Efficient Genome Wide Tagging by Reduction to SAT

Arthur Choi[1], Noah Zaitlen[2], Buhm Han[3], Knot Pipatsrisawat[1], Adnan Darwiche[1], and Eleazar Eskin[1,4,⋆]

[1] Department of Computer Science, University of California Los Angeles, Los Angeles, CA, 90095
[2] Bioinformatics Program, University of California, San Diego, La Jolla, CA, 92093
[3] Department of Computer Science and Engineering, University of California, San Diego, La Jolla, CA, 92093
[4] Department of Human Genetics, University of California Los Angeles, Los Angeles, CA, 90095
Tel.: +1-310-594-5112 Fax: +1-310-825-2273
eeskin@cs.ucla.edu

Abstract. Whole genome association has recently demonstrated some remarkable successes in identifying loci involved in disease. Designing these studies involves selecting a subset of known single nucleotide polymorphisms (SNPs) or tag SNPs to be genotyped. The problem of choosing tag SNPs is an active area of research and is usually formulated such that the goal is to select the fewest number of tag SNPs which "cover" the remaining SNPs where "cover" is defined by some statistical criterion. Since the standard formulation of the tag SNP selection problem is NP-hard, most algorithms for selecting tag SNPs are either heuristics which do not guarantee selection of the minimal set of tag SNPs or are exhaustive algorithms which are computationally impractical. In this paper, we present a set of methods which guarantee discovering the minimal set of tag SNPs, yet in practice are much faster than traditional exhaustive algorithms. We demonstrate that our methods can be applied to discover minimal tag sets for the entire human genome. Our method converts the instance of the tag SNP selection problem to an instance of the satisfiability problem, encoding the instance into conjunctive normal form (CNF). We take advantage of the local structure inherent in human variation, as well as progress in knowledge compilation, and convert our CNF encoding into a representation known as DNNF, from which solutions to our original problem can be easily enumerated. We demonstrate our methods by constructing the optimal tag set for the whole genome and show that we significantly outperform previous exhaustive search-based methods. We also present optimal solutions for the problem of selecting multi-marker tags in which some SNPs are "covered" by a pair of tag SNPs. Multi-marker tags can significantly decrease the number of tags we need to select, however discovering the minimal number of

⋆ Corresponding author.

K.A. Crandall and J. Lagergren (Eds.): WABI 2008, LNBI 5251, pp. 135–147, 2008.

multi-marker tags is much more difficult. We evaluate our methods and perform benchmark comparisons to other methods by choosing tag sets using the HapMap data.

1 Introduction

Whole genome association is a powerful method for discovering the genetic basis of human diseases. Recently, it has been successfully employed to reveal novel loci correlated with risks for diseases including coronary artery disease, bipolar disorder, type 1 and type 2 diabetes, amongst many others[11]. A typical association study collects genotype information at a set of single nucleotide polymorphism (SNP) and compares the allele frequency at each SNP in a case and a control population using a statistical test in order to determine which loci are associated with the disease.

Even with the tremendous technological advances that have driven down the cost of collecting SNP genotypes, collecting all known SNPs is prohibitively expensive. Genetic association studies take advantage of the fact that genotypes at neighboring SNPs are often in linkage disequilibrium (LD) or are correlated with each other. This correlation allows for "indirect association" where a SNP which is associated with the disease is detected not by collecting genotypes at the SNP directly, but instead by collecting genotypes at a neighboring "tag" SNP or SNP that is correlated with the associated SNP. The availability of reference data sets such as those provided by the HapMap project[10] allow for us to measure the linkage disequilibrium patterns between SNPs and use this information when determining which SNPs to select as tags. *Tag SNP selection* is a central problem in designing association studies and has been extensively studied [27].

Research on the tag SNP selection problem can be roughly split into two categories: the statistical criteria used for selecting tag SNPs and the algorithms for choosing a tag set given this statistical criteria. Many statistical criteria have been proposed for tag SNP selection [27]. The most popular criterion considers the square correlation coefficient r^2 between SNPs. Under this formulation of the tag SNP selection problem the goal is to choose a subset of the SNPs as tags such that each SNP not selected in the tag set has an r^2 value with a tag SNP above a minimal threshold. This problem is commonly represented as a graph where each SNP is represented as a node and an edge connects two nodes if their corresponding SNPs have a correlation above the threshold. A set of tags which covers the remaining tag SNPs corresponds to a vertex cover in this graph. This problem is NP-complete [1] and it was widely believed that we cannot obtain the minimal solutions to this problem for the whole genome[27]. Relatively few algorithmic approaches have been proposed for this problem with the greedy algorithm being the most widely used[7]. While the greedy approach has been used to construct the current generation of commercial products used for whole genome association studies, the greedy algorithm provides no guarantees that the chosen tag set has the minimal number of tag SNPs necessary to cover the region. Relatively few approaches have been presented to obtaining minimal tag sets

