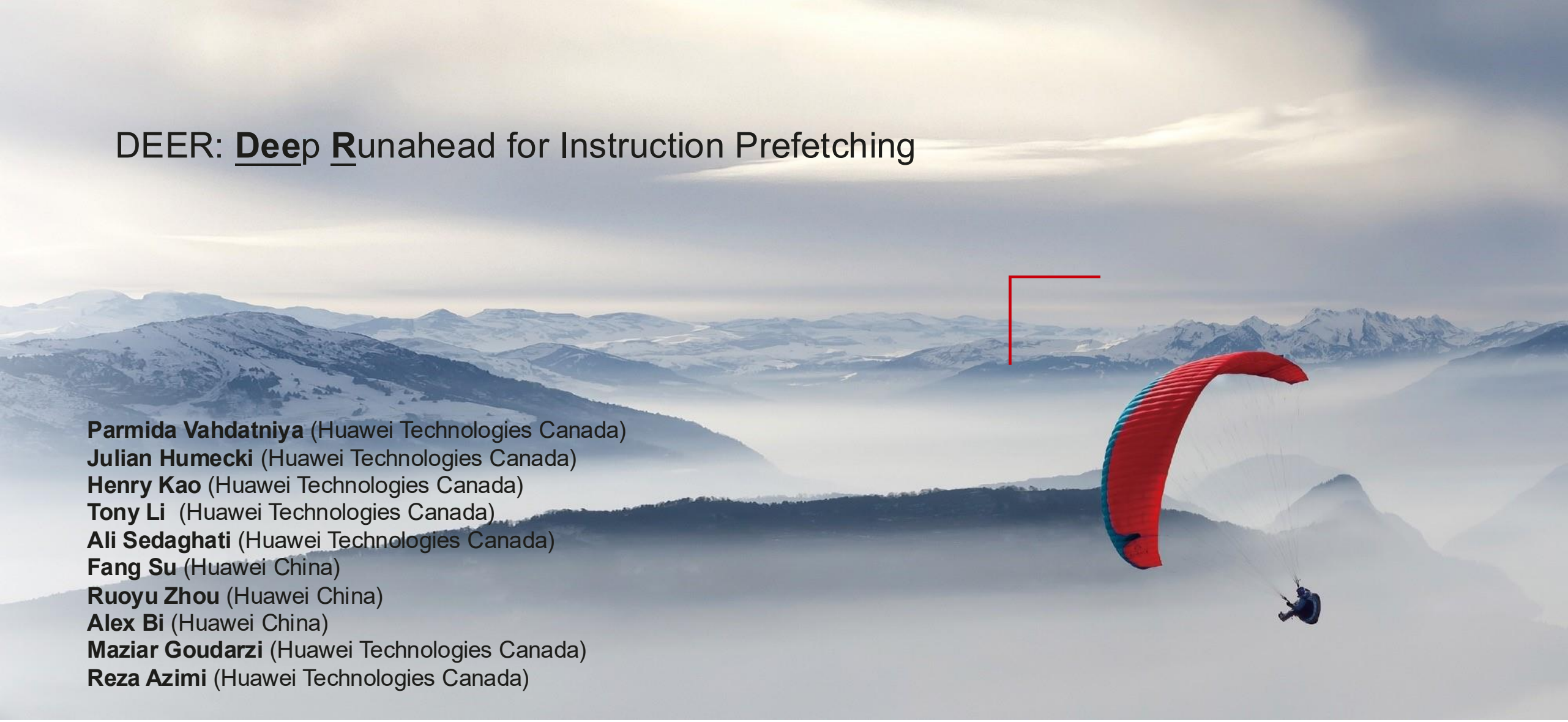
DEER: **Dee**p **R**unahead for Instruction Prefetching



