Efficient Algorithms for Genome-Wide TagSNP Selection across Populations via the Linkage Disequilibrium Criterion

Lan Liu, Yonghui Wu, Stefano Lonardi and Tao Jiang[*]
Department of Computer Science and Engineering, University of California,
Riverside, CA 92507, USA
[*]*Email:jiang@cs.ucr.edu*

In this paper, we study the tagSNP selection problem on multiple populations using the pairwise r^2 linkage disequilibrium criterion. We propose a novel combinatorial optimization model for the tagSNP selection problem, called the *minimum common tagSNP selection* (MCTS) problem, and present efficient solutions for MCTS. Our approach consists of three main steps including (i) partitioning the SNP markers into small disjoint components, (ii) applying some data reduction rules to simplify the problem, and (iii) applying either a fast greedy algorithm or a Lagrangian relaxation algorithm to solve the remaining (general) MCTS. These algorithms also provide lower bounds on tagging (*i.e.* the minimum number of tagSNPs needed). The lower bounds allow us to evaluate how far our solution is from the optimum. To the best of our knowledge, it is the first time tagging lower bounds are discussed in the literature. We assess the performance of our algorithms on real HapMap data for genome-wide tagging. The experiments demonstrate that our algorithms run 3 to 4 orders of magnitude faster than the existing single-population tagging programs like FESTA, LD-Select and the multiple-population tagging method MultiPop-TagSelect. Our method also greatly reduces the required tagSNPs compared to LD-Select on a single population and MultiPop-TagSelect on multiple populations. Moreover, the numbers of tagSNPs selected by our algorithms are almost optimal since they are very close to the corresponding lower bounds obtained by our method.

1. INTRODUCTION

The rapid development of high-throughput genotyping technologies has recently enabled genome-wide association studies to detect connections between genetic variants and human diseases. *Single-nucleotide polymorphism* (SNP) is the most frequent form of polymorphism in the human genome. Common SNPs with *minor-allele frequency* (MAF) of 5% have been estimated to occur once every $\sim$600 bps [18], and there are more than 10 million verified SNPs in dbSNP [11]. Given these numbers, it is currently infeasible to consider all the available SNPs to carry out association studies. This motivates the selection of a *subset* of informative SNPs, called *tagSNPs*.

The selection of tagSNPs *in silico* is a well-studied research topic. Existing computational methods for tagSNP selection can be classified into two categories: *haplotype-based* methods [1, 12, 17, 19, 24, 28, 31, 32, 34] and *haplotype-independent* methods [5, 15, 16, 20–22, 25, 27, 26]. The haplotype-based methods require phased multi-locus haplotypes, whereas the haplotype-independent methods do not require haplotype information. The main shortcoming of haplotype-based methods is that the preprocessing step (*i.e.* the inference of haplotypes from genotypes) is computationally demanding. In addition, since there is not an authoritative inference method, the haplotypes generated by the existing haplotype inference methods are often quite different [7, 32, 35]. Consequently, the tagSNPs selected by the haplotype-based methods would be quite different. Recently, Carlson *et al.* [5] proposed a haplotype-independent method that employs the r^2 *linkage disequilibrium* (LD) statistical criterion to measure the association between SNPs. The tagSNPs selected by this method are shown to be effective in disease association mapping studies, because the measure r^2 is directly related to the statistical power of association mapping. Because this method has comparable performance at a lower computational cost than many other methods [33, 27], tagging approaches based on r^2 LD statistics have gained popularity among researchers in the SNP community [2, 5, 8, 22, 26, 33].

Most approaches using the r^2 criterion require that tagSNPs be defined within a single population, because LD patterns (see the caption of Figure 1(A) for a definition) are quite susceptible to population stratification [5]. In two populations with different evolutionary histories, a pair of SNPs having remarkably different allele frequencies and very weak LD may show strong LD in the admixed population (see such an example in Table 1). Recent study [6] shows that the LD patterns and allele frequencies across populations are very different [6, 29] in fact. For example, among the populations collected in the HapMap project (*i.e.* YRI, CEU, CHB and JPT), 81% of the SNPs in YRI population have a near perfect proxy (*i.e.* SNPs that have $r^2 \geq 0.8$ with other SNPs), while in the other three populations, 91% of the SNPs have a near perfect proxy [9]. Therefore, tagSNPs picked from the combined populations or one of the populations might not be sufficient to capture the variations in all populations. In order to maintain the power of association mapping, we need generate a common (or universal) tagSNP set to type all the populations with sufficient accuracy.

A simple approach to select a universal tagSNP set is to tag one population first and then select a supplementary set for each of the other populations one by one

[*]Corresponding author.

Table 1. r^2 statistics for a pair of SNP markers in a single and admixed populations. One SNP has alleles denoted as A and a while the other SNP has alleles denoted as B and b. Population 3 is an even mixture of populations 1 and 2.

Population 1				Population 2				Population 3			
	B	b			B	b			B	b	
A	0.9025	0.0475	0.95	A	0.0025	0.0475	0.05	A	0.4525	0.0475	0.5
a	0.0475	0.0025	0.05	a	0.0475	0.9025	0.95	a	0.0475	0.4525	0.5
	0.95	0.05	$r^2 = 0$		0.05	0.95	$r^2 = 0$		0.5	0.5	$r^2 = 0.6561$

[2, 23, 22]. For instance, we can select a tagSNP set for non-African populations and a supplement for populations with significant African ancestry [23]. However, this sequential approach might not give a satisfactory solution, as the tagSNP set selected for one population might be far from being adequate to type the SNPs of the remaining populations. As a result, the supplementary tagSNP sets are large and the total number of tagSNPs chosen is far from the optimum. Moreover, the performance of the approach is sensitive to the specific order of the input populations. In order to generate the smallest set of tagSNPs on K populations, one would have to execute the tagging procedure $K!$ times considering all possible orderings, which would be extremely inefficient for genome-wide tagging. We can improve the performance of the tagging approach by evaluating multiple populations at the same time. When choosing tagSNPs, we prefer those with "good properties" with respect to the collection of populations as a whole. An example of our tagging strategy is given in Figure 1.

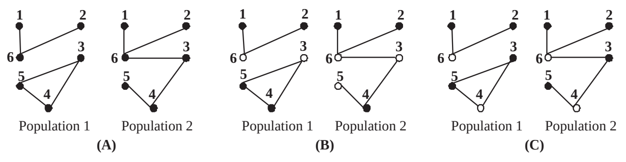

Fig. 1. (A). LD patterns in two populations. The vertices denote the SNP markers and the edges denote pairs of markers with strong LD (*i.e.* the r^2 measure between the markers is greater than a given threshold). (B). Tagging results of the above simple sequential approach. We first choose markers 3 and 6 to tag population 1 and then choose an additional marker 5 to tag population 2. Three markers are selected in total to tag both populations. (C). Tagging results of an improved approach. We select markers 4 and 6 considering both populations simultaneously. Only two markers are selected in total to tag both populations.

Previous work on tagSNP selection based on the linkage disequilibrium criterion. There is a large body of scientific literature on the problem of selecting tagSNPs based on the r^2 LD criterion. Carlson *et al.* suggested a greedy procedure called LD-Select, which works as follows: (i) select the SNP with the maximum number of proxies, (ii) remove the SNP and its proxies from consideration, and (iii) repeat the above two steps until all SNPs have been tagged [5]. This algorithm is very simple, however it may miss solutions with the smallest number of tagSNPs in general, as shown in [26]. More recently, Qin *et al.* implemented a comprehensive search algorithm called FESTA, which first breaks down a large set of markers into disjoint pieces (called *precincts*), and then performs an exhaustive search on each piece if the estimated computational cost is below a certain threshold [26]. FESTA usually gives a better solution than LD-Select, but due to the fact that it employs exhaustive search, it is too slow to be practical for genome-wide tagSNP selection.

The above methods are only applicable to single population tagSNP selection. Recently, Howie *et al.* presented an algorithm for multiple populations, called MultiPop-TagSelect. MultiPop-TagSelect combines the tagSNPs selected for each population by LD-Select to produce a universal tagSNP set for a collection of populations [13]. The algorithm works reliably, and it could in principle be used with any tagSNP selection method for single populations. However, its accuracy highly depends on the performance of the single-population tagSNP selection method. Magi *et al.* [22] also designed a software tool called REAPER which is rather similar to LD-Select if applied to a single population. To select a universal tagSNP set for several populations, it first selects a tagSNP set for one population, and then it selects a supplement for the remaining populations one by one. As mentioned above, the performance of the method crucially depends on the choice of the initial tagSNP set and the ordering of the populations. It is not clear, moreover, how one should select tagSNPs for the first population so as to minimize the size of the final solution.

