High Performance Computing for Haplotyping: Models and Platforms

Andrea Tangherloni[1(✉)], Leonardo Rundo[1,5], Simone Spolaor[1], Marco S. Nobile[1,6], Ivan Merelli[2], Daniela Besozzi[1], Giancarlo Mauri[1,6], Paolo Cazzaniga[3,6], and Pietro Liò[4]

[1] Department of Informatics, Systems and Communication, University of Milano-Bicocca, Milan, Italy
andrea.tangherloni@disco.unimib.it
[2] Institute of Biomedical Technologies, Italian National Research Council, Segrate, Italy
[3] Department of Human and Social Sciences, University of Bergamo, Bergamo, Italy
[4] Computer Laboratory, University of Cambridge, Cambridge, UK
[5] Institute of Molecular Bioimaging and Physiology, Italian National Research Council, Cefalù, PA, Italy
[6] SYSBIO.IT Centre of Systems Biology, Milano, Italy

Abstract. The reconstruction of the haplotype pair for each chromosome is a hot topic in Bioinformatics and Genome Analysis. In Haplotype Assembly (HA), all heterozygous Single Nucleotide Polymorphisms (SNPs) have to be assigned to exactly one of the two chromosomes. In this work, we outline the state-of-the-art on HA approaches and present an in-depth analysis of the computational performance of GenHap, a recent method based on Genetic Algorithms. GenHap was designed to tackle the computational complexity of the HA problem by means of a *divide-et-impera* strategy that effectively leverages multi-core architectures. In order to evaluate GenHap's performance, we generated different instances of synthetic (yet realistic) data exploiting empirical error models of four different sequencing platforms (namely, Illumina NovaSeq, Roche/454, PacBio RS II and Oxford Nanopore Technologies MinION). Our results show that the processing time generally decreases along with the read length, involving a lower number of sub-problems to be distributed on multiple cores.

Keywords: Future-generation sequencing
Genome Analysis Haplotype Assembly
High Performance Computing · Master-Slave paradigm

1 Introduction

The advent of second-generation sequencing technologies revolutionized the field of genomics, enabling a more complete view and understanding of the genome of different species. However, despite their great contribution to the field, the

G. Mencagli et al. (Eds.): Euro-Par 2018 Workshops, LNCS 11339, pp. 650–661, 2019.
https://doi.org/10.1007/978-3-030-10549-5_51

data produced by these technologies are still unsuitable for several applications, including Haplotype Assembly (HA). This problem consists in assigning all heterozygous Single Nucleotide Polymorphisms (SNPs) to exactly one of the two homologous chromosomes, leveraging data from sequencing experiments. The short length of the reads produced by second-generation sequencing technologies might be not long enough to span over a relevant number of SNP positions, leading to the reconstruction of short haplotype blocks [8,43] and ultimately hindering the possibility of reconstructing the full haplotypes.

In recent years, however, a third-generation of sequencing technologies was developed and paved the way to the production of sequencing data characterized by reads covering hundreds of kilobases, thus able to span different SNP loci at once [16,32,33]. Unfortunately, the increase in length comes at the cost of a decrease in the accuracy of the reads, compared to the short and precise ones produced by second-generation sequencing technologies, such as NovaSeq (Illumina Inc., San Diego, CA, USA) [31]. In order to compensate for this inadequacy, there is a need for increasing the read coverage. Formally, the coverage of a sequencing experiment is the average number of times that each nucleotide is expected to be covered by a read. This value is given by the following relationship:

$$\text{cov} = (L \cdot N)/G, \tag{1}$$

where cov stands for the coverage, L for the read length, N for the number of reads and G for the length of the haploid region of the genome on which the reads are mapped [20]. Equation (1) shows that longer reads or a higher amount of reads are needed to increase the coverage. In practice, an average coverage higher than 30× is the *de facto* standard for accurate SNP detection [38]. Along with the HA issues, novel challenges—e.g., poliploidity, metagenomics, analysis of cancer cell heterogeneity and chromosomal capture experiments—require sequencing data with a high coverage.

