Barcelona
Supercomputing
Center
Centro Nacional de Supercomputación

EXCELENCIA
SEVERO
OCHOA

# Memory system performance

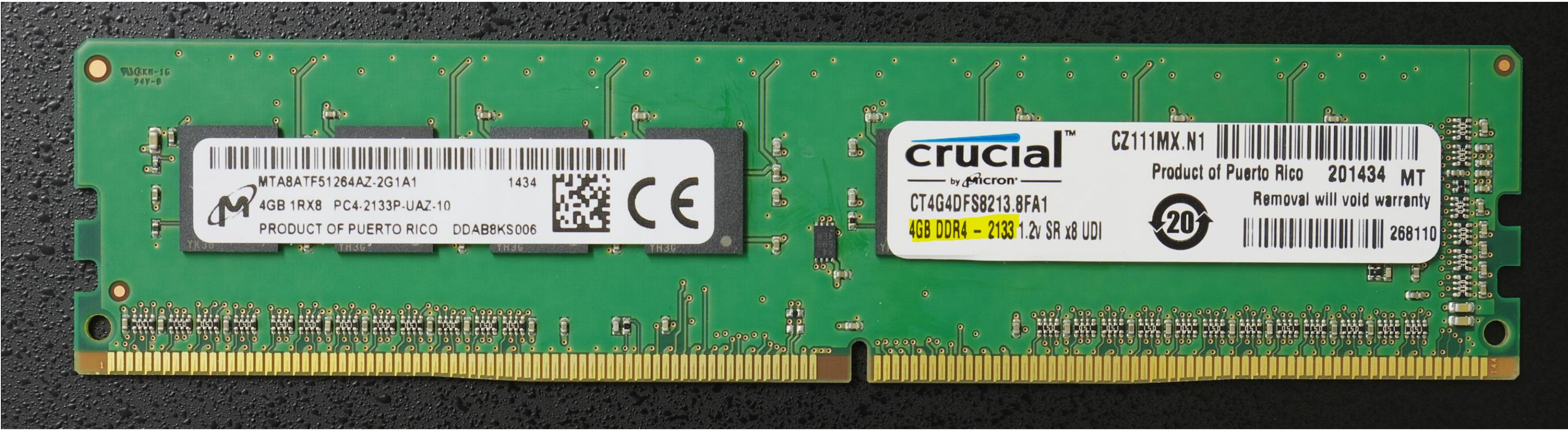
Petar Radojkovic
BSC memory systems team leader

5. November 2021.

DEEP-SEA weekly seminars

# What is memory system performance?

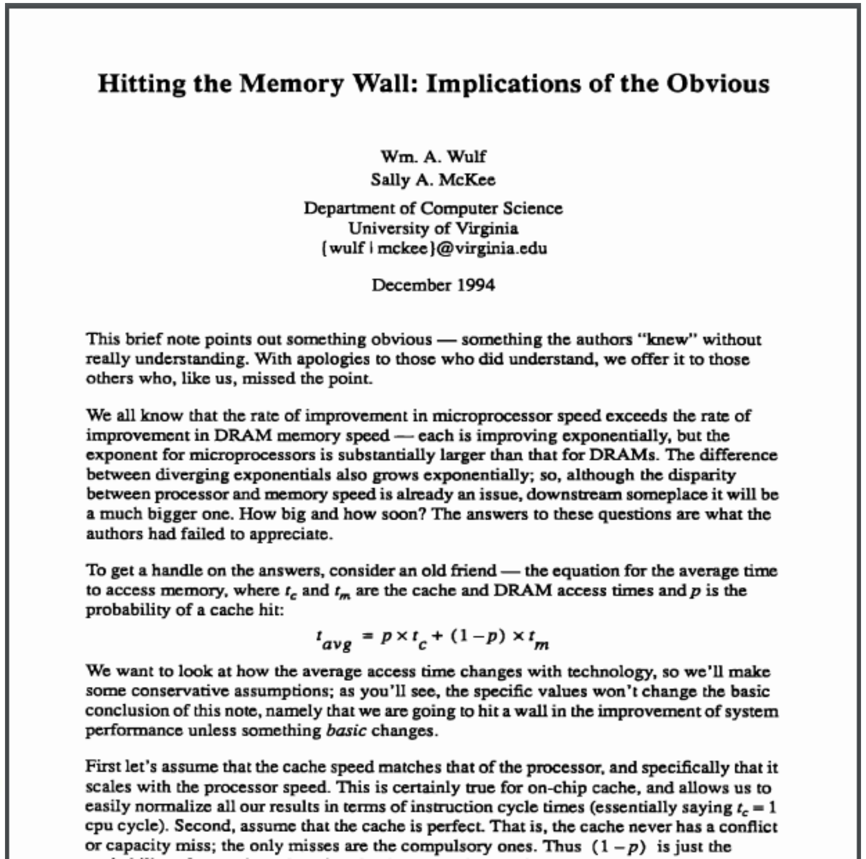
# Memory performance = Bandwidth


MTA8ATF51264AZ-2G1A1
1434
4GB 1RX8 PC4-2133P-UAZ-10
PRODUCT OF PUERTO RICO DDAB8KS006
crucial™
by Micron
CZ111MX.N1
Product of Puerto Rico 201434 MT
CT4G4DFS8213.8FA1
4GB DDR4 – 2133 1.2v SR x8 UDI
Removal will void warranty
268110
20

- DDR4-2133
    - Bandwidth = 2133 * 64Byte * …

# Memory performance = Latency

- Memory latency is what stalls the pipeline (execution)
- Memory wall is about *latency*



## Hitting the Memory Wall: Implications of the Obvious

Wm. A. Wulf
Sally A. McKee

Department of Computer Science
University of Virginia
{wulf | mckee}@virginia.edu

December 1994

This brief note points out something obvious — something the authors "knew" without really understanding. With apologies to those who did understand, we offer it to those others who, like us, missed the point.

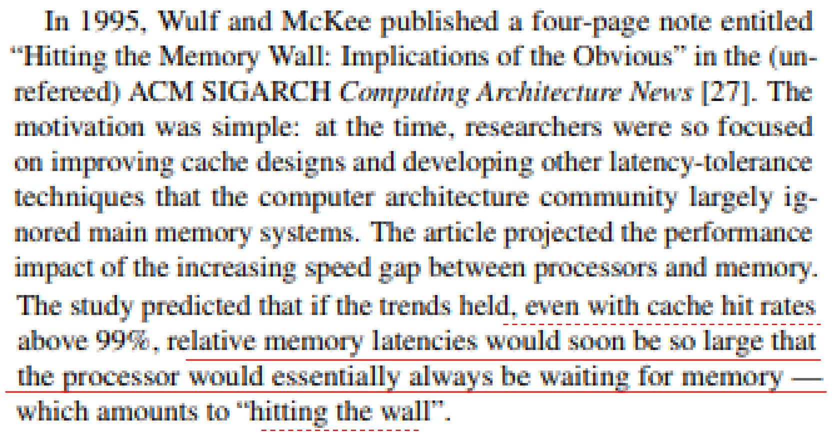
We all know that the rate of improvement in microprocessor speed exceeds the rate of improvement in DRAM memory speed — each is improving exponentially, but the exponent for microprocessors is substantially larger than that for DRAMs. The difference between diverging exponentials also grows exponentially; so, although the disparity between processor and memory speed is already an issue, downstream someplace it will be a much bigger one. How big and how soon? The answers to these questions are what the authors had failed to appreciate.

