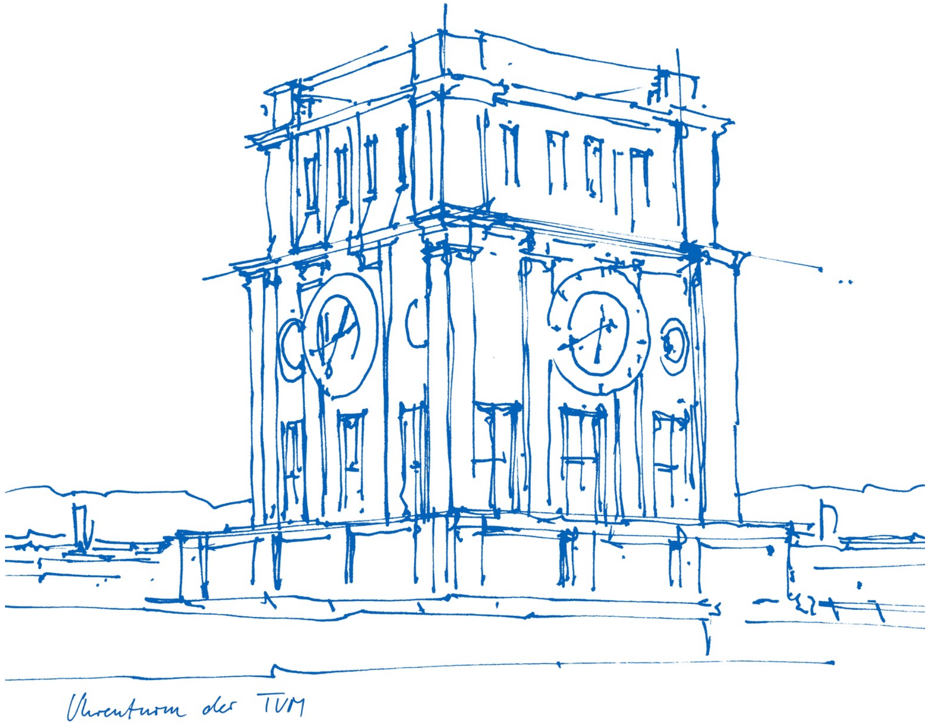
Toward Dynamic Orchestration of Data/Power/Process Management for Hybrid Memory Based Systems

Eishi Arima, Carsten Trinitis, Martin Schulz

Special thanks to: Michael Gerndt

CAPS, TUM

eishi.arima@tum.de



Summary

- *As memory systems become more and more complicated, the memory access/usage behavior play a more significant role in various system optimiztions incl. data, power, and process management*
- Modern systems/applications are prone to be bottlenecked by memory accesses, thus memory performance directly affects total system performance in many cases
- Due to the more ***complicated*** memory configurations, the memory/system performance becomes more ***difficult to predict***
- We have observed the impact of memory acsssess/usage behavior on various optimizations on hybrid-memory-based systems in our prior studies

We should revisit system optimizations so that they become more aware of ***memory-related factors*** *and operate in a* ***coordinated*** *and* ***dynamic*** *manner*

Technology trends

Background: Moore's law is slowing down, and the end is inevitable

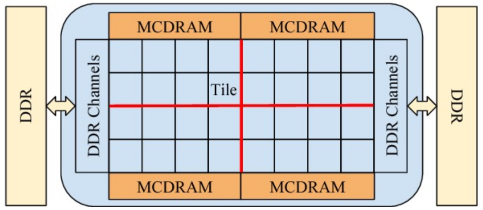
Current/future trend: Extremely heterogeneous system architecture

- Equiped with multiple different accelalators or devices at each component
- GPUs, FPGAs, AI chips, in-memory accelalators, and even quantum computers

Our focus today: Hybrid-memory-based systems

- Memory systems composed of multiple different memory technologies
- HBM, NVRAM, DDR DRAM – they all have pros and cons

