**INTRODUCTION**

Data deduplication has been demonstrated to be an effective technique in Cloud backup and archiving applications to reduce the backup window, improve the storage-space efficiency and network bandwidth utilization. Recent studies reveal that moderate to high data redundancy clearly exists in VM (Virtual Machine) [6], [17], enterprise [20], [29], [37], [8] and High-Performance Computing (HPC) [19], [27] storage systems. These studies have shown that by applying the data deduplication technology to large-scale data sets, an average space saving of 30%, with up to 90% in VM and 70% in HPC storage systems, can be achieved [6], [37], [27]. For example, the time for the live VM migration in the Cloud can be significantly reduced by adopting the data deduplication technology [46]. The existing data deduplication schemes for primary storage,n such as iDedup [37] and Offline-Dedupe [8], are capacityoriented in that they focus on storage capacity savings and only select the large requests to deduplicate and bypass all the *\_ B. Mao and S. Wu are with Xiamen University, Xiamen, Fujian, China.* *E-mail: fmaobo,* [*suzheng@xmu.edu.cn*](mailto:suzheng@xmu.edu.cn) *\_ H. Jiang and L. Tian is with the Department of Computer Science and* *Engineering, University of Nebraska-Lincoln, Lincoln, NE, USA.* *E-mail:* [*jiang@cse.unl.edu*](mailto:jiang@cse.unl.edu) *\_ This is an extended version of our manuscript published in the Proceedings* *of the 28th IEEE International Parallel & Distributed Processing Symposium* *(IPDPS’14), Pheonix, AZ, USA, May 2014.* small requests (*e.g.*, 4KB, 8KB or less). The rationale is that the small I/O requests only account for a tiny fraction of the storage capacity requirement, making deduplication on them unprofitable and potentially counterproductive considering the substantial deduplication overhead involved. However, previous workload studies have revealed that small files dominate in primary storage systems (more than 50%) and are at the root of the system performance bottleneck [29], [23], [4], [40], [27]. Furthermore, due to the buffer effect, primary storage workloads exhibit obvious I/O burstiness [23], [4]. From a performance perspective, the existing data deduplication schemes fail to consider these workload characteristics in primary storage systems, missing the opportunity to address one of the most important issues in primary storage, that of performance. With the explosive growth in data volume, the I/O bottleneck has become an increasingly daunting challenge for big data analytics [39] in terms of both performance and capacity. Recent International Data Corporation (IDC) studies indicate that in past five years the volume of data has increased by almost 9 times to 7ZB per year and a more than 44-fold growth to 35ZB is expected in the next ten years [41]. *Managing the data deluge on storage to support (near) real-time data analytics becomes an increasingly critical challenge for Big Data* *analytics in the Cloud, especially for VM platforms where the* *sheer number and dominance of small files overwhelm the I/O* *data path in the Cloud [40], [10]*. Moreover, our workload analysis, detailed in Section 2.1, IEEE Transactions on Computers,Volume:65,Issue:6,Issue Date :June.1.2016 2 shows that data redundancy on the critical I/O path is much more pronounced than on the storage devices, largely due to the high temporal locality of small I/O requests. This suggests that, if such redundant I/Os can be removed from the critical I/O path, the performance bottleneck of a primary storage system may be significantly alleviated, if not removed. Thus, we argue that, for primary storage systems in the Cloud, it may be at least as important, if not more so, to deduplicate the redundant I/Os on the critical I/O path for the sake of performance as to deduplicating redundant data on storage devices for the sake of capacity savings. On the other hand, our experimental studies suggest that directly applying data deduplication to primary storage systems will likely cause space contention in the main memory and data fragmentation on disks. This is in part because data deduplication introduces significant index-memory overhead to the existing system and in part because a file or block is split into multiple small data chunks that are often located

in non-sequential locations on disks after deduplication. This fragmentation of data can cause a subsequent read request to invoke many, often random, disk I/O operations, leading to performance degradation. Our preliminary evaluations on the VM disk images reveal that the restore times with deduplication are much higher than those without deduplication, by an average of 2.9*\_* and up to 4.2*\_* [24]. These two problems will be particularly acute with the deployment of the data deduplication technology into the primary storage systems for big data analytics in the Cloud. To address the important performance issue of primary storage in the Cloud, and the above deduplication-induced problems, we propose a Performance-Oriented data Deduplication scheme, called POD, rather than a capacity-oriented one (*e.g.*, iDedup), to improve the I/O performance of primary storage systems in the Cloud by considering the workload characteristics. POD takes a two-pronged approach to improving the performance of primary storage systems and minimizing performance overhead of deduplication, namely, a request-based selective deduplication technique, called Select- Dedupe, to alleviate the data fragmentation and an adaptive memory management scheme, called iCache, to ease the memory contention between the bursty read traffic and the bursty write traffic. More specifically, Select-Dedupe takes the workload characteristics of small-I/O-request domination into the design considerations. It deduplicates all the write requests if their write data is already stored sequentially on disks, including the small write requests that would otherwise be bypassed from by the capacity-oriented deduplication schemes. For other write requests, Select-Dedupe does not deduplicate their redundant write data to maintain the performance of the subsequent read requests to these data. iCache dynamically adjusts the cache space partition between the index cache and the read cache according to the workload characteristics, and swaps these data between memory and back-end storage devices accordingly. During the write-intensive bursty periods, iCache enlarges the index cache size and shrinks the read cache size to detect much more redundant write requests, thus improving the write performance. During the read-intensive bursty periods, on the other hand, the read cache size is enlarged to cache more hot read data to improve the read performance. Thus, the memory efficiency is maximized. The prototype of the POD scheme is implemented as an embedded module at the block-device level and a subfile deduplication approach is used. To examine the net effect of the POD scheme, in our trace-driven evaluation we use the blocklevel traces that were collected beneath the memory buffer cache so that the caching/buffering effect of the storage stack is already fully captured by the traces. In other words, all the small I/O requests in our evaluation are issued from the buffer cache to the block devices after the former has processed the filesystem-issued requests. The extensive trace-driven experiments conducted on our lightweight prototype implementation of POD show that POD significantly outperforms iDedup in the I/O performance measure of primary storage systems without sacrificing the space savings of the latter. Moreover, as an application of the POD technology to a background I/O task in primary cloud storage, it is shown to significantly improve the online RAID reconstruction performance by reducing the user I/O intensity during recovery.