Advanced RDMA-based Admission Control for Modern Data-Centers

P. Lai S. Narravula K. Vaidyanathan D. K. Panda
Department of Computer Science and Engineering
The Ohio State University
{laipi, narravul, vaidyana, panda}@cse.ohio-state.edu

***Abstract*—Current data-centers employ admission control mechanism to maintain low response time and high throughput under overloaded scenarios. Existing mechanisms use internal (on the overloaded server) or external (on the front-end proxies) admission control approaches. The external admission control is preferred since it can be performed transparently without any modifications to the overloaded server and global decisions can be made based on the load information of all the back-end servers. However, external admission control mechanisms are bound to use TCP/IP communication protocol to get the load information from the back-end servers and rely on coarse-grained load monitoring due to the overheads associated with fine-grained load monitoring. In this paper, we provide a fine-grained external admission control mechanism by leveraging the one-sided RDMA feature of high-speed interconnects and consequently provide superior performance, response time guarantees and overload control in a data-center environment. Our design is implemented over InfiniBand-based clusters working in conjunction with Apache based servers. Experimental evaluations with single file, world cup and zipf traces show that our admission control can improve the response time by up to 28%, 17% and 23%, respectively, as compared to performing TCP/IP-based admission control and 51%, 36% and 42%, respectively, as compared to the base performance without any admission control. Further, our evaluations also show that RDMA-based admission control mechanism can provide better QoS guarantees as compared to TCP/IP-based admission control and no admission control approaches.**

I. Introduction

There has been a tremendous growth of internet-based applications in the fields of e-commerce, bio-informatics, satellite weather image analysis, etc., in recent years. Typically, these applications are hosted through a cluster-based data-center. Figure 1 shows the common components involved in designing such a cluster-based data-center. Requests from clients (over Wide Area Network (WAN)) first pass through a front-end proxy (Tier 0) which performs basic triage on each request to determine if it can be satisfied by a static content web server or if it requires more complex dynamic content generation. The proxies also usually do some amount of caching. Tier 1 is generally responsible for all application-specific processing such as performing an online purchase or building a query to filter some data. At the back end of the processing stack is the data repository/database server (Tier 2) with the associated storage. This is the prime repository of all the content that is delivered or manipulated.

With increasing interest in on-line businesses and personalized services hosted in such cluster-based data-centers, a large number of clients request for either the raw or some kind of processed data simultaneously. Unfortunately, the request workloads vary widely over time due to phenomena like time-of-day effects and flash crowds [3], [10]. In these situations, data-centers receive huge bursts of requests, leading to overload scenarios, thus making the data-center extremely slow to respond to clients. Moreover, in business environments, clients pay for the data-center resources and in turn expect QoS (quality of service) guarantees even under overload situations. Such requirements make the management of data-center resources a challenging task [11], and more importantly increase the need for an efficient admission control mechanism to help improve the data-center and meet these performance guarantees in the presence of overload.

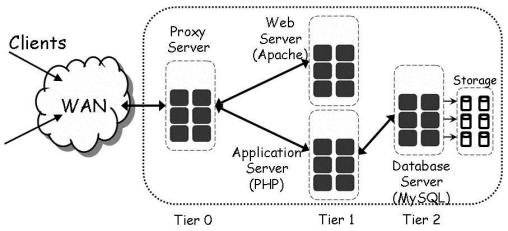

Fig. 1. Cluster-based data-center

In order to tackle this problem, researchers have come up with a number of techniques [6], [9] which focus on performing admission control either on the overloaded servers (internal admission control) or on the external front-end tier nodes (external admission control) that monitor the load information of the overloaded servers and accordingly use this information to determine whether a request should be admitted. Typically, the external admission control mechanism is preferred since admission control can be performed transparently without any modifications to the overloaded server and global decisions can be made based on the load information of all the back-end servers. The external admission control mechanisms are bound to use TCP/IP communication protocol to get the load information from the back-end servers and rely on coarse-grained load monitoring due to the overheads associated with fine-grained load monitoring. However,

as demonstrated by recent literature [15], [7], the resource usage of requests is becoming extremely divergent and unpredictable, thus increasing the need to perform admission control in a fine-grained manner. Moreover, as mentioned in [12], the load on the back-end servers can significantly affect the responsiveness of load monitoring since TCP/IP is a two-sided communication protocol which requires some amount of CPU on both sides of the communication network.

On the other hand, Remote Direct Memory Access (RDMA) is emerging as the central feature of modern network interconnects like InfiniBand(IBA) [2] and 10-Gigabit Ethernet [1]. RDMA operations allow the network interface to transfer data between local and remote memory buffers without any interaction with the operating system or CPU intervention. In this paper, we leverage the RDMA capabilities to design more efficient external admission control mechanisms and consequently provide superior performance, response time guarantees and overload control in a data-center environment.

