Supplementary Note 1: BiForce algorithm and implementation

Outline

One of the major challenges of studying epistasis in genome-wide association studies (GWAS) is the computational efficiency. Current methods are either unable to handle high density SNP data and therefore execute a pre-filtering or screening step prior to statistical tests to reduce the search space or apply high-performance computing solutions using advanced or supercomputer systems (e.g. graphics processing units (GPU), cloud and grid-based solutions, computer clusters). The majority of these methods concern only binary (i.e. case-control) data. Many complex (quantitative) traits may require even more computational resources. On the other hand, the number of SNPs used in GWAS has increased greatly providing an unprecedented opportunity to investigate epistasis on hundreds of billions of SNP pairs. Fast computational methods are desperately needed for GWAS to fill the gap between this opportunity and computational challenges. Here we present a novel tool named BiForce to tackle the challenges. BiForce is a powerful and versatile method able to carry out high throughput pair-wise epistasis detection for both binary (case-control) and quantitative traits by

1. Converting SNP genotype data into Boolean bit values and storing them in special arrays (called bit sets) for counting samples in each SNP genotype combination in a fast and CPU-efficient way
2. Efficiently calculating pair-wise SNP interactions using a logical AND operation over bit sets
3. Applying a fast, multithreaded approach to perform a full pair-wise genome scan for epistasis in a parallel manner

Experimental results show that BiForce can reduce the running time dramatically and make epistasis analysis in GWAS possible even on a single computer.

Data structure

GWAS raw data are stored as phenotype and genotype files. Phenotype data contains information about each individual and its phenotype that can be either a class label (i.e. case or control) in a disease trait, or a decimal number in a quantitative trait. Genotype data contain the genetic information of each individual as SNP genotypes: the homozygous of A allele (AA or 0), heterozygous (AB or 1) and homozygous of B allele (BB or 2). Table_S1_1 shows an example of a typical raw data format for a binary and quantitative dataset with six individuals (samples) and four SNPs.

Table_S1_1: A typical raw data format of GWAS data with four SNPs from six individual samples.

I. Phenotype data

I/1. Binary (case-control) data

Sample ID	Value
I_1	1
I_2	0
I_3	1
I_4	1
I_5	0
I_6	0

I/2.Quantitative data

Sample ID	Value
I_1	1.4
I_2	2.4
I_3	-1.3
I_4	-3.1
I_5	2.4
I_6	1.2

II. Genotype data

	Sample ID					
SNP ID	I_1	I_2	I_3	I_4	I_5	I_6
SNP_1	AA	AA	AB	AA	AB	BB
SNP_2	AB	AA	AB	AB	BB	BB
SNP_3	AA	AB	AA	AB	AB	AA
SNP_4	AA	BB	BB	AA	AB	AA

For contingency table based statistical tests of SNP interactions, phenotype values for each joint genotype need to be calculated first (see below). Because the running time of a pair-wise genome scan significantly depends on how efficient such a calculation is, the data structure in which genotype values are stored is critically important. Instead of storing SNP genotypes as integer values (e.g. 0, 1, 2) each using 16 or 32 bits of computer memory, BiForce use bitwise arrays to manage the genotype data where each component of a bitwise array has a Boolean value, i.e. either on/true/1 or off/false/0. Because each SNP can only have three genotypes, we can represent each genotype as one bit – if a sample has the genotype the bit value is 1 otherwise 0. Therefore, each SNP S_X can be represented by 3 bit sets, S_X^0, S_X^1 and S_X^2, corresponding to the three genotypes 0, 1 and 2. If a sample has a genotype at the SNP, one and only one of the 3 bits will be set to 1. Otherwise, the genotype is missing (e.g.NA) so none of the 3 bits will be set (i.e. all remain 0).

Thus, by applying the bitwise data structure, BiForce will save 13 or 29 bits (from 16 or 32 bits to only 3 bits) for each SNP and consequently is very memory efficient in light of a large number of SNPs used in GWAS. More importantly, such bitwise data structures can cope with missing genotypes easily without the need of imputing and enable very fast bitwise operations (see section of Construction of Contingency Tables below). To use the bitwise data structures, the raw GWAS data need to be converted and stored in bitwise arrays first as illustrated in Table_S1_2. BiForce uses Java BitSet to implements the bitwise arrays that can be increased in size if necessary and indexed by a nonnegative integer corresponding to each individual sample to allow fast storage and retrieval.

