ParaHaplo Manual

Program for Genome-Wide Association Study Using Parallel Computing

Version 4.0

2013/2/28

Kazuharu Misawa [a] and Naoyuki Kamatani [b]

[a] Research Program for Computational Science, Research and Development Group for Next-Generation Integrated Living Matter Simulation, Fusion of Data and Analysis Research and Development Team, RIKEN, 4-6-1 Shirokane-dai, Minato-ku, Tokyo 108-8639, Japan.

[b] Laboratory for Statistical Analysis, RIKEN Center for Genomic Medicine, Tokyo, Japan

2010/9/10 Manual Revised
2010/10/25 Manual Revised
2010/11/15 Manual Revised
2010/12/25 Manual Revised
2011/1/26 Manual Revised
2011/2/28 Version 3.0 Released
2013/2/15 Version 4.0 Released

Table of Contents

Introduction 1

The purpose of this program package 1

Program overivew .... エラー! ブックマークが定義されていません。

Operating systems and platforms 3

Install 4

Download and expand 4

How to install for PC-cluster 6

How to install for RIKEN RICC 7

How to install for RIKEN FX1 8

How to install for K-computer 9

Usage of programs in the packages 10

Imputation 10

Before imputation .... エラー! ブックマークが定義されていません。

How to run the MPI version of imputation on PC clusters 10

How to run the non-parallel version of imputation on PC clusters 12

Haprotype phasing 13

Before using haplotype phasing .... エラー! ブックマークが定義されていません。

How to run the MPI version of haplotype phasing on PC clusters 13

How to run the non-parallel version of haplotype phasing 14

Calculation of type I error rates by using MCMC method 15

How to run the MPI version in parallel on PC clusters in C 15

How to run the OpenMP version in parallel on PC clusters in C 16

How to run the MPI+OpenMP version in parallel on PC clusters in C 17

How to run the non-parallel version in C 18

Calculation of type I error rates by using MCMC/RAT method 19

How to run the MPI version in parallel on K-computer or RIKEN RICC in C 19

How to run the OpenMP version in parallel on K-computer or RIKEN RICC in C 20

How to run the MPI+OpenMP version in parallel on K-computer or RIKEN RICC in C 22

How to run the non-parallel version on K-computer or RIKEN RICC in C 24

Calculation of type I error rates by using MCMC/RAT method (use divided file). 25

Exact Calculation of type I error rates 26

How to run the version in C 26

Permutation test based on RAT algorithm (5) 28

How to run the MPI version in parallel in C 28

How to run the MPI+OpenMP version in parallel in C 30

How to run the non-parallel version in C 31

Standard Permutation test (SPT) 32
How to run the MPI version in parallel on PC clusters in C 32
How to run the non-parallel version in C 34
Data formats of input and output files 35
Data conversion from BioBank Japan format to HapMap format 35
Data conversion from PED/MAP format to HapMap format 35
Haplotype block file 36
Haplotype blocks defined by user 36
Sliding window method 36
Output file 37
The header part 37
The result part 37
Calculating the global P value 38
Literature cited ..... エラー！ ブックマークが定義されていません。

Introduction

The purpose of this software package

Recent advances in high-throughput genotyping technologies have allowed us to test allele-frequency differences between case and control populations on a genome-wide scale (1). Accurate and complete genotypes have been obtained for more than one million single nucleotide polymorphisms (SNPs), and thousands of people are now being genotyped (2,3).

One of the crucial problems in genome-wide association studies (GWAS) is correcting for multiple comparisons. Bonferroni’s correction for the P value is typically used to account for multiple testing. However, when SNP loci are in linkage disequilibrium, Bonferroni’s correction is too conservative and could drop truly significant SNPs (4,5).

To correct for the multiple comparisons at multiple SNP loci in linkage disequilibrium, Misawa et al. (4) have developed new algorithms that treat linked loci as one haplotype block. They developed the method to calculate the exact probability of the type I error with the following prerequisite conditions: (i) that the haplotype frequencies in the population are known, and the number of haplotype copies in sample follows a multinomial distribution. The permutation test also can handle this problem (5).

Running time is a problem in calculating the exact probability (4) as well as in performing permutation tests (5). In this study, we developed ParaHaplo (6) , parallel computation programs for calculating exact probability (4) as well as for permutation tests (5) for GWAS. ParaHaplo is based on data parallelism, a programming technique for splits a large data set into small data sets to be processed in parallel. ParaHaplo was developed based on the Intel Message Passing Interface (MPI) and was designed to run on PC clusters.

To measure the efficiency of ParaHaplo in computational time, the difference in haplotype frequency between CHB and JPT HapMap (7) on chromosome 22 was analyzed using the new programs. The result showed that ParaHaplo is more than 100 times faster than existing solutions.