In this paper, we briefly describe the state-of-the-art on haplotype computational tools, providing a taxonomy based on the employed computational techniques. Then, we focus on GenHap [40], an evolutionary method that exploits High Performance Computing (HPC) architectures. We show how GenHap performs on data produced by four different sequencing platforms, namely:

- Illumina NovaSeq (Illumina Inc., San Diego, CA, USA) [31]: the most used and widespread platform belonging to the class of second-generation sequencing technologies, able to produce a huge number of short and precise reads (up to 150 bp);
- Roche/454 (Roche AG, Basel, Switzerland) [23]: a second-generation sequencing technology able to produce accurate and slightly longer reads than Illumina sequencers (up to 700 bp);
- PacBio RS II (Pacific Biosciences of California Inc., Menlo Park, CA, USA) [32,33]: a third-generation sequencing technology able to produce long reads (up to 30000 bp);

– Oxford Nanopore Technologies (ONT) MinION (ONT Ltd., Oxford, United Kingdom) [16,17,36]: the latest developed third-generation sequencing technology, able to produce reads that are tens of kilobases long.

The manuscript is structured as follows. Section 2 describes and classifies the most used HA approaches, focusing on HPC potential provided by GenHap. The achieved results, in terms of scalability and efficiency on multi-core architectures, are shown and analyzed in Sect. 3. Finally, future directions and possible fruitful connections with other research fields, such as machine learning and security in distributed computing, are mentioned in Sect. 4.

2 HPC in Haplotype Assembly

Current human Whole Genome Sequencing (WGS) approaches do not generally provide phasing information, limiting the identification of clinically-relevant samples, estimation of compound heterozygosity as well as population-level phenomena, including haplotype diversity and Linkage Disequilibrium patterns that could help to resolve migratory patterns and mutation origins [7].

Several computational HA approaches for human genome phasing have been proposed in literature [7]. Most of these methods solve the NP-hard Minimum Error Correction (MEC) problem, which aims at inferring the haplotype pair that yields two disjoint sets of the sequencing reads characterized by the minimum number of SNP values to be corrected [41]. An additional variant of MEC exists, called weighted MEC (wMEC) [14], which takes into account also the information concerning the quality of the reads.

In what follows, we concisely describe the most diffused HA methods and graphically represent them by means of a "phylogenetic tree"-like diagram (Fig. 1). Then, we focus on the functioning of the distributed GenHap implementation on multi-core architectures [40].

2.1 Related Work

Beagle [5] is one of the earliest heuristic approaches based on Hidden Markov Models (HMMs). Considering the genotype information of an individual, Beagle finds the most likely haplotype pair among different possible haplotype solutions. It has a quadratic computational complexity with respect to the input data.

SHAPEIT [10] starts from genotyping data related to a population and, given the genotype data of an individual, exploits an HMM-based approach to estimate the haplotype pair. The population data are used to apply constraints on the graph, which denotes all possible haplotypes compatible with the input data, in order to determine the haplotype of that individual. At each iteration, SHAPEIT has a linear complexity with respect to the number of haplotypes.

Eagle2 [22] is a phasing algorithm that exploits the Burrows-Wheeler transform to encode the information from large external reference panels. It relies on an HMM to explore only the most relevant phase paths among all possible paths. The authors showed that Eagle is 20 times faster than SHAPEIT [10].

