SAT in Bioinformatics: Making the Case with Haplotype Inference

Inês Lynce[1] and João Marques-Silva[2]

[1] IST/INESC-ID, Technical University of Lisbon, Portugal
ines@sat.inesc-id.pt
[2] School of Electronics and Computer Science, University of Southampton, UK
jpms@ecs.soton.ac.uk

Abstract. Mutation in DNA is the principal cause for differences among human beings, and Single Nucleotide Polymorphisms (SNPs) are the most common mutations. Hence, a fundamental task is to complete a map of haplotypes (which identify SNPs) in the human population. Associated with this effort, a key computational problem is the inference of haplotype data from genotype data, since in practice genotype data rather than haplotype data is usually obtained. Recent work has shown that a SAT-based approach is by far the most efficient solution to the problem of haplotype inference by pure parsimony (HIPP), being several orders of magnitude faster than existing integer linear programming and branch and bound solutions. This paper proposes a number of key optimizations to the the original SAT-based model. The new version of the model can be orders of magnitude faster than the original SAT-based HIPP model, particularly on biological test data.

1 Introduction

Over the last few years, an emphasis in human genomics has been on identifying genetic variations among different people. This allows to systematically test common genetic variants for their role in disease; such variants explain much of the genetic diversity in our species. A particular focus has been put on the identification of Single Nucleotide Polymorphisms (SNPs), point mutations found with only two possible values in the population, and tracking their inheritance. However, this process is in practice very difficult due to technological limitations. Instead, researchers can only identify whether the individual is heterozygotic at that position, i.e. whether the values inherited from both parents are different. This process of going from genotypes (which include the ambiguity at heterozygous positions) to haplotypes (where we know from which parent each SNP is inherited) is called *haplotype inference*.

A well-known approach to the haplotype inference problem is called Haplotype Inference by Pure Parsimony (HIPP). The problem of finding such solutions is APX-hard (and, therefore, NP-hard) [7]. Current methods for solving the HIPP problem utilize Integer Linear Programming (ILP) [5,1,2] and branch and bound algorithms [10]. Recent work [8] has proposed the utilization of SAT for

A. Biere and C.P. Gomes (Eds.): SAT 2006, LNCS 4121, pp. 136–141, 2006.

the HIPP problem. Preliminary results are significant: on existing well-known problem instances, the SAT-based HIPP solution (SHIPs) is by far the most efficient approach to the HIPP problem, being orders of magnitude faster than *any* other alternative exact approach for the HIPP problem. Nevertheless, additional testing revealed that the performance of SHIPs can deteriorate for larger problem instances. This motivated the development of a number of optimizations to the basic SHIPs model, which are described in this paper. The results of the improved model are again very significant: the improved model can be *orders of magnitude faster* on biological test data than the basic model.

2 Haplotype Inference

A *haplotype* is the genetic constitution of an individual chromosome. The underlying data that forms a haplotype can be the full DNA sequence in the region, or more commonly the SNPs in that region. Diploid organisms pair homologous chromosomes, and thus contain two haplotypes, one inherited from each parent. The *genotype* describes the conflated data of the two haplotypes. In other words, an *explanation* for a genotype is a pair of haplotypes. Conversely, this pair of haplotypes explains the genotype. If for a given site both copies of the haplotype have the same value, then the genotype is said to be *homozygous* at that site; otherwise is said to be *heterozygous*.

Given a set $\mathcal{G}$ of n genotypes, each of length m, the haplotype inference problem consists in finding a set $\mathcal{H}$ of $2{\cdot}n$ haplotypes, such that for each genotype $g_i \in G$ there is at least one pair of haplotypes (h_j, h_k), with h_j and $h_k \in \mathcal{H}$ such that the pair (h_j, h_k) explains g_i. The variable n denotes the number of individuals in the sample, and m denotes the number of SNP sites. g_i denotes a specific genotype, with $1 \leq i \leq n$. (Furthermore, g_{ij} denotes a specific site j in genotype g_i, with $1 \leq j \leq m$.) Without loss of generality, we may assume that the values of a SNP are always 0 or 1. Value 0 represents the wild type and value 1 represents the mutant. A haplotype is then a string over the alphabet {0,1}. Moreover, genotypes may be represented by extending the alphabet used for representing haplotypes to {0,1,2}. Homozygous sites are represented by values 0 or 1, depending on whether both haplotypes have value 0 or 1 at that site, respectively. Heterozygous sites are represented by value 2.

