JMT/MTP Evaluation-I

Wamique Zia
2023CSM1020

Project Description

1. Adaptive Allocation:
   Cache Replacement Policies at L2/L3 level.
   - Under the mentorship of Gautam Hazari @ AMD.

2. Exploiting parallelism to enhance Memory Performance.
   - Under the mentorship of Dr. T V Kalyan @ IIT Ropar.

Project 1 : Adaptive Allocation
Cache Replacement Policies at L2/L3 level.

➔ The basic idea relies on the '**Set Dueling**' concept.

➔ Compares the performance of two policies on '**leader sets**' and winning is policy is applied to remaining sets called '**follower sets**'.

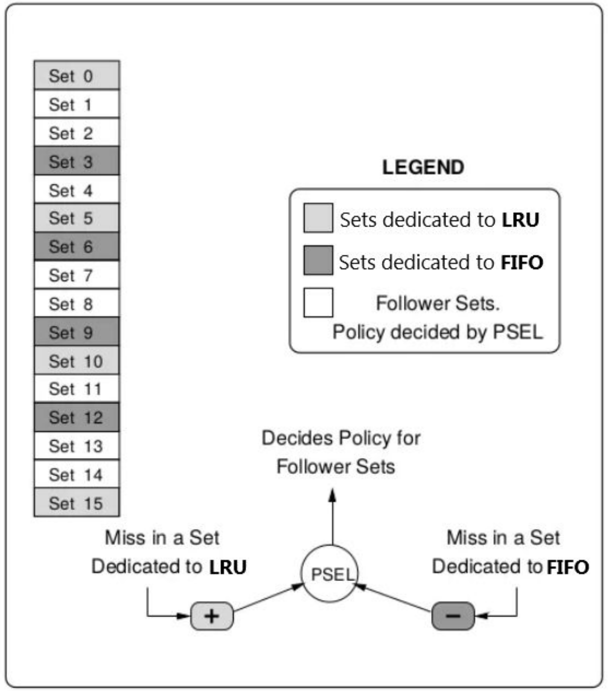


Fig. 1. Set Dueling b/w LRU and FIFO

Project 2 :
Exploiting parallelism to enhance Memory Performance

1. Introduction
2. Literature Survey.
3. Trace collection for Graph and Genomics Applications.
4. DDR Frequency Sensitivity Analysis.

Project 2 : Introduction

- **Why do we require it ?**
  - Increased Memory Footprint
  - Cache Locality Limitations
  - Performance Improvement
  - Energy Efficiency

Project 2 : Literature Survey

- Different ideas/ways to enhance memory performance.
  - Dynamic ROW ACTivation
  - DRAM Row Cache
  - Employing NoC packet routing mechanisms
  - Row/Page Management Policies
  - Perceptron Learning

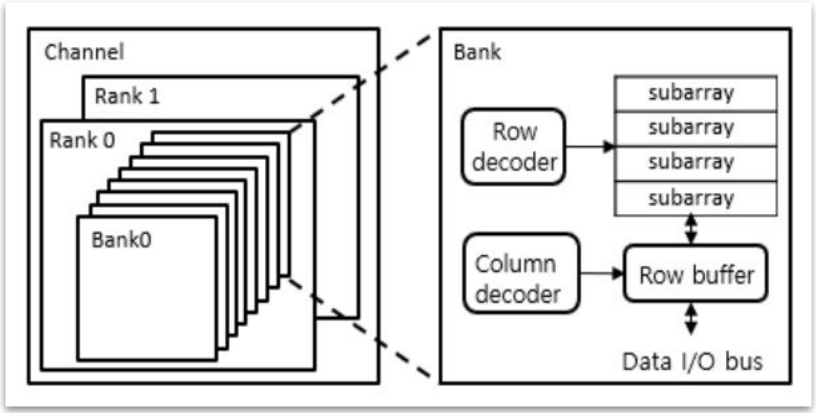
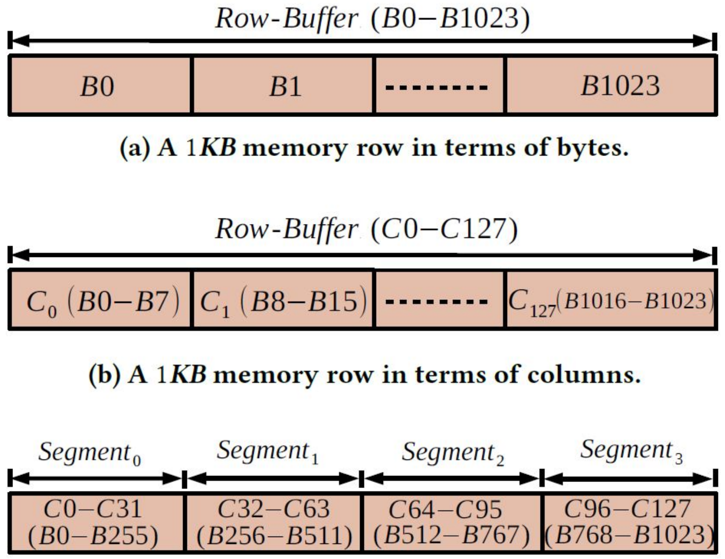


Fig. 2. Conventional DRAM Organization

Dynamic/Partial ROW ACTivation

- ➔ Dynamic Detection of Row Size
  - ◆ Monitor Access Behavior, and
  - ◆ History Information Maintenance
- ➔ Calculating the Permutation Rate
  - ◆ Access Patterns
  - ◆ Granularity of Tracking
- ➔ Dynamic Activation Process
  - ◆ Row Activation
  - ◆ Early Precharge



(a) A 1*KB* memory row in terms of bytes.

