GPU-Based Acceleration for Protein Sequence Alignment Using BLAST

Eric Arezza
School of Computer Science
Carleton University, Ottawa, Canada
ericarezza@cmail.Carleton.ca

Outline

1. Background of Sequence Alignment
   - What is it and why is it important?
   - What tools and problems exist?
2. Introduction to BLAST
   - What is BLAST and the need for acceleration?
3. BLAST Algorithm Overview
   - Four steps of BLAST
   - Using GPUs for speedup
4. GPU Implementations of BLAST
   - Comparisons and targeted steps for speedup
5. Independent Evaluations of Speedup
   - Description of hardware, tests, and results
6. Concluding Remarks

Sequence Alignment

Background:

- Sequencing technology provides new genomic and proteomic information from many organisms

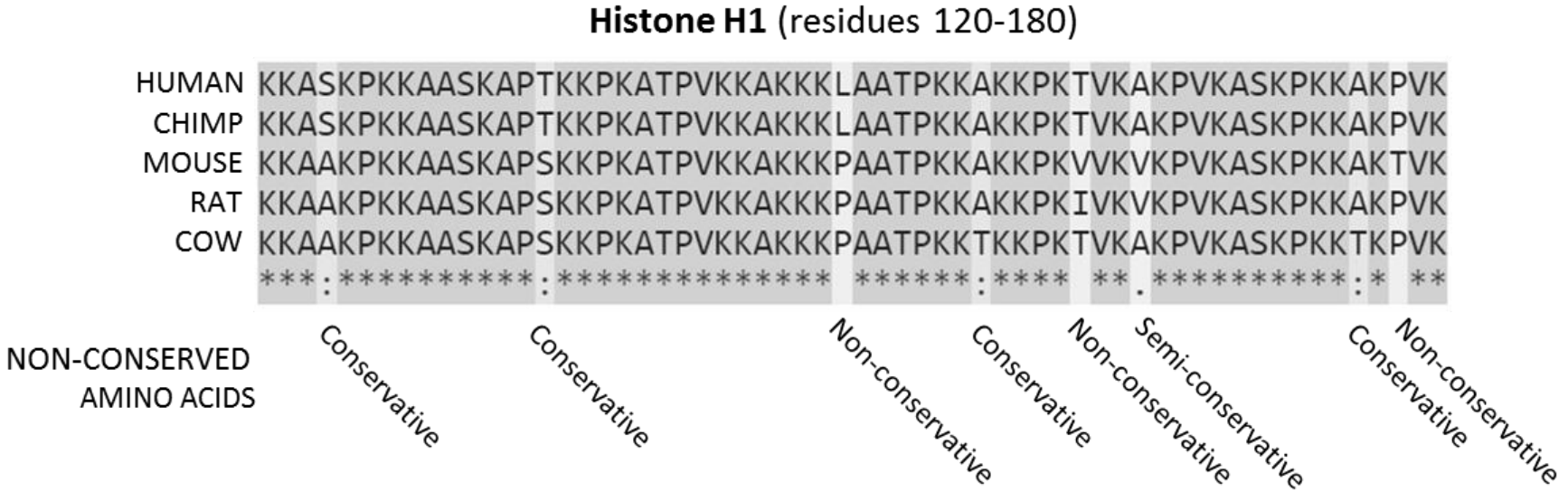
DNA (4 nucleotides): ACGT

Proteins (~20 amino acids): ARNDQ…WYV

- Functional expression of genes
- Foundation for cellular processes

Sequence Alignment

- Subsequences of matching nucleotides/amino acids between two or more sequences



Why?

- Finds if similar sequences already in a database
- Identifies functionally similar regions of proteins
- Helps understand evolutionary relatedness between organisms/cellular components

Sequence Alignment

- Many tools to perform alignment
- Pairwise/multiple alignments
- Global vs. local alignments
- DNA/RNA vs. proteins
- Some faster than others
- Sensitivities of alignments
- Different inputs/outputs

BLAST

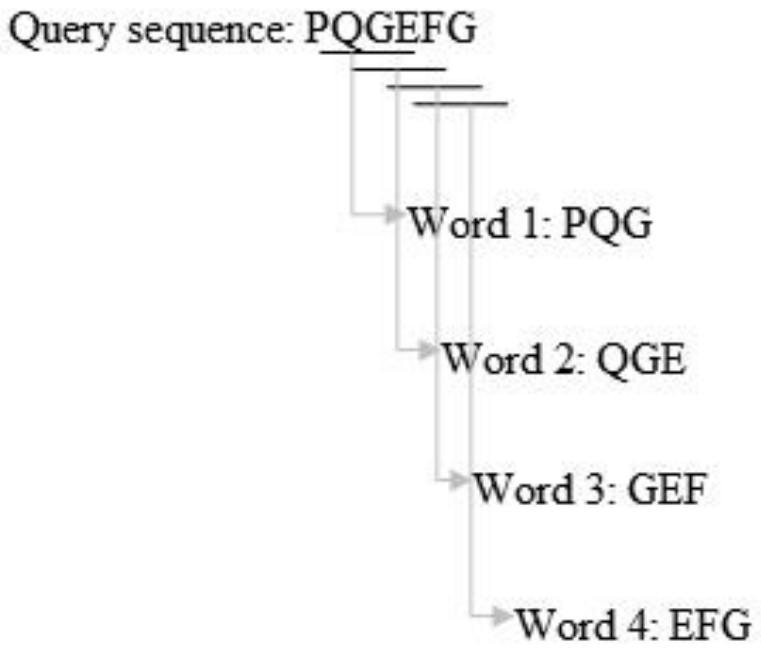
Background:

- Basic Local Alignment Search Tool
- Heuristic approach
- Developed in 1990, improved in 1997
- NCBI standard tool for sequence alignment
- Web interface or command-line execution
- Pairwise alignment against database of sequences

