

ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP 2026)

# Faster and Cheaper: Pushing the Sequence Alignment Throughput with Commercial CPUs

Zhonghai Zhang<sup>1,2</sup>, Yewen Li<sup>3</sup>, Ke Meng<sup>1</sup>, Chunming Zhang<sup>1</sup>, Guangming Tan<sup>1</sup>

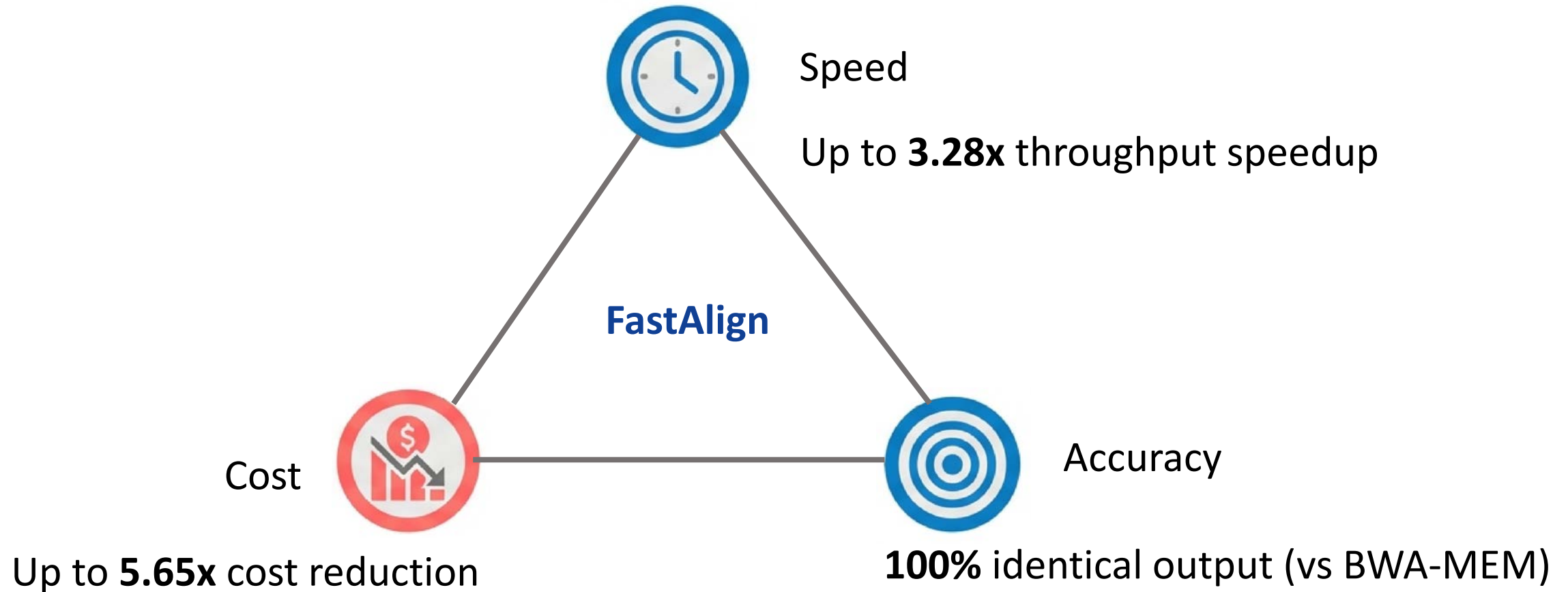
<sup>1</sup>Institute of Computing Technology, Chinese Academy of Sciences

<sup>2</sup>University of Chinese Academy of Sciences

<sup>3</sup>HongKong University of Science and Technology

2026-02-03 Sydney, Australia

# Abstract: Redefining Sequence Alignment on CPUs



FastAlign delivers a **high-performance** and **cost-efficient** sequence alignment solution on general-purpose CPU platforms