Software overview

ParaHaplo tests the difference in allele frequency between case (experimental) and control population, or between any two populations. ParaHaplo can calculate and display the Pearson score for chi-square test and Fst, depending on a command-line option. ParaHaplo calculates the exact probability of type I errors of allele frequency tests (4). The program also includes an implementation of the algorithm that asymptotically calculates the type I error rates using a Markov-chain Monte Carlo sampler (4). This program also supports the standard permutation test (SPT) and RAT-based permutation tests (5).

ParaHaplo was implemented on a message passing interface (MPI) multithreaded package, which allowed us to construct parallel-computing programs on multiprocessors. The genome-wide polymorphism data is broken into haplotype blocks that are user-defined. Then, the MPI-Bcast function is used to distribute a single set of haplotype block data into each processor. The haplotype frequency data of one block are analyzed by a single-processor. In this step, ParaHaplo calculates the probability of local type I error, given the significance level of each SNP locus.

When the analysis on each haplotype block is complete, the results are combined into a single genome-wide data using the MPI-Gatherv function. Then, the global type I error is obtained from the local type I error using Bonferroni’s correction, because different haplotype blocks are considered to be independent of each other, although SNPs within haplotype blocks are not independent.

ParaHaplo requires an input file containing the haplotype block boundaries and two files containing population data. ParaHaplo HapMap data format and BioBank Japan data format are supported for chromosome data files. ParaHaplo is compatible with OpenMPI version 1.2.5 and with MPICH version 1.2.7p1.The source code can be compiled using either a GCC compiler or an Intel C compiler. Both C programs and Java programs are available for single-processor machines. This software package contains the following programs:

(i) Markov-chain Monte Carlo (MCMC) algorithm to asymptotically calculate the probability of the type I error (4)

(ii) Calculation of exact type I error rates (4)

(iii) Permutation test based on RAT algorithm (5)

(iv) Standard permutation test (SPT)

(v) Data conversion from BioBank Japan format or PED/MAP format to HapMap format

Operating systems and platforms

The parallel and single-processor versions coded in C were tested on the following machine:

Machine	PC cluster with 1,024 nodes CPU: Intel Core2Duo 1.5 GHz Memory: 2 GB HDD: 4 GB
OS	Linux OS（CentOS 4.5）
Parallel computing	OpenMPI and MPICH
Compiler	GCC4, Intel C compiler, Fujitsu compiler

The single-processor versions coded in Java were tested on the following machine:

Machine 1	PC cluster with 1,024 nodes CPU: Intel Core2Duo 1.5 GHz Memory: 2 GB HDD: 4 GB
Machine 2	PC with two processors CPU: Intel Xeon 3 GHz Memory: 4 GB HDD: 400 GB
OS	Windows Vista and Linux OS（CentOS 4.5）

Installation

Downloading and expanding

The source programs and executable binaries are available for download at:

http://sourceforge.jp/projects/parallelgwas/?_sl=1

After *.tar.gz is downloaded, run:

```
%> gunzip *.tar.gz
%> tar xvf *.tar
```

This procedure will create the SNP directory that contains the files and directories shown in Table 1.

Table 1. Files and directories of the ParaHaplo statistical software package

Directory name	Contents
SNP	Root directory for ParaHaplo
SNP/bin	Executable binaries of ParaHaplo
SNP/HaplotypePhasing	Source programs of the single-processor version as well as the MPI parallel version of haplotype phasing (Misawa and Kamatani, unpublished)
SNP/SNP_MultiLocus	Source programs of the single-processor version as well as the MPI parallel version of the Markov-chain Monte Carlo (MCMC) algorithm that asymptotically calculates the probability of type I errors (4)
SNP/SNP_Exact	Source programs for the single-processor version that calculates the exact probability of type I errors (4)
SNP/SNP_RATParallel	Source programs for the MPI parallel version of the permutation test based on the RAT algorithm (5)
SNP/SNP_RAT_Hybrid	Source programs for the MPI+OpenMP parallel version of the permutation test based on the RAT algorithm (5)
SNP/SNP_RAT	Source programs of the single-processor version of the permutation test based on the RAT algorithm (5)
SNP/SNP_PrimitiveParallel	Source programs for the MPI parallel version of the permutation test based on the standard permutation algorithm

SNP/SNP_Primitive	Source programs of the single-processor version of the permutation test based on the standard permutation algorithm
SNP/SNP_LIB	Library of functions
SNP/dSFMT	Source program of random-number generator of the Mersenne twister algorithm (*)
SNP/tool	Additional scripts and tools
SNP/script	Scripts for the PC cluster using gcc compiler
SNP/script_RICC	Scripts for RIKEN RICC
SNP/script_FX	Scripts for RIKEN FX1
SNP/script_K	Scripts for the K-computer using the Fujitsu compiler.

