Rooflines …

Aleksandar Ilic and Alexandre Rodrigues

Leonel Sousa, José Morgado, Diogo Matos,
Diogo Marques, Frederico Pratas

inesc id lisboa 20 YEARS
DEFINING TECHNOLOGY

dare Co-funded by the European Union EuroHPC Joint Undertaking FCT Fundação para a Ciência e a Tecnologia

The DARE SGA1 project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101202459. The JU receives support from the European Union's Horizon Europe research and innovation programme and Spain, Germany, Czechia, Italy, Netherlands, Belgium, Finland, Greece, Croatia, Portugal, Poland, Sweden, France and Austria. Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or of the granting authority. Neither the European Union nor the granting authority can be held responsible for

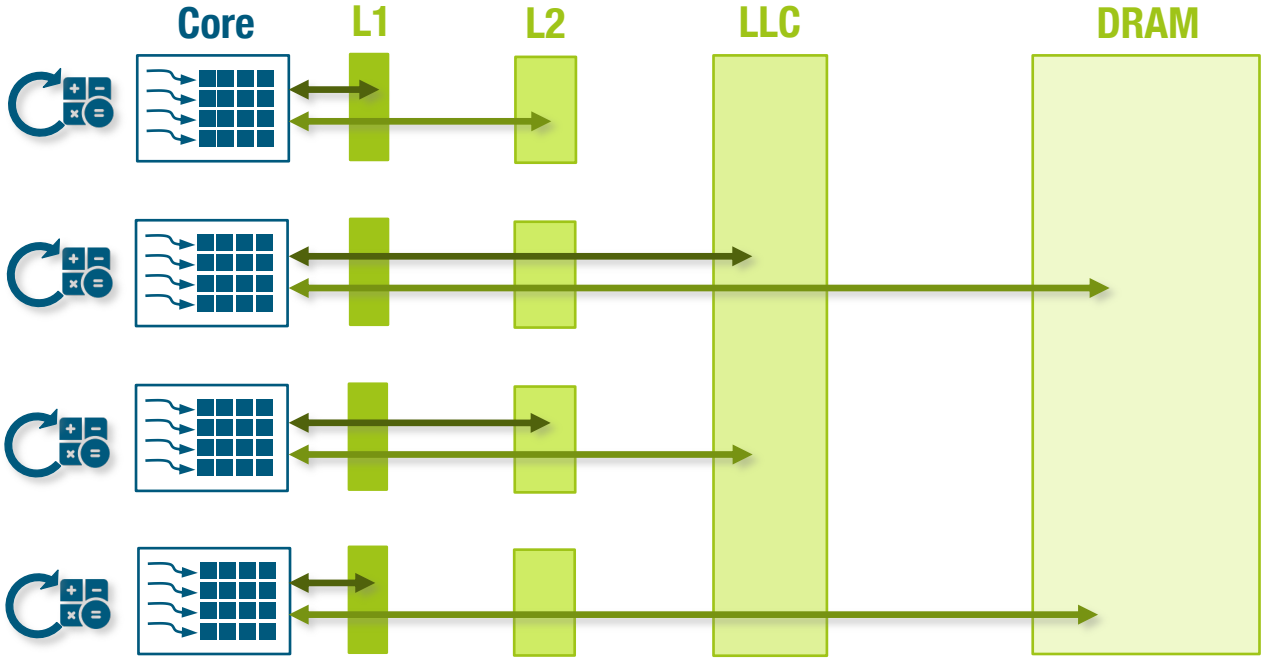
Cache-aware Roofline Model

Cache-aware Roofline Model: Outline

Cache-aware Roofline Model

A. Ilic, F. Pratas and L. Sousa, “Cache-aware Roofline Model: Upgrading the Loft”, IEEE Computer Architecture Letters (2014)
J. Morgado, L. Sousa and A. Ilic, “CARM Tool: Cache-Aware Roofline Model Automatic Benchmarking and Application Analysis”, IISWC (2024)

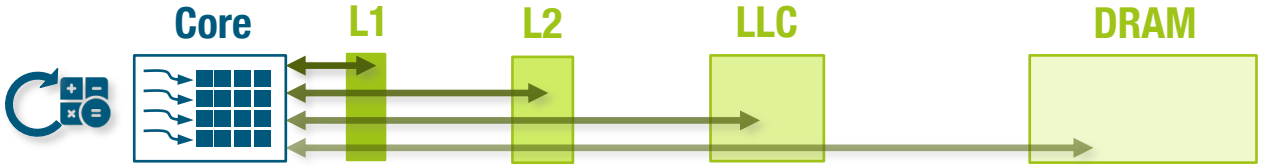
Roofline in a nutshell



Communication overlapped with computation

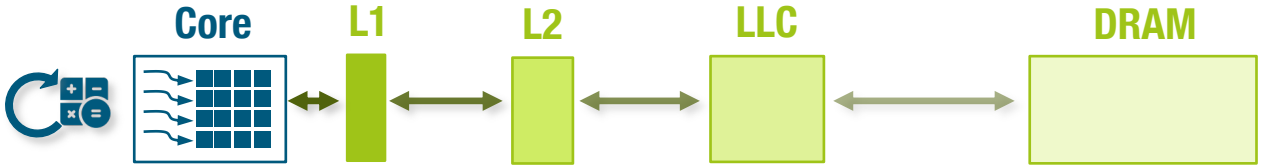
Max performance capped by peak compute throughput or available bandwidth (processor's view)

What is bandwidth?



Cache-aware Roofline Model (CARM)[1]: Bandwidth as seen by the core

- Obtained via micro-benchmarking



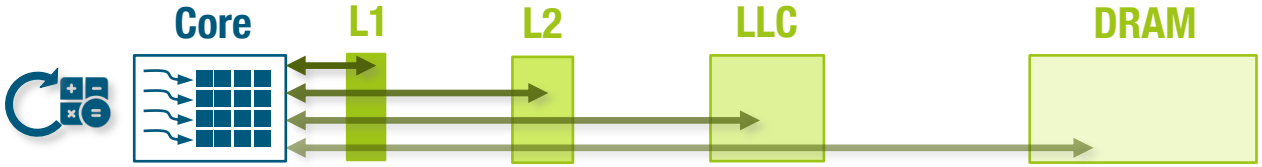
Original Roofline Model (ORM)[2]: Bandwidth between memory levels

- Can be obtained from data-sheets

[1] A. Ilic, F. Pratas and L. Sousa, "Cache-aware Roofline Model: Upgrading the Loft", IEEE Computer Architecture Letters (2014)

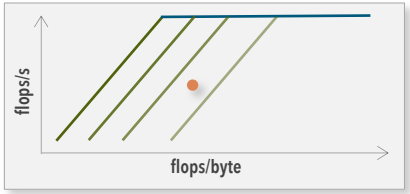
[2] S. Williams, A. Waterman, D. Patterson, "Roofline: An Insightful Visual Performance Model for Multicore Architectures", Commun. ACM (2009)

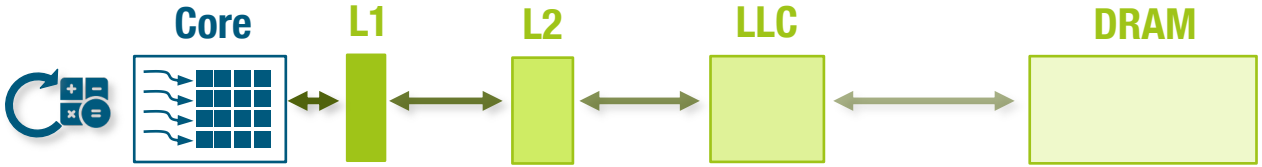
Implications …



Cache-aware Roofline Model[1]

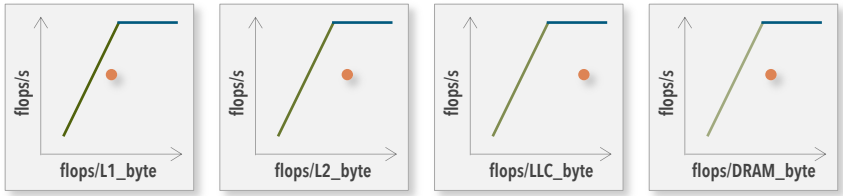
- One model, one arithmetic intensity
- One application "point"





Original Roofline Model[2]

