Dynamic Page Migration in Software DSM Systems

Thomas Repantis[1] Christos D. Antonopoulos[2] Vana Kalogeraki[1] Theodore S. Papatheodorou[2]

[1] Department of Computer Science & Engineering
University of California, Riverside
Riverside, CA 92521, USA

[2] Computer Engineering & Informatics Department
University of Patras
26500 Patras, Greece

http://www.cs.ucr.edu/~trep/

Abstract

We introduce a protocol for dynamically migrating memory pages in home-based Software DSM systems. In these systems each page has a designated home node; yet our protocol allows a node that heavily modifies a page to become its new home. The process is dynamic and totally transparent to the applications programmer. The benefits of our page migration mechanism include the reduction of remote page modifications, faster memory accesses, and less communication overhead.

Introduction

- Computer clusters are a flexible and inexpensive platform for high performance computing.
- Using the architecture of Distributed Shared Memory (DSM) in computer clusters makes the work of the application programmer easier. A Software DSM library provides the notion of a shared address space to the programmer, although application threads may execute on different cluster nodes with physically distributed memory.

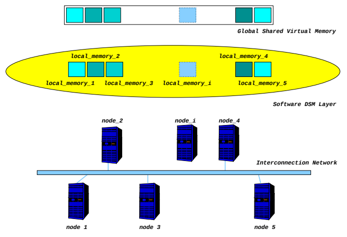


Figure 1: The Software DSM layer provides the applications with the illusion of shared memory.

- In home-based Software Distributed Shared Memory systems each memory page has a home:
  - The home gathers all page modifications (diffs).
  - All page access faults can be resolved with one request to the home of the particular page.
  - The performance however depends greatly on the distribution of pages across nodes, with respect to the memory access pattern of the application. Remote page accesses incur extra communication cost.

Motivation

Dynamic page migration (explicitly moving the home of a page from some node to another at run-time) offers several advantages when employed in DSMs:

- Reduces the average latency of memory accesses by increasing the locality.
- Enables the system to adapt to the applications' memory access patterns.
- Improves resource utilization. It allows the system to consider the computational and communicational needs of the applications and also to adapt to changing resource availability.
- Achieves the above with lower overhead than traditional approaches that rely on thread migration.

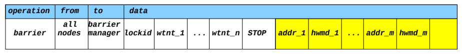
The Dynamic Page Migration Protocol

We propose a simple and efficient page migration mechanism, that dynamically allocates shared memory pages to home nodes:

- Each page has an initial, designated home node.
- Nodes that heavily modify the pages can become their new homes.
- The migration decision is taken locally, at the home of each page, according to the sizes of the diffs that have been created for each page by the nodes that modified it.
- To avoid redundant page transfers, we perform migration only when the number of modifications of a page by a remote node becomes higher than a threshold.

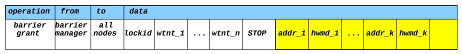
Propagation of the Migration Decisions

- The migration information is piggybacked on the existing synchronization (barrier) messages to minimize the communication overhead. Thus our implementation targets a DSM that uses scope-consistency. However this does not affect the generality of our approach.
- Each node arriving at a barrier sends its migration decisions to the barrier manager.
- The barrier manager then propagates the collected migration decisions to all nodes.
- Thus the page tables of all nodes are updated before any thread is released from the barrier.
- Pages are transferred from the old to the new homes, if the latter don't have an accurate copy already (if they were not the single modifiers during the last barrier interval).
- Only after a migrating page has arrived, does the new home refer to it. The old home does not unmap a migrating page, before it is sent to its new home.



operation	from	to	data										
barrier	all nodes	barrier manager	lockid	wtnt_1	...	wtnt_n	STOP	addr_1	hwmd_1	...	addr_m	hwmd_m	

Figure 2: The modified barrier request message, incorporating the addresses of the pages that will migrate from this home, as well as the hosts with the maximum sizes of applied diffs, which will become the pages' new homes.



operation	from	to	data										
barrier grant	barrier manager	all nodes	lockid	wtnt_1	...	wtnt_n	STOP	addr_1	hwmd_1	...	addr_k	hwmd_k	

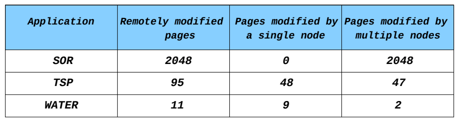
Figure 3: The modified barrier grant message, incorporating all hosts' migration decisions, which were gathered by the barrier manager.

Experimental Setup

- We have implemented our mechanism in the JIAJIA software DSM and evaluated it using real application benchmarks.
- – 4 Intel Pentium III nodes, 256KB cache, 256MB RAM each.
  - GigaBit Ethernet
  - Linux 2.4.18, gcc 3.2.2
- We have compared our mechanism to JIAJIA's home migration protocol, where a page moves to a new home only if that was the only modifier during the last barrier interval.

Applicability of Dynamic Page Migration Protocol

- For several applications there are pages that are remotely modified by multiple nodes.



Application	Remotely modified pages	Pages modified by a single node	Pages modified by multiple nodes
SOR	2048	0	2048
TSP	95	48	47
WATER	11	9	2

Figure 4: Number of pages that are remotely modified for some applications.

- In such cases a migration protocol that relies on detecting a single modifier pattern will not be triggered and will offer no optimization.

Preliminary Performance Evaluation

We have compared the performance of the dynamic page migration protocol to the execution without migration and to JIAJIA's home migration protocol, for a variety of metrics.

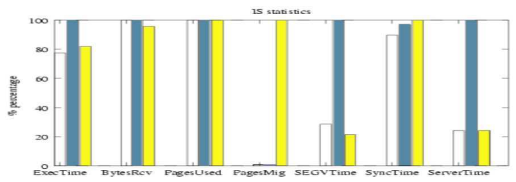


Figure 5: Performance of bucket sort (white: no migration, blue: JIAJIA's home migration protocol, yellow: the new, dynamic page migration protocol).

Our dynamic page migration protocol:

- Reduces remote page modifications.
- Reduces transfers of pages and diffs (bytes received).
- Improves average memory access latencies (time spent in serving segmentation violations).

Conclusions and Future Work

- Dynamic page migration, when employed in Software DSM systems, reduces remote page modifications.
- That results to faster memory accesses and less communication overhead.
- The cost of executing the algorithm and of migrating the pages is amortized by the benefits gained.

Our future work includes:

- Extensive experimental evaluation of our protocol.
- Study and evaluation of more elaborate migration policies.