Identifying interacting genetic variations by fish-swarm logic regression

Xuanping Zhang
Xi'an Jiaotong University

Jiayin Wang
Washington University School of Medicine in St. Louis

Aiyuan Yang
Xi'an Jiaotong University

Chunxia Yan
Xi'an Jiaotong University

Feng Zhu
Xi'an Jiaotong University

See next page for additional authors

Follow this and additional works at: https://digitalcommons.wustl.edu/open_access_pubs

Recommended Citation
https://digitalcommons.wustl.edu/open_access_pubs/1621
Research Article

Identifying Interacting Genetic Variations by Fish-Swarm Logic Regression

Xuanping Zhang,1 Jiayin Wang,2 Aiyuan Yang,1 Chunxia Yan,3,4,5 Feng Zhu,6 Zhongmeng Zhao,1 and Zhi Cao1

1 Department of Computer Science and Technology, Xi’an Jiaotong University, Xi’an, Shaanxi 710049, China
2 The Genome Institute, Washington University, St. Louis, MO 63108, USA
3 College of Medicine and Forensics, Xi’an Jiaotong University Health Science Center, Xi’an, Shaanxi 710061, China
4 Key Laboratory of the Ministry of Health for Forensic Sciences, Xi’an Jiaotong University, Xi’an, Shaanxi 710061, China
5 Key Laboratory of the Ministry of Education for Environment and Genes Related to Diseases, Xi’an Jiaotong University, Xi’an, Shaanxi 710061, China
6 Center for Translational Medicine, The First Affiliated Hospital of Xi’an Jiaotong University College of Medicine, Xi’an, Shaanxi 710061, China

Correspondence should be addressed to Jiayin Wang; jiw09003@gmail.com

Received 2 April 2013; Revised 8 June 2013; Accepted 2 July 2013

Academic Editor: Eugénio Ferreira

Copyright © 2013 Xuanping Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Understanding associations between genotypes and complex traits is a fundamental problem in human genetics. A major open problem in mapping phenotypes is that of identifying a set of interacting genetic variants, which might contribute to complex traits. Logic regression (LR) is a powerful multivariant association tool. Several LR-based approaches have been successfully applied to different datasets. However, these approaches are not adequate with regard to accuracy and efficiency. In this paper, we propose a new LR-based approach, called fish-swarm logic regression (FSLR), which improves the logic regression process by incorporating swarm optimization. In our approach, a school of fish agents are conducted in parallel. Each fish agent holds a regression model, while the school searches for better models through various preset behaviors. A swarm algorithm improves the accuracy and the efficiency by speeding up the convergence and preventing it from dropping into local optimums. We apply our approach on a real screening dataset and a series of simulation scenarios. Compared to three existing LR-based approaches, our approach outperforms them by having lower type I and type II error rates, being able to identify more preset causal sites, and performing at faster speeds.

1. Introduction

Understanding the genotype-phenotype association is one of the major problems in human genetics. Much effort has been devoted to mapping complex traits with one or pairwise single nucleotide polymorphisms (SNPs). These studies were mainly supported by the “common disease-common variant (CDCV)” hypothesis [1, 2], which suggests that complex diseases can be largely attributable to a moderate number of common variants, each of which explains partial risk among a population [3]. According to the CDCV hypothesis, the genetic cause is considered to be either a large number of small-effect common variants across the entire allele frequency spectrum, which is also called “the infinitesimal genetic model” [4], or some combination of genotypic, environmental, and epigenetic interactions, known as “the broad sense heritability model” [5].

In the broad sense heritability model, there is a focus on two types of interactions in the quantitative research, which are the genotype-by-genotype interactions, also known as epistasis, and genotype-by-environment interactions. The genotype-by-genotype interactions consider that the effect of one genetic variation is conditional on genotypes at one or more other unlinked loci, while the genotype-by-environment interactions consider that the effect of one genetic variation is conditional on environmental factors, such as behaviors and temperature [3].
Along with the growing evidence of genotype-by-genotype interactions being important contributors to genetic variations in complex human diseases, there are many different formulations in modeling both types of interactions [6]. Some heuristic, learning-based, and Bayesian-based approaches are also proposed, especially for addressing nonlinear interactions and high-dimensional data. This type of approach, to our knowledge, includes machine learning approaches (e.g., neural networks [7, 8]), data-mining approaches (e.g., pattern mining [9]), and regression-based approaches (e.g., classification and regression trees (CARTs), pattern-based logistic regression [10], and logic regression). Logic regression (LR) is one of the approaches for finding multiple interactions and has been successfully applied on several datasets [6, 11–17]. A series of logic regression-based approaches have been developed and are reported to outperform other existing approaches [12–15, 17].

Genetic studies now generate SNP data with thousands or millions of variants from more than ten thousand sampled individuals. A main deficiency of existing LR-based approaches is that these approaches are not efficient enough to handle large-scale data. These approaches often suffer from slow convergence when finding the optimal solutions in a very large solution space. Because of the design of the logic tree (LT, the basic computational unit in logic regression), the size of the solution space of the logic trees increases factorially when the number of SNPs becomes larger. A way of speeding up the logic regression is to design a better regression algorithm. The greedy strategy [11], the simulated annealing algorithm [12, 13], and the bootstrap strategy [14, 15] have been successfully applied in different scenarios.

Motivated by previous studies, in this paper, a novel regression algorithm on the logic regression framework is described. This new algorithm incorporates fish-swarm optimization [18], which is a widely used particle swarm algorithm that is based on swarm intelligence. The basic idea of the fish-swarm algorithm is to introduce a school of fish, which are implemented by threads in the computation. Each fish (fish agent) holds a logic tree and explores the solution space according to a set of preset individual behaviors and swarm behaviors. To speed up the convergence of the search process, the new algorithm also improves the behaviors by introducing selection probability distributions. These probability distributions lead to the selection of more suitable behaviors in the search.

