Power-Efficient Breadth-First Search with DRAM Row Buffer Locality-Aware Address Mapping

Satoshi Imamura*, Yuichiro Yasui*, Koji Inoue*,
Takatsugu Ono*, Hiroshi Sasaki**, Katsuki Fujisawa*

*Kyushu University, **Columbia University

HPGDMP ’16, Salt Lake City, November 13, 2016

Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.

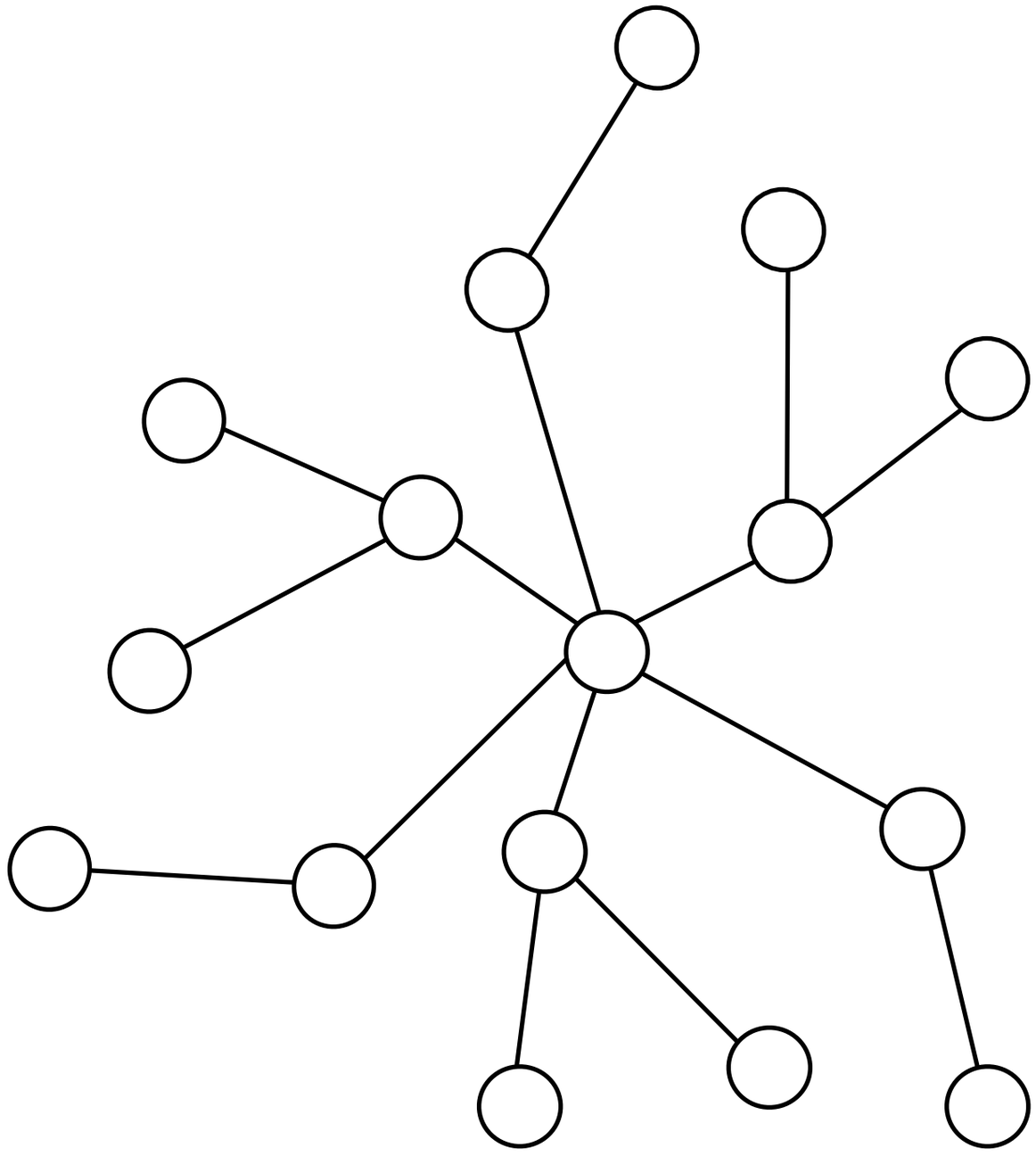
Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.

- Graphs
  - Consist of vertices and edges
  - Can represent relationships between vertices

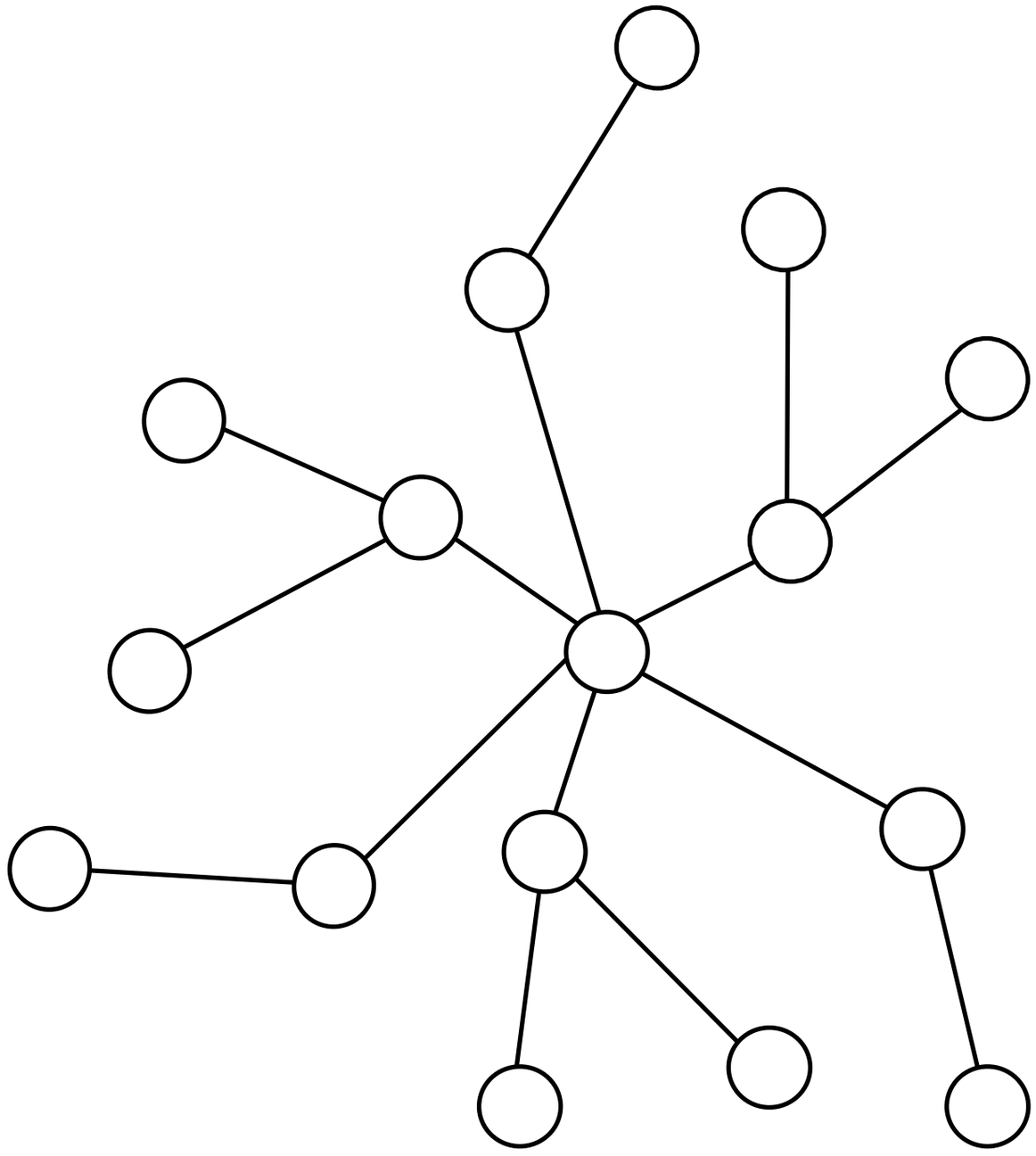
Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.
- Graphs
  - Consist of vertices and edges
  - Can represent relationships between vertices



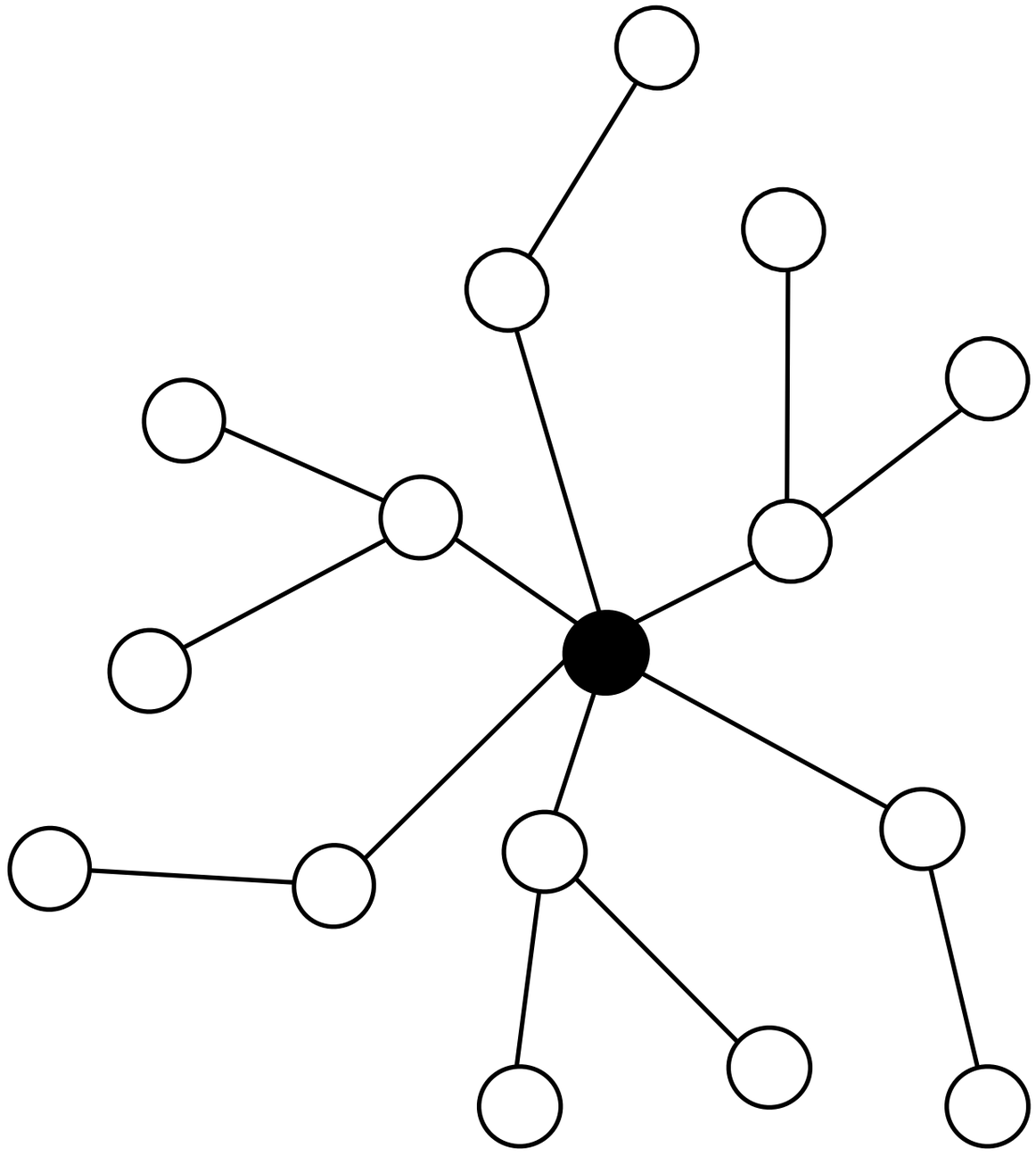
Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.
- Graphs
  - Consist of vertices and edges
  - Can represent relationships between vertices
- Breadth-first search (BFS)
  - Many graph algorithms are based on BFS



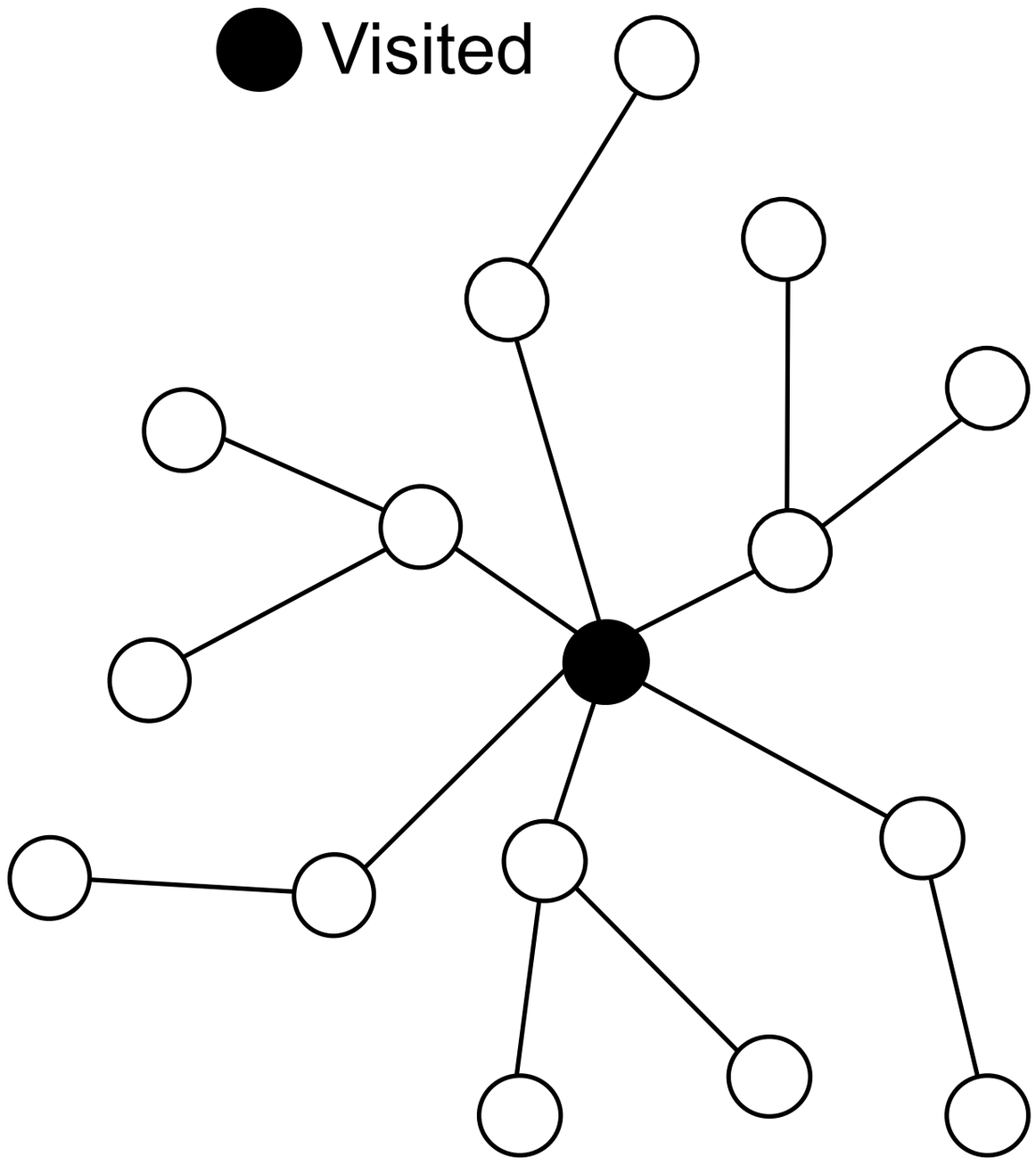
Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.
- Graphs
  - Consist of vertices and edges
  - Can represent relationships between vertices
- Breadth-first search (BFS)
  - Many graph algorithms are based on BFS



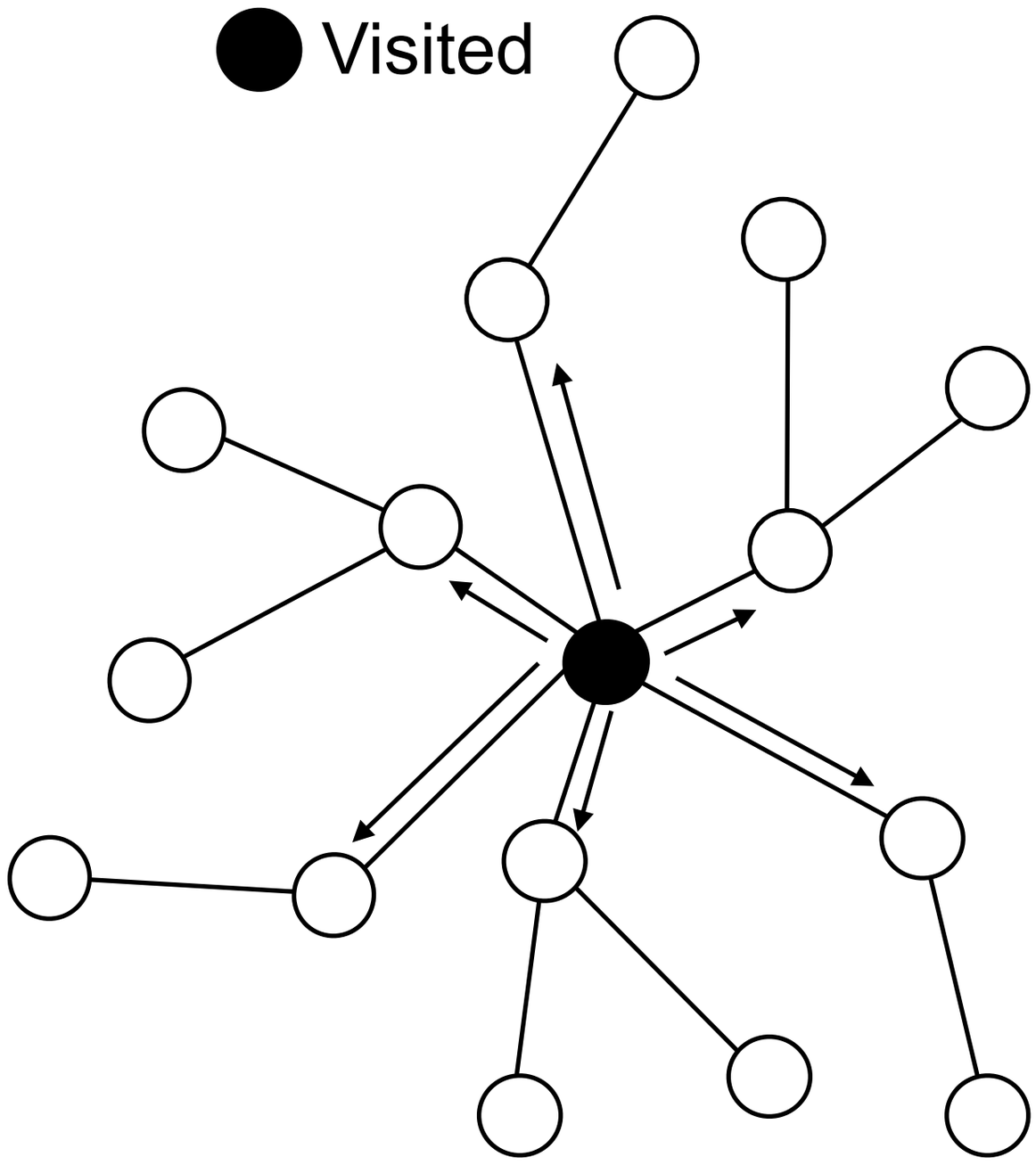
Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.
- Graphs
  - Consist of vertices and edges
  - Can represent relationships between vertices
- Breadth-first search (BFS)
  - Many graph algorithms are based on BFS



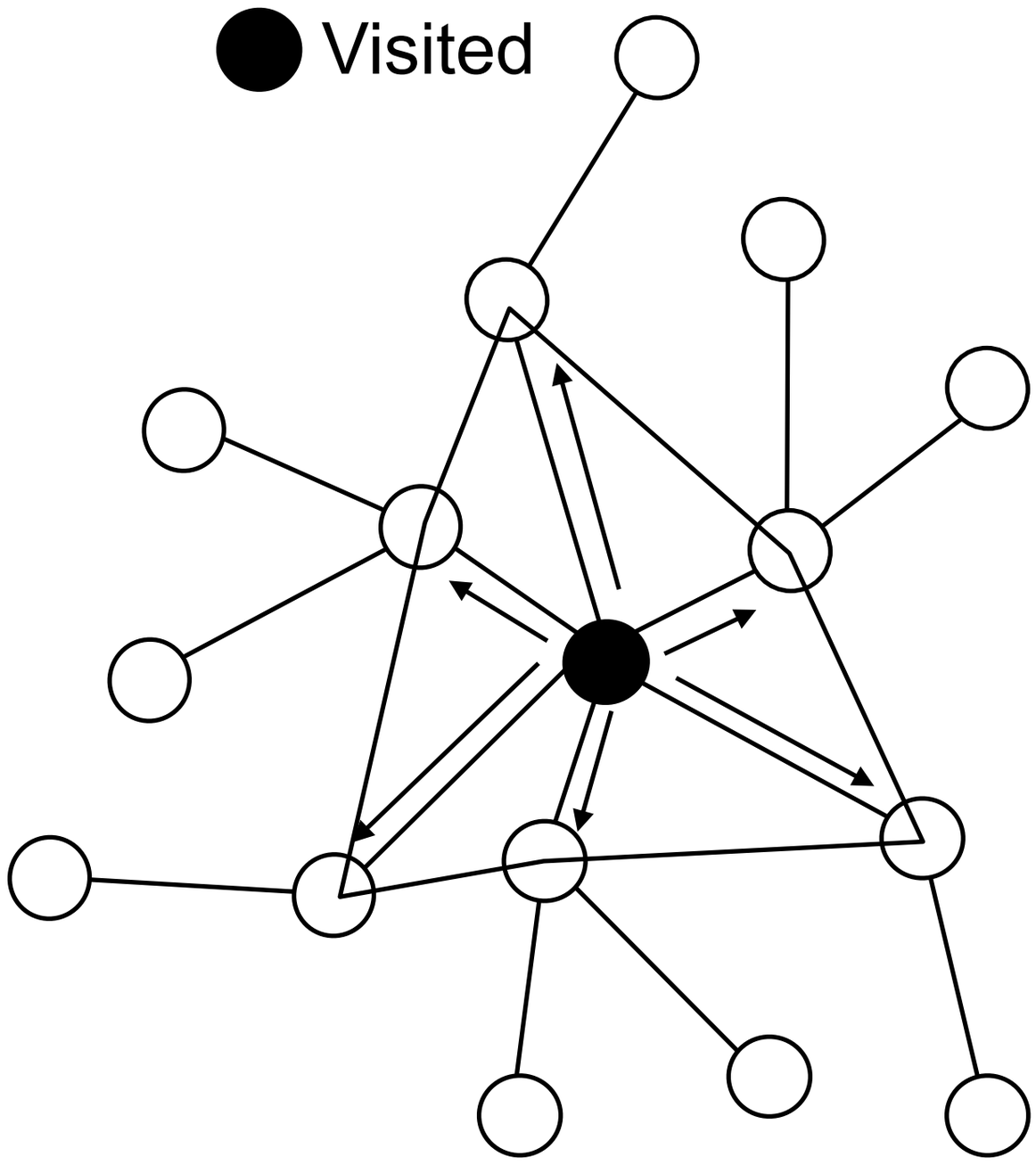
Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.
- Graphs
  - Consist of vertices and edges
  - Can represent relationships between vertices
- Breadth-first search (BFS)
  - Many graph algorithms are based on BFS



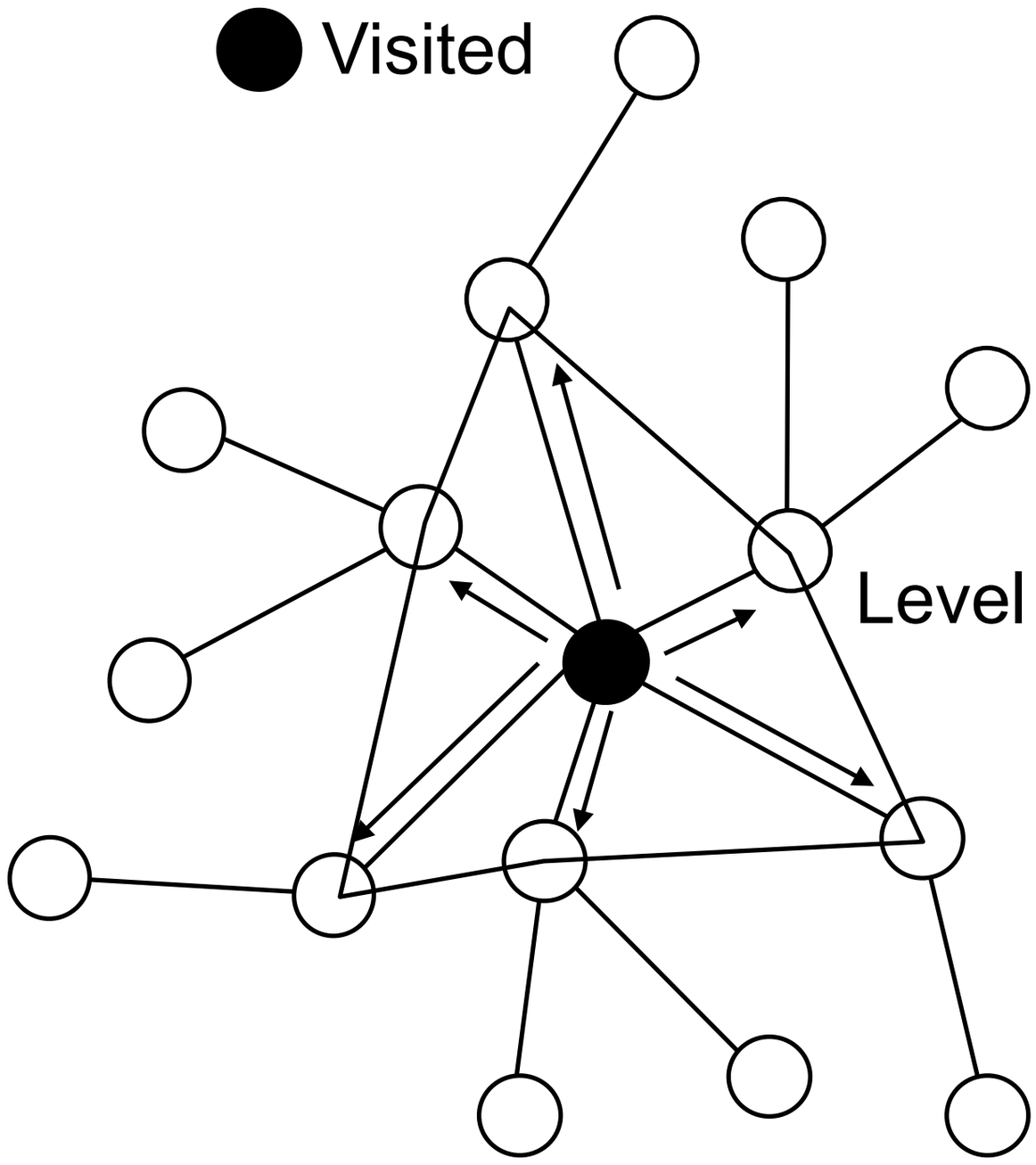
Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.
- Graphs
  - Consist of vertices and edges
  - Can represent relationships between vertices
- Breadth-first search (BFS)
  - Many graph algorithms are based on BFS



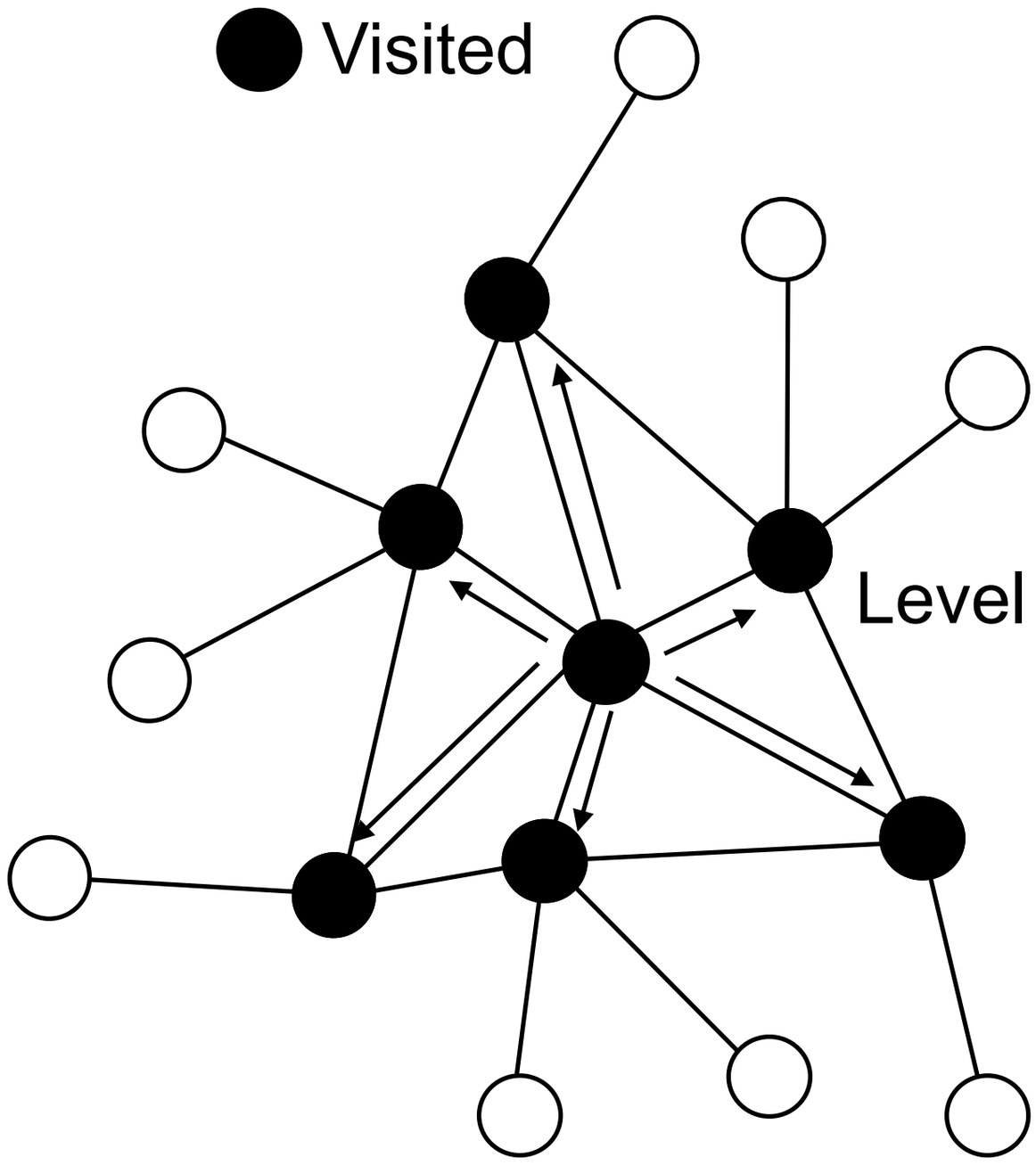
Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.
- Graphs
  - Consist of vertices and edges
  - Can represent relationships between vertices
- Breadth-first search (BFS)
  - Many graph algorithms are based on BFS



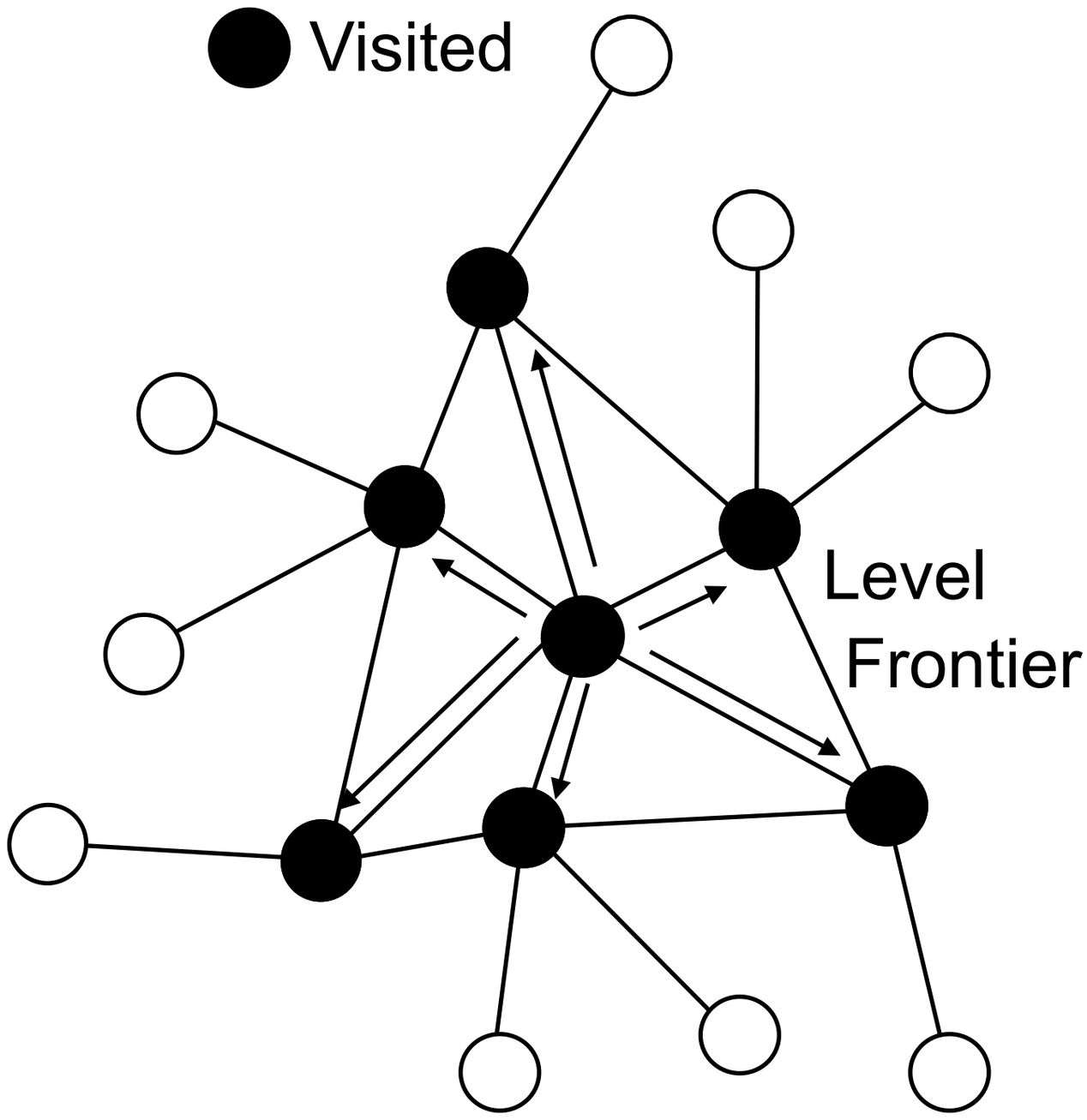
Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.
- Graphs
  - Consist of vertices and edges
  - Can represent relationships between vertices
- Breadth-first search (BFS)
  - Many graph algorithms are based on BFS



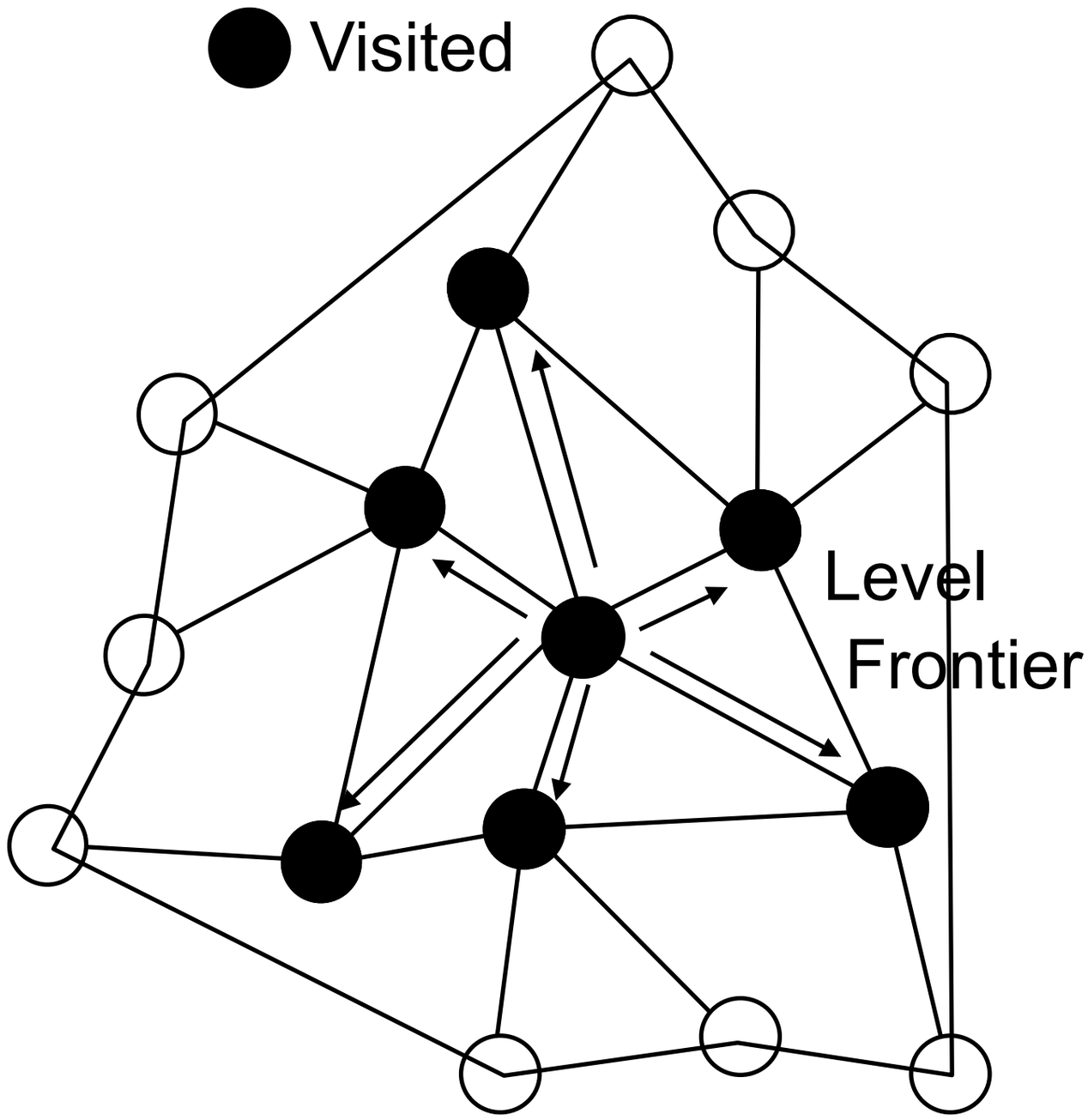
Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.
- Graphs
  - Consist of vertices and edges
  - Can represent relationships between vertices
- Breadth-first search (BFS)
  - Many graph algorithms are based on BFS



Graph Analysis & BFS

- Graph analysis can be used to various real services
  - Social networking services
  - Road traffic analysis, etc.
- Graphs
  - Consist of vertices and edges
  - Can represent relationships between vertices
- Breadth-first search (BFS)
  - Many graph algorithms are based on BFS



Power Problem on Servers

- Performance has been improved as the amount of hardware resources is increased
- Power consumption also continues to grow
  - Proportional to # of active transistors
- One of the most critical design constraints

Summary of This Work

- Objective: to improve the power efficiency (i.e., performance per watt) of a state-of-the-art BFS implementation
- Contributions:

Summary of This Work

- Objective: to improve the power efficiency (i.e., performance per watt) of a state-of-the-art BFS implementation
- Contributions:
  - Investigate the memory access pattern of its main algorithm using a simulator

Summary of This Work

- Objective: to improve the power efficiency (i.e., performance per watt) of a state-of-the-art BFS implementation
- Contributions:
  - Investigate the memory access pattern of its main algorithm using a simulator
  - Reveal that conventional address mapping schemes of memory controllers do NOT efficiently exploit DRAM

Summary of This Work

- Objective: to improve the power efficiency (i.e., performance per watt) of a state-of-the-art BFS implementation
- Contributions:
  - Investigate the memory access pattern of its main algorithm using a simulator
  - Reveal that conventional address mapping schemes of memory controllers do NOT efficiently exploit DRAM
  - Propose a novel scheme and improve DRAM power efficiency by 30.3%

Agenda

- State-of-the-art BFS implementation
- DRAM mechanisms
- Memory access analysis with conventional address mapping schemes
- Proposed: per-row channel interleaving
- Evaluation of power efficiency

Agenda

- State-of-the-art BFS implementation
- DRAM mechanisms
- Memory access analysis with conventional address mapping schemes
- Proposed: per-row channel interleaving
- Evaluation of power efficiency

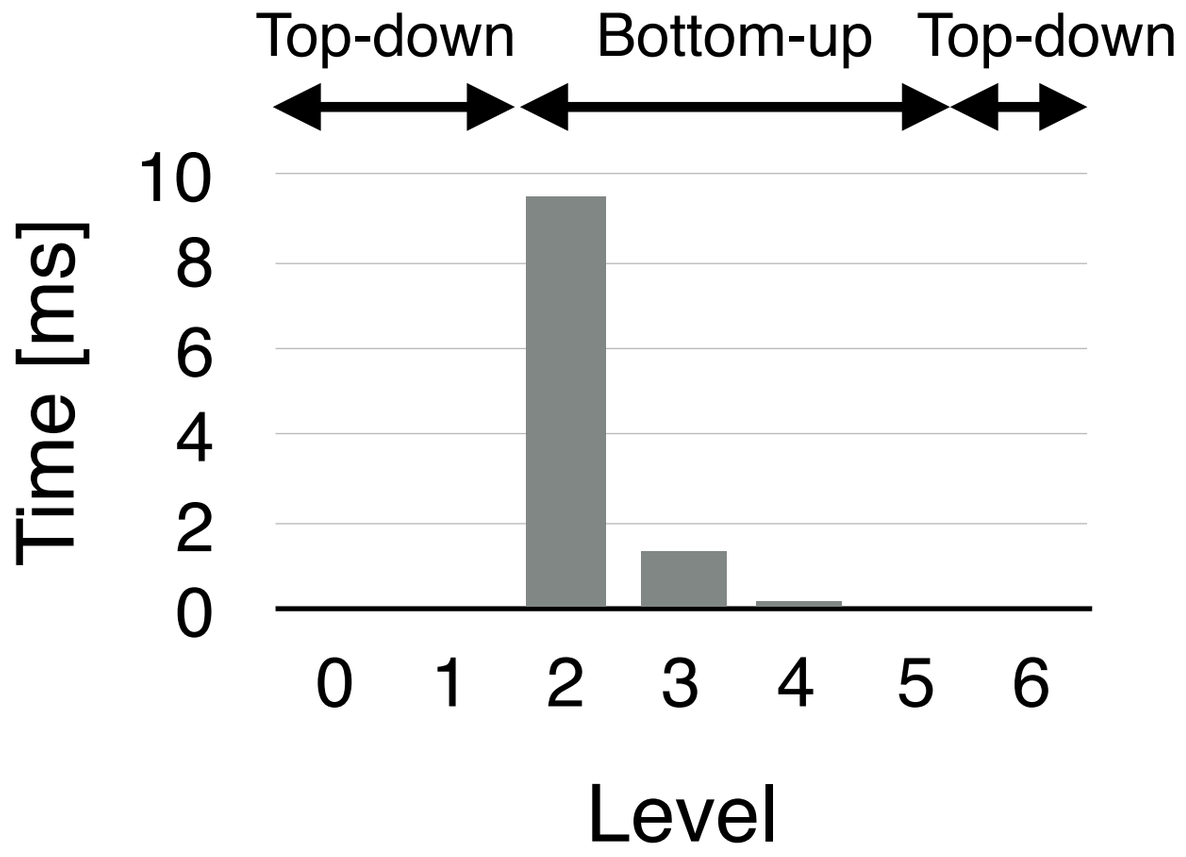
Yasui16 Implementation

[Yasui+, HPGP'16]

- Achieved the best performance for a single-node system in the June 2016 Graph500 list

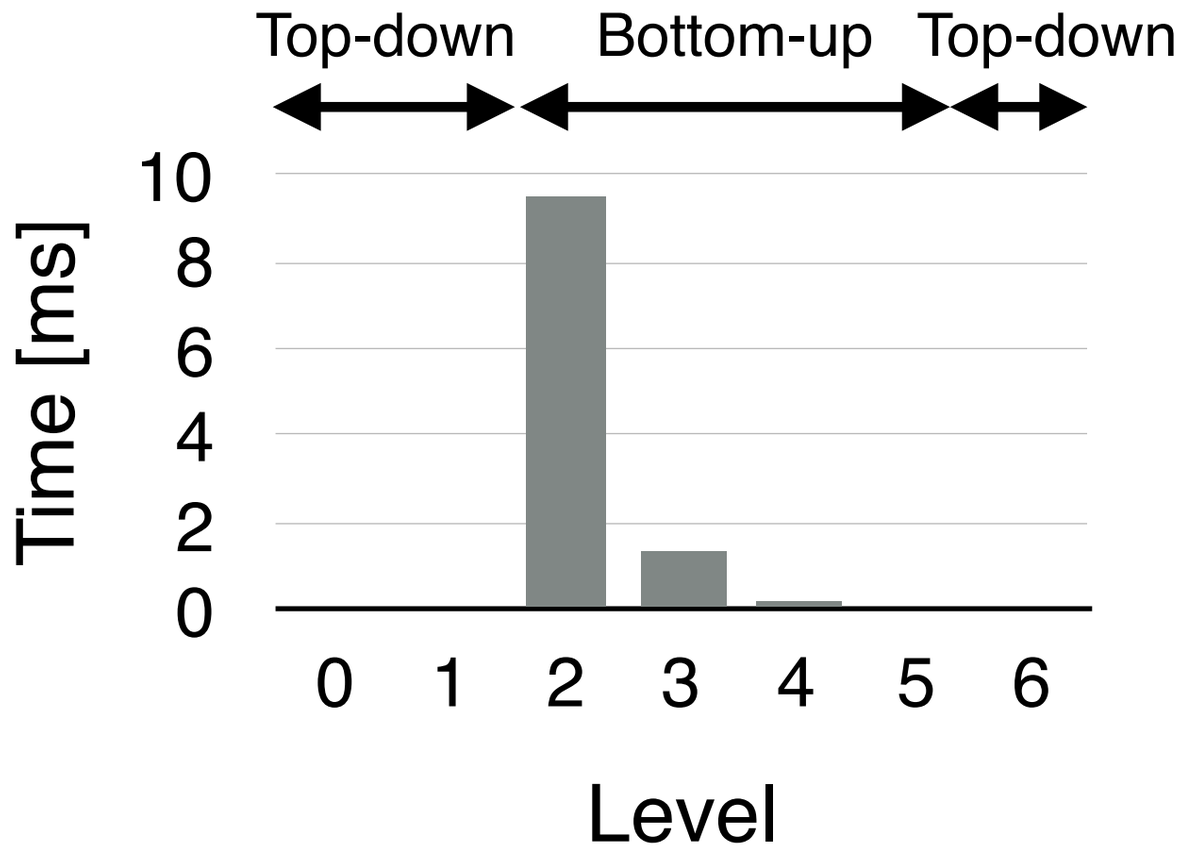
- Applies several tuning techniques
  - NUMA-aware data layout [Yasui+, BigData'13]
  - Adjacency list and vertex sorting [Yasui+, HPCS'15]
  - Direction optimizing [Beamer+, SC'12]
    - ✓ Significantly reduces # of edge traversals by switching two algorithms at each level: ***top-down*** or ***bottom-up***

Time Breakdown of BFS



BFS is executed with scale 22 on 10-core Haswell machine

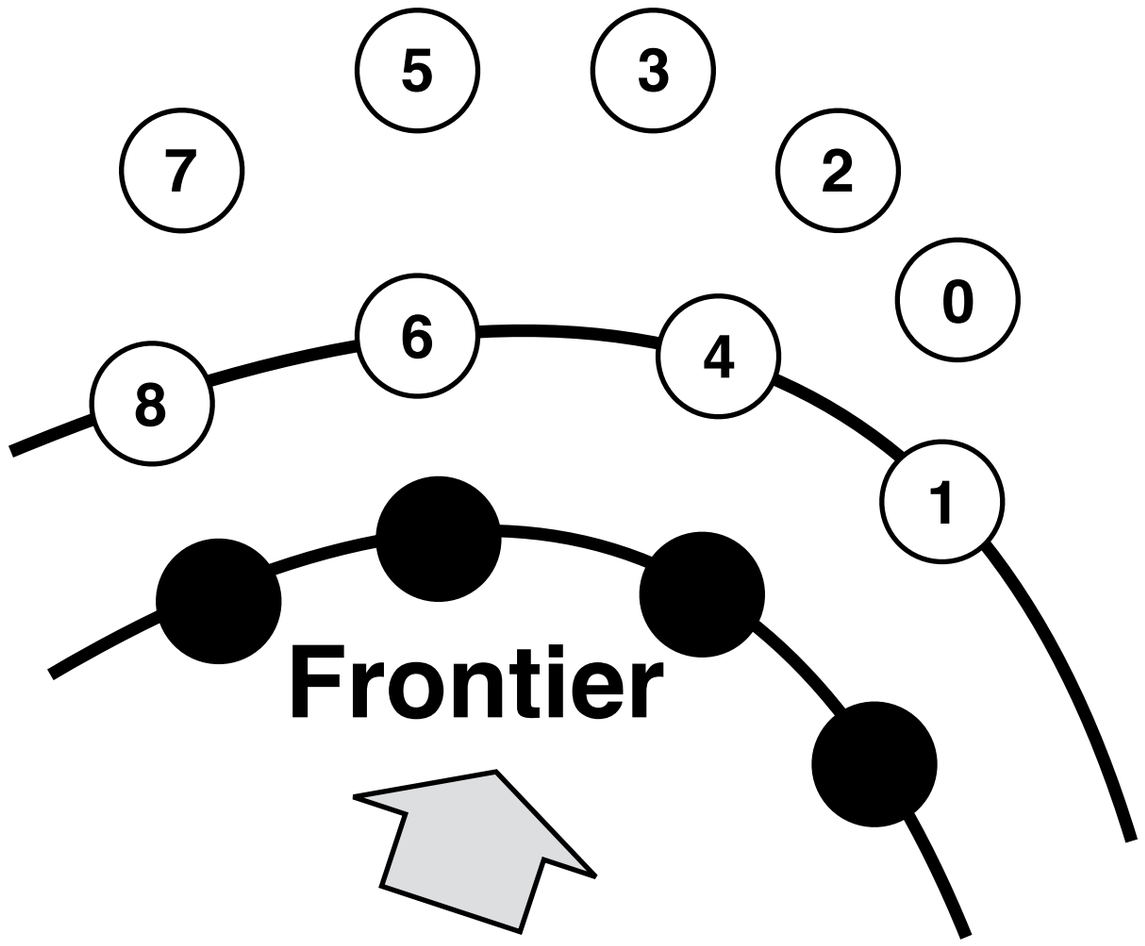
Time Breakdown of BFS



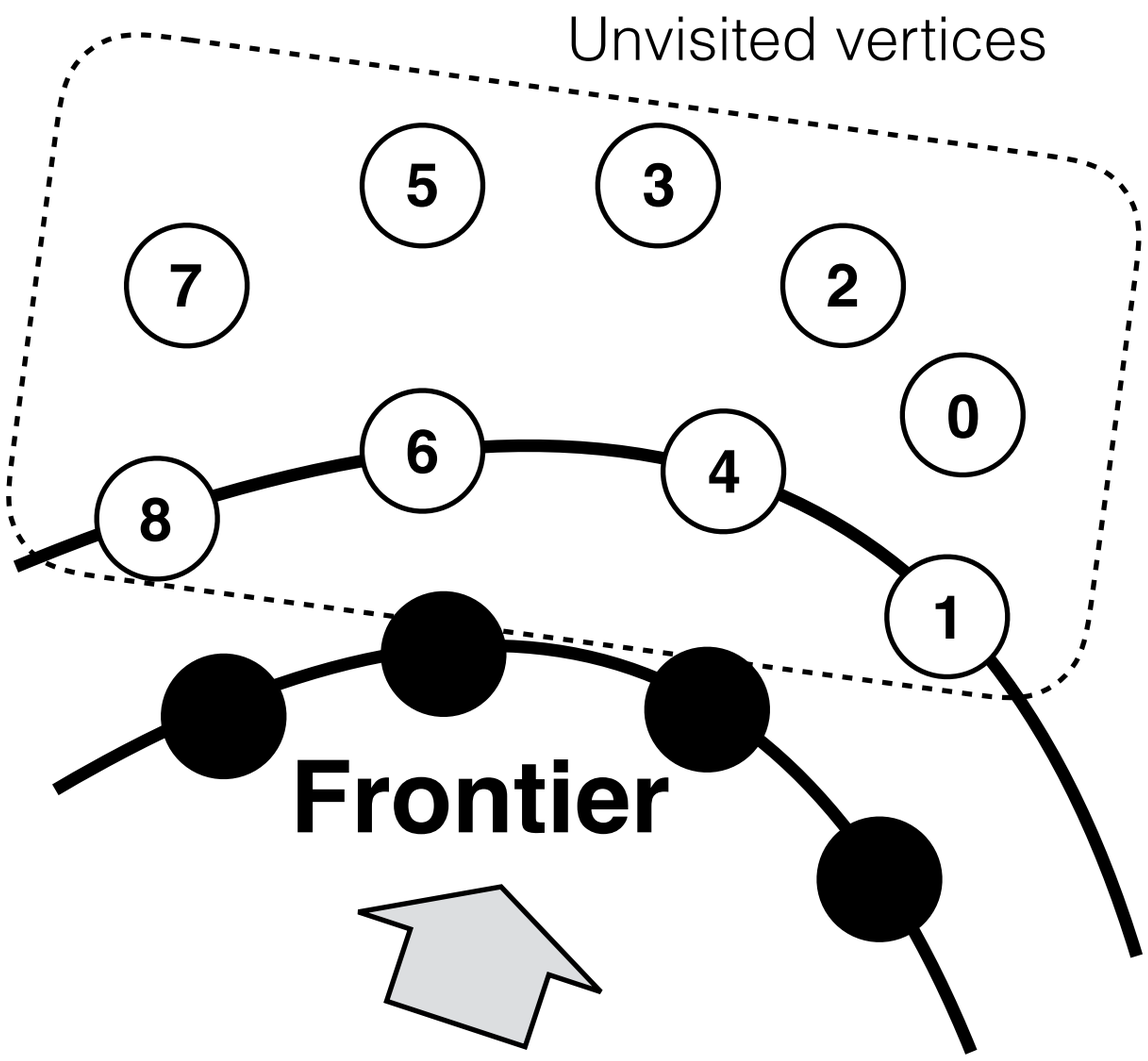
BFS is executed with scale 22 on 10-core Haswell machine

- Over 99% of exe. time is spent by bottom-up algorithm
- We aim to improve the power efficiency of bottom-up