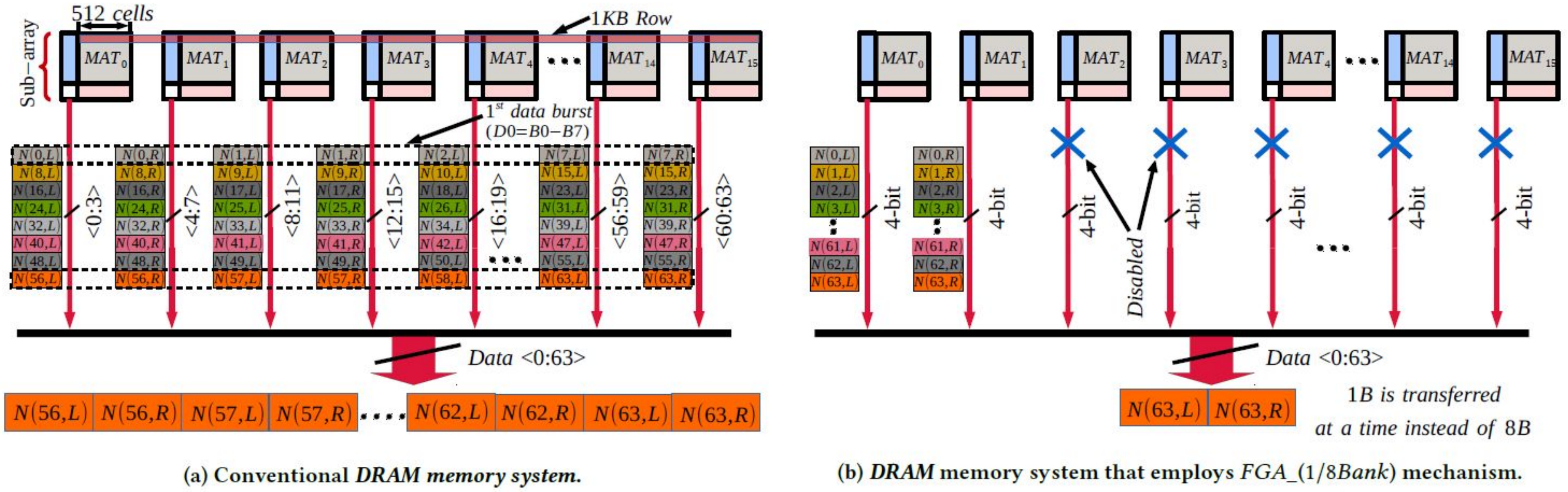
(b) A 1*KB* memory row in terms of columns.

(c) A 1*KB* memory row divided into 4 equal-sized segments.

Fig. 3. 1KB memory row (page)

Source: *Dynamic row activation mechanism for multi-core systems*, published in "18th ACM International Conference on Computing Frontiers, 2021"

Dynamic/Partial ROW ACTivation (contd.)


512 cells
1KB Row
Sub−array
MAT0
MAT1
MAT2
MAT3
MAT4
MAT14
MAT15
1st data burst (D0=B0−B7)
<0:3>
<4:7>
<8:11>
<12:15>
<16:19>
<56:59>
<60:63>
Data <0:63>
N(56,L) N(56,R) N(57,L) N(57,R) N(62,L) N(62,R) N(63,L) N(63,R)
4-bit
Disabled
1B is transferred at a time instead of 8B

(a) Conventional *DRAM memory system.*

(b) *DRAM* memory system that employs *FGA_(1/8Bank)* mechanism.

Fig. 4. An example shows the data fetching and accommodating in conventional DRAM memory systems and proposed DRA mechanism.

Source: *Dynamic row activation mechanism for multi-core systems*, published in "18th ACM International Conference on Computing Frontiers, 2021"

Dynamic/Partial ROW ACTivation (contd.)

➔ Result:
- ◆ The Dynamic Row Activation (DRA) mechanism was tested using workloads from the MediaBench and PARSEC benchmark suites.
- ◆ DRA improves DRAM access latency by 12.3%.
- ◆ Achieves an average 39% reduction in ACTivaiton power.
- ◆ 12.8% reduction total DRAM energy consumption.

Source: *Dynamic row activation mechanism for multi-core systems*, published in “18th ACM International Conference on Computing Frontiers, 2021”

DRAM Row Cache

- ➔ The Row Buffer Cache (RBC) architecture is designed to cache the contents of the row buffer in an SRAM cache located within the memory controller. This allows the system to maintain access to frequently used rows, even when conflicts occur.