This work contains several research contributions:

1) We present an architecture for achieving fine-grained admission control for multi-tier data-centers. This architecture requires minimal changes to legacy data-center applications. It is currently implemented over InfiniBand with Apache based servers. It could as such be used with any protocol layer; at the same time, it allows us to take advantage of the advanced features provided by InfiniBand to further improve performance and scalability of admission control in data-center environments.
2) Our experimental evaluations with single file, world cup and zipf traces show that the fine-grained admission control approach can improve the response time by up to 28%, 17% and 23%, respectively, as compared to performing admission control using the TCP/IP communication and 51%, 36% and 42%, respectively, as compared to the base performance. Further, our evaluations show that the system with RDMA-based admission control mechanism can provide better QoS guarantees as compared to systems with TCP/IP-based admission control mechanism and traditional systems.
3) Our results also show that one-sided operations such as the RDMA operations can provide better performance robustness to load in the data-center as compared to two-sided protocols such as TCP/IP over the same IBA network. This feature becomes more important because of the unpredictability of load in a typical data-center environment which supports large-scale dynamic services.

The rest of the paper is organized as follows: Section II provides an overview of high-speed interconnects and fine-grained resource monitoring services. In Section III, we discuss the design and implementation of our admission control approach in detail. We analyze the experiment results in Section IV, discuss the related work in Section V and summarize our conclusions and possible future work in Section VI.

II. Background

In this section, we provide a brief introduction of high-speed interconnects and fine-grained resource monitoring.

A. High-Speed Interconnects

Modern high performance interconnects, such as InfiniBand, iWARP/10-Gigabit Ethernet, Quadrics, etc., not only provide high performance in terms of low latency and high bandwidth but also provide a range of novel features such as remote memory operations (RDMA read and write), atomic operations, protocol offload, etc. Remote Direct Memory Access (RDMA) operations are used to allow the initiating node to directly access the memory of remote-node without the involvement of the remote-side CPU. Therefore, a RDMA operation has to specify the memory address for the local buffer as well as that for the remote buffer. In addition, RDMA operations are allowed only on pinned memory locations thus securing the remote node from accessing any arbitrary memory location. There are two kinds of RDMA operations: RDMA Write and RDMA Read. In a RDMA write operation, the initiator directly writes data into the remote node's memory. Similarly, in a RDMA read operation, the initiator reads data from the remote node's memory. In this paper, we leverage the one-sided RDMA operations for facilitating efficient admission control.

B. Fine-grained Resource Monitoring

Efficiently identifying the amount of resources used in data-center environments has been a critical research issue in the past several years. Traditionally, several techniques periodically monitor the resources used in the cluster and use this information to make various decisions such as admission decision, load-balancing, reconfiguration, etc. Many techniques rely on coarse-grained monitoring in order to avoid the overheads associated with fine-grained resource monitoring. However, on the other hand, the resource usage of requests is becoming increasingly divergent, thus increasing the need for fine-grained resource monitoring.

Efficient fine-grained resource monitoring approach attempts to achieve two goals: (i) to get an accurate picture of the current resource usage at very high granularity (in the order of milliseconds) and with low overhead and (ii) to be resilient to loaded conditions in a data-center environment. This approach uses RDMA operations to actively capture the resource usage of the back-end nodes and completely avoids TCP/IP communication to get this information. Due to the one-sided nature of RDMA operations, this approach helps in getting an accurate picture of resource usage, especially when back-end nodes are heavily loaded since RDMA operations

remove the communication overhead on the nodes that are being monitored. We encourage the readers to refer to our previous work [12] for detailed design and its associated benefits.

III. The Proposed Design

Admission control is a critical requirement for providing stable and feasible services in high-performance data-centers. It protects the servers and guarantees performance in the presence of overload by determining whether to accept new connections without jeopardizing the already established connections (services), i.e., admission control allows for graceful degradation in performance of the data-center servers during loaded conditions. In addition, it is also important for admission control mechanisms to handle the modern clustered multi-tiered data-center architectures.

In this section, we describe our design of the admission control and its implementation for multi-tier data-centers by leveraging the features of high-speed interconnects.

In our design we have the following main components: (i) *Admission control module*, (ii) *Load monitoring daemon* and (iii) *Load gathering daemon*. The *load gathering daemon* on the loaded servers are designed to collect the load information of the server. The external *admission control module* and the *load monitoring daemon* communicate with the *load gathering daemon* and provide the required services.