To get a handle on the answers, consider an old friend — the equation for the average time to access memory, where $t_c$ and $t_m$ are the cache and DRAM access times and $p$ is the probability of a cache hit:

$$t_{avg} = p \times t_c + (1-p) \times t_m$$

We want to look at how the average access time changes with technology, so we'll make some conservative assumptions; as you'll see, the specific values won't change the basic conclusion of this note, namely that we are going to hit a wall in the improvement of system performance unless something *basic* changes.

First let's assume that the cache speed matches that of the processor, and specifically that it scales with the processor speed. This is certainly true for on-chip cache, and allows us to easily normalize all our results in terms of instruction cycle times (essentially saying $t_c = 1$ cpu cycle). Second, assume that the cache is perfect. That is, the cache never has a conflict or capacity miss; the only misses are the compulsory ones. Thus $(1-p)$ is just the



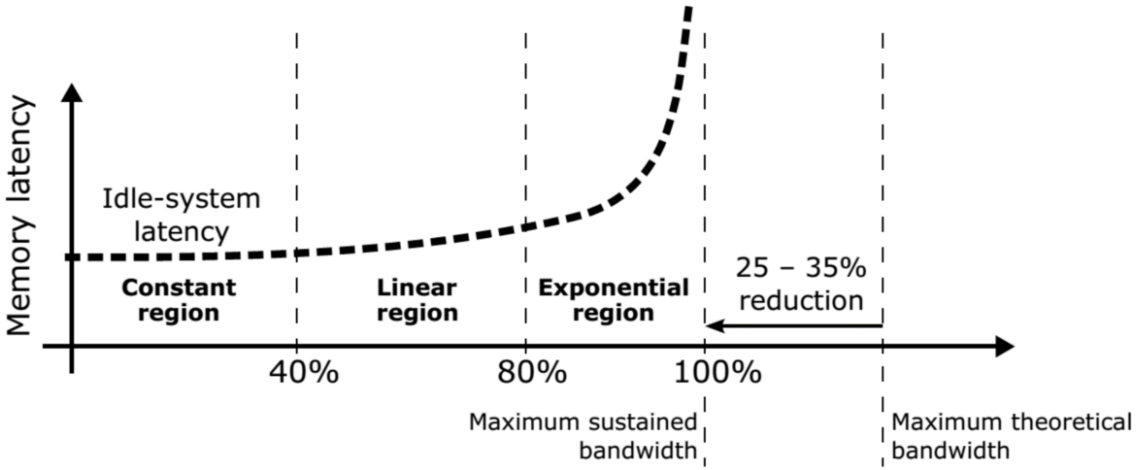
In 1995, Wulf and McKee published a four-page note entitled "Hitting the Memory Wall: Implications of the Obvious" in the (unrefereed) ACM SIGARCH *Computing Architecture News* [27]. The motivation was simple: at the time, researchers were so focused on improving cache designs and developing other latency-tolerance techniques that the computer architecture community largely ignored main memory systems. The article projected the performance impact of the increasing speed gap between processors and memory. The study predicted that if the trends held, even with cache hit rates above 99%, relative memory latencies would soon be so large that the processor would essentially always be waiting for memory — which amounts to "hitting the wall".

Wm. A. Wulf and S. A. McKee.
*Hitting the memory wall: Implications of the obvious.*
Computer architecture news, 1995.

# Memory latency is not a single number

- Memory latenc<u>ies</u>
  - 1: Lead-off (unloaded) latency
  - 2: Loaded latency

- Memory latency curve
  - Memory latency = f(memory system stress)

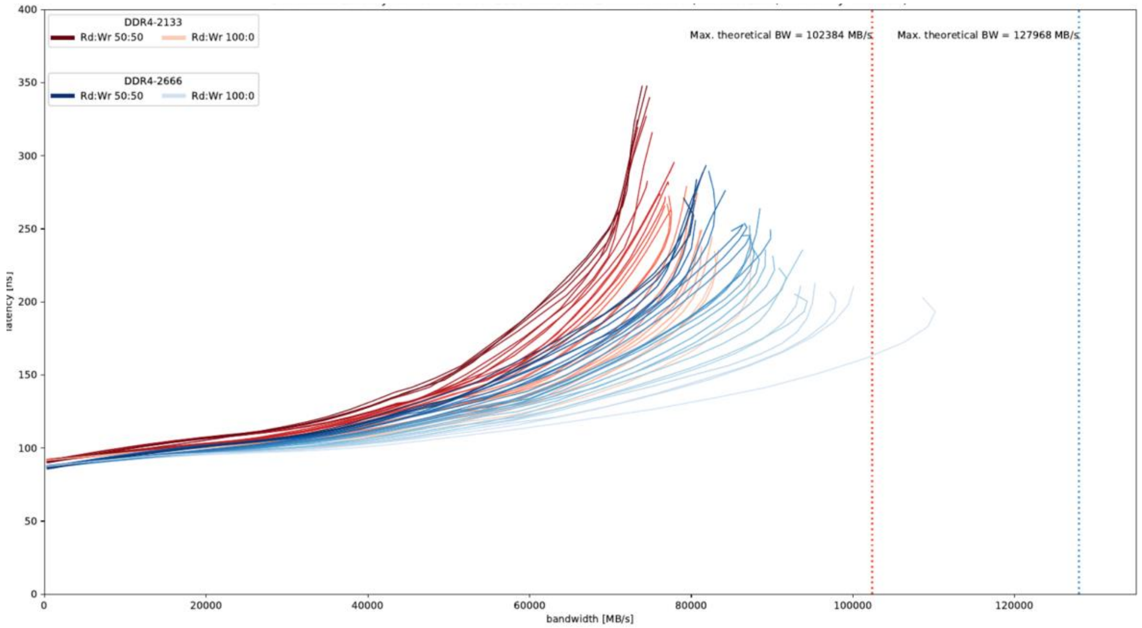


B. L. Jacob.
*The memory system: You can’t avoid it, you can’t ignore it, you can’t fake it.*
Synthesis Lectures on Computer Architecture, 2009.

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

# Memory latency curves

- Actual measurements
  - Dual socket Intel Xeon 8260 CPU (Cascade Lake)
    - 24 cores @ 2.4 GHz
  - 6 DDR4 memory channels per socket