### 2. Background and Related Studies

#### 2.1. Basic Logic Regression Model

Logic regression (LR), which was first proposed in [11], attempts to identify a set of Boolean combinations (interactions) among candidate variables (SNPs) for the prediction of a case-control phenotype. A Boolean combination involves interacting SNPs and logic interactions among them. The basic logic regression [11] attempts to find a single Boolean expression that best “explains” the given genotypes behind the observed phenotype. One “explanation” means that the phenotype value predicted by the Boolean expression on a genotype is the same as that of the corresponding phenotype of this genotype.

The expression with the highest number of explanations is the output of the regression process. The number of explanations of an expression is also called the score of this expression.

Because of the combinatorial explosion of potential Boolean combinations, the logic tree (LT) model is suggested to represent a Boolean expression, where each leaf of a logic tree corresponds to a SNP site, while the internal nodes are associated with logical operators (e.g., AND or OR). A greedy strategy and a simulated annealing algorithm are designed separately to search for a better logic tree that fits the given genotype-phenotype dataset better. Note that every Boolean expression can be represented as a logic tree; see Figure 1. The logic regression is considered to be an exact approach in association studies.

The process of seeking a better logic tree is operated by changing the components or modifying the topology of the current logic tree. In basic logic regression approaches, three tree operations are suggested: add, delete, and change; see Figure 1. The add operation is to add a SNP (or its negation) with a specific operator (AND or OR) to the current logic tree. The delete operation removes a SNP and its parent internal node, the operator, from the current LT. The change operation updates the current LT by changing a SNP (or operator) to a different SNP (resp., operator). Without loss of generality, considering a set of LTs that together influence the phenotype, the basic model of the logic regression is as follows [11]:

\[
g(E(P)) = \beta_0 + \sum_{i=1}^{k} \beta_i L_i,
\]

where each \( L_i \) represents a logic tree among the set of \( k \) logic trees; these LTs jointly affect the case-control phenotype \( P \in \{0, 1\} \). The total number of SNPs involved in all of the logic trees is defined as the logic tree model size \( s \). Here, \( \beta \) is a series of unknown regression parameters, each for a specific logic tree. \( E(P) \in [0, 1] \) is the expectation of the phenotype. An invertible link function, \( g \), is introduced to map the real-valued predictor \( E(P) \) onto the range \([0, 1]\). According to the existing logic regression-based approaches [11–13], we also adopt the same “sigmoid” function: \( g(\eta) = 1/(1 + \exp(-\eta)) \).

However, many Boolean expressions can fit equally or almost equally well, and there are no universal algorithms to reduce the Boolean expressions. Furthermore, the best Boolean expression may be an overfitted expression rather than the true one. This situation occurs more frequently due to noisy data [12]. To overcome these weaknesses, a Monte Carlo logic regression (MCLR) approach is designed [12], which partly incorporates Bayesian model selection techniques and reports a group of plausible Boolean expressions for further investigation. Different statistical tests are applied on the Boolean expressions, and different interesting features are identified. In the MCLR approach, the model size and the number of models with the same size are treated as random variables following a geometric prior distribution. The prior for a model \( M \) of size \( s \) is given by

\[
P(M) = P(s)P(M|s) = \theta (1 - \alpha) \times \alpha^s \times \frac{1}{s_N},
\]

where \( s_N \) is the number of models with size \( s \). The parameter \( \alpha \in (0, 1) \) in the prior is used to penalize large models.
The larger a best model is, the more likely it can be achieved by chance. It is reported that the penalty parameter $\alpha$ somehow controls the overfitting issue [12]. However, $\alpha$ and another model parameter, the maximum number of logic trees $k$, must be predetermined in MCLR. In practice, these two parameters are often suggested by experts on the applications. In another logic regression-based approach, full Bayesian parameters are oftensuggested by expertsonthe applications. In practice, these two model parameter, the maximum number of logic trees controls the overfitting issue [12]. However, $\alpha$ and another model parameter, the maximum number of logic trees $k$, must be predetermined in MCLR. In practice, these two parameters are often suggested by experts on the applications.

As an optimization framework, different behaviors may be considered and implemented for different problems and solution spaces. Overall, AFSA is suggested as one of the best swarm intelligence optimization methods due to its high convergence speed, flexibility, fault tolerance, and many other advantages [20]. For more details on fish-swarm optimization, Neshat and others wrote a comprehensive review in [20].

2.3. Problem Statement. Suppose that we are given a set of $M$ sampled individuals. We use a binary vector $P$ to represent the phenotype of these $M$ individuals. For any individual $i$, $P_i = 1$ if $i$ is a case (affected by the phenotype) and $P_i = 0$ if $i$ is a control (not affected by the phenotype). Let $h_i$ represent a SNP genotype of individual $i$ with a set of $N$ sites. When we consider the "recessive-set" genetic model, we assume that this SNP genotype is a binary vector that shows the allele types at all sites (which are assumed to contain two alleles each). For the "recessive" genetic model, the logic regression framework can also handle diploid genotypes by encoding each site into a 2-bit binary variable, which is the same as all existing LR-based approaches [11–13]. Thus, we have a binary matrix with $M$ rows (each for one of the $M$ haplotypes) and $N + 1$ columns, where the first $N$ columns correspond to the $N$ SNPs and the last column corresponds to the phenotype. Let $S_j$ denote the $j$th SNP. $s_{ij}$ denotes the allelic value of individual $i$ at site $j$. We assume that $s_{ij} = 0$ if the SNP $j$ of individual $i$ shows a wild-type allele, and $s_{ij} = 1$ if the SNP $j$ of individual $i$ shows a mutation. The goal of our problem is to find the interaction(s) among the SNPs (a subset of the given SNPs) that might explain the phenotype better.