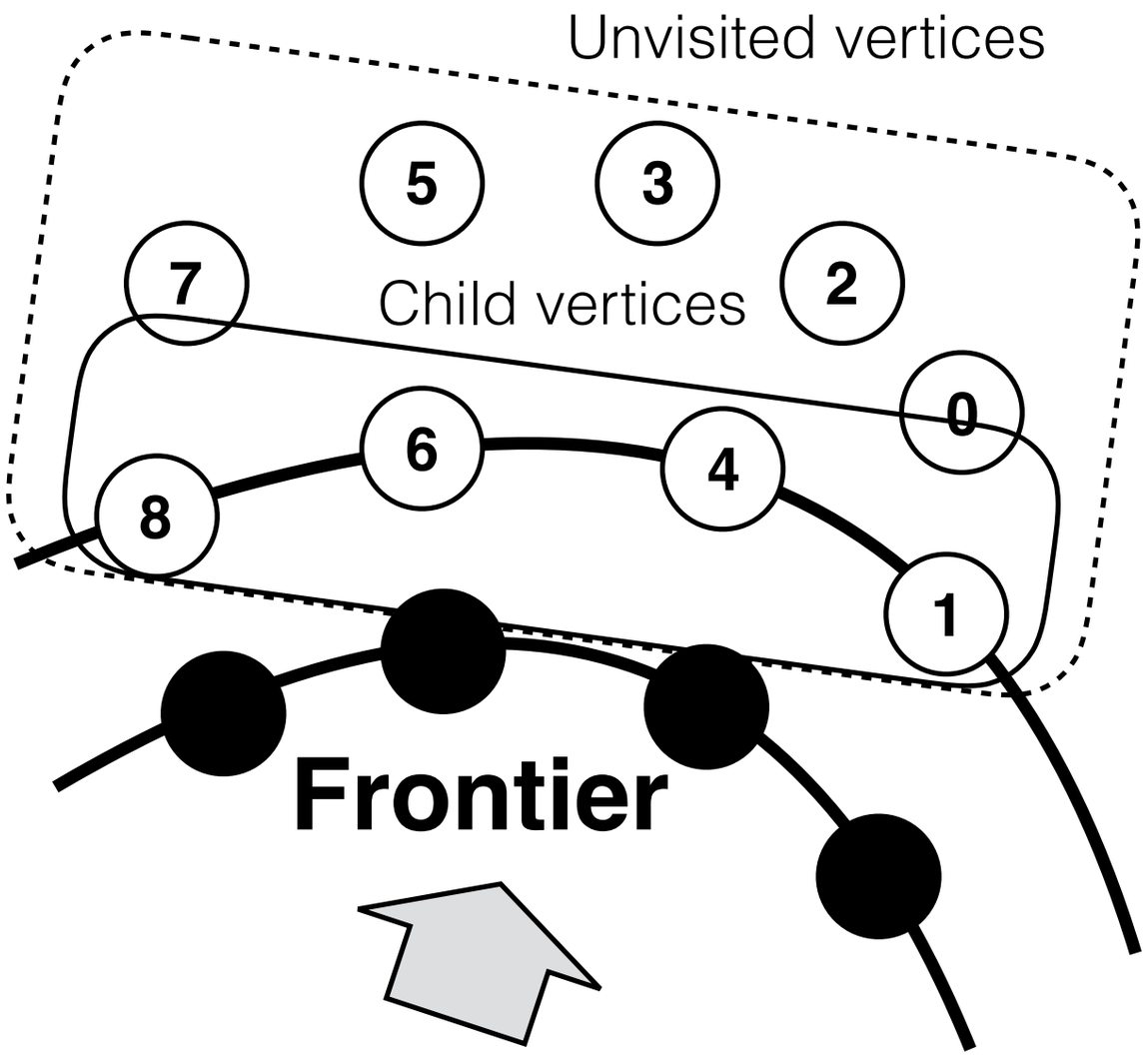
Bottom-up Algorithm



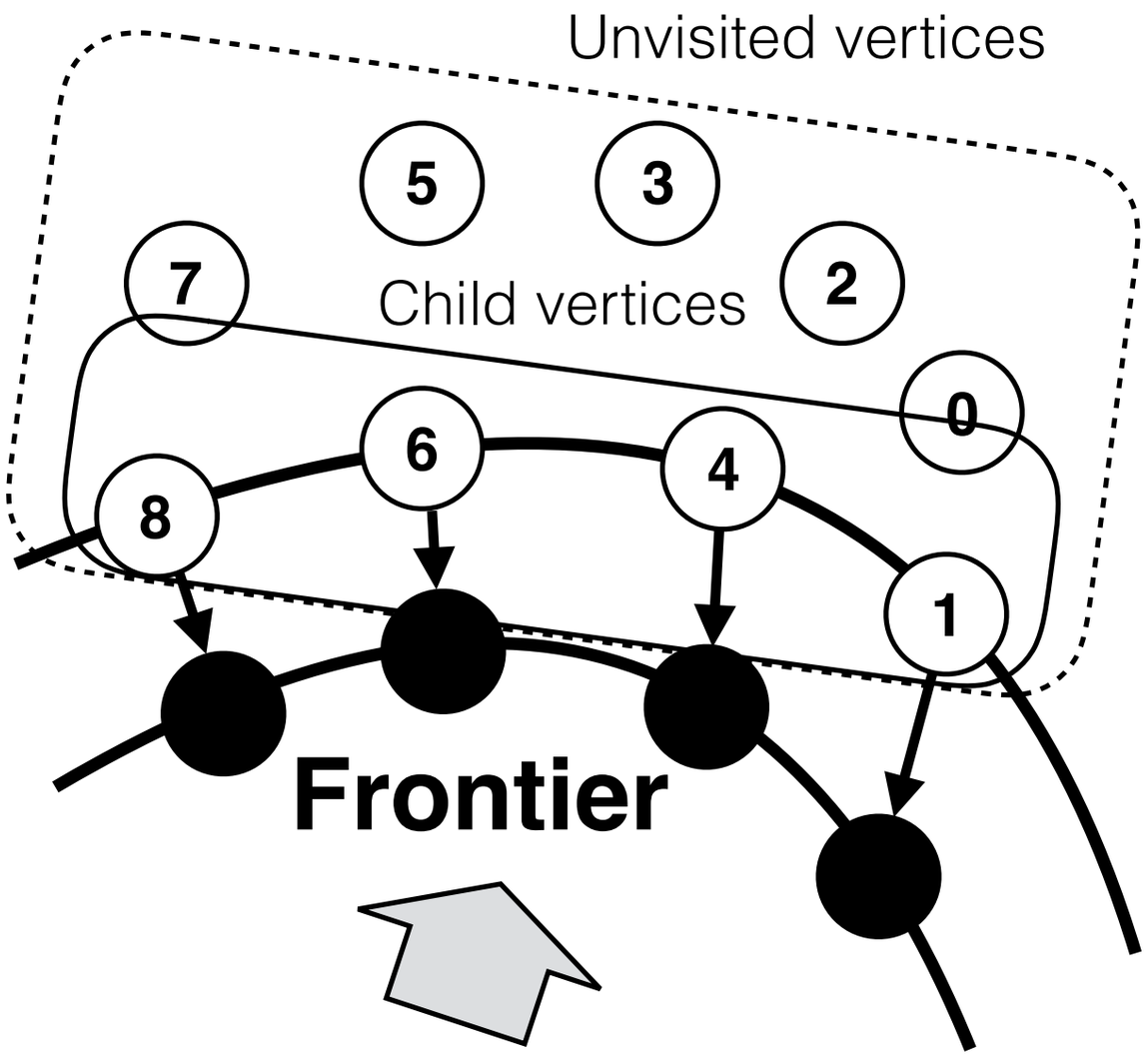
Bottom-up Algorithm



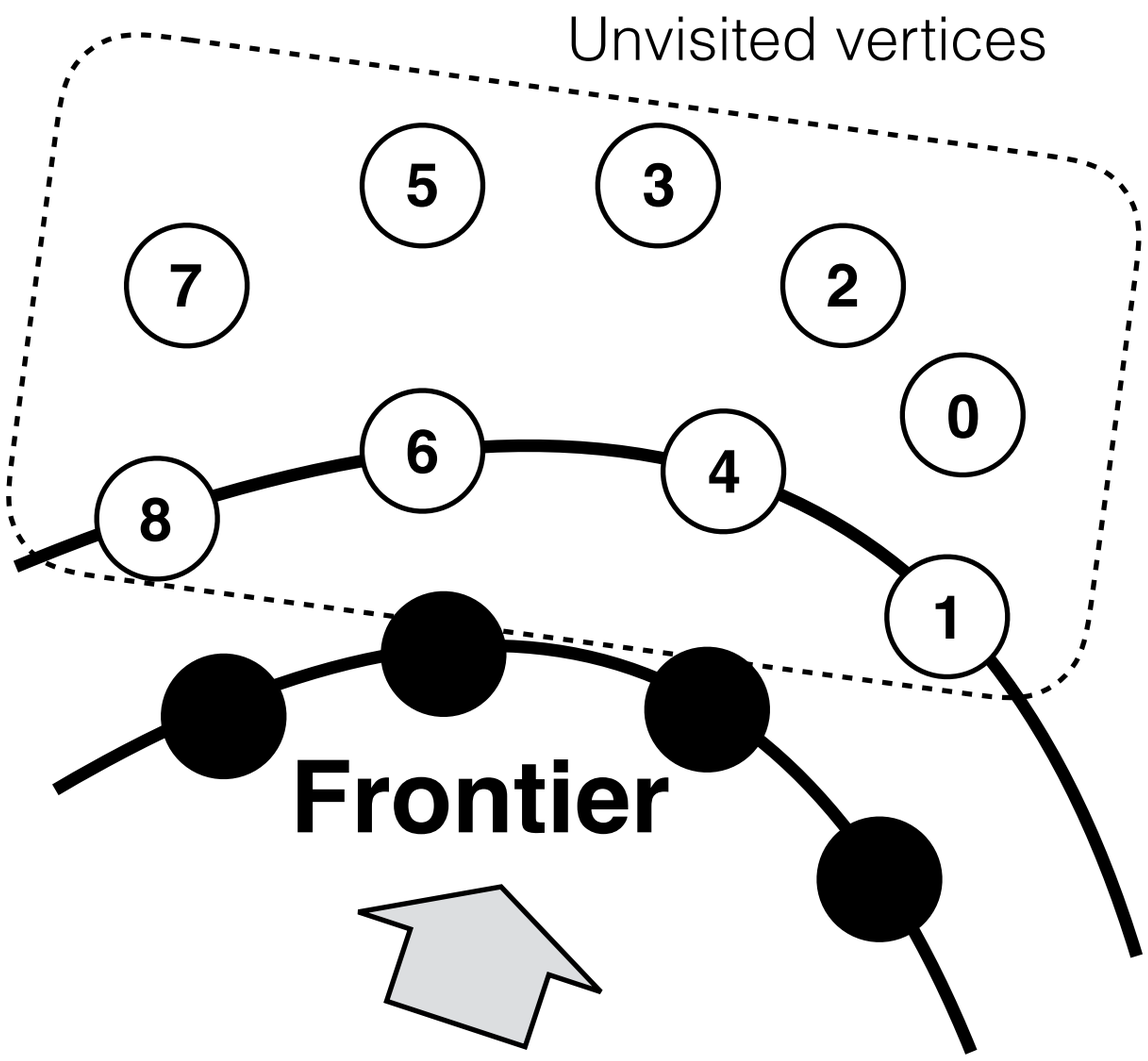
Bottom-up Algorithm



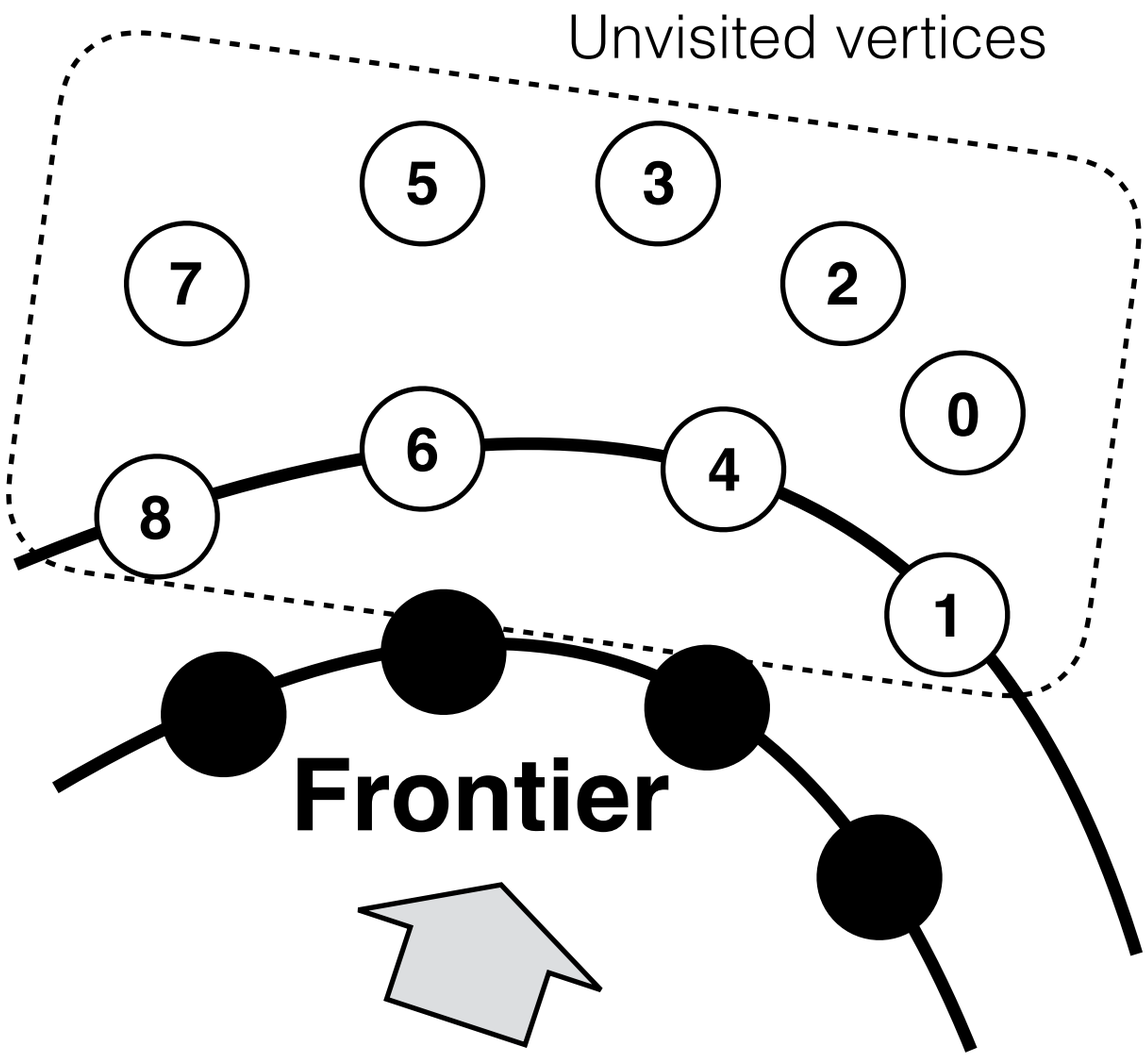
Bottom-up Algorithm



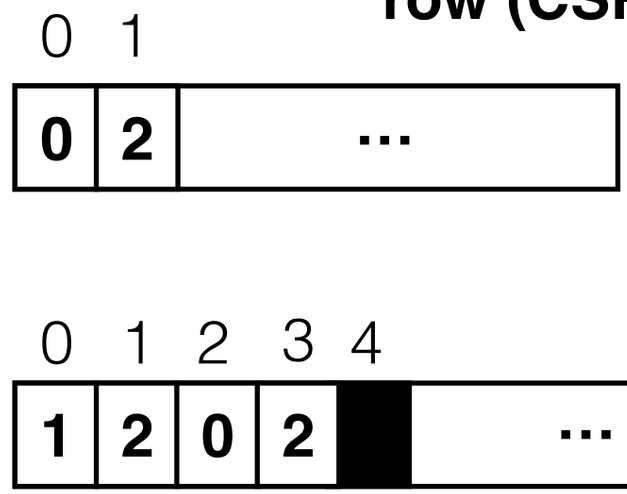
Bottom-up Algorithm



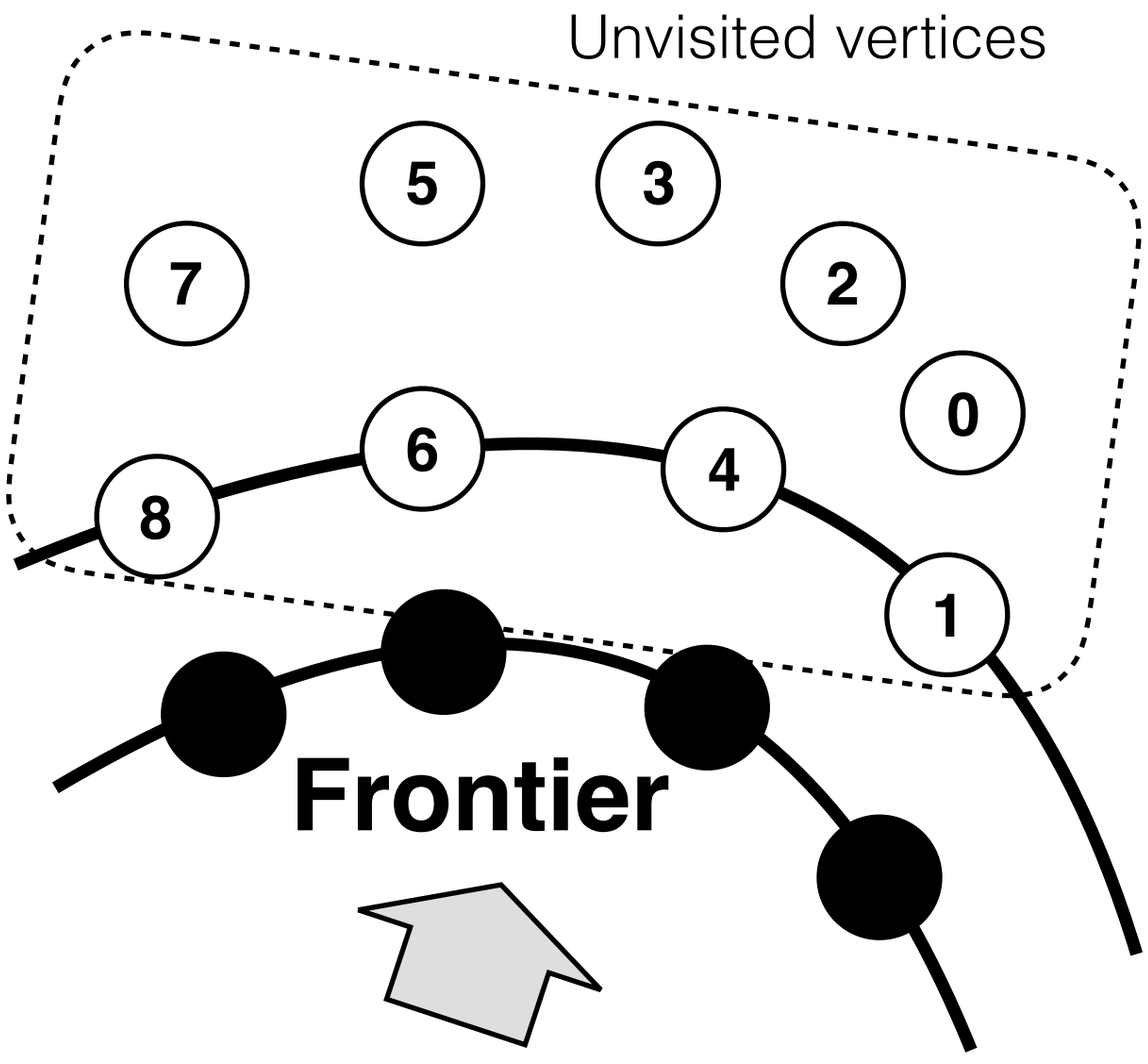
Bottom-up Algorithm



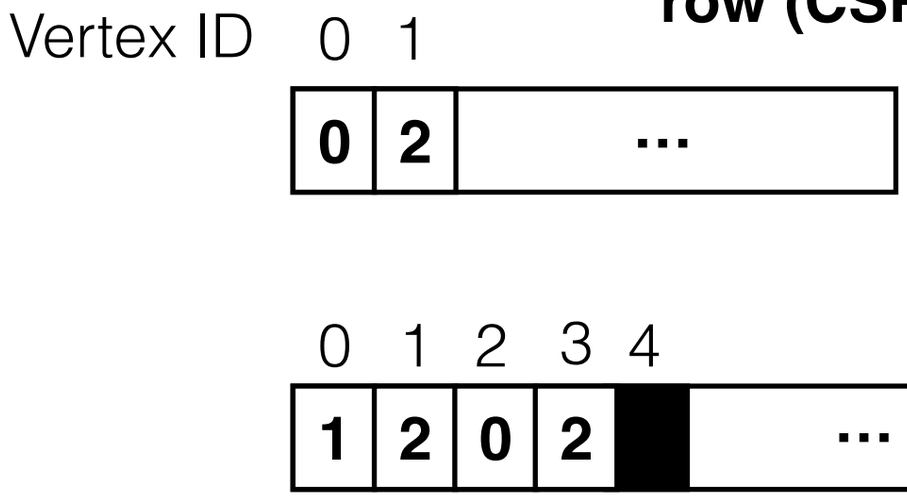
Compressed sparse row (CSR)



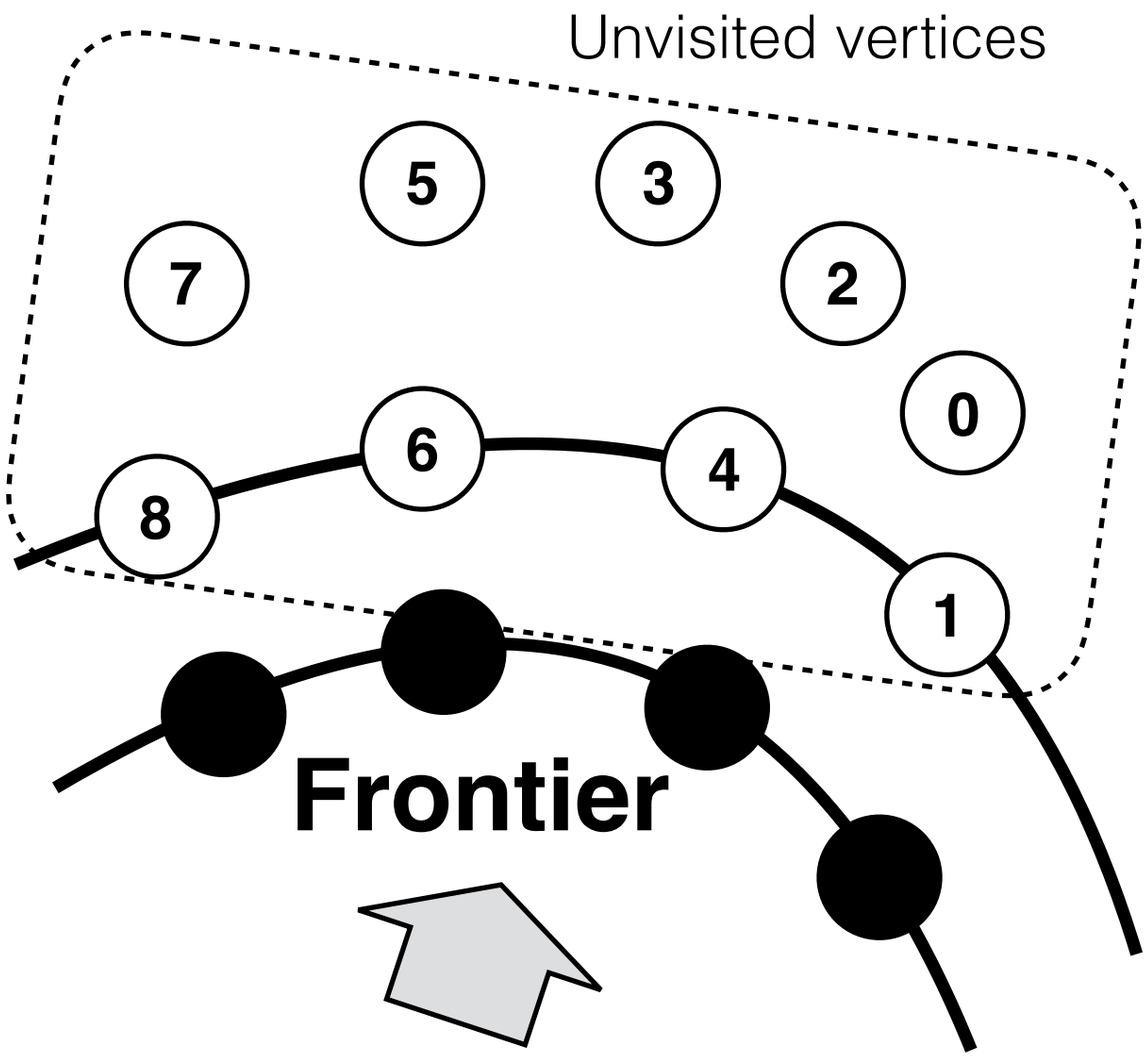
Bottom-up Algorithm



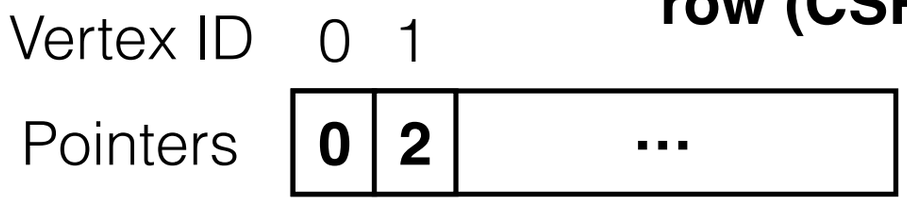
Compressed sparse row (CSR)

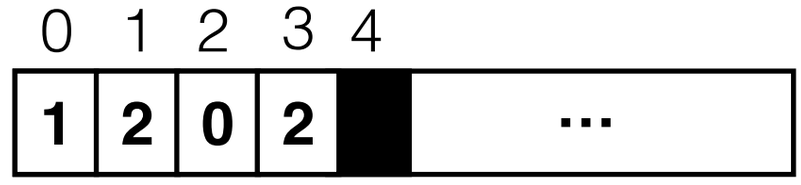


Bottom-up Algorithm

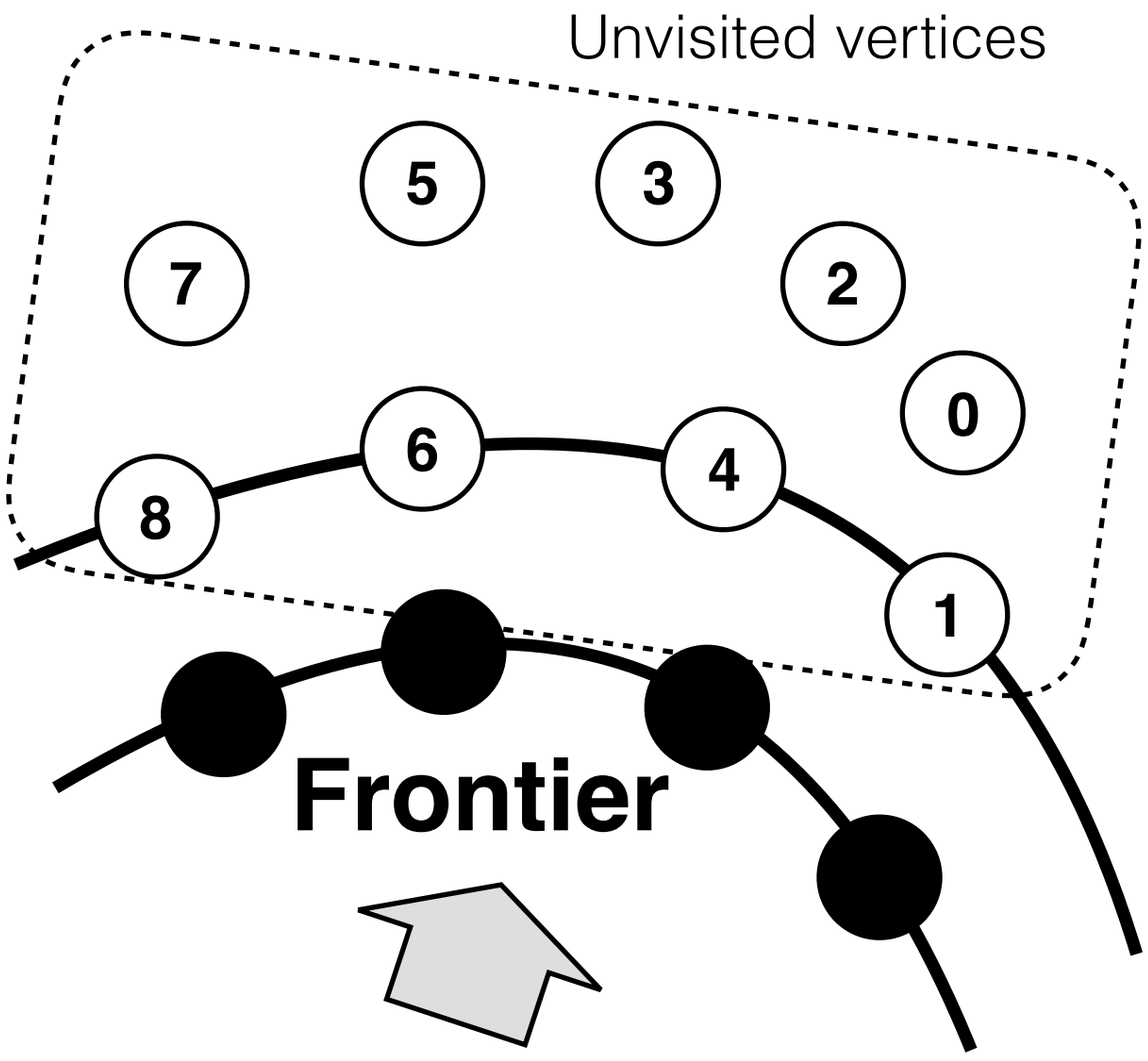


Compressed sparse row (CSR)

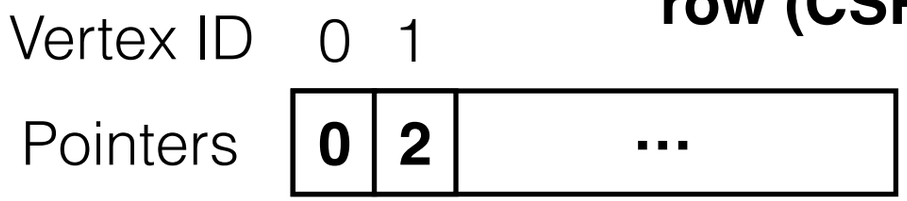


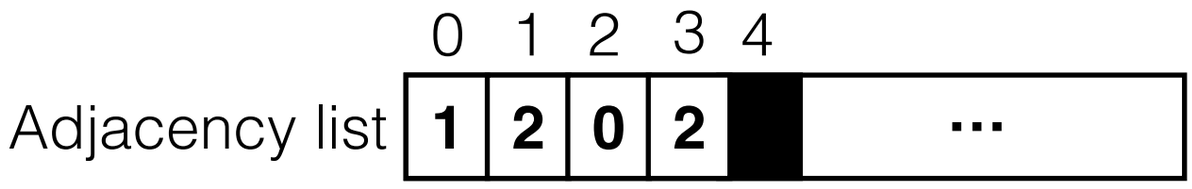


Bottom-up Algorithm

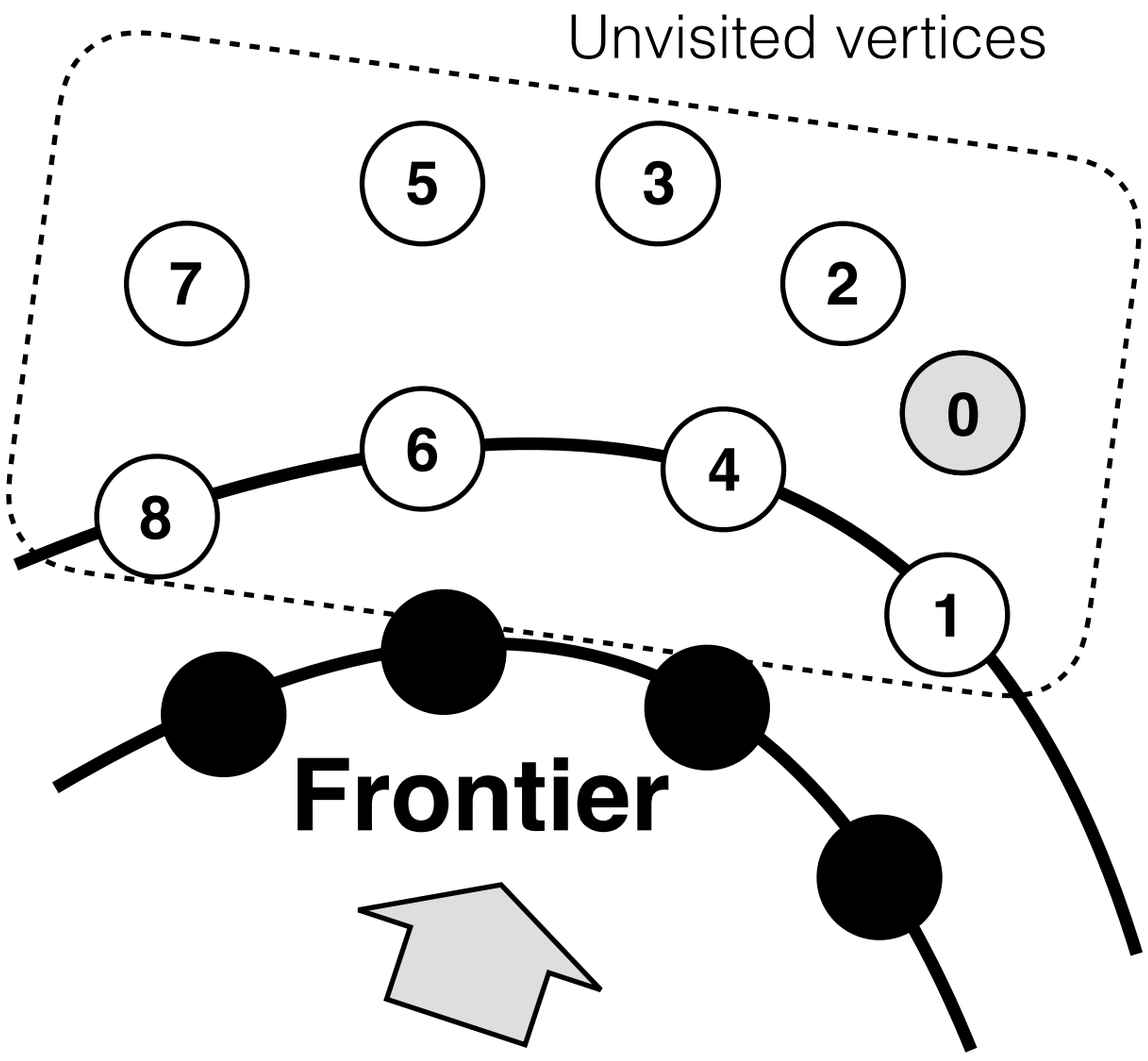


Compressed sparse row (CSR)

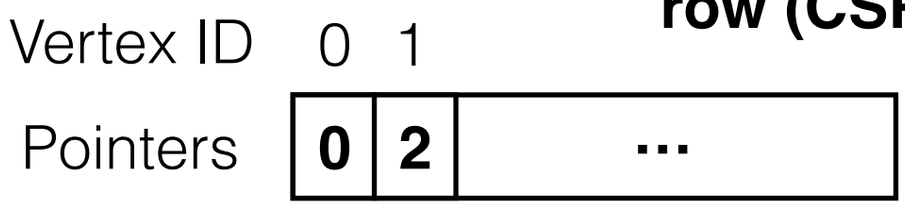


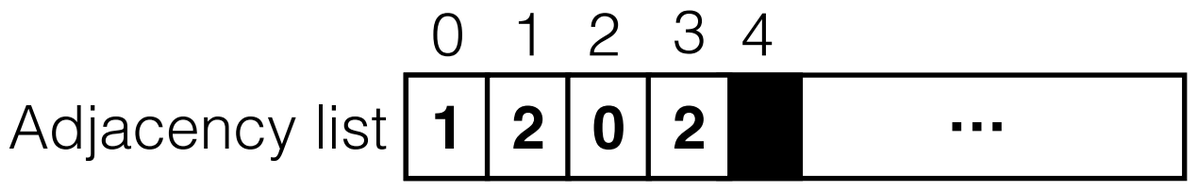


Bottom-up Algorithm

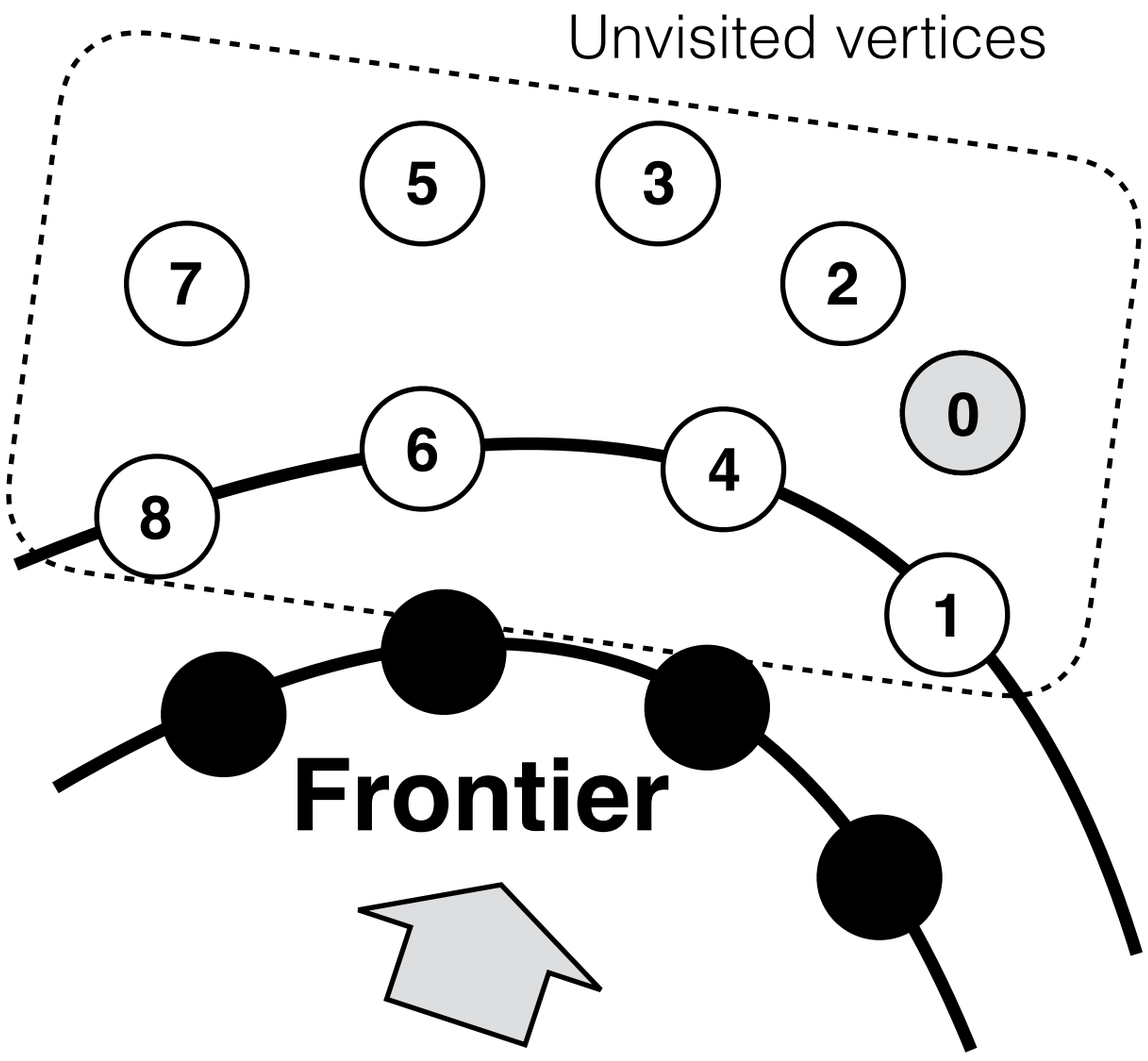


Compressed sparse row (CSR)

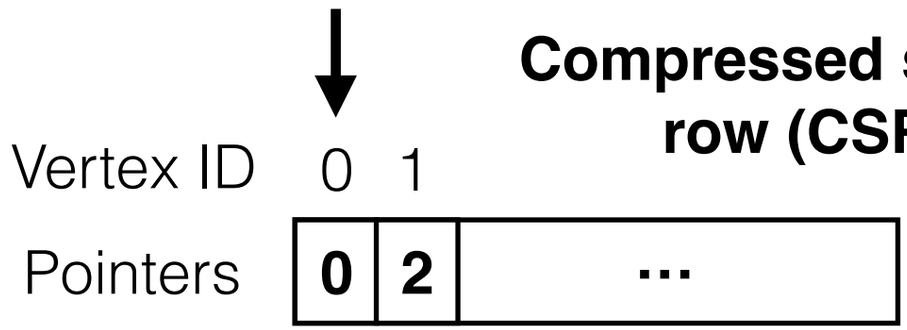


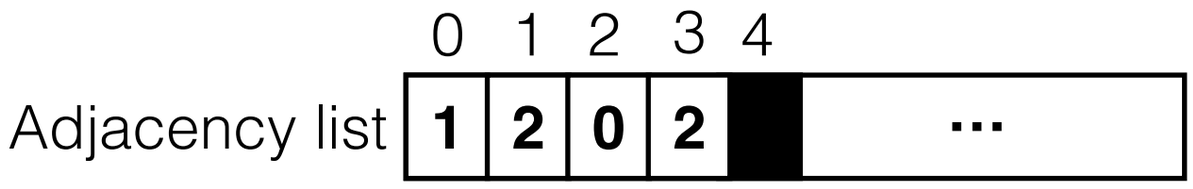


Bottom-up Algorithm

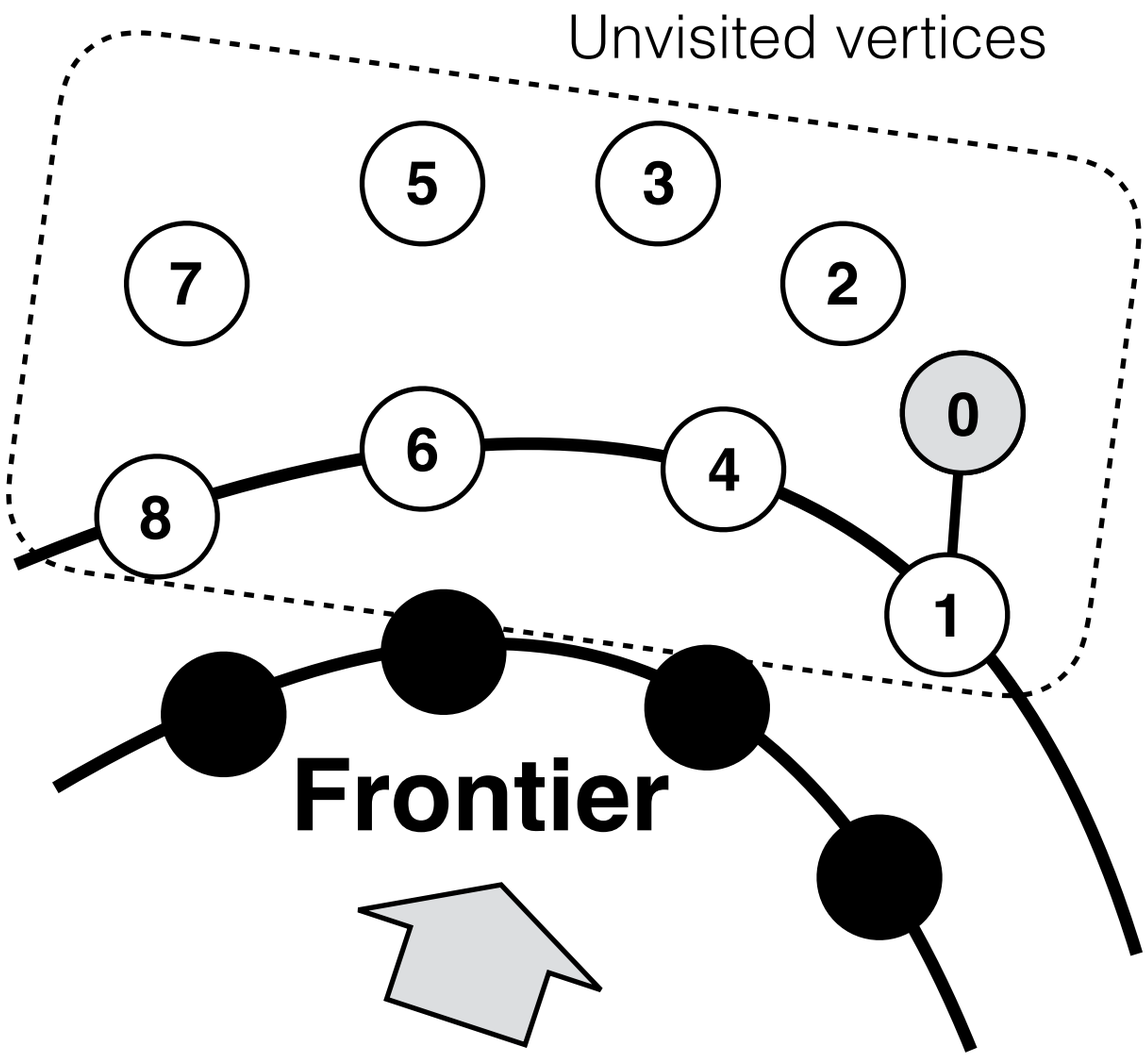


Compressed sparse row (CSR)

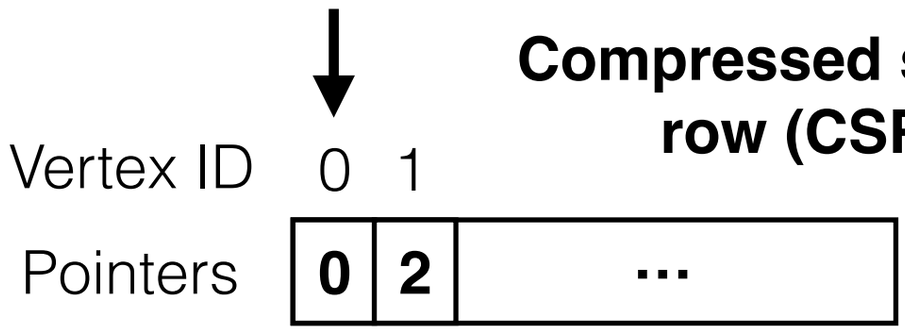


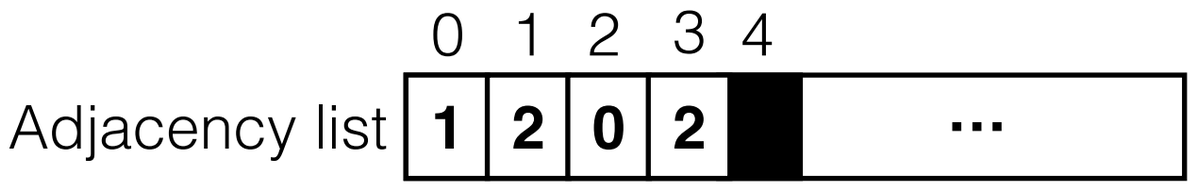


Bottom-up Algorithm

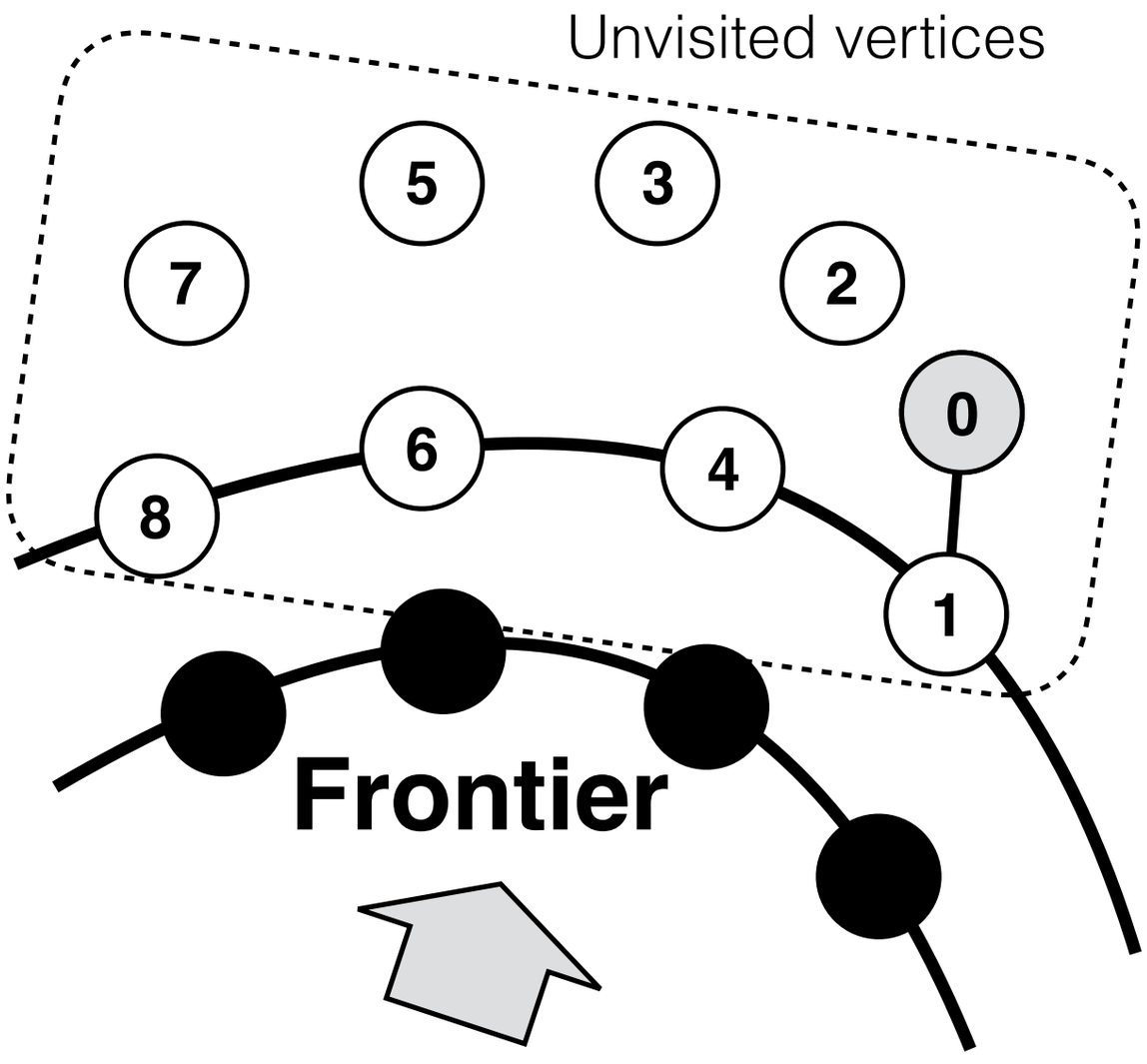


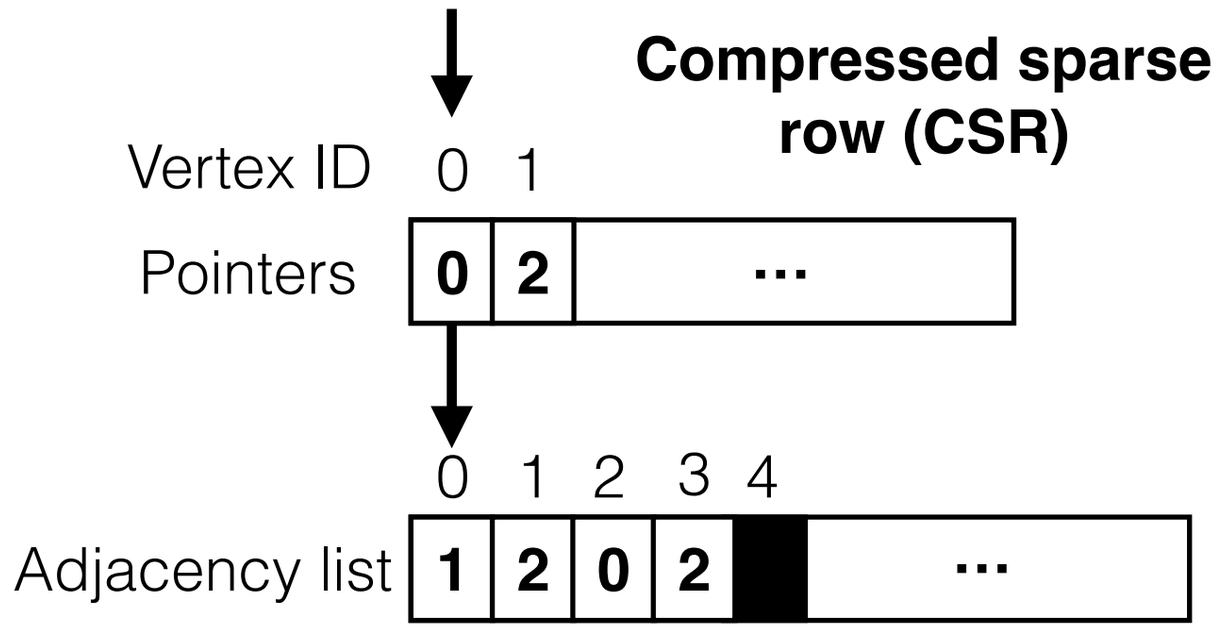
Compressed sparse row (CSR)