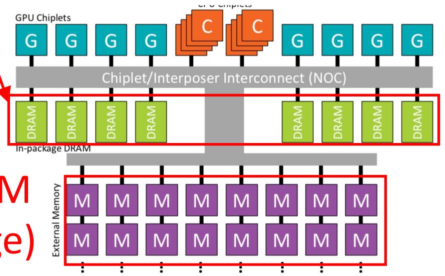
3D stacked MCDRAM (Fast)
+ DDR4 DRAM (Large)



Intel Knights Landing

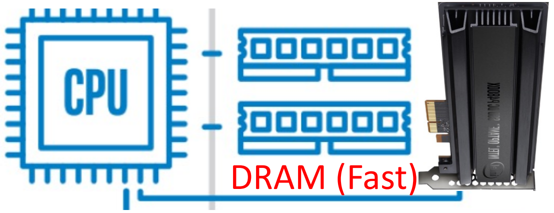
HBM (Fast)

DIMM
(Large)



AMD Future Exascale APU
[T. Vijayaraghavan+ HPCA17]

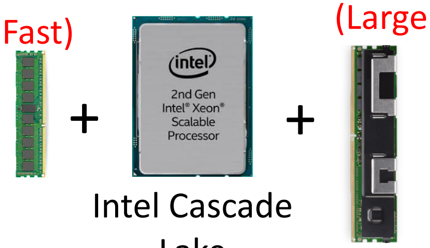
NVRAM (Large)–used
as an extension of
main memory



DRAM DIMM + Optane SSD

DRAM
(Fast)

NVRAM
(Large)



Intel Cascade
Lake

DRAM DIMM + Optane DIMM

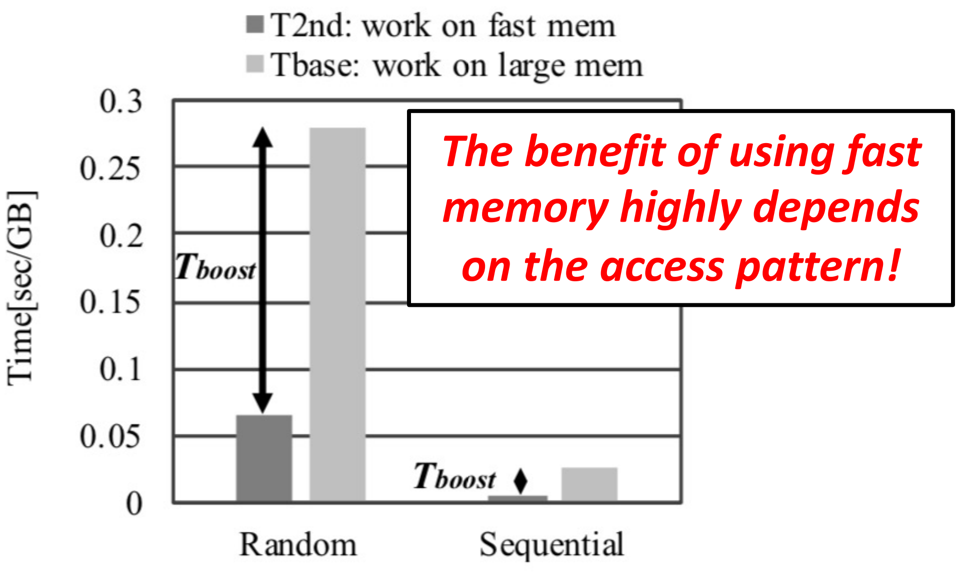
Our previous studies around hybrid memory systems

Pattern-Aware Staging [ISC’20]: An access-pattern-aware data allocation optimization