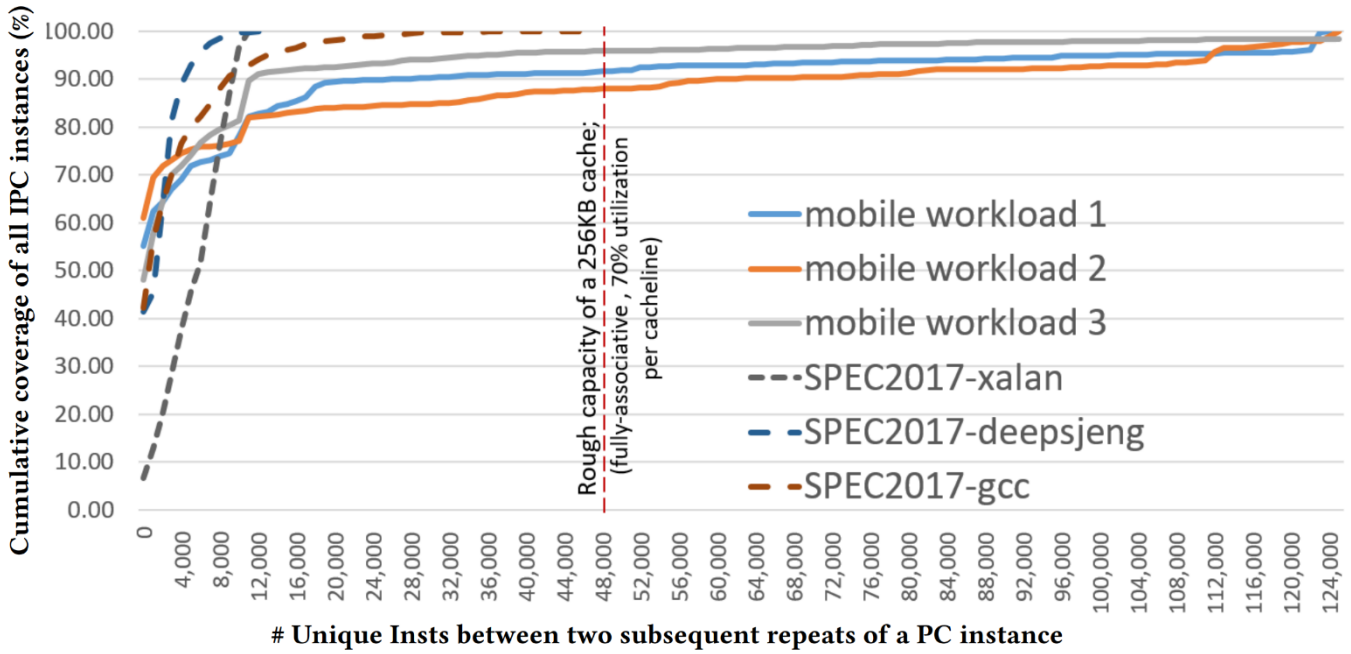
Parmida Vahdatniya (Huawei Technologies Canada)
Julian Humecki (Huawei Technologies Canada)
Henry Kao (Huawei Technologies Canada)
Tony Li (Huawei Technologies Canada)
Ali Sedaghati (Huawei Technologies Canada)
Fang Su (Huawei China)
Ruoyu Zhou (Huawei China)
Alex Bi (Huawei China)
Maziar Goudarzi (Huawei Technologies Canada)
Reza Azimi (Huawei Technologies Canada)

ARM-based General-Purpose Computing @ International Symposium on Computer Architecture (ISCA), June 21st 2025, Tokyo, Japan

HUAWEI

Introduction and Motivation

- Mobile workloads face heavy frontend stalls due to large code footprints and long repeat cycles.
  - > Complex
  - > Using hundreds of libraries with thousands of functions
  - > Deep cross-library calls
  - > Exhibiting a long tail in PC repeat distance
  - > Not representable by SPEC CPU, GeekBench, etc…



- Existing prefetchers (hardware and software) struggle with:
  - > Insufficient coverage
  - > High storage and energy costs

