STUDIA INFORMATICA 2014
Volume 35 Number 4 (118)

Krzysztof A. CYRAN, Dariusz MYSZOR
Silesian University of Technology, Institute of Informatics

HOW MESSAGE PASSING INTERFACE (MPI) ACCELERATES A COALESCENT-BASED WHOLE GENOME SIMULATOR

Summary. Computer simulations are one of the pillars of contemporary science. In the current paper we present next type of improvements introduced into GENOME: A rapid coalescent-based whole genome simulator. The modifications are based on parallelization of processes with the use of MPI technology. The influence of introduced modification, has been tested on Ziemowit HPC cluster which is installed in Silesian Biofarma. Results point out that process of outcomes generation can be reduced significantly if proposed modifications are applied.

Keywords: coalescence, MPI, parallelization, whole genome generation

JAK INTERFEJS PRZEKAZYWANIA KOMUNIKATÓW MPI PRZYSPIESZA SYMULATOR CAŁEGO GENOMU OPARTY NA KOALESCENCJI

S**treszczenie**. Symulacje komputerowe uważane są za jeden z filarów współczesnej nauki. W artykule opisano kolejny rodzaj optymalizacji programu GENOME: A rapid coalescent-based whole genome simulator, mającej na celu skrócenie czasu oczekiwania na wyniki. Modyfikacje bazują na zrównoleglaniu wykonywania procesów z wykorzystaniem technologii MPI oraz klastrów HPC. W celu przetestowania uzyskanego rozwiązania wykorzystano klaster HPC Ziemowit, będący na wyposażeniu Śląskiej Biofarmy. Wyniki wskazują, iż wprowadzone modyfikacje pozwalają na znaczne skrócenie czasu wykonywania aplikacji.

Słowa kluczowe: koalescencja, MPI, zrównoleglanie, generacja całego genomu

1. Introduction

In the post-genomic era, which started with the completion of the Human Genome Project, the extremely large amount of genetic data of various types creates an informational noise that requires rapid and constant development of new advanced methods, as well as technologies, for data mining and processing the diversity of these data to understand biological and environmental processes contributing to genetic variation. In particular, currently launched 1000 Genomes Project (1000GP) and Cancer Genome Project (CGP) constitute an important challenge for information sciences to search for causative variants by genome-wide association studies (GWAS). Unfortunately, it has been shown that synthetic associations [1] can be responsible for many GWAS results. Such synthetic association which obscure actual reasoning can be proved by using whole genome simulators such as GENOME [2] considered as a gold standard and used by Dickson et al. (2010) [1] for the above purpose. Whole genome simulators supply advanced machine learning (ML) and bio-statistical models with large examples of sequences required for training, before application to genome-wide studies. This may finally result in better understanding of the genetic component of susceptibility to common human diseases with complex scheme of inheritance. As stated by Stefansson (2010) [3], at the EMBO|EMBL Symposium Human Variation: Cause and Consequence, the understanding of the genetics of complex diseases will always be fragmentary unless considered in the context of the natural selection and human diversity in general. Therefore, easy access to genetic variation patterns obtained from whole genome simulators is so crucial for studying implications of the common disease – rare variant (CDRV) paradigm and for effective use of data mining methods in the case-control genome-wide association studies (GWAS). Most of the GWAS research up-to-date has been focused on common variants [4, 5], which invariably have small effects. However, the role of rare variants in complex disease susceptibility, which can have larger effect sizes [1, 6] is currently recognised and emphasised. The analysis of lower frequency polymorphisms necessitates utilization of larger sample sizes as well as development of new analytical approaches in order to increase power. The 1000GP will improve our understanding of variation at the lower end of the frequency spectrum and is expected to motivate the enhancement of information capture and interpretation of genetic association studies [5].

In our study, GENOME [2] has been used as a software for parallel coalescent-based [7, 8] generation of genome-wide sequences for studying synthetic associations (SA) between rare variants and common marker SNPs in GWAS case-control studies. We tried to perform time optimization of this software by using Message Passing Interface (MPI) technology. With that respect, our method will make possible to study SA between rare variants and common SNP markers faster. Verification of proposed methodology has been done at Sile-

sian Ziemowit cluster. The methodology is based on multiprocessing in a coalescent-based simulations for generating whole genomes. As proved by Dickson et al. (2010) [1] it allows for studying of the effect of synthetic associations (SA) between multiple rare variants and marker common SNPs using large amounts of data simulated on parallel computational clusters.