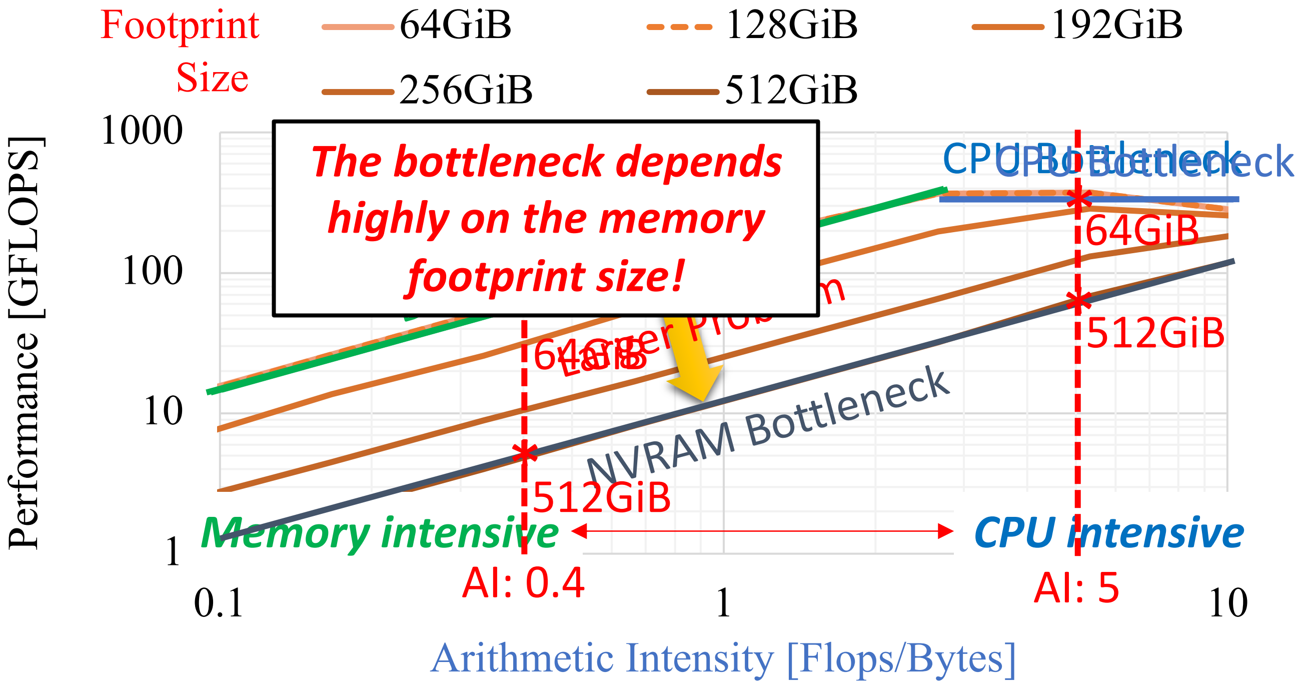
Footprint-Aware Power Capping [ISC’20]: A memory-footprimt-aware power management

Footprint-Aware Co-Scheduling: A process or job scheduling concept that is also explicitly aware of memory footprint

These studies are aware of the memory access/utilization behavior on hybrid memory systems!



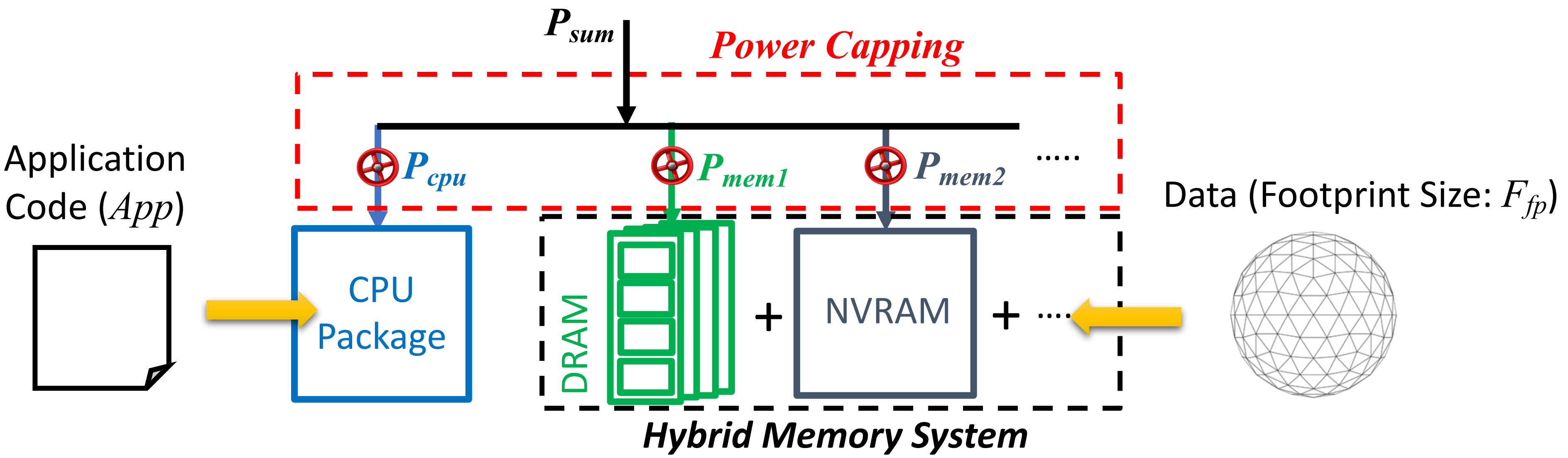
The benefit of using fast memory highly depends on the access pattern!