arXiv paper: https://arxiv.org/abs/2504.20387

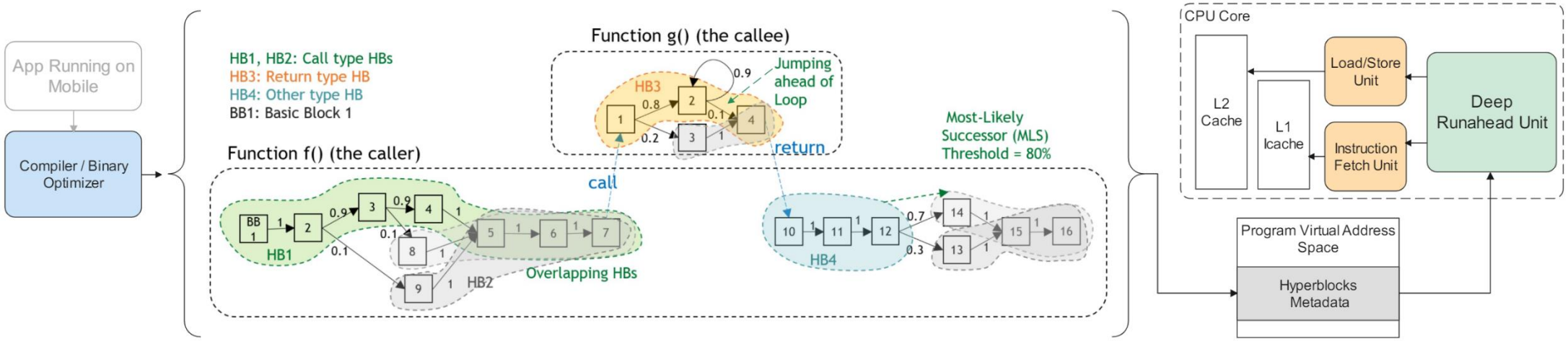
HUAWEI

Challenges of Existing Prefetching Techinques

- Software-Only Prefetchers:
  - High runtime overhead
  - Hard to place in the right locations
  - Limited effectiveness with cross library prefetching
- Hardware-Only Prefetchers:
  - Need large, expensive storage to track instruction streams
  - Power & area costs
  - Limited by the accuracy of branch predictors
- Record-and-Replay Approaches:
  - Offer some improvement
  - Significant area and power overhead
  - Not good for resource-constrained mobile environments

DEER (**Dee**p **R**unahead Prefetcher) Overview

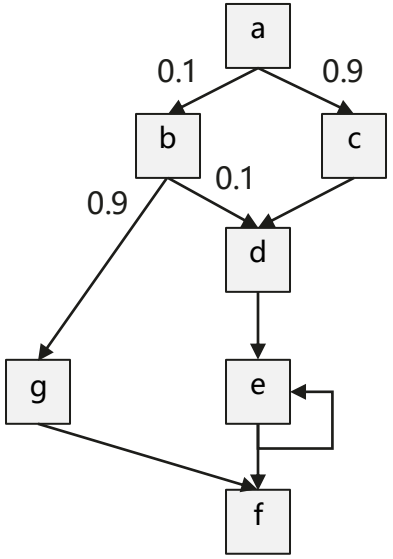
- Software-Hardware co-designed technique
- Offline Record: profiling & offline analysis to predict future instruction streams, even across complex control flows
- Compact metadata describes likely future instruction cache lines
- Online Replay: Hardware components uses this metadata to prefetch instructions

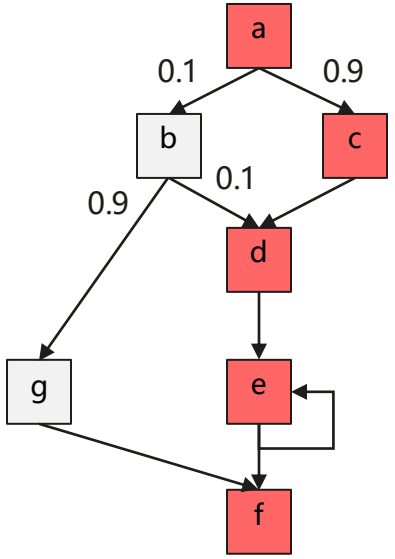


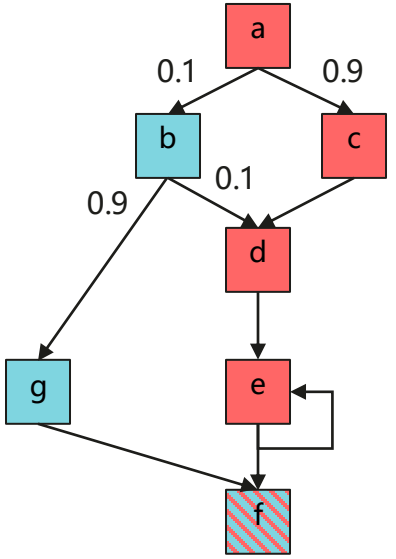
HUAWEI

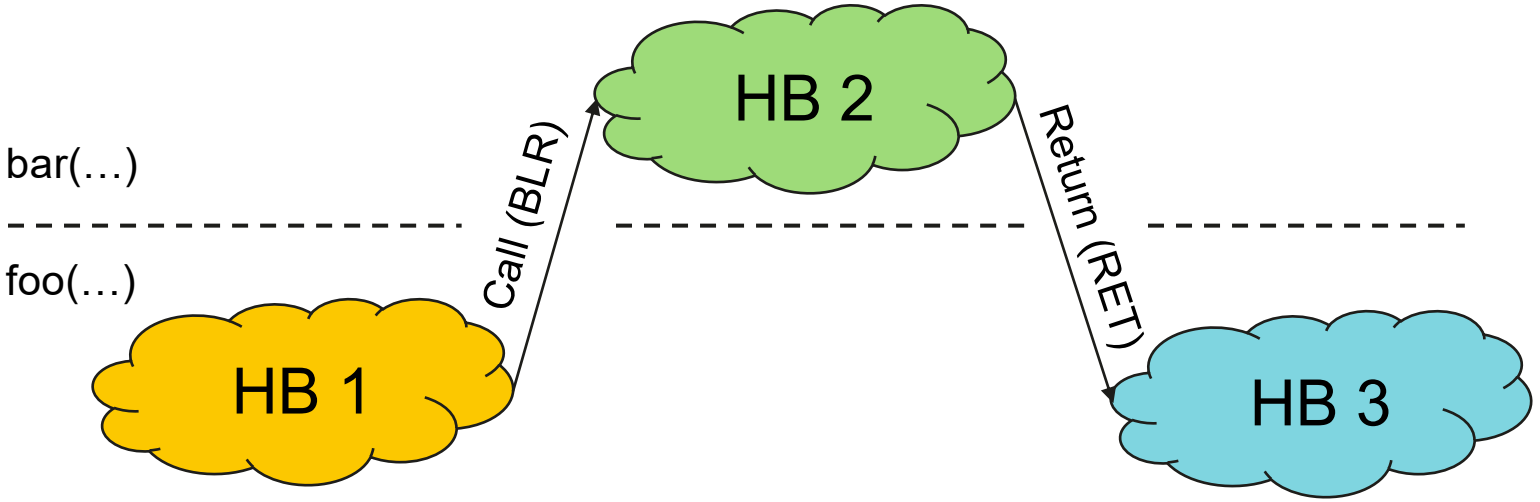
Offline Profiling + Analysis & HyperBlocks