Our contribution on tagSNP section based on the linkage disequilibrium criterion. In this paper, we take a different approach to the multi-population tagSNP selection problem. Contrary to the previous methods, we do not generate a tagSNP set for each individual population separately, but rather we evaluate all the populations at the same time. The method that we propose could be used to generate a universal or cosmopolitan tagSNP set for multi-ethnic, ethic-unknown or admixed populations [13].

The main idea of our approach is to transform a multi-population tagSNP selection problem, called the *minimum common tagSNP selection* (MCTS) problem (to be defined more precisely later in the paper), into a minimum common dominating vertex set problem on multiple graphs. Each graph corresponds to one of the populations under consideration. The vertices in a graph correspond to the SNP markers of the population, and there is an edge between two markers when they are in strong LD (according to some given threshold). To find an optimal solution MCTS, we first decompose it into disjoint subproblems, each of which is essentially a connected component of the union graph[a] and represents a precinct as defined in [26]. Then, for each precinct, we apply three data reduction rules repeatedly to further reduce the size of the subproblem, until none of the rules can be applied anymore. Finally, the reduced subproblems are solved by either a simple greedy approach (similar to *cosmopolitan tagging* [2]) or a more sophisticated Lagrangian relaxation heuristic. Both algorithms will be explained in detail later in the paper. Along with the solution produced by our algorithm, we also obtain lower bounds on the minimum number of tagSNPs required, which allows us to quantitatively assess how close our solution is from the optimum.

We evaluate the performance of our method on real HapMap data for genome-wide tagging. The experimental results demonstrate that our algorithms run 3 to 4 orders of magnitude faster than the existing single-population tagging programs like FESTA, LD-Select and the multiple-population tagging method MultiPop-TagSelect. Our method also greatly reduces the required tagSNPs compared to LD-Select on a single population and MultiPop-TagSelect on multiple populations. Moreover, the numbers of tagSNPs selected by our algorithms are almost optimal since they are very close to the corresponding lower bounds provided by our method. For example, the gap between our solution and the lower bound is 1061 SNPs with r^2 threshold being 0.5 and 142 SNPs with the r^2 threshold being 0.8, given the entire human genome with 2,862,454 SNPs (MAF being 5%).

The rest of the paper is organized as follows. In Section 2, we first propose a combinatorial optimization model for the MCTS problem and then present a computational complexity result. In Section 3, we introduce three rules to reduce the size of the problem, and devise a greedy tagging algorithm, called GreedyTag, and a Lagrangian relaxation heuristic, called LRTag. After showing the experimental results in Section 4, we conclude the paper with some remarks about the performance of our tagging method in Section 5. Due to page limit, some of the illustrative figures and tables are given in the appendix.

2. FORMULATION OF THE MCTS PROBLEM

Consider K distinct populations and a set V of biallelic SNP markers $v_1, v_2, \ldots, v_n$. Since the r^2 coefficient is unreliable for rare SNPs when the sample size is small [5], we will consider only SNPs with MAF $\geq 5\%$. The set of SNPs might be different from population to population. We use $V_i \subseteq V$ to denote the SNP set in population i. Clearly, we have $V = V_1 \cup V_2 \cup \cdots \cup V_k$.

For a pair of SNP markers v_{j_1} and v_{j_2} in a population i (for any $1 \leq i \leq K$), the r^2 coefficient between them is denoted by $r_i^2(v_{j_1}, v_{j_2})$. Markers v_{j_1} and v_{j_2} are said to be in *high LD* in population i, if $r_i^2(v_{j_1}, v_{j_2}) \geq \gamma_0$, where γ_0 is a pre-defined threshold (γ_0 will be set to 0.5 or higher in our study). Moreover, v_{j_1} (or v_{j_2}) is considered being the *tagSNP* or *proxy* for v_{j_2} (or v_{j_1}, respectively) in population i. For convenience, we define E_i to be the set containing all the high-LD marker pairs in population i, *i.e.* $E_i = \{(v_{j_1}, v_{j_2}) | r_i^2(v_{j_1}, v_{j_2}) \geq \gamma_0,\ v_{j_1}, v_{j_2} \in E_i\}$. Now we can formally define the MCTS problem.

MINIMUM COMMON TAGSNP SELECTION(MCTS)
Instance: A collection of K populations and a set V of biallelic SNP markers. Each population i ($1 \leq i \leq K$) has its marker set $V_i \subseteq V$ and LD patterns $E_i = \{(v_{j_1}, v_{j_2}) | r_i^2(v_{j_1}, v_{j_2}) \geq \gamma_0,\ v_{j_1}, v_{j_2} \in V_i\}$, where γ_0 is a pre-defined threshold.
Feasible solution: A subset $T \subseteq V$ such that for any marker $v \in V_i, v \notin T$ from some population i, there exists a marker v' in $T \cap V_i$ with $(v, v') \in E_i$ (that is, $r_i^2(v, v') \geq \gamma_0$).
Objective: Minimize |T|.

It is easy to observe that any feasible solution to the MCTS problem is a common dominating vertex set in the graphs $\{G_i | 1 \leq i \leq K\}$, where $G_i = (V_i, E_i)$. In particular, the smallest set of tagSNPs for a single population is a minimum dominating vertex set of the corresponding graph. Obviously, the MCTS problem is NP-hard, since it is a generalization of the minimum dominating vertex set problem, which is known to be NP-hard [4].

Theorem 2.1. *The MCTS problem is NP-hard.*

We introduce some additional notations to be used later. To differentiate the occurrences of a marker in different populations, we use v_j^i to represent the j^{th} marker appearing in the i^{th} population. Given a marker $v_j \in V$, we define the following two sets:

$$N^i(v_j) = \{v_{j'}^i | (v_j, v_{j'}) \in E_i, v_j, v_{j'} \in V_i\} \cup \{v_j^i | v_j \in V_i\}$$
$$N^*(v_j) = \bigcup_{1 \leq i \leq K} N^i(v_j) \tag{1}$$

The set $N^i(v_j)$ represents the subset of markers in strong-LD with v_j in population i, and the set $N^*(v_j)$ represents the union of such subsets for all the populations. Note that, $N^i(v_j)$ is empty if $v_j \notin V_i$. Given a marker $v_j \in V_i$ in population i, we define the following set:

$$C(v_j^i) = \{v_{j'} | (v_j, v_{j'}) \in E_i, v_j, v_{j'} \in V_i\} \cup \{v_j\} \tag{2}$$

The set $C(v_j^i)$ is the subset of markers each of which can tag the occurrence v_j^i, whereas $N^*(v_j)$ is the subset of occurrences that the marker v_j can tag.

[a]Given graphs $G_i = (V_i, E_i)(1 \leq i \leq k)$, the union graph is defined as $G = (V, E)$, where $V = \bigcup_i V_i$ and $E = \bigcup_i E_i$.

Based on the above definitions, the MCTS problem can also be viewed as the following set cover problem. Given the universe $\mathcal{U} = \bigcup_{1 \leq i \leq K} \{v_j^i | v_j \in V_i\}$ and the collection $\mathcal{C} = \{N^*(v_j) | v_j \in V\}$, find a subcollection of sets from $\mathcal{C}$ to cover $\mathcal{U}$. Clearly, the number of sets in a minimum set cover is equal to the number of markers in a minimum tagSNP set.

Consequencely, approximation algorithms that solve set cover can be applied to MCTS. In practice, greedy algorithms are commonly used for set cover due to their simplicity and effectiveness. The simplest greedy algorithm for set cover, which picks the set that covers the most number of uncovered elements each time, achieves an approximation ratio of $\log(m)$, where m is the number of elements to be covered [30]. This implies a $\log(Kn)$ approximation algorithm for MCTS, $|\mathcal{U}| \leq Kn$.

However, the approximation ratio of $\log(Kn)$ could be too large in practice, due to the fact that V may contain millions of markers and $n = |V|$. Therefore, the solution produced by the above greedy approach may not be sufficiently small. We aim to design efficient heuristics to provide better solutions.

3. OPTIMIZATION TECHNIQUES TO SOLVE THE MCTS PROBLEM

In principle, a minimum common tagSNP set can be found by exhaustive search. In reality, there are millions of markers, and it is infeasible to conduct the exhaustive search. Since human chromosomes consist of high-LD regions (*i.e.* haplotype blocks) interspersed with *recombination hotspots*, we partition the markers into precincts such that markers in strong LD belong in the same precinct. In this way, we could narrow down the search space and improve the efficiency of our algorithm.