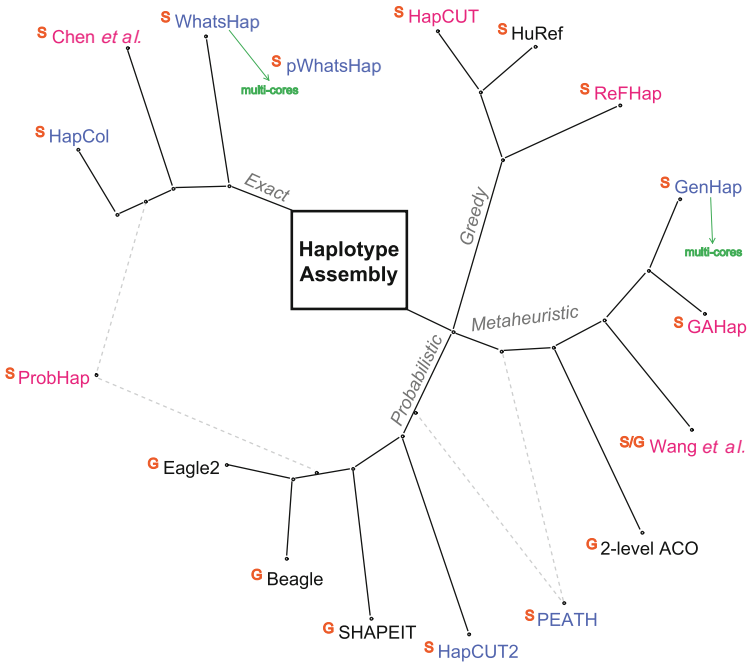

Fig. 1. The "phylogeny" of haplotyping methods. Over the past few years, the repertoire of tools for haplotyping has rapidly expanded. A "phylogenetic tree"-like diagram is used here to depict the division of the algorithms in 4 different classes, namely: exact, greedy, probabilistic, metaheuristic. Hybrid methods are connected with dashed lines to the implemented multiple computational techniques. The orange superscript denotes the analyzed data: sequencing (S) and genotyping (G). Methods that solve either the MEC or the wMEC problem are denoted with blue or magenta, respectively. Finally, the HA methods that exploit HPC are highlighted with a green arrow directed to the used computational resources. See text in Sect. 2.1 for descriptions of the most common software representatives of branches, and for the definitions of abbreviations. (Color figure online)

HapCUT [1] leverages sequencing data (i.e., the entire set of reads is considered) instead of population genotypes. It infers the haplotype pair of an individual by partitioning the set of reads solving the MEC problem. The MEC problem is reduced to the max-cut problem, which is greedily solved over the graph representation of the input instance.

HapCUT2 [12] is a recent heuristic approach that exploits a haplotype likelihood model for the sequencing reads. A partial likelihood function is used to evaluate the likelihood of a subset of the fragments. Differently from its previous version (HapCUT [1]), which is based on a greedy max-cut algorithm, HapCUT2 optimizes the likelihood to find a max-cut in graph representation of the input instance.

ProbHap [18] relies on an exact likelihood optimization technique to solve a generalized version of the MEC problem. It exploits a dynamic programming algorithm capable of exactly optimizing a likelihood function, which is specified by a probabilistic graphical model that generalizes the MEC problem.

ReFHap [11] is based on a heuristic algorithm to find the max-cut. ReFHap solves the Maximum Fragments Cut (MFC) problem instead of the classic MEC problem. The max-cut problem is reduced to the MFC problem, which is addressed using a greedy approach.

HuRef [21] is a heuristic approach that aims at inferring the heterozygous variants of an individual. It is based on a greedy algorithm that iteratively refines the initial partial haplotype solutions. The authors leveraged this HA approach to study non-SNP genetic alterations considering the diploid nature of the human genome.

Chen et al. [6] proposed an exact approach for the MEC problem using an integer linear programming solver. First, the fragment matrix is decomposed into small independent sub-matrices. Each of these sub-matrices is used to define an integer linear programming problem that is then exactly solved.

WhatsHap [29] is an exact method relying on a dynamic programming algorithm used to solve wMEC. It implements a fixed parameter tractable algorithm, where the fixed parameter is the maximum coverage of the input instance, to deal with the NP-hardness of the wMEC problem. This method does not assume the all-heterozygosity of the phased positions.

pWhatsHap [4] is an efficient version of WhatsHap [29], which was designed to leverage multi-core architectures in order to obtain a relevant reduction of the execution time required by WhatsHap. The proposed implementation exploits the physical shared memory of the underlying architecture to avoid data communication among threads.

HapCol [30] implements a dynamic programming algorithm to solve an alternative version of the wMEC problem, called k-MEC, which is used to take into account the distribution of sequencing errors of future-generation technologies. In this strategy, the number of corrections per column is bounded by the parameter k. No all-heterozygous assumption is required.

Two-Level ACO [2] is based on the Ant Colony Optimization (ACO) technique, which is a metaheuristic designed to deal with combinatorial problems on graphs generated starting from the genotyping data given as input. This approach is based on the pure parsimony criterion to find the smallest set of distinct haplotypes that solves the HA problem.

Probabilistic Evolutionary Algorithm with Toggling for Haplotyping (PEATH) [26] is based on the Estimation of Distribution Algorithm (EDA), which is a metaheuristic suitable for continuous problems. During each iteration of EDA, the promising individuals are used to build probabilistic models that are sampled to explore the search space. This metaheuristic is exploited to deal with noisy sequencing reads, aiming at inferring one haplotype, under the all-heterozygous assumption.

