

Barcelona Supercomputing Center Centro Nacional de Supercomputación



AccelCom

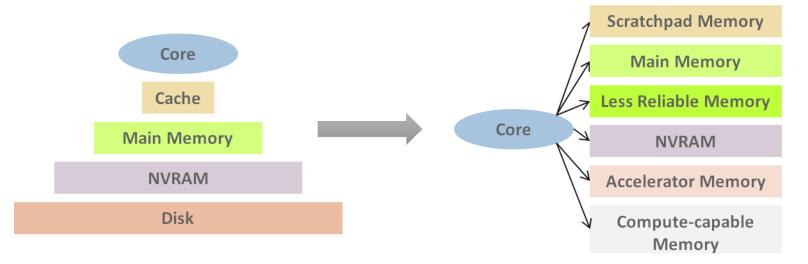
# An Ecosystem of Tools for Broad Heterogeneous Memory Usage

27 June, 2022

Marc Jordà

# **Overview & Motivation**

- To efficiently exploit heterogeneous memory:
  - Bring them as first-class citizens
  - Move from hierarchical to explicitly managed

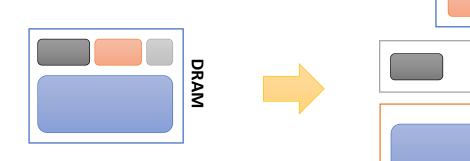


- Application's data distribution?
  - OS? Heuristics? On-the-fly monitoring? Hardware-assisted? Historic data? User hints?
  - Need ecosystem to assist users/developers: tools
    - Profilers, libraries, runtime systems



### **Overview & Motivation**

- Heterogeneous Memory Systems
  - KNL: DRAM + MCDRAM (↑ BW, ↑ Lat.) → R.I.P.
  - Byte-addressable NVRAM DIMMs (↓ BW, ↑ Lat., persistent)
    - Intel<sup>®</sup> Optane<sup>™</sup> Persistent Memory (PMem)
- Goal: Assess optimal data distribution
  - Maximize performance
  - Minimize energy





...

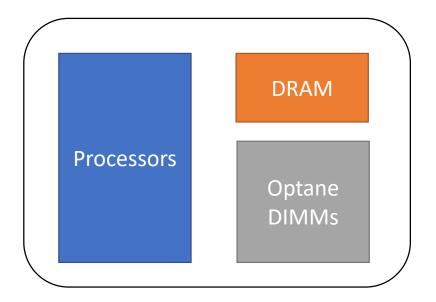
DRAM

Optane DIMMs

SSDs

# Systems w/ Optane DIMMs

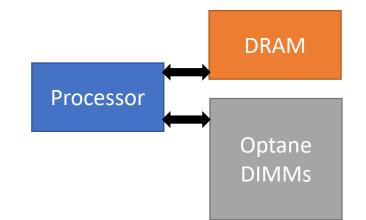
- Two levels of memory
  - Main memory (DRAM)
    - Processor has direct access to all of main memory
    - Regular DRAM latency/bandwidth
  - Intel<sup>®</sup> Optane<sup>™</sup> PMem
    - Very high capacity + persistency
    - Higher latency, but much better than SSDs

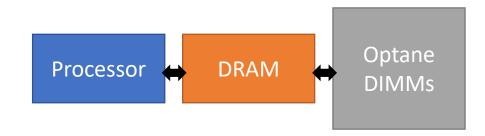




# **Optane PMem Modes**

- App Direct Mode (Heterogeneous Memory)
  - DRAM and Optane DIMMs are both available
  - More overall memory available
  - Software managed (applications need to handle themselves)





- Memory Mode (Deep Memory)
  - DRAM as cache for Optane DIMMs
  - Only Optane DIMMs address space
  - Done in hardware (applications and OS don't need to be modified)



# Framework



Barcelona Supercomputing Center Centro Nacional de Supercomputación

#### **Our Framework**

- Based on offline profiling + user-level interposition
- Manages memory at the dynamic allocation granularity

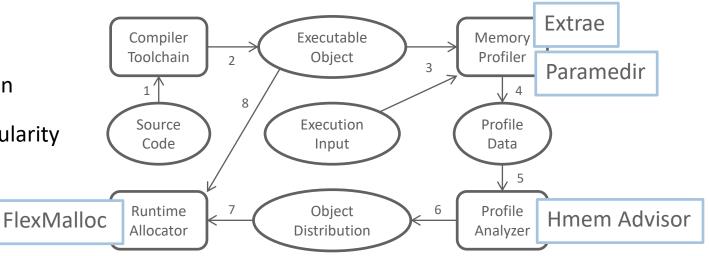
#### Steps

- Profiling (Extrae & Paramedir)
  - Collects allocation's address, size, and allocation point callstack (our object ID)
  - Sampling of precise profiling events (PEBS) for LLC load and L1 store misses, which include the accessed address
- Analysis (Hmem Advisor)
  - Computes per-object cost using heuristics based on profiling data to assign each object to a memory tier
  - Greedy relaxation of the 0/1 multiple knapsack problem + optional fine-tuning heuristics (e.g. BW-aware)
- Execution with data placement
  - Execution of the original binary with the FlexMalloc library, which interposes memory allocation functions to redirect each allocation to the corresponding memory tier