![Figure 1: Logic tree representation of $X_1 \land X_2 \land (\neg X_3 \lor X_4)$ and three permissible moves for logic trees. Starting tree, $T_0$, is at top left.](image-url)
3. Methods

Our new method is the fish-swarm logic regression algorithm. The main motivation for developing this algorithm is to conduct a more efficient and accurate regression process and to extend the algorithm to a parallel framework. To perform initializations, we first generate \( F \) fish (fish agents) and \( F \) initial logic trees. Each fish agent holds a logic tree, separately. The details of initialization are described in Section 3.1. This school of fish agents is seeking better solutions iteratively. In each iteration, fish agents communicate with each other to identify the best (or one of the best) logic tree(s) among all of the logic trees that are held by the school, according to the scores. Then, each fish agent chooses one of the present behaviors, which are described in Section 3.2, by comparing the current logic tree with the best logic tree. After updating the current logic tree, the new logic tree is accepted to replace the current one by criteria with different conditions. In contrast to most of the existing logic regression-based approaches, the total number of iterations is no longer preset, and our algorithm will terminate when the best logic tree converges.

Our approach takes advantage of swarm optimizations. By incorporating a swarm framework, the algorithm searches the solution space from multiple start points (different logic trees) instead of continuing to apply modifications on one logic tree. Thus, it is obvious that we have a higher probability of converging into local optimum(s) or global optimum(s), and this framework speeds up the previous search process. In particular, we use the “fish agent” framework rather than other swarm intelligence frameworks because of the high similarity between the mechanism of a fish swarm and the genotype-phenotype association problem. In a natural scenario with a school of fish, a fish forages independently in a small space around it, while it also might follow other fish that could lead to a space that has more food. However, each fish always keeps a distance from the other fish, to control the school density. This arrangement is one of the major differences of the fish swarm from the other swarm algorithms. Intuitively, we would like to prevent logic trees from gathering together, because if they do so, then the algorithm might actually perform similar to the algorithm that has only one logic tree performing, and it could fall into a local optimum rapidly. Moreover, as mentioned before, selecting only the best logic tree is not sufficient; the mechanism of the fish swarm fits well with the problem and the requirements.

3.1. Logic Tree Space. The fish-swarm algorithm models the natural environment and animal behaviors; however, we cannot blindly or mechanically copy this framework because of two reasons: (1) the solution space that comprises the logic trees is significantly different from the 3D space (the natural environment), and (2) behaviors in the natural environment are not able to directly apply to the logic tree space.

We modify the fish agent framework to fit this specific problem. Suppose that we have generated \( F \) fish agents, and this school will explore the solution space then. The first issue of building up the framework is whether we could map the solution space to the “environment.” An associated problem is how to describe the differences between the logic trees for the communications among fish agents. In a real-world scenario, the environment is a three-dimensional space, and the space coordinates are communicated. The solution space, on the other hand, comprises all of the possible logic trees, where the logic trees could have multiple dimensions. For example, two different logic trees not only might bring a different number of SNPs but also might be built up by different topologies or different logic operators on the internal nodes. Here, we propose a practical way to map logic trees in a four-dimensional space and measure the differences among them, which is enlightened by a full Bayesian framework on logic regression in [13].

First, in the logic regression framework, there are three major unknown parameters: the number of logic trees in one regression model \( n_l \), the size of the regression model \( s \), which is equal to the number of SNPs involved in the model, and the SNPs involved in each logic tree \( q \). The priors of these parameters are written in a factorized form:

\[
p(\text{Model}) = p(n_l) \cdot p(s) \cdot \prod_{i=1}^{n_l} p(s_i) \cdot p(q_i | s_i),
\]

where \( s_i \) is the size of the logic tree \( i \) in the regression model \( 1 \leq i \leq n_l \) and vector \( q_i \) is a binary indicator for logic tree \( i \), in which \( q_{ij} = 1 \) denotes that the SNP \( j \) is selected as a leaf of logic tree \( i \). In other words, \( s_i \) is also equal to the number of 1s in \( q_i \).

We assume that the size of every logic tree has a prior distribution of \( p(s_i) = U(1, \ldots, s_{\text{max}}) \), where \( U(\cdot) \) represents a uniform distribution and \( s_{\text{max}} \) is the maximum size of a logic tree that can be preset. \( p(n_l) \) is also assigned a prior distribution of \( p(n_l) = U(1, \ldots, n) \), which indicates the lack of knowledge of the number of logic trees. In a specific case, both the prior distributions of \( p(s_i) \) and \( p(n_l) \) can be preset to incorporate more prior knowledge.

A logic tree not only makes up the SNPs but also connects with the logic operators. The logic operators describe complex interactions among the SNPs. Different SNPs could have different functions, including “causal,” “neutral,” and “protective.” The causal variants increase the risk of cases, while the protective variants decrease the risks. The neutral variants are considered to be independent of the phenotype. For the “additive” genetic model, the “AND” operators are adopted to connect the causal SNPs. For the “dominant” genetic model, the causal SNPs are connected by “OR” operators. If we split a logic tree into two sublogics (sub-LTs) at an “OR” operator, according to the genetic model, each new sub-LT affects the phenotype independently, which is the same as the original logic tree. In other words, these two new sub-LTs still contain the same information as the original logic tree. This arrangement implies that splitting the logic tree at an “OR” operator will not cause information loss. Thus, if we split a logic tree at the “OR” operators recursively, we obtain a forest (a set of logic trees) that comprises the sub-LTs with only the “AND” operators.

We highlight the split process for two reasons.

(i) The sub-LTs in the forest contain only “AND” operator(s), and thus the topologies of these sub-LTs are
no longer considerable because of the commutative law. The differences between any two sub-LTs, sub-LT \( i \) and sub-LT \( j \), can be represented by the differences between vector \( \vec{d}_i \) and \( \vec{d}_j \).