# Outline

1

**Background**

2

Motivation and Challenges

3

Design of FastAlign

4

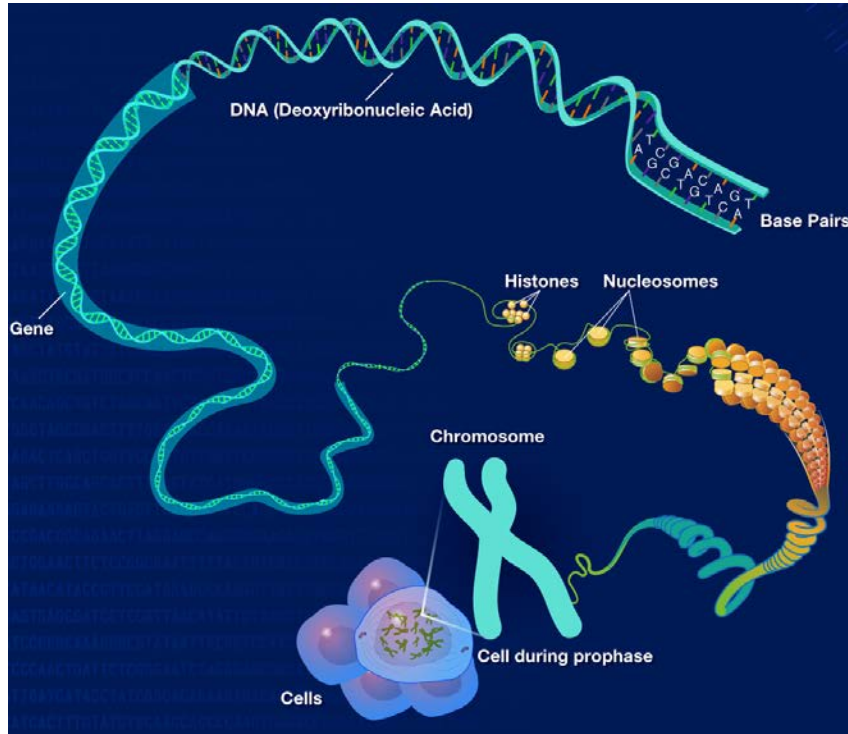
Evaluation

5

Conclusion

# Background: Genomics Influences Our World

## Genome

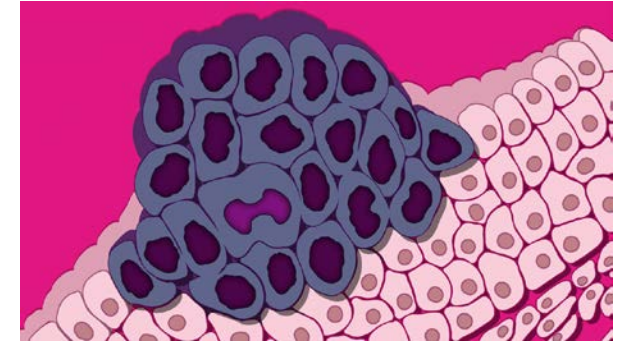


Source: NIH, NHGRI

## Human Genomic



## Cancer Therapy



## Rare Diseases



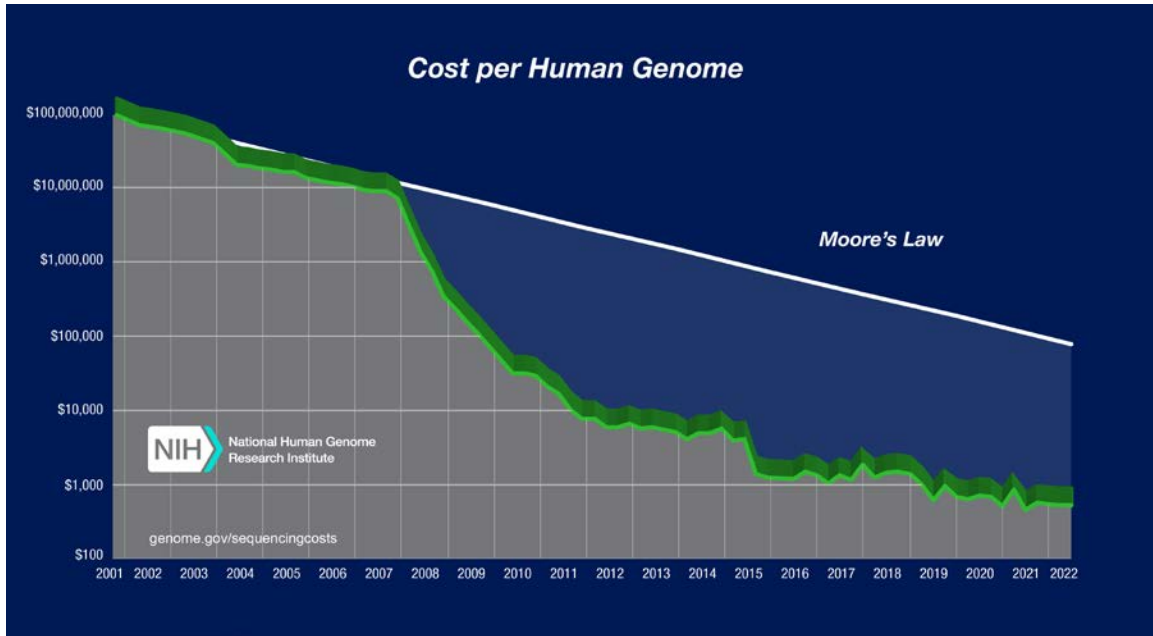
## Prenatal Diagnosis



Source: NIH, NHGRI

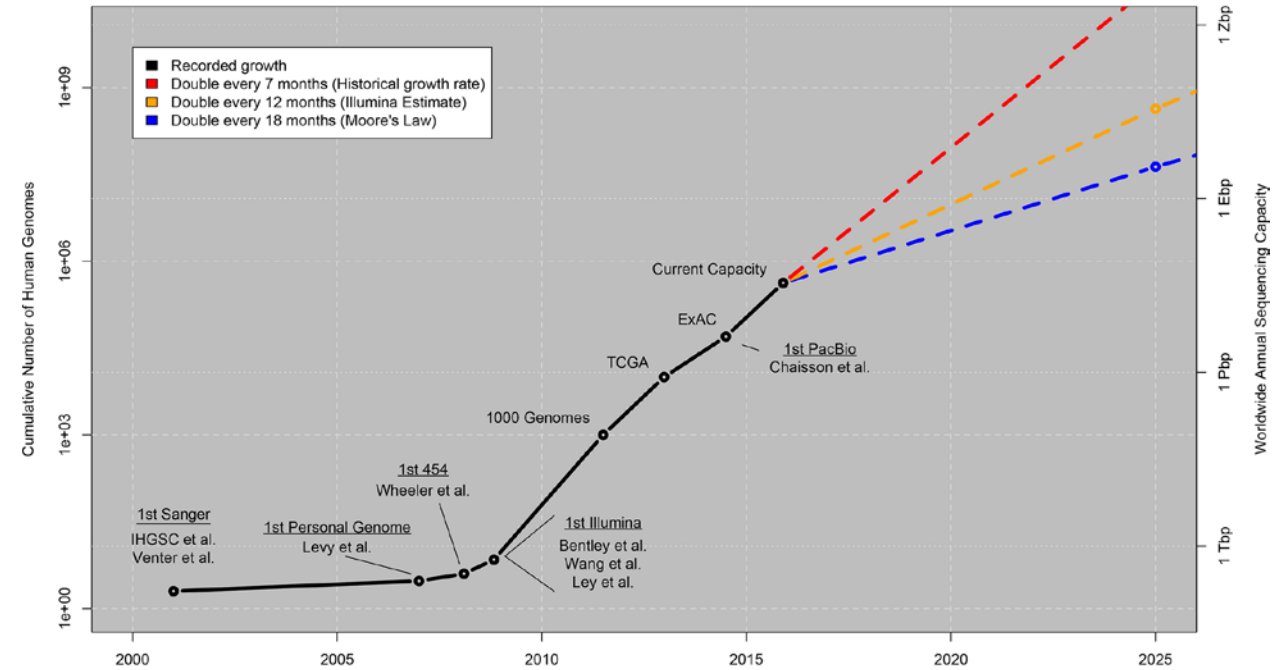
# Background: Cost Reduction and Data Growth

## Plummeting genome sequencing costs



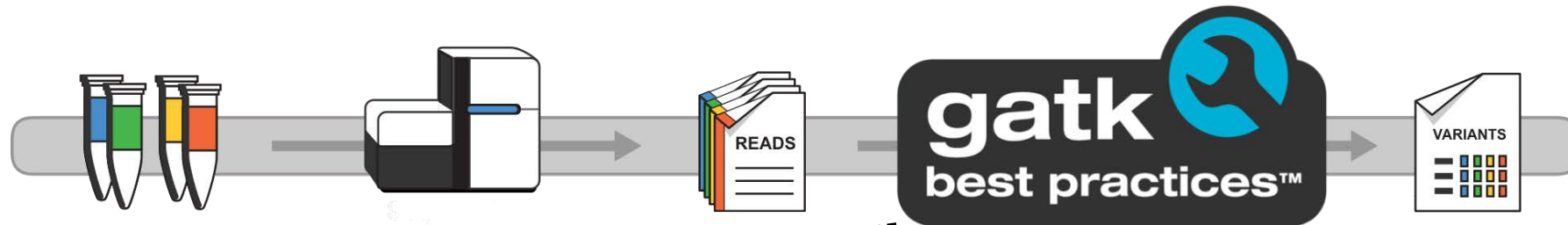
Source: NIH, NHGRI

## Exponential growth of genomic data



Source: Astronomical or Genomical, PLOS Bio 2015

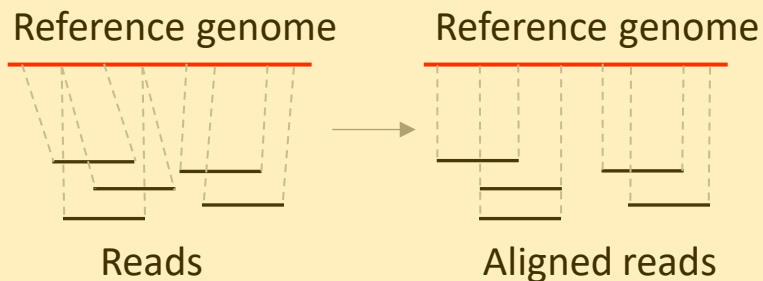
# Background: Genome Analysis Pipeline



Source: GATK, Broad Institute

## Read Alignment / Mapping

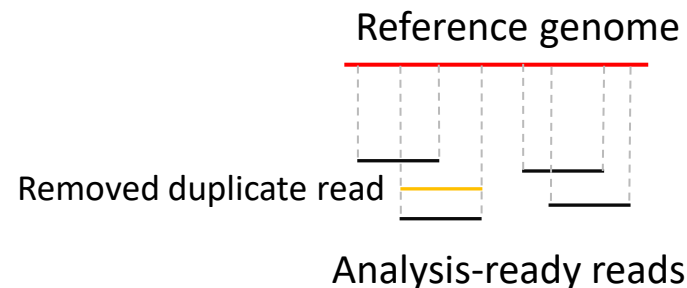
Mapping reads to their positions in the reference genome



