Fat Caches for Scale-Out Servers

Fat Caches for Scale-Out Servers*

Stavros Volos[†]
Microsoft Research

Djordje Jevdjic[†]
University of Washington

Babak Falsafi
EcoCloud, EPFL

Boris Grot
University of Edinburgh

Abstract

Emerging scale-out servers are characterized by massive memory footprints and bandwidth requirements. On-chip stacked DRAM caches have been proposed to provide the required bandwidth for manycore servers through caching of secondary data working sets. However, the disparity between provided capacity and working set sizes precludes their effective deployment in servers, calling for high-capacity cache architectures. High-capacity caches—enabled by the emergence of high-bandwidth memory technologies—exhibit high spatio-temporal locality due to coarse-grained access patterns and long cache residency periods stemming from skewed dataset access distributions. The observed spatio-temporal behavior favors a page-based organization that naturally exploits spatial locality while minimizing tag storage requirements and enabling a practical in-SRAM tag array architecture. By storing tags in SRAM, caches avoid the complexity of in-DRAM metadata found in state-of-the-art DRAM caches.

1. Introduction

Scale-out datacenters host a variety of data-intensive services, such as search and social connectivity. To concurrently support billions of users, latency-sensitive online services and analytic engines creating user-specific content (e.g., advertisements and recommendations) rely on large amounts of memory to minimize dataset access latency. The ever-growing popularity of the in-memory computing paradigm—which will be further broadened by the emergence of non-volatile memory—leads to datacenter deployments in which memory accounts for a big share of the datacenter's total cost of ownership (TCO) [1].

Optimizing for datacenter's TCO calls for customized architectures that maximize compute density. Following a considerable amount of research, identifying the requirements of scale-out workloads, and indicating that these workloads benefit from thread-level parallelism and fast access to multi-megabyte instruction footprints [2] [3], industry has started employing specialized manycore processors with modestly-sized last-level caches (e.g., Cavium ThunderX, EZchip Tile-MX) due to the substantial performance and TCO advantages offered by specialization.

Memory systems in scale-out servers are of paramount importance as they need to sustain the vast bandwidth demands of manycore CMPs [3] [4]. Recent advances in *on-chip stacked DRAM* technology [5] eliminate the bandwidth bottleneck that plagues conventional DRAM. As this technology is capacity-limited due to thermal constraints, prior research advocates for using it as a cache to provide access to secondary data working sets [4] [6] [7] [8].

Our analysis shows that on-chip stacked DRAM caches are unattractive for scale-out servers. We find that memory accesses follow power-law distributions so that a hot portion of memory (~10%) accounts for the majority of accesses (65–95%). Thus, while the vast working sets of scale-out workloads are amenable to caching, *high-capacity* caches (10s of GB) are required given main memory sizes trending toward 100s of GB. The required cache capacities greatly exceed those of *low-capacity* caches, including on-chip stacked DRAM caches.

This work seeks to develop a scalable, high-capacity, and high-bandwidth memory system for scale-out servers by leveraging emerging *high-bandwidth memory modules* as a high-capacity cache. High-bandwidth interconnect technologies allow for connecting the processor to multiple high-bandwidth memory modules via a silicon interposer (e.g., Hynix HBM) forming an *on-package* cache, or high-speed serial links (e.g., Micron HMC) forming an *off-package* cache.

In contrast to prior stacked DRAM cache proposals, which advocate for block-based [7] [8] and sector-based organizations [4] [6], we find that page-based organizations are favored in scale-out servers. High-capacity caches—effective in capturing the secondary data working sets of scale-out workloads—uncover significant spatio-temporal locality across dataset objects due to long cache residency periods. The improved spatio-temporal locality allows for employing a page-based cache organization, thereby minimizing tag storage requirements and enabling a practical in-SRAM tag array architecture, which can be implemented in the logic die of the high-bandwidth memory modules. This design offers fundamental complexity advantages over state-of-the-art DRAM caches, which suffer from high tag/metadata overheads that mandate in-DRAM storage.

[†] Most of the work was done while the authors were at EPFL.

* The work was partially funded by the NanoTera project "YINS."

Table 1. Requirements of one scale-out server.

Year	Processor		Memory System	
	Cores	Bandwidth	Bandwidth	Capacity
2015	96	115 GB/s	288 GB/s	384 GB
2018	180	216 GB/s	540 GB/s	720 GB
2021	320	384 GB/s	960 GB/s	1280 GB

2. Emerging Scale-Out Servers and DRAM Technologies

In this section, we examine the memory requirements of emerging scale-out servers and also review the features of emerging DRAM technologies.

2.1. Scale-Out Server Requirements

Processor and system vendors resort to manycore processors (e.g., Cavium ThunderX) to boost server throughput and rely on buffer-on-board chips (e.g., Cisco's *extended memory technology* [9]) to increase memory capacity. In doing so, datacenter operators can deploy fewer servers for the same throughput requirements and dataset size, thus lowering TCO significantly [9] [10].

We quantify the memory bandwidth and capacity requirements of emerging scale-out servers for various manufacturing technologies in Table 1. Our configuration maximizes throughput by integrating maximum number of cores for a given die area and power budget of 250–280 mm^2 and 95–115 Watt. The modeled organization resembles that of manycore servers, such as Cavium ThunderX.

Bandwidth. We measure processor's off-chip bandwidth demands by scaling per-core bandwidth consumption with the total number of cores. We measure per-core bandwidth by simulating a 16-core server finding that per-core bandwidth ranges from 0.4GB/s to 1.2GB/s. Peak bandwidth demands are 115GB/s (2015), 216GB/s (2018), and 384GB/s (2021).