BLAST Algorithm

Step 1: Hit Detection/Seeding

- k-mer words + neighbor words scoring

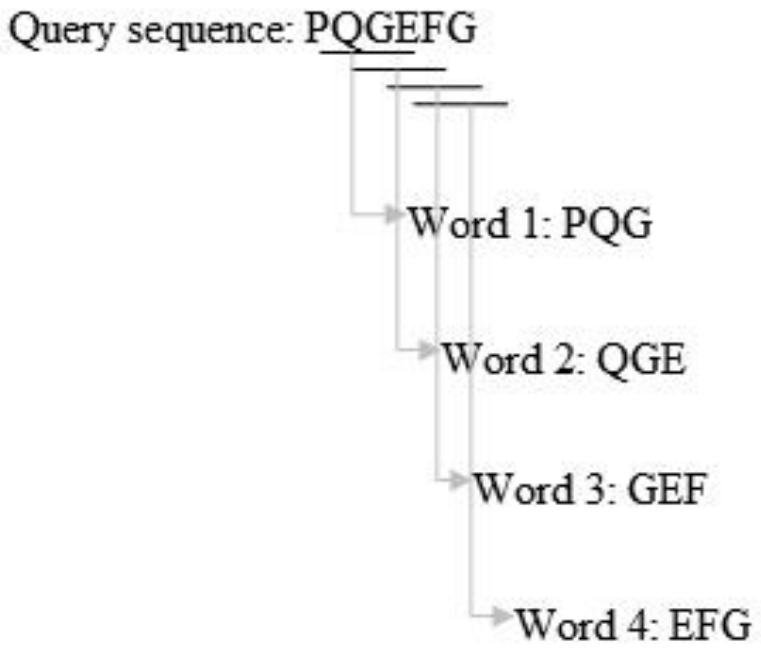


BLOSUM62 Scoring Matrix

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	−1	5																		
Asn	−2	0	6																	
Asp	−2	−2	1	6																
Cys	0	−3	−3	−3	9															
Gln	−1	1	0	0	−3	5														
Glu	−1	0	0	2	−4	2	5													
Gly	0	−2	0	−1	−3	−2	−2	6												
His	−2	0	1	−1	−3	0	0	−2	8											
Ile	−1	−3	−3	−3	−1	−3	−3	−4	−3	4										
Leu	−1	−2	−3	−4	−1	−2	−3	−4	−3	2	4									
Lys	−1	2	0	−1	−3	1	1	−2	−1	−3	−2	5								
Met	−1	−1	−2	−3	−1	0	−2	−3	−2	1	2	−1	5							
Phe	−2	−3	−3	−3	−2	−3	−3	−3	−1	0	0	−3	0	6						
Pro	−1	−2	−2	−1	−3	−1	−1	−2	−2	−3	−3	−1	−2	−4	7					
Ser	1	−1	1	0	−1	0	0	0	−1	−2	−2	0	−1	−2	−1	4				
Thr	0	−1	0	−1	−1	−1	−1	−2	−2	−1	−1	−1	−1	−2	−1	1	5			
Trp	−3	−3	−4	−4	−2	−2	−3	−2	−2	−3	−2	−3	−1	1	−4	−3	−2	11		
Tyr	−2	−2	−2	−3	−2	−1	−2	−3	2	−1	−1	−2	−1	3	−3	−2	−2	2	7	
Val	0	−3	−3	−3	−1	−2	−2	−3	−3	3	1	−2	1	−1	−2	−2	0	−3	−1	4

Step 1: Hit Detection/Seeding

- 3-mer words for PQGEFG = l – k + 1 = 6 – 3 + 1 = 4 words



BLOSUM62 Scoring Matrix

	Ala	Arg	Asn	Asp	Cys	Gln	Glu	Gly	His	Ile	Leu	Lys	Met	Phe	Pro	Ser	Thr	Trp	Tyr	Val
Ala	4																			
Arg	−1	5																		
Asn	−2	0	6																	
Asp	−2	−2	1	6																
Cys	0	−3	−3	−3	9															
Gln	−1	1	0	0	−3	5														
Glu	−1	0	0	2	−4	2	5													
Gly	0	−2	0	−1	−3	−2	−2	6												
His	−2	0	1	−1	−3	0	0	−2	8											
Ile	−1	−3	−3	−3	−1	−3	−3	−4	−3	4										
Leu	−1	−2	−3	−4	−1	−2	−3	−4	−3	2	4									
Lys	−1	2	0	−1	−3	1	1	−2	−1	−3	−2	5								
Met	−1	−1	−2	−3	−1	0	−2	−3	−2	1	2	−1	5							
Phe	−2	−3	−3	−3	−2	−3	−3	−3	−1	0	0	−3	0	6						
Pro	−1	−2	−2	−1	−3	−1	−1	−2	−2	−3	−3	−1	−2	−4	7					
Ser	1	−1	1	0	−1	0	0	0	−1	−2	−2	0	−1	−2	−1	4				
Thr	0	−1	0	−1	−1	−1	−1	−2	−2	−1	−1	−1	−1	−2	−1	1	5			
Trp	−3	−3	−4	−4	−2	−2	−3	−2	−2	−3	−2	−3	−1	1	−4	−3	−2	11		
Tyr	−2	−2	−2	−3	−2	−1	−2	−3	2	−1	−1	−2	−1	3	−3	−2	−2	2	7	
Val	0	−3	−3	−3	−1	−2	−2	−3	−3	3	1	−2	1	−1	−2	−2	0	−3	−1	4