(*) Copyright (c) 2006, 2007 Mutsuo Saito, Makoto Matsumoto, and Hiroshima University. All rights reserved. Copyright notice of the Mersenne twister algorithm is also included in the same directory.

Before compiling and running, the paths must be set as follows:

```
        PATH=$PATH:[$SNPinstalldir]/SNP/bin
        export SNP_HOME=[$SNPinstalldir]/SNP
        export PATH=$PATH:[$mpich2-installdir]/bin
        export
```

Compiling for a PC cluster

To compile the non-parallel version of the programs:

```
%> cd SNP
%> make –f Makefile.linux single
```

To compile the MPI parallel version of the programs:

```
%> cd SNP
%> make –f Makefile.linux parallel
```

To compile the MPI+OpenMP parallel version of the programs:

```
%> cd SNP
%> make –f Makefile.linux hybrid
```

To compile the OpenMP parallel version of the programs:

```
%> cd SNP
%> make –f Makefile.linux openmp
```

To compile the tools:

```
%> cd SNP
%> make –f Makefile.linux tools
```

To compile all programs:

```
%> cd SNP
%> make –f Makefile.linux all
```

To clean up the working directories:

```
%> cd SNP
%> make –f Makefile.linux clean
```

After compilation, all executable binaries will be in the SNP/bin directory.

Compiling for the RIKEN RICC

To compile the non-parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_RICC.linux single
```

To compile the MPI parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_RICC.linux parallel
```

To compile the MPI+OpenMP parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_RICC.linux hybrid
```

To compile the OpenMP parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_RICC.linux openmp
```

To compile the tools:

```
%> cd SNP
%> make –f Makefile_RICC.linux tools
```

To compile all programs:

```
%> cd SNP
%> make –f Makefile_RICC.linux all
```

To clean up the working directories:

```
%> cd SNP
%> make –f Makefile_RICC.linux clean
```

After compilation, all executable binaries will be in the SNP/bin directory.

Compiling for the RIKEN FX1

To compile the non-parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_FX1.linux single
```

To compile the MPI parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_FX1.linux parallel
```

To compile the MPI+OpenMP parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_FX1.linux hybrid
```

To compile the OpenMP parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_FX1.linux openmp
```

To compile the tools:

```
%> cd SNP
%> make –f Makefile_FX1.linux tools
```

To compile all programs:

```
%> cd SNP
%> make –f Makefile_FX1.linux all
```

To clean up the working directories:

```
%> cd SNP
%> make –f Makefile_FX1.linux clean
```

After compilation, all executable binaries will be in the SNP/bin directory.

Compiling for the K-computer

To compile the non-parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_K.linux single
```

To compile the MPI parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_K.linux parallel
```

To compile the MPI+OpenMP parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_K.linux hybrid
```

To compile the OpenMP parallel version of the programs:

```
%> cd SNP
%> make –f Makefile_K.linux openmp
```

To compile the tools:

```
%> cd SNP
%> make –f Makefile_K.linux tools
```

To compile all programs:

```
%> cd SNP
%> make –f Makefile_K.linux all
```

To clean up the working directories:

```
%> cd SNP
%> make –f Makefile_K.linux clean
```

After compilation, all executable binaries will be in the SNP/bin directory.

Using the programs in the package

This chapter presents the usage of the following programs in the package:

- Imputation (MPI parallel and non-parallel)
- Haplotype phasing (MPI parallel and non-parallel)
- Calculation of type I error rates using the MCMC method on a PC cluster in C (MPI parallel, OpenMP parallel, MPI+OpenMP parallel, and non-parallel)
- Calculation of type I error rates using the MCMC/RAT method on a K-computer or a RIKEN RICC in C (MPI parallel, OpenMP parallel, MPI+OpenMP parallel, and non-parallel)
- Exact calculation of type I error rates in C (non-parallel)

The usage instructions for the following tests are also included:

- Permutation test based on RAT algorithm in C (MPI parallel, MPI+OpenMP parallel, non-parallel)
- Standard permutation test in C (MPI parallel, non-parallel)

Finally, the formats and contents of the input and output files are discussed:

- Data conversions from BioBank Japan format or PED/MAP format to HapMap format
- Haplotype block file
- Output file

Imputation

Imputation is performed to fill in the missing data with probable data to improve statistical accuracy. This package includes two versions of the imputation program: (i) the MPI parallel version, and (ii) the non-parallel version.

Prior to running the imputation program, the MACH haplotyper (8) must be installed, and the paths for these programs must be appropriately set.

Running the MPI parallel version of the imputation program

To run the MPI parallel version of the imputation program, use ImputationParallel.sh, which is located in the SNP/script, SNP/script_FX1, SNP/script_RICC, and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

For PC cluster, RIKEN FX1, RIKEN RICC:

```
%> ImputationParallel.sh $1 $2 $3 $4 $5 $6 $7 $8 $9
```

For K-computer:

