Very Large Scale ReliefF Algorithm on GPU for Genome-Wide Association Study

Kwan-Yeung Lee[1,*], **Pengfei Liu**[2,*], **Kwong-Sak Leung**[3] **and Man-Hon Wong**[4]
Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong, China

Abstract— *The advancement in DNA sequencing technology has led to an information bloom on sequencing data and the rise of new data-driven researches like genome-wide association study (GWAS). One major challenge in GWAS is to identify a small set of disease associated single nucleotide polymorphisms (SNPs) out of millions of them in the human genome. Feature selection is a popular technique to reduce the number of features which can boost the efficiency of follow-up analyses. In this paper, we optimized a feature selection algorithm specifically designed for GWAS called Very Large Scale ReliefF (VLSRF) through General-Purpose Computing on Graphics Processing Unit (GPGPU). Experimental results showed that our GPU-based VLSRF and ReliefF achieved up to 100 times speed up relative to a CPU based ReliefF on synthetic datasets. On the other hand, our GPU-based VLSRF was tested on a real dataset from a GWAS Parkinson's disease study in the National Human Genome Research Institute [1]. The SNPs identified by our GPU-based VLSRF are consistent with existing literature.*

Keywords: GPU, feature selection, VLS Relief, Parallization

1. Introduction

Nowadays, next-generation sequencing technologies allow scientists to sequence a large number of samples in a reasonable time and cost [2], [3] which encourages biologist to perform research in data-driven manner e.g. genome-wide association studies (GWASs). In GWAS, statistical or computational analyses are applied to compare the DNA sequences of healthy samples (controls) and patients of a specific genetic disease (cases) in order to identify single nucleotide polymorphisms (SNPs) [4] that are associated to complex genetic diseases like cancer [5] and hepatitide [6].

A single nucleotide polymorphism (SNP) is a genetic variation of a single nucleotide at a specific location of the DNA sequences across the samples in a population. As human is a diploid and each chromosome has two different copies, the genotype of a SNP is composed by two alleles (nucleotides). A SNP can be associated to a genetic disease as a single SNP or as a SNP-SNP interaction. SNPs which are associated to a genetic disease independently could be found through statistical test like p-value testing. Although most of the independent disease associated SNPs were found in various research, those SNPs alone were insufficient in explaining the heritability of those genetic diseases they were associated to. One possible reason is that some disease associated SNP-SNP interactions remain undiscovered. However, it is difficult to detect these interactions. It is well-known that independent marginal effect of each SNP in most SNP-SNP interactions is small [7], [8]. Therefore non-linear SNP-SNP interactions can model the cause of a genetic disease better than linear models. However, detecting non-linear disease-associated SNP-SNP interactions suffers from the curse of dimensionality. One possible approach to improve the efficiency is to reduce the search space through carefully selecting a subset of SNPs that have high potential to be disease associated.

Relief-F is one of the most popular feature selection algorithms in analyzing GWAS datasets. Relief-F is proved to be better in selecting the interacting SNPs than traditional feature selection algorithms like chi-square or information gain which are based on the statistic of each individual feature [9]. Since the accuracy of ReliefF does not scale up well under the large number of SNPs in real GWAS datasets, an enhanced version of Relief-F called Very Large Scale Relief-F (VLSRF) is developed to improve its accuracy. However, VLSRF had a significant higher time complexity than its predecessor [10].

Since ReliefF and VLSRF has a high potential to be parallelized, we propose a GPU implementation on them using CUDA to significantly reduce the time for biologists in prioritizing a list of potential disease associated SNPs for their researches. We believe this can directly accelerate the process of finding SNPs and/or genes that relating to diseases and designing new drugs and diagnosis.

GPU is a high-performance multi-core processor with high computation and data throughput [11]. A modern GPU can perform significantly faster than a CPU in many complex operations like vector and matrix operations or floating point arithmetics. Hence, it has become a popular computing solution for analyses on experimental data in various areas, such as weather forecasting [12], molecular dynamics [13] and fluid-flow [14]. Moreover, it can be programmed easily to perform general purpose computation through high-level programming language like CUDA or OpenCL. CUDA is shown to perform better than OpenCL [15], and CUDA is capable of handling large biological experimental data [16].

[*]These two authors contribute equally to the work.

Table 1: Encoding Table for SNP Genotype

Orginal Genotype (A = major allele, a = minor allele)	Encoded Value
Missing	0
AA	1
Aa	2
aa	3

In most cases, an allele of a SNP could only has two nucleotide types. A major allele has a nucleotide type that appears more frequently in its corresponding SNP across the population while a minor allele has a nucleotide type that appears less frequently in its corresponding SNP across the population.

