HybHap: A Fast and Accurate Hybrid Approach for Haplotype Inference on Large Datasets

Rogério S. Rosa and Katia S. Guimarães

Informatics Center
Federal University of Pernambuco, UFPE
Recife, Brazil
{rsr,katiag}@cin.ufpe.br

Abstract. We introduce HybHap, a new approach for haplotype inference problem on large genotype datasets. HybHap is a hybrid method, based on the Parsimonious tree-grow idea, which resorts to Markov chains, in order to maximize the probability that the haplotypes will be shared by more genotypes in the dataset. Several experiments with large biological datasets taken from HapMap were performed to compare HybHap with two well known algorithms: fastPHASE and PTG. The results show that HybHap is a rather robust, reliable, and efficient method that runs orders of magnitude faster than the others, producing results of comparable accuracy, hence being much more suitable to deal with the challenge of genome wide tasks.

Keywords: haplotype inference, hybrid algorithms, markov chains, tree-grow.

1 Introduction

An alteration of one isolated nucleotide base which occurs with considerable frequence in the DNA of a given population is known as Single Nucleotide Polymorphism (SNP) [1] [2] [3]. Occurrences of SNPs have been associated with specific phenotypic traits and also with several illnesses [4]. Hence, it is important to map the occurrences of SNPs, but that has shown to be a huge challenge.

An haplotype can be defined as a set of SNPs from a copy of a specific chromosome. Much of the difficulty of finding these alterations is due to the lack of haplotype data in large scale, mostly because of the high cost of collecting that information directly.

One possible way of acquiring haplotype data is to infer them from genotype data, which are highly abundant. That motivates the Haplotype Inference (HI) problem, whose computational cost depends on the evolutionary model considered. One such model is based on the biological sound Parsimony Principle, but it is proved to be NP-hard [5], meaning that all algorithms currently known can only solve it in time that is exponential on the number and size of the DNA sequences, which is prohibitive. Several computational methods were developed aiming at finding solutions that may be biologically plausible, but they usually

J.C. Setubal and N.F. Almeida (Eds.): BSB 2013, LNBI 8213, pp. 24–35, 2013.

present high computational costs. In view of real applications, the current challenge is to infer haplotypes from large scale genotypes. Hence, computationally efficient methods with acceptable accuracy are in great demand. In this paper we present a hybrid approach that combines the efficiency of Parsimonious Tree-Grow with Markov chain choices. The result is a method that is orders of magnitude faster than the known methods, delivering results of comparable accuracy.

The rest of this paper is organized as follows. Section 2 gives a formal definition of the Haplotype Inference Problem and discusses related works. In Section 3, the proposed Markov chain used by the HybHap method is presented. The HybHap approach is introduced in Section 4. In Section 5 there is a description of the datasets and how the benchmark was organized, and in Section 6, the results of experiments are provided to demonstrate the accuracy and efficiency of the proposed hybrid method. Finally, we present several remarks, concluding the paper in Section 7.

2 Haplotype Inference Methods

We adopt the notation used by Rosa and Guimarães [6]. A genotype can be computationally represented by a vector on the alphabet {0, 1, 2}, where a symbol 2 represents an ambiguous site. Then a genotype vector g, with n sites, can be explained by two haplotype vectors h_1 and h_2, where each site $h_1(i)$ and $h_2(i)$, $1 \leq i \leq n$, has $h_1(i)$, $h_2(i) \in \{0,1\}$, and follows the rule given by: (A) $h_1(i) = h_2(i) = g(i)$, if $g(i) \in \{0,1\}$; and (B) $h_1(i) = 1 - h_2(i)$, if $g(i) = 2$. The sites of g that have a symbol 0 or 1 are called homozygous (non ambiguous sites) and those with a symbol 2 are called heterozygous (ambiguous sites). The Haplotype Inference Problem basically consists of finding, for each genotype g, haplotypes h_1 and h_2 such that h_1 and h_2 explain g in a biologically plausible way. For instance, if $g = (0, 1, 2, 2, 1, 2)$, possible solutions are $h_1 = (0, 1, 0, 0, 1, 0)$ and $h_2 = (0, 1, 1, 1, 1, 1)$, or else $h_1 = (0, 1, 0, 1, 1, 1)$ and $h_2 = (0, 1, 1, 0, 1, 0)$, among other possibilities. It is easy to see that there are 2^{h-1} candidate haplotype pairs to explain g, where h is the number of ambiguous sites in g. Obviously, there are many plausible solutions for a given input g, so a biological criterion is needed to define a good solution.

