NEAR DATA FILTERS: TAKING ANOTHER BRICK FROM THE MEMORY WALL

Diego G. Tomé
CWI, The Netherlands
diego.tome@cwi.nl

Tiago R. Kepe
UFPR and IFPR, Brazil
trkepe@inf.ufpr.br

Marco A. Z. Alves
UFPR, Brazil
mazalves@inf.ufpr.br

Eduardo C. de Almeida
UFPR, Brazil
eduardo@inf.ufpr.br
NEAR DATA FILTERS: TAKING ANOTHER BRICK FROM THE MEMORY WALL

INTRODUCTION
MEMORY WALL PROBLEM

- The disparity between CPU performance and main memory latency has grown tightly.
The disparity between CPU performance and main memory latency has grown tightly.
MEMORY WALL PROBLEM

- The disparity between CPU performance and main memory latency has grown tightly.

Hitting the Memory Wall
W. Wulf and McKee S.A.
ACM Computer Architecture News, 1994
The disparity between CPU performance and main memory has grown tightly.

Hitting the Memory Wall
W. Wulf and McKee S.A.
ACM Computer Architecture News, 1994
The disparity between CPU performance and main memory has grown tightly.

**Hitting the Memory Wall**
W. Wulf and McKee S.A.
ACM Computer Architecture News, 1994

**Hybrid memory cube**
J. Jeddeloh and B. Keet.
Symposium on VLSI (VLSIT), 2012
QUERY PROCESSING: WHERE DOES TIME GO?

Figure 1: Top time consuming database operators in MonetDB [9] running the 100 GB TPC-H benchmark.

TPC-H Queries - operators breakdown

Monetdb: Two decades of research in column-oriented database architectures
QUERY PROCESSING: WHERE DOES TIME GO?

Figure 1: Top time consuming database operators in MonetDB [9] running the 100 GB TPC-H benchmark.

76% IS SPENT IN SELECTION AND PROJECTION

TPC-H Queries - operations breakdown

Monetdb: Two decades of research in column-oriented database architectures
DATA MOVEMENT IN DATABASE SYSTEMS

Memory Hierarchy

- CPU Registers: 1000 bytes, 300 ps
- Level 1 Cache: 64 KB, 1 ns
- Level 2 Cache: 256 KB, 10 ns
- Last Level Cache: 4 MB, 20 ns
- Main Memory: 16 GB, 100 ns

Access time increasing

Capacity increasing

NEXT REQUESTS

Vector Scan

NEAR DATA FILTERS: TAKING ANOTHER BRICK FROM THE MEMORY WALL
DATA MOVEMENT IN DATABASE SYSTEMS
NEAR-DATA PROCESSING (NDP)

- Cost of moving data is significant
- Performs the data processing in the main memory
- Reduces energy consumption (data movement)
- Provides faster response time (no cache latency)
- With HMC the execution inside memory became tangible

Near-Data Processing: Insights from a MICRO-46 Workshop

Hybrid memory cube
J. Jeddeloh and B. Keet.
Symposium on VLSI (VLSIT), 2012
HYBRID MEMORY CUBE (HMC)

Hybrid memory cube
J. Jeddeloh and B. Keet.
Symposium on VLSI (VLSIT), 2012

TSV - THROUGH-SILICON VIAS
HYBRID MEMORY CUBE (HMC)

- Composed of 32 vaults
- Bandwidth up to 320GB/s
- Instruction execution (16 bytes at a time)
- 256 Bytes of row buffer
1. What happens when database systems run the select scan over the current x86 architecture using the HMC as ordinary DRAM?
DATABASE SYSTEMS PREDICATE PROCESSING

“SELECT <COLUMNS> FROM <TABLES> WHERE <CONDITION> AND/OR <CONDITION>”
DATABASE SYSTEMS PREDICATE PROCESSING

Query Plan

Processor Core

Pipeline

Load (value)

16 BYTES REQUEST

Cache Hierarchy

Cache line
64 Bytes

16 Bytes Operation

x86

HMC
DATABASE SYSTEMS PREDICATE PROCESSING

Query Plan: Selection Scan → Project

Processor Core:
- Pipeline: Load (value)
- 16 Bytes Request
- Cache Hierarchy:
  - Cache line: 64 Bytes