```
%> ImputationParallel.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10
```

Command-line options are as follows:

$1:	Input file to be imputed (HapMap data format)
$2:	Reference file (phased HapMap data format)
$3:	Haplotype block file
$4:	Recombination file If $8 is 0, the recombination rate will be read from the file. If $8 is larger than 0, the recombination rate will be estimated and written to the file.
$5:	Error rate file
$6:	Output file
$7:	Sample size per division of pedigree
$8:	Sample size for estimating parameters (If 0 is set, parameters will not be estimated.)
$9:	Number of processing units that run in parallel
$10:	Number of nodes that run in parallel (For the K-computer only.)

Running the non-parallel version of the imputation program

To run the non-parallel version of the imputation program, use Imputation.sh, which is located in the SNP/script, SNP/script_FX1, SNP/script_RICC, and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

```
%> Imputation.sh $1 $2 $3 $4 $5 $6 $7 $8
```

Command-line options are as follows:

$1:	Input file to be imputed (HapMap data format)
$2:	Reference file (phased HapMap data format)
$3:	Haplotype block file
$4:	Recombination file If $8 is 0, the recombination rate will be read from the file. If $8 is larger than 0, the recombination rate will be estimated and written to the file.
$5:	Error rate file
$6:	Output file
$7:	Sample size per division of pedigree
$8:	Sample size for estimating parameters (If 0 is set, the parameters will not be estimated.)

Haplotype phasing

The haplotype phasing program calculates the probability of haplotype phasing. This package includes two versions of the haplotype phasing program: (i) the MPI parallel version, and (ii) the non-parallel version.

Prior to running this program, PHASE 2.1 (9) and SNPHAP 1.3.1 (9,10) must be installed, and the paths to these programs must be set appropriately.

Running the MPI parallel version of the haplotype phasing program

To run the MPI parallel version of the haplotype phasing program, use HaplotypePhasingParallel.sh, which is located in the SNP/script, SNP/script_FX1, SNP/script_RICC, and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

For PC cluster, RIKEN FX1, RIKEN RICC:

```
%> HaplotypePhasingParallel.sh $1 $2 $3 $4 $5 $6
```

For K-computer:

```
%> HaplotypePhasingParallel.sh $1 $2 $3 $4 $5 $6 $7
```

Command-line options are as follows:

$1:	Input file (HapMap data format)
$2:	Haplotype block file
$3:	Output file
$4:	Option for phasing program
	0: Use SNPHAP 1.3
	1: Use PHASE 2.1
$5:	Upper bound of the number of SNPs
$6:	Number of processing units that run in parallel
$7:	Number of nodes that run in parallel (For the K-computer only.)

Running the non-parallel version of the haplotype phasing program

To run the non-parallel version of the haplotype phasing program, use HaplotypePhasing.sh, which is located in the SNP/script, SNP/script_FX1, SNP/script_RICC, and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

```
%> HaplotypePhasing.sh $1 $2 $3 $4 $5
```

Command-line options are as follows:

$1:	Input file (HapMap data format)
$2:	Haplotype block file
$3:	Output file
$4:	Option for phasing program
	0: Use SNPHAP 1.3
	1: Use PHASE 2.1
$5:	Upper bound of the number of SNPs

Calculation of type I error rates using the MCMC method on a PC cluster

The TypeI programs are used to asymptotically calculate the probability of type I errors. This set uses the MCMC method. This package includes the following versions of the TypeI programs using the MCMC method on a PC cluster: (i) the MPI parallel version in C, (ii) the OpenMP parallel version in C, (iii) the MPI+OpenMP parallel version in C, (iv) the non-parallel version in C, and (v) the non-parallel version in Java.

Running the MPI parallel version of the TypeI program using MCMC in C

To run the MPI parallel version of the TypeI program using the MCMC method in C, use TypeIParallel.sh, which is located in the SNP/script and SNP/script_FX1 directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

```
%> TypeIParallel.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10
```

Command-line options are as follows:

$1:	The first data file, as the case population
$2:	The second data file, as the control population
$3:	Output file
$4:	Haplotype block option 0: User-defined haplotype blocks 1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score 0: Pearson score 1: Fst
$7:	Number of repeats
$8:	Number of MCMC generations
$9:	Input data format 0: HapMap data 1: Haplotype data 2: Phased HapMap data
$10:	Number of processing units that run in parallel

Running the OpenMP parallel version of the TypeI program using MCMC in C

To run the OpenMP parallel version of the TypeI program using the MCMC method in C, use TypeIOpenMP.sh, which is located in the SNP/script and SNP/script_FX1 directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