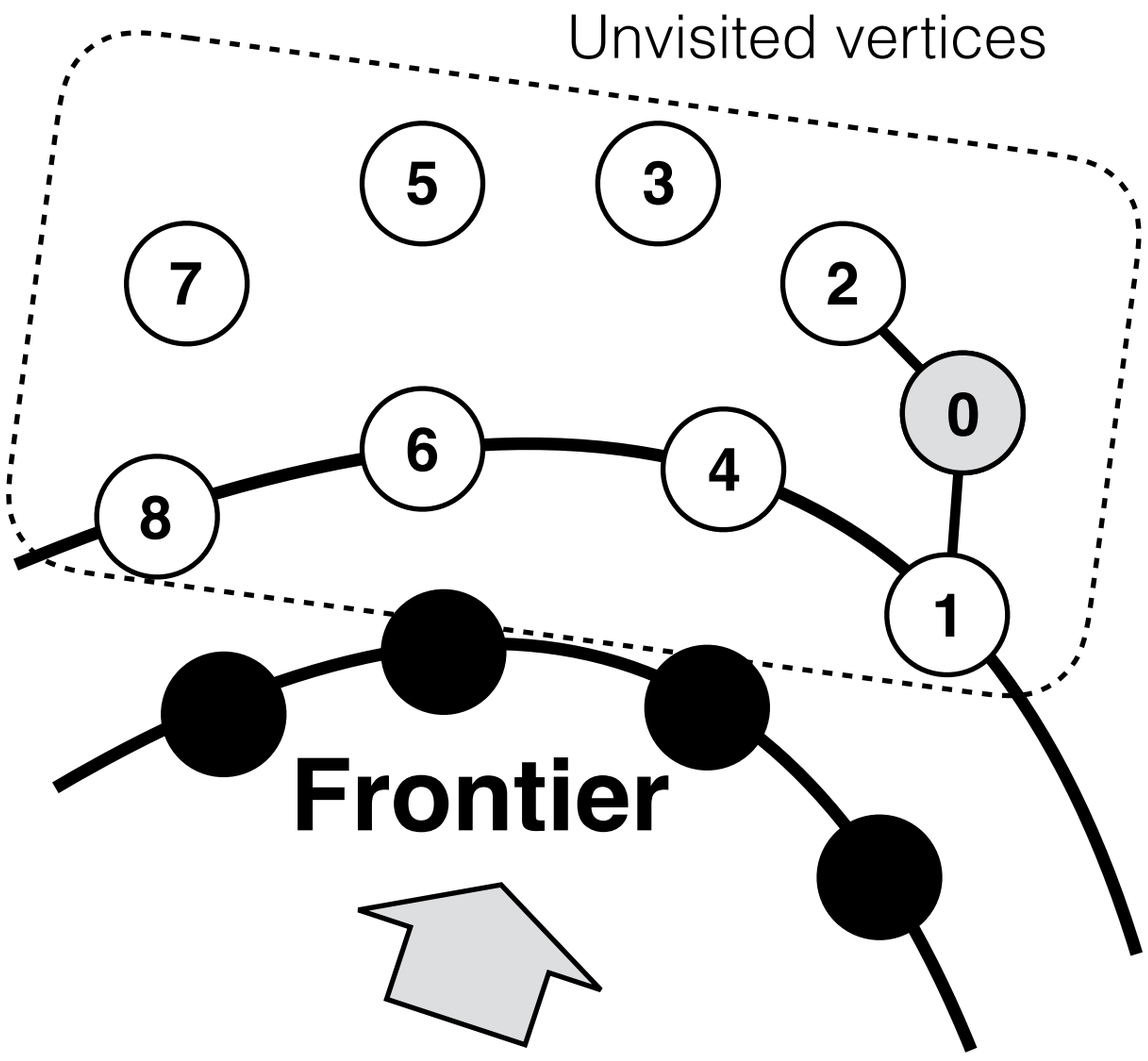


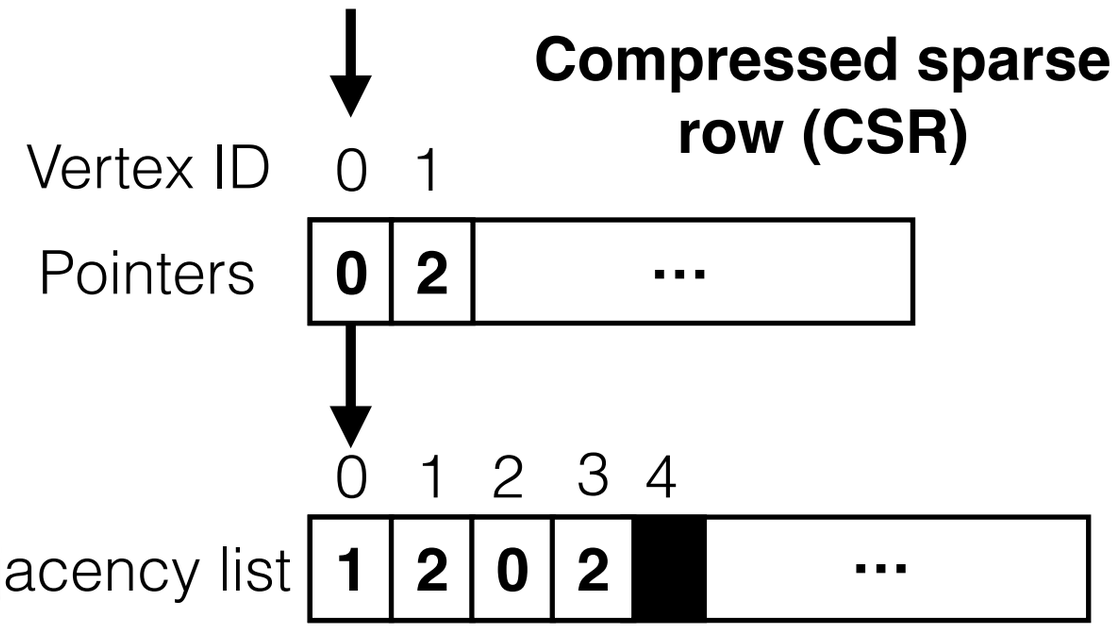
Bottom-up Algorithm



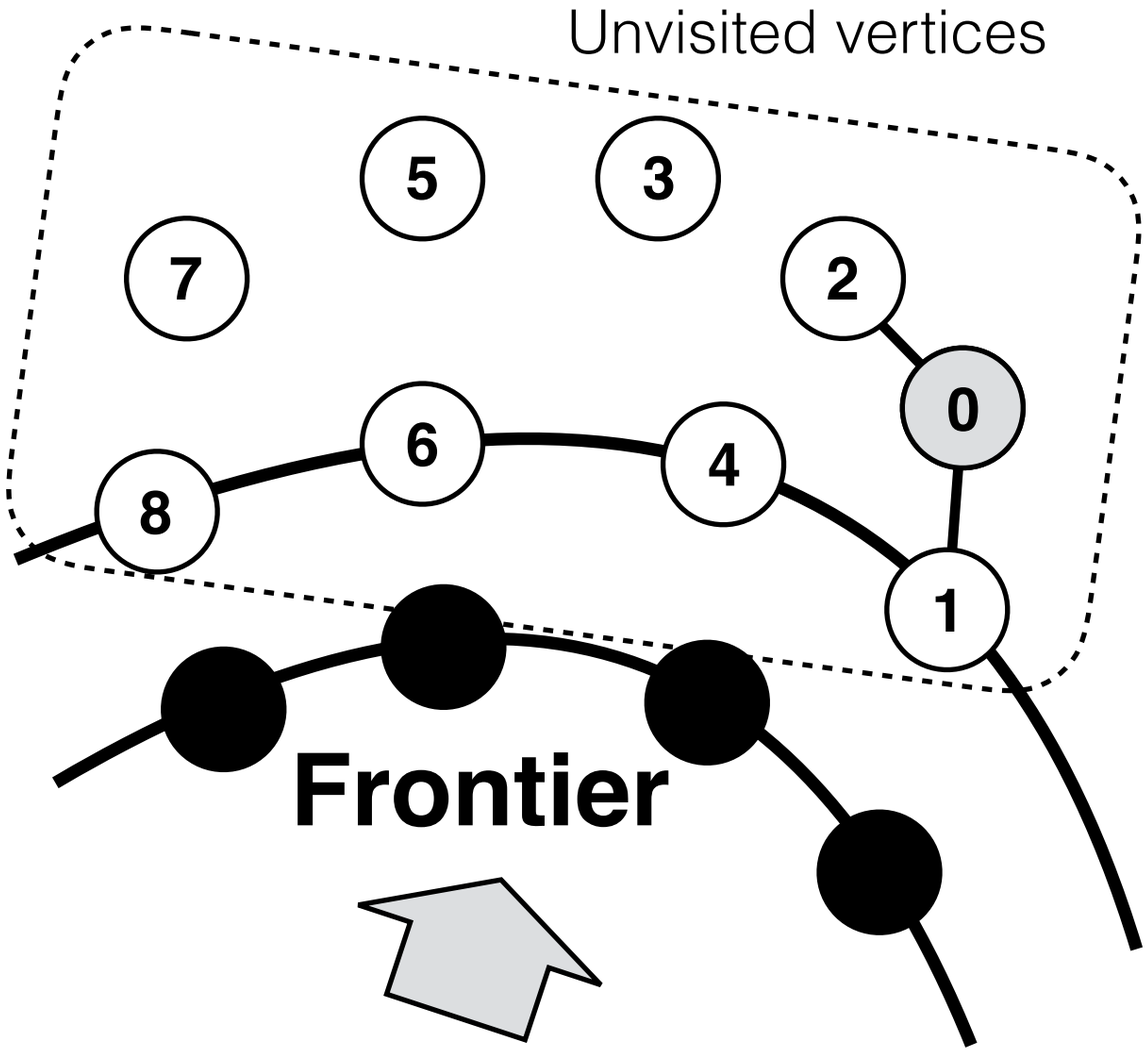


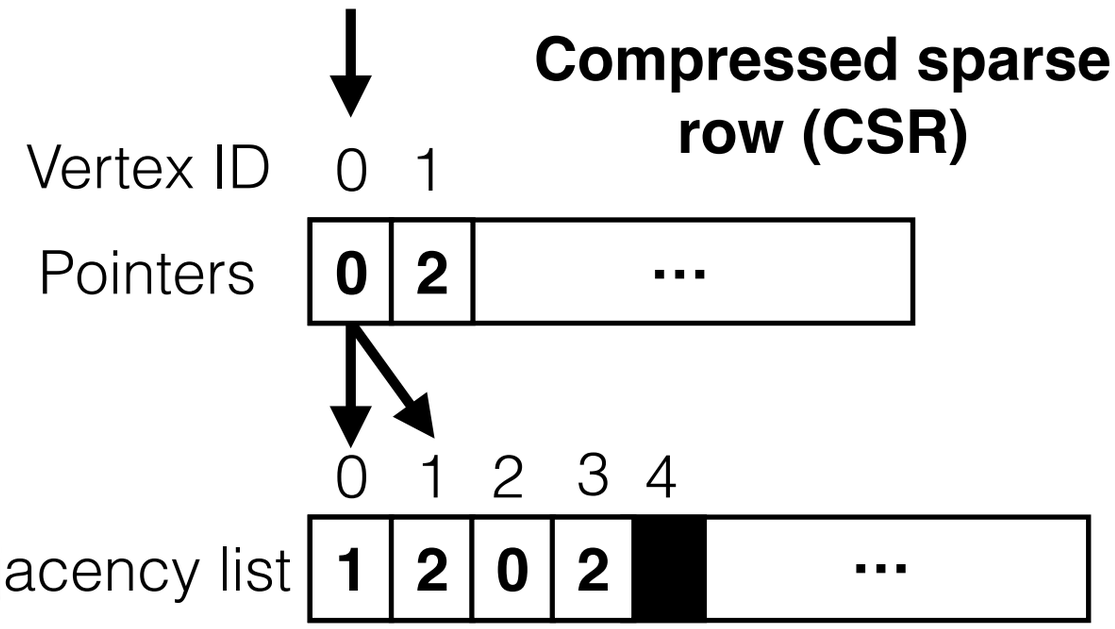
Bottom-up Algorithm



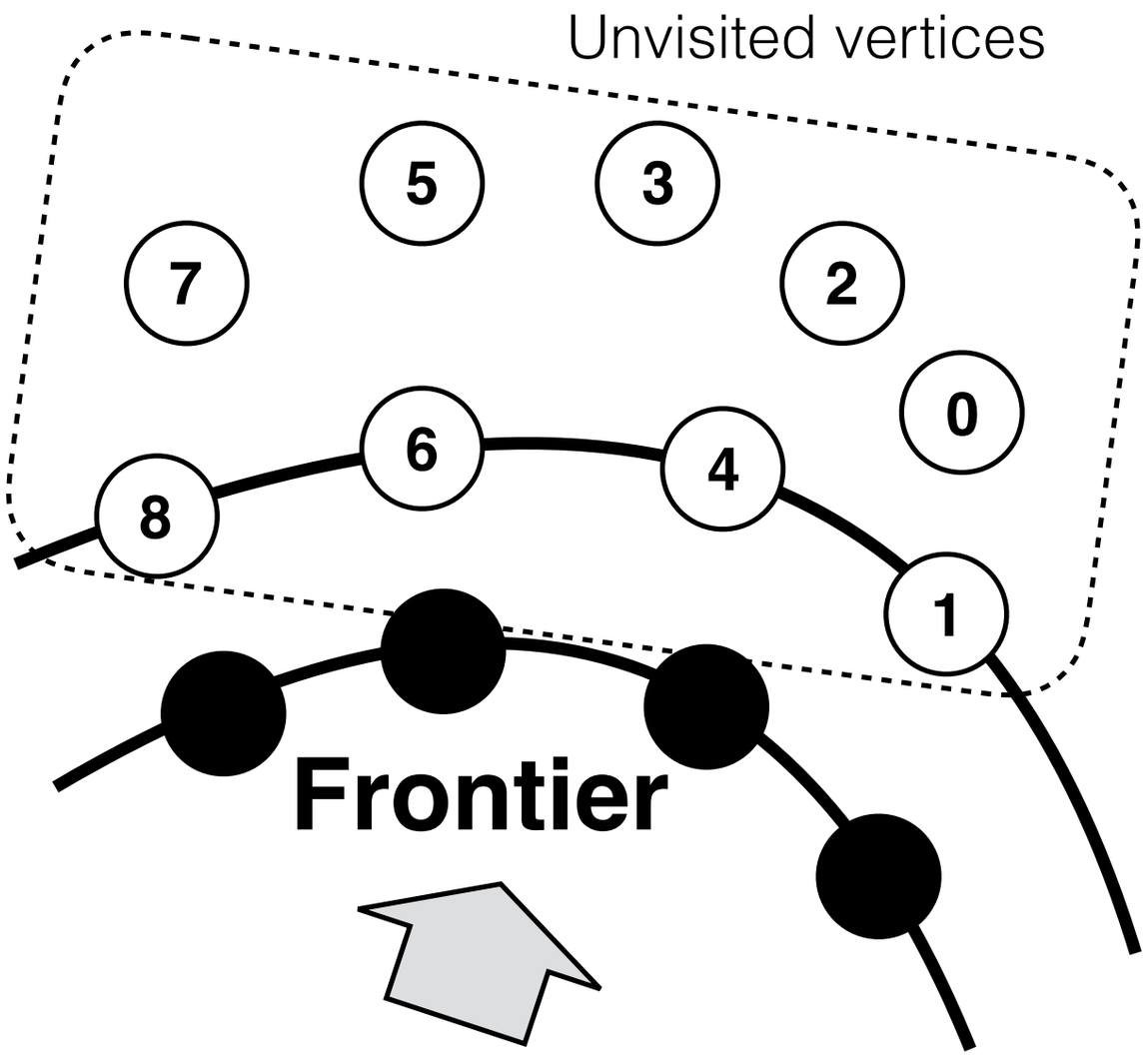


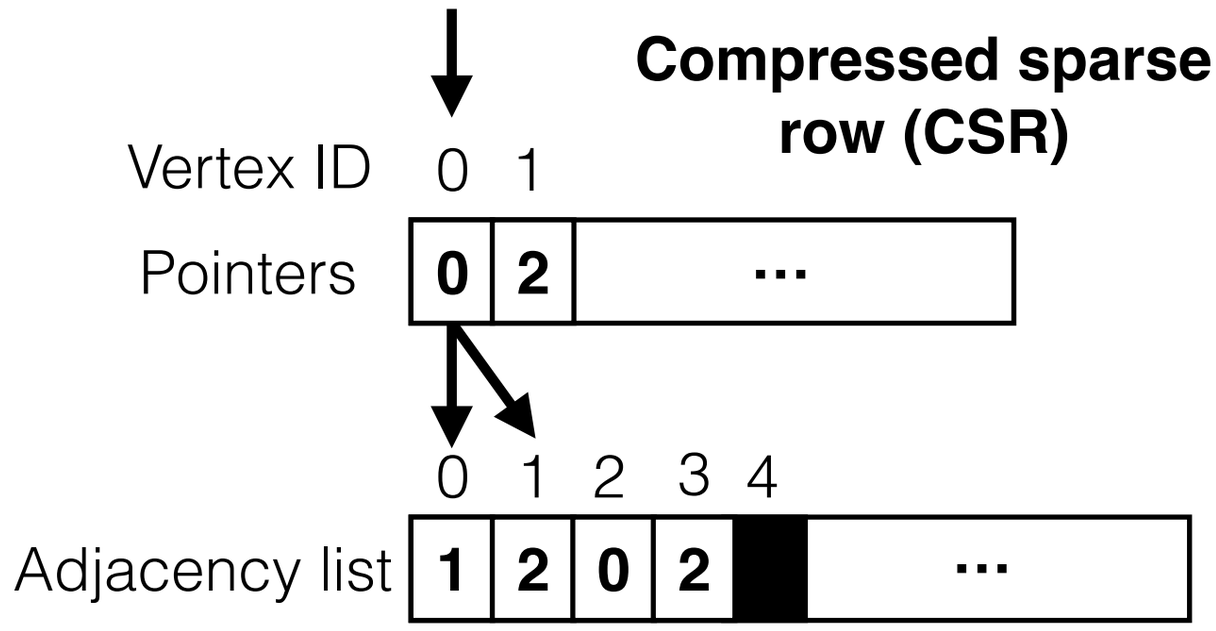
Bottom-up Algorithm



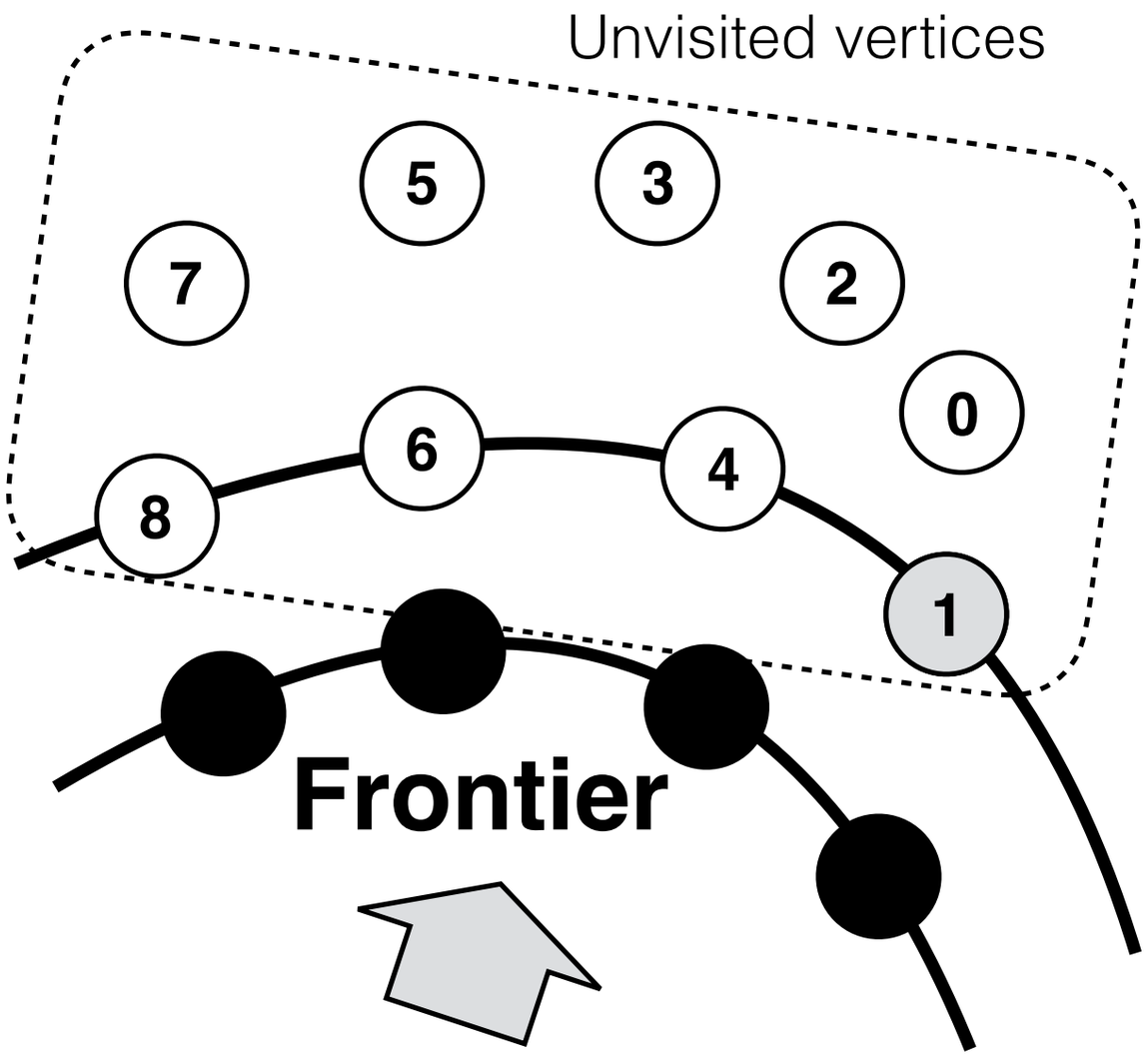


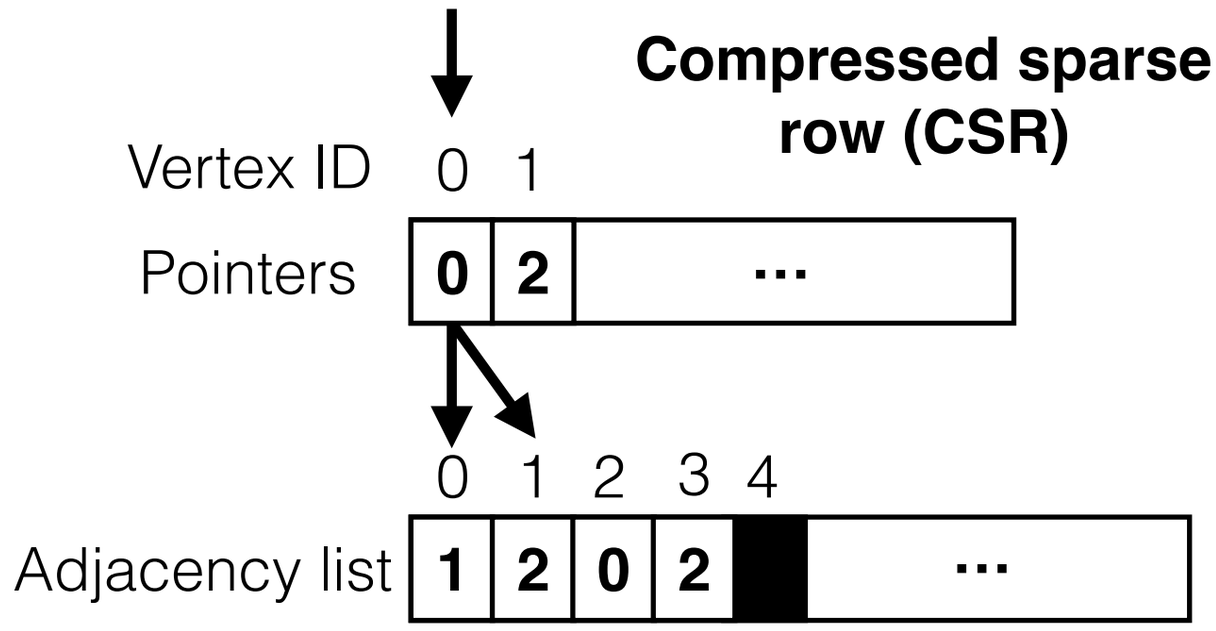
Bottom-up Algorithm



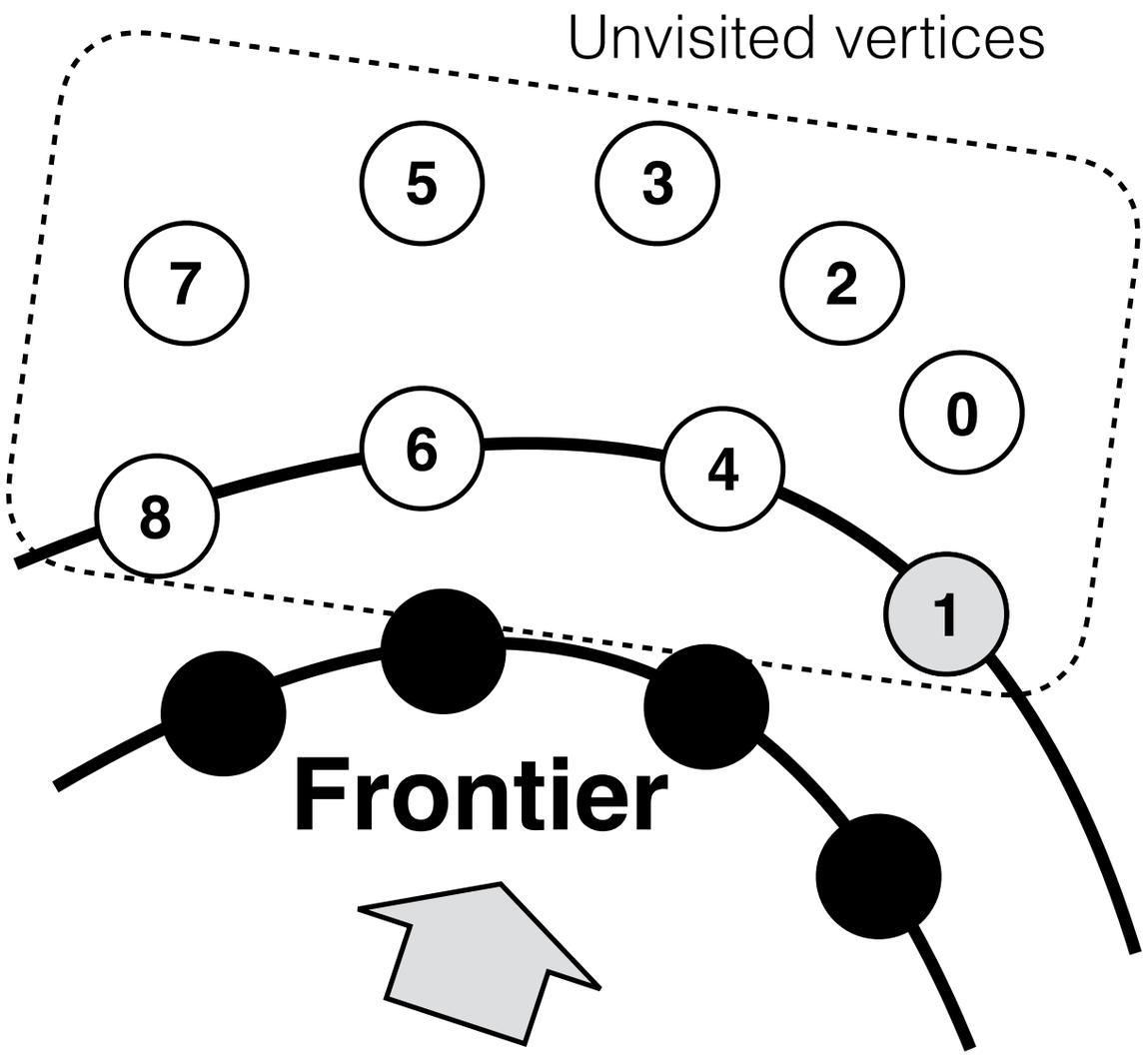


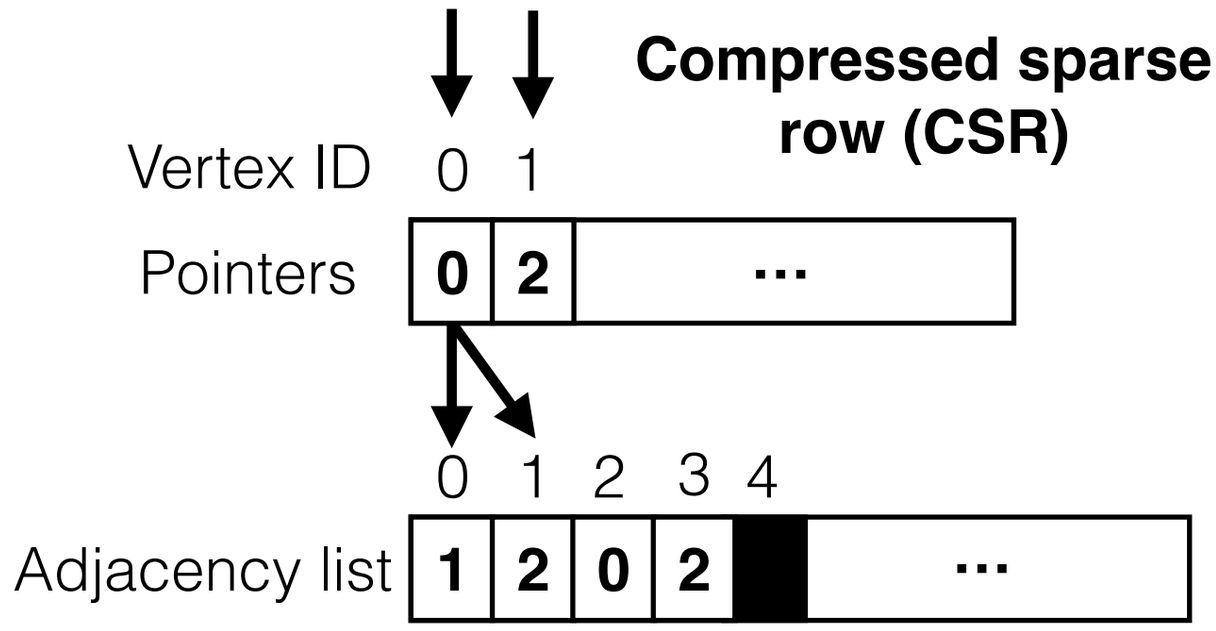
Bottom-up Algorithm



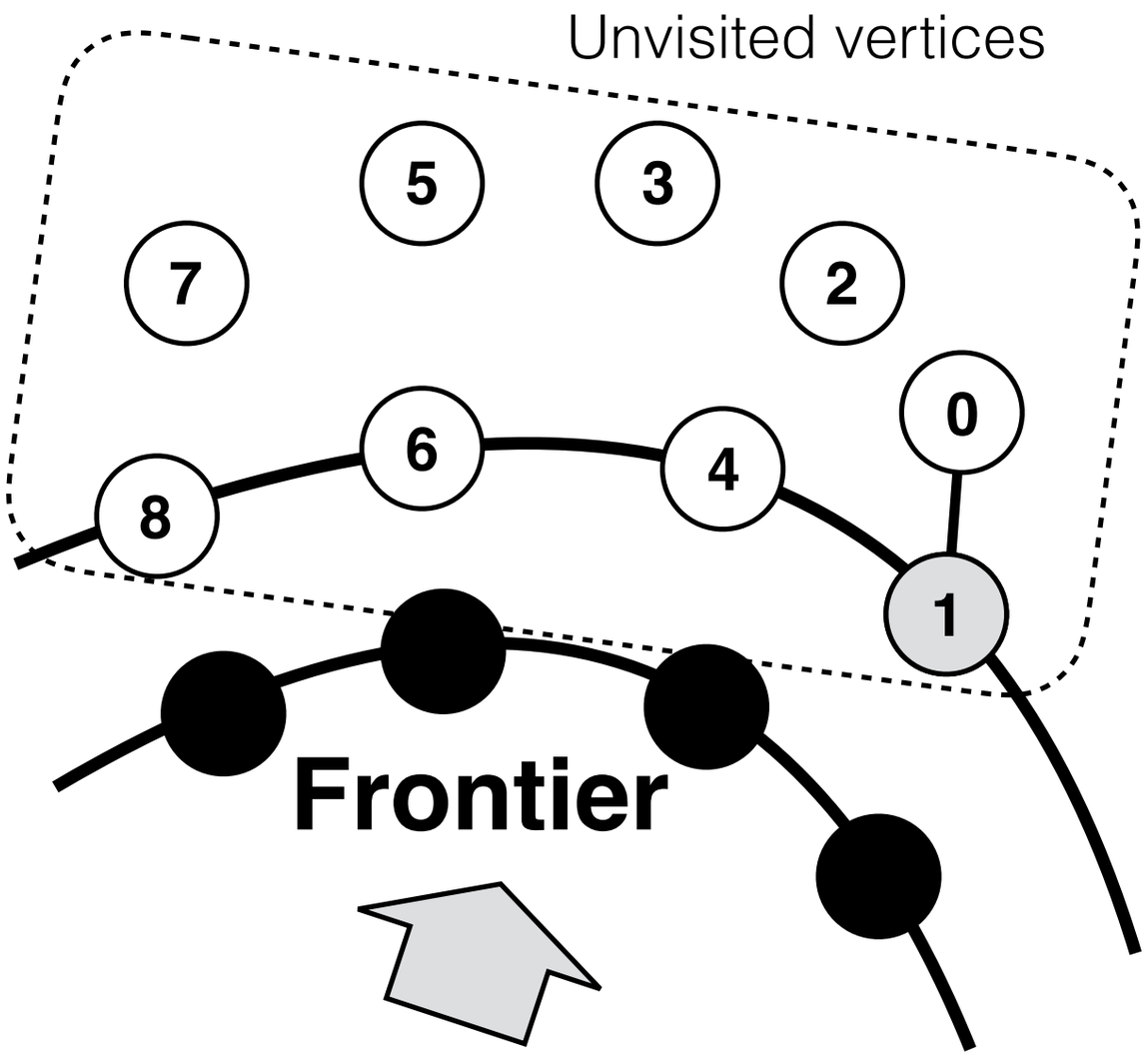


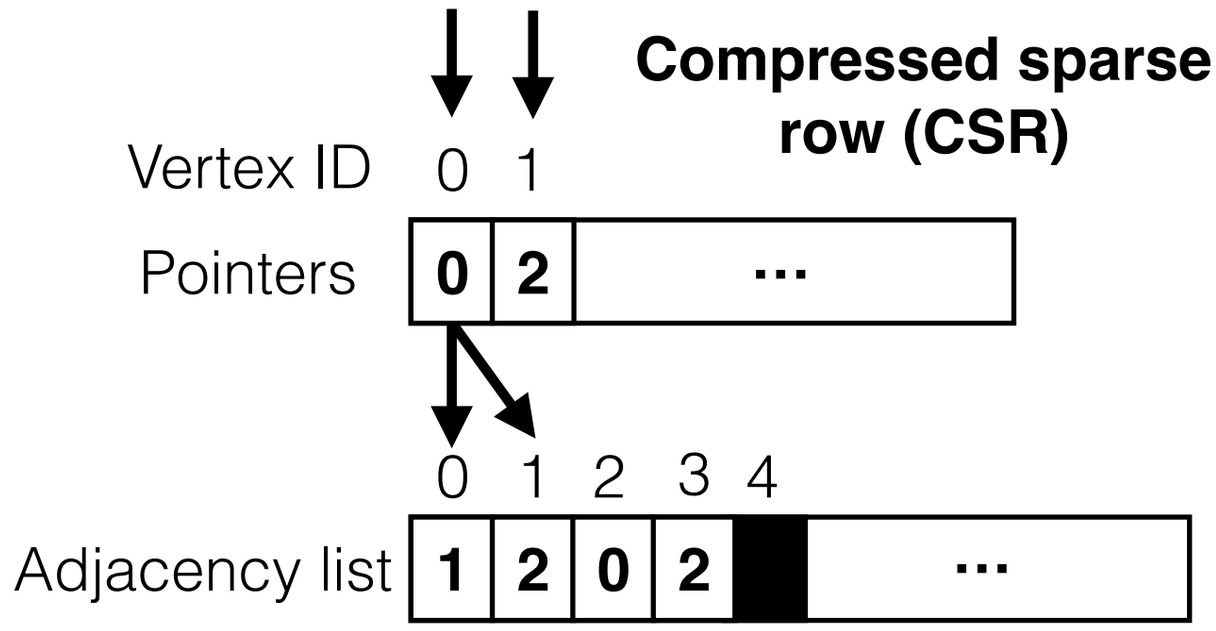
Bottom-up Algorithm



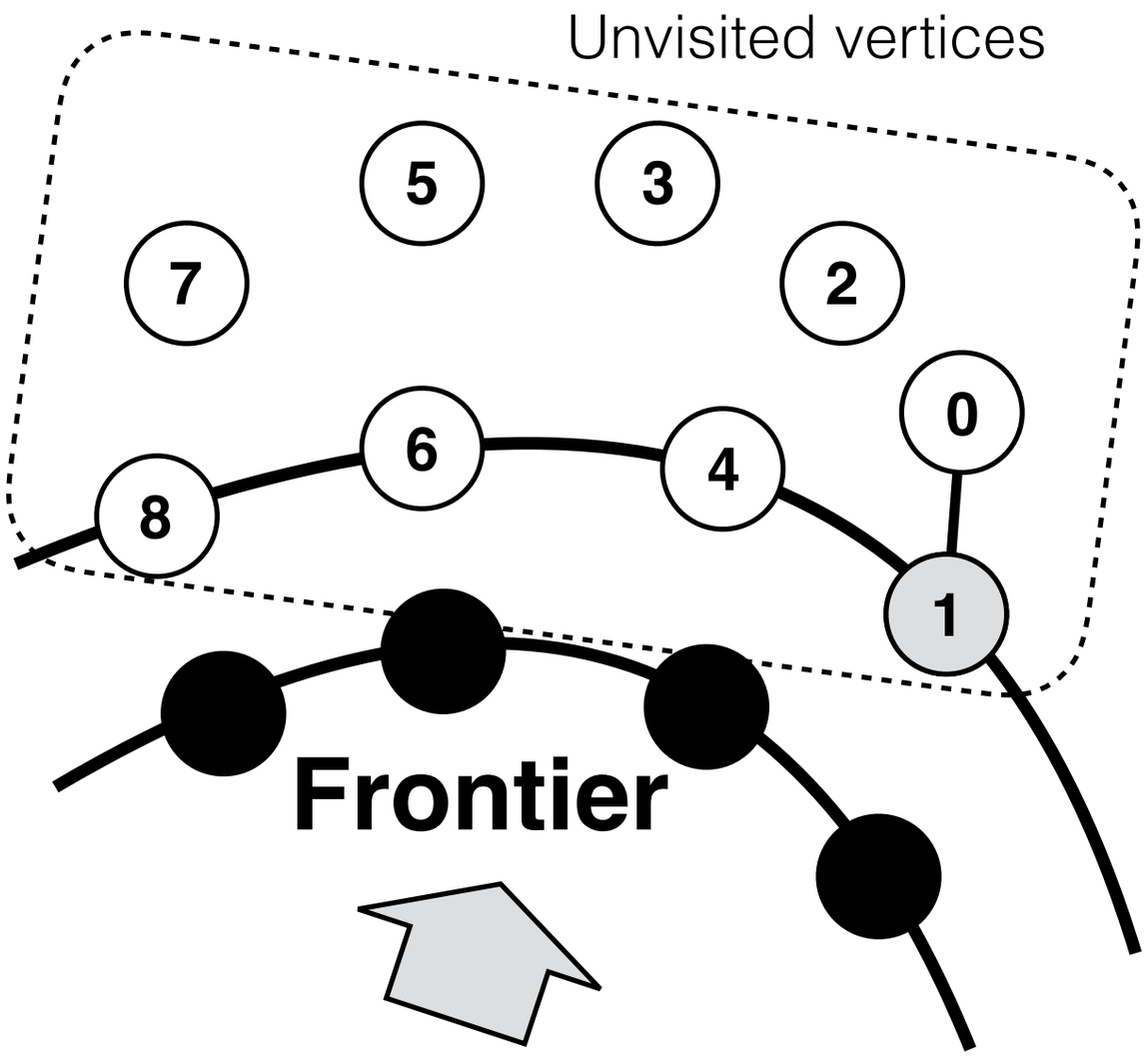


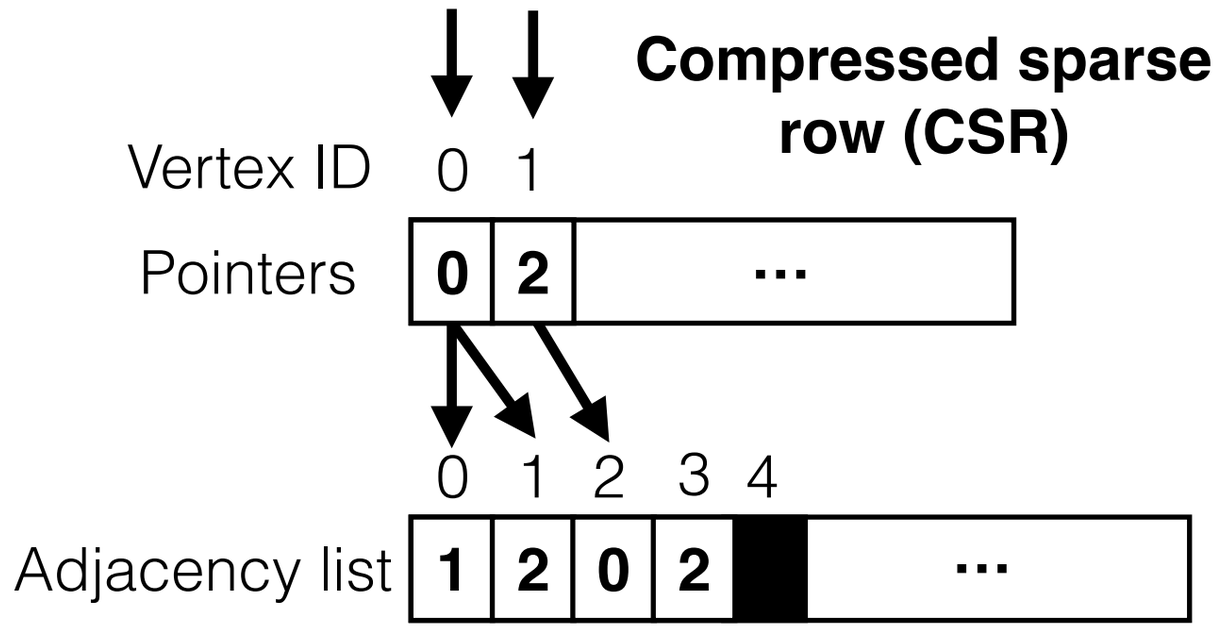
Bottom-up Algorithm



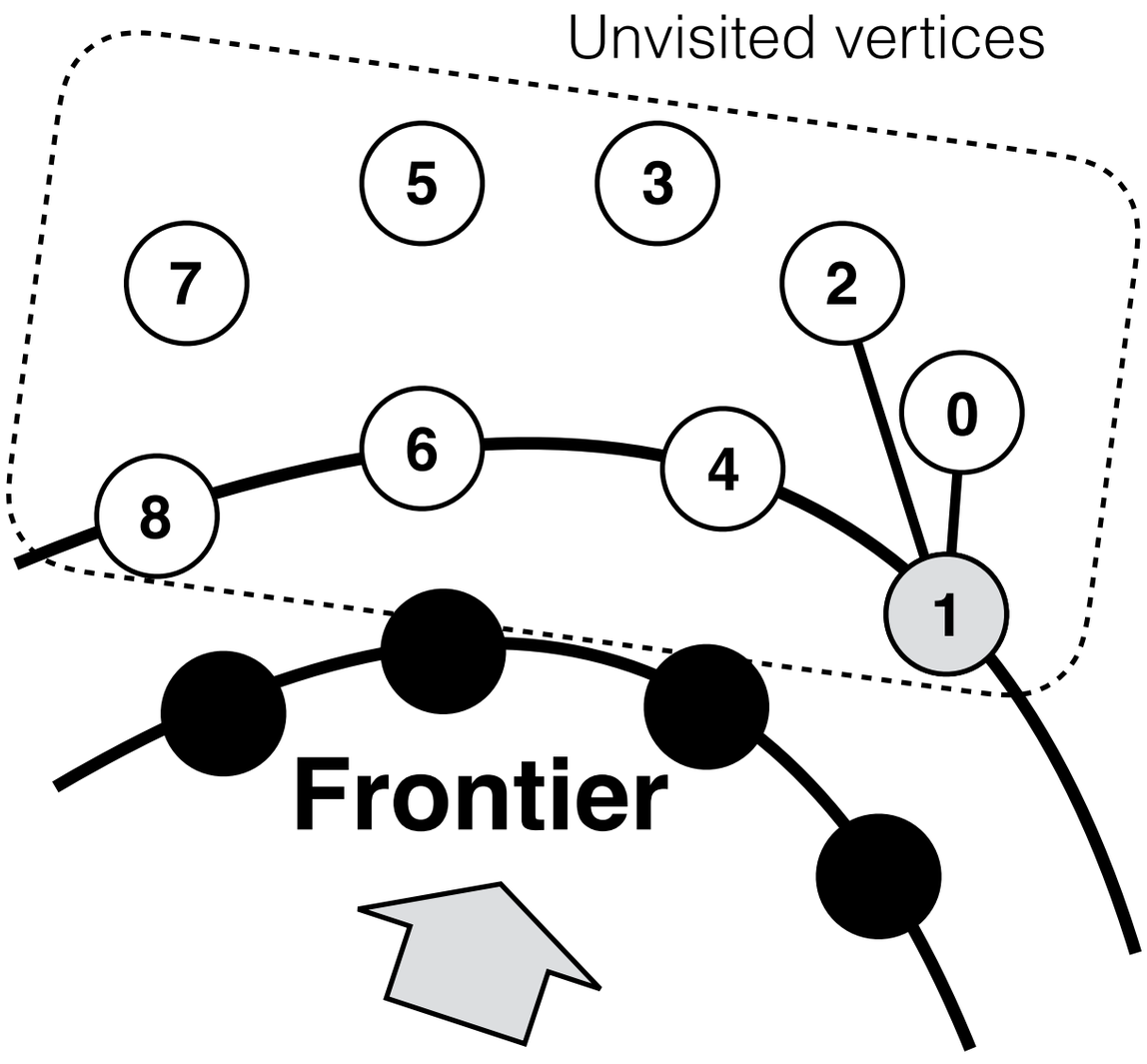


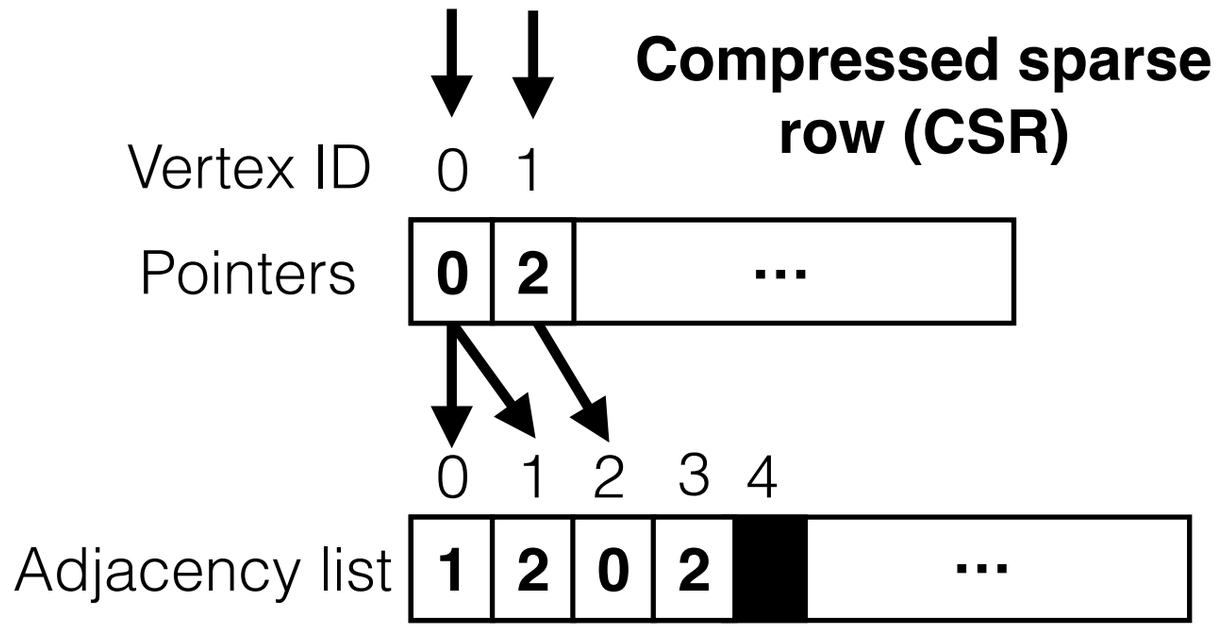
Bottom-up Algorithm



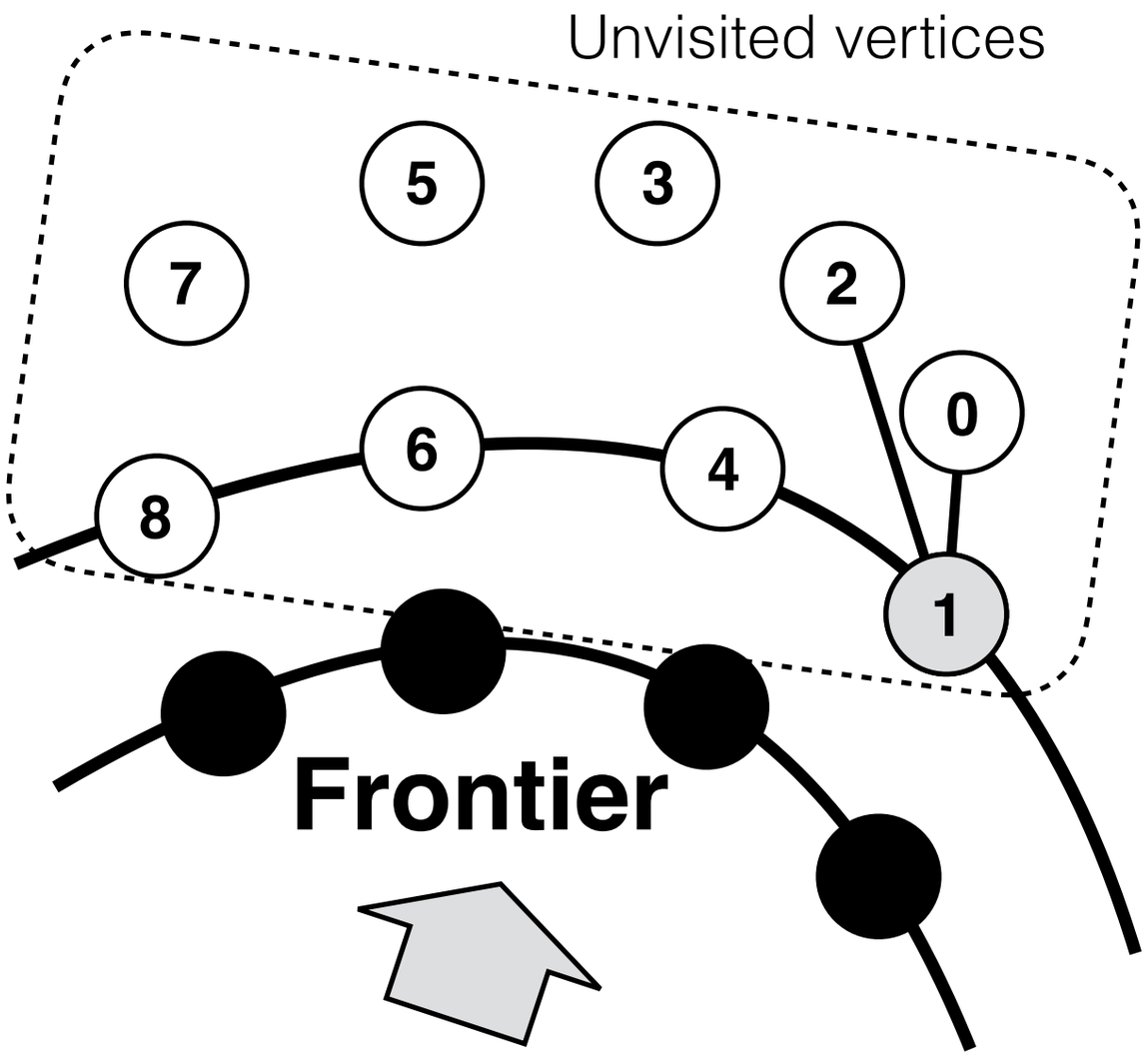


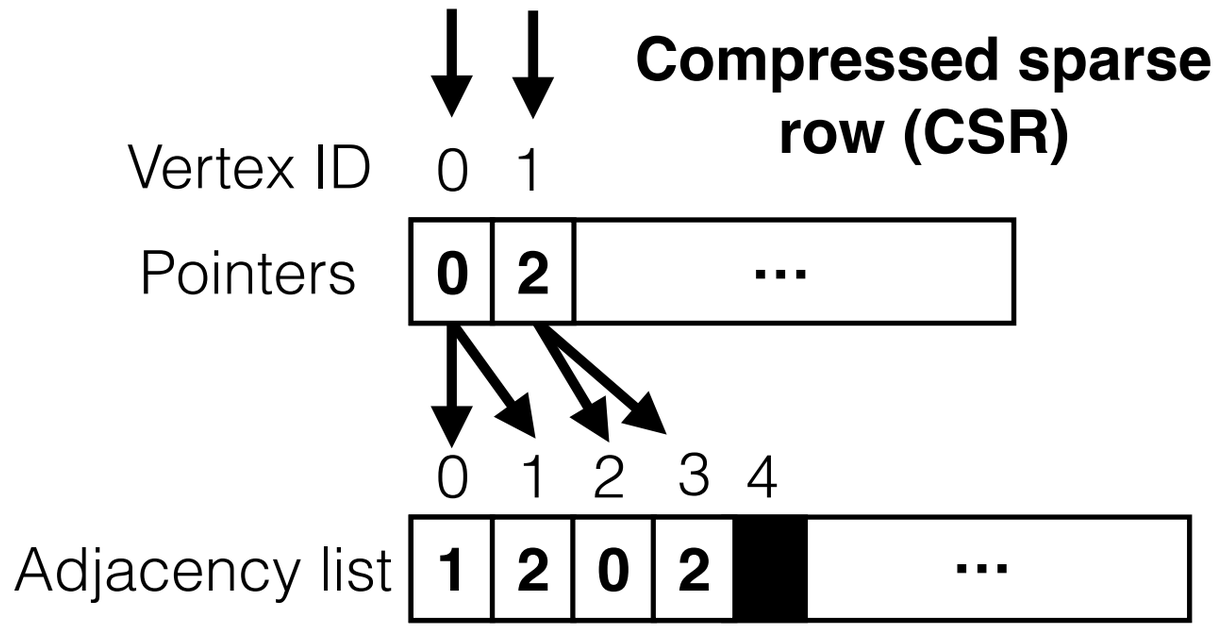
Bottom-up Algorithm



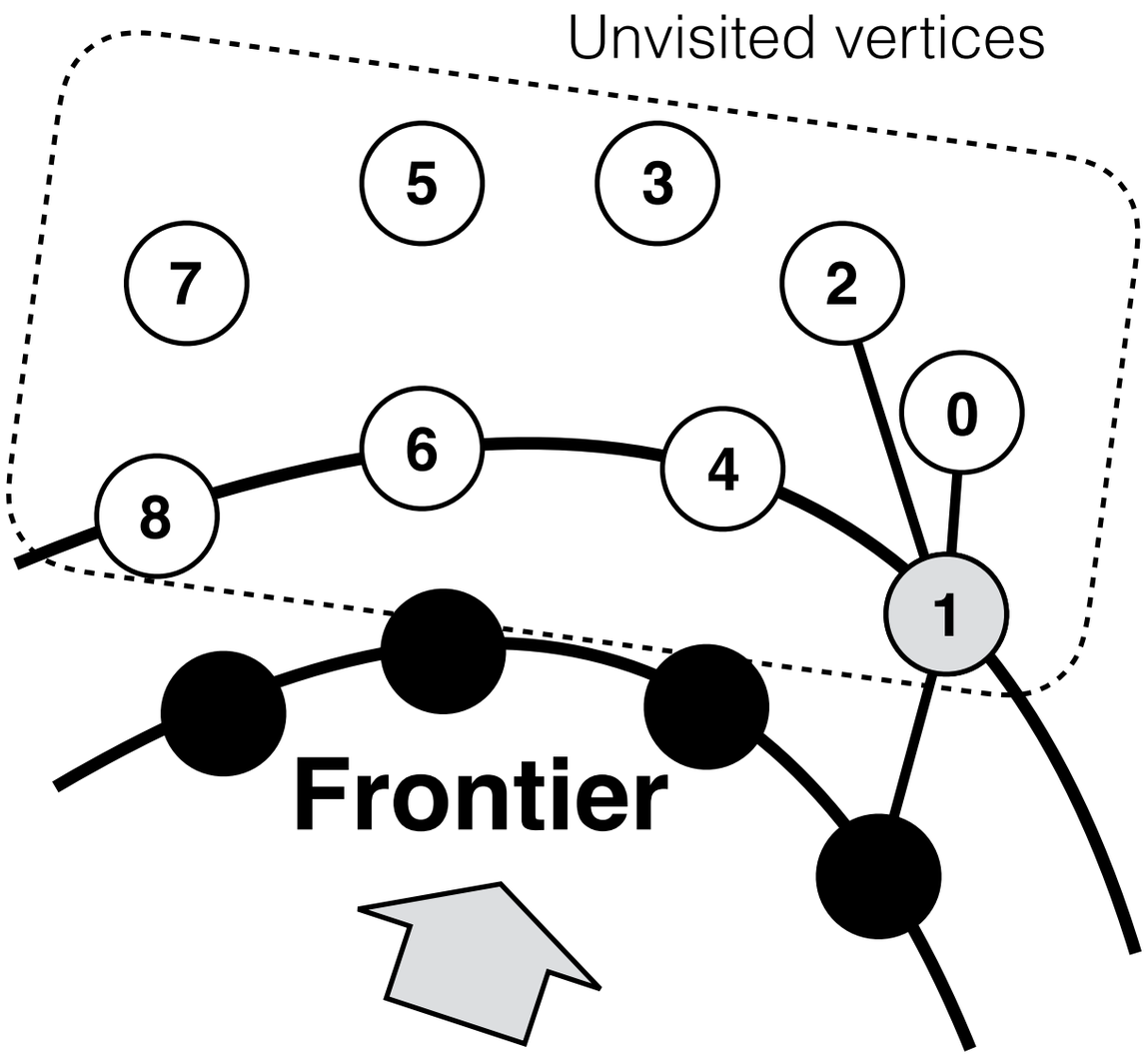


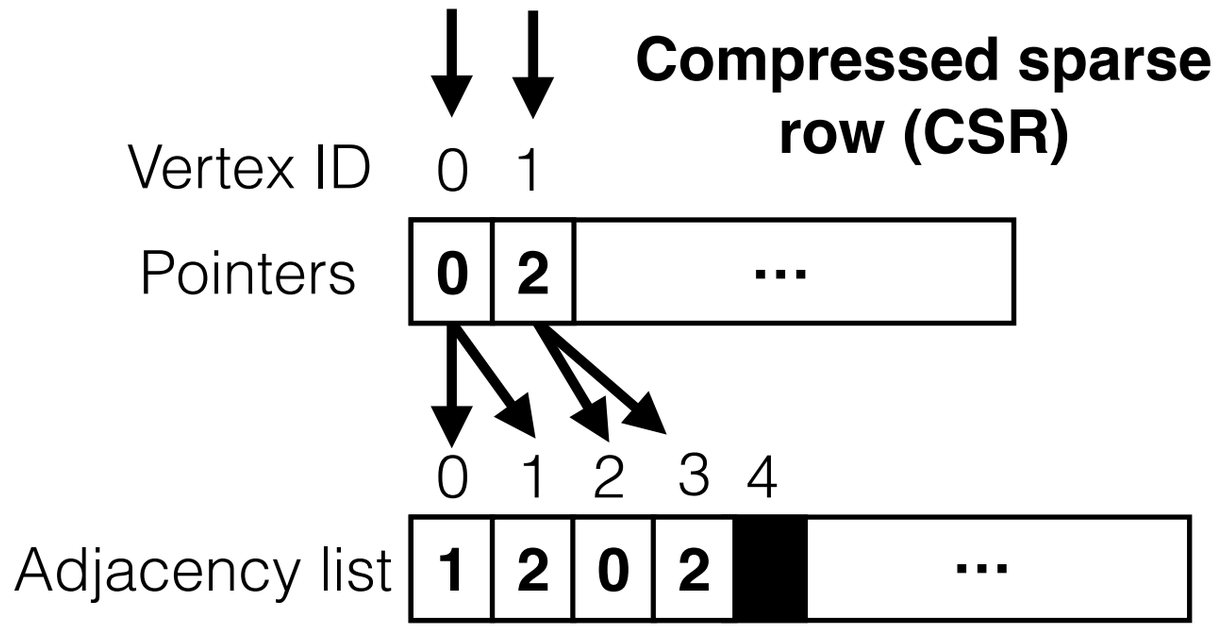
Bottom-up Algorithm



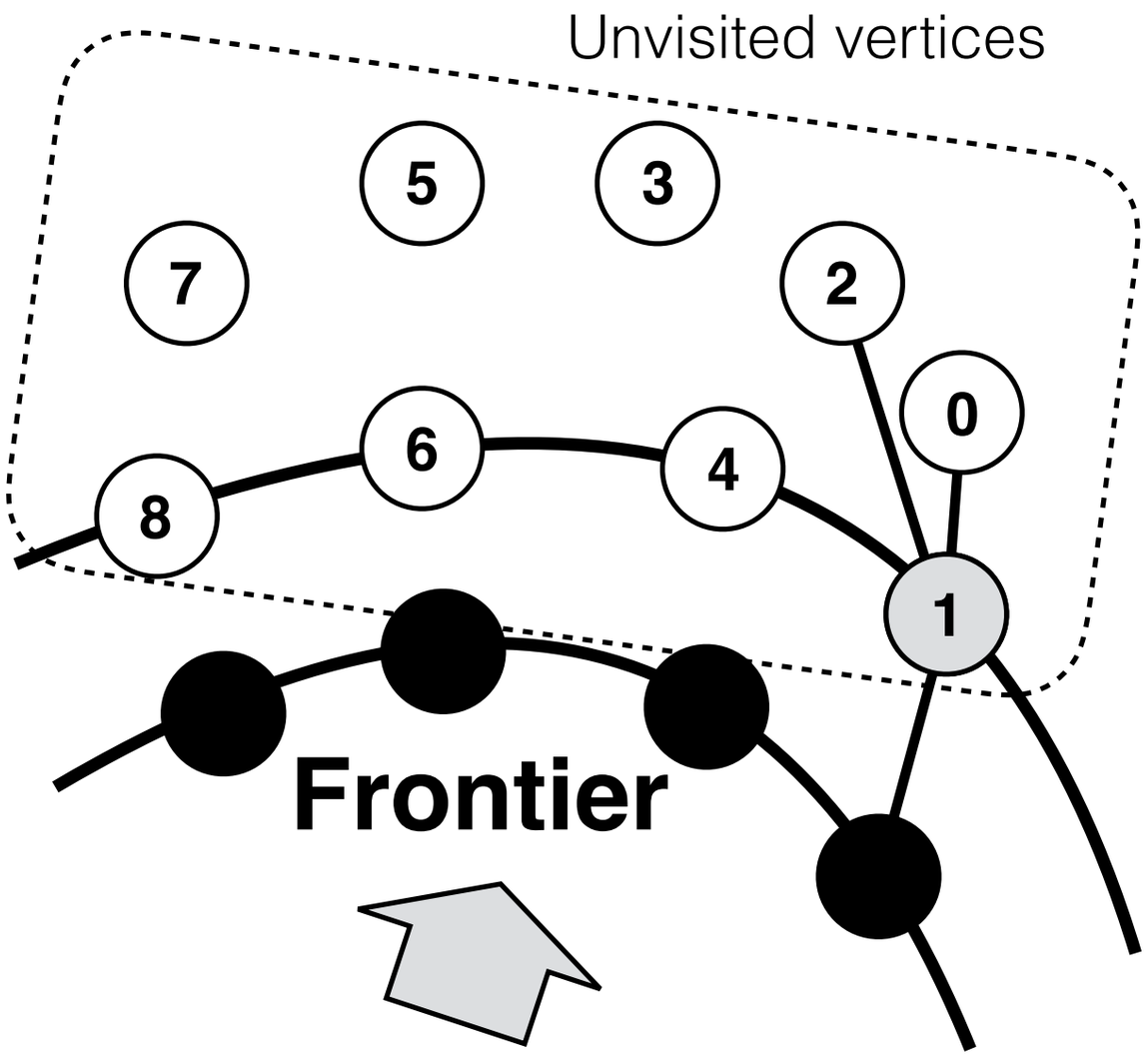


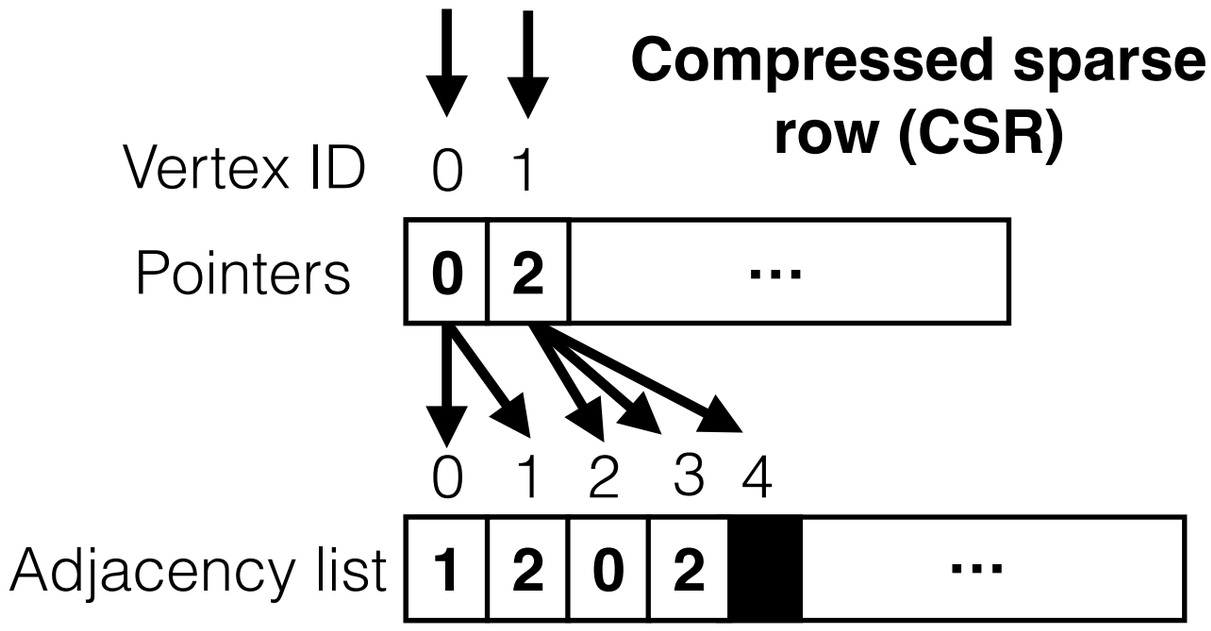
Bottom-up Algorithm



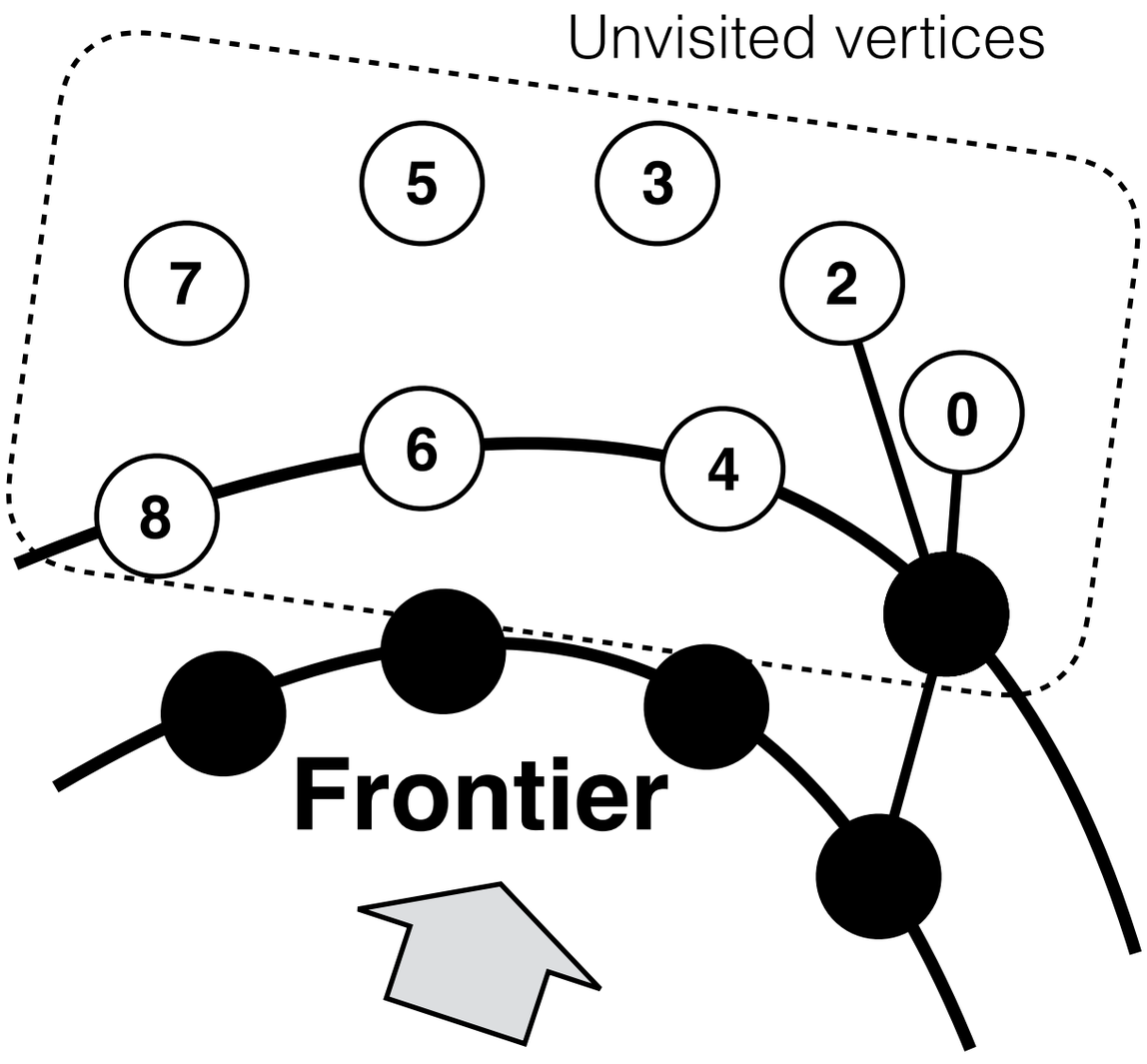


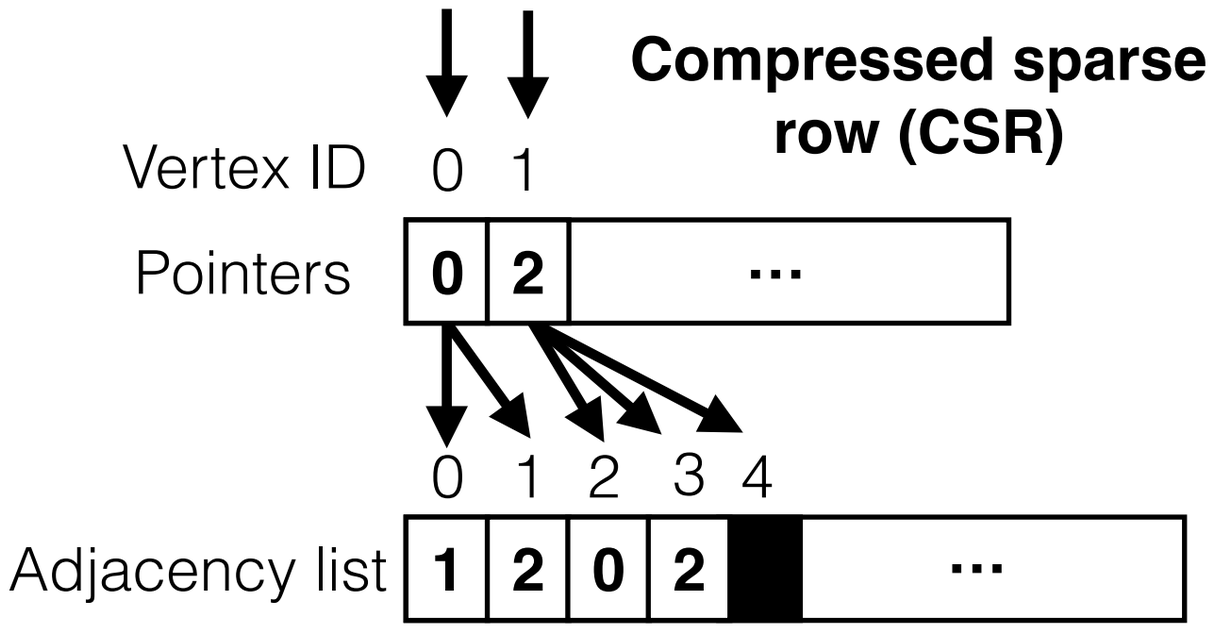
Bottom-up Algorithm



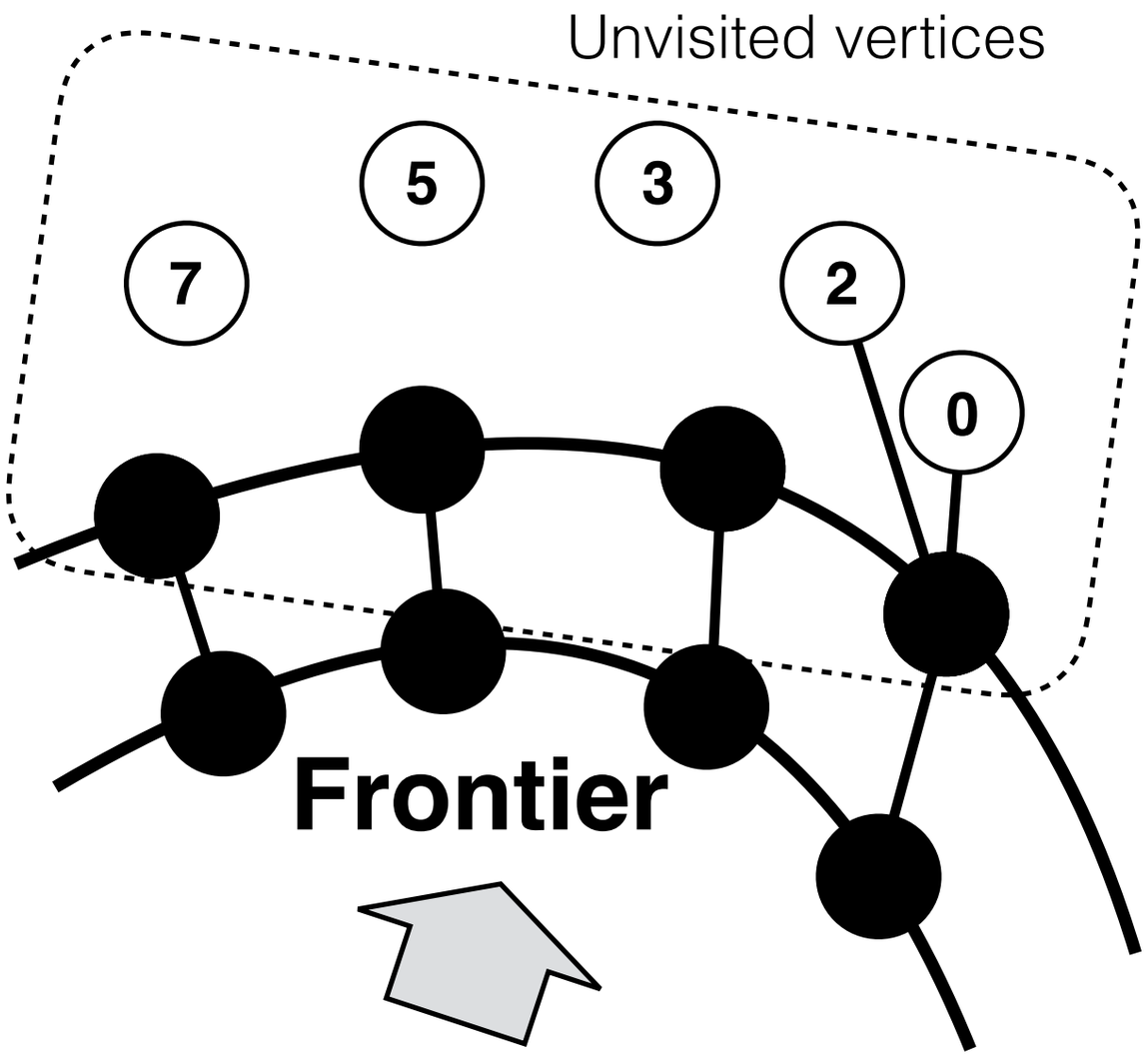


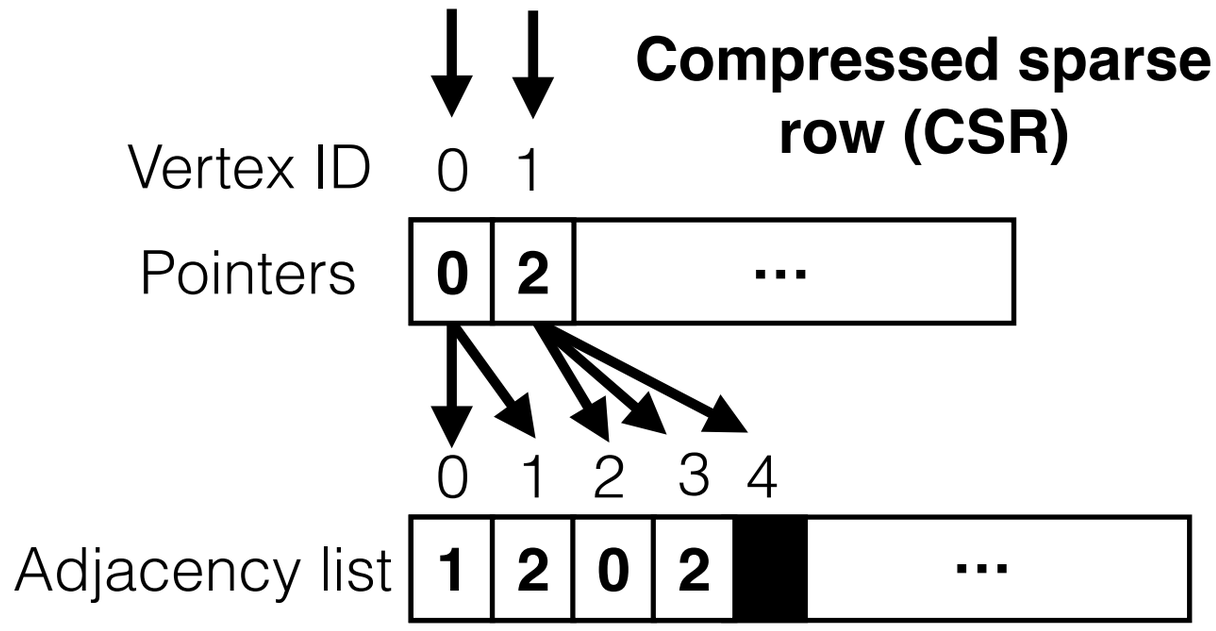
Bottom-up Algorithm



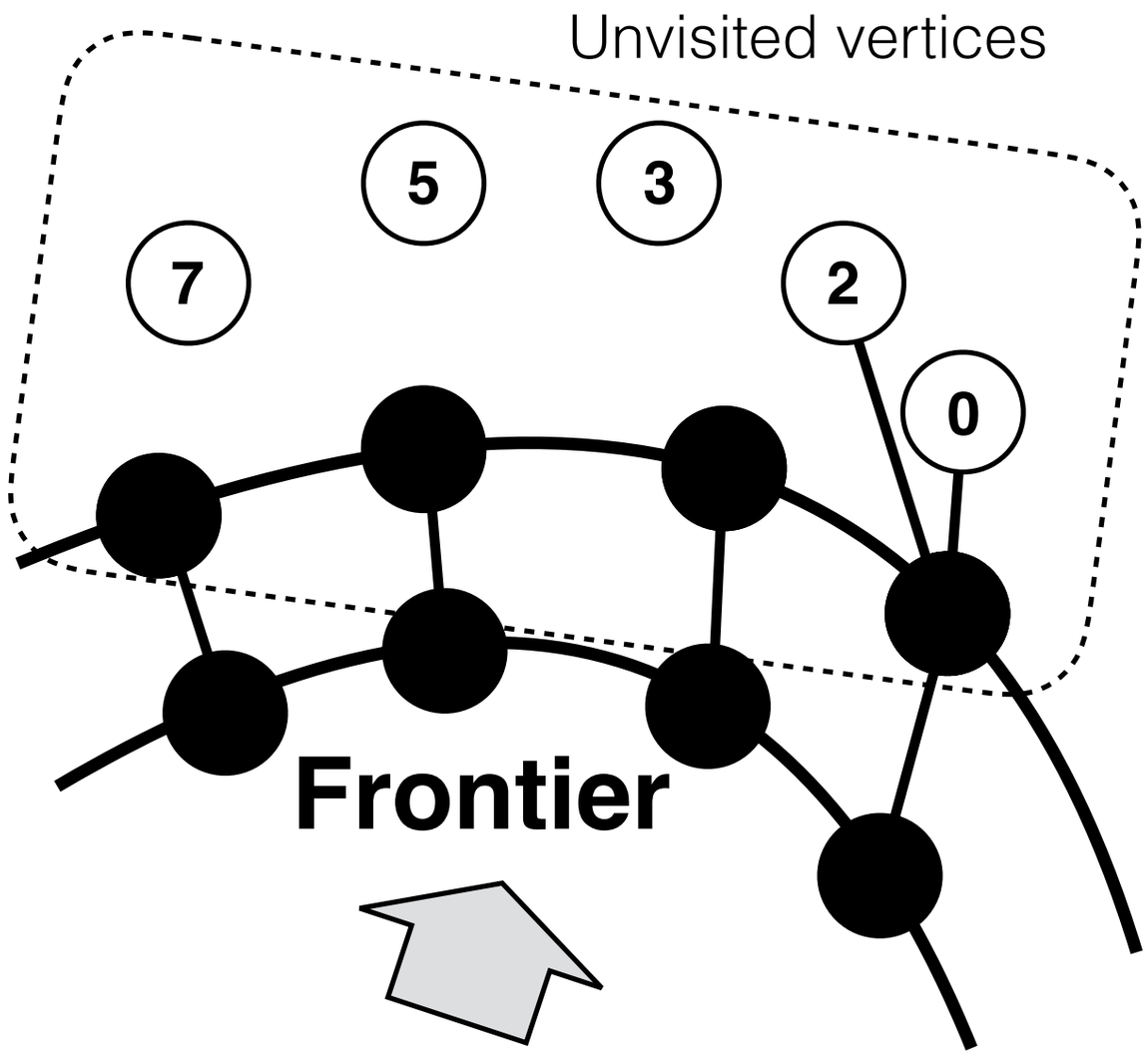


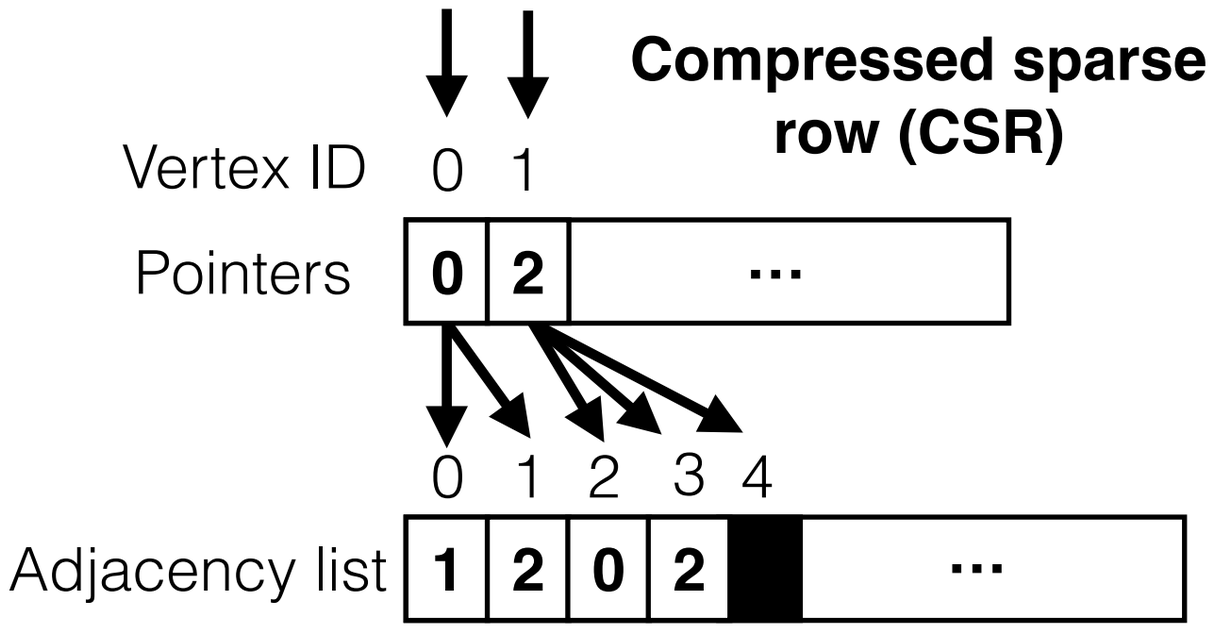
Bottom-up Algorithm





Bottom-up Algorithm



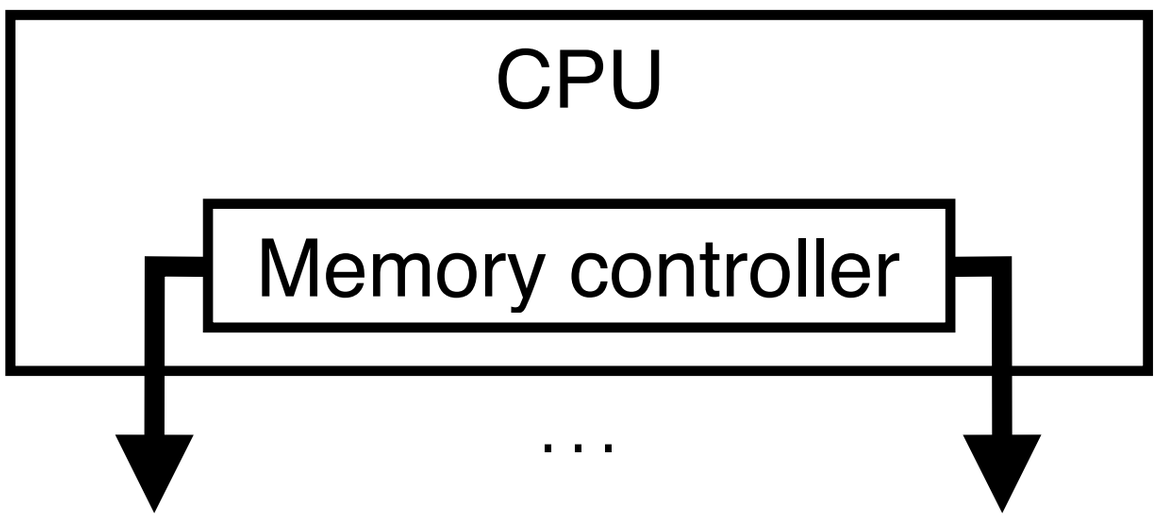


- This search is parallelized using OpenMP
- Memory-intensive

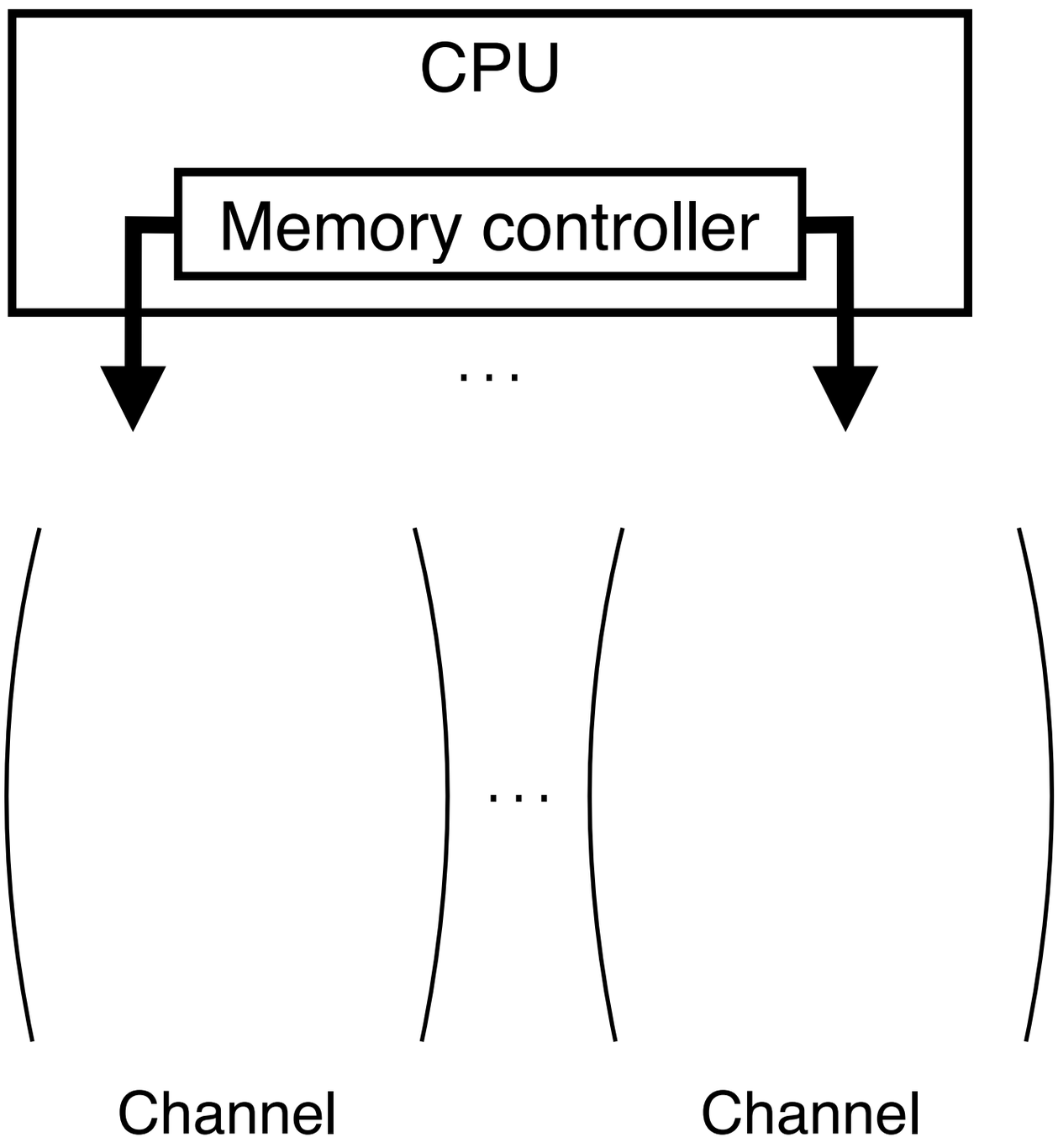
Agenda

- State-of-the-art BFS implementation
- DRAM mechanisms
- Memory access analysis with conventional address mapping schemes
- Proposed: per-row channel interleaving
- Evaluation of power efficiency

