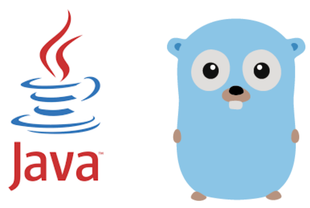
Rethinking the Memory Hierarchy for Modern Languages

Po-An Tsai, Yee Ling Gan, and Daniel Sanchez
{poantsai, elainegn, sanchez}@csail.mit.edu

Motivation

Object-based memory model enables memory safety & automatic memory management (GC)

+ *simplifies programming*
+ *prevents memory bugs*
- *adds overheads*



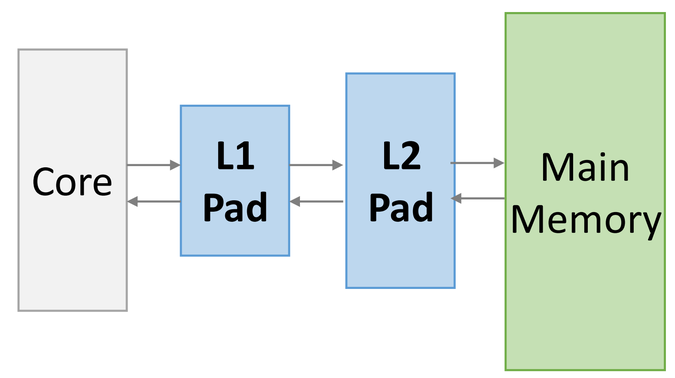
Overheads due to mismatch between flat memory systems and object-based, memory-safe languages

Key Insight: Hiding memory layout in hardware enables new optimizations for object-based programs

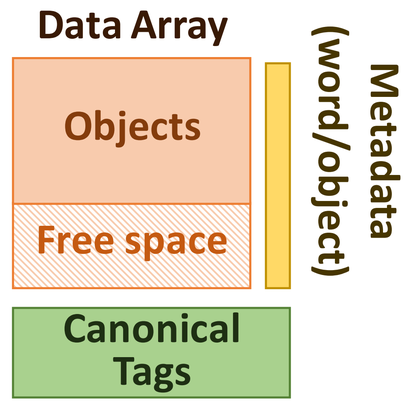
Hotpads Overview

A novel **memory hierarchy designed for memory-safe, object-based languages** that support objects and pointers at the ISA level