**Major Bottleneck**

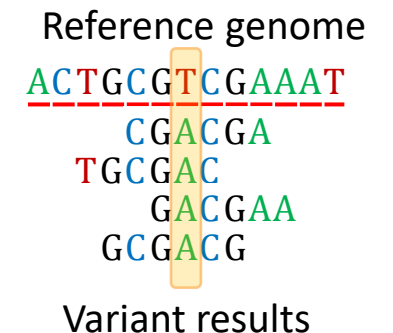
## Alignment Post Processing

Marking duplicates and recalibrating base quality scores

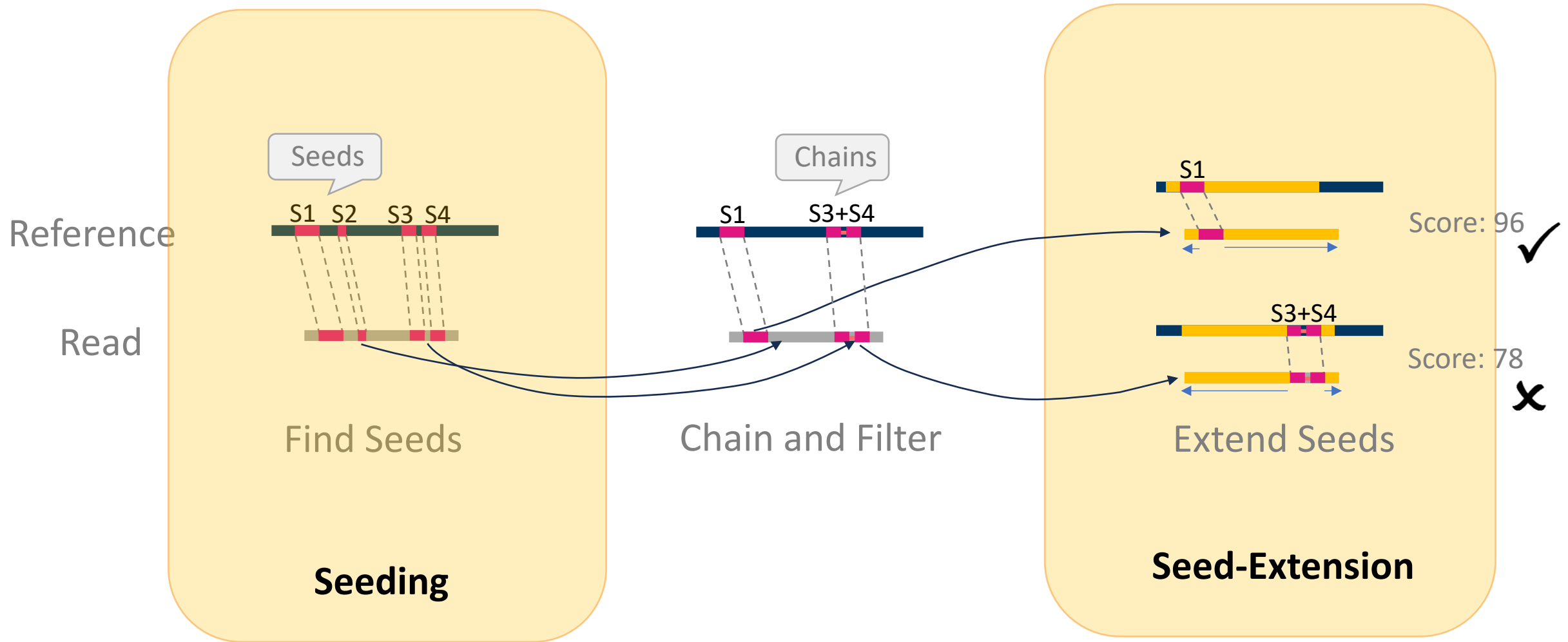


## Variant Calling

Detecting genetic variants using analysis-ready reads



# Background: Sequence Alignment (Read Alignment)



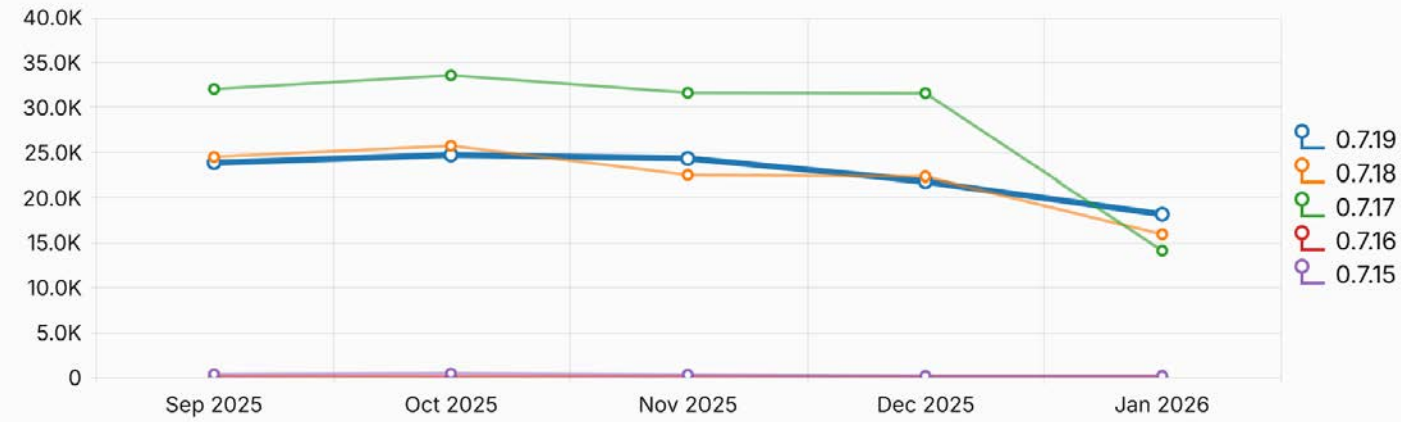
# Motivation: Cost-Efficient Solution



CPU-based BWA-MEM remains the dominant solution

Downloads (Last 6 months): 366.8K

60K downloads every month



cheap but slow



GPU/FGPA-based solutions require expensive hardware

- e.g. BWA-MEM-GPU (ICS'23), ERT (ISCA'21)

fast but expensive

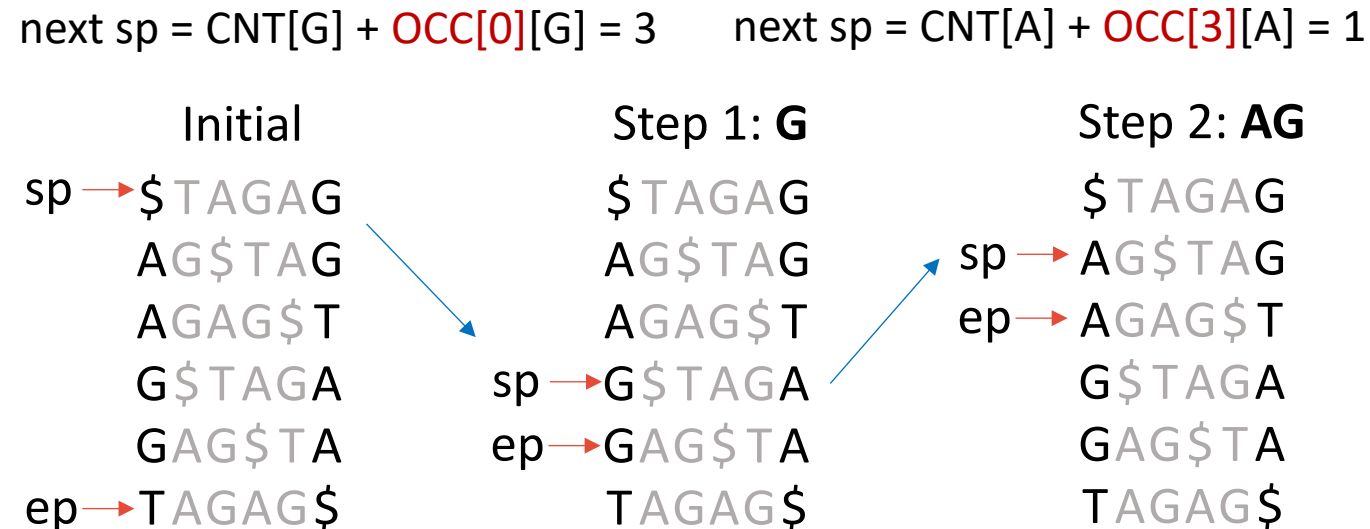
**Our Target:** cheap and fast sequence alignment solution on CPUs

# Challenges: Low Memory Efficiency in Seeding

- FM-Index processing only one base at a time
- Accessing non-contiguous memory each step
- Sampling storage, real-time rebuilding for not-stored data

Reference (R)	Suffix Array (SA)	Burrows-Wheeler Matrix (BWT)	Occurrence Table (OCC)
T A G A G \$	5	\$ T A G A G	A C G T 0 0 1 0
	3	A G \$ T A G	0 0 2 0
	1	A G A G \$ T	0 0 2 1
	4	G \$ T A G A	1 0 2 1
	2	G A G \$ T A	2 0 2 1
	0	T A G A G \$	2 0 2 1
Count (CNT)			
\$ A C G T			
0 1 1 3 5			

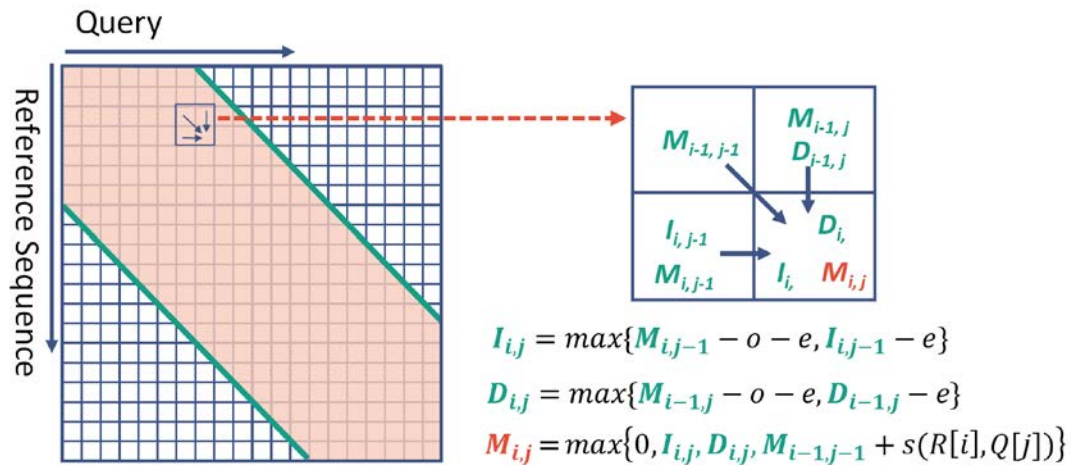
FM-Index Data Structure for Seeding



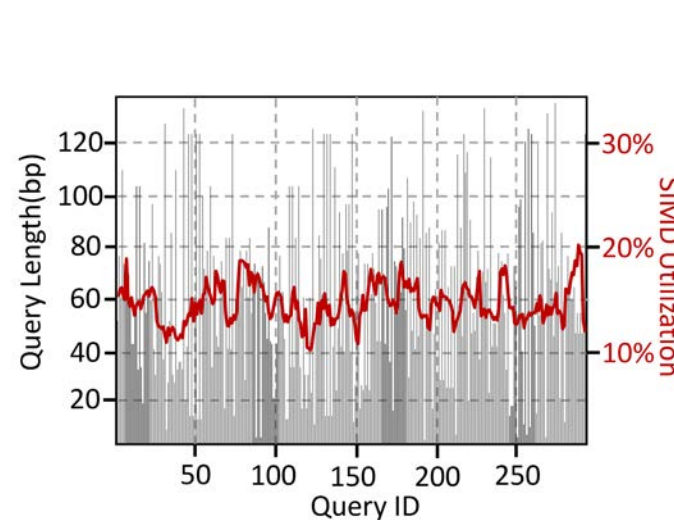
FM-Index search for query (AG)

