HPCA-2014

Improving DRAM Performance by Parallelizing Refreshes with Accesses

Kevin Chang

Donghyuk Lee, Zeshan Chishti, Alaa Alameldeen, Chris Wilkerson, Yoongu Kim, Onur Mutlu

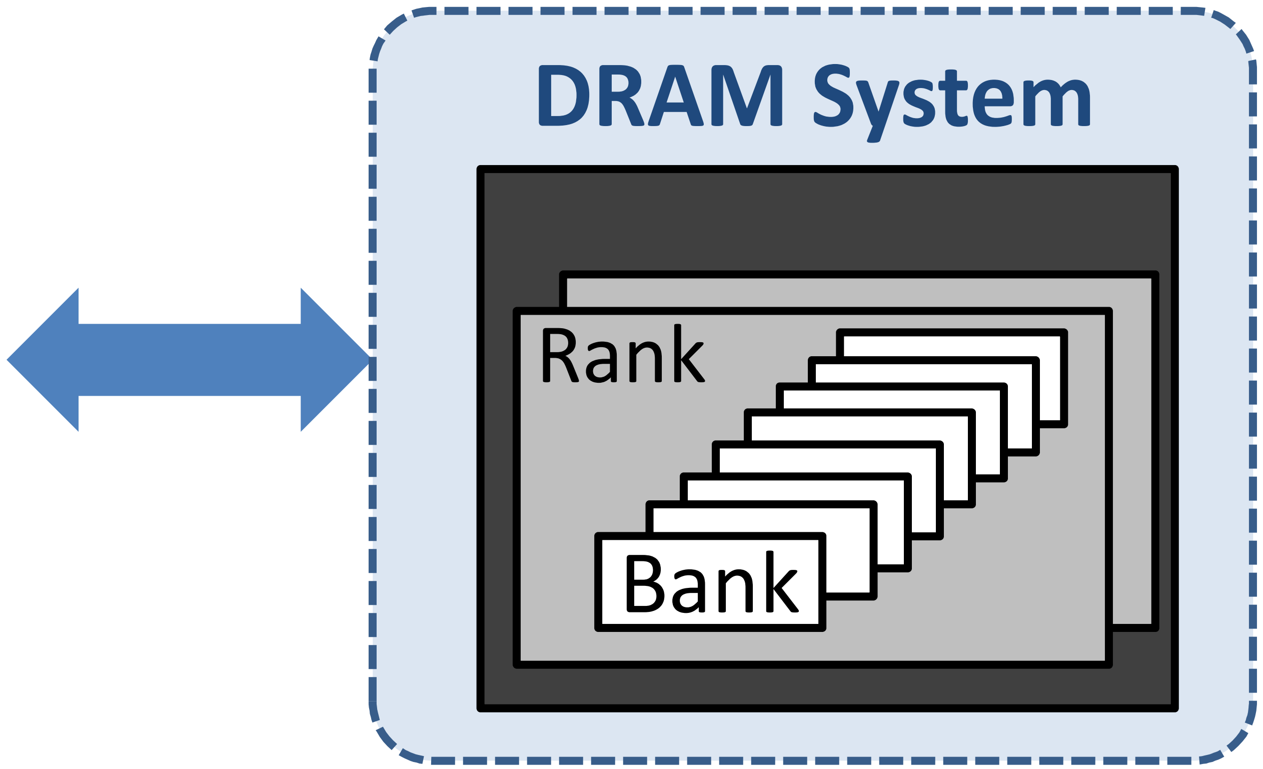
Paper Overview

- Issue: **Refresh in Modern DRAM(eg: DDR3, LPDDR2) : All Bank or Per Bank: simple and straight forward method**
  - Impacting system performance ← and energy efficiency ←
  - Will be prominent in near future as DRAM density increases
- Objective: Serve (more)memory accesses with refreshes to reduce latency on demand requests
- Method:
  - 1. Enable more parallelization between refreshes and accesses across different banks with new per-bank refresh scheduling algorithms
  - 2. Enable serving accesses concurrently with refreshes in the same bank by Using DRAM subarrays
- Result: Improve system performance and energy efficiency for a wide variety of different workloads and DRAM densities
  - 20.2% and 9.0% for 8-core systems using 32Gb DRAM
  - Very close to the ideal scheme without refreshes

Outline

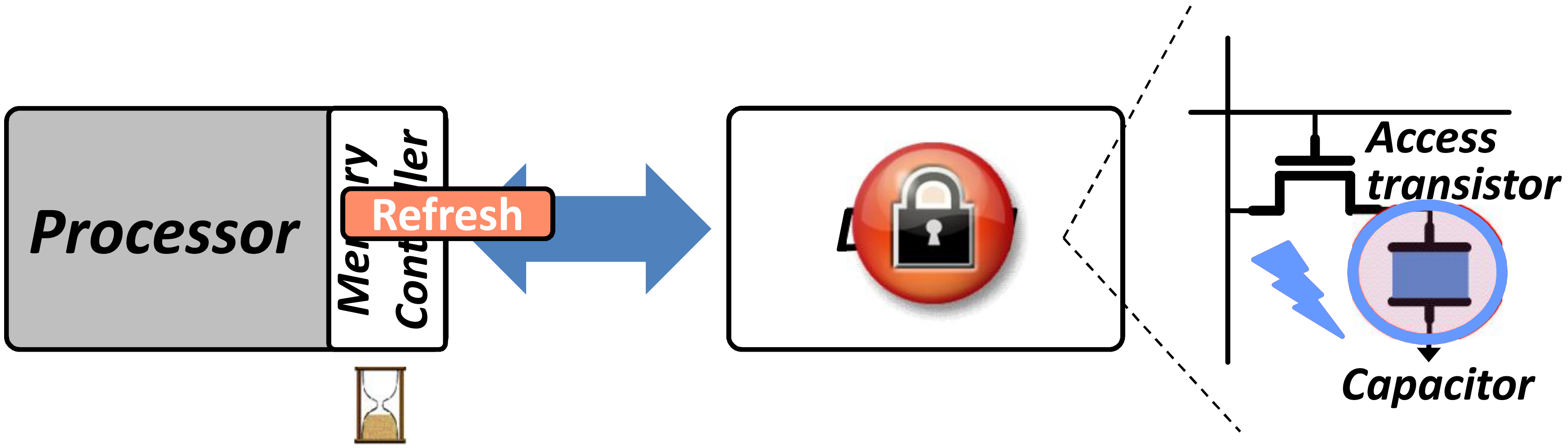
- **DRAM and Refresh Background**
- Motivation and Key Ideas
- Mechanisms
- Results