In this paper, we are developing a GPU-based VLSRF implementation optimized for GWAS dataset. By exploiting the parallelization power of CUDA, reducing number of feature subsets through a novel feature subset enumeration process and reducing the time cost and space used in storing and loading the dataset through simple compression, our GPU-based VLSRF out-performed CPU-based Relief and CPU-based VLSRF in terms of both performance and accuracy. Furthermore, our experiments demonstrated that our GPU-based VLSRF could detect disease associated SNPs in a real Parkinson's disease dataset.

Here is the layout of this paper. First, related work will be given in section 3. Then, details of our implementation will be discussed in section 4. Finally, experimental results will be shown and analyzed in section 5.

2. Problem definition

Input: A dataset that stores the genotypes of a common list of features (SNPs) of a number of case and control samples. The genotype of each SNP is encoded through the schema shown in table 1.
Output: An association score for each SNP against the target genetic disease of the inputted dataset.

3. Related Work

With the increasing popularity of data mining, the application of feature selection has been widen in various research area [17]–[21]. Generally speaking, there are two different kinds of feature selection algorithms. The first kind utilized one or more statistical properties of each feature such as information entropy and t-test value, to search for potential useful features [22]. The second kind exploits the topology or the structure in the data space [23]–[25] to search for potentially important features by measuring the inner relationships between samples and features.

As SNP-SNP interactions are most likely to be non-linear, pure statistical algorithms which measured the statistical property of each SNP perform poorly in GWAS. Meanwhile structural based algorithms like Relief algorithm has been proven to be efficient and effective in prioritizing SNPs in synthetic GWAS datasets [9].

```
Relief Algorithm
Initialize feature score vector $w$
for n = 1 : no. of iteration pre-defined
    Randomly select a sample $x$ from sample set $S$
    Find the nearest hit $NH_x$ and nearest miss $NM_x$ of $x$
    for i = 1 : no. of features
        $w[i] = w[i] + |x[i] - NM_x[i]| - |x[i] - NH_x[i]|$
    end
end
```

Fig. 1: Pseudocode of Relief Algorithm

Relief is a probabilistic algorithm which evaluates every feature in a dataset in the following steps [24]. First, a sample χ is randomly selected and its pairwise distances with other samples are calculated. Second, the nearest neighbor with the same phenotype (hit) or nearest neighbor with a different phenotype (miss) are found. Third for each feature, its value in the selected sample χ is compared against its value in the corresponding nearest hit and miss. Features that are more consistent within samples from the same class and inconsistent across samples from different classes will obtain a higher score. Otherwise, it will have a lower score. Finally, the score of each feature is updated through repeating the three steps above multiple times. Its pseudocode is shown in figure 1.

On the other hand, ReliefF is a Relief based algorithm which can handle datasets with multiple classes and continuous value features while maintaining a similar time complexity as its predecessor [25]. The major improvement of ReliefF over its predecessor is that it compares each randomly selected sample against a small group of nearest hit and miss during the feature scoring process instead of only one nearest hit and miss. This eliminates the aversive effects of outliners and has improved the overall accuracy. Moreover, ReliefF was further enhanced and a new algorithm called Tuned ReliefF was developed [9]. Tuned ReliefF achieved a higher detection power than its predecessor through repeatedly executing ReliefF algorithm and gradually removing a small number of low ReliefF score features after each ReliefF execution. As ReliefF is executed iteratively, Tuned ReliefF is significant slower than ReliefF algorithm.

Iterative ReliefF is another Relief based algorithm [26]. Similar to ReliefF algorithm, it compares each randomly selected sample to more than one nearest hit and one nearest miss. First, it calculates the probability of each randomly selected sample χ being a outliner and the probability of other sample being the nearest hit or the nearest miss of

```
VLSReliefF Algorithm
1 Initialize global feature score vector w
2 for i = 1 : no. of feature subset
3     Enumerate the i^th feature subset S
4     Form a reduced dataset D′ with features in S only
5     Perform ReliefF on D′ to calculate local score w′
6     for i =  1 : no. of features
7         w[i] = max(w[i], w′[i])
8     end
9 end
```

Fig. 2: Pseudocode of VLS ReliefF Algorithm.

χ. After that, it compares each randomly selected sample χ against all other samples and updates the score of each feature according to the nearest hit/miss probability of other samples and outlining probability of the selected sample previously calculated. The above two steps will be repeated until the scores of all features converge.