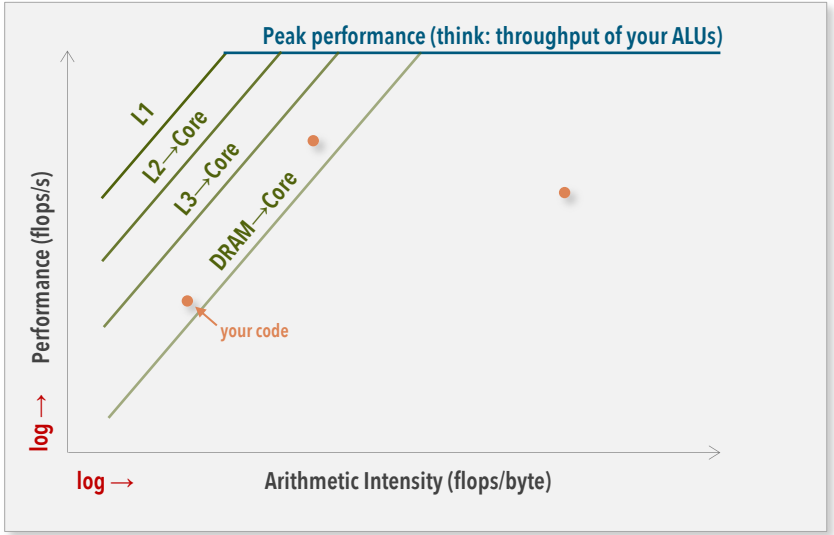
- Several models, several intensities
- Several application "points"



[1] A. Ilic, F. Pratas and L. Sousa, "Cache-aware Roofline Model: Upgrading the Loft", IEEE Computer Architecture Letters (2014)

[2] S. Williams, A. Waterman, D. Patterson, "Roofline: An Insightful Visual Performance Model for Multicore Architectures", Commun. ACM (2009)

Implications … bring cool features



Cache-aware Roofline Model

- Shows absolute architecture maximums

(You can't break them! Can your application exploit them?)

How to "plot" my code?

- CARM arithmetic intensity is exactly what you expect it to be!

code

```
...
  float a = A[i];
  float b = B[i];
  float c = a*b;
  C[i] = c;
...
```

1 flop

12 bytes

AI = 1/12

assembly

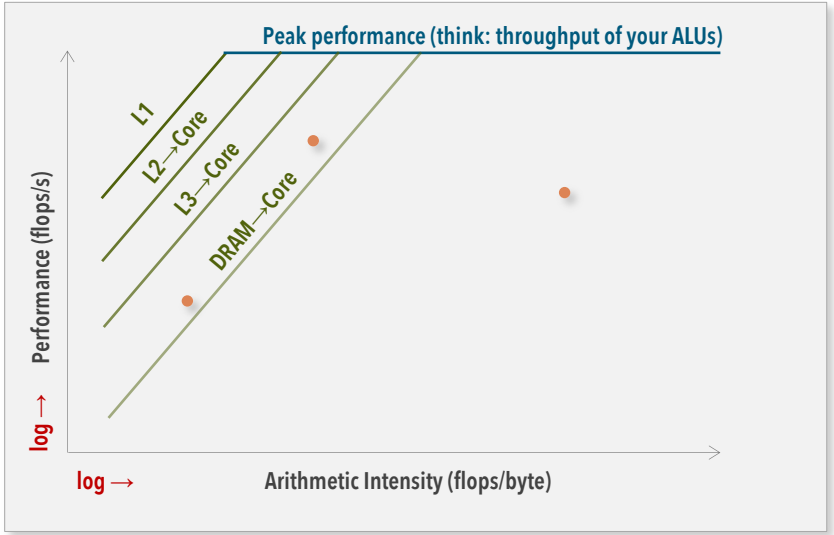
```
...
ld  r1, mem[add1]
ld  r2, mem[add2]
mul r3, r1, r2
st  mem[add3], r3
...
```

counters

```
...
MEM_RETIRED.LOADS
MEM_RETIRED.STORES
INST_RETIRED.FLOPS
CPU_CLK_CYCLES.ALL
...
```

 A. Ilic, F. Pratas and L. Sousa, "Cache-aware Roofline Model: Upgrading the Loft", IEEE Computer Architecture Letters (2014)

Implications … bring cool features



Cache-aware Roofline Model

- Shows absolute architecture maximums

(You can't break them! Can your application exploit them?)

How to "plot" my code?

- CARM arithmetic intensity is exactly what you expect it to be!

Intel Advisor Roofline feature

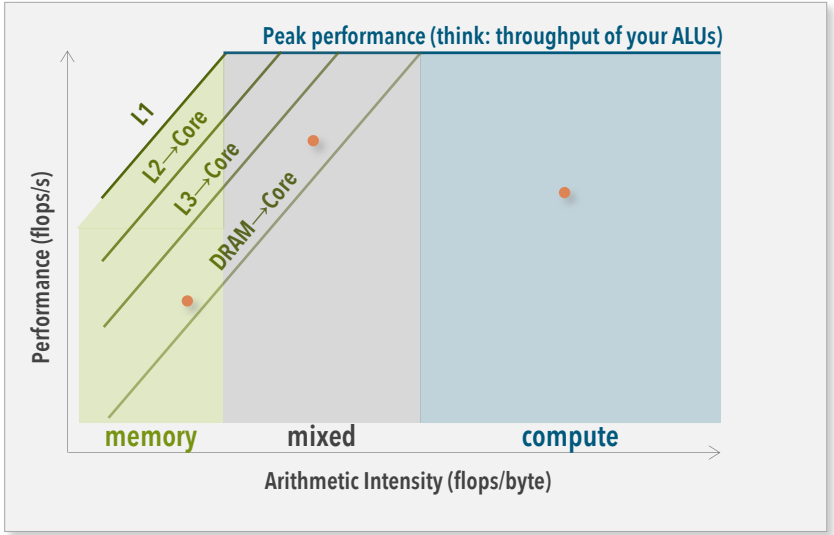
- CARM is there since 2017

CARM Tool (in-house)

- Open-source at Champ Hub @ Github
- Support all major CPUs and GPUs
- PMU- and DBI-based application profiling

A. Ilic, F. Pratas and L. Sousa, "Cache-aware Roofline Model: Upgrading the Loft", IEEE Computer Architecture Letters (2014)
J. Morgado, L. Sousa and A. Ilic, "CARM Tool: Cache-Aware Roofline Model Automatic Benchmarking and Application Analysis", IISWC (2024)

Implications … bring cool features



memory bound
(improve access pattern, use of caches)

mixed
(all kinds of everything)

compute bound
(vectorize, parallelize…)

Cache-aware Roofline Model

- Shows absolute architecture maximums

(You can't break them! Can your application exploit them?)

How to "plot" my code?

- CARM arithmetic intensity is exactly what you expect it to be!

How to use CARM?

① Detect the boundness region

- What are my expected maximums?
- Provides first optimization hints

② Draw an imaginary vertical line

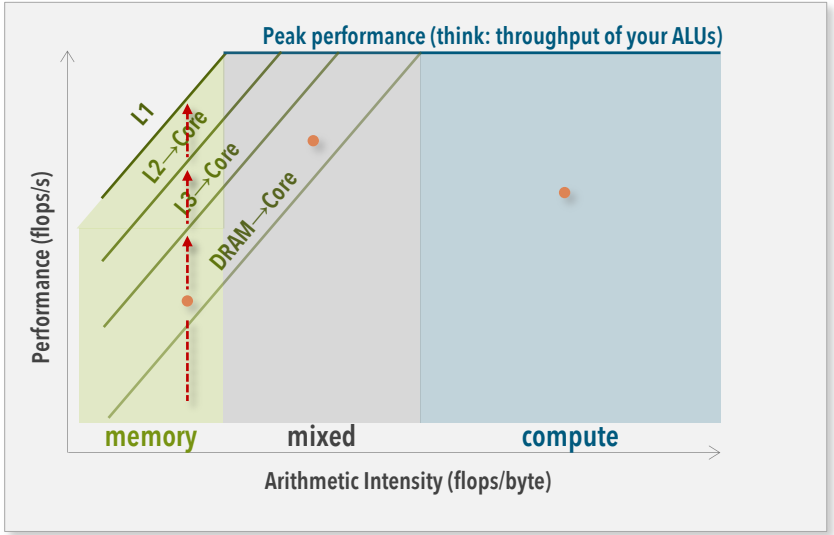
- What are my main bottlenecks? (observe intersected lines)
- Focus your optimization (aim at surpassing the line above)

③ Optimize your code: Break above roofs!

