Far East Journal of Mathematical Sciences (FJMS)

Published Online: August 2015
http://dx.doi.org/10.17654/FJMSOct2015_347_364
Volume 98, Number 3, 2015, Pages 347-364 **ISSN: 0972-0871**

CALCULATION OF INDIVIDUAL STRATIFICATION BY MULTICORE PROCESSORS

A. Burutarchanai* and P. Chongstitvatana

Department of Computer Engineering
Chulalongkorn University
Bangkok, 10330, Thailand
e-mail: alongkot.bu@gmail.com
prabhas.c@chula.ac.th

Abstract

This work proposed a parallel paradigm to speedup the calculation for the analysis of individual admixture for individual identification by genetic data. The calculation is developed from the Frappe program. The calculation method is based on a likelihood calculation, which requires a large number of iterations to obtain the solution. The likelihood calculation is performed using expectation maximization (EM). The approach is developed to utilize the calculation power on parallel machines. Multicore processors are suitable because they are widely available with reasonable cost. Our software can achieve speedup of the calculation 3-fold with 4-core processors. The program was implemented with a 64-bit multicore processor to improve the performance on large datasets. The algorithm was adapted to the multiprocessor architecture. The program also has a milestone feature allowing it to re-execute from the previous stop point which is very important for a long running job.

Received: April 29, 2015; Revised: June 29, 2015; Accepted: July 6, 2015
2010 Mathematics Subject Classification: 68-XX.
Keywords and phrases: individual identification, admixture analysis, expectation maximization, multithreading processor.
*Corresponding author
Communicated by K. K. Azad

1. Introduction

The analysis of admixture is used to study the stratification of populations. It is useful for genetic association studies [1] and admixture mapping [2, 3]. The calculation can be applied in epidemiology genetics [3, 4]. The analysis requires genetic data in the format of single nucleotide polymorphisms (SNPs) and the number of ancestral groups. The result is an inheritance ratio from each ancestral group that transferred to the individual.

Admixture analysis estimates ancestry based on a maximum likelihood relation. There are several software programs that can perform such calculations. In 2003, STRUCTURE was proposed by Falush et al. [5]. The calculation model was based on the Markov chain. The proposed model applied biological knowledge of linkage disequilibrium and allele frequency to the calculation. STRUCTURE provides the result of mixing ratios at a group level, not an individual level.

Pfaff et al. [6] proposed a calculation method based on the ancestry informative marker (AIM) information. AIM uses a particular DNA sequence and its location. To avoid the problem of lacking historical data, Chikhi et al. [7] proposed a calculation model that used a genetic relationship and Bayesian relationship. The proposed model is very useful for two groups of ancestry and with the knowledge of the time of mixing. Tang et al. [8] applied a combination relationship without a mixing generation constraint to generalize the model. Tang et al. [8] proposed a calculation method to solve the proposed calculation model by expectation maximization (EM). The calculation is guaranteed to converge, although it may converge to a local maximum [9]. Alexander et al. [10] claimed that the Newton-Raphson calculation is better than EM. They proposed a block relaxation method that was a quasi-Newton method, and his calculation used a conjugate gradient, which reduces the computing cost of the Jacobian matrix. Most of the proposed methods work with small sizes of genetic data.

2. Methods

This section explains the relationship of the genetic data in SNPs format

to the calculation and then proposes the new approach. The original calculation of admixture was proposed by Tang et al. using EM. The new approach was adapted from Alexander's method by making it more specific to the input data type. Alexander et al. proposed a quasi-Newton method to solve the equation. However, this work uses an EM method instead of a quasi-Newton method because the EM method guarantees convergence. The original calculation is described in the next section.

2.1. The original method

SNP data represents a difference in a single DNA building block in the living organism. The SNPs data is encoded by values of 0, 1 and 2. The value of SNPs data represents as 2 for homozygosity wild type which is major allele for both traits, 1 for heterozygosity wild type which is one trait major allele and one minor allele and 0 for homozygosity variant which is minor allele for both traits. The likelihood calculation is developed from mixture ratio. The mixing ratio drives from the calculation of the probability marker value, to be major or minor allele, of each allele of a selected locus of each individual in the hybrid population from their SNPs ancestry population. The calculation is a linear combination of each ancestry population as shown in Table 1.