Very Large Scale ReliefF (VLSRF) selects important features from a dataset with enormous number of features through divide and conquer [10]. First, it splits the original feature set into a full collection of feature subsets with a user specific size. The size of feature subset is significantly smaller than the original feature set to ensure that ReliefF can maintain a high detection power on those subsets. Second, the local score of every feature under each feature subset is calculated independently through ReliefF algorithm. Finally, all local feature scores calculated by each independent ReliefF execution are merged and the final global score of each feature is its maximum local score. Although VLSRF maintains a high detection accuracy under extremely large number of features, it has a much higher time complexity than ReliefF as a trade off. To ensure at least one feature subset contains all important features and such that their association can be measured, an exponential number of feature subsets are needed to be evaluated. Its pseudocode is shown in figure 2.

4. Implementation Detail

In this section, we will describe the details of our GPU-based VLSRF implementation. The overall flow chart of this implementation is shown in figure 3.

4.1 Data Compression

As a SNP has at most 3 different genotypes, every feature in a GWAS dataset has a very small domain and 2 bits are sufficient for representing any genotype of a SNP. Therefore, any dataset inputted will be compressed through converting every feature into a 2 bit integer and tightly packing every 4 features into a byte before storing into the host main memory and GPU device memory. This reduces the amount of memory required for storing the dataset significantly and allowed GPU with a limited device memory can store a larger dataset. It also reduced the run-time overhead in copying the dataset from main memory to GPU device memory. Finally, only a constant number of binary operations are needed for accessing the value of a feature from a compressed dataset, so it maintains the performance of ReliefF while improving the efficiency of loading data into GPU device memory.

4.2 Feature Subset Enumeration

To reduce the number of feature subsets to be evaluated, we proposes a new feature subset formation process. In this new process, all features in the inputted dataset are divided into different groups. A feature subset is then generated by combining several feature groups in stead of randomly selecting a number of features from the original set of features. The details of this approach is shown in figure 3.

We apply a binary operation based subset enumeration algorithm called GosperâĂŹs hack [27] to select feature groups for forming feature subsets. In this algorithm, a binary string is representing the membership of each feature group of a subset and it has a length of the number of feature groups. If the ith bit in the binary string is one, then the ith feature group is a member of the current feature subset. Otherwise, it is not a member of the subset. Furthermore, in this algorithm, a binary operation based enumerator is used to enumerate the membership of feature groups of the next new subset in O(No. of features) binary operations based on a given subset. Therefore, every subset can be generated through iterative use of the enumerator. Therefore, this algorithm minimized the time needed in enumerating each feature subset.

4.3 GPU-Based ReliefF Algorithm

The ReliefF algorithm is implemented to be executed entirely under GPU to minimize the data transfer between the host PC memory and the GPU device memory. Some operations in the ReliefF are optimized through exploiting the parallelization processing power of GPU. First, the distance of a pair of sample does not have dependency with the distance of other pairs of samples and can be calculated independently. Second, a list of randomly selected samples are compared against their corresponding group of nearest hits and group of nearest misses in a feature by feature wise manner. These feature value comparisons are independent from each other and can be performed simultaneously. Therefore, two separated CUDA kernel programs are developed for performing these two operations. These two CUDA kernel programs are executed under multiple GPU threads in parallel automatically.

5. Result and Discussion

In order to evaluate the performance of our GPU based VLSRF implementation, numerous experiments were per-