One of the approaches to the haplotype inference problem is called Haplotype Inference by Pure Parsimony (HIPP). A solution to this problem minimizes the total number of distinct haplotypes used. Experimental results provide support for this method: the number of haplotypes in a large population is typically very small, although genotypes exhibit a great diversity.

3 SAT-Based Haplotype Inference

This section summarizes the model proposed in [8], where a more detailed description of the model (and associated optimizations) can be found. The SAT-based formulation models whether there exists a set $\mathcal{H}$ of haplotypes, with

$r = |\mathcal{H}|$ haplotypes, such that each genotype $g_i \in \mathcal{G}$ is explained by a pair of haplotypes in $\mathcal{H}$. The SAT-based algorithm considers increasing sizes for $\mathcal{H}$, from a lower bound lb to an upper bound ub. Trivial lower and upper bounds are, respectively, 1 and $2 \cdot n$. The algorithm terminates for a size of $\mathcal{H}$ for which there exists $r = |\mathcal{H}|$ haplotypes such that every genotype in $\mathcal{G}$ is explained by a pair of haplotypes in $\mathcal{H}$. In what follows we assume n genotypes each with m sites. The same indexes will be used throughout: i ranges over the genotypes and j over the sites, with $1 \leq i \leq n$ and $1 \leq j \leq m$. In addition r candidate haplotypes are considered, each with m sites. An additional index k is associated with haplotypes, $1 \leq k \leq r$. As a result, $h_{kj} \in \{0,1\}$ denotes the j^{th} site of haplotype k. Moreover, a haplotype h_k, is viewed as a m-bit word, $h_{k\,1} \ldots h_{k\,m}$. A valuation $v : \{h_{k\,1}, \ldots, h_{k\,m}\} \rightarrow \{0,1\}$ to the bits of h_k is denoted by h_k^v. Observe that valuations can be extended to other sets of variables.

For a given value of r, the model considers r haplotypes and seeks to associate two haplotypes (which can possibly represent the same haplotype) with each genotype g_i, $1 \leq i \leq n$. As a result, for each genotype g_i, the model uses *selector* variables for selecting which haplotypes are used for explaining g_i. Since the genotype is to be explained by *two* haplotypes, the model uses two sets, a and b, of r selector variables, respectively s^a_{ki} and s^b_{ki}, with $k = 1, \ldots, r$. Hence, genotype g_i is explained by haplotypes h_{k_1} and h_{k_2} if $s^a_{k_1 i} = 1$ and $s^b_{k_2 i} = 1$. Clearly, g_i is also explained by the same haplotypes if $s^a_{k_2 i} = 1$ and $s^b_{k_1 i} = 1$.