(ii) The forest represents all of the information of an original logic tree, and thus the differences between two logic trees are computable by measuring the forests that are derived from them. For example, the number of “OR” operators in a logic tree is equal to the number of sub-LTs in the forest.

In summary, we define a three-dimensional hyperspace as follows: the first dimension, the scalar \( n \), indicates the number of sub-LTs that can be derived from a logic tree; the second dimension, vector \( \vec{s} \), indicates the size of each sub-LT; and the third dimension, scalar \( Y \), indicates the score of the original logic tree. For a simpler version, the second dimension can be replaced by a scalar \( s \) of the size of the original logic tree. According to our experience, the accuracy of this simplified version is roughly the same as the original one.

3.2. Behaviors of Fish Agents. When we have the search space, fish agents conduct behaviors that search the solution space simultaneously. Thus, to define the behaviors that regulate the search strategy is another important part of a swarm algorithm. Behaviors are often dependent on the solution space that they work on. For the specific logic tree space, note that we have a collapsing solution space from the set of all possible logic trees rather than a bijective solution space. For example, one point in the logic tree space could correspond to multiple logic trees. This correspondence occurs because of the complexity of both the tree topology and the logic operators. For a bijective solution space, defining the swarm behaviors is adequate in most cases; however, in the logic tree space, the fish agent should harbor necessary behaviors itself, in addition to the swarm behaviors, to update the logic tree that it holds even when it keeps the location in the space. In this section, we will describe the behaviors for a fish agent, while in Section 3.3, we will define the behaviors of the fish swarm. For a fish agent \( f_i \), we define four fish agent behaviors that allow it to alter its current logic tree to a new logic tree. These four behaviors are the following.

1. ADD SNP: select a SNP and add it to the LT.
2. DEL SNP: select a SNP on the LT and remove the SNP from all of the sub-LTs.
3. ALT SNP: select a SNP on the LT and alter the SNP by another SNP.
4. ALT OPT: select an operator on the LT and alter the operator by the opposite operator.

The probability distribution of choosing a behavior affects the preferences of the behaviors. The simplest way is to adopt a uniform distribution; for example, each behavior has the same probability, 25% of them being chosen. However, to accelerate the convergence, it is better to reflect preferences among the behaviors. Suppose that, after one iteration, the fish agent that holds the logic tree with the highest score is announced. Let this fish agent be \( f_{\text{best}} \). Intuitively, when the \( f_{\text{best}} \) is announced, the difference in the sizes between fish agents \( f_i \) and \( f_{\text{best}} \) can be measured by \( |\vec{s}_i - \vec{s}_{\text{best}}| \). Thus, we adopt a normal distribution to obtain the chosen probability of each behavior, where the probability density function subjected to \( \vec{s} \) is

\[
f(s) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(s-s_{\text{best}})^2/2\sigma^2},
\]

where the mean of this normal distribution is set to \( s_{\text{best}} \), which is the size of \( f_{\text{best}} \), and the variance of this normal distribution is set to \( \sigma^2 = (1/m) \sum_{i=1}^{m} (s_i - s_{\text{best}})^2 \).

Furthermore, for a specific SNP, we should also consider the probability that this SNP will be chosen. We obtain the probability distribution of selecting a SNP by measuring the importance of each SNP. The measurement of importance is a statistic [14, 15]. Intuitively, an important SNP is assumed to be the SNP that occurs most frequently. However, although some SNPs either may be explanatory for a small subset of cases and controls or could be actually very important for the correct prediction of some of the phenotypes, such SNPs are considered to be unimportant [14]. Every several iterations, the algorithm updates the importance of all of the SNPs and generates a new probability distribution.

To compute the importance, each fish agent records the correctly classified out-of-bag (OOB) observations. Let \( P(x_i) \) represent the probability of SNP \( j \) being added to a fish agent \( f_i \). The importance of \( j \) is

\[
V(j) = \frac{1}{F} \sum_{k=1}^{F} \left( N_k - N_k^{(j)} \right),
\]

where \( F \) is the number of fish agents and \( N_k \) is the number of correctly classified OOB observations of fish agent \( f_k \). This measurement is much more robust than the previous measurement based on \( \beta \) distributions [17]. Note that each fish agent applies a bootstrap sampling on the given data \( D \) instead of working on the whole \( D \), because it should not be computed on the same data on which the classification rule has been trained but instead is computed on independent data that contains new observations [14].

Suppose there is an index vector \( S_{f_i} \) for fish \( f_i \) that comprises the SNPs in \( f_i \) and an index vector \( S_{f_j} \) of fish \( f_j \) that comprises the SNPs in \( f_j \). Then, the distance between \( f_i \) and \( f_j \) in the solution space can be computed by the distance between \( S_{f_i} \) and \( S_{f_j} \) as follows:

\[
d(S_{f_i}, S_{f_j}) = \left| \sum_{f_i} V(i) - \sum_{f_j} V(j) \right|.
\]

3.3. Behaviors of the Fish Swarm. Here, we continue to introduce the swarm behaviors. The behaviors of fish \( f_i \) are listed as follows.

1. HOLD. If \( f_i \) is \( f_{\text{best}} \), then only “HOLD” behavior is allowed. At this time, \( f_{\text{best}} \) does nothing but retains the current best
Figure 2: “FOLLOW” behavior is illustrated. When \( s_i > s_{\text{best}} \) (\( s_{\text{best}} \) is equal to 4, and \( s_i \) is equal to 6; shown at (a)), the probability of “DEL” operations (shown as the green shadow) is larger than the probability of “ALT” operations (shown as the purple shadow). When \( s_i < s_{\text{best}} \) (\( s_{\text{best}} \) is equal to 6, and \( s_i \) is equal to 3; shown at (d)), the probability of “ADD” operations (shown as the green shadow) is larger than the probability of “ALT” operations (shown as the purple shadow).

logic tree. Otherwise, \( f_i \) will select one of the following three behaviors.