DRAM System Organization



- Banks can serve multiple requests in parallel

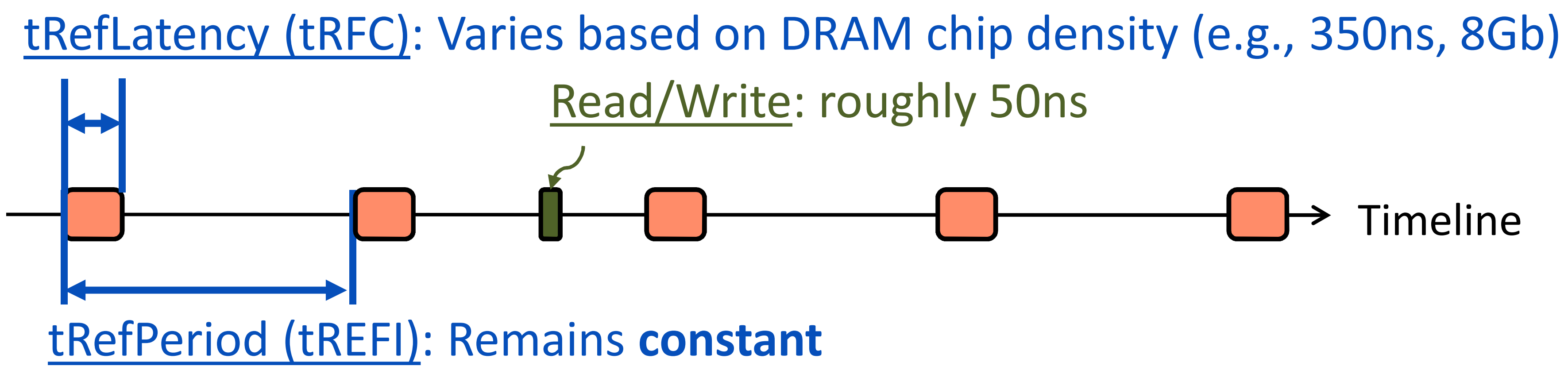
During Refresh no memory Access



Refresh delays requests by 100s of ns

DRAM Refresh Frequency

- DRAM (JEDEC) standard requires memory controllers to send **periodic refreshes** to DRAM


tRefLatency (tRFC): Varies based on DRAM chip density (e.g., 350ns, 8Gb)
Read/Write: roughly 50ns
Timeline
tRefPeriod (tREFI): Remains **constant**

Increasing Performance Impact

- DRAM is unavailable to serve requests for $\frac{tRefLatency}{tRefPeriod}$ of time

- **6.7%** for today's 4Gb DRAM

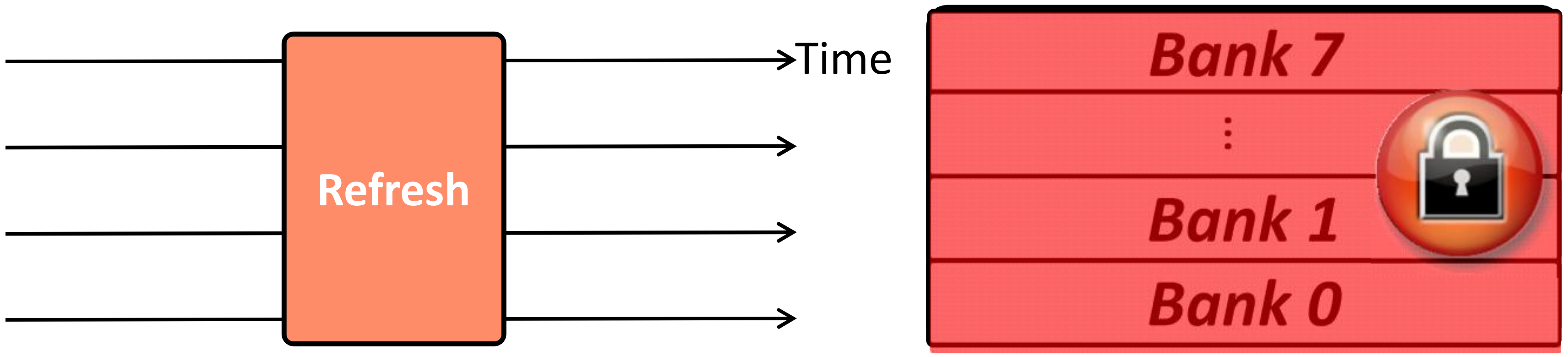
- Unavailability increases with higher density due to higher *tRefLatency*
  - **23% / 41%** for **future 32Gb / 64Gb DRAM**

