Published online 3 October 2019

NAR Genomics and Bioinformatics, 2020, Vol. 2, No. 1 **1**
doi: 10.1093/nargab/lqz009

GPU empowered pipelines for calculating genome-wide kinship matrices with ultra-high dimensional genetic variants and facilitating 1D and 2D GWAS

Wenchao Zhang[1], Xinbin Dai[1], Shizhong Xu[2,*] and Patrick X. Zhao[1,*]

[1]Noble Research Institute, LLC, 2510 Sam Noble Parkway, Ardmore, OK 73401, USA and [2]Department of Botany and Plant Sciences, University of California, Riverside, CA 92521, USA

Received May 31, 2019; Revised August 22, 2019; Editorial Decision September 16, 2019; Accepted September 25, 2019

ABSTRACT

Genome-wide association study (GWAS) is a powerful approach that has revolutionized the field of quantitative genetics. Two-dimensional GWAS that accounts for epistatic genetic effects needs to consider the effects of marker pairs, thus quadratic genetic variants, compared to one-dimensional GWAS that accounts for individual genetic variants. Calculating genome-wide kinship matrices in GWAS that account for relationships among individuals represented by ultra-high dimensional genetic variants is computationally challenging. Fortunately, kinship matrix calculation involves pure matrix operations and the algorithms can be parallelized, particular on graphics processing unit (GPU)-empowered high-performance computing (HPC) architectures. We have devised a new method and two pipelines: KMC1D and KMC2D for kinship matrix calculation with high-dimensional genetic variants, respectively, facilitating 1D and 2D GWAS analyses. We first divide the ultra-high-dimensional markers and marker pairs into successive blocks. We then calculate the kinship matrix for each block and merge together the block-wise kinship matrices to form the genome-wide kinship matrix. All the matrix operations have been parallelized using GPU kernels on our NVIDIA GPU-accelerated server platform. The performance analyses show that the calculation speed of KMC1D and KMC2D can be accelerated by 100–400 times over the conventional CPU-based computing.

INTRODUCTION

Due to the great success in identifying causal single-nucleotide polymorphisms (SNPs) conferring complex traits or diseases (1), the genome-wide association study (GWAS) has revolutionized quantitative genetics (2). The data and results generated by GWAS have led to remarkable discoveries, including the development of evolutionary biology, annotation of gene functions, understanding of disease mechanisms, and translation toward new clinic diagnosis and therapeutics (3). In the original GWAS, individuals were assumed to be independent and GWAS populations were assumed to be homogeneous (4,5). However, individuals within a GWAS population are always related to a certain degree, and the population may be heterogeneous due to a mixture of multiple genetically distinct subgroups (6). Population structure and cryptic relatedness between individuals are widespread confounding factors and will cause spurious associations. If not properly controlled, these confounding factors will generate misleading results, which is very deleterious during the GWAS analysis (7). Therefore, a fine association mapping analysis requires careful selection of individuals to exclude obvious subgroups and needs to account for the relatedness among individuals (8). At present, the unified K+Q linear mixed model (LMM) has emerged as a popular method for simultaneously accounting for population structure and cryptic relatedness (4,9). To solve the LMM problem, a kinship matrix, which is used to describe the similarities between pairs of individuals, must be mathematically defined (10,11). There are two distinct and related definitions of kinship matrix. One is the matrix holding the coancestry coefficients between pairs of individuals that can be calculated based on pedigree information. This is an identical-by-descent (IBD) based kinship matrix (12). The other is based on marker genotypes instead of the pedigree information and is known as an identical-by-state (IBS) based kinship matrix (13–15). It has been re-

*To whom correspondence should be addressed. Tel: +1 580 224 6725; Email: pzhao@noble.org
Correspondence may also be addressed to Shizhong Xu. Tel: +1 951 827 5898; Email: shizhong.xu@ucr.edu

ported that the marker-based kinship matrix represents the realized genetic relationships between individuals and thus is more accurate than the pedigree-based kinship matrix (8).

The widely used K+Q LMM considers only the additive genetic effects, which may only account for a fraction of the heritability (16). An important factor that accounts for the missing heritability is the comprehensive genetic architecture of the trait, i.e. epistatic effects defined as gene-by-gene interactions (GxG) and gene-by-environment interactions (GxE) (8,17,18). To accurately dissect the phenotypic variance and address the G × G and G × E interaction effects, Xu *et al.* proposed several novel LMMs that incorporate multiple polygenic covariance structures (19), and several related tools have been developed (20,21). Again, to solve the polygenic LMMs, we need to define and calculate kinship matrices for polygenic epistatic effects.

Advances in high-throughput sequencing technology make it possible to generate ultra-high-dense SNP maps (e.g. 10 million scale) (22). The high density markers allow us to calculate accurate kinship matrices for high-resolution GWAS (23). Statistic power is defined as the probability to reject a null hypothesis (H_0), while the alternative hypothesis (H_A) is true. The GWAS statistical power is primarily determined by the sample size: the larger the sample size, the higher the power (24).

Kinship matrix calculation primarily involves matrix multiplication, which is complex (19). If n is the sample size and m is the number of markers, the main effect kinship matrix calculation mathematically follows a complexity of $O(mn^2)$. This means that there is a huge computational burden to calculate the main effect kinship matrix when both the sample size and the number of markers are large. In the epistasis analysis, it is the marker pairs instead of markers that are involved for kinship matrix calculation. Therefore, the polygenic kinship matrix mathematically follows a complexity of (m^2n^2) (20,21). This means that we need to overcome a huge computational burden in polygenic kinship matrix calculation for a 2D GWAS analysis, even with a moderate number of markers.

During the past decade, Graphics Processing Units (GPUs), typically containing thousands of hardware processor cores, have rapidly evolved to become a standard high-performance accelerators for large-scale data computing (25). Because of its parallelization featured as data-driven architecture, and capability to compute large-scale data, the GPU-empowered high-performance computing (HPC) platform is particularly suitable for large-scale matrix operations, including addition, multiplication and convolution (26).

We studied the mathematical principle and the complexity of computing the marker-inferred main effect kinship matrix and marker-pair-inferred epistatic effect kinship matrix. We found that the complexity to calculate a kinship matrix is linear on the number of markers or marker pairs. Such linearity allows us to partition the large-scale genotypic data into blocks, so a sub-kinship matrix can be computed from each block and eventually obtain the final form of a kinship matrix by integrating all the sub-kinship matrices. Based on this fundamental analysis, we have successfully developed two web-based GPU-empowered pipelines: KMC1D for main effect kinship matrix calculation and KMC2D for epistatic effect kinship matrix calculation. Both operate at up to a hundred times the speed of the gold standard counterpart (the CPU-based sequential calculation).

Specifically, users only need to provide the required genotype marker matrix data files. The KMC1D/KMC2D will upload the files, parse the parameters, partition the whole markers or marker pairs into blocks, and call the GPU kernels for parallel computing and integrating all of the sub-kinship matrices into a full kinship matrix. For KMC1D, the genotype data may contain millions of markers and thousands of individuals so that the storage may quickly reach hundreds of GBs. Uploading such a big file can be difficult. To solve this problem, we took advantage of HTML5 and developed a multi-threading resumable file uploading module to upload very large genotype data.