Table 1. The probability of allele inheritance of a selected trait from the ancestry groups I and II. The table on the left hand side shows the ratio and the allele frequency of the groups I and II that transfer to their children on the right hand table. The major allele of *A* is the sum of all the probabilities of the major allele to be a probability value of the major allele, vice versa for the probability of the minor allele

Ancestral	I	II
Ancestral ratio	0.9	0.1
Major allele (A)	0.7	0.5
Minor allele (a)	0.3	0.5

Hybrid		$P \times Q$
Allele A	I	0.63
	II	0.05
Allele a	I	0.27
	II	0.05

Table 1 shows an example of the linear combination property of genetic transfer from generation to generation. The probability of the selected trait

has potential to be major allele at 68%, but each locus composes of two traits. The hybrid population genetic G composes of M markers from every I individual, each marker has two traits, the genetic value is referred as g_{im}. The g_{im} is calculated by the sum of product of the ratio of each population group and their genetic value as (1) for major allele and (2) for minor allele:

$$P(g_{im} = Major\ allele | P, Q) = \Sigma_K q_{ik} \times (p_{km}), \quad (1)$$

$$P(g_{im} = Minor\ allele | P, Q) = \Sigma_K q_{ik} \times (1 - p_{km}). \quad (2)$$

g_{im} : represents the SNP value of the mth marker belonging to the ith individual.

p_{km} : represents major allele frequency of the mth marker of the kth ancestor. The minor allele frequency can be inferred from $1 - p_{km}$.

q_{ik} : represents the mixing ratio of the kth groups of ancestral in the ith individual.

The g_{im} for value of 2, each trait is inherited from major allele. The probability is calculated by $[\Sigma_K q_{ik} \times (p_{km})]^2$. For g_{im} value 1, one trait is inherited from major allele and one from minor allele, the probability is $[\Sigma_K q_{ik} \times (p_{km})] \times [\Sigma_K q_{ik} \times (1 - p_{km})]$. The g_{im} for value of 0, each trait is inherited from minor allele, the probability is calculated by $[\Sigma_K q_{ik} \times (1 - p_{km})]^2$.

The admixture calculation was a method to find a solution of ancestor in the left hand table from the mixed data. This is called *likelihood calculation*. The likelihood calculation is a reverse calculation of the probability calculation as the mixture calculation. The calculation maximizes the likelihood value of ancestry allele frequency, P, and ancestry admix ratio, Q, from the hybrid genetic value, G.

The likelihood relation is proposed as (3). The input is genetic data, G, in SNP format. The relation is developed from conditional product of every trait

of every loci of marker and every individual

$$L(P, Q|G) = \prod_I \prod_M \begin{cases} \prod_K (q_{ik} \times p_{km})^2, & g_{im} = 2, \\ \prod_K (q_{ik} \times p_{km})(q_{ik} \times (1 - p_{km})), & g_{im} = 1, \\ \prod_K (q_{ik} \times (1 - p_{km}))^2, & g_{im} = 0. \end{cases} \quad (3)$$

Equation (3) can be transformed into (4). The conditional product can be changed into power form in (4):

$$L(P, Q|G) = \prod_I \prod_M \prod_K (q_{ik} \times p_{km})^{g_{im}} (q_{ik} \times (1 - p_{km}))^{(2-g_{im})}. \quad (4)$$

There are several approaches of the likelihood relation as above. This work proposed an approach which used expectation maximization (EM). The EM calculates the likelihood relation by taking log of the likelihood, call *log-likelihood*. The log-likelihood is addressed as (5):

$$L(P, Q) = \Sigma_I \Sigma_M \{(g_{im}) \log \Sigma_K q_{ik} \times (p_{km}) + (2 - g_{im}) \log \Sigma_K q_{ik} \times (1 - p_{km})\}. \quad (5)$$

The data for this calculation are genetic values with the size of I individuals $\times$ M markers in SNP format. The solutions of admixture calculation are the parameters of the likelihood that maximize the likelihood value with respect to the current genetic data.