HMC:
- 64 Bytes Request
DATABASE SYSTEMS PREDICATE PROCESSING

Query Plan
- Project
- Selection Scan
- x86

Processor Core
- Pipeline
  - Load (value)
- 16 BYTES REQUEST

Cache Hierarchy
- Cache line 64 Bytes

HMC
- 256 BYTES AVAILABLE
- 64 BYTES REQUEST

NEAR DATA FILTERS: TAKING ANOTHER BRICK FROM THE MEMORY WALL
DATABASE SYSTEMS PREDICATE PROCESSING

Query Plan
- Project
- Selection Scan
- x86

Processor Core
- Pipeline
  - Load (value)
- Cache Hierarchy
  - Cache line
    - 64 Bytes

HMC
- 256 Bytes Available

64 BYTES RESPONSE
DATABASE SYSTEMS PREDICATE PROCESSING

Query Plan

Processor Core

Pipeline

Load (value)

16 Bytes Response

Cache Hierarchy

Cache line
64 Bytes

HMC

64 Bytes Returned

64 BYTES RETURNED

NEAR DATA FILTERS: TAKING ANOTHER BRICK FROM THE MEMORY WALL
DATABASE SYSTEMS PREDICATE PROCESSING

DATA-MOVEMENT

Query Plan

Processor Core

Pipeline

Load (value)

Cache Hierarchy

Cache line
64 Bytes

HMC

16 BYTES REQUEST

64 BYTES REQUEST

256 BYTES AVAILABLE

64 BYTES RESPONSE

16 BYTES RESPONSE
DATABASE SYSTEMS PREDICATE PROCESSING
DO MITIGATE DATA-MOVEMENT

Query Plan
Processor Core
HMC

16 Bytes Operation
Pipeline
Load (value)
Cache Hierarchy
Cache line
64 Bytes

x86
Selection Scan
Project

16 BYTES REQUEST
64 BYTES REQUEST
256 BYTES AVAILABLE
64 BYTES RESPONSE
16 BYTES RESPONSE

DATABASE SYSTEMS PREDICATE PROCESSING
DO MITIGATE DATA-MOVEMENT

NEAR DATA FILTERS: TAKING ANOTHER BRICK FROM THE MEMORY WALL
NEAR-DATA FILTERS FOR DATABASE SYSTEMS

- Allow the execution of filtering inside the HMC
- Take advantage of the HMC instruction execution and high bandwidth capacity to mitigate the “memory wall”
- Increase the parallelism of the filtering execution
- With extensions for the HMC ISA, filtering is done by branch-less decisions
1. What happens when database systems run the select scan over the current x86 architecture using the HMC as ordinary DRAM? **Memory wall is still a problem.**

2. Can we use the current HMC Instruction Set Architecture (ISA) to implement the near-data filter?
HMC-SCAN LIMITATIONS

SELECT COLUMN FROM TABLE WHERE COLUMN < 10
HMC-SCAN LIMITATIONS

SELECT COLUMN FROM TABLE WHERE COLUMN < 10 AND COLUMN 1 < 10
HMC-SCAN LIMITATIONS

Branch Execution

A
LOAD bitmap1
CMP  bitmap1
true
B
LOAD p_discount1
CMP  p_discount1
false
C
LOAD bitmap2
CMP  bitmap2
HMC-SCAN LIMITATIONS

Branch Execution

A
LOAD bitmap1
CMP bitmap1
true
LOAD p_discount1
CMP p_discount1
false
LOAD bitmap2
CMP bitmap2

Predicated Execution

A
LOAD bitmap1
CMP bitmap1 → p1
B
LOAD p_discount1
CMP p_discount1 if (p1)
C
LOAD bitmap2
CMP bitmap2
3. What are the extensions to the HMC ISA to leverage the full potential of the hardware and reduce the interleaving with x86 instructions?"
HIPE-FILTER: **HMC INSTRUCTION PREDICATION EXTENSION**

- A comparison instruction between address and immediate
- 32 Vector Functional Units, one per vault
- Bigger operations, with up to 256 bytes
- 36 Registers 256 Bytes wide
- Interleaves x86 with HMC instructions
“SELECT<_COLUMNS> FROM <TABLES> WHERE<CONDITION> AND/OR <CONDITION>”
DATABASE SYSTEMS PREDICATE PROCESSING