2. Original implementation

GENOME: A rapid coalescent-based whole genome simulator [2], is currently used as one of gold standards in generating the whole sequences for population genetics studies (see for example how it is used for proving the existence of synthetic associations in Genome Wide Association Studies GWAS [1]). Since authors of GENOME software allowed for introduction of modifications into original code if certain condition are met (redistributions of source code must retain specific copyright notice, this list of conditions and specific disclaimer; redistributions in binary form must reproduce the specific copyright notice, this list of conditions and specific disclaimer in the documentation and/or other materials provided with the distribution; the names of project contributors may not be used to endorse or promote products derived from this software without specific prior written permission), therefore keeping in mind the above conditions, we tried to optimise the code in order to achieve higher performance. One of such modification, parallelization based on introduction of multithreading, has been presented in our first in a series paper [9]. Here, the second level of parallelization is considered, namely using multiple processes in MPI technology.

Original code of a GENOME is based on a standard coalescent model [10, 11, 13]. It allows for simulation of sequences (on the whole genome scale) which are drawn from a population that is under the Wright-Fisher neutral model [12]. At the beginning of the simulation user defines number of subpopulations (s), effective population size (N) and number of samples drawn from every population (n), number of independent regions (chromosomes, c), number of fragments (genes, g) in a single independent region and length of each fragment (nucleotides, l). In addition recombination rate (r), mutation rate (mt) and migration rate (m) can be set. User can define complex subpopulation scenarios such as bottlenecks, subpopulations growths / shrinks, merge of subpopulations etc. There is also ability of definition of frequency of different recombination rates.

Simulation application works in two phases. First phase is responsible for creation of gene genealogy for every gene from pool of sampled individuals. It takes into account coalescent, recombination and inter-subpopulation migration events. Interestingly for genome wide analysis, coalescent events occur so often that execution of code which is skipping gen-

erations that are not informative can take more time than application of simpler approach that is recreating every generation back in time.

Second phase is responsible for application of mutation process on top of created gene genealogy, that is in accordance with user defined values of mutation coefficient.

Original code was written in C++ and is portable between various operating system. All calculations performed by GENOME application are done in a single process which utilizes single thread approach. Therefore, an assumption could be made that introduction of modifications in the code, which will allow for utilization of several nodes of High Performance Computing Cluster (HPCC), should allow for decreasing of result await time.

3. Parallel programming

In order to be able to run the application on variety of systems Message Passing Interface (MPI), which is standardized and portable system for message passing across computation units, is applied. MPI is commonly used in scientific parallel computations because it allows achieving of high performance, scalability and portability. Huge advantages of MPI is abundance of free and commercial implementations. As a result ability of cooperation with many programming languages as well as hardware and software platforms is achieved. All these features makes application of MPI a perfect fit for distributed memory environment e.g. of high performance computing clusters.

Among many available distributions, Intel MPI will be chosen because it is optimized for Intel based processors, therefore it is argued that application of this library will results in additional speed-up of communication operations, which is crucial for achieving of meaningful results of biological processes simulation in acceptable time.

3.1. High Performance Computing (HPC) cluster

Silesian University of Technology is a Member of Bio-Farma Consortium, which is in a possession of high performance computing cluster, called HPC Ziemowit. Hardware architecture of HPC Ziemowit is composed of 106 computing nodes (IBM Blade Center HS22V). Each node possess two Intel Xenon 2.66 GHz processors which are internally equipped with 6 cores. As a result 1272 computation units are available. Utilization of 40 Gbit InfiniBand as a connection between nodes, allows to take full advantage of parallel computing abilities. It is worth to mention that additional 1Gbit Ehternet network is available for management purposes. Each computing node within the cluster is equipped with 36 GB of RAM shared between two processors.

Ziemowit HPC offers 216 TB of disk space, that is distributed through three disk arrays and is accessible as one virtual drive. Storage is based on Lustre parallel distributed file system that ensures efficient way of storage data access. In order to ensure fast communication between computing units and storage devices, 10 additional nodes plugged to InfiniBand on one end and to storage server on the other, are utilized. As a result overall theoretical computational power of HPC Ziemowit is estimated at 13.5 TFLOPS.

