From Processing-in-Memory to Processing-in-Storage (PRinS)

Roman Kaplan, Leonid Yavits and Ran Ginosar

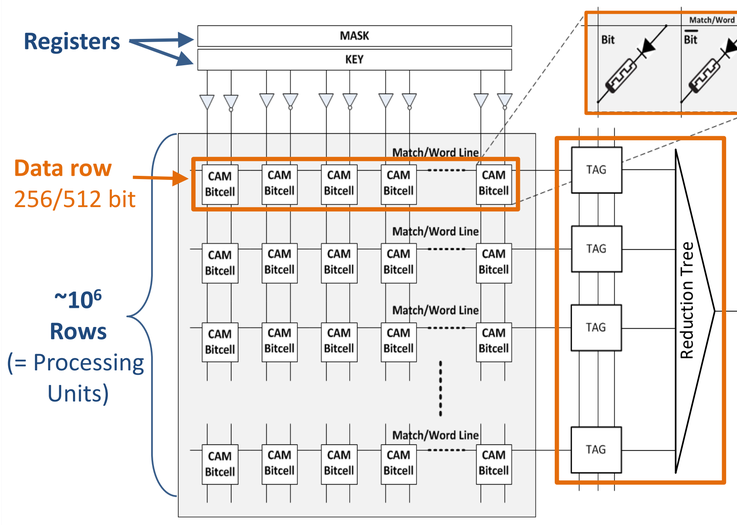
Technion, Israel Institute of Technology

Problem and Solution Overview

Storage ⇔ Memory Bottleneck Problem

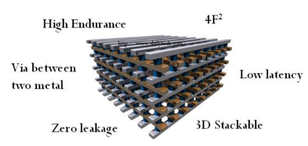
- PiM performs the bulk of computation in main memory
  - ✓ Reduces Mem→CPU transfers
- **Storage → Mem data transfers are the main bottleneck**
- Processing <u>in-storage</u> can increase performance and reduce energy by 100×

Resistive Content Addressable Memory (ReCAM): A PRinS Device



Resistive Bitcells

- Based on resistive elements (e.g., memristors)
- High density & non-volatile

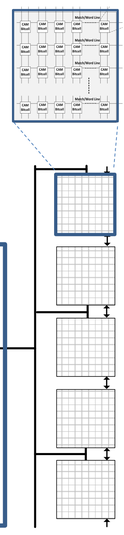


Logic next to data: *in-situ* processing

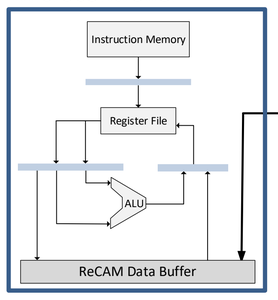
- Processing unit (PU) per row
- All PUs can operate in parallel
- **Instruction are based on bit-serial associative processing**

Full ReCAM-Based Storage System

- Divided to multiple ReCAM ICs
  - ✓ E.g., IC can be 256MB = 8M rows
- Linearly scalable performance
  - ➔More ICs = More parallelism



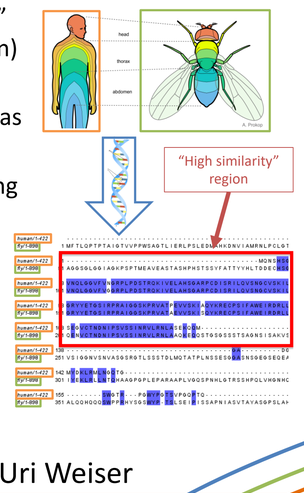
Microcontroller



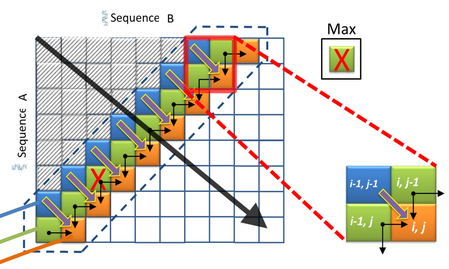
PRinS Application
DNA Local Sequence Alignment*

Smith-Waterman Algorithm

- Finds regions with "highest similarity" between two sequences (DNA\protein)
  - ✓ Proven to be optimal
- Every "match", "mismatch" & "gap" has a score
- S-W is based on dynamic programming
  - Fills a $m \times n$ matrix
  - Has a quadratic computational complexity $O(m \cdot n)$
- We focus on the computationally-heavy scoring step