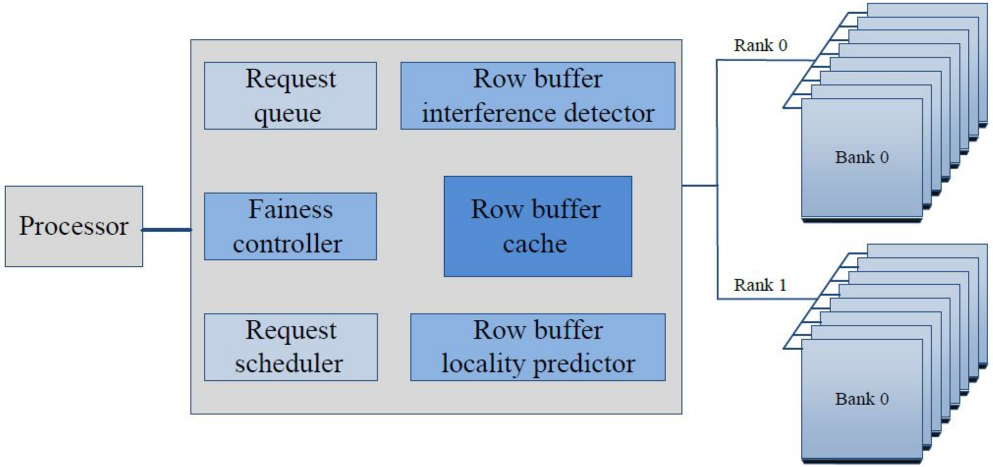


Fig. 5. Simplified View of Row Buffer Cache (RBC)

Source: *RBC: A memory architecture for improved performance and energy efficiency*, published in “Tsinghua Science and Technology, 2021”

DRAM Row Cache (contd.)

- ➔ **Row Buffer Interference Detector:** This component quantifies the severity of row buffer interference for each row, identifying which rows are most affected by conflicts.
- ➔ **Row Buffer Locality Predictor:** This predictor uses the interference data to determine which rows have good locality and should be cached in the RBC.
- ➔ **Fairness Controller:** This ensures that all applications receive fair access to the RBC, preventing any single application from monopolizing the cache.

DRAM Row Cache (contd.)

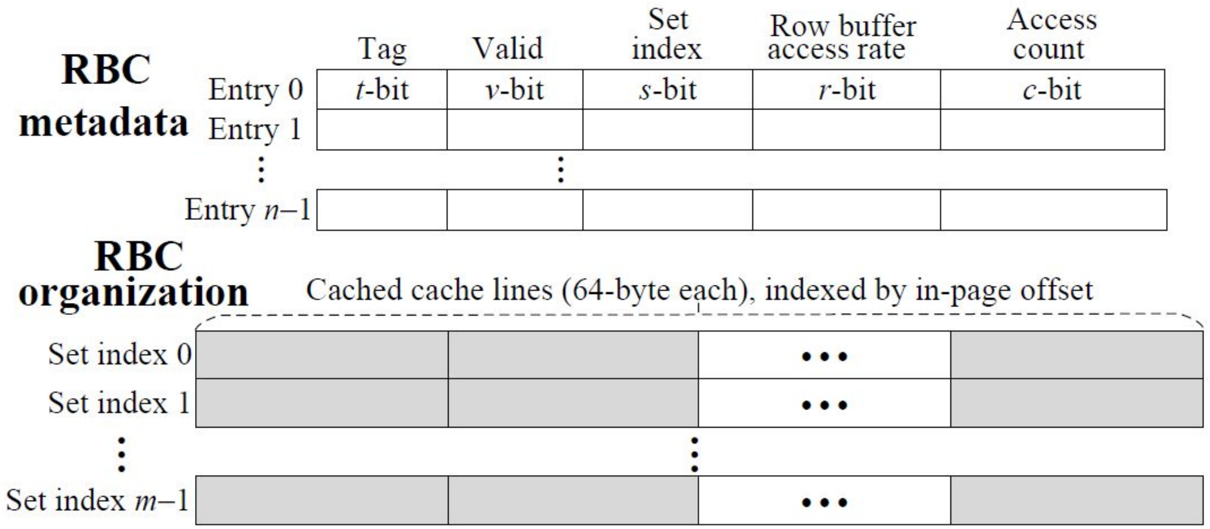


Fig. 6. Organization of RBC

Source: *RBC: A memory architecture for improved performance and energy efficiency*, published in “Tsinghua Science and Technology, 2021”

DRAM Row Cache (contd.)

➔ Result:

- ◆ In single-core simulations, RBC achieved up to a 2.24x performance improvement (16.1% on average).
- ◆ In multi-core simulations, performance increased by up to 1.55x (17% on average).
- ◆ Memory energy consumption was reduced by up to 68.2% (23.6% on average) in single-core simulations, and up to 35.4% (21:3% on average in multi-core simulations.

Employing NoC packet routing mechanisms

➔ Challenges with heterogeneous MPSoCs:
- ◆ Memory Locality Dilution
- ◆ Inadequate QoS Support
- ◆ Increased Latency in Network-on-Chip (NoC)