BLAST Algorithm

Step 1: Hit Detection/Seeding

- Using first word PQG, generate neighbor words within given threshold score

PQG
PQA 7+5+0=12
LQA -3+5+4=6
LFG 4-3+0=1
…

20 x 20 x 20 = 8000 words for each k-mer word in sequence

e.g. threshold = 10, then PQG, PQA,are added to seed list for extension

- This reduces total number of words for seeding

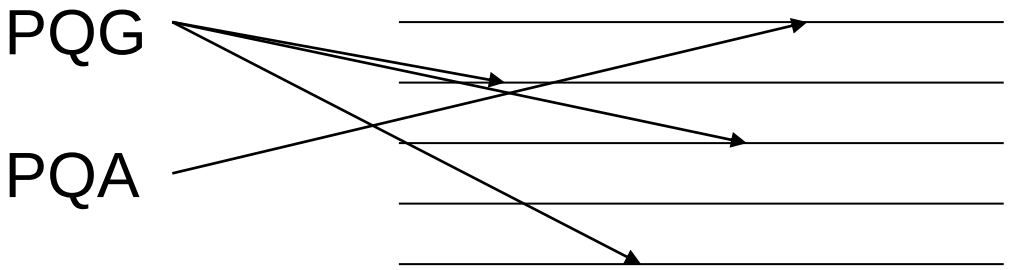
BLAST Algorithm

Step 1: Hit Detection/Seeding

- Place words into a table with index of word locations
- Organize into a binary search tree for lookup
- Repeat for all words in query sequence

Then:

- Scan database sequences for exact matches (hits) to words and record as array of tuples (queryPosition, subjectPosition)



Step 2: Ungapped Seed Extension

- High-scoring Segment Pairs

Query sequence: R P P Q G L F

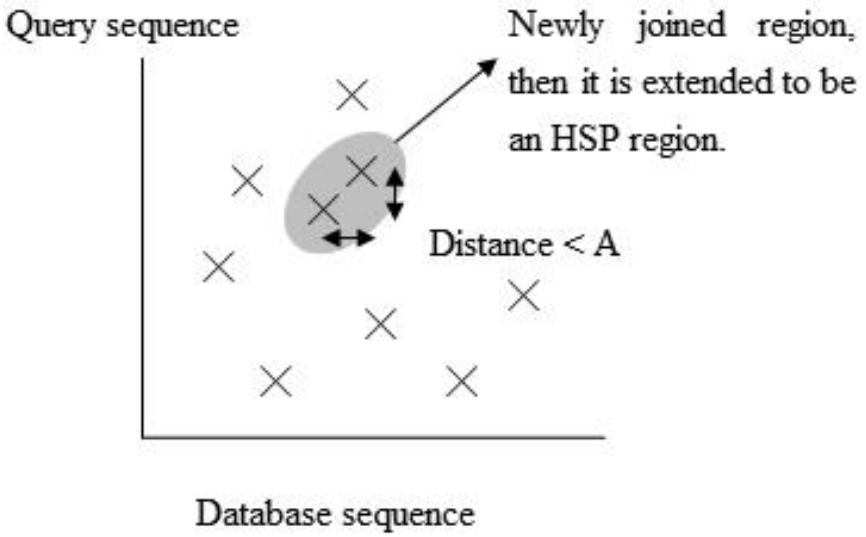
Database sequence: D P **P E G** V V

Exact match is scanned.