and these approaches reduce the problem in order to make it computationally feasible. Halldorson et al. [5] restrict LD patterns to a window and FESTA [24] solves the problem by partitioning the SNPs into precincts which do not have any linked SNPs in them and then exhaustively enumerating the solutions within the precinct.

In this paper we present a novel approach to solving the tag SNP selection problem which can discover all minimal solutions and can scale to the whole genome. Our method encodes the instance of the tag SNP selection problem as an instance of the satisfiability (SAT) problem. Here, our SAT instances are clauses in conjunctive normal form (CNF) where a variable assigned to `true` corresponds to the inclusion of a SNP into the tag set. We take advantage of the local structure inherent in human variation, as well as progress in knowledge compilation, and convert our CNF encoding into a representation known as DNNF, from which solutions to our original problem can be easily enumerated. A satisfying assignment of variables to truth values in the SAT instance yields a valid solution to the tagging problem (and vice versa). As we shall clarify, a "minimal" satisfying assignment yields a minimal set of tags.

We compare the results of our method on the single SNP r^2 tag SNP selection problem to FESTA [24] and Halldorson's method [5] over the ENCODE [10] data set. We also demonstrate that our methods scales to the whole genome HapMap data. Consistent with previous studies, minimal solutions for the tag SNP selection method use only slightly less tags than greedy solutions[24]. One advantage of our framework is that we can characterize the entire set of optimal solutions, but in a tractable form that allows for flexible designs. Another advantage is that our approach extends to more challenging variants of the tag SNP selection problem where we are selecting mutli-marker tags in which a SNP can be "covered" by a pair of tags. Multi-marker tags have been shown to significantly increase the power of association studies[23,30].

2 Methods

We present methods for choosing the minimal number of tags SNP for several variations of the *tagging* problem. First we show how to solve the single SNP r^2 tagging problem in which we search for a minimum set of tag SNPs which cover the remaining SNPs in a region of the genome with an r^2 above some minimum threshold. Second, we present a method for combining our optimal solutions in local regions to an optimal solution for the entire genome. Third, we extend our solution to multi-marker tags or tags which combine two or more SNPs. The use of multi-marker tags can significantly reduce the number of tags which need to be collected in order to cover a region, but the optimization procedure is much more difficult.

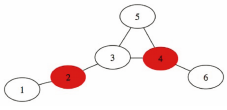
2.1 Single SNP r^2 Tagging

Let $S = \{s_i\}_{i=1}^n$ be a set of SNPs. We say SNP s_i "covers" SNP s_j if their correlation coefficient r^2, exceeds some threshold r^2_{min}. If $T' \subseteq S$, and $\forall s_j \in$

(a)

r^2_{ij}	s_1	s_2	s_3	s_4	s_5	s_6
s_1	1.0	0.9	0.5	0.4	0.2	0.4
s_2	0.9	1.0	0.9	0.5	0.3	0.2
s_3	0.5	0.9	1.0	0.9	0.8	0.1
s_4	0.4	0.5	0.9	1.0	0.9	0.8
s_5	0.2	0.3	0.8	0.9	1.0	0.5
s_6	0.4	0.2	0.1	0.8	0.5	1.0

(b)



(c)

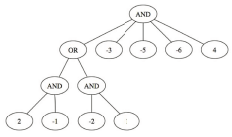


Fig. 1. (a) Single SNP r^2 table (b) Graph of cover problem (c) NNF equivalent to CNF

$S, \exists s_i \in T'$ such that $r^2_{ij} \geq r^2_{min}$, we call T' a *valid* cover of S. Our goal is to select the smallest set $T' \subseteq S$ that is a *valid* cover of S.