- Matrix-fill order is on the main diagonal
- Entire antidiagonal is calculated in parallel
  - ✓ Antidiagonal per group of ReCAM columns
- Search for the maximal score
- Only 3 antidiagonals are required in each iteration

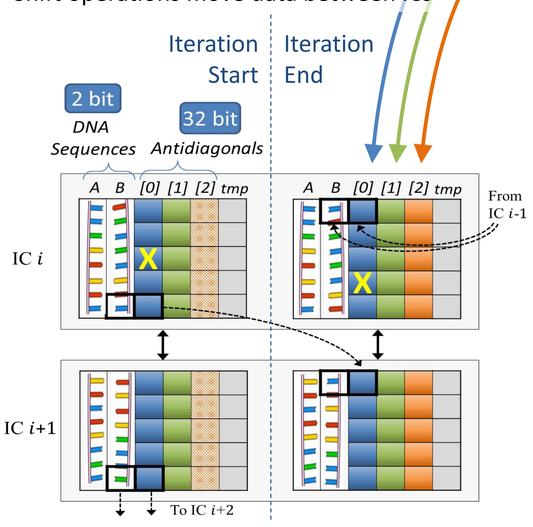


* This work was done with prof. Uri Weiser

PRinS Implementation & Performance Comparison

In-Storage Computation

- IC = Integrated Circuit
- Shift operations move data between ICs



Performance Comparison

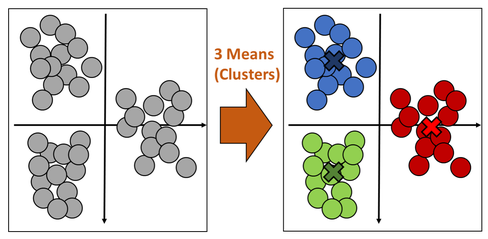
- Cycle-accurate simulator: 8GB of storage running at 500MHz
- Compared to multi-accelerator state-of-the-art solutions: FPGA, Xeon Phi and GPU
- ReCAM shows **4.7**× higher performance than a 384-GPU solution

Accelerator	FPGA	Xeon Phi	GPU	ReCAM
Performance (TCUPS)	6.02	0.23	11.08	**52.68**
# of ICs	128	4	**384**	32
Publication	L.Wienbrandt. **The FPGA-based High-Performance Computer RIVYERA for Applications in Bioinformatics.** In *Language, Life, Limits: 10th CiE*, pages 383-392. Springer 2014.	Y. Liu and B. Schmidt. 2014b. **SWAPHI: Smith-Waterman protein database search on Xeon Phi coprocessors.** In *IEEE ASAP*, 184–185.	Sandes, E. F. O., et al. **"CUDAlign 4.0: Incremental Speculative Traceback for Exact Chromosome-Wide Alignment in GPU Clusters."** IEEE	

PRinS Application
K-Means Clustering

K-Means Clustering Algorithm

- Common Machine Learning algorithm
- For each sample, finds its group among possible K
- Widely used in many fields, including:
  - Image processing
  - Anomaly detection
- Data intensitve
  - Multiple iterations over entire dataset



<u>Two Main Loops</u>

1. For each Sample:
   For each Center:
   Calculate distance(Sample, Center)
2. For each Center:
   For each Sample:
   Sum += Sample

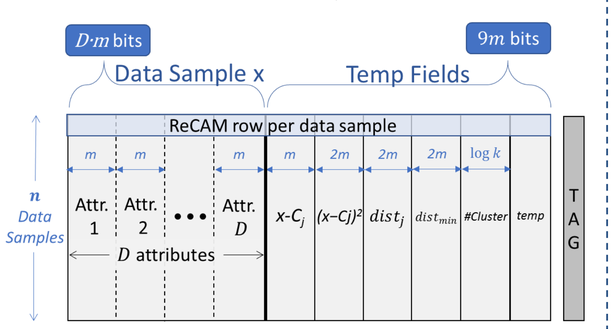
$O(n)$ in a von Neumann machine

$\boldsymbol{O(1)}$ **in ReCAM**

PRinS Implementation & Performance Comparison

In-Storage Computation

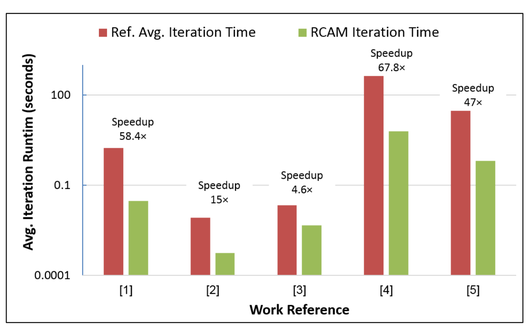
- In large datasets: #Clusters #Data Samples
- Computation over **ALL** samples in one instruction
- Each sample attributes requires multiple temp fields
  - Difference between center's attribute
  - Squared difference
  - Total distance between sample and center