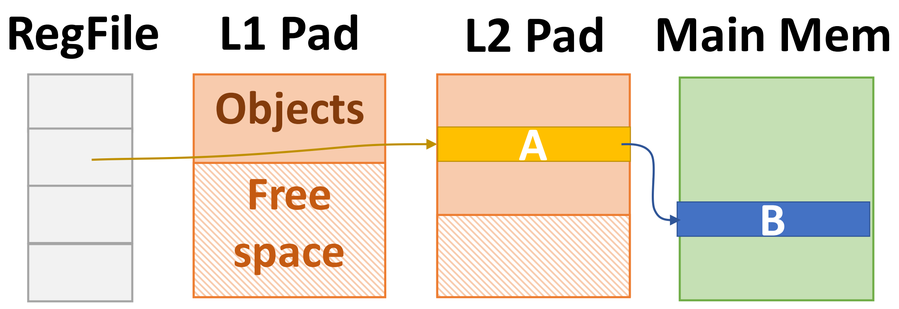
Example Hotpads hierarchy



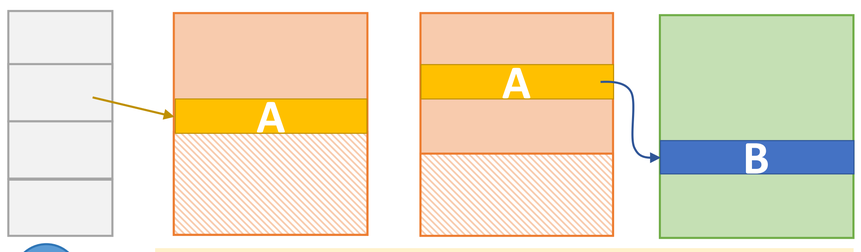
Pad Organization



Hotpads Key Features

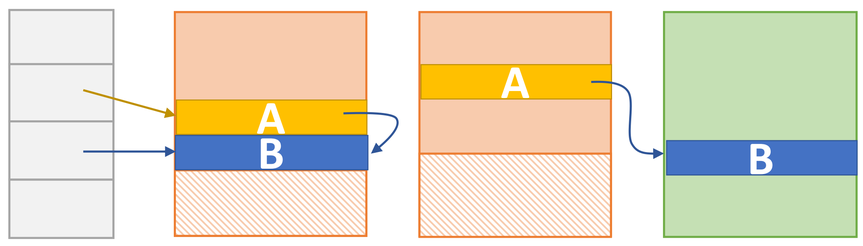


1 Initial state with 2 objects A and B



2 Read A

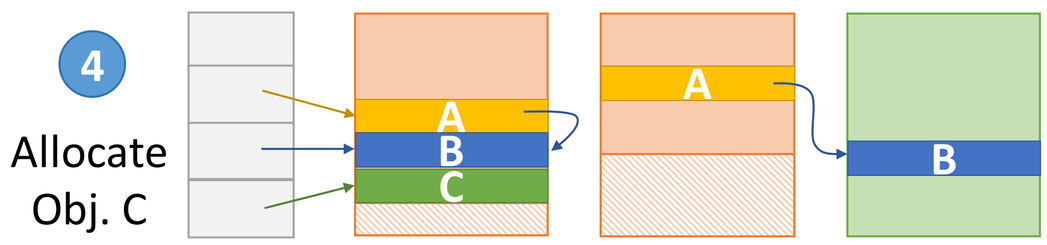
Key Feature 1: Implicit object movement in response to accesses



3 Read B via A

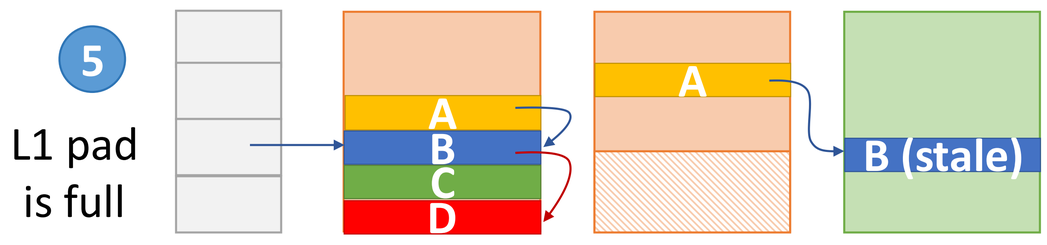
Key Feature 2: Pointer rewriting

Future dereferences avoid associative lookups and thus consume less energy.



Objects are allocated in the fast, efficient L1 pad, rather than allocated in and fetched from main memory.

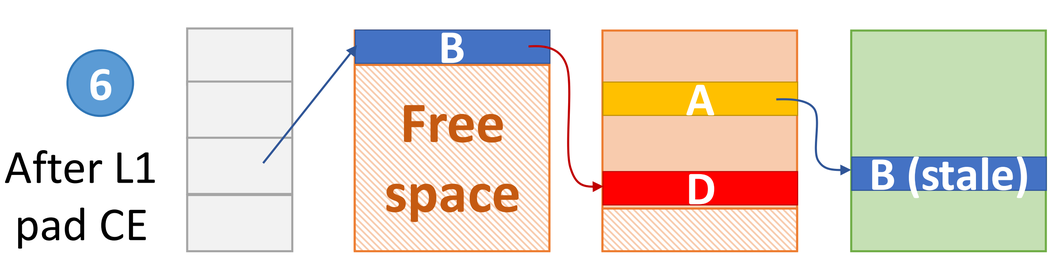
Key Feature 3: In-hierarchy object allocation to reduce memory traffic



C is dead (not pointed by). **D** is alive (pointed by B). **A** and **B** are not collected because they are copies . **B** is recently-used.