There are two main biological models used to infer haplotypes: Pure Parsimony and Perfect Phylogeny. Inferring haplotypes assuming perfect phylogeny was shown to be a linear problem [7]. However the assumption that the DNA sequences were not subject to recombination events is not realistic.

Haplotype inference by pure parsimony principle (HIPP) has been used by many approaches because of its innate simplicity and biological soundness. As said before, unfortunately the HIPP problem is NP-hard [8]. Some approaches based on Integer Programming have been proposed for it [9] [10] [11] [12].

Another method for the HIPP problem is the Parsimonious Tree-Grow (PTG) method [13], which explains a set of m genotypes of length n in time $O(m^2 n)$. In the PTG method a tree is constructed, where each edge is labelled by a haplotype symbol (0 or 1), and nodes contain the genotypes (id) that are explained

by haplotypes formatted by a trace from the root to that specific node. Many operations in PTG are random, so it is necessary to run the method many times, selecting the best solution using some metric, in order to have reliable results.

Methods based on Markov chain models have been proposed successfully. These methods basically build a Markov chain in which each state is associated to a symbol (0 or 1) and the transition probabilities are calculated from the input data. Heuristics based on Dynamic Programming and Expectation Maximization algorithms are also applied [14] [15] [16] [17].

Statistical methods have considered the Parsimony Principle as accessory. PHASE [18] and fastPHASE [19] are considered good classical approaches for the HI Problem. These methods use maximum likelihood to estimate haplotype frequencies. The objective is to estimate the maximum value of this likelihood function. Such methods are stochastic, and each execution of the program may result in a different solution, since the derivations are dependent on the initial configuration, which is randomly selected. Basically, fastPHASE is a variation of PHASE for resolving large data sets.

3 Computing the Markov Chain

The probability that a haplotype fragment will be part of a solution, considering the parsimony criterion can be efficiently estimated using a Markov chain. Given a genotype matrix G, with m rows and n columns, a Markov chain C is created with $2n+2$ states, each state representing the start (C_{start}) or the end of the chain (C_{end}), or a possible symbol s (0 or 1) in the j-th site of a haplotype fragment, $1 \leq j \leq n$, ($C_j(s)$). There are three types of state transitions: $(C_{start}, C_1(s))$, $(C_{j-1}(s_1), C_j(s_2))$, and $(C_n(s), C_{end})$.

The initial probabilities are computed as an *a priori* probability of symbol s occurring in the first site of all the $2m$ haplotypes to be inferred from G (1). The absolute frequency of symbol s being in the first site of the matrix is calculated according to Equation 2.

$$(C_{start}, C_1(s)) = A(1, s)/(2m) \tag{1}$$

$$A(j, s) = \sum_{y=1}^{m} f_1(G(y, j), s), \tag{2}$$

with

$$f_1(x, s) = \begin{cases} 2, & \text{if } x = s \\ 1, & \text{if } x = 2 \\ 0, & \text{otherwise} \end{cases}$$

The transition probabilities whose source state is not the initial state (C_{start}) and the destination is not the final state (C_{end}) are denoted by $(C_{j-1}(s_1), C_j(s_2))$, where $2 \leq j \leq n$. These are conditional probabilities: probability of s_2 occurring in the j-th site of the $2m$ haplotypes inferred from G, given that s_1 occurred in

the $(j-1)$-th site of said set of haplotypes (Equation 3). That depends on the absolute frequency of haplotypes inferred with s_1 in the $(j-1)$-th site (1), and an estimation of the expected frequency of symbol s_2 in the j-th site of those same haplotypes (Equation 4).