- You should move up (as your performance improves)
- Unless you restructure the code, or your compiler decides so…

A. Ilic, F. Pratas and L. Sousa, "Cache-aware Roofline Model: Upgrading the Loft", IEEE Computer Architecture Letters (2014)
D. Marques, et.al., "Performance analysis with Cache-aware Roofline Model in Intel Advisor", HPCS (2017)

Implications … bring cool features



memory bound
(improve access pattern, use of caches)

mixed
(all kinds of everything)

compute bound
(vectorize, parallelize…)

Cache-aware Roofline Model

- Shows absolute architecture maximums

(You can't break them! Can your application exploit them?)

How to "plot" my code?

- CARM arithmetic intensity is exactly what you expect it to be!

How to use CARM?

① Detect the boundness region

- What are my expected maximums?
- Provides first optimization hints

② Draw an imaginary vertical line

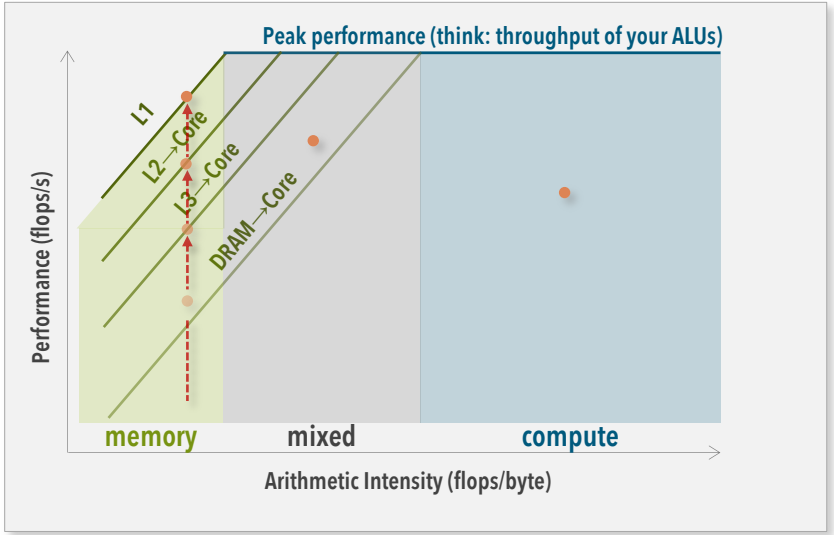
- What are my main bottlenecks? (observe intersected lines)
- Focus your optimization (aim at surpassing the line above)

③ Optimize your code: Break above roofs!

- You should move up (as your performance improves)
- Unless you restructure the code, or your compiler decides so…

A. Ilic, F. Pratas and L. Sousa, "Cache-aware Roofline Model: Upgrading the Loft", IEEE Computer Architecture Letters (2014)
D. Marques, et.al., "Performance analysis with Cache-aware Roofline Model in Intel Advisor", HPCS (2017)

Implications … bring cool features



memory bound
(improve access pattern, use of caches)

mixed
(all kinds of everything)

compute bound
(vectorize, parallelize…)

Cache-aware Roofline Model

- Shows absolute architecture maximums

(You can't break them! Can your application exploit them?)

How to "plot" my code?

- CARM arithmetic intensity is exactly what you expect it to be!

How to use CARM?

① Detect the boundness region

- What are my expected maximums?
- Provides first optimization hints

② Draw an imaginary vertical line

- What are my main bottlenecks? (observe intersected lines)
- Focus your optimization (aim at surpassing the line above)

③ Optimize your code: Break above roofs!

- You should move up (as your performance improves)
- Unless you restructure the code, or your compiler decides so…

A. Ilic, F. Pratas and L. Sousa, "Cache-aware Roofline Model: Upgrading the Loft", IEEE Computer Architecture Letters (2014)
D. Marques, et.al., "Performance analysis with Cache-aware Roofline Model in Intel Advisor", HPCS (2017)

Not all rooflines are the same

A. Ilic, F. Pratas and L. Sousa, “Cache-aware Roofline Model: Upgrading the Loft”, IEEE Computer Architecture Letters (2014)
S. Williams, A. Waterman, D. Patterson, "Roofline: An Insightful Visual Performance Model for Multicore Architectures”, Commun. ACM (2009)

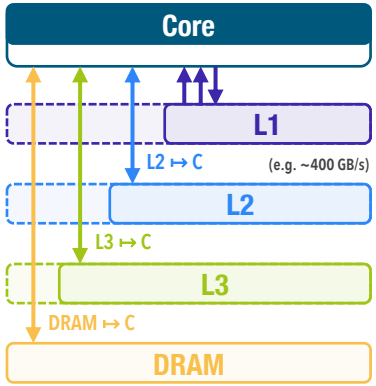
CARM vs. ORM: Construction and Bandwidth

CARM: Cache-aware Roofline

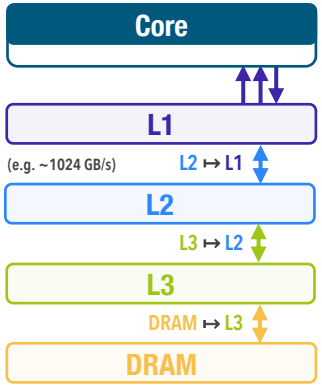
ROOFS
Micro-bencharking (realistic)

BANDWIDTH
As seen by the core

ROOFLINE
Single-plot, single app point



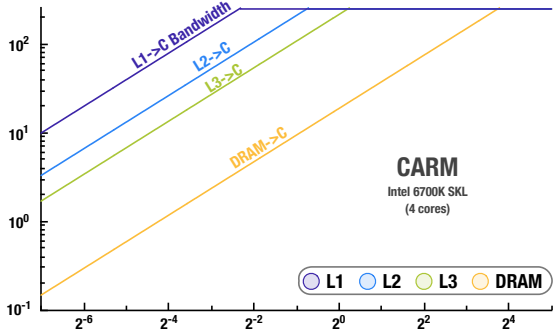
ORM: Original Roofline

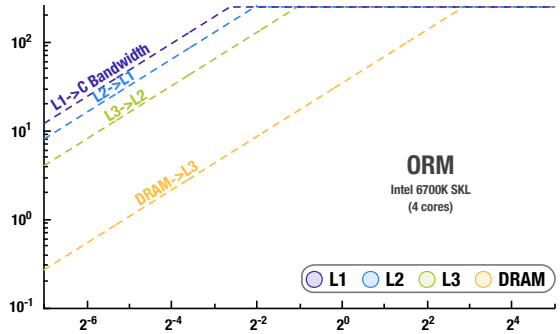


ROOFS
Data-sheets (theoretical)

BANDWIDTH
Between two memory levels

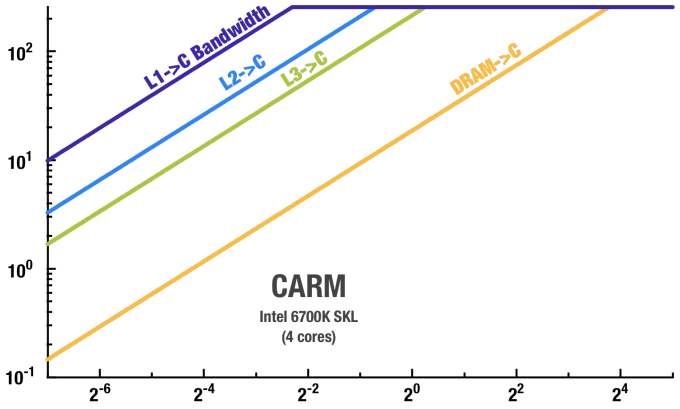
ROOFLINE
Multi-plot, multiple app points
(one for each memory level)