- Coarse-grained control flow stability exists in programs
- Form HBs to represent a stable unit of execution within a function
- Branch Profiling using ARM BRBE
- Offline tools to build stable paths from BRBE profiles and compiler analysis
- Chain HB together to form interprocedural stable paths







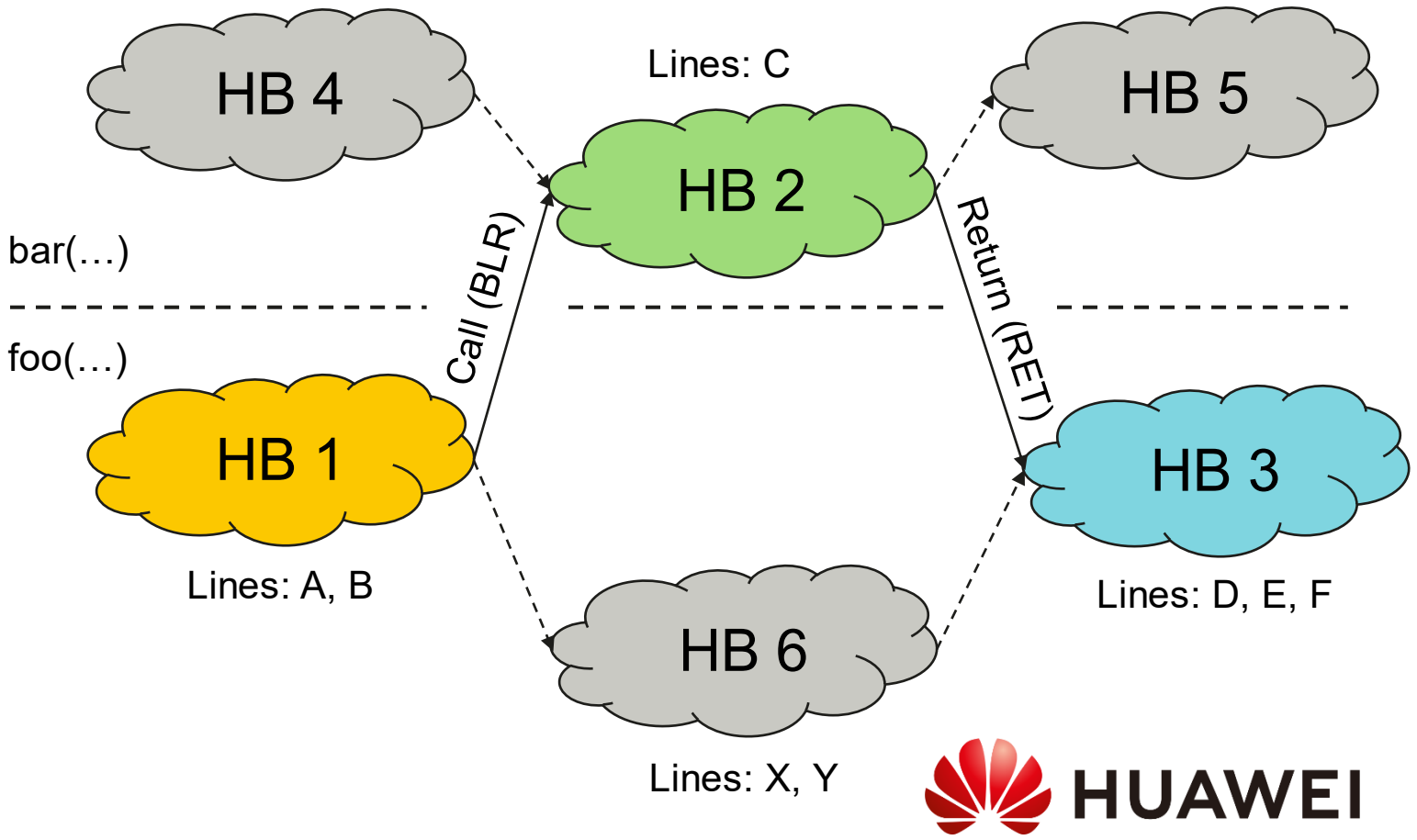


HUAWEI

Encoding Prefetch Chains

HyperBlock	Prefetch Lines
HB1	A, B, C, D, E, F
HB2	C
HB3	D, E, F

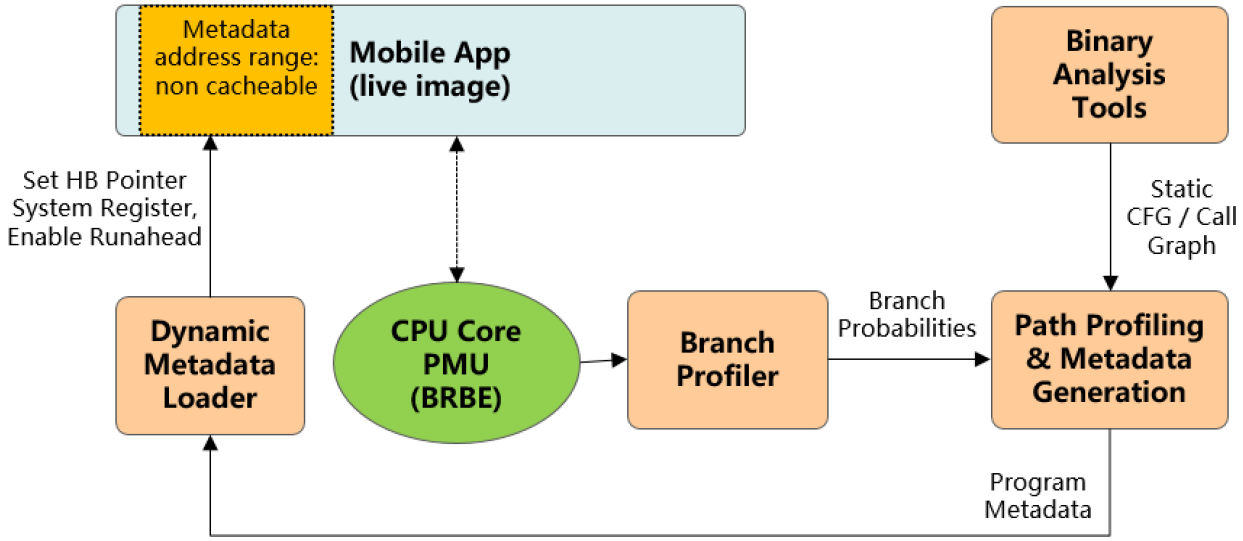
- Embed stable HB chains as metadata into memory
- Prefetch the lines when PC reached the start of an HB