Consider the example in Figure 1, where we have 6 SNPs $s_1, \ldots, s_6$, and the pairwise r^2 values described in the table in Figure 1(a). Suppose that we have the threshold $r^2_{min} = 0.8$. We can represent the SNPs as the graph shown in Figure 1(b) where an edge denotes an r^2 above the minimum threshold. The standard greedy algorithm [7,18] picks tag SNPs by repeatedly selecting the SNP with the largest number of uncovered neighbors. We can easily see that there are two optimal solutions, $T = \{s_4, s_2\}$ and $T = \{s_4, s_1\}$. Note that one greedy solution will select SNP s_3 in the first step resulting in a non-optimal solution $T = \{s_3, s_1, s_6\}$. Our approach to the tag SNP selection problem will characterize all optimal solutions in a compact directed acyclic graph, which in this case happens to be a tree in the example of Figure 1(c).

We shall reduce the problem of identifying a valid selection of SNPs to the problem of identifying a satisfying assignment to a propositional sentence in conjunctive normal form (CNF). In particular, we want a sentence in CNF where satisfying assignments correspond to a valid selection of SNPs. We create a literal for every SNP and a clause for every SNP consisting of literals that can cover that SNP.

Given a threshold r^2_{min}, consider a sentence in CNF: $\Phi = \phi_1 \wedge \cdots \wedge \phi_n$ with as many clauses ϕ_i as there are SNPs s_i, where each clause is of the form:

$$\phi_i = \bigvee_{r^2_{ij} \geq r^2_{min}} s_j$$

Each SNP $s_j \in S$ is a positive literal in the CNF sentence Φ, and appears in clause ϕ_i if and only if SNP s_j can cover SNP s_i. A valid selection T' of SNPs then corresponds precisely to a satisfying assignment of Φ.

In order to find a *minimally* valid selection T of SNPs, we seek a minimum cardinality model of our propositional sentence, where a minimum cardinality model is a satisfying assignment with a minimal number of positive literals.

Consider the example in Figure 1 with six SNPs $s_1, \ldots, s_6$. Given the threshold $r^2_{min} = 0.8$ we have the following CNF formula $(s_1 \vee s_2) \wedge (s_1 \vee s_2 \vee s_3) \wedge (s_2 \vee s_3 \vee s_4 \vee s_5) \wedge (s_3 \vee s_4 \vee s_5 \vee s_6) \wedge (s_3 \vee s_4 \vee s_5) \wedge (s_4 \vee s_6)$ We have two minimum cardinality models, $(\neg s_1, s_2, \neg s_3, s_4, \neg s_5, \neg s_6)$ and $(s_1, \neg s_2, \neg s_3, s_4, \neg s_5, \neg s_6)$, corresponding to our two minimally valid selection of SNPs.

Not surprisingly, identifying a minimum cardinality model for a given sentence in CNF is also an NP–hard problem. Our approach is based on converting our sentence Φ in CNF into a logically equivalent sentence Δ in *decomposable negation normal form* (DNNF) [12,13,15,16]. DNNF is a logical representation that allows certain queries, which are in general intractable, to be computed in time polynomial in the size of the DNNF sentence. For example, if a conversion from CNF to DNNF does indeed result in a sentence of manageable size, we can efficiently test whether the original sentence is satisfiable, count and enumerate its models, and identify another sentence in DNNF that characterizes all its minimum cardinality models. By enumerating the models of the resulting sentence, we can enumerate all of the minimally valid selections of SNPs. In general, there are no guarantees that a CNF can be converted to a DNNF of reasonable size, but we demonstrate that for the tag SNP selection problem, due to the inherent local structure of the problem, our approach is tractable.

This conversion is performed here by the C2D compiler, which compiles CNF instances into DNNF [6].[1] The C2D compiler has already been successfully employed in a number of other applications, serving as a backbone reasoning system in support of higher level tasks. For example, C2D was used as the backbone for planning systems [4,20], for diagnostic systems [19,3,22,2], for probabilistic reasoning [28,9,25,8], and for query rewrites in databases [29]. In each one of these applications, high level reasoning problems were encoded into CNF, which was compiled into DNNF by C2D. The resulting compilation was then used to solve the original problem by posing polytime queries to it.

Decomposable negation normal form. A negation normal form (NNF) is a rooted and directed acyclic graph in which each leaf node is labeled either by a literal (say i for a positive literal, and $-i$ for a negative literal), or simply by true or false. Each internal node is labeled with a conjunction ($\wedge$ or AND) or a disjunction ($\vee$ or OR); Figure 1(c) depicts an example. A negation normal form is decomposable (DNNF) if it satisfies the *Decomposability* property: for each conjunction in the NNF, the conjuncts do not share variables. The NNF in Figure 1(c) is also in DNNF.