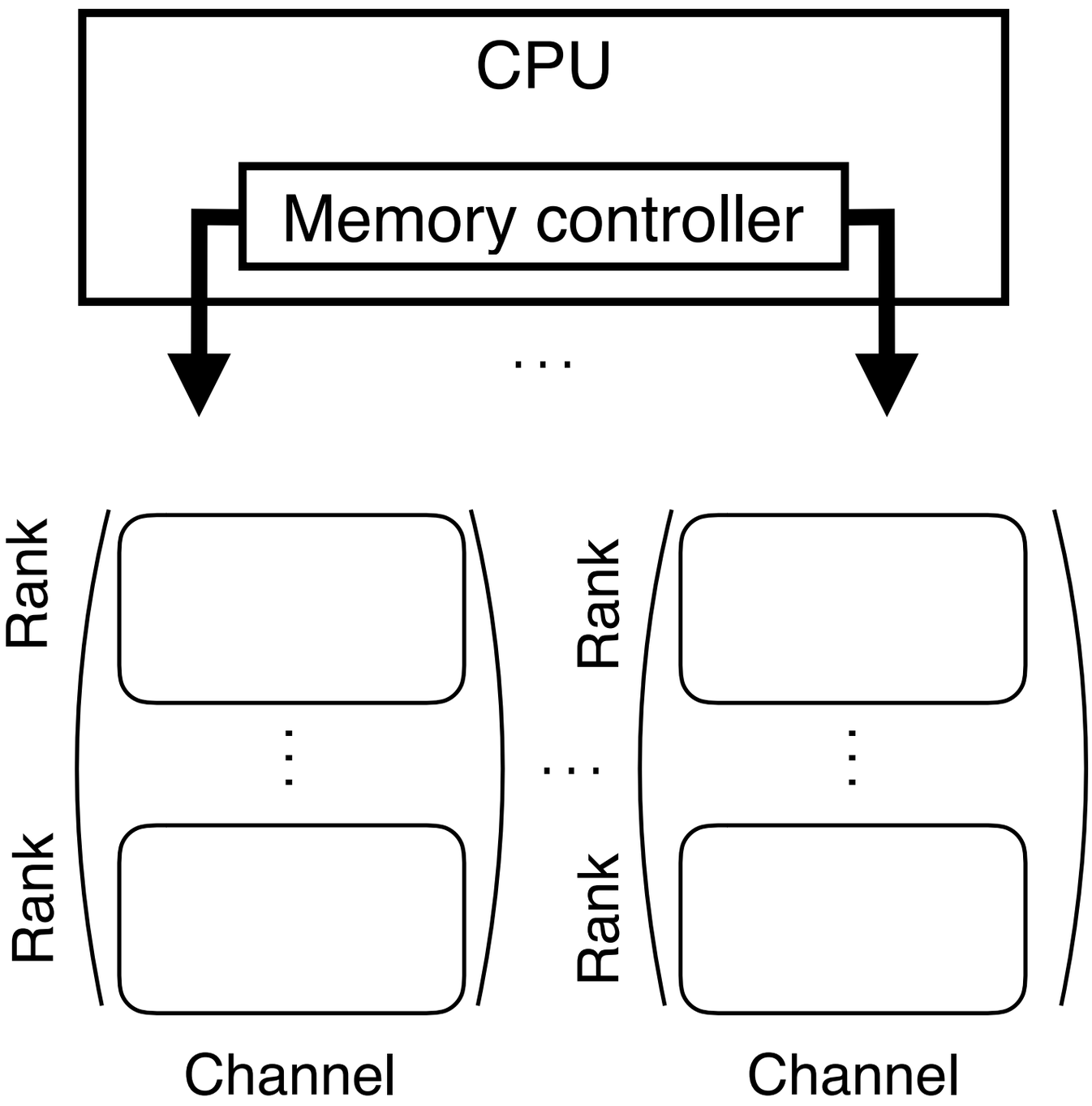
Multibank architecture



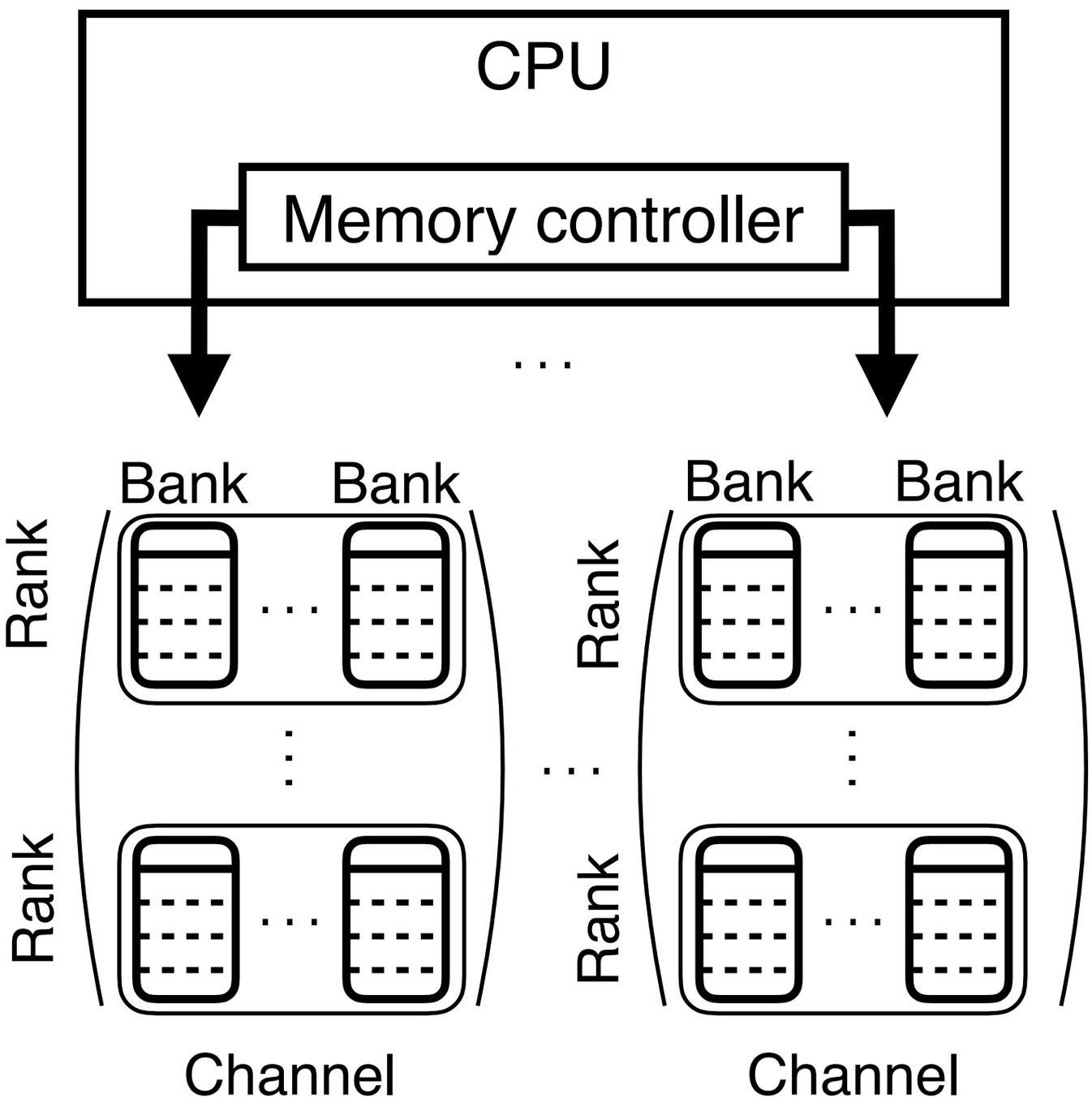
Multibank architecture



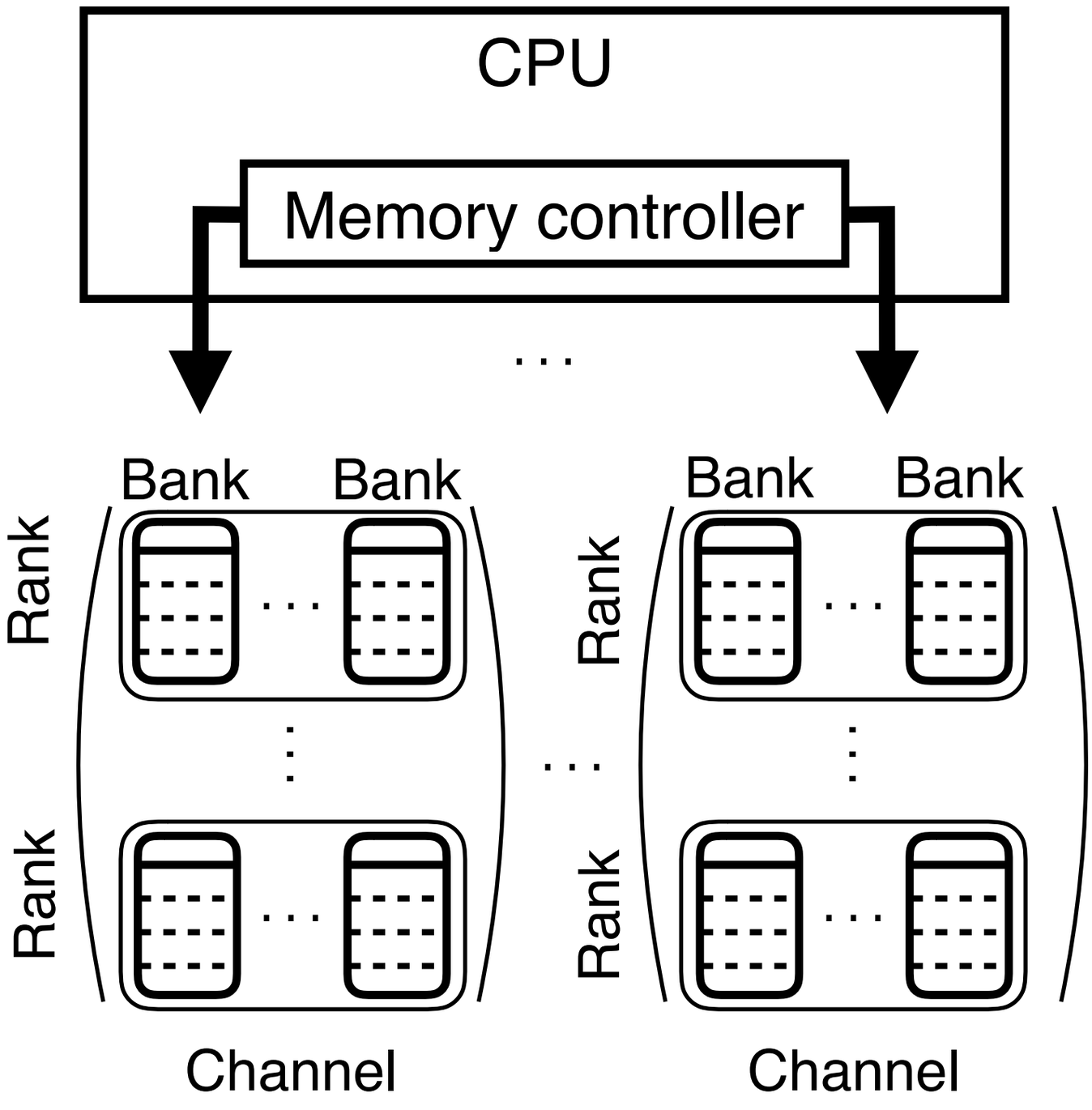
Multibank architecture

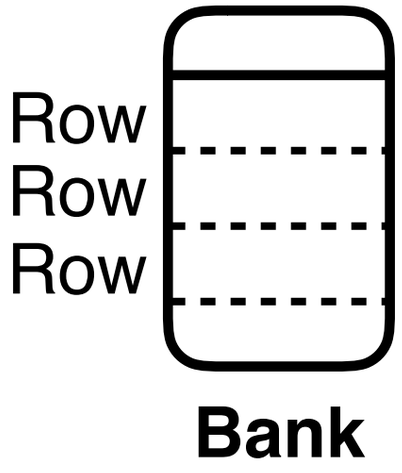


Multibank architecture

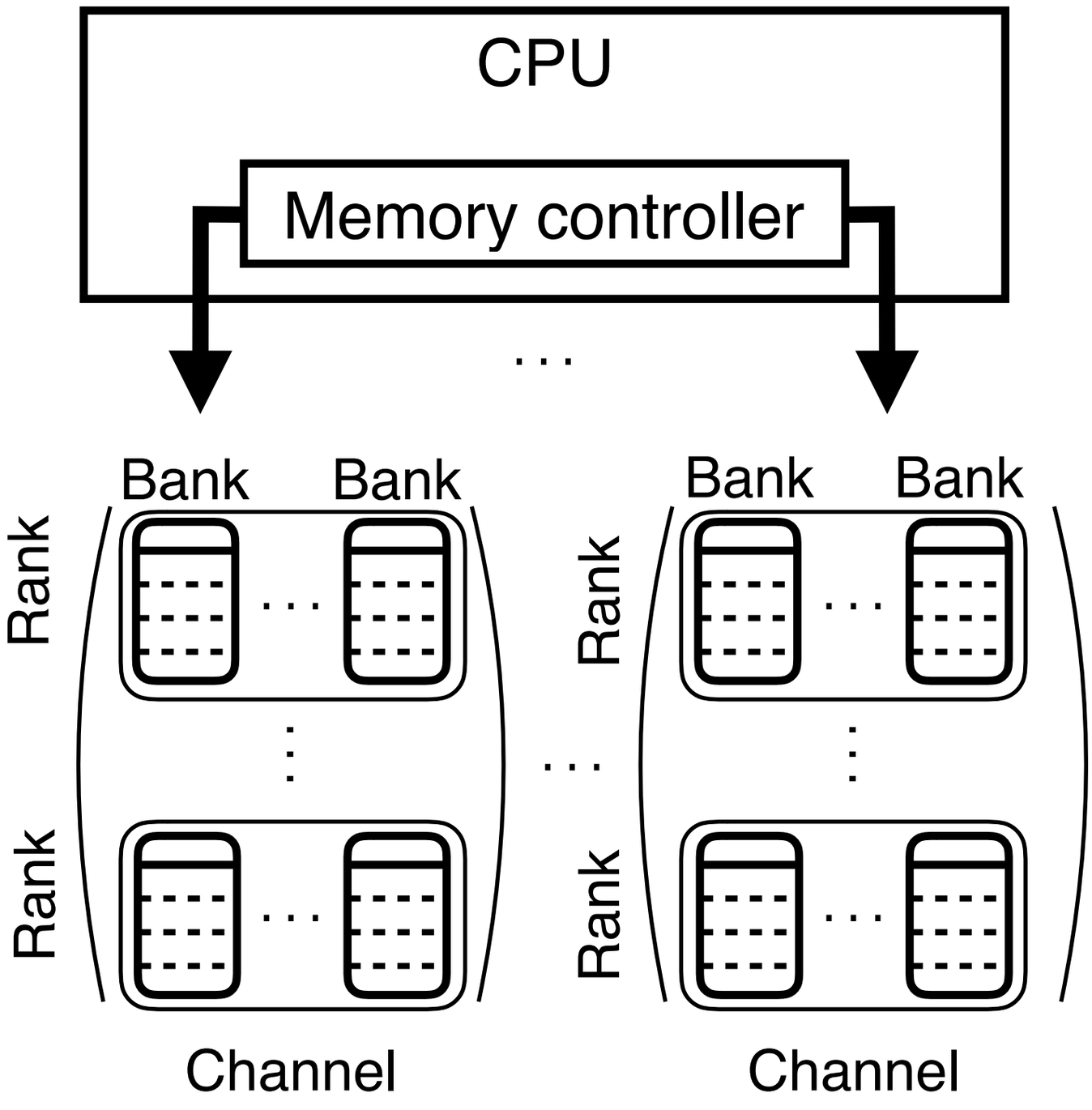


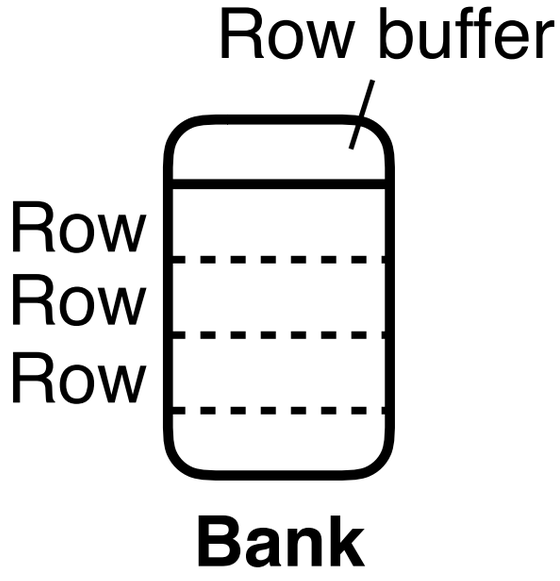
Multibank architecture



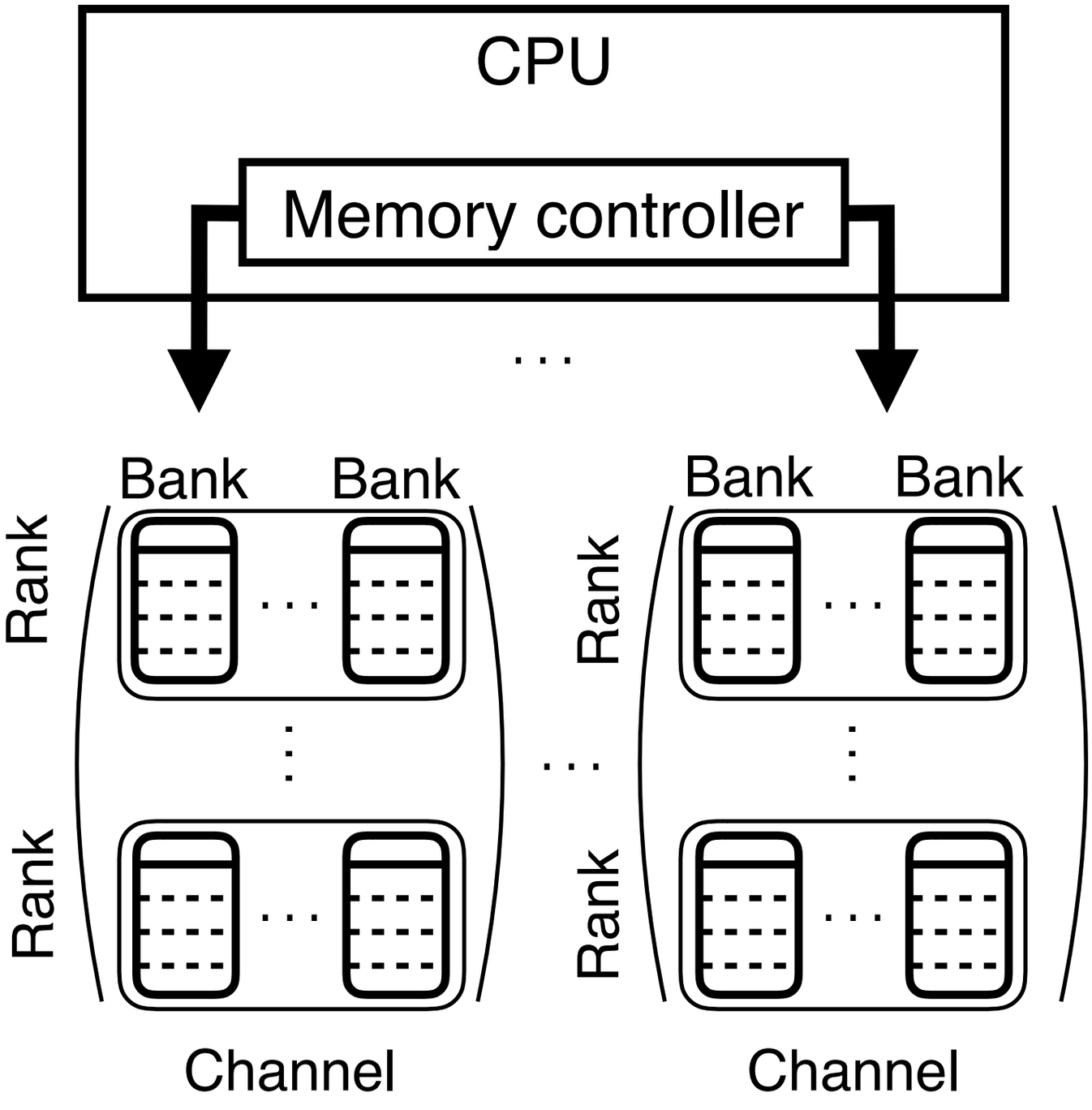


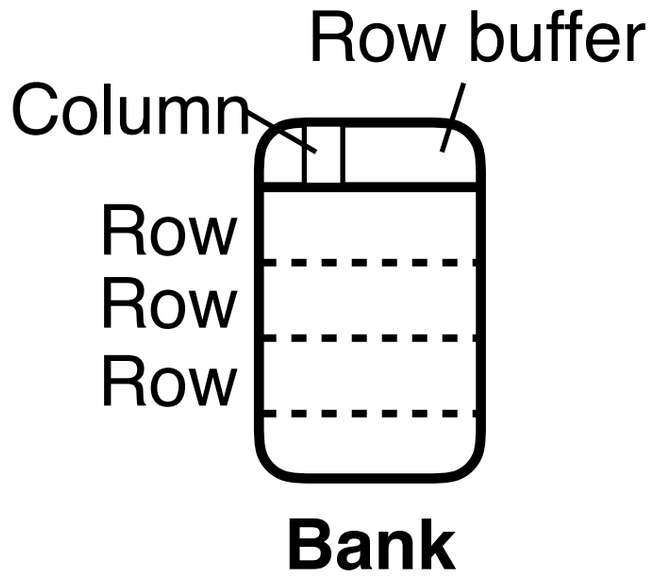
Multibank architecture



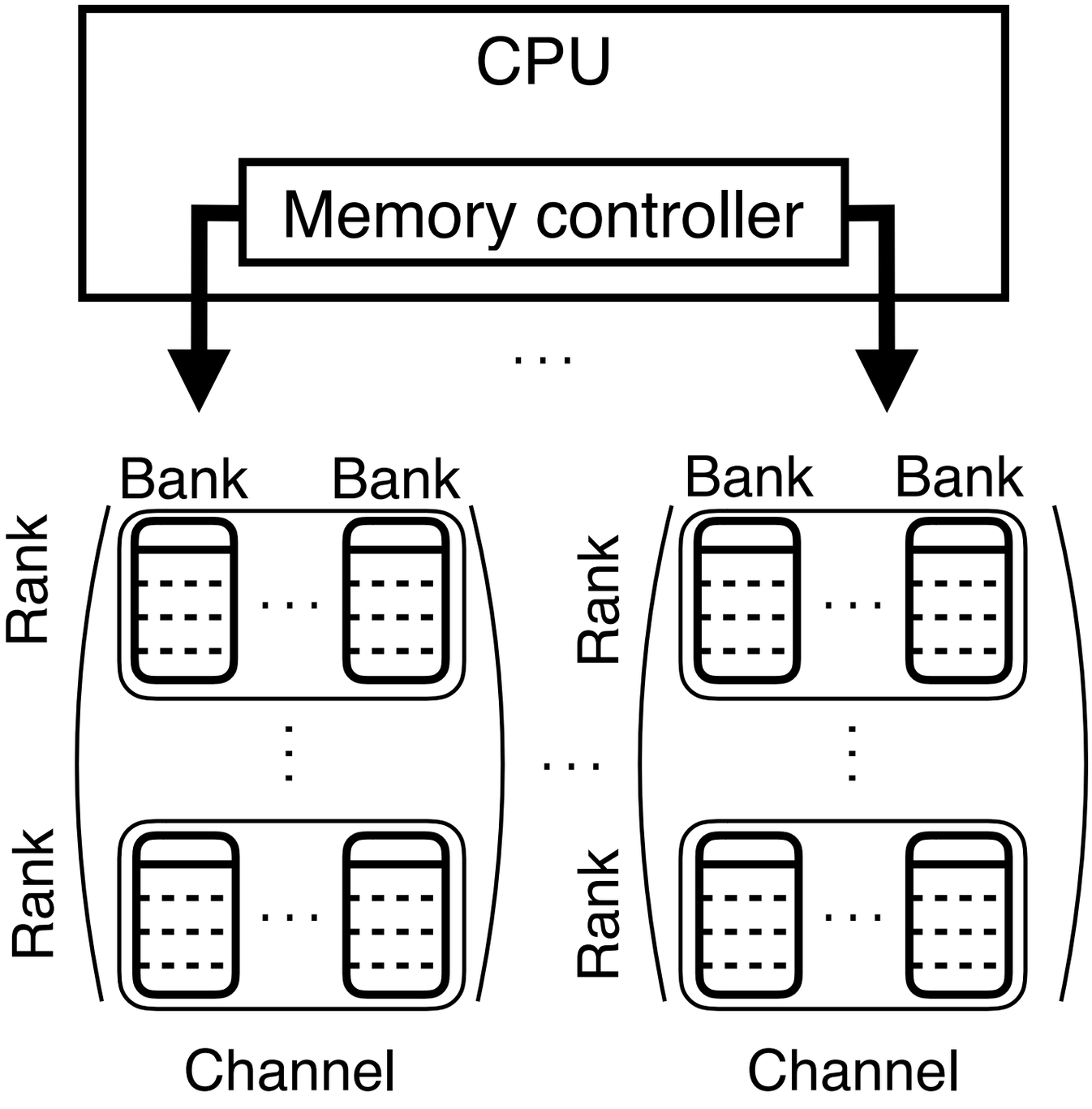


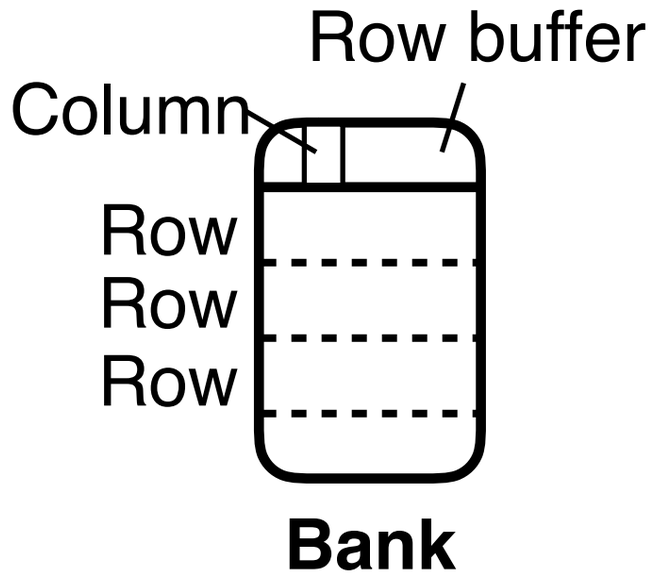
Multibank architecture





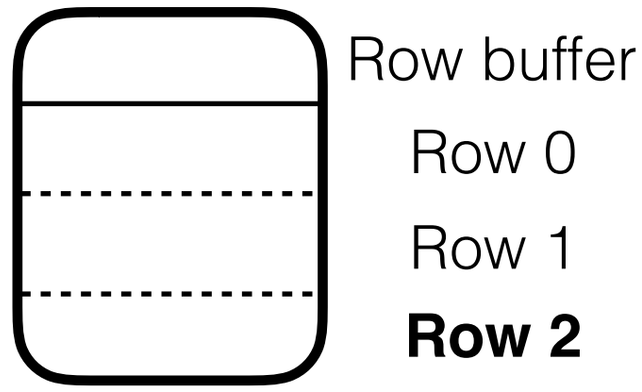
Multibank architecture



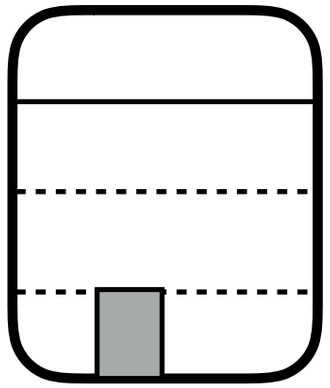


Multiple banks can be accessed in parallel
→ ***bank parallelism***

Three Types of Row Buffer Accesses

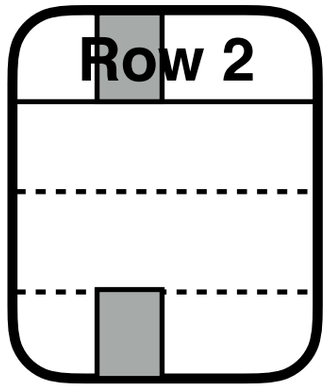


Three Types of Row Buffer Accesses



Access to data in **row 2**

Three Types of Row Buffer Accesses



Row buffer

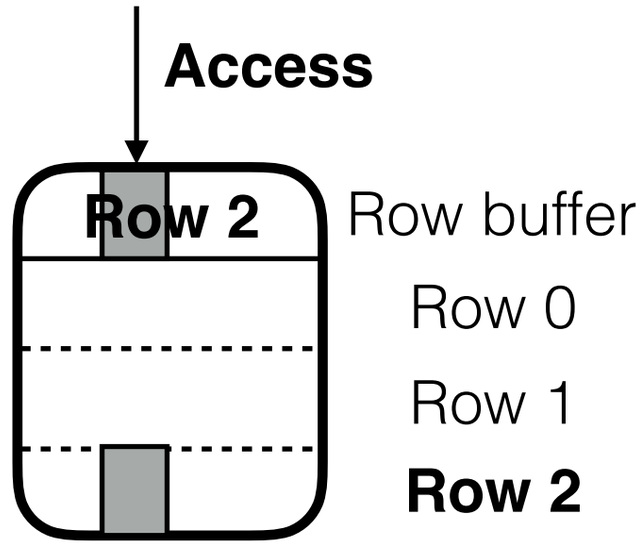
Row 0

Row 1

Row 2

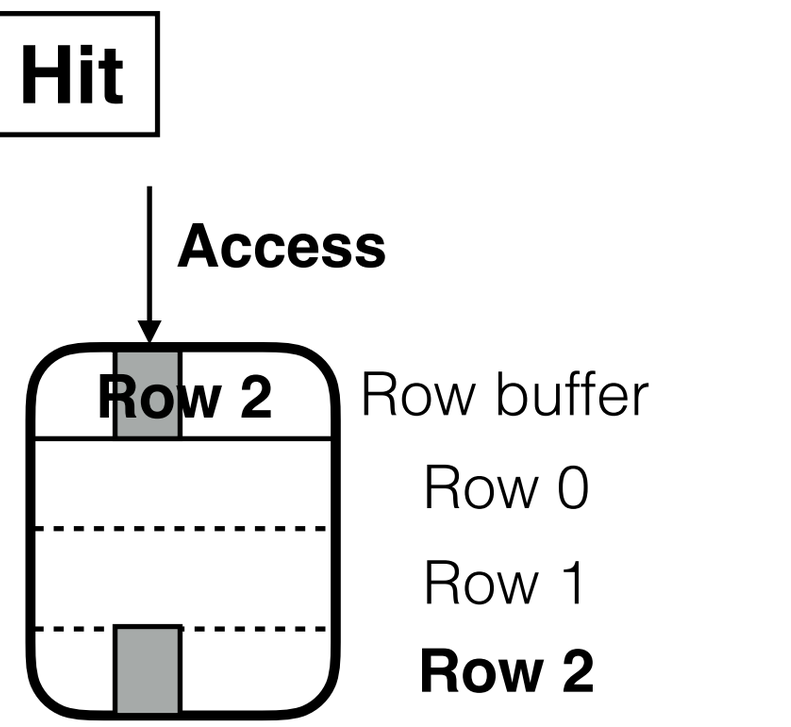
Access to data in **row 2**

Three Types of Row Buffer Accesses



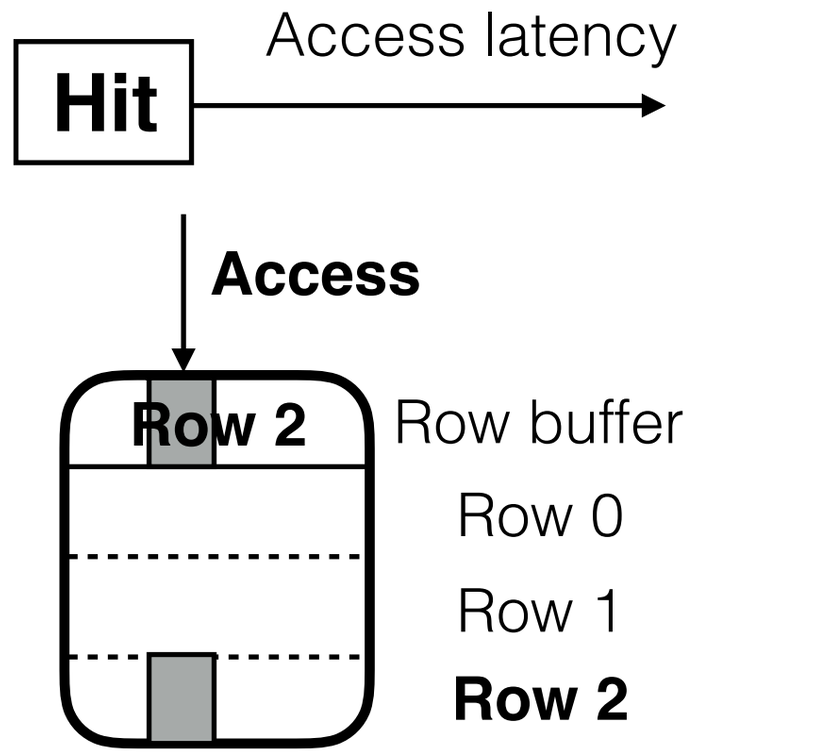
Access to data in **row 2**

Three Types of Row Buffer Accesses



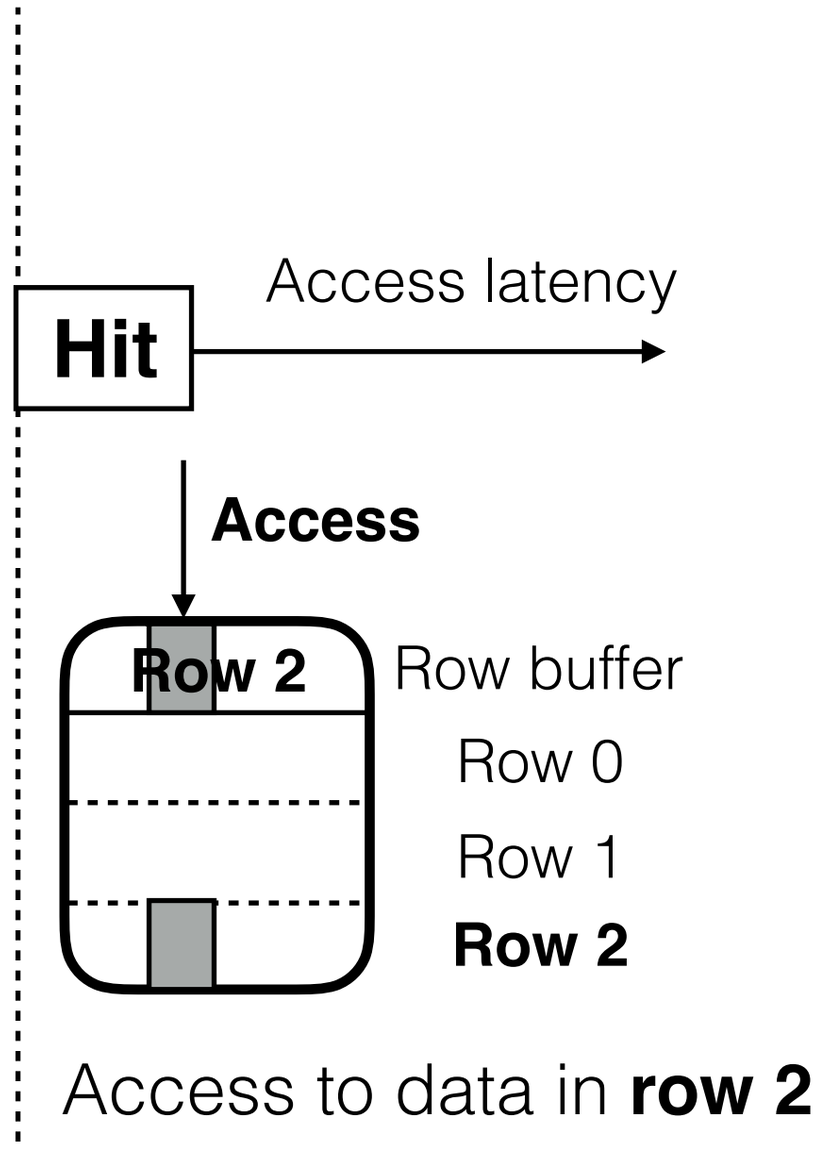
Access to data in **row 2**

Three Types of Row Buffer Accesses

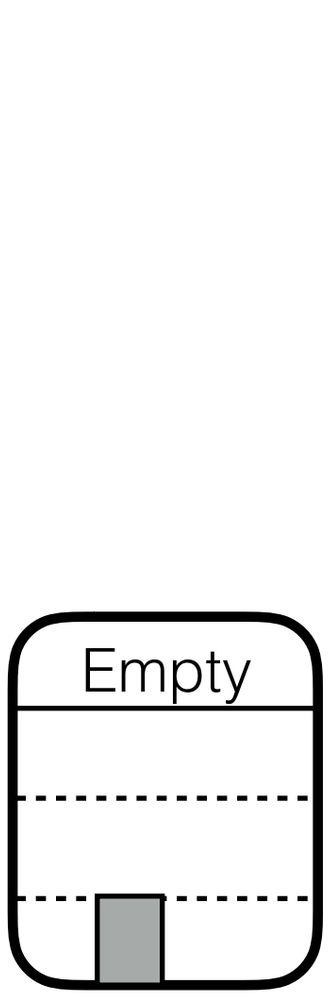


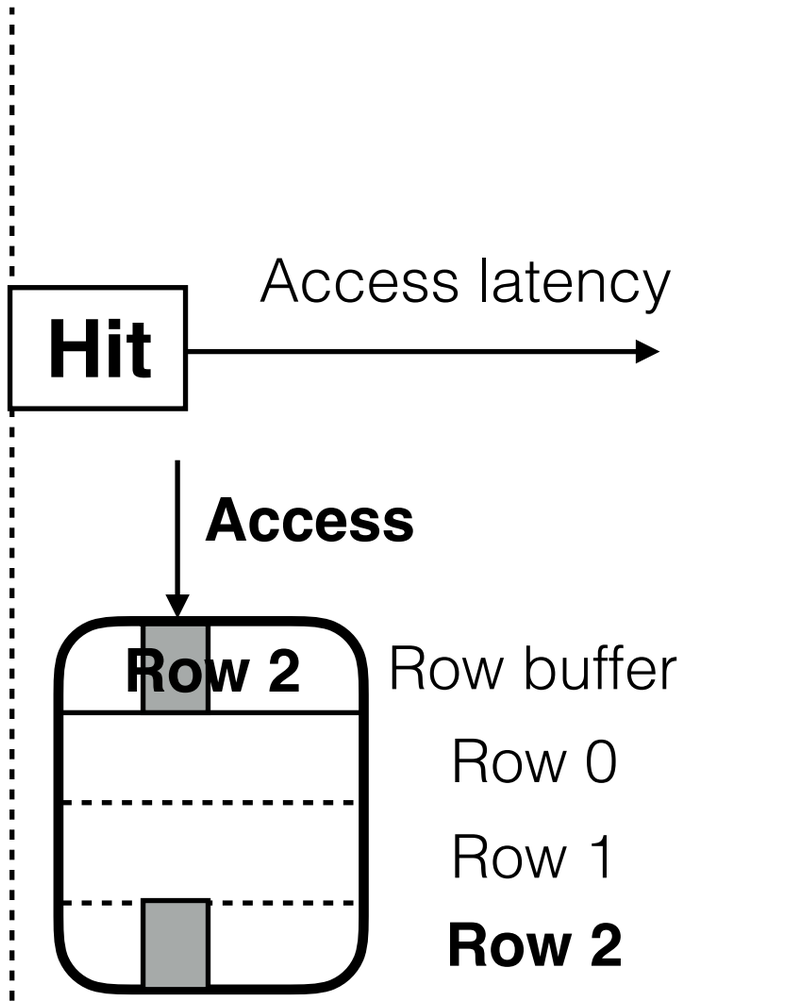
Access to data in **row 2**

Three Types of Row Buffer Accesses



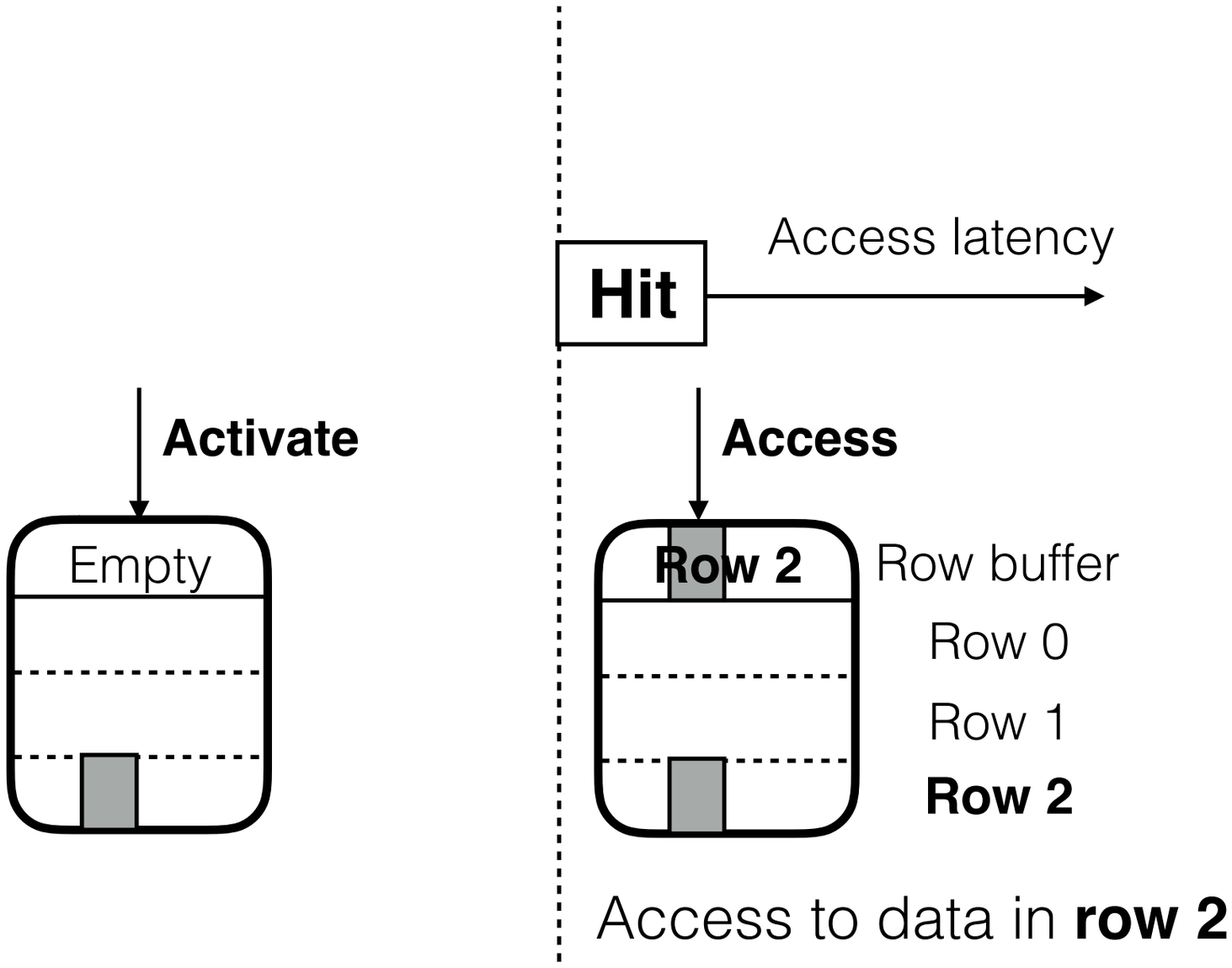
Three Types of Row Buffer Accesses



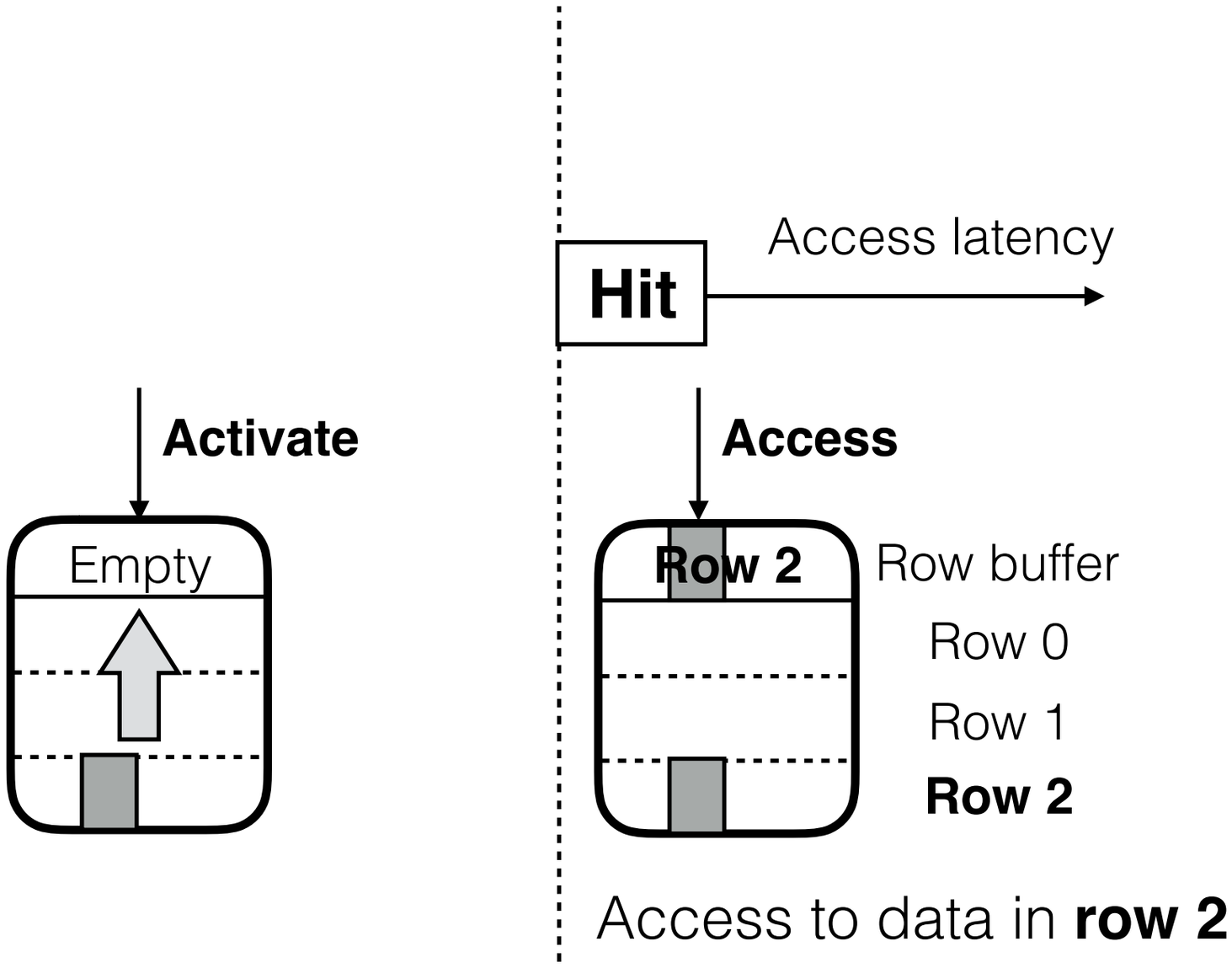


Access to data in **row 2**

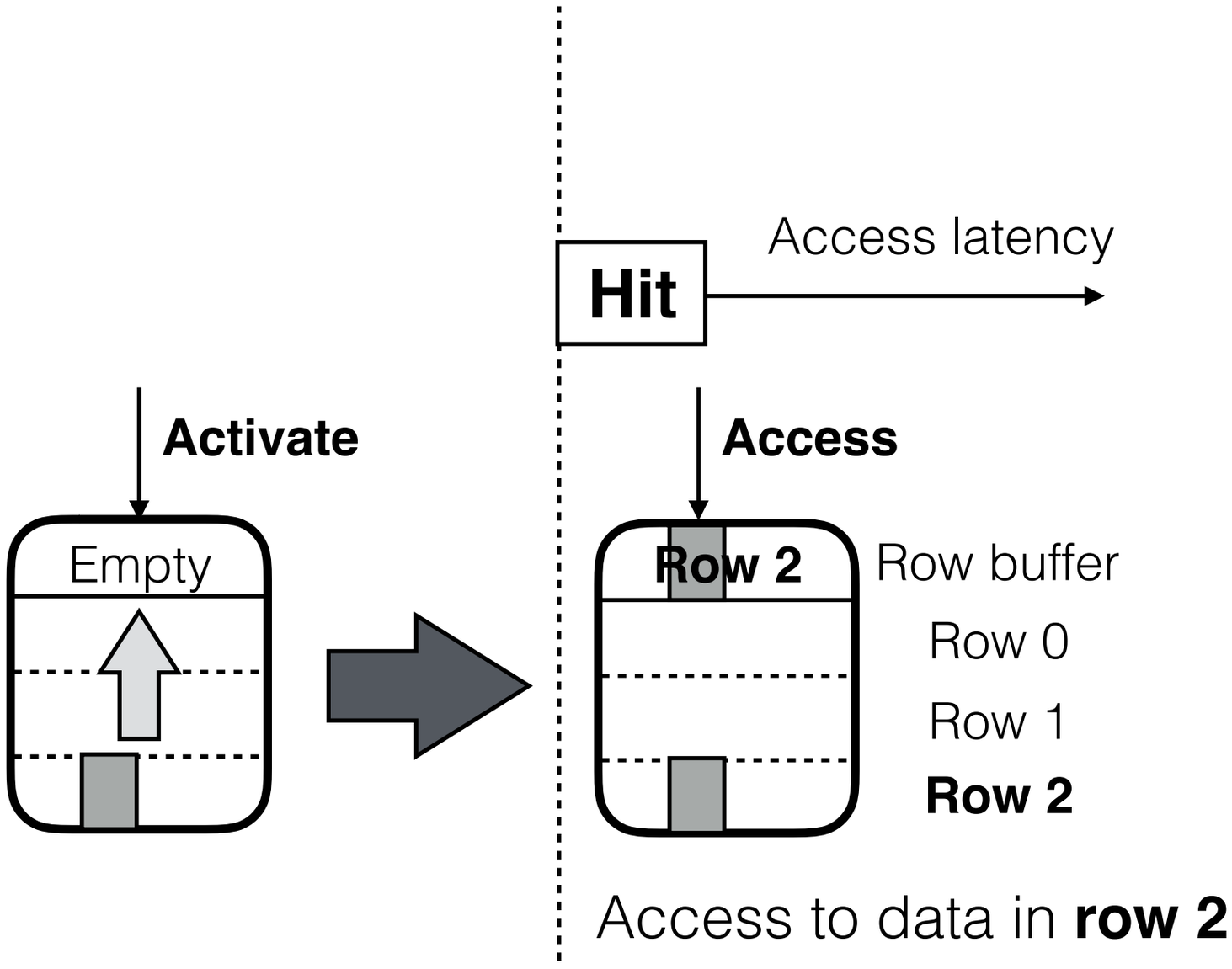
Three Types of Row Buffer Accesses



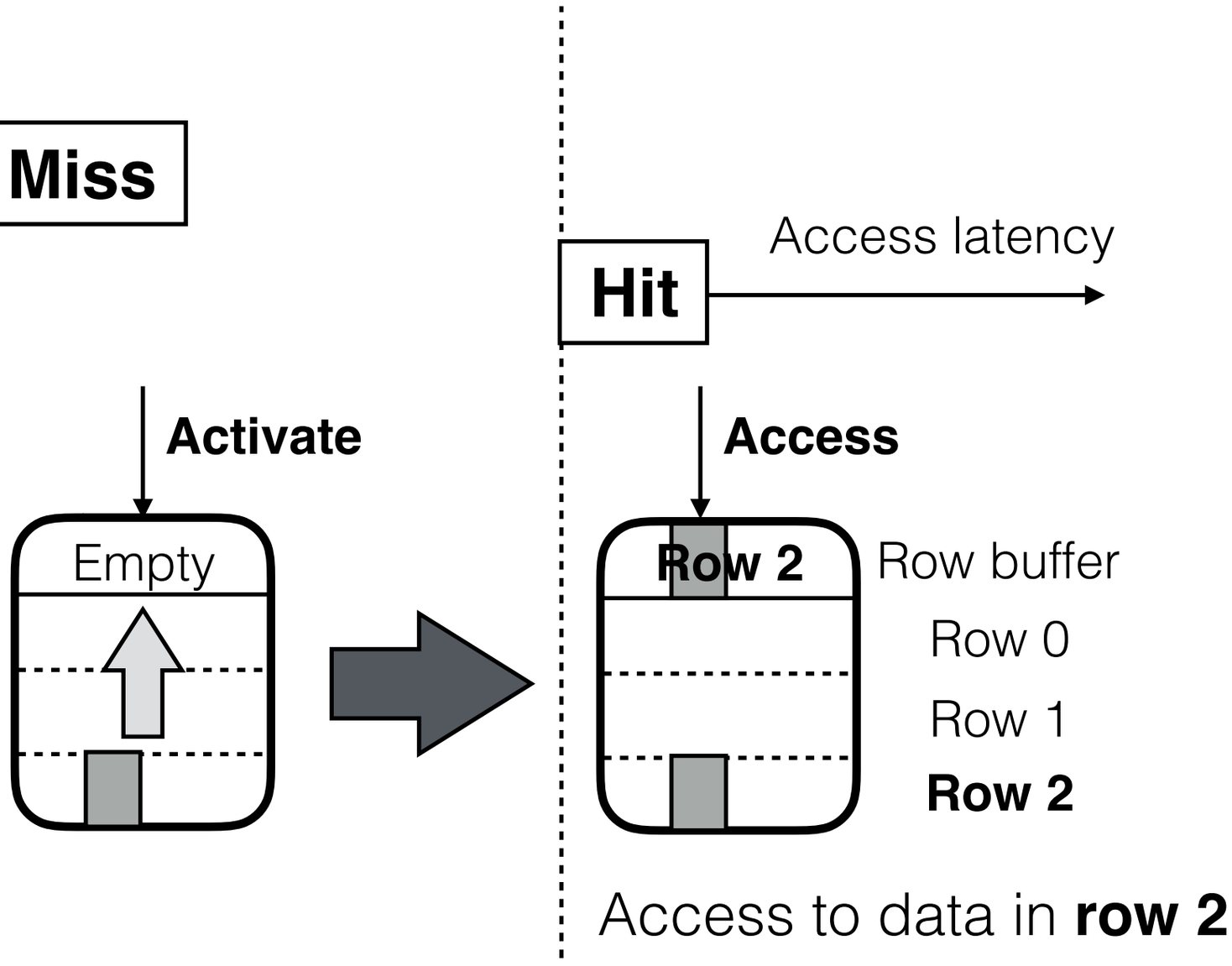
Three Types of Row Buffer Accesses



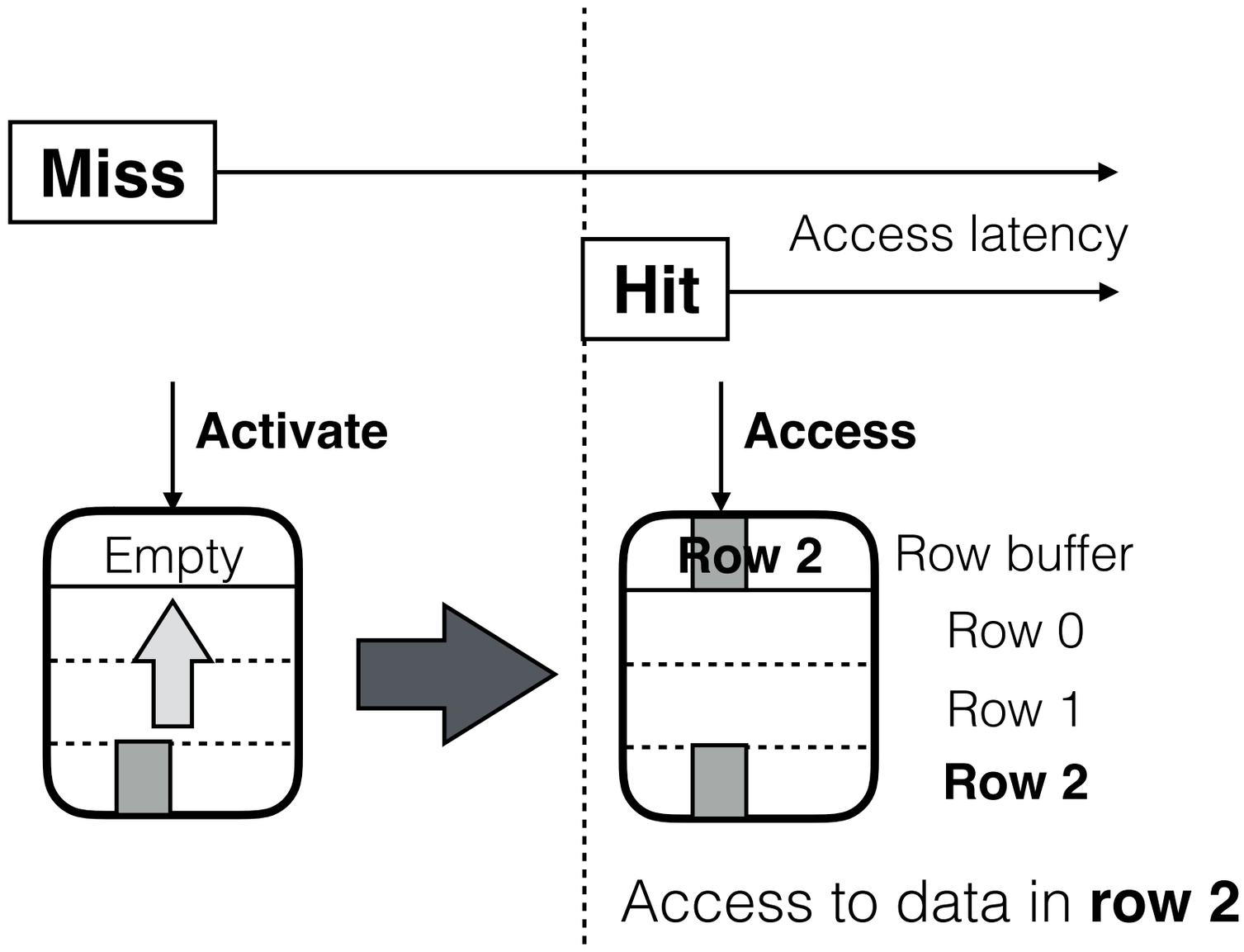
Three Types of Row Buffer Accesses



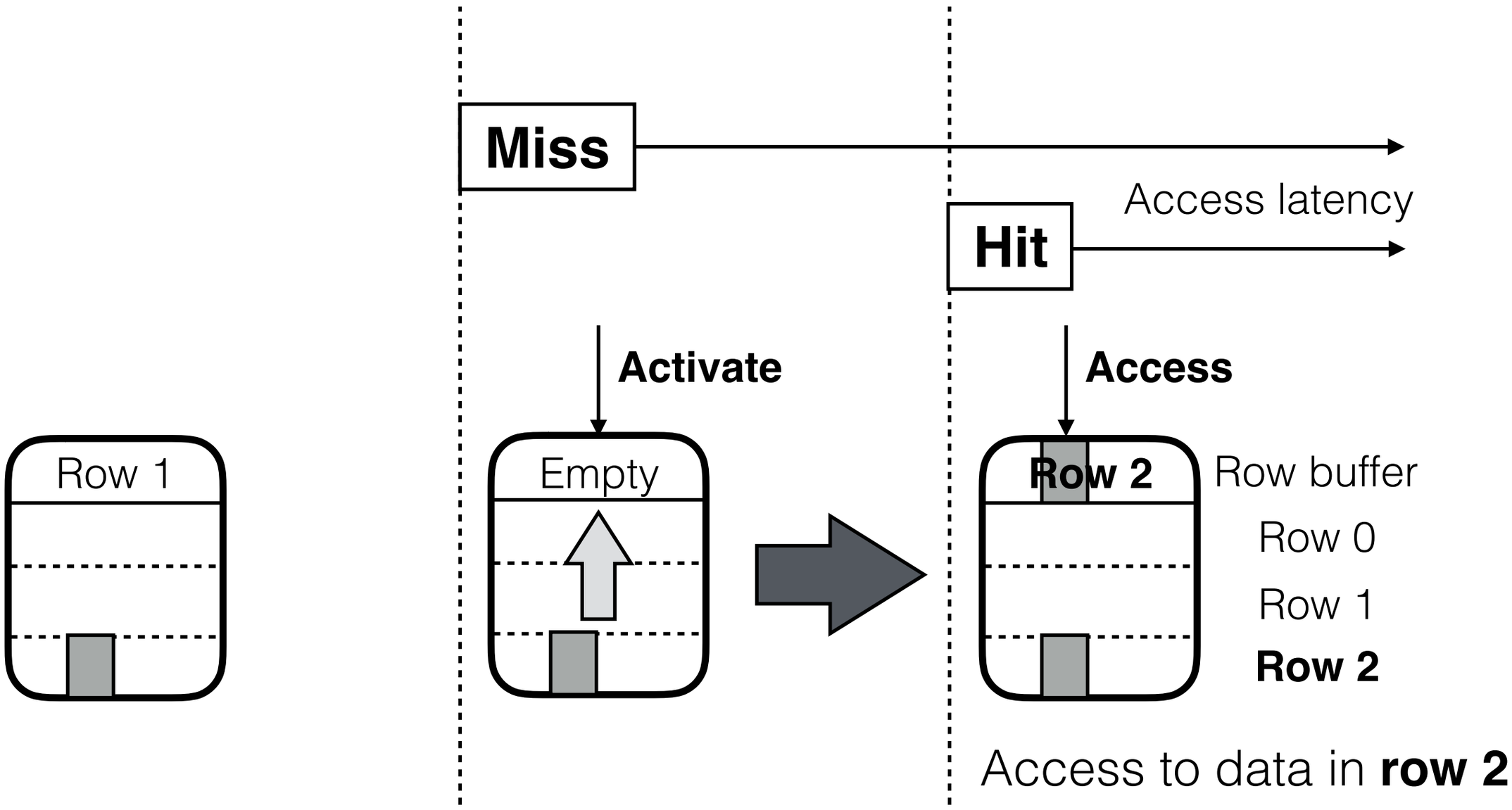
Three Types of Row Buffer Accesses



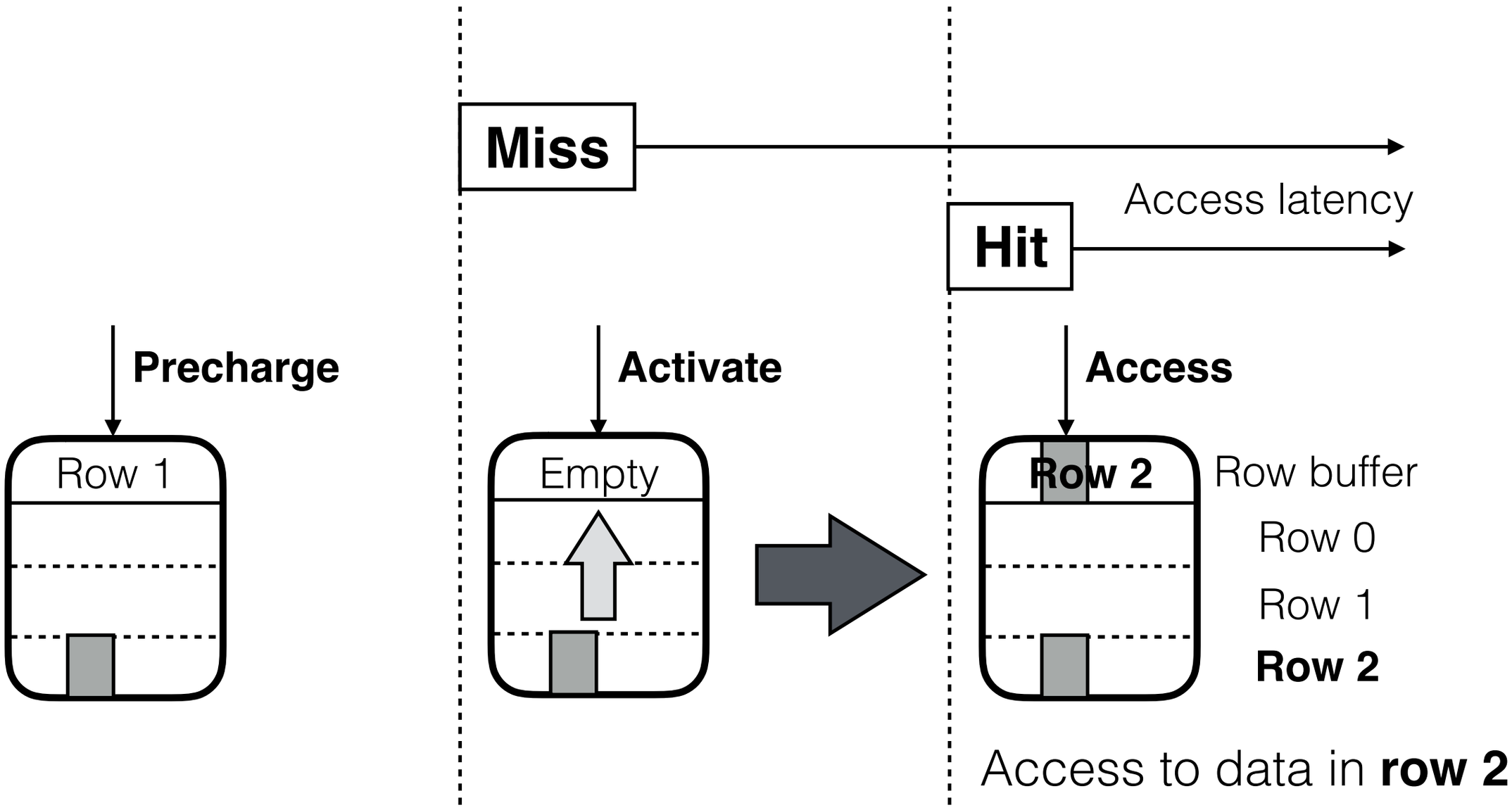
Three Types of Row Buffer Accesses



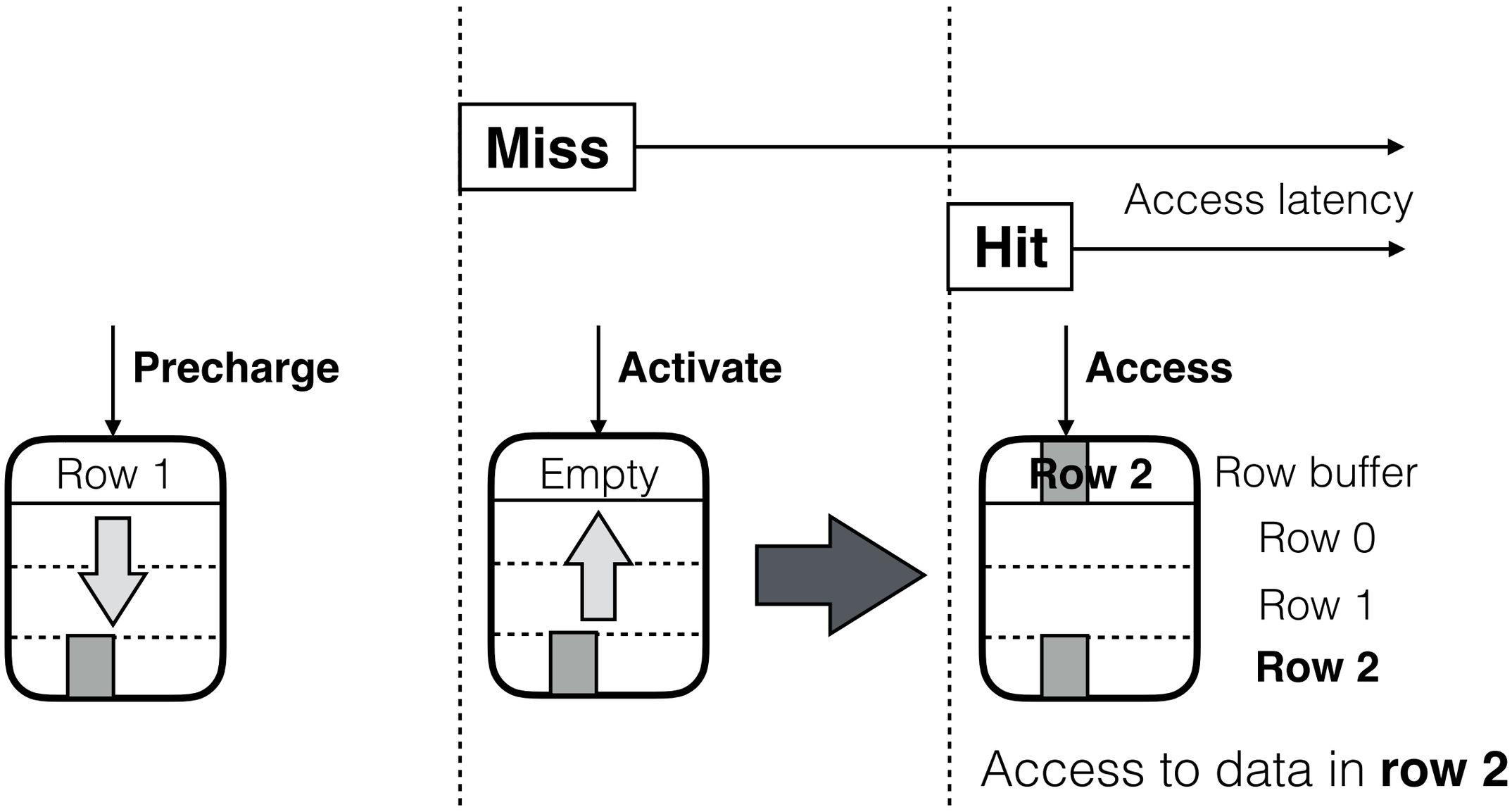
Three Types of Row Buffer Accesses



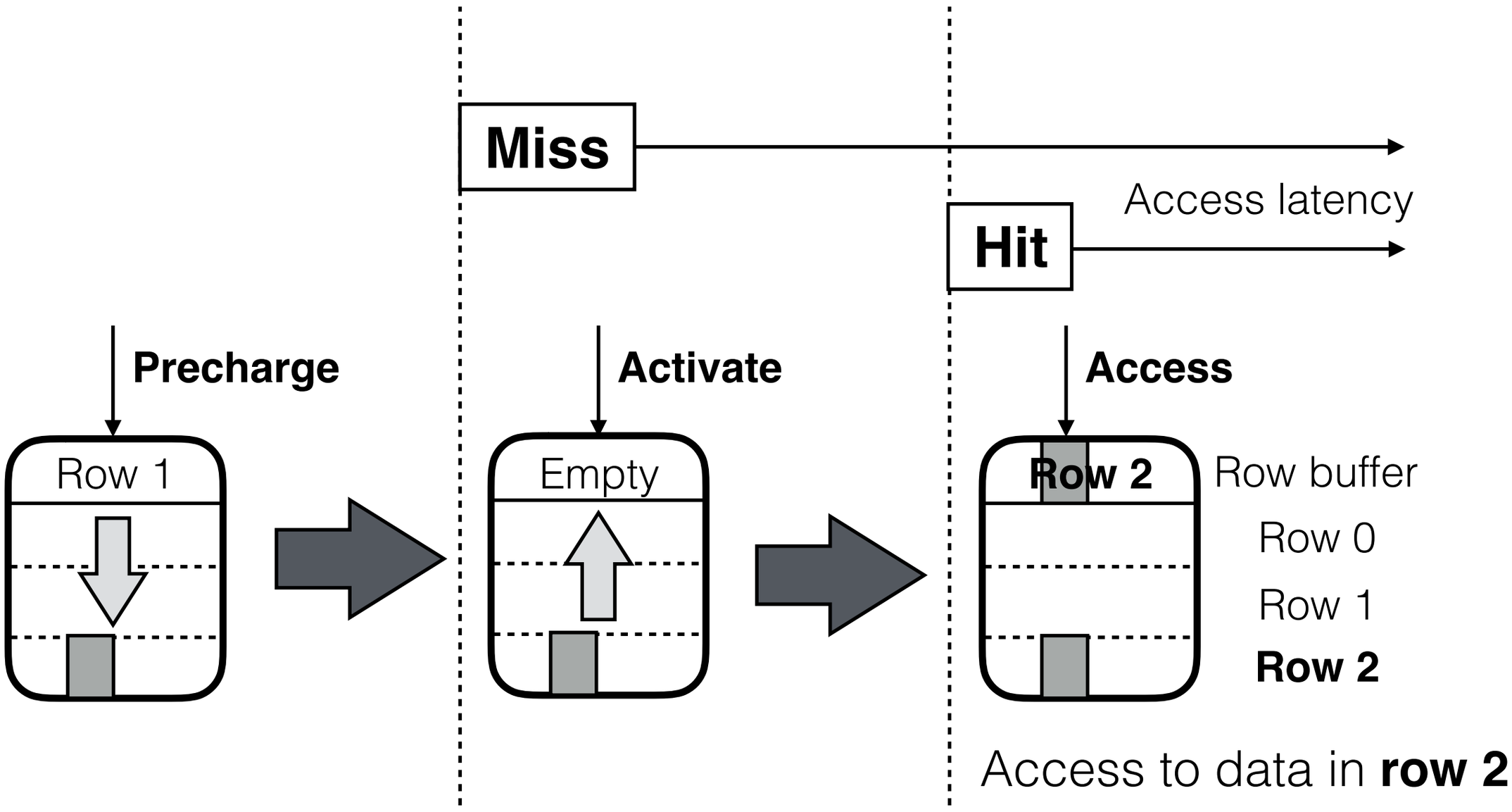
Three Types of Row Buffer Accesses



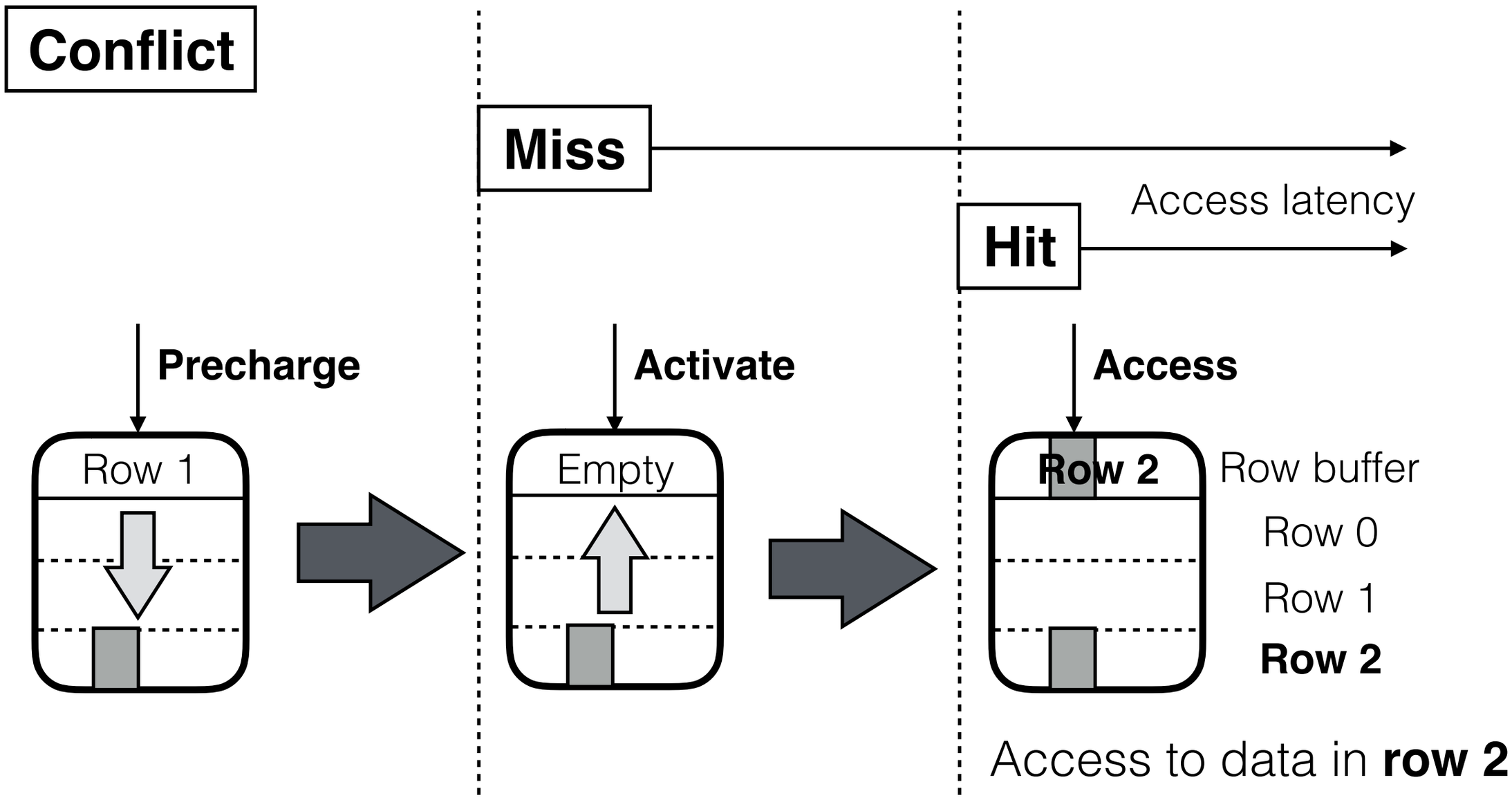
Three Types of Row Buffer Accesses



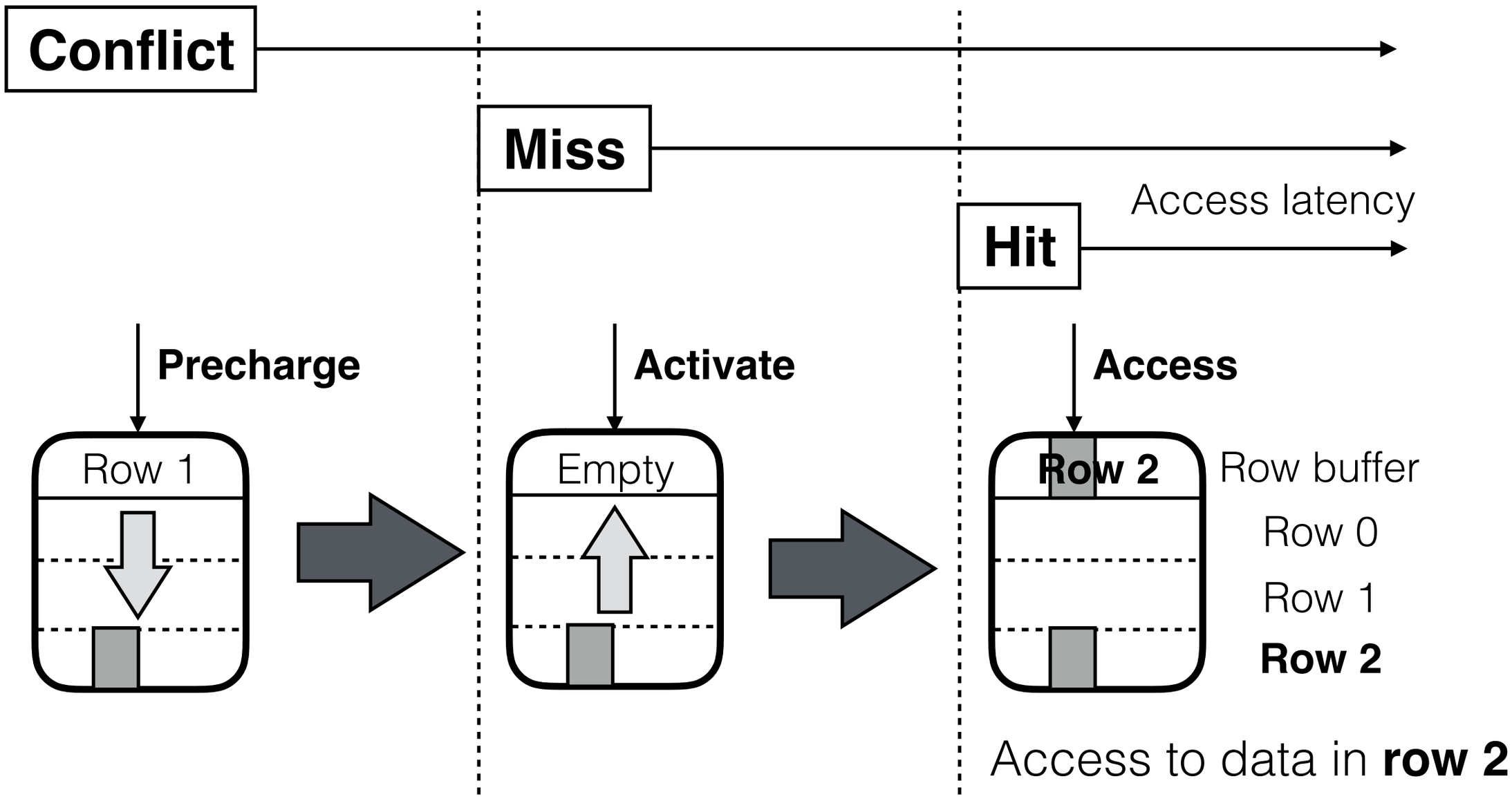
Three Types of Row Buffer Accesses



Three Types of Row Buffer Accesses



Three Types of Row Buffer Accesses



Row Buffer Management Polices

- Open-page policy
  - Row buffers are NOT precharged (closed) after accessed
    - ✓ Pros: accesses to the same row become “hits”
    - ✓ Cons: accesses to different rows become “conflicts”

Row Buffer Management Polices

- Open-page policy
  - Row buffers are NOT precharged (closed) after accessed
    - ✓ Pros: accesses to the same row become “hits”
    - ✓ Cons: accesses to different rows become “conflicts”

- Closed-page policy
  - Row buffers are closed immediately after accessed
    - ✓ Pros/Cons: all accesses become “misses”

Row Buffer Management Polices

- Open-page policy
  - Row buffers are NOT precharged (closed) after accessed
    - ✓ Pros: accesses to the same row become “hits”