- Prefetch redundancy might exist
- However, necessary in the case where code takes an alternate path
- No context-sensitivity

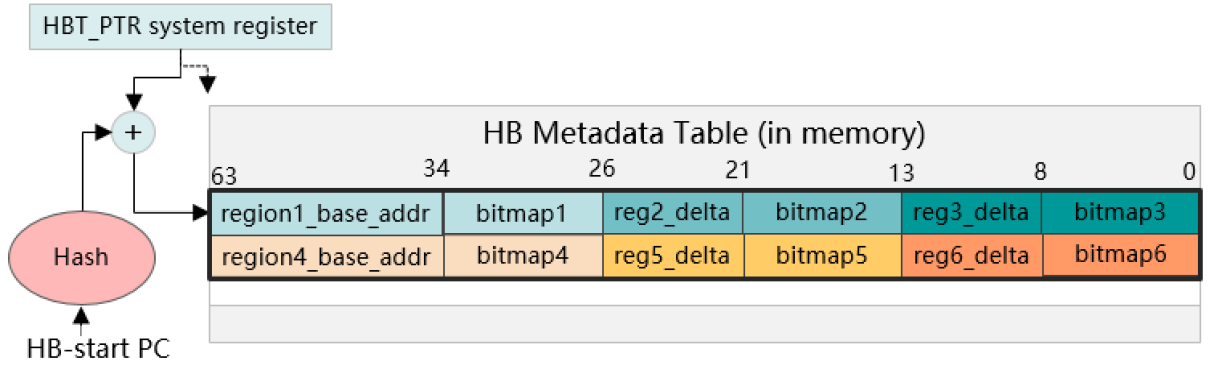


HUAWEI

Embedding the Prefetch Metadata

- Metadata embedded into the binary from the dynamic loader
- Start address of the metadata in memory stored in a system register (saved and restored in context-switch)
- Each HB metadata uses 16B and encodes up to 48 instruction cache lines.
- Hash of PC and Start address of metadata table is used to fetch prefetch metadata from memory