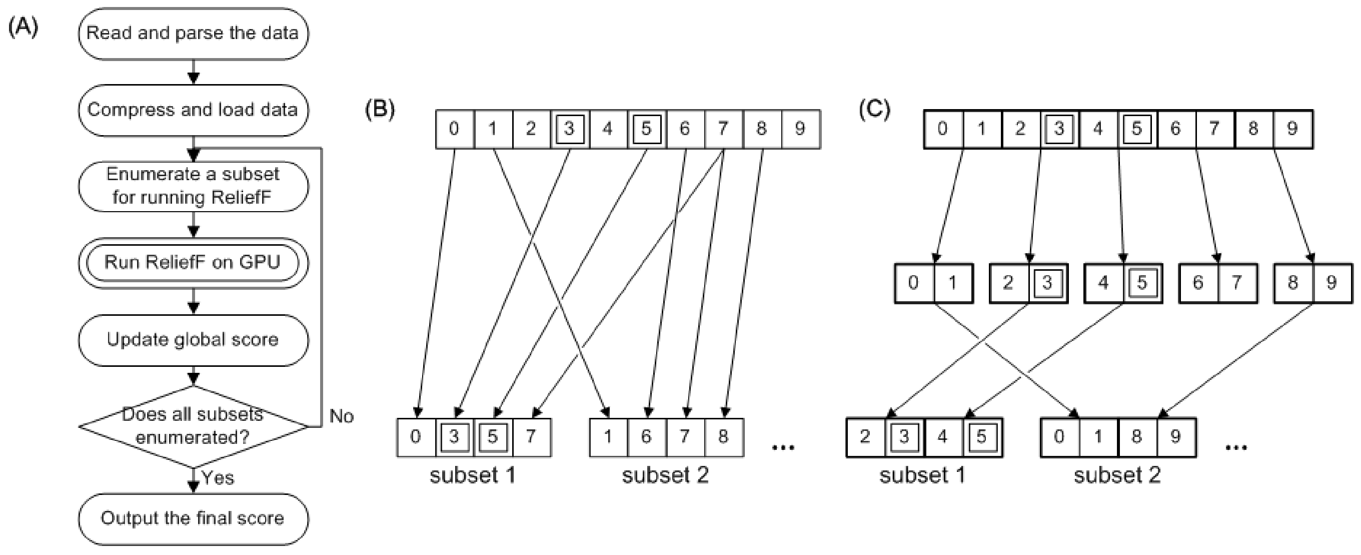

Fig. 3: (A) shows the execution flow of our GPU-based VLS ReliefF. In (A), the block with double line border is performed under GPU and other blocks are executed under CPU. (B) shows an example on the original process of enumerating feature subsets in VLSRF. Under 10 features and process (B), there are totally $\binom{10}{4} = 210$ ways to form feature subsets with 4 features. (C) shows an example on our novel process for enumerating feature subsets in VLSRF under 10 features. First, the original feature set is split into 5 feature groups where each of these group carries 2 features. Second, each feature subset is composed by 2 randomly selected feature groups. As a result, there are $\binom{5}{4} = 5$ total possible feature subsets with size = 4 which is significant fewer than process (B). On the other hand, the new process will still able to select all important features into the same feature subset and ensure at least 1 pass of Relief algorithm can detect these features as long as all possible subsets are tested. As seen in (C), feature '3' and '5' (i.e. the block with double line border) are both important features and they are both the member of subset 1.

formed on various synthetic and real datasets. In this section, we will discuss the results of those experiments in detail.

5.1 Machine Configuration

All experiments on CPU implementations were performed on a workstation with one Intel I5-4670 3.4GHZ CPU and 8GB DDR3 main memory. On the other hand, all experiments on the GPU implementations were performed on a high performance Linux server with two Intel Xeon E5-2670 2.6GHZ CPU, 128GB ECC DDR3 main memory and a NVIDIA GK110GL Kepler GPU.

5.2 Experiment on Synthetic Datasets

We performed experiments to compare the speed and accuracy of our GPU-based VLSRF against CPU-based ReliefF and CPU-based VLSRF. The performance metrics were the ranking of predictive feature and execution time.

5.2.1 Details of Synthetic Dataset Generator

All synthetic datasets were generated by GAMETES which is a dataset generator specifically designed for generating highly accurate GWAS synthetic dataset [28]. We generated four groups of datasets under different heritabilities using GAMETES and each dataset group contained 50 datasets. The configurations on generating those datasets are shown in table 2.

5.3 Runtime Parameters

The parameters used to run the experiments are shown in table 3.

5.4 Predictive feature Ranking Comparison

Numerous experiments were conducted to compare the accuracy between ReliefF and VLSRF. Features in each dataset in the four dataset groups were ranked and the ranking of the two predictive features were compared as the measurement for accuracy. Furthermore, we performed our experiments on VLSRF under two different feature grouping sizes to understand the behaviour of our novel feature subset enumeration procedure. All the results were plotted as charts and shown in figure 4.