    - ✓ Cons: accesses to different rows become “conflicts”
- Closed-page policy
  - Row buffers are closed immediately after accessed
    - ✓ Pros/Cons: all accesses become “misses”
- AMD’s adaptive open-page policy
  - Row buffers are closed after 128 cycles from the last access

Row Buffer Management Polices

- Open-page policy
  - Row buffers are NOT precharged (closed) after accessed
    - ✓ Pros: accesses to the same row become “hits”
    - ✓ Cons: accesses to different rows become “conflicts”
- Closed-page policy
  - Row buffers are closed immediately after accessed
    - ✓ Pros/Cons: all accesses become “misses”

Used in this work

- AMD’s adaptive open-page policy
  - Row buffers are closed after 128 cycles from the last access

Agenda

- State-of-the-art BFS implementation
- DRAM mechanisms
- Memory access analysis with conventional address mapping schemes
- Proposed: per-row channel interleaving
- Evaluation of power efficiency

Address Mapping Schemes

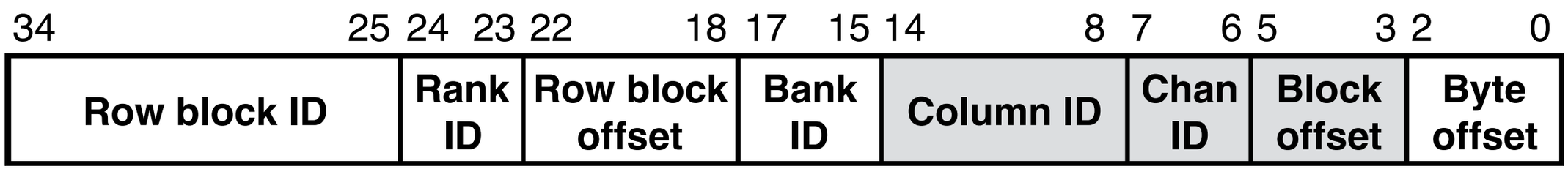
- Determine the location of data in DRAM based on a physical address
- Implemented in memory controllers

Address Mapping Schemes

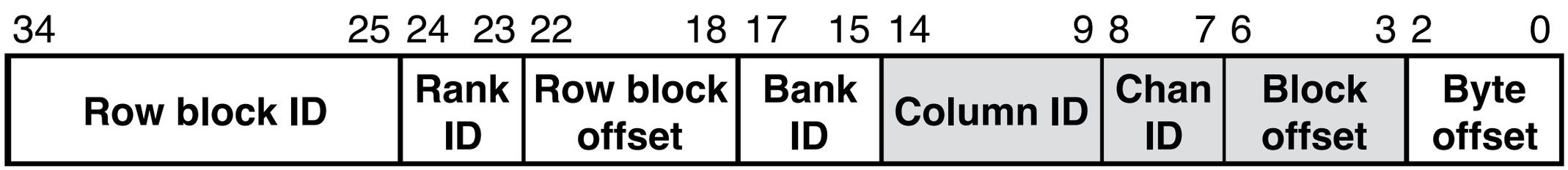
- Determine the location of data in DRAM based on a physical address
- Implemented in memory controllers
- Conventional schemes:
  - Per-cache-line channel interleaving (PCL)
    - ✓ 64 B blocks are interleaved across channels
    - ✓ Applied to Intel Nehalem processors [Park+, ASPLOS ’13]
  - Per-2-cache-line channel interleaving (P2CL)
    - ✓ 128 B blocks are interleaved across channels
    - ✓ Applied to Intel SandyBridge and Haswell processors

Breakdown of Physical Addresses

Per-cache-line channel interleaving (PCL)

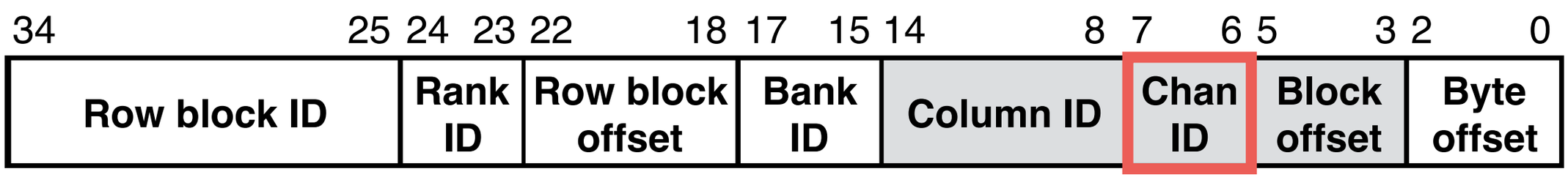


Per-2-cache-lines channel interleaving (P2CL)

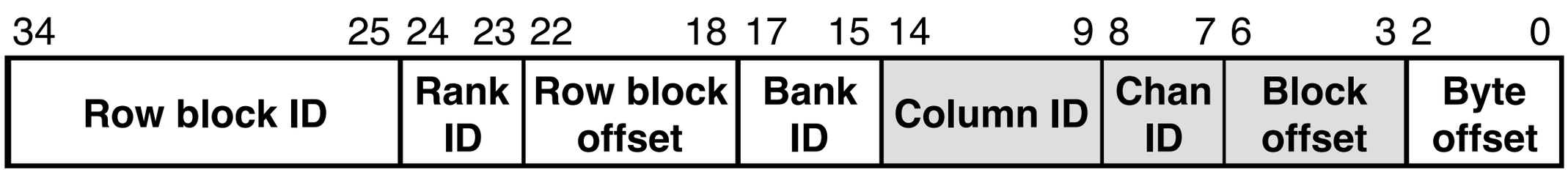


Breakdown of Physical Addresses

Per-cache-line channel interleaving (PCL)



Per-2-cache-lines channel interleaving (P2CL)



Breakdown of Physical Addresses

Per-cache-line channel interleaving (PCL)

6 bits → 64 B

34	25	24	23	22	18	17	15	14	8	7	6	5	3	2	0
Row block ID		Rank ID		Row block offset		Bank ID		Column ID		Chan ID		Block offset		Byte offset	

Per-2-cache-lines channel interleaving (P2CL)

34	25	24	23	22	18	17	15	14	9	8	7	6	3	2	0
Row block ID		Rank ID		Row block offset		Bank ID		Column ID		Chan ID		Block offset		Byte offset	

Breakdown of Physical Addresses

Per-cache-line channel interleaving (PCL)

6 bits → 64 B

34	25	24	23	22	18	17	15	14	8	7	6	5	3	2	0
Row block ID		Rank ID		Row block offset		Bank ID		Column ID		Chan ID		Block offset		Byte offset	

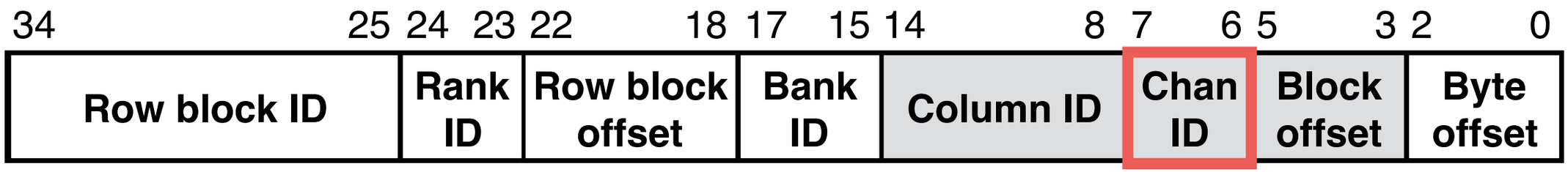
Per-2-cache-lines channel interleaving (P2CL)

34	25	24	23	22	18	17	15	14	9	8	7	6	3	2	0
Row block ID		Rank ID		Row block offset		Bank ID		Column ID		Chan ID		Block offset		Byte offset	

Breakdown of Physical Addresses

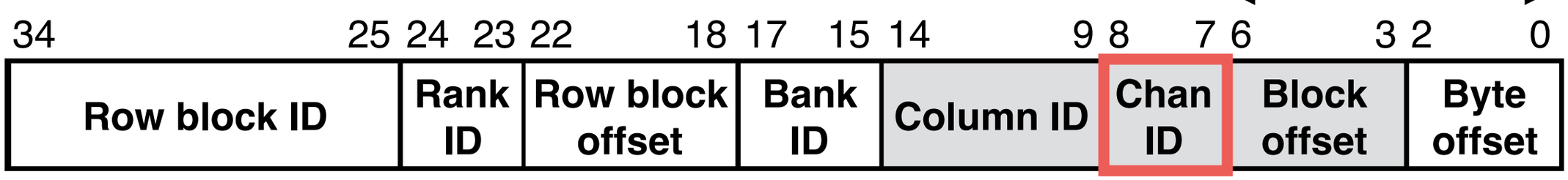
Per-cache-line channel interleaving (PCL)

6 bits → 64 B



Per-2-cache-lines channel interleaving (P2CL)

7 bits → 128 B



Simulator Setup

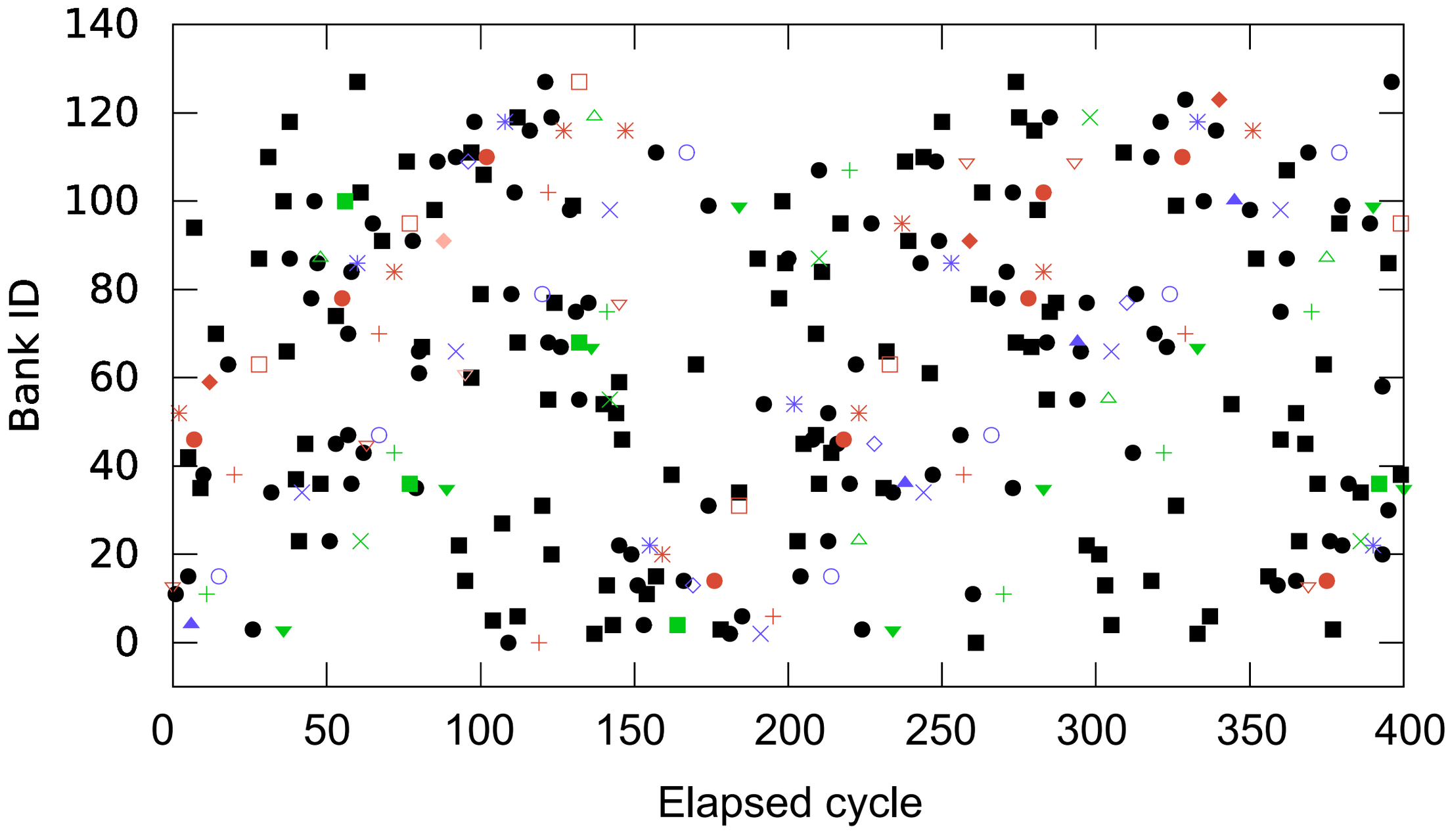
- Cycle-accurate multicore simulator: MARSSx86
- DRAM simulator: DRAMsim2