Outline

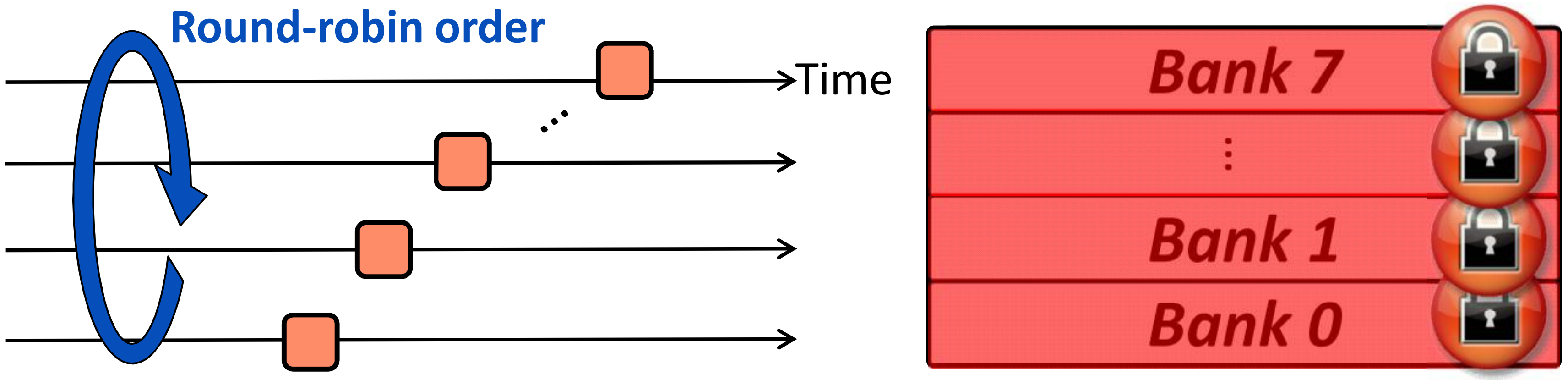
- DRAM and Refresh Background
- **Motivation and Key Ideas**
- Mechanisms
- Results

Refresh in Modern DRAM

All-bank refresh in commodity DRAM (DDRx,LPDDRx)



Per-bank refresh in mobile DRAM (LPDDRx)



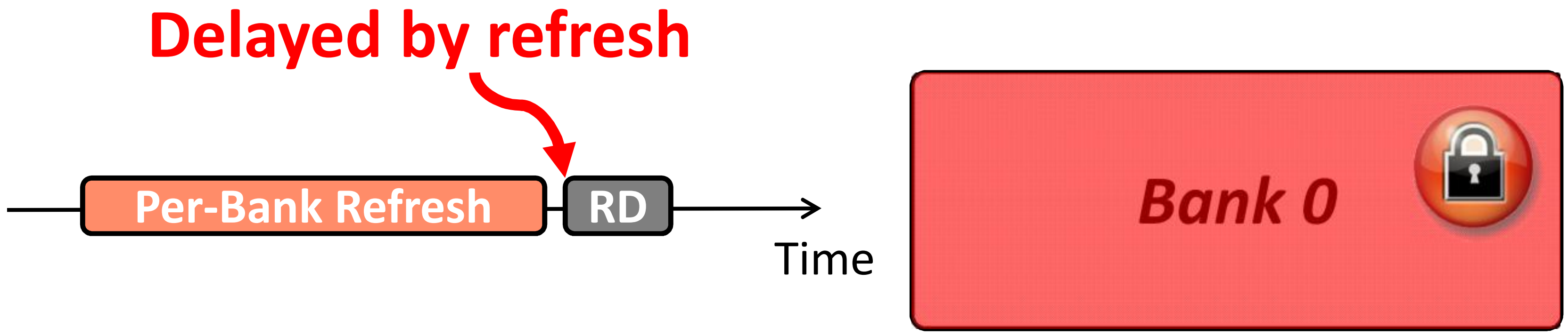
Shortcomings of Per-Bank Refresh

- Problem 1: Refreshes to different banks are scheduled in a strict round-robin order
  - The static ordering is hardwired into DRAM chips
  - Refreshes busy banks with many queued requests when other banks are idle

- Key idea: Schedule per-bank refreshes to idle banks opportunistically in a dynamic order

Shortcomings of Per-Bank Refresh

- Problem 2: Banks that are being refreshed cannot concurrently serve memory requests



Key idea: Exploit **subarrays** within a bank to parallelize refreshes and accesses across **subarrays**

Outline

- Motivation and Key Ideas
- DRAM and Refresh Background
- Mechanisms
  - 1. Dynamic Access-Refresh Parallelization (DARP)
  - 2. Subarray Access-Refresh Parallelization (SARP)
- Results

DARP

- **Dynamic Access-Refresh Parallelization (DARP)**
  - An improved scheduling policy for per-bank refreshes
  - Exploits refresh scheduling flexibility in DDR DRAM

- Component 1: **Out-of-order per-bank refresh**
  - Avoids poor static scheduling decisions

  By Dynamically issues per-bank refreshes to idle banks

- Component 2: **Write-Refresh Parallelization**
  - Avoids refresh interference on latency-critical reads