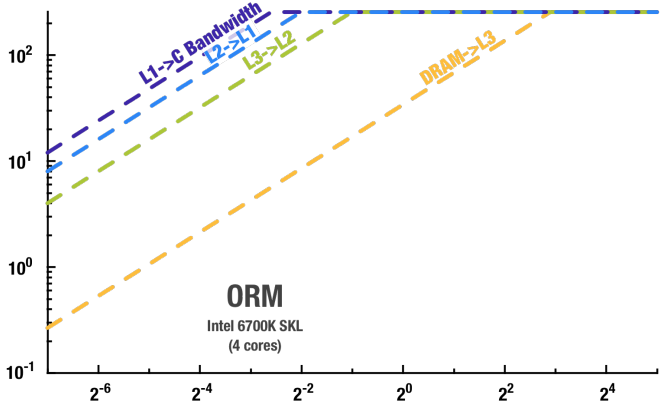




Example: L3 Benchmark

Memory-bound Application: Benchmark to obtain the sustainable bandwidth of LLC





iteration: 1

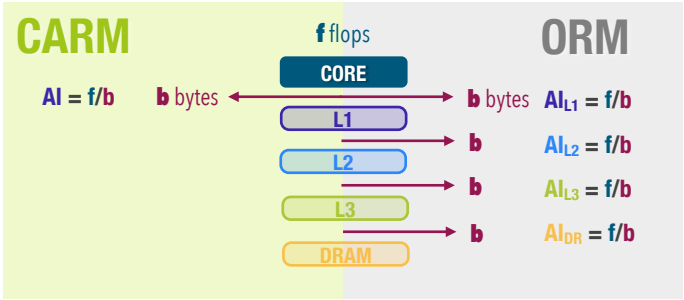
"cold caches" (misses in all caches)

L3 Benchmark

```
for it=1..N do
   mops  (2LD+ST)
   mops  (2LD+ST)
   mops  (2LD+ST)
   flops (FMAD)
end for
```

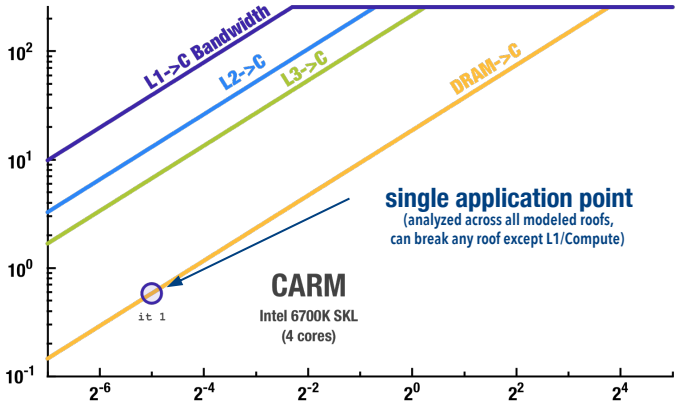
→ **b** bytes

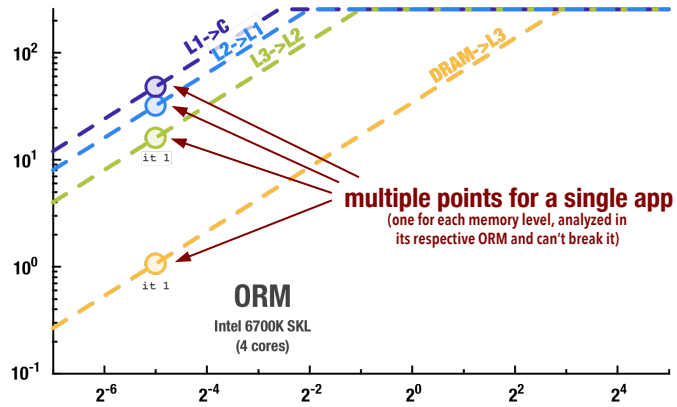
→ **f** flops



Example: L3 Benchmark

Memory-bound Application: Benchmark to obtain the sustainable bandwidth of LLC





iteration: 1

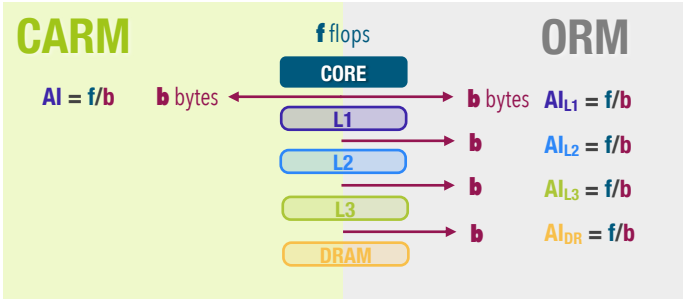
"cold caches" (misses in all caches)

L3 Benchmark

```
for it=1..N do
   mops  (2LD+ST)
   mops  (2LD+ST)
   mops  (2LD+ST)
   flops (FMAD)
end for
```

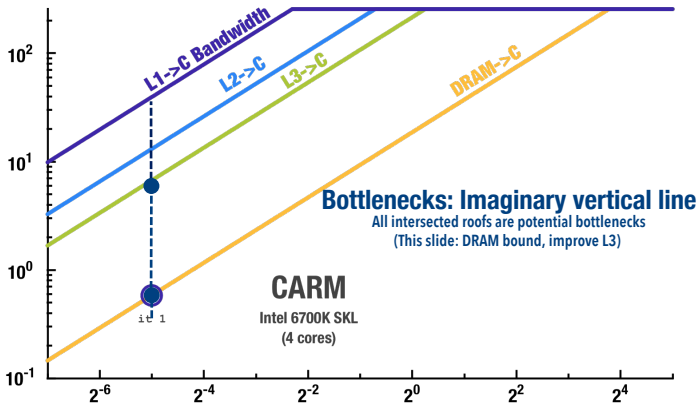
b bytes

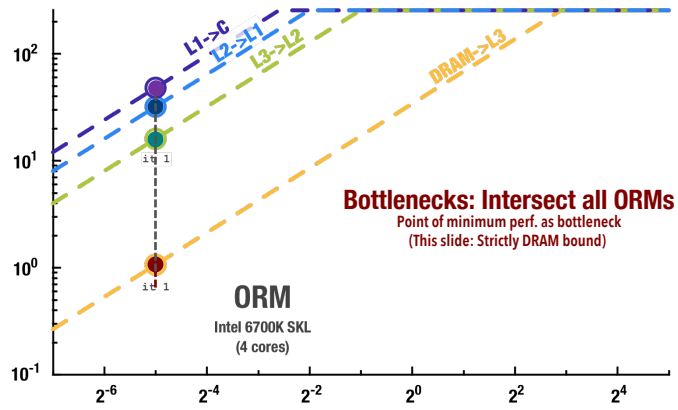
f flops



Example: L3 Benchmark

Memory-bound Application: Benchmark to obtain the sustainable bandwidth of LLC





iteration: 1

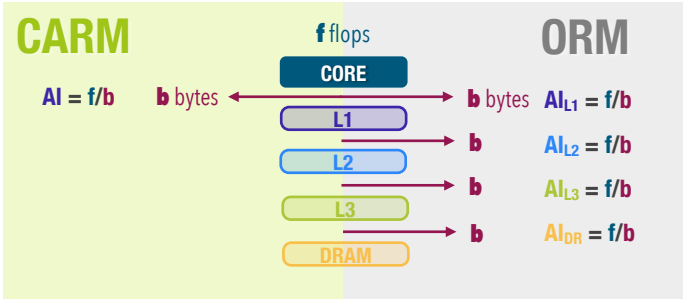
"cold caches" (misses in all caches)

L3 Benchmark

```
for it=1..N do
   mops  (2LD+ST)
   mops  (2LD+ST)
   mops  (2LD+ST)
   flops (FMAD)
end for
```

