Adaptive Scheduling for Systems with Asymmetric Memory Hierarchies

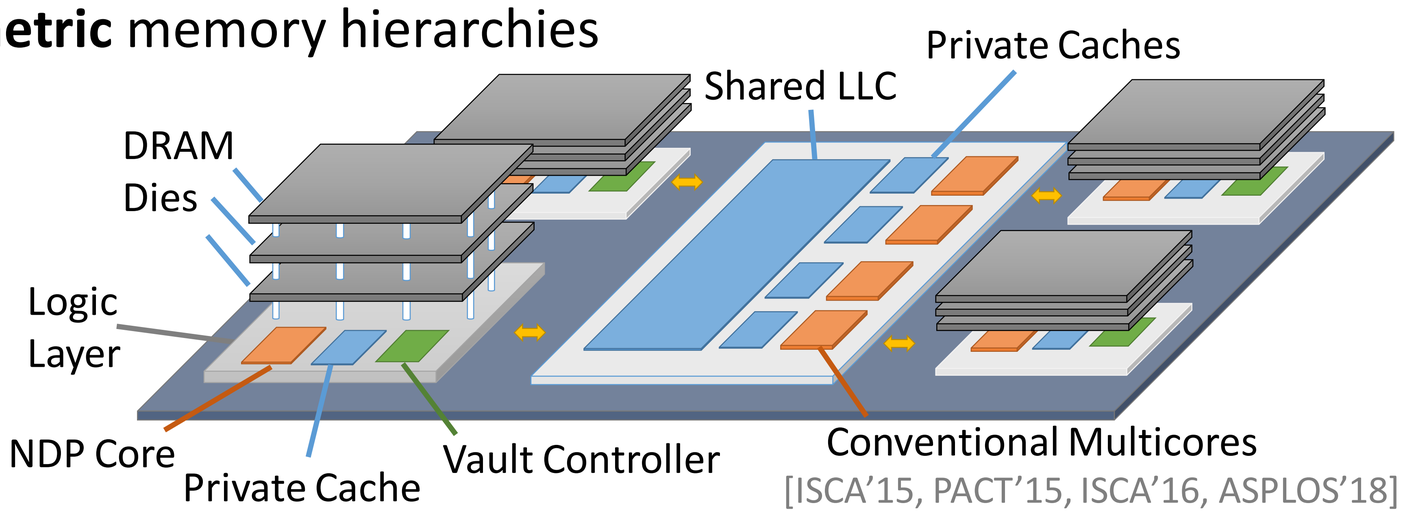
Po-An Tsai, Changping Chen, and Daniel Sanchez
{poantsai, cchen, sanchez}@csail.mit.edu

Background

3D stacking has enabled systems with **asymmetric** memory hierarchies

Deep hierarchy: Conventional multi-core processors with multi-level cache hierarchy

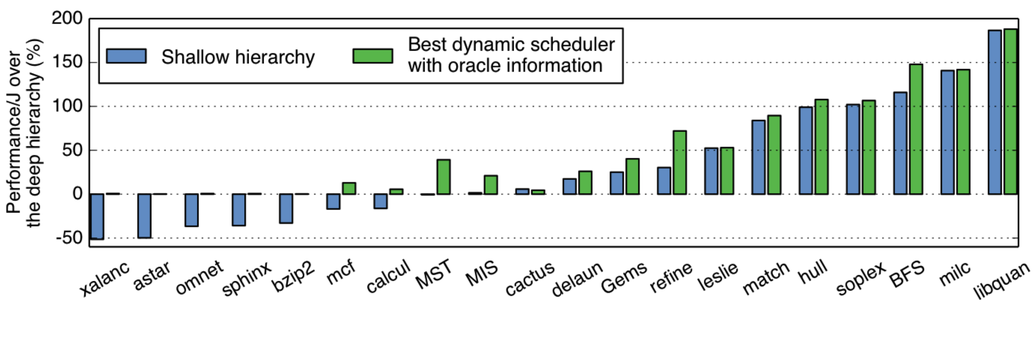
Shallow hierarchy: Near-data processing cores with shallow hierarchies using few cache levels between cores and memories



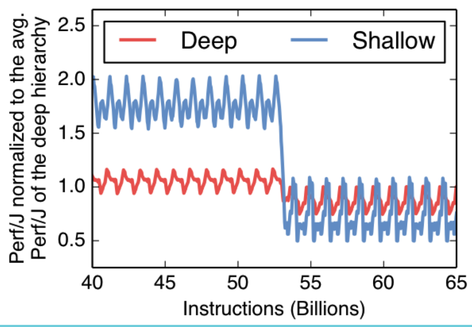
[ISCA'15, PACT'15, ISCA'16, ASPLOS'18]

