Journal of Network and Computer Applications 32 (2009) 642–651

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

An efficient design for fast memory registration in RDMA

Li Ou [a], Xubin He [b,*], Jizhong Han [c]

[a] DELL Inc., USA
[b] Electrical and Computer Engineering Department, Tennessee Technological University, Cookeville, TN 38505, USA
[c] Institute of Computing Technology, Chinese Academy of Sciences, China

ARTICLE INFO

Article history:
Received 7 October 2007
Received in revised form
26 March 2008
Accepted 5 July 2008

Keywords:
RDMA
Memory registration
Cache
Performance evaluation

ABSTRACT

Remote Direct Memory Access (RDMA) improves network bandwidth and reduces latency by eliminating unnecessary copies from network interface card to application buffers, but the communication buffer management to reduce memory registration and deregistration cost is a significant challenge to be addressed. Previous studies use pin-down cache and batched deregistration, but only simple LRU is used as a replacement algorithm to manage cache space. In this paper, we evaluate the cost of memory registration in both user and kernel spaces. Based on our analysis, we reduce the overhead of communication buffer management in two aspects simultaneously: utilize a Memory Registration Region Cache (MRRC), and optimize the RDMA communication process of clients and servers with Fast RDMA Read and Write Process (FRRWP). MRRC manages memory in terms of memory region, and replaces old memory regions according to both their sizes and recency. FRRWP overlaps memory registrations between a client and a server, and allows applications to submit RDMA write operations without being blocked by message synchronization. We compare the performance of MRRC and FRRWP with traditional RDMA operations. The results show that our new design improves the total cost of memory registrations and overall communication latency by up to 70%.

1. Introduction

The advent of networking technologies facilitates the service of storage over networks. Remote Direct Memory Access (RDMA) is emerging as the central feature in modern network interconnects. It offers low latency, high throughput, and low CPU overhead communication in network storage systems. These enabling technologies eliminate or reduce costs of memory copy, network access, interrupt, and protocol processing in the network subsystem. Interconnects like InfiniBand (Infiniband Trade Association, 2000), Myrinet (Boden et al., 1995), and Quadrics (Petrini et al., 2001) have long introduced RDMA in LAN environments. RDMA over IP has been developed to extend the benefits of RDMA across the WAN/Internet. The RDMA Consortium has proposed the RDMA Protocol Verbs Specifications (RDMAVS 1.0) (Hilland et al., 2003) to standardize the efforts.

While RDMA improves network bandwidth and decreases latency by eliminating unnecessary copies from network interface cards (NICs) to application buffers, a number of challenges must be addressed. One of the most significant issues is efficient communication buffer management to reduce memory registration and deregistration costs. Previous research (Bell and Bonachea, 2003; Rangarajan and Iftode, 2004; Tezuka et al., 1998; Wu et al., 2003a, b; Zhou et al., 2002) shows that memory registration is an expensive operation since it requires pinning of pages in physical memory and accessing the on-chip memory of the NIC, such as InfiniBand Host Control Adapter (HCA) and RNIC of RDMA over IP. Experimental results (Tezuka et al., 1998) from Myrinet and an extremely old Pentium Pro machine (200 MHz) show that one memory page transfer (4 KB) only takes 25.6 μs while the memory registration cost is approximately 26 μs. Even with a much faster configuration (InfiniBand HCA and Intel Xeon 2.4 GHz processor) (Wu et al., 2003b), the registration of a memory page still costs about 7 μs, almost the same as the transfer time for that page. The cost and overhead of memory registration dramatically degrade the performance of RDMA and increase network latency in the critical data path of I/O operations.

Several attempts (Bell and Bonachea, 2003; Rangarajan and Iftode, 2004; Tezuka et al., 1998; Wu et al., 2003a, b; Zhou et al., 2002) have been made to reduce the overhead of memory registration in RDMA. In some special environments (Bell and Bonachea, 2003; Liu et al., 2003; Wu et al., 2004), the memory region used by applications is predefined and can be preregistered in the initialization phase; thus in the critical path of data transferring, memory registration is not necessary. In general applications, since dynamic registration and deregistration cannot

* Corresponding author. Tel.: +1 931 372 3462; fax: +1 931 372 3436.
E-mail addresses: li_ou@dell.com (L. Ou), hexb@tntech.edu (X. He), hjz@ict.ac.cn (J. Han).

1084-8045/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jnca.2008.07.008

be avoided, a pin-down cache (Tezuka et al., 1998) is incorporated in the memory manager. A pin-down cache delays deregistration of registered buffers and caches their registration information for future accesses of the same memory region. Several cache designs for memory registration (Rangarajan and Iftode, 2004; Wu et al., 2003a) are proposed based on the pin-down cache to take advantage of temporal locality of memory accesses of RDMA. Current memory registration caches manage memories at the page level and only consider LRU as the replacement algorithm. Most applications using RDMA register and deregister memory regions containing multiple continuous or noncontiguous memory pages; thus, page level management for registration caches is not efficient enough. Furthermore, with multiple page memory regions, the locality of memory accesses is also changed, and the general LRU algorithm is probably not the best choice.