A. System Architecture

Figure 2 depicts the overall system architecture. While our approach is applicable to any of the tiers in a data-center, we show our design in the context of proxy server and web-server tiers. The *admission control module* is embedded into the proxy servers which use Apache to provide services. We can dynamically load this module in Apache (proxy server tier) and perform admission control on the requests forwarded to the web-server tier.

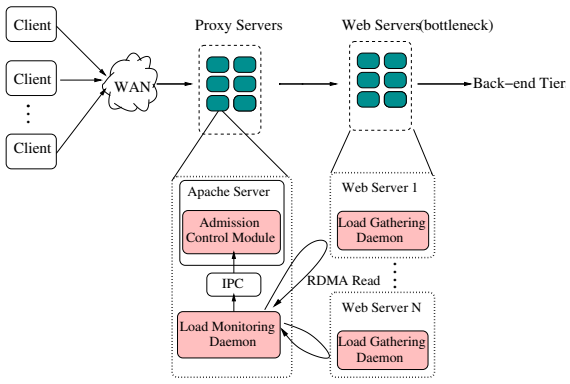

Fig. 2. System architecture

In our architecture, we apply this external admission control mechanism due to the following reasons. Firstly, it helps in making the global decisions based on the load information obtained from all back-end servers rather than that of a single server. Secondly, the admission control can be performed transparently to the back-end servers and requires no modification on their operating systems or applications. Thirdly, the proxy servers are usually less likely to get overloaded and can be easily and economically expanded to avoid being overloaded. It is also flexible to deploy them before any potentially overloaded tier without affecting the system functionality.

B. Load Monitoring and Load Gathering Daemons

Within the above framework, two parameters are necessary: the load limit (i.e. the maximum allowed load) and the load information of the protected servers. Given the load information (e.g., CPU utilization, memory utilization, network load or number of simultaneous socket connections etc.), the admission control decision is straightforward. In our design, a load monitoring daemon is running on the proxy server for fetching the load information from the back-end web servers. Correspondingly, a light-weight load gathering daemon is running on each of the web servers to collect the instantaneous load status of that server.

These two daemons themselves are not complicated. The critical part is to design effective and efficient communication between them. TCP/IP communication is often a traditional choice, but it is not efficient in terms of overhead and responsiveness in the overloaded conditions. In our design, the load monitoring daemon uses RDMA read operation to periodically fetch the load information from load gathering daemons. It can be achieved at a very high granularity and the back-end servers are not involved in explicit communications. Accordingly, the overhead is very low.

Two aspects are important for the performance of our admission control mechanism. One is the accuracy and granularity of the load information. Clearly, the system has better performance with more accurate and more frequent updates of load information. The other aspect is the extra load introduced by the admission control itself. If this is large, the performance may not improve and may even degrade. As stated above, our approach takes advantage of RDMA read which can get accurate load information at very high granularity and at the same time affects the loaded servers as little as possible. RDMA operation is especially beneficial when the resource bottleneck is the CPU, since it does not compete for this bottleneck resource. The benefits are shown in Section IV by comparing this approach to the similar scheme which instead uses TCP/IP protocol to transmit the load information.

C. Admission Control Module

In this section, we will give a detailed description of the admission control module.

The admission control module is responsible for making admission decisions, thus needs to interact with the

load monitoring daemon to get the load information. Since they are on the same proxy server, any IPC (Inter Process Communication) mechanism can be used. We utilize shared memory for this, due to its high efficiency and low overhead. It is to be noted that this interaction is asynchronous with the communication between the load monitoring daemon and the remote load gathering daemon, so that the load information can be read into the admission control module very quickly.

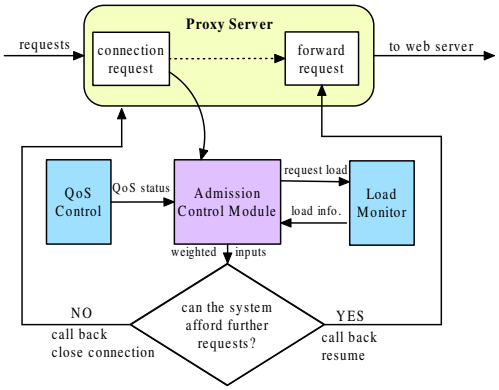

Fig. 3. Admission control process

Regarding the design, as mentioned earlier, the admission control module is implemented as a dynamically loadable Apache module. Apache provides a standard interface for adding the third-party modules for extending functionalities. It has internal handlers, hook functions and call-back functions for the ease of adding a new module. We add the admission control module through this standard programming interface.