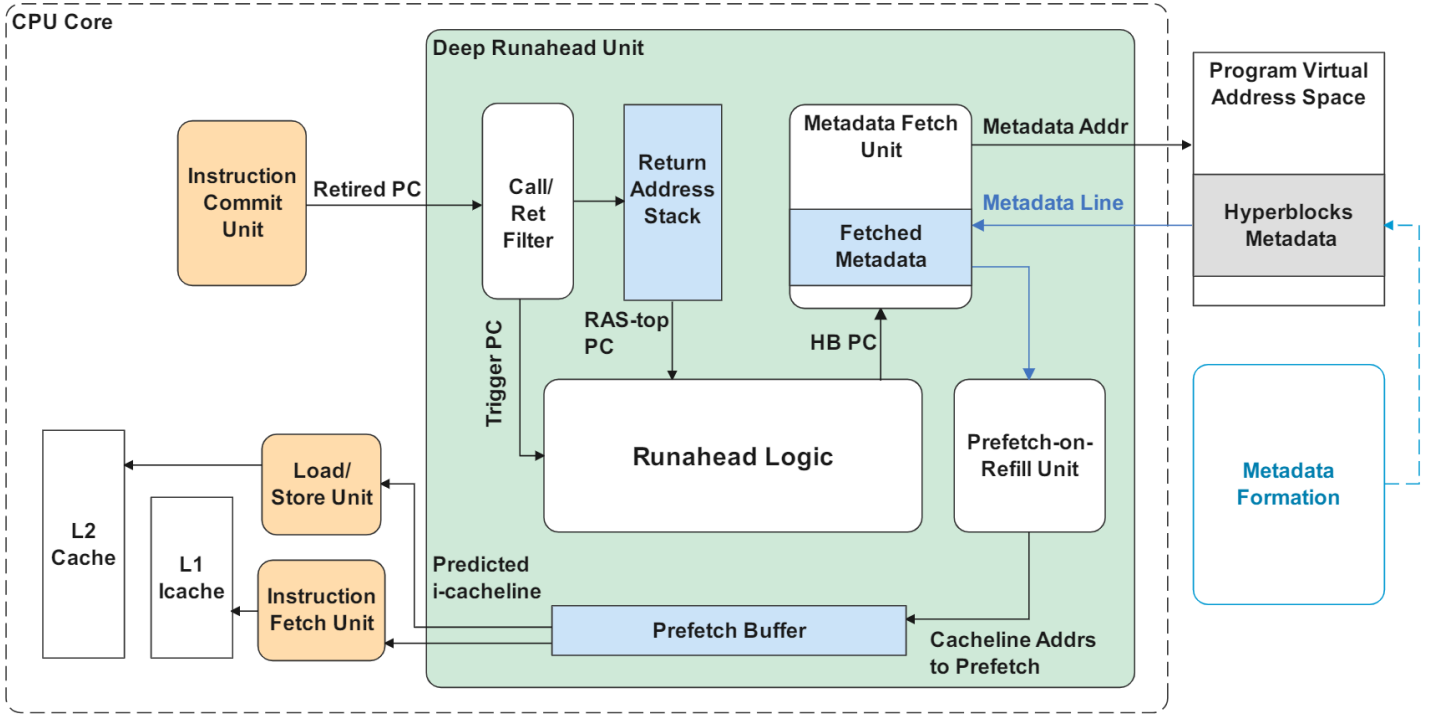




HUAWEI

Hardware Modifications

- Use retired/commited PC to generate a metadata fetch address from memory
- Prefetch line addresses get extracted from the metadata once returned from memory
- Prefetch address get pushed onto the prefetch buffer



HUAWEI

Evaluation Setup

- gem5 in SE mode
- O3 ARM core
- 256KB L1 I/D, 2MB unified L2
- Stride prefetchers

- 15 simpoints captured from real mobile apps across various categories (news, games, video players, social networks, etc…)