High bandwidth utilization levels can adversely impact end-to-end memory latency due to heavy contention on memory resources. As performance of scale-out services is characterized by tail latencies, memory latency and queuing delays must be minimized. Thus, system designers over-provision memory bandwidth to ensure low utilization (<40%) and avoid queuing [2]. As such, memory systems need to supply 288GB/s (2015), 540GB/s (2018), and 960GB/s (2021). Such requirements exceed the capabilities of conventional DRAM systems by 5.5–7.5x.

Capacity. We estimate required memory capacity by examining various system deployments. Today, data analytic engines are provisioned with 2–8GB per core (Cloudera), web search engines deploy 64GB for 16 cores (Microsoft Bing) while web and streaming servers require 1–2GB per core [2]. With the emergence of extended memory technology and non-volatile memory, we anticipate that datacenter operators will continue deploying 4GB of per-core memory cost-effectively, resulting in deployment of several 100s of GB of memory per server.

2.2. Emerging DRAM Technologies

Stacked DRAM can provide an order of magnitude higher (memory core) bandwidth than conventional DRAM due to dense through-silicon vias. It also offers low latency and low DRAM energy due to reduced wire spans and smaller page sizes. However, existing deployment options for stacked DRAM fail to satisfy the joint capacity, bandwidth, and power requirements mandated by scale-out servers. Next, we review the deployment options for stacked DRAM and their respective limitations.

On-Chip and On-Package Stacked DRAM. Through-silicon vias provide high-bandwidth connectivity between the processor and on-chip stacked DRAM. Thermal constraints, however, limit the number of DRAM stacks that can be integrated on top of the processor, confining On-Chip Stacked DRAM to sizes that are two-to-three orders of magnitude smaller than the memory capacity demands of servers. Similarly, the high cost of big packages and area-intensive silicon interposers limit the number of stacked DRAM modules in On-Package Stacked DRAM systems. When combined with the thermally-constrained capacity of a few GB per module, an On-Package DRAM solution fails to provide the requisite memory capacity for servers.

Off-Package Stacked DRAM. High-speed serial interfaces can break the bandwidth wall by connecting the processor to multiple Off-Package Stacked DRAM modules. The high signal integrity of serial interfaces allows for achieving an order of magnitude higher data rates than DDR with the same number of pins.

Although off-package stacked DRAM systems deliver much greater memory capacity than on-chip and on-package stacked DRAM systems, there are two main factors that prevent such systems from replacing conventional DRAM. First, serial channels impose high idle power as keep-alive packets must be sent at frequent intervals to maintain lane alignment across the channel's lanes. Second, thermal constraints limit the number of stacked layers per module and necessitate a blade-level network of these modules for a big-memory server. Such a network comes at the cost of high idle power consumption due to the use of many serial links resulting from a multi-hop chip-to-chip network.

2.3. State-of-the-art DRAM Caches

Given the disparity between memory capacity requirements and the capacity provided by emerging DRAM technologies, most proposals advocate employing stacked

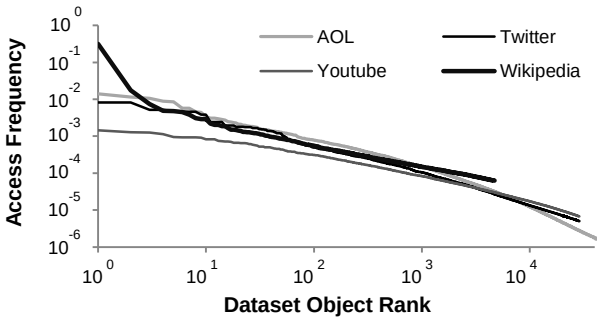


Figure 1. Dataset object popularity exhibits power-law distribution. Please note that power-law relationships show linear trends in log-log scale.

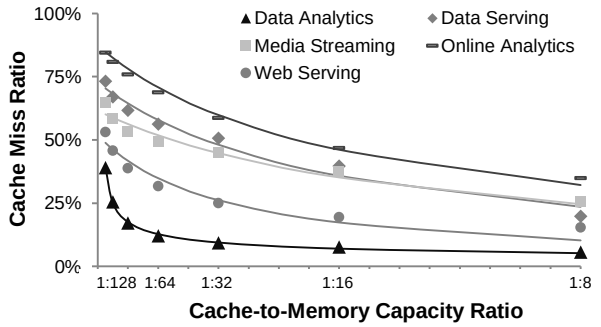


Figure 2. Miss ratio for various Cache-to-Memory Capacity Ratios. Lines denote x-shifted power-law fitting curves.

DRAM as a cache to filter accesses to main memory. State-of-the-art cache proposals leveraging mainly On-Chip Stacked DRAM have to contend with relatively high miss rates due to its limited capacity. As a result, they are primarily optimized for low cache-memory bandwidth utilization through block-based organizations [7] [8], sector-based footprint-predicting organizations [4] [6], and address-correlated filter-based caching mechanisms [11].

Unfortunately, such organizations come with high tag/metadata overhead and high design complexity, making such cache designs impractical. For instance, state-of-the-art block-based and footprint-predicting caches require 4GB and 200MB of tags, respectively, for a capacity of 32GB. Due to the prohibitive tag array overhead, recent proposals implement the tag array in DRAM [6] [7] [8]. In-DRAM tag arrays, however, require substantial engineering effort, making state-of-the-art caches less attractive. In addition, footprint-predicting caches [4] [6] rely on instruction-based prediction. However, the program counter of an instruction is not available in the memory hierarchy, thus requiring the core-to-cache transfer of the program counter for all memory references, further increasing design complexity.