# Challenges: Load Imbalance in Seed-Extension

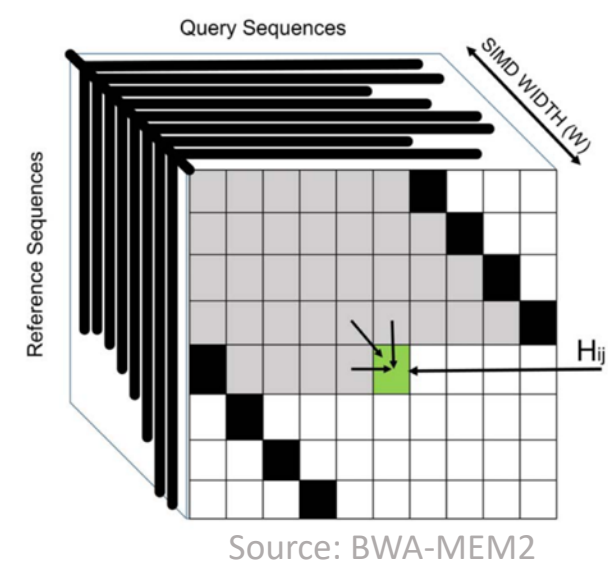
- Banded-Smith-Waterman is computing-intensive (quadratic complexity)
- Adjacent queries have highly variable lengths
- Load imbalance for inter-query SIMD parallelization