In order to deal with multiple populations, we extend the concept of precinct defined originally in [26]. We say that two markers are in the same *precinct* if and only if they are in strong LD in some population. Based on the simple observation that no marker in a precinct can tag a marker in another tag a marker in another precinct, we can obtain a minimum tagSNP set for the combining the minimum tagSNP sets for each precinct. The precincts can be easily identified by running a breath first search (BFS) in the union graph G. By partitioning the markers into precincts, we decompose the original problem into a set of disjoint subproblems of much smaller sizes. We then select tagSNPs for each precinct independently, which could save a lot of running time.

3.1. Data Reduction Rules

We introduce three simple data reduction rules to further reduce the subproblem sizes and improve efficiency.

Rule 1: *Pick all irreplaceable markers.* If a marker v_j has no proxy from population i (that is, v_j is a singleton in $G_i = (V_i, E_i)$), then marker v_j must be in the minimum tagSNP set.

Rule 2: *Remove less informative markers.* Given two markers $v_{j'}$ and v_j, if $N^*(v_{j'}) \subseteq N^*(v_j)$, we say that v_j is more *informative* than $v_{j'}$. Similarly, given a set of markers v_{j_1}, v_{j_1},..., v_{j_k}, if $N^*(v_{j_1}) \subseteq N^*(v_{j_2}) \subseteq \ldots \subseteq N^*(v_{j_k})$, v_{j_k} is called the *maximally informative* SNP marker in the set. It is clear that we can discard less informative SNPs and only keep those maximally informative ones without degrading the quality of the solution.

Rule 3: *Remove less stringent occurrences.* Given two occurrences $v_{j'}^{i'}$ and v_j^i, if $C_j^i \subseteq C_{j'}^{i'}$, we say that $v_{j'}^{i'}$ is *less stringent* than v_j^i. Similarly, given a set of occurrences $v_{j_1}^{i_1}$, $v_{j_2}^{i_2}$,..., $v_{j_k}^{i_k}$, if $C_{j_k}^{i_k} \subseteq \ldots \subseteq C_{j_2}^{i_2} \subseteq C_{j_1}^{i_1}$, the occurrence $v_{j_k}^{i_k}$ is called the *most stringent* occurrence in the set. Observe that the markers selected to tag the most stringent occurrences will also tag the less stringent occurrences. Therefore, we consider only the most stringent occurrences and discard the others.

The above rules can also be viewed as data reduction rules applied to a 0/1 matrix obtained as follows. Given the notations of the occurrence set $\mathcal{U}$, the marker set V and the neighborhood collections $\mathcal{C}$ introduced in previous section, the rows in the matrix represent $\mathcal{U}$, the columns denote V, and each cell (i, j) indicates whether the marker corresponding to column j can tag the occurrence corresponding to row i (*i.e.* the value of a cell is set to 1 if the marker can tag the occurrence, and 0 otherwise). Thus, Rule 2 (or Rule 3) is equivalent to redundant column deletion (or row deletion, respectively).

The above rules can be applied repeatedly and in any combination whenever applicable. The reduced problem obtained after the application of the above data reduction rules will be subject to our greedy algorithm or Lagrangian relaxation (LR) algorithm, as explained next.

3.2. A Greedy Algorithm for MCTS

Greedy algorithms are often desirable due to their simplicity and efficiency. The greedy algorithm, *GreedyTag*, below is adapted from the greedy algorithm for the set cover problem as presented in [30]. By first applying the above data reduction rules, we will show later in the paper that GreedyTag greatly outperforms the other greedy algorithms such as LD-Select and MultiPop-TagSelect. Moreover, a lower bound, called *GreedyTag_lb*, is produced by GreedyTag, which is equal to the number of tagSNPs selected by data reduction Rule 1. Even though the lower bound is is somewhat loose since we only consider Rule 1, it turned out to be pretty tight in our experiments on real data (see Section 4 for more details). Due to space constraint, we present the pseudo-code of GreedyTag in the appendix.

3.3. A Lagrangian Relaxation Algorithm for MCTS

A subset T of SNPs can be denoted by its characteristic vector $\boldsymbol{t} = t_1 t_2 \ldots t_n$, where $t_i = 1$ if $v_i \in T$, and $t_i = 0$ otherwise. It is thus easy to formulate the following integer linear program for MCTS.

$$\begin{aligned} &\text{Minimize } |T| = \textstyle\sum_{1 \le j \le n} t_j \\ &\text{subject to } \textstyle\sum_{v_{j'} \in C(v_j^i)} t_{j'} \ge 1 \quad 1 \le i \le K \text{ and } 1 \le j \le n \\ &\qquad t_j \in \{0,1\}, 1 \le j \le n \end{aligned} \tag{3}$$

Our second algorithm for MCTS is based on the Lagrangian relaxation framework. We assign a non-negative vector $\boldsymbol{\lambda} = \lambda_{11}\lambda_{12}\ldots\lambda_{K,n}$ of Lagrangian multipliers to the inequalities, and obtain the following relaxed integer program.

$$\begin{aligned} &\text{Minimize } L(\boldsymbol{t}, \boldsymbol{\lambda}) \\ &\qquad = \textstyle\sum_{1 \le j \le n} t_j + \sum_{1 \le i \le K, 1 \le j \le n} \lambda_{i,j} (1 - \sum_{v_{j'} \in C(v_j^i)} t_{j'}) \\ &\text{subject to } t_j \in \{0,1\}, \lambda_{i,j} \ge 0, 1 \le i \le K, 1 \le j \le n \end{aligned} \tag{4}$$

For a given $\boldsymbol{\lambda}$, define $L(\boldsymbol{\lambda}) = min\ L(\boldsymbol{t}, \boldsymbol{\lambda})$. Observe that the size of any feasible tagSNP set T would be an upper bound for $L(\boldsymbol{\lambda})$ in (4), and any $L(\boldsymbol{\lambda})$ would be a lower bound for $|T|$. Hence, we look for $max\ L(\boldsymbol{\lambda})$, which gives the best lower bound for $min\ |T|$.

For any given $\boldsymbol{\lambda}$, we can easily obtain $L(\boldsymbol{\lambda})$ in (4) as follows. For convenience, we define $s(t_j)(1 \le j \le n)$ as:

$$s(t_j) = 1 - \sum_{1 \le i \le K,\ v_{j'} \in C(v_j^i)} \lambda_{i,j'} = 1 - \sum_{v_{j'}^i \in N^*(v_j)} \lambda_{i,j'}$$

which are the *Lagrangian costs* associated with t_j in (4). Rearranging the terms in (4), we have the objective function $L(\boldsymbol{t}, \boldsymbol{\lambda}) = \sum_{1 \le i \le K, 1 \le j \le n} \lambda_{i,j} + \sum_{1 \le j \le n} s(t_j) \cdot t_j$. In order to minimize the objective function, we have to set $t_j = 0$ if $s(t_j) > 0$, $t_j = 1$ if $s(t_j) < 0$, and t_j an arbitrary value if $s(t_j) = 0$.

The vector $\boldsymbol{t}$ obtained above may not be a feasible solution to (3). In other words, some occurrence might not be tagged by any marker in $T = \{v_j | t_j = 1, 1 \le j \le n\}$ induced by the characteristic vector $\boldsymbol{t}$. We will adopt a strategy *reduced cost heuristic* (RCH) introduced by Balas and Ho [3] to deal with this issue (the details are given in the pseudo-code in the appendix).

Next we need find a good multiplier vector $\boldsymbol{\lambda}$, *i.e.* one that gives a near optimal lower bound. We utilize a standard optimization technique called *subgradient optimization* [3], which iteratively updates the solution toward the subgradient direction to reach the optimum. We can define

$$S(\lambda_{i,j}) = 1 - \sum_{v_{j'} \in C(v_j^i)} t_{j'}$$

which simplifies $L(\boldsymbol{t}, \boldsymbol{\lambda}) = \sum_{1 \le j \le n} t_j + \sum_{1 \le i \le K, 1 \le j \le n} S(\lambda_{i,j}) \cdot \lambda_{i,j}$. Obviously, $\nabla\boldsymbol{\lambda} = (\nabla\lambda_{11}, \nabla\lambda_{12}, ..., \nabla\lambda_{K,n})$ where $\nabla\lambda_{i,j} = S(\lambda_{i,j})$. Starting from a initial setting $\boldsymbol{\lambda}^0$, we sequentially generate $\boldsymbol{\lambda}^1, \boldsymbol{\lambda}^2, \boldsymbol{\lambda}^3, \ldots$, based on the following formula

$$\boldsymbol{\lambda}^{k+1} = max\{\mathbf{0}, \boldsymbol{\lambda}^k + \alpha_k \frac{|T^*| - L^*}{||\nabla\boldsymbol{\lambda}^k||^2} \nabla\boldsymbol{\lambda}^k\} \tag{5}$$

where T^* is the smallest feasbile tagSNP set found so far (*i.e.* the best upper bound for $max\ L(\boldsymbol{t})$), L^* is the largest of $max\ L(\boldsymbol{t})$ found so far (*i.e.* the best lower bound for $max\ L(\boldsymbol{t})$), and $\{\alpha_0, \alpha_1, \ldots\}$ is a decreasing sequence of pre-defined scalars.