Wang et al. [41] relies on Genetic Algorithms (GAs), which are a family of metaheuristics designed to tackle combinatorial and discrete problems. This method was proposed to address an extended version of the MEC problem in which genotyping data are considered during the SNP correction process.

GAHap [42] uses GAs to infer the haplotype pair of an individual working on nucleotide strings. During the optimization, GAHap solves the MEC problem by means of a majority rule that takes into account allele frequencies. No all-heterozygous assumption is required.

GenHap [40] is a novel computational method based on GAs to solve the wMEC problem. This method exploits a *divide-et-impera* approach to partition the entire problem into smaller and manageable overlapped sub-problems. In order to solve in parallel the sub-problems, GenHap was developed using a Master-Slave approach to leverage multi-core architectures.

2.2 GenHap: A Distributed Computing Implementation for HA

Hereafter, we briefly recall the peculiarities of GenHap [40], by focusing on the HPC implementation. GenHap tackles the HA problem by solving the wMEC problem, exploiting an approach based on GAs. Since the execution time and the problem difficulty increase with the number of reads and SNPs of the input data, GenHap follows a *divide-et-impera* approach [24] in which the wMEC problem is efficiently solved by splitting the fragment matrix $\mathbf{M}$ into $\Pi = \lfloor m/\gamma \rfloor$ sub-matrices consisting of γ reads (where γ depends on the coverage value and on the nature of the sequencing technology). By so doing, the problem difficulty is reduced by solving the sub-problems by means of independent GA executions that eventually converge to solutions having two sub-haplotypes with the least number of corrections to the SNP values. Finally, these sub-haplotypes are combined to achieve the complete haplotype pair. It is worth noting that GenHap considers all phased positions [19] as heterozygous during the optimization phase with GAs. As soon as the sub-haplotypes are obtained, all possible uncorrected heterozygous sites are removed and the correct value is assigned by checking the columns of the sub-partitions.

GenHap makes use of a Master-Slave distributed programming paradigm [39] to speed up the overall execution (Fig. 2) [35]. It was developed using the C++ programming language and the Message Passing Interface (MPI) specifications to leverage multi-core Central Processing Units (CPUs). The Master-Slave strategy of GenHap consists of the following phases: **(1)** the Master process (MPI rank 0) proceeds by (*i*) allocating the necessary resources, (*ii*) partitioning the matrix into Π sub-matrices, and (*iii*) offloading the data onto the available Σ Slave processes. Each Slave σ (with MPI rank $1 \leq \sigma \leq \Sigma$) proceeds by randomly generating the initial population of the GA; **(2)** each Slave executes the assigned wMEC sub-task by means of an independent GA instance. If multiple cores are available, the Slave processes are executed in a parallel fashion; **(3)** as soon as the wMEC sub-tasks are terminated, the Master process recombines the sub-solutions received from the Slaves, and yields the complete wMEC solution.

According to the GA settings analysis provided in [40], we used here 100 individuals, tournament selection with size equal to 10 individuals, crossover and mutation rates equal to 0.9 and 0.05, respectively. Finally, the elitism strategy is exploited to copy the best individual from the current population into the next one without undergoing the genetic operators.

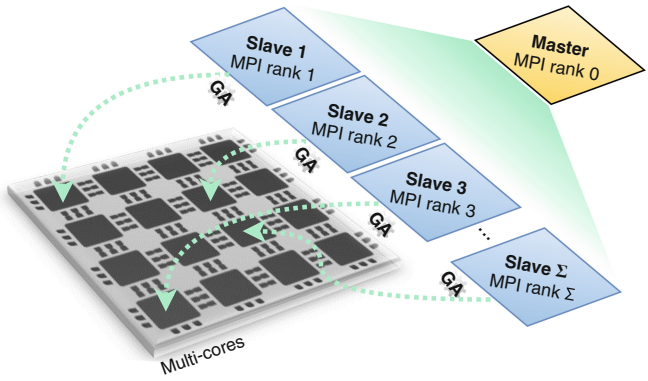

Fig. 2. Graphical representation of the Master-Slave approach implemented by GenHap: the Master process handles all the Σ Slaves by sending one or more sub-partitions to each Slave, which then solves the assigned wMEC sub-task leveraging a core.