Banded-Smith-Waterman

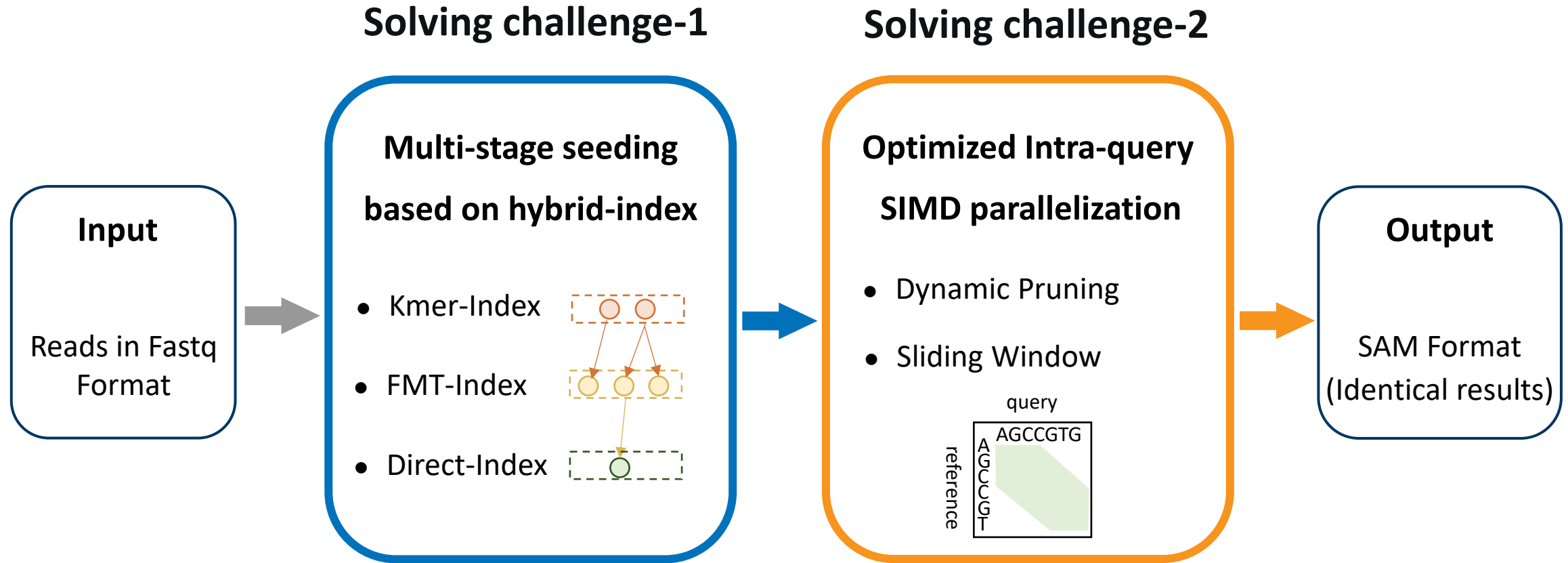


The length of adjacent queries



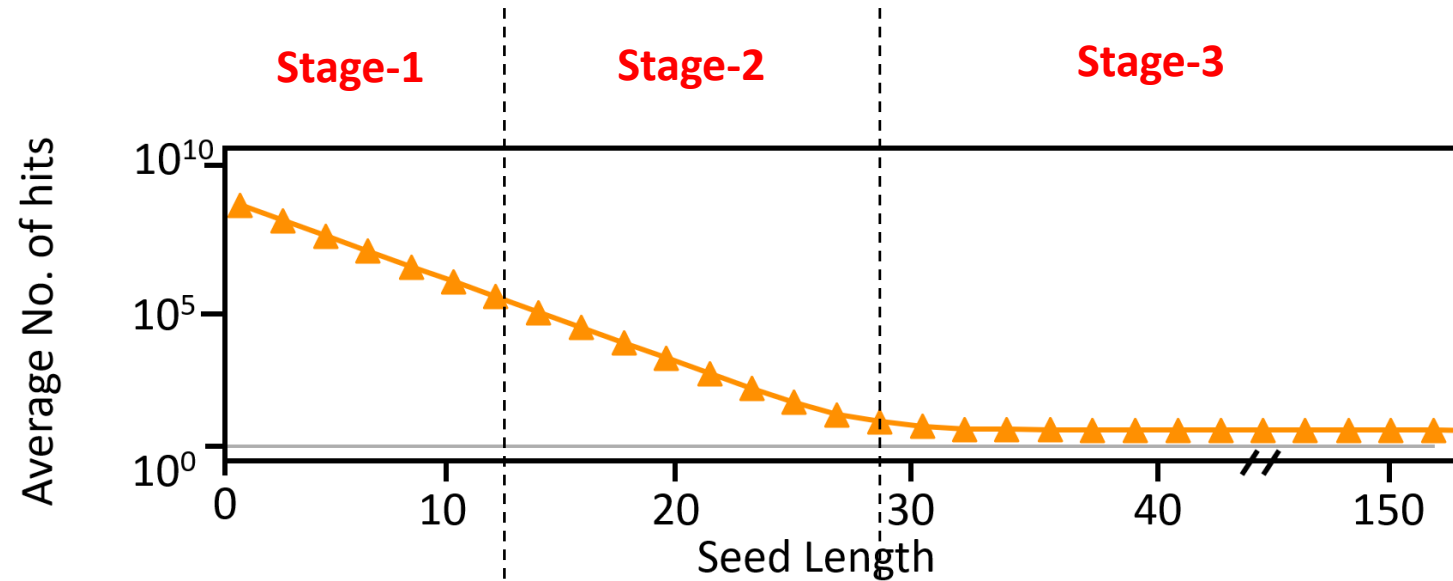
Inter-query SIMD

# FastAlign: Architecture Overview

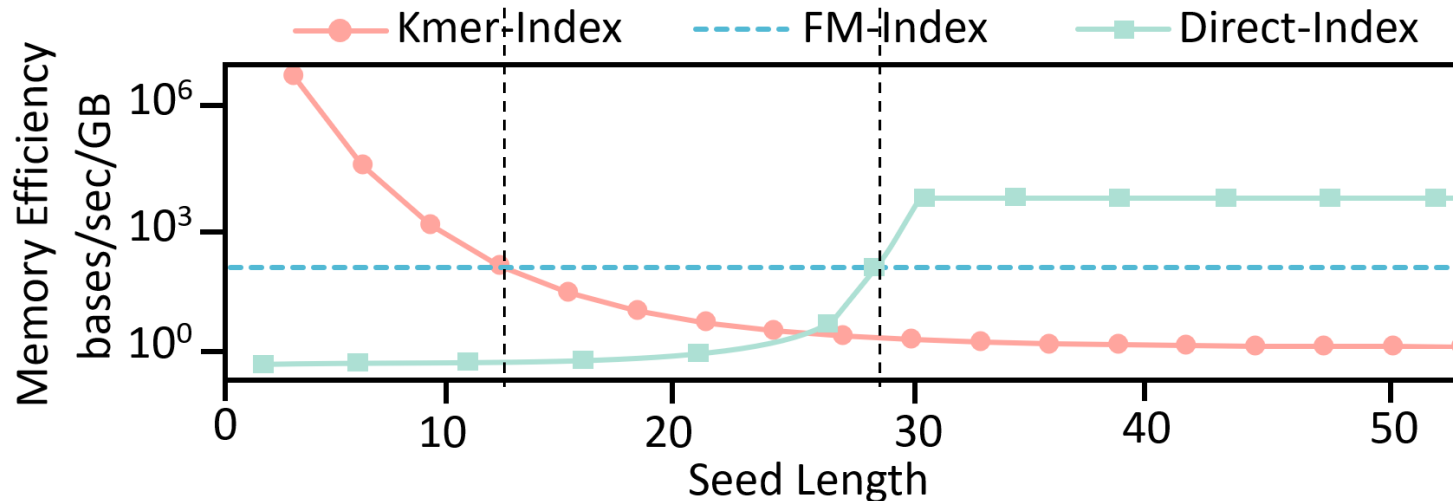


End-to-end, software-only optimization with no need for dedicated hardware accelerators.

# Opportunity: Hybrid Indexing in Seeding Phase



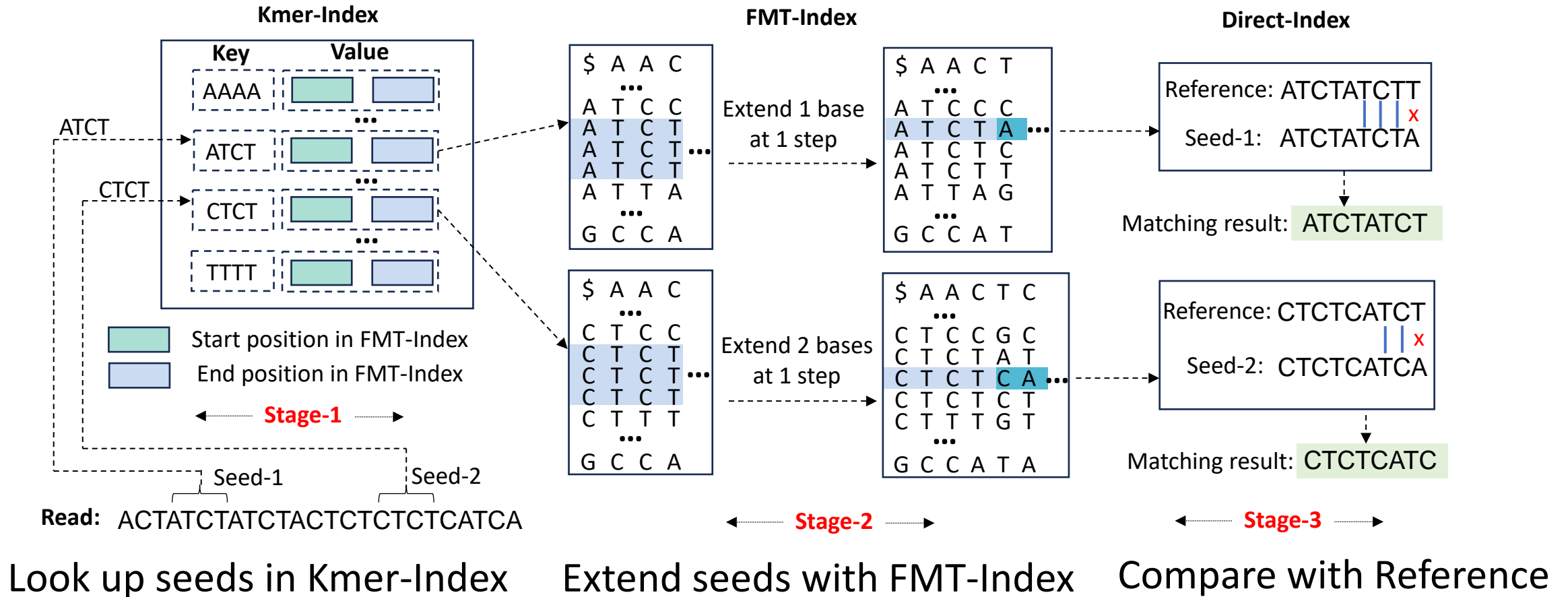
Exact-match number drops rapidly as seed length increases



Each index method has its own optimal efficiency scope.

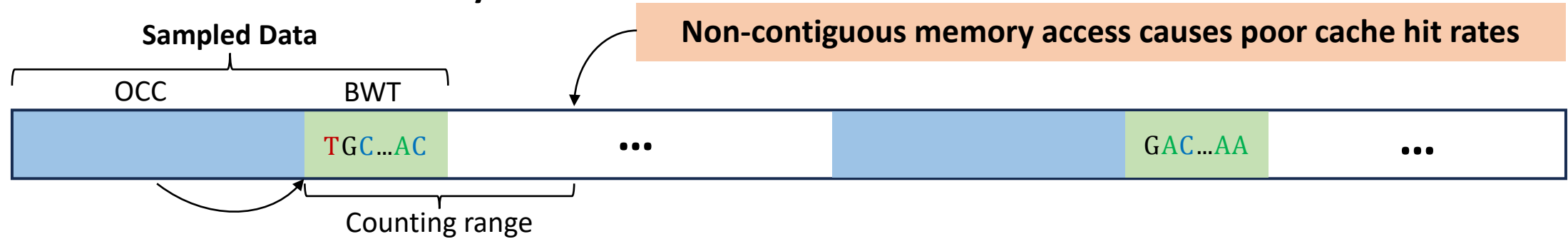
# Multi-Stage Seeding with Hybrid Index

**Reference Sequence:** AACTCCGCTATCTATCTTTTGTCTAGGTGTAGCTCTCATCTTTATTAGAAAAAGCCATATCCCTA



# FMT-Index: Improve Seeding Performance

## Standard FM-Index in memory



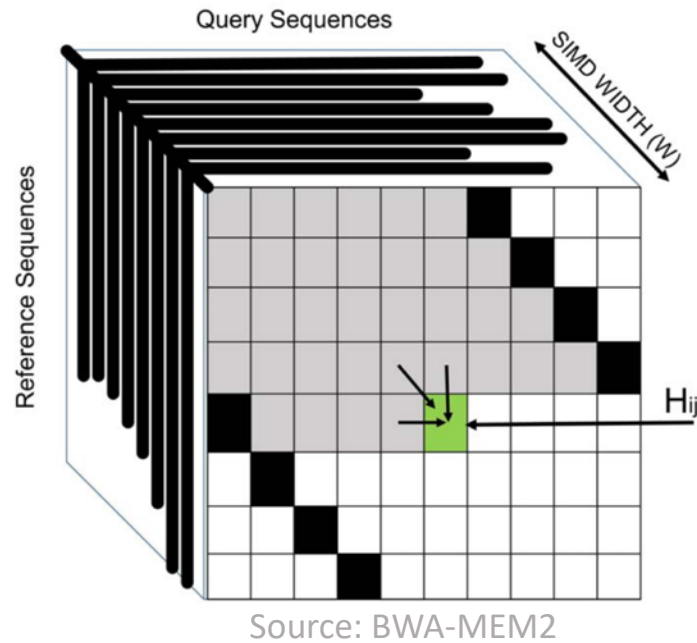
## FastAlign FMT-Index in memory



- 50% reduction in random accesses via TOCC and TBWT
- MOCC significantly reduces counting operations
- FMT-Index remains low memory usage

# Opportunity: Intra-query Parallelization in Seed-Extension

- Less load imbalance from query length variation
- Element-wise operations impact the performance