The hardware architecture of HPC Ziemowit cluster is controlled by Red Hat Enterprise 5 operating system. In addition, Intel Cluster Studio compiler, which improves thread performance and performs code optimization, therefore allows to gain additional speed-up of code execution and ensures maximum utilization of available computing platform, is available. Furthermore implementation of Intel MPI library, which is optimized for Intel based processor is installed.

3.2. Pseudorandom number generator

Processes of development of real population are based on randomness. Thus ability of generation of random sequences of data is also required in case of coalescence approach to the population development. However computer algorithms, designated to run on contemporary computers, are deterministic therefore, obtainment of truly randomness in computer environment is not an easy task. Fortunately in case of simulation researches truly random sequences are not required, as a result application of specialized deterministic algorithms which are able to generate sequences of data, that capture all important statistical properties of true random series, is possible.

Generated sequences should be able to pass tests designated to validate its pseudorandom statistic properties such as die hard battery of tests. In order to generate sequence of values PRNG must be initialized with seed. For a given seed PRNG should always return the same sequence of values. It is worth to mention that such a behaviour introduces additional benefits among them facilitation of application testing and ability of easy recreation of phenomenon which were observed during simulation approach execution.

There are many available algorithms, that generate pseudo random values (PRNG), however only fraction can successfully pass statistical tests and at the same time is characterized by sufficiently long period (maximum length of repetition free sequence for all possible seeds) [14] and ability of efficient generation of data series. In conducted simulations Mersenne-Twister pseudorandom number generator was utilized [15]. It has good k-distribution property and passes most of available tests for statistical randomness, in addition it is characterize by long period equal to 2^{19937}-1 . In addition Mersenne Twister PRNG util-

izes memory in an efficient way and is characterized by high speed of generation of consecutive values, which are created in packages of 625 entries each.

Introduction of multi-process application includes new challenges in the area of random number generators. Parallel PRNG should be able to generate sequences of pseudorandom values, without correlation at the level of node and between nodes and at the same time it should require minimum amount of communication between nodes, moreover it should work correctly for any number of computing nodes. The main techniques of parallelization of pseudorandom number generators for N computation nodes include [16]:

Leapfrog – values X_j that are generated by pseudorandom number generator, are assigned to the computation nodes j according to the formula X_j , X_{j+R} , X_{j+2R} , ...

Sequence splitting – the pseudo-random series X, characterized by period d is divided into R non overlapping sub-sequences. Each subsequence contains M =dR values. Individual group is assigned to each node, so node j generates following values from the sequence X_{jM} , X_{jM+1} , . . .

Independent PRNG – each node utilizes different algorithm of pseudorandom values generation or the same PRNG algorithm with various values of internal state constants, which are set in such a way that generators are independent. There are various libraries which provides above mentioned solution e.g. Intel® Math Kernel Library which implements set of 6024 independent Mersenne Twister pseudorandom generators each with period equal to 2^{2203}-1.

Independent sequences – each node initializes instance of PRNG with different seed, these seeds should be random and independent at each node. Initialization of seed at each node, can depend on the time, however such approach might result in achievement of identical seeds across many nodes. In order to obtain proper set of seeds, additional PRNG is required, then an array of seeds is initialized at master node which ensures correctness of generated seeds and values are sent across the cluster.

The first two solutions are applicable only for PRNGs which allow for easy jumping to the arbitrary values in the random sequence. What is more there is a possibility of the occurrence of long range correlation between sequences in different nodes. On the other hand the independent sequence solution does not guarantee non overlapping sequences in different nodes, however it is usually applied for PRNG with long periods, therefore such a possibility is practically ruled out. Independent sequences method is a good option, however it usually requires utilization of commercial libraries.

In case of Mersenne Twister generator application of leapfrog or sequence splitting mechanism would take significant fraction of computation time, therefore the decision was made that independent sequences method will be applied. Master node initializes vectors with values and sends to the respective worker nodes.