3 Test Battery and Results

In what follows, we present some computational results obtained by considering different sequencing technologies, namely: Illumina NovaSeq, Roche/454, PacBio RS II, and ONT MinION. In [40], GenHap was shown to be faster than HapCol achieving up to 20× speed-up on PacBio RS II instances, reconstructing haplotypes characterized by a very low haplotype error rate. Moreover, GenHap was capable of solving in about 10 min a real PacBio RS II instance characterized by #SNPs $\simeq$ 28000 and #reads $\simeq$ 140000, with average and maximum coverages equal to 29 and 565, respectively. Notice that a direct comparison with the only other parallel method, pWhatsHap [4], was not possible since the source code of that tool is no longer publicly available.

In order to assess the computational performance of GenHap, we used the General Error-Model based SIMulator (GemSIM) toolbox [25] to generate different instances of synthetic (yet realistic) data, compliant with these sequencing technologies. GemSIM generates the instances relying on empirical error models and distributions learned from real NGS data. A detailed description of the whole pipeline is described in [40]. For each sequencing technology, we generated a single instance varying the following parameters: (i) #SNPs $\in$

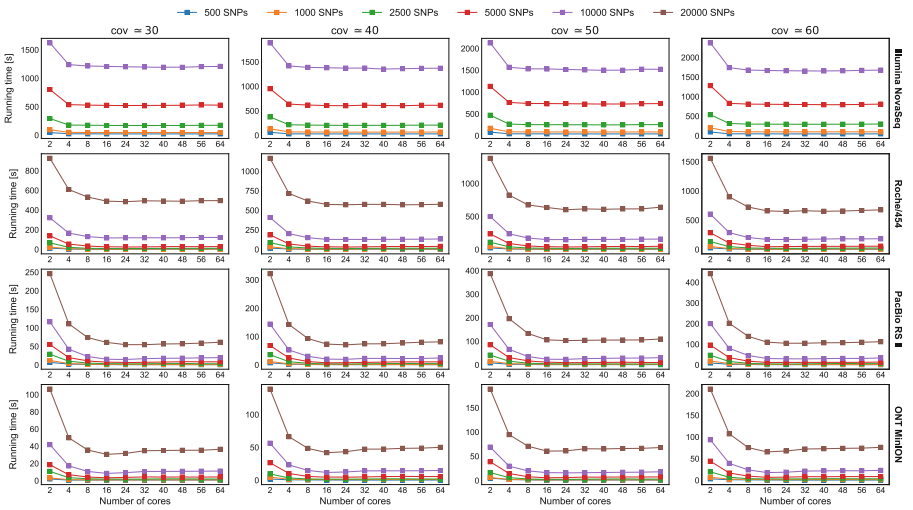

Fig. 3. Comparison of the running time required by GenHap on sequencing data generated by four sequencing technologies (Illumina NovaSeq, Roche/454, PacBio RS II, ONT MinION) by varying the coverage values. Note that the instances generated using the Illumina NovaSeq technology and characterized by #SNPs = 20000 required more RAM than the amount of memory available on the computing nodes used for the tests. The tests were executed by increasing the number of cores exploited to run GenHap, to evaluate the scalability of the implementation based on distributed computing.

$\{500, 1000, 5000, 10000, 20000\}$; (*ii*) cov $\in \{\sim 30\times, \sim 40\times, \sim 50\times, \sim 60\times\}$; (*iii*) average $f_{\text{SNPs}} = 200$ (i.e., one SNP every 200 bp exists [13,27]).

These instances were used to evaluate the scalability of GenHap by varying the number of cores, that is, #cores $\in \{2, 4, 8, 16, 24, 32, 40, 48, 56, 64\}$. All tests were performed on the MARCONI supercomputer, which is based on the Lenovo NeXtScale System® platform (Morrisville, NC, USA), provided by the Italian inter-university consortium CINECA (Bologna, Italy). Three different partitions running on CentOS 7.2 are available on this supercomputer:

*A*1 Broadwell (BDW) partition consists of 720 compute nodes, each one equipped with 2 Intel® Xeon® E5-2697 v4 (18 cores at 2.30 GHz) and 128 GB RAM;
*A*2 Knights Landing (KNL) partition consists of 3600 compute nodes, each one equipped with an Intel® Knights Landing (68 cores at 1.40 GHz and 16 GB MCDRAM), which is the next-generation of the Intel® Xeon Phi™ product family for many-core architectures, and 93 GB RAM;
*A*3 Skylake (SKL) partition consists of 92 compute nodes, each one equipped with 2 Intel® Xeon® 8160 CPU (18 cores at 2.10 GHz) and 192 GB RAM.