MATERIALS AND METHODS

LMM, kinship matrices, 1D and 2D GWAS

The genetic effects of markers and the phenotypic values of quantitative traits can be connected through a kind of LMM. The software engineering an LMM usually can produce a specific GWAS analysis package. The additive effect and dominance effect can be conducted through 1D GWAS (9,10), and the canonical LMM model is

$$\boldsymbol{y} = \boldsymbol{X\beta} + \sum_{i=1}^{m} \widehat{\boldsymbol{Z}}_{\boldsymbol{i}} a_i + \sum_{i=1}^{m} \widehat{\boldsymbol{W}}_{\boldsymbol{i}} d_i + \boldsymbol{e} \qquad (1)$$

where $\boldsymbol{y}$ is an $n \times 1$ vector of phenotypic values with n being the sample size; $\boldsymbol{X\beta}$ represents some systematic nongenetic effects (year effect, location effect and so on); $\boldsymbol{Z}$ and $\boldsymbol{W}$ are $m \times n$ genotype matrices with m being the marker size; $\widehat{\boldsymbol{Z}}_{\boldsymbol{i}}$ is an $n \times 1$ vector of additive effect indicator variables for all individuals at marker i, and the a_i is the additive genetic effect of marker i; $\widehat{\boldsymbol{W}}_{\boldsymbol{i}}$ is a $n \times 1$ vector of dominance effect indicator variables for all individuals at marker i, and the d_i is the dominance genetic effect of marker i; and $\boldsymbol{e}$ is an $n \times 1$ vector of residual errors. If the two alleles of a diploid individual at one specific locus are represented as A_1 and A_2, the additive effect indicator variable for individual j at marker i can be defined as $Z_{ji} = 1$ for A_1A_1, $Z_{ji} = 0$ for A_1A_2 and $Z_{ji} = -1$ for A_2A_2; the dominance effect indicator variance can be defined as $W_{ji} = 1$ for A_1A_2 and $W_{ji} = 0$ for A_1A_1 or A_2A_2. Assume that $a_i \sim N(0, \sigma_a^2)$ and $d_i \sim N(0, \sigma_d^2)$ for all $i = 1, \cdots, m$, where σ_a^2 and σ_d^2 are the polygenic additive variance and the polygenic dominance variance. The expectation of the model is $E(\boldsymbol{y}) = \boldsymbol{X\beta}$ and the variance of the model is

$$Var\ (\boldsymbol{y}) = \boldsymbol{K_a}\,\sigma_a^2 + \boldsymbol{K_d}\sigma_d^2 + \boldsymbol{I}\sigma^2 \qquad (2)$$

where $\boldsymbol{K_a}$ and $\boldsymbol{K_d}$ are $n \times n$ marker inferred kinship matrices for the additive and dominance effects, respectively. Formulas to calculate the main effect kinship matrices are presented later.

To conduct the epistatic effect GWAS, Xu (19) proposed the following polygenic model:

$$\begin{aligned}\boldsymbol{y} = \boldsymbol{X\beta} + \sum_{i=1}^{m} \widehat{\boldsymbol{Z}}_i a_i + \sum_{i=1}^{m} \widehat{\boldsymbol{W}}_i d_i + \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \left(\widehat{\boldsymbol{Z}}_i \# \widehat{\boldsymbol{Z}}_j\right)(aa)_{ij} \\ + \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \left(\widehat{\boldsymbol{Z}}_i \# \widehat{\boldsymbol{W}}_j\right)(ad)_{ij} \\ + \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \left(\widehat{\boldsymbol{W}}_i \# \widehat{\boldsymbol{Z}}_j\right)(da)_{ij} \\ + \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \left(\widehat{\boldsymbol{W}}_i \# \widehat{\boldsymbol{W}}_j\right)(dd)_{ij} + \boldsymbol{e} \end{aligned} \quad (3)$$

The model contains two types of main effects (additive and dominance), and four types of epistatic effects (additive-additive, additive-dominance, dominance-additive and dominance-dominance). These effects are denoted by a_i, d_i, $(aa)_{ij}$, $(ad)_{ij}$, $(da)_{ij}$ and $(dd)_{ij}$. In addition, $\widehat{\boldsymbol{Z}}_i \# \widehat{\boldsymbol{W}}_j$ represents element-wise two $n \times 1$ vectors' multiplication, which will generate another $n \times 1$ vector. The expectation of the model is $E(\boldsymbol{y}) = \boldsymbol{X\beta}$ and the variance of the above epistatic model is

$$\begin{aligned} Var\,(\boldsymbol{y}) = \boldsymbol{K_a}\,\sigma_a^2 + \boldsymbol{K_d}\sigma_d^2 + \boldsymbol{K_{aa}}\sigma_{aa}^2 + \boldsymbol{K_{ad}}\sigma_{ad}^2 \\ + \boldsymbol{K_{da}}\sigma_{da}^2 + \boldsymbol{K_{dd}}\sigma_{dd}^2 + \boldsymbol{I}\sigma^2 \end{aligned} \quad (4)$$

where $\boldsymbol{K_a}$ and $\boldsymbol{K_d}$ are the two main effect kinship matrices while $\boldsymbol{K_{aa}}$, $\boldsymbol{K_{ad}}$, $\boldsymbol{K_{da}}$ and $\boldsymbol{K_{dd}}$ are the four epistatic effect kinship matrices. Based on the polygenic LMM, we have successfully developed a pipeline, PEPIS (20), for the polygenic epistatic effect analysis, which essentially includes main effect 1D GWAS and epistatic effect 2D GWAS.

The GWAS using the canonical LMM or the polygenic LMM requires these kinship matrices, which allow us to dissect a complex phenotype into several meaningful genetic components and guide the genome-wide marker association mapping. The additive and dominance genotype indicator matrices are denoted by $\boldsymbol{Z}$ and $\boldsymbol{W}$. The main effect kinship matrices are defined by

$$\begin{cases} \boldsymbol{K_a^*} = \boldsymbol{Z}^t\ \boldsymbol{Z} \\ C_a = mean\left(diag\left(\boldsymbol{K_a^*}\right)\right) \\ \boldsymbol{K_a} = \left({}^{1}/_{C_a}\right)\boldsymbol{K_a^*} \end{cases} \quad (5)$$

$$\begin{cases} \boldsymbol{K_d^*} = \boldsymbol{W}^t\ \boldsymbol{W} \\ C_d = mean\left(diag\left(\boldsymbol{K_d^*}\right)\right) \\ \boldsymbol{K_d} = \left({}^{1}/_{C_d}\right)\boldsymbol{K_d^*} \end{cases} \quad (6)$$

To develop the four types of epistatic effect kinship matrices, we need to generate four types of element-wise marker pairs according to Equations (7–10):

$$\begin{aligned}\boldsymbol{U_{aa}} = \left[\widehat{\boldsymbol{Z}}_1\#\widehat{\boldsymbol{Z}}_2; \widehat{\boldsymbol{Z}}_1\#\widehat{\boldsymbol{Z}}_3; \ldots; \widehat{\boldsymbol{Z}}_1\#\widehat{\boldsymbol{Z}}_m;\right. \\ \left.\widehat{\boldsymbol{Z}}_2\#\widehat{\boldsymbol{Z}}_3; \ldots; \widehat{\boldsymbol{Z}}_2\#\widehat{\boldsymbol{Z}}_m; \ldots; \widehat{\boldsymbol{Z}_{m-1}}\#\widehat{\boldsymbol{Z}}_m\right] \end{aligned} \quad (7)$$

$$\begin{aligned}\boldsymbol{U_{ad}} = \left[\widehat{\boldsymbol{Z}}_1\#\widehat{\boldsymbol{W}}_2; \widehat{\boldsymbol{Z}}_1\#\widehat{\boldsymbol{W}}_3; \ldots; \widehat{\boldsymbol{Z}}_1\#\widehat{\boldsymbol{W}}_m;\right. \\ \left.\widehat{\boldsymbol{Z}}_2\#\widehat{\boldsymbol{W}}_3; \ldots; \widehat{\boldsymbol{Z}}_2\#\widehat{\boldsymbol{W}}_m; \ldots; \widehat{\boldsymbol{Z}_{m-1}}\#\widehat{\boldsymbol{W}}_m\right] \end{aligned} \quad (8)$$

$$\begin{aligned}\boldsymbol{U_{da}} = \left[\widehat{\boldsymbol{W}}_1\#\widehat{\boldsymbol{Z}}_2; \widehat{\boldsymbol{W}}_1\#\widehat{\boldsymbol{Z}}_3; \ldots; \widehat{\boldsymbol{W}}_1\#\widehat{\boldsymbol{Z}}_m;\right. \\ \left.\widehat{\boldsymbol{W}}_2\#\widehat{\boldsymbol{Z}}_3; \ldots; \widehat{\boldsymbol{W}}_2\#\widehat{\boldsymbol{Z}}_m; \ldots; \widehat{\boldsymbol{W}_{m-1}}\#\widehat{\boldsymbol{Z}}_m\right] \end{aligned} \quad (9)$$

$$\begin{aligned}\boldsymbol{U_{dd}} = \left[\widehat{\boldsymbol{W}}_1\#\widehat{\boldsymbol{Z}}_2; \widehat{\boldsymbol{W}}_1\#\widehat{\boldsymbol{W}}_3; \ldots; \widehat{\boldsymbol{W}}_1\#\widehat{\boldsymbol{W}}_m;\right. \\ \left.\widehat{\boldsymbol{W}}_2\#\widehat{\boldsymbol{W}}_3; \ldots; \widehat{\boldsymbol{W}}_2\#\widehat{\boldsymbol{W}}_m; \ldots; \widehat{\boldsymbol{W}_{m-1}}\#\widehat{\boldsymbol{W}}_m\right] \end{aligned} \quad (10)$$