3. Memory Access Characterization of Scale-Out Servers

High-bandwidth memory modules are an ideal building block for a high-capacity high-bandwidth cache. However, state-of-the-art DRAM caches are hindered by the need to keep metadata in DRAM. In this section, we study the application characteristics that enable architecting an effective, practical, and scalable cache.

3.1. Temporal Characterization

We examine the memory access distribution of scale-out applications by looking at the characteristics of the dominant types of memory accesses.

Dataset accesses. We examine the dataset object popularity (i.e., how frequent a dataset object is accessed) of search query terms (AOL), tweets (Twitter), videos (Youtube), and web pages (Wikipedia) based on publicly available data. Figure 1 plots the probability for a dataset object to be referenced as a function of popularity, showing that the dataset object popularity is highly skewed with a small set of dataset objects (10–20%) contributing to most of the dataset object accesses (65–80%). For instance, a small fraction of users and their pictures account for most of the user traffic in picture sharing services, such as Flickr. Due to power-law popularity distributions, dataset accesses in data stores, object caching systems, streaming servers, web search engines, and web servers exhibit power-law distributions.

Accesses to dynamically allocated memory. Server applications frequently access dynamically allocated memory with high temporal reuse. Examples include:

- Server applications utilize software caches to keep a set of hot objects—e.g., rows in data stores and compiled script code in web servers. As they host dataset-relevant data/metadata, the distributions of their accesses will follow those of the datasets.
- Server applications and operating systems employ various data structures per client/network connection, such as buffers for media packets in streaming servers and OS data structures storing TCP/IP state. The large number of concurrent connections in manycore CMPs results in a footprint that dwarfs on-chip cache capacity. The reuse of these structures is high as they are accessed multiple times during a connection.

The skew in object popularity and temporal reuse of dynamically allocated memory is expected to be mirrored in the memory access distribution. To confirm this, we examine the memory access distribution of a simulated 16-core scale-out server. To estimate the hot memory footprint

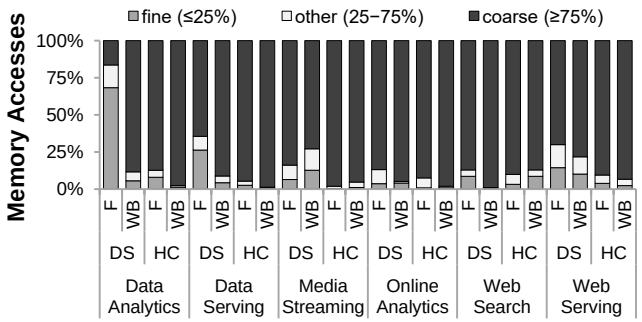


Figure 3. Granularity at which page-sized lines are fetched (F) from and written back (WB) to DRAM for Die-Stacked (DS) and high-capacity cache (HC) of 1:128 and 1:8 Cache-to-Memory Capacity Ratio, respectively.

of scale-out applications, we employ a state-of-the-art DRAM cache and measure its miss ratio for various capacities [8].

Figure 2 plots the cache miss ratio for various Cache-to-Memory Capacity Ratios. The markers denote measurements while contiguous lines show x-shifted power-law fitted curves. The figure shows that memory accesses are skewed so that 6.25–12.5% of the memory footprint accounts for 65–95% of total accesses. The figure confirms that existing low-capacity caches (left points), such as on-board SRAM caches (IBM Centaur), on-package eDRAM caches, and on-chip stacked DRAM caches cannot exploit temporal locality in scale-out servers. In extreme cases, such as Data Serving and Online Analytics, on-chip stacked DRAM caches are bandwidth-constrained with less than 40% of memory accesses filtered. We thus conclude that the combination of poor cache performance and technological complexity of die stacking limits the usefulness of on-chip stacked DRAM caches in servers.

3.2. Spatial Characterization

Scale-out applications often operate on bulk objects (e.g., database rows), thus exhibiting a high incidence of coarse-grained accesses [12]. To allow for retrieving an object in sub-linear time, objects are pinpointed through pointer-intensive indexing structures, such as hash tables and trees. For instance, data stores and object caching systems use a hash table to retrieve data objects. While objects are accessed at coarse granularity, finding them requires performing a sequence of pointer dereferences. Thus, a non-negligible fraction of accesses are fine-grained [12].

We examine the granularity at which high-capacity (HC) caches access memory by measuring the access density at which page-sized lines are fetched from and written back to memory in Figure 3. We define *page access density* as the fraction of 64-byte blocks within a page accessed between the page's first access and the page's eviction from the cache. We use a page of 2KB as it reduces the tag array size significantly with limited tolerance for overfetch. Thus, fine-grained pages have low access density (up to 8 unique cache blocks accessed) while coarse-grained pages have high access density (at least 24 unique cache blocks accessed). For comparison, we include a low-capacity cache, labeled as Die-Stacked (DS).

We find that Die-Stacked exhibits bimodal memory access behavior—i.e., fine-grained and coarse-grained accesses account for 21% and 68% of accesses, respectively. While coarse-grained accesses are prevalent, the frequent incidence of fine-grained accesses must also be accommodated effectively. Due to the limited capacity of on-chip stacked DRAM caches, pointer-containing pages show low temporal reuse and are frequently evicted. To avoid massive bandwidth waste in accesses to such pages, state-of-the-art DRAM caches rely on block-based or sector-based footprint-predicting organizations that are bandwidth-frugal, but carry a high metadata storage cost.