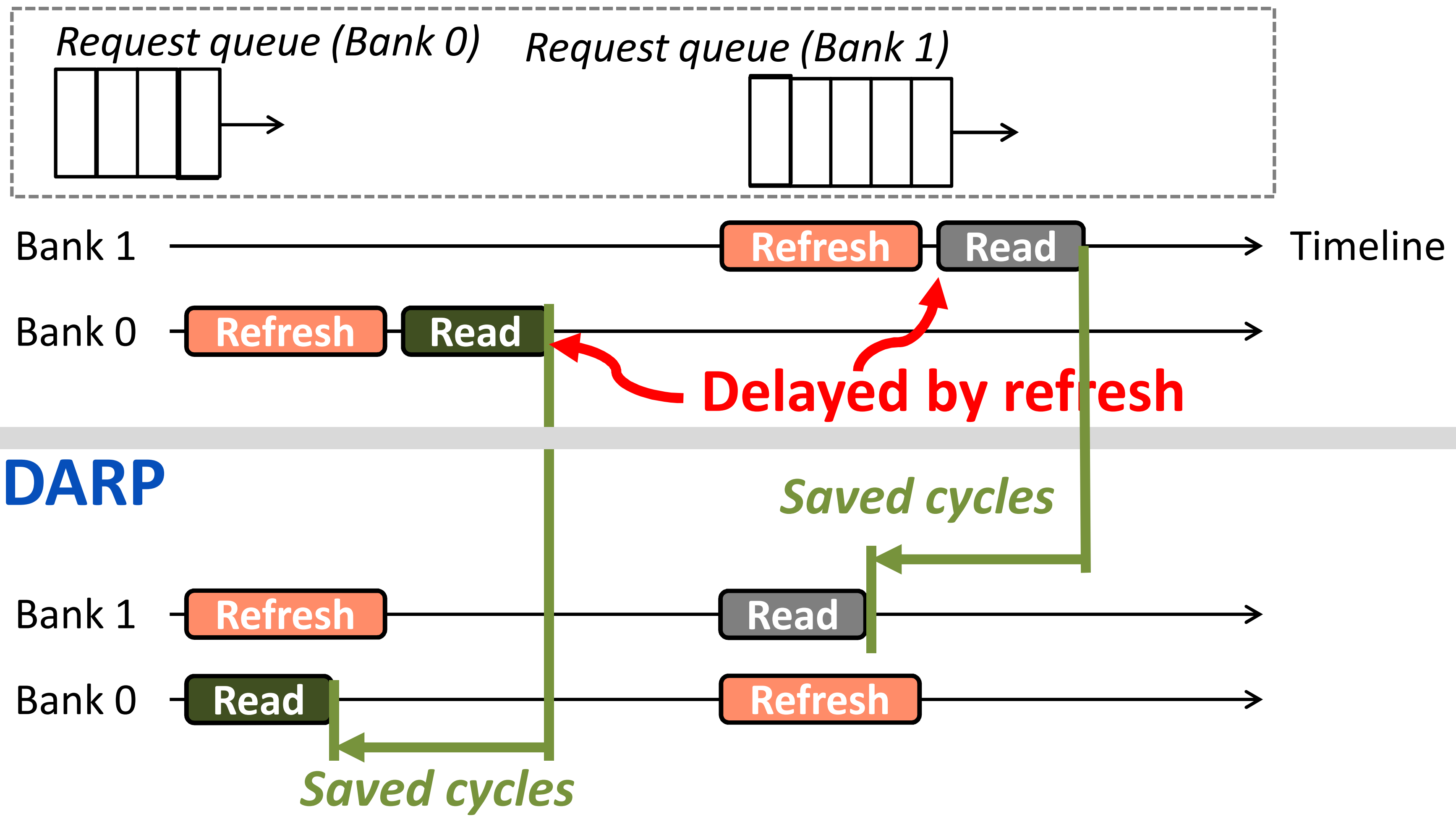
  By Hiding(parallelize) refreshes with **a batch of writes**

1) Out-of-Order Per-Bank Refresh

- **Dynamic scheduling policy** that prioritizes refreshes to idle banks
- **Memory controllers** decide which bank to refresh

1) Out-of-Order Per-Bank Refresh

Baseline: Round robin

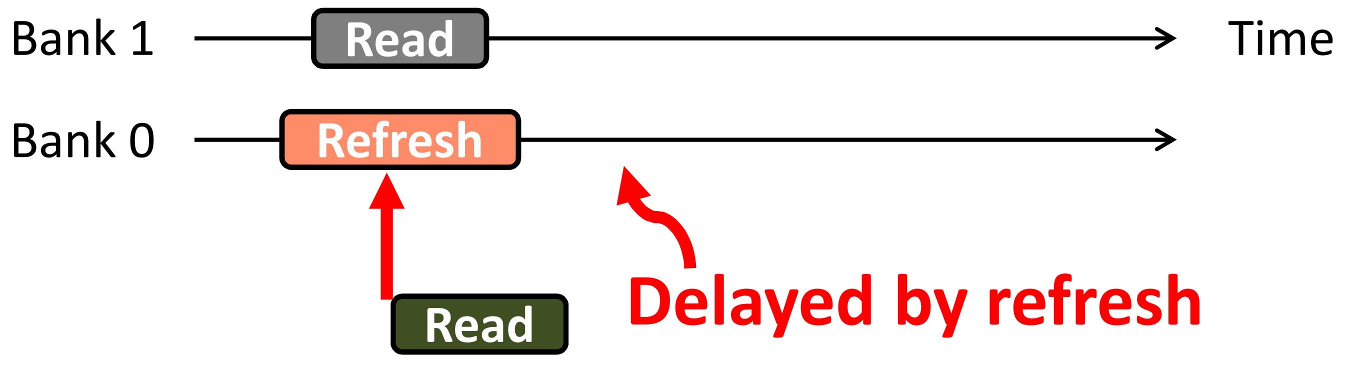


Outline

- Motivation and Key Ideas
- DRAM and Refresh Background
- Mechanisms
  - 1. Dynamic Access-Refresh Parallelization (DARP)
    - 1) Out-of-Order Per-Bank Refresh
    - **2) Write-Refresh Parallelization**
  - 2. Subarray Access-Refresh Parallelization (SARP)
- Results