From these pair-wise genotype indicator variables, the epistatic effect kinship matrices can be calculated by

$$\begin{cases} \boldsymbol{K_{aa}^*} = \boldsymbol{U_{aa}^t}\ \boldsymbol{U_{aa}} \\ C_{aa} = mean\left(diag\left(\boldsymbol{K_{aa}^*}\right)\right) \\ \boldsymbol{K_{aa}} = \left({}^{1}/_{C_{aa}}\right)\boldsymbol{K_{aa}^*} \end{cases} \quad (11)$$

$$\begin{cases} \boldsymbol{K_{ad}^*} = \boldsymbol{U_{ad}^t}\ \boldsymbol{U_{ad}} \\ C_{ad} = mean\left(diag\left(\boldsymbol{K_{ad}^*}\right)\right) \\ \boldsymbol{K_{ad}} = \left({}^{1}/_{C_{ad}}\right)\boldsymbol{K_{ad}^*} \end{cases} \quad (12)$$

$$\begin{cases} \boldsymbol{K_{da}^*} = \boldsymbol{U_{da}^t}\ \boldsymbol{U_{da}} \\ C_{da} = mean\left(diag\left(\boldsymbol{K_{da}^*}\right)\right) \\ \boldsymbol{K_{da}} = \left({}^{1}/_{C_{da}}\right)\boldsymbol{K_{da}^*} \end{cases} \quad (13)$$

$$\begin{cases} \boldsymbol{K_{dd}^*} = \boldsymbol{U_{dd}^t}\ \boldsymbol{U_{dd}} \\ C_{dd} = mean\left(diag\left(\boldsymbol{K_{dd}^*}\right)\right) \\ \boldsymbol{K_{dd}} = \left({}^{1}/_{C_{dd}}\right)\boldsymbol{K_{dd}^*} \end{cases} \quad (14)$$

From the above formulas, we can estimate the computational complexities for computing these kinship matrices. Let m be the number of markers and n be the sample size. The number of multiplications for a main effect kinship matrix is of order $O(mn^2)$. The number of multiplications for an epistatic kinship matrix is of order $O(m^2n^2)$. Therefore, we can compute a main effect kinship matrix from millions of markers with tens of thousands of individuals. However, we can only handle about tens of thousands of markers and individuals for an epistatic effect kinship matrix. As a result, some strategies to speed up the kinship matrix calculation are badly needed.

Calculate main effect kinship matrix by partitioning high-dimensional markers into blocks

After SNP and genetic variant calling (27,28), the genotype at one specific locus is represented by AA, Aa and aa for one homozygote, the heterozygote and the other homozygote, respectively. The three genotypes are then numerically coded by −1, 0, 1 or 0, 1, 2 for the additive effect and 0, 1, 0 for the dominance effect. From Equations (5) and (6), we can easily calculate the two main effect kinship matrices, e.g. K_a and K_d. With the high-throughput sequencing technology, we can easily generate millions of SNP data. The

large volume of data is hard to handle using traditional matrix multiplications. Luckily, we can take advantage of the linearity of matrix theory and partition the whole genotype data into many blocks. Kinship matrix is calculated for each block, then the block specific kinship matrices are added to generate the final kinship matrix.

Let $\boldsymbol{G}$ be a $m \times n$ matrix for the additive or dominance genotype effect, where m is the number of markers and n is the sample size. If we partition $\boldsymbol{G}$ into L blocks, and each block $\boldsymbol{G_k}$ has n individuals and m_k markers so that $m = \sum_{k=1}^{L} m_k$. The partitioned matrix can be written as $\boldsymbol{G} = [\boldsymbol{G}_1; \boldsymbol{G}_2; \ldots, \boldsymbol{G_k}; \ldots; \boldsymbol{G_L}]$, according to matrix block theory, it is easy to prove that

$$\boldsymbol{K^*} = \boldsymbol{G^t}\ \boldsymbol{G} = \begin{bmatrix} \boldsymbol{G}_1 \\ \boldsymbol{G}_2 \\ \cdots \\ \boldsymbol{G_k} \\ \cdots \\ \boldsymbol{G_L} \end{bmatrix} [\boldsymbol{G}_1; \boldsymbol{G}_2; \ldots, \boldsymbol{G_k}; \ldots; \boldsymbol{G_L}] = \sum_{k=1}^{L} \boldsymbol{G_k^t G_k} \quad (15)$$

Figure 1A illustrates the principle to generate the coded additive and dominance genotypic effect and calculate the main kinship matrix through partitioning the additive/dominance marker into blocks.

Generate all of the combinational genotype marker pairs and calculate epistatic effect kinship matrix by partitioning the marker pairs into blocks

In the polygenic LMMs, two main effects: a_i, d_i, and four epistatic effects: $(aa)_{ij}$, $(ad)_{ij}$, $(da)_{ij}$ and $(dd)_{ij}$, are defined. Main effects correspond to the marker's direct effect, while epistatic effects correspond to the combinational marker's pairs. Similar to the main effect kinship matrix calculation, we can use the corresponding marker pairs to calculate the four epistatic effect kinship matrices. Given m markers, there are $C_m^2 = m(m-1)/2$ marker pairs. Therefore, the number of marker pairs can be huge for a large m. For example, it can reach hundreds of millions for a typical number of markers, say $m = 20000$.

Again, we can partition the huge number of marker pairs into small blocks and calculate the sub-kinship matrix one by one and then merge them into a whole kinship matrix. Figure 1B illustrates the mathematical principle to generate all of the epistatic genotypic marker pairs and calculate the epistatic kinship matrix through partitioning the marker pairs into blocks.

Design and implementation

To calculate the main effect kinship matrices for high-dimensional markers and the epistatic effect kinship matrices for a typical number of markers and sample size, we take advantage of the linearity of matrix operation to partition the markers or marker pairs into blocks. Furthermore, the matrix operations for kinship matrix calculation include matrix transposition, matrix–matrix multiplication, matrix addition and matrix normalization, all of which can be parallelized on the GPU-based HPC platform.

Overall pipeline architecture

We examined the biological fundamentals to calculate the main and epistatic effect kinship matrices, and particularly investigated the mathematical linearity of matrix block operation and used it to partition the high-dimensional markers or marker pairs into blocks for efficient kinship matrix calculation. Based on these, we developed two parallel computing pipelines: KMC1D for the main effect kinship matrix calculation and KMC2D for the epistatic effect kinship matrix calculation.

The GPU parallel scheme is essentially a data-driven parallelism, which requires the developer to take the responsibility for organizing the low-level algorithm design, data structure wrapping and data streaming between CPU and GPU. Such a GPU-employed parallel architecture includes (i) the host codes sequentially running in the CPU part and (ii) the device kernel codes parallel running on the GPU part. The GPU part mainly coordinates the parallel kernel function modules, while the host part is responsible for data loading and GPU kernel launching.

