Efficient Haplotype Inference with Answer Set Programming

Ferhan Türe and **Esra Erdem**
Faculty of Engineering and Natural Sciences
Sabancı University, Orhanlı, Tuzla, Istanbul 34956, TURKEY

Abstract

Identifying maternal and paternal inheritance is essential to be able to find the set of genes responsible for a particular disease. Although we have access to genotype data (genetic makeup of an individual), determining haplotypes (genetic makeup of the parents) experimentally is a costly and time consuming procedure due to technological limitations. With these biological motivations, we study a computational problem, called Haplotype Inference with Pure Parsimony (HIPP), that asks for the minimal number of haplotypes that form a given set of genotypes. We introduce a novel approach to solving HIPP, using Answer Set Programming (ASP). According to our experiments with a large number of problem instances (some automatically generated and some real), our ASP-based approach solves the most number of problems compared to other approaches based on, e.g., integer linear programming, branch and bound algorithms, SAT-based algorithms, or pseudo-boolean optimization methods.

Introduction

A genotype is the specific genetic makeup of an individual. Each genotype has two copies, one from the mother and one from the father. These two copies are called haplotypes, and they combine to form the genotype. Haplotype Inference with Pure Parsimony (HIPP) problem is, for a given set of genotypes, to infer the haplotypes that explain them. This is essential to be able to map disease genes, and find the set of genes responsible for a particular disease. Due to technological limitations, we have access only to genotype data rather than haplotype data, and determining haplotypes experimentally is a costly and time consuming procedure. Therefore, inferring haplotypes computationally is a highly motivated problem in terms of biological applications.

There have been various approaches to solving the HIPP problem. A SAT-based approach was introduced in (Lynce & Marques-Silva 2006) and a PBO-based approach in (da Graça *et al.* 2007). Other approaches include ILP-based methods (Brown & Harrower 2004; Gusfield 2003), and a branch and bound algorithm (Wang & Xu 2003).

In this paper we introduce a novel approach to this problem, using Answer Set Programming (ASP), which we call Haplo-ASP. Experiments indicate that Haplo-ASP solves more number of problems than any of the existing systems does. Also, unlike the existing approaches, our approach allows us to add domain specific information such as haplotype patterns observed for some gene family.

Haplotype Inference

Given a set G of n genotypes each with m sites, the HIPP problem is to find a minimal set H of haplotypes such that each genotype in G is *explained* by two haplotypes in H. Let each site in a genotype take a value from $\{0, 1, 2\}$; and each site in a haplotype take a value from $\{0, 1\}$. Then, two haplotypes h_1 and h_2 *explain* a genotype g if for every site j the following hold (Gusfield 2003):

- if $g[j] = 2$ then either $h_1[j] = 0$ and $h_2[j] = 1$ or $h_2[j] = 0$ and $h_1[j] = 1$;
- if $g[j] = 1$ then $h_1[j] = 1$ and $h_2[j] = 1$; and
- if $g[j] = 0$ then $h_1[j] = 0$ and $h_2[j] = 0$.

We consider the decision version of HIPP, called HIPP-DEC: Given a set G of n genotypes each with m sites, and a positive integer k, decide whether there is a set H of at most k unique haplotypes such that each genotype in G is explained by two haplotypes in H. HIPP-DEC is proved to be NP-hard (Gusfield 2003; Lancia, Pinotti, & Rizzi 2004).

Assume H is a set that contains $2 * n$ haplotypes, $h_1, \ldots, h_{2n}$, and that every genotype g_i in G is mapped to two haplotypes, h_{2i} and h_{2i-1}, in H. Then H is a solution to HIPP-DEC if the following hold:

C1 For every genotype g in G, for every site j of g with value 2, the values of the j'th sites of the corresponding two haplotypes are different.

C2 For every genotype g in G, for every site j of g with value 1 or 0, the values of the j'th sites of the corresponding two haplotypes are $g[j]$.

C3 There are at most k unique haplotypes in H.

In ASP, we consider genotypes and haplotypes as sets of literals. The existence of $amb(i, j)$ (resp. $\neg amb(i, j)$) in an answer set represents $g_i[j] = 2$ (resp. $g_i[j] = 1$), and their absence represents $g_i[j] = 0$. Similarly, haplotypes are described by atoms of the form $h(i, j)$. The ASP program describing HIPP-DEC is presented in the language of the ASP solver CMODELS (Lierler & Maratea 2004). Some parts of

```
% Generate a value for each site of haplotype H
{h(H,J)} :- haplo(H).

% Test the generated haplotypes wrt C1--C3
% C1
:- amb(G,J), h(2*G,J), h(2*G-1,J).
:- amb(G,J), not h(2*G-1,J), not h(2*G,J).

% C2
:- not h(2*G-1,J),-amb(G,J).
:- not h(2*G,J),-amb(G,J).
:- h(2*G-1,J), not -amb(G,J), not amb(G,J).
:- h(2*G,J), not -amb(G,J), not amb(G,J).

% C3
:- k+1 {unique(H):haplo(H)}.
```

Figure 1: Description of HIPP in LPARSE.

the ASP program is shown in Fig. 1; here `G` is a genotype, `J` is a site, and `unique` describes unique haplotypes. The program is available at (Haplo-ASP 2008).