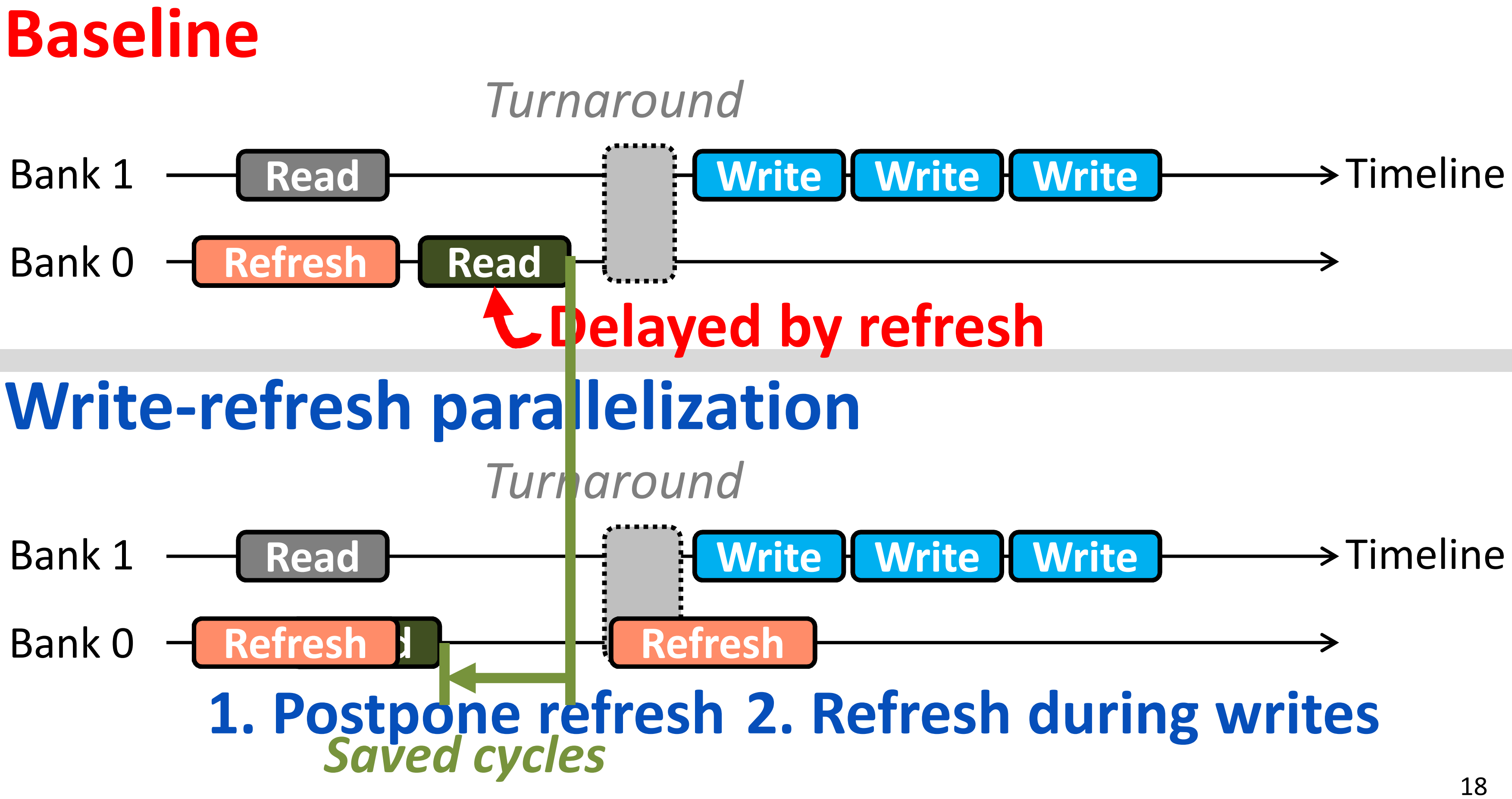
Refresh Interference on Upcoming Requests

- Problem: A refresh may collide with an upcoming request in the near future



2) Write-Refresh Parallelization

- Proactively schedules refreshes when banks are serving **write batches**



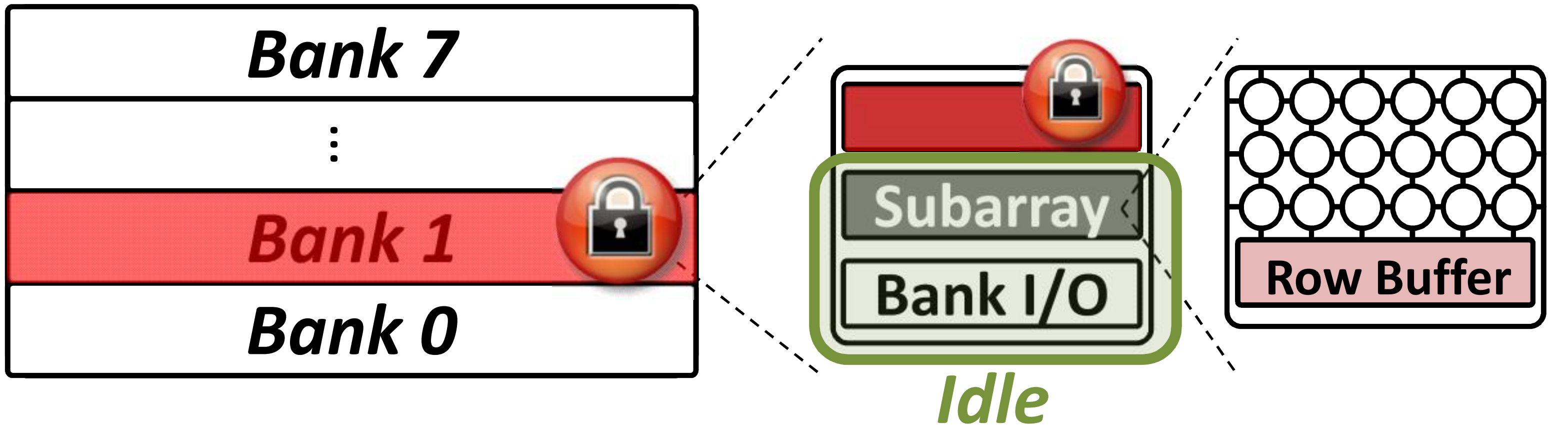
Outline

- Motivation and Key Ideas
- DRAM and Refresh Background
- Mechanisms
  - 1. Dynamic Access-Refresh Parallelization (DARP)
  - 2. Subarray Access-Refresh Parallelization (SARP)
- Results