In contrast, high-capacity caches exhibit coarse-grained memory access behavior—i.e., 93% of all accesses. This behavior is attributed to two phenomena. First, the lifetime of pages in the cache is on the order of 10s to 100s of milliseconds. Thus, pages containing a collection of fine-grained objects (e.g., hash bucket headers) can enjoy spatial locality uncovered through long cache residency times, stemming from skewed access distributions. Second, low-access-density pages containing pointer-intensive indexing structures with good temporal reuse (e.g., intermediate tree nodes) are preserved across accesses.

3.3. Summary

Our study demonstrates that high-capacity caches are needed to capture the skewed memory access distributions of servers. We also find that the improved spatio-temporal behavior of high-capacity caches offers an opportunity to use a simple page-based organization, thus avoiding the storage and complexity overheads associated with state-of-the-art stacked DRAM caches.

4. Memory System Architecture for Scale-Out Servers

We present MeSSOS, a *Memory System* architecture for *Scale-out Servers* that provides the required bandwidth and capacity for a scale-out server. High bandwidth is delivered through caching of data working sets in a high-capacity *Scale-Out Cache* (soCache), which consists of multiple off-package stacked DRAM modules. High memory capacity is achieved through the deployment of multiple conventional (DDR-based) DIMMs.

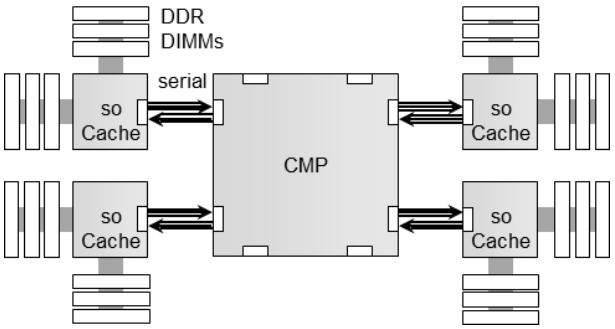
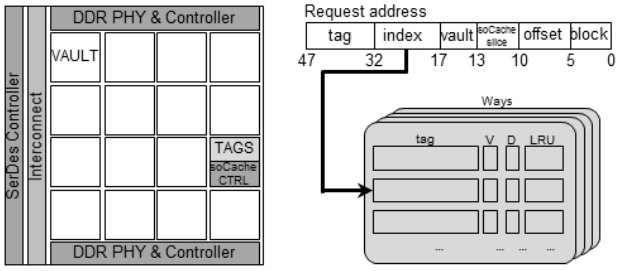


Figure 4. MeSSOS overview.



(a) Logic die organization (b) Tag array architecture

Figure 5. The organization of an soCache slice.

Figure 4 shows the design overview. In MeSSOS, an on-board building block consists of an soCache slice fronting a set of conventional DIMMs. The design of each building block (e.g., serial link and DDR bandwidth, cache-to-memory capacity ratio) is guided by our memory access characterization. Capacity and bandwidth can be seamlessly scaled by adjusting the number of building blocks. Next, we examine the soCache architecture and its integration with the rest of the system.

4.1. soCache Architecture

MeSSOS utilizes multiple off-package stacked DRAM modules as a high-capacity cache. To avoid communication between soCache slices, memory addresses are statically interleaved across the slices. Figure 5a shows the organization of an soCache slice. As shown in the figure, stacked DRAM is internally organized as a set of vaults, which are connected to the serial link via an interconnect.

Cache organization. soCache uses a page-based organization leveraging the observation that high-capacity caches uncover spatial locality that is beyond the temporal reach of lower-capacity caches. The page-based design not only naturally captures spatial locality, but also minimizes metadata storage requirements over block-based and footprint-predicting designs thanks to smaller tag entries and/or fewer cache sets. The page-based design also reduces dynamic DRAM energy by exploiting DRAM row buffer locality and fetching the entire page with one DRAM row activate, thus minimizing the number of DRAM row activates, which dominate energy consumption in conventional DRAM [12].

Based on page-size sensitivity analysis, we find that a page size of 2KB offers a good trade-off between tag array size and bandwidth overhead stemming from overfetch. We also observe that low associativity (4-way in the preferred design) is sufficient for minimizing the incidence of conflicts while also reducing tag and LRU metadata costs.

Tag array. The page-level organization reduces the tag array overhead significantly. For instance, a soCache of 32GB, consisting of eight 4GB slices, requires 5MB of tags per slice, or 20mm^2 in 40nm technology (obtained using CACTI).[1] The small tag array size allows us to embed it in the logic die of the modules comprising soCache. These logic dies are under-utilized, typically housing per-vault memory controllers, an on-chip interconnect, and off-chip I/O interfaces and controllers. In our specialized HMC, these components occupy ~70mm^2 in 40nm technology (estimated by scaling die micrographs) leaving sufficient room for the tags on a typical HMC logic die (~100mm^2).

To enable low tag lookup latency, we distribute the tag array across the high-bandwidth memory module, placing each tag array slice beneath a vault. For a 4GB soCache slice, each slice of the in-SRAM tag array requires only 320KB of storage and 3–4 cycles of access latency (obtained using CACTI). Low associativity and small in-SRAM tags allow for searching the ways in parallel at small latency and energy overheads, allowing for a feasible and practical a set-associative cache organization.