If we are able to efficiently compile a CNF instance into DNNF, many queries are straightforward to compute [12]. For example, we can test if a DNNF sentence is satisfiable by simply traversing the graph bottom-up, while visiting children before parents. If a leaf node is labeled by a literal (i or $-i$), or true, then it is satisfiable; otherwise, it is unsatisfiable (labeled with false). A subsentence rooted at an OR node is satisfiable iff any of its children are satisfiable. For an AND node, it is satisfiable iff all of its children are satisfiable (since AND nodes are decomposable). We can compute the minimum cardinality of a sentence and enumerate its models in a similar way [12,17].

Before we proceed to describe how to compile a sentence in CNF into DNNF, consider the *conditioning* of a sentence Δ on an instantiation α, denoted $\Delta \mid \alpha$. This operation yields a sentence that can be obtained by replacing every literal in

[1] C2D further enforces the *determinism* property, and more specifically, compiles CNF instances into d-DNNF [16].

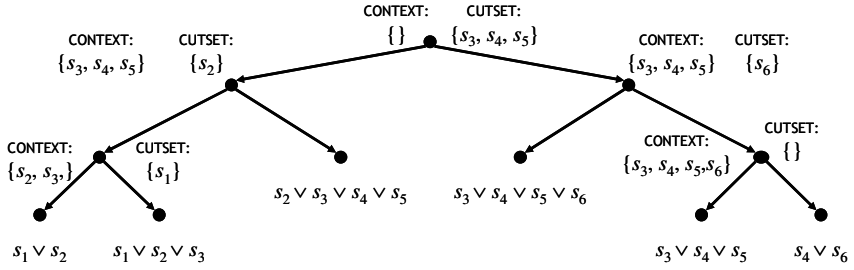


Fig. 2. A d-tree for the CNF used for the example in Figure 1. Internal nodes are labeled with their contexts and cutsets.

Δ with true (respectively, false) if it is consistent (inconsistent) with instantiation α. For example, conditioning the DNNF $(\neg a \wedge \neg b) \vee (b \wedge c)$ on instantiation $b \wedge d$ gives $(\neg a \wedge \mathsf{false}) \vee (\mathsf{true} \wedge c)$. Note that DNNF is closed under conditioning, i.e., conditioning a DNNF Δ on an instantiation α results in another DNNF. Moreover, the resulting sentence $\Delta \mid \alpha$ does not mention variables assigned by α.

Consider the following theorem, due to [12], which motivates the compilation procedure underlying C2D.

Theorem 1 (Case Analysis). *Let Δ_1 and Δ_2 be two sentences in* DNNF*, and let Δ be the sentence $\bigvee_\alpha (\Delta_1 \mid \alpha) \wedge (\Delta_2 \mid \alpha) \wedge \alpha$, where α are instantiations of variables mentioned in both Δ_1 and Δ_2. Then Δ is in* DNNF*, and is equivalent to $\Delta_1 \wedge \Delta_2$.*

This theorem suggests a recursive algorithm $\mathsf{DNNF1}(\Phi)$ that converts a sentence Φ in CNF into a sentence Δ in DNNF:

1. If Φ contains a single clause ϕ, return $\mathsf{DNNF1}(\Phi) \leftarrow \phi$. Note that a clause is vacuously decomposable.
2. Otherwise, return

$$\mathsf{DNNF1}(\Phi) \leftarrow \bigvee_\alpha \mathsf{DNNF1}(\Phi_1 \mid \alpha) \wedge \mathsf{DNNF1}(\Phi_2 \mid \alpha) \wedge \alpha,$$

where Φ_1 and Φ_2 is a partitioning of clauses in Φ, and α is an instantiation of the variables mentioned in both Φ_1 and Φ_2.

We can see that this procedure gives us the decomposability property, but at the expense of increasing the size of the original sentence. This increase is incurred primarily due to the case analysis performed, and the extent of this increase is sensitive to the way we decide to partition the clauses of the input sentence Φ. In particular, we would want to minimize the number of common variables between Φ_1 and Φ_2, as the complexity of case analysis is exponential in this number.

Partitioning can be guided by decomposition trees, or simply d-trees [12].

Definition 1. *A d-tree $\mathcal{T}$ for a CNF Φ is a binary tree whose leaves correspond to the clauses in Φ.*

An example d-tree for the CNF used for the example in Figure 1 is shown in Figure 2. Intuitively, the above compilation procedure traverses the d-tree, starting from the root, where case analysis is performed based on how the d-tree partitions the clauses of the CNF Φ. In particular, each interior node t is associated with the set of clauses that appear below it, and the partition is determined by the clauses of t's left and right children.