SARP

Observations:

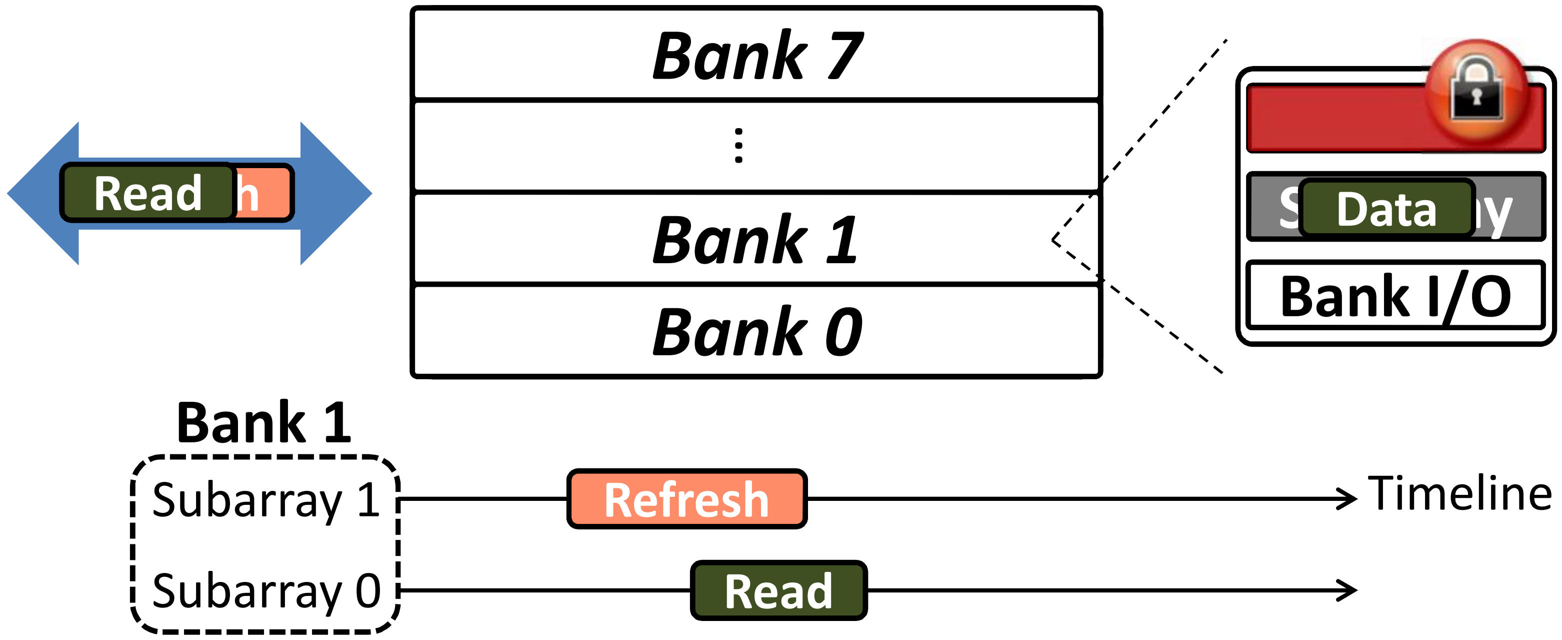
1. A bank is further divided into **subarrays**
   - Each has its own row buffer to perform refresh operations



2. Some **subarrays** and **bank I/O** remain completely **idle** during refresh

SARP

- <u>**Subarray Access-Refresh Parallelization (SARP)**</u>**:**
  - Parallelizes refreshes and accesses **within a bank**