Score: -2 **7 7 2 6 1** -1

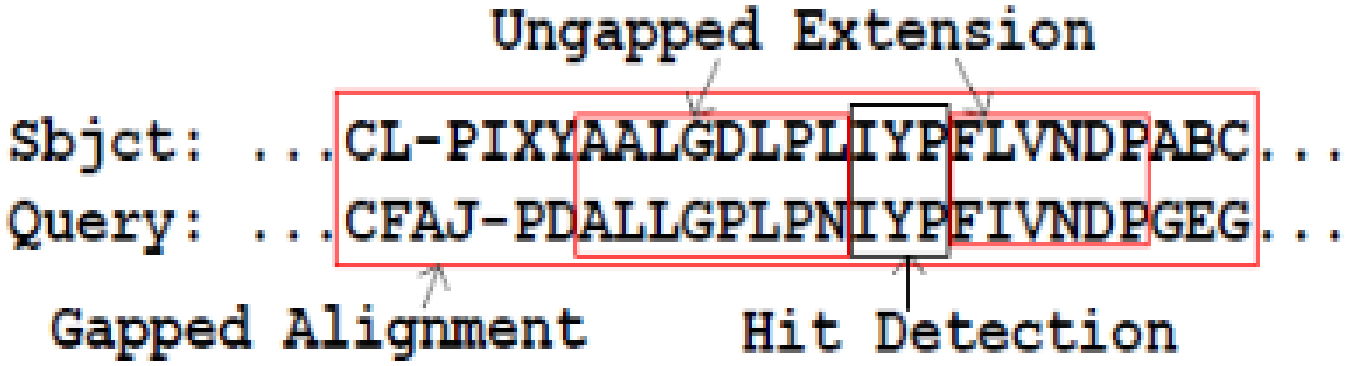
HSP

Optimal accumulated score = 7+7+2+6+1 = 23



Step 3: Gapped Seed Extension

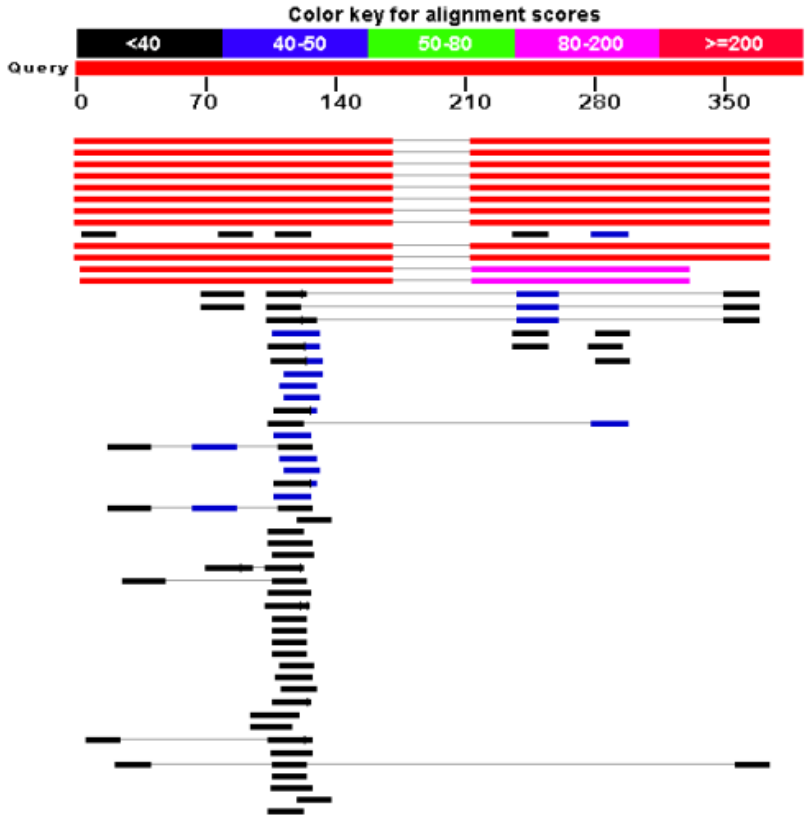
- Use ungapped HSPs to further extend including gaps
- Accounts for insertion/deletions in sequences
- Concludes a local alignment between query and subject



BLAST Algorithm

Step 4: Traceback and Output

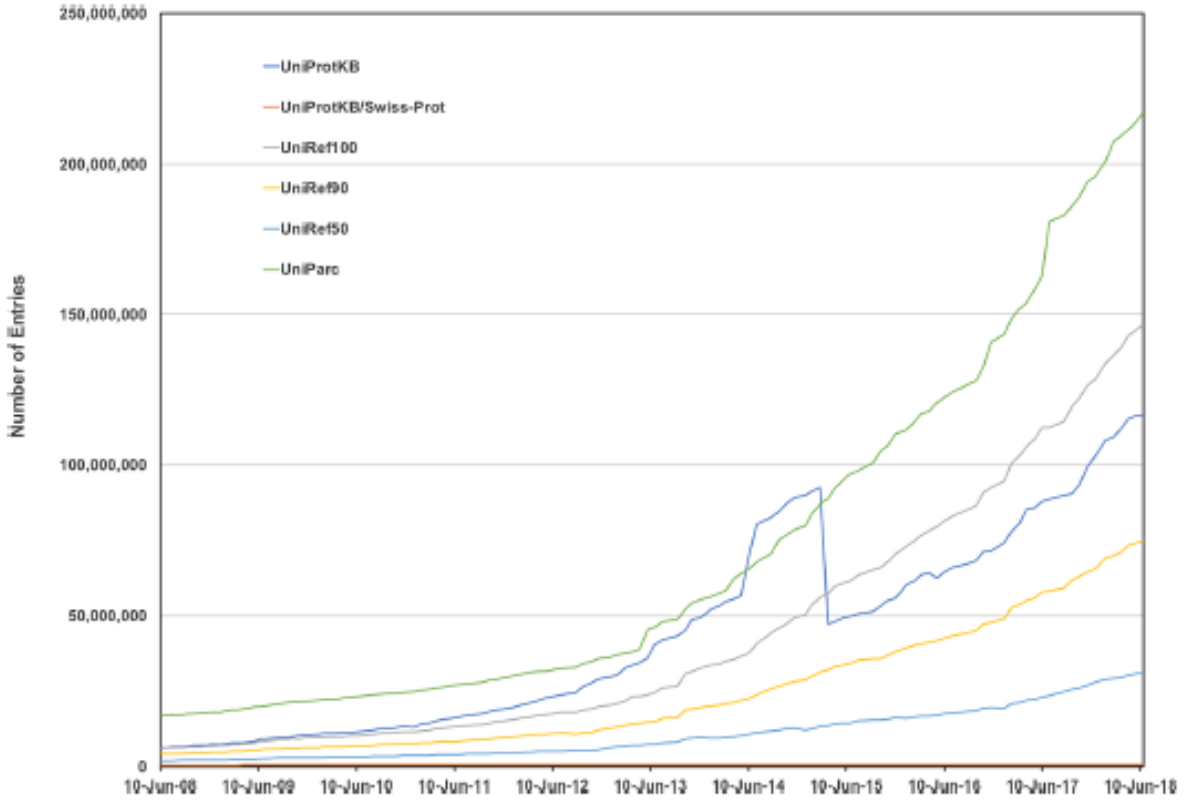
- Re-score and report all matches above given thresholds