$$(C_{j-1}(s_1), C_j(s_2)) = B(j, s_1, s_2)/A(j-1, s_1) \tag{3}$$

$$B(j, s_1, s_2) = \sum_{y=1}^{m} f_2(G(y, j-1), G(y, j), s_1, s_2), \tag{4}$$

with

$$f_2(x_1, x_2, s_1, s_2) = \begin{cases} 2, & \text{if } x_1 = s_1 \text{ and } x_2 = s_2 \\ 0.5, & \text{if } x_1 = 2 \text{ and } x_2 = 2 \\ 1, & \text{if } x_1 = 2 \text{ and } x_2 = s_2 \\ 1, & \text{if } x_1 = s_1 \text{ and } x_2 = 2 \\ 0, & \text{otherwise} \end{cases}$$

After constructing Markov chain C as described above, a tree T is computed which contains the $2m$ haplotypes that resolve G. The HybHap method uses the information contained in C to choose more promising branches, trying to keep T with the minimum possible number of branches, so as to approach an optimal solution, according to the Pure Parsimony criterion.

4 The HybHap Method

A tree T, which has $n+1$ layers, each one denoted by $T(j)$, is computed. A layer can have $2m$ nodes in the worst case (maximum possible number of distinct haplotypes to be inferred). A node in layer $T(j)$ is denoted by $T_{(j,k)(s)}$, where s is the node type (0 or 1), and k is the number sequence of the node in layer j.

A node $T_{(j,k)(s)}$ is labelled by $(i_{r_1}, i_{r_2}, ..., i_{r_g})$, $1 \leq g \leq m$, which represents the genotype fragments explained by that node. The root is labelled by all genotype Ids $(i_1, i_2, ..., i_m)$. For each layer j of T, for each node in j, for each genotype Id i_r in the label of the current node, if G, in site j of genotype, has value 1 (0), then a node of type 1 (0) is created on the next layer connected to the current node, and it is labelled by all genotype Ids in the present node that have value 1 (0) in site j. If the value in G in layer j and genotype Id i_r has value 2, then it is checked if a value 2 was previously explained for genotype i_r. If that is not the case, then site j in genotype i_r is resolved by adding two new nodes in the $(j+1)$-th layer, connected to the current node, and labelled i_r. In case a value 2 has been previously resolved, then the genotype Id and the current node are reserved to be processed after the current layer is treated. An example of tree is illustrated in Figure 1.

Random operations may occur in the processing of the genotypes and nodes reserved in a layer. In HybHap the Markov chain C constructed will be used to decide which haplotype fragment is the most promising one. Each trace, from

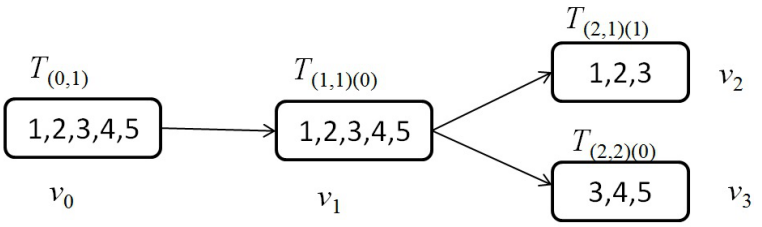

Fig. 1. Node v_0 is the root of tree T; this node is in level 0 and is node number 1 of that level ($T_{(0,1)}$). The root is labelled by five genotypes Ids (1,2,3,4, and 5), representing all five genotypes in the input. Node v_1, denoted by $T_{(1,1)(0)}$ is in level 1 and it is the node number 1 of that level; this node is of type 0. Node v_3, denoted by $T_{(2,2)(0)}$, is in level 2 and it is the node number 2 of that level, this node is of type 0, and it is labelled by genotype Ids 3,4, and 5.