(2) RANDOM. If \( f_i \) selects “RANDOM” behavior, it searches the space randomly. All of the operations are selected with the same probability 1/4.

(3) FOLLOW. If \( f_i \) selects “FOLLOW” behavior, then it will follow the \( f_{\text{best}} \), which indicates that \( f_i \) will attempt to

(i) have the same size as \( f_{\text{best}} \) or
(ii) have the same SNPs as \( f_{\text{best}} \).

To achieve this goal, \( f_i \) checks the following \( s \) value.

(i) If \( s_i > s_{\text{best}} \), then \( f_i \) holds more SNPs than \( f_{\text{best}} \). We force that \( f_i \) can only choose “DEL” operation or two “ALT” operations. \( f_i \) may select “DEL” operations with probability \( f(s_i) \) (delete a SNP \( j \) with probability \( 1 - P_i(j) \)) and “ALT” operations with probability \( 1 - f(s_i) \) (select a SNP \( j \) with probability \( 1 - P_i(j) \) and replace it by another SNP \( s'_{\text{best}} \) selected with probability \( 1 - P_i(j') \)); see Figure 2.

(ii) If \( s_i < s_{\text{best}} \), then \( f_i \) holds fewer SNPs than \( f_{\text{best}} \). We force that \( f_i \) can only choose “ADD” operation or two “ALT” operations. Then, \( f_i \) may select “ADD” operations with probability \( 1 - f(s_i) \) (add a SNP \( j \) on a LT or add \( j \) as a new LT with probability \( P_i(j) \)) and “ALT” operations with probability \( f(s_i) \) (select a SNP \( j \) with probability \( 1 - P_i(j) \) and replace it by another SNP \( s'_{\text{best}} \) selected with probability \( 1 - P_i(j) \)); see Figure 2.

(iii) If \( s_i = s_{\text{best}} \), then \( f_i \) holds the same number of SNPs as \( f_{\text{best}} \). We force \( f_i \) to choose “ALT” with a probability of 1 (select a SNP \( j \) with a probability of \( 1 - P_i(j) \) and replace it by another SNP \( s'_{\text{best}} \) that is selected with probability \( 1 - P_i(j') \)).

(4) KPDIST. If \( f_i \) selects “KPDIST” behavior, then it will keep a distance from \( f_{\text{best}} \), which indicates that \( f_i \) will attempt to

(i) change to a different size from \( f_{\text{best}} \)
(ii) select different SNPs with \( f_{\text{best}} \).

To achieve this goal, \( f_i \) checks the \( s \) value.

(i) If \( s_i > s_{\text{best}} \), then we force that \( f_i \) can choose only “ADD” operation, with probability of \( 1 - f(s_i) \), and choose two “ALT” operations with probability \( f(s_i) \).
(ii) If \( s_i < s_{\text{best}} \), then we force that \( f_i \) can only choose "ALT" operations and "DEL" operation with probability \( f(s_i) \) and 1 – \( f(s_i) \), respectively.

(iii) If \( s_i = s_{\text{best}} \), then we require that \( f_i \) can only choose "ADD" and "DEL" operations with the same probability of 1/2.

After applying a series of behaviors, each fish agent holds a new logic tree. If the new logic tree obtains a higher score than the previous logic tree, then the new logic tree explains more genotypes; next, this new logic tree is accepted and replaces the previous one. Otherwise, the new logic tree is rejected with a probability of \( Q \) [12]:

\[
Q = \min \left\{ 1, \frac{\nu^{k/2}|\hat{\Sigma}|^{1/2}}{\nu^{k/2}|\hat{\Sigma}|^{1/2} \exp \left( \frac{a}{a^*} \right)} \right\}, \tag{7}
\]

where \( \nu \) determines the prior variance, \( |\hat{\Sigma}| \) is the determinant of the posterior variance covariance matrix, and \( a \) is an error term to measure the fitness to the data \( Y \). where a **gamma function** is applied as \( a = \nu' \Omega - \hat{\beta}' \hat{\Sigma}^{-1} \hat{\beta} \). \( \Omega \) is the covariate matrix within the regression framework, \( \hat{\beta} \) is the associated vector of coefficients, and \( \hat{\Sigma} \) is the maximum a posteriori estimate of \( \beta \) which is found by Newton’s method. The superscript * refers to the parameters of the proposed updated model, and other parameters are denoted for those of \( f_{\text{best}} \). This acceptance probability \( Q \) in an equation is modified to check the proximity of the current solution (LT \( M^\theta \)) with the global optimal (LT \( M^* \)).

**3.4. Accepting the Candidate Models.** Each fish agent could store a local optimal logic tree during the search process, while the whole swarm always announces the current best logic tree (the global optimal). After several iterations, the reversible jump method is implemented. In other words, the acceptance probability of a newly proposed logic tree could decrease, but it might be closer to the best LT in the current iteration.

In addition, we consider a stepwise regression process. The stepwise regression eliminates insignificant SNPs iteratively and drops them off. The stepwise mechanism checks the active SNPs (SNPs not removed) every \( t_{\text{max}} \) iteration and determines whether a SNP should be removed or not, according to the results of an F statistic. At the same time, the level of significance \( F_{\text{out}} \) under the F statistic is introduced for determining whether an independent SNP should be removed or not. To achieve this goal, we calculate the total sum of squares (TSS) after dropping SNP \( i \) (\( i = 1, 2, \ldots, n \)):

\[
\text{TSS} = \text{RSS}_k + \text{ESS}_k, \tag{8}
\]

where \( k = 1, \ldots, n \) and \( k \neq i \). RSS is the sum of squares due to the regression, and ESS is the sum of squares due to the random errors or the residuals. We drop SNP \( i \) if the \( F \) statistic of SNP \( iF_i \) ≤ \( F_{\text{out}} \), where \( F_{\text{out}} \) is a preset threshold.