Our module traps into the Apache request processing. It takes actions immediately after the TCP connection between the client and the proxy server is established. Generally, after the connection is established, Apache has several protocol modules for processing requests before forwarding them or generating appropriate replies. It reads the request, parses request header, checks the user ID or performs authentication etc. We drop the requests if required, at the earliest possible stage on the proxy server to minimize the involved overheads.

Further, as shown in Figure 3, when a request arrives, the Apache thread will call the admission control module after the TCP connection is established. The admission control module then gets the necessary information to make the decision. Current load status is certainly the most important information. Other information such as QoS requirements and previous latency or throughput can also be integrated to balance the decision. In our implementation, we use a simple policy that if acceptance of the arriving request does not exceed the load limit (e.g. the total TCP connections) of any web server, the admission control module returns the call back of OK and then the Apache thread proceeds; otherwise, it returns HTTP Service Temporarily Unavailable to notify the client and then close the connection. Note that other advanced decision making mechanisms can also be utilized. Such mechanisms are orthogonal to our design and can be completely complementary.

IV. Experimental Results

In this section, we present a comprehensive analysis of the experiment results, demonstrating the benefits of the design. We first show the basic micro-benchmark level evaluation of the protocols used on our test bed. Next, we study our admission control approach with several experimental and real traces. In all our experiments, the web-server tier will potentially get overloaded and the admission control is applied at the preceding proxy server tier, thereby, limiting the number of forwarded requests to better manage the overloaded web-server tier. It is to be noted that the proposed admission control can be applied to any of the tiers of the data-center as well as to other multi-tier applications.

Experimental Test bed: Our experimental test bed is a 32-node cluster. Each node is equipped with dual Intel Xeon 3.6 GHz processors and 2 GB memory, running RHEL4 U4 with the kernel 2.6.9.34. The cluster is equipped with IB DDR memfree MT25208 HCAs, and OFED 1.2 drivers are used. Apache 2.2.4 is used for web severs and proxy servers.

A. Micro-benchmarks

In this section, we show the basic micro-benchmark results that characterize our experimental test bed. Latency of the communication primitive (RDMA Read) used in our design is illustrated.

The latency achieved by the VAPI-level RDMA Read communication model and IPoIB (TCP/IP over IBA) for various message sizes is shown in Figure 4(a). RDMA Read achieves a latency of 5.2μs for 1 byte messages as compared to 18.9μs achieved by IPoIB. Further, with increasing message sizes, we see that the difference tends to increase.

In the following experiment, we demonstrate the benefits of RDMA operations over traditional TCP/IP-based communication protocols (here it is IPoIB) under loaded conditions. The emulated load is added and the RDMA Read test (emulating the process of fetching the load information) is performed from the proxy server to the loaded web server. Figure 4(b) shows that the performance of IPoIB degrades significantly with increase in background load. On the other hand, one-sided communication operations such as RDMA read show no degradation in performance. These results show the capability and effectiveness of one-sided communication primitives in the overloaded data-center environment, providing the support for using them in our admission control.

B. Data-Center-level Evaluation

In this section, we present the results for the data-center level experiments to verify the benefits of our design. The

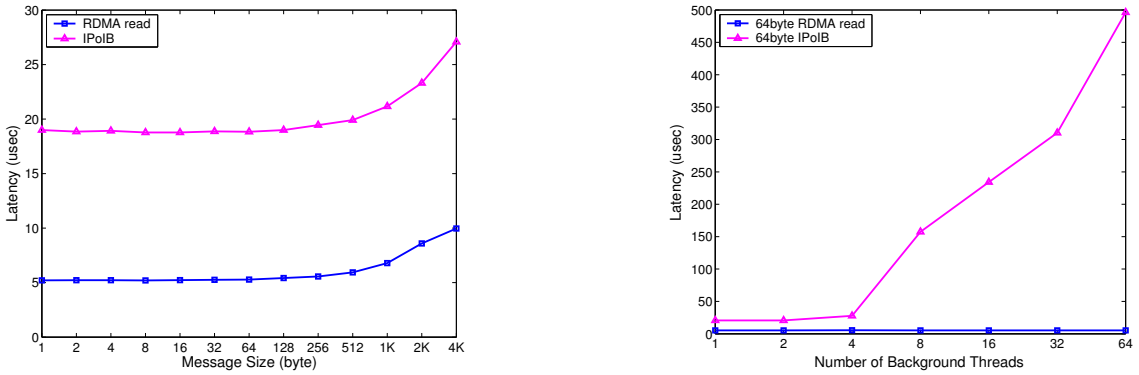

Fig. 4. Latency of basic RDMA read: (a)different message size (b) with background computation

workload is generated using a large number of clients and the average client-perceived response time is measured. In this context we utilize the following three load traces (i) single file trace, (ii) zipf like trace [16] and (iii) World Cup trace [4].