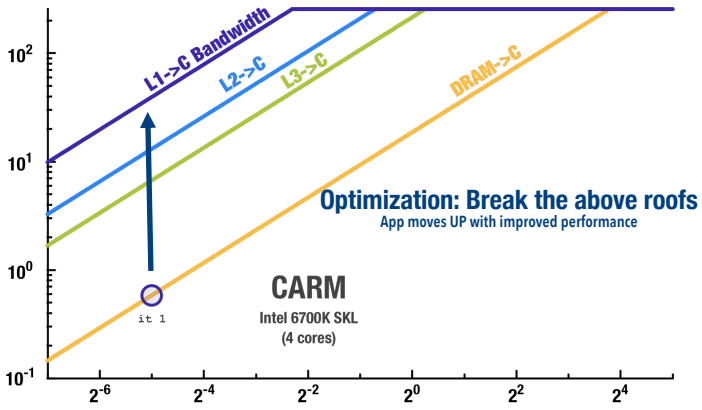
→ **b** bytes

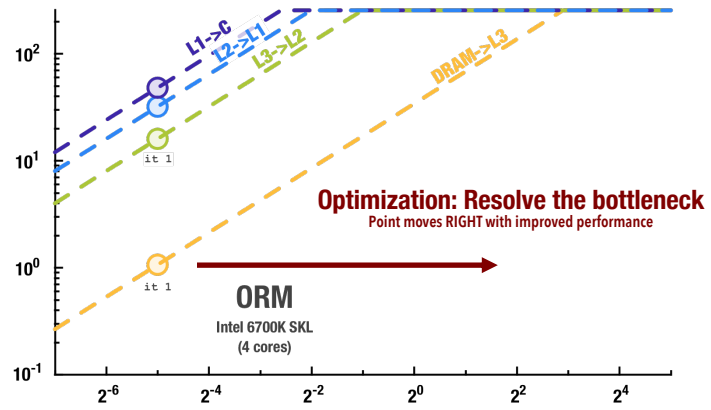
→ **f** flops



Example: L3 Benchmark

Memory-bound Application: Benchmark to obtain the sustainable bandwidth of LLC





iteration: **1**

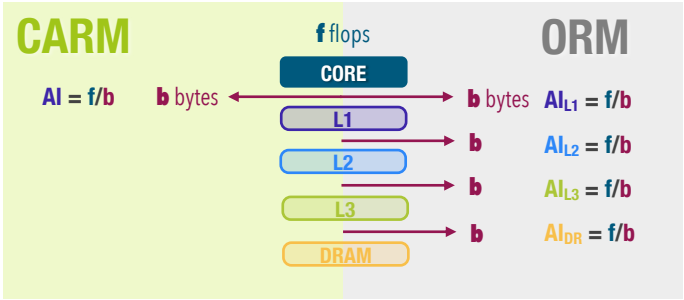
"cold caches" (misses in all caches)

L3 Benchmark

```
for it=1..N do
   mops  (2LD+ST)
   mops  (2LD+ST)
   mops  (2LD+ST)
   flops (FMAD)
end for
```

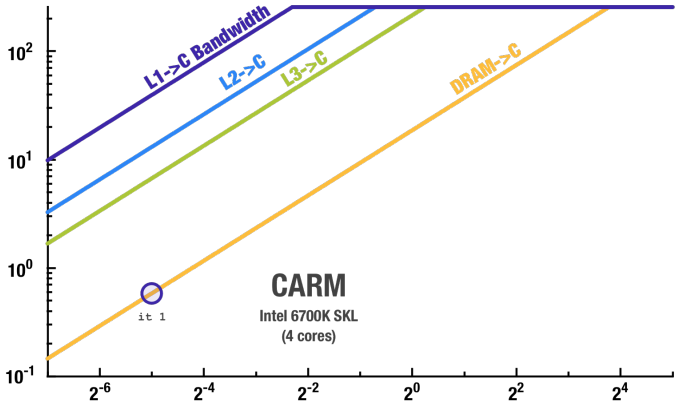
→ **b** bytes

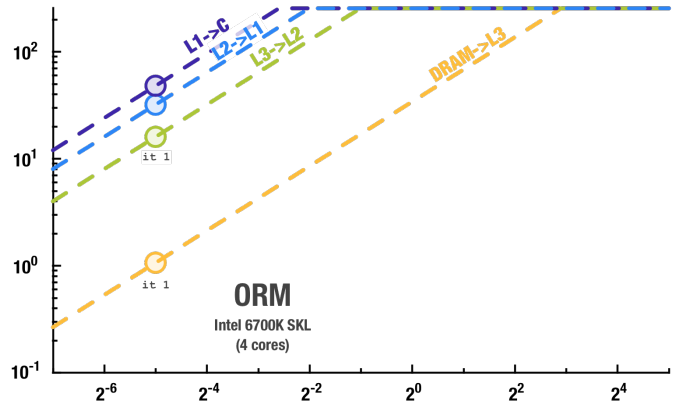
→ **f** flops



Example: L3 Benchmark

Memory-bound Application: Benchmark to obtain the sustainable bandwidth of LLC





iterations: **1 and 2**

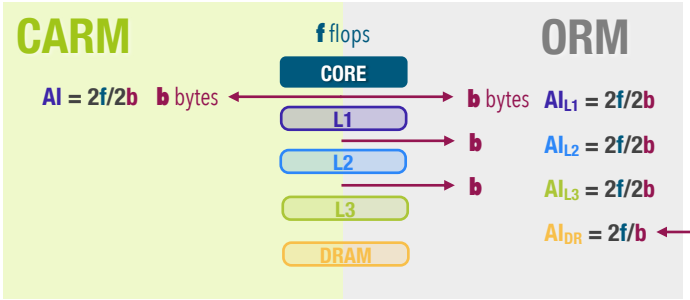
All data reused from L3

L3 Benchmark

```
for it=1..N do
   mops  (2LD+ST)
   mops  (2LD+ST)
   mops  (2LD+ST)
   flops (FMAD)
end for
```

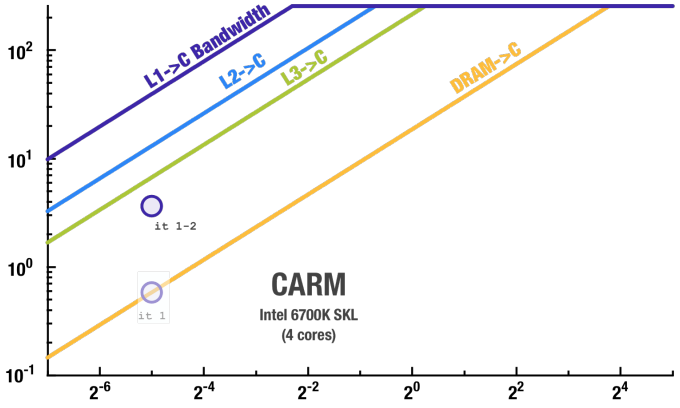
→ **b** bytes

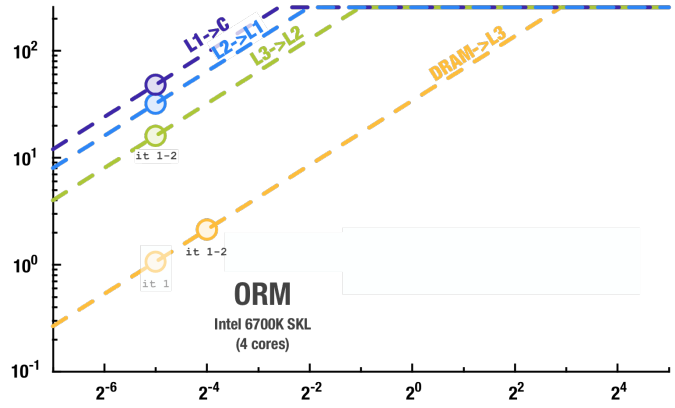
→ **f** flops



Example: L3 Benchmark

Memory-bound Application: Benchmark to obtain the sustainable bandwidth of LLC





iterations: **1 and 2**

All data reused from L3

L3 Benchmark

```
for it=1..N do
   mops  (2LD+ST)
   mops  (2LD+ST)
   mops  (2LD+ST)
   flops (FMAD)
end for
```

b bytes

f flops

CARM

f flops

AI = 2f/2b **b** bytes

CORE

b bytes

L1

b

L2

b

L3

DRAM

ORM

AI_{L1} = 2f/2b

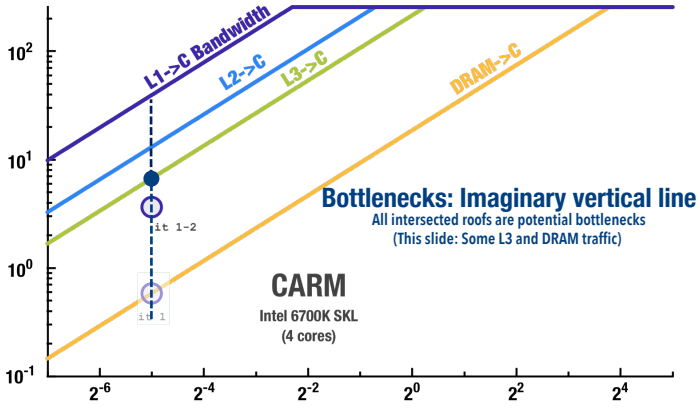
AI_{L2} = 2f/2b

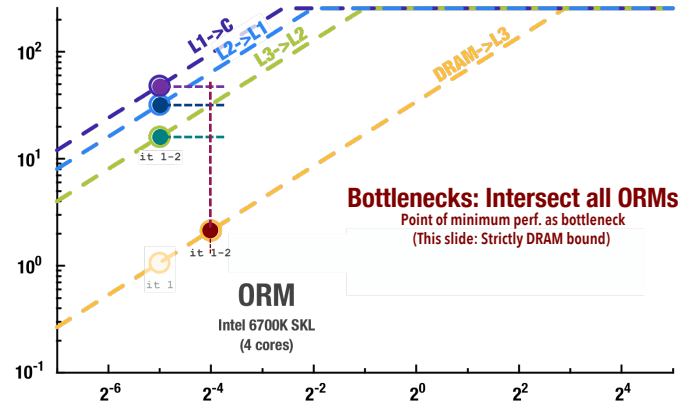
AI_{L3} = 2f/2b

AI_{DR} = 2f/b ← **DRAM** accessed only in the 1st iteration!