Inter-query SIMD



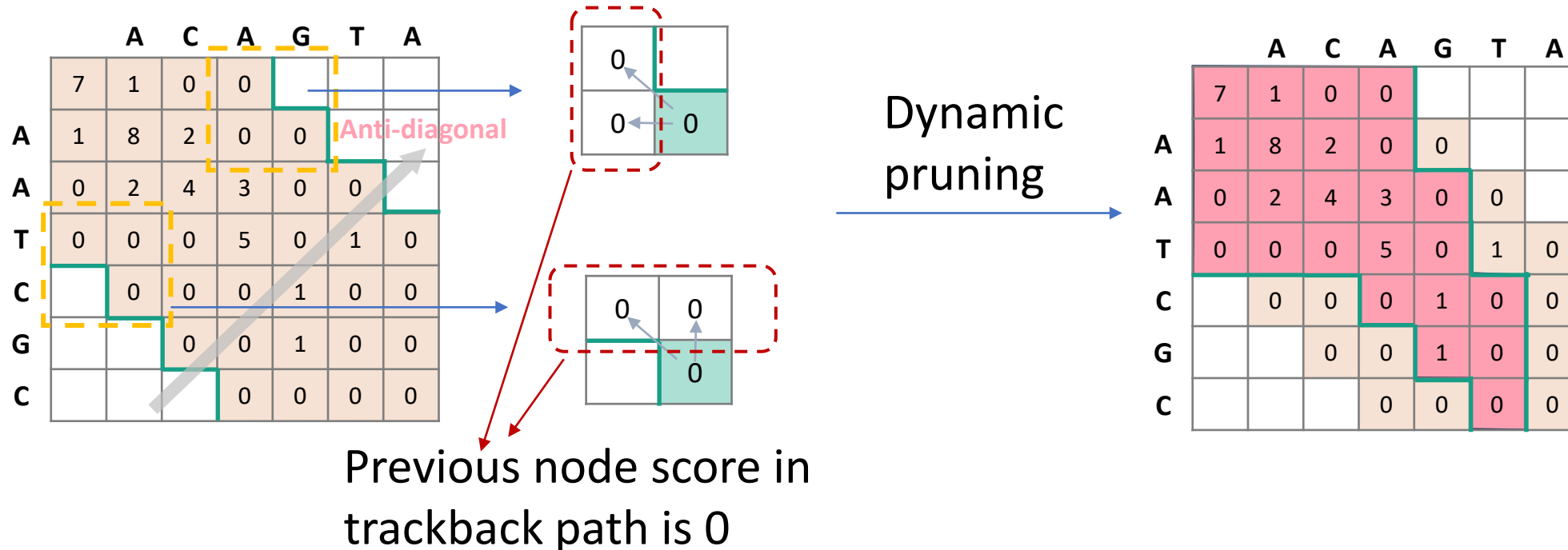
	T	G	T	T	A	C
	0	0	0	0		
G	0	0	3	1		
G	0	0	3			
T	0	3				
T	0					
G						
A						

SIMD WIDTH

Intra-query SIMD

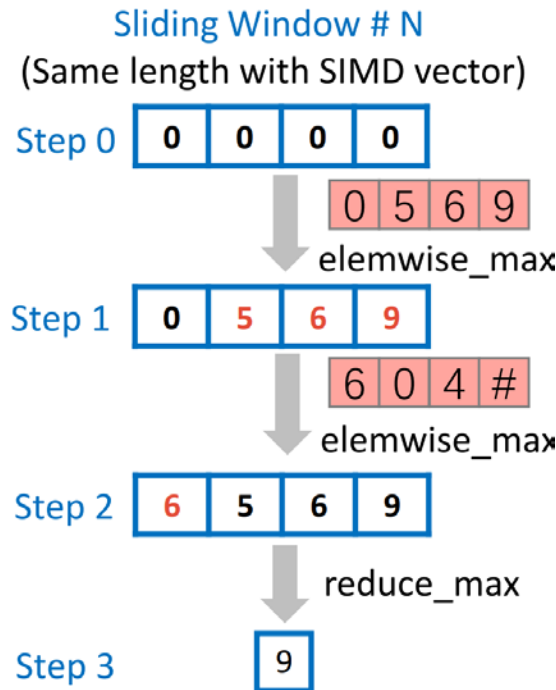
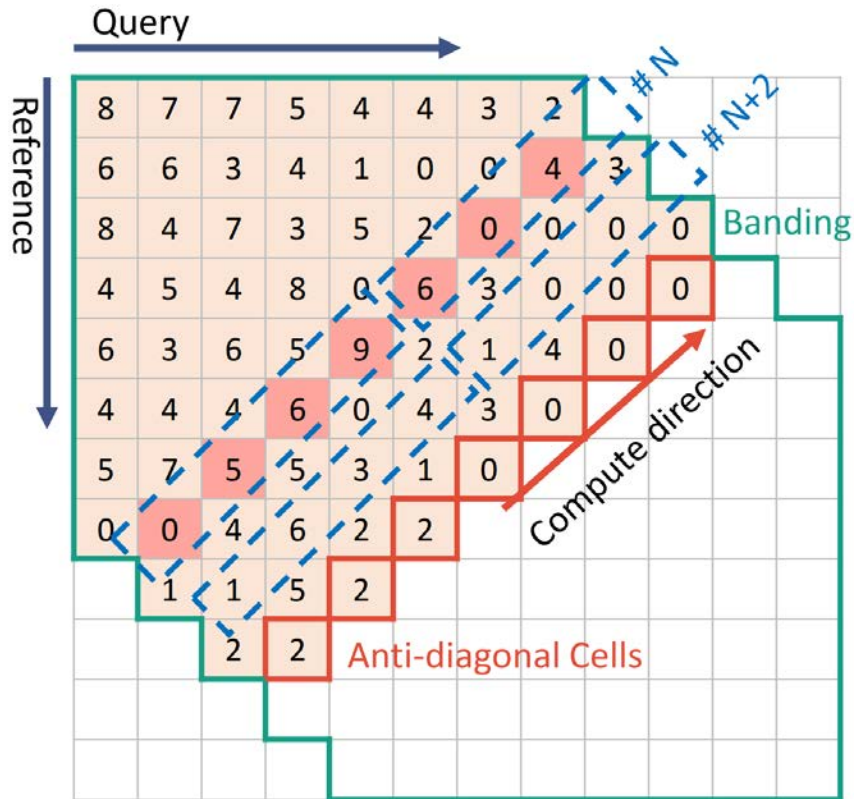
# Dynamic Pruning: Reduce Computation Scope

- Observation: Scores in the traceback path must be non-zero in BWA-MEM extension phase
- Optimization: Reduce the computation scope along the anti-diagonal after each round.



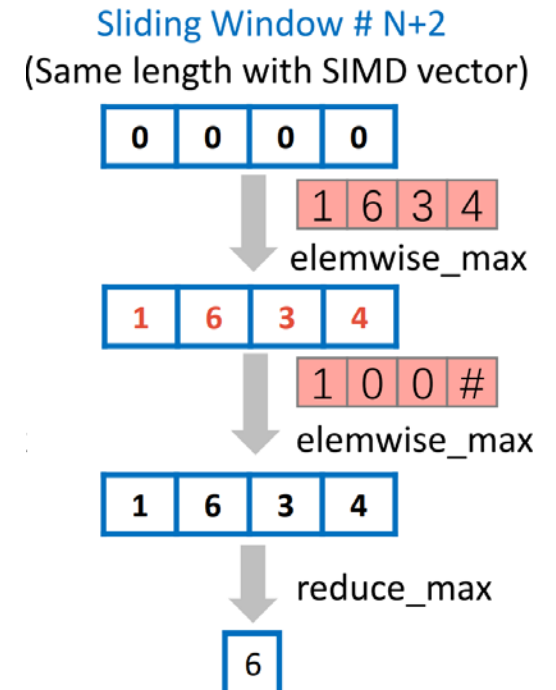
# Sliding Window: Eliminate redundant computation

- Observation: In most rounds (>60%), no maximum value update
- Optimization: Sliding-Window maximum tracking and unnecessary update avoidance



Update max score from 8 to 9

Calculate max score position



No update max score is 9

Ignore max score position

# Evaluation Methodology

## □ Experimental Setup:

- CPU (AMD EPYC 7702 64-Core Processor with 256 GB of RAM)
- GPU (NVIDIA v100)

## □ Datasets:

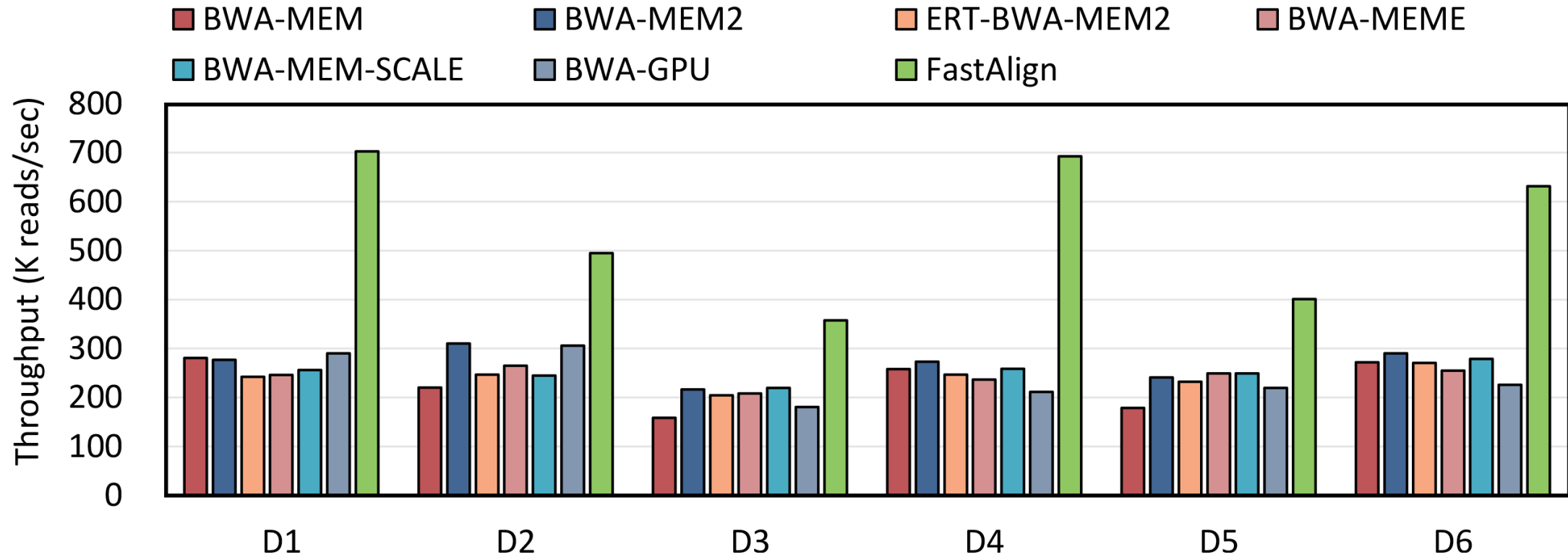
- Six experimental human datasets (from NCBI)