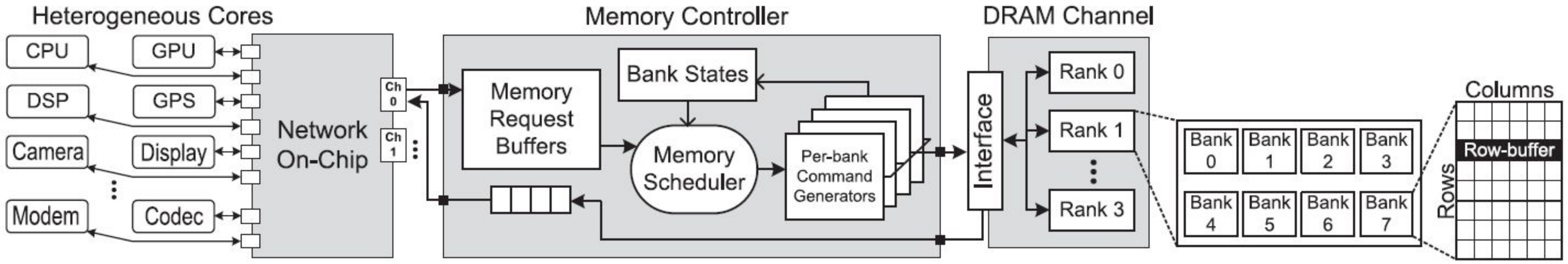
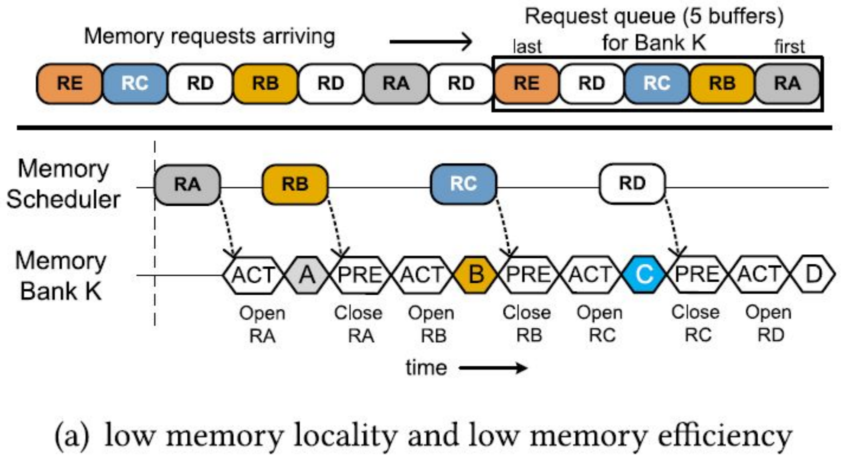


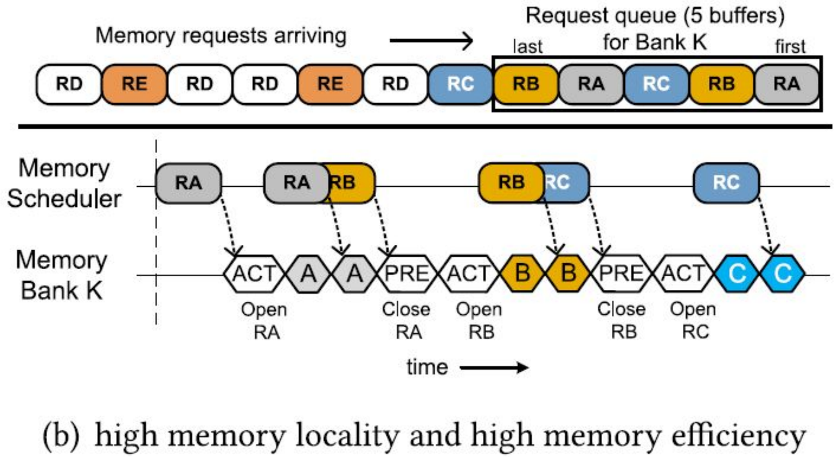
Fig. 7. Architecture of memory subsystem with a NoC for heterogeneous MPSoC system

Source: *Improving Memory Efficiency in Heterogeneous MPSoCs through Row-Buffer Locality-aware Forwarding*, published in “ACM Transactions on Architecture and Code Optimization (TACO), 2020”

Employing NoC packet routing mechanisms (contd.)



(a) low memory locality and low memory efficiency



(b) high memory locality and high memory efficiency

Fig. 8. An example of memory locality of NoC traffic affecting memory efficiency in DRAM

Source: *Improving Memory Efficiency in Heterogeneous MPSoCs through Row-Buffer Locality-aware Forwarding*, published in "ACM Transactions on Architecture and Code Optimization (TACO), 2020"

Employing NoC packet routing mechanisms (contd.)

- ➔ Locality-aware Forwarding Mechanism
  - Request Handling: When a memory request arrives at the NoC router, the router examines the row-index cache to determine if there are other requests for the same row.
  - Forwarding Strategy: Router uses a locality-aware allocation scheme that considers both the row indices and the QoS requirements of the requests.
  - Latency Reduction: Ensures that requests for the same row are forwarded in close succession

Source: *Improving Memory Efficiency in Heterogeneous MPSoCs through Row-Buffer Locality-aware Forwarding*, published in “ACM Transactions on Architecture and Code Optimization (TACO), 2020”

Employing NoC packet routing mechanisms (contd.)

- Result:
  - The locality-aware NoC router achieves a 58.32% increase in Row-Buffer Hits.
  - The average memory latency is reduced by 14.45%.
  - The design also results in a net reduction in energy costs for both DRAM and NoC operations by 27.82%.