KMC1D for main effect kinship matrix calculation can handle very large genotypic data in the CSV (Comma Separated Values) format. Therefore, it is not optimal to load the whole data into the memory. Instead, the KMC1D pipeline loads the marker data one block at a time. Once the uploaded marker block is full, KMC1D will launch the GPU kernel to calculate a sub-kinship matrix in a parallel manner. However, pipeline KMC2D is at a very different scenario to perform the epistatic effect kinship matrix calculation. The KMC2D pipeline deals with marker pairs, which can be huge even for a moderate number of markers. Therefore, we let KMC2D read the two types of marker matrices (Z and W) into memory and generate elementwise marker pairs one block at a time. Once a marker pair block is full, KMC2D will launch the GPU kernel to parallel calculate a sub-kinship matrix. Figure 2 illustrates the parallel architecture of GPU-empowered kinship matrix calculation.

Host part design for CPU involved serial process

As discussed above, we sequentially designed the host part by partitioning the huge genotype marker or marker pair data into blocks and feeding each block to the GPU kernel for parallel computing. We particularly considered two scenarios to handle (i) the large-scale genotype markers that are directly loaded from external files in KMC1D and (ii) the marker pairs that are combinatorically calculated from two types of related genotype effect matrix in KMC2D. We allocated the global memory variables: Z_b, K_b, K_{sum} to represent the block of marker or marker pairs, intermediate kinship matrix based on a block of markers or marker pairs and the final integrated sum kinship matrix, respectively. All of this global memory resides in device DRAM for transfers between the host and device as well as for data transfer to and from kernels. To objectively compare the performance of CPU employed parallel computing against the golden CPU serial computing, we designed three run models coded as 0, 1 or 2 to allow the whole pipeline to be run at GPU parallel, CPU golden serial, and read file and loop through mode, respectively. The CPU golden serial run mode provides a basic calculation capability, and the read

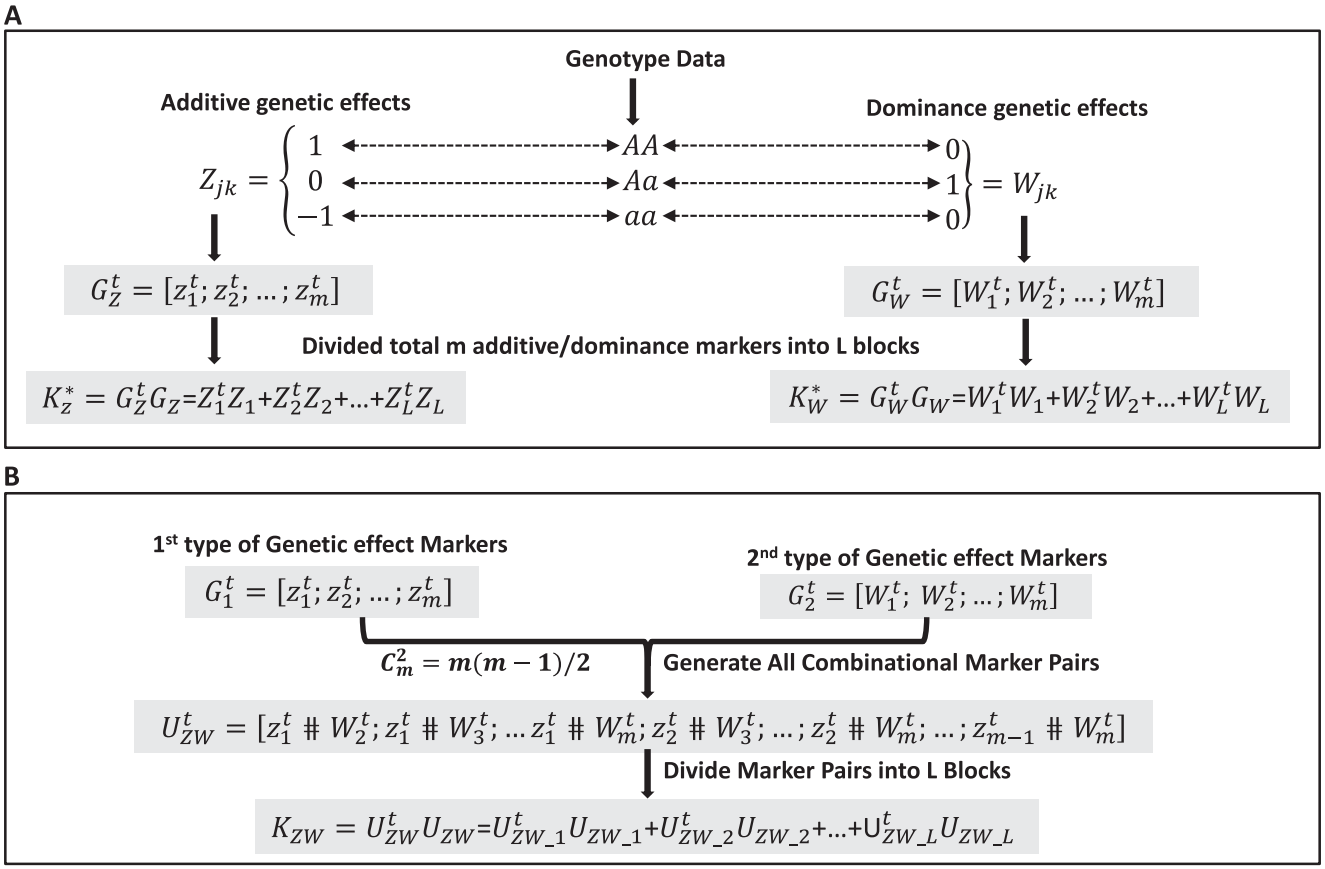

Figure 1. Mathematical principle to calculate the high-dimensional kinship matrices. (**A**) Principle to generate the coded additive and dominant genotypic effect and calculate the main kinship matrix through partitioning the additive/dominance marker into blocks. (**B**) Principle to generate all of the epistatic genotypic marker pairs and calculate the epistatic kinship matrix through partitioning the marker pairs into blocks.

file and loop through mode provides the basic platform cost. In the later section, we will discuss how to use the running time of three modes to calculate the acceleration up ratio of GPU empowered parallel scheme against the traditional CPU serial scheme.

Device kernel design for GPU involved parallel calculation

In the GPU empowered HPC platform, the crucial work is to design the kernel function modules and allocate the specific data for each GPU thread, which physically corresponds to one GPU core. In CUDA system, all of the GPU threads can be organized into 1D, 2D or 3D structure. CUDA gives each a unique thread ID, which can be linearly calculated by the CUDA system built-in variables, such as threadIdx.*, blockIdx.* and gridIdx.*. As discussed earlier, procedures to calculate the kinship matrix can be refined as transpose of a matrix, multiplication and addition of two matrices, and normalization of a matrix. Therefore, we coded four kernels for the four matrix-level mathematical matrix operations.

The addition of two matrices and normalization of a matrix can be easily implemented as matrix-entry-based individual addition or division operation, which can be assigned to a specific GPU thread. Therefore, 1D GPU thread architecture is a good fit to above two kinds of matrix operations. Each GPU thread is distinguished by a unique CUDA built-in variable threadIdx.x and responsible for a corresponding matrix-entry based operation.

The transpose of a matrix and the matrix–matrix multiplication are a little bit complex, because they involve the 2D vertical–horizontal structure of a matrix. The matrix–matrix multiplication has always been chosen as a benchmark of scoring the GPU parallel architecture performance (29). A more efficient strategy is to adopt a tiled structure by defining a warp that allows a collection of GPU threads to access the shared memory and run concurrently (26,29). In technology, GPU threads access the shared memory at least one scale faster than the device's global memory. There are many studies that discuss the performance of the deployed kernels for matrix–matrix multiplication. During our implementation, 2D GPU thread architecture can be deployed and the CUDA built-in variables threadIdx.x, blockIdx.x, threadIdx.y, blockIdx.y are used to indicate the horizontal and vertical indices, which, in this study, correspond to the individual (genotype data matrix column) and the marker (genotype data matrix row) respectively. In the tiled version, the tile dimension size TILE_DIM is a parameter that can be customized and constrained by the GPU device. Our Tesla K80 GPU server allows the maximum warp size as 32. To simplify and achieve the optimal memory-access performance, we hard-configured the TILE_DIM as 32 for the