4. Introduced modifications

Introduction of concurrency into base code can be done at various levels of simulation process. Therefore, at the solution design level areas, which can be simulated as independently from each other as possible, should be recognized. In addition computational architecture and hardware should be taken into account in order to utilize its advantages and to avoid creation of architecture induced bottlenecks. As mentioned earlier Ziemowit HPC possesses over one thousands of computation units, it allows for high level of tasks granulation however utilization of MPI causes that communication between these tasks is done through exchanging of message between processes, therefore synchronization tasks such as barrier synchronization might take sufficient amount of simulation time.

Simulation process has two phases first one is generation of genealogy for every gene from every chromosome of every individual sampled from the first population. During this phase following processes are simulated:

- searching for predecessors of every individual,
- recombination process causes that one individual can have various areas of a single chromosome derived from various parents,
- migration process introduces ability of possession of predecessors from different subpopulations.

Second phase of simulated process includes application of mutations. Gene genealogy and predefined mutation process are also taken into account. Initial performance analysis point out that first phase of simulation takes sufficient amount of total simulation time therefore, priority was put at parallelization of processes from this simulation's phase.

Parallelization can be introduced at the level of genes, individuals, populations or chromosomes simulation. Introduction of concurrency at the level of genes processing was ruled out because sequences of genes (in chromosome of a single individual) should emerged from the same parental individual unless recombination event occurred, as a result dependency, between consecutive genes exists which prevents ability of efficient parallelization.

Another level on which concurrency can be introduced is analysis of individuals. All genes of a single individual, from given population, can be analysed in a package independently from other individuals in a given population. Issue in such a solution is the number of individuals which should be analysed, which significantly exceeds the number of computation units, possessed by Ziemowit HPC. This issue can be mitigated by assigning of groups of individuals to every HPC's node. In such a solution two types of nodes are distinguished main node and worker nodes. Main node is responsible for initialization of worker nodes, data collection, normalization when first phase of computation is done and execution of second phase of simulation process i.e. assigning mutation process on generated gene genealogy.

Every worker node, on the other hand, is responsible for simulation of group of descendants, as well as parental individuals, and keeping track of local coalescence events. The issue is that in such a solution parent can be located in a different worker node than its descendants. Therefore necessity of internode communication occurs. Node which stores parent must get unique id of a child, and its location in a sparse structure, as well as list of genes which given parents passing on particular child. To make matter worse all nodes have to process the same generation at the same time frame, in order to be able to assign children (from one node) to its parent (from the other node) and to track coalescent events efficiently. Termination of simulation requires additional communication event based on collective operation MPI_AllReduce, with is summing the amount of calescent events. When proper value is obtained, first phase of simulation is terminated on all worker nodes and arrays describing local coalescence events are send to the main node. Next step is execution of phase two of simulation process.

In order to test performance of such a solution, application in C++, based on GENOME, GENOME: A rapid coalescent-based whole genome simulator project, was implemented and experiments were conducted in Ziemowit environment. For original code, average time of computation, for very small data set: p=1, N=12 , n=7, g=3, r=0.0001, m=0.00025, l=10000, for 100 independent repetition of simulation run took 0.175369 [s]. When suggested multi process approach was applied simulation took more than 3 seconds. Results of experiments showed that extensive amount of communications between worker nodes and necessity of synchronization between all nodes leads to the significant extension of the time which is required in order to execute first phase of simulation. As a result this approach was abandoned.

Population consists of subpopulations, between them migration of individuals is limited, so time of simulation can potentially be reduced if a single subpopulation is simulated by single worker node. It allows for minimization of amount of data which are transferred among worker nodes, still all worker nodes have to work on the same generation at the same timeframe, so in order to start to process the next generation there is a need for synchronization between nodes.

In order to mitigate effect of data dependency, parallelisation at the level of simulation of chromosomes was introduced. GENOME application does not track dependencies between genes from various chromosomes of a single individual, such an approach allows for simulation of chromosomes by independent computation units. As in previous method there is a master node, which is responsible for initialization of worker nodes, and collection of data after completion of simulation. Importantly additional speed up is obtained because worker nodes, in this version of simulation application, could be responsible for conduction of both phases of simulation process i.e. creation of gene genealogy and application of mutations.

