GHOSTM: A GPU-Accelerated Homology Search Tool for Metagenomics

Shuji Suzuki[1], Takashi Ishida[1] and Yutaka Akiyama[1]
[1] Graduate School of Information Science and Engineering, Tokyo Institute of Technology

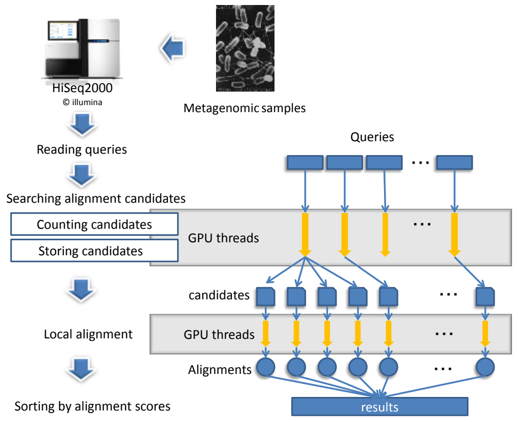
Abstract

Background:
A large number of sensitive homology searches are required for mapping DNA sequence fragments to known protein sequences in public and private databases during metagenomic analysis. BLAST[1] is currently used for this purpose, but its calculation speed is insufficient, especially for analyzing the large quantities of sequence data obtained from a next-generation sequencer. However, faster search tools, such as BLAT[2], do not have sufficient search sensitivity for metagenomic analysis. Thus, a sensitive and efficient homology search tool is in high demand for this type of analysis.

Methodology:
We developed a new, highly efficient homology search algorithm suitable for graphics processing unit (GPU) calculations that was implemented as a GPU system that we called GHOSTM[3]. The system first searches for candidate alignment positions for a sequence from the database using pre-calculated indexes and then calculates local alignments around the candidate positions before calculating alignment scores. We implemented both of these two processes on GPUs. The system achieved calculation speeds that were 165 and 438 times faster than BLAST with 1 GPU and 4 GPUs.

Conclusions:
We developed a GPU-optimized algorithm to perform sensitive sequence homology searches and implemented the system as GHOSTM. Currently, sequencing technology continues to improve, and sequencers are increasingly producing larger and larger quantities of data. This explosion of sequence data makes computational analysis with contemporary tools more difficult. We developed GHOSTM, which is a cost-efficient tool, and offer this tool as a potential solution to this problem.

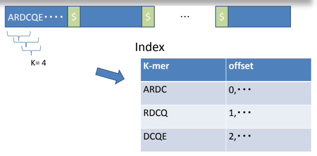
Overview of the method



1.Indexes for database is constructed beforehand.
2.The candidate alignment positions for a sequence from the database are found by using the indexes.
3.The local alignments are calculated around the candidate positions using the Smith – Waterman algorithm.
4.The alignments are sorted by the alignment scores and the alignments are outputted as results.

Construction of database indexes

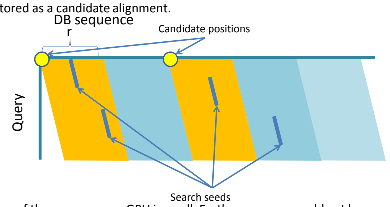
- Sequences are connected with inserting delimiters, to make a long DB sequence.
- And then, every offset of k-mer in DB sequence are added the index



K-mer	offset
ARDC	0,・・・
RDCQ	1,・・・
DCQE	2,・・・

Search for alignment candidates

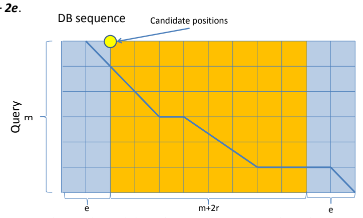
- The DNA query sequences were initially translated into protein sequences in all six open reading frames.
- The index keys of protein sequences were generated in the same way as the database indexes but with *s* characters skips.
- For checking matches, a database sequence was first divided into regions of size *r*, and the key of each query was compared with the keys of the database sequences. If more than a threshold number *t* of keys matched in a region and the right adjacent region, the position was stored as a candidate alignment.



- The size of the memory on a GPU is small. Furthermore, we could not know, a priori, the number of candidates and the size of the results to be stored when we generated a candidate for a large number of queries. Consequently, storage of the results often failed because of the shortage of GPU memory. To overcome this problem, The processing is performed as follows.
  1.Count the number of candidates at the alignment position
  2.Divide the queries into subqueries whose results could be store in the GPU.

Local alignment

- After searching for alignment positions, optimal local alignment was performed for the region around each candidate position using the Smith-Waterman algorithm, and the alignment score for each candidate position was calculated. When calculating the local alignment, we restricted the alignment target of a database sequence to a small region of size *m + 2r + 2e*.

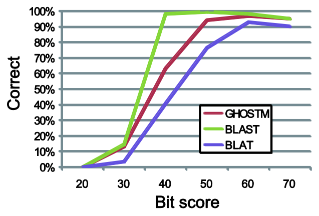


- A thread was assigned to each candidate alignment position and synchronization among threads was removed.
- All threads randomly and frequently accessed the scoring matrix. Thus, the matrix data were stored on the texture memory of a GPU because the access speed was much faster than the global memory of a GPU.

Results

☐Evaluation of search accuracy

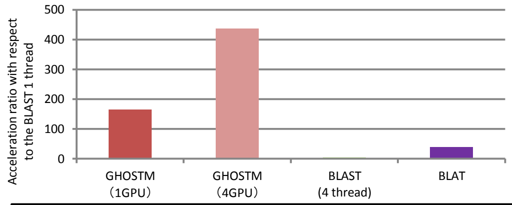
To evaluate the search accuracy, we compared the search results obtained with the Smith-Waterman local alignment method implemented in SSEARCH, and these results were assumed to be the correct answers. We analyzed the performance of a particular method in terms of the fraction of its results that corresponded to the correct answers obtained by SSEARCH.



Database : KEGG gene.pep (2.0GB)
Query : metagenomic data of soil microbes (60~75 bp, 10,000 reads)

- The search accuracy of GHOSTM was clearly higher than that of BLAT.
- The search accuracy of GHOSTM was lower than that of BLAST, especially for those hits whose scores were below 40. However, low scoring hits (e.g., < 50) are generally not used in practice because such hits can occur by chance. With the exception of the low score hits, GHOSTM successfully identified more than 90% of the hits identified by SSEARCH. This suggests that GHOSTM is sufficiently accurate for general usage.

☐Evaluation of computing time



Database : KEGG gene.pep (2.0GB)
Query : metagenomic data of soil microbes (60~75 bp, 100,000 reads)

- The GHOSTM with 1 GPU achieved a calculation speed 165.1 times faster than BLAST.
- The GHOSTM with 1 GPU achieved a calculation speed 4.1 times faster than BLAT .
- The GHOSTM program with 4 GPUs achieved a calculation speed 437.6 times faster than BLAST.

References

1.Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ, Basic local alignment search tool, Journal of molecular biology, 215:403-10, 1990
2.Kent WJ, "BLAT---The BLAST-Like Alignment Tool", *Genome Research*, 12(4):656-664, 2002
3.S. Suzuki, T. Ishida, Y. Akiyama," Fast short DNA sequence mapping on GPUs", *IPSJ-SIG Technical Report*, 2010-BIO-21(30), 1-6, 2010.

Acknowledgment

This research was supported in part by HPCI STRATEGIC PROGRAM in Computational Life Science and Applications in Drug Discovery and Medical Development by MEXT of Japan, Cancer Research Development Funding by the National Cancer Center of Japan and the CUDA COE Program by NVIDIA.