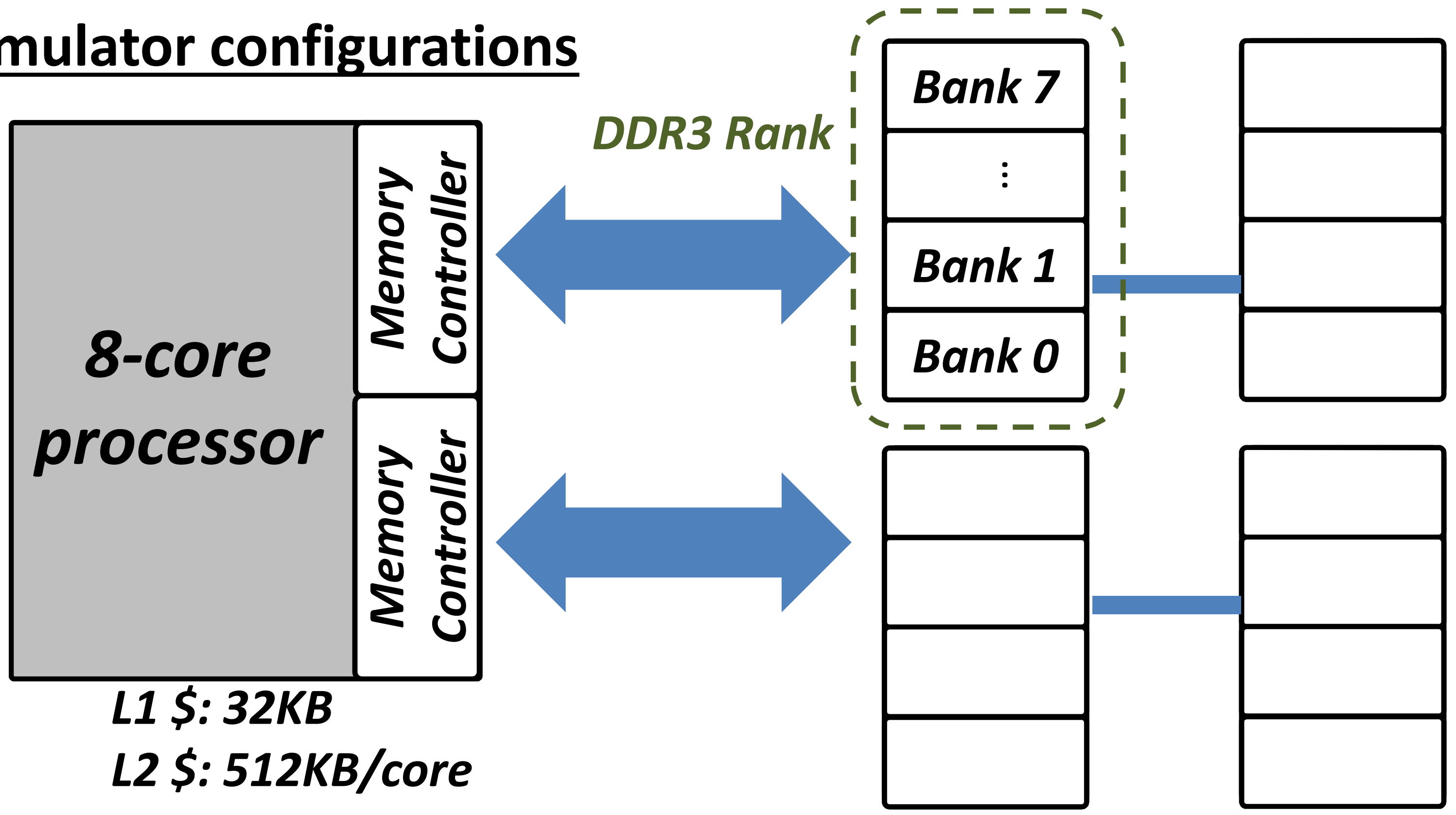
Very modest DRAM modifications: 0.71% die area overhead

Outline

- Motivation and Key Ideas
- DRAM and Refresh Background
- Mechanisms
- **Results**

Methodology

Simulator configurations

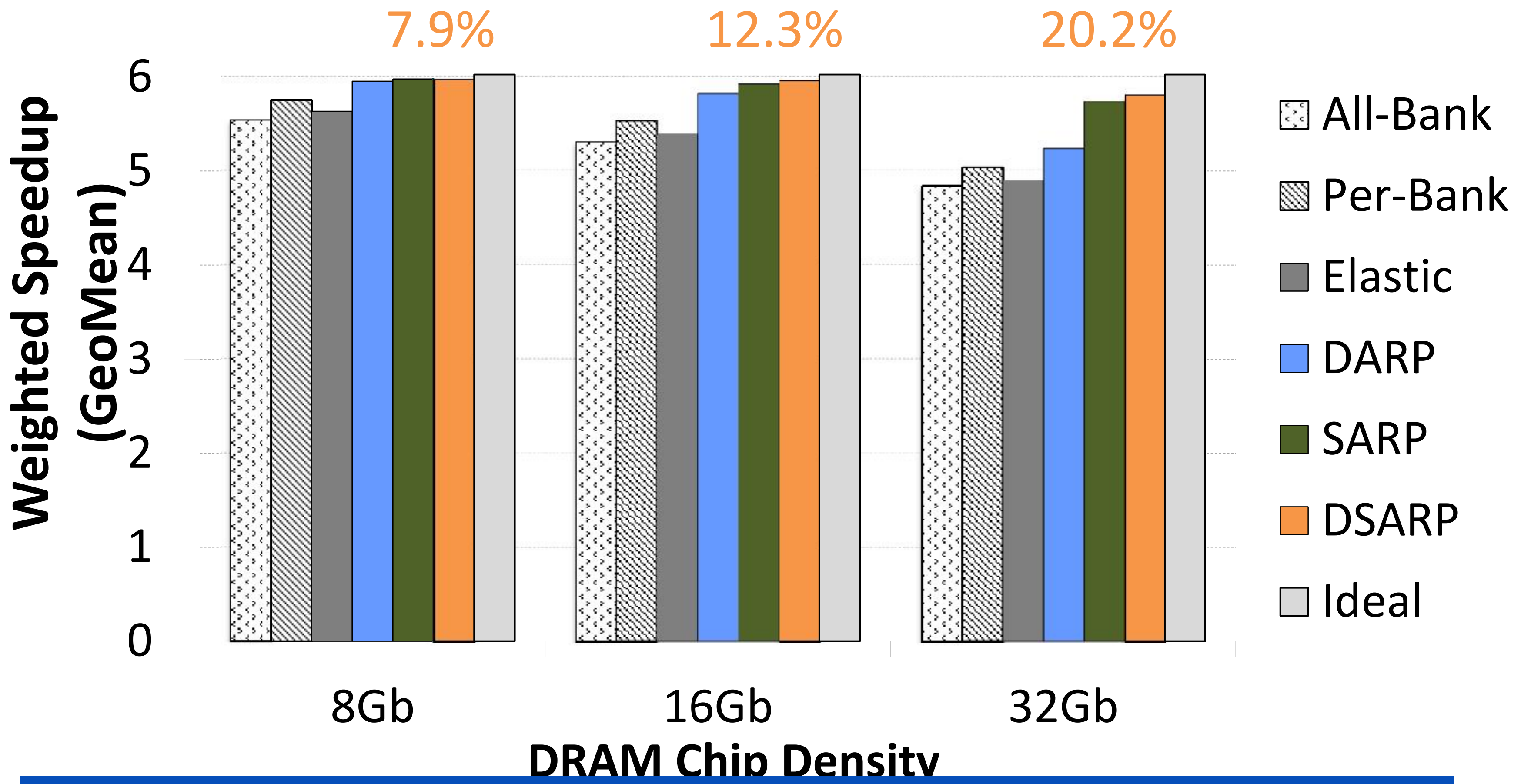


- **100 workloads**: SPEC CPU2006, STREAM, TPC-C/H, random access
- *Weighted speedup*