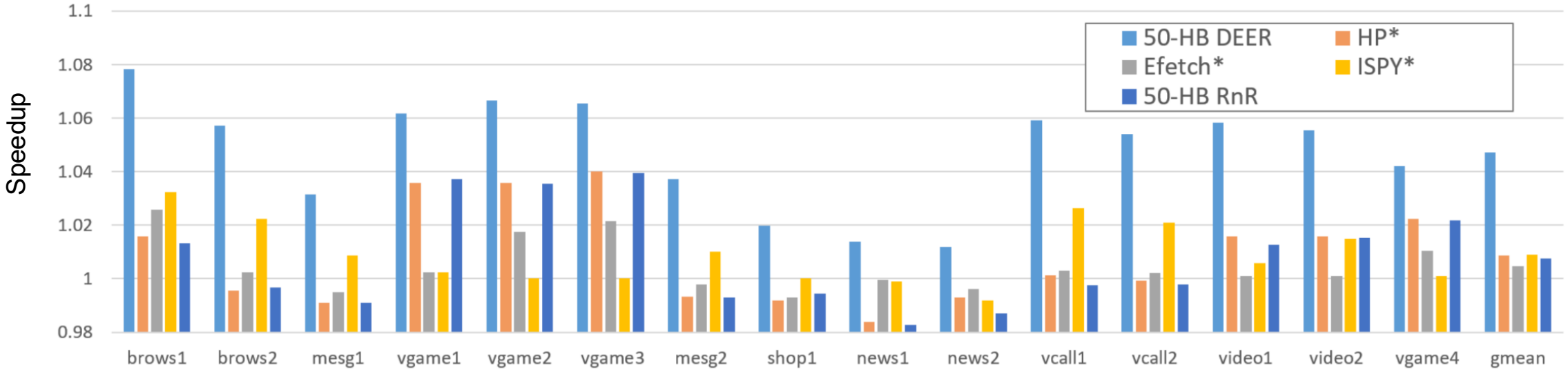
Evaluated Prior Art:

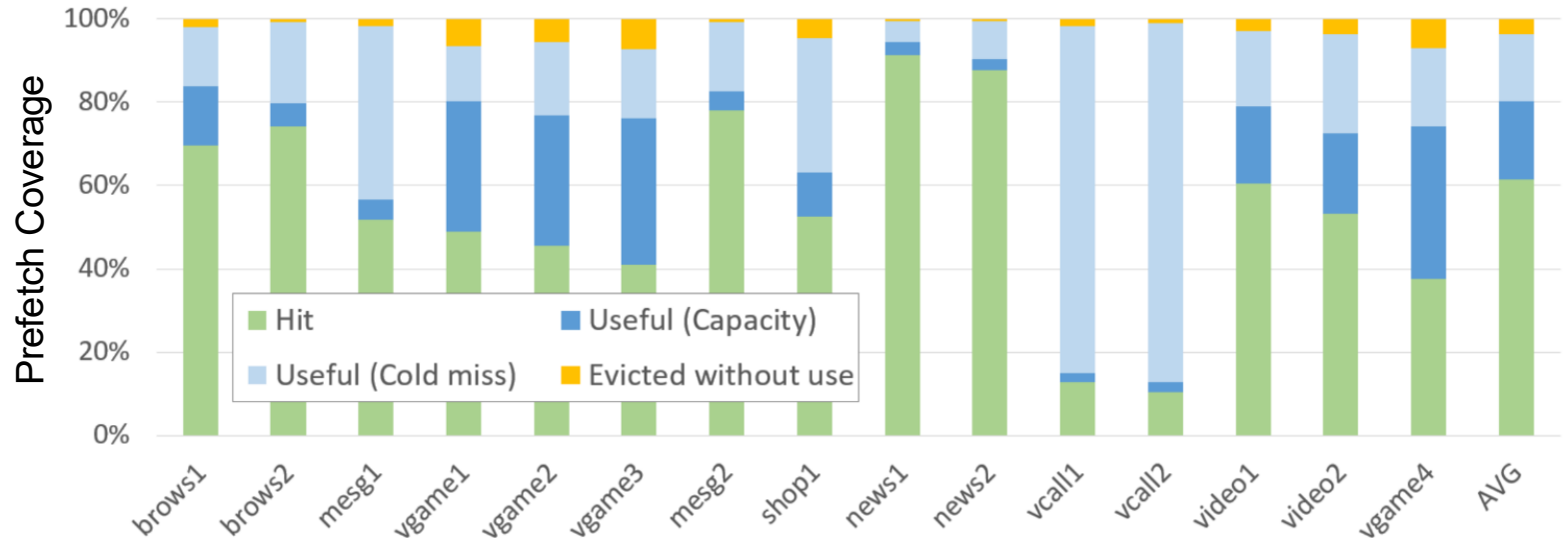
- Record and Replay: Record and Replay the last 50 HBs
- Hierarchical-Prefetching* (ASPLOS'25): Record and Replay last 50 HBs based on trigger PCs
- I-Spy* (MICRO'20): Software prefetching without the dynamic instruction overheads
- EFetch* (PACT'14): Prefetch down call chains based on software hints