the root to the current node in T, is a valid path in C. There are three situations in which random choices may be needed, the others are symmetric; in those situations, new nodes are computed through C, and the choice is based on the maximum probability found. In case we need to choose among existing nodes to explain the reserved genotype, then we compute the Euclidean distance between the sites that have not yet been processed in the reserved genotype and all sites that have not been processed in the genotypes that are partially resolved by candidate nodes, the choice is based on the least distance. When the probabilities or distances are the same between candidate nodes, then a random choice is needed, but the chances of that actually occurring are slim.

The HybHap method (Algorithm 1) has three main steps: Initialization, Resolution of genotype prefix with known solution (genotype fragments that have only homozygous sites or one heterozygous site), as described in Algorithm 2, and explanation of genotype fragments that have no previous resolution (more than one heterozygous site), as described in Algorithm 3. In initialization the Markov chain is computed as described before, the root is created and labelled with all genotype Ids of G.

The 2 explains the genotype fragments (prefix) that have at most one heterozygous site. In this case, when a site with symbol 2 is resolved for genotype i, we make $f(i) = true$. All genotypes marked in the prior step ($f(i) = true$), for a specific SNP (a layer of tree T), will be processed after all non-ambiguous genotype fragments of that layer are resolved.

In Algorithm 3, the fragments of genotypes reserved before are explained. In 2, a genotype i was associated to two nodes (two is the maximum number of nodes that can explain a genotype with at least one heterozygous site). In this Algorithm 3, a Markov chain is used to decide which is the best branching option. Equation 5 is used, in which $P(v_1)$ denotes the probability that the haplotype fragment represented by node v_1 will be part of solution, according to the parsimony criterion (conservation), and $t(v_1)$ denotes the type of v_1. There are three cases in which the Markov chain is applied: (1) There are no branches

growing from v_1 and v_2; (2) There is a single branch (v'_1, v'_2) growing from each of v_1 and v_2, both of the same type s; and (3) There are two branches growing from v_1 but no branches growing from v_2; the other cases are symmetric. Those three situations are addressed in Algorithm 3, and illustrated in Figure 2-C.

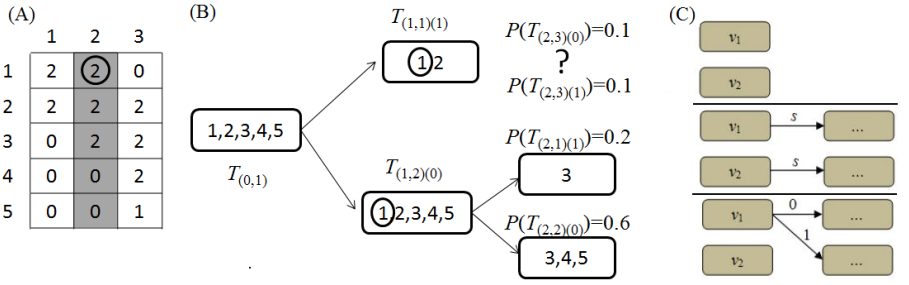

Fig. 2. Example of Algorithm Execution

$$P(v_1)(C_j(t(v_1)), C_{j+1}(s_1)) + P(v_2)(C_j(t(v_2)), C_{j+1}(s_2)) \quad (5)$$

After building tree T, the final solution can be recovered by tracing from the root to each leaf of T, concatenating the types of the nodes in the path. The result will be the haplotype matrix that explains G following the parsimony principle.