The pseudo-code for the Lagrangian relaxation algorithm, *LRTag*, is given in the appendix. In the algorithm, we start from a initial setting of $\boldsymbol{\lambda}^0$, generate a solution to $\boldsymbol{t}^0$ and extend it to a valid tagSNP set as mentioned above. Then we update $\boldsymbol{\lambda}^0$ into $\boldsymbol{\lambda}^1$ according to the formula (5). We repeat the process until we cannot improve $\boldsymbol{\lambda}$ or a pre-defined number of maximum iterations is reached. Over the entire iterative process, the smallest feasible set of tagSNPs found by LRTag would be output as a solution to the MCTS problem, and the largest $L(\boldsymbol{t})$ would be a lower bound for tagSNP selection, called *LRTag_lb*.

4. EXPERIMENTAL RESULTS

In our experiments, we test the algorithms GreedyTag and LRTag on the Hapmap populations, and compare their performance and efficiency with single-population tagging programs LD-Select and FESTA, and a multiple-population tagging program MultiPop-TagSelect. For convenience, we will also denote by GreedyTag the cardinality of a feasible tagSNP set obtained by the GreedyTag algorithm. We use similar notations for LRTag, LD-Select, FESTA and MultiPop-TagSelect.

Both of our algorithms calculate lower bounds on the minimum number of required tagSNPs, one of which is found by GreedyTag (*i.e.* GreedyTag_lb) and the other by LRTag (*i.e.* LRTag_lb). We define *gap* as the difference between the highest lower bound and the cardinality of the smallest tagSNP set found by our algorithms, which will be used to measure the quality of the solutions.

We apply all the methods on the entire human genome data involving chromosomes 1 through 22 and on all ENCODE regions (ENm010, ENm013, ENm014, ENr112, ENr113, ENr123, ENr131, ENr213, ENr232 and ENr321) genotyped by the HapMap project (release #19, NCBI build 34, October 2005). For the ENCODE data, we estimate the r^2 statistics by using a two-marker EM algorithm to compute the maximum-likelihood values of the four gamete frequencies, which is also commonly adopted by LD-Select and HaploView [2]. For the entire human genome data, we directly download the r^2 statistics from the HapMap website [10], generated by HaploView to save computational cost. Note that, HaploView only calculates LD for markers up to 250 kbps apart, which is reasonable because the LD for markers that are farther than 250 kbps would normally be very weak anyway, and high LD in such a case can happen only purely by chance. In order to save running time for dealing with the entire human genome data, we prune the LD pattern data downloaded from the HapMap website by keeping only entries with r^2 no less than 0.5.

We ran all the programs on a 32-processor SGI Altix 4700 supercomputer system with 1.6HZ CPU and 64 GB shared memory in the Computer Science Department, University of California - Riverside. Our GreedyTag and LRTag algorithms used up to 15 threads in parallel, while each of the other programs is single-threaded.[b]

4.1. Tagging the ENCODE Regions

A dense set of SNPs across ten large genomic regions have been produced by the HapMap ENCODE project. These regions serve as the foundation to evaluate the development of methodologies and technologies for detecting functional elements in human DNA. Each region is about 500Kb in length and has an SNP density about 1 SNP per 600 bps.

Tagging ENCODE regions for a single population. We tag each HapMap population separately by LD-Select, FESTA, and our new algorithms GreedyTag and LRTag. For illustration purposes, we only show the results for tagging the CEU population and compare the performance of the above algorithms in Table 2.

When the r^2 threshold is set as 0.5, the number of tagSNPs selected by our algorithm is on the average 9.3% of the total number of markers (the actual percentage number ranges from 5.1% to 15.3%). With a more stringent r^2 threshold of 0.8, the average number of tagSNPs rises to 17.6% of the total number of markers (ranging from 11.4% to 24.9%). The same trend was observed when applying our algorithms on the other populations (results are not shown due to space constraint).

On each ENCODE region, we observe that the gap between LRTag_lb and LRTag is at most one with the r^2 threshold being 0.5, and there is no gap when the r^2 threshold is set as 0.8. This demonstrates that our algorithm LRTag found near-optimal solutions in all test cases. In general, LRTag and GreedyTag always generated the smallest sets of tagSNPs, FESTA selected at most three more tagSNP, and LD-Select might select up to eight more tagSNPs.

Since our algorithms and FESTA are all near-optimal, we compare the time efficiency of these programs in Table 3. Because LD-Select takes genotype data as input and the other programs take pairwise LD data as input, we do not compare LD-Select's running times directly with those of the others here (generally speaking, it takes LD-Select from 30m to 2h on an ENCODE region). From Table 3, we can see that the running time of FESTA varied widely from 1s to 64h on different regions, while our algorithms GreedyTag and LRTag consistently took 1-2s on all regions. In conclusion, our algorithms were 3 to 4 orders of magnitude faster than FESTA in most of the cases, and found slightly smaller sets of tagSNPs.

Tagging ENCODE regions for multiple populations. We tag each and the entire ENCODE regions for all four HapMap populations by MultiPop-TagSelect, GreedyTag and LRTag. The tagging results of these methods on each ENCODE region are summarized in Table 4. We also highlight the results for region ENm013 and for the entire ENCODE region in Figure 2.

With the r^2 threshold set as 0.5, the number of tagSNPs selected by our algorithms is on the average 18.3% of the total number of markers (the actual percentage number ranges from 11.0% to 34.5%). With a more stringent r^2 threshold of 0.8, the average number of tagSNPs increases to 33.7% (ranging from 24.0% to 50.5%). We observe that LRTag always performs the best in these tests, followed by the GreedyTag algorithm, and MultiPop-TagSelect always performs worst. When r^2 threshold is set as 0.5, LRTag requires 16.4% fewer markers on the average than MultiPop-TagSelect. When the r^2 threshold is 0.8, LRTag usually requires 5.1% fewer markers on the average than MultiPop-TagSelect.

The gap between LRTag_lb and LRTag is at most two for each ENCODE region and totally six for all ENCODE regions with the r^2 threshold being 0.5. There is no gap with the r^2 threshold being 0.8.

4.2. Genome-wide Tagging

Because both LD-Select and MultiPop-TagSelect (written in Perl) took more than 20 hours to tag a single chromosome, we re-implemented their algorithms in C++, called *LD-Select** and *MultiPop-TagSelect**, respectively, in order give a fair comparison of the programs. Since FESTA's "greedy-exhaustive hybrid search" is very computational demanding and hard to emulate, we exclude FESTA from the following comparative study.

Tagging the human genome for a single population. We apply LD-Select*, GreedyTag and LRTag on each HapMap population separately. For illustration purposes, we only discuss the results for tagging the CEU population and compare the performance of the above three algorithms. The details can be found in Table 5 given in the appendix.

[b]Note that if a program runs in time of t with 15 threads, then its running time with one thread would be $15t$. This transformation can be used to compare the running times of our programs and those of the other programs on a single thread mode.

Table 2. Summary of tagSNPs identified by FESTA, LD-Select, GreedyTag and LRTag for a single population, CEU, on all ENCODE regions.

Region	ENm010	ENm013	ENm014	ENr112	ENr113	ENr123	ENr131	ENr213	ENr232	ENr321
# SNP	525	692	904	947	1080	864	990	612	457	544
$r^2 \geq 0.5$										
# precinct	39	27	47	52	40	30	83	42	64	52
# tagSNP (upper bound)										
LD-Select	62	38	65	84	77	69	112	62	72	68
FESTA	57	35	63	76	73	65	107	61	70	65
GreedyTag	56	35	63	76	73	62	107	61	70	64
LRTag	56	35	63	76	73	62	107	61	70	64
# tagSNP (lower bound)										
LRTag_lb	55	35	63	76	73	62	107	60	70	64
GreedyTag_lb	50	33	63	72	69	54	101	55	70	62
Gap	1	0	0	0	0	0	0	1	0	0
$r^2 \geq 0.8$										
# precinct	116	69	121	139	131	129	175	105	106	107
# tagSNP (upper bound)										
LD-Select	123	82	129	152	146	139	189	110	115	109
FESTA	122	79	129	152	143	139	186	110	114	109
GreedyTag	122	79	129	152	143	139	186	110	114	109
LRTag	122	79	129	152	143	139	186	110	114	109
# tagSNP (lower bound)										
GreedyTag_lb	122	79	129	152	143	139	186	110	114	109
LRTag_lb	122	79	129	150	143	139	186	107	110	109
Gap	0	0	0	0	0	0	0	0	0	0

Table 3. The time efficiency of FESTA, GreedyTag and LRTag for selecting tagSNPs from a single population, CEU, on all ENCODE regions. The running times are obtained on a 32-processor SGI Altix 4700 supercomputer system.