The EM method decomposes the likelihood calculation into three steps, the initialization step with bias, the expectation step and the maximization step. The initialization step is biased to perturb the search direction. Based on the experiment, this bias is important for the solution to converge. The EM method repeats the steps of maximization and expectation until the solution is converged. The maximization step computes likelihood values from the parameter values. The expectation step re-estimates the parameter values from the new likelihood values from the maximization step. The

maximization step computes equations (6)-(7) and the expectation step computes equation (8):

$$p_{mlk}^{(n)} = \frac{\Sigma_{i=1}^{I}\Sigma_{a=1}^{2}1(G_{ima} = l)E_{imak}^{(n-1)}}{\Sigma_{i=1}^{I}\Sigma_{a=1}^{2}E_{imak}^{(n-1)}}, \tag{6}$$

$$q_{ik}^{(n)} = \frac{\Sigma_{m=1}^{M}\Sigma_{a=1}^{2}E_{imak}^{(n-1)}}{2M}, \tag{7}$$

$$E_{imak}^{(n+1)} = \frac{p_{mlk}^{(n)}q_{ik}^{(n)}}{\Sigma_{k=1}^{K}p_{mlk}^{(n)}q_{ik}^{(n)}}. \tag{8}$$

The calculation of maximization, (6) and (7) can be performed separately because the inputs are independent. The calculation reads the genetic value and E and computes the update values of the ancestry allele frequency, p, and the individual admixture, q. The stopping criterion is satisfied when the likelihood values become stable (9):

$$L(n) - L(n-1) < \varepsilon. \tag{9}$$

2.2. The new method

The calculation approach is adapted to support a multiple calculation unit. There are two considerations in proposing a new method. The first is the technological aspect. As there are many available technologies to speedup a computation, the choice of technology is explained. The second consideration is the mathematic calculation that utilizes a multiple calculation unit. These two considerations are detailed in the next subsection.

2.2.1. Multiprocessor technology

There are several technologies to speedup a computation such as using Graphic Processing Units, using many machines in cloud computing or using the multithreading available in modern computer systems. Each technology has its own merit. The multithreading technology is chosen because it is available at low cost to the researchers and there is good software development support. The Microsoft dot Net 4.0 framework supports the use

of multicore processors. The framework provides a method of *parallel-for* to implement a parallel version of a for-loop. The *parallel-for* construct distributes multiple tasks of calculation to many individual cores. The calculation of equations (5)-(8) consists of a nested for-loop, and therefore it fits this framework. The *parallel-for* was used in the outer for-loop. The compiler of the dot Net framework automatically assigns each inner for-loop to each calculation unit. This mechanism can gain speed by using many cores at very low overhead. The implemented software runs on a 64-bit Microsoft operation system version for extended use of large memory.

To achieve good speedup on multiple processors, the pattern of access to the memory must be considered. The aim is to reduce the amount of data exchange between many computing units. There are several access patterns: row-wise, column-wise and block-wise. The row-wise pattern partitions the task of calculation into row stripes. Each row of data is assigned to each computing unit, and vice versa, the column-wise pattern partitions the task into column stripes. The block-wise pattern partitions a task into several small blocks and assigns each block to each computing unit. In considering the size of each partition (the amount of work to be done), the calculation time and communication time must be balanced. If a sub-task is too small, then the amount of communication between tasks will be increased because there will be a large number of sub-tasks. Furthermore, when using a shared value between several computing units, the concurrent update to shared values must be handled appropriately to reduce the amount of conflict access to the memory and to ensure the proper update sequence. This conflict can avoid by adjusting the calculation as below.

2.2.2. Parallel calculation

The value of likelihood in (5) is a major cause of conflicting access to the memory because it includes a variable shared between many threads of calculation. To reduce this conflict, the shared variable is replaced by an array of variables, each owned by an individual computing unit. Equation (10) reflects this change. The likelihood variable (L) is divided amongst i computing threads. The row-wise pattern was chosen because of the nature of

genetic data. The number of markers is much larger than the number of individuals. The calculation is performed simultaneously on every row. The likelihood value was changed from a single variable to an array of values individually updated by each computing unit:

$$L(P, Q, i) = \Sigma_M \{(g_{im}) \log \Sigma_K q_{ik} \times (p_{km}) + (2 - g_{im}) \log \Sigma_K q_{ik} \times (1 - p_{km})\}. \quad (10)$$