4.2. Processor-soCache Interface

The processor is connected to soCache via high-bandwidth serial links. Both processor and soCache slices implement simple controllers to orchestrate communication (Figure 4). The controllers consist of a pair of queues to buffer incoming and outgoing packets, and a SerDes interface. Processor-side controllers serialize outgoing requests into packets, before routing them to the soCache slice based on corresponding address bits (Figure 5b), and deserialize incoming data and forwards them to the last-level cache. An soCache-side controller deserializes incoming memory requests and forwards them to the vault's soCache controller based on corresponding address bits (Figure 5b), and serializes outgoing data into packets and forwards them to the processor.

As scale-out workloads exhibit variable read-write ratios [12], each serial link consists of 16 request lanes and 16 response lanes. Thus, a serial link requires ~70 pins (control

[1] Per soCache slice, a 4-way cache consists of 32K sets per vault, occupying 320KB of tags. Tag entries are 20-bit; 15 bits for the tag, 2 page-level valid and dirty bits, 3 bits for maintaining the pseudo-LRU tree.

and double-ended signalling for data lanes) as opposed to a DDR channel, which requires ~150 pins. The lower number of per-serial-link pins allows for integrating a high number of processor-side SerDes channels without increasing the number of the processor's pins compared to a processor with DDR channels, thereby keeping the cost associated with the processor's packaging constant.

4.3. soCache-Main Memory Interface

The off-package high-bandwidth memory modules provide the communication bridge between processor and main memory. Memory requests that miss in soCache are forwarded directly to local memory modules. To do so, the soCache slice integrates DDR controllers to control the local DDR channels, requiring the implementation of the DDR PHY and protocol in the logic die of the soCache modules.

DDR channels. Thanks to the high degree of bandwidth screening provided by soCache, the DDR channels operate at low frequency to reduce idle power. Compared to conventional HMCs hosting four SerDes interfaces, each of our specialized HMC hosts only one SerDes interface (of area ~9mm^2 and power 1.5Watt), freeing up area and power resources for the required low-frequency DDR interfaces (~10mm^2 each). Our estimates show that the power consumption of an soCache slice lies within the power budget of conventional HMCs. The total number of pins required by each soCache slice matches that of on-board chips in buffer-on-board systems.

DDR controllers. They employ FR-FCFS open-row policy with page-level address interleaving. We map an entire soCache's page-sized cache line to one DRAM row by using the following addressing scheme *Row:ColumnHigh:Rank: Bank:LocalChannel:soCacheSlice:ColumnLow:WordOffset*, where ColumnHigh(ColumnLow) is 2(8) bits. To guarantee that requests missing in an soCache slice are served by local DRAM, the mapping scheme interleaves addresses across local channels using the least significant vault bit.

4.4. System-Level Considerations

High-bandwidth memory technology. While we choose off-package stacked DRAM as our cache substrate, our insights on high-capacity cache design are also applicable to on-package stacked DRAM. Such design can lower cache access latency by avoiding chip-to-chip links, but at the cost of lower cache hit rates in big-memory systems, and additional buffer-on-board chips, which would be required to afford high memory capacity with conventional DIMMs given the pin-count limitations of a single package.

Scalability. MeSSOS delivers high memory capacity in a scalable manner while relying on cost-effective DIMMs. MeSSOS distributes the required number of DDR channels and their pins across multiple soCache modules as opposed to a single processor chip. This approach resembles that of buffer-on-board systems, which employ on-board chips to boost memory capacity in a cost-effective way. In contrast to these systems, MeSSOS does not require additional on-board buffer chips as the functionality of those chips is implemented in the logic die of the soCache modules.

TCO. MeSSOS achieves substantial system cost savings due to lower acquisition and operating costs. By providing the required bandwidth and capacity for a server, MeSSOS maximizes server throughput, thus reducing the number of servers required for the same throughput. MeSSOS also lowers memory energy by minimizing the static power footprint of its underlying memory interfaces. As MeSSOS employs off-package stacked DRAM as a cache, it (i) bridges the processor-bandwidth gap with a minimal number of power-hungry serial links, (ii) efficiently utilizes serial link bandwidth and amortizes their high idle power consumption, and (iii) filters a high degree of memory accesses, and thus infrequent main memory accesses can be served by under-clocked DIMMs.

5. Experimental Methodology

We evaluate MeSSOS performance and energy efficiency using a combination of cycle-accurate full-system simulations, analytic models, and technology studies.

5.1. Scale-Out Server Organization

We model chips with an area of 250–280 mm^2, and a power budget of 95–115 Watt. We use the scale-out processor methodology to derive the optimal ratio between core count and cache size in each technology [3]. The configuration resembles that of specialized manycore CMPs, such as Cavium ThunderX.

Table 2 summarizes the details of the evaluated designs across technology nodes. For a given technology node, the processor configuration and memory capacity are fixed. We evaluate the following memory systems: (i) DDR-only memory; (ii) buffer-on-board (BOB) system [9], which relies on on-board chips to boost bandwidth and capacity through additional DDR channels, but at the cost of higher end-to-end memory latency and energy consumption due to (processor-BOB) serial links and intermediate buffers; (iii) high-bandwidth memory modules (HBMM), which replaces DDR-based memory with off-package stacked DRAM—i.e., stacked DRAM is deployed as main memory. HBMM employs a tree network topology to reduce the number of network hops—average and maximum number of network hops is three and four, respectively; (iv) Die-stacked cache with a block-based organization [8] that maximizes effective capacity and minimizes off-chip bandwidth waste. The

Table 2. System configuration.