1) Performance with Single File Trace: In this experiment, we use the single file trace which contains only a single html document that is requested by several clients. This workload is used to study the basic performance achieved by the data-center environment for different file systems without being diluted by other interactions in more complex workloads. Since the typical static files accessed in data-centers is tens of kilo bytes, we use the 16 KBytes file in these experiments.

(i). Basic results

Figure 5(a) shows the average response time as a function of number of clients. The load monitoring granularity is 1 ms, i.e., the load information on the proxy server is updated every 1 ms. This granularity is high enough to capture the workload change if the load monitoring module is actually able to retrieve the information quickly enough. For each load configuration as shown by the x axis, we let the corresponding number of clients fire requests for about 30 minutes. The first two minutes period is considered as a warm-up stage and its data is excluded.

Results of three systems are shown: the system with admission control using RDMA Read, marked as *with AC (RDMA)*, the system with admission control using TCP/IP protocol, marked as *with AC (TCP/IP)* and the original system without admission control, marked as *No AC*. As shown in Figure 5(a), the average response time of the original system increases as the workload increases and starts to increase dramatically with more than 240 clients, which indicates the threshold of overload condition. However, admission control in the other two systems begins taking effect from this point since the improvement becomes apparent after 240 clients. Comparing the original system and the system with TCP/IP-based admission control, we see that the latter system has much better performance with the improvement varying from 24% to 44%, e.g., when the original system reaches the maximum response time of 192.31 ms, it has 142.29 ms in the same scenario (with 26% improvement). This difference demonstrates the necessity of admission control in overload conditions.

Further, with RDMA-based admission control approach, we find significantly more benefits as compared to the TCP/IP-based approach especially under extremely overloaded scenarios. The average response time sees a benefit of up to 28% (and up to about 50% as compared to the original system). This is because RDMA read doesn't require extra resources or CPU time from the overloaded web server and thus will not be delayed to update the load information on the proxy server, whereas the TCP/IP-based scheme may be stuck with this process. Hence accurate load information is always available in a timely manner for RDMA-based admission control and the corresponding decisions are more accurate and prompt.

Admission control guarantees the performance of the already accepted requests at the cost of dropping or throttling some other requests. In order to make a fair comparison of the above three systems, we also show the system aggregate TPS (Transactions Per Second) in Figure 5(b). It reflects the system performance taking the drop rate into account. We can see that the systems with admission control has higher TPS than the original system by about 15%, although they have more requests dropped. (In fact, the original system also has some drops due to the static constraint on web servers imposed by Apache itself.) It further verifies a well-known trait that it is better to serve lesser number of requests with acceptable performance instead of serving too many requests but with high impact on performance.

It is to be noted that even though both the TCP/IP-based admission control and the RDMA-based admission control observe similar drop rates explaining the benefits of admission control, the overall performance difference between them is due to the difference in the accuracy of the load information they depend on. In order to analyze

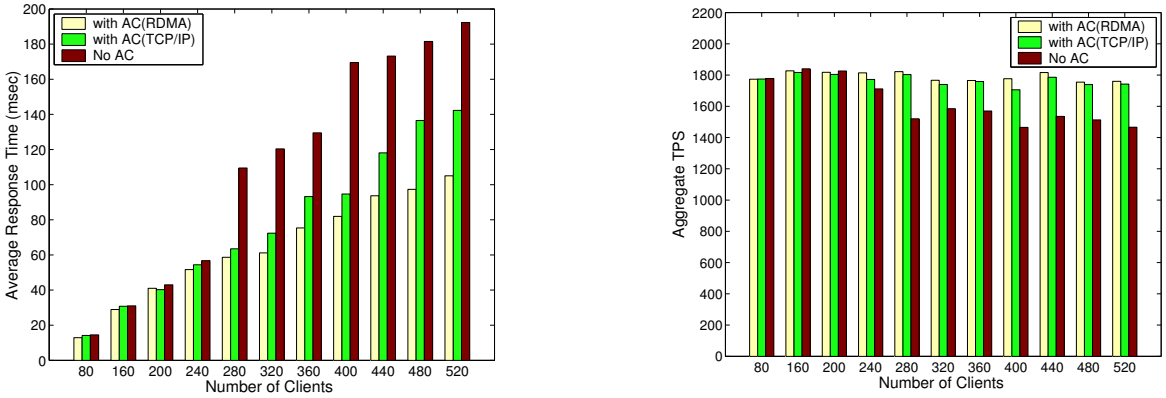

Fig. 5. Performance with single file (16KB): (a) average response time (b) aggregate TPS

this further, we perform the following experiments.