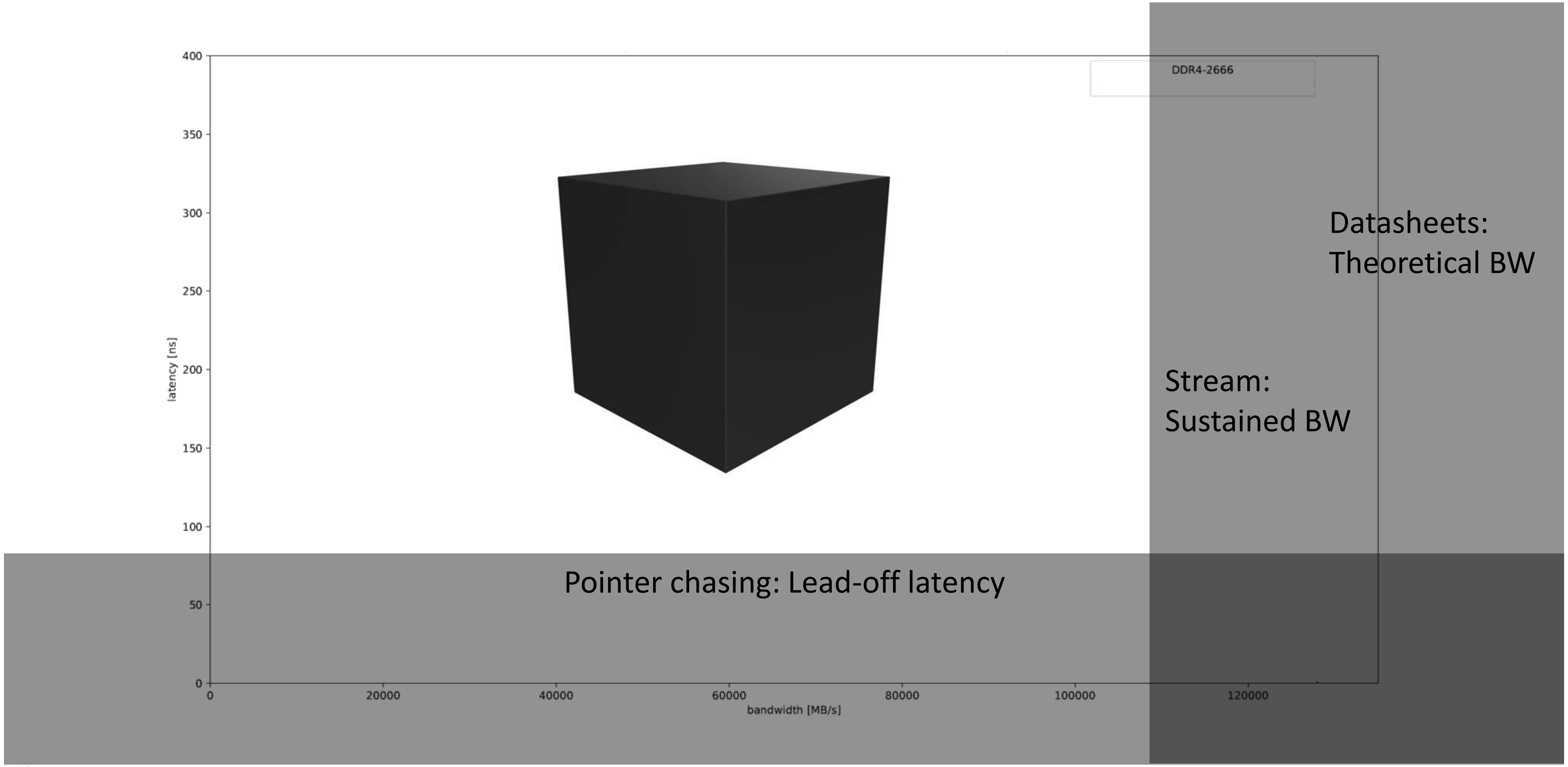
  - Measured with memory stressing benchmark (developed by the BSC)
    - Enhanced Stream benchmark (memory stress, X-axis) & Pointer chasing benchmark (latency , Y-axis)

  - Open source: [https://github.com/bsc-mem/PROFET](https://github.com/bsc-mem/PROFET)

