Parahaplo: A Parallel Tree-based Solver for Haplotype Assembly on CPU-GPU Systems

Robert Clucas
University of the Witwatersrand
School of Electrical and Information Engineering
Johannesburg, South Africa
robert.clucas@students.wits.ac.za

Sasha Naidoo
University of the Witwatersrand
School of Electrical and Information Engineering
Johannesburg, South Africa
sasha.naidoo2@students.wits.ac.za

***Abstract*—Haplotyping is an important problem in bioinformatics, and can significantly impact areas such as disease diagnosis and drug discovery. Organisms such as humans have two copies of each chromosome, and haplotyping partitions genotype calls between the two chromosomes. The Minimum error correction (MEC) formulation of the haplotype assembly (HA) problem is NP-hard and involves correcting the minimum number of SNPs to infer the haplotypes from a set of aligned reads. Most existing solutions either make assumptions to reduce complexity, or require impractical computation times. In this work, we present Parahaplo, a parallel tree-based solver for the general case HA problem where columns are both heterozygous and homozygous, and reads may contain multiple gaps.**

Input matrices obtained from DNA sequencing technology are typically extremely large, up to hundreds of Gigabytes, and hence cannot fit into GPU memory. To overcome this, Parahaplo uses many-core CPUs to split the input matrices into uncorrelated sub-blocks, which are then represented as a binary tree, where each node is a position in the haplotype. Additionally, the CPUs find the root node of each tree, as well as the correlation between the haplotype positions. The trees for each sub-block are solved in parallel on the GPUs using a parallel, breadth first branch-and-bound algorithm. The breadth first search utilises the massively-parallel nature of GPUs and ensures the tree is always balanced, minimizing the number of underutilized cores and limiting thread synchronization. Highly parallel bounding operators are designed to efficiently determine the upper and lower bound, allowing nodes which cannot provide an optimal solution to be pruned early in the search, limiting the size of the search tree.

***Keywords*-Haplotype; CPU; GPU; Tree; Branch; Bound;**

I. Introduction

Haplotypes form a key part in the field of bioinformatics and plays a fundamental role in genetic analysis such as in genetic variation analyses, gene disease diagnoses and drug discovery [1]. A haplotype is defined as an ordered set of single nucleotide polymorphisms (SNPs) occurring at specific sites on a single chromosome. Humans have a pairs of these chromosomes and are such termed diploid organisms. Therefore, humans have a haplotype associated with each chromosome. It is common for there to be only two nucleotide bases for most SNPs, known as biallelic SNPs [2]. This is a common assumption made when trying to solve for haplotypes. DNA sequencing reads for both alleles at a specific position, known was the genotype, but is unable to determine which chromosome the allele originated from [3]. This is known as the haplotyping problem. It is also known that a correlation exists between alleles in close proximity. Inference methods use this fact in conjunction with genotype data to infer the haplotypes. This class of methods, however, relies on the correlation between allele and thus struggles to identify rare or unique haplotypes [3]. New high-throughput sequencing technology produces a set short read fragments which can be combined to determine the two correct haplotypes [1]. This class of methods is termed *Haplotype Assembly* (HA). Computational methods are therefore required in order to process this data.

II. Background and Literature Review

A. Haplotype Assembly Problem

actual problem
solutions

B. Hardware Implementations

cpu
gpu
cpu-gpu

III. Parahaplo Solver

data transferral diagram

A. Data Processing and Manipulation

1) Data Conversion:
2) Block Decomposition:
3) Data Correlation:

B. Solving

1) Node Selection:
2) Tree Creation: Memory management
3) Tree Search:

IV. Results

Plain cpu cpu-gpu
small datasets, large datasets

V. Future Recommendations

VI. Conclusion

The conclusion goes here. this is more of the conclusion

Acknowledgment

References

[1] Z.-Z. Chen, F. Deng, and L. Wang, "Exact algorithms for haplotype assembly from whole-genome sequence data," *Bioinformatics*, vol. 29, no. 16, pp. 1938–1945, 2013. [Online]. Available: http://bioinformatics.oxfordjournals.org/content/25/1/68.abstract

[2] S. Climer, G. Jger, A. R. Templeton, and W. Zhang, "How frugal is mother nature with haplotypes?" *Bioinformatics*, vol. 25, no. 1, pp. 68–74, 2009. [Online]. Available: http://bioinformatics.oxfordjournals.org/content/25/1/68.abstract

[3] D. He, A. Choi, K. Pipatsrisawat, A. Darwiche, and E. Eskin, "Optimal algorithms for haplotype assembly from whole-genome sequence data," *Bioinformatics*, vol. 26, no. 12, pp. i183–i190, 2010. [Online]. Available: http://bioinformatics.oxfordjournals.org/content/26/12/i183.abstract