Region	ENm010	ENm013	ENm014	ENr112	ENr113	ENr123	ENr131	ENr213	ENr232	ENr321
$r^2 \geq 0.5$										
FESTA	3h14m	3h16m	4h51m	3h18m	14h24m	64h4m	5h13m	2h24m	1h38m	45m19s
GreedyTag	1s	1s	1s	1s	1s	1s	1s	1s	1s	1s
LRTag	1s	1s	1s	2s	1s	2s	1s	1s	1s	1s
$r^2 \geq 0.8$										
FESTA	1s	3s	28m20s	44m6s	12m8s	50m16s	3s	1s	2s	1s
GreedyTag	<1s	<1s	<1s	<1s	<1s	<1s	<1s	<1s	<1s	<1s
LRTag	<1s	<1s	<1s	<1s	<1s	<1s	<1s	<1s	<1s	<1s

Table 4. Summary of tagSNPs identified by MultiPop-TagSelect, GreedyTag and LRTag for all HapMap populations on ENCODE regions.

Region	ENm010	ENm013	ENm014	ENr112	ENr113	ENr123	ENr131	ENr213	ENr232	ENr321
# SNP	783	1063	1261	1158	1485	1221	1186	900	777	1025
$r^2 \geq 0.5$										
# precinct	65	38	48	44	67	38	73	56	126	53
# tagSNP (upper bound)										
MultiPop-TagSelect	206	150	201	238	260	181	306	200	286	233
GreedyTag	179	117	164	184	228	141	257	173	268	193
LRTag	179	117	164	184	228	141	257	173	268	193
# tagSNP (lower bound)										
LRTag_lb	178	117	162	182	226	141	256	173	268	193
GreedyTag_lb	169	107	149	168	218	139	250	173	264	189
Gap	1	0	2	2	0	0	1	0	0	0
$r^2 \geq 0.8$										
# precinct	156	111	146	101	209	106	194	170	210	191
# tagSNP (upper bound)										
MultiPop-TagSelect	338	275	321	425	454	329	462	324	402	396
GreedyTag	322	255	305	391	437	300	445	318	392	377
LRTag	322	255	305	389	437	300	445	318	392	377
# tagSNP (lower bound)										
LRTag_lb	322	255	305	389	437	300	445	318	392	377
GreedyTag_lb	319	253	303	374	435	300	443	318	392	377
Gap	0	0	0	0	0	0	0	0	0	0

With the r^2 threshold set as 0.5, the number of tagSNPs selected by our algorithms is 14.4% of the total number of markers on the average (the actual percentage ranges from 11.2% to 21.4%). With a more stringent r^2 threshold of 0.8, the average number of tagSNPs increases to 26.6% (ranging from 22.2% to 35.5%). Similar trends were observed when applying our algorithms to the other populations (the results are not shown due to space limitation).

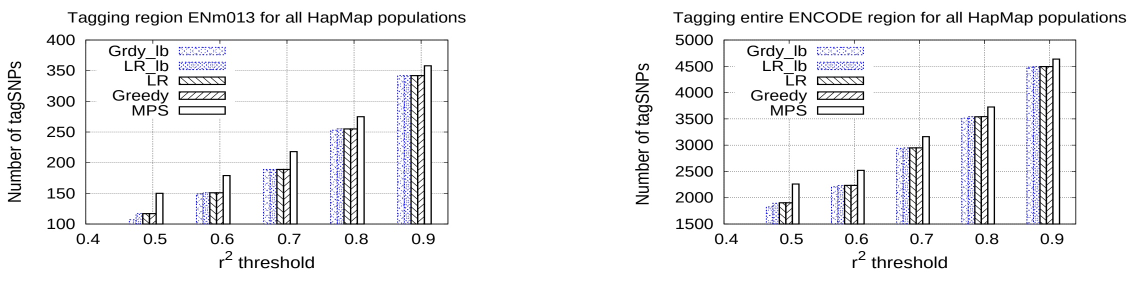

Fig. 2. (A). Tagging for HapMap populations on region ENm013 with 783 markers. (B). Tagging for HapMap populations on all ENCODE regions with 10,859 markers. Here, "Grdy_lb" stands for "GreedyTag_lb", "LRTag_lb" stands for "LRTag_lb", "LR" stands for "LRTag", "Greedy" stands for "GreedyTag", and "MPS" stands for "MultiPopTagSelect".

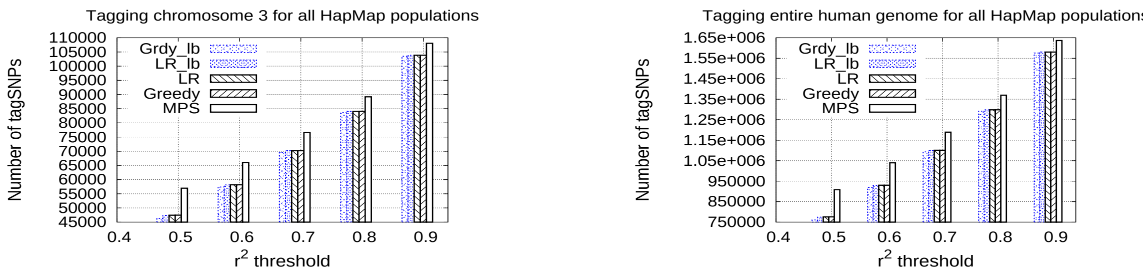

Fig. 3. (A). Tagging chromosome 3 for all HapMap populations with 196,535 markers. (B). Tagging the entire human genome for all HapMap populations with 2,862,454 markers. See the caption of Figure 2 for the definitions of the legends in the subfigures.

We observe that LRTag always performs the best, followed by the GreedyTag algorithm, and LD-Select* always performs the worst. With the r^2 threshold set as 0.5, LRTag usually requires 4.9% fewer tagSNPs (the actual percentage number ranges from 2.8% to 6.3%) on average than LD-Select* on each chromosome. When the r^2 threshold is increased to 0.8, LRTag usually requires 1.2% fewer tagSNPs (ranging from 0.07% to 1.5%) on average than LD-Select*.

We can see that, on each chromosome, the gap between the lower bound from LRTag_lb and the upper bound obtained by LRTag is on the average 7 (the actual number ranges from 1 to 20) with the r^2 threshold set as 0.5 and less than 1 (ranging from 0 to 2) with the r^2 threshold set being 0.8. This demonstrates that LRTag finds near-optimal solutions in all test cases even for genome-wide tagging on a single population. In fact, the performance of GreedyTag is not bad either.

Tagging the human genome for multiple populations. Finally, we tag the entire human genome for all four HapMap populations by MultiPop-TagSelect, GreedyTag and LRTag. We summarize the tagging results of these methods on each chromosome in Table 8 (given in the appendix), and then highlight the results for chromosome 3 and all chromosomes in Figure 3.

With the r^2 threshold set as 0.5, the number of tagSNPs selected by our methods is on the average 27.3% of the total number of markers (the actual percentage ranges from 21.9% to 47.2%). With a more stringent r^2 threshold of 0.8, the average number of tagSNPs increases to 46.0% (ranging from 29.4% to 60.4%). Based on Table 8, we observe that LRTag always performs slightly better than GreedyTag and significantly better than MultiPop-TagSelect*. With the r^2 threshold set as 0.5, LRTag requires 6.8% fewer tagSNPs on average (the actual number ranges from 4.0% to 8.0%) than MultiPop-TagSelect* on each chromosome. With the r^2 threshold set as 0.8, LRTag requires 3.6% fewer markers on average (ranging from 2.7% to 4.3%) than MultiPop-TagSelect*.

The gap between the lower bound from LRTag_lb and upper bound of LRTag is on the average 48 for each chromosome (the actual number ranges from 6 to 109) with the r^2 threshold set as 0.5, and 6.5 (ranging from 0 to 16) with the r^2 threshold set being 0.8, as shown in Table 8.