The bottleneck depends highly on the memory footprint size!

One example: footprint-aware power capping

- Based on the observation, we proposed a power management concept called ***footprint-aware power capping***
  - Under a given total power constraint (P_{sum}), we optimize the power allocation combination $\{P_{cpu}, P_{mem1}, P_{mem2}, \ldots\}$ while explicitly considering the data footprint size (F_{fp}) in addition to other features of application (App) such as arithmetic intensity.
  - Inputs: $\{App, F_{fp}, P_{sum}\}$ → Outputs: $\{P_{cpu}, P_{mem1}, P_{mem2}, \ldots\}$



Our key insight based on our prior works

What we learnt from our prior studies were:

- The ***memory access/utilization behavior*** matters for optimizing hybrid-memory-based systems as it can impact performance more significantly than ever before
- System optimizations should be aware of the ***memory-related factors***, and they should be conducted in an ***orchestrated*** and ***dynamic*** manner

Orchestrated:

- They are basically ***connected*** and ***interacting*** each other because they are fucntions of the the memory access/utilization behavior as well as the data management policy (these aspects are less important for traditional monolithic memories)

Dynamic:

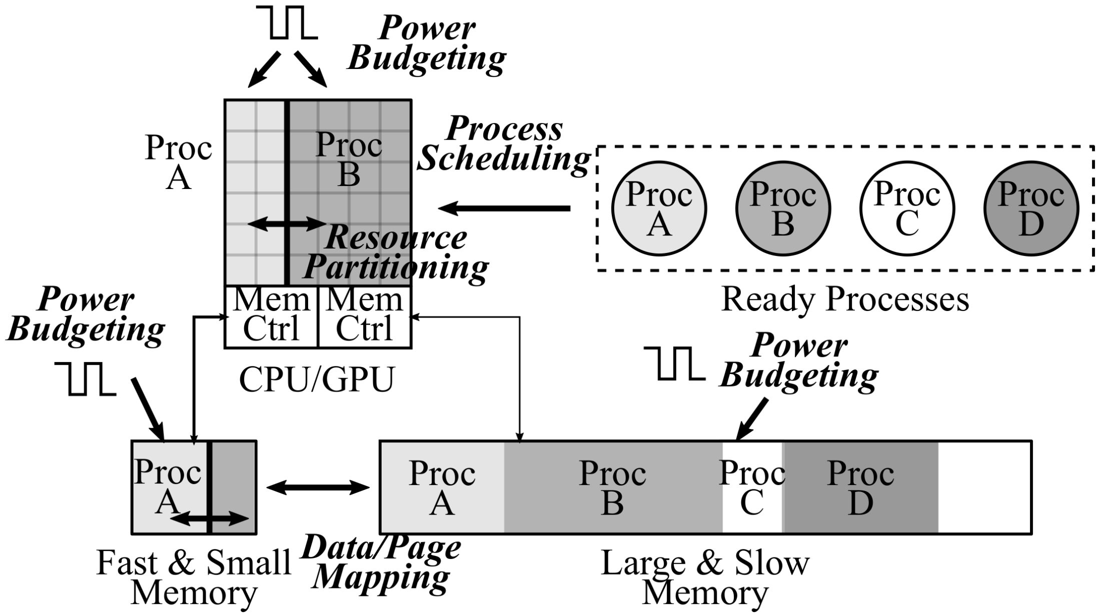
- The memory access/utilization behavior is ***dynamic information***, and thus dynamic analyses/optimizations are also required for this purpose
- Suited for the ***operating system layer*** as well as have to be ***co-designed*** with the hardware side