In our experiment, for ReliefF all the predictive features were ranked top half in all the experiments. It is clear that ReliefF can differentiate predictive features from other features effectively. The numbers of datasets that the first and second predictive features were ranked top 20% under heritability 0.1, 0.2, 0.3 and 0.4 were 17, 9, 19, and 16 and 9, 18, 25 and 22 out of 50 datasets respectively. Therefore generally speaking, the performance of ReliefF increased with

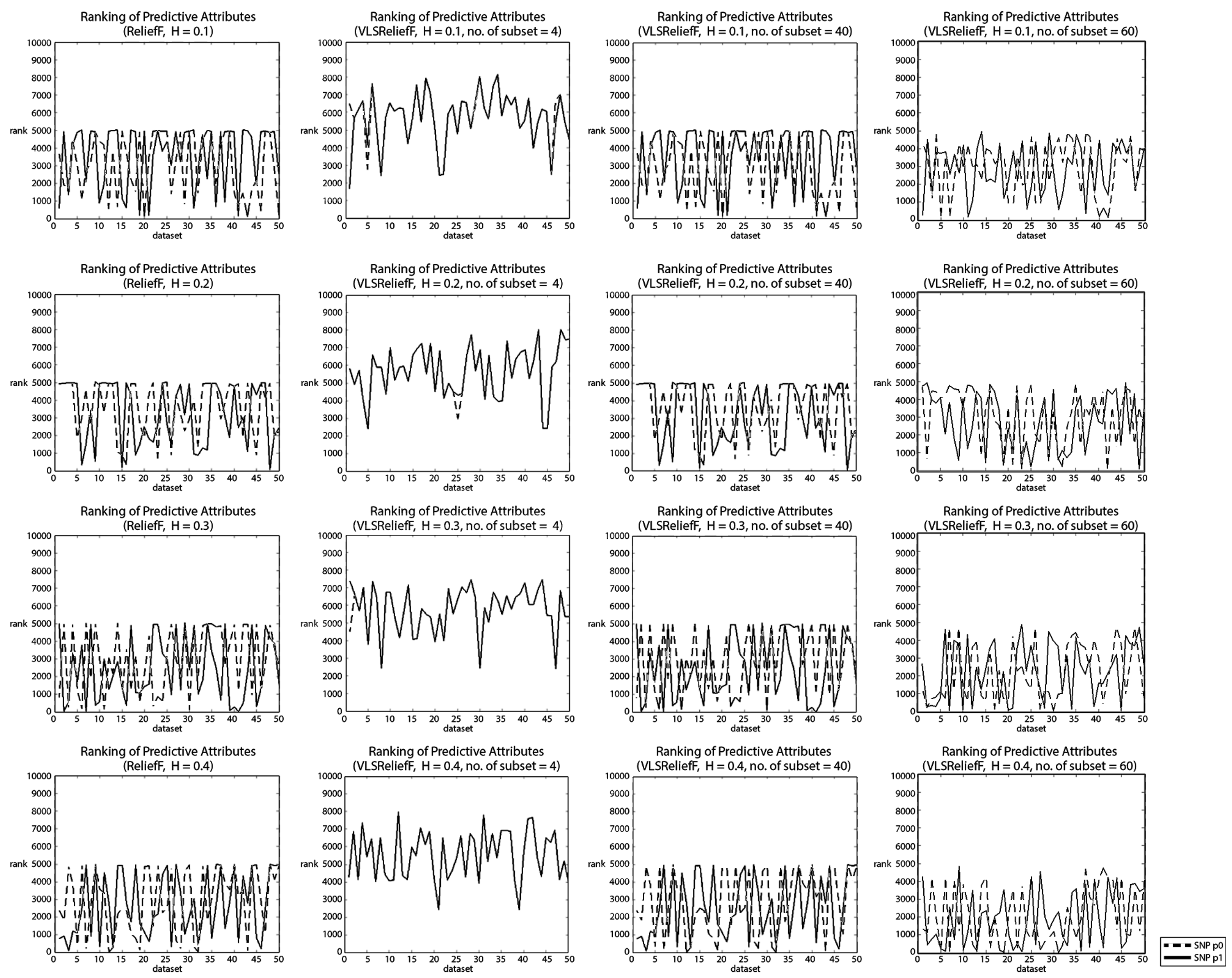

Fig. 4: Ranking of two predictive features of groups of synthetic datasets with different heritabilities under ReliefF and VLS ReliefF algorithm. For each chart, the horizontal axis stands for a dataset and the vertical axis stands for the ranking of predictive features. Moreover, the dotted line and solid line in each chart represent the rankings of the first(p0) and second(p1) predictive feature across the datasets.

the increasing heritability. To conclude, ReliefF performed as expected and it can be used as a benchmark for comparison.

By observing the graph, for VLSReliefF we can see that as the number feature subset increases, the ranks of the two predictive features increases. This shows that under the same number of feature subset enumerated and tested, the scores of the features converge faster with a larger feature group size. This shows that our novel feature subset enumeration procedure is effective in reducing feature subsets needed to be enumerated.

