Taming Workload Bursts in Data Centers

Lanyue Lu (Student) and Peter Varman[*]

The increasing complexity of managing stored data and the economic benefits of consolidation are driving storage systems towards a service-oriented paradigm, in which personal and corporate clients purchase storage space and bandwidth to store and retrieve their data. In this scenario, trade-offs with regard to QoS performance guarantees, pricing, and resource provisioning for hosting data centers assume increasing importance. A fundamental challenge in data center operations is the need to deal effectively with high-variance bursty workloads arising in the network and storage server traffic. Although only a small portion of the workload, the bursts have a disproportionate effect on the performance of the entire workload. This tail wagging the dog situation results in the server being forced to make unduly conservative estimates of resource requirements, resulting in excess resource commitments with associated monetary and energy consumption costs, and unnecessary throttling of the number of the clients admitted into the system.

In this abstract, we present a novel workload shaping framework to improve client performance and slim resource provisioning. In our approach we modify the characteristics of the arriving workload so that its behavior is dominated by the majority well-behaved portion of the request stream; the portions of the workload comprising the tail are identified and isolated so that their effects are localized. This results in more predictable behavior, and significantly lower resource requirements. The performance SLA consequently is specified by a distribution of response times rather than a single worst-case measure. For instance, rather than specifying a single upper bound r on the response time for all requests, a client may relax the requirements and instead require that 99% of the requests meet the bound r and the remaining requests meet a more relaxed latency r'. An n-tier SLA is represented by the set of n pairs of frequency and response times: $\{(f_i, r_i) | 1 \leq i \leq n, 0 < f_{i-1} < f_i \leq 1, f_n = 1.0\}$. A 3-tier response time distribution $\{(0.9, 20ms)$, $(0.99, 50ms)$, $(1.0, 500ms)\}$ indicates that no more than 10% of the requests can exceed 20ms latency, no more than 1% should exceed 50ms, while all requests must be served within a 500ms response time.

The workload shaping procedure consists of two complementary operations: *decomposition* and *recombination*. In the decomposition phase, the workload of a single application (or client) is partitioned into several classes with different performance guarantees. The requests belonging to the different classes are directed to separate queues (primary and secondary). In the recombination phase the requests of the separated classes are multiplexed in a suitable manner to satisfy the individual performance constraints. We provide an optimal decomposition algorithm, which can efficiently identify a maximal-sized set of requests that can meet the deadline, given the workload, capacity and response time bound. For the recombination, we provide a slack-based algorithm which schedules the requests from different queues in a response time sensitive manner for QoS guarantees.

Current Progress: We have implemented our workload shaping framework in DiskSim as a new I/O scheduler, and evaluated it using several block level storage traces: WebSearch, OLTP, TPC-D and OpenMail. The results show several advantages of our framework: (i) it reduces the storage server resource requirements (capacity and power) significantly while affecting QoS guarantees minimally; (ii) it results in better response time distribution than the unshaped workload; (iii) it provides much more accurate resource planning for single and multiple clients compared to simple worst-case resource requirements aggregation. We are currently implementing this framework as a new I/O scheduler in the Linux kernel. In applying this high level idea there are several directions for improvements that are currently under investigation.

Data Locality: Disk throughput highly depends on the spatial locality in the workload. During decomposition, a stream of sequential requests may be separated into different queues destroying locality in the original request stream. To avoid throughput degradation in the recombination process, the scheduling must be sensitive to the underlying data locality. Therefore, we not only schedule the requests across different queues to meet response time bounds, but also try to improve the system throughput by re-ordering the requests in the individual queues using seek-distance minimization schedulers like SSTF.

Adaptive Decomposition: The presence of locality in the request stream can be leveraged to improve the decomposition performance. A static decomposition scheme distributes requests to different queues using random access service time estimates. Request streams with high locality will finish faster than estimated, and the decomposition component should dynamically adjust its partitioning decision to leverage the free disk bandwidth. An adaptive algorithm will choose between scheduling requests from a secondary queue, or continue giving service to the primary queue, to achieve performance tuned to the current access pattern.

Multiple Clients: In a shared storage environment, each client or application may have different QoS requirements. An open question is: Where should our workload shaping framework reside? There are two options: local or global. The first choice means that we maintain separate request queues for each client during the decomposition process. The aggregate storage bandwidth is then multiplexed among the clients based on their reservations or requirements. The second option means that we maintain shared queues for all the clients during the decomposition process. The scheduler records the number of requests from each client and uses a global strategy to decide how to decompose the workload when new requests arrive. Also, it re-orders the requests in a fair-queueing manner that guarantees the response time requirements of all the clients.

[*]Rice University, email:{ll2,pjv}@rice.edu