Figure 2-B illustrates an application of Markov chain C during the construction of tree T. First the root $T_{(0,1)}$ is created and labelled by all genotype Ids of the matrix identified in Figure 2-A. Then the nodes descendent from $T_{(0,1)}$ are created. Since no ambiguity has been resolved in T yet, nodes $T_{(1,1)(1)}$ and $T_{(1,2)(0)}$ are created to explain genotypes (1,2) and (1,2,3,4,5), respectively. Since in column 2, genotypes 1 and 2 have symbol 2, and sites of that type have been previously resolved for those genotype, their Ids are kept to be processed after all sites on the second column that do not present ambiguity or that have all previous sites without ambiguity are resolved. Hence, genotypes 3, 4, and 5 are resolved, by creating nodes $T_{(2,1)(1)}$ and $T_{(2,2)(0)}$.

After that, genotypes 1 and 2 are dealt with. There are two nodes, $T_{(2,1)(1)}$ and $T_{(2,2)(0)}$, branching from $T_{(1,2)(0)}$ that can explain genotype 1, and none from $T_{(1,1)(1)}$. In order to decide which of those nodes should be created branching from $T_{(1,1)(1)}$: $T_{(2,3)(0)}$ or $T_{(2,3)(1)}$. Markov chain C is then used to estimate which node maximizes the probability of being also used in the resolution of other genotypes (parsimony). In the case of this example, there are the following node combinations: $P(T_{(2,3)(0)} + T_{(2,1)(1)}) = 0.3$ and $P(T_{(2,3)(1)} + T_{(2,2)(0)}) = 0.7$. The choice is for the option that maximizes the probability, hence, node $T_{(2,3)(1)}$ is added branching from node $T_{(1,1)(1)}$.

Algorithm 1. HybHap

```
input  : a matrix of Genotypes G
output: a tree T
Initialization;
foreach layer j in T do
    foreach node v in current layer do
        KnownSolution(v, j);
    end
    UnknownSolution(j);
end
```

5 Experiments Design

The dataset used for the experiments was the same one used in a previous work where the performances of well known haplotype inference algorithms when dealing with data of different sizes and levels of conservation are compared [6]. It is comprised by sequences originally taken from the HapMap Project [20], which were collected from Chromosome 20 of population CEU (Caucasians resident in the state of Utah (USA) with northern European ancestry). The original dataset is composed of 13 subsets which vary in sequence length and in number of distinct haplotypes.

The set contains haplotypes of sizes 100, 200, 400, 800, 1600, and 3200 SNPs with 88 individuals, separated by size into classes A, B, C, D, E, and F, respectively. From each class, we chose randomly three instances.

The metrics used in the benchmark were Error Rate [21] and computational time. The Error Rate tells us about the capacity that one method has to correctly infer a haplotype set from a genotype set, based on a known haplotype set. The computational time is an empiric metric used to estimate computational costs; although it is not the best technique for it, in this case theoretical analysis cannot be applied to all methods.

For the comparison experiments, PTG was implemented in MATLAB 2008. Version 1.2.3 for Windows of fastPHASE was used. The experiments ran individually in a computer with an Intel Quad Core 2.33GHz processor, with 3GB of RAM. The results are shown in Table 1. For each experiment, the execution time (Time) and Error Rate (ER) attained are given.

6 Experiments Results

The measures described earlier were applied to each instance. Since PTG and fastPHASE have a stochastic behavior, for comparison purposes the average over 30 executions with every single dataset was used to establish the Error Rate. Although HybHap is not a deterministic algorithm, in practice it presents standard deviation virtually equal to zero, meaning that for different executions with the same input dataset (including the same genotype order), it generates the same haplotype set.