Source: *Improving Memory Efficiency in Heterogeneous MPSoCs through Row-Buffer Locality-aware Forwarding*, published in "ACM Transactions on Architecture and Code Optimization (TACO), 2020"

Row/Page Management Policies

➔ Problems:
- ◆ Fallback Allocation
- ◆ Promotion Failure
- ◆ Migration to CPU-less Nodes
- ◆ Inefficient Page Reclamation

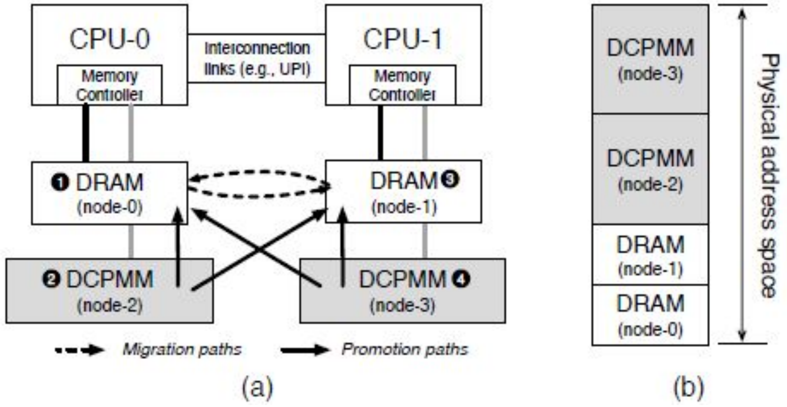


Fig. 9. Multi-tiered memory system augmented on the NUMA architecture

Source: *Exploring the Design Space of Page Management for Multi-Tiered Memory Systems*, published in "USENIX ATC , 2021"

Row/Page Management Policies

- ➔ Auto-Tiering :
  - ◆ Promotion and Migration
    - ● CPM (Conservative Promotion and Migration)
    - ● OPM (Opportunistic Promotion and Migration)
  - ◆ Latency Hiding
  - ◆ Reclamation

Source: *Exploring the Design Space of Page Management for Multi-Tiered Memory Systems*, published in "USENIX ATC , 2021"

Row/Page Management Policies

- Results:
  - Graph500 and GraphMat show performance increase of 17% and 19%, respectively.
  - SPECAccel Workloads shows mixed results, with some sub-tests having 7-8% performance increase.
  - OPM with Background Demotion provided up to a 6.17× speedup.
  - OPM-BD outperformed Intel's Tiering approach by up to 6.99× in workloads like Graph500 and GraphMat.

Source: *Exploring the Design Space of Page Management for Multi-Tiered Memory Systems*, published in “USENIX ATC , 2021”

Perceptron Learning

➔ Traditional page management policies (i.e., static open- or close-page policies) are unable to adapt to dynamically changing memory behaviors, leading to inefficiencies. Existing dynamic page policies are either access-based or time-based but still lack the ability to fully adapt to long-term memory access behavior, which changes over time.

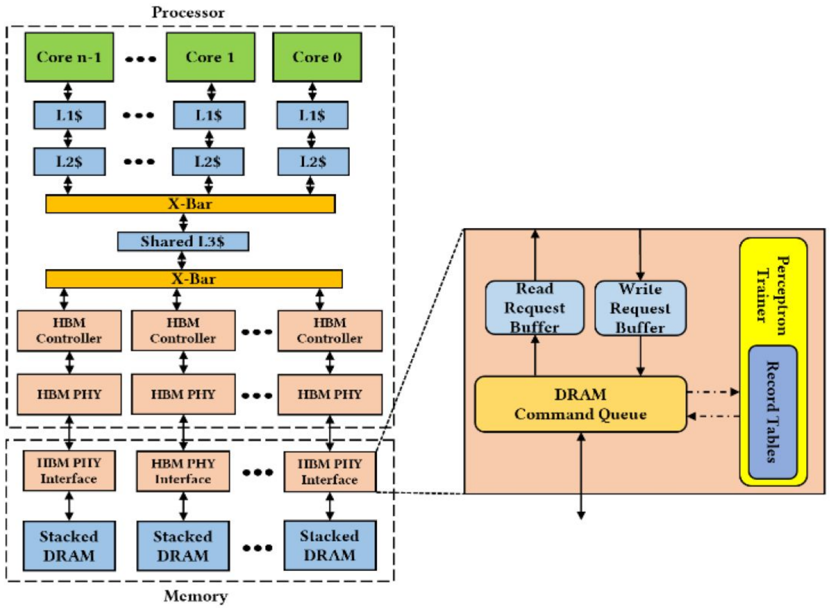


Fig. 10. Perceptron Trainer in HBM system

Perceptron Learning

➔ **DYMPL**, a perceptron-based dynamic page policy that adapts to the changing memory access behavior of applications by learning from past access patterns.