Example: L3 Benchmark

Memory-bound Application: Benchmark to obtain the sustainable bandwidth of LLC





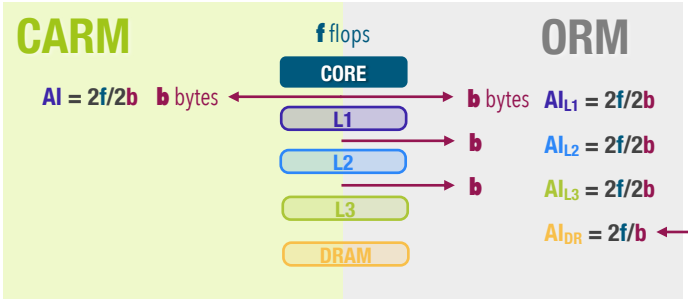
iterations: 1 and 2
All data reused from L3

L3 Benchmark

```
for it=1..N do
   mops  (2LD+ST)
   mops  (2LD+ST)
   mops  (2LD+ST)
   flops (FMAD)
end for
```

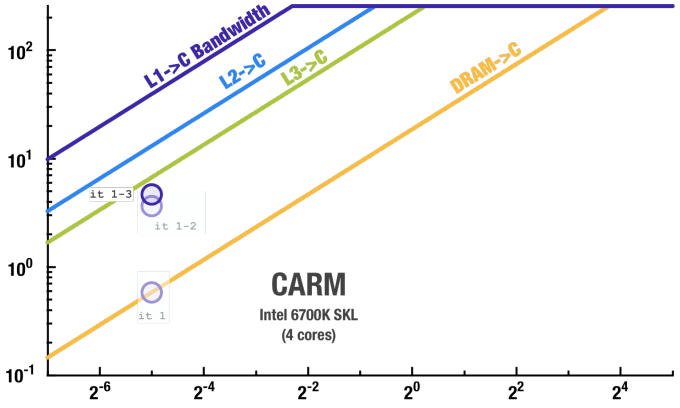
→ **b** bytes

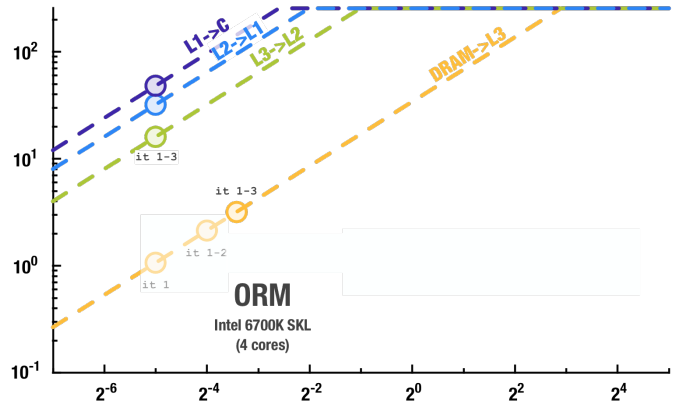
→ **f** flops



Example: L3 Benchmark

Memory-bound Application: Benchmark to obtain the sustainable bandwidth of LLC





iterations: **1 to 3**

All data reused from L3

L3 Benchmark

```
for it=1..N do
   mops  (2LD+ST)
   mops  (2LD+ST)
   mops  (2LD+ST)
   flops (FMAD)
end for
```

→ **b** bytes

→ **f** flops

CARM

f flops

AI = 3f/3b **b** bytes → CORE → **b** bytes

L1 → **b**

L2 → **b**

L3

DRAM

ORM

AI_{L1} = 3f/3b

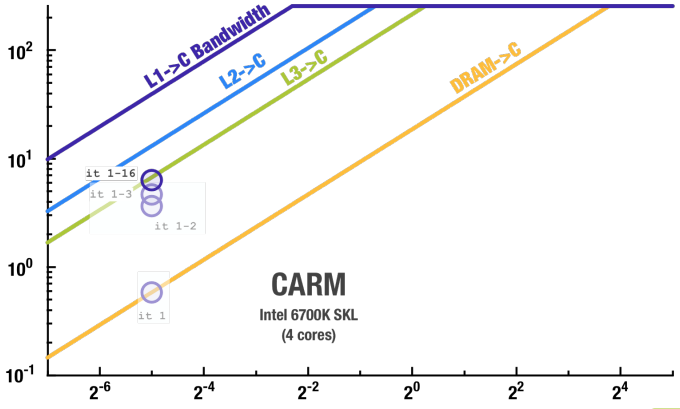
AI_{L2} = 3f/3b

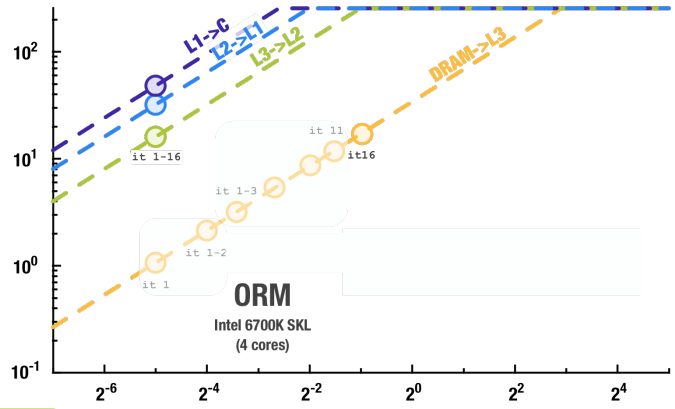
AI_{L3} = 3f/3b

AI_{DR} = 3f/b ← **DRAM** accessed only in the 1st iteration!

Example: L3 Benchmark

Memory-bound Application: Benchmark to obtain the sustainable bandwidth of LLC





iterations: **1 to 16**

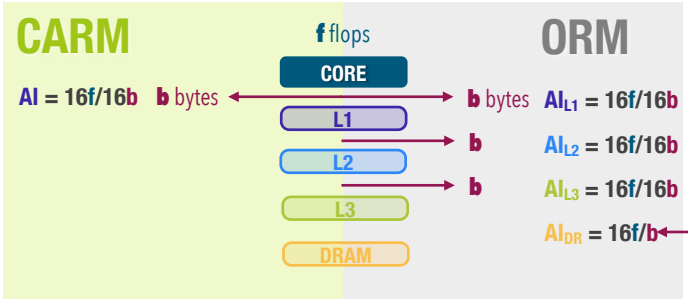
All data reused from L3

L3 Benchmark

```
for it=1..N do
   mops  (2LD+ST)
   mops  (2LD+ST)
   mops  (2LD+ST)
   flops (FMAD)
end for
```