Comparing HybHap to PTG, considering Error Rate, the accuracy of HybHap and PTG were very close, slightly favoring HybHap for the larger datasets. In all

Algorithm 2. KnownSolution(v: a node, j: a SNP)

```
1  foreach genotype i in v do
2      if G(i,j+1)=2 and f(i)=true then
3          Associate to genotype i the node v;
4      else
5          if G(i, j + 1) = s and s ∈ {0, 1} then
6              if there is no branch with target node of type s growing from node v
               then
7                  Add a node of type s growing from node v and label this new
                   node with i;
8              else
9                  Add i to the set of Ids in the label of this node;
10             end
11         else
12             f(i) ← true;
13             for s = 0, 1 do
14                 if there is no node growing from v of type s then
15                     Add a new node from v in layer j + 1 of type s and add i to
                       the set of Ids of this new node;
16                 else
17                     Add i to the set of Ids of this node;
18                 end
19             end
20         end
21     end
22 end
```

cases, HybHap was much faster than PTG. For instance, HybHap solved the largest dataset (F) in about 3 minutes, while PTG took about 39 minutes to find a less accurate solution.

Comparing HybHap to the classical approach fastPHASE, we observed that the accuracy performances considering Error Rate were very close, and for the larger datasets in the benchmark, the differences between the Error Rates for the two methods were smaller than 2%. It is important to notice that, for the largest dataset, F, while HybHap needed only about 1 minute to find a solution with 13.67% of error, fastPHASE resolved this instance with 11.93% of error in about 72 hours. The difference of Error Rate in this case was 1.74%, however, the time necessary for fastPHASE to resolve it was approximately 1080 times longer than the time required by HybHap.

Figure 3 shows graphical comparisons of HybHap with fastPHASE and PTG, in regard to Error Rate (Figure 3-A) and computational time (Figure 3-B). Since the values for computational time are so different, the values in Figure 3-B are depicted in log scale. It can be seen that the time of fastPHASE grows much faster than HybHap, as the length of the sequences in the datasets increases. On the other hand, while the Error Rate of HybHap is always higher than that of fastPHASE, the difference in Error Rate is virtually constant.

Algorithm 3. UnknownSolution(j: a SNP)

```
1  foreach genotype i associated to a node pair (v1, v2) in SNP j do
2      if there is a single branch growing from v1 and v2 of different types or there
       are two branches and a single branch growing from v1 or v2 then
3          Add i to the set of Ids of a node that grows from a node that has a
           single branch (v1 or v2) and add i to the set of Ids of a node of opposite
           type that grows from the node left;
4      else
5          if there are two branches growing from v1 and two branches growing
           from v2 then
6              Compute the Euclidean distance among the unresolved suffix of i
               and the unresolved suffixes of genotypes explained by nodes growing
               from v1 and v2. Explain i in nodes that minimize the distance.
7          else
8              Compute value of Equation 5 for (s1 = 0, s2 = 1) and for
               (s1 = 1, s2 = 0). Take the pair (s1, s2) that maximizes v, if value of
               the two pairs are same, then select a pair randomly;
9              switch Ambiguity cases in v1 and v2 do
10                 case 1
11                     Grow a node of type s1 from v1, Grow a node of type s2
                       from v2. Add i to the set of Ids of these new nodes;
12
13                 case 2
14                     If s = s1, then add genotype i to the set of node v1' and
                       grow a node of type s2 from v2, labelled by i. Otherwise, do
                       the same symmetrically;
15
16                 case 3
17                     Add genotype i to the set of Ids of the node of type s1
                       growing from v1, and add a node of type s2 growing from
                       node v2, including i in the set of Ids of this new node (other
                       cases are symmetric);
18
19             endsw
20         end
21     end
22 end
```

Table 1. Comparison Results: Error Rate (ER) and Time in seconds (s), minutes (m) or hours (h)

Set	HybHap		PTG		fastPHASE	
	ER	Time	ER	Time	ER	Time
A	12.9%	02.88 s	13.5%	53.88 s	09.3%	00 h 30 m
B	09.9%	04.83 s	09.7%	01.62 m	05.8%	01 h 00 m
C	12.7%	12.28 s	12.6%	03.73 m	09.2%	02 h 30 m
D	11.1%	26.11 s	12.3%	07.67 m	08.7%	05 h 24 m
E	13.6%	01.06 m	14.2%	16.98 m	11.7%	18 h 00 m
F	13.7%	03.00 m	14.2%	39.00 m	11.9%	72 h 00 m