On one hand, applied approach limits utilization of HPC nodes to the number of analysed chromosomes e.g. for simulation of whole human genome which possess 46 chromosomes only 47 nodes (one master node and 46 worker nodes) are utilized, therefore there is no way of increase of performance through utilization of HPC machine which more computation nodes. On the other hand, there is no need of communication between worker nodes during simulation process, therefore time which has to be sacrifice for the sake of initialization and communication is minimized as a result obtained speed up, should be significant.

5. Results

Several tests, with various values of initialization parameters, were performer in order to validate performance gain for implemented solution. Default values, of simulation input variables were as follow: p=1, N = 10000, n = 6000, g = 30, r = 0.0001, m= 0.00025, l=10000. For every test scenario 30 independent simulation runs were performed.

Results were presented in on fig. 1 and fig. 2. Influence of the number of analysed chromosomes was presented. Noteworthy performance gain is obtained for more than 1 chromosomes. Because of ability of minimization of communication between nodes during simulation, almost linear speed up can be observed.

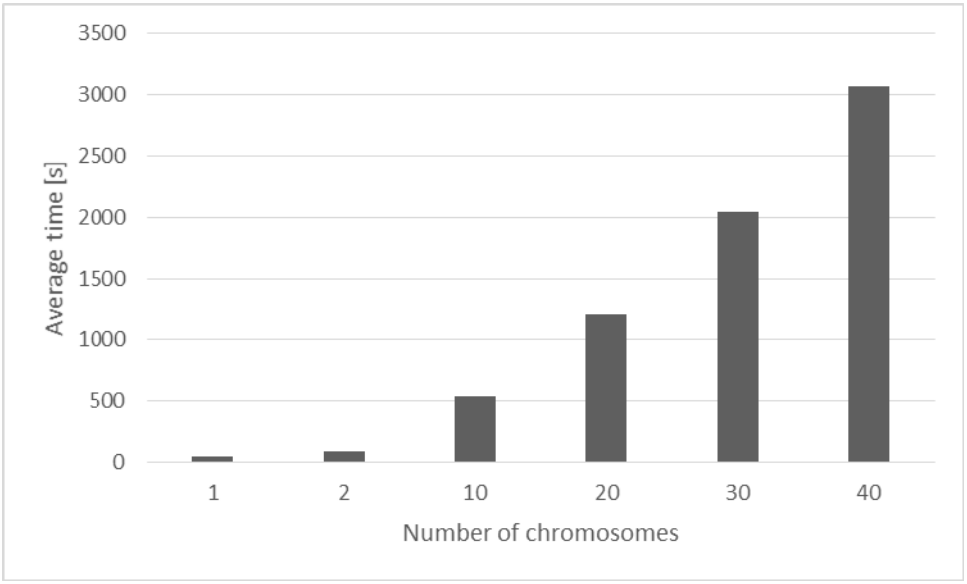

Fig. 1. Time of simulation application execution for various number of involved threads (p=1, N = 10000, n = 6000, g = 30, r = 0.0001, m= 0.00025), when single computation process is utilized

Rys. 1. Czas wykonywania programu symulacyjnego dla różnej liczby chromosomów (p=1, N = 10000, n = 6000, g = 30, r = 0.0001, m= 0.00025), gdy aplikacja wykorzystuje jeden proces obliczeniowy