(ii). Analysis

Figure 6 presents the instant response times observed in the three systems during a small time window. We applied a 400 client workload. As shown in the figure, it is clear that the system without admission control has many requests with very high response times while the systems with admission control have much less such requests. This is more evident in RDMA-based admission control as almost all of the requests can be served with acceptable response time. For the TCP/IP-based admission control, the load update is not as accurate as the RDMA-based admission control and hence the admission control module sometimes reads the stale information and makes wrong admission decisions. As expected, its control on the response time is not as good as RDMA-based approach.

In order to confirm this we further observe the drop rates for a 100 second interval with the same workload. We average the drop rates across all of the clients every one second and illustrate them in Figure 7. As shown in the figure, the drop rate fluctuates rapidly using the RDMA-based admission control, which reflects the instantaneous changing load on web servers very accurately. Comparatively, the TCP/IP-based scheme shows longer streaks of continuous drops or continuous acceptances due to the delay in updating the load information. Finally, the system without admission control has a lot of acceptances even though the system has been overloaded.

As shown in figures 6 and 7, we see the real reactions of the three systems in the overload condition, demonstrating the cause for improvement achieved by the RDMA-based admission control shown earlier in Figure 5(a).

(iii). QoS

We had seen earlier in Figure 6 that there are relatively lesser requests served with longer response times in systems using admission control. From the QoS (Quality of Service) perspective, this indicates that the admission control can provide higher quality of service. In this section, we will analyze the QoS capabilities of these

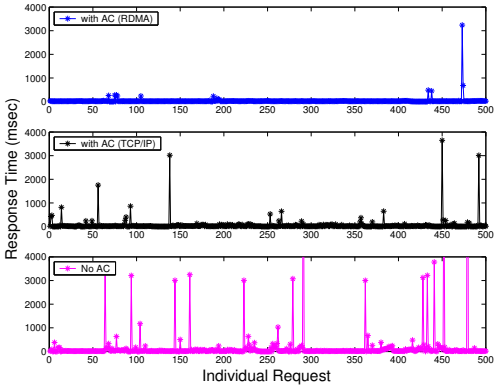

Fig. 6. Snapshot of instant response time

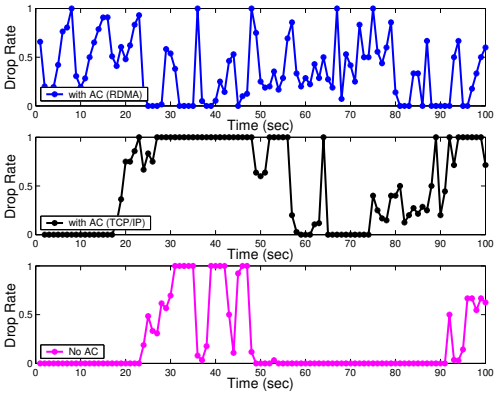

Fig. 7. Instant drop rate

systems.

A response time snapshot is shown in Figure 8. We use a log scale because of the large variation of values. A QoS threshold of 1 second is also shown in the figure. We can see that the system with RDMA-based admission control has much more capability of satisfying the QoS requirement. The systems with TCP/IP-based admission control and without admission control have a lot of unsatisfactory requests whereas the system with RDMA-based admission control shows almost no unsatisfactory

requests.