5. CONCLUSION

Our LRTag and GreedyTag algorithms run quickly on ENCODE regions and the entire human genome for both single and multiple populations. On an ENCODE region with the r^2 threshold being 0.5, it takes our algorithms no more than 2 seconds to tag a single population (as shown in Table 3) and less than 7 seconds to tag multiple populations (as displayed in Table 9 in the appendix). On a human chromosome, it takes no more than 4 minutes to tag a single population (as shown in Table 6 in the appendix) and less than 12 minutes to tag on multiple populations (as displayed in Table 7 in the appendix). For r^2 thresholds greater than 0.5, our algorithms run faster. Hence, for any given r^2 threshold, it takes our algorithms less than a minute to tag the entire ENCODE region and less than an hour to tag the entire human genome.

If the number of populations of interest increases, the genotyping density increases or the r^2 threshold increases, the number of required tagSNPs also increases. For example, on multiple HapMap populations with the r^2 threshold being 0.5, we need to tag one SNP for about every 6 SNPs on the densely genotyped ENCODE regions. We need to tag one SNP for about every 4 SNPs on sparsely genotyped HapMap chromosomes.

All the lower and upper bounds produced by the discussed methods are shown in Figure 2 and Figure 3. In the figures, we tag the ENCODE regions and human genome on the HapMap populations with the r^2 thresholds being 0.5, 0.6, 0.7 and 0.8 separately. From all these test cases, we observe that LRTag always chooses the smallest set of tagSNPs, closely followed by GreedyTag, while MultiPop-TagSelect chooses the largest set.

LRTag_lb always provides the best lower bound and LRTag the best upper bound among all methods considered. The simple greedy algorithm, GreedyTag, chooses slightly more tagSNPs than LRTag and the lower bound GreedyTag_lb is slightly lower than LRTag_lb, which indicates that the data reduction rules in Section 3 are very powerful.

When the r^2 threshold increases, the size of the precincts decreases. Consequently, the gap between the lower bound and the upper bound decreases. For the entire human genome with 2,862,454 markers, the gap between LRTag and LRTag_lb is 1061 when the r^2 threshold is 0.5, and 142 when the r^2 threshold increases to 0.8. The small gap shows that LRTag finds near-optimal solutions for genome-wide tagging.

ACKNOWLEDGEMENT

The research was supported in part by NSF grant CCR-0309902, NIH grant LM008991-01, NSFC grant 60528001, NSF CAREER IIS-0447773, NSF DBI-0321756 and a Changjiang Visiting Professorship at Tsinghua University. Our programs (GreedyTag and LRTag) are available upon request.

References

1. H. Avi-Itzhak *et al.*. Selection of Minimum Subsets of Single Nucleotide Polymorphisms to Capture Haplotype Block Diversity. *Proc. Pac. Symp. Biocomput.*, 466-477, 2003.
2. P. Bakker *et al.*. Transferability of tag SNPs in Genetic AssociationStudies in Multiple Populations *Nat. Genet.*, 38:1298-1303, 2006.
3. E. Balas and M. C. Carrera A Dynamic Subgradient-Based Branch-And-Bound Procedure for Set Covering. Operations Research, 44:875-890, 1996.
4. R. Bar-Yehuda and S. Moran. On Approximation Problems related to the Independent Set and Vertex Cover Problems. Disc. Appl. Math., 9:1-10, 1984.
5. C.S. Carlson *et al.*. Selecting a Maximally Informative Set of Singlenucleotide Polymorphisms for Association Analyses using Linkage Disequilibrium. *Am. J. Hum. Genet.*, 74:106-20, 2004.
6. D.F.Conrad *et al.* A Worldwide Survey of Haplotype Variation and Linkage Disequilibrium in the Human Genome *Nature Genet.* 38:1251-1260, 2006.
7. K. Ding *et al.* The Effect of Haplotype-block Definitions on Inference of Haplotype-block Structure and htSNPs Selection. *Mol. Biol. Evol.*, 22: 148-159, 2005.
8. D. A. Hinds *et al.*. Whole-Genome Patterns of Common DNA Variation in Three Human Populations *Science*, 307:1072-1079, 2005.
9. International HapMap Consortium. *Nature* 437, 1299C1320 2005.
10. HapMap LD data. `http://www.hapmap.org/downloads/ld_data/2005-10/`
11. The International SNPWorking Group. A Map of Human Genome Sequence Variation Containing 1.42 Million Single Nucleotide Polymorphisms. *Nature*, 409: 928-933, 2001.
12. S.B. Gabriel *et al.*. The Structure of Haplotype Blocks in the Human Genome. *Science*, 296:2225-2229, 2002.
13. B.N.Howie *et al.* Efficient Selection of Tagging Single-Nucleotide Polymorphisms in Multiple Populations *Hum. Genet.*, 120:58-68, 2006.
14. Whole-Genome Patterns of Common DNA Variation in Three Human Populations *Science*, 307:1072-1079, 2005.
15. E. Halperin, G. Kimmel, and R. Shamir. Tag SNP Selection in Genotype Data for Maximizing SNP Prediction Accuracy. *Bioinformatics*, 21(Suppl 1):i195-i203, 2005.
16. J. Hampe, S. Schreiber and M. Krawczak. Entropy-based SNP Selection for Genetic Association Studies. *Hum Genet.* 114(1):36-43, 2003.
17. G.C. Johnson *et al.*. Haplotype Tagging for the Identification of Common Disease Genes. *Nat. Genet.*, 29:233-237, 2001.
18. L. Kruglyak and D. Nickerson. Variation is the Spice of Life. *Nat. Genet.*, 27:234-236, 2001.
19. X. Ke and L.R. Cardon. Efficient Selective Screening of Haplotype Tag SNPs. *Bioinformatics*, 19:287-288, 2003.
20. Z. Lin and R.B. Altman. Finding Haplotype Tagging SNPs by Use of Principal Components Analysis. *Am. J. Hum. Genet.*, 75:850-861, 2004.
21. Z. Liu, S. Lin and M. Tan. Genome-Wide Tagging SNPs with Entropy-Based Monte Carlo Method. *J. Comput. Biol.*, 13(9):1606-1614, 2006.
22. R. Magi, L. Kaplinski and M. Remm The Whole Genome TagSNP Selection and Transferability Among HapMap Populations *Pacific Symposium on Biocomputing*, 11:535-543, 2006.
23. A. C. Need and D.B. Goldstein Genome-wide Tagging for Everyone *Nat. Genet.*, 38:1227-1228, 2006.
24. N. Patil *et al.*. Blocks of Limited Haplotype Diversity Revealed by High-resolution Scanning of Human Chromosome 21. *Science*, 294:1719-1723, 2001.
25. T. M. Phuong, Z. Lin and R. B. Altman. Choosing SNPs Using Feature Selection *Proc. Computaional Systems Bioinformatics Conference(CSB)*, 301-309, 2005.
26. Z. S. Qin, S. Gopalakrishnan and G. R. Abecasis. An Efficient Comprehensive Search Algorithm for TagSNP Selection using Linkage Disequilibrium Criteria *Bioinformatics*, 22: 220-225, 2006.
27. D.O. Stram *et al.*. Choosing Haplotypetagging SNPs based on Unphased Genotype Data using a Preliminary Sample of Unrelated Subjects with an Example from the Multiethnic Cohort Study. *Hum. Hered.*, 55:27C36, 2003.
28. P. Sebastiani *et al.*. Minimal Haplotype Tagging. *Proc. Natl. Acad. Sci. USA*, 100:9900-9905, 2003.
29. S. L. Sawyer, et al. Linkage Disequilibrium Patterns Vary Substantially among Populations. *Eur J Hum Genet*, 13:677-686, 2005.
30. V. V. Vazirani. Approximation Algorithms, 2003.

31. N. Wang *et al.* . Distribution of Recombination Crossovers and the Origin of Haplotype Blocks: The Interplay of Population History, Recombination, and Mutation. *Am. J. Hum. Genet.* 71: 1227-1234,2002.
32. K. Zhang *et al.*. A Dynamic Programming Algorithm for Haplotype Partitioning. *Proc. Natl. Acad. Sci. USA*, 99:7335-7339, 2002.
33. K. Zhang and L. Jin. HaploBlockFinder: Haplotype Block Analyses. *Bioinformatics* 19:1300-1301, 2003.
34. K. Zhang *et al.*. HapBlock: Haplo-type Block Partitioning and Tag SNP Selection Software Using a Set of Dynamic Programming Algorithms. *Bioinformatics* 21:131-134, 2005.
35. E. Zeggini *et al.*. Characterisation of the Genomic Architecture of Human Chromosome 17q and Evaluation of Different Methods for Haplotype Block Definition. *BMC Genet.* 6: 21, 2005.

