International Journal of Scientific Research and Engineering Development-– Volume 2 Issue 6, Nov- Dec 2019

Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

# A Study of Disaggregated Memory Management Techniques with Hypervisor Based Technique

Lakshmi Prasad[1], Smita C Thomas[2]

[1](P G Scholar, Department of Computer Science and Engineering,Mount Zion College of Engineering, Kadammanitta
Email: lakshmiprsd24@gmail.com)

[2] (Research Scholar, VelsUniversity,Chennai
Email: smitabejoy@gmail.com)

----------**********----------

## Abstract:

Increasing importance of in-memory processing, cloud service suppliers have launched massive memory virtual machine services to accommodate memory intensive workloads. Such massive memory services victimization low volume scaled-up machines square measure so much less cost-effective than scaled-out services consisting of high-volume artefact servers. By exploiting memory usage imbalance across cloud nodes, disaggregated memory will rescale the memory capability for a virtual machine during a cost-efficient approach. Disaggregated memory permits obtainable memory in remote nodes to be used for the virtual machine requiring a lot of memory than its domestically obtainable memory. It supports high performance with the quicker direct memory whereas satisfying the memory capability demand with the slower remote memory. This paper compares some existing technologies with planned system. It additionally proposes a new hypervisor-integrated disaggregated memory system for cloud computing. The hypervisor-integrated style has many new contributions in its disaggregated memory style and implementation. initially the tight hypervisor integration, investigates a new page management mechanism and policy tuned for disaggregated memory in virtualized systems. Second, it restructures the memory management procedures and relieves the measurability concern for supporting massive virtual machines. Third, exploiting page access records obtainable to the hypervisor, it supports application-aware elastic block sizes for taking remote memory pages with totally different granularities. reckoning on the degrees of spacial neighborhood for various regions of memory during a virtual machine, the optimum block size for every memory region is dynamically chosen. The experimental results with the implementation integrated to the KVM hypervisor, shows that the disaggregated memory will give on the average half-dozen % performance degradation compared to the best local-memory solely machine.

*Keywords* **—Disaggregated memory; cloud computing; virtualization; remote memory**

----------**********----------

## I. INTRODUCTION

The growthof data-intensive workloads, such as inmemory databases, data caching, bioinformatics, andgraph processing, has been tremendously increasing the memory capacity requirements in cloud servers. To accommodate such big memory applications, cloud providers have begun to offer large machine types with more than 1 TB of memory. Google Compute Engine announced a plan to provide 1 TB of memory in 2017. Amazon already started to support EC2 x1e.32xlarge with 4 TB ofmemory in four AWS regions and announced a plan to launch EC2 instances with 16 TB of memory. With the ever-increasing demand for fast in-memory big-data processing, the cloud providers are expected to adopt large VM instances more and more widely. However, offering such large VM

  

instances requires massive investment for the existing infrastructure currently composed of commodity volume servers connected through high-speed interconnects. For example, the EC2 large virtual machine instance with 4 TB memory is supported only in 4 regions, limiting the benefits of large memory machines for remotely located users. Furthermore, large memory machines with TBs of memory and high core counts, provide far less performance per cost or performance per watt than the commodity volume servers. Meanwhile, in the cloud systems, the heterogeneity of workloads commonly incurs the imbalance of memory usages in each node. Such variance in memory usages can cause memory shortages in some machines, while other machines have ample free memory. The inherent memory imbalance can open a new opportunity to provide a large memory virtual machine (VM) cost-efficiently by combining the free memory in multiple servers into a single unified memory.

To support large memory with volume commodity servers, disaggregated memory allows distributed memory across different physical servers to be used as a single memory, creating an illusion of larger memory than the physical memory limit on a single machine. This paper proposes a new hypervisor-based disaggregated memory, providing large scalable memory for guest VMs transparently. With the disaggregated cloud memory (dcm) based on VMs, an application on a VM can use the remote memory in other machines in addition to the local memory without any modification in the application binary and guest operating systems.Instead, the hypervisor hides all the complexity ofaccessing remote memory via page-oriented remote memory accesses. If the accessed memory page from a guest VMis not in the local memory, a page fault will initiate an access to the remote memory, and the new local memory page will be allocated for the faulting VM.

## II. LITERATURE SURVEY

Disaggregated memory is an exciting technology proposed to improve memory utilization in cloud data centres. Its basic idea is to detach ("disaggregate") most of the memory of each machine, placing it on a common fabric, where it forms a cluster memory pool; this memory can be assigned to machines when and if they need it. To realize disaggregated memory, new hardware architectures are under development in both academic and industrial settings. While new hardware could be interesting, its cost could be prohibitive and its availability may be limited.