Finally, when \( f_{\text{best}} \) does not change in \( B \) iterations (the threshold \( B \) is preset), the regression algorithm terminates and outputs all of the logic trees that are held by all of the fish in the school.

**4. Results and Discussion**

We first apply our fish-swarm logic regression (FSLR) approach on a real screening dataset and then apply it on a series of simulated datasets under different configurations to test the performance of our approach compared to other logic regression-based approaches. The software tool, FSLR, is available at http://www.engr.uconn.edu/~jiwo09003/.

Three existing LR-based approaches are compared, which are Monte Carlo logic regression (MCLR) [12], full Bayesian logic regression (FBLR) [13], and the SNP(s) importance measurement approach (LogicFS) [14]. The software package attached to MCLR is LogicReg, the software package for FBLR is SCRIME, and for the SNP importance measurement approach, LogicFS is the name of the software package. We adopt two groups of parameters in MCLR: (1) \( \alpha = 1/2 \) with the total number of LTs \( K = 2 \), and (2) \( \alpha = 1/\sqrt{2} \) with \( K = 3 \), which are suggested in [12, 13]. Both MCLR and FBLR are preset to run 100,000 iterations with an additional 10,000 burn-in iterations in all of the experiments, as suggested.

**4.1. Fish Swarm on Real Mutation Screening Data.** The real dataset is from our own study, which focuses on the genetic association between the dopamine receptor D1 (DRD1) gene polymorphisms and the risk of opioid dependence. Seven possible functional single nucleotide polymorphisms, rs4867798, rs1799914, rs686, rs4532, rs5326, rs10063995, and rs10078866, in the regulatory or coding regions of DRD1 were identified by DNA sequencing in 20 heroin addicts and were further genotyped in 425 heroin addicts and 514 healthy controls.

Several genes that encode dopamine receptors have been confirmed to be associated with a risk of heroin addiction. Our previous studies [21, 22] as well as some studies from other labs [23–25] indicate that TaqI RFLP in the dopamine receptor D2 gene (DRD2) and –521 C/T in the Dopamine receptor D4 gene (DRD4) modulate the predisposition to heroin dependence. Several noncoding but potentially functional polymorphisms in flanking or untranslated regions of DRD1 have been identified, such as rs686 (+1403 T/C), rs4532 (−48 A/G), and rs5326 (−94 G/A). The rs686 polymorphism has been proven to affect the expression levels of DRD1 and might influence the level of DRD1 stimulation in PFC [26, 27].

Two types of drug dependence, nicotine [26] and alcohol [28], and addictive behavior (pathological gambling) [29] have both been shown to be associated with DRD1 (rs686, rs4532, and rs265981), which suggests that there are common effects of DRD1 on the susceptibility to addiction.

We applied our approach, FSLR, on this dataset. When considering the homozygote mutations, the logic regression model reports the highest score, which is 516 (among 939 individuals). rs4532 is the SNP with the highest importance. Two interactions, rs4532-rs686 and rs10078866-rs4532, are accepted much more than other interactions. When considering both the homozygote and the heterozygote mutations, two interactions, rs4532-rs1799914 and rs1799914-rs686, are accepted much more than the others, with the highest score being 518. These results, which are for candidate associations, are supported by clinical knowledge.
4.2. Simulated Data Preparation. For each simulation configuration, we generate 100 datasets. All of the datasets are generated by the ms-series simulator [30]. The ms-series is widely used for generating haplotypes/genotypes with preset parameters, for example, the mutation rate and recombination rate. For each dataset, we first use ms to generate 40,000 haplotypes with the same number of segregating sites equal to 1000. The neutral mutation rate is equal to $10^{-7}$, and the crossover probability between adjacent base pairs is equal to $10^{-7}$ per generation. Then, we randomly pick up a specific set of preset causal sites and generate a Boolean expression among them. According to the Boolean expression, we compute the phenotype of each haplotype: if the output of the expression is equal to 1, then the haplotype is a case; otherwise, the haplotype is a control. Note that complex traits are often affected by multiple factors. So we define the level of risk as equal to the probability of the phenotype being the same as the output of the Boolean expression. In other words, an individual that has mutations on causal sites has a higher probability of being a case, rather than having 100% chance of being a case. This probability is equal to the risk. Finally, we randomly sample 1,000 haplotypes from cases and 1,000 haplotypes from controls to make up one dataset. Moreover, datasets always have some errors. To get closer to the real datasets, we add noise (errors) on the simulated datasets. For the generated haplotypes, we define the level of noise as equal to the probability of randomly altering an allelic value from wild type to mutation or from mutation to wild type. We add the noise on the haplotypes randomly after the 2,000 haplotypes are sampled.

In the following sections, we will present the comparison results on three aspects: (1) the accuracy of each approach (measured by the type I and type II error rates), (2) the performance under different levels of risk and different levels of noise, and (3) the running time. To ensure confidence in the results, we conducted 100 repeats for each configuration used in the comparison.

4.3. Accuracy for the Different Numbers of Causal SNPs. We first compared the accuracy. The accuracy is measured by the type I error rate and the type II error rate, separately. The type I error rate is computed as the percentage of missed causal sites divided by the number of selected SNPs, while the type II error rate is computed as the percentage of wrong selections of noncausal SNPs among all of the SNPs involved in a regression model. For each simulation configuration, the number is computed based on 100 repeats.

Table 1: Accuracy for different numbers of causal SNPs. The column “Causal” shows the number of casual sites. The type I error rate is the percentage of missed causal sites divided by the number of selected SNPs. The type II error rate is the percentage of wrong selections of noncausal SNPs among all of the SNPs involved in a regression model. For each simulation configuration, the number is computed based on 100 repeats.