If a site g_{ij} of a genotype g_i is either 0 or 1, then this is the value required at this site and so this information is used by the model. If a site g_{ij} is 0, then the model requires, for $k = 1, \ldots, r$, $(\neg h_{kj} \vee \neg s^a_{ki}) \wedge (\neg h_{kj} \vee \neg s^b_{ki})$. If a site g_{ij} is 1, then the model requires, for $k = 1, \ldots, r$, $(h_{kj} \vee \neg s^a_{ki}) \wedge (h_{kj} \vee \neg s^b_{ki})$. Otherwise, one requires that the haplotypes explaining the genotype g_i have opposing values at site i. This is done by creating two variables, $g^a_{ij} \in \{0,1\}$ and $g^b_{ij} \in \{0,1\}$, such that $g^a_{ij} \neq g^b_{ij}$. In CNF, the model requires two clauses, $(g^a_{ij} \vee g^b_{ij}) \wedge (\neg g^a_{ij} \vee \neg g^b_{ij})$. In addition, the model requires, for $k = 1, \ldots, r$, $(h_{kj} \vee \neg g^a_{ij} \vee \neg s^a_{ki}) \wedge (\neg h_{kj} \vee g^a_{ij} \vee \neg s^a_{ki}) \wedge (h_{kj} \vee \neg g^b_{ij} \vee \neg s^b_{ki}) \wedge (\neg h_{kj} \vee g^b_{ij} \vee \neg s^b_{ki})$. Clearly, for each i, and for a or b, it is necessary that exactly one haplotype is used, and so exactly one selector variable be assigned value 1. This can be captured with cardinality constraints, $(\sum_{k=1}^{r} s^a_{ki} = 1) \wedge \left(\sum_{k=1}^{r} s^b_{ki} = 1\right)$. Since the proposed model is purely SAT-based, a simple alternative solution is used, which consists of the CNF representation of a simplified adder circuit [8].

The model described above is not effective in practice. Hence a number of improvements have been added to the basic model. One technique, common to other approaches to the HIPP problem, is the utilization of structural simplifications techniques, for reducing the number of genotypes and sites [2,8]. Another technique is the utilization of lower bound estimates, which reduce the number of iterations of the algorithm, but also effectively prune the search space. Finally, one additional key technique for pruning the search space is motivated by observing the existence of symmetry in the problem formulation. Consider two haplotypes h_{k_1} and h_{k_2}, and the selector variables $s^a_{k_1 i}$, $s^a_{k_2 i}$, $s^b_{k_1 i}$ and $s^b_{k_2 i}$. Furthermore, consider Boolean valuations v_x and v_y to the sites of haplotypes h_{k_1}

and h_{k_2}. Then, $h_{k_1}^{v_x}$ and $h_{k_2}^{v_y}$, with $s_{k_1 i}^a s_{k_2 i}^a s_{k_1 i}^b s_{k_2 i}^b = 1001$, corresponds to $h_{k_1}^{v_y}$ and $h_{k_2}^{v_x}$, with $s_{k_1 i}^a s_{k_2 i}^a s_{k_1 i}^b s_{k_2 i}^b = 0110$, and one of the assignments can be eliminated. To remedy this, one possibility is to enforce an ordering of the Boolean valuations to the haplotypes [1]. Hence, for any valuation v to the problem variables we require $h_1^v < h_2^v < \ldots < h_r^v$ (see [8] for further details).

4 Improvements to SAT-Based Haplotype Inference

Motivated by an effort to apply the SHIPs model to biological test data, we were able to identify a number of additional improvements to the basic model. For difficult problem instances, the run time is very sensitive to the number of g variables used. The basic model creates two variables for each heterozygous site. One simple optimization is to replace each pair of g variables associated with a heterozygous site, g_{ij}^a and g_{ij}^b, by a single variable t_{ij}. Consequently, the new set of constraints becomes, $(h_{kj} \vee \neg t_{ij} \vee \neg s_{ki}^a) \wedge (\neg h_{kj} \vee t_{ij} \vee \neg s_{ki}^a) \wedge (h_{kj} \vee t_{ij} \vee \neg s_{ki}^b) \wedge (\neg h_{kj} \vee \neg t_{ij} \vee \neg s_{ki}^b)$. Hence, if selector variable s_{ki}^a is activated (i.e. assumes value 1), then h_{kj} is equal to t_{ij}. In contrast, if selector variable s_{ki}^b is activated, then h_{kj} is the complement of t_{ij}. Observe that, since the genotype has at least one heterozygous site, then it must be explained by *two different* haplotypes, and so s_{ki}^a and s_{ki}^b cannot be simultaneously activated.