System	2015 (22nm)	2018 (18nm)	2021 (14nm)
CMP	96 cores, 3-way OoO, 2.5GHz	180 cores, 3-way OoO, 2.5GHz	320 cores, 3-way OoO, 2.5GHz
LLC	24 MB	45 MB	80 MB
Memory	384 GB	720 GB	1280 GB
DDR	4 DDR-1600	5 DDR-2133	6 DDR-2667
	Memory latency: 55ns including off-chip link (15ns) and DRAM core (40ns)		
HBMM	8 32-lane @ 10Gbps	10 32-lane @ 15Gbps	12 32-lane @ 20Gbps
	Memory latency: hop-count*35ns (SerDes & pass-through logic) + 20ns (stacked DRAM access)		
BOB	8 32-lane @ 10Gbps 16 DDR-1600	10 32-lane @ 15Gbps 20 DDR-2133	12 32-lane @ 20Gbps 24 DDR-2667
	Memory latency: 95ns including SerDes & buffer (40ns), buffer-DDR link (15ns) and DRAM core (40ns)		
Die-Stacked	Cache: 1GB	Cache: 2GB	Cache: 4GB
	Hit latency: ~20ns including predictor lookup and stacked DRAM access (20ns) Miss latency: ~55ns including predictor lookup and off-chip DRAM access (55ns)		
	Off-chip: 4 DDR-1600	Off-chip: 5 DDR-2133	Off-chip: 6 DDR-2667
MeSSOS	CMP-Cache: 8 32-lane @ 10Gbps Cache: 8x4GB	CMP-Cache: 10 32-lane @ 15Gbps Cache: 10x8GB	CMP-Cache: 12 20-lane @ 20Gbps Cache: 12x8GB
	Tag lookup latency: 35ns including SerDes (30ns) and distributed tag array lookup (5ns) Hit latency: 55ns including tag lookup (35ns) and stacked DRAM access (20ns) Miss latency: 95ns including tag lookup (35ns) and off-chip DRAM access (60ns)		
	Cache-Memory: 16 DDR-1066	Cache-Memory: 20 DDR-1066	Cache-Memory: 24 DDR-1066

cache is backed by DDR-based memory; and (v) MeSSOS that deploys stacked DRAM modules as a cache in front of DDR-based memory.

5.2. Performance and Energy Models

Due to space constraints, we present only a summary of our framework. The details of the framework, including system performance, energy modeling and projection to future technologies can be found elsewhere [13] [14].

Performance. We measure performance using analytic models, which are validated against cycle-accurate full-system simulations of a 16-core CMP with high accuracy (5% average error). Our model extends the classical average memory access time analysis to predict per-core performance for a given memory system. The model is parameterized by 16-core full-system simulations results (using Flexus [15]), including core performance, miss rates of on-chip and stacked DRAM caches, and interconnect delay. For off-chip access latency, we include link latency, memory core latency, and queuing delays. We model queuing delays by running cycle-accurate simulations to measure memory latency for various bandwidth utilization levels for each workload separately.

Energy. We develop a custom energy modeling framework to include various system components, such as cores, on-chip interconnects, caches, memory controllers, and memory. Our framework draws on several specialized tools (e.g., CACTI, McPAT) to maximize fidelity through detailed parameter control.

Future technologies. To understand the effect of technology scaling on the examined memory systems, we model our systems in 2018 and 2021. Per ITRS estimates, processor supply voltages will scale from 0.85V (2015) to 0.8V (2018) and 0.75V (2021). We use Micron's datasheets to examine the impact of data rate and memory density on DDR energy based on Micron's datasheets. We also study the impact of manufacturing technology on power consumption and data rate of SerDes interfaces based on numerous published measurements.

Workloads. Our analysis is based on a wide range of scale-out workloads taken from CloudSuite 2.0 [2]. We also evaluate online analytics running a mix of TPC-H queries on a modern column-store database engine, MonetDB.

6. Evaluation

We compare MeSSOS to various memory systems in terms of system performance and energy efficiency across technology generations.

6.1. Performance and Energy Efficiency Implications

We begin our study with a 96-core CMP in the 22nm technology. Figure 6 plots the fraction of memory requests that are served by soCache for various Cache-to-Memory Capacity Ratios. The figure demonstrates the ability of MeSSOS to serve the bulk (>95%) of those using its soCache thanks to temporal locality arising from skewed access distributions (gray bar) and spatial locality arising from page-based organizations and high cache residency times stemming from high cache capacity (white bar).

The figure (right) illustrates the DDR bandwidth consumption compared to the DDR baseline. As expected,