Simulator Setup

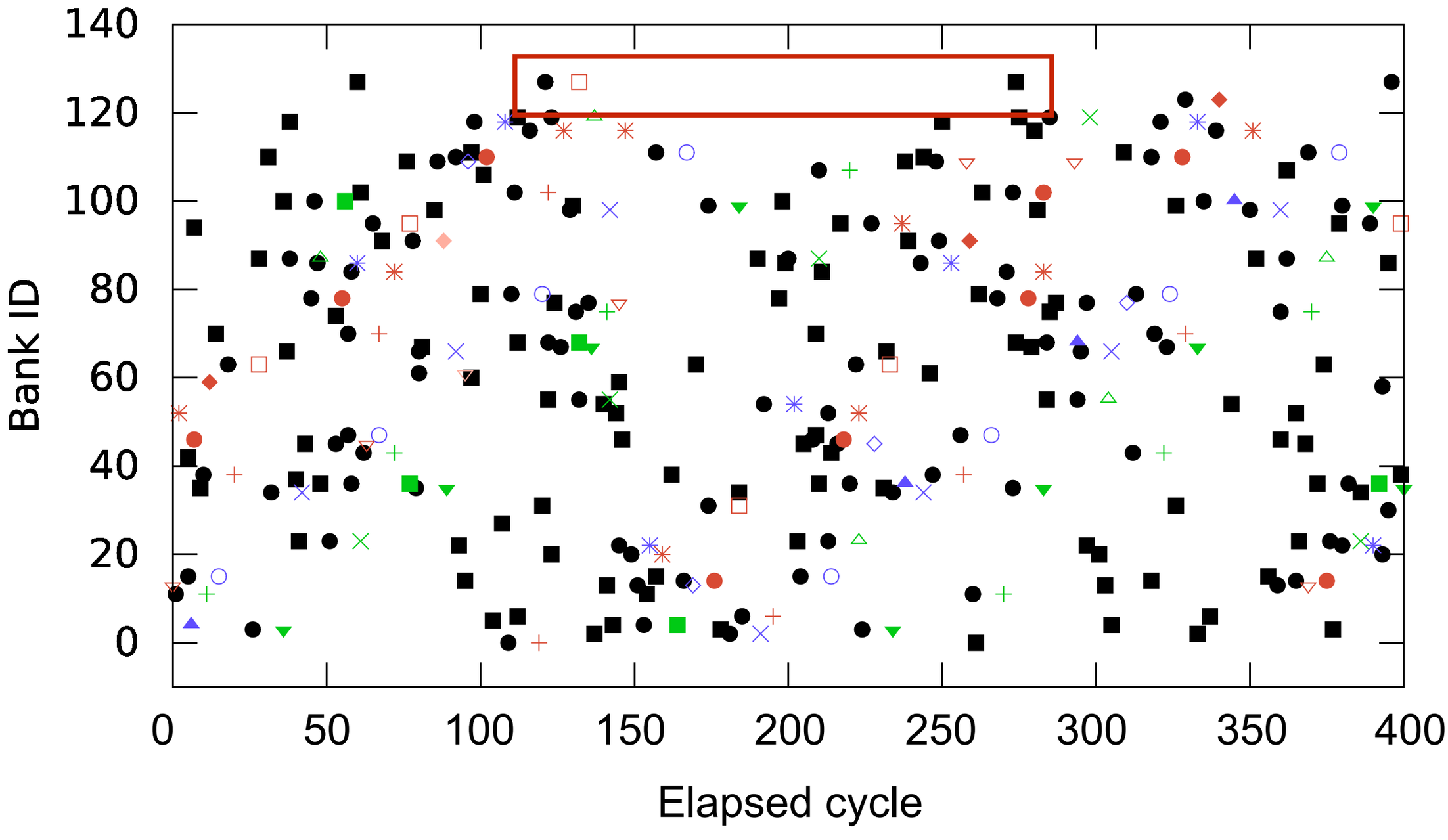
- Cycle-accurate multicore simulator: MARSSx86
- DRAM simulator: DRAMsim2

Parameter	Setting
Processor	**16 Out-of-Order cores**, 3.0 GHz
Cache	Private 32 KB L1 I/D, Private 256 KB L2 Shared 16 MB L3
Memory Controller	**Adaptive open-page policy (128 cycles)** FR-FCFS scheduling policy [Rixner+, ISCA'00]
DRAM	**32 GB, DDR3-1333, 8 KB row** **4 channels, 4 ranks/channel, 8 banks/rank (128 banks)**

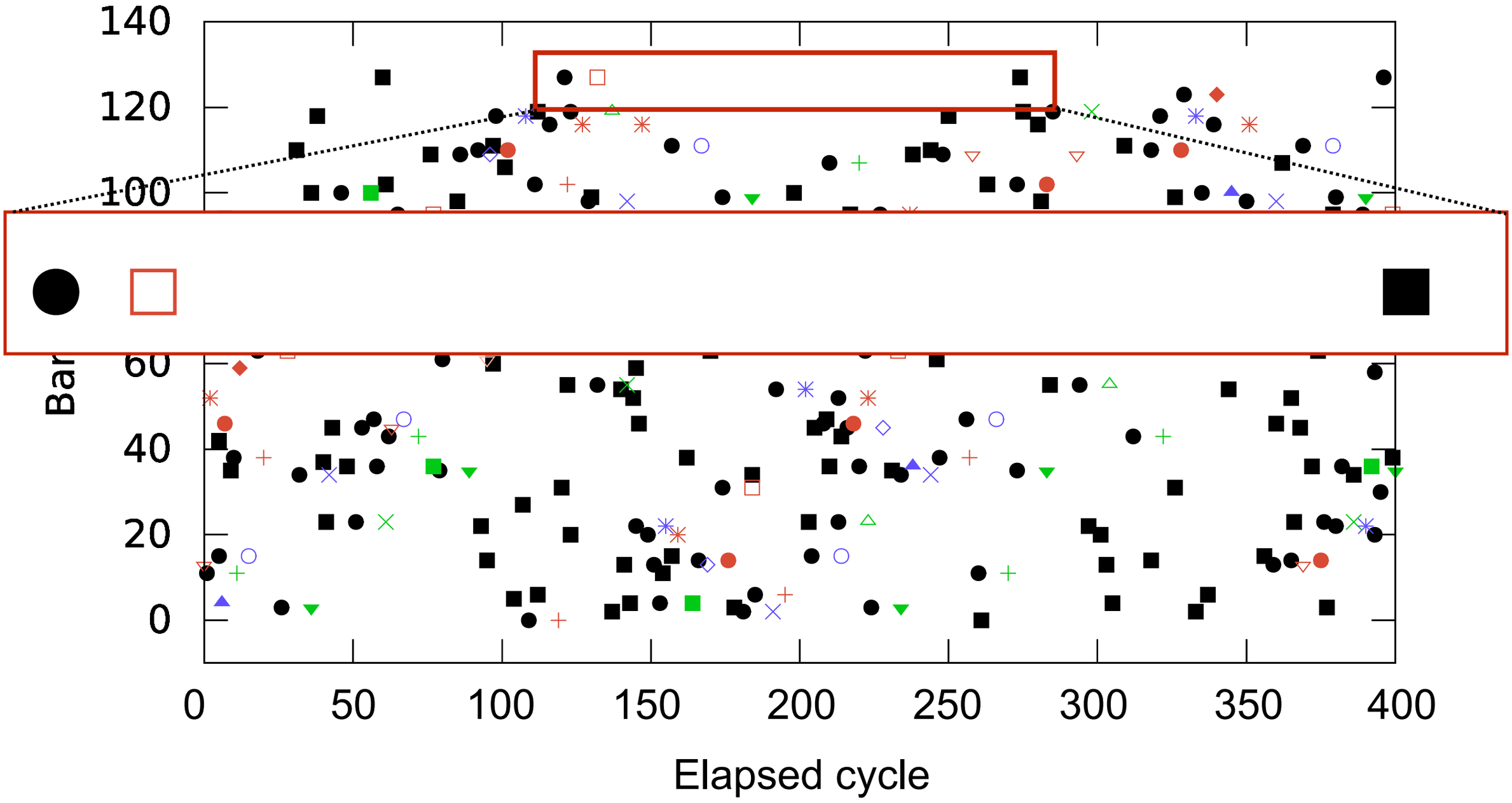
Memory Access Trace of Bottom-up with PCL



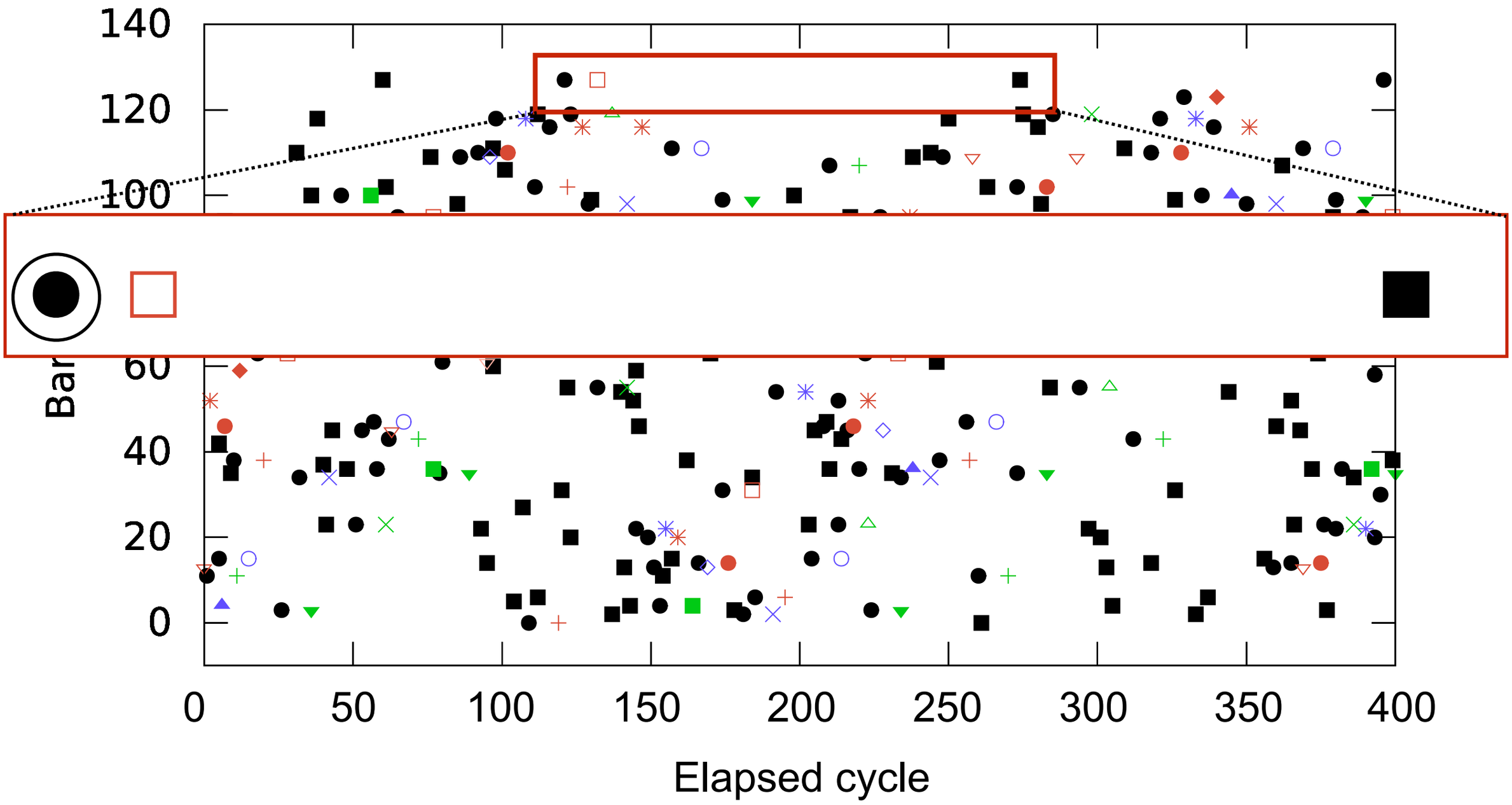
Memory Access Trace of Bottom-up with PCL



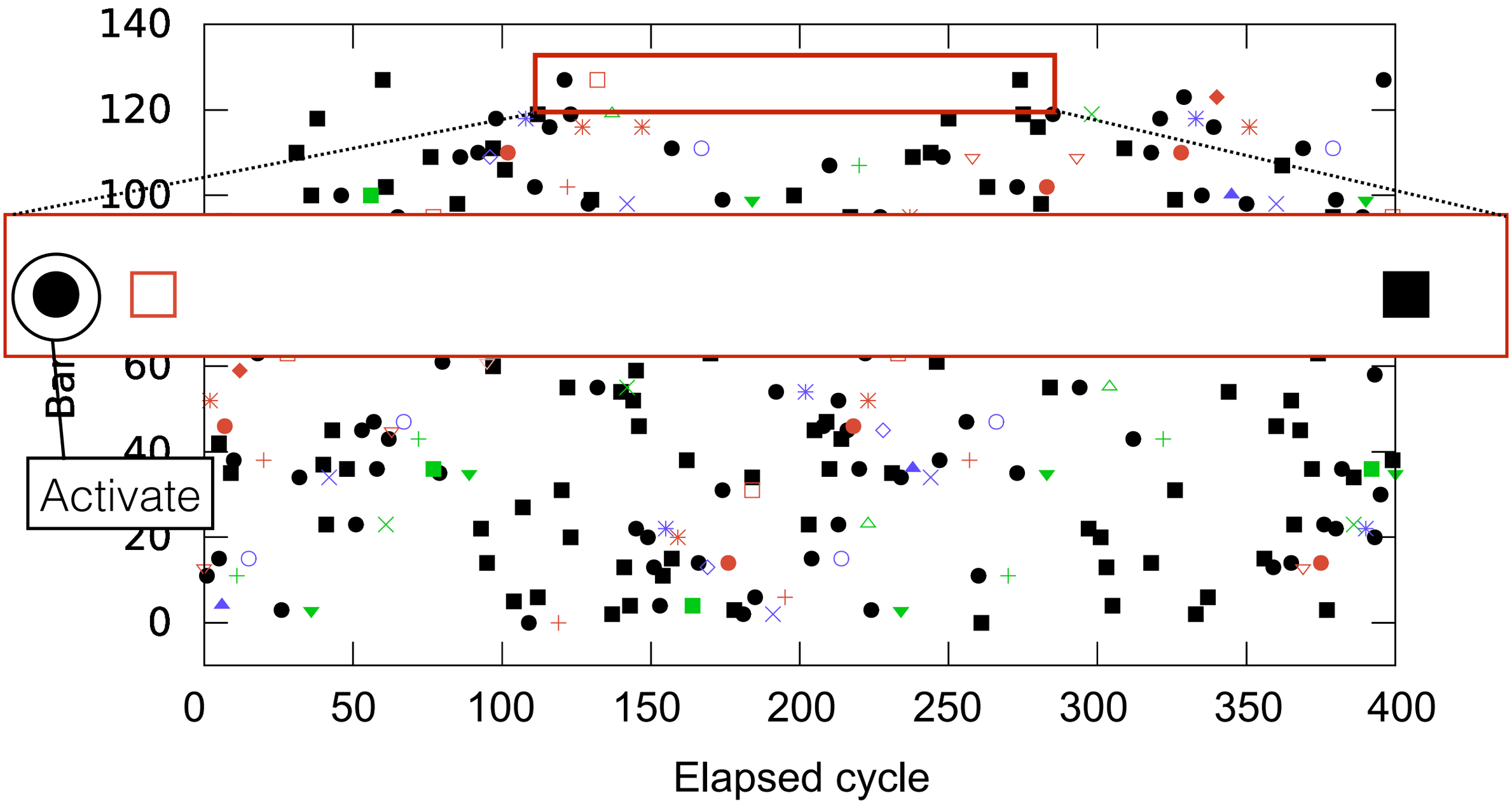
Memory Access Trace of Bottom-up with PCL



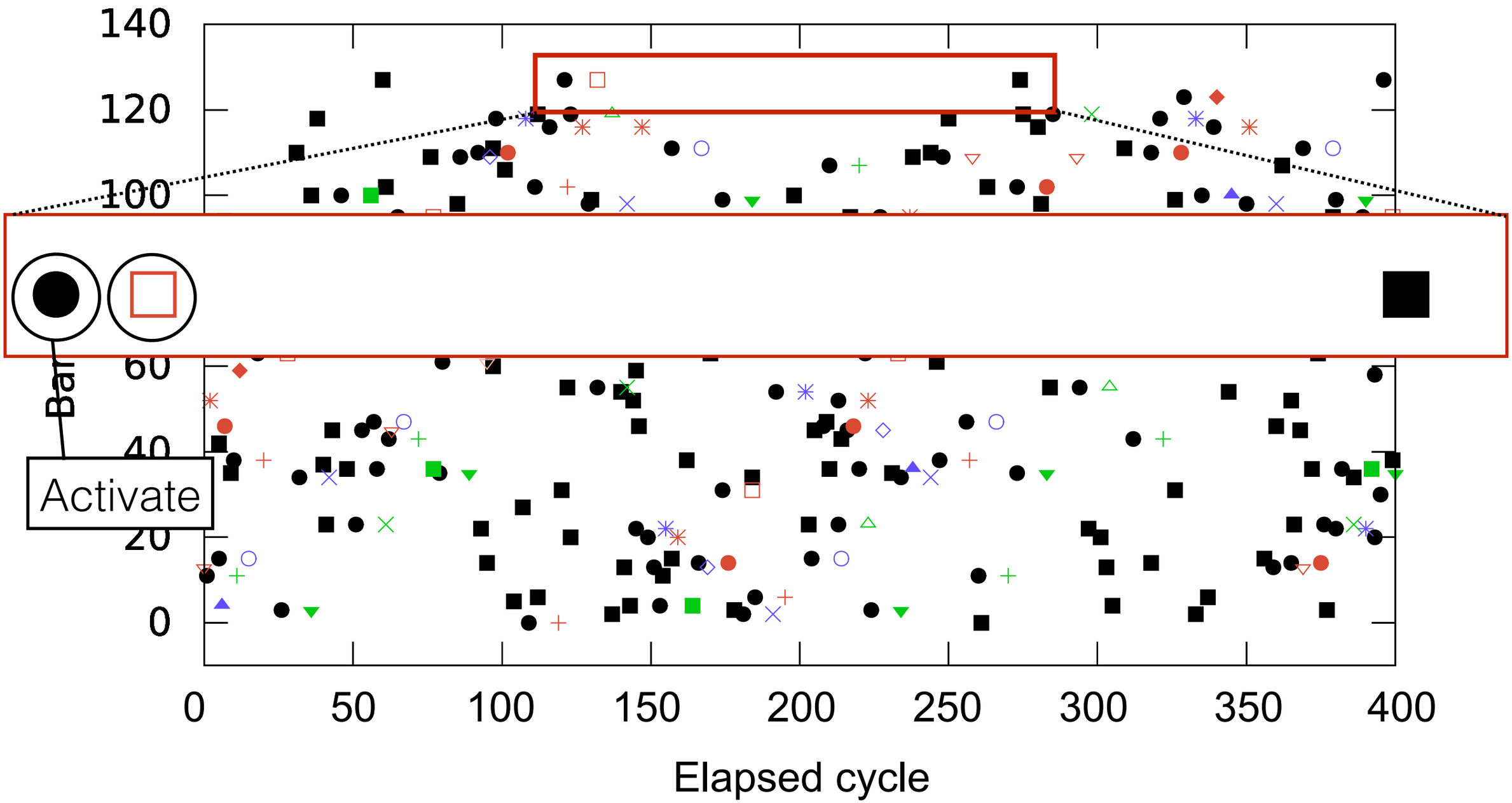
Memory Access Trace of Bottom-up with PCL



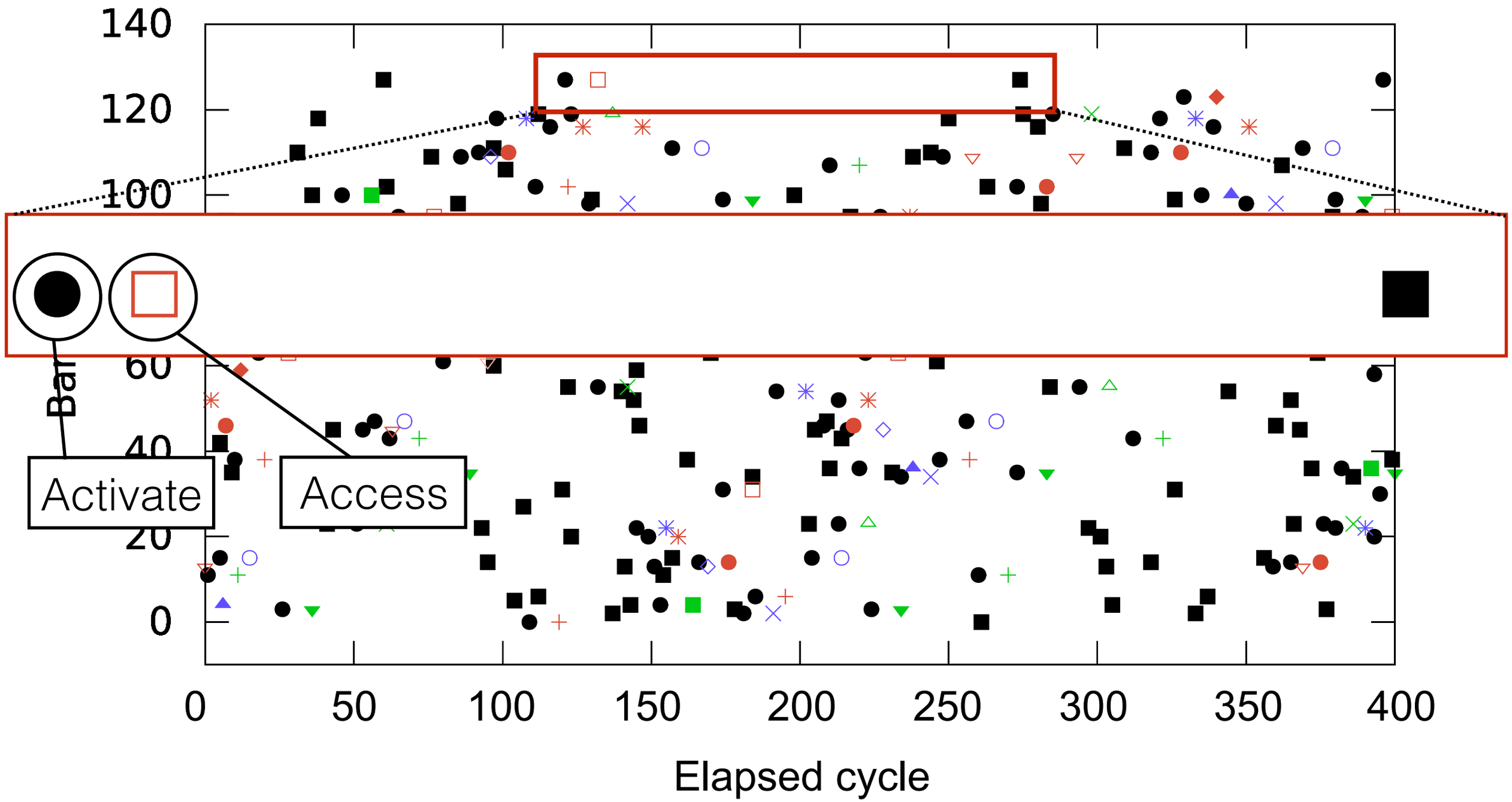
Memory Access Trace of Bottom-up with PCL



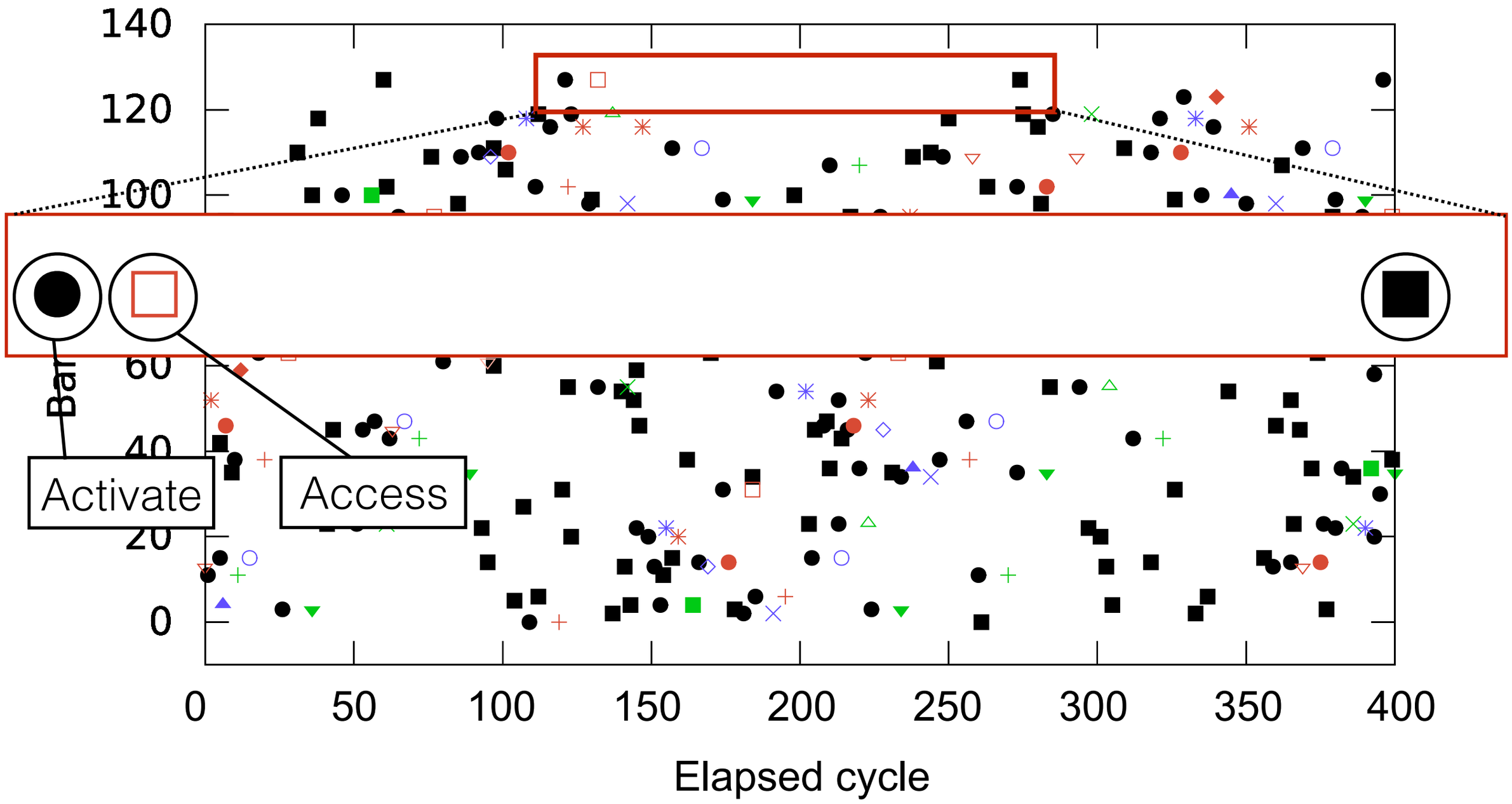
Memory Access Trace of Bottom-up with PCL



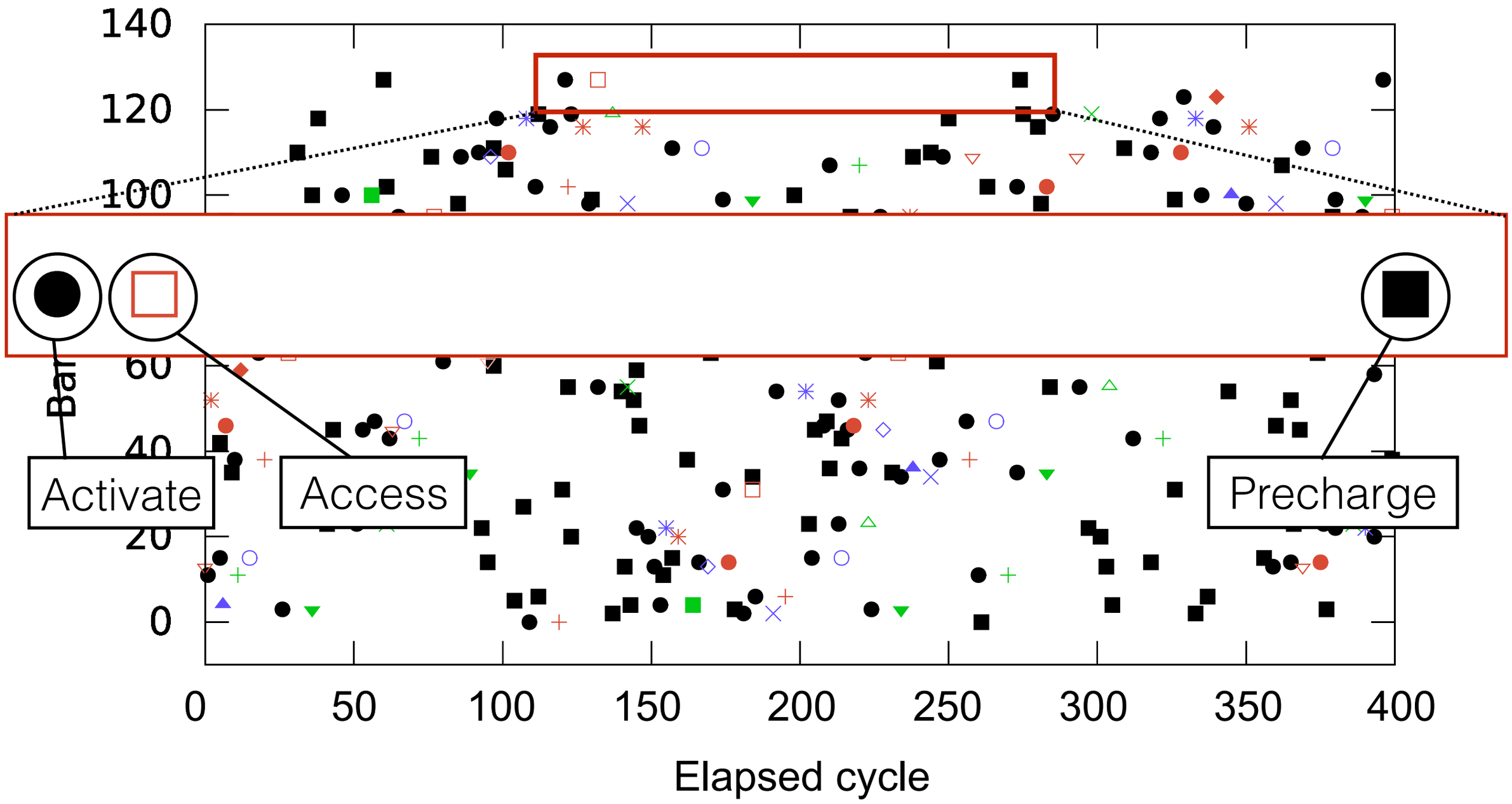
Memory Access Trace of Bottom-up with PCL



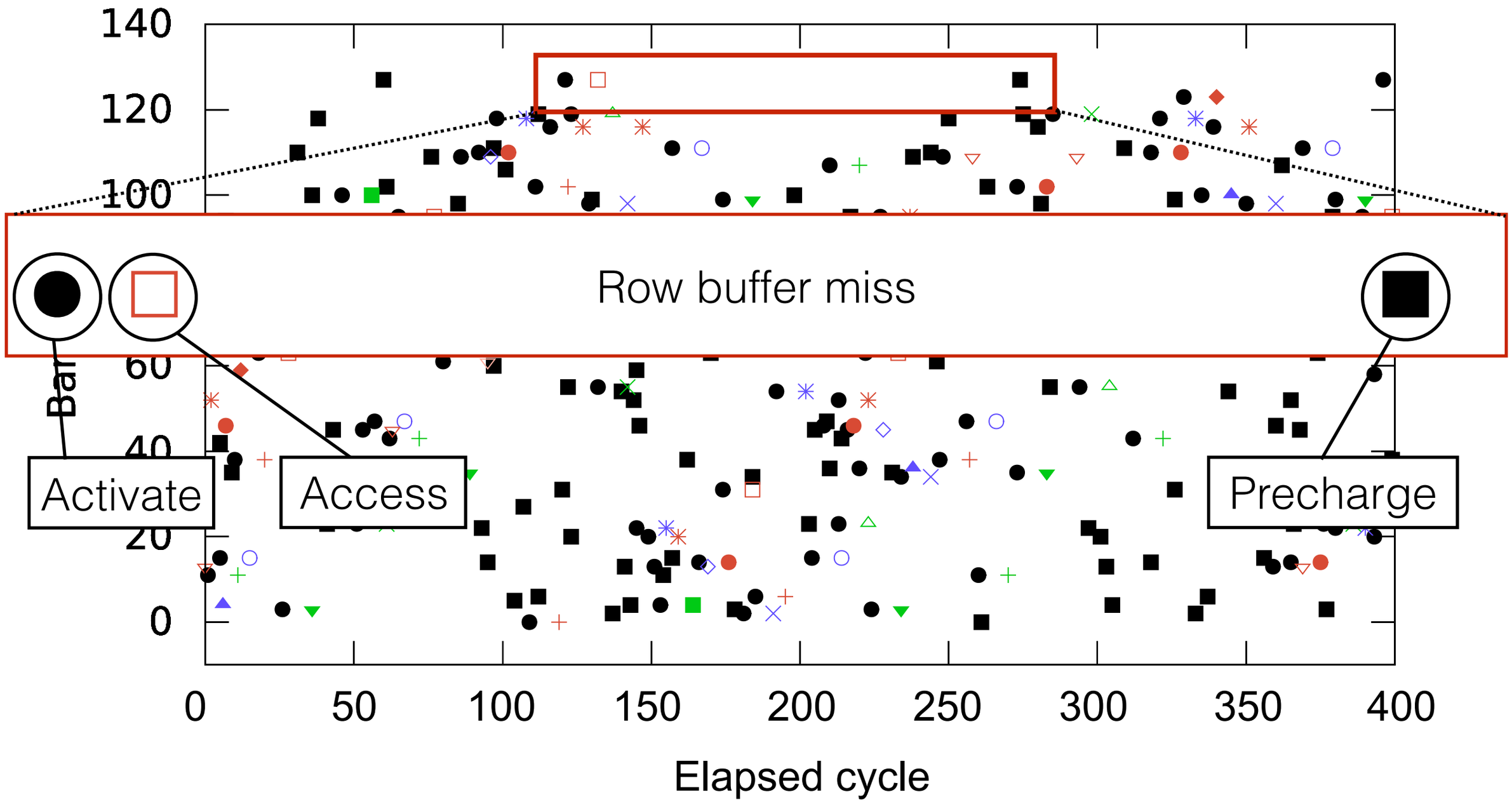
Memory Access Trace of Bottom-up with PCL



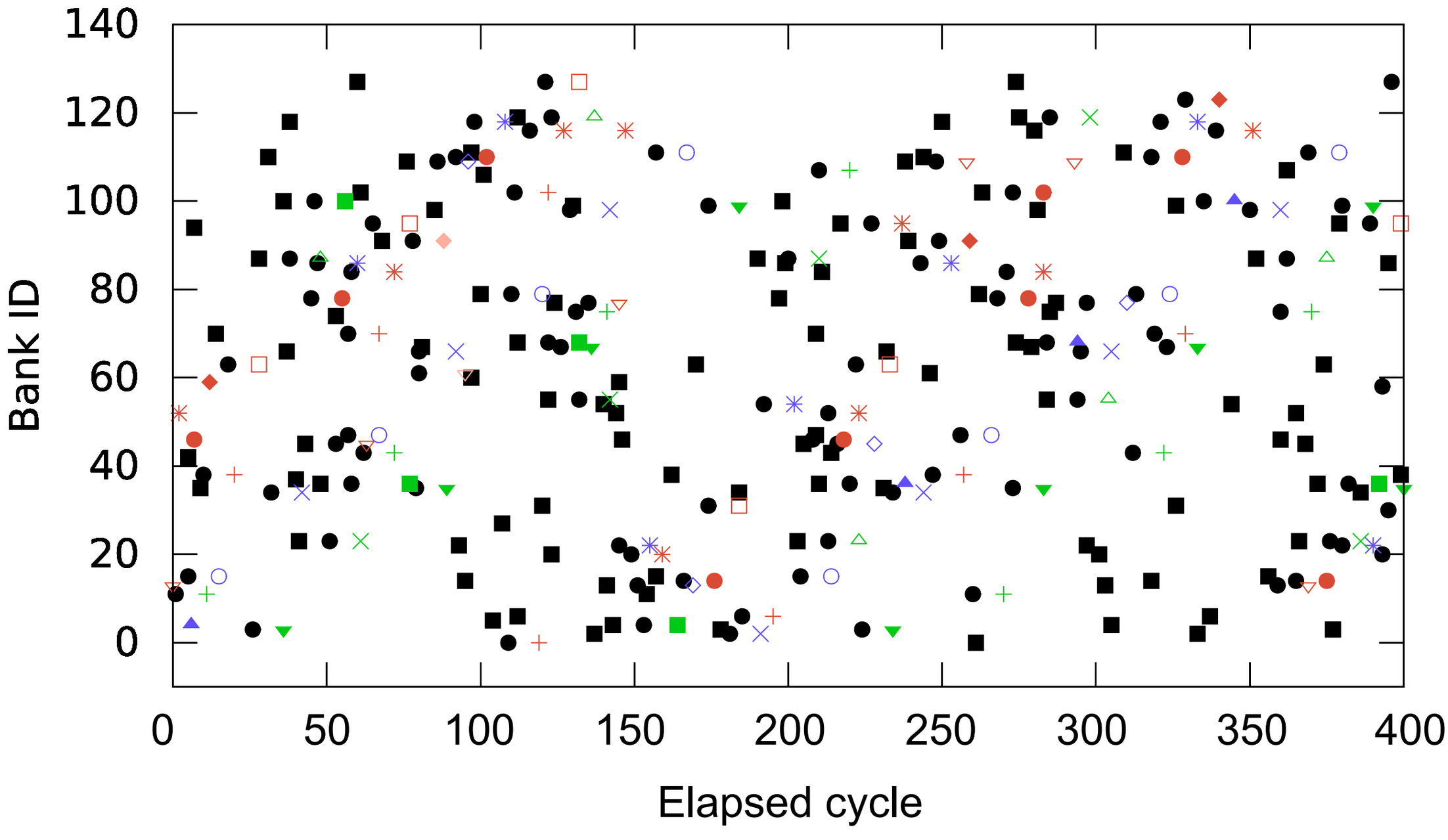
Memory Access Trace of Bottom-up with PCL



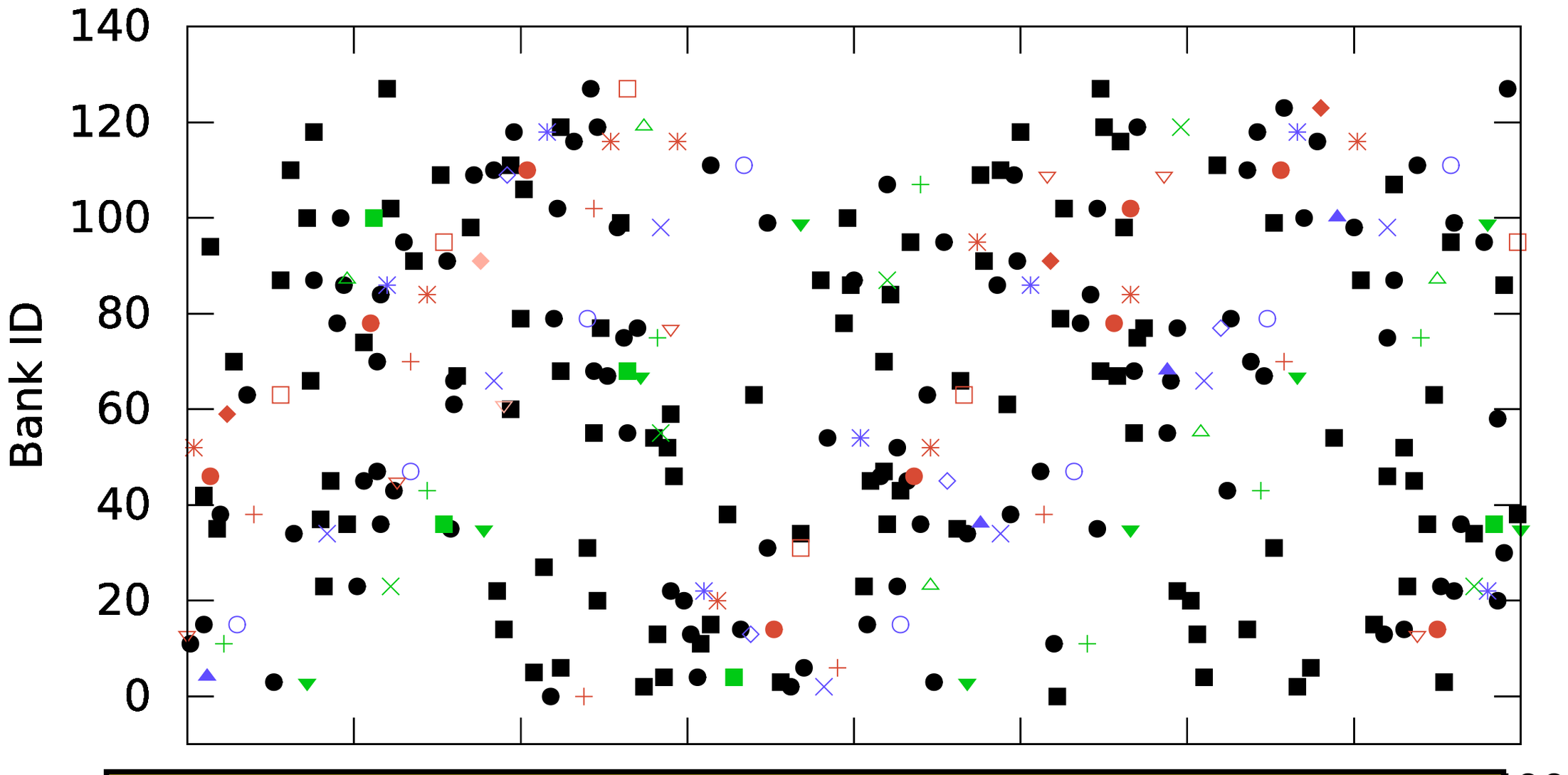
Memory Access Trace of Bottom-up with PCL



Memory Access Trace of Bottom-up with PCL

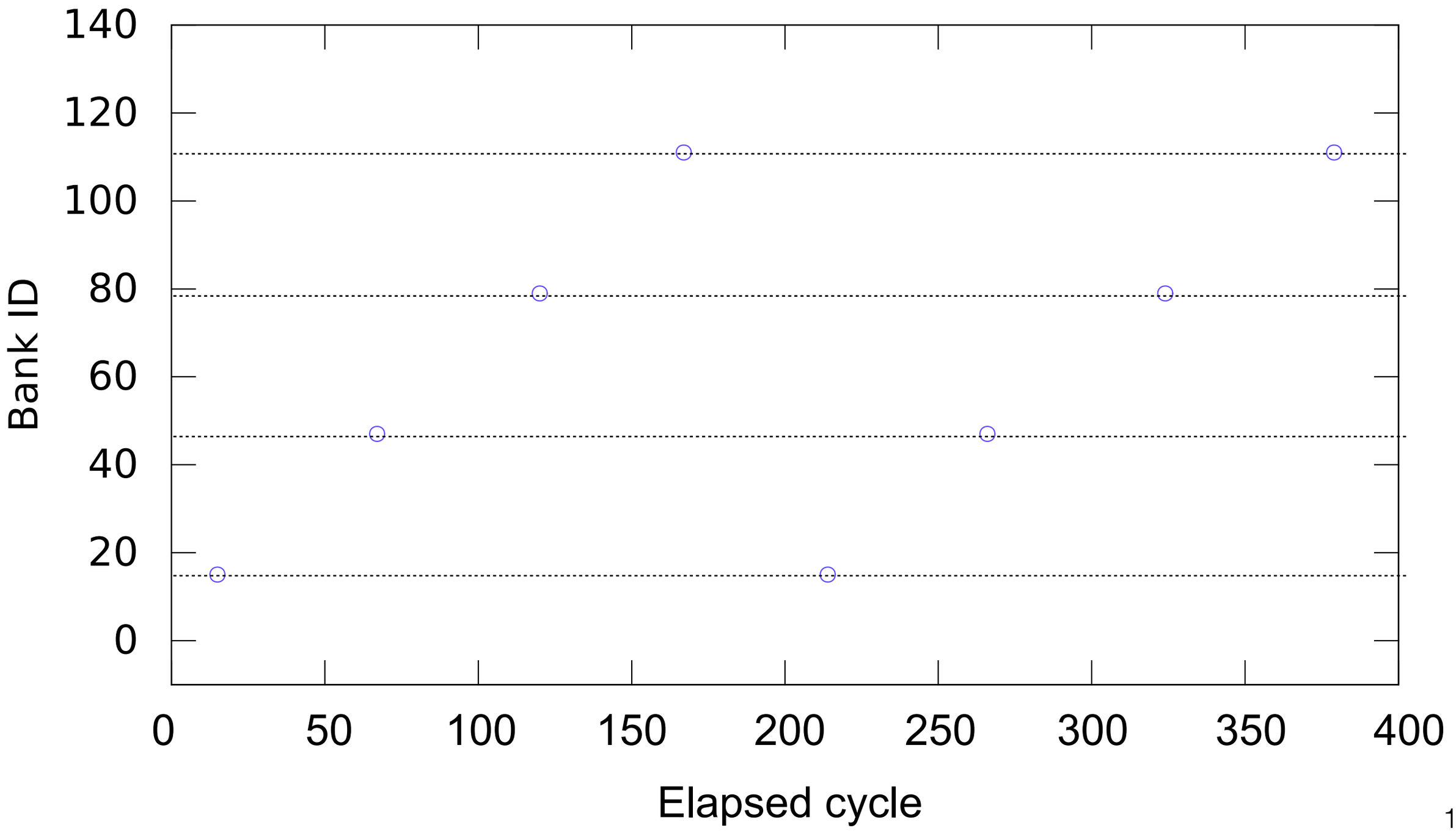


Memory Access Trace of Bottom-up with PCL

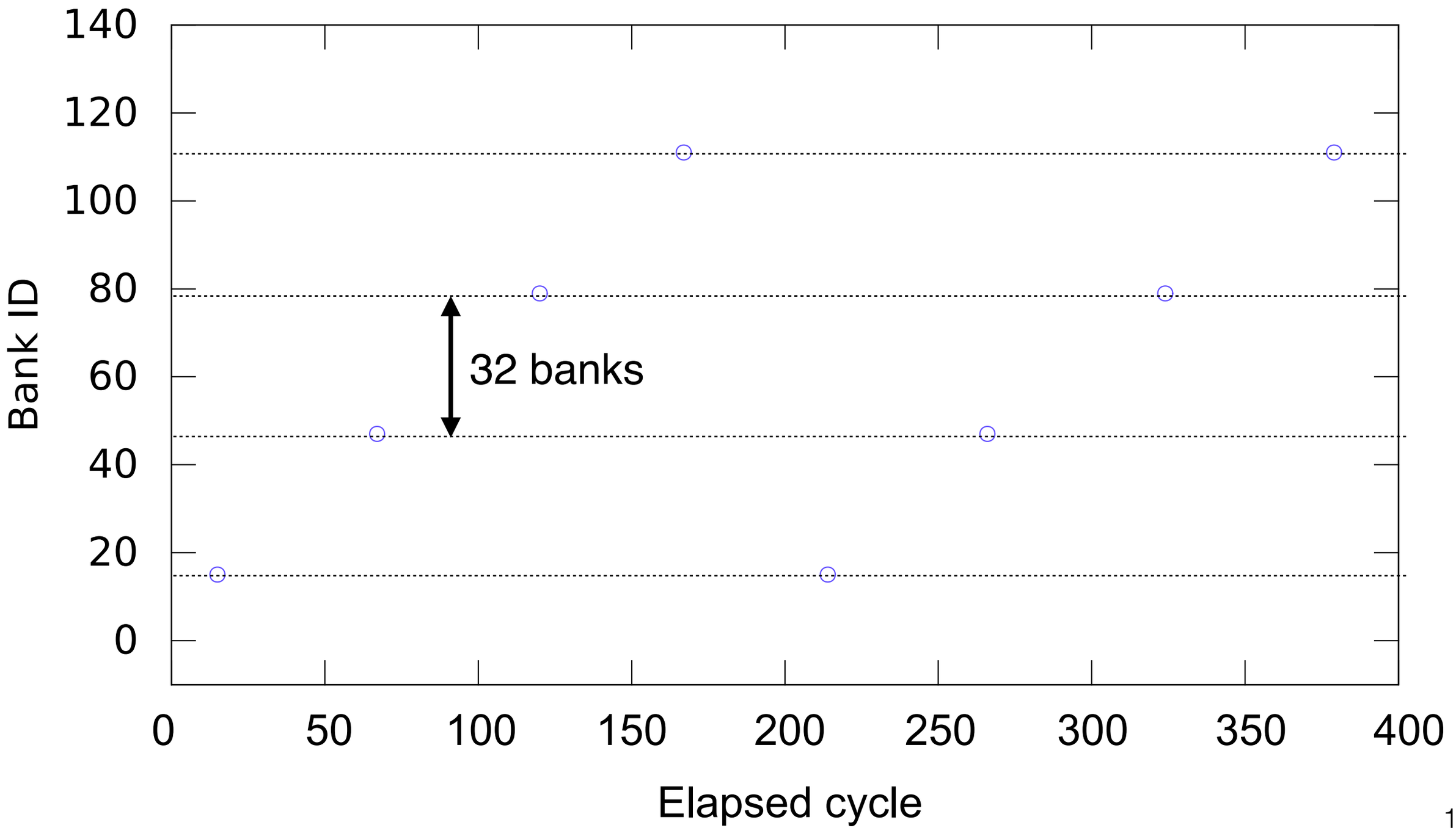


- Almost all accesses are row buffer misses
- Almost all threads access distinct banks

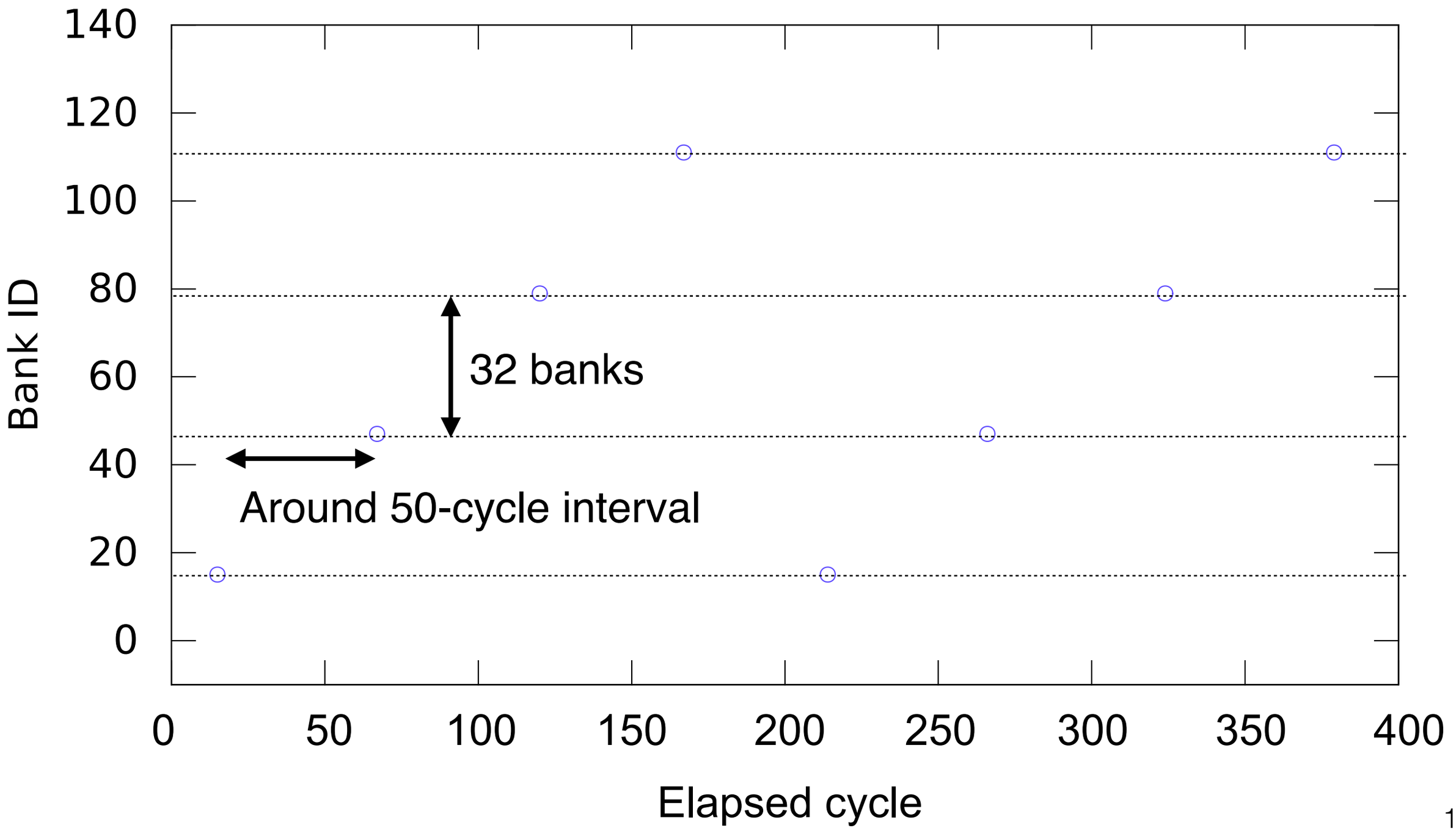
Memory Access Trace of Bottom-up with PCL (1 thread)



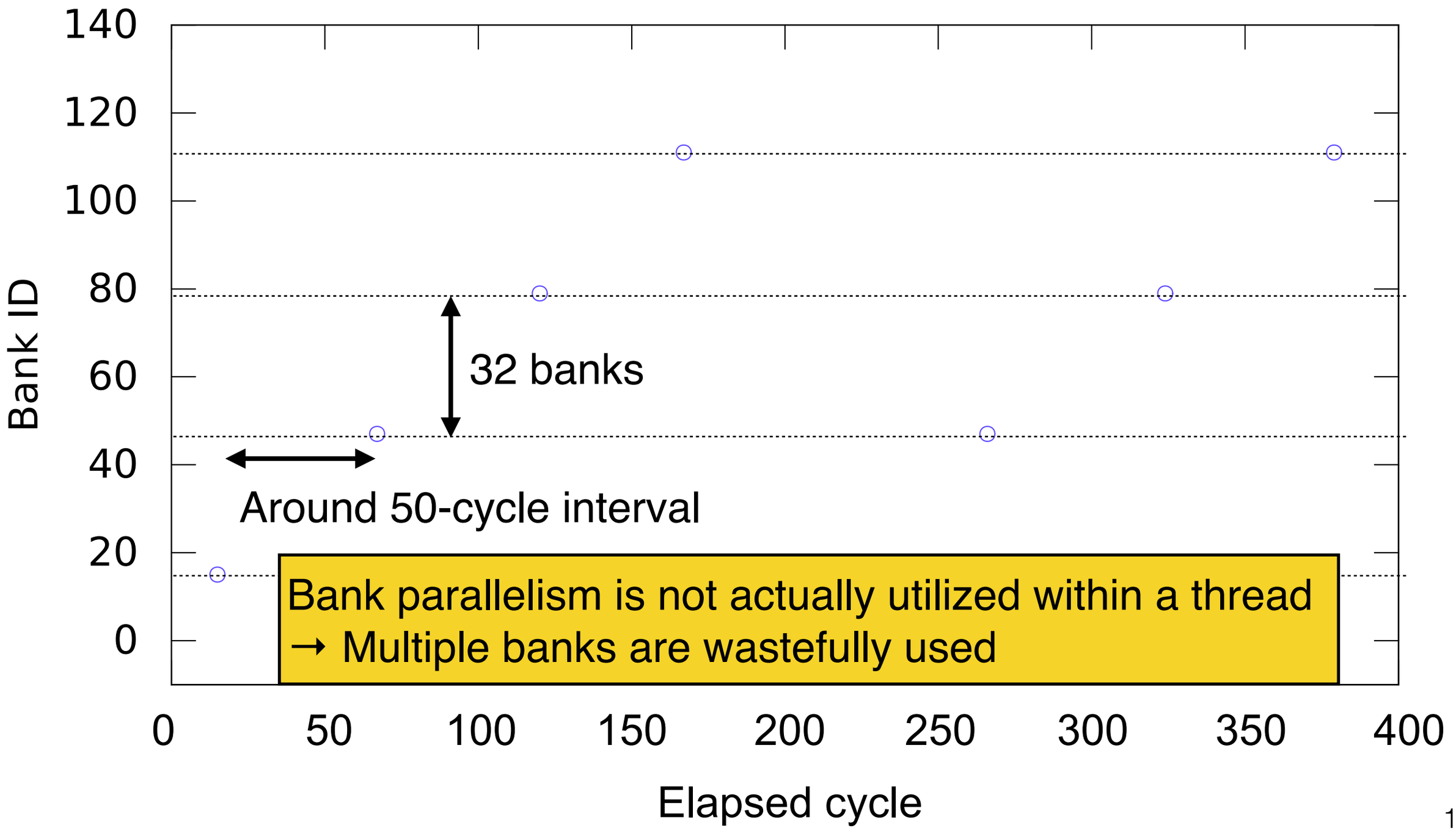
Memory Access Trace of Bottom-up with PCL (1 thread)



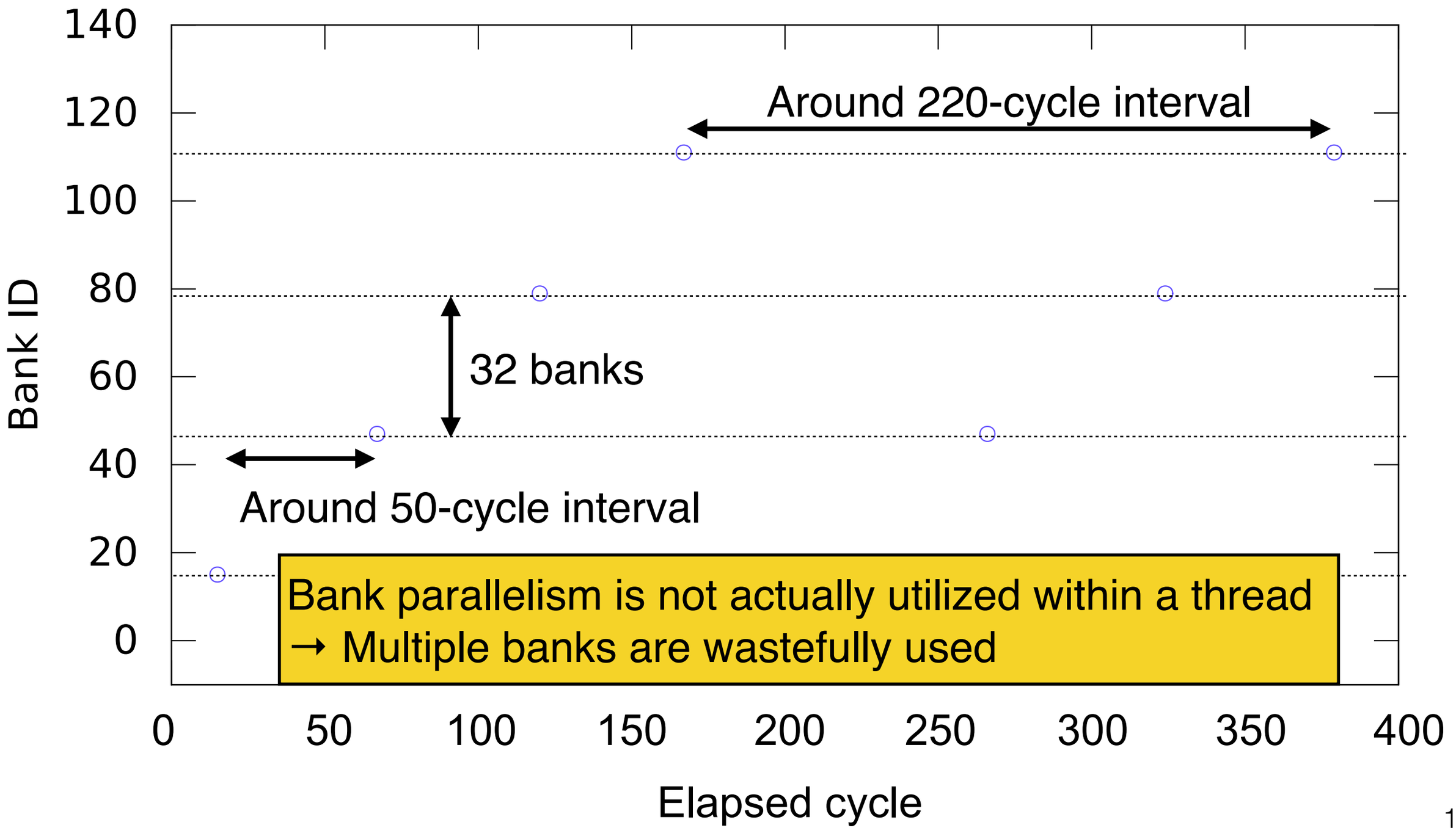
Memory Access Trace of Bottom-up with PCL (1 thread)