According to Aguilera et al.[1],disaggregated memory can be fully realized in software alone, without new hardware architectures, new standards, or new interconnects, using instead commodity of-the-shelf hardware available today. To clearly distinguish the software from the hardware approaches, we call the former remote memory, while reserving the term disaggregated memory for the hardware solution. With remote memory, each machine has a conventional memory architecture and it contributes parts of its memory to the cluster memory pool. Machines that need memory then access the memory of another machine over the network. Using the virtual memory subsystem, accesses to the remote memory are transparent, appearing like accesses to local memory. Remote memory brings many benefits, such as huge memories, better utilization, and more efficient data exchange. Remote memory is an old idea, similar to distributed shared memory and swapping to the network. These ideas were studied extensively 20 years ago, and now they are getting revived in different ways. What is different today? In the 2000s, network latency was three orders of magnitude higher than memory latency (hundreds ofμs vs. hundreds of ns). Since then, network latencies have improved significantly, while memory latencies have not, so there is a gradual convergence of performance. So, might remote memory work now? It turns out that efficiently realizing remote memory requires overcoming several challenges, from virtual memory overheads, crashes of remote machines, sharing model, virtualization, scalability, and placement. The paper enumerates the challenges, discuss their feasibility, explain how some of them are addressed by recent work, and indicate other promising directions to explore. Currently, no system addresses all of the challenges. Some of the challenges remain open

  

problems, while others have been studied but not extensively. We believe much more research is required in this topic, and we hope to provide a broad research agenda around it, by proposing more problems than solutions.

According to Rao et al.[2],Achieving efficiency in data processing requires balanced computing, meaning that a system has the right mix of CPU, memory, storage IO, and network IO so that one part of the computation is not bottlenecked waiting for results from another part of the computation. Getting this balance just right is a moving target, since the input data, number and type of queries, presence of failures, and network conditions are in a constant state of flux. Correctly provisioning baremetal servers is a challenge since one must determine their specific configuration at entirely the wrong timescales, well before they are put into production. While virtual machines play an important role in providing more flexibility in balancing resources, they are not enough. Even with VMs, you are limited to configurations implementable in a single server, you are subject to "fragmentation" of resources, and you are not able to upgrade individual components like CPU and memory independently of each other. These challenges have led to disaggregated server designs, where individual components such as CPU, memory, and storage are interconnected over a network, rather than over a bus within a single chassis. The advantages of disaggregation include more efficient utilization of resources, and the ability to independently upgrade different system components. The challenge for disaggregation is the "memory wall". Today storage is commonly disaggregated via SANs and other network-based file storage protocols, and Facebook has introduced a disaggregated system-on-- chip (SoC) platform called Yosemite, which relies on networked storage. Yet there is a growing gap in the rate at which CPUs can execute instructions and the rate at which data can be fetched into the CPU from main memory. For this reason in Yosemite (as well as other designs), memory and CPU are still tightly integrated in the same chassis. The paper puts aside the issue of disaggregating memory in general, and instead examines disaggregating memory for a common and increasingly deployed type of application: analytics queries. Using Spark SQL as a motivating platform, we measure the actual rate at which threads of execution access memory and process records, and using these measurements, determine the feasibility of disaggregating memory. Spark is an example of a growing set of data-parallel frameworks which exhibit minimal data-dependent branches, and as such, can take advantage of significant amounts of pipelining. For this reason, they are largely latency insensitive, further enabling the use of disaggregated memory. The initial results show that even after significant optimization, Spark SQL analytics queries access memory an order of magnitude slower than the underlying components permit, opening up the possibility of disaggregating memory from compute. In fact, the requirements on the underlying network are modest, and can be met with existing commercial products such as 40- and 100-Gb/s NICs (e.g., the Mellanox ConnectX-4 NIC [17]). We conclude by recommending further changes that improve Spark SQL's ability to support memory disaggregation.

According to Samihet al.[3],Highly-integrated distributed systems such as Intel Micro Server and SeaMicro Server are increasingly becoming a popular server architecture. Designers of such systems face interesting memory hierarchy design challenges while attempting to reduce/eliminate the notorious disk storage swapping. Disk swapping activities slow down applications' execution drastically. Swapping to the free remote memory - nearby nodes, through Memory Collaboration has demonstrated its cost-effectiveness compared to overprovisioning memory for peak load requirements. Recent studies propose several ways to access the under-utilized remote memory in static system configurations, without detailed exploration of dynamic memory collaboration. Dynamic collaboration is an important aspect given the run-time memory usage fluctuations in clustered systems. Furthermore, with the growing interest in memory collaboration, it is crucial to understand the existing performance bottlenecks, overheads, and potential optimizations. The paper addresses these two issues. Here, proposes an Autonomous

  

Collaborative Memory System (ACMS) that manages memory resources dynamically at run time, to optimize performance, and provide QoS measures for nodes engaging in the system and implements a prototype realizing the proposed ACMS, experiment with a wide range of real-world applications, and show up to 3x performance speedup compared to a non-collaborative memory system, without perceivable performance impact on nodes that provide memory. Second, analysed, in depth, the end-to-end memory collaboration overhead and bottlenecks. Based on this analysis, here provides insights on several corresponding optimizations to further improve the performance.