Table_S1_2: Raw data represented as in bitwise data structure.

Genotype data after conversion (Binary data)

Cases	I_1	I_3	I_4	Cont.	I_2	I_5	I_6
$S_1^0 =$	1	0	1	$S_1^0 =$	1	0	0
$S_1^1 =$	0	1	0	$S_1^1 =$	0	1	0
$S_1^2 =$	0	0	0	$S_1^2 =$	0	0	1
$S_2^0 =$	0	0	0	$S_2^0 =$	1	0	0
$S_2^1 =$	1	1	1	$S_2^1 =$	0	0	0
$S_2^2 =$	0	0	0	$S_2^2 =$	0	1	1
$S_3^0 =$	1	1	0	$S_3^0 =$	0	0	1
$S_3^1 =$	0	0	1	$S_3^1 =$	1	1	0
$S_3^2 =$	0	0	0	$S_3^2 =$	0	0	0
$S_4^0 =$	1	0	1	$S_4^0 =$	0	0	1
$S_4^1 =$	0	0	0	$S_4^1 =$	0	1	0
$S_4^2 =$	0	1	0	$S_4^2 =$	1	0	0

Genotype data after conversion (Quantitative data)

	I_1	I_2	I_3	I_4	I_5	I_6
$S_1^0 =$	1	1	0	1	0	0
$S_1^1 =$	0	0	1	0	1	0
$S_1^2 =$	0	0	0	0	0	1
$S_2^0 =$	0	1	0	0	0	0
$S_2^1 =$	1	0	1	1	0	0
$S_2^2 =$	0	0	0	0	1	1
$S_3^0 =$	1	0	1	0	0	1
$S_3^1 =$	0	1	0	1	1	0
$S_3^2 =$	0	0	0	0	0	0
$S_4^0 =$	1	0	0	1	0	1
$S_4^1 =$	0	0	0	0	1	0
$S_4^2 =$	0	1	1	0	0	0

The bitwise data structures can also be applied to manage the phenotypes in binary traits where two bit sets are used to represent case and control respectively. Again these data structures allow BiForce to use bitwise operations to compute the contingency tables in a fast, CPU-efficient manner. The bitwise data structures however, are not applicable to phenotypes in quantitative traits.

Construction of Contingency Tables

In order to calculate the interactions between two SNPs, S_X and S_Y, first a contingency table C is constructed with three variables: the trait value (e.g. disease status, i.e. case or control) and the two SNPs. We define the contingency table using a binary trait as an example as follows. Let C be defined as a $|t|$ x $(|x| \cdot |y|)$ matrix, where a matrix element $n_{t,z}$ represents the number of samples with joint genotype z (0 to 8) and trait value t, where $t \in \{0,1\}$, the first SNP's genotype code $x \in \{0,1,2\}$ and the second SNP's genotype code $y \in \{0,1,2\}$ (Table_S1_3). $n_{t,z}$ is calculated using a logical AND function (or intersection) of the two SNPs' bit sets S_X^x and S_Y^y as follows:

$$n_{t,z} = \left|S_X^x \wedge S_Y^y\right|, z = 3 \cdot x + y.$$

Table_S1_3: Contingency table for GWAS binary data.

Trait	$S_X^0 \wedge S_Y^0$	$S_X^0 \wedge S_Y^1$	$S_X^0 \wedge S_Y^2$	$S_X^1 \wedge S_Y^0$	$S_X^1 \wedge S_Y^1$	$S_X^1 \wedge S_Y^2$	$S_X^2 \wedge S_Y^0$	$S_X^2 \wedge S_Y^1$	$S_X^2 \wedge S_Y^2$
t_0	$n_{0,0}$	$n_{0,1}$	$n_{0,2}$	$n_{0,3}$	$n_{0,4}$	$n_{0,5}$	$n_{0,6}$	$n_{0,7}$	$n_{0,8}$
t_1	$n_{1,0}$	$n_{1,1}$	$n_{1,2}$	$n_{1,3}$	$n_{1,4}$	$n_{1,5}$	$n_{1,6}$	$n_{1,7}$	$n_{1,8}$

To illustrate how contingency table is built, let's take the case samples and SNPs S_1 and S_2 in Table_S1_2 to calculate $n_{1,1}$ (i.e. the number of case samples with S_1 taking genotype 0 and S_2 taking genotype 1):

$$n_{1,1} = |S_1^0 \wedge S_2^1| = |1\ 0\ 1 \wedge 1\ 1\ 1| = |1\ 0\ 1| = 2.$$

Note that the logical AND function ($\wedge$) gives a TRUE (1) only if both operands are TURE and the final result of 2 is obtained by counting the 1s. The full contingency table constructed for the pair of SNPs S_1 and S_2 is given in Table_S1_4.