<table>
<thead>
<tr>
<th>Causal FSLR</th>
<th>Type I</th>
<th>Type II</th>
<th>MCLR</th>
<th>Type I</th>
<th>Type II</th>
<th>FBLR</th>
<th>Type I</th>
<th>Type II</th>
<th>LogicFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.65%</td>
<td>65.00%</td>
<td>1.38%</td>
<td>88.30%</td>
<td>0.52%</td>
<td>52.00%</td>
<td>0.63%</td>
<td>63.00%</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>1.38%</td>
<td>69.00%</td>
<td>1.21%</td>
<td>94.75%</td>
<td>1.34%</td>
<td>67.00%</td>
<td>1.47%</td>
<td>73.50%</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>1.75%</td>
<td>58.33%</td>
<td>1.20%</td>
<td>96.13%</td>
<td>2.15%</td>
<td>71.67%</td>
<td>2.21%</td>
<td>73.67%</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>2.53%</td>
<td>63.25%</td>
<td>1.18%</td>
<td>97.30%</td>
<td>3.02%</td>
<td>75.50%</td>
<td>3.22%</td>
<td>80.50%</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>3.72%</td>
<td>69.40%</td>
<td>1.14%</td>
<td>97.64%</td>
<td>4.05%</td>
<td>81.00%</td>
<td>3.98%</td>
<td>79.60%</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>3.80%</td>
<td>63.33%</td>
<td>1.10%</td>
<td>97.90%</td>
<td>4.73%</td>
<td>78.83%</td>
<td>4.90%</td>
<td>81.67%</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>4.62%</td>
<td>66.00%</td>
<td>1.08%</td>
<td>98.17%</td>
<td>5.78%</td>
<td>82.57%</td>
<td>5.82%</td>
<td>83.14%</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>5.40%</td>
<td>67.50%</td>
<td>1.09%</td>
<td>98.48%</td>
<td>6.24%</td>
<td>78.00%</td>
<td>6.58%</td>
<td>82.25%</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>5.38%</td>
<td>59.79%</td>
<td>1.10%</td>
<td>98.91%</td>
<td>7.24%</td>
<td>80.44%</td>
<td>7.67%</td>
<td>85.22%</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>6.44%</td>
<td>64.40%</td>
<td>1.05%</td>
<td>98.40%</td>
<td>7.76%</td>
<td>77.60%</td>
<td>8.47%</td>
<td>84.70%</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Comparisons on identifying preset causal sites. The column “Causal” shows the number of causal sites. A column under the name of an approach shows the average number (among 100 repeats) of successfully identified preset causal sites among the number of casual sites.

| Causal FSLR MCLR FBLR LogicFS |
|---|---|---|---|
| 10 | 3.5 | 1.23 | 4.8 | 3.7 |
| 20 | 6.2 | 1.05 | 6.6 | 5.3 |
| 30 | 12.5 | 1.16 | 8.5 | 7.9 |
| 40 | 14.7 | 1.08 | 9.8 | 7.8 |
| 50 | 12.8 | 1.18 | 9.5 | 10.2 |
| 60 | 22.0 | 1.26 | 12.7 | 11.0 |
| 70 | 23.8 | 1.28 | 12.2 | 11.8 |
| 80 | 26.0 | 1.22 | 17.6 | 14.2 |
| 90 | 36.2 | 0.98 | 17.5 | 13.3 |
| 100 | 35.6 | 1.60 | 22.4 | 15.3 |
Table 3: Accuracy for different numbers of causal SNPs with risks and noise. The level of risk is equal to the probability of the phenotype being the same as the output of the Boolean expression. The level of noise is equal to the probability of randomly altering an allelic value from wild type to mutation or from mutation to wild type. The type I and II error rates are similar. For each simulation configuration, the number is computed based on 100 repeats.

<table>
<thead>
<tr>
<th></th>
<th>FSLR Type I</th>
<th>FSLR Type II</th>
<th>MCLR Type I</th>
<th>MCLR Type II</th>
<th>FBLR Type I</th>
<th>FBLR Type II</th>
<th>LogicFS Type I</th>
<th>LogicFS Type II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Risk</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5%</td>
<td>12.8%</td>
<td>58.80%</td>
<td>1.16%</td>
<td>98.88%</td>
<td>8.70%</td>
<td>73.60%</td>
<td>6.80%</td>
<td>71.80%</td>
</tr>
<tr>
<td>10%</td>
<td>12.7%</td>
<td>59.00%</td>
<td>1.12%</td>
<td>98.44%</td>
<td>9.00%</td>
<td>77.60%</td>
<td>6.70%</td>
<td>74.40%</td>
</tr>
<tr>
<td>15%</td>
<td>12.3%</td>
<td>59.20%</td>
<td>1.19%</td>
<td>98.92%</td>
<td>8.90%</td>
<td>74.40%</td>
<td>6.60%</td>
<td>73.90%</td>
</tr>
<tr>
<td>Noise</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1%</td>
<td>12.5%</td>
<td>59.00%</td>
<td>1.17%</td>
<td>98.56%</td>
<td>9.00%</td>
<td>77.80%</td>
<td>6.70%</td>
<td>73.40%</td>
</tr>
<tr>
<td>2%</td>
<td>13.5%</td>
<td>58.00%</td>
<td>1.17%</td>
<td>98.76%</td>
<td>9.00%</td>
<td>81.80%</td>
<td>6.80%</td>
<td>74.60%</td>
</tr>
<tr>
<td>3%</td>
<td>14.8%</td>
<td>56.60%</td>
<td>1.08%</td>
<td>99.19%</td>
<td>8.90%</td>
<td>78.40%</td>
<td>7.30%</td>
<td>76.60%</td>
</tr>
</tbody>
</table>

4.4. Accuracy at Different Noise Levels. We also compared the accuracy under different levels of risk and different levels of noise. All of the datasets applied in this group of experiments have a total of 10 preset causal sites among the 1000 sites. We first varied the levels of risk from 5% to 15%; then, we varied the levels of noise on the haplotypes from 1% to 3%.