The basic model utilizes lower bounds, which are obtained by identifying incompatibility relations among genotypes. These incompatibility relations find other applications. Consider two *incompatible* genotypes, g_{i_1} and g_{i_2}, and a candidate haplotype h_k. Hence, if either $s_{ki_1}^a$ or $s_{ki_1}^b$ is activated, and so h_k is used for explaining genotype g_{i_1}, then haplotype h_k *cannot* be used for explaining g_{i_2}; hence both $s_{ki_2}^a$ and $s_{ki_2}^b$ *must not* be activated. The implementation of this condition is achieved by adding the following clauses for each pair of incompatible genotypes g_{i_1} and g_{i_2} and for each candidate haplotype h_k, $(\neg s_{ki_1}^a \vee \neg s_{ki_2}^a) \wedge (\neg s_{ki_1}^a \vee \neg s_{ki_2}^b) \wedge (\neg s_{ki_1}^b \vee \neg s_{ki_2}^a) \wedge (\neg s_{ki_1}^b \vee \neg s_{ki_2}^b)$.

One of the key techniques proposed in the basic model is the utilization of the sorting condition over the haplotypes, as an effective symmetry breaking technique. Additional symmetry breaking conditions are possible. Observe that the model consists of selecting a candidate haplotype for the a representative and another haplotype for the b representative, such that each genotype is explained by the a and b representatives. Given a set of r candidate haplotypes, let h_{k_1} and h_{k_2}, with $k_1, k_2 \leq r$, be two haplotypes which explain a genotype g_i. This means that g_i can be explained by the assignments $s_{k_1 i}^a s_{k_2 i}^a s_{k_1 i}^b s_{k_2 i}^b = 1001$, but also by the assignments $s_{k_1 i}^a s_{k_2 i}^a s_{k_1 i}^b s_{k_2 i}^b = 0110$. This symmetry can be eliminated by requiring that only one arrangement of the s variables can be used to explain each genotype g_i. One solution is to require that the haplotype selected by the s_{ki}^a variables always has an index *smaller* than the haplotype selected by the s_{ki}^b variables. This requirement is captured by the conditions $\left(s_{k_1 i}^a \rightarrow \bigwedge_{k_2=1}^{k_1-1} \neg s_{k_2 i}^b\right)$

[1] See for example [4] for a survey of work on the utilization of lexicographic orderings for symmetry breaking.

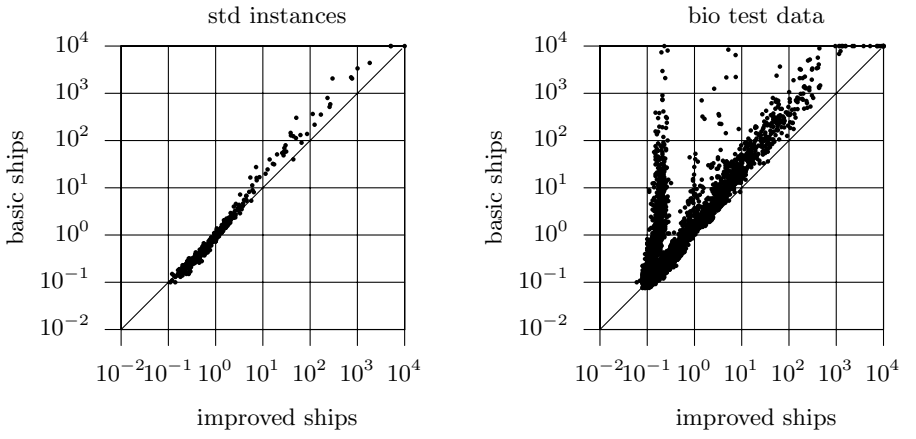

Fig. 1. Comparison of the basic and improved SHIPs models (run times)

and $\left(s^{b}_{k_2 i} \rightarrow \bigwedge_{k_1=k_2+1}^{r} \neg s^{a}_{k_1 i}\right)$. Clearly, each condition above can be represent by a single clause. Moreover, observe that for genotypes with heterozygous sites, the upper limit of the first constraint can be set to $k_1 - 1$ and the lower limit of the second condition can be set to $k_2 + 1$.