Key Feature 4: Unifying hierarchical GC and data movement in hardware

Many objects die young and never reach main memory. Frequently used objects are kept in small, fast pads.



Collection-Eviction (CE):

1. Find roots
2. Mark live objects
3. Compact & evict live objects
4. Update pointers

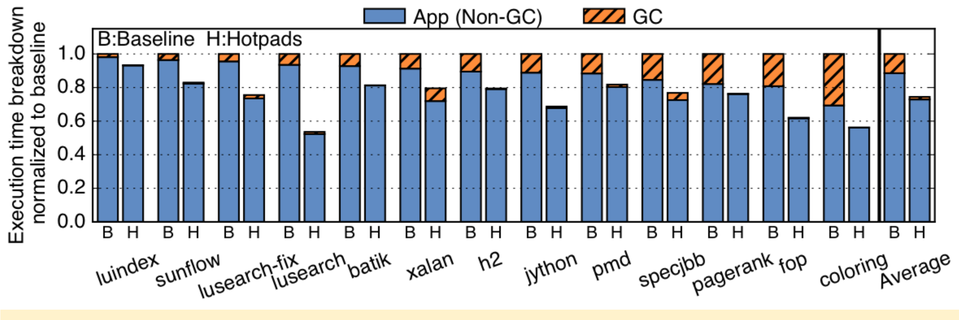
Evaluation

Methodology:

- Simulate Hotpads using Maxsim simulator (Zsim+Maxine JVM)
- **13 Java Applications** from Dacapo, Jgrapht, SpecJBB
- 4 Westmere-like OOO cores
- 3-level cache/pad hierarchy

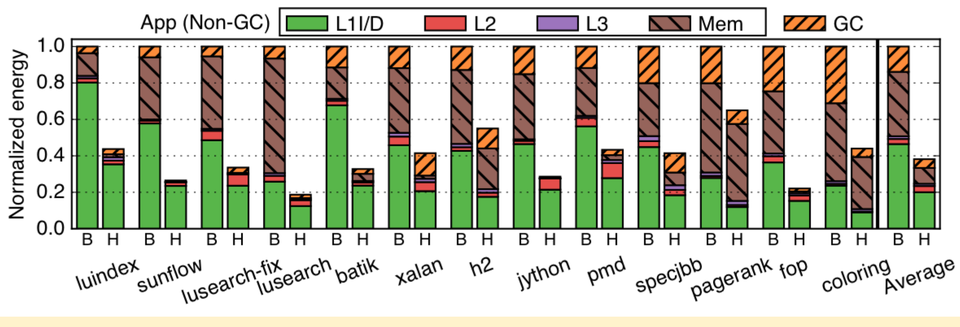
Execution time breakdown:

- GC cost reduced by 8x with Hardware CE
- 34% performance improvement



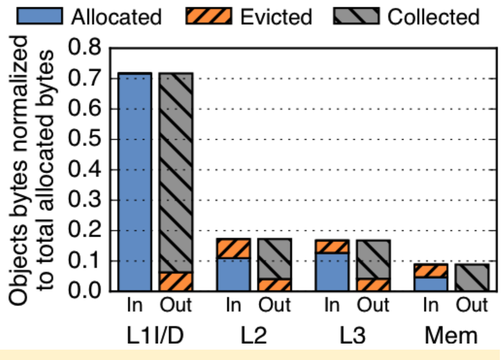
Dynamic memory hierarchy energy:

- Dynamic energy reduced by 2.7x
- Large reductions in L1D, Mem and GC



Hotpads acts like a super-generational collector:

Most objects are collected in the L1 pad. 90% of them never reach main memory



See the paper (https://goo.gl/eXzG6a) for: Pointer rewriting and CE analysis, legacy mode for running conventional apps, coherence, and more!