The results are compared in Tables 3 and 4. According to these results, we can conclude that the performance of FSLR, FBLR, and LogicFS is weakened either by the risk or by the noise. However, FSLR can identify more causal sites than FBLR and LogicFS. For example, FSLR successfully reports approximately 20 preset causal sites, while FBLR finds only ~11 and LogicFS only identifies 13 approximately. The reason that FSLR has higher type 1 error rates is that the regression models (the number of candidate causal sites) reported by FSLR are smaller than those of the other two methods. Thus, although FSLR misses fewer preset causal sites, the type I error rates are still higher. On the other hand, MCLR is not affected significantly by these, but the original performance of MCLR might not be good enough.

Table 4: Comparisons on identifying preset causal sites with risks and noise. A column under the name of an approach shows the average number (among 100 repeats) of successfully identified preset causal sites under the particular level of noise.

<table>
<thead>
<tr>
<th>Noise</th>
<th>FSLR</th>
<th>MCLR</th>
<th>FBLR</th>
<th>LogicFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>5%</td>
<td>20.6</td>
<td>0.72</td>
<td>11.1</td>
<td>13.3</td>
</tr>
<tr>
<td>10%</td>
<td>21.0</td>
<td>0.62</td>
<td>9.1</td>
<td>13.7</td>
</tr>
<tr>
<td>15%</td>
<td>21.7</td>
<td>0.50</td>
<td>10.8</td>
<td>11.7</td>
</tr>
<tr>
<td>1%</td>
<td>20.6</td>
<td>0.56</td>
<td>13.2</td>
<td>14.1</td>
</tr>
<tr>
<td>2%</td>
<td>20.5</td>
<td>1.28</td>
<td>11.2</td>
<td>13.3</td>
</tr>
<tr>
<td>3%</td>
<td>19.9</td>
<td>0.54</td>
<td>12.8</td>
<td>12.5</td>
</tr>
</tbody>
</table>

Table 5: Comparisons on running time. The running time is measured in seconds.

<table>
<thead>
<tr>
<th>Causal</th>
<th>FSLR</th>
<th>MCLR</th>
<th>FBLR</th>
<th>LogicFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>17.43</td>
<td>56.59</td>
<td>1659</td>
<td>12.23</td>
</tr>
<tr>
<td>20</td>
<td>18.48</td>
<td>53.64</td>
<td>1559</td>
<td>12.50</td>
</tr>
<tr>
<td>30</td>
<td>18.72</td>
<td>58.37</td>
<td>1603</td>
<td>12.12</td>
</tr>
<tr>
<td>40</td>
<td>18.96</td>
<td>57.76</td>
<td>1463</td>
<td>11.88</td>
</tr>
<tr>
<td>50</td>
<td>19.45</td>
<td>58.31</td>
<td>1520</td>
<td>12.10</td>
</tr>
<tr>
<td>60</td>
<td>19.94</td>
<td>57.72</td>
<td>1418</td>
<td>12.43</td>
</tr>
<tr>
<td>70</td>
<td>20.69</td>
<td>59.58</td>
<td>1482</td>
<td>12.11</td>
</tr>
<tr>
<td>80</td>
<td>22.49</td>
<td>58.04</td>
<td>1366</td>
<td>12.57</td>
</tr>
<tr>
<td>90</td>
<td>24.35</td>
<td>58.54</td>
<td>1466</td>
<td>12.79</td>
</tr>
<tr>
<td>100</td>
<td>24.35</td>
<td>59.13</td>
<td>1346</td>
<td>12.65</td>
</tr>
</tbody>
</table>

FSLR runs faster than MCLR and FBLR, while it has a similar speed to LogicFS.

4.5. Comparisons of the Running Time. In addition, we compare the running time among FSLR, MCLR, FBLR, and LogicFS. We record the average running time on 1000 repeats. Because both MCLR and FBLR rely on the Monte Carlo Markov chain (MCMC) to seek a better regression model, the number of iterations of MCMC might dominate the running time. Following the suggestions in the papers, we preset 100,000 iterations with an additional 10,000 burn-in iterations. LogicFS is preset by 20 iterations with bootstrap sampling. FSLR is applied on a cluster with 12 laptops. The collections of running time are shown in Table 5. Intuitively,

FSLR runs faster than MCLR and FBLR, while it has a similar speed to LogicFS.

5. Conclusions

In this paper, we present a novel logic regression-based approach, fish-swarm logic regression (FSLR), to detect the interacting SNPs that are associated with a phenotype. We designed a new regression algorithm, which incorporates the advantages of a swarm framework, to improve both the accuracy and the efficiency of logic regression. In contrast to previous swarm algorithms, in this approach, we design...
a specific solution space into which all possible logic trees are mapped. Then, two types of behaviors, agent behaviors and swarm behaviors, are suggested to rule the search strategy. A series of simulation experiments are performed to compare the accuracy under different scenarios of three logic regression-based approaches. The running times among the approaches are also collected. Our approach, fish-swarm logic regression-based approaches. The running times among the others.

Conflict of Interests

The authors declare that they have no competing financial interests.

Authors’ Contribution

Jiayin Wang, Xuanping Zhang, and Chunxia Yan conducted this research. Jiayin Wang, Aiyuan Yang, and Feng Zhu designed algorithms and experiments. Aiyuan Yang and Zhi Cao developed the software packages and participated in the performance analysis and the experiments on the real dataset. Jiayin Wang, Chunxia Yan, and Zhongmeng Zhao wrote this paper. All authors have read and approved the final manuscript.

Acknowledgments

This work was supported by the National Science Foundation [CCF-116175], the Ph.D. Program Foundation of the Ministry of Education of China [20100201110063], and the National Science Foundation of China [81172903].

References