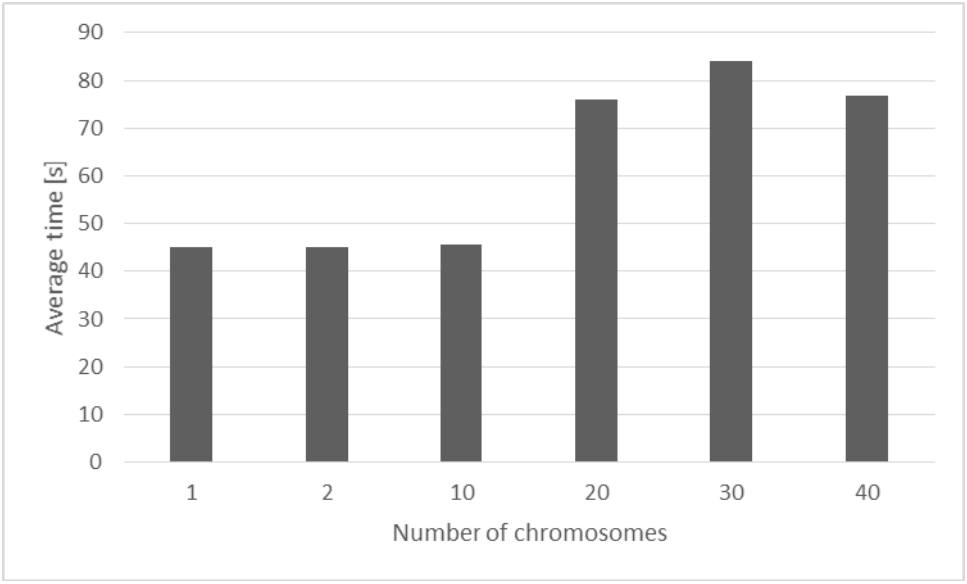

Fig. 2. Time of simulation application execution for various number of involved threads (p=1, N = 10000, n = 6000, g = 30, r = 0.0001, m= 0.00025), when number of utilized processes is equal to the number of simulated chromosomes

Rys. 2. Czas wykonywania programu symulacyjnego dla różnej liczby wątków (p=1, N = 10000, n = 6000, g = 30, r = 0.0001, m= 0.00025), gdy liczba wykorzystywanych procesów równa jest liczbie symulowanych chromosomów

6. Discussion

As compared to the state-of-the-art software simulating the whole genome data, the GENOME [2], the advantage of method is the proposed development of a parallel algorithm for implementation in distributed cluster environment such as Silesian BIOFARMA. Although not all nodes of HPC cluster were utilized obtained performance gain is sufficient. Unfortunately limiting factor, on a way of additional speedup obtainment and increase of utilization of computation cluster, are exchange of data between nodes and necessity of synchronization of execution of successive generations. Conducted experiments point out that in analysed scenario message passing between nodes is taking too much time in order to obtain speedup, what is even worst, it causes significant slowdown of the simulation process. In order to overcome this issue in the future work internal architecture of available computing cluster can be taken into account. In HPC Ziemowit set of nodes utilizes shared memory, communication through shared memory is significantly faster than communication with messages that are send across the network. Therefore an assumption could be made that proper utilization of threads, which would take benefit from existing architecture, can lead to further improvement in the results await time.

The results obtained by us will accelerate searching for causative variants in simulated data using combined, machine learning and statistical models. Without distributed processing the task would require much more time, therefore the clear advantage of the proposed methodology is its time effectiveness. After choosing the appropriate data mining strategy using simulated data, this strategy may be easily applied to the actual anonymised data (from 1000GP and WTCCC) genotyped from large cohorts of controls and cases with complex diseases (such as for example autoimmune thyroid disease or breast cancer). The use of developed algorithms has been verified in the Institute of Informatics of Silesian Univeristy of Technology [9] and in European BIO-FARMA Consortium platforms located in Gliwice, Poland (current paper). Using this parallel environment we performed a series of experiments and obtained in parallel way data for studying the effect of synthetic associations between multiple rare variants and common marker SNPs. The results achieved will make searching for causative variants in simulated and real (WTCCC or 1000GP) data more reliable because the effect of synthetic association can be assessed much faster than with original GENOME software.

7. Summary

It is widely believed that simulation, in addition to theory and experiments, is one of the pillar of science. Computer simulations are important tools because they usually allow for obtainment of results in shorter time, and with smaller costs, than conduction of real experiments. Utilization of these methods is especially important in biological sciences, where they can be utilized in order to validate hypothesis and confirm correctness of undertaken approaches. Therefore work over reduction of results await time is important, especially now because new methods utilized in biological science, e.g. genome sequencing techniques allow for efficient obtainment of vast amount of real data.

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the PEOPLE Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme FP7/2007-2013/ under REA grant agreement no. 298995. Krzysztof Cyran has been supported by the above mentioned grant, Dariusz Myszor has been supported by BK-215/Rau2/2013. Calculations were carried out using the computer cluster Ziemowit (http://www.ziemowit.hpc.polsl.pl) funded by the Silesian BIO-FARMA project No. POIG.02.01.00-00-166/08 in the Computational Biology and Bioinformatics Laboratory of the Biotechnology Centre in the Silesian University of Technology.