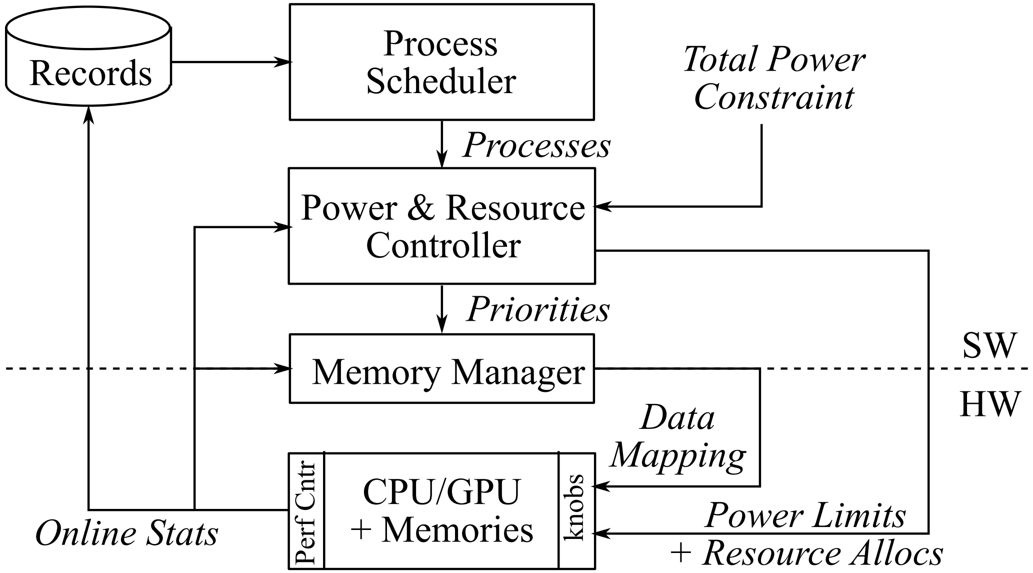
Goal and overall solution

Goal: maximizing a given objective func. (e.g., system throughput) by dynamically orchestrating data/power/process management on hybrid-memory-based systems