Scheduling Applications to the Right Hierarchy is Challenging

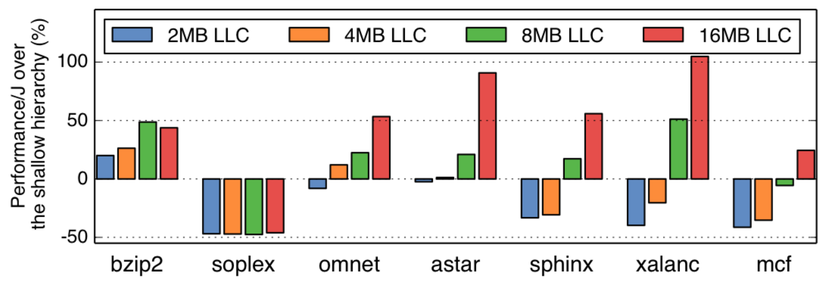
Challenge 1: Apps have different preferences



Challenge 2: Preferences change over time



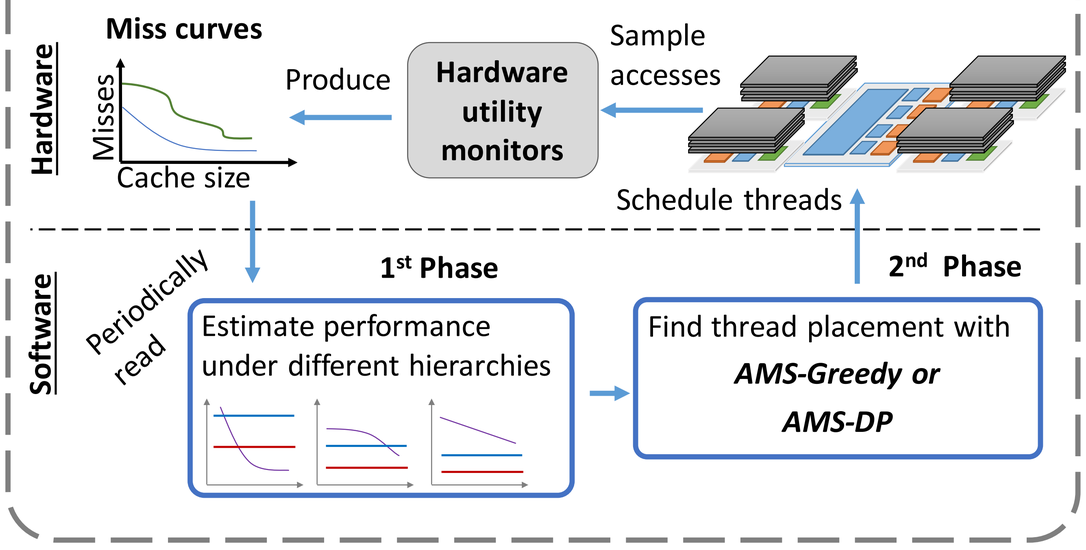
Challenge 3: Preferences change when LLC capacity is contented



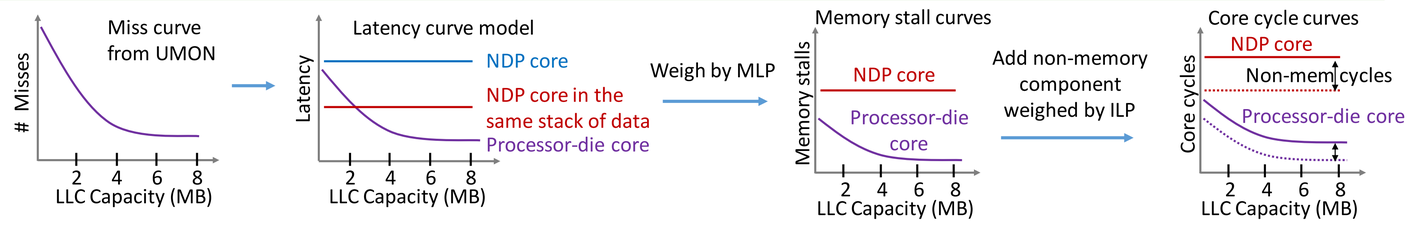
AMS: Adaptive Scheduling for Asymmetric Memory Systems

Insight: Modeling a thread's preferences to different hierarchies bears a strong resemblance to the cache partitioning problem!