➔ Working:

- ◆ Feature Extraction
- ◆ Prediction
- ◆ Training
- ◆ Adaptation

Perceptron Learning

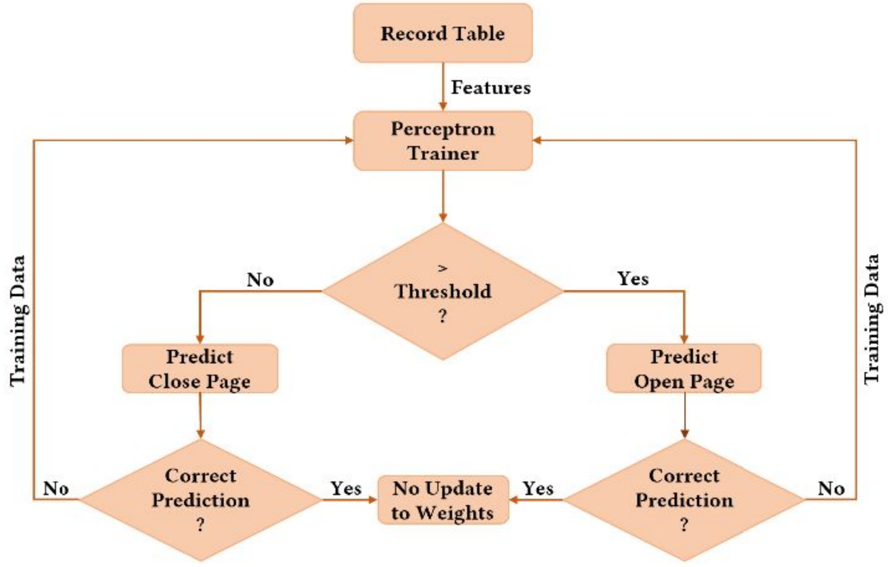


Fig. 11. Perceptron Training

Source: *Dynamic Page Policy Using Perceptron Learning*, published in "International Symposium on Memory Systems (MEMSYS), 2022"

Perceptron Learning

- Results:
  - DYMPL reduced row-buffer conflicts by 19.4% on average compared to static open-page policies.
  - Overall system performance improved by 5.5% on average.
  - Compared to statically implemented open- and close-page policies, the perceptron-based approach achieved performance improvements of 12.6% and 19.8%, respectively.

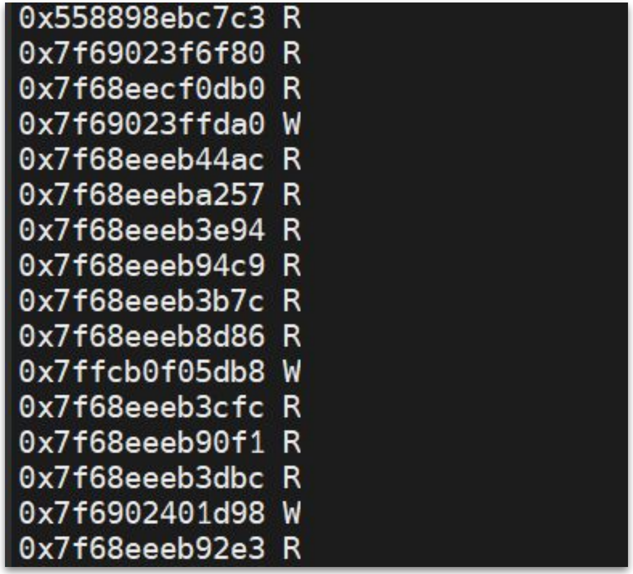
Source: *Dynamic Page Policy Using Perceptron Learning*, published in “International Symposium on Memory Systems (MEMSYS), 2022”

Trace Collection

1. GAPBS
   - BFS, PR, SSSP

2. GenomicsBench
   - BSW, CHAIN, DBG, PILEUP

3. Used Intel Pintool for tracing.



```
0x558898ebc7c3 R
0x7f69023f6f80 R
0x7f68eecf0db0 R
0x7f69023ffda0 W
0x7f68eeeb44ac R
0x7f68eeeba257 R
0x7f68eeeb3e94 R
0x7f68eeeb94c9 R
0x7f68eeeb3b7c R
0x7f68eeeb8d86 R
0x7ffcb0f05db8 W
0x7f68eeeb3cfc R
0x7f68eeeb90f1 R
0x7f68eeeb3dbc R
0x7f6902401d98 W
0x7f68eeeb92e3 R
```