    **New code release coming soon!**

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

# Memory performance: Looking inside the box

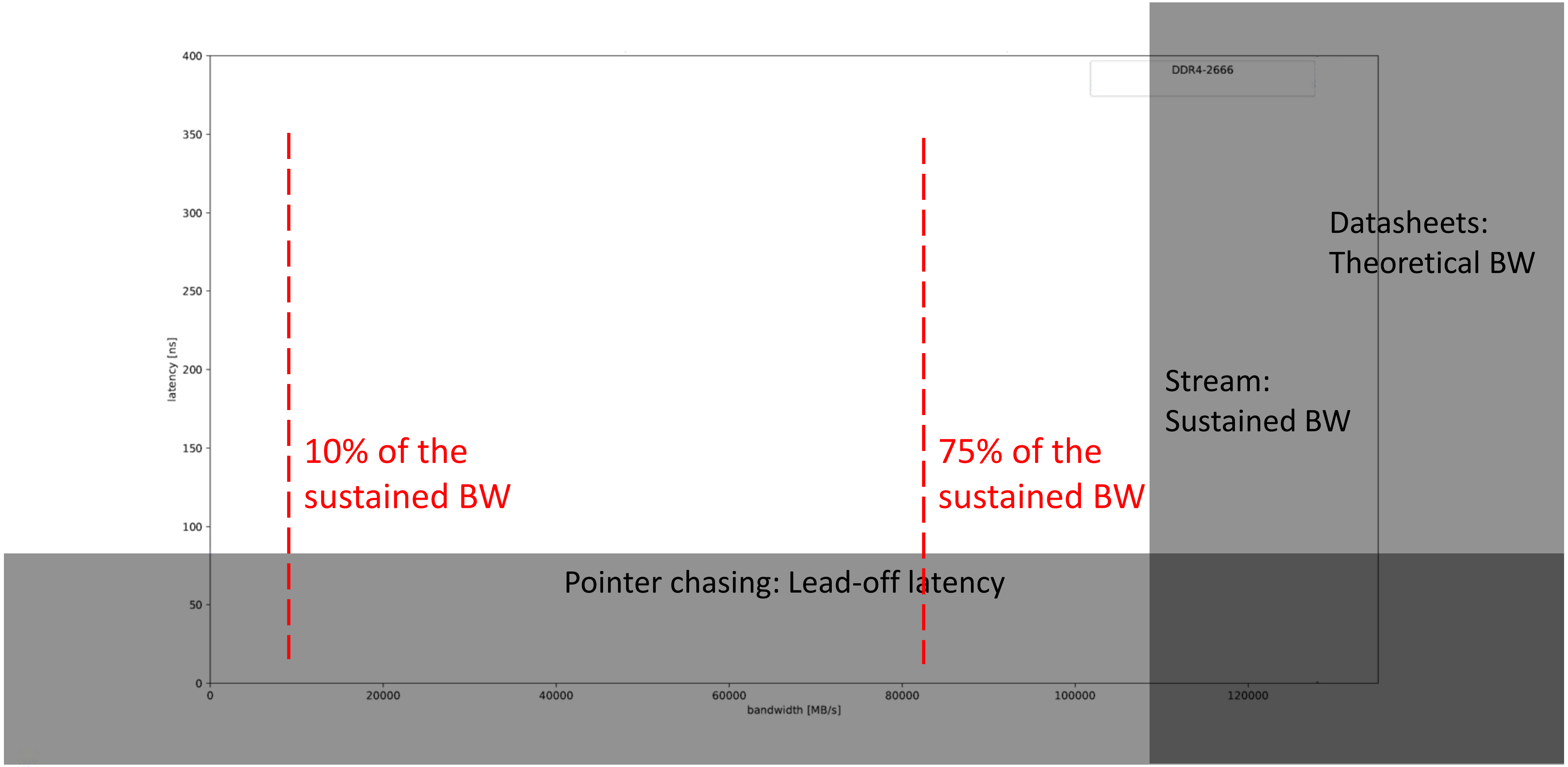


Datasheets: Theoretical BW

Stream: Sustained BW

Pointer chasing: Lead-off latency

# Use case 1: Application profiling



10% of the sustained BW

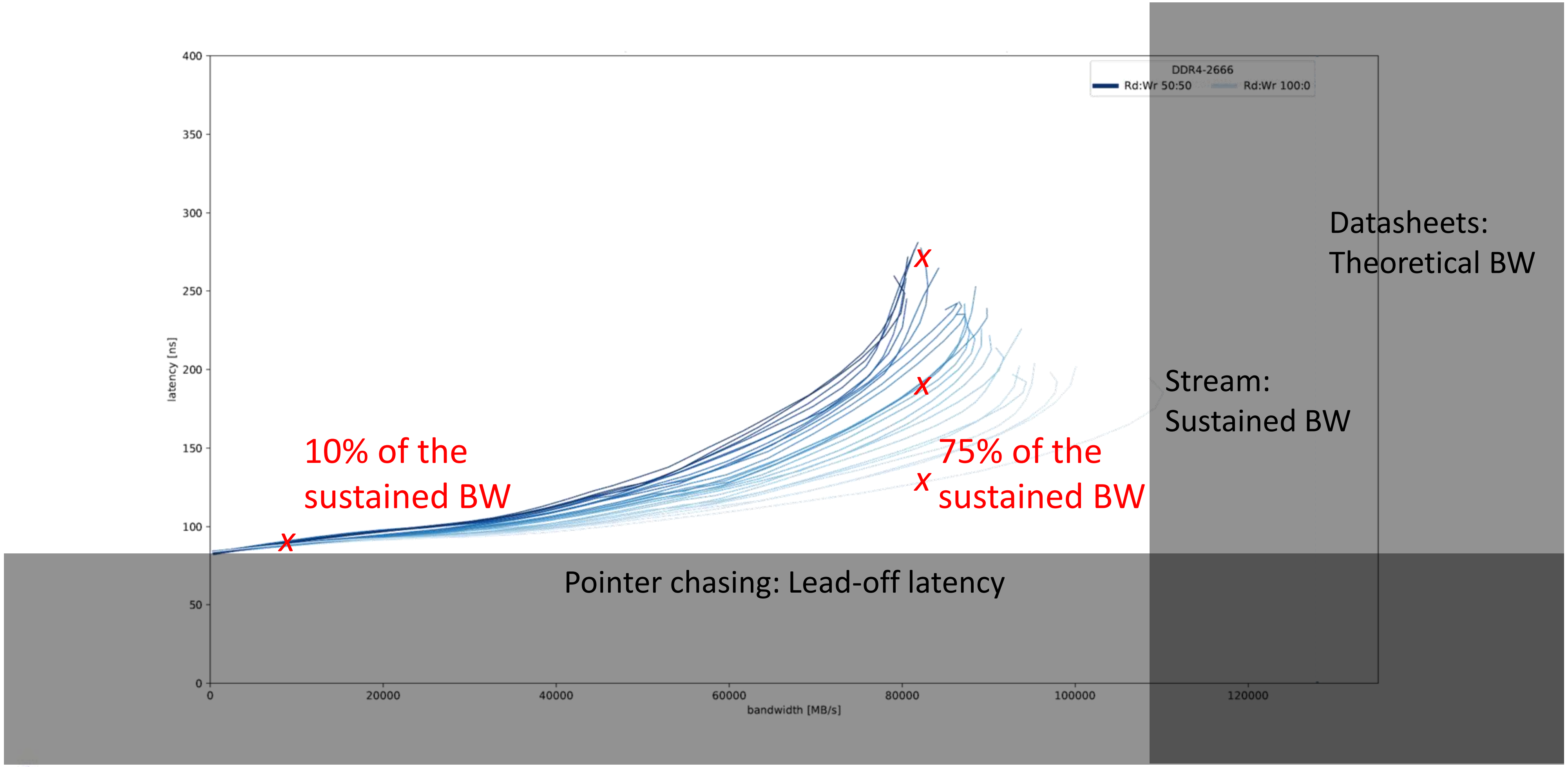
75% of the sustained BW

Stream: Sustained BW

Datasheets: Theoretical BW

Pointer chasing: Lead-off latency

# Use case 1: Application profiling



Datasheets: Theoretical BW

Stream: Sustained BW

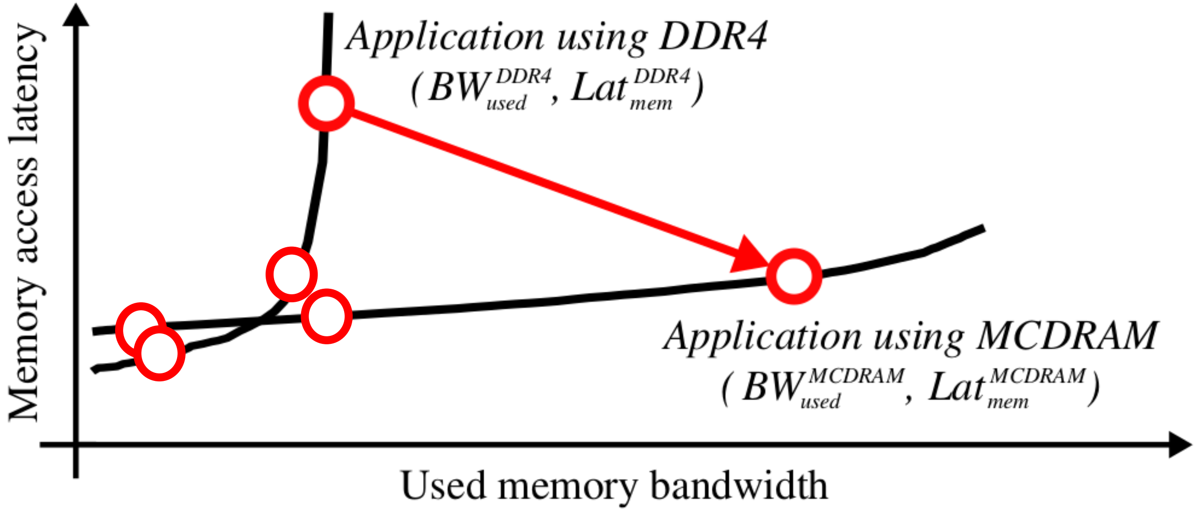
10% of the sustained BW

75% of the sustained BW

Pointer chasing: Lead-off latency

# Use case 2:
# What (not) to expect from a novel memory system

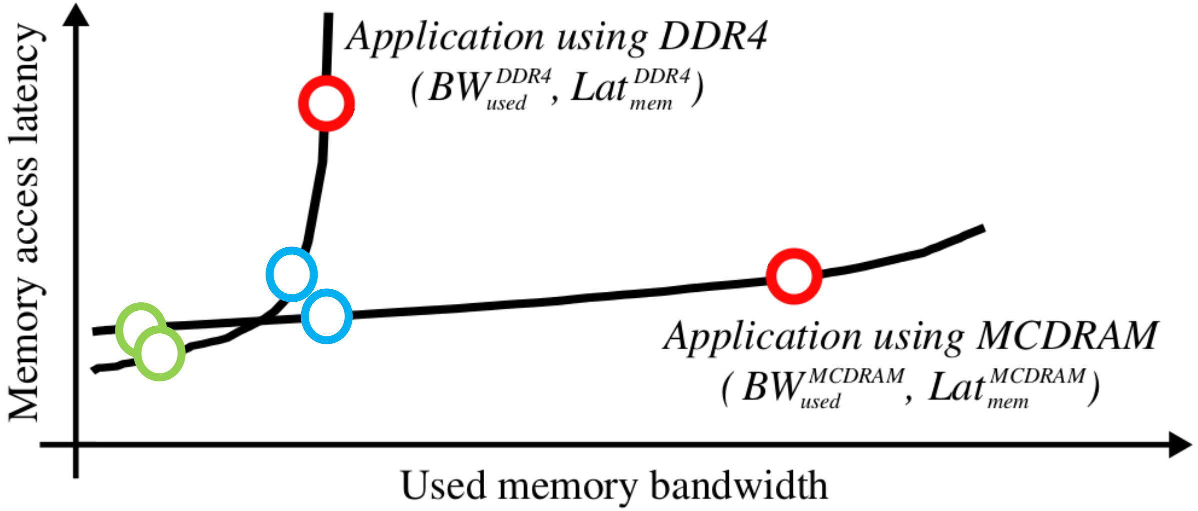
- DDR4 and MCDRAM at KNL platform
  - *“MCDRAM provides an N-fold higher performance”*



- Similar-ish analysis can be done for DDR4 and Optane (Storage Class Memory)

# Use case 3:
# Data distribution in heterogeneous memory systems

- How to distribute red, blue and green data structures in heterogeneous memory system
    - DDR4 and MCDRAM at KNL platform