Comparison Points

- **All-bank refresh** [DDR3, LPDDR3, …]
- **Per-bank refresh** [LPDDR3]
- **Elastic refresh** [Stuecheli et al., MICRO '10]:
  - Postpones refreshes by a time delay based on the predicted rank idle time to avoid interference on memory requests
  - Proposed to schedule all-bank refreshes without exploiting per-bank refreshes
  - Cannot parallelize refreshes and accesses within a rank
- **Ideal (no refresh)**

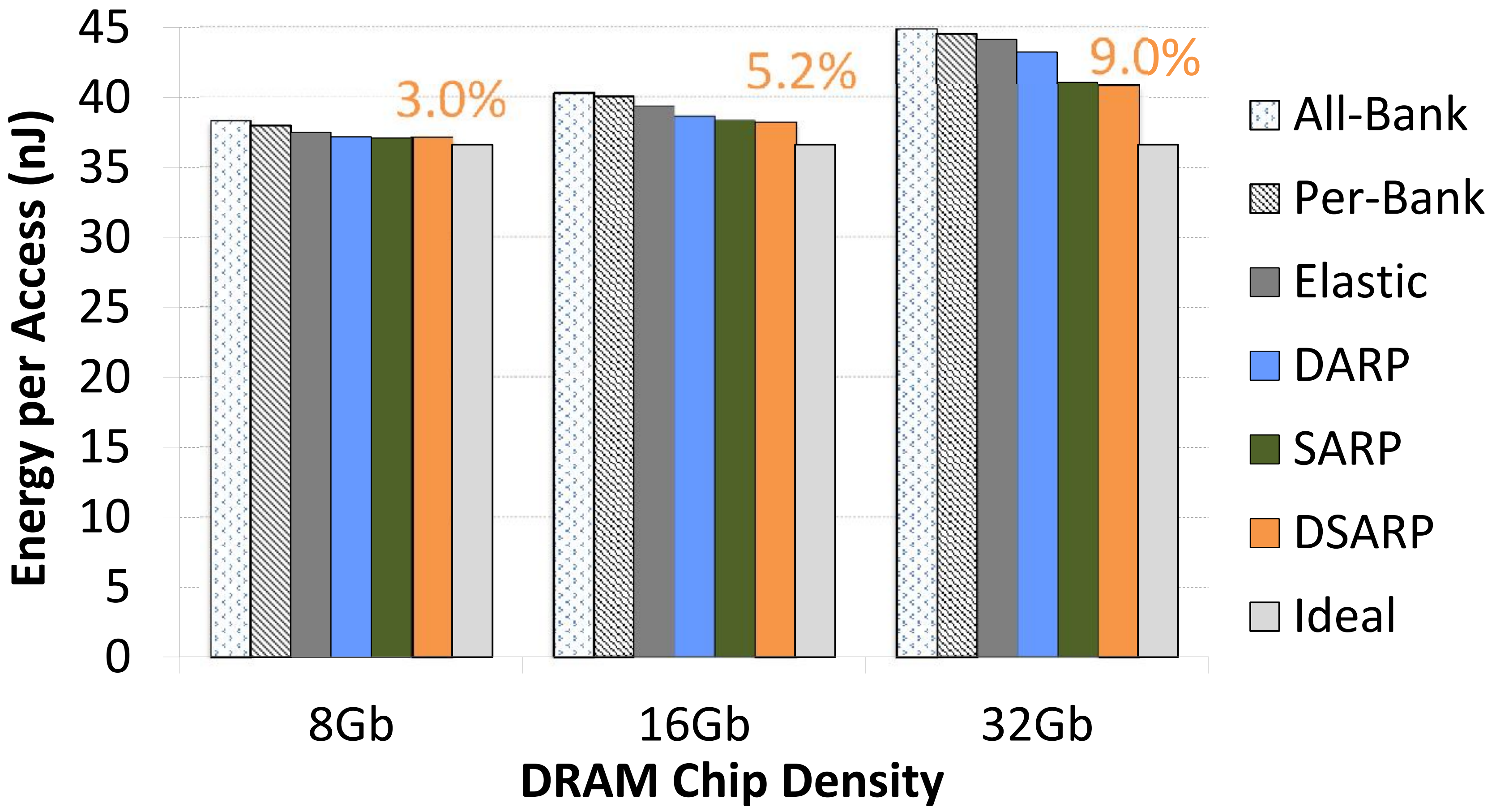
System Performance



1. Both DARP & SARP provide performance gains and combining them (DSARP)
.

*2. Consistent system performance improvement across DRAM densities (within **0.9%, 1.2%, and 3.8%** of ideal)*

Energy Efficiency



Consistent reduction on energy consumption

Other Results and Discussion in the Paper

- Detailed multi-core results and analysis
- Result breakdown based on memory intensity
- Sensitivity results on number of cores, subarray counts, refresh interval length, and DRAM parameters
- Comparisons to DDR4 fine granularity refresh

Improving DRAM Performance by Parallelizing Refreshes with Accesses