```
%> TypeIOpenMP.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10
```

Command-line options are as follows:

$1:	The first data file, as the case population
$2:	The second data file, as the control population
$3:	Output file
$4:	Haplotype block option
	0: User-defined haplotype blocks
	1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score
	0: Pearson score
	1: Fst
$7:	Number of repeats
$8:	Number of MCMC generations
$9:	Input data format
	0: HapMap data
	1: Haplotype data
	2: Phased HapMap data
$10:	Number of threads that run in parallel

Running the MPI+OpenMP parallel version of the TypeI program using MCMC in C

To run the MPI+OpenMP version of the MCMC method, use TypeIHybrid.sh, which is located in the SNP/script and SNP/script_FX1 directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

```
%> TypeIHybrid.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10 $11
```

Command-line options are as follows:

$1:	The first data file, as the case population
$2:	The second data file, as the control population
$3:	Output file
$4:	Haplotype block option
	0: User-defined haplotype blocks
	1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score
	0: Pearson score
	1: Fst
$7:	Number of repeats
$8:	Number of MCMC generations
$9:	Input data format
	0: HapMap data
	1: Haplotype data
	2: Phased HapMap data
$10:	Number of processing units that run in parallel
$11:	Number of threads that run in parallel

Running the non-parallel version of the TypeI program using MCMC in C

To run the non-parallel version of the TypeI program using the MCMC method in C, use TypeI.sh, which is located in the SNP/script and SNP/script_FX1 directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

```
%>  TypeI.sh $1 $2 $3 $4 $5 $6 $7 $8 $9
```

Command-line options are as follows:

$1:	The first data file, as the case population	
$2:	The second data file, as the control population	
$3:	Output file	
$4:	Haplotype block option	
	0:	User-defined haplotype blocks
	1:	Blocks with equal number of SNPs
$5:	Haplotype block file	
$6:	Output score	
	0:	Pearson score
	1:	Fst
$7:	Number of repeats	
$8:	Number of MCMC generations	
$9:	Input data format	
	0:	HapMap data
	1:	Haplotype data
	2:	Phased HapMap data

Calculation of type I error rates using the MCMC/RAT method on K-computer or RIKEN RICC

The TypeI programs are used to asymptotically calculate the probability of type I errors. This set uses the MCMC/RAT method. This package includes several versions of the TypeI programs using the MCMC/RAT method on a K-computer or a RIKEN RICC: (i) the MPI parallel version in C, (ii) the OpenMP parallel version in C, (iii) the MPI+OpenMP parallel version in C, and (iv) the non-parallel version in C.

Running the MPI parallel version of the TypeI program using MCMC/RAT in C

To run the MPI parallel version of the TypeI program using the MCMC/RAT method in C, use TypeIFast.sh, which is located in the SNP/script_RICC and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

For RIKEN RICC:

```
%> TypeIFast.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10 $11
```

For K-computer:

```
%> TypeIFast.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10 $11 $12
```

Command-line options are as follows:

$1:	The first data file, as the case population
$2:	The second data file, as the control population
$3:	Output file
$4:	Haplotype block option 0: User-defined haplotype blocks 1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score 0: Pearson score 1: Fst
$7:	Number of repeats
$8:	Number of MCMC generations
$9:	Input data format 0: HapMap data 1: Haplotype data 2: Phased HapMap data

$10:	Calculation Type
	0: RAT Method
	1: MCMC Method
$11:	Number of processing units that run in parallel
$12:	Number of nodes that run in parallel (For the K-computer only)

Running the OpenMP parallel version of the TypeI program using MCMC/RAT in C

To run the OpenMP version of the TypeI program using the MCMC/RAT method in C, use TypeIOpenMP.sh, which is located in the SNP/script_RICC and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

For RIKEN RICC:
```
%> TypeIOpenMP.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10 $11
```

For K-computer:
```
%> TypeIOpenMP.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10 $11 $12
```

Command-line options are as follows:

$1:	The first data file, as the experimental population
$2:	The second data file, as the control population
$3:	Output file
$4:	Haplotype block option
	0: User-defined haplotype blocks
	1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score
	0: Pearson score
	1: Fst
$7:	Number of repeats
$8:	Number of MCMC generations
$9:	Input data format
	0: HapMap data
	1: Haplotype data
	2: Phased HapMap data
$10:	Calculation Type

	0:	RAT Method
	1:	MCMC Method
$11:	Number of threads	
$12:	Number of nodes that run in parallel (For the K-computer only)	

Running the MPI+OpenMP parallel version of the TypeI program using MCMC/RAT in C

To run the MPI+OpenMP parallel version of the TypeI program using the MCMC/RAT method in C, use TypeIHybridR.sh, which is located in the SNP/script_RICC and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

For RIKEN RICC:
```
%> TypeIHybridR.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10 $11 $12
```

For K-computer:
```
%> TypeIHybridR.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10 $11 $12 $13
```

Command-line options are as follows:

$1:	The first data file, as the case population
$2:	The second data file, as the control population
$3:	Output file
$4:	Haplotype block option 0: User-defined haplotype blocks 1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score 0: Pearson score 1: Fst
$7:	Number of repeats
$8:	Number of MCMC generations
$9:	Input data format 0: HapMap data 1: Haplotype data 2: Phased HapMap data
$10:	Calculation Type 0: RAT Method 1: MCMC Method
$11:	Number of processing units that run in parallel
$12:	Number of threads
$13:	Number of nodes that run in parallel (For the K-computer only)

Running the non-parallel version of the TypeI program using MCMC/RAT in C

To run the non-parallel version of the TypeI program using the MCMC/RAT method, use TypeI.sh, which is located in the SNP/script_RICC and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