5 Experimental Results

The models described in the previous section, referred to as SHIPs (Sat-based Haplotype Inference by Pure Parsimony), have been implemented as a Perl script, which iteratively generates CNF formulas to be given to a SAT solver. Currently, MiniSAT [3] is used.

With the purpose of comparing the basic and the improved versions of SHIPs, two sets of problem instances are considered. The first set of instances were generated using Hudson's program `ms` [6] (denoted `std instances`). The second set of instances are the instances currently available from biological test data (denoted `bio test data`) (e.g. from [9]).

The results are shown in Figure 1. Each plot compares the CPU time required by both the basic and the improved SHIPs models for solving each problem instance. The limit CPU time was set to 10000s using a 1.9 GHz AMD Athlon XP with 1GB of RAM running RedHat Linux. For the `std` instances the results are clear. The improved model is consistently faster than the basic model, especially for the most difficult problem instances. For problem instances requiring more than 10 CPU seconds, and with a single exception, the improved model is always faster than the basic model. For most of these instances, and by noting that the plot uses a log scale, we can conclude that the speedups range from a factor of 2 to a factor of 10. For the `bio test data` instances the performance differences become even more clear. The improved model significantly outperforms the basic

model. Observe that the speedups introduced by the improved model can exceed 4 orders of magnitude.

6 Conclusions and Future Work

This paper provides further evidence that haplotype inference is a new very promising application area for SAT. The results in this paper and in [8] provide unquestionable evidence that the utilization of SAT yields the most efficient approach to the problem of haplotype inference by pure parsimony. Indeed, the SAT-based approach is the *only* approach currently capable of solving a large number of practical instances. Moreover, the optimizations proposed in this paper are shown to be essential for solving challenging problem instances from biological test data. Despite the promising results, several challenges remain. Additional biological test data may yield new challenging problem instances, which may motivate additional optimizations to the SAT-based approach.

Acknowledgments. The authors thank Arlindo Oliveira for having pointed out the haplotype inference by pure parsimony problem. This work is partially supported by FCT under research project POSC/EIA/61852/2004.

References

1. D. Brown and I. Harrower. A new integer programming formulation for the pure parsimony problem in haplotype analysis. In *Workshop on Algorithms in Bioinformatics (WABI'04)*, 2004.
2. D. Brown and I. Harrower. Integer programming approaches to haplotype inference by pure parsimony. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(2):141–154, April-June 2006.
3. N. Eén and N. Sörensson. An extensible SAT-solver. In *International Conference on Theory and Applications of Satisfiability Testing (SAT)*, pages 502–518, 2003.
4. A. Frisch, B. Hnich, Z. Kiziltan, I. Miguel, and T. Walsh. Global constraints for lexicographic orderings. In *International Conference on Principles and Practice of Constraint Programming (CP)*, 2002.
5. D. Gusfield. Haplotype inference by pure parsimony. In *14th Annual Symposium on Combinatorial Pattern Matching (CPM'03)*, pages 144–155, 2003.
6. R. R. Hudson. Generating samples under a wright-fisher neutral model of genetic variation. *Bioinformatics*, 18(2):337–338, February 2002.
7. G. Lancia, C. M. Pinotti, and R. Rizzi. Haplotyping populations by pure parsimony: complexity of exact and approximation algorithms. *INFORMS Journal on Computing*, 16(4):348–359, 2004.
8. I. Lynce and J. Marques-Silva. Efficient haplotype inference with Boolean satisfiability. In *National Conference on Artificial Intelligence (AAAI)*, July 2006.
9. M. J. Rieder, S. T. Taylor, A. G. Clark, and D. A. Nickerson. Sequence variation in the human angiotensin converting enzyme. *Nature Genetics*, 22:481–494, 2001.
10. L. Wang and Y. Xu. Haplotype inference by maximum parsimony. *Bioinformatics*, 19(14):1773–1780, 2003.