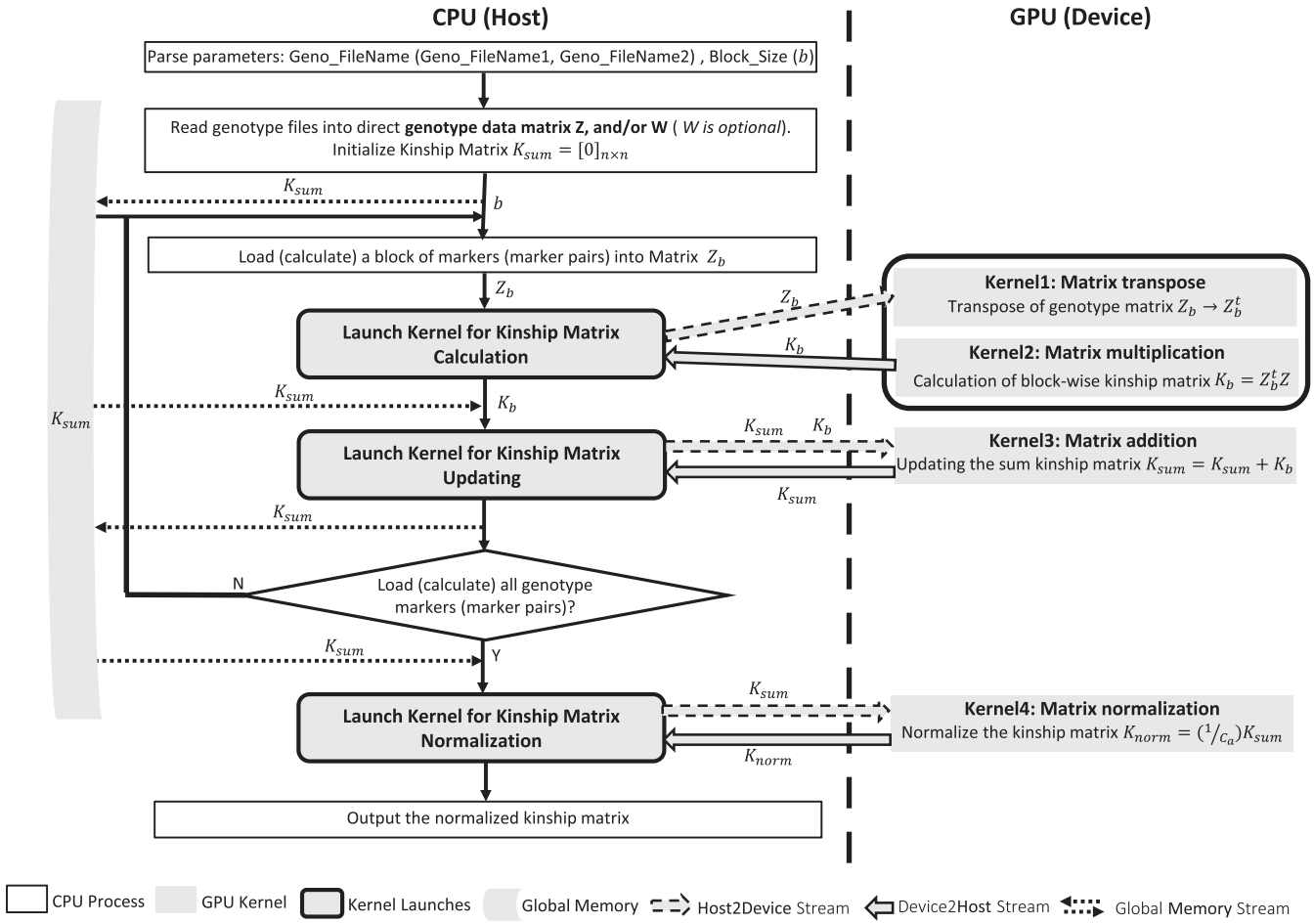

Figure 2. Parallel structure of GPU empowered kinship matrix calculating.

kernel of transpose of matrix and the multiplication of two matrices.

The kernel source codes (Supplementary File S1) for four kinds of matrix operation are written in C++ and could be compiled in NVCC.

HTML5-based resumable transmitting for uploading large genotype data files

KMC1D is responsible for calculating the main effect kinship matrix from the original genotype data file. The genotype data file can reach to several hundred GBs with tens of millions genotype markers and tens of thousands individuals.

Uploading such a large text file from remote client-side to the KMC1D's web server can be very difficult. The genotype data, being as pure text file, can be compressed in lossless format such as .zip, .bzip and .gzip, which have been widely used in GNU tools. However, the lossless compression only can acquire limited and unfixed compression ratio depending on the information content.

To further solve big file uploading issue, we used the HTML5 API File.slice function to develop a specific multithreading resumable file uploading module for uploading the huge genotype data file. In the client-side, the large genotype data file is cut into slices, which are parallel transmitted to the server with its start and end positions at the multithread model. In the server side, the file slices are parallel received and merged into one file. The genotype data for uploading and transmitting can be uncompressed text file or its compressed format. Our tests indicate that KMC1D can efficiently upload a file larger than 200 GB, which can be difficult to handle through traditional methods.

RESULTS

KMC1D and KMC2D are two GPU empowered pipelines designed for high-performance computing of the high-dimensional kinship matrices for main effect and epistatic effect, respectively. We submitted our simulated genotype dataset with variable numbers of genotypic markers and individuals to the two pipelines, recorded the run time at each scenario, and compared the performances of GPU parallel model against golden CPU serial model. Our GPU parallel computing performance analysis results show that KMC1D can achieve about ∼100 times of calculating acceleration, and KMC2D can achieve ∼ 400 times of calculating acceleration when compared with the golden CPU serial running mode.

Implementation environment

The KMC1D and KMC2D pipelines were developed in C/C++ and deployed on a Linux CUDA (Compute Unified Device Architecture) server, which is equipped with a NVIDIA Tesla K80 GPU accelerator. Table 1 gives the details of the implementation environment about the host CPU and device GPU.

Submission of representative genotype file or files to KMC1D and KMC2D

KMC1D requires a genotype matrix file as input, while the KMC2D requires two genotype matrix data files as input. Once the required genotype data file/files are provided, KMC1D and KMC2D will start the data pre-process, launch the GPU cores for parallel computing and return the final integrated kinship matrix.

To efficiently utilize KMC1D and KMC2D, we developed user-friendly web interfaces to provide online services. In the backend, several Linux shell scripts were developed as wrappers to streamline the whole pipeline. The users only need to specify the required genotype file/files, configure one parameter to partition the high-dimensional markers or marker pairs into blocks, and click the Submit or Upload & Submit button. KMC1D and KMC2D will return the analyzed result once the calculation is finished. Figure 3 shows the web user interfaces for submitting genotype data and downloading analysis results. Users also can download the source codes and compile them into executables, and run them through command lines on their own GPU-equipped Linux servers.

Performance and discussion

To evaluate the performances achieved through our GPU empowered parallel model against the golden CPU serial running model, we used real IMF2 Rice genotypic data (Supplementary File S2) with the dimension of 1619 $\times$ 278 and generated a series of simulated data through extendedly repeating the markers (rows) and individuals (columns). Table 2 lists the dimension details of the simulated data for the benchmark performance analysis of KMC1D and KMC2D. We ran the same data at three modes as GPU parallel computing, golden CPU serial computing, and data file read or loop through only, recorded the three running time, then calculated the acceleration ratios according to the formula 16 below. Here, the acceleration ratios represent the objective performance of our GPU empowered pipeline.

$$Acceleration\ Ratio = \frac{(CPU_{Serial} - Read\_Loop\_Through)}{(GPU_{Para} - Read\_Loop\_Through)} \quad (16)$$

Case study for KMC1D

To evaluate the performance of KMC1D that can be affected by the number of SNP markers and individuals, we classified the simulated genotype data files with a fixed individual number and variable marker numbers to group Test 1 and vice versa for group Test 2. We ran each genotype data at three models and recorded all running times. Finally, we used formula 16 to calculate the acceleration ratios. Figure 4 gives the performance plots.