As we can see in Figure 2, each internal node is labeled with two variable sets: the *cutset* and the *context*. At a given node t, the cutset tells our compilation algorithm which variables to perform case analysis on. The context tells us those variables that appear in both of t's children, but have already been instantiated for case analysis by an ancestor. An instantiation α of the context variables can then be used as a key for a cache that stores the results of compiling the subset of clauses $\Phi \mid \alpha$. When a node is revisited with the same context, then the algorithm can simply return the DNNF sentence $\Delta \mid \alpha$ already computed. For example, at the root of the tree, the cutset contains $\{s_3, s_4, s_5\}$ since those variables appear in both children. If we follow the left branch twice, the context is now $\{s_2, s_3\}$, which was instantiated by the root and its left children. Note that this node will be visited multiple times for different instantiations of $\{s_4, s_5\}$, but only different instantiations of the context yield different subproblems. Thus, when this node is revisited with the same context instantiation, we simply fetch the result from the cache. It is this subformula re-use that allows compilation to moderate the exponential growth of the formula caused by case analysis.[2] The C2D compiler, while based on this approach, employs more advanced techniques to further improve on the efficiency of compilation [14].

Scaling to Whole Genome Tagging. The C2D compiler is capable of computing minimal tag sets, for several thousands of SNPs. Unfortunately, memory becomes an issue when we try to compile even larger regions of the genome. To encode the entire genome as a CNF, however, we must use 3.8 million literals and 3.8 million clauses. Clearly, we need new techniques to scale to this problem size. Roughly, our approach is to compile sufficiently small regions of the genome into DNNF, which are then "stitched" together to construct a DNNF for the entire genome. Due to space limitations, we omit the detail of this approach.

2.2 Multi-marker SNP Tagging

Recent work has shown that using statistical tests based on haplotypes over multiple SNPs improves the power of whole genome association studies[23,30]. In the context of tagging, this permits combinations of tag SNPs (multi-marker tags) to cover a SNP, allowing for a smaller set of tags to cover the SNPs.

In this situation, an even smaller set T' may be a valid cover of SNPs.

Again, we reduce the problem of identifying a valid set of SNPs to satisfiability. Given a threshold r^2_{min}, we now have two classes of clauses Φ and Ψ. Clauses Φ, as before, enforce constraints that require each SNP in S to be covered:

[2] In particular, the complexity of compilation can be bounded in terms of the size of the context and the cutset [12].

$\Phi = \phi_1 \wedge \cdots \wedge \phi_n$, where there are as many clauses ϕ_i as there are SNPs s_i, but where each clause is now of the form:

$$\phi_i = \Big(\bigvee_{r^2_{j \to i} \geq r^2_{min}} s_j \Big) \vee \Big(\bigvee_{r^2_{j,k \to i} \geq r^2_{min}} p_{jk} \Big).$$

In this case, either a positive literal s_j representing SNP s_j or a positive literal p_{jk} representing a SNP pair (s_j, s_k) can also satisfy clause ϕ_i and cover SNP s_i.

Clauses Ψ enforce the constraints that if a pair literal p_{jk} is true, then both s_j and s_k are true (i.e., in the tag set):

$$\Psi = \bigwedge_{r^2_{j,k \to i} \geq r^2_{min}} (p_{jk} \equiv s_j \wedge s_k)$$

where $p_{jk} \equiv s_j \wedge s_k$ are equivalence constraints that ensure that p_{jk} is selected iff the corresponding pair (s_j, s_k) is selected. In clausal form, this equivalence constraint is given by three clauses: $\neg p_{jk} \vee s_j$, $\neg p_{jk} \vee s_k$ and $p_{jk} \vee \neg s_j \vee \neg s_k$.