Initial framework work:

H. Servat, A. J. Peña, G. Llort, E. Mercadal, H. C. Hoppe, and J. Labarta "Automating the application data placement in hybrid memory systems" IEEE Cluster 2017



# **Experimental Evaluation**

#### Hardware

- Intel Xeon Platinum 8260L CPU @ 2.30GHz, HT disabled
- 16 GB of DRAM
- Two Optane configurations
  - PMem6: 6x 512 GB Optane<sup>™</sup> PMem
  - PMem2: 2x 512 GB Optane<sup>™</sup> PMem
- Software stack
  - Fedora 27 (kernel 4.18.8-100.fc27.x86\_64)
  - Intel Compiler Suite 2019u3
  - Intel MPI

- Hmem Advisor parameters
  - DRAM limit: 4, 8, 12 GB
  - Profiling data: Loads only, loads+stores
- Comparison with
  - Memory Mode
  - Experimental Kernel-level page migration (tiering-0.7 kernel)
  - ProfDP



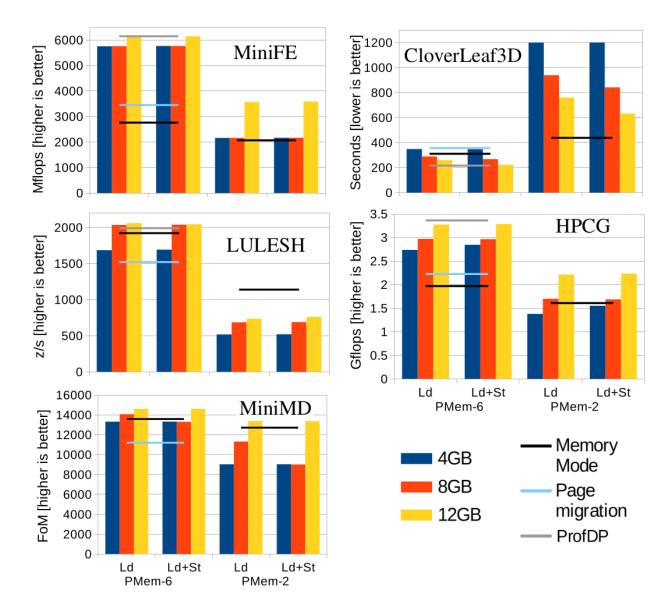
#### **Some Results**

#### Mini apps

- Comparison with memory-mode and an experimental kernel-level page migration approach
- With 12GB, in all PMem-6 and most Pmem-2 cases, our framework performs better than MM and KPM
  - In some of them even using 4x less DRAM
- Similar to ProfDP performance

#### Full apps

- LAMMPS
  - Ld: 4% overhead vs. MM
  - Ld+St: 3% overhead
- OpenFOAM
  - Ld: 3% speedup
  - Ld+St: 6.2% speedup
- More complex memory access patterns -> harder to capture object relevance in the cost heuristics
- We plan to analyse more applications in the future



Miniapps performance results



# **Ongoing & Future work**

- Applications:
  - Try out ecoHMEM with more HPC applications (DEEP-SEA project, ...)
  - TensorFlow
- Automatic application code modification
  - Alternative to Flexmalloc's runtime-interposition approach
  - Source-to-source compiler to modify app's allocation points according to Hmem Advisor output
- Integration with Kernel-level page migration approaches
  - Initial placement from ecoHMEM



# ecoHMEM 1.0 Released

- Download tarball from BSC SW repository:
  - <u>https://www.bsc.es/discover-bsc/organisation/scientific-structure/accelerators-and-communications-hpc/team-software</u>
- Source code currently available from EPEEC's Project repository:
  - <u>https://github.com/epeec/ecoHMEM</u>
- Flexible Memory Allocation Tool also available:
  - <u>https://github.com/intel/flexmalloc</u>



# Acknowledgements

- European Commission's EuroHPC project DEEP-SEA under grant agreement 955606
- Intel-BSC Exascale Laboratory SoW 5.1
- European Union's H2020 Marie Sklodowska-Curie grant agreement No. 749516
- European Union's H2020 Project EPEEC under grant agreement No. 801051
- The Spanish Ministerio de Ciencia e Innovación—Agencia Estatal de Investigación











Barcelona Supercomputing Center Centro Nacional de Supercomputación



# Thank you

marc.jorda@bsc.es