In Figure 4, we demonstrated that our GPU empowered pipeline KMC1D acquired 10–100 acceleration ratios. Figure 4 also shows that (i) the running time at CPU serial mode has a simple linear and square relationships with the variable marker number m and individual number n, respectively, which verified that the complexity of main effect kinship matrix follow as $O(mn^2)$; (ii) the close entanglement of the performance curves between GPU parallel mode and read and loop through model indicates that the main burden of KMC1D is from the genotype data pre-processing, such as file reading; (iii) as the individual number or marker number increases, the acceleration rates increase quickly and then slowly to reach plateau, which can be explained by the reason that the more and more GPU cores were gradually deployed for calculation until saturated.

We also submitted a very large genotype matrix with the dimension of 10000000 $\times$ 5000 and the file size of $\sim$119 GB to KMC1D for performance test. It took about 5 h for KMC1D to complete the calculation. We specifically separated and recorded the times for data file uploading and running time on the same large genotype data file at read and loop through mode. It reported 50 min for data uploading, and about 4 h for read and loop through the data matrix. This evidence again shows us that the main burden of KMC1D is from the file uploading and data file reading.

Case study for KMC2D

Similarly, the simulated genotype data for KMC2D are classified as with a group with a same individual number (n = 1390) but varying marker numbers, and another group with a fixed marker number (m = 16190) but varying individual numbers. For each genotype data, we ran it at three modes as GPU parallel computing, golden CPU serial, and data file read and loop through only, then recorded the running times. Again, we use formula 16 to calculate the acceleration ratios. Figure 5 illustrates the performance plots.

From Figure 5, we demonstrated that our GPU empowered pipeline KMC2D achieved acceleration rates at several hundred times. Our analyses also suggested that (i) the running time of CPU serial model or GPU parallel model had a square relationship with the variable marker number m and individual number n, which validated that the complexity of main effect kinship matrix followed $O(m^2n^2)$; (ii) the running time of file read and loop through only model had a square and linear relationship with the variable marker number m and individual number n, respectively, because the number of marker pairs was equal to $C_m^2 = (\boldsymbol{m} - 1)\ \boldsymbol{m}/2$; (iii) before the GPU cores were fully utilized, the acceleration ratio continued to rise with the increase of marker number or individual number, but after that, it reached a plateau. For the genotype file with 16 190 marker, there will be $(16190 - 1) \times 16190/2 = \sim$ 131 million marker pairs, considering the 1390 individuals, which is far beyond the capability of any existing GWAS platform. It only took KMC2D about 30 min to complete the epistatic kinship matrix calculation, while it would take sev-

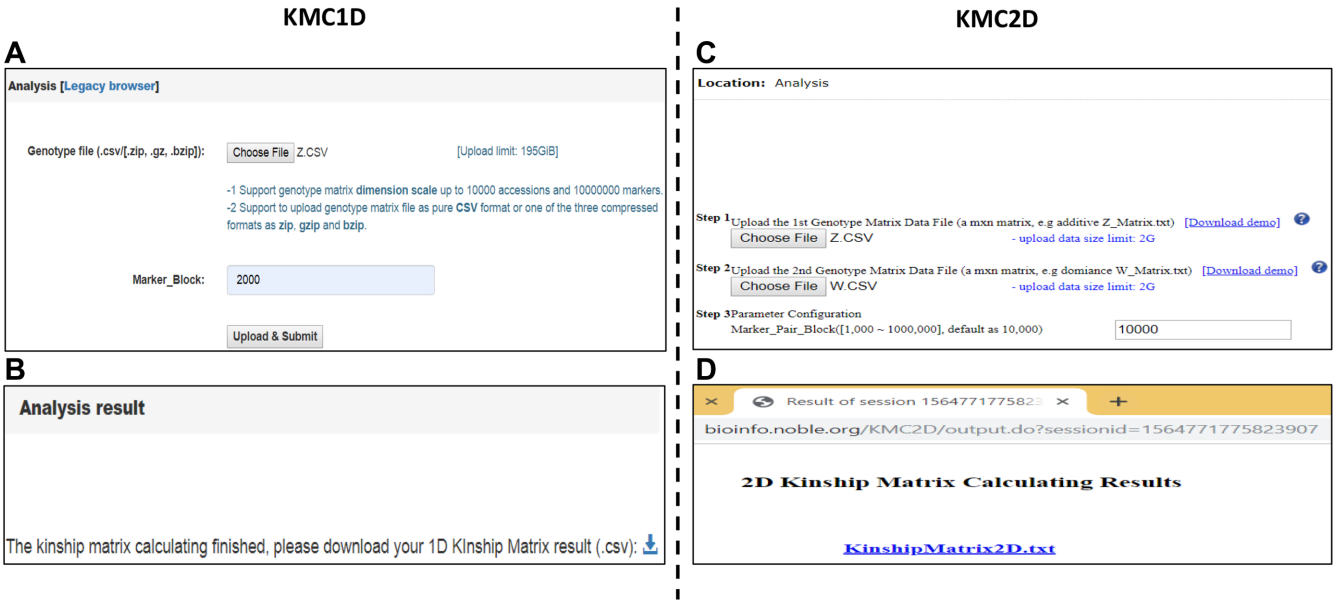

Figure 3. User Interface for submission of genotype data and downloading the kinship matrix. (**A**) KMC1D user interface for uploading the very large genotype data. (**B**) KMC1D user interface for downloading the calculated main effect kinship matrix. (**C**) KMC2D user interface for uploading the two genotype data. (**D**) KMC2D user interface for downloading the calculated epistatic effect kinship matrix.

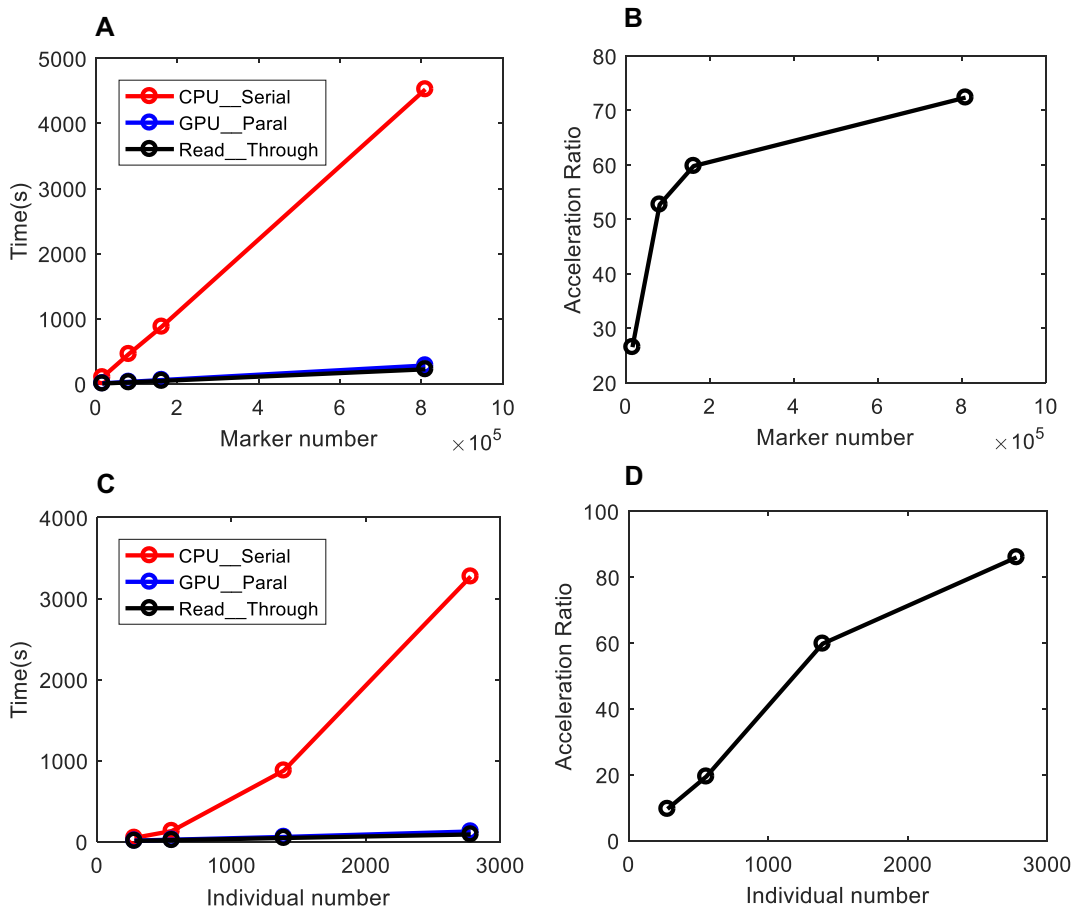

Figure 4. Performance of KMC1D with variable marker number and a fixed individual number (Test 1) or vice versa (Test 2). Test 1 (A and B) is set for a fixed individual number at 1390. (**A**) Relationship of running times with variable marker number. (**B**) Relationship of acceleration ratios with the variable marker number. Test 2 (C and D) is set for a fixed marker number at 161 900. (**C**) Relationship of running times with variable individual number. (**D**) Relationship of acceleration ratios with the variable individual number.