BIBLIOGRAPHY

1. Dickson A. et al.: Rare variants create synthetic genome-wide associations. PloS Biology, 2010, vol. 8, no. 1, p. 1÷12.
2. Liang L., Zollner S., Abecasis G.R.: GENOME: a rapid coalescent-based whole genome simulator. Bioinformatics, 2007, vol. 23, no. 12, p. 1565÷1567.
3. Stefansson K.: Genetic common/complex traits and the juxtaposition of nature and nurture. EMBO|EMBL Symposium on Human Variation: Cause and Consequence, EMBL Advanced Training Centre, Heidelberg, Germany, 2010.
4. Peng B., Kimmel M.: Simulations provide support for the common disease – common variant hypothesis. Genetics, 2007, vol. 175, p. 763÷776.
5. Panoutsopoulou K., Zeggini E.: Finding common susceptibility variants for complex diseases: past, present, and future. Briefings in Functional Genomics and Proteomics, 2010, vol. 8, no. 5, p. 345÷352.

6. Gorlov I. P., Gorlova O. Y., Sunyaev S. R., Spitz M. R., Amos C. I.: Shifting paradigm of association studies: value of rare single-nucleotide polymorphisms. Am. J. Hum. Genet., 2008, vol. 82, p. 100÷112.
7. Hein J., Schierup M., Wiuf C.: Gene Genealogies, Variation and Evolution: A Primer in Coalescent Theory. Oxford University Press, 2004.
8. Kingman J.F.C.: Origins of the coalescent: 1974–1982. Genetics, 2000, vol. 156, p. 1461÷1463.
9. Cyran K.A., Myszor D.: Multithread parallelization of a rapid coalescent-based whole genome Simulator. Studia Informatica, vol. 35, no. 4 (118), 2014, p. 73÷88.
10. Hudson R.R.: Properties of a neutral allele model with intragenic recombination. Theoretical Population Biology, 1983, vol. 23, p. 183÷201.
11. Hudson R.R.: Gene genealogies and the coalescent process. Oxford Surveys in Evolutionary Biology, 1990, vol. 7, p. 1÷44.
12. Hudson R.R.: Generating samples under a Wright-Fisher neutral model. Bioinformatics, 2002, vol. 18, p. 337÷378.
13. Donnelly P., Tavare S.: Coalescents and genealogical structure under neutrality. Annual Review of Genetics, 1995, vol. 29, p. 401÷421.
14. Cyran K. A., Kimmel M.: Interactions of neanderthals and modern humans: what can be inferred from mitochondrial DNA? Math Biosci Eng, 2005, vol. 2, no. 3, p. 487÷98.
15. Matsumoto M., Nishimura T.: Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. ACM Transactions on Modeling and Computer Simulation, 1998, vol. 8, p. 3÷30.
16. Coddington P. D.: Random number generators for parallel computers. The NHSE Review, 1997, vol. 2.

Omówienie

W artykule przedstawiono zestaw modyfikacji wprowadzonych do kodu aplikacji GENOME: A rapid coalescent-based whole genome simulator. Celem było skrócenie czasu oczekiwania na wyniki, gdy aplikacja uruchamiana jest w środowisku klastra obliczeniowego HPC. Aplikacja została rozbita na wiele procesów, do komunikacji pomiędzy procesami wykorzystano biblioteki MPI.

Kolejne sekcje artykułu opisują klaster obliczeniowy Ziemowit HPC, który został wykorzystany do przeprowadzania testów wydajnościowych aplikacji zmodyfikowanej oraz bazowej. Przedstawiono także problem wykorzystania generatora liczb pseudolosowych w środo-

wisku wieloprocesowym. Omówiono i przetestowano kilka scenariuszy pozwalających na zrównoleglenie kodu.

Uzyskane wyniki wskazują, iż wprowadzone modyfikacje pozwalają na znaczące skrócenie czasu generacji wyników, gdy liczba symulowanych chromosomów była większa od jednego.

Addresses

Krzysztof CYRAN: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16
44-100 Gliwice, Polska, krzysztof.cyran@polsl.pl.
Dariusz MYSZOR: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16
44-100 Gliwice, Polska, dariusz.myszor@polsl.pl.