APPENDIX

Algorithm 5.1 (GreedyTag: Greedy Algorithm for TagSNP Selection in Multiple Populations)

Input: A set V of biallelic SNP markers and their pairwise r^2 LD statistics in K distinct populations. A pre-defined threshold γ_0 for r^2 LD statistics.
Output: A feasible tagSNP set $T \subseteq V$, and a lower bound LB.
Begin

```
  Partition markers into precincts. Let the set of precincts be $\mathcal{P}$.
  For each precinct $p \in \mathcal{P}$ $\{$the following will be executed in parallel on a multi-processor machine$\}$
    Let $U$ be the set of SNPs and $W$ the set of marker occurrences in $p$.

    Step 1: Apply the three data reduction rules.
    $T_p \Leftarrow \emptyset$; $LB_p \Leftarrow 0$; UPDATED $\Leftarrow$ true;
    While UPDATED $\{$ execute the optimal rules iteratively$\}$
      UPDATED $\Leftarrow$ false;
      If $\exists$ an irreplaceable marker $v_j \in U$ $\{$Rule 1$\}$
        $U \Leftarrow U - \{v_j\}$;
        $W \Leftarrow W - N^*(v_j)$; $\{N^*(v_j)$ is defined in Equation (1) $\}$
        $T_p \Leftarrow T_p \cup \{v_j\}$;   $LB_p \Leftarrow LB_p + 1$;
        UPDATED $\Leftarrow$ true;
      If $\exists$ a less informative marker $v_j \in U$ $\{$Rule 2$\}$
        $U \Leftarrow U - \{v_j\}$;   UPDATED $\Leftarrow$ true;
      If $\exists$ a less stringent occurrence $v_j^i \in W$ $\{$Rule 3$\}$
        $W \Leftarrow W - \{v_j^i\}$; UPDATED $\Leftarrow$ true;
    For each $v_j \in U$
      $D(v_j) \Leftarrow N^*(v_j) \cap W$;

    Step 2: Select tagSNPs greedily.
    While $W$ is non-empty $\{$there are markers to be tagged$\}$
      Let $v_{j_0} \Leftarrow argmax_{v_j \in U} |D(v_j)|$;
      $T_p \Leftarrow T_p \cup \{v_{j_0}\}$; $U \Leftarrow U - \{v_{j_0}\}$;
      $W \Leftarrow W - N^*(v_{j_0})$;
      For each $v_j \in U$
        $D(v_j) \Leftarrow D(v_j) \cap W$;

  $T \Leftarrow \bigcup_{p \in \mathcal{P}} T_p$;   $LB \Leftarrow \sum_{p \in \mathcal{P}} LB_p$
  Output $T$ , $LB$ $\{$output the solution and lower bound$\}$
End
```

Algorithm 5.2 (LRTag: Lagrangian relaxation Algorithm for TagSNP Selection in Multiple Populations)

Input: A set V of n biallelic SNP markers and their pairwise r^2 LD statistics in K distinct populations. A pre-defined threshold γ_0 for r^2 LD statistics. A pre-defined initial scalar α_0 and threshold α_{min} for subgradient optimization. A pre-defined maximum number $Iter_{max}$ of iterations and a pre-defined threshold K_{max} of maximum trials. **Output**: A feasible tagSNP set $T \subseteq V$, and a lower bound LB.
Begin

```
  Partition markers into precincts. Let the set of precincts be $\mathcal{P}$

  For each precinct $p \in \mathcal{P}$ $\{$the following will be executed in parallel$\}$
    Let $U$ be the SNP set and $W$ be the marker occurrences set in $p$.

    Step 1: Apply the three data reduction rules and obtain a temporary tagSNP set $T_p$ and a lower bound $LB_p$.
    $\{$The same as the rules in algorithm 5.1)$\}$.

    Step 2: Select tagSNPs under a LR framework.
    Generate Lagrangian relaxation formula as in Equation (4);
    $k \Leftarrow 0$; $\alpha \Leftarrow \alpha_0$; $Iter \Leftarrow 0$;
    Initialize $\boldsymbol{\lambda}$ being an arbitrary non-negative vector;
    $LB_{p1} \Leftarrow 0$; $T_{p1} \Leftarrow U$;

    While $(\alpha > \alpha_{min})$ and $(Iter < Iter_{max})$
      $Iter \Leftarrow Iter + 1$; $new\_LB \Leftarrow \sum_{1 \le i \le K, 1 \le j \le n} \lambda_{i,j}$;
      $new\_T \Leftarrow \emptyset$;

      $\{$Calculate a new lower bound $new\_LB$$\}$
      For each $v_j \in U$
       $s_j \Leftarrow 1 - \sum_{v_{j'}^i \in N^*(v_j)} \lambda_{i,j'}$; $\{N^*(v_j)$ is given in Equation(1)$\}$
        If $s_j \le 0$   $t_j \Leftarrow 1$;   Else   $t_j \Leftarrow 0$;
        $new\_LB \Leftarrow new\_LB + s_j \cdot t_j$;

      $\{$Obtain a feasible tagSNP set $new\_T$ by the RCH method [3]$\}$
      For each $v_j \in U$      $RCH\_s_j \Leftarrow s_j$;
      For each $v_j^i \in W$     $RCH\_\lambda_{i,j} \Leftarrow \lambda_{i,j}$;
      For each $v_j^i \in W$
        If $\sum_{v_{j'} \in C(v_j^i)} t_{j'} < 1$   $\{C(v_j^i)$ is defined in Equation (2) $\}$
          $RCH\_s_m \Leftarrow min\{RCH\_s_{j'} : v_{j'} \in C(v_j^i)\}$;
          $RCH\_\lambda_{i,j} \Leftarrow RCH\_\lambda_{i,j} + RCH\_s_m$;
          For each $v_{j'} \in C(v_j^i)$
            $RCH\_s_{j'} \Leftarrow RCH\_s_{j'} - RCH\_s_m$;
            If $RCH\_s_{j'} \le 0$   $t_{j'} \Leftarrow 1$;
      For each $v_j \in U$
        If $t_j = 1$   $new\_T \Leftarrow new\_T \cup \{v_j\}$;

      $\{$Update the lower bound $LB_{p1}$ and the tagSNP set $T_{p1}$ $\}$
      If $new\_LB \le LB_{p1}$   $k \Leftarrow k + 1$;
        If $(k \ge K_{max})$   $\alpha \Leftarrow \alpha/2$; $k \Leftarrow 0$;
      Else   $LB_{p1} \Leftarrow new\_LB$; $k \Leftarrow 0$;
      If $|new\_T| < |T_{p1}|$   $T_{p1} \Leftarrow new\_T$;

      $\{$Update the Lagrangian multipliers $\boldsymbol{\lambda}$ by the subgradient optimization method $\}$
      For each $v_j^i \in W$   $\nabla\lambda_{i,j} \Leftarrow 1 - \sum_{v_{j'} \in C(v_j^i)} t_{j'}$;

      $\boldsymbol{\lambda} \Leftarrow max\{\mathbf{0}, \boldsymbol{\lambda} + \alpha \frac{|T_{p1}| - LB_{p1}}{||\nabla\boldsymbol{\lambda}||^2} \nabla\boldsymbol{\lambda}\}$;
    $\{$Combine the solutions from step 1 and step 2$\}$
    $T_p \Leftarrow T_p \cup T_{p1}$; $LB_p \Leftarrow LB_p + LB_{p1}$;

  $T \Leftarrow \bigcup_{p \in \mathcal{P}} T_p$; $LB \Leftarrow \sum_{p \in \mathcal{P}} LB_p$
  Output $T$, $LB$ $\{$output the solution and the lower bound$\}$
End
```

Table 5. Summary of the tag SNPs selected by LD-Select, GreedyTag and LRTag for a single population, CEU, on each human chromosome.

