

SBMA

Preliminaries

BDMPI

Overview

SBMA

Motivation Hypothesis and key question SBMA framework

Results

Benchmarks

Experimental setu

Experiments

Conclusions

A Memory Management System Optimized for BDMPI's Memory and Execution Model

Jeremy Iverson¹ George Karypis¹

¹University of Minnesota, Minneapolis, MN, USA

EuroMPI 2015

October 4, 2015



Consider this parallel application





One *simple* solution





A more realistic solution



Preliminaries

BDMPI

Overview

SBMA

Motivation Hypothesis and key question SBMA framework

Results

Denchmarks

Experimental setu

Experiments

Conclusions







But what if hardware is fixed?





Let's look at a serial application





Let's look at a serial application





Now recall the parallel application...





... and apply the serial solution





Remember the more realistic solution?

SBMA

Preliminaries

BDMPI

Overviev

SBMA

Motivation Hypothesis and key question SBMA framework

Results

Dencimarks

Experimental setu

Experiments

Conclusions







What if we could just...





Enter BDMPI

SBMA

BigData MPI (BDMPI)

• Transparent layer between an MPI application and an MPI runtime

Node-level co-operative multi-tasking (execution model)

- MPI process will run until it blocks for a communication operation (collective, recv)
- Cost of loading data from disk is amortized over large segments of computation

Constrained memory over-subscription (memory model)

- Assumes the problem is decomposed s.t. each MPI process can fit its working set in memory
- Manages the scheduling of MPI processes per compute node to reduce pressure on OS swapping mechanism

BDMPI

Overview

SBMA

Motivation Hypothesis and key question SBMA framework

Results

Benchmarks

Experimental setu

Conclusions



Enter BDMPI

SBMA

BigData MPI (BDMPI)

• Transparent layer between an MPI application and an MPI runtime

Node-level co-operative multi-tasking (execution model)

- MPI process will run until it blocks for a communication operation (collective, recv)
- Cost of loading data from disk is amortized over large segments of computation

Constrained memory over-subscription (memory model)

- Assumes the problem is decomposed s.t. each MPI process can fit its working set in memory
- Manages the scheduling of MPI processes per compute node to reduce pressure on OS swapping mechanism

BDMPI

Overview

SBMA

Motivation Hypothesis and key question SBMA framework

Results

Benchmarks

Experimental setu

.



Enter BDMPI

SBMA

BigData MPI (BDMPI)

• Transparent layer between an MPI application and an MPI runtime

Node-level co-operative multi-tasking (execution model)

- MPI process will run until it blocks for a communication operation (collective, recv)
- Cost of loading data from disk is amortized over large segments of computation

Constrained memory over-subscription (memory model)

- Assumes the problem is decomposed s.t. each MPI process can fit its working set in memory
- Manages the scheduling of MPI processes per compute node to reduce pressure on OS swapping mechanism

BDMPI

Overview

SBMA

Motivation Hypothesis and key question SBMA framework

Results Benchmark

Experimental set

Experiments

Conclusions



Overview

SBMA



Overview

SBMA

- Motivation
- Hypothesis and key question
- SBMA framework

Results 2

- Benchmarks
- Experimental setup
- Experiments





Pitfalls of OS swapping in BDMPI





$\label{eq:posterior} \mbox{Pitfalls of OS swapping in BDMPI}$





$\label{eq:posterior} \mbox{Pitfalls of OS swapping in BDMPI}$





Pitfalls of OS swapping in BDMPI





Let's back up...



rank 1 compute

rank 1 comm



Let's back up...





... and reduce disk contention





Important perspective

SBMA

Preliminaries

BDMPI

Overview

SBMA

Motivation

Hypothesis and key question SBMA framework

Results

Benchmarks

Experimental setu

Experiments

Conclusions

Hypothesis

• Exploiting the BDMPI memory and execution models will lead to reduced disk contention compared with deferring to the OS VMM

Key question

• How aggressively should a process' virtual address space be exchanged between physical memory and disk to maintain to prevent memory over-subscription?



Important perspective

SBMA

Preliminaries

BDMPI

Overview

SBMA

Motivation

Hypothesis and key question SBMA framework

Results

Benchmarks

Experimental setu

Experiments

Conclusions

Hypothesis

• Exploiting the BDMPI memory and execution models will lead to reduced disk contention compared with deferring to the OS VMM

Key question

• How aggressively should a process' virtual address space be exchanged between physical memory and disk to maintain to prevent memory over-subscription?



SBMA Overview

SBMA

Preliminaries

- BDMPI
- Overview

SBMA

- Motivation
- Hypothesis and key question

SBMA framework

Results

- Benchmarks
- Experimental setup

C. . . I

What it is. . .

- Storage-Backed Memory Allocation (SBMA)
- Built as part of the BDMPI library
- User space virtual memory manager

How it works...

- Uses C interposition to fulfill applications' memory allocation requests
- Relies on memory protection and signal handling to track status of allocated pages



An illustrative example





- BDMPI
- Overview

SBMA

- Motivation Hypothesis and question
- SBMA framework
- Results
- Benchmarks
- Experimental setu
- Conclusions



free(arr);



Memory access patterns

SBMA

Preliminaries BDMPI Overview

SBMA

Motivation

Hypothesis and key question

SBMA framework

Results

Benchmarks

Experimental setup

Experiments

Conclusions











Benchmarks

SBMA

Renchmarks

Synthetic

- Sequence of reads and writes
- Used to quantify the overhead introduced by the SBMA library

PageRank

- Memory footprint fixed
- Multiplying a sparse matrix by a vector

ParMetis

- Memory footprint changes throughout execution
- Recursively contracting a graph

SPLATT

- Memory footprint fixed, but has different phases requiring different amounts of memory
- Multiplying a sparse tensor and dense matrices



Experimental setup

SBMA

Experimental setup

System

• Four machine cluster with an aggregate 16GB DRAM and 1.2TB swap

Datasets

- Synthetic dynamically generated random data (4GB in memory)
- PageRank 6.6B edges, ordered randomly (35GB in memory)
- ParMetis 760M edges (13GB in memory)
- SPLATT 2.9M×2.1M×25.5M with 143.6M non-zeros (26GB in memory)



Synthetic benchmark

SBMA

eliminaries DMPI		Read(OS	x == y) SBMA	Write OS	(x = y) SBMA	Read/W OS	rite (x += y) SBMA
verview	AI	1195	1194	514	373	472	352
otivation	LI	1195	