Our analysis was carried out by using the computing nodes of partition *A*2, which was chosen due to the availability of a higher number of computing cores.

Figure 3 depicts the running times required by GenHap to infer the pairs of haplotypes. As expected, the processing time generally decreases along with the

read length: indeed, according to Eq. (1), the same coverage can be obtained by means of long reads coupled with a lower number of reads. This circumstance leads to a lower number of sub-problems to be solved, reducing the necessary computational effort. Moreover, the lowest running time is achieved on the instances generated relying on the ONT MinION, which is capable of producing long reads (up to 6000 bp) with accuracy greater than 92%. As a matter of fact, the amount of SNPs to be corrected decreases when reads characterized by high accuracy are taken into account, allowing the GA instances to have a fast convergence to the optimal solutions. The results obtained for each sequencing platform are summarized as follows. (*i*) Illumina NovaSeq: independently from the coverage, the lowest running time is achieved by exploiting 24 cores to parallelize the GA instances when #SNPs = 10000. When #SNPs < 10000, 16 or 24 cores require the minimum running time to infer the haplotype pairs; (*ii*) Roche/454: when #SNPs $\geq$ 5000, the best GenHap's performance is achieved by exploiting 16 or 24 cores, otherwise the best choice is 24 cores; (*iii*) PacBio RS II: in every test, the fastest executions are generally obtained by exploiting 24 cores to parallelize the GA instances, except when #SNPs = 500 is taken into account. In this case, the running time decreases when 16 cores are exploited to effectively distribute the GA instances on multiple cores. Since the reads generated by relying on this technology have a low accuracy (approximately 87%), which makes the problem more difficult to be solved (i.e., the amount of SNPs to be corrected increases), the scalability of GenHap is emphasized; (*iv*) ONT MinION: in all tests, the best choice is 16 cores that allow for efficiently distributing the computational load. In every test, a number of cores greater than 24 does not reduce the running time since the overhead introduced by MPI is not entirely mitigated by the required computational load. Furthermore, when the number of sub-problems is lower than the number of available cores, our Master-Slave approach exploits a number of cores equal to the number of sub-problems. On the one hand, when technologies producing short reads are considered, the number of haplotype blocks increases along with #SNPs. Since these blocks are solved sequentially and are generally characterized by a number of sub-problems lower than the available cores, 16 or 24 cores allow for balancing the computational load. On the other hand, technologies producing long reads generate a small number of reads that lead to a low number of sub-problems to be solved. Notice that exploiting the accuracy of the reads produced using Illumina NovaSeq, Roche/454 and ONT MinION, the GA instances have a fast convergence to the optimal solutions requiring only a dozen of generations.

4 Conclusion and Future Trends

In this paper, we presented a complete overview on the currently available HA computational tools, focusing on the potential of HPC in this research area. In particular, we investigated the computational performance of GenHap [40], a recent evolutionary method leveraging multi-core architectures.

As a future development, we plan to extend GenHap to deal with HA in organisms characterized by different ploidity. Differently from diploid organisms

having two copies of each chromosome set, polyploid organisms have multiple copies of their chromosome sets. Polyploidy has gained scientific interest in the study of the ongoing species diversification phenomena [28]. This characteristic is mainly present in plant genomes, but also in animals (such as salmonid fishes and African clawed frogs) [34]. In these comparative genomic studies, haplotype-aware assemblies play a crucial role in elucidating genetic and epigenetic regulatory evolutionary aspects. Unfortunately, the computational burden of the HA problem is emphasized in the case of polyploid haplotypes with respect to diploids [9]. Therefore, HPC represents a key element for efficient, accurate, and scalable methods for HA of both diploid and polyploid organisms.

An interesting future trend in Genome Analysis is related to its connection with machine learning. As a matter of fact, deep learning has been successfully applied in population genetic inference and learning informative features of data [37]. Combining population genetics inference and HA can provide insights on patterns regarding the genetic diversity in DNA polymorphism data, especially for rapid adaptation and selection [15].

An additional issue worth of notice is that, although the integration of various types of information (e.g., electronic health records and genome sequences) conveys a wealth of information, it is giving rise to unique challenges in bioinformatics analysis even in terms of secure genomic information sharing [3]. With reference to secure Genome-Wide Association Study (GWAS) in distributed computing environments, multi-party computation schemes based on conventional cryptographic techniques achieve limited performance in practice [7]. Therefore, HPC could become an enabling factor also in this context.