Consider the example from the previous section with six SNPs $s_1, \ldots, s_6$. Suppose that the pair (s_1, s_3) can cover s_6, and that $r^2_{1,3 \to 6} = 0.9$. Given the threshold $r^2_{min} = 0.8$, we gain a third optimal solution s_1, s_3, to go with the two solutions s_4, s_2 and s_4, s_1 from before. Encoding the problem, we have the following formula: $(s_1 \vee s_2) \wedge (s_1 \vee s_2 \vee s_3) \wedge (s_2 \vee s_3 \vee s_4 \vee s_5) \wedge (s_3 \vee s_4 \vee s_5 \vee s_6) \wedge (s_3 \vee s_4 \vee s_5) \wedge (s_4 \vee s_6 \vee p_{13}) \wedge (p_{13} \equiv s_1 \wedge s_3)$. We again want a minimum cardinality assignment, but minimizing only the number of positive s_i literals. We can introduce constraints $p_{jk} \equiv \neg q_{jk}$ to cancel out the contribution of the p_{jk}'s to the cardinality with the q_{jk}'s; we can then convert to DNNF and minimize, as before.[3]

3 Results

We downloaded the complete HapMap build 22 data including all ENCODE regions. These data are genotypes on 270 individuals in 4 populations and over 3.8 million SNPs. They represent the most complete survey of genotype data currently available and are used as our test data sets. The 10 ENCODE regions span 5 MB and are believed to have complete ascertainment for SNPs with frequency greater than 5%. They are commonly used to estimate the performance of association study design methods and tag SNP selection methods since there are still many unknown common SNPs in the rest of the genome.

3.1 ENCODE Single SNP r^2 Comparison

We compared the performance of our method to the two other optimal methods as well as the non-optimal greedy algorithm over each of the ENCODE regions

[3] We can also existentially quantify out the p_{jk} literals; this operation is supported by C2D [6].

Region	n_s	Greedy	Halldorson(W=15)	FESTA(L=10^7)	Optimal
ENm010	567	159	243(12m)	157(0m)	157
ENm013	755	93	309(23m)	90(0m)	90
ENm014	914	164	393(33m)	157(0m)	157
ENr112	927	180	340(34m)	173(34m)	173
ENr113	1072	179	395(46m)	176(274m)	176
ENr123	937	174	463(35m)	172(0m)	172
ENr131	1041	228	414(44m)	221(0m)	221
ENr213	659	122	248(17m)	122(0m)	122
ENr232	533	142	181(11m)	140(36m)	140
ENr321	599	132	202(14m)	131(0m)	131

Fig. 3. Comparison of several *tagging* algorithms over the Encode regions in the CEU population. n_s is the number of SNPs in the region. The table shows the tag set size for each of the methods various methods (smaller is better). Running times are given in parentheses in minutes. All running times for Optimal are less than one minute (0m).

in each of the populations. Haldorsson *et al.*[5] restricts the maximum length of correlations and uses a dynamic programming procedure which guarantees to find an optimal solution. Given a window size W, Haldorsson *et al.*[5] examines all 2^W possible choices of tag SNPs in the window and then uses dynamic programming to extend this to a longer region. FESTA [24] extends the standard greedy algorithm in a natural way. Given an r^2 threshold, FESTA partitions the SNPs into precincts where SNPs are correlated only within the precinct. For a small precinct FESTA enumerates all possible tag sets in search of the minimal tag set. For a larger precinct, FESTA applies a hybrid exhaustive enumeration and greedy algorithm by first selecting some SNPs using exhaustive enumeration and then applying greedy algorithm. The user defines a threshold L so that the hybrid method is applied when $\binom{n}{k} > L$ where n is the number of SNPs in a precinct and k is the number of tags that need to be selected from the precinct. We use $L = 10^7$ in our experiments.

The results are presented in table 3. Surprisingly, the Halldorson *et al.*[5] method, at the maximum limit of what is computationally feasible ($W = 15$) performs worse than the simple greedy algorithm and is much slower than than both FESTA and our approach. FESTA and our approach both recover optimal solutions for the ENCODE regions, however, FESTA ends up spending a very large amount of computational time in very large precincts, taking several hours to complete some of the data sets while our approach requires less than a minute.

3.2 Whole Genome Single SNP r^2

We ran our approach and the greedy approach over the entire genome wide HapMap data for the CEU population in order to find the minimal tag set to cover all SNPs with MAF ≥ 0.05 and $r^2 \geq 0.8$. Greedy resulted in 472729 tag SNPs while our approach needed only 468967 over the entire 1692323 SNP data set. This modest decrease shows that in the single SNP r^2 tag SNP selection

approach, greedy search performs almost as well as optimal search. Our program required less than one day on a single CPU to run over the whole genome. Once we compile into DNNF, which is done just once, we can perform a variety of queries and operations on the set of all optimal solutions [16], allowing for flexible design or optimization on a secondary criteria. For example, we can efficiently obtain the set of optimal solutions that contain some SNPs and do not contain other SNPs.