According to Lim et al.[4],Technology and cost barriers prevent the straightforward use of capacity scaling to meet the growing need in systems for increased memory capacity. Disaggregated memory is a design that can provide a cost-effective way to scale memory capacity. To explore its software implications, we developed a software-based prototype of disaggregated memory by adding hypervisor support for remote memory. We evaluated the impact of two replacement policies, Round-robin and Clock, and two replacement schemes, on-demand and pre-evict, on disaggregated memory. The findings show that the low latency of remote memory favours approaches that minimize replacement time, with simpler Round-robin outperforming the more accurate Clock which has higher latency for page selection, and with pre-evict greatly outpacing on-demand. Here demonstrates that disaggregated memory has synergy with content-based page sharing (CBPS), with the combination outperforming either technique alone. Finally, the case study of an interactive web caching workload (memcached) shows that disaggregated memory provides similar response time performance at a lower cost compared to scaling out on multiple compute blades. The study demonstrates feasibility of the software infrastructure required for disaggregated memory and shows that it can provide performance-per-dollar benefits for important emerging workloads.

## III. PROPOSED METHOD

In the proposed hypervisor-based design, the disaggregated memory support is directly integrated to the page management in the KVM hypervisor. Unlike the prior study which supports the remote memory as a block device and uses the existing storage-based swap mechanism,the proposed integrated design can provide finegrainedadjustments of memory eviction and high scalability with the hypervisor integration.

One of the key observations from the disaggregated cloud memory is that the granularity of memory fetching from remote machines affect the frequency of remote memory accesses significantly. If the memory access patterns have high spatial locality, increasing the granularity of memory fetching (block size) can reduce future page faults by a prefetching effect. If the memory access pattern exhibits small random accesses, a small block can reduce the remote memory access overhead for each fault. This paper proposes a dynamic block size adjustment technique, called elastic block, to find the optimal block size for each VM. The proposed mechanism can assign the optimal block size not only for each VM but also different memory regions in VMs, as it tracks the spatial locality in the hypervisor-managed memory map for each VM.

Compared to the prior work, the paper has the following new contributions.

- The paper compares various existing technologies with the proposed system.
- The paper proposes an integrated hypervisor-baseddesign for disaggregated memory. Instead of relyingon the conventional swap system designed for slowdisks, the disaggregated memory support is directlyadded to the memory management in the KVMhypervisor.
- The integrated design overhauls thememory managementmechanisms and policies with a new replacementscheme turned for disaggregated memory, alatency hiding mechanism by overlapped memoryreclamation and network operations, and scalabilityimprovements.

  

- This study identifies the importance of selecting rightmemory fetching sizes from remote systems in disaggregatedmemory designs. The best block size mayvary by many factors including access patterns,availablelocalmemory, I/Activities, and programphases.
- The design allows a fine-grained adjustment of blocksize to minimize the overhead of remote memoryaccesses, while maximizing prefetching effects. Blocksizes are dynamically adjusted for each page addressin a virtual machine, and thus the adjustment mechanismcan find the best block size for different regionseven within a virtual machine

## IV. CONCLUSION

This paper proposed a new memory disaggregation system backed by RDMA supported high bandwidth networks. Theproposed hypervisor-based design for disaggregated memory provides memory extension to the remote memory transparently to guest operating systems and applications. Its new design proposed a new replacement scheme, overlapped memory reclaim and network transfer, and scalability supports by per-vCPU data structures and lockless writeback operations. In addition, the elastic block maximizes the performancebenefit of exploiting the spatial locality, as itdynamically adapts to changing access patterns. The experimental results showed that the disaggregated memory can provide on average 6 percent performance degradation compared to the ideal local-memory only machine, even though the direct memory capacity is only 50 percent of the total memory footprint. In addition, the proposed design provides scalable performance with increasing numbers of vCPUs. With the advent of high bandwidth non-volatile memory technologies, the proposed disaggregated memory will be able to expand its support for general hierarchicalmemory systems, such as conventional DRAM and new non-volatile memory. To prove its design flexibility, the paper also showed the preliminary performance evaluation with the new Optane SSD as the indirect memory.

## REFERENCES

[1] M. K. Aguilera, N. Amit, I. Calciu, X. Deguillard, J. Gandhi, P. Subrahmanyam, L. Suresh, K. Tati, R. Venkatasubramanian, and M. Wei, “Remote memory in the age of fast networks,” in Proc. Symp. Cloud Comput., 2017, pp. 121–127.

[2] P. S. Rao and G. Porter, “Is memory disaggregation feasible?: A case study with spark SQL,” in Proc. Symp. Archit. Netw. Commun. Syst., 2016, pp. 75–80.

[3] A. Samih, R. Wang, C. Maciocco, M. Kharbutli, and Y. Solihin, “Collaborative memories in clusters: Opportunities and challenges,” in Transactions on Computational Science XXII, Berlin, Germany: Springer, 2014, pp. 17–41.

[4] K. Lim, Y. Turner, J. R. Santos, A. AuYoung, J. Chang, P. Ranganathan, and T. F.Wenisch, “System-level implications of disaggregated memory,” in Proc. IEEE 18th Int. Symp. High-Perform. Comput. Archit., 2012, pp. 1–12.

  