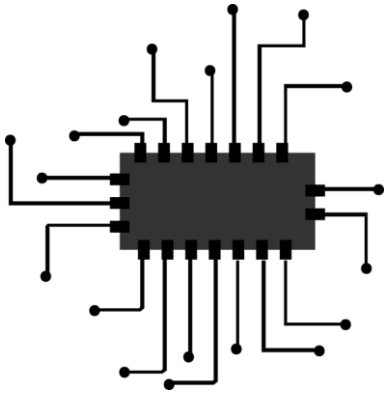

```
Database: SwissProt
           474,714 sequences; 179,658,219 total letters



Query= sp|Q9VCA8|ANKHM_DROME Ankyrin repeat and KH domain-containing
protein mask OS=Drosophila melanogaster OX=7227 GN=mask PE=1 SV=2

Length=4001
                                                                    Score     E
Sequences producing significant alignments:                        (Bits)  Value

Q9VCA8.2 RecName: Full=Ankyrin repeat and KH domain-containing pr...  8192    0.0
Q8IWZ3.1 RecName: Full=Ankyrin repeat and KH domain-containing pr...  814     0.0
O75179.3 RecName: Full=Ankyrin repeat domain-containing protein 1...  806     0.0
Q99NH0.2 RecName: Full=Ankyrin repeat domain-containing protein 1...  805     0.0
Q21920.3 RecName: Full=Ankyrin repeat and KH domain-containing pr...  516     6e-146
Q60J38.3 RecName: Full=Ankyrin repeat and KH domain-containing pr...  509     7e-144
Q8C8R3.2 RecName: Full=Ankyrin-2; Short=ANK-2; AltName: Full=Anky...  237     1e-60
```

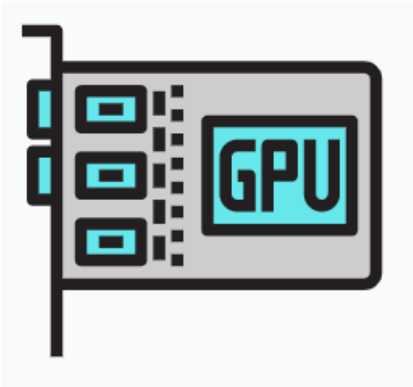
Speeding Up BLAST

- Growth of protein databases and sequencing data
- e.g. nr db ~82G, blastp on protein (length=4000) against swissprot database (size 133M) takes ~1.3 minutes



Speeding Up BLAST

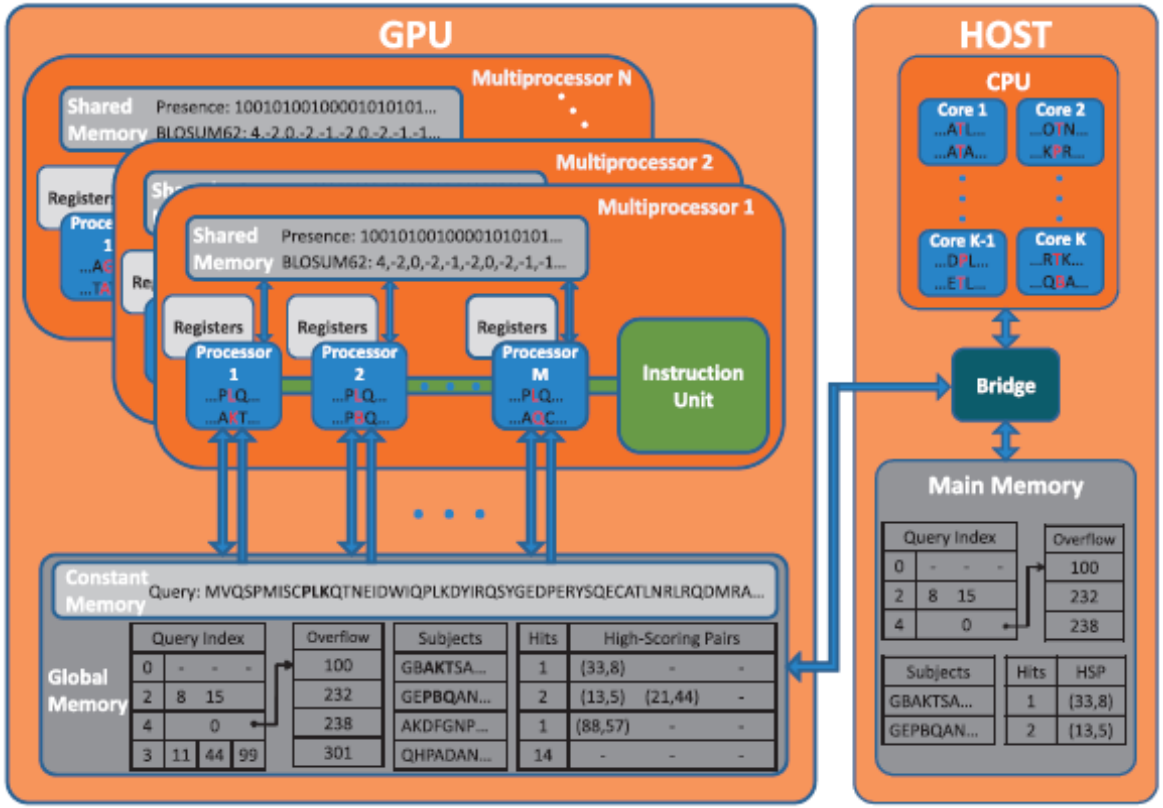




- Clusters + FPGAs
  - Hadoop & Spark
  - Many machines
  - Time-investment
  - Expensive
  - Resource accessibility
- GPUs
  - General-purpose
  - Accessible resources
  - Simpler single-machine
  - Cheaper than FPGA