Fig. 12. Sample Memory-Driven Trace File

Frequency Sensitivity Analysis

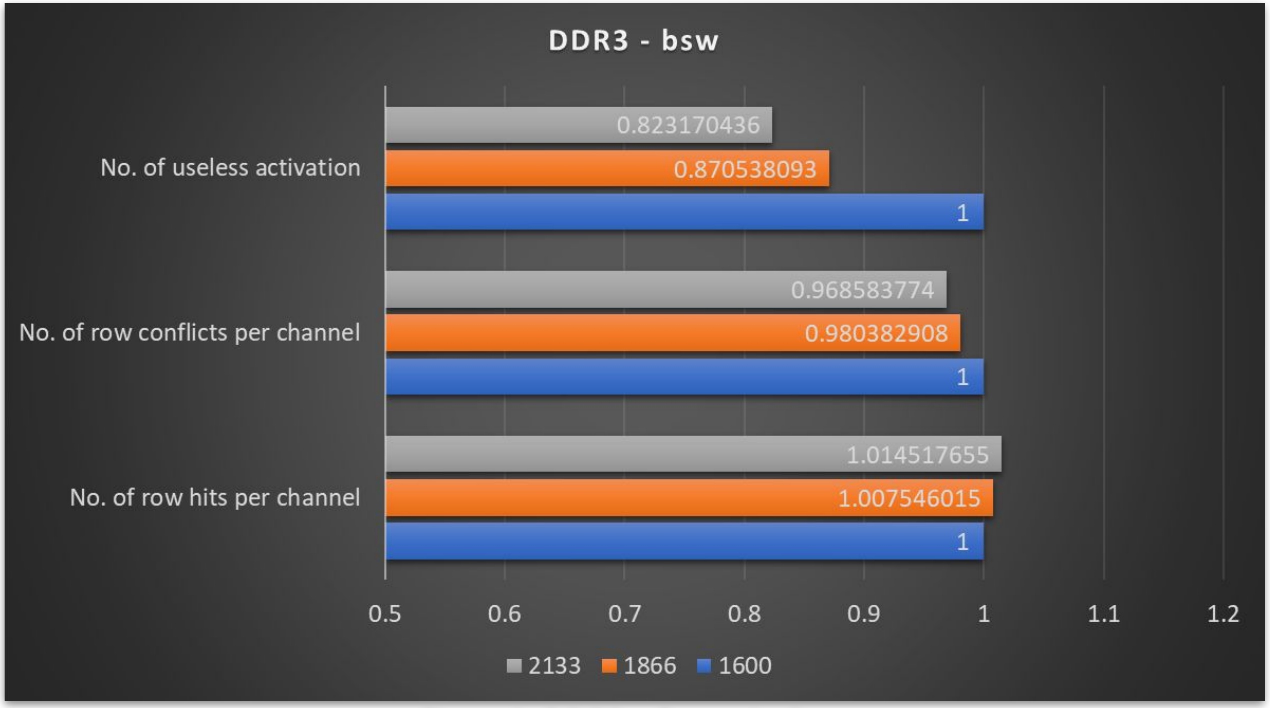


Chart. 1. DDR3 - Genomics/bsw

Frequency Sensitivity Analysis (contd.)

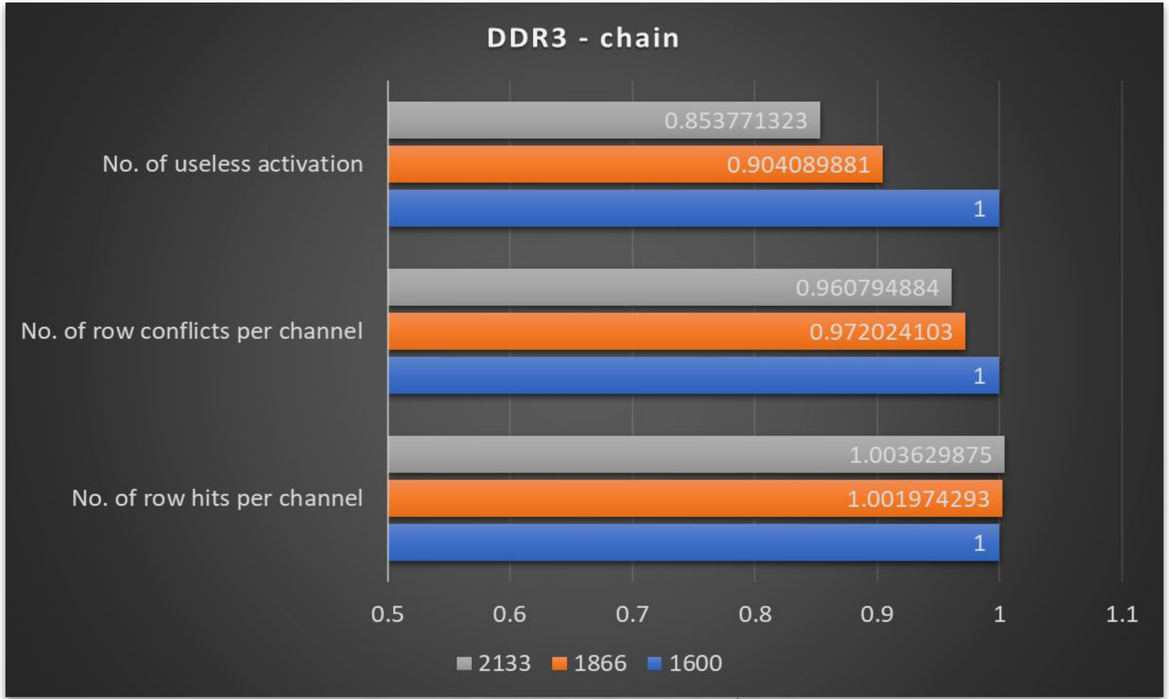


Chart. 2. DDR3 - Genomics/chain

Frequency Sensitivity Analysis (contd.)