Acknowledgment. We acknowledge the CINECA for the availability of High Performance Computing resources and support.

References

1. Bansal, V., Bafna, V.: HapCUT: an efficient and accurate algorithm for the haplotype assembly problem. Bioinformatics **24**(16), i153–i159 (2008)
2. Benedettini, S., Roli, A., Di Gaspero, L.: Two-level ACO for haplotype inference under pure parsimony. In: Dorigo, M., Birattari, M., Blum, C., Clerc, M., Stützle, T., Winfield, A.F.T. (eds.) ANTS 2008. LNCS, vol. 5217, pp. 179–190. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87527-7_16
3. Bianchi, L., Liò, P.: Opportunities for community awareness platforms in personal genomics and bioinformatics education. Brief. Bioinform. **18**(6), 1082–1090 (2016)
4. Bracciali, A., et al.: pWhatsHap: efficient haplotyping for future generation sequencing. BMC Bioinform. **17**(Suppl. 11), 342 (2016)
5. Browning, S.R., Browning, B.L.: Rapid and accurate haplotype phasing and missing-data inference for whole-genome association studies by use of localized haplotype clustering. Am. J. Hum. Genet. **81**(5), 1084–1097 (2007)
6. Chen, Z.Z., Deng, F., Wang, L.: Exact algorithms for haplotype assembly from whole-genome sequence data. Bioinformatics **29**(16), 1938–1945 (2013)
7. Choi, Y., Chan, A.P., Kirkness, E., Telenti, A., Schork, N.J.: Comparison of phasing strategies for whole human genomes. PLoS Genet. **14**(4), e1007308 (2018)

8. Daly, M.J., Rioux, J.D., Schaffner, S.F., Hudson, T.J., Lander, E.S.: High-resolution haplotype structure in the human genome. Nat. Genet. **29**(2), 229 (2001)
9. Das, S., Vikalo, H.: SDhaP: haplotype assembly for diploids and polyploids via semi-definite programming. BMC Genomics **16**(1), 260 (2015)
10. Delaneau, O., Marchini, J., Zagury, J.F.: A linear complexity phasing method for thousands of genomes. Nat. Methods **9**(2), 179 (2012)
11. Duitama, J., Huebsch, T., McEwen, G., Suk, E., Hoehe, M.: ReFHap: a reliable and fast algorithm for single individual haplotyping. In: Proceedings of the First ACM International Conference on Bioinformatics and Computational Biology, pp. 160–169. ACM (2010)
12. Edge, P., Bafna, V., Bansal, V.: HapCUT2: robust and accurate haplotype assembly for diverse sequencing technologies. Genome Res. **27**(5), 801–812 (2017)
13. Gabriel, S.B., et al.: The structure of haplotype blocks in the human genome. Science **296**(5576), 2225–2229 (2002)
14. Greenberg, H.J., Hart, W.E., Lancia, G.: Opportunities for combinatorial optimization in computational biology. INFORMS J. Comput. **16**(3), 211–231 (2004)
15. Hermisson, J., Pennings, P.S.: Soft sweeps and beyond: understanding the patterns and probabilities of selection footprints under rapid adaptation. Methods Ecol. Evol. **8**(6), 700–716 (2017)
16. Jain, M., Fiddes, I.T., Miga, K.H., Olsen, H.E., Paten, B., Akeson, M.: Improved data analysis for the MinION Nanopore sequencer. Nat. Methods **12**(4), 351 (2015)
17. Jain, M., et al.: Nanopore sequencing and assembly of a human genome with ultra-long reads. Nat. Biotechnol. **36**(4), 338 (2018)
18. Kuleshov, V.: Probabilistic single-individual haplotyping. Bioinformatics **30**(17), i379–i385 (2014)
19. Kuleshov, V., et al.: Whole-genome haplotyping using long reads and statistical methods. Nat. Biotechnol. **32**(3), 261–266 (2014)
20. Lander, E.S., Waterman, M.S.: Genomic mapping by fingerprinting random clones: a mathematical analysis. Genomics **2**(3), 231–239 (1988)
21. Levy, S., et al.: The diploid genome sequence of an individual human. PLoS Biol. **5**(10), e254 (2007)
22. Loh, P.R., et al.: Reference-based phasing using the haplotype reference consortium panel. Nat. Genet. **48**(11), 1443 (2016)
23. Luo, C., Tsementzi, D., Kyrpides, N., Read, T., Konstantinidis, K.T.: Direct comparisons of Illumina vs. Roche 454 sequencing technologies on the same microbial community DNA sample. PloS One **7**(2), e30087 (2012)
24. Maisto, D., Donnarumma, F., Pezzulo, G.: Divide et impera: subgoaling reduces the complexity of probabilistic inference and problem solving. J. R. Soc. Interface **12**(104), 20141335 (2015)
25. McElroy, K.E., Luciani, F., Thomas, T.: GemSIM: general, error-model based simulator of next-generation sequencing data. BMC Genomics **13**(1), 74 (2012)
26. Na, J.C., Lee, J.C., Rhee, J.K., Shin, S.Y.: PEATH: single individual haplotyping by a probabilistic evolutionary algorithm with toggling. Bioinformatics **34**(11), 1801–1807 (2018)
27. Nachman, M.W.: Single nucleotide polymorphisms and recombination rate in humans. Trends Genet. **17**(9), 481–485 (2001)
28. Otto, S.P., Whitton, J.: Polyploid incidence and evolution. Annu. Rev. Genet. **34**(1), 401–437 (2000)
29. Patterson, M., et al.: WhatsHap: weighted haplotype assembly for future-generation sequencing reads. J. Comput. Biol. **22**(6), 498–509 (2015)