- Similar-ish analysis can be done for DDR4 and Optane (Storage Class Memory)

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

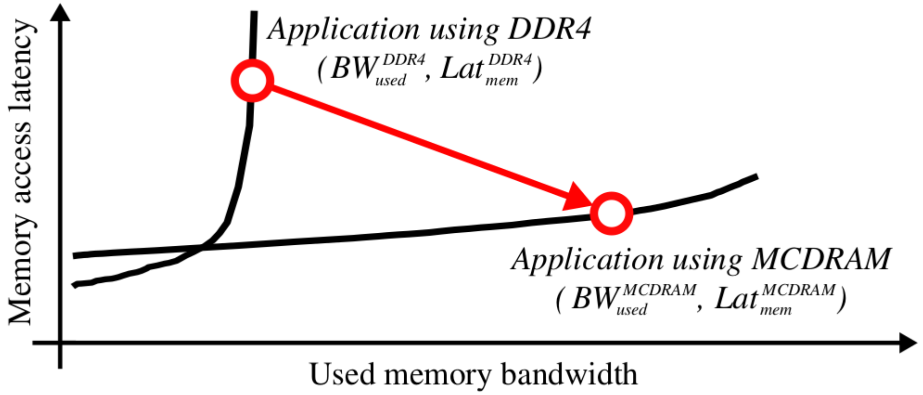
# Use case 4:
# Memory system design space exploration

- Overall system performance = f (mem system)?

- *“Can we estimate application performance with <u>different (new) main memory</u>?”*
  - Example: High-end CPU with DDR4-XXXX
    - DDR4 (different frequencies)
    - DDR5
    - HBM2/3
    - LPDDR4/5
    - …
    - Optane
    - Memory over CXL (CCIX, etc.)
    - …
    - Novel memory designs

# Our idea: PROFET – PROfile & EsTimate

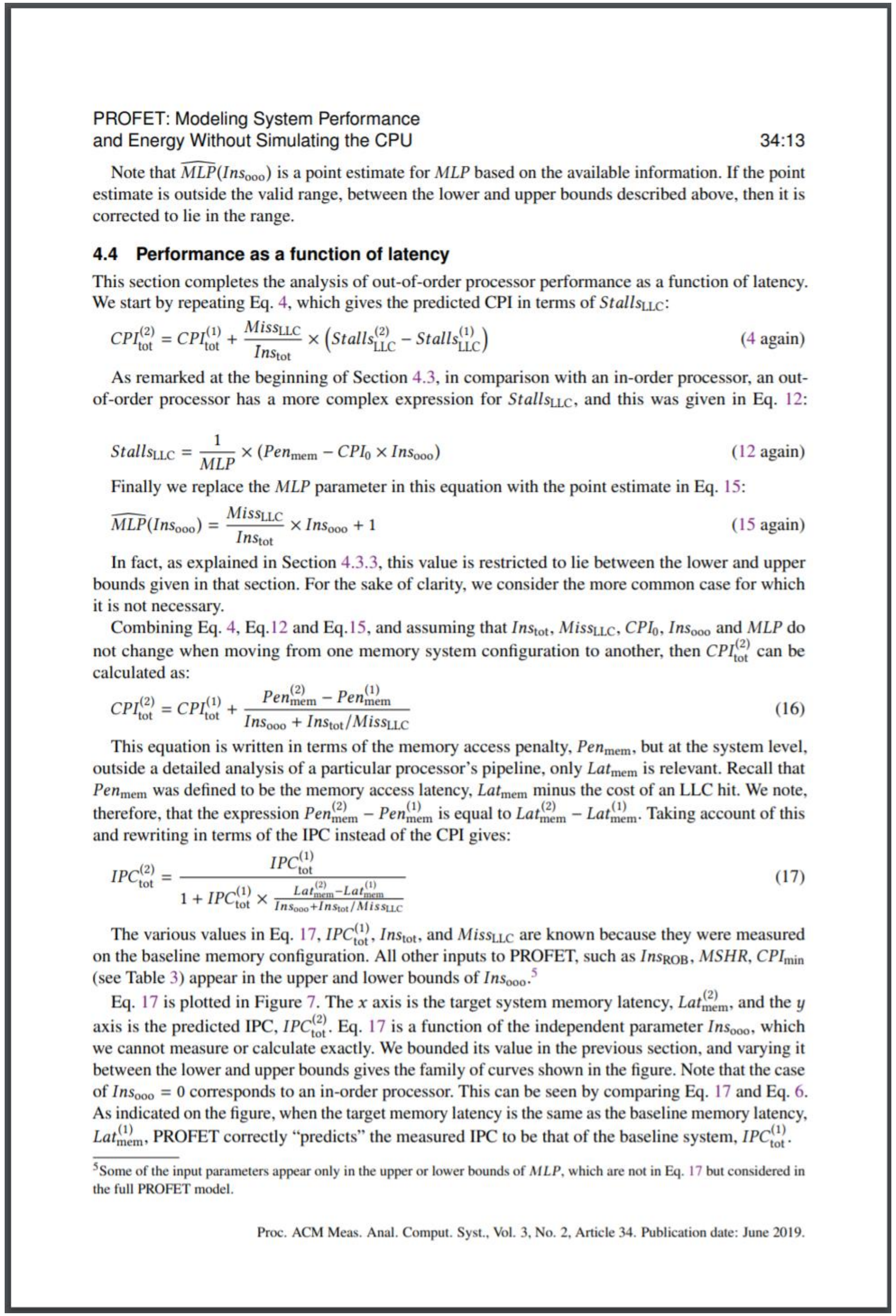
- **PROfile**
  - Memory latency curves
    - Conventional memory system
    - Novel memory system
  - Application running on an actual system



- **EsTimate**
  - Application performance, power and energy with the novel memory system

# PROFET modeling

- Based on strong profiling capabilities of high-end processors (hardware counters)
- Parameters that we cannot measure
  - Set the boundaries based on the microarchitectural specification (e.g., row-buffer size)
- Analytically model performance – Equations
- Solve the equations
- Published:
  Radulovic et al., ***PROFET: Modeling System Performance and Energy Without Simulating the CPU***. SIGMETRICS, 2019.
- Open source:
  https://github.com/bsc-mem/PROFET
  **New code release coming soon(-ish)! ;-)**



Note that $\widehat{MLP}(Ins_{\text{ooo}})$ is a point estimate for $MLP$ based on the available information. If the point estimate is outside the valid range, between the lower and upper bounds described above, then it is corrected to lie in the range.

### 4.4 Performance as a function of latency

This section completes the analysis of out-of-order processor performance as a function of latency. We start by repeating Eq. 4, which gives the predicted CPI in terms of $Stalls_{\text{LLC}}$:

$$CPI_{\text{tot}}^{(2)} = CPI_{\text{tot}}^{(1)} + \frac{Miss_{\text{LLC}}}{Ins_{\text{tot}}} \times \left(Stalls_{\text{LLC}}^{(2)} - Stalls_{\text{LLC}}^{(1)}\right) \tag{4 again}$$

As remarked at the beginning of Section 4.3, in comparison with an in-order processor, an out-of-order processor has a more complex expression for $Stalls_{\text{LLC}}$, and this was given in Eq. 12:

$$Stalls_{\text{LLC}} = \frac{1}{MLP} \times (Pen_{\text{mem}} - CPI_0 \times Ins_{\text{ooo}}) \tag{12 again}$$

Finally we replace the $MLP$ parameter in this equation with the point estimate in Eq. 15:

$$\widehat{MLP}(Ins_{\text{ooo}}) = \frac{Miss_{\text{LLC}}}{Ins_{\text{tot}}} \times Ins_{\text{ooo}} + 1 \tag{15 again}$$

In fact, as explained in Section 4.3.3, this value is restricted to lie between the lower and upper bounds given in that section. For the sake of clarity, we consider the more common case for which it is not necessary.