Chromosome	1	2	3	4	5	6	7	8	9	10	11
# SNP	151195	181499	143472	130823	138817	149514	113037	122646	100352	110942	104661
$r^2 \geq 0.5$											
# precinct	15752	29426	12901	11906	11998	11831	10512	9900	9438	10153	9979
# tagSNP (upper bound)											
LD-Select*	21865	36238	19063	17212	17765	17921	15418	15140	13800	14882	14307
GreedyTag	20806	35083	17984	16295	16769	16815	14584	14203	13066	14041	13600
LRTag	20800	35065	17977	16286	16756	16798	14577	14196	13058	14038	13589
# tagSNP (lower bound)											
LRTag_lb	20793	35059	17958	16279	16736	16784	14569	14182	13049	14031	13578
GreedyTag_lb	20123	34202	17155	15675	15965	16086	14021	13568	12530	13477	13089
Gap	7	6	19	7	20	14	8	14	9	7	11
$r^2 \geq 0.8$											
# precinct	35990	51098	31916	28650	29931	30632	26181	26120	23739	25186	23544
# tagSNP (upper bound)											
LD-Select*	38944	54612	35092	31590	32978	33723	28754	28822	26008	27698	25826
GreedyTag	38534	54081	34602	31124	32502	33229	28362	28394	25666	27302	25485
LRTag	38534	54080	34601	31123	32501	33227	28362	28393	25665	27302	25484
# tagSNP (lower bound)											
LRTag_lb	38534	54080	34600	31123	32501	33225	28361	28391	25664	27301	25483
GreedyTag_lb	38269	53687	34310	30824	32189	32962	28083	28110	25396	27077	25276
Gap	0	0	1	0	0	2	1	2	1	1	1
Chromosome	12	13	14	15	16	17	18	19	20	21	22
# SNP	100437	84184	68485	58491	57083	47505	62666	29341	51206	27955	26996
$r^2 \geq 0.5$											
# precinct	9960	7476	6751	6740	7184	6764	6534	5291	5874	3270	3829
# tagSNP (upper bound)											
LD-Select*	14243	10996	9703	9364	9962	8656	9115	6464	7972	4470	5029
GreedyTag	13554	10374	9215	8930	9503	8355	8652	6286	7637	4258	4831
LRTag	13548	10370	9212	8923	9500	8354	8649	6284	7634	4257	4831
# tagSNP (lower bound)											
LRTag_lb	13539	10363	9203	8920	9500	8353	8646	6283	7630	4256	4830
GreedyTag_lb	13048	9988	8867	8589	9241	8180	8332	6190	7380	4125	4714
gap	9	7	9	3	0	1	3	1	4	1	1
$r^2 \geq 0.8$											
# tagSNP (upper bound)											
# precinct	23809	18509	16391	15629	16869	13942	15262	10019	13177	7390	8240
LD-Select*	25887	20221	17723	16908	18194	14778	16498	10494	14194	7912	8727
GreedyTag	25579	19967	17546	16722	18012	14670	16299	10420	14052	7844	8652
LRTag	25579	19967	17545	16722	18012	14669	16299	10420	14052	7844	8652
# tagSNP (lower bound)											
LRTag_lb	25578	19966	17545	16722	18012	14668	16299	10420	14051	7844	8652
GreedyTag_lb	25387	19778	17405	16608	17836	14588	16181	10382	13943	7774	8601
Gap	1	1	0	0	0	1	0	0	1	0	0

Table 6. The speeds of GreedyTag and LRTag for tagging the human genome for a single population, CEU, with the r^2 threshold being 0.5. The running time is evaluated on a 32-processor SGI Altix 4700 supercomputer system.

Chromosome	1	2	3	4	5	6	7	8	9	10	11
LRTag	1m18s	1m44s	1m28s	1m12s	1m27s	3m7s	1m3s	1m15s	57s	1m6s	1m8s
GreedyTag	1m17s	1m41s	1m16s	1m15s	1m24s	3m11s	58s	1m16s	57s	1m6s	1m10s
Chromosome	12	13	14	15	16	17	18	19	20	21	22
LRTag	56s	50s	34s	28s	23s	46s	31s	9s	23s	11s	10s
GreedyTag	56s	50s	37s	27s	20s	47s	31s	10s	22s	11s	9s

Table 7. The speeds of GreedyTag and LRTag for tagging the entire human genome for all HapMap populations with the r^2 threshold being 0.5. The running time is evaluated on a 32-processor SGI Altix 4700 supercomputer system.

Chromosome	1	2	3	4	5	6	7	8	9	10	11
LRTag	3m4s	2m2s	3m9s	2m51s	3m37s	11m4s	2m12s	3m45s	2m24s	2m49s	2m20s
GreedyTag	3m11s	1m13s	3m43s	2m46s	3m20s	10m45s	2m25s	2m52s	2m18s	2m55s	2m16s
Chromosome	12	13	14	15	16	17	18	19	20	21	22
LRTag	2m55s	2m11s	1m28s	1m	48s	1m10s	1m17s	27s	56s	25s	30s
GreedyTag	3m	2m27s	1m16s	52s	23s	1m9s	1m16s	27s	50s	25s	30s

Table 8. Summary of the tagSNPs selected by MultiPop-TagSelect, GreedyTag and LRTag for all HapMap populations on each human chromosome.

Chromosome	1	2	3	4	5	6	7	8	9	10	11
# SNP	216357	249136	196535	182273	187924	205496	155224	170136	138047	156089	144083
$r^2 \geq 0.5$											
# precinct	16234	26836	12835	12251	12414	11862	10332	10101	9254	10220	9568
# tagSNP (upper bound)											
MultiPop-TagSelect*	64892	126408	56978	52828	54087	54454	45943	46927	41341	45226	41556
GreedyTag	59126	122372	51266	47650	48661	48817	41169	42206	37289	40713	37365
LRTag	55016	117537	47450	44223	45186	44987	38150	39149	34439	37554	34590
# tagSNP (lower bound)											
LRTag_lb	54942	117511	47362	44141	45102	44878	38090	39076	34381	37486	34537
GreedyTag_lb	53937	117155	46330	43239	44145	43845	37280	38161	33534	36713	33778
Gap	74	26	88	82	84	109	60	73	58	68	53
$r^2 \geq 0.8$											
# precinct	42450	56135	35192	33434	33211	33228	28543	28948	25485	28277	25428
# tagSNP (upper bound)											
MultiPop-TagSelect*	100062	155505	89195	82835	84998	86313	72024	74934	65442	70817	64679
GreedyTag	94797	150664	84091	78077	80188	80981	67818	70678	61708	66676	60721
LRTag	94797	150664	84090	78076	80186	80980	67817	70677	61706	66674	60718
# tagSNP (lower bound)											
LRTag_lb	94788	150660	84079	78072	80174	80964	67808	70667	61699	66663	60705
GreedyTag_lb	94362	150393	83585	77674	79663	80507	67461	70285	61321	66291	60294
Gap	9	4	11	4	12	16	9	10	7	11	13

Chromosome	12	13	14	15	16	17	18	19	20	21	22
# SNP	141943	119080	94528	81687	79898	64645	89024	40549	70877	39400	39523
$r^2 \geq 0.5$											
# precinct	10086	7810	6532	6667	7328	6952	6875	5127	6139	3290	3884
# tagSNP (upper bound)											
MultiPop-TagSelect*	42362	33477	28465	27847	28987	23601	27109	15768	23243	13100	13895
GreedyTag	38563	30183	25706	25408	26432	21931	24789	14785	21319	12010	12980
LRTag	35493	27927	23932	23721	24791	20647	23174	14007	19994	11253	12174
# tagSNP (lower bound)											
LRTag_lb	35449	27881	23903	23686	24761	20636	23141	14001	19971	11238	12160
GreedyTag_lb	34833	27306	23440	23229	24294	20366	22697	13814	19601	11035	12017
Gap	44	46	29	35	30	11	33	6	23	15	14
$r^2 \geq 0.8$											
# precinct	27027	21084	17723	17526	18943	16278	17866	11289	15438	8366	9480
# tagSNP (upper bound)											
MultiPop-TagSelect*	65521	52863	44226	42380	43913	34289	41893	22274	35251	19990	20624
GreedyTag_lb	61828	49797	41867	40250	41726	32862	39833	21465	33676	19060	19742
LRTag_lb	61826	49796	41867	40250	41724	32862	39833	21464	33675	19060	19741
# tagSNP (lower bound)											
LRTag_lb	61816	49791	41860	40247	41721	32860	39832	21464	33673	19059	19739
GreedyTag_lb	61450	49498	41642	40029	41497	32740	39625	21377	33525	18996	19660
Gap	10	5	7	3	3	2	1	0	2	1	2

Table 9. The speeds of GreedyTag and LRTag for tagging the entire ENCODE region for all HapMap populations with the r^2 threshold being 0.5. The running time is evaluated on a 32-processor SGI Altix 4700 supercomputer system.

Region	ENm010	ENm013	ENm014	ENr112	ENr113	ENr123	ENr131	ENr213	ENr232	ENr321
LRTag	1s	4s	3s	5s	7s	5s	1s	1s	1s	2s
GreedyTag	1s	4s	4s	5s	7s	6s	1s	1s	1s	2s