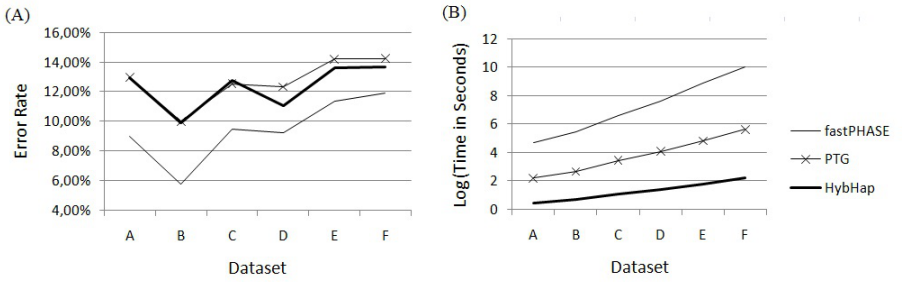

Fig. 3. Computational Time (in seconds) and Error Rate attained in each dataset class. Each class has 88 genotypes and different number of SNPs: (A) 100, (B) 200, (C) 400, (D) 800 and (F) 1600.

7 Discussion and Conclusion

In this paper we have proposed a hybrid method for haplotype inference. The proposed method is very stable, since in practice it presents a standard deviation of zero. In our experiments, the highest number of random operations for any instance was two, but that seldom happened. Due to that and to the efficiency of the operations in HybHap, its computational time is very low when compared to PTG and to fastPHASE.

With the enormous growth in the number of genomes available, efficient methods to deal with large datasets are highly desirable. There are many approaches to infer haplotypes with high quality, but they are applied only to small datasets, and it is not in line with the current inference requirements, which are on large scale. In face of that, HybHap presents desirable properties. The proposed method is computationally very efficient and in large datasets produces results with accuracy very close to that of more costly methods. That is most valuable, due to the growing number of genetic variation studies, which are performed by Computational Biology groups most of which have limited computational processing resources available.

An important point is the fact that PTG is based solely on parsimony, disregarding any other type of information or precondition about the genotype sequences, while methods based on Markov chains, such as fastPHASE, use the parsimony criterion as a help, applying additional techniques, models, and insights to find a biologically more plausible solution. Nonetheless, that combination leads to an extremely high computational time requirement. Hence, as the length of the genotype sequences grow, those methods become non-viable.

We also believe that the wide gap between the performances of HybHap and fastPHASE with respect to some cases of Error Rate is due to the fact that HybHap has no strategy to cluster together segments of different with similar characteristics regarding conservation. Since HybHap presents excellent computational cost, the original algorithm, presented in this paper, can be improved by strategies to associate similar regions, as it is done in fastPHASE, for instance. We are currently working on that aspect of the method. Missing data is another aspect that needs to be addressed.

The experimental analysis shows that the HybHap method is more adequate for dealing with long genome sequences. We are currently working on a theoretical argument for the fact that HybHap requires less computational time, as well as on improving its accuracy.

Acknowledgments. This work was developed with financial support from Brazilian sponsoring agency FACEPE (process no. PBPG-0070-1.03/10). The authors thank the anonymous reviewers for their insightful comments.