Combining Eq. 4, Eq.12 and Eq.15, and assuming that $Ins_{\text{tot}}$, $Miss_{\text{LLC}}$, $CPI_0$, $Ins_{\text{ooo}}$ and $MLP$ do not change when moving from one memory system configuration to another, then $CPI_{\text{tot}}^{(2)}$ can be calculated as:

$$CPI_{\text{tot}}^{(2)} = CPI_{\text{tot}}^{(1)} + \frac{Pen_{\text{mem}}^{(2)} - Pen_{\text{mem}}^{(1)}}{Ins_{\text{ooo}} + Ins_{\text{tot}}/Miss_{\text{LLC}}} \tag{16}$$

This equation is written in terms of the memory access penalty, $Pen_{\text{mem}}$, but at the system level, outside a detailed analysis of a particular processor's pipeline, only $Lat_{\text{mem}}$ is relevant. Recall that $Pen_{\text{mem}}$ was defined to be the memory access latency, $Lat_{\text{mem}}$ minus the cost of an LLC hit. We note, therefore, that the expression $Pen_{\text{mem}}^{(2)} - Pen_{\text{mem}}^{(1)}$ is equal to $Lat_{\text{mem}}^{(2)} - Lat_{\text{mem}}^{(1)}$. Taking account of this and rewriting in terms of the IPC instead of the CPI gives:

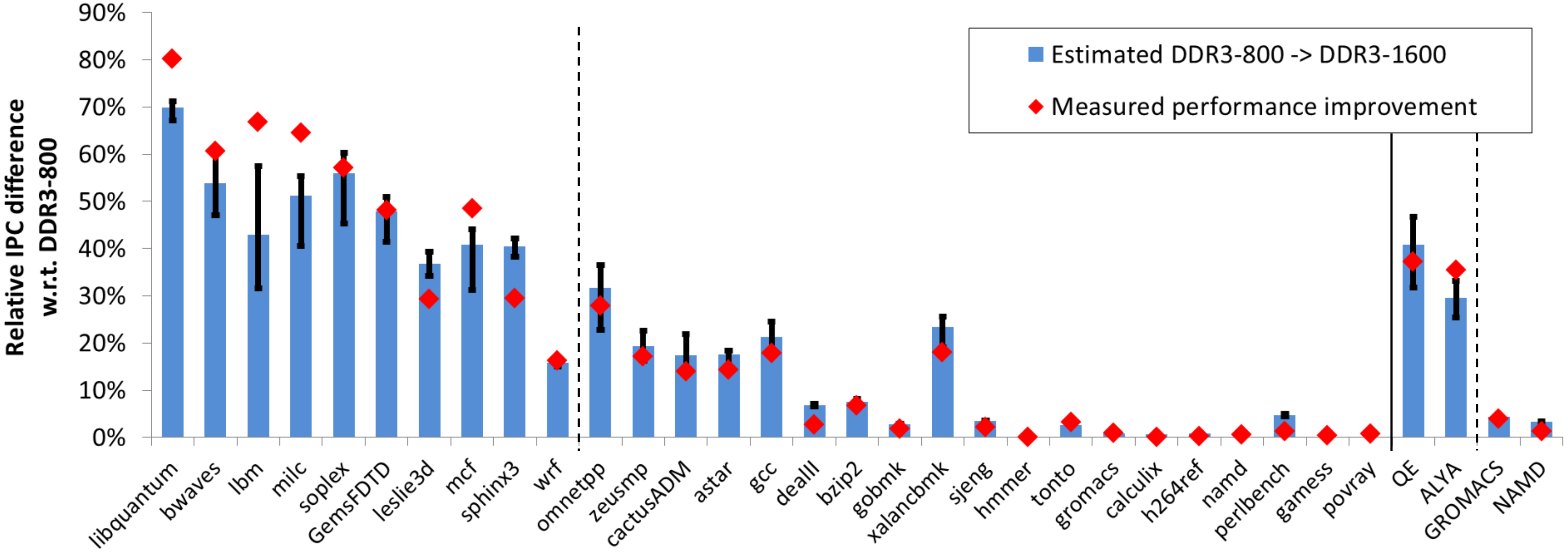
$$IPC_{\text{tot}}^{(2)} = \frac{IPC_{\text{tot}}^{(1)}}{1 + IPC_{\text{tot}}^{(1)} \times \frac{Lat_{\text{mem}}^{(2)} - Lat_{\text{mem}}^{(1)}}{Ins_{\text{ooo}} + Ins_{\text{tot}}/Miss_{\text{LLC}}}} \tag{17}$$

The various values in Eq. 17, $IPC_{\text{tot}}^{(1)}$, $Ins_{\text{tot}}$, and $Miss_{\text{LLC}}$ are known because they were measured on the baseline memory configuration. All other inputs to PROFET, such as $Ins_{\text{ROB}}$, $MSHR$, $CPI_{\text{min}}$ (see Table 3) appear in the upper and lower bounds of $Ins_{\text{ooo}}$.[5]

Eq. 17 is plotted in Figure 7. The $x$ axis is the target system memory latency, $Lat_{\text{mem}}^{(2)}$, and the $y$ axis is the predicted IPC, $IPC_{\text{tot}}^{(2)}$. Eq. 17 is a function of the independent parameter $Ins_{\text{ooo}}$, which we cannot measure or calculate exactly. We bounded its value in the previous section, and varying it between the lower and upper bounds gives the family of curves shown in the figure. Note that the case of $Ins_{\text{ooo}} = 0$ corresponds to an in-order processor. This can be seen by comparing Eq. 17 and Eq. 6. As indicated on the figure, when the target memory latency is the same as the baseline memory latency, $Lat_{\text{mem}}^{(1)}$, PROFET correctly "predicts" the measured IPC to be that of the baseline system, $IPC_{\text{tot}}^{(1)}$.

[5] Some of the input parameters appear only in the upper or lower bounds of $MLP$, which are not in Eq. 17 but considered in the full PROFET model.