We can easily add to the ASP program some domain specific information, e.g., some haplotype patterns observed for the gene family. For instance, we can express that all haplotypes have value 1 at site 2 by the constraint:

```
:- not h(H,2), haplo(H).
```

Experiments

We compared the performance of haplotype inference systems RPoly (da Graça *et al.* 2007), SHIPs (Lynce & Marques-Silva 2006), and Haplo-ASP on a workstation with 1.5GHz Xeon processor and 4x512MB RAM, running Red Hat Linux(Version 4.3). Problem instances were all obtained from the authors of (Brown & Harrower 2004), some of which are real biological problems and some are artificially generated (Hudson 2002). All problem instances were simplified as described in (Lynce & Marques-Silva 2006). A lower bound for k can be computed as explained in (Lynce & Marques-Silva 2006). We used the lower bounds computed by SHIPs, then performed a binary search on the values of k to find the minimal set of unique haplotypes explaining the given set of genotypes.

We conducted experiments as in (da Graça *et al.* 2007). For each system we assigned 1000 sec.s of CPU time to solve each problem. We then sorted all problem instances according to CPU times, and plotted a graph of problems and the corresponding CPU time values. The graph is available at (Haplo-ASP 2008).

Table 1: Experimental results

Category	# of problems	SHIPs	Haplo-ASP	Rpoly
Real	294	282	292	291
Artificial	40	40	40	40
Total	334	322	332	331

Table 1 shows the results comparing all haplotype inference systems for 334 instances (40 automatically generated, 294 real). The best performance in terms of total number of problems solved is by Haplo-ASP. Rpoly aborts for only one problem that Haplo-ASP solves, but in most of the problems for which it has a solution, it is faster than Haplo-ASP by a magnitude of up to 300. SHIPs aborts on 12 problems out of 334, but can solve a problem that neither Haplo-ASP nor Rpoly can solve in 1000 sec.s. We also tried Hapar (Wang & Xu 2003), and obtained results similar to (Lynce & Marques-Silva 2006). For instance, Hapar can only solve 12 out of 90 for a specific set of problems (ibd).

Conclusion and Future Work

In this paper we presented Haplo-ASP, an ASP-based HIPP solver that describes the HIPP problem in a formal representation and uses an underlying engine to find a solution to the problem. Unlike the existing approaches, Haplo-ASP allows us to add domain specific information. Haplo-ASP solves more problems compared to the state-of-the-art solvers, and uses less computation time in large problem instances. However, Rpoly is faster than Haplo-ASP on many problem instances, so several improvements could be done to increase performance. Some improvements to Haplo-ASP would be to add more symmetry-breaking constraints, and to decrease program size by some preprocessing step, as implemented in the existing systems; this is a part of our ongoing work.

Acknowledgments

Pierre Flener, Serdar Kadıoğlu, and Uğur Sezerman shared with us their experiences with the existing haplotype inference systems. Inês Lynce and Joao Marques-Silva provided us with the executables for SHIPs. Daniel Brown, Ian Harrower, and Inês Lynce provided us with the benchmarks.

References

Brown, D., and Harrower, I. 2004. A new integer programming formulation for the pure parsimony problem in haplotype analysis. In *Proc. of WABI*, 254–265.

da Graça, A.; Marques-Silva, J.; Lynce, I.; and Oliveira, A. 2007. Efficient haplotype inference with pseudo-boolean optimization. In *Proc. of AB*:125–139.

Gusfield, D. 2003. Haplotype inference by pure parsimony. In *Proc. of CPM*, 144–155.

Haplo-ASP. 2008. `http://people.sabanciuniv.edu/esraerdem/haplo-asp.html`.

Hudson, R. 2002. Generating samples under a wrightfisher neutral model of genetic variation. *Bioinf.* 18:337–338.

Lancia, G.; Pinotti, M. C.; and Rizzi, R. 2004. Haplotyping populations by pure parsimony: Complexity of exact and approximation algorithms. *INFORMS Journal on Computing* 16(4):348–359.

Lierler, Y., and Maratea, M. 2004. Cmodels-2: SAT-based answer set solver enhanced to non-tight programs. In *Proc. of LPNMR-7*, 346–350.

Lynce, I., and Marques-Silva, J. 2006. Efficient haplotype inference with boolean satisfiability. In *Proc. of AAAI*, 239–250.

Wang, L., and Xu, Y. 2003. Haplotype inference by maximum parsimony. *Bioinformatics* 19:1773–1780.