Label	Type	Len	#Reads	Source
D1	WES	144 bp	$1.1 \times 10^8$	SRR25735653
D2	WGS	101 bp	$1.5 \times 10^9$	ERR194147
D3	WGS	150 bp	$1.0 \times 10^9$	SRR25735658
D4	WES	150 bp	$1.2 \times 10^8$	SRR25735654
D5	WGS	150 bp	$3.0 \times 10^8$	ERR9129380
D6	WES	145 bp	$1.8 \times 10^8$	SRR8381429

## □ Baselines:

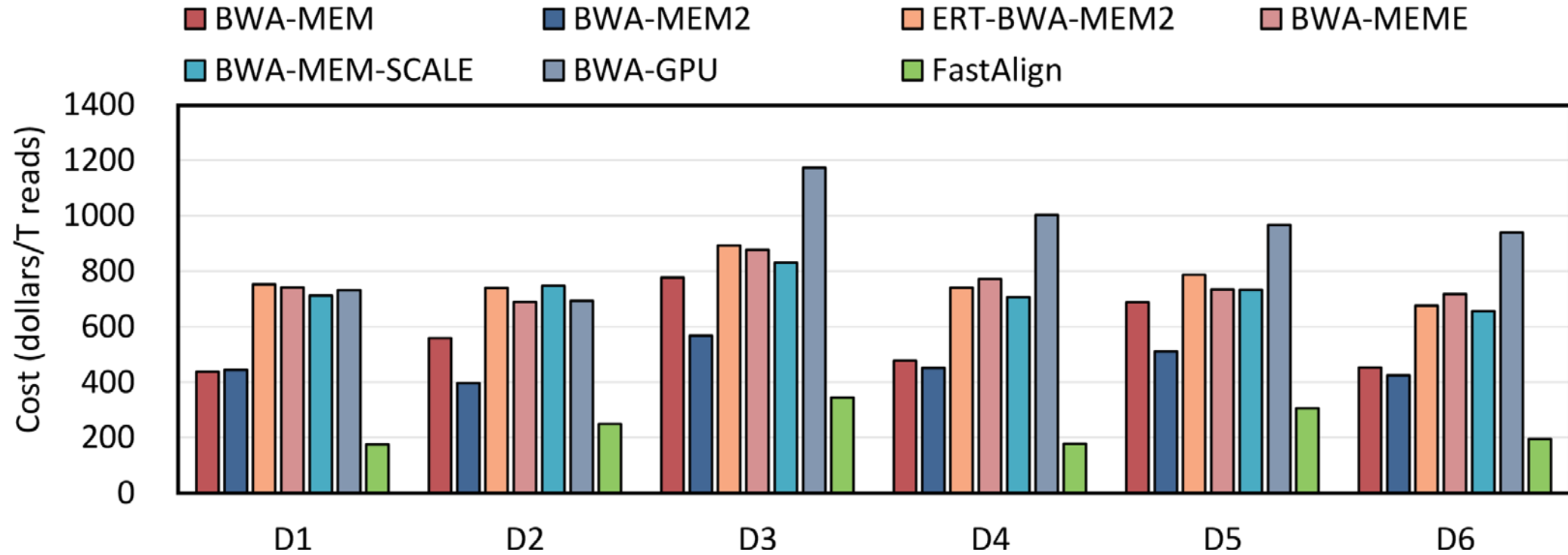
- BWA-MEM<sup>[BIO'10]</sup>, BWA-MEM2<sup>[IPDPS'19]</sup>, ERT-BWA-MEM<sup>[ISCA'21]</sup>, BWA-MEME<sup>[BIO'21]</sup>  
BWA-MEM-SCALE<sup>[ICPP'22]</sup>, BWA-GPU<sup>[ICS'23]</sup>

# Key Results – Performance



**FastAlign achieves up to 2.69×, 2.54×, 2.90×, 2.93×, 2.27×, and 3.28× speedup, respectively**

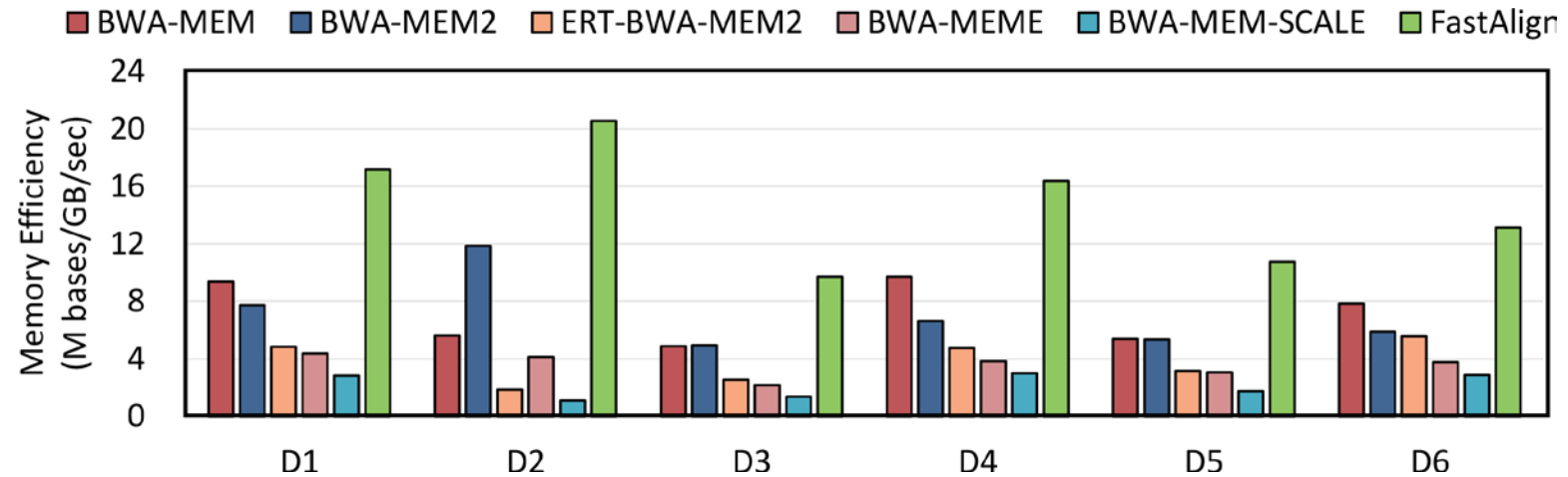
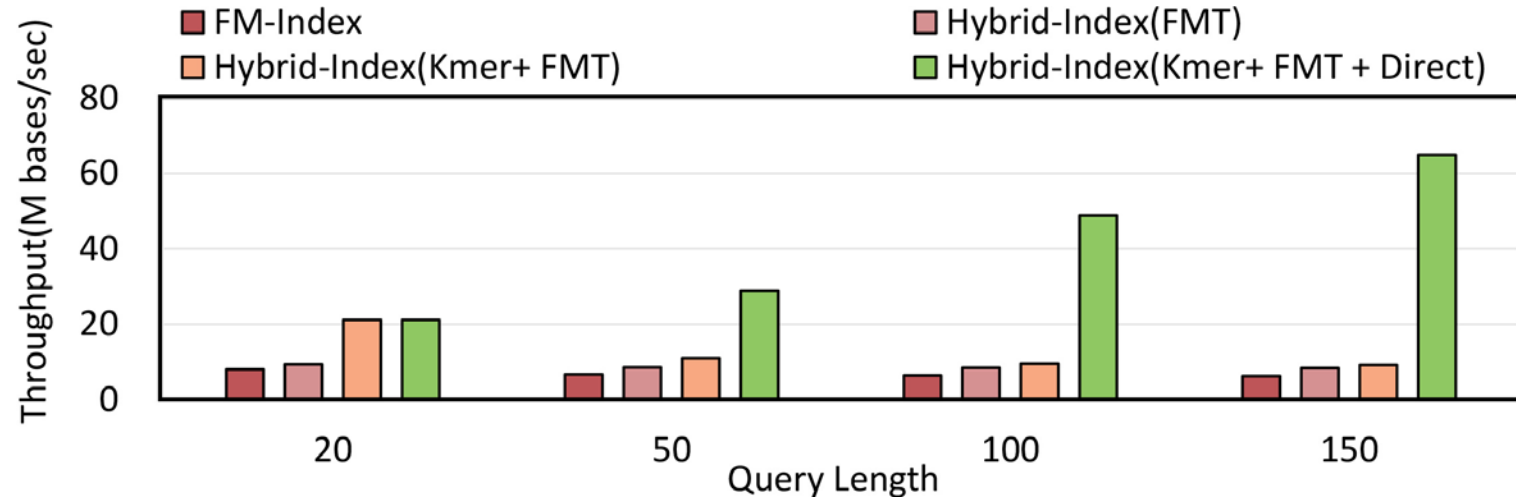
# Key Results – Cost