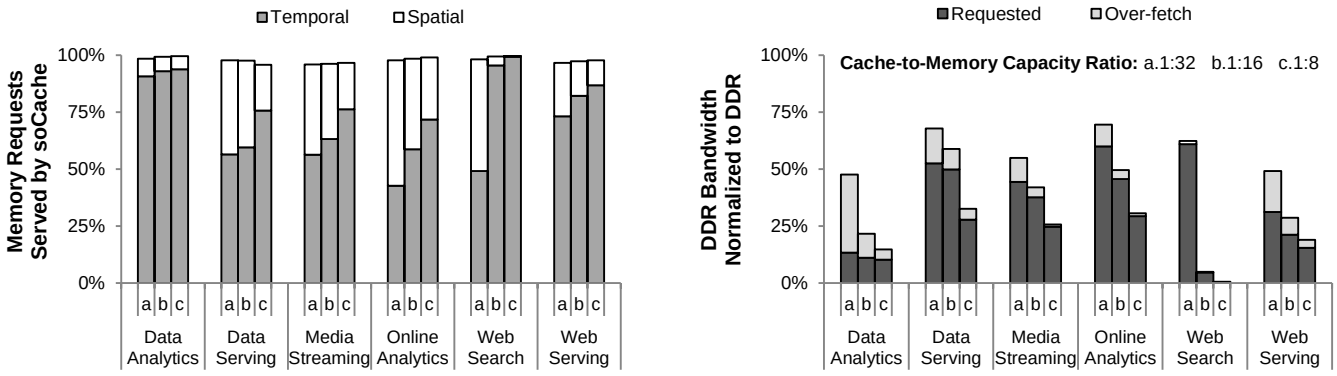


Figure 6. MeSSOS effectiveness for various Cache-to-Memory Capacity Ratio: (a) 1:32, (b) 1:16, (c) 1:8.

DDR bandwidth savings increase with bigger caches. For a 1:8 Cache-to-Memory Capacity Ratio, soCache captures the hot data working sets, and hence is able to absorb 65–95% of memory traffic. The light gray bars illustrate the extra traffic generated due to coarse-grained transfers between soCache and the DIMMs. The absolute increase in traffic is small (3% on average). For the rest of the evaluation, we use 1:8 Cache-to-Memory Capacity Ratio, unless stated otherwise.

Performance. Figure 7 compares MeSSOS to the DDR baseline as well as HBMM (which employs high-bandwidth memory modules as main memory), BOB, and Die-Stacked. BOB and HBMM improve performance over DDR by 49% and 33%, respectively, as they provide sufficient bandwidth to the processor. However, the bandwidth increase comes at the cost of higher memory latency. BOB adds an extra 40ns while HBMM requires a point-to-point network, which adds a latency of 35ns per network hop. Because HBMM accesses are frequently multi-hop, BOB outperforms HBMM by 12%. Our analysis (not shown) also finds that on-board SRAM caches found in some BOB chips exhibit low temporal locality (average hit ratio of 25%), and thus provide negligible performance gains.

MeSSOS outperforms all systems due to its ability to provide high bandwidth at low latency. Compared to the DDR baseline, MeSSOS improves system performance by ~2x. MeSSOS outperforms BOB and HBMM by 28% and 43%, respectively, due to lower memory latency.

MeSSOS outperforms Die-Stacked by 23% due to lower off-chip bandwidth pressure, resulting from its greater cache capacity. On average, MeSSOS filters 84% of DDR accesses as compared to 45% in Die-Stacked. For Data Serving and Online Analytics, MeSSOS outperforms Die-Stacked by 81% and 61%, as Die-Stacked is bandwidth-constrained due to is inability to filter off-chip bandwidth (only 38% and 13% of accesses). One exception is Data Analytics where memory accesses are extremely skewed,

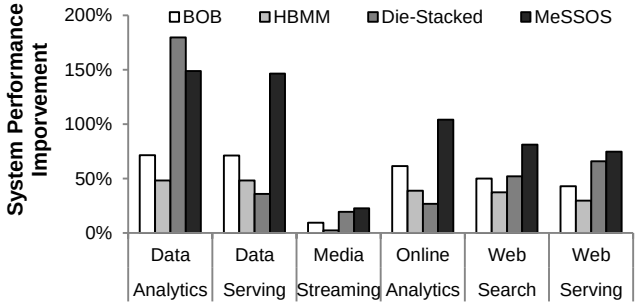


Figure 7. System performance improvement of various memory systems over DDR.

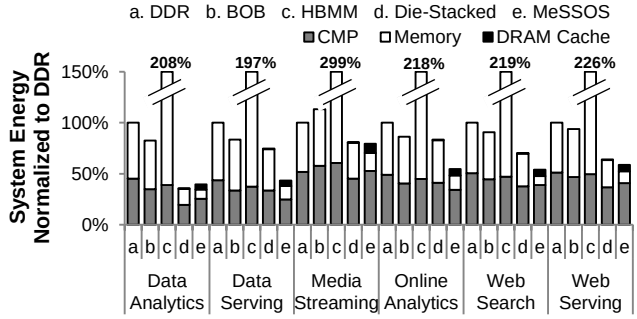


Figure 8. System energy breakdown.

and hence Die-Stacked achieves high hit ratio, outperforming MeSSOS due to lower cache access latency.

Energy. Figure 8 plots system energy for the examined designs normalized to the DDR baseline. BOB reduces energy by 12% compared to DDR mainly due to performance gains. HBMM increases energy by 2.3x compared to DDR due to its power-hungry memory network.