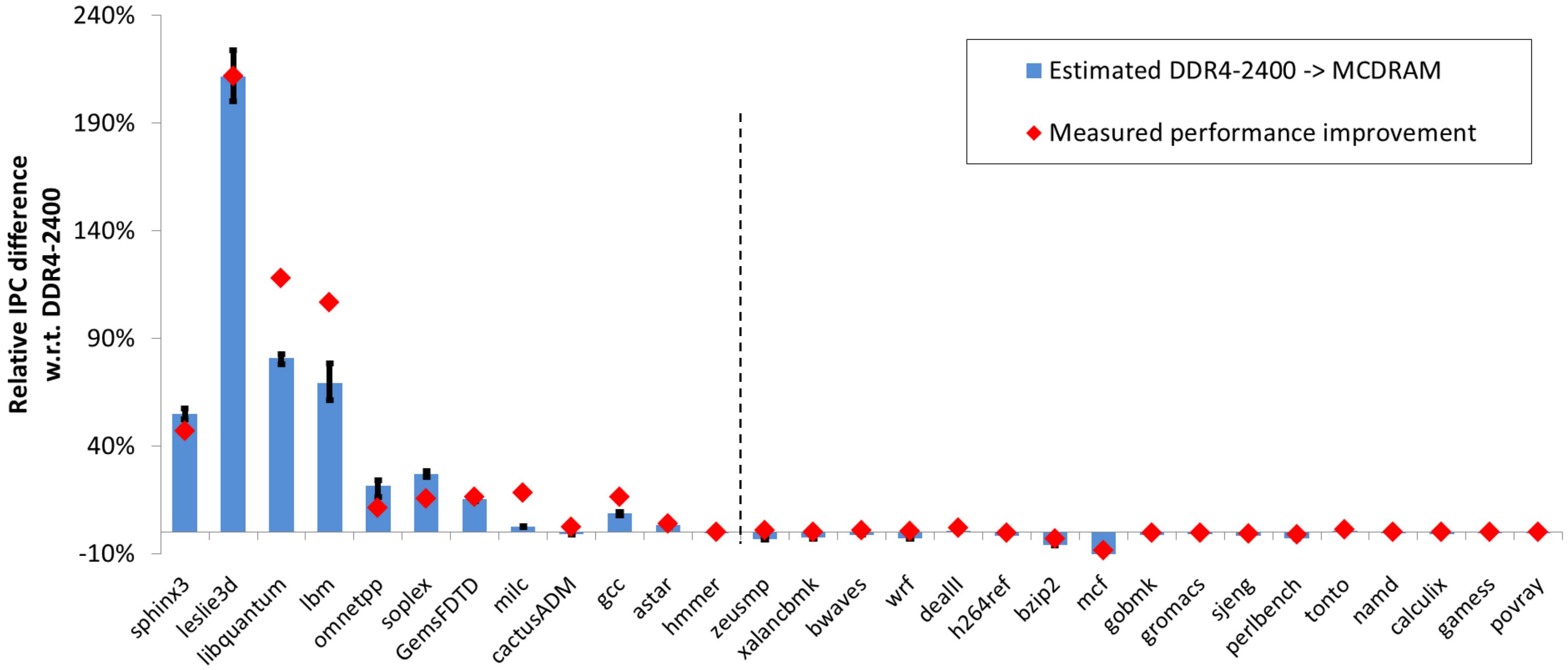
# Evaluation: Sandy Bridge DDR3-800 → DDR3-1600



- Performance estimation error
  - High-bandwidth benchmarks: 5.3%
  - All benchmarks: 2.9%
- Power and energy
  - All benchmarks: 1% and 2%

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

# Evaluation: KNL DDR4-2400 → MCDRAM



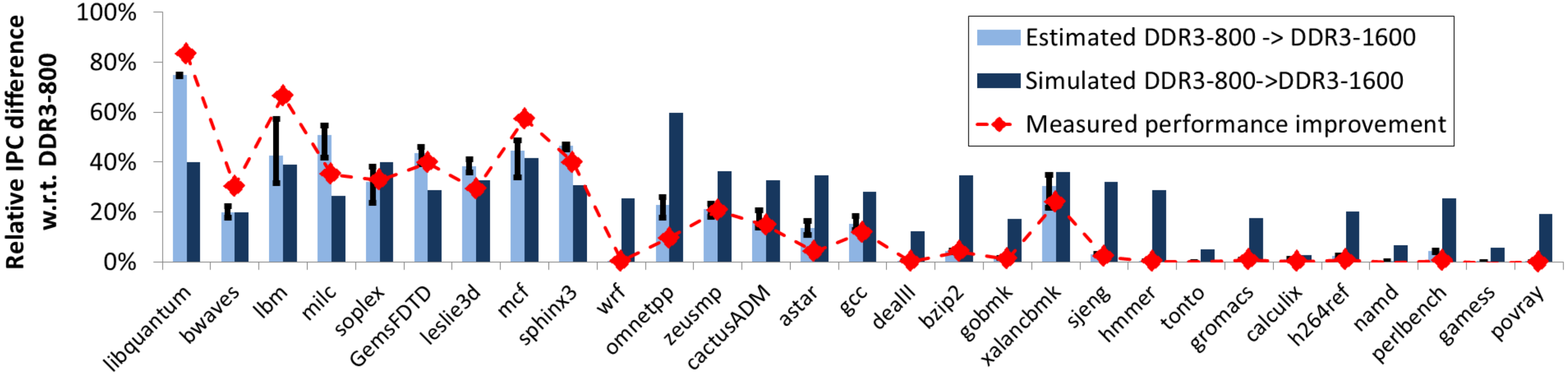
- Performance estimation error
    - High-bandwidth benchmarks: 7%
    - All benchmarks: 3.8%

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

# PROFET *vs.* Simulators

- Simulators:
  - CPU simulator: Zsim - updated and validated for Intel Sandy Bridge CPU
  - Main memory simulator: DRAMSim2
  - Workloads: SPEC CPU2006, 150 billions of instructions

- DDR3-800 → DDR3-1600

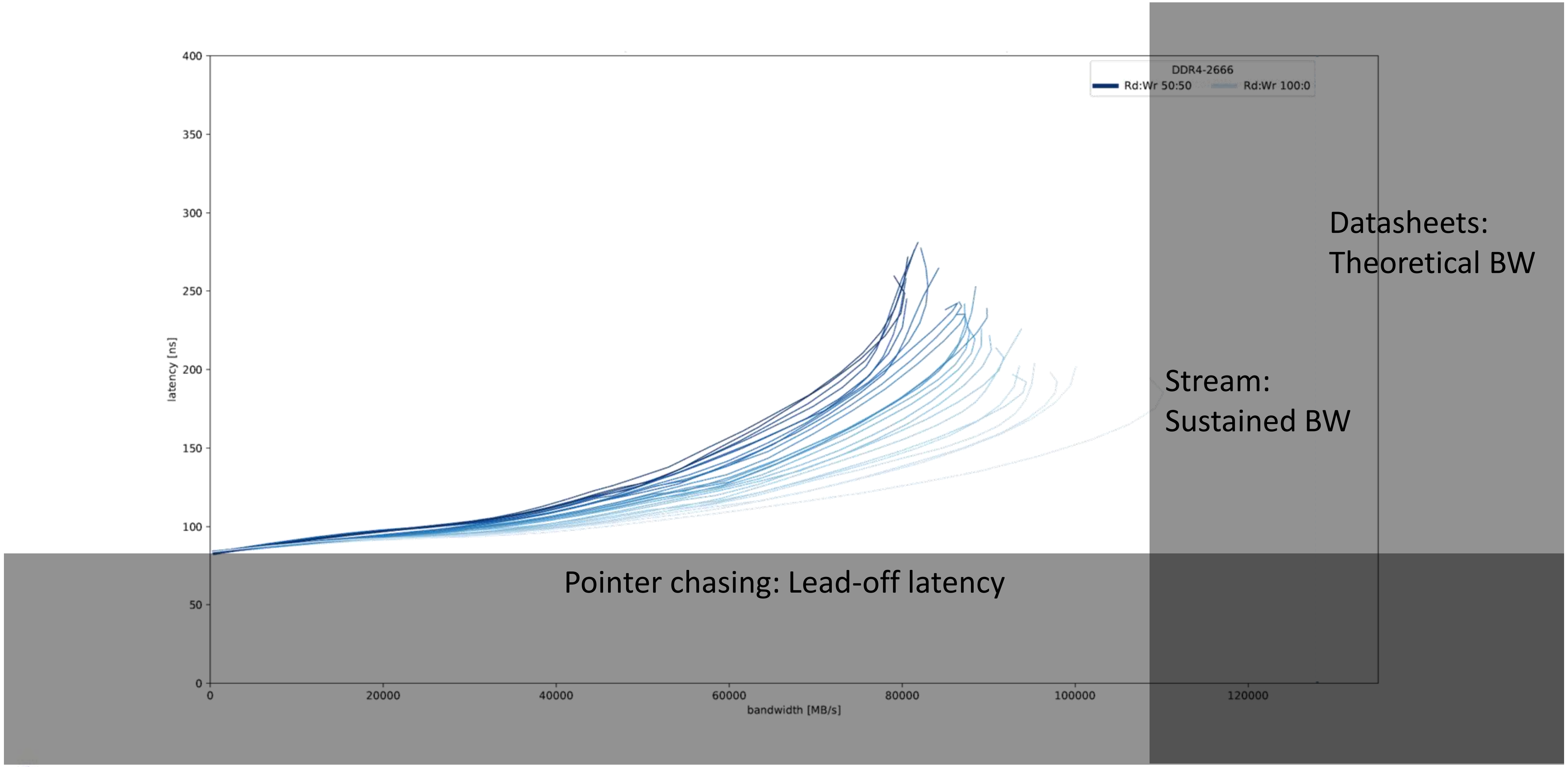


- Average error:
  - PROFET: 3.6%
  - Simulations: 15.7%

- Model estimations follow the trend better than simulations
- PROFET faster than the simulator: **Three orders of magnitude**

