Accelerate GWAS through Parallel

Weixiao Zhan

March 15, 2024

1 Introduction

The motivation of the project is to address the inefficiencies, specifically time-latency, observed in `plink` computing Genome-Wide Association Studies (GWAS, Uffelmann et al. [2021]). Recognizing the computation of GWAS is highly parallelizable: effect size (β) and significance (p-value) of each Single Nucleotide Polymorphism (SNP) does not depend on another's allele frequency, this project, **parallel GWAS**, aim to reimplement GWAS leveraging hardware parallel capabilities. In addition, run time comparison between `plink` and parallel GWAS, is conducted on two different platforms to evaluate performance.

2 Methods

Matrix arithmetic are well implemented in parallel by open-source libraries. The methodology is to convert GWAS regression problem into matrix arithmetic, and leverage existing libraries to accelerate computation. Specifically, parallel GWAS uses PyTorch library by Paszke et al. [2019].

2.1 preprocessing

Firstly, construct allele frequency matrix X and phenotype matrix Y from file using stream IO.

$$X = \begin{array}{cc} \begin{bmatrix} & \vdots & \\ \cdots & x_{i,j} \in \{0,1,2\} & \cdots \\ & \vdots & \end{bmatrix} & \begin{matrix} \uparrow \\ i \in \text{sample} \\ \downarrow \end{matrix} \\ \leftarrow j \in \text{SNP} \rightarrow & \end{array}$$

$$Y = \begin{bmatrix} \vdots \\ y_i \\ \vdots \end{bmatrix} \quad \begin{matrix} \uparrow \\ i \in \text{sample} \\ \downarrow \end{matrix}$$

Then, center X, Y at their means:

$$\begin{aligned} X_{:,j} &\leftarrow X_{:,j} - \text{mean}\left(X_{:,j}\right) \quad \forall j \\ Y &\leftarrow Y - \text{mean}\left(Y\right) \end{aligned}$$

2.2 effect size

The effect size can be computed as:

$$\hat{\beta} = \frac{X^T Y}{X^T X}$$

2.3 significance (p value)

The null hypothesis H_0 is no association between the phenotype and the genotype, i.e. $\beta = 0$. Thus the test statistics can be computed as

$$\text{stat} = \frac{\hat{\beta} - 0}{SE\left(\hat{\beta}\right)}$$

$$SE\left(\hat{\beta}\right) = \text{var}_{(-2)}(Y - \hat{\beta}X)$$

in which, $\text{var}_{(-2)}(\cdot)$ computes the variance with (n-2) correction.

This statistics follows t-distribution with (number of samples -2) degree of freedom. Thus p-value can be computed as:

$$\text{stat} \sim T_{df=\text{samples}-2}$$

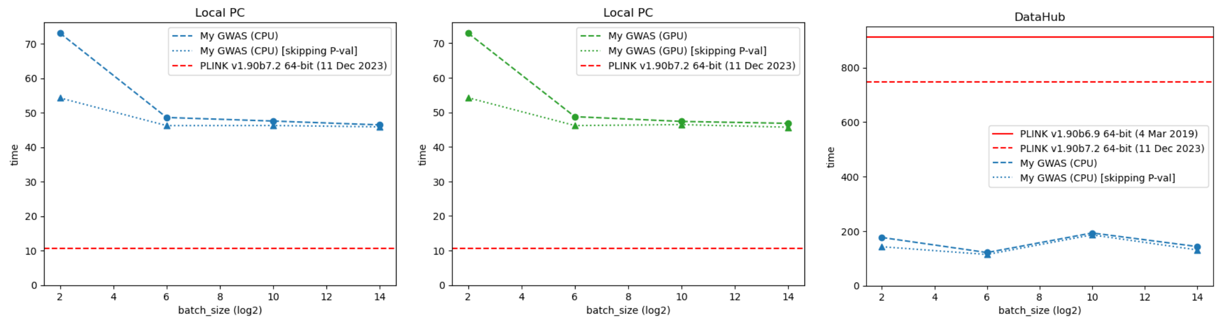
$$p\text{-values} = 2\int_{-\infty}^{-|\text{stat}|} T_{df=\text{samples}-2}(t)dt$$

It is worth mentioning that there is no close formula for t distribution. The last step to compute p-value is performed in batch operations using `scipy` library instead of parallel. Later of this report will discuss the effect of different batch sizes.

3 Results

The correctness of parallel GWAS result is verified by `plink` results.

Run time comparison is conducted on a local PC equipped with Nvidia GPU and datahub. Figure 1a and 1b shows the runtime on local PC using different hardware. Figure 1c shows the runtime on datahub. Since there is no GPU resource available, only CPU version of parallel GWAS is used.



(a) CPU runtime (b) GPU runtime (c) runtime on datahub.

Figure 1: Runtime comparison

4 Discussion

1. In Figure 1, the difference between dash and dot line is the time to compute CDF of T distribution. As the batch size increases, the time computing p-values decreases. However, large batch size may overwhelm the system memory, for instance, in Figure1c, 2^6 batch size produced peak performance. Thus batch size is an important parameter to tune on hardware.

2. In Figure 1a and 1b, runtime difference between CPU and GPU is negligible. Since GWAS has relatively simple matrix computation, benefit of having extra parallel cores on GPU canceled out with the extra communication cost.

3. Comparing Figure 1a and 1c, `plink` performed dramatically different: 80× faster on local PC. Potential cause is `plink 1.9` might used some Intel specific libraries that are not fully supported by AMD. Meanwhile, `plink 2.0`, currently still under development, promise to have versions specificity compiled and built AMD, which should help speed up calculation on datahub machines in the future.

Future work:

1. Add support to handle co-variables, such as principle components of population, etc.

5 Code

github repo: https://github.com/weixiao-zhan/CSE284_ParallelGWAS

References

E. Uffelmann, Q.Q. Huang, N.S. Munung, et al. Genome-wide association studies. *Nat Rev Methods Primers*, 1:59, 2021. doi: 10.1038/s43586-021-00056-9. Accepted: 13 July 2021, Published: 26 August 2021.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.