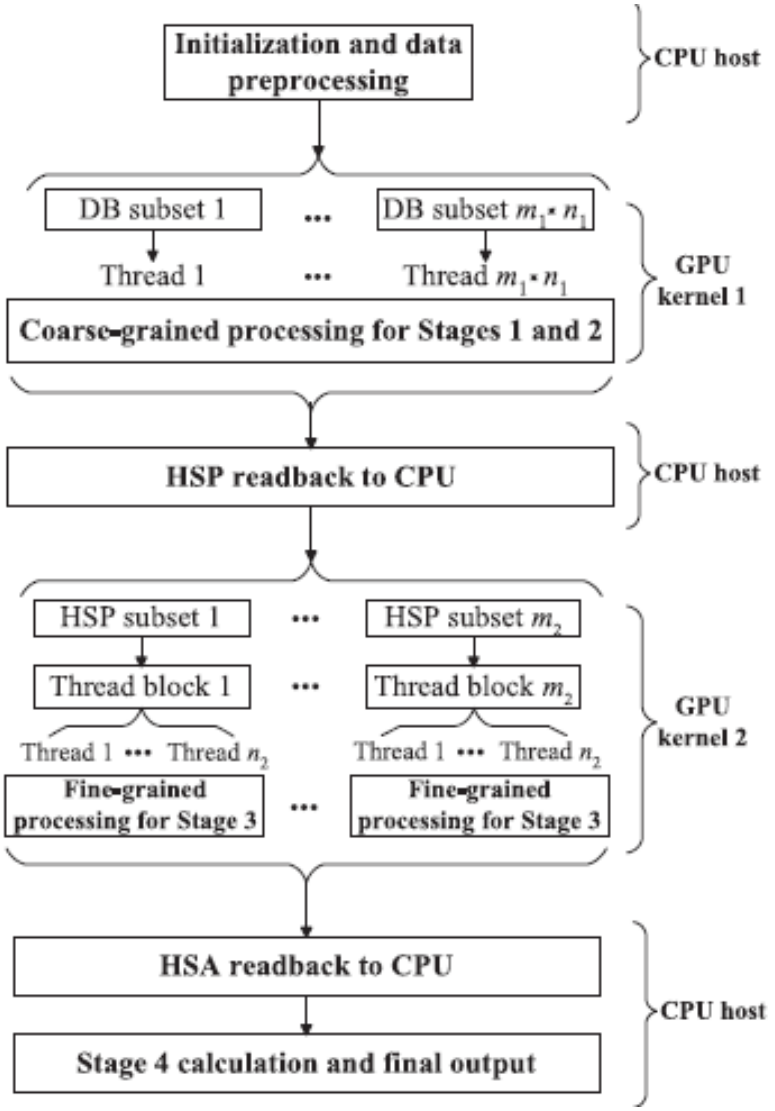
NCBI-GPU-BLASTP (2010)

- Focused on seeding & ungapped extension
- Most frequently accessed data structures put to fastest memory access locations
- Database sequences sorted, given to each GPU thread
- Presence vector helps speed up hit detection



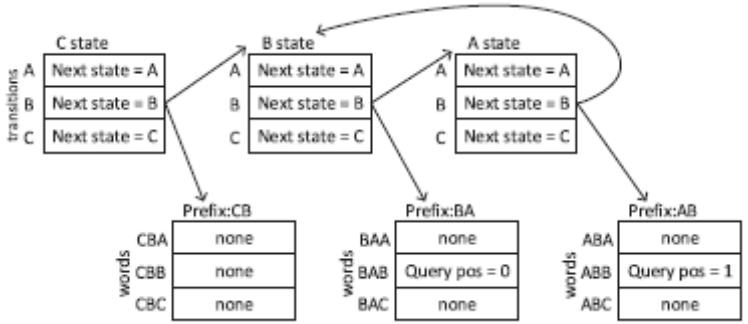
CUDA-BLASTP (2011)

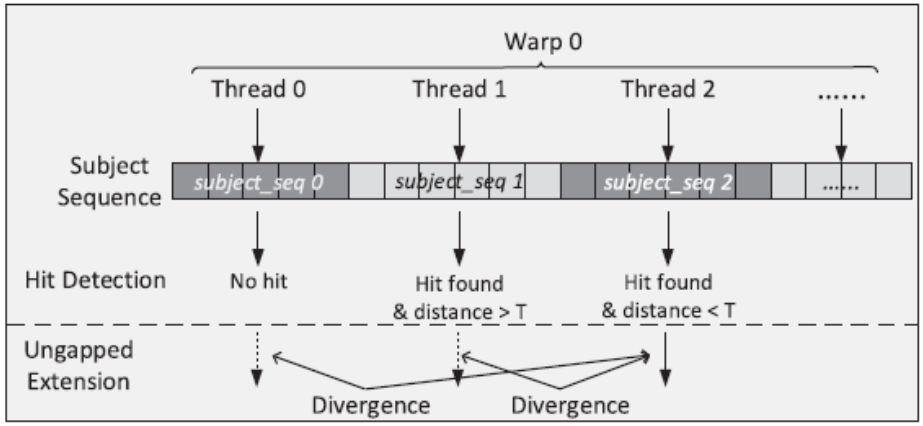
- Course-grained algorithm for seed generation and ungapped extension
  - Database sequences divided evenly into subsets over threads
- Fine-grained algorithm for gapped extension
  - HSPs distributed evenly to thread blocks