Solution: A top-down and feedback-driven approach

- Top-down to reduce the complexity; Feedback for adaptive optimization

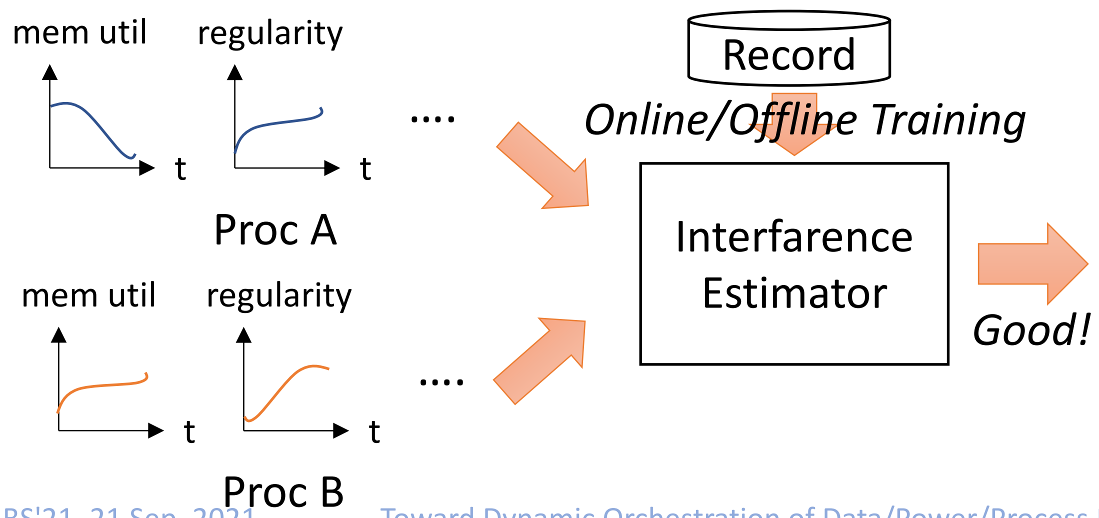


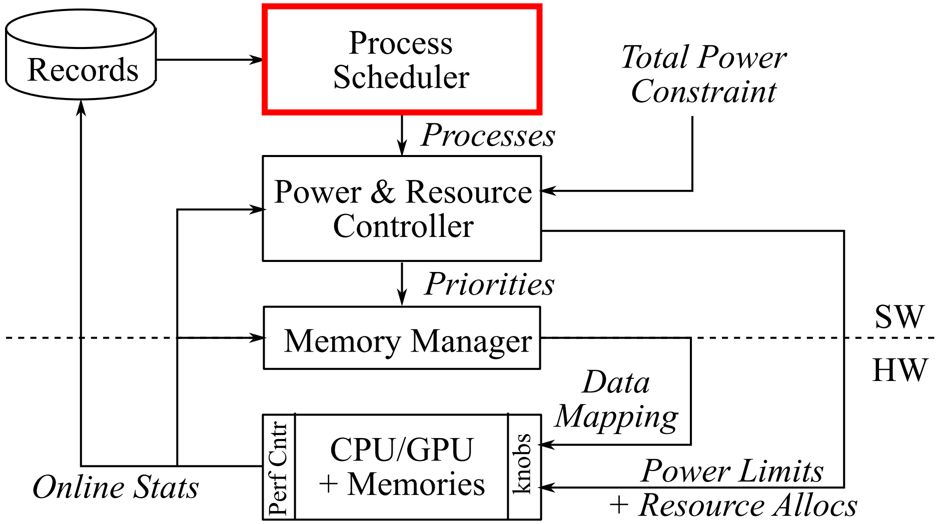


Components and major challenges

Process Scheduler: responsible for selection of co-running processes

- A history-based approach: model/evaluate the co-run intereference among arbitrary process combinations using the stats of previous runs
- Should be aware of the history of ***memory access/utilization behavior*** in a time series format
- Doesn’t matter what the other components are doing – they are a blackbox
- ***Applying AI to the model will be promissing***





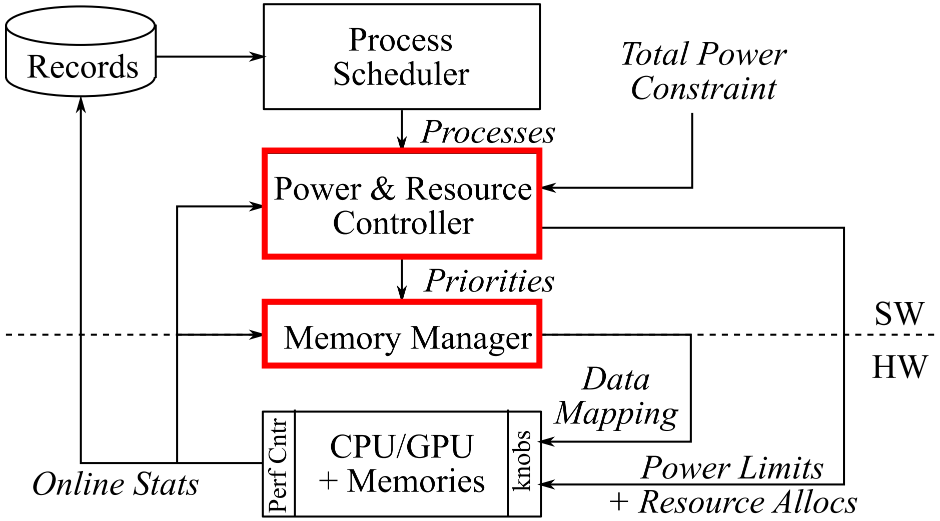
Components and major challenges 2

Power & Resource Controller: responsible for power/resource allocations and data allocation priorities for a given set of co-run processes

- Need power/performance modeling and allocation algorithms using dynamic stats
- A control-theory-/AI-based approach is a promising direction
- ***What is the necessary & sufficient set of stats?***