References

1. Sonis, S., Antin, J., Tedaldi, M., Alterovitz, G.: SNP-based Bayesian networks can predict oral mucositis risk in autologous stem cell transplant recipients. Oral Dis (2013)
2. Jin, Y., Lee, C.G.L.: Single Nucleotide Polymorphisms Associated with MicroRNA Regulation. Biomolecules 3(2), 287–302 (2013)
3. Martinez-Herrero, S., Martinez, A.: Cancer protection elicited by a single nucleotide polymorphism close to the adrenomedullin gene. J. Clin. Endocrinol. Metab (2013)
4. Wang, E.Y., Liang, W.B., Zhang, L.: Association between single-nucleotide polymorphisms in interleukin-12a and risk of chronic obstructive pulmonary disease. DNA Cell Biol. 31(9), 1475–1479 (2012)
5. Gusfield, D.: Haplotyping as perfect phylogeny: Conceptual framework and efficient solutions. In: International Conference on Research in Computational Molecular Biology (RECOMB), pp. 166–175 (2002)
6. Rosa, R.S., Guimarães, K.S.: Insights on haplotype inference on large genotype datasets. In: Ferreira, C.E., Miyano, S., Stadler, P.F. (eds.) BSB 2010. LNCS (LNBI), vol. 6268, pp. 47–58. Springer, Heidelberg (2010)
7. Ding, Z., Filkov, V., Gusfield, D.: A linear-time algorithm for the perfect phylogeny haplotyping (PPH) problem. In: Miyano, S., Mesirov, J., Kasif, S., Istrail, S., Pevzner, P.A., Waterman, M. (eds.) RECOMB 2005. LNCS (LNBI), vol. 3500, pp. 585–600. Springer, Heidelberg (2005)
8. Gusfield, D.: Inference of haplotypes from samples of diploids populations: Complexity and algorithms. Journal of Computational Biology 8, 305–324 (2001)

9. Gusfield, D.: Haplotype inference by pure parsimony. In: Baeza-Yates, R., Chávez, E., Crochemore, M. (eds.) CPM 2003. LNCS, vol. 2676, pp. 144–155. Springer, Heidelberg (2003)
10. Lancia, G., Pinotti, M.C., Rizzi, R.: Haplotyping populations by pure parsimony: Complex of exact and approximation algorithms. INFORMS J. Computing 16, 348–359 (2004)
11. Halldórsson, B.V., Bafna, V., Edwards, N., Lippert, R., Yooseph, S., Istrail, S.: A survey of computational methods for determining haplotypes. In: Istrail, S., Waterman, M.S., Clark, A. (eds.) DIMACS/RECOMB Satellite Workshop 2002. LNCS (LNBI), vol. 2983, pp. 26–47. Springer, Heidelberg (2004)
12. Brown, D.G., Harrower, I.M.: Integer programming approaches to haplotype inference by pure parsimony. IEEE/ACM Trans. Comput. Biol. Bioinformatics, pp. 141–154 (2006)
13. Li, Z., Zhou, W., Zhang, X., Chen, L.: A parsimonious tree-grow method for haplotype inference. Oxford Bioinformatics 21, 3475–3481 (2005)
14. Sun, S., Greenwood, C.M., Neal, R.M.: Haplotype inference using a bayesian hidden markov model. Genetic Epidemiology 31, 937–948 (2007)
15. Zhang, J.H., Wu, L.Y., Chen, J., Zhang, X.S.: A fast haplotype inference method for large population genotype data. Computational Statistics and Data Analysis 52, 4891–4902 (2008)
16. Eronen, L., Geerts, F., Toivonen, H.: Haplorec: efficient and accurate large-scale reconstruction of haplotypes. BMC Bioinformatics 7, 542 (2006)
17. Eronen, L., Geerts, F., Toivonen, H.: A markov chain approach to reconstruction of long haplotypes. In: Pacific Symposium on Biocomputing, pp. 104–115 (2004)
18. Stephens, M., Smith, N.J., Donnelly, P.: A new statistical method for haplotype reconstruction from population data. American Journal of Human Genetics 68, 59–62 (2001)
19. Scheet, P., Stephens, M.: A fast and flexible statistical model for large-scale population genotype data: applications to inferring missing genotypes and haplotypic phase. American Journal of Human Genetics 78, 629–644 (2006)
20. The International HapMap Consortium: The international hapmap project. Nature 426, 789–796 (2003)
21. Niu, T., Qin, Z.S., Xu, X., Liu, J.S.: Bayesian haplotype inference for multiple linked single-nucleotide polymorphism. American Journal of Human Genetics 70, 157–169 (2002)