FastAlign achieves up to 2.69×, 2.54×, 4.30×, 4.24×, 4.07×, and 5.65× cost reduction, respectively.

# Key Results – Seeding

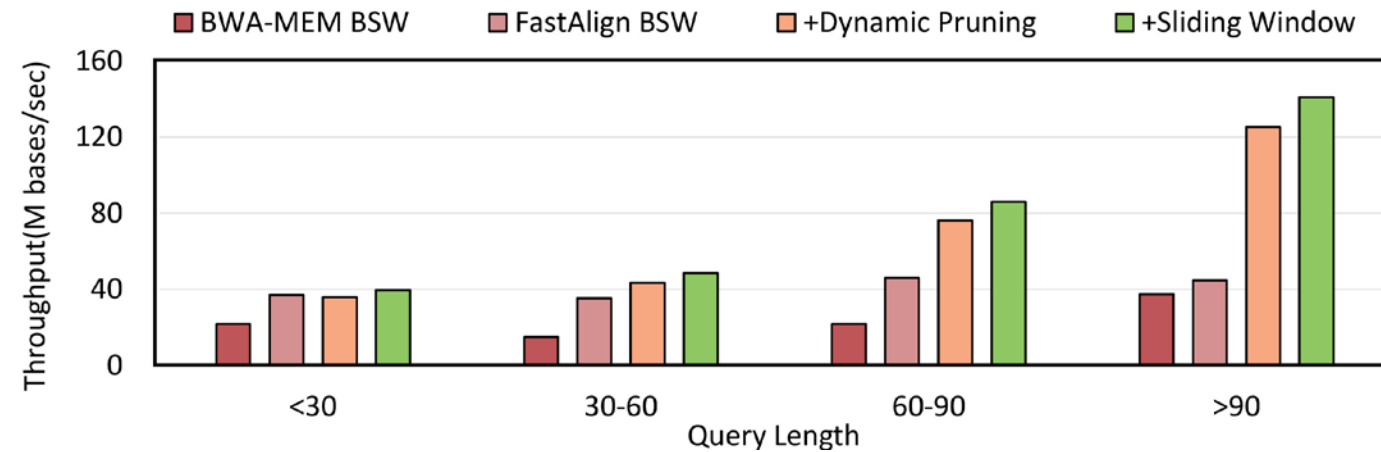
- The ablation results for the seeding phase
- The memory efficiency results for the seeding phase



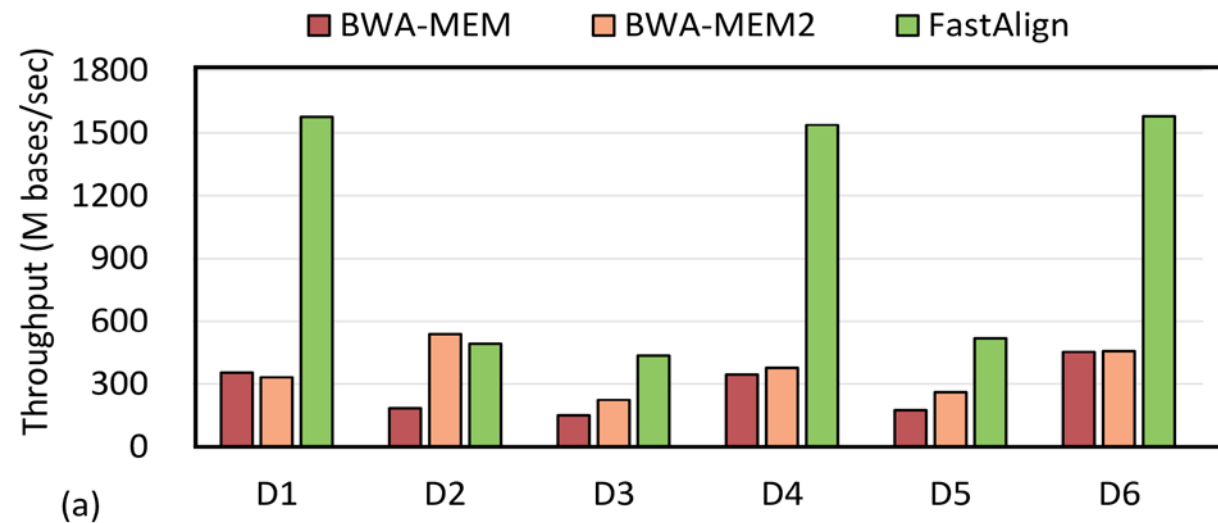
**FastAlign achieves up to 3.67x, 2.47x, 11.11x, 4.96x, and 18.92x speedup, respectively.**

# Key Results – Seed-Extension

- The ablation results for the seed-extension phase



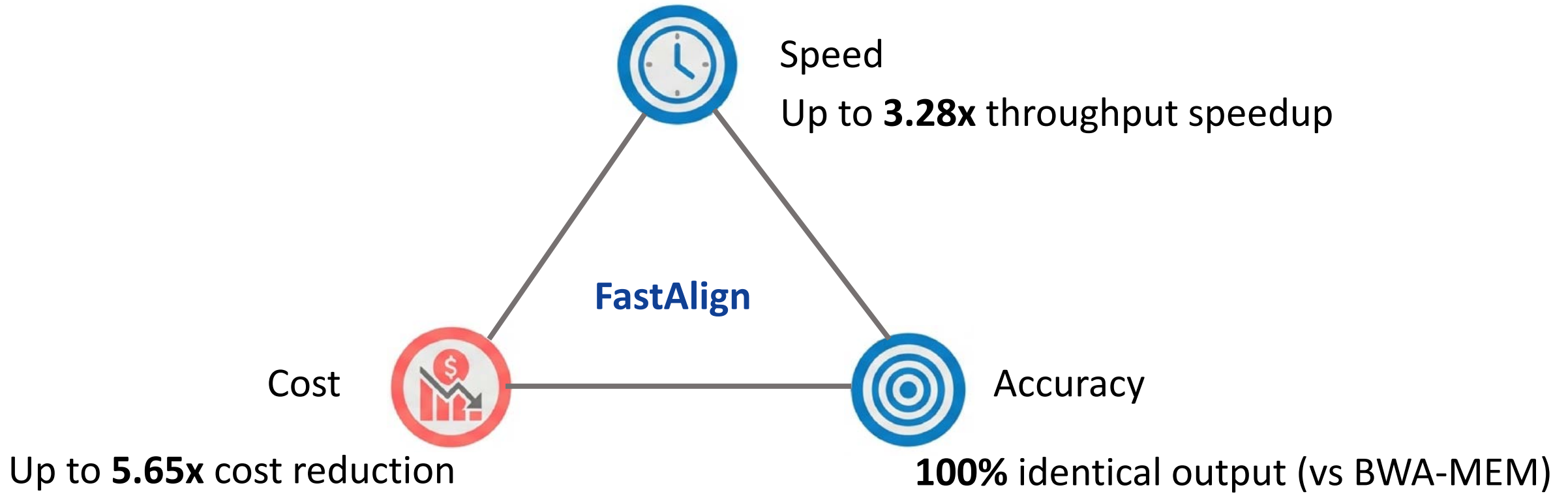
- The throughput results for the seed-extension phase



(a)

FastAlign achieves up to 4.44× and 4.77× throughput speedup, respectively in seed-extension phase.

# Conclusion



Open-source at <https://github.com/zzhofict/BWA-FastAlign>



ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP 2026)

# Faster and Cheaper: Pushing the Sequence Alignment Throughput with Commercial CPUs

Thanks for your attention



Open-source at <https://github.com/zzhofict/BWA-FastAlign>