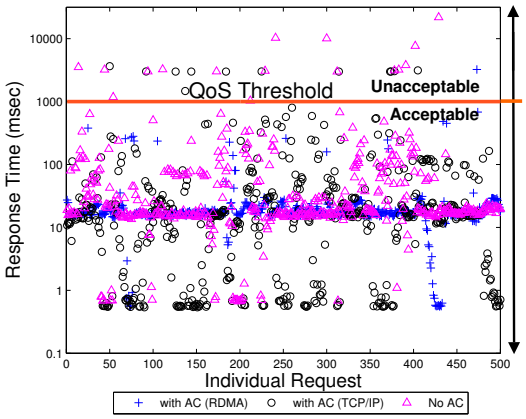

Fig. 8. Variations in instant response time

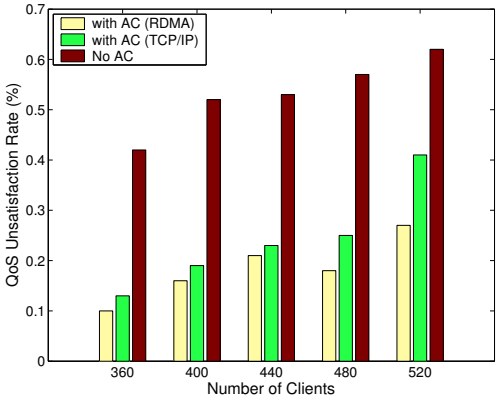

Fig. 9. Overall QoS for varying workloads

While the above trend is illustrated for a small time period, we average them over the entire duration of the experiments and present the results in Figure 9. presents the overall percentage of the requests which cannot meet the QoS requirements with varying workload. We see that the system with RDMA-based admission control shows significantly better QoS as compared to the system without admission control and marginal improvements as compared to the system with TCP/IP-based admission control for heavily overloaded scenarios. On the other hand, with the same requirement of average response time, the system with RDMA-based approach seen earlier is capable of serving much more clients than the other two systems. Thus, RDMA-based admission control has a better QoS control.

2) Performance with Zipf and World Cup Traces: We further demonstrate the benefits of our design using two widely used traces; the zipf trace and the world cup trace. It has been well acknowledged in the community that most workloads follow a Zipf-like distribution for static content, i.e., the relative probability of a request for the i'th most popular document is proportional to $1/i^{\alpha}$, where α is a factor that determines the randomness of file accesses. In our experiment, we use the zipf trace with a high α (α=0.9) for evaluation. The world cup trace is extracted from the real data during the World Cup 1998.

Due to the space limitation, here we present the performance only in terms of average response time. The results with Zipf trace and world cup trace are shown in Figures 10(a) and 10(b), respectively. We see similar trends as seen with the single file trace. The system with RDMA-based admission control has the best performance and the benefits increase with increasing workload. For 520 clients workload, we see that RDMA-based admission control outperforms the system with TCP/IP-based admission control and the system without admission control by more than 23% and 42%, respectively with Zipf trace, and 17% and 36%, respectively with world cup trace.

V. Discussion and Related Work

Addressing the web server overload has been an important issue for data-centers since the explosive development of the internet, and many mechanisms have been proposed.

The simplest way for overload control is resource containment. A predefined resource limit is an internal system parameter with which the server keeps accepting requests until the resource consumption exceeds the limit. This whole process is static and rigid which well-known applications like Apache employ. It bounds the number of service threads and stops spawning more when it reaches the maximum limit. Consequently, the incoming requests will wait until more threads become available again. The waiting time can be unbounded due to TCP's exponential backoff on SYN retransmission and the performance can degrade as shown in Section IV.

The second mechanism focuses on shedding some amount of workload to maintain the performance for the existing clients. Many of these techniques rely on an explicit control to bound the resource consumption or request rates according to the system load. [14] describes an adaptive approach to bound the 90th-percentile response time in the context of SEDA web server in which admission control is performed on several stages. Cherkasova and Phaal [8] proposed to perform admission control on sessions instead of requests in order to increase the successful sessions. Our RDMA-based admission control mechanism complements these approaches.

The third mechanism is the service differentiation. It differentiates the classes of clients so that the high-priority clients are not subjected to the service degradation under overloaded scenarios. [13] proposed a kernel-based approach that controls the socket listen queue by request URL and client IP address. Our work can be extended to provide this service and RDMA operations can be further exploited for performance guarantees.

There are also many other schemes such as service degradation and control theory application. Actually, the

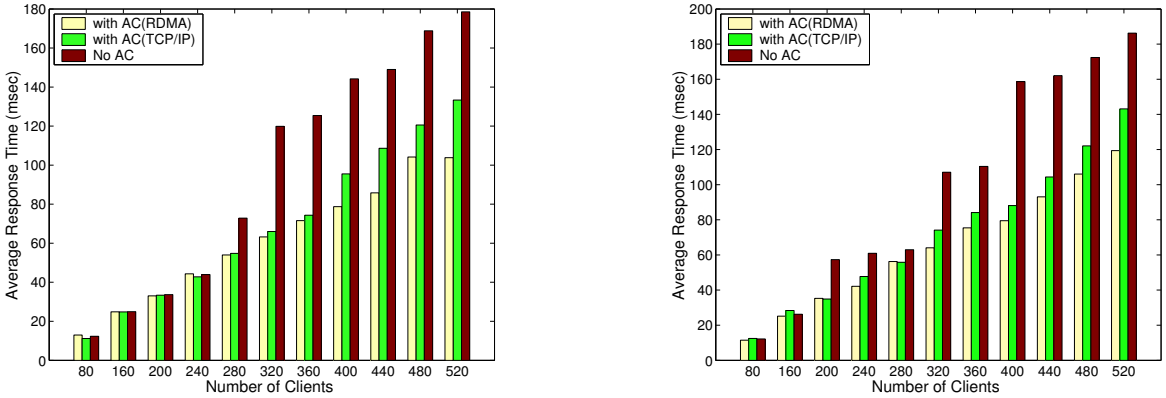

Fig. 10. Average response time for: (a) Zipf trace (b) World Cup trace

overload control mechanism can not be strictly classified as above. Much work has integrated different mechanisms into a combined one. Bhatti and Friedrich [5] has proposed a framework supporting tiered web services to differentiated clients through admission control, request classification and request scheduling. Bhoj et al. [6] presented a Web2K server which performs admission control based on the accept queue length and the arrival and service rates of a particular client class.