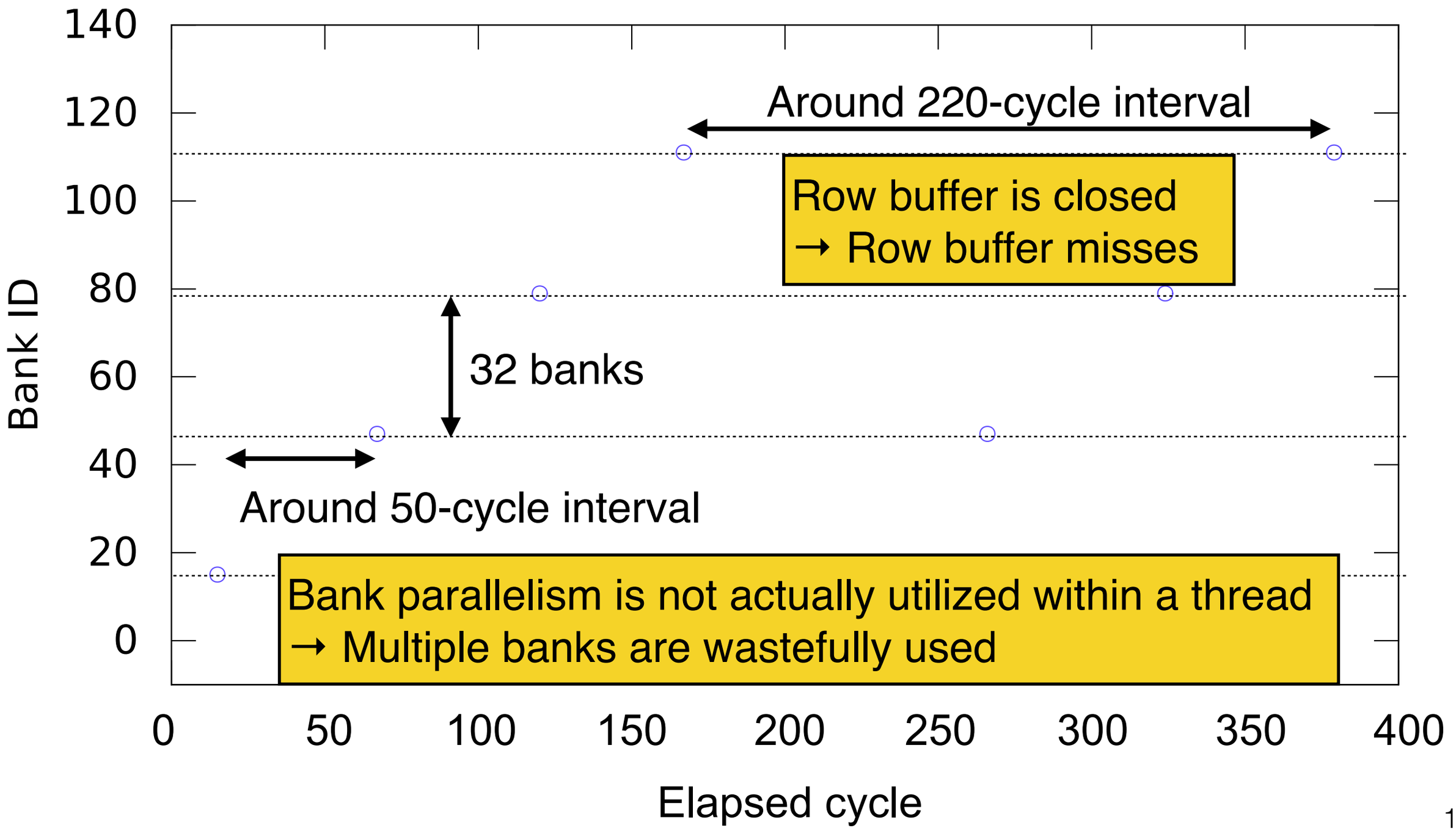
Memory Access Trace of Bottom-up with PCL (1 thread)



Memory Access Trace of Bottom-up with PCL (1 thread)



Memory Access Trace of Bottom-up with PCL (1 thread)



Agenda

- State-of-the-art BFS implementation
- DRAM mechanisms
- Memory access analysis with conventional address mapping schemes
- Proposed: per-row channel interleaving
- Evaluation of power efficiency

Idea

- If each thread sequentially accesses the entire row in the same bank…
  - Row buffer hit ratio (RBHR) should be improved
  - Bank parallelism should be exploited among different threads
  - #banks used at a time should be reduced

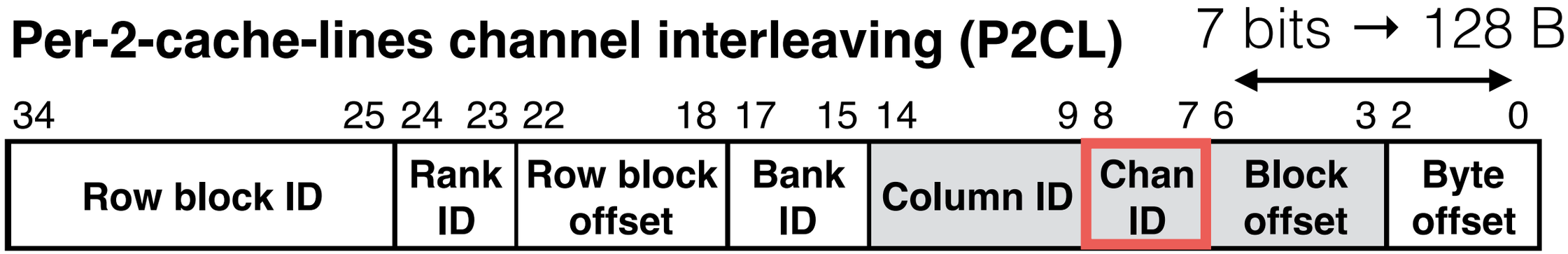
Per-Row Channel Interleaving (PR)

- Interleaves a contiguous memory region across channels per row size (8 KB in this work)

Per-Row Channel Interleaving (PR)

- Interleaves a contiguous memory region across channels per row size (8 KB in this work)

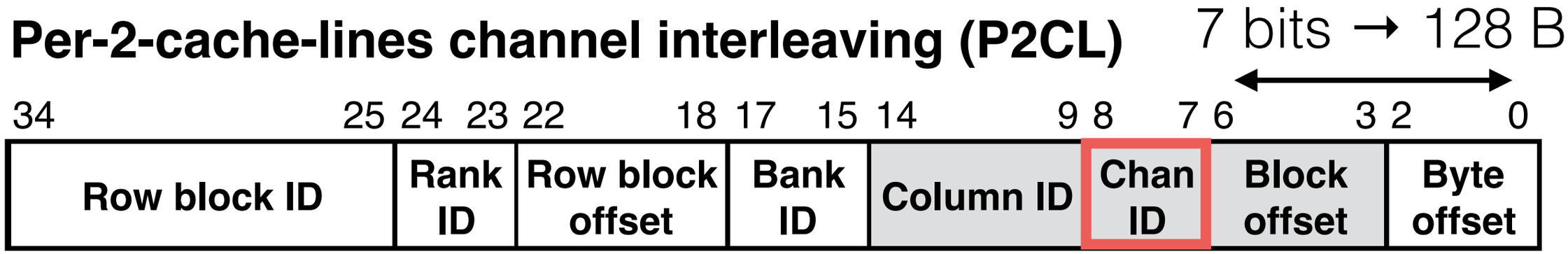
Per-2-cache-lines channel interleaving (P2CL)

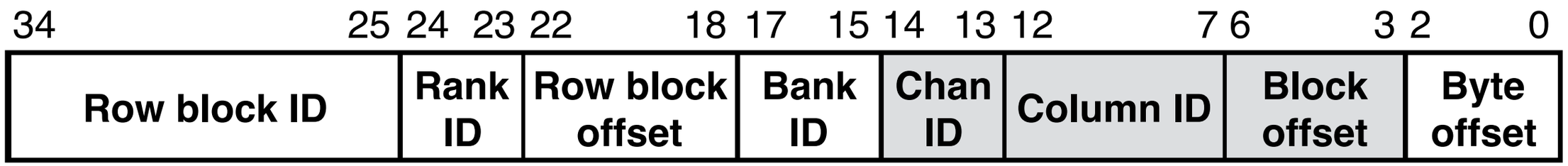


Per-Row Channel Interleaving (PR)

- Interleaves a contiguous memory region across channels per row size (8 KB in this work)

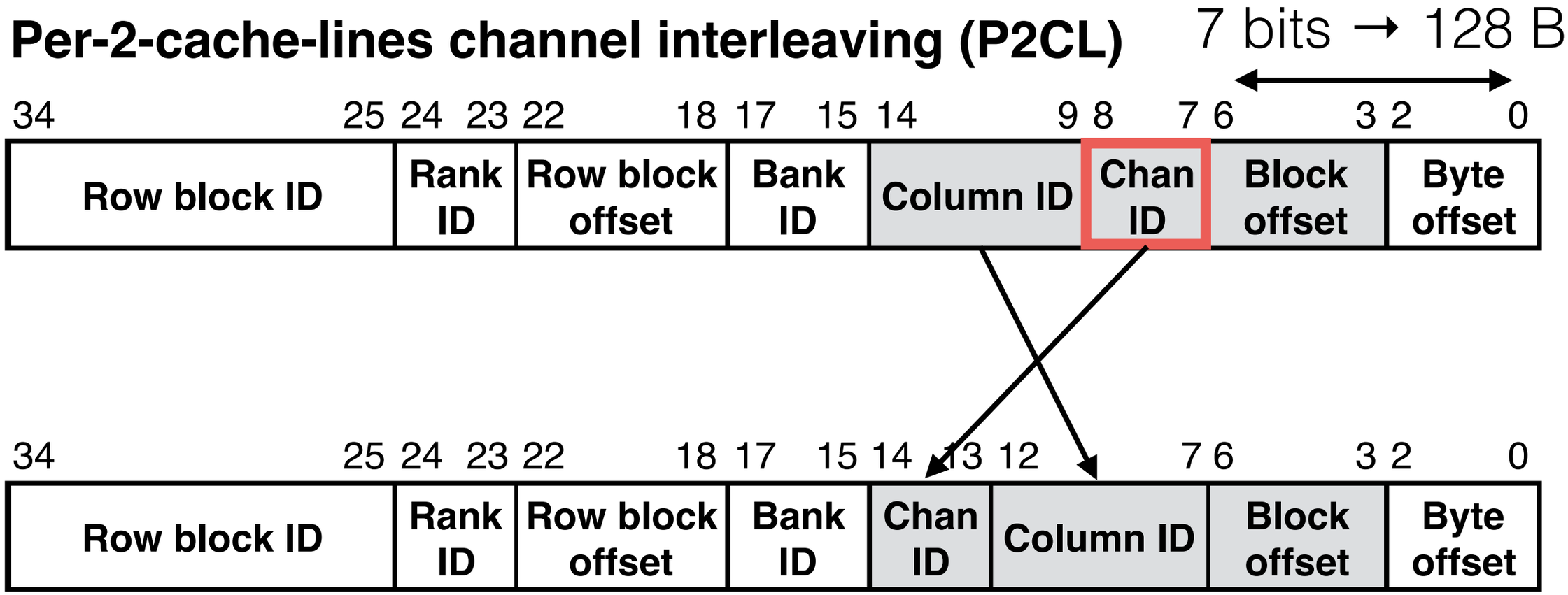
Per-2-cache-lines channel interleaving (P2CL)





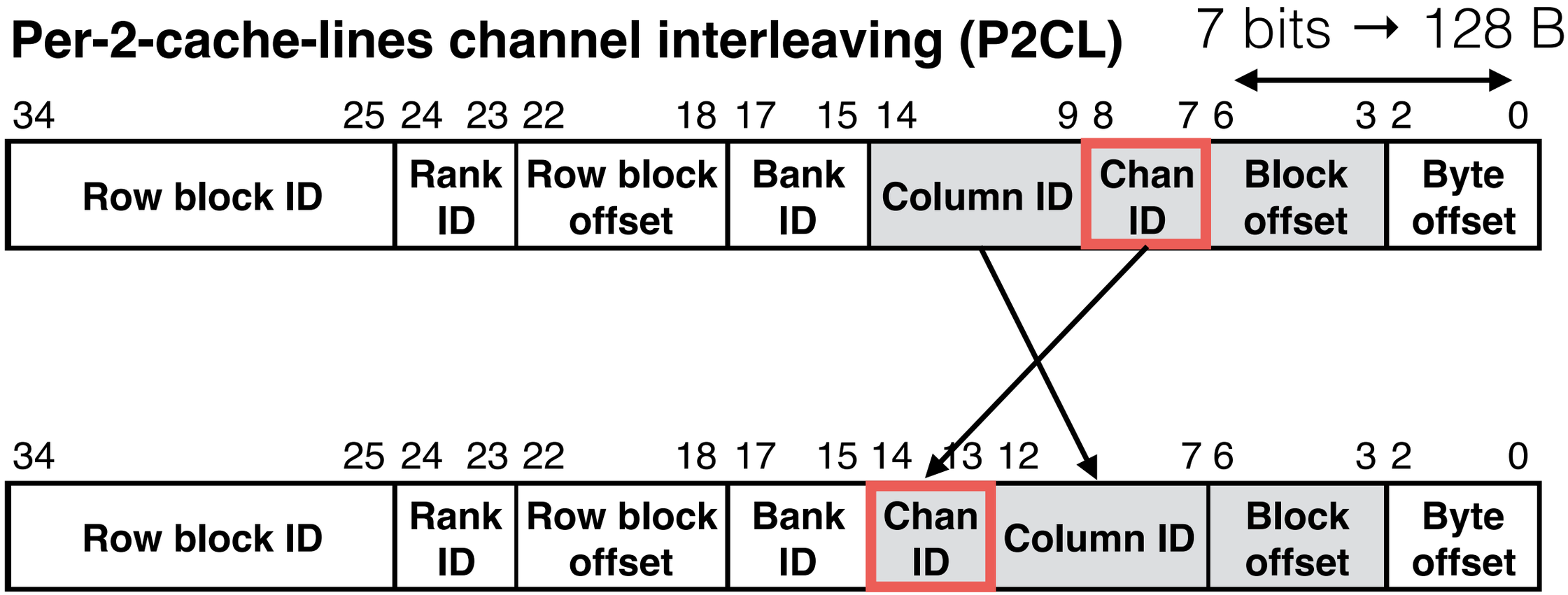
Per-Row Channel Interleaving (PR)

- Interleaves a contiguous memory region across channels per row size (8 KB in this work)



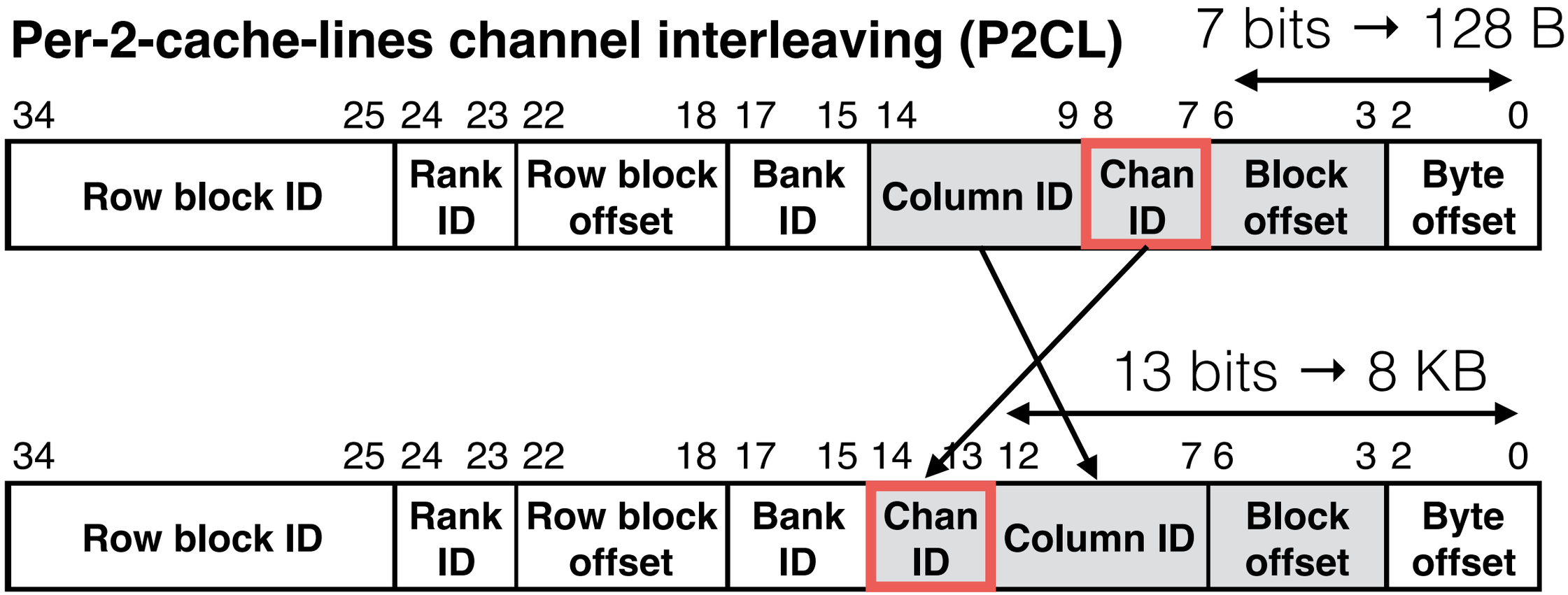
Per-Row Channel Interleaving (PR)

- Interleaves a contiguous memory region across channels per row size (8 KB in this work)

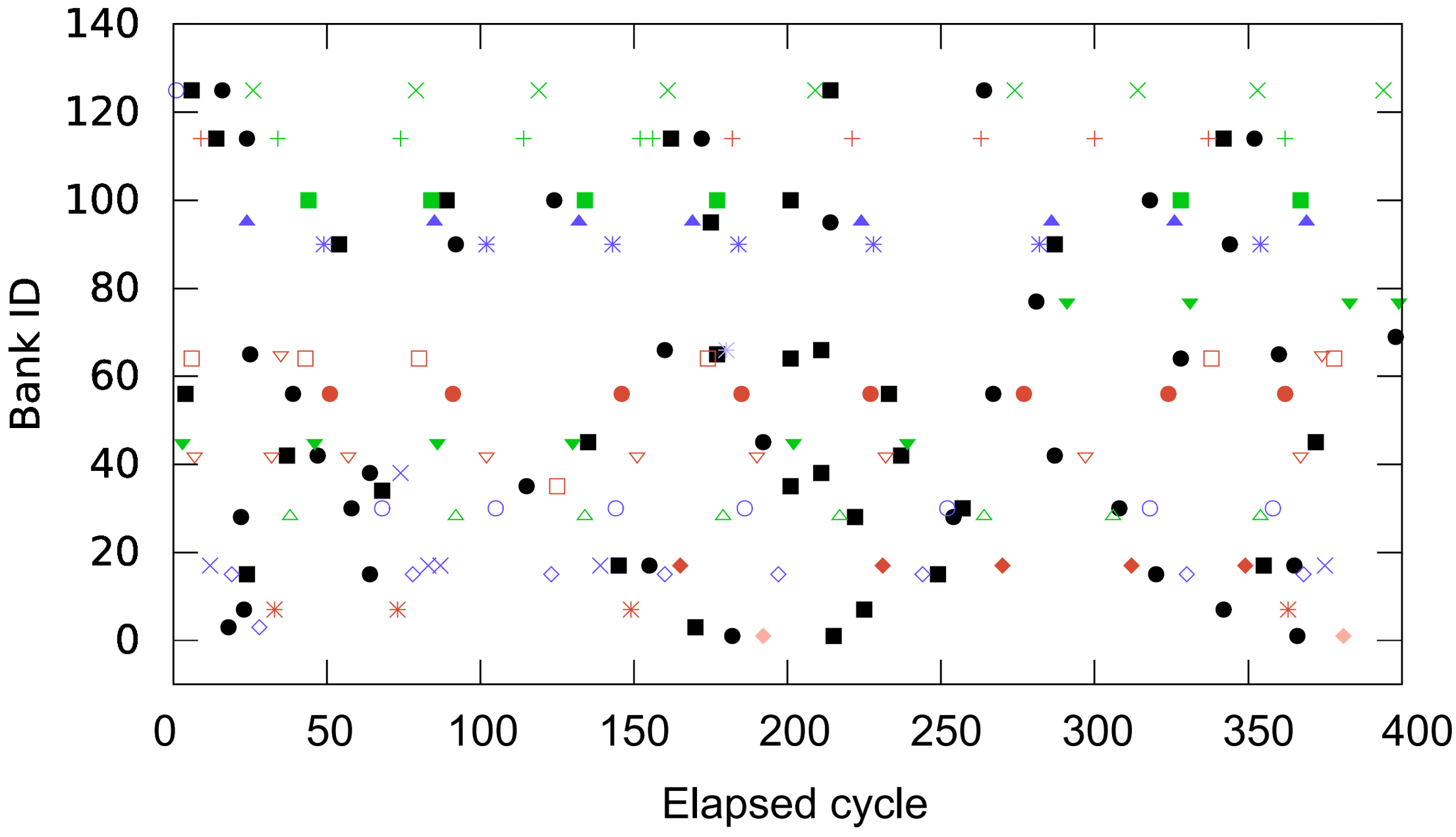


Per-Row Channel Interleaving (PR)

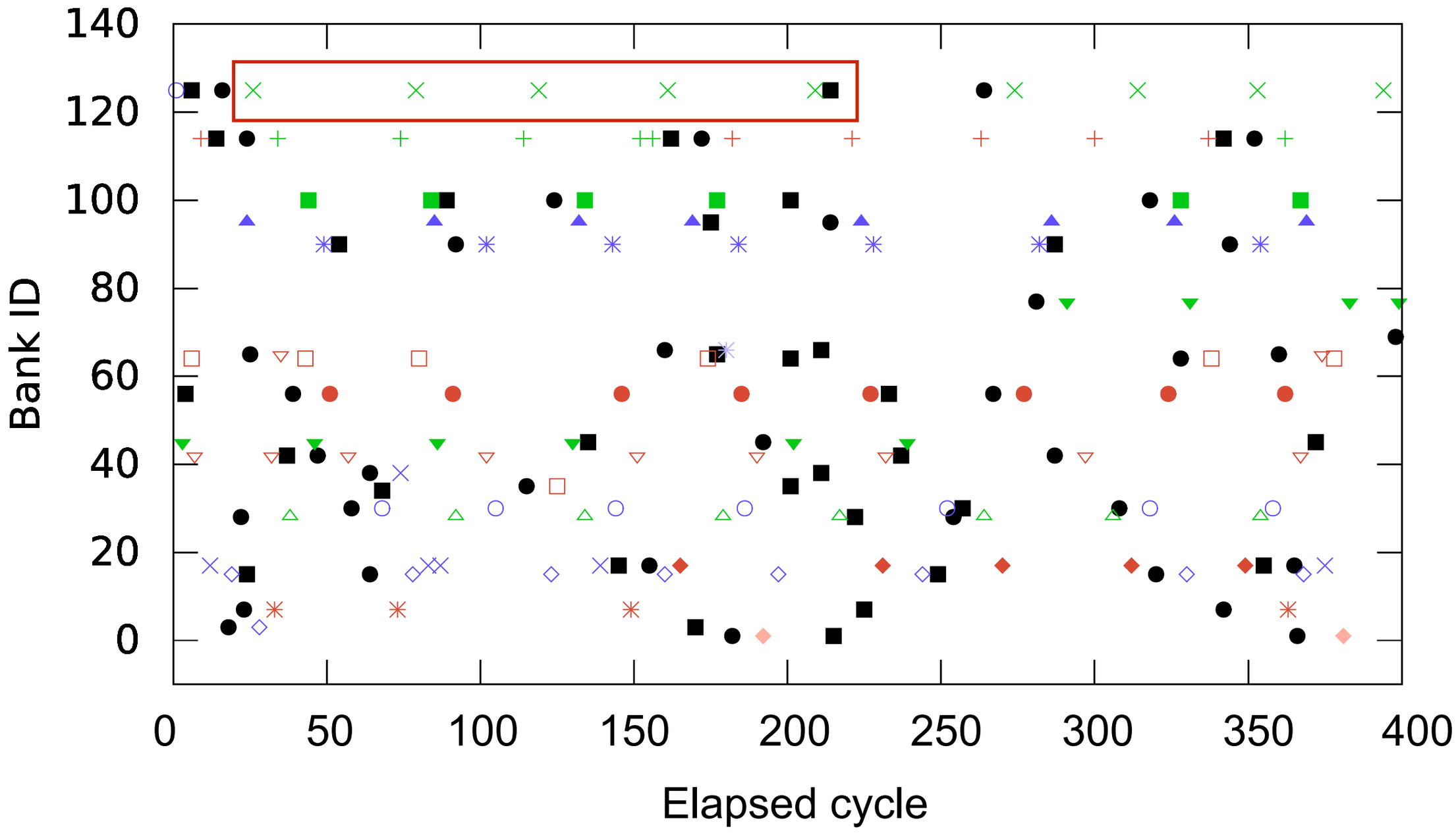
- Interleaves a contiguous memory region across channels per row size (8 KB in this work)



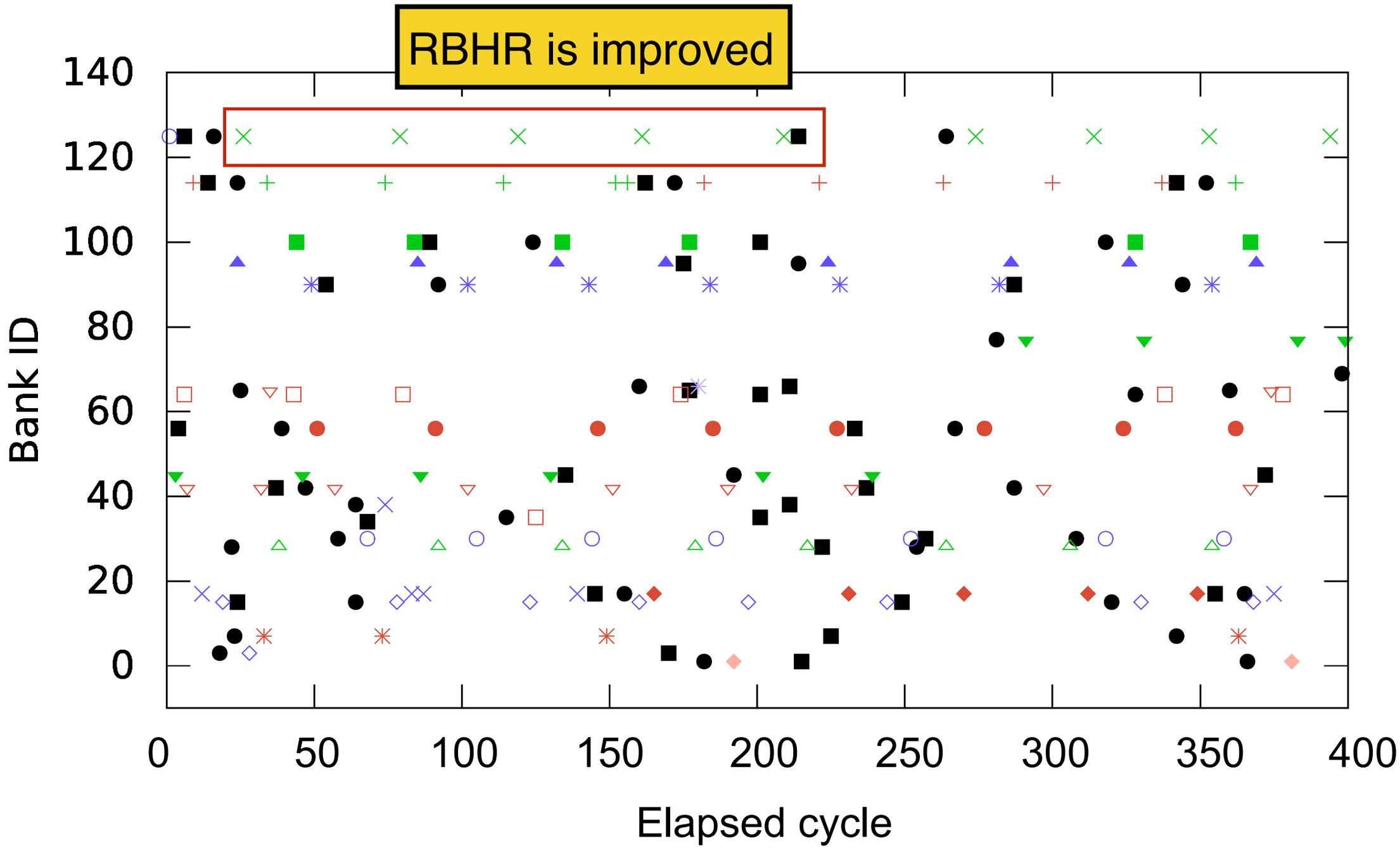
Memory Access Trace with PR



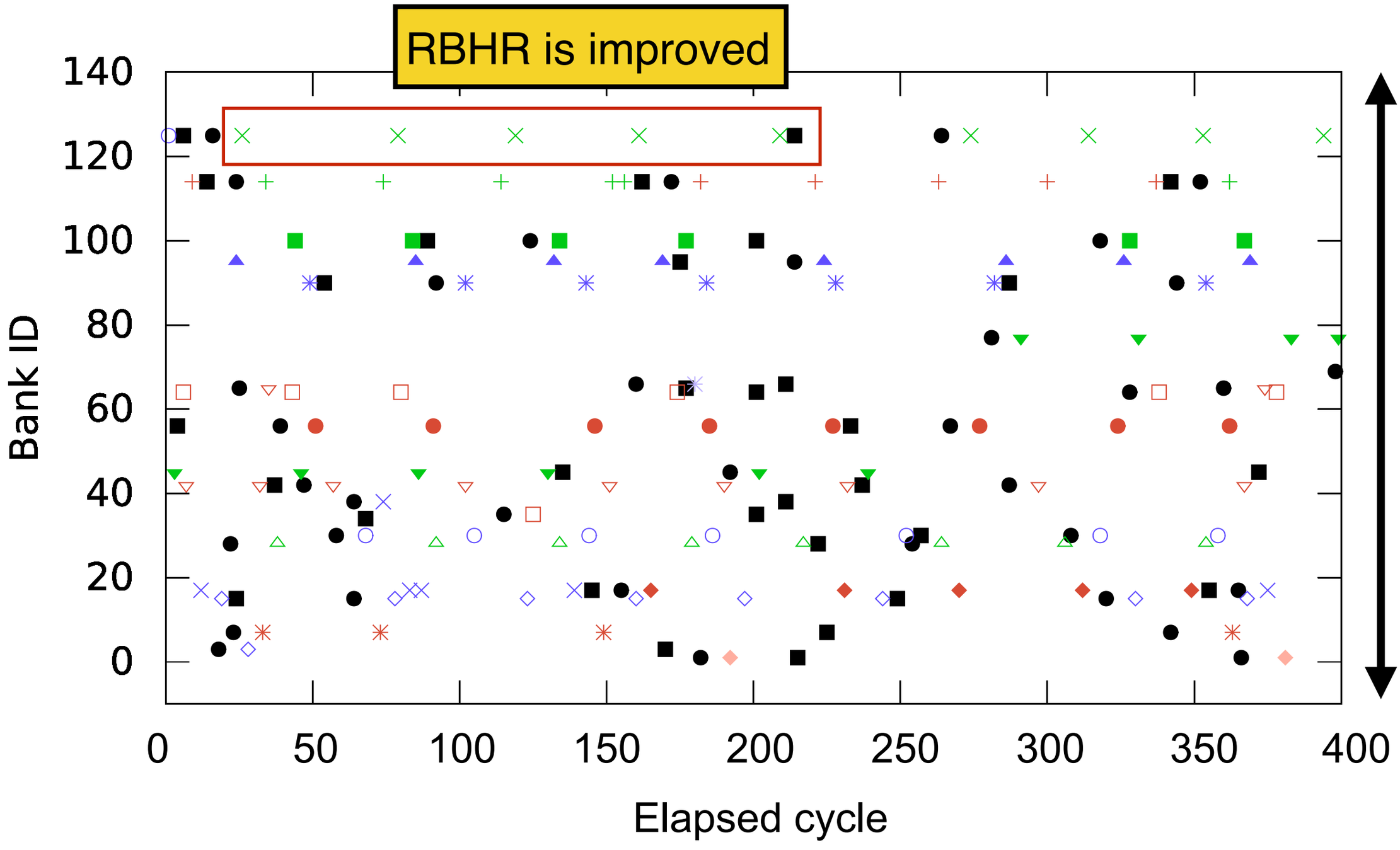
Memory Access Trace with PR



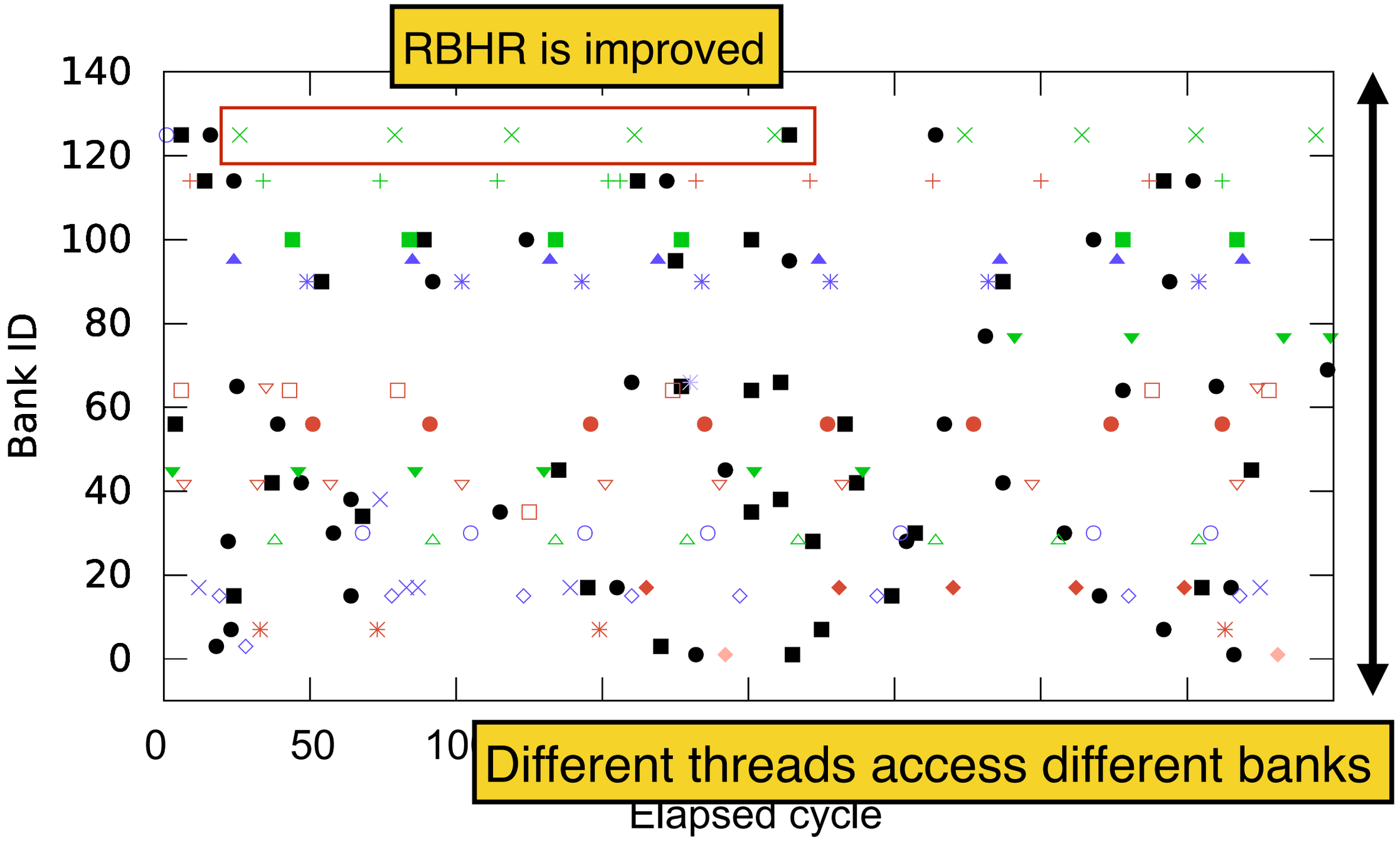
Memory Access Trace with PR



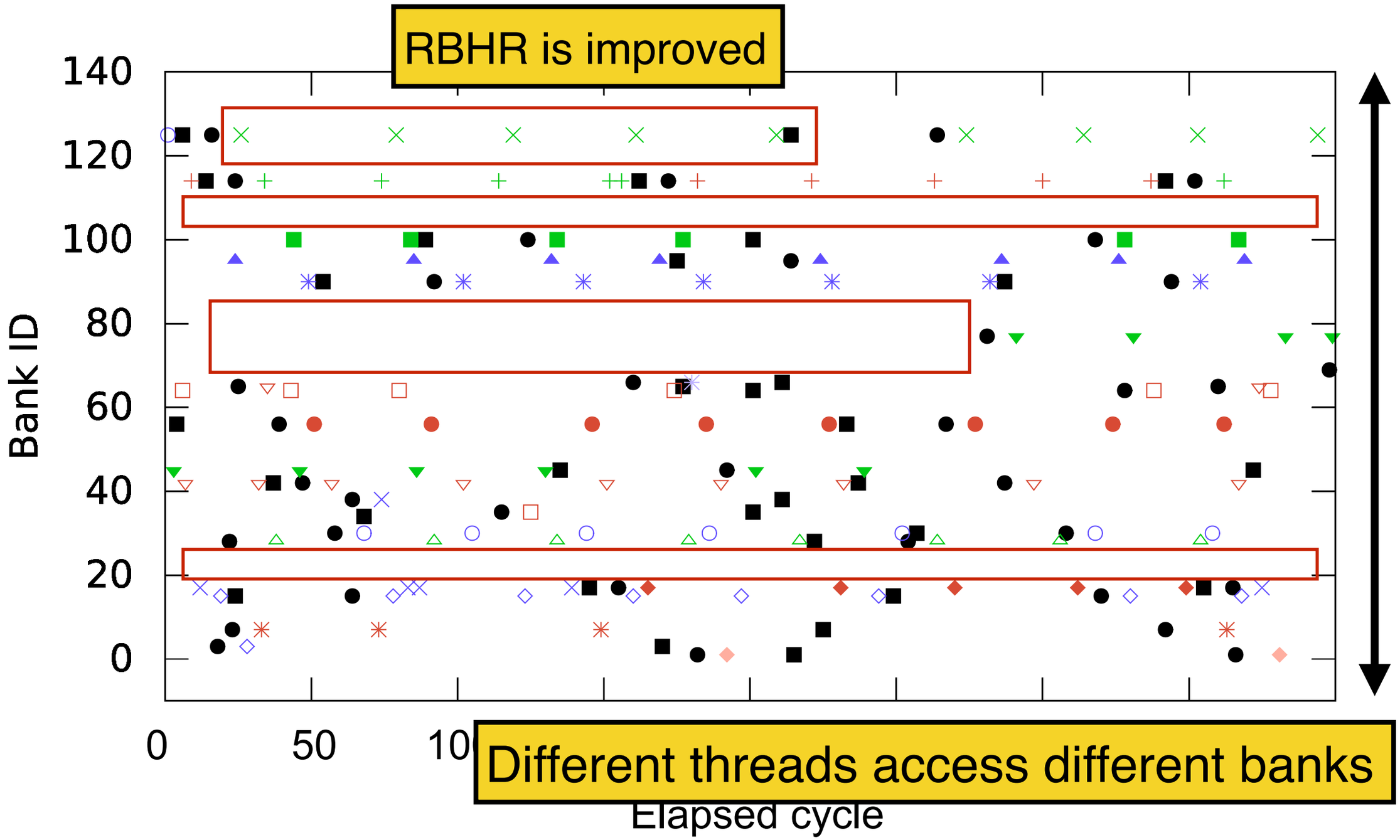
Memory Access Trace with PR



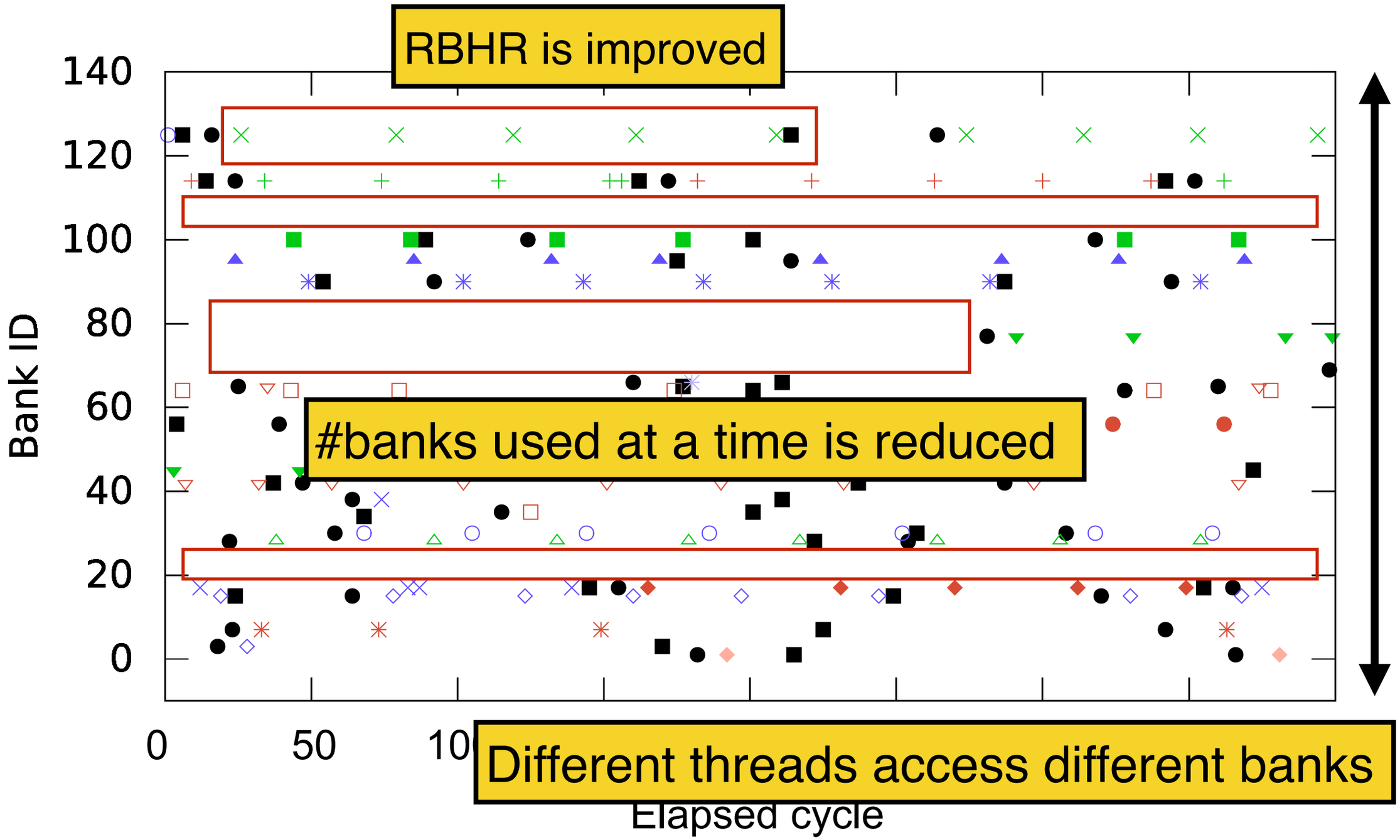
Memory Access Trace with PR



Memory Access Trace with PR



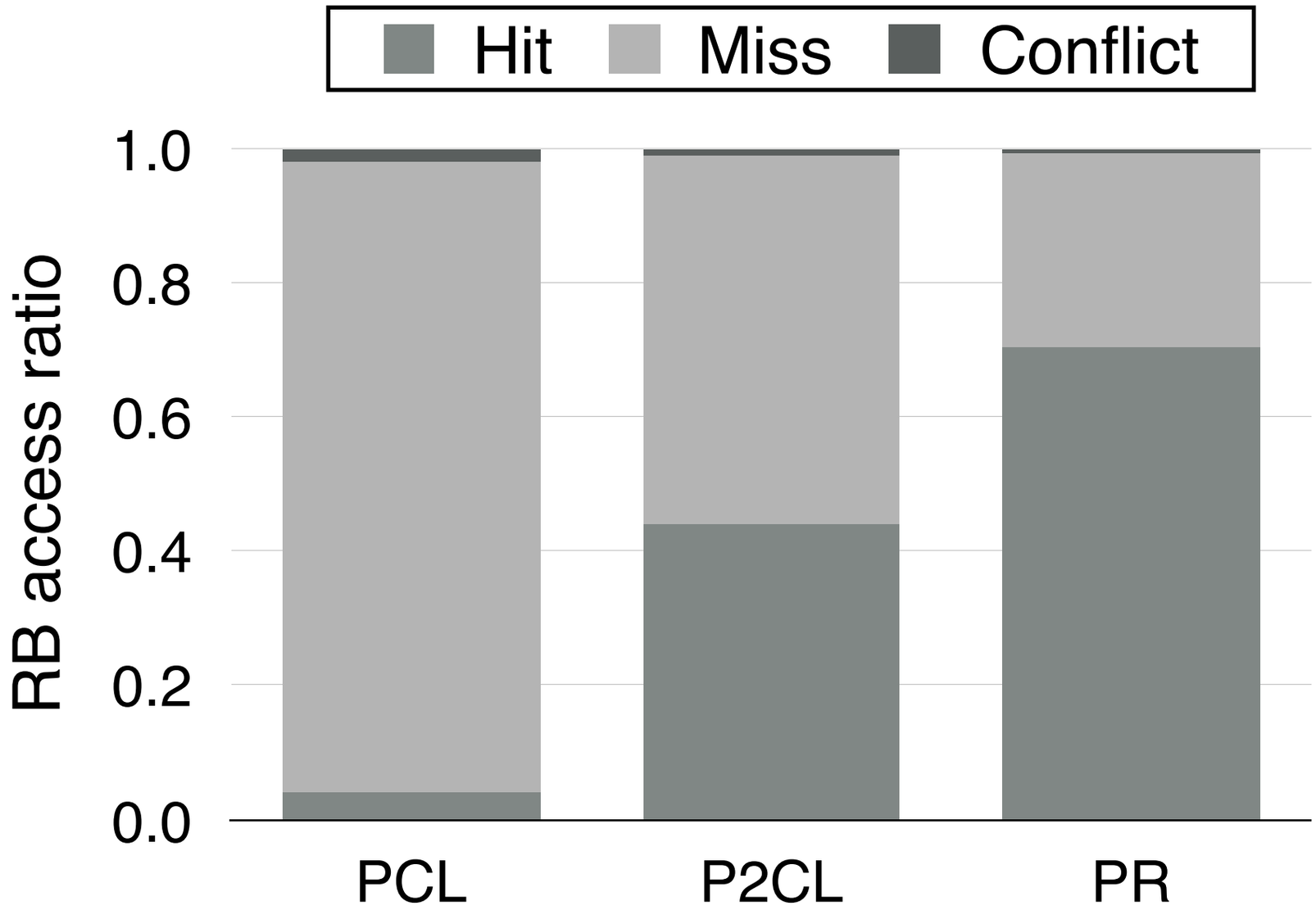
Memory Access Trace with PR



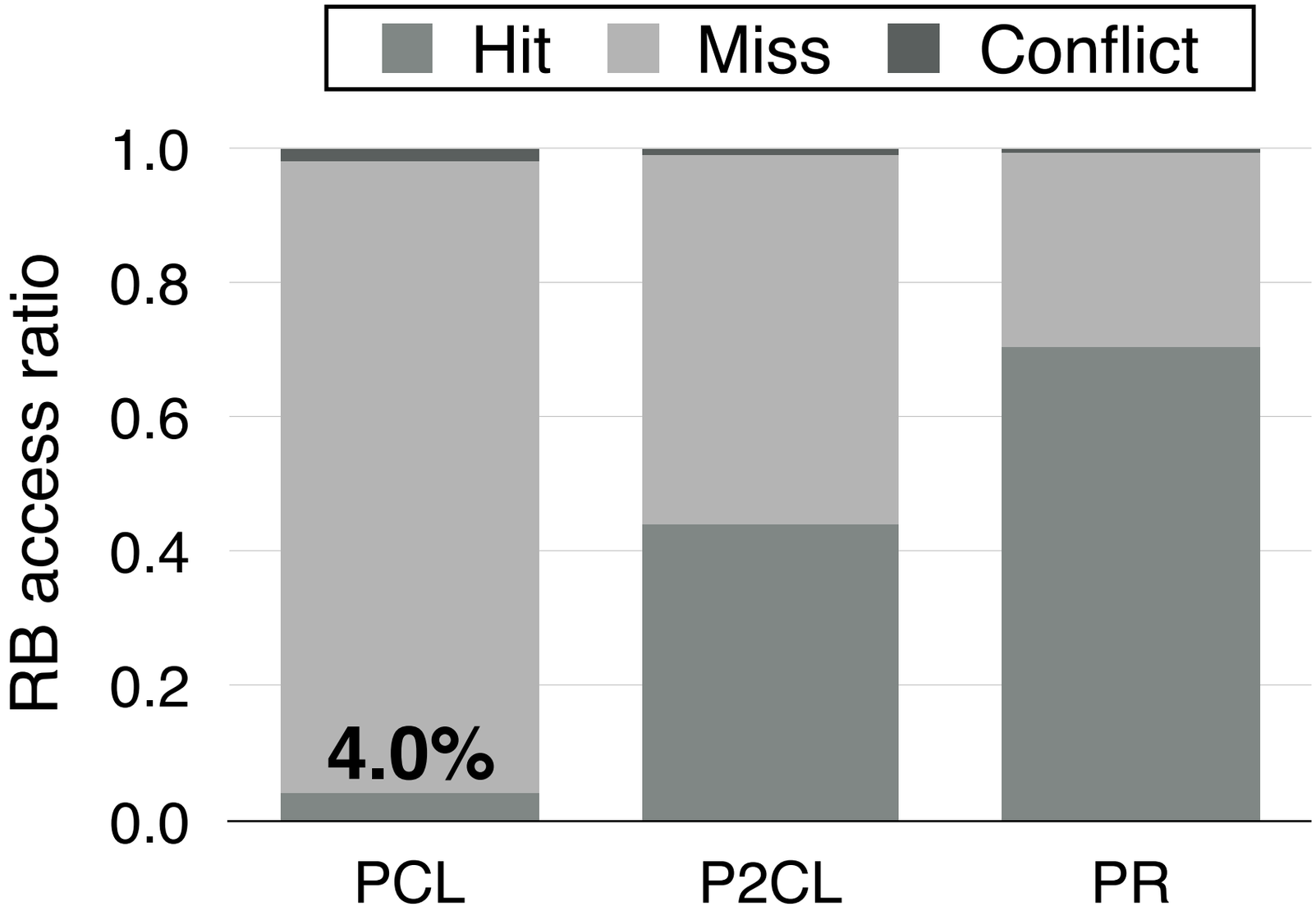
Agenda

- Graph analysis applications and Graph500
- State-of-the-art Graph500 implementation
- DRAM mechanisms
- Memory access analysis with conventional address mapping schemes
- Proposed: per-row channel interleaving
- Evaluation of power efficiency