Query Plan
- Project
- Selection Scan
- HIPE-SCAN

Processor Core
- Pipeline
  - Load (value)
- Cache Hierarchy
  - Cache line
    - 64 Bytes

HMC

NEAR DATA FILTERS: TAKING ANOTHER BRICK FROM THE MEMORY WALL
DATABASE SYSTEMS PREDICATE PROCESSING

Query Plan

Processor Core

Cache Hierarchy

HMC

256 BYTES OPERATION
DATABASE SYSTEMS PREDICATE PROCESSING

Query Plan

Project

Selection Scan

Pipeline

Load (value)

Cache Hierarchy

Cache line

64 Bytes

16 Bytes Operation

HMC

HIPE-FILTER

STORE
DATABASE SYSTEMS PREDICATE PROCESSING

Query Plan

Project
Selection Scan

Processor Core

Pipeline
Load (value)

Cache Hierarchy
Cache line 64 Bytes

16 Bytes Operation

RETURN STATUS

BYPASS CACHE HIERARCHY

HMC
HIPE-FILTER

NEAR DATA FILTERS: TAKING ANOTHER BRICK FROM THE MEMORY WALL
DATABASE SYSTEMS PREDICATE PROCESSING

DATA-MOVEMENT

Query Plan

Processor Core

HMC

16 BYTES REQUEST -> 64 BYTES REQUEST

64 BYTES REQUEST

256 BYTES AVAILABLE

64 BYTES RESPONSE

16 BYTES RESPONSE
DATABASE SYSTEMS PREDICATE PROCESSING

NEAR-MEMORY OPERATIONS

Query Plan

Processor Core

HMC

Selection Scan

Pipeline

Load (value)

BYPASS CACHE HIERARCHY

Cache Hierarchy

Cache line
64 Bytes

16 Bytes Operation

RETURN STATUS

HIPE-SCAN

HMC INSTRUCTION

256 BYTES OPERATION

NEAR-MEMORY OPERATIONS: TAKING ANOTHER BRICK FROM THE MEMORY WALL
NEAR-DATA FILTERS BENEFITS

- Reduce cache pollution
- No Cache latency
- Mitigate DRAM operations (signals)
NEAR DATA FILTERS: TAKING ANOTHER BRICK FROM THE MEMORY WALL

EXPERIMENTS
METHODOLOGY AND SETUP

- Using a cycle-accurate simulator to evaluate
- Dataset 1GB from TPC-H for OLAP scenario
- Micro-benchmark TPC-H Query 06
Figure 10: Evaluating execution time of TPC-H Q6 varying the selectivity factor in the different hardware architectures.

TPC-H query 06 Select scan - Performance
Figure 10: Evaluating execution time of TPC-H Q6 varying the selectivity factor in the different hardware architectures.

TPC-H query 06 Select scan - Performance
SELECTIVE LOAD FILTER IS 3X MORE EFFICIENT IN ENERGY CONSUMPTION

TPC-H query 06 Select scan - Energy
Figure 13: Execution time using the HIPE-Filters in the column projection varying the loop unroll depth.

TPC-H query 06 Projection
Figure 12: Execution time using the HIPE-Filters in the Nested-Loop join varying the loop unroll depth.
1. What happens when database systems run the select scan over the current x86 architecture using the HMC as ordinary DRAM? The memory wall is still a problem.

2. Can we use the current HMC Instruction Set Architecture (ISA) to implement the near-data select scan? Yes, but it is not suitable to mitigate the memory wall.

3. What are the extensions to the HMC ISA to leverage the full potential of the hardware to mitigate the memory wall and reduce the interleaving with x86 instructions? Branch-less near data filters.
NEAR DATA FILTERS: TAKING ANOTHER BRICK FROM THE MEMORY WALL

CONCLUSION
CONCLUSION

- With the usage of the HMC as ordinary DRAM the memory wall is still a problem.
- Performing filtering operations inside the HMC improves performance for OLAP.
- Near-data processing is suitable for OLAP mitigating the memory wall for DBMSs.
- The correct filter algorithm on different selectivity factors may have great impact on performance.