3.3 Mutli-marker Tag SNP Selection ENCODE Results

Although for the single SNP r^2 tag SNP selection problem, the greedy algorithm achieves a solution close to the optimal solution, this is not the case for multi-marker tag SNP selection. We compare our method to the popular Tagger [18] method over the Encode region ENm010 in the CEU population. This regions contains 567 SNPs with minor allele frequency (MAF) greater than 0.05. Tagger first computes a single SNP tag set using greedy search resulting in 159 tag SNPs and then uses a "roll back" procedure in which a SNP s_i is removed from the tag set if another pair of SNPs in the tag set cover s_i with $r^2 = 1.0$. That is, redundant SNPs are removed from the tag set. Tagger's multi-marker approach does reduce the number of SNPs required to cover an ENCODE region to 141 SNPs compared to 157 optimal single SNP tags, but the reduction is far from optimal. Our method requires only 72 SNPs to cover the ENCODE region a 54% and 40% reduction over single SNP tags and the Tagger's multi-marker tags.

4 Discussion

We have presented a novel method for solving a variety of tag SNP selection problems which guarantee a solution the the minimal number of tag SNPs. Since the problem is NP-hard, our methods provide no guarantees on the running time of the algorithms. However, we have demonstrated that in practice, our approach is very efficient and much faster than previously proposed exhaustive methods. Improvements over the classic single SNP r^2 tagging problem are modest compared to greedy search. The FESTA [24] algorithm also achieves these results over the ENCODE regions, but is not guaranteed to discover a minimal tag set in the general case. We outperform FESTA in terms of running time. In addition, our approach allows us to characterize *all* optimal solutions as opposed to just those containing perfectly linked SNPs. This permits flexible tag set choice that can be further optimized over secondary criteria. This method is also extensible to other measures of SNP coverage besides r^2.

Recent work has shown the multi-marker methods are more powerful than single SNP techniques in the context of association studies. While a variety of multi-marker statistical tests exist, the current optimal tagging methods such as FESTA are not able to tag for multiple markers efficiently. Our SAT based method is able to find optimal multi-marker tags for pairs of SNPs over the dense ENCODE regions. The gain for optimal tagging over greedy search in this

context is significantly better than for the single SNP with improvement over the popular Tagger [18] method reaching 40%. Since the cost of custom genotype arrays remains high, this tool is valuable for follow up association studies. That is, once genome wide results have been found, further genotyping must be done to localize the region containing the causal variant. Intelligent choice of tag sets for follow up studies can greatly improve their power and until now, multi-marker tagging for follow up has been non-optimal.

In addition to r^2, other factors that influence power are the minor allele frequency of the causal variant and the strength of the effect of a variant on disease risk. We can also formulate a variant of the tag SNP selection problem in which the goal is to optimize the statistical power given that we are collecting a fixed number of tag SNPs. For this problem, formulating the problem as a SAT problem is difficult because the constraint of using a fixed number of tags creates long range dependencies which complicate the decomposition of the problem into a DNNF. A typical approach to solve problems with such constraints is to formulate the problem as a Max-SAT problem where each clause has a weight and we are trying to satisfy the largest number of clauses. In future work, we will explore formulating the maximum power tag SNP selection problem as a Max-SAT problem in order to choose tags which optimally maximize the statistical power.

Our method and whole genome optimal data sets are available for use via webserver at http://whap.cs.ucla.edu.

Acknowledgments

N.Z. is supported by the Microsoft Graduate Research Fellowship. N.Z., B.H. and E.E. are supported by the National Science Foundation Grant No. 0513612, and National Institutes of Health Grant No. 1K25HL080079.

References

1. Bafna, V., Halldorsson, B.V., Schwartz, R., Clark, A., Istrail, S.: Haplotypes and informative snp selection: Don't block out information. In: RECOMB, pp. 19–27 (2003)
2. Barrett, A.: From hybrid systems to universal plans via domain compilation. In: Proceedings of the 14th International Conference on Planning and Scheduling (ICAPS), pp. 44–51 (2004)
3. Barrett, A.: Model compilation for real-time planning and diagnosis with feedback. In: Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI), pp. 1195–1200 (2005)
4. Bonet, B., Geffner, H.: Heuristics for planning with penalties and rewards using compiled knowledge. In: Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR), pp. 452–462 (2006)
5. Halldorsson, B.V., Bafna, V., Lippert, R., Schwartz, R., De La Vega, F.M., Clark, A.G., Istrail,: Optimal haplotype block-free selection of tagging snps for genome-wide assoaciation studies. Genome Research 14, 1633–1640 (2004)
6. The c2d compiler, `http://reasoning.cs.ucla.edu/c2d/`