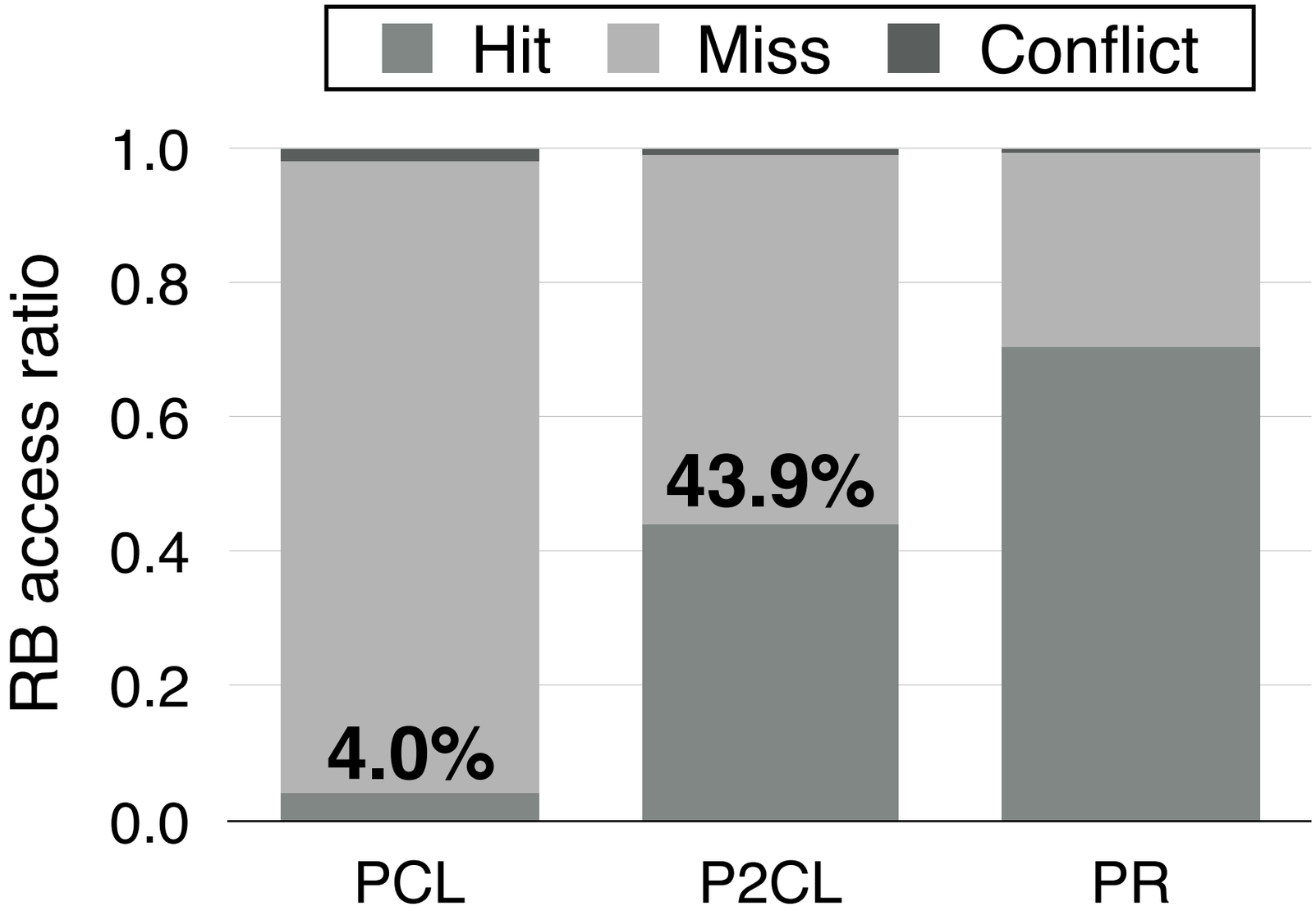
Row Buffer Access Ratio



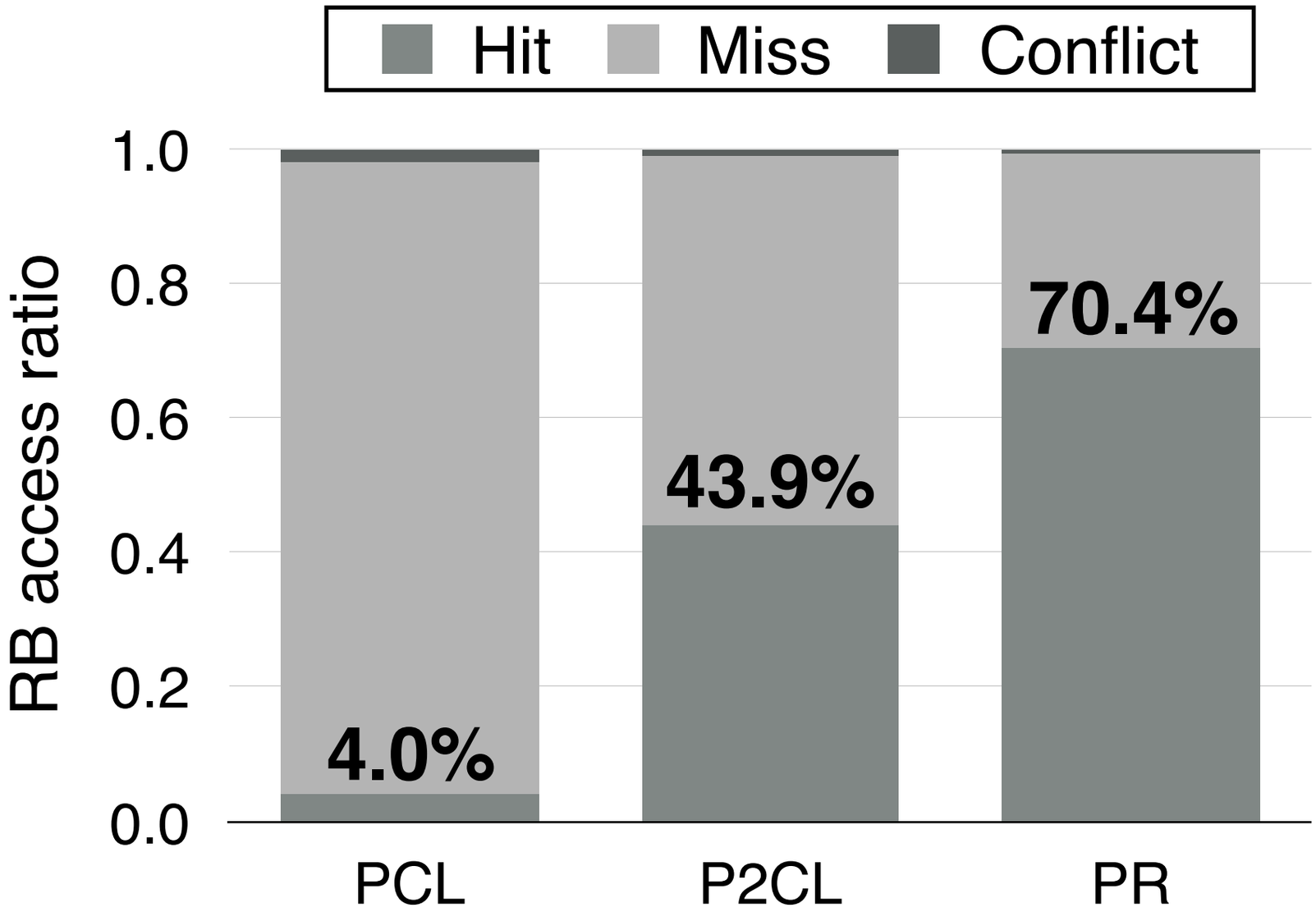
Row Buffer Access Ratio



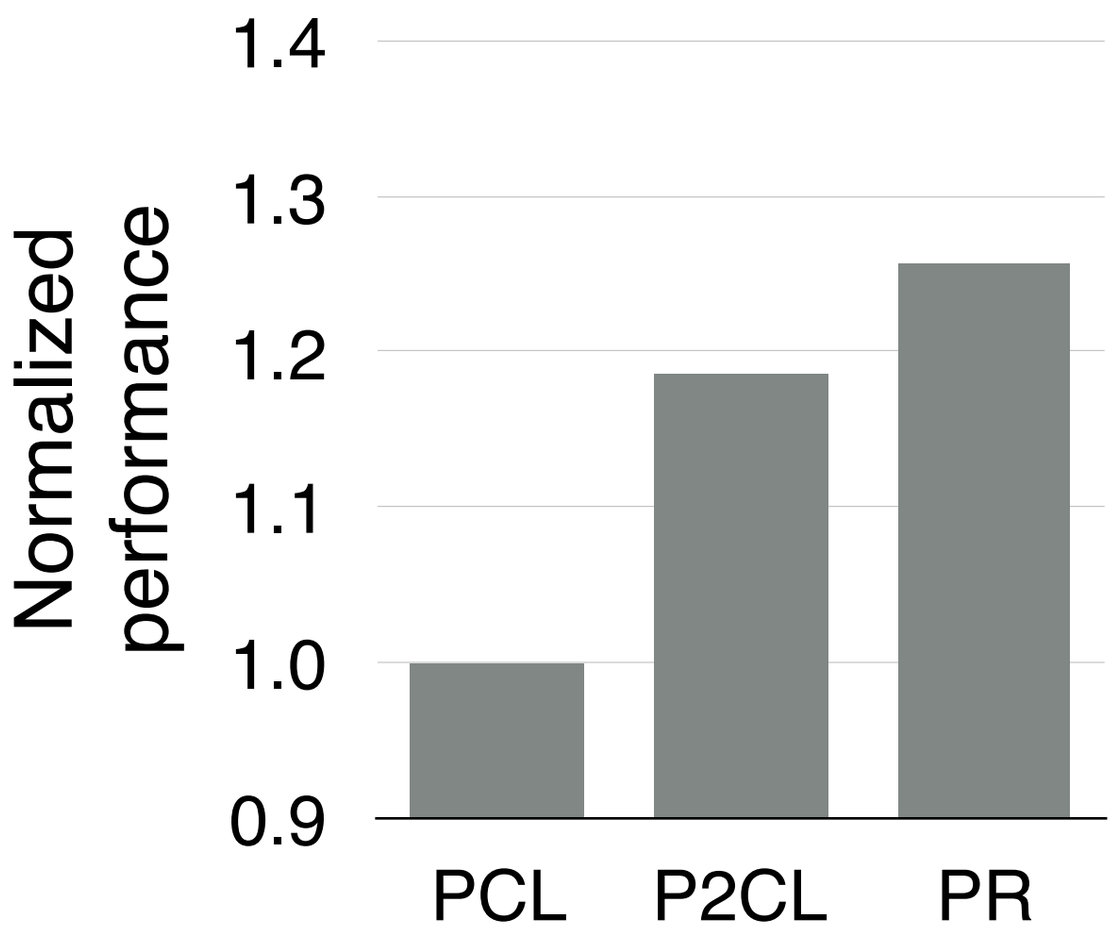
Row Buffer Access Ratio



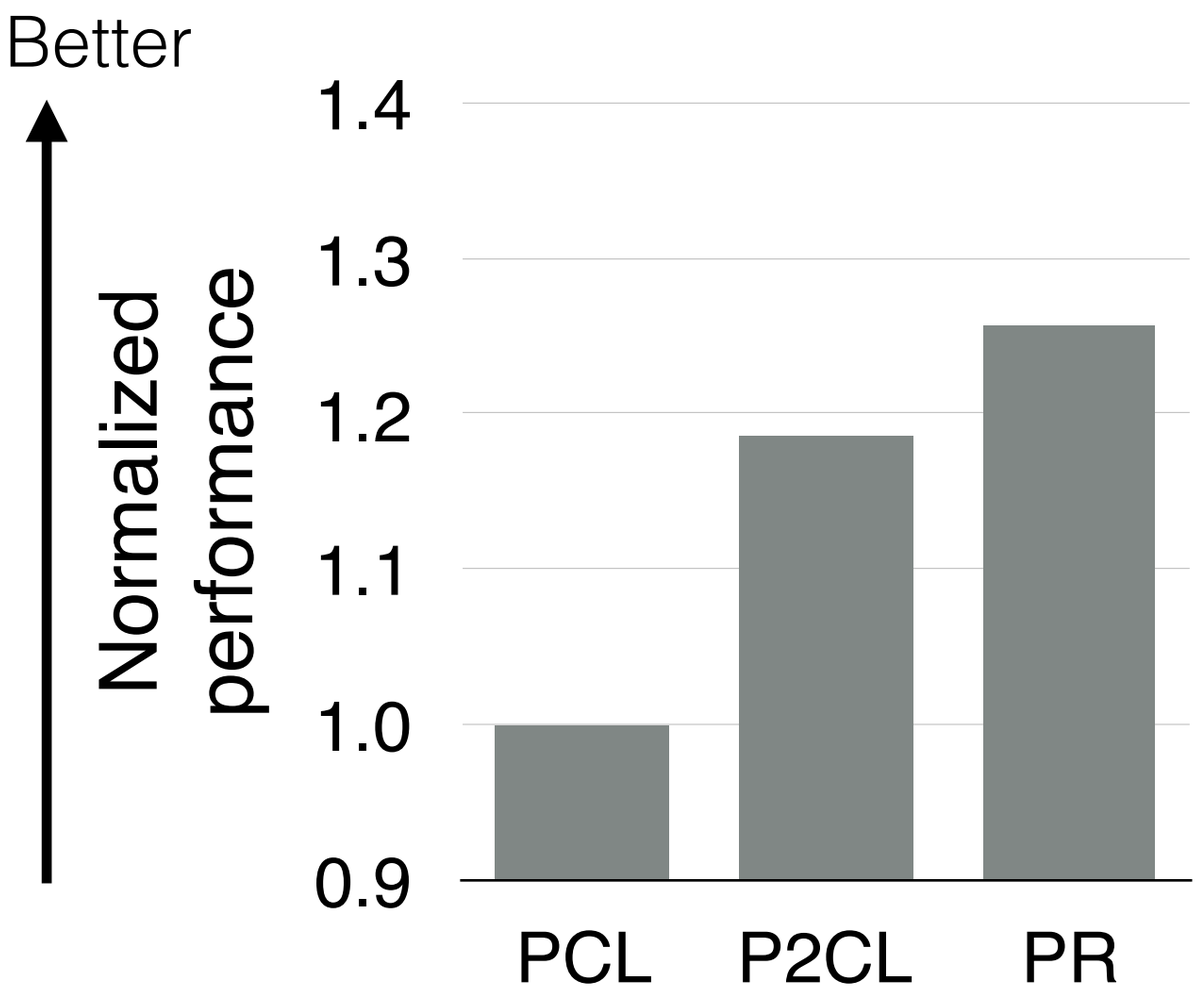
Row Buffer Access Ratio



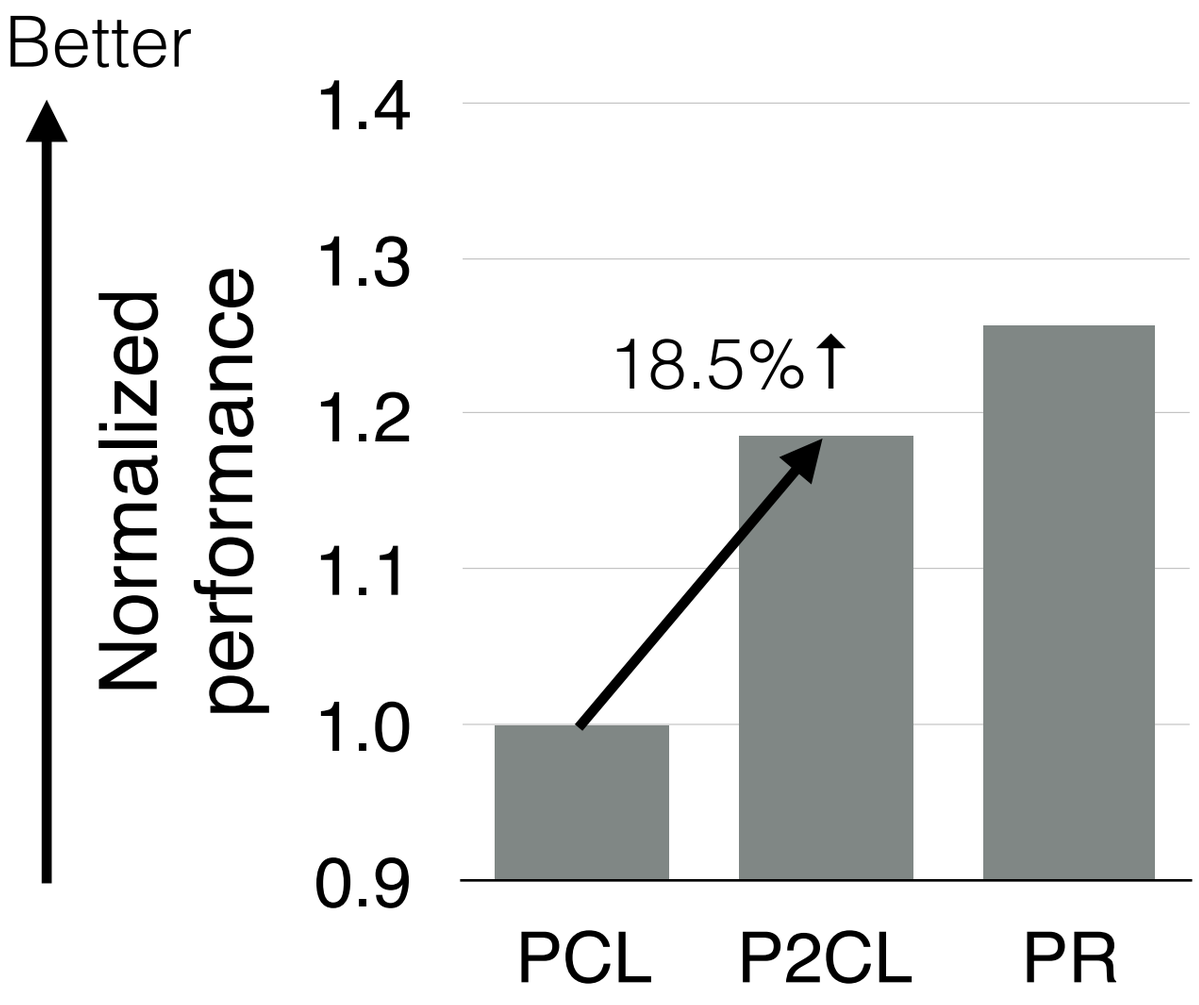
Performance & Power



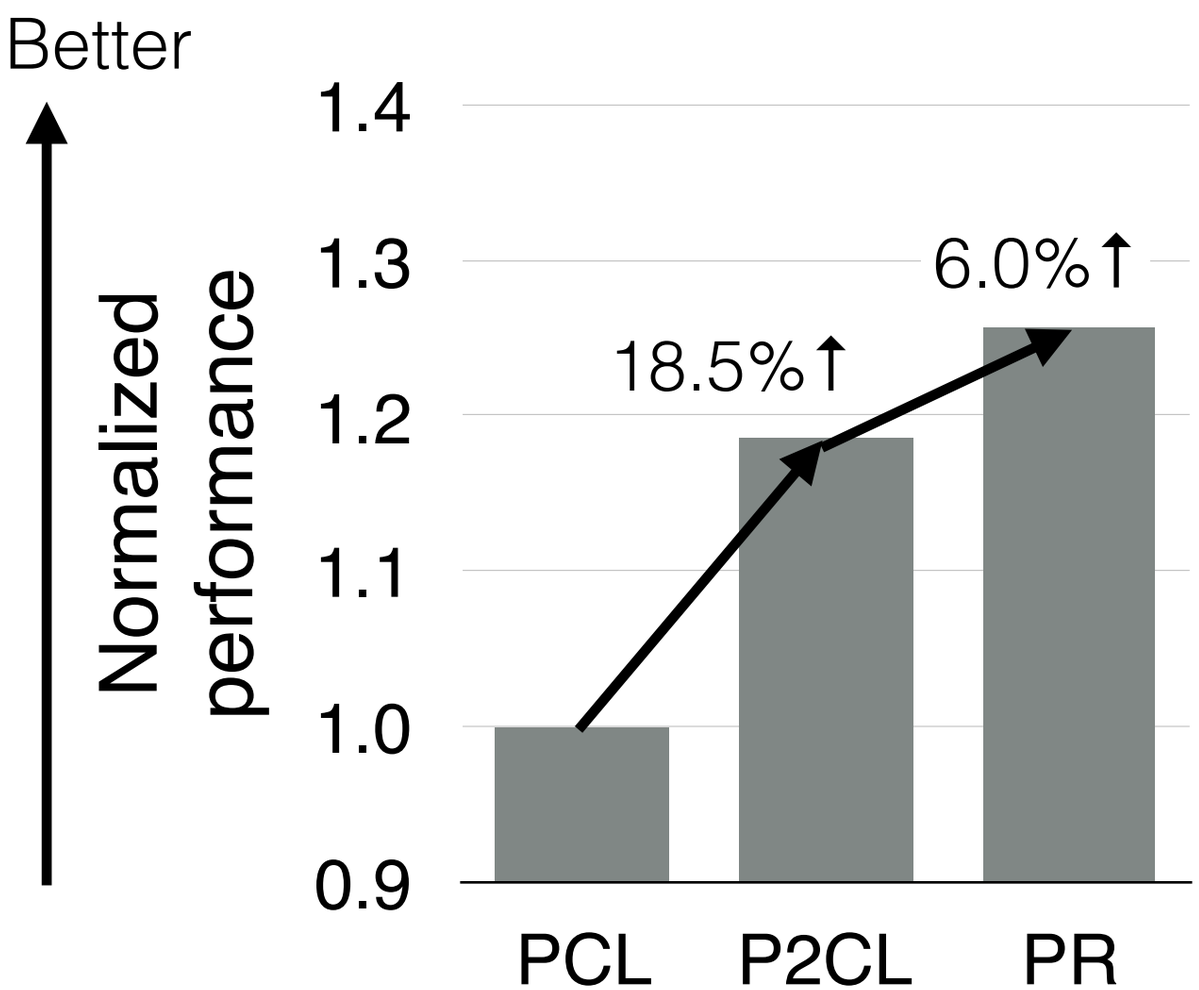
Performance & Power



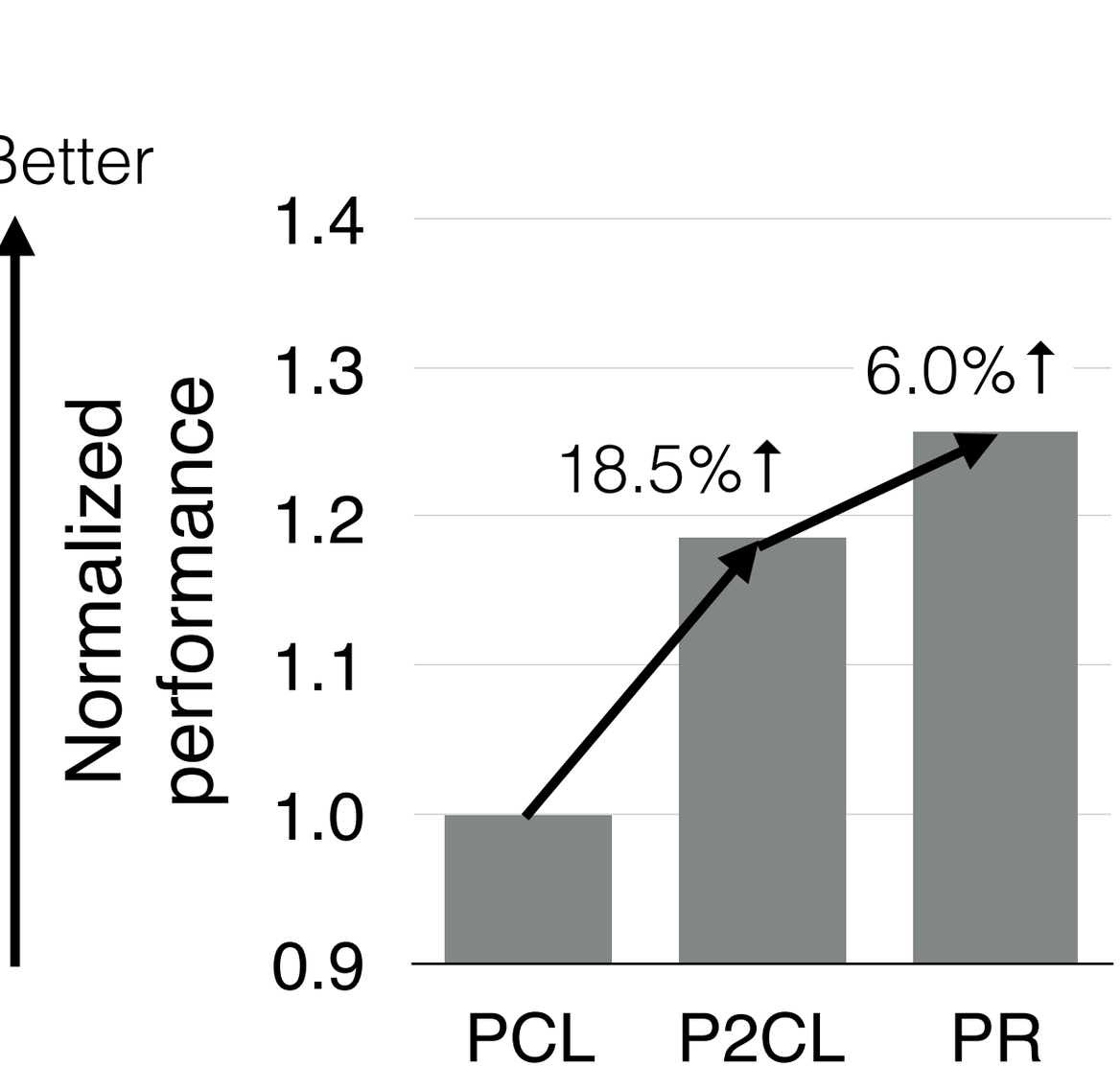
Performance & Power

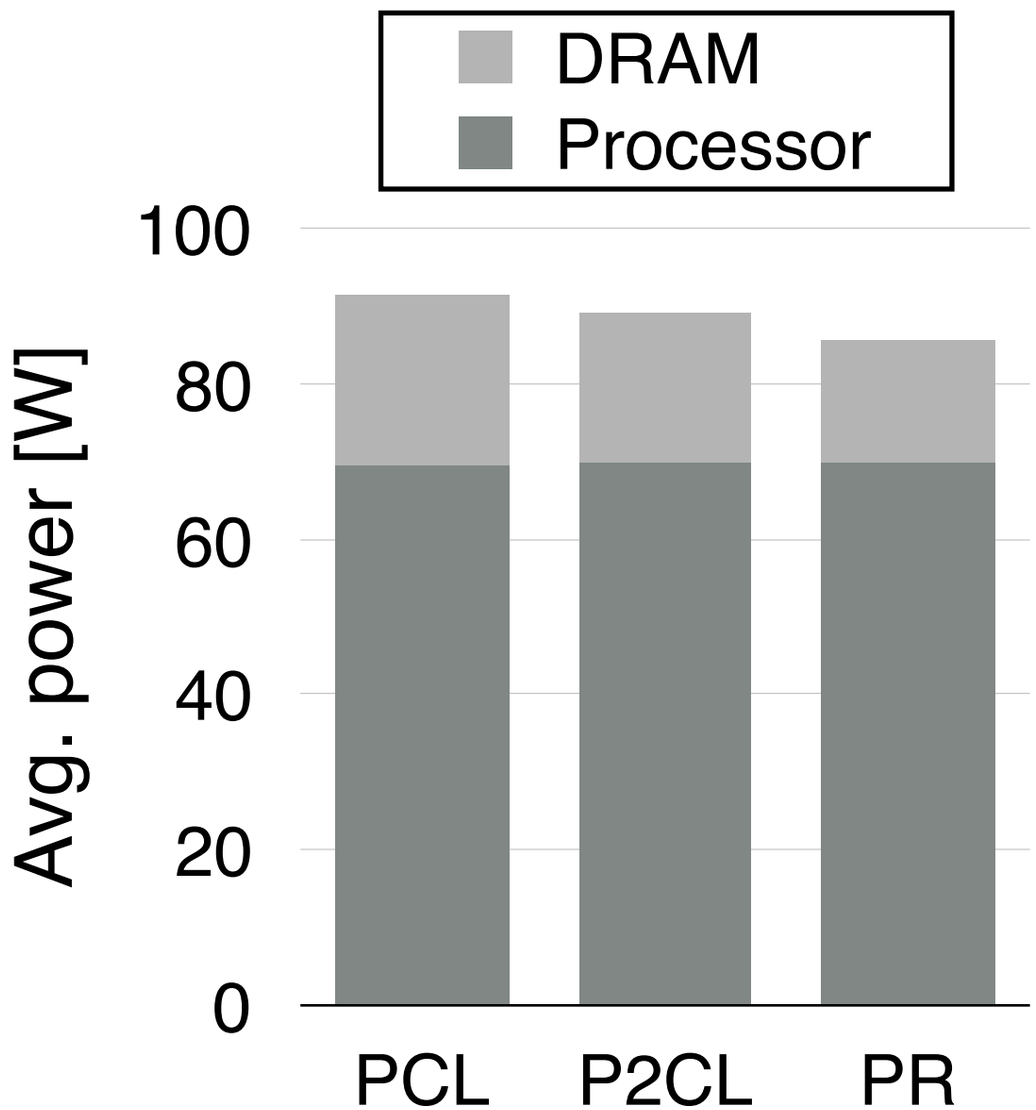


Performance & Power

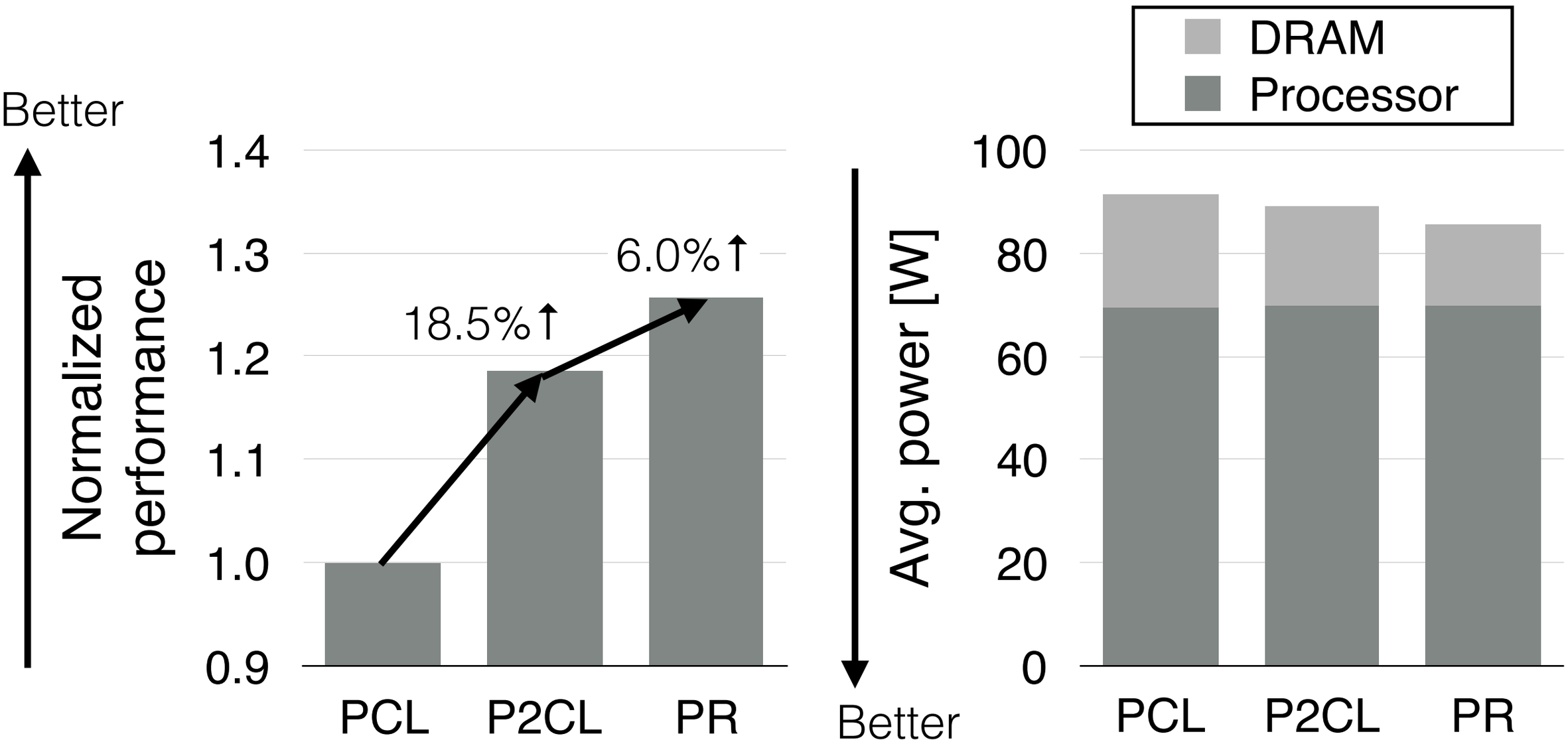


Performance & Power

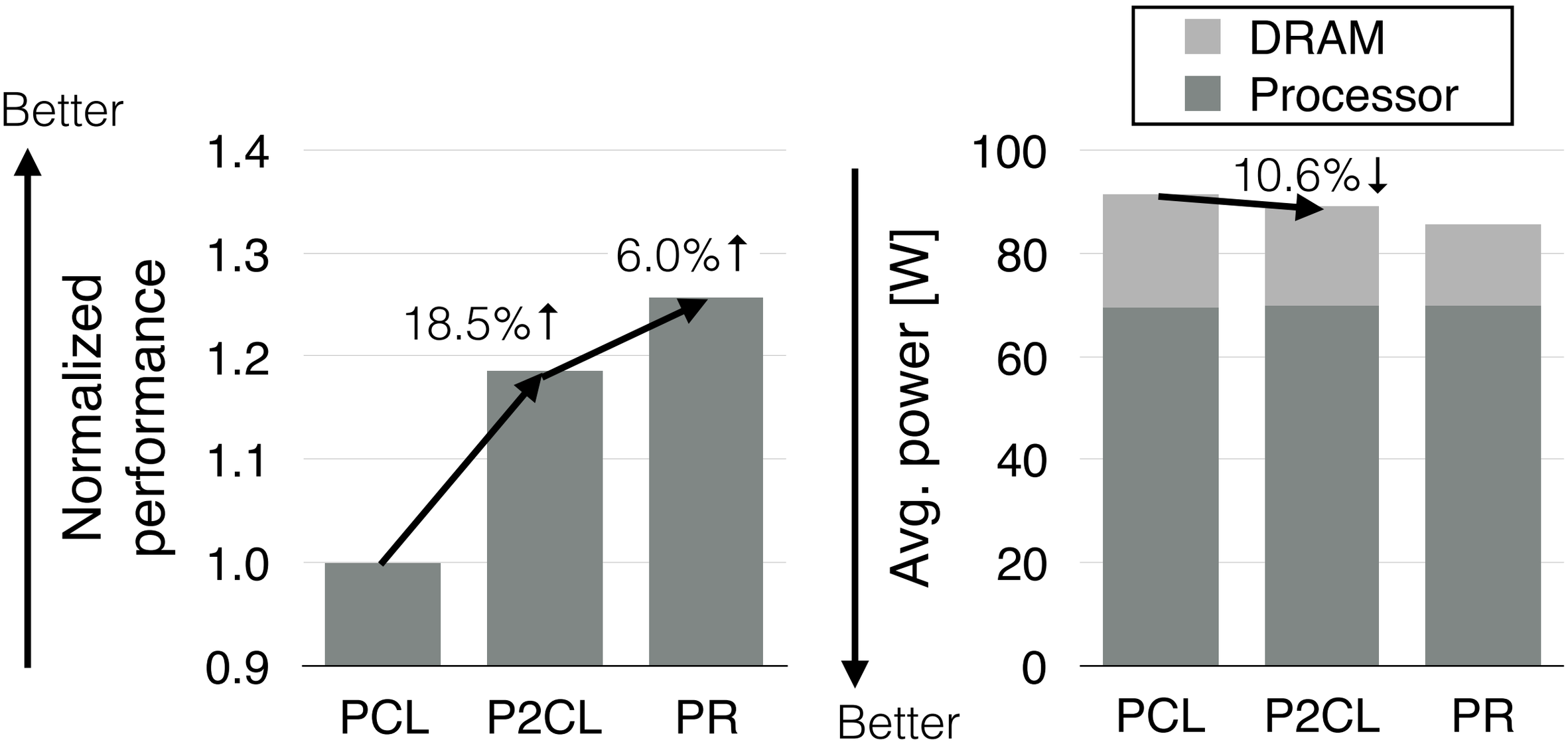




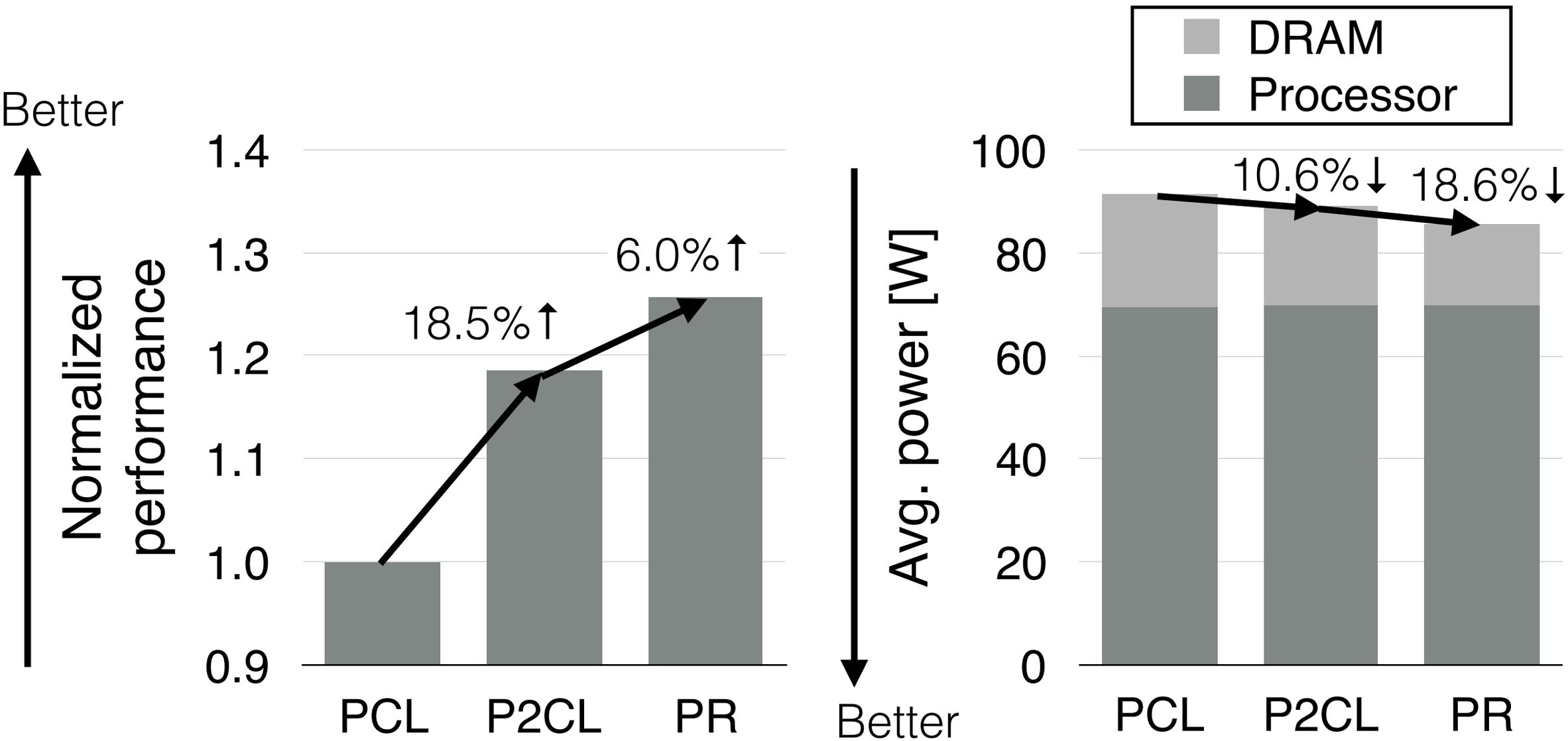
Performance & Power



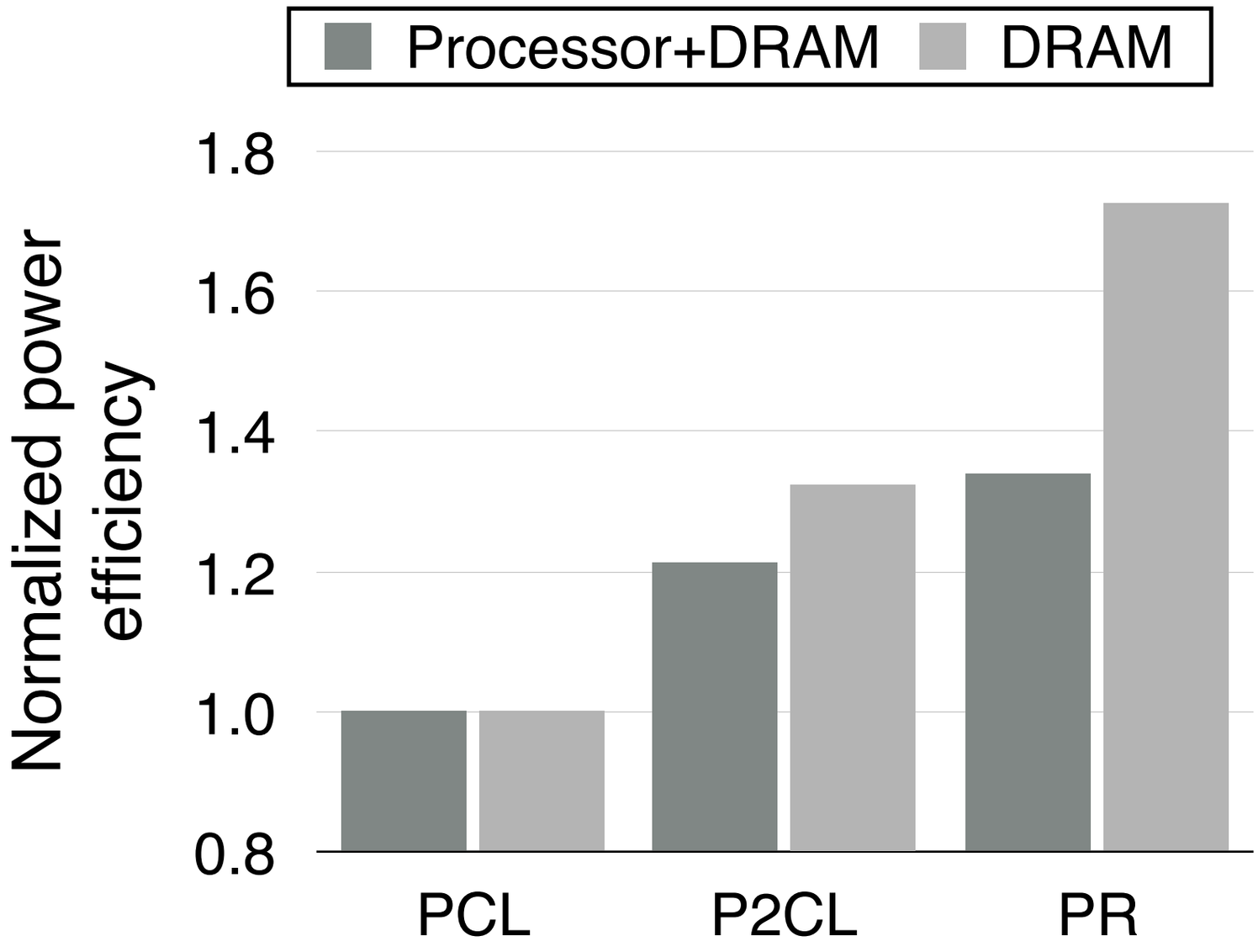
Performance & Power



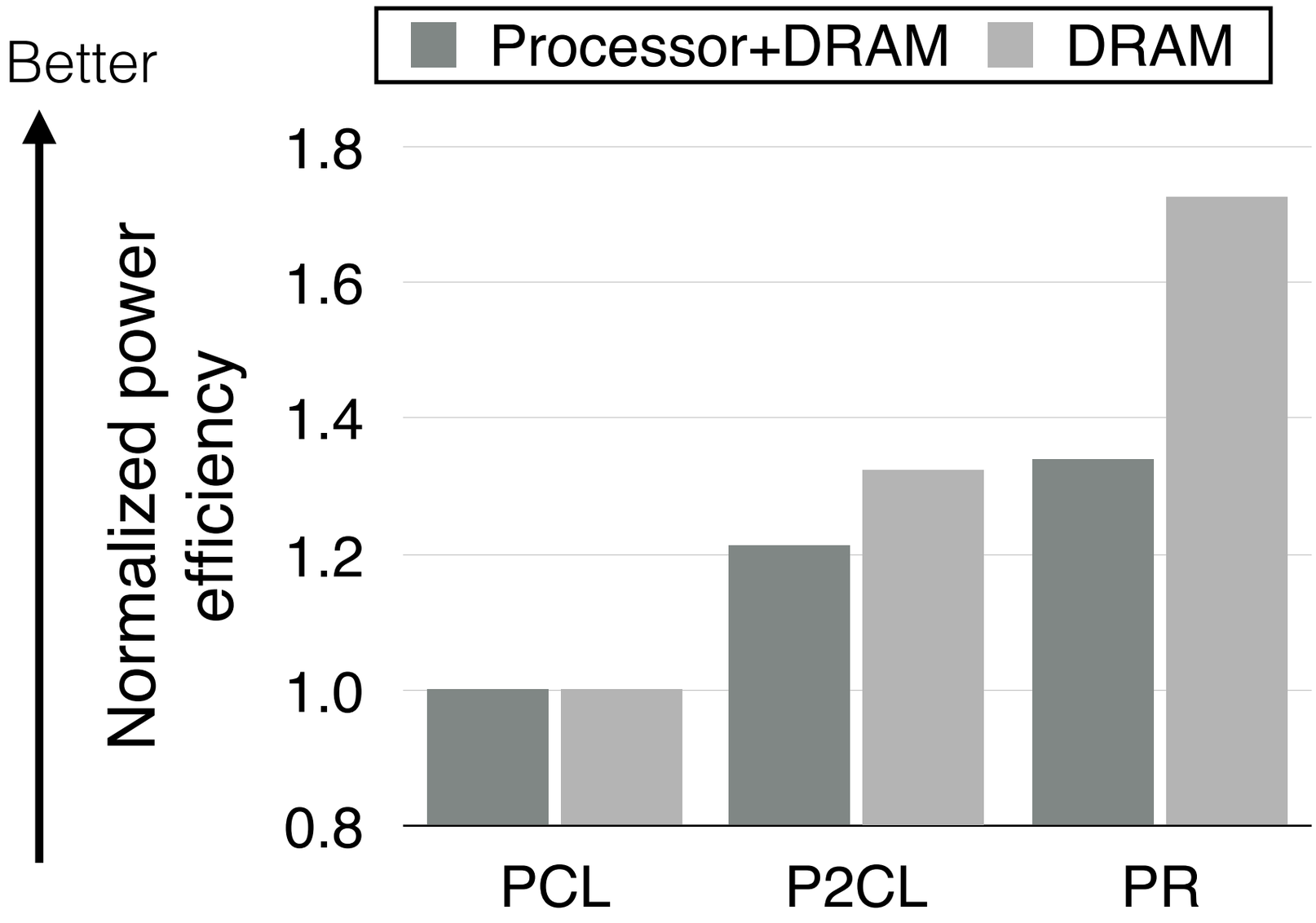
Performance & Power



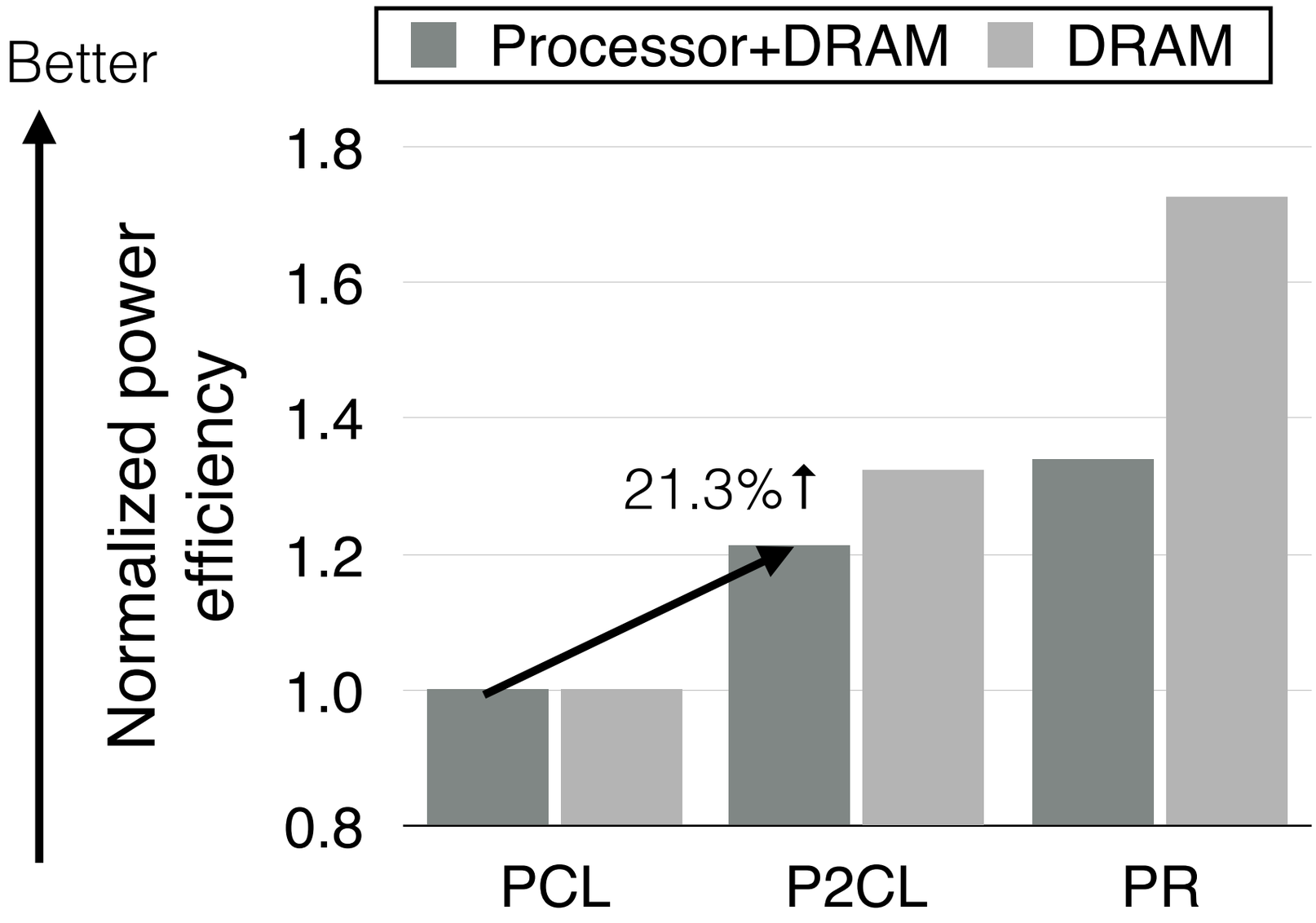
Power Efficiency



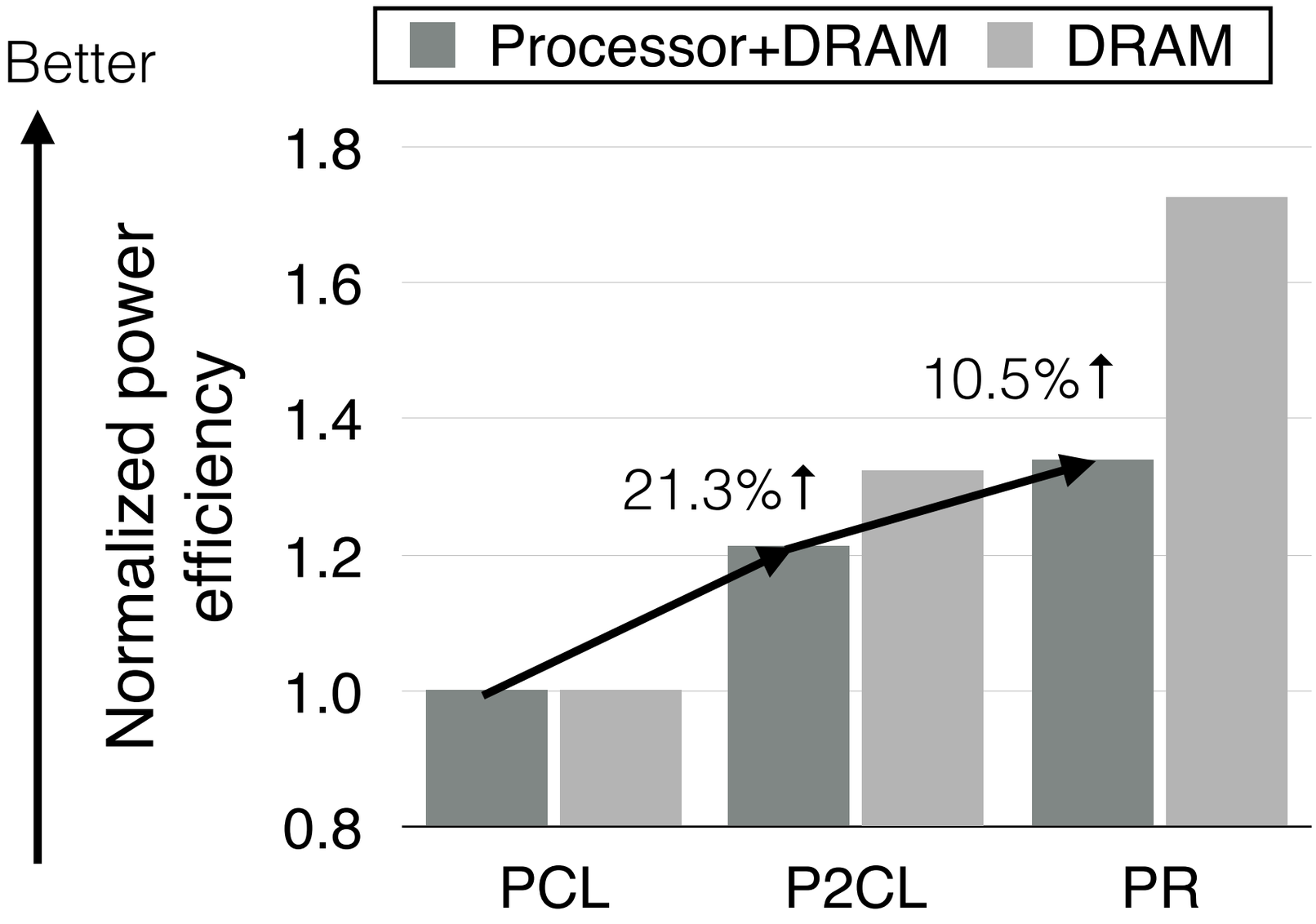
Power Efficiency



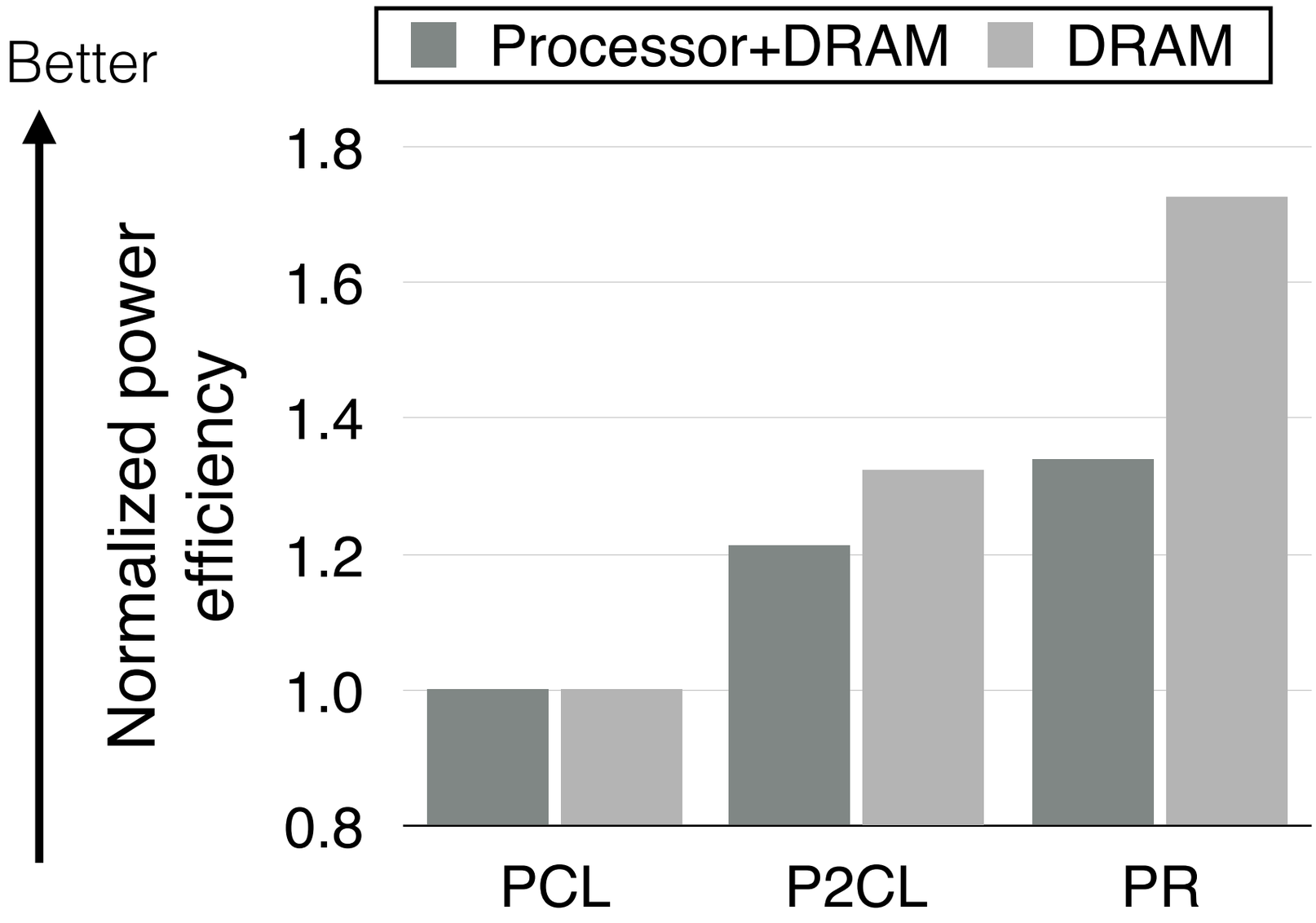
Power Efficiency



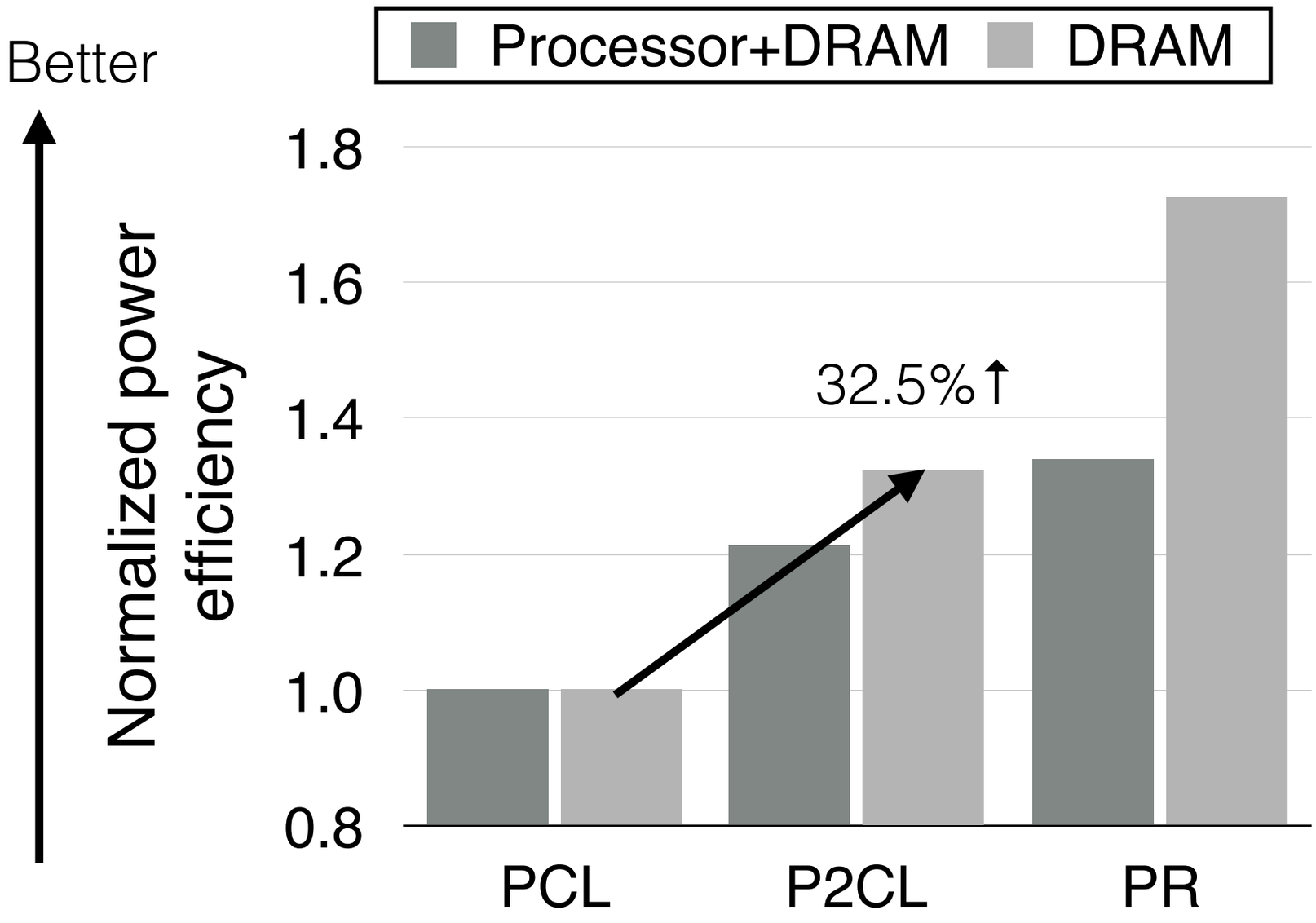
Power Efficiency



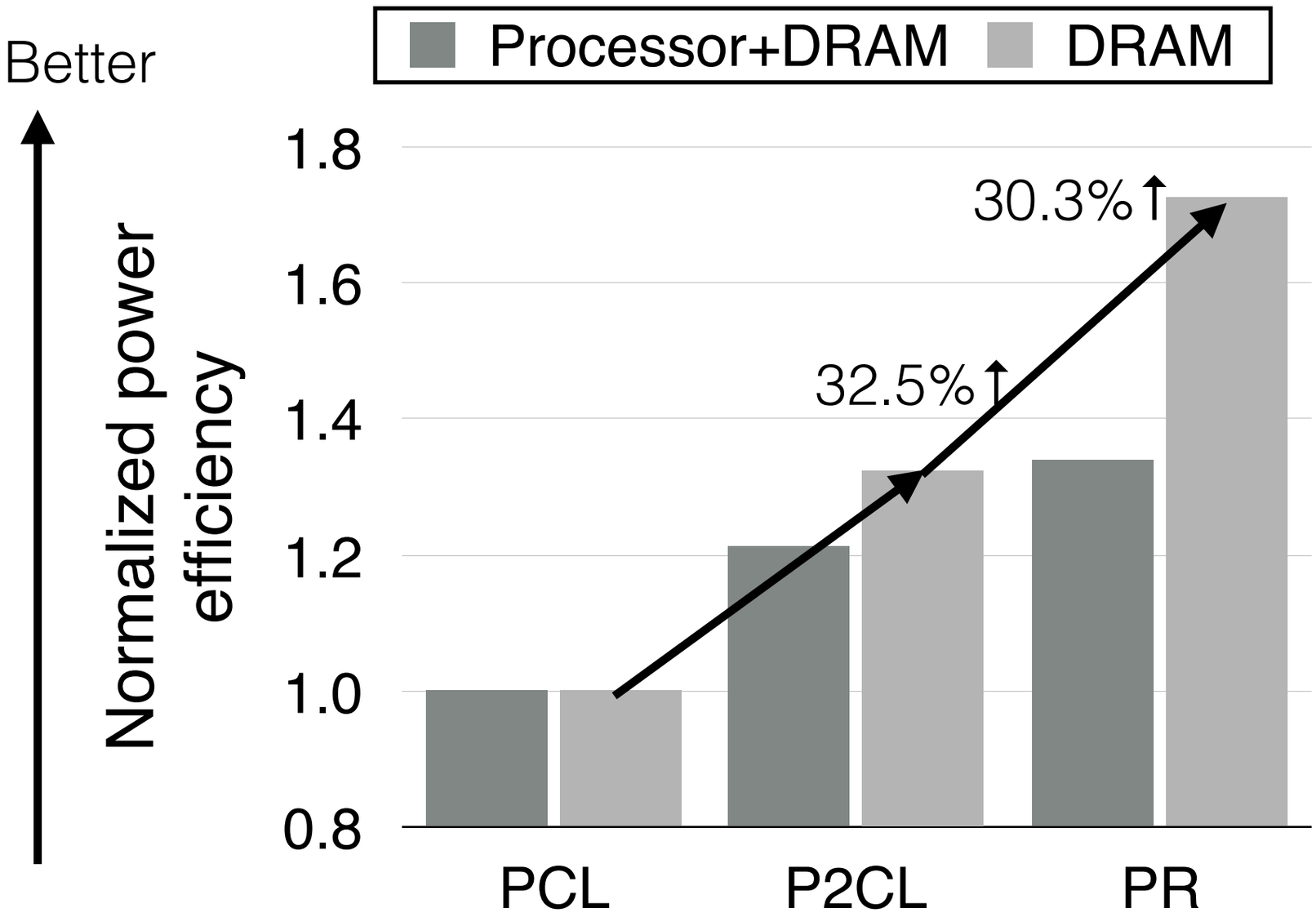
Power Efficiency



Power Efficiency



Power Efficiency



Conclusions

- It is a big challenge to improve the power efficiency of BFS
- Conventional address mapping schemes do not efficiently utilize DRAM in bottom-up algorithm
- We propose per-row channel interleaving
  - It improves RBHR and DRAM power efficiency by 30.3%
- Future work
  - Evaluate PR with other graphs/algorithms/applications with various memory access patterns

Power-Efficient Breadth-First Search with DRAM Row Buffer Locality-Aware Address Mapping

Satoshi Imamura*, Yuichiro Yasui*, Koji Inoue*,
Takatsugu Ono*, Hiroshi Sasaki**, Katsuki Fujisawa*

*Kyushu University, **Columbia University