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

# Memory performance: Looking inside the box

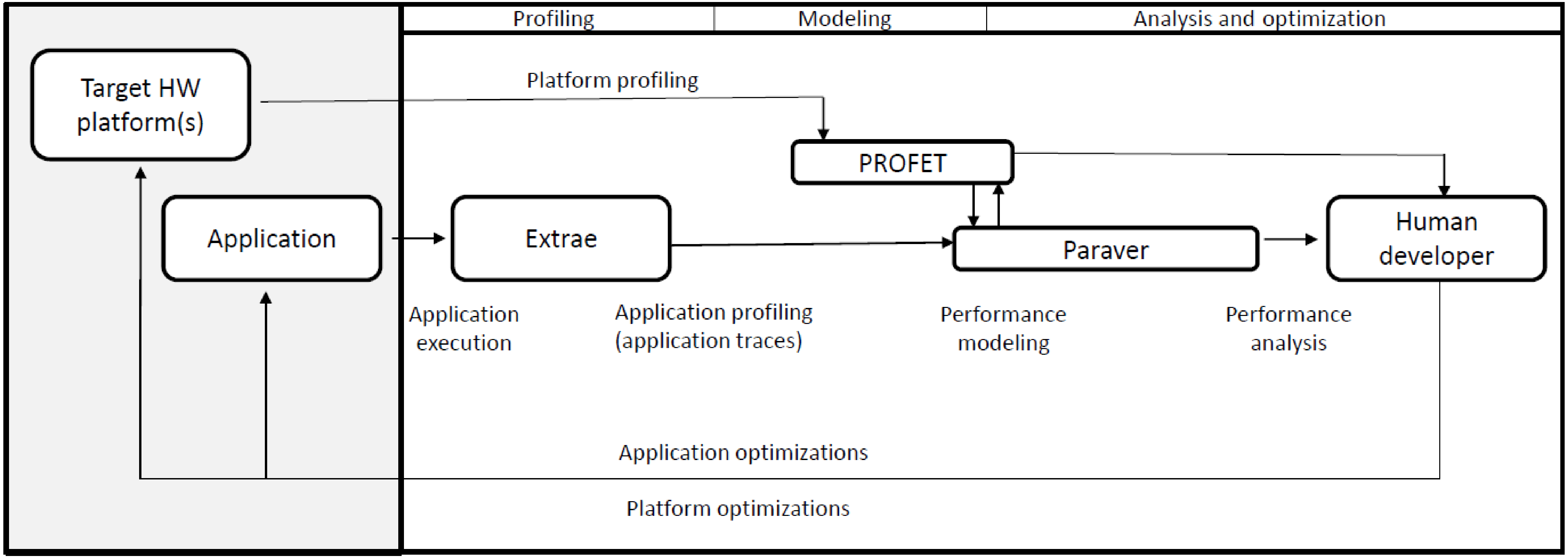


Datasheets: Theoretical BW

Stream: Sustained BW

Pointer chasing: Lead-off latency

# Extrae-PROFET-Paraver optimization cycle



Barcelona Supercomputing Center
Centro Nacional de Supercomputación

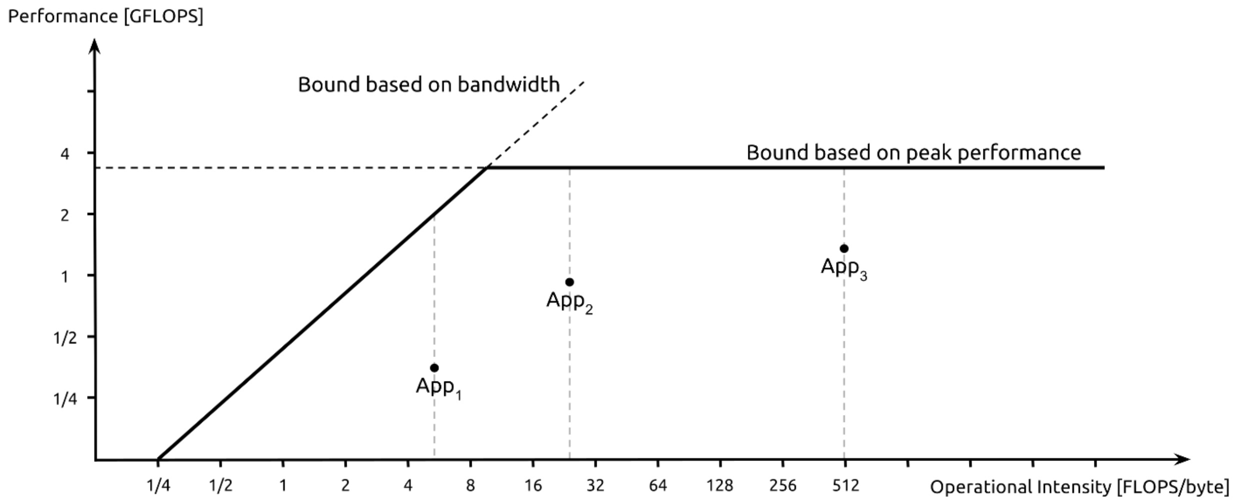
BSC Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA
SEVERO
OCHOA

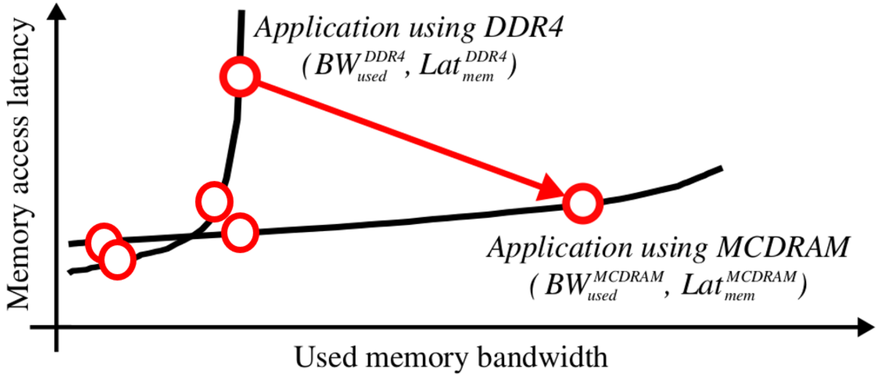
# Thank you

petar.radojkovic@bsc.es

# The roofline model



- DDR4 and MCDRAM at KNL platform
  - *“MCDRAM provides an N-fold higher performance”*


Memory access latency
Application using DDR4
$(BW_{used}^{DDR4}, Lat_{mem}^{DDR4})$
Application using MCDRAM
$(BW_{used}^{MCDRAM}, Lat_{mem}^{MCDRAM})$
Used memory bandwidth