7. Carlson, C.S., Eberle, M.A., Rieder, M.J., Yi, Q., Kruglyak, L., Nickerson, D.A.: Selecting a maximally informative set of single-nucleotide polymorphisms for association analyses using linkage disequilibrium. Am. J. Hum. Genet. 74(1), 106–120 (2004)
8. Chavira, M., Darwiche, A.: Compiling Bayesian networks with local structure. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI), pp. 1306–1312 (2005)
9. Chavira, M., Darwiche, A., Jaeger, M.: Compiling relational Bayesian networks for exact inference. International Journal of Approximate Reasoning 42(1–2), 4–20 (2006)
10. The International HapMap Consortium. A haplotype map of the human genome 437(7063), 1299–1320 (2005)
11. The Wellcome Trust Case Control Consortium. Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. Nature 447, 661–678 (2007)
12. Darwiche, A.: Decomposable negation normal form. Journal of the ACM 48(4), 608–647 (2001)
13. Darwiche, A.: On the tractability of counting theory models and its application to belief revision and truth maintenance. Journal of Applied Non-Classical Logics 11(1-2), 11–34 (2001)
14. Darwiche, A.: A compiler for deterministic, decomposable negation normal form. In: Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI), pp. 627–634. AAAI Press, Menlo Park (2002)
15. Darwiche, A.: New advances in compiling CNF to decomposable negational normal form. In: Proceedings of European Conference on Artificial Intelligence, pp. 328–332 (2004)
16. Darwiche, A., Marquis, P.: A knowledge compilation map. Journal of Artificial Intelligence Research 17, 229–264 (2002)
17. Darwiche, A., Marquis, P.: Compiling propositional weighted bases. Artificial Intelligence 157(1-2), 81–113 (2004)
18. de Bakker, P.I.W., Yelensky, R., Pe'er, I., Gabriel, S.B., Daly, M.J., Altshuler, D.: Efficiency and power in genetic association studies. Nat. Genet. 37(11), 1217–1223 (2005)
19. Elliott, P., Williams, B.: Dnnf-based belief state estimation. In: Proceedings of the 21st National Conference on Artificial Intelligence (AAAI 2006) (2006)
20. Darwiche, A., Palacios, H., Bonet, B., Geffner, H.: Pruning conformant plans by counting models on compiled d-dnnf representations. In: Proceedings of the 15th International Conference on Planning and Scheduling (ICAPS), pp. 141–150. AAAI Press, Menlo Park (2005)
21. Huang, J.: Complan: A conformant probabilistic planner. In: Proceedings of the 16th International Conference on Planning and Scheduling (ICAPS) (2006)
22. Huang, J., Darwiche, A.: On compiling system models for faster and more scalable diagnosis. In: Proceedings of the 20th National Conference on Artificial Intelligence (AAAI), pp. 300–306 (2005)
23. Pe'er, I., de Bakker, P.I.W., Maller, J., Yelensky, R., Altshuler, D., Daly, M.: Evaluating and improving power in whole genome association studies using fixed marker sets. Nature Genetics 38, 663–667 (2006)
24. Qin, Z.S., Gopalakrishnan, S., Abecasis, G.R.: An efficient comprehensive search algorithm for tagSNP selection using linkage disequilibrium criteria. Bioinformatics 22(2), 220–225 (2006)

25. Sang, T., Beame, P., Kautz, H.: Solving Bayesian networks by weighted model counting. In: Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI 2005), vol. 1, pp. 475–482. AAAI Press, Menlo Park (2005)
26. Siddiqi, S., Huang, J.: Hierarchical diagnosis of multiple faults. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI) (2007)
27. Halldorsson, B.V., Istraila, S., De La Vegab, F.M.: Optimal selection of snp markers for disease association studies. Human Heredity 58, 190–202 (2004)
28. Wachter, M., Haenni, R.: Logical compilation of bayesian networks. Technical Report iam-06-006, University of Bern, Switzerland (2006)
29. Yolifè Arvelo, M.-E.V., Bonet, B.: Compilation of query–rewriting problems into tractable fragments of propositional logic. In: Proceedings of AAAI National Conference (2006)
30. Zaitlen, N., Kang, H.M., Eskin, E., Halperin, E.: Leveraging the HapMap correlation structure in association studies. Am. J. Hum. Genet. 80(4), 683–691 (2007)