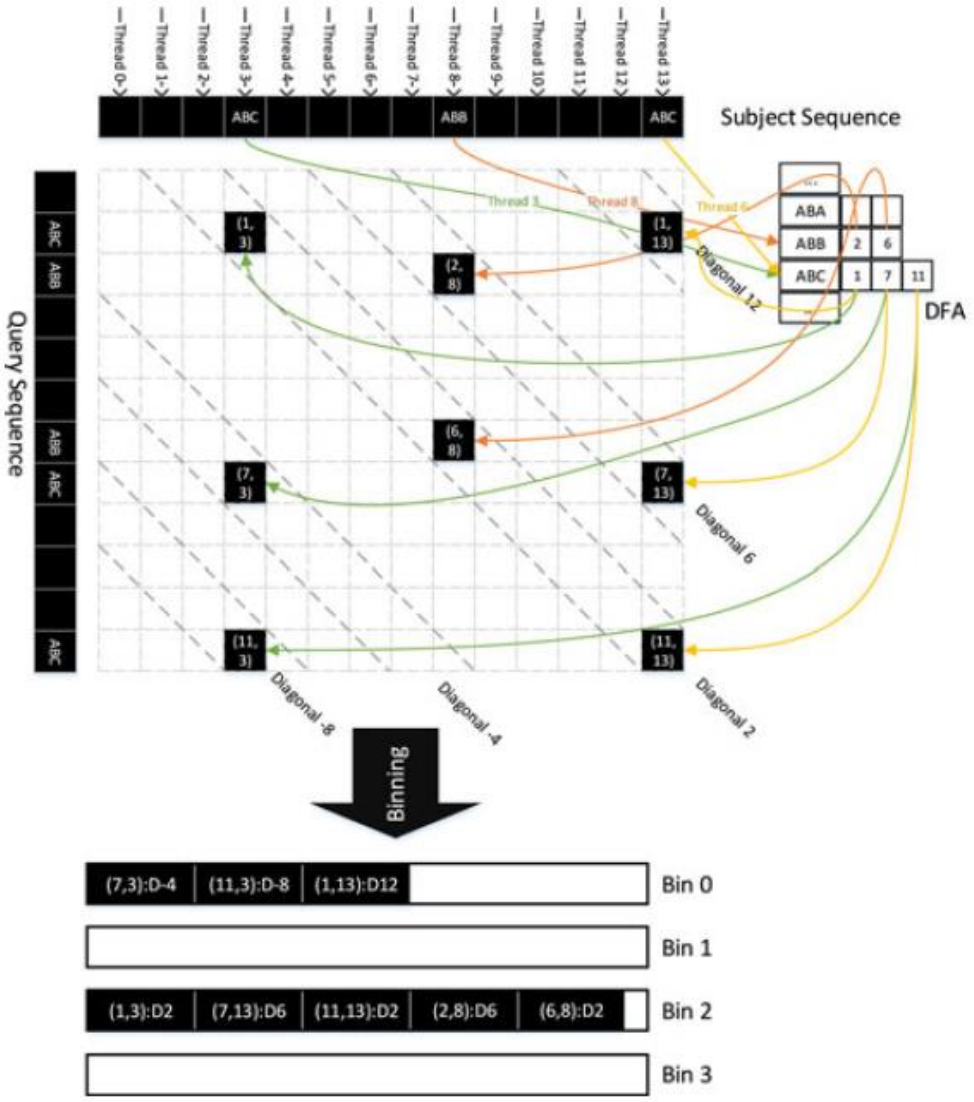


cuBLASTP (2017)

- Focused on diagonal execution of hit detection + ungapped extension
- Uses deterministic finite automation to find word matches
- Binning-sorting-filtering approach to reorder memory accesses

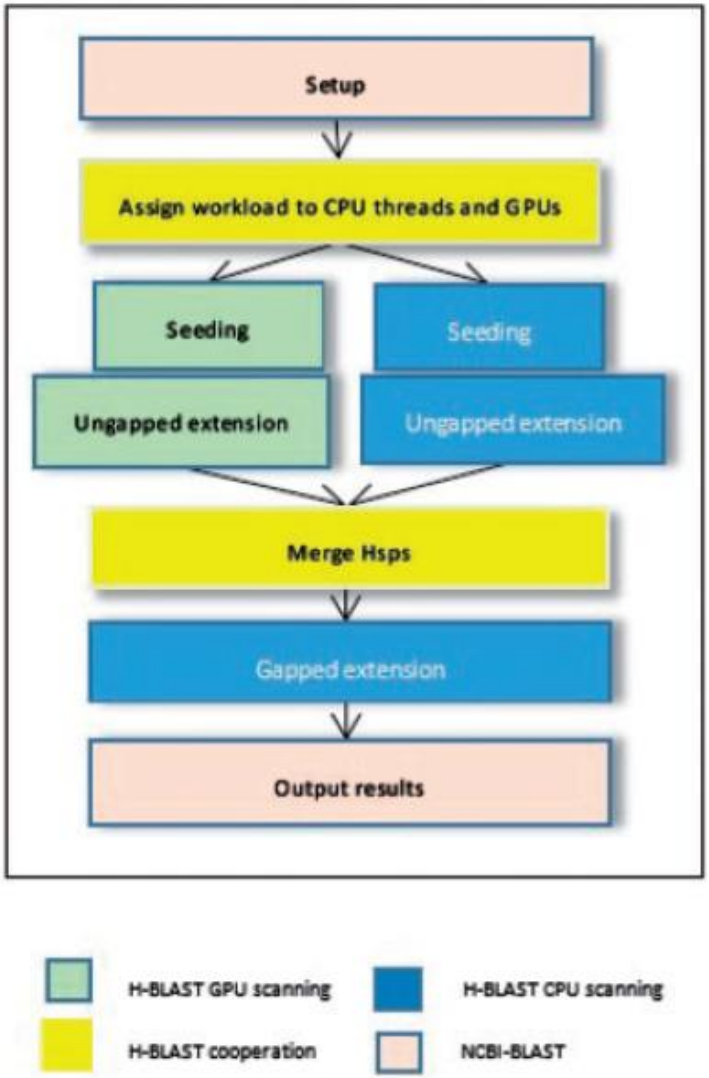






H-BLAST (2017)

- Has additional functionality for BLASTX (nucleotides->proteins)
- Seeding and ungapped extension
- Maps alignment tasks of database sequences to GPU threads
- Sort hits in queue, push to extension queues
- Extensions grouped based on subject sequence lengths to perform in batches (balances workload)