Each ith row of calculation is assigned to each calculation thread. The stopping criterion must be extended to (11):

$$V_i[L(n, i) - L(n - 1, i)] < \varepsilon'. \quad (11)$$

The maximization step in (6) and (7) can be simultaneously calculated in one *parallel-for* loop. The row-wise pattern was applied. The calculation time complexity is $O(M)$, which means that the number of markers determines the calculation time. Using the method of separately updating the values of P and Q, the calculation time complexity will become to $O(M + I)$, where M is the number of markers and I is the number of individuals. Similarly, the expectation step was implemented using a row-wise pattern.

Here is the summary of the computation of the likelihood value. The calculation is performed by three independent loops: initialization, maximization and expectation. Maximization and expectation are executed concurrently until meeting the stopping criterion (11).

Initialization. This step was run only once at the start to initialize the value of E_{imak} of (6) and (7). The random values have bias and the sum of each vector E_{ima} must be exactly 1.

The following steps are repeated until the likelihood values are not significantly different from the previous iteration.

Maximization. Calculate (6) to maximize the likelihood value by P and (7) to maximize the likelihood value by Q. The calculations can be

simultaneously executed because there is no dependency of the inputs of the calculation. The inputs are the ancestry allele frequency and the ancestry admix ratio.

Expectation. Calculate the E_{imak} of the maximized P and Q. The likelihood value (8) can be simultaneously calculated in this step. The last column of the likelihood value is used to test the stopping criterion.

Most of the calculation steps are performed on the row-wise pattern. This pattern reduces the number of memory accesses from every calculation unit. For example, in the expectation step, the Q value of each individual was accessed only one time for each row with the row-wise pattern. If the block-wise pattern was used, then there would be a very large number of memory accesses for Q by M times, and for P by I times. Comparatively, in the column-wise pattern, P was read only one time and Q by I times. Finally, the row-wise pattern reads Q one time and reads P by M times. Because the genetic data has a much larger number of columns than the number of individuals, the row-wise pattern reduces the overhead of memory access.

3. Results

The purpose of this work is to speedup the calculation of large data by optimizing the performance of multicore calculation units. The bovine dataset from Bovine HapMap Consortium [12] was chosen to validate the proposed method. This dataset consists of 44,706 markers from 1,121 individuals. The test was run on input ranging from 3 groups of ancestral through 30 groups of ancestral. The test was performed using an Intel Xeon E5520 with 24 GB of main memory running on 64-bit Window 7.

The result of EM calculation provides three types of data. The first type is the ancestral admixture ratio of each individual. The second type is the ancestral allele frequency from each group in each locus. The last type is latent variable in each individual allele copy. This information can be used to infer the source of the allele copy. The Newton-Raphson method cannot provide this information. The EM method is robust because it can guarantee

convergence even when the starting point is not close to the solution. Figure 1 shows the speedup of the proposed method on a 4-core processor versus the original Frappe on the same processor.