When VLSRF was executed under four feature subsets, only the second predictive feature was ranked top 20% once. Moreover the numbers of datasets that the first and second predictive features were ranked top 20% under heritability 0.1, 0.2, 0.3 and 0.4 were 18, 11, 16, and 16 and 13, 16, 25 and 22 out of 50 datasets respectively if VLSRF is executed under 40 feature subsets. It is obvious that VLSRF do not outperform ReliefF subset in these circumstances. On the other hand, it had a significantly higher accuracy than ReliefF under 60 feature subsets as the numbers of datasets that the first and second predictive features were ranked top 20% under heritability 0.1, 0.2, 0.3 and 0.4 were 15, 13, 26, and 29 and 15, 18, 19 and 26 out of 50 datasets respectively. It is clear that VLSRF can perform better than ReliefF as long as the number of feature subset evaluated is large enough.

Table 2: Parameters for Generating Synthetic Dataset

Parameter	Value	Description
Total No. of features	10000	The number of SNPs in the dataset
Predictive feature No	2 (p0, p1)	The number of disease SNPs in the dataset
Heritability	0.1, 0.2, 0.3, 0.4	Proportion of cases that are associated by disease associated SNP
Population Prevalence	Random	The probability for a random sample having disease
Minor Allele Frequency	0.2	The frequency of allele that occur less in the population
No. of Cases and Controls	1000(500:500)	The number of cases and controls in the dataset
No. of EDM	1	The number of model used in generating the dataset

Table 3: Runtime Parameters for ReliefF and VLSReliefF

Parameter	ReliefF	VLSRF
Number of NNs	1	1
Size of Subset	-	5000
Number of Subset	-	4, 40, 60

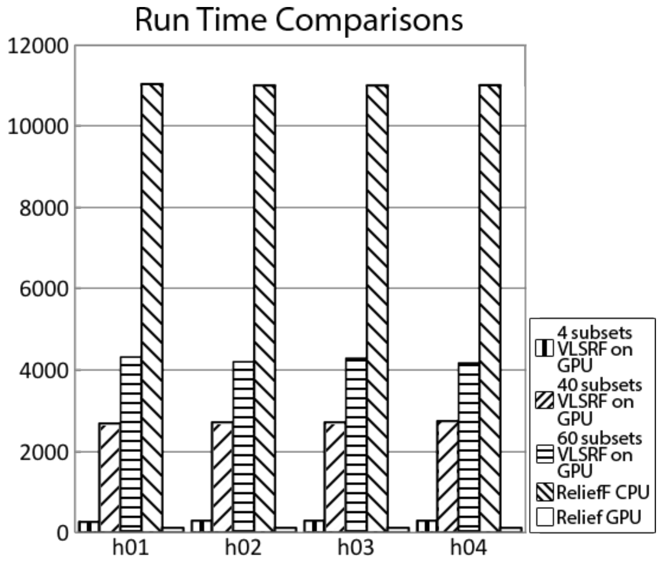

Fig. 5: Run time comparison between CPU and GPU implementations of ReliefF and VLS ReliefF. The horizontal axis stands for heritability of the dataset group and the vertical axis stands for the execution time in seconds.

5.5 Execution Time Comparison

Numerous experiments were conducted to compare the execution time between CPU-based ReliefF and our GPU-based VLSRF. We measured and compared the time needed for both algorithms to finish the execution of the datasets in a dataset group. The experimental time cutoff was 5 hours. All the results were plotted as charts and shown in figure 5.

Our experiments have showed that ReliefF costed around 11,000 seconds and 110 seconds to finish processing all the dataset in a dataset subgroup on CPU and GPU respectively. The GPU version outperformed the CPU version by around 100 times. It is obvious that GPU alone significantly improve the efficiency of ReliefF algorithm.

The run time of VLSRF was not reported as it failed to finish its execution on CPU within the time limit under any number of feature subset we tested. This result is consistent with our understanding of VLSRF. According to section 3, the run time of VLSRF was approximately proportional to the run time of ReliefF by a factor of the feature subset number. Therefore, we could roughly estimate the run time of VLSRF under 4 feature subsets to be 4x11000 = 44000 seconds = 12 hours. Therefore, no CPU based VLSRF could finish its execution before reaching the time limit. On the other hand, GPU-based VLSRF could finish its execution with at most 4300s. This clearly shows that it is far more practical to perform VLSRF on GPU rather than GPU.

The run time of VLSRF under 4, 40 and 60 feature subsets were around 250s, 2700s and 4500s respectively. It is clear that our novel feature selection process is efficient and does not hinder the performance of VLSRF.