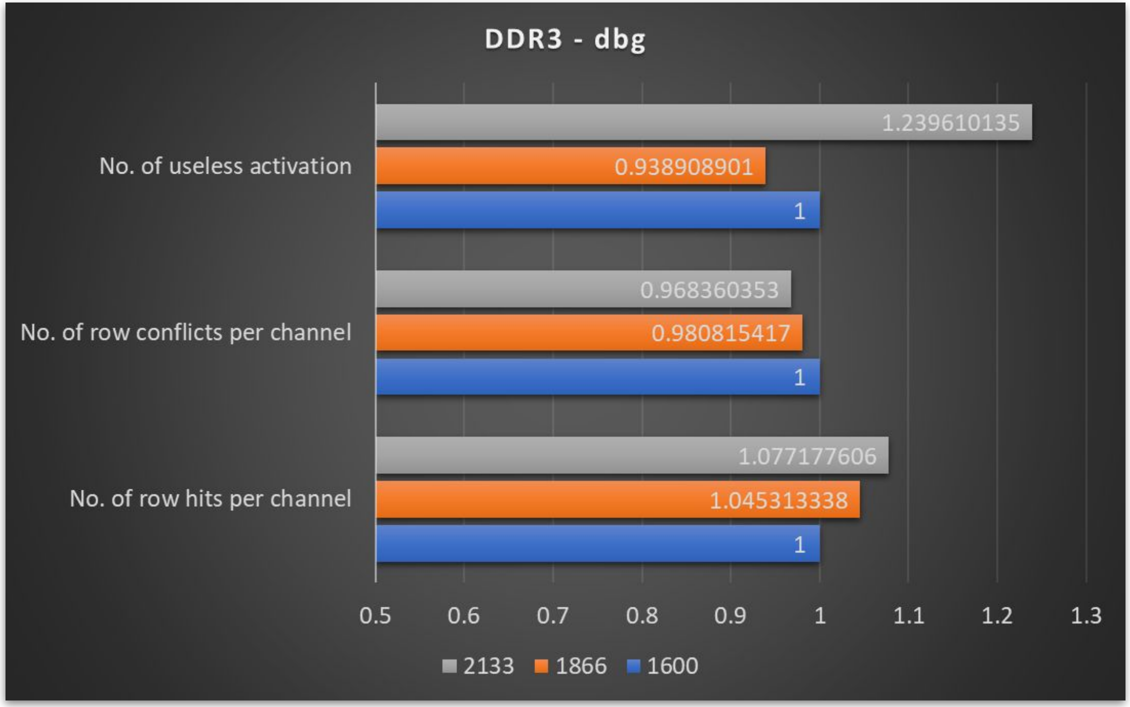


Chart. 3. DDR3 - Genomics/dbg

Frequency Sensitivity Analysis (contd.)

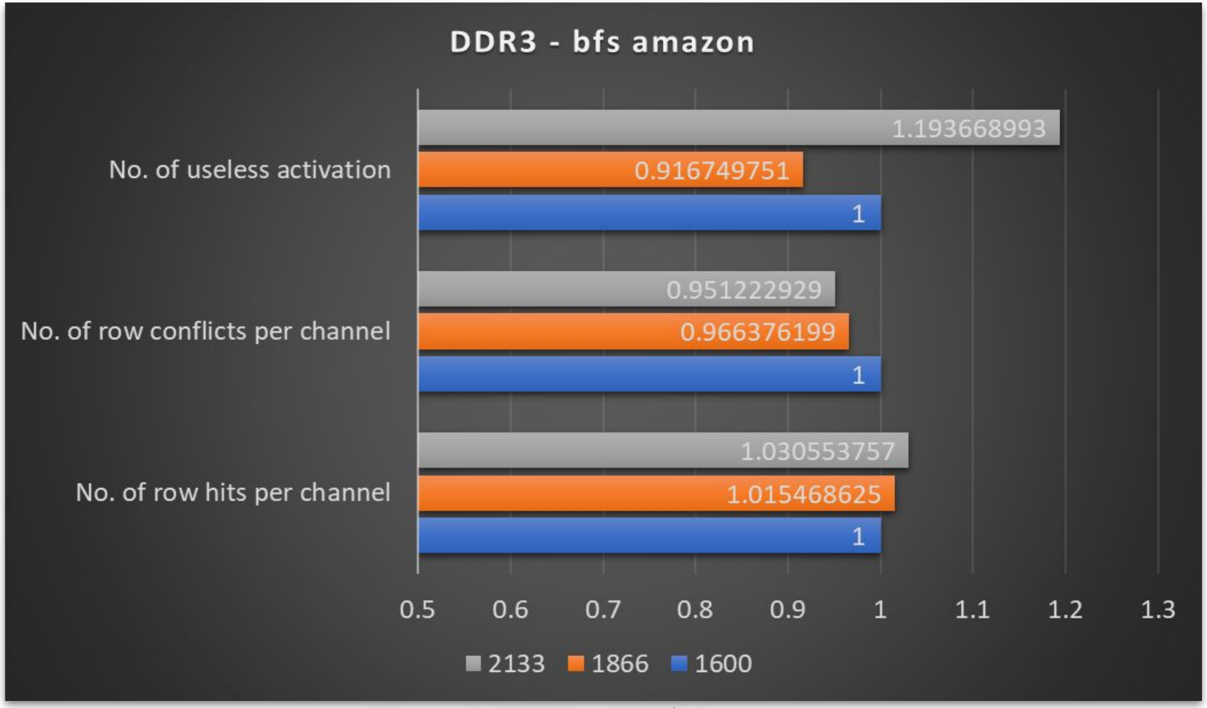


Chart. 4. DDR3 - GAP/bfs_amazon

Tools Used

1. **Ramulator** by CMU Safari group.
   a. It is a fast and cycle-accurate DRAM simulator.
   b. Supports memory-driven and cpu-driven trace.
   c. Supports various DRAM standards, such as, DDR3, DDR4, LPDDR3, LPDDR4, GDDR5, and many more.

Future Work

1. Trace Collection for more applications.
2. Data bus width sensitivity analysis.
3. Per Column detailed study of memory access patterns.

Thank-You