Table_S1_4: The contingency table for the pair of SNPs *S_1* and *S_2* with binary data from Table_S1_2.

Trait	$S_1^0 \wedge S_2^0$	$S_1^0 \wedge S_2^1$	$S_1^0 \wedge S_2^2$	$S_1^1 \wedge S_2^0$	$S_1^1 \wedge S_2^1$	$S_1^1 \wedge S_2^2$	$S_1^2 \wedge S_2^0$	$S_1^2 \wedge S_2^1$	$S_1^2 \wedge S_2^2$
t_0	0	2	0	0	1	0	0	0	0
t_1	1	0	0	0	0	1	0	0	1

The definitions above are also applied to construct contingency tables for quantitative traits where each table has only a single row instead of two for binary traits. As bitwise operations cannot be used to calculate the counts and sums (mean = sum/count) in the contingency table for quantitative traits, a simple loop is used over all bits in a BitSet. This results in longer running time for quantitative traits.

Statistical tests of pair-wise SNP interactions

Binary Traits

BiForce uses the contingency tables constructed as input to calculate test statistics of interaction between each pair of SNPs. For binary traits, BiForce adopts the approximation step and the log-likelihood ratio tests as implemented in BOOST (Wan *et. al.* 2010, American Journal of Human Genetics, 87: 325-340) (http://bioinformatics.ust.hk/BOOST.html) as a default option to accelerate the full pair-wise genome scan. The approximation step can be dismissed when necessary and thus perform the exhaustive scan using only the log-likelihood ratio tests. Briefly, considering a SNP pair of S_1 and S_2 in case (t_1) and control (t_0) in Table_S1_4 with a total number of samples of n:

a. calculate conditional probabilities $P(S_1|S_2)$, $P(S_2|t)$, $P(t|S_1)$
b. for t_0 (and then t_1)
    i. calculate *probability*log(probability)* for each of the 9 joint genotypes and the sum (*int_sum_a* for t_0 and *int_sum_c* for t_1)
    ii. calculate $P(S_1|S_2)*P(S_2|t)*P(t|S_1)$ for each of the 9 joint genotypes and then the sum (*tao_a* for t_0 and *tao_b* for t_1)
    iii. calculate *-probability**log($P(S_1|S_2)*P(S_2|t)*P(t|S_1)$) for each of the 9 joint genotypes and the sum (*int_sum_b* for t_0 and *int_sum_d* for t_1)
c. log-likelihood ratio for the interactions is calculated as:

$$2*n*((int\_sum\_a+int\_sum\_b+int\_sum\_c+int\_sum\_d)+\log(tao\_a+tao\_b))$$

Quantitative traits

For quantitative traits, BiForce uses F-ratio tests implemented using ANOVA. Considering SNP pair of S_1 and S_2 in quantitative trait t with a total number of samples of n and a contingency table of number of samples in each joint genotype similar to Table_S1_4, BiForce first calculates the overall mean μ, the total phenotypic variance $SST = \sum_i^n (t_i - \mu)^2$ and then derives a new 3 x 3 contingency table of means of each joint genotype m_{ij} with cnt_{ij} number of samples quoted from the previous contingency table (i ~ 0 to 2, j ~ 0 to 2), row (r_0 to r_2) and column (c_0 to c_2) means (Table_S1_5).