5.6 Details of Real Dataset

5.6.1 Dataset Source and Pre-processing

Our GPU-based VLSRF was applied on a Parkinson's disease study called Mayo-Perlegen LEAPS (Linked Efforts to Accelerate Parkinson's Solutions) Collaboration which is originated from National Human Genome Research Institute [1] to evaluate its performance on a real dataset. This study had two tiers and we picked the tier 2 for performing our experiment. Tier 2 had 660 samples (332 case and 332 control) and each sample had 3000 SNPs. Before we performed experiments on this dataset, we had performed data cleansing and converted it into ARFF file format with the encoding scheme described in section 3 such that it could be processed by our programs easily.

5.6.2 Result Analysis

After we computed the score of all SNPs using our GPU-based VLSRF, we ranked those SNPs according to their score and extracted the top 10% SNPs for further analysis. Among those top 10% SNPs, there are 3 SNPs which are associated to Parkinson's disease related gene. These 3 SNPs are rs2287403, rs1979687 and rs2303703 which are associated to YLPM1, USP47 and MYO10 respectively and these 3 genes were all reported to be associated to Parkinson's disease by a Parkinson's disease database [29]. Furthermore, gene USP47 and MYO10 were separately reported to be associated to Parkinson's disease under 2 more literatures [30], [31]. All these showed that our GPU-based VLSRF algorithm are able to select disease associated SNPs effectively.

6. Conclusion

In this paper, we have demonstrated a new GPU-based VLSRF implementation and tested it thoroughly using multiple sets of synthetic datasets and a real GWAS dataset on Parkinson's disease. Under various experiments, our GPU-based VLSRF was shown to be an effective and efficient algorithm in identifying important SNPs. It outperformed existing CPU based VLSRF and ReliefF in term of speed while maintaining a comparable accuracy with them. Under the experiment of Parkinson's disease dataset, it has identified SNPs consistent to existing literature. All these show GPU-based VLSRF is effective and efficient under both real and synthetic datasets.

References

[1] D. M. Maraganore, M. de Andrade, T. G. Lesnick, K. J. Strain, M. J. Farrer, W. A. Rocca, P. K. Pant, K. A. Frazer, D. R. Cox, and D. G. Ballinger, "High-resolution whole-genome association study of parkinson disease," *The American Journal of Human Genetics*, vol. 77, no. 5, pp. 685–693, 2005.

[2] J. N. Hirschhorn and M. J. Daly, "Genome-wide association studies for common diseases and complex traits," *Nature Reviews Genetics*, vol. 6, no. 2, pp. 95–108, 2005.

[3] W. Y. Wang, B. J. Barratt, D. G. Clayton, and J. A. Todd, "Genome-wide association studies: theoretical and practical concerns," *Nature Reviews Genetics*, vol. 6, no. 2, pp. 109–118, 2005.

[4] D. E. Reich and E. S. Lander, "On the allelic spectrum of human disease," *TRENDS in Genetics*, vol. 17, no. 9, pp. 502–510, 2001.

[5] D. F. Easton and R. A. Eeles, "Genome-wide association studies in cancer," *Human Molecular Genetics*, vol. 17, no. R2, pp. R109–R115, 2008.

[6] N. P. Paynter, D. I. Chasman, G. Paré, J. E. Buring, N. R. Cook, J. P. Miletich, and P. M. Ridker, "Association between a literature-based genetic risk score and cardiovascular events in women," *Jama*, vol. 303, no. 7, pp. 631–637, 2010.

[7] J. H. Moore, F. W. Asselbergs, and S. M. Williams, "Bioinformatics challenges for genome-wide association studies," *Bioinformatics*, vol. 26, no. 4, pp. 445–455, 2010.

[8] J. H. Moore, "The ubiquitous nature of epistasis in determining susceptibility to common human diseases," *Human heredity*, vol. 56, no. 1-3, pp. 73–82, 2003.

[9] J. H. Moore and B. C. White, "Tuning relieff for genome-wide genetic analysis," in *Evolutionary computation, machine learning and data mining in bioinformatics*. Springer, 2007, pp. 166–175.

[10] M. J. Eppstein and P. Haake, "Very large scale relieff for genome-wide association analysis," in *Computational Intelligence in Bioinformatics and Computational Biology, 2008. CIBCB'08. IEEE Symposium on*. IEEE, 2008, pp. 112–119.