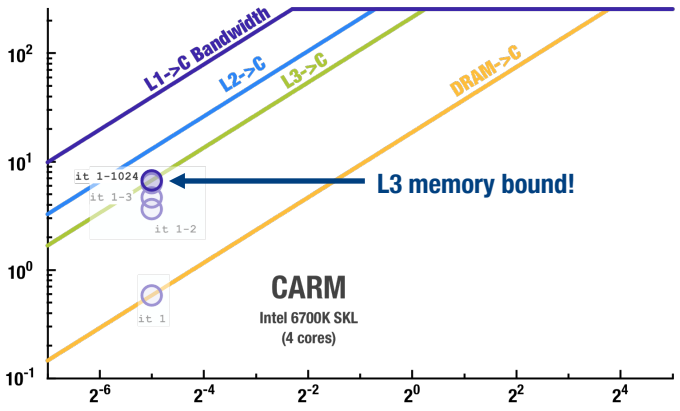
b bytes

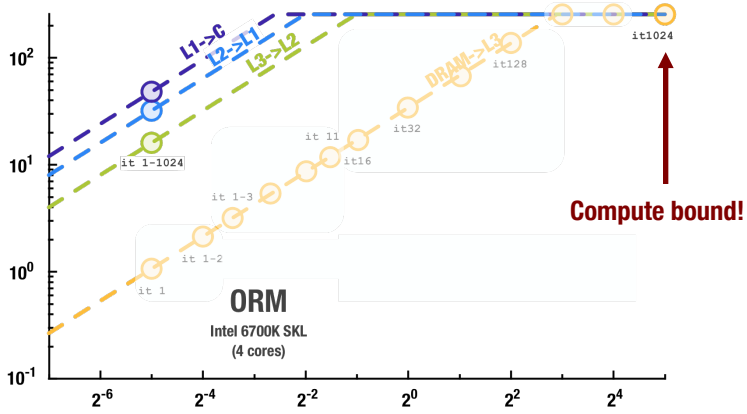
f flops



Example: L3 Benchmark

Memory-bound Application: Benchmark to obtain the sustainable bandwidth of LLC





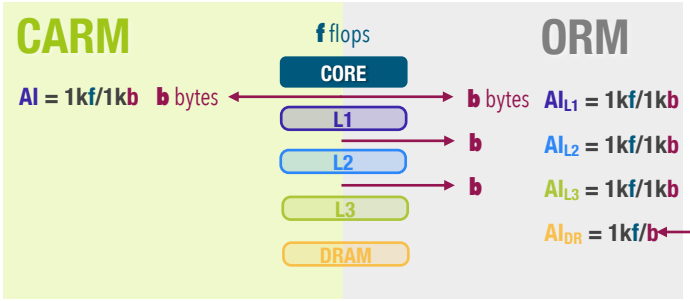
iterations: **1 to 1024**
All data reused from L3

L3 Benchmark

```
for it=1..N do
   mops  (2LD+ST)
   mops  (2LD+ST)
   mops  (2LD+ST)
   flops (FMAD)
end for
```

→ **b** bytes

→ **f** flops



Don’t do it manually!

J. Morgado, L. Sousa and A. Ilic, “CARM Tool: Cache-Aware Roofline Model Automatic Benchmarking and Application Analysis”, IISWC (2024)

State of the art – Roofline Tools

	Intel Advisor	AMD uProf	ERT
Supported Architectures	Intel	AMD	Intel | AMD
Supported Roofline	CARM	ORM	? CARM ?
Application Analysis	DBI	PMUs	No support
Open-Source	No	No	Yes

 J. Morgado, L. Sousa and A. Ilic, "CARM Tool: Cache-Aware Roofline Model Automatic Benchmarking and Application Analysis", IISWC (2024)

State of the art – Roofline Tools

	Intel Advisor	AMD uProf	ERT	CARM Tool
Supported Architectures	Intel	AMD	Intel | AMD	Intel | AMD ARM | RISC-V
Supported Roofline	CARM	ORM	? CARM ?	CARM
Application Analysis	DBI	PMUs	No support	DBI | PMUs
Open-Source	No	No	Yes	Yes

SUPPORTED ISA (SIMD extensions)

intel AMD — SCALAR | SSE | AVX2 | AVX512

arm — SCALAR | NEON | SVE

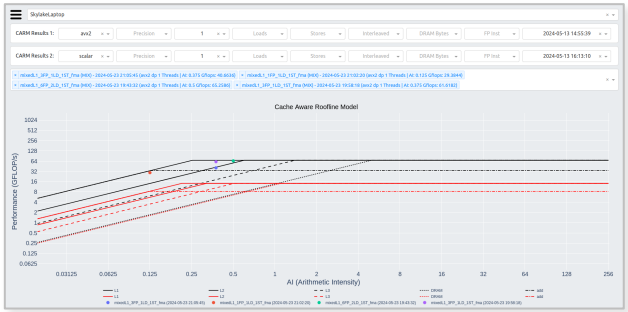
RISC-V — SCALAR | RVV 0.7.1 | RVV 1.0

 J. Morgado, L. Sousa and A. Ilic, "CARM Tool: Cache-Aware Roofline Model Automatic Benchmarking and Application Analysis", IISWC (2024)

The CARM Tool :: High-level overview

DEMO (SOON)

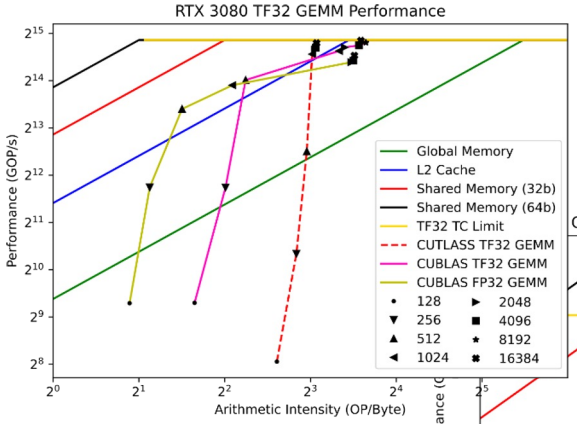
- **Interface: Two types**
  - Command Line Interface (CLI)
  - Graphical User Interface (GUI)
- **Automatic Benchmarking**
  - CARM Benchmarks
  - Memory / FP / Mixed Benchmarks
- **Application Analysis**
  - PMU: PAPI
  - DBI: DynamoRIO, Intel SDE

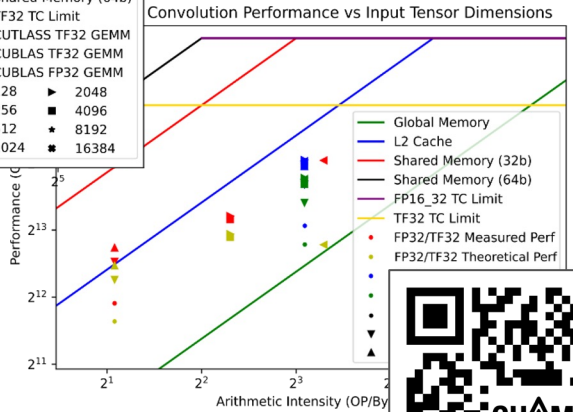


 J. Morgado, L. Sousa and A. Ilic, "CARM Tool: Cache-Aware Roofline Model Automatic Benchmarking and Application Analysis", IISWC (2024)

The CARM Tool :: GPU Extension

- **Nvidia GPU + Tensor CARM**
  - Compute: CUDA + Tensor Cores
  - Memory: Global, L2 cache, Shared
- **AMD GPU Extension**
  - Targeting MI GPUs via HIP
  - AMD Roc Prof for profiling
- **GPU App Profiling: Nsight Compute**
  - Entire app or specific GPU kernels
  - Potential shown in profiling convolutions etc.



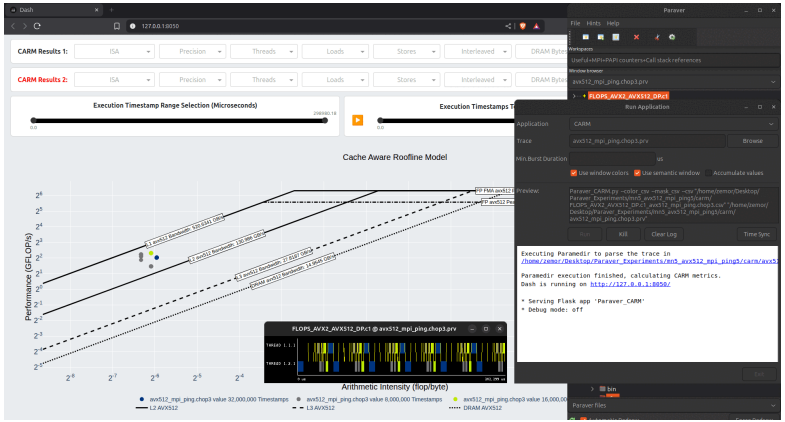


The CARM Tool :: Paraver integration

- **CARM-based application profiling in Paraver**
  - Via analysis of Extrae traces
  - CARM Tool provides the CARM results
  - Integrated with the Paraver Tool as a separate module