30. Pirola, Y., Zaccaria, S., Dondi, R., Klau, G., Pisanti, N., Bonizzoni, P.: HapCol: accurate and memory-efficient haplotype assembly from long reads. Bioinformatics **32**(11), 1610–1617 (2015)
31. Quail, M.A., et al.: A large genome center's improvements to the Illumina sequencing system. Nat. Methods **5**(12), 1005 (2008)
32. Rhoads, A., Au, K.F.: PacBio sequencing and its applications. Genomics Proteomics Bioinform. **13**(5), 278–289 (2015)
33. Roberts, R.J., Carneiro, M.O., Schatz, M.C.: The advantages of SMRT sequencing. Genome Biol. **14**(6), 405 (2013)
34. Rodriguez, F., Arkhipova, I.R.: Transposable elements and polyploid evolution in animals. Curr. Opin. Genet. Dev. **49**, 115–123 (2018)
35. Rundo, L., et al.: MedGA: a novel evolutionary method for image enhancement in medical imaging systems. Expert Syst. Appl. **119**, 387–399 (2019)
36. Senol Cali, D., Kim, J.S., Ghose, S., Alkan, C., Mutlu, O.: Nanopore sequencing technology and tools for genome assembly: computational analysis of the current state, bottlenecks and future directions. Brief. Bioinform., bby017 (2018)
37. Sheehan, S., Song, Y.S.: Deep learning for population genetic inference. PLoS Comput. Biol. **12**(3), e1004845 (2016)
38. Sims, D., Sudbery, I., Ilott, N.E., Heger, A., Ponting, C.P.: Sequencing depth and coverage: key considerations in genomic analyses. Nat. Rev. Genet. **15**(2), 121 (2014)
39. Tangherloni, A., Rundo, L., Spolaor, S., Cazzaniga, P., Nobile, M.S.: GPU-powered multi-swarm parameter estimation of biological systems: a master-slave approach. In: Proceedings of the 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP), pp. 698–705. IEEE (2018)
40. Tangherloni, A., et al.: GenHap: a novel computational method based on genetic algorithms for haplotype assembly. BMC Bioinform. (2018, in press)
41. Wang, R., Wu, L., Li, Z., Zhang, X.: Haplotype reconstruction from SNP fragments by minimum error correction. Bioinformatics **21**(10), 2456–2462 (2005)
42. Wang, T.C., Taheri, J., Zomaya, A.Y.: Using genetic algorithm in reconstructing single individual haplotype with minimum error correction. J. Biomed. Inform. **45**(5), 922–930 (2012)
43. Zhang, K., Calabrese, P., Nordborg, M., Sun, F.: Haplotype block structure and its applications to association studies: power and study designs. Am. J. Hum. Genet. **71**(6), 1386–1394 (2002)