Table_S1_5: The 3 x 3 contingency table of joint genotype means for the pair of SNPs *S_1* and *S_2*

	S_2^0	S_2^1	S_2^2	row_mean
S_1^0	m_{00}	m_{01}	m_{02}	r_0
S_1^1	m_{10}	m_{11}	m_{12}	r_1
S_1^2	m_{20}	m_{21}	m_{22}	r_2
col_mean	c_0	c_1	c_2	μ

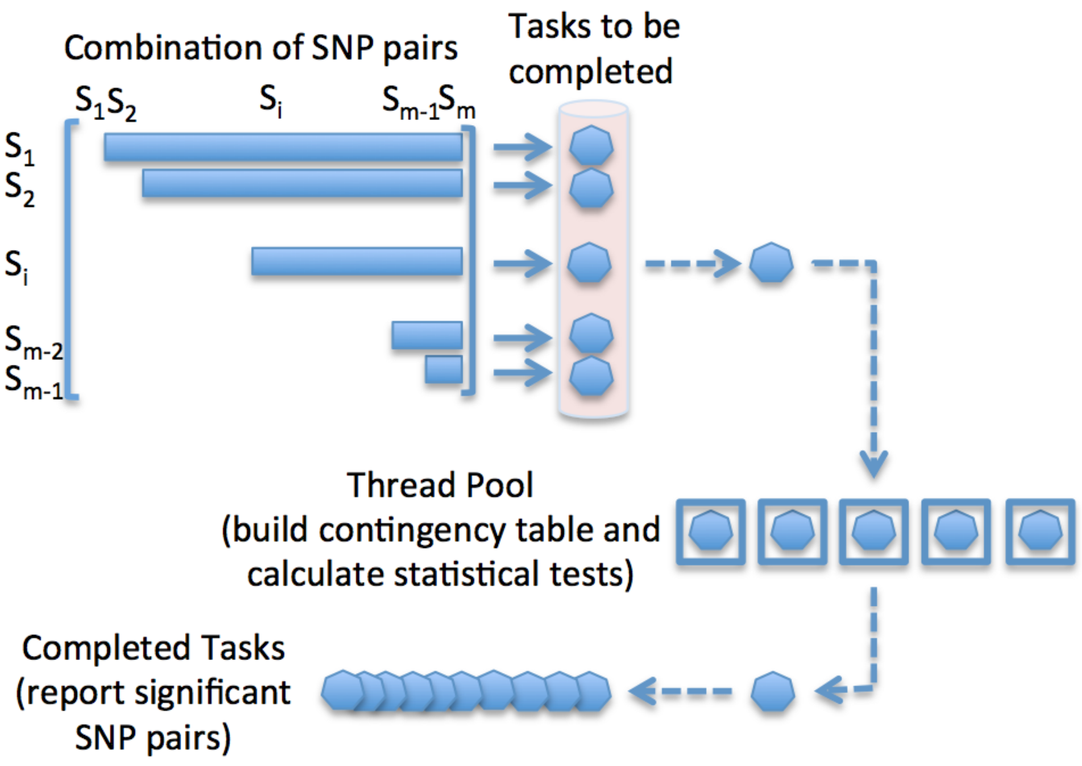
BiForce then computes the between group variance $SSB = \sum_{i=0}^{2}\sum_{j=0}^{2} cnt_{ij}(m_{ij} - \mu)^2$ and interaction variance $SSI = \sum_{i=0}^{2}\sum_{j=0}^{2} cnt_{ij}(m_{ij} - r_i - c_j + \mu)^2$ and then the residual variance $SSW = SST - SSB$. The F ratio of interaction is $F_{\text{int}} = (SSI/dfI)/(SSW/dfW)$ where dfW and dfI are the degrees of freedom for the residual (e.g. n-9) and the interaction (e.g. 4) respectively.

Multithreaded approach for full pair-wise genome scans

During the past few years, computer processors went through a major evolution in computing technology. Multi-core processors have eventually become the standard computing model because of the provided performance benefits beyond the capabilities of single-core processors. Therefore, it is becoming more and more common for computer systems to have multiple processors or processors with multiple execution cores. At the same time, however, there is a lack of method development among computational biologists to exploit the opportunities that this computing model promises. Almost all of the methods reported for genome wide association studies restrict their access to a single thread and thus, a single core, regardless of the available number of cores. Multithreaded approach, in contrast, allows access to two or more threads depending solely on the given computer power. In a multithreaded environment, each thread of a process can run on a separate processor at the same time, resulting in a parallel execution. Even on a single processor, a multithreaded method is running faster than a method developed merely for a single thread as computer systems even with a single execution core have many active threads.