MeSSOS reduces system energy by 1.9x, 1.7x, and 4.3x compared to DDR, BOB, and HBMM. As bulk of the accesses are served by soCache, MeSSOS exploits the low-

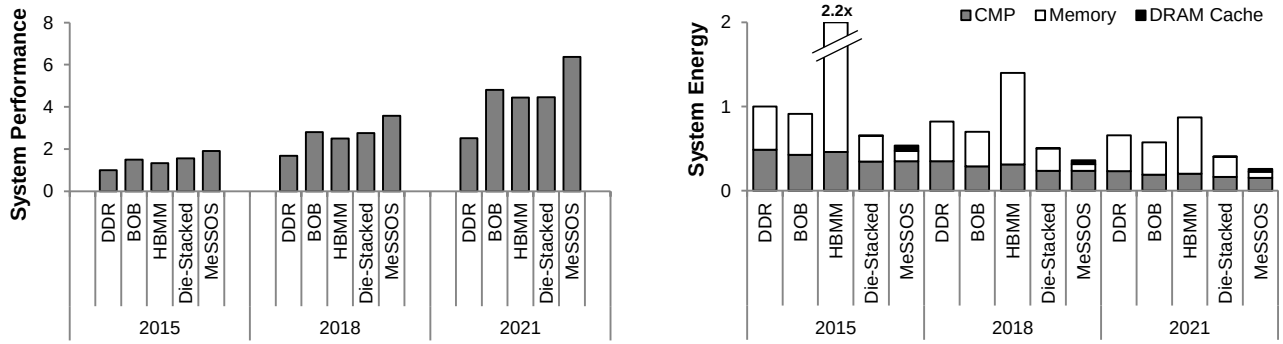


Figure 9. System performance and energy consumption for various technologies normalized to DDR-2015.

energy access of stacked DRAM modules, thus reducing memory energy consumption significantly. Furthermore, MeSSOS enforces coarse-grained data movement between soCache and DRAM, thus amortizing energy-intensive DRAM row activates [12]. Compared to Die-Stacked, MeSSOS reduces energy by 23% due to lower DDR energy resulting from lower off-chip bandwidth consumption.

6.2. Projection to Future Technologies

In Figure 9, we study the effect of technology scaling on MeSSOS's performance and energy efficiency in 14nm (2018) and 11nm (2021) technologies.

MeSSOS leverages the abundant bandwidth provided by SerDes, increasing performance almost linearly with the number of cores and by 3.7x (2018) and 6.6x (2021) compared to DDR-2015. Due to poor scalability of DDR interfaces, the bandwidth gap between DDR-based systems and the processor is increasing rapidly. Thus, MeSSOS's performance improvement over DDR and Die-Stacked increases across technologies. MeSSOS improves performance by 2.3x (2018) and 2.7x (2021) over DDR, and by 30% (2018) and 43% (2021) over Die-Stacked.

Regarding energy efficiency, the DDR energy footprint increases across technologies. Because MeSSOS employs under-clocked DIMMs, its energy footprint increases by only a small factor. Thus, MeSSOS reduces energy by 1.7x (2015), 2x (2018), and 2.6x (2021) compared to DDR and BOB, and by 23% (2015), 40% (2018), and 60% (2021) compared to Die-Stacked. Compared to HBMM, MeSSOS reduces energy by 4-4.4x.

7. Conclusion

We proposed a memory system architecture that utilizes multiple high-bandwidth memory modules as a scale-out cache, which is effective in capturing the secondary data working sets of scale-out workloads. Unlike state-of-the-art stacked DRAM caches employing in-DRAM block-level metadata, the proposed cache employs a page-based organization with low-overhead in-SRAM metadata as coarse-grained access patterns are dominant in high-capacity caches. The proposed memory system architecture provides the required memory bandwidth and capacity for scale-out servers.

References

[1] L. A. Barroso and U. Holzle, "The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machine," Madison: Morgan & Claypool, 2009.

[2] M. Ferdman et al., "Clearing the Clouds: A Study of Emerging Scale-out Workloads on Modern Hardware," in *ASPLOS*, 2012.

[3] P. Lotfi-Kamran et al., "Scale-Out Processors," in *ISCA*, 2012.

[4] D. Jevdjic et al., "Die-stacked DRAM Caches for Servers: Hit Ratio, Latency, or Bandwidth? Have it All with Footprint Cache," in *ISCA*, 2013.

[5] B. Black et al., "Die Stacking (3D) Microarchitecture," in *MICRO*, 2006.

[6] D. Jevdjic et al., "Unison Cache: A Scalable and Effective Die-stacked DRAM Cache," in *MICRO*, 2014.

[7] G. H. Loh and M. D. Hill, "Efficiently Enabling Conventional Block Sizes for Large Die-stacked DRAM Caches," in *MICRO*, 2011.

[8] M. Qureshi and L. H. Gabriel, "Fundamental Latency Trade-off in Architecting DRAM Caches: Outperforming Impractical SRAM Tags with Simple and Practical Design," in *MICRO*, 2012.

[9] Cisco, "Extended Memory Technology," [Online]. Available: http://www.cisco.com/c/dam/en/us/products/collateral/servers-unified-computing/ucs-5100-series-blade-server-chassis/at_a_glance_c45-555038.pdf.

[10] B. Grot et al., "Optimizing Datacenter (TCO) with Scale-Out Processors," *IEEE Micro*, vol. 32, no. 5, pp. 52-63, 2012.

[11] X. Jiang et al., "CHOP: Adaptive Filter-based DRAM Caching for CMP Server Platforms," in *HPCA*, 2010.

[12] S. Volos et al., "BuMP: Bulk Memory Access Prediction and Streaming," in *MICRO*, 2014.

[13] S. Volos et al., "An Effective DRAM Cache Architecture for Scale-Out Servers," Technical Report, MSR-TR-2016-20, Microsoft Research, 2016.

[14] S. Volos, "Memory Systems and Interconnects for Scale-Out Servers," Doctoral Dissertation, EPFL-THESIS-6682, Dept. of Computer & Communication Sciences, EPFL, 2015.

[15] T. F. Wenisch et al., "SimFlex: Statistical Sampling of Computer System Simulation," *IEEE Micro*, vol. 26, no. 4, pp. 18-31, 2006.