[1] Bhimani, Janki, *et al.* "*Accelerating K-Means clustering with parallel implementations and GPU computing.*" HPEC 2015.
[2] Li, Zhehao, *et al.* "*High-performance K-means Implementation based on a Coarse-grained Map-Reduce Architecture.*" arXiv:1610.05601 (2016).
[3] Ramanathan, Nadesh, *et al.* "*A Case for Work-stealing on FPGAs with OpenCL Atomics.*" SIGDA 2016.
[4] Ding, Yufei, *et al.* "*Yinyang k-means: A drop-in replacement of the classic k-means with consistent speedup.*" ICML 2015.
[5] Rossbach, Christopher J. *et al.* "Dandelion: a compiler and runtime for heterogeneous systems.", SOSP 2013.

Performance Comparison to other Works

Work Reference	Platform	Dataset			Clusters
		Samples	Attributes	Size on disk	
[1]	GPU	1.4M	5	21.3MB	240
[2]	FPGA	2M	4	31.6MB	4
[3]	FPGA	1M	1	4MB	128
[4]	Intel i7	2.5M	68	318.8MB	10000
[5]	10-GPU Cluster	1B	40	152.7GB	120

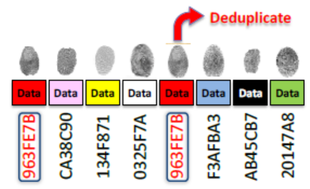


PRinS Application: **In-Storage Deduplication**

What is Deduplication?

- Deduplication is a technique for storing a single copy of each data block in storage
  - ✓ Can reach 10x reduction in data volume
- How it works:
  1. Data is broken into fixed blocks

  2. A fingerprint (FP) is calculated for each block
  3. Only pointers are stored for identical blocks

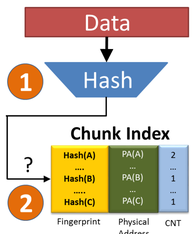


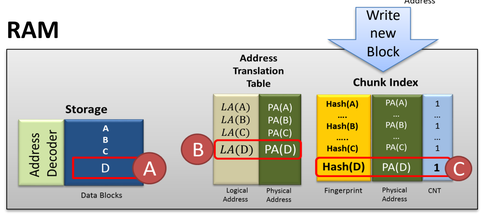
Traditional Systems vs. ReCAM

Traditional (RAM+CPU) Systems

New block write:

1. Hash (create key)
2. Search in key table
3. Write to three tables in RAM (A B & C)

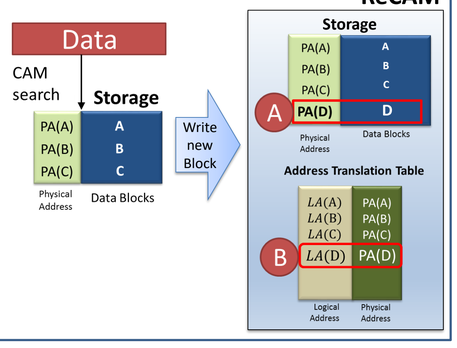




In-ReCAM Deduplication

Use CAM operations:

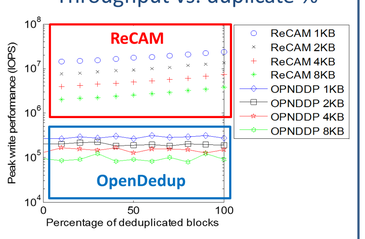
1. CAM search → No need to hash
2. Write block + 1 pointer



Performance Evaluations

- ReCAM was simulated with a cycle-accurate simulator
- ReCAM Parameters: 256GB @1GHz
- Opendedup executed on high-end server: 4×8 octa-core CPU, 64GB RAM, 800GB SSD drive
- **ReCAM has 100× higher throughput than deduplication with RAM+CPU**
- Energy consumption is similar or lower for the common block sizes (4 & 8KB)

Throughput vs. duplicate %



Energy vs. duplicate %