```
%>  TypeI.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10
```

Command-line options

$1:	The first data file, as the case population
$2:	The second data file, as the control population
$3:	Output file
$4:	Haplotype block option
	0: User-defined haplotype blocks
	1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score
	0: Pearson score
	1: Fst
$7:	Number of repeats
$8:	Number of MCMC generations
$9:	Input data format
	0: HapMap data
	1: Haplotype data
	2: Phased HapMap data
$10:	Calculation Type
	0: RAT Method
	1: MCMC Method

Dividing files for the calculation of type I error rates using the MCMC/RAT method

To divide files needed for the calculation of type I error rates using the MCMC/RAT method, use typeIDivFile.exe, which is located in the SNP/bin directory. This function is available only for PC clusters and the K-computer.

Usage:

For PC cluster and K-computer:
```
%>  typeIDivFile.exe $1 $2 $3 $4 $5 $6 $7
```

Command-line options

$1:	Data type 0: HapMap 1: Haplotype data 2: Phasing HapMap
$2:	The first input data file
$3:	The second input data file
$4:	Haplotype block option 0: User-defined haplotype blocks 1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	File divide number
$7:	Output file

To create a batch file, use batch_typeParallelD_ricc.exe, which is located in the SNP/bin directory.

Usage:

For PC cluster, RIKEN FX1, RIKEN RICC:
```
%>batch_typeParallelD_ricc.exe $1 $2 $3 $4 $5 $6 $7 $8 $9
```

For K-computer:
```
%>batch_typeParallelD_ricc.exe $1 $2 $3 $4 $5 $6 $7 $8 $9 $10
```

Command-line options

$1:	Exec file (located in SNP/bin directory)
$2:	Number of processes that run in parallel
$3:	Generated XML file
$4:	Calculation method

	0: Pearson score
	1: Fst
$5:	Number of repeats
$6:	Number of MCMC generations
$7:	Output file
$8:	Batch script name
$9:	Calculation type
	0: RAT Method
	1: MCMC Method
$10:	Number of nodes that run in parallel (For the K-computer only)

Example:

For PC cluster, RIKEN FX1, RIKEN RICC:

```
%>  ./batch_typeIParallelD_ricc.exe  ./typeIParallelDFast.exe  1
/home/riken/test.xml 0 10 10 ./out.txt./GO.sh 0
%> qsub ./GO.sh
```

For K-computer:

```
%>  ./batch_typeIParallelD_ricc.exe  ./typeIParallelDFast.exe  1
/home/riken/test.xml 0 10 10 ./out.txt./GO.sh 0 1
%> qsub ./GO.sh
```

Exact calculation of type I error rates

Running the non-parallel version of the exact calculation of type I error rates in C

To run the non-parallel version to obtain the exact calculation of type I error rates, use Exact.sh, which is located in the SNP/script, SNP/script_RICC, SNP/script_FX1, and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

```
%> Exact.sh $1 $2 $3 $4 $5 $6 $7
```

Command-line options:

$1:	The first data file, as the case population
$2:	The second data file, as the control population

$3:	Output file
$4:	Haplotype block option
	0: User-defined haplotype blocks
	1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score
	0: Pearson score
	1: Fst
$7:	Input data format
	0: HapMap data
	1: Haplotype data
	2: Phased HapMap data

Permutation test using the RAT algorithm (5)

This permutation test calculates the P value using the RAT algorithm. This package includes the following versions of the permutation test: (i) the MPI parallel version in C, (ii) the MPI+OpenMP parallel version in C, and (iii) the non-parallel version in C.

Running the MPI parallel version of the permutation test using RAT in C

To run the MPI parallel version of the permutation test using the RAT algorithm in C, use RatParallel.sh, which is located in the SNP/script, SNP/script_RICC, SNP/script_FX1, and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

For PC cluster, RIKEN FX1, RIKEN RICC:

```
%> RatParallel.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10
```

For K-computer:

```
%> RatParallel.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10 $11
```

Command-line options:

$1:	The first data file, as the case population
$2:	The second data file, as the control population
$3:	Output file
$4:	Haplotype block option 0: User-defined haplotype blocks 1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score 0: Pearson score 1: Fst
$7:	Number of MCMC generations
$8:	Number of burning generations
$9:	Input data format 0: HapMap data 1: Haplotype data 2: Phased HapMap data
$10:	Number of processing units that run in parallel
$11:	Number of nodes that run in parallel (For the K-computer only)

Running the MPI+OpenMP parallel version of the permutation test using RAT in C

To run the MPI+OpenMP parallel version of the permutation test using the RAT algorithm in C, use RatHybrid.sh, which is located in the SNP/script_RICC and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

For RIKEN RICC:

```
%> RatHybrid.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10 $11
```

For K-computer:

```
%> RatHybrid.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10 $11 $12
```

Command-line options:

$1:	The first data file, as the case population
$2:	The second data file, as the control population
$3:	Output file
$4:	Haplotype block option 0: User-defined haplotype blocks 1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score 0: Pearson score 1: Fst
$7:	Number of MCMC generations
$8:	Number of burning generations
$9:	Input data format 0: HapMap data 1: Haplotype data 2: Phased HapMap data
$10:	Number of processing units that run in parallel
$11:	Number of thread
$12:	Number of nodes that run in parallel (For the K-computer only)

Running the non-parallel version of the permutation test using RAT in C

To run the non-parallel version of the permutation test using the RAT algorithm in C, use Rat.sh, which is located in the SNP/script, SNP/script_RICC, SNP/script_FX1, and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

```
%> Rat.sh $1 $2 $3 $4 $5 $6 $7 $8 $9
```

Command-line options:

$1:	The first data file, as the case population
$2:	The second data file, as the control population
$3:	Output file
$4:	Haplotype block option
	0: User-defined haplotype blocks
	1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score
	0: Pearson score
	1: Fst
$7:	Number of MCMC generations
$8:	Number of burning generations
$9:	Input data format
	0: HapMap data
	1: Haplotype data
	2: Phased HapMap data

Standard permutation test (SPT)

This permutation test calculates the P value using the standard permutation algorithm. This package includes the following versions of the permutation test: (i) the MPI parallel version in C, and (ii) the non-parallel version in C.

Running the MPI parallel version of the standard permutation test in C

To run the MPI parallel version of the standard permutation test in C, use PrimitiveParallel.sh, which is located in the SNP/script, SNP/script_RICC, SNP/script_FX1, and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

For PC cluster, RIKEN FX1, RIKEN RICC:

```
%> PrimitiveParallel.sh $1 $2 $3 $4 $5 $6 $7 $8 $9
```

For K-computer:

```
%> PrimitiveParallel.sh $1 $2 $3 $4 $5 $6 $7 $8 $9 $10
```

Command-line options:

$1:	The first data file, as the case population
$2:	The second data file, as the control population
$3:	Output file
$4:	Haplotype block option
	0: User-defined haplotype blocks
	1: Blocks with equal number of SNPs
$5:	Haplotype block file
$6:	Output score
	0: Pearson score
	1: Fst
$7:	Number of repeats
$8:	Input data format
	0: HapMap data
	1: Haplotype data
	2: Phased HapMap data
$9:	Number of processing units that run in parallel
$10:	Number of nodes that run in parallel (For the K-computer only)

Running the non-parallel version of the standard permutation test in C

To run the non-parallel version of the standard permutation test in C, use Primitive.sh, which is located in the SNP/script, SNP/script_RICC, SNP/script_FX1, and SNP/script_K directories. If this script is run outside these directories, the path must be set appropriately.

Usage:

```
%> Primitive.sh $1 $2 $3 $4 $5 $6 $7 $8
```

Command-line options:

$1:	The first data file, as the case population		
$2:	The second data file, as the control population		
$3:	Output file		
$4:	Haplotype block option		
	0:	User-defined haplotype blocks	
	1:	Blocks with equal number of SNPs	
$5:	Haplotype block file		
$6:	Output score		
	0:	Pearson score	
	1:	Fst	
$7:	Number of repeats		
$8:	Input data format		
	0:	HapMap data	
	1:	Haplotype data	
	2:	Phased HapMap data	

Data formats of input and output files

ParaHaplo supports both the HapMap data format (7) and the BioBank Japan data format (2). The example data files are included in the SNP/data directory.

Data conversion from BioBank Japan format to HapMap format

To convert the genotype file from the BioBank Japan format to the HapMap format, use illumina2hapmap.exe, which is located in the SNP/bin directory.

Usage:

```
%> illumina2hapmap.exe $1 $2 $3
```

Command-line options:

$1:	Data file in BioBank Japan style
$2:	Information file of SNPs
$3:	Output data file in HapMap format

Data conversion from PED/MAP format to HapMap format

To convert the genotype file from the PED/MAP format to the HapMap format, use pedmap2hapmap.exe, which is located in the SNP/bin directory.

Usage:

```
%> pedmap2hapmap.exe $1 $2
```

Command-line options:

$1:	Input PED/MAP file
$2:	Output data file in HapMap format

Haplotype block file

In this program package, the haplotype-block file could be defined by specifying the haplotype block boundaries or the sliding window parameters.

Haplotype block boundaries

The boundaries of the haplotype blocks could be listed in the haplotype-block file to define haplotype blocks that do not overlap. The last line must be empty. The following is an example of a haplotype-block file with block boundaries:

```
14000000
15000000
16000000
17000000

48000000
49000000
50000000
```

Sliding window parameters

Instead of specifying haplotype block boundaries, the window size and the step size could be specified in the haplotype-block file. The first line of the file must be the window size, and the second line must be the step size. Both are measured by the number of SNPs. If the step size is smaller than the window size, each window is overlapping. If the step size is larger than the window size, SNPs between windows are ignored. If each SNP must be analyzed separately, both window size and step size must be set to 1. The third line must be an empty string.

The following is an example of a haplotype-block file with sliding window parameters, where the window size is 100 and the step size is 200:

```
 100
 200
```

Output file

The programs within the program package return the result of analyses in a similar format. Each output file consists of two sections: the header and the result. The differences among the output files of different versions of each program are in the header section.

The header section

The first and second lines of the header section show the names of the case data file (CaseData) and the control data file (ControlData). The third line shows the number of runs of the MCMC chains. The fourth line shows the number of generations of the MCMC chains.

The result section

The result section shows the results of the analyses. The first column shows the range of the haplotype block within the chromosome, and the second column shows the number of SNPs within that haplotype block. The remaining columns provide information about the SNP with the highest score in that block. The third, fourth, and fifth columns show its rs number, its position within the chromosome, and its score, respectively. The sixth column shows the type I error. If no polymorphic site is found in the haplotype block, the program displays “NoData.”

```
CaseData       = /phased/JPT/chr22.txt
ControlData    = /phased/CHB/chr22.txt
Repeat         = 4
Generation     = 100000
BlockArea        SNPNum   rsNumber Position  Score         Type I error
14657412-14893245  100  rsxxxxx1  14870204  7.9158529679   0.0161200000

14976512-16148952  100  rsxxxxx2  15853229  10.0341711103  0.4292800000

16485214-17065485  100  rsxxxxx3  16643268  16.3205869262  0.0189700000

17165489-18221578  100  NoData
18345862-19147853  100  rsxxxxx5  18582729  28.5802577195  0.0000000000

19348562-19965482  100  rsxxxxx6  19660266  14.2802131465  0.0318300000
```

Calculating the global P value

The data for case (experimental) and control populations are typically separated into one chromosome per file; therefore, the global P value must be calculated by gathering the results from all chromosomes. To do so, use global.awk, which can be found in the SNP/Tools directory.

Usage:

```
%> gawk –f global.awk datafile1.txt datafile2.txt … >all.txt
```

Acknowledgment

We thank Dr. Yumi Yamaguchi-Kabata for her kind comments on the manuscript.

References

1. Hirschhorn, J.N. and Daly, M.J. (2005) Genome-wide association studies for common diseases and complex traits. *Nat Rev Genet*, **6**, 95-108.
2. Nakamura, Y. (2007) The BioBank Japan Project. *Clin Adv Hematol Oncol*, **5**, 696-697.
3. Yamaguchi-Kabata, Y., Nakazono, K., Takahashi, A., Saito, S., Hosono, N., Kubo, M., Nakamura, Y. and Kamatani, N. (2008) Japanese population structure, based on SNP genotypes from 7003 individuals compared to other ethnic groups: effects on population-based association studies. *Am J Hum Genet*, **83**, 445-456.
4. Misawa, K., Fujii, S., Yamazaki, T., Takahashi, A., Takasaki, J., Yanagisawa, M., Ohnishi, Y., Nakamura, Y. and Kamatani, N. (2008) New correction algorithms for multiple comparisons in case-control multilocus association studies based on haplotypes and diplotype configurations. *J Hum Genet*, **53**, 789-801.
5. Kimmel, G. and Shamir, R. (2006) A fast method for computing high-significance disease association in large population-based studies. *Am J Hum Genet*, **79**, 481-492.
6. Misawa, K. and Kamatani, N. (submitted) ParaHaplo: A program package for whole-genome association study using parallel computing
7. The International HapMap Consortium. (2005) A haplotype map of the human genome. *Nature*, **437**, 1299-1320.
8. Li, Y. and Abecasis, G.R. (2008) Mach 1.0: rapid haplotype reconstruction and missing genotype inference. *Am J Hum Genet*, **S79**, 2290.
9. Stephens, M., Smith, N.J. and Donnelly, P. (2001) A new statistical method for haplotype reconstruction from population data. *Am J Hum Genet*, **68**, 978-989.
10. Clayton, D. (2005).