All adapted to prefetch into L2 and without instruction overhead for fairness

HUAWEI

Evaluation





HUAWEI

Conclusion & Future Work

- DEER is a software-hardware co-design prefetcher that predicts likely instruction paths far into the future
- Offline Record: Push the burden of recording/predicting deep instruction paths to offline software analysis by leveraging ARM's BRBE profiling extension
- Online Replay: Replay the instruction cache lines in the deep instruction paths provided from offline recording
- Outperforms hardware-centric techniques by covering cold misses
- Outperforms software-centric techniques by reducing dynamic instruction overheads

- Particularly useful for mobile systems which may contains an abundance of cold misses (TODO: need to measure this)
  - > Application startups
  - > Context Switching
  - > Thread Migrations

HUAWEI

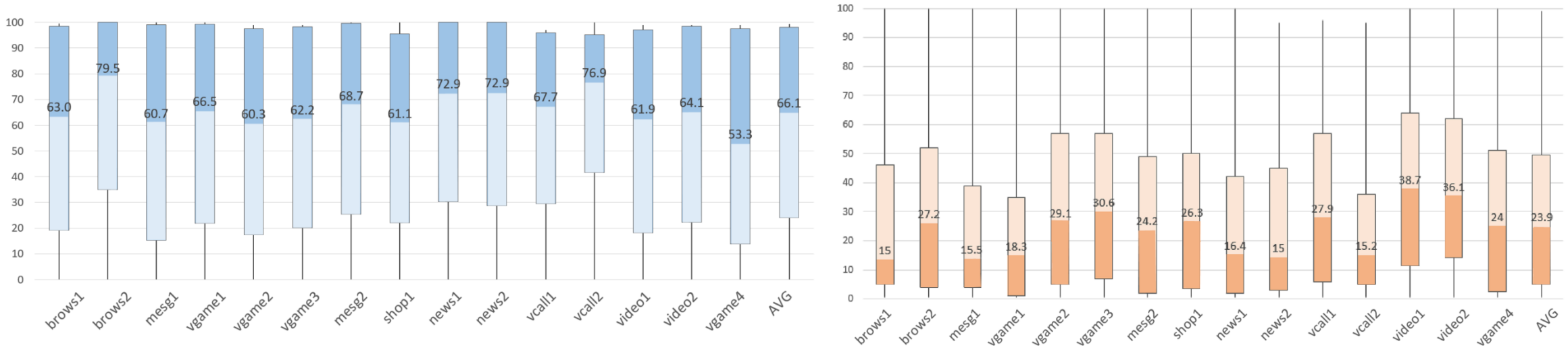
Thank you.

Bring digital to every person, home and organization for a fully connected, intelligent world.

arXiv paper: https://arxiv.org/abs/2504.20387

HUAWEI

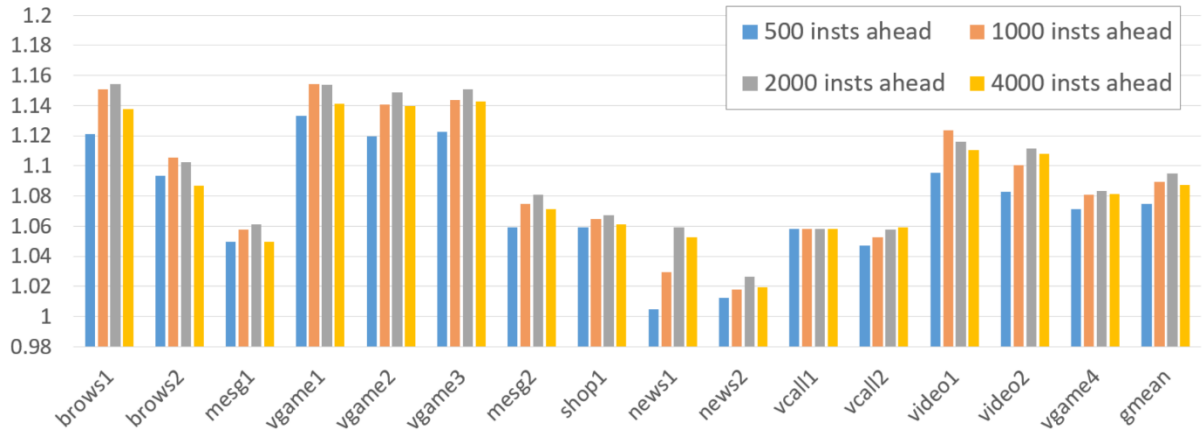
Evaluation



DEER accuracy of predicted upcoming cachelines vs Hierarchical-Prefetching* accuracy

HUAWEI

DEER (Deep Runahead Prefetcher): Upper Bound Gains



Upper bound ipc speedup by an oracle DEER prefetcher

HUAWEI