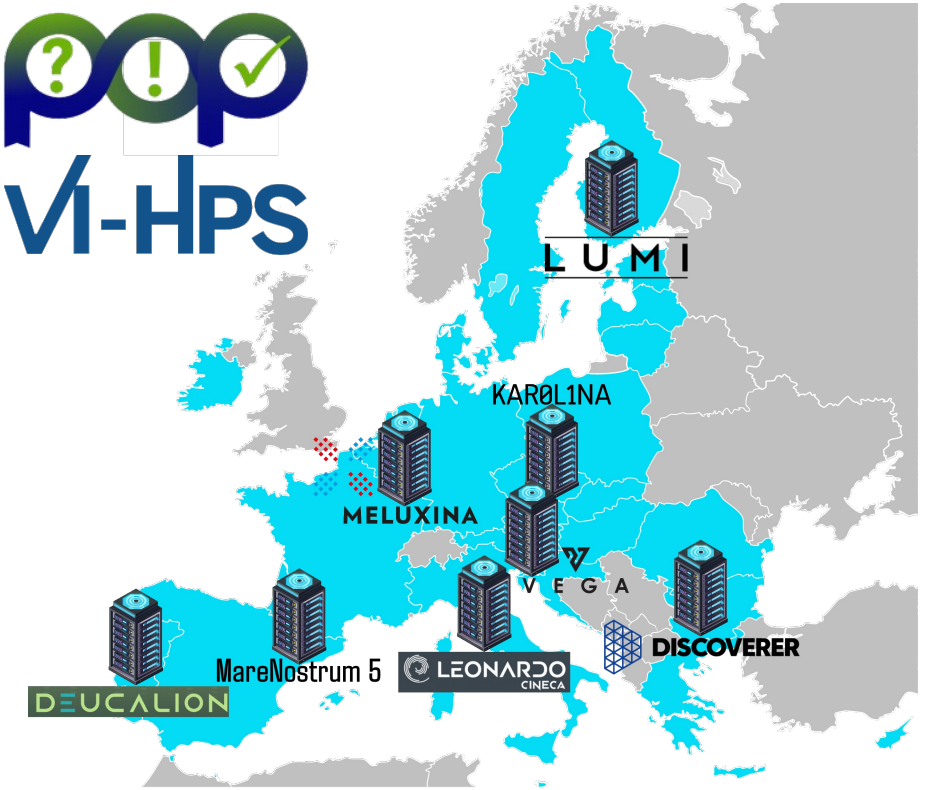
- **CARM-Paraver integration features**
  - Synchronization with Paraver timelines
  - Importing Paraver timestamp colouring and masks
  - Exporting CARM-related metrics to Paraver

- **Synchronization with Paraver Timelines**
  - Synchronization point defined via Paraver GUI
  - Re-sync possible via CARM GUI



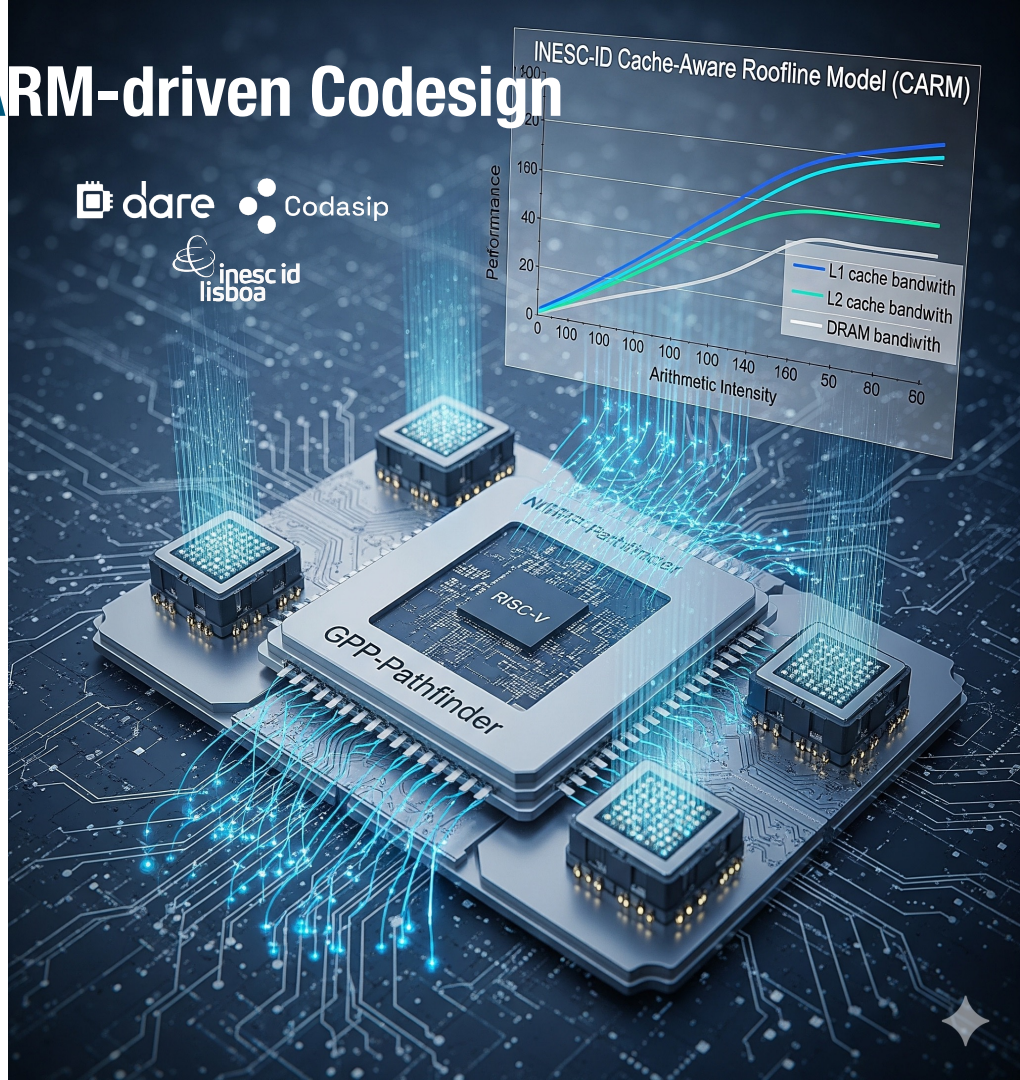
The CARM Tool :: Integration into HPC Centres

- **CARM-based application profiling**
  - Via the CARM Tool
  - Via Extrae traces and Paraver integration
- **POP3: Deployment to HPC Centres**
  - Karolina automated deployment complete
  - Available on EuroHPC centres soon
- **CARM Dissemination Initiatives**
  - POP3 Webinars
  - VI-HPS Tools Guide and Blog
  - https://www.vi-hps.org/tools/carm.html



DARE Project :: CARM-driven Codesign

- **DARE SGA1 Project**
  - Driving Europe's next-generation HPC systems
  - Focus on performance, efficiency, and sovereignty
- **Codasip: RISC-V GPP Pathfinding**
  - Low-level architecture-aware support for CARM
  - Help optimizing the apps running on the GPP core
- **Insights about characteristics of the architecture**
- **Collaborate with the DARE SW TA**



CARM Tool – Available on Github

- **The CARM Tool is open source and available on Github**
  - https://github.com/champ-hub/carm-roofline

- **The first of many tools to be developed in the scope of CHAMP hub**

CHAMP

Heterogeneous Computing and Performance Modeling Hub

Hub para Computação Heterogénea e Modelação de Performance

Finally, something to really see!

J. Morgado, L. Sousa and A. Ilic, “CARM Tool: Cache-Aware Roofline Model Automatic Benchmarking and Application Analysis”, IISWC (2024)

inesc id lisboa 20 YEARS

DEFINING TECHNOLOGY

Thank you!

dare

Co-funded by the European Union

EuroHPC Joint Undertaking

FCT Fundação para a Ciência e a Tecnologia

The DARE SGA1 project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 101202459. The JU receives support from the European Union's Horizon Europe research and innovation programme and Spain, Germany, Czechia, Italy, Netherlands, Belgium, Finland, Greece, Croatia, Portugal, Poland, Sweden, France and Austria. Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or of the granting authority. Neither the European Union nor the granting authority can be held responsible for