Apart from these existing works, our efforts mainly focus on leveraging the features of modern network interconnects to improve the design and implementation rather than on designing new algorithm for admission control. As our experimental results in Section IV indicate, we believe it is meaningful and promising to extend and improve these existing techniques in the context of multi-tier data-centers over high-speed interconnects.

VI. Conclusions and Future Work

In this paper, we leveraged the RDMA features of high-speed interconnects in designing an efficient admission control mechanism and consequently provided superior performance, response time guarantees and overload control in a data-center environment. Our design is implemented over InfiniBand-based clusters working in conjunction with Apache based servers. Experimental evaluations with single file, worldcup and zipf traces showed that our admission control can improve the response time by up to 28%, 17% and 23%, respectively as compared to performing admission control using the TCP/IP communication and 51%, 36% and 42%, respectively, as compared to base performance without any admission control. Further, our evaluations also showed that the system with DMA-based admission control mechanism can provide better QoS guarantees as compared to the system with TCP/IP-based admission control and that without any admission control.

As a part of future work, we propose to utilize RDMA operations in kernel space in order to use other metrics such as the number of pending interrupts to perform more efficient admission control. We further propose to improve and extend other sophisticated overload control algorithms by exploiting more of the advanced features of modern interconnects. We also plan to include our earlier work on reconfiguration and resource monitoring to provide an integrated resource management service for data-centers.

References

[1] Chelsio Communications Inc. . `http://www.chelsio.com/`.

[2] InfiniBand Trade Association. http://www.infinibandta.com.

[3] S. Adler. The Slashdot Effect, An Analysis of Three Internet Publications. In *Linux Gazette*, March 1999.

[4] The Internet Traffic Archive. http://ita.ee.lbl.gov/html/traces.html.

[5] N. Bhatti and R. Friedrich. Web server support for tiered services. *IEEE Network*, 13(5):64–71, 1999.

[6] P. Bhoj, S. Ramanathan, and S. Singhal. Web2K: Bringing QoS-aware web servers. In *Proceedings of the IEEE International Conference on HPDC*, June 2002.

[7] Jeffrey S. Chase, Darrell C. Anderson, Prachi N. Thakar, Amin Vahdat, and Ronald P. Doyle. Managing Energy and Server Resources in Hosting Centres. In *Symposium on Operating Systems Principles*, 2001.

[8] L. Cherkasova and P. Phaal. Session based admission control: a mechanism for overloaded commercial web server. In *MASCOTS*, 2000.

[9] S. Elnikety, E. Nahum, J. Tracey, and W. Zwaenepoel. A method for transparent admission control and request scheduling in E-commerce web sites. In *Proceedings of the 13th World Wode Web Conference*, May 2004.

[10] W. LeFebvre. CNN.com: Facing a World Crisis, 2002.

[11] D. Patterson. Availability and Maintainability >> Performance: New Focus for a New Century, 2002.

[12] K. Vaidyanathan, H. W. Jin, and D. K. Panda. Exploiting RDMA operations for Providing Efficient Fine-Grained Resource Monitoring in Cluster-based Servers. In *Workshop on Remote Direct Memory Access (RDMA): Applications, Implementations, and Technologies (RAIT), in conjunction with Cluster Computing September*, 2006.

[13] T. Voigt, R. Tewari, D. Freimuth, and A. Mehra. Kernel mechanisms for service differentiation in overloaded Web servers. In *Proceedings of USENIX Annual Technical Conference*, June 2002.

[14] M. Welsh, D. Culler, and E. Brewer. SEDA: An Architecture for Well-Conditioned, Scalable Internet Services. In *the Eighteenth Symposium on Operating Systems Principles (SOSP-18)*, 2001.

[15] J. Yin, L. Alvisi, M. Dahlin, and A. Iyengar. Engineering Web Cache Consistency. *ACM Transactions on Internet Technology, 2:3,*, August. 2002.

[16] George Kingsley Zipf. Human Behavior and the Principle of Least Effort. Addison-Wesley Press, 1949.