[11] M. Harris and D. Luebke, "Gpgpu: General-purpose computation on graphics hardware," in *International Conference on Computer Graphics and Interactive Techniques: ACM SIGGRAPH 2005 Courses: Los Angeles, California*, vol. 2005, 2005.

[12] J. Michalakes and M. Vachharajani, "Gpu acceleration of numerical weather prediction," *Parallel Processing Letters*, vol. 18, no. 04, pp. 531–548, 2008.

[13] J. A. Anderson, C. D. Lorenz, and A. Travesset, "General purpose molecular dynamics simulations fully implemented on graphics processing units," *Journal of Computational Physics*, vol. 227, no. 10, pp. 5342–5359, 2008.

[14] P. Bailey, J. Myre, S. D. Walsh, D. J. Lilja, and M. O. Saar, "Accelerating lattice boltzmann fluid flow simulations using graphics processors," in *Parallel Processing, 2009. ICPP'09. International Conference on*. IEEE, 2009, pp. 550–557.

[15] K. Karimi, N. G. Dickson, and F. Hamze, "A performance comparison of cuda and opencl," *arXiv preprint arXiv:1005.2581*, 2010.

[16] R. Jiang, F. Zeng, W. Zhang, X. Wu, and Z. Yu, "Accelerating genome-wide association studies using cuda compatible graphics processing units," in *Bioinformatics, Systems Biology and Intelligent Computing, 2009. IJCBS'09. International Joint Conference on*. IEEE, 2009, pp. 70–76.

[17] S. Beniwal and J. Arora, "Classification and feature selection techniques in data mining," in *International Journal of Engineering Research and Technology*, vol. 1, no. 6 (August-2012). ESRSA Publications, 2012.

[18] F. Min, Q. Hu, and W. Zhu, "Feature selection with test cost constraint," *International Journal of Approximate Reasoning*, vol. 55, no. 1, pp. 167–179, 2014.

[19] J. Tang and H. Liu, "Unsupervised feature selection for linked social media data," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012, pp. 904–912.

[20] ——, "Feature selection with linked data in social media." in *SDM*. SIAM, 2012, pp. 118–128.

[21] C.-P. Lee and Y. Leu, "A novel hybrid feature selection method for microarray data analysis," *Applied Soft Computing*, vol. 11, no. 1, pp. 208–213, 2011.

[22] G. Brown, A. Pocock, M.-J. Zhao, and M. Luján, "Conditional likelihood maximisation: a unifying framework for information theoretic feature selection," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 27–66, 2012.

[23] K. Kira and L. A. Rendell, "A practical approach to feature selection," in *Proceedings of the ninth international workshop on Machine learning*, 1992, pp. 249–256.

[24] ——, "The feature selection problem: Traditional methods and a new algorithm," in *AAAI*, vol. 2, 1992, pp. 129–134.

[25] I. Kononenko, E. Šimec, and M. Robnik-Šikonja, "Overcoming the myopia of inductive learning algorithms with relieff," *Applied Intelligence*, vol. 7, no. 1, pp. 39–55, 1997.

[26] Y. Sun, "Iterative relief for feature weighting: algorithms, theories, and applications," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 6, pp. 1035–1051, 2007.

[27] R. W. Gosper, "Item 175 in beeler, m., gosper, rw, and schroeppel, r., hakmem," 1972.

[28] R. J. Urbanowicz, J. Kiralis, N. A. Sinnott-Armstrong, T. Heberling, J. M. Fisher, and J. H. Moore, "Gametes: a fast, direct algorithm for generating pure, strict, epistatic models with random architectures," *BioData mining*, vol. 5, no. 1, pp. 1–14, 2012.

[29] J. O. Yang, W.-Y. Kim, S.-Y. Jeong, J.-H. Oh, S. Jho, J. Bhak, and N.-S. Kim, "Pdbase: a database of parkinson's disease-related genes and genetic variation using substantia nigra ests," *BMC genomics*, vol. 10, no. Suppl 3, p. S32, 2009.

[30] F. Simunovic, M. Yi, Y. Wang, L. Macey, L. T. Brown, A. M. Krichevsky, S. L. Andersen, R. M. Stephens, F. M. Benes, and K. C. Sonntag, "Gene expression profiling of substantia nigra dopamine neurons: further insights into parkinson's disease pathology," *Brain*, vol. 132, no. 7, pp. 1795–1809, 2009.

[31] K. Gousset, L. Marzo, P.-H. Commere, and C. Zurzolo, "Myo10 is a key regulator of tnt formation in neuronal cells," *Journal of cell science*, vol. 126, no. 19, pp. 4424–4435, 2013.