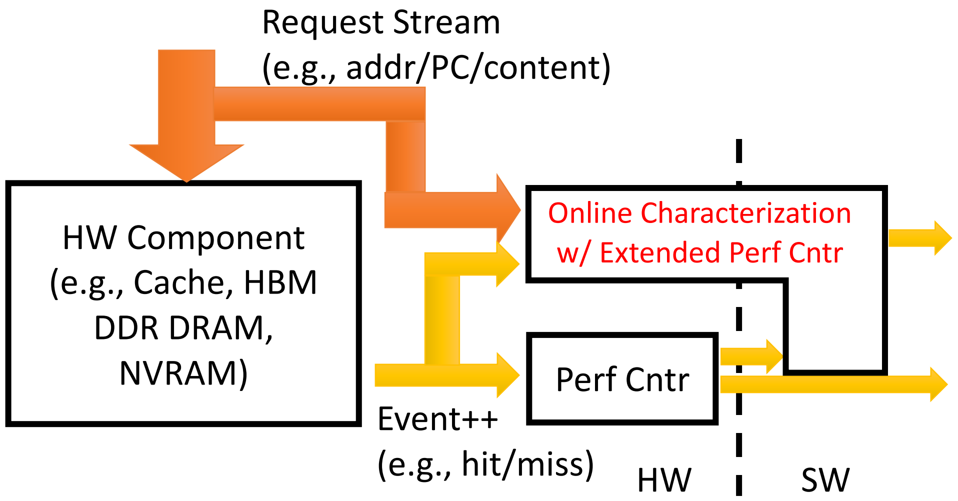
Memory Manager: HW and/or SW

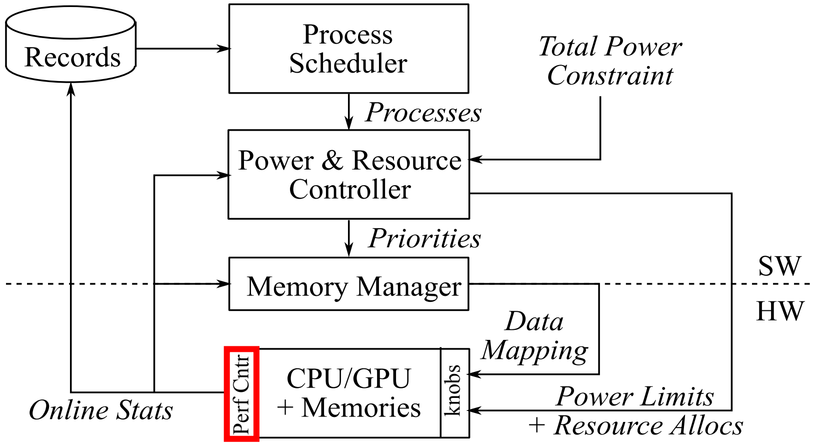
- Data management policy – a pattern-aware approach is promising
    - e.g., our ISC’20 paper
- Need a priority setting function
    - e.g., partitioning
- Making the policy selectable by others will be a good option



Further challenges: revisiting performance counters

- We may need to revisit also the sensor side in the optimization loop, i.e., the perfromance counters
- Conventional performance counters just count the number of events on each component, which could be more benefitial if they would provide more info
  - Today's memory analysis SW tools won't be suitable for ***dynamic*** analysis due to the OH
- This is a good SW/HW co-design research opportunity





Conclusion

The memory access/usage behavior as well as the memory management play an important role in a variety of optimizations on hybrid memroy based systems

- Power management, process scheduling, and others
- These optimizations should be aware of ***memory-related aspects*** and should work in a ***coordinated/dynamic*** manner

This must be the case also for other disruptive memory architectures or concepts, not limited to hybrid memory systems

- Systems with near/in-memory accelelators; non-volatility support in main memories
- We are interested also in how these disruptive technologies affect the system optimizations:
  - What parameters we shoud focus on
  - How the modeling and algorithms should be changed
  - How the optimization methodologies/frameworks should be like
  - How we should extend ours to support them

Thank you

for your attention!