BiForce is designed and implemented to fully exploit the advantage of multithreaded environment. The main steps of the method can be summarized as follows:

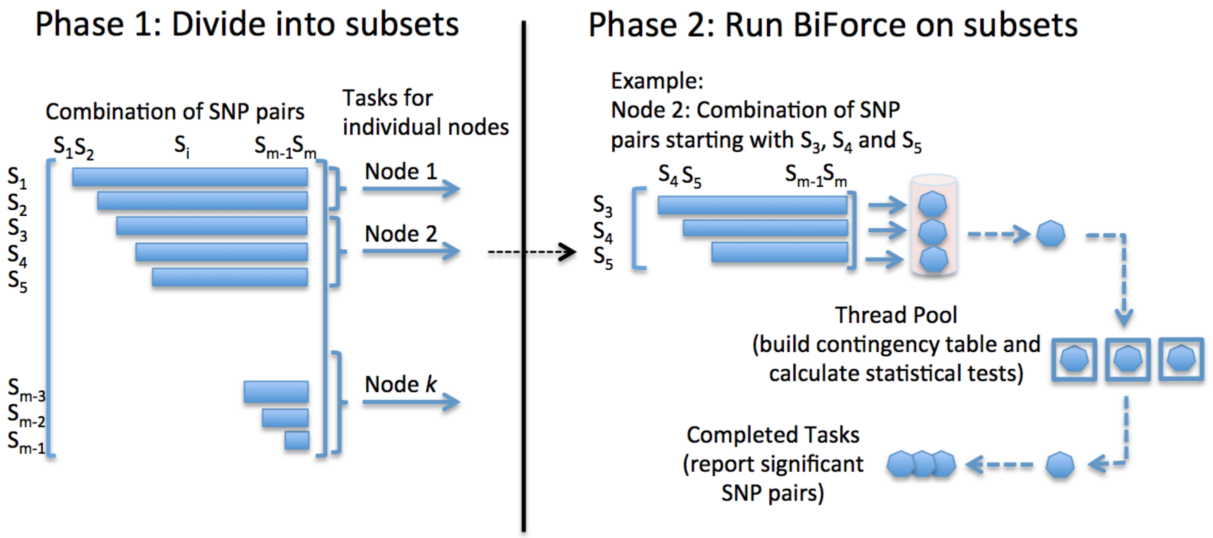
- Step 1: Convert raw data to bit sets as described above.
- Step 2: For each SNP S_i, i={*1*,*2*,..,*m*-1}, collect all (S_i, S_j) SNP pairs, where j={i+1, i+2,…,*m*}. Let $Task_i$ denote all SNP pairs having S_i as the first SNP of the pair (see Figure_S1_1).
- Step 3: Send all $Task_i$, i={1,2,…,*m*-1} to the Thread Pool. (While a thread can be thought as an actual worker, Thread Pool is a group of workers.) For each task $Task_i$ in the pool, build contingency table and calculate statistical tests for its SNP pairs.
- Step 4: Report SNP pairs with interaction values above the required threshold.



Figure_S1_1: Illustration of multithreaded steps applied in the BiForce method.

BiForce on parallel computing platforms

In addition to single computers with multi-cores, BiForce is able to run on various parallel computing platforms including clusters, MPPs and grids. These systems are specifically designed to take large data, divide them into subparts, allowing the individual computer nodes to process their own parts. To ensure the best running time, BiForce performs the division by splitting up the data into closely equal parts applying a simple greedy strategy before distributing the data parts to the nodes (Figure_S1_2). Applying BiForce on parallel platforms, the running time can be further reduced from hours to minutes and from minutes to seconds depending on the number of available nodes.


Phase 1: Divide into subsets
Combination of SNP pairs
Tasks for individual nodes
S_1S_2
S_i
$S_{m-1}S_m$
S_1
S_2
S_3
S_4
S_5
S_{m-3}
S_{m-2}
S_{m-1}
Node 1
Node 2
Node *k*
Phase 2: Run BiForce on subsets
Example:
Node 2: Combination of SNP pairs starting with S_3, S_4 and S_5
S_4S_5
$S_{m-1}S_m$
S_3
S_4
S_5
Thread Pool
(build contingency table and calculate statistical tests)
Completed Tasks
(report significant SNP pairs)

Figure_S1_2: Parallel distribution method applied in the BiForce method.