Table 1. The implementation environment for host CPU and device GPU

Language	Platform	CUDA Devices	Driver Version	Run Version	CUDA Cores	Device GPU Clock	Host CPU Clock
C/C++	Linux	Tesla K80	9.0	9.0	2496	824MHz	1.74G

Table 2. Dimension of the simulated data for KMC1D and KMC2D

KMC1D					KMC2D				
	Test 1. Fix #Individual = 1390					**Test 1**. Fix #Individual = 1390			
#Marker	16 190	80 950	161 900	809 500	#Marker	1619	3238	8095	16 190
	Test 2. Fix #Marker = 161 900					**Test 2**. Fix #Marker = 16 190			
#Individual	278	556	1390	2780	#Individual	278	556	834	1390

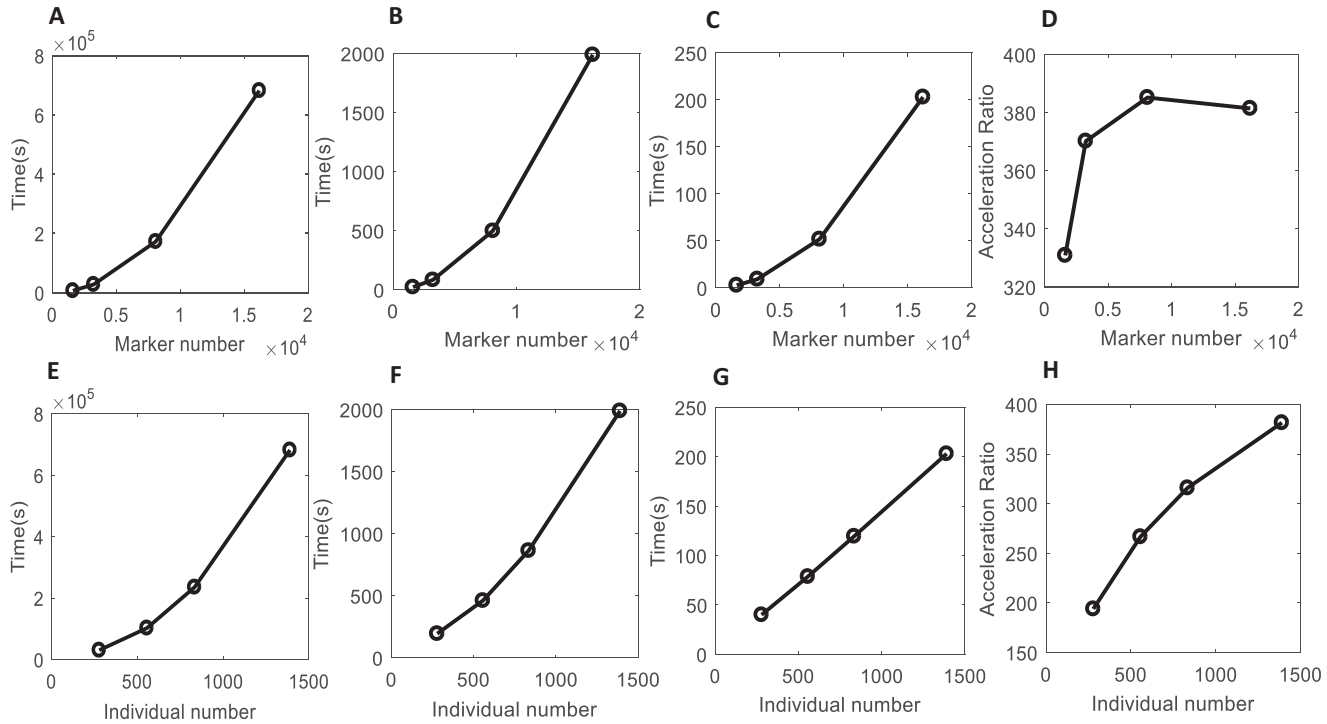

Figure 5. Performance of KMC2D with variable marker number and a fixed individual number (Test 1) or vice versa (Test 2). Test 1 (A–D) is set for a fixed individual number at 1390. (**A–C**) Relationship of running time with variable marker number at CPU Serial mode, GPU Parallel mode, and Loop Through mode respectively. (**D**) Relationship of acceleration ratios with the variable marker number. Test 2 (E–H) is set for a fixed marker number at 16 190. (**E–G**) Relationship of running times with variable individual number at CPU Serial mode, GPU Parallel mode, and Loop Through mode, respectively. (**H**) Relationship of acceleration ratios with the variable individual number.

eral weeks at the CPU serial mode to complete the same calculation.

Study on the parameter of partition block size

The block size for partitioning the markers or marker pairs into blocks is the only parameter we need to configure. The block size essentially determines the allocated RAM needed in the host CPU, which, in general, follows a linear relationship. On the other hand, the block size is also related to the 2D threads architecture in GPU device part, and the larger block size demands the more GPU cores that are deployed. Therefore, the partition block size is a very important parameter, which greatly affects the two pipelines' performance. We tested the same genotype data by partitioning the markers or marker pairs into different block sizes, and separately submitted them to KMC1D or KMC2D for main or epistatic effect kinship matrix calculations. Supplementary Tables S1 and S2 provided the dimension details for the test data and the performance results at different block sizes. Based on these performance tests, we found that when the block sizes were set >2000, it only minimally affected the computing performance. This can be understood by the 1D/2D GPU parallel computing architecture. All the available GPU cores/threads can be arranged as a 2D architecture, which span over to one direction (e.g. threadIdx.x, blockIdx.x) for the individuals, and the other direction (e.g. threadIdx.y, blockIdx.y) for the marker or marker pairs. When the block parameters for marker or marker pair were set above 2000, the GPU threads in the marker or marker pair direction would be fully utilized. Additionally, the marker or marker pairs are partitioned into

blocks, therefore, KMC1D or KMC2D only need to allocate a memory block from hundreds of KBs to several hundred MBs, which greatly decrease the required RAM size.

CONCLUSION

Kinship matrix is widely used to efficiently measure the relatedness among individuals, and the kinship matrix calculation is the first and most important step when conducting a traditional 1D GWAS analysis. Polygenic LMMs and epistatic effect analysis can reveal the phenotypic variance, which has been shown to explain the missing heritability. To solve the polygenic LMMs and conduct an epistatic effect 2D GWAS analysis, we must generate the combinational marker pairs, then calculate the epistatic effect kinship matrices. However, the typical main effect kinship matrix calculation can be a challenge, if there are tens of millions of genotype markers and tens of thousands of individuals; the epistatic effect kinship matrix can be more challenging even for typical genotype data consisting of several thousands of markers and hundreds to thousands of individuals.

To overcome these challenges, we systematically investigated the mathematical principle to calculate the main and epistatic effect kinship matrices. We then adopted a strategy to partition the marker or marker pairs into blocks, calculate each sub-kinship matrix and merge them into one using the linearity property of matrix operation. Based on these strategies, we further refined the kinship matrix calculation into several basic matrix operations including transpose of a matrix, matrix–matrix multiplication, matrix addition and matrix normalization. All of these matrix operations are time-consuming but can be parallel computed using hundreds to thousands of GPU device cores.

We successfully developed two GPU empowered pipelines, KMC1D and KMC2D, which were deployed on our Linux server quipped with a Tesla K80 GPU accelerator, for main effect and epistatic effect kinship matrix calculation. The matrix operations have been coded into GPU kernels. All of the source codes, including the host CPU part and device GPU part, are written in CUDA C/C++ and compiled by NVCC. Our GPU parallel computing performance test results show that KMC1D can achieve about ∼100 times of calculating acceleration, and KMC2D can achieve ∼400 times of calculating acceleration when compared with the golden CPU serial running mode. We also implemented a multi-threading and resumable data-transferring module, utilizing HTML5 API File.slice function, for uploading large-size data files.

DATA AVAILABILITY

Both KMC1D and KMC2D are developed in C++ and compiled with NVCC++ in Linux environment. The C++ and CUDA source codes, the details about the procedures to compile the source codes into executables and the instructions about using command lines to run the executables on the public repository GitHub at https://github.com/noble-research-institute/KMC1D and https://github.com/noble-research-institute/KMC2D, respectively. There is no restriction for end-users to download, modify, compile and use the two packages in their own GPU environments.

The KMC1D and KMC2D web programs are publicly available at https://bioinfo.noble.org/KMC1D/ and https://bioinfo.noble.org/KMC2D/, respectively.

SUPPLEMENTARY DATA

Supplementary Data are available at NARGAB Online.

FUNDING

National Science Foundation Collaborative [DBI-1458597 to P.X.Z.; DBI-1458515 to S.X.]; Noble Research Institute (in part, to P.X.Z.).
Conflict of interest statement. None declared.

REFERENCES

1. Manolio,T.A. (2010) Genomewide association studies and assessment of the risk of disease. *N. Engl. J. Med.*, **363**, 166–176.
2. Stranger,B.E., Stahl,E.A. and Raj,T. (2011) Progress and promise of Genome-Wide association studies for human complex trait genetics. *Genetics*, **187**, 367–383.
3. Visscher,P.M., Wray,N.R., Zhang,Q., Sklar,P., McCarthy,M.I., Brown,M.A. and Yang,J. (2017) 10 years of GWAS discovery: biology, function, and translation. *Am. J. Human Genetics*, **101**, 5–22.
4. Hoffman,G.E. (2013) Correcting for population structure and kinship using the linear mixed model: theory and extensions. *PLoS One*, **8**, e75707.
5. Kang,H.M., Sul,J.H., Service,S.K., Zaitlen,N.A., Kong,S.Y., Freimer,N.B., Sabatti,C. and Eskin,E. (2010) Variance component model to account for sample structure in genome-wide association studies. *Nat. Genet.*, **42**, 348–354.
6. Patterson,N., Price,A.L. and Reich,D. (2006) Population structure and eigenanalysis. *PLos Genet.*, **2**, e190.
7. Mangin,B., Siberchicot,A., Nicolas,S., Doligez,A., This,P. and Cierco-Ayrolles,C. (2011) Novel measures of linkage disequilibrium that correct the bias due to population structure and relatedness. *Heredity*, **108**, 285–291.
8. Myles,S., Peiffer,J., Brown,P.J., Ersoz,E.S., Zhang,Z., Costich,D.E. and Buckler,E.S. (2009) Association mapping: critical considerations shift from genotyping to experimental design. *Plant Cell*, **21**, 2194–2202.
9. Yu,J., Pressoir,G., Briggs,W.H., Vroh Bi,I., Yamasaki,M., Doebley,J.F., McMullen,M.D., Gaut,B.S., Nielsen,D.M., Holland,J.B. *et al.* (2006) A unified mixed-model method for association mapping that accounts for multiple levels of relatedness. *Nat. Genet.*, **38**, 203–208.
10. Yang,J., Lee,S.H., Goddard,M.E. and Visscher,P.M. (2011) GCTA: a tool for Genome-wide complex trait analysis. *Am. J. Human Genetics*, **88**, 76–82.
11. Bradbury,P.J., Zhang,Z., Kroon,D.E., Casstevens,T.M., Ramdoss,Y. and Buckler,E.S. (2007) TASSEL: software for association mapping of complex traits in diverse samples. *Bioinformatics*, **23**, 2633–2635.
12. Speed,D. and Balding,D.J. (2014) Relatedness in the post-genomic era: is it still useful? *Nat. Rev. Genet.*, **16**, 33–44.
13. Astle,W. and Balding,D.J. (2009) Population structure and cryptic relatedness in genetic association studies. *Statist. Sci.*, **24**, 451–471.
14. Bernardo,R., Murigneux,A. and Karaman,Z. (1996) Marker-based estimates of identity by descent and alikeness in state among maize inbreds. *Theor. Appl. Genet.*, **93**, 262–267.
15. Chang,C.C., Chow,C.C., Tellier,L.C., Vattikuti,S., Purcell,S.M. and Lee,J.J. (2015) Second-generation PLINK: rising to the challenge of larger and richer datasets. *GigaScience*, **4**, doi:10.1186/s13742-015-0047-8.
16. Pandey,A., Davis,N.A., White,B.C., Pajewski,N.M., Savitz,J., Drevets,W.C. and McKinney,B.A. (2012) Epistasis network centrality analysis yields pathway replication across two GWAS cohorts for bipolar disorder. *Transl. Psychiatry*, **2**, e154.
17. Carlborg,Ö. and Haley,C.S. (2004) Epistasis: too often neglected in complex trait studies? *Nat. Rev. Genet.*, **5**, 618–625.

18. Eichler,E.E., Flint,J., Gibson,G., Kong,A., Leal,S.M., Moore,J.H. and Nadeau,J.H. (2010) Missing heritability and strategies for finding the underlying causes of complex disease. *Nat. Rev. Genet.*, **11**, 446–450.
19. Xu,S. (2013) Mapping quantitative trait loci by controlling polygenic background effects. *Genetics*, **195**, 1209–1222.
20. Zhang,W., Dai,X., Wang,Q., Xu,S. and Zhao,P.X. (2016) PEPIS: a pipeline for estimating epistatic effects in quantitative trait locus mapping and genome-wide association studies. *PLoS Comput. Biol.*, **12**, e1004925.
21. Zhang,W., Dai,X., Xu,S. and Zhao,P.X. (2018) 2D association and integrative omics analysis in rice provides systems biology view in trait analysis. *Commun. Biol.*, **1**, 153.
22. Lipka,A.E., Tian,F., Wang,Q., Peiffer,J., Li,M., Bradbury,P.J., Gore,M.A., Buckler,E.S. and Zhang,Z. (2012) GAPIT: genome association and prediction integrated tool. *Bioinformatics*, **28**, 2397–2399.
23. Wang,B., Sverdlov,S. and Thompson,E. (2017) Efficient estimation of realized kinship from single nucleotide polymorphism genotypes. *Genetics*, **205**, 1063–1078.
24. Hong,E.P. and Park,J.W. (2012) Sample size and statistical power calculation in genetic association studies. *Genomics Inform.*, **10**, 117–122.
25. Brodtkorb,A.R., Hagen,T.R. and Sætra,M.L. (2013) Graphics processing unit (GPU) programming strategies and trends in GPU computing. *J. Parallel Distrib. Comput.*, **73**, 4–13.
26. Tomaž Dobravec,P.B. (2017) Comparing CPU and GPU implementations of a simple matrix multiplication algorithm. *Int.J. Comput. Electric.Eng.*, **9**, 430–438.
27. Nielsen,R., Paul,J.S., Albrechtsen,A. and Song,Y.S. (2011) Genotype and SNP calling from next-generation sequencing data. *Nat. Rev. Genet.*, **12**, 443–451.
28. Nielsen,R., Korneliussen,T., Albrechtsen,A., Li,Y. and Wang,J. (2012) SNP calling, genotype calling, and sample allele frequency estimation from new-generation sequencing data. *PLoS One*, **7**, e37558.
29. Cecilia,J.M., García,J.M. and Ujaldón,M. (2009) The GPU on the matrix-matrix multiply: performance study and contributions. *PARCO*, **19**, 331–340.