AMS overview



Contribution 1: Analytical model to account for asymmetries

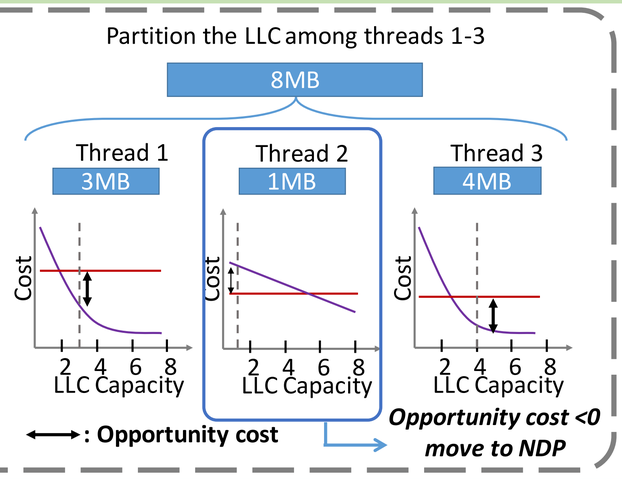


Contribution 2: Two thread placement algorithms that extend techniques originally designed for cache partitioning

1. AMS-Greedy

Performs multiple rounds of cache partitioning and uses its outcomes to greedily map threads to hierarchies

Low-overhead yet high-quality



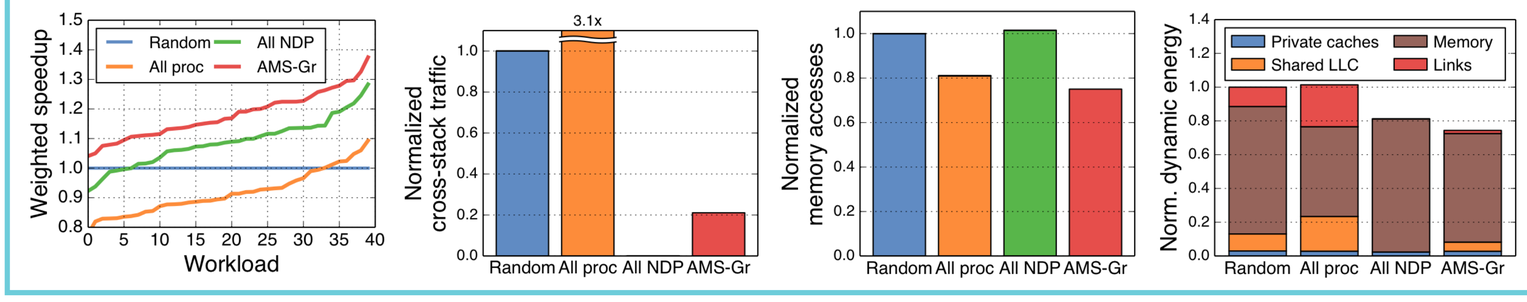
2. AMS-DP

Leverages **dynamic programming** to explore the full space of decisions efficiently, finding the optimal schedule given the analytical model

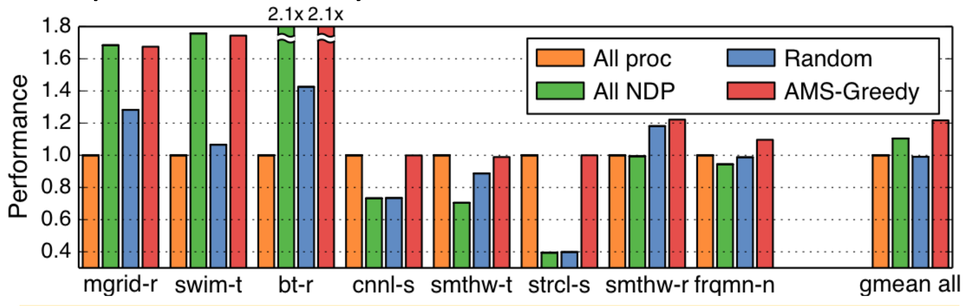
More expensive but finds the optimal schedule under the model

Evaluation

1. **Methodology:**
   8-core processor die; 3-level deep hierarchy with 16MB shared LLC
   4 NDP stacks; 2 cores per stack; 2-level private-cache-only shallow hierarchy
2. **Multi-programmed results**
   AMS-Greedy finds the right hierarchy for each application. It never hurts performance and improves weighted speedup by up to 37% and by 18% on average over the Random baseline.



3. **Multi-threaded results**
   AMS also handles multithreaded workloads under asymmetric hierarchies, improving gmean performance by 22% over the Random baseline



See the paper (https://goo.gl/3yjDmK) for more results: case study of AMS adapting to phases, comparison with prior contention- and core-asymmetry-aware schedulers