In this paper, we evaluate the cost of memory registration in both user and kernel spaces. We analyze latency of memory registration and find three main parts which contributes most to the total costs. Based on our analysis, we reduce the overhead of memory registration in two aspects simultaneously: utilize a memory registration cache, and optimize the RDMA communication process of clients and servers.

We propose a new cache management scheme, Memory Registration Region Cache (MRRC), to minimize the cost of memory registration and deregistration in the critical data path. MRRC manages memory in terms of memory regions, which contain one or more memory pages, and considers pipelining between RDMA operations and memory registrations. MRRC organizes the cache stack using the LRU algorithm, but divides the stack into three sections and evicts memory regions from the *eviction section* according to both the size and recency.

We then propose a new communication scheme between an RDMA client and server, Fast RDMA Read and Write Process (FRRWP), to minimize the cost of memory registration in the critical data path. FRRWP re-schedules the communication process of RDMA to overlap memory registrations between the client and the server. It allows issues of RDMA operations without being blocked by the synchronization messages: the applications may submit an RDMA write immediately after they finish local memory registrations, without waiting for the confirmation of registrations from the peer node.

The performance of MRRC is compared with traditional RDMA memory registration operations and other typical registration cache management algorithms such as pin-down cache (Tezuka et al., 1998) and FMRD (Wu et al., 2003a). The results show that compared to traditional RDMA memory registration, MRRC improves the total cost of memory registrations by up to 70%. We compare the latency of FRRWP with traditional RDMA operations using a mathematic model. The results show that FRRWP reduces the total communication latency in the critical data path by 68%.

The rest of the paper is organized as follows. Background material is presented in Section 2. Section 3 examines related work. Section 4 evaluates the cost of memory registration in both user and kernel space. Sections 5 and 6 describe the design issues of MRRC and FRRWP in detail, respectively. MRRC is compared to the previous efforts to improve RDMA performance in Section 7. The latency of FRRWP is analyzed in Section 8. Section 9 draws the conclusions.

2. Background review

In RDMA, an NIC (RNIC, RDMA NIC) or InfiniBand HCA writes or reads user specified buffers directly without unnecessary copies, so before each RDMA operation, it is required to register a memory region where the user buffers are located. In the process of registration, the device driver first maps the virtual memory address to the physical address, then pins the memory region to make sure that in the operations of RDMA, the memory region is not swapped out from physical memory. After mapping and pinning, the driver reports the information of the memory region to NIC, in which a table is used to keep information of all registered memory regions. A memory region cannot be pinned forever; otherwise, the effective size of physical memory used for other purpose is reduced. On the other side, the number of entries in the registration table is limited. When the number of registered buffers exceeds this limit, the application needs to deregister memory and free resources on the NIC, which involves the unpinning of the memory region and remove the entry from the table. Memory registration and deregistration are time-consuming operations.

The cost of memory registration and deregistration varies with the performance of hosts. For instance, in a pretty old Pentium Pro machine (200 MHz), one memory page (4 KB) registration takes 26 μs (Tezuka et al., 1998), while the same operation only need 7 μs with a much faster Intel Xeon 2.4 GHz processor (Wu et al., 2003b). Although high performance servers reduce time of memory registrations, the cost is still almost same as the network latency of the contemporary interconnect used by servers (Tezuka et al., 1998; Wu et al., 2003b). If every RDMA operation is blocked by the registration and deregistration, the overhead is very large and overall communication latency is very high. Previous studies (Tezuka et al., 1998; Wu et al., 2003b) show that without any optimization, the RDMA performance is hurt by the memory registration and deregistration so much that even the traditional send and receive operations, which involve several memory copies, could outperform RDMA if the message size is small. Experiments (Tezuka et al., 1998; Wu et al., 2003b) show that if the message size of most operations is less than 1 K, RDMA with normal memory registration may not provide better performance than the traditional way, and in some cases, even worse.

The simple solution of pre-registering all buffers at application startup is not general and cannot be applied in most systems, since they use large caches and require large amount of memory buffers. Dynamic memory registration is not avoided if applications keep using different buffers. To improve RDMA performance, it is very important to reduce the overhead of memory registration and deregistration.

3. Related work

Several studies have improved the performance of memory registration and deregistration of RDMA. Tezuka et al. (1998) propose a *pin-down cache* for Myrinet. A pin-down cache delays the deregistration of registered buffers and caches their registration information for future accesses of the same memory region. LRU is used as a replacement algorithm in a pin-down cache to manage memory registrations. Zhou et al. (2002) eliminate pinning and unpinning from the registration and deregistration paths by combining memory pinning and allocation together. They also demonstrated that *batched deregistration* is an efficient way to reduce the average cost of deregistration memory. Wu et al. (2003a), propose a two-level architecture, *FMRD*, for memory registration by adopting both a pin-down cache and batched deregistration. *FMRD* also takes advantage of the Mellanox fast memory region registration extension in VAPI (Mellanox Technologies, 2003). Based on the pin-down cache, a *lazy cache* is proposed in Rangarajan and Iftode (2004), which combines a cache of registration mappings with a lazy approach to memory deregistration. The *lazy cache* is implemented using an