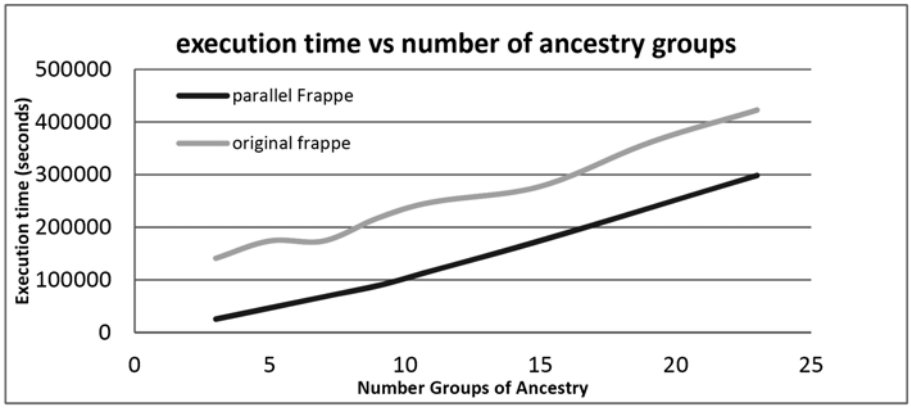

Figure 1. Comparing the calculation speed of the original Frappe and the proposed method on various sizes of the ancestry group. The vertical axis is the execution time, in second, until the algorithm terminates. The horizontal axis is the number of groups of ancestry *k*. The results show that *k* has a huge effect on the speed of calculation. The graph shows steeper slopes at some *k*, such as at $k = 7$ and $k = 15,$ in the original method, which may be caused by the result of memory paging. The Microsoft dot Net seems to make a good effort at memory management that makes the graph smoother between each ancestry group in parallel Frappe.

The likelihood value in Figure 2 represents the convergence of the value according to the theory of EM. Figure 3 shows the admix ratio of the original method on the upper panel and the proposed method on the lower panel. The results show little difference between the original version and the proposed version. The existing difference came from the difference of the initial point. However, the quality of the overall admix ratio is acceptable.

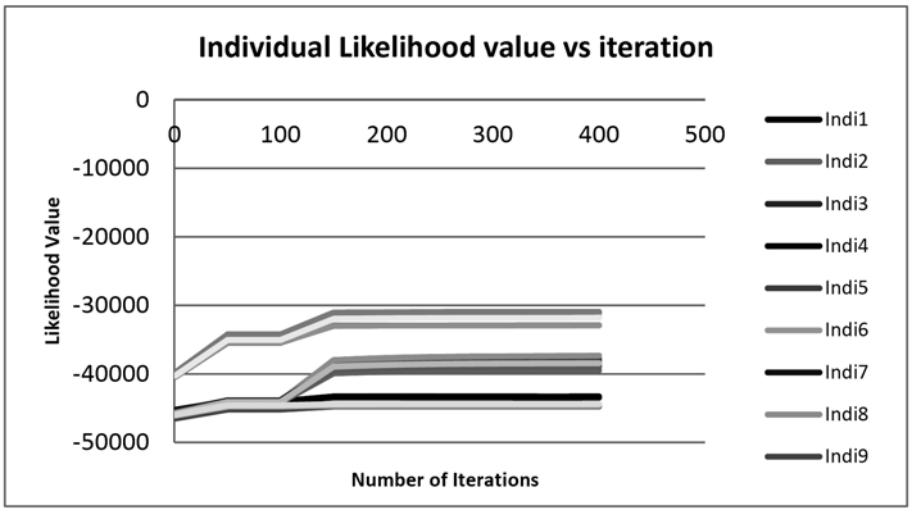

Figure 2. The convergence of the likelihood value of the proposed method. The vertical axis shows the likelihood value of each individual and the horizontal axis shows the iteration number. This graph is obtained from the Bovine dataset which consists of 44707 SNPs, 1121 individual and 7 ancestry groups. The convergence was obtained at approximately 200 iterations.