GPU Implementations of BLAST

	NCBI-GPU-BLASTP	CUDA-BLASTP	GPU-BLASTP	CuBLASTP	H-BLAST
Database (proteins size)	env_nr (6,031,291)	GenBank nr (9,230,955)	Ncbi nr (9,874,397)	env_nr (6,000,000), swissprot (300,000)	Ncbi nr (14,324,397)
Query Proteins	51 mouse (UniProt)	P14144, P42018, Q52TG9, Q52KR2, P08678	4 proteins	3 proteins	250 (swissprot), 6 groups (100 to 600)
Query Lengths/Amino Acids	Up to 4498	127, 254, 517, 1054, 2026	1000 to 4000	127, 517, 1054	100 to 5000, maximum 9000
FSA or NCBI Base Code	NCBI	NCBI	FSA	FSA	NCBI
Reported Comparisons	3-4x BLASTP	10x BLASTP	BLASTP, CUDA-BLASTP	2.5-2.8x BLASTP, FSA-BLAST, CUDA-BLASTP, GPU-BLASTP	4-10x BLASTP, GPU-BLASTP
GPU Used	Fermi C2050	GeForce GTX 280, GeForce GTX 295	Tesla, C1060, Fermi C2050	Kepler K20c	K20x, K40m
Available	http://archimedes.cheme.cmu.edu/?q=gpublast	https://sites.google.com/site/liuweiguohome/software	N/A	https://github.com/vtsynergy/cuBLASTP	https://github.com/Yeyke/H-BLAST

Independent Evaluation

Project Objective:

- Evaluate GPU methods for BLAST independently using common parameters
- Assess speedup comparisons and analyze results

Independent Evaluation of Speedup

Hardware, Setup, and Testing Parameters:

Resources:

- Carleton University SCS VMs
- GPU: Geforce RTX 2080 Super
- vCPU: 6
- RAM: 24GB
- OS: Ubuntu 20.04

Evaluation:

- Swissprot Database
- Arbitrary query proteins of varied length
- Default BLAST+ parameters

Independent Evaluation of Speedup

Results:

TBD

Conclusion

Remarks:

- BLAST has inherent opportunities for parallelization
- Distributing database sequences over threads is common
- Size of data structures limits fast-access memory optimizations
- Pending results…

References

- Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. J Mol Biol. 1990 Oct 5;215(3):403-10. doi: 10.1016/S0022-2836(05)80360-2. PMID: 2231712.
- Altschul SF, Madden TL, Schäffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 1997 Sep 1;25(17):3389-402. doi: 10.1093/nar/25.17.3389. PMID: 9254694; PMCID: PMC146917.
- Xiong, J. (2006). SEQUENCE ALIGNMENT. In Essential Bioinformatics (pp. 29-94). Cambridge: Cambridge University Press.
- The UniProt Consortium, UniProt: a worldwide hub of protein knowledge, Nucleic Acids Research, Volume 47, Issue D1, 08 January 2019, Pages D506–D515, https://doi.org/10.1093/nar/gky1049
- S. Xiao, H. Lin and W. Feng, "Accelerating Protein Sequence Search in a Heterogeneous Computing System," 2011 IEEE International Parallel & Distributed Processing Symposium, Anchorage, AK, 2011, pp. 1212-1222, doi: 10.1109/IPDPS.2011.115.
- Ling, Cheng, and Khaled Benkrid. “Design and Implementation of a CUDA-Compatible GPU-Based Core for Gapped BLAST Algorithm.” Procedia computer science 1.1 (2010): 495–504. https://doi.org/10.1016/j.procs.2010.04.053

References

- M. Said, M. Safar, M. Taher and A. Wahba, "Accelerating iterative protein sequence alignment on a heterogeneous GPU-CPU platform," 2016 International Conference on High Performance Computing & Simulation (HPCS), Innsbruck, 2016, pp. 403-410, doi: 10.1109/HPCSim.2016.7568363.
- Liu W, Schmidt B, Müller-Wittig W. CUDA-BLASTP: accelerating BLASTP on CUDA-enabled graphics hardware. IEEE/ACM Trans Comput Biol Bioinform. 2011 Nov-Dec;8(6):1678-84. doi: 10.1109/TCBB.2011.33. PMID: 21339531.
- Vouzis, P. D., & Sahinidis, N. V. (2011). GPU-BLAST: using graphics processors to accelerate protein sequence alignment. Bioinformatics (Oxford, England), 27(2), 182–188. https://doi.org/10.1093/bioinformatics/btq644
- Zhang J, Wang H, Feng WC. cuBLASTP: Fine-Grained Parallelization of Protein Sequence Search on CPU+GPU. IEEE/ACM Trans Comput Biol Bioinform. 2017 Jul-Aug;14(4):830-843. doi: 10.1109/TCBB.2015.2489662. Epub 2015 Oct 12. PMID: 26469393.
- Glasco, D. (2012). An Analysis of BLASTP Implementation on NVIDIA GPUs. [Online] Available: https://www.semanticscholar.org/paper/An-Analysis-of-BLASTP-Implementation-on-NVIDIA-GPUs-Glasco/42bb3ca76542c08566547de2d828b2e3e61af4f3

References

- Rani, S., Gupta, O.P. CLUS_GPU-BLASTP: accelerated protein sequence alignment using GPU-enabled cluster. J Supercomput 73, 4580–4595 (2017). https://doi-org.proxy.library.carleton.ca/10.1007/s11227-017-2036-4
- Weicai Ye, Ying Chen, Yongdong Zhang, Yuesheng Xu, H-BLAST: a fast protein sequence alignment toolkit on heterogeneous computers with GPUs, Bioinformatics, Volume 33, Issue 8, 15 April 2017, Pages 1130–1138, https://doi.org/10.1093/bioinformatics/btw769
- M. Cameron, H.E. Williams, and A. Cannane, ``Improved Gapped Alignment in BLAST'', IEEE/ACM Transactions on Computational Biology and Bioinformatics, 1(3), 116-129, 2004.

Questions

1. What is sequence alignment?
2. Why is BLAST a standard tool?
3. Why do we want to speed up BLAST?