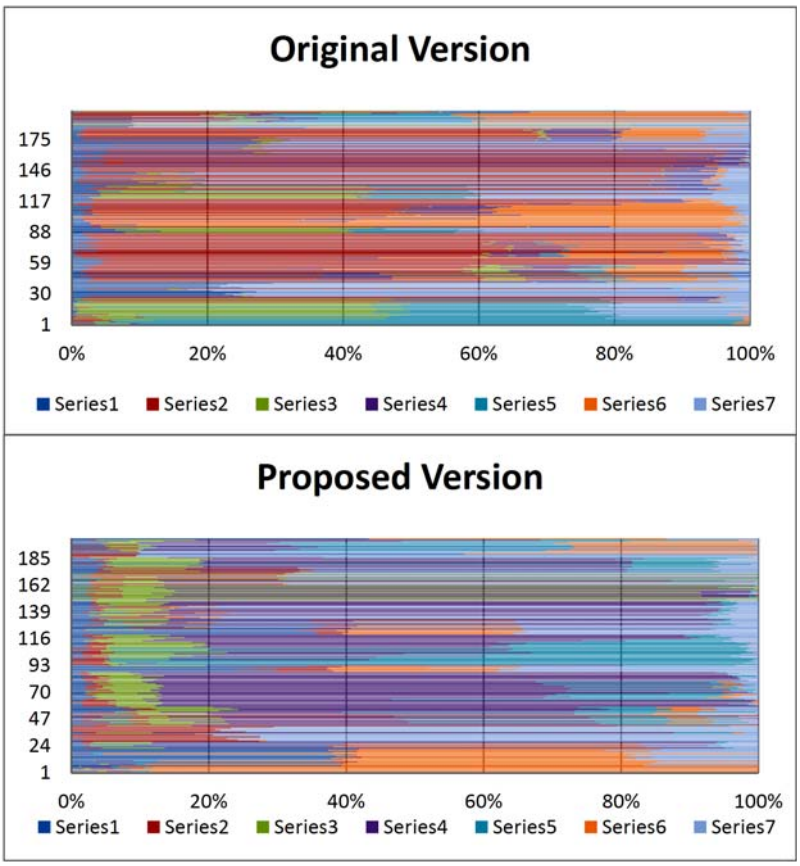

Figure 3. Comparing the results of the original Frappe and the proposed method. Two graphs show the admix ratio sampled from the same two hundred rows of individuals.

4. Conclusions

The adjustment of mathematic calculation can significantly improve the performance in a multi-processor machine. This work provides an adjustment of calculation method that best fits with the genetic data in SNPs format. The value of genetic data can directly be used in (5) instead of (3). The result shows the calculation speedup of the proposed method by Microsoft dot Net 4.0.

There are various types of parallel processing such as using Graphic

Processing Units, a cluster computing platform or the multithread method. An advantage of multicore processors is the power efficiency per computation. Multicore processors also have low communication time between multiple processes compared to cluster computing which uses a computer network to communicate across multiple machines. These advantages are especially suitable for the problem of likelihood calculation. Graphic Processing Units are a popular choice recently due to their power efficiency per computation. However, gaining speedup from these platforms requires hard work. The tools for GPU programming are difficult to generalize across different platforms. There are various types of parallel programming tools such as OpenMP, CUDA, Stream and OpenCL [13-15]. These tools require programmers to carefully balance computation load across multiple processors. These tools are platform dependent to some extent; for example, Nvidia GPU uses CUDA [13] and ATI GPU uses Stream [14]. Currently, Amazon Web Services [16] provides easy to use, on demand, self-configured machines allowing users to assemble a large system for computation intensive tasks easily. The proposed method can be used in such systems.

The proposed method uses multiple cores to execute the likelihood calculation in parallel. The high performance is achieved by carefully considering the memory access pattern to avoid memory access conflict between multiple concurrent calculations. The choice of software also supports the improvement of performance because the load balancing is performed automatically by the compiler. The *parallel-for* construct in the programming language simplifies the programming task.

The experiment on large data shows that the proposed method achieves good speedup without sacrificing the accuracy of the results. The reported results have not been compared to the results of the ADMIXTURE software proposed by Alexander et al. because his software runs on the different platforms than the one used in this work (they run only on Linux and OSX platforms). However, the block relaxation method does not guarantee convergence from random initial points so it may not be a worthwhile comparison.

Appendix

Implementation

This section explains how to transform for-loop into a *parallel-for* construct. The following part of the code shows the calculation of maximization and expectation. The software program starts from allocate the latent variable, E, which consists of I individuals, M markers and K ancestry groups. This variable is used to represent the inheritance ratio of the mth marker, ith individual from each ancestry group. The value of E has two sides, each side represents each trait. E' represents the other side of a trait. The calculation of equation (3) is shown in the section below.

```
updateQ_Origina1 {
    for every Individuals, i
        for every Ancestry groups, k
            for every Markers, m
                case the last column :    q_ik = (q_ik + E_imk + E'_imk)/ (2 × Marker size)
                case the beginning   :    q_ik = E_imk + E'_imk
                otherwise            :    q_ik += E_imk + E'_imk
}
```

The code shows that at the beginning of every column, the previous value must be cleared. The sum is accumulated through the last column. The last individual admix ratio will be divided by 2 × marker because there are two sides of trait per one marker. The ancestry allele frequency in equation (2) is a kind of conditional cumulative sum which is slightly different from the calculation in equation (3). The ancestry allele frequency, P_{km}, is calculated similar to the allele frequency calculation. P_{km} represents the major ancestry allele frequency and P'_{km} represents the minor allele frequency. The code section of the ancestry allele frequency calculation is shown below.

```
updateP_Original {
    Clear value of P_km and norm_km
    for every Markers, m {
        for every Individuals, i
            case Homozygosity wild type
                for every Ancestry groops, k
                    p_km += E_imk + E'_imk
                    norm_km += E_imk + E'_imk
            case Heterozygosity wild type
                for every Ancestry groups, k
                    p_km += E_imk
                    p'_km += E'_imk
```

```
                    normkm += Eimk + E′imk
            case Homozygosity variance
                for every Ancestry groups, k
                    p′km += Eimk + E′imk
                    normkm += Eimk + E′imk
    }
    normalize the allele frequency of each ancestry group.
    for every markers, m
        for every Ancestry groups, k
            pkm /= normkm
            p′km /= normkm
}
```

The code above can be transformed into parallel version as follows.

```
calculateQ (i,k,m) {
         case the last column  :  qik = (qik + Eimk + E′imk)/(2 × Marker size)
         case the beginning    :  qik = Eimk + E′imk
         otherwise             :  qik += Eimk + E′imk
}
updatePQ_Parallel {
  parallel_for every Markers, m {
        for every Ancestry groups, k
            pkm = ⊘.⊘
            p′km = ⊘.⊘
            normkm = ⊘.⊘
  }
  parallel_for enery Markers, m {
        for every Individuals, i
            case Homozygosity wild type
                for every Ancestry groups, k
                    pkm += Eimk + E′imk
                    normkm += Eimk + E′imk
                    calculateQ (i,k,m)
            case Heterogeneous wild type
                for every Ancestry groups, k
                    pkm += Eimk
                    p′km += E′imk
                    normkm += Eimk + E′imk
                    calculateQ (i,k,m)
            case Homozygosity variance
                for every Ancestry groups, k
                    p′km += Eimk + E′imk
                    normkm += Eimk + E′imk
                    calculateQ (i,k,m)
   }
   parallel_for every Markers, m {
       for every Ancestry groups, k
           pkm /= normkm
           p′km /= normkm
  }
}
```

The code shows the simultaneous calculation of *P* and *Q* in the single *parallel-for* which save the calculation time. The implementation in the expectation step has a major difference from single variable access of likelihood. The likelihood values are distributed as an array of values to avoid the bottleneck caused by many calculation threads accessing the same variable. Although Microsoft dot Net framework provides a method to support mutual exclusion, it can slow down the execution. The critical section was used to protect against multiple accesses from more than one

calculation unit. The likelihood calculation is embedded in the expectation step. For the ancestry allele frequency, only the major allele frequency was stored. The minor allele frequency can be inferred by 1-major allele frequency. Other values such as homozygosity wild type, heterozygosity wild type and homozygosity variance can be derived from the allele frequency. The original code (non-parallel) section is shown below.

```
updateEandCalLikelihood_Original {
    for every Individuals, i
        for every Markers, m
            case Homozygosity wild type
                E_imk = q_ik×p_km
                E'_imk = q_ik×p_km
                likelihood += 2×log(q_ik×p_km)
            case Heterozygosity wild type
                E_imk = q_ik×p_km
                E'_imk = q_ik×p'_km
                likelihood += log(q_ik×p_km) + log(q_ik×p'_km)
            case Homogygosity variance
                E_imk = q_ik×p'_km
                E'_imk = q_ik×p'_km
                likelihood += 2×log(q_ik×p'_km)
}
```

The section of code below shows the application of the *parallel-for* construct to the original code. The likelihood value needs a critical section to protect it from concurrent access.

```
updateEandCalLikelihood_CriticalSection {
    parallel_for everylndividual, i {
         for every Markers, m
            case Homozygosity wild type
                E_imk = q_ik×p_km
                E'_imk = q_ik×p_km
                critical section { likelihood += 2×log(q_ik×p_km) }
            case Heterozygosity wild type
                E_imk = q_ik×p_km
                E'_imk = q_ik×p'_km
                critical section { likelihood += log(q_ik×p_km) + log(q_ik×p'_km) }
            case Homozygosity variance
                E_imk = q_ik×p'_km
                E'_imk = q_ik×p'_km
                critical section { likelihood += 2×log(q_ik×p'_km) }
    }
}
```

Finally, the likelihood value is distributed as an array of values and hence allows many computation units to access them concurrently.

```
updateEandCalLikelihood _parallel{
    parallel_for every Individuals, i {
         for every Markers, m
            case Homozygosity wild type
                E_imk = q_ik×p_km
                E'_imk = q_ik×p_km
                Likelihood_i += 2×log(q_ik×p_km)
```

```
            case Heterozygosity wild type
                E_imk = q_ik×p_km
                E'_imk = q_ik×p'_km
                Likelihood_i += log(q_ik×p_km) + log(q_ik×p'_km)
            case Homozygosity variance type
                E_imk = q_ik×p'_km
                E'_imk = q_ik×p'_km
                Likelikood_i += 2×log(q_ik×p'_km)
    }
}
```

Acknowledgment

The authors would like to thank the National Centre of Genetic Engineering and Biotechnology (BIOTEC), Thailand, for the data used in the experiments.

References

[1] J. C. Long, The genetic structure of admixed population, Genetics 127 (1991), 417-428.

[2] N. Patterson, N. Hattangadi, B. Lane, K. E. Lohmueller, D. A. Hafler, J. R. Oksenberg, S. L. Hauser, M. W. Smith, S. J. O'Brien, D. Altshuler, M. J. Daly and D. Reich, Methods for high-density admixture mapping of disease genes, Am. J. Human Genetics 74 (2004), 979-1000.

[3] C. L. Hanis, R. Chakraborty, R. E. Ferrell and W. J. Schull, Individual admixture estimates: disease associations and individual risk of diabetes and gallbladder disease among Mexican-Americans in Starr County, Texas, Am. J. Phys. Anthropol. 70(4) (1986), 433-441.

[4] C. Bonilla, M. D. Shriver, E. J. Parra, A. Jones and J. R. Fernández, Ancestral proportions and their association with skin pigmentation and bone mineral density in Puerto Rican women from New York city, Hum. Genet. 115 (2004), 57-68.

[5] D. Falush, M. Stephens and J. K. Pritchard, Inference of population structure using multilocus genotype data: linked loci and correlated allele frequencies, Genetics 164 (2003), 1567-1587.

[6] C. L. Pfaff, J. Barnholtz-Sloan, J. K. Wagner and J. C. Long, Information on ancestry from genetic markers, Genetic Epidemiology 26(4) (2003), 305-315.

[7] L. Chikhi, M. W. Bruford and M. A. Beaumont, Estimation of admixture proportions: a likelihood-based approach using Markov chain Monte Carlo, Genetics 158 (2001), 1347-1362.

[8] H. Tang, P. Jie, P. Wang and N. J. Risch, Estimation of individual admixture: analytical and study design considerations, Genetic Epidemiology 28 (2005), 289-301.

[9] A. P. Dempster, N. M. Laird and D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, JSTOR Series B 39 (1977), 1-38.

[10] D. H. Alexander, J. Novembre and K. Lange, Fast model-based estimation of ancestry in unrelated individuals, Genome Res. 19(9) (2009), 1655-1664.

[11] Parallel Framework. http://msdn.microsoft.com/en-us/library/dd460693(v=vs.110).aspx, 2014.

[12] Bovine HapMap Consortium, R. A. Gibbs, Genome-wide survey of SNP variation uncovers the genetic structure of cattle breeds, Science 324(5926) (2009), 528-532.

[13] CUDA: http://developer.nvidia.com/category/zone/cuda-zone, 2014.

[14] ATI Stream: http://developer.amd.com/archive/gpu/ATIStreamSDKv2.3, 2014.

[15] OpenCL, 2014, https://www.khronos.org/opencl/.

[16] Amazon Elastic Compute Cloud, 2014, http://aws.amazon.com/ec2.

[17] A. Heinecke, M. Klemm and H.-J. Bungartz, From GPGPU to many-core: Nvidia Fermi and Intel many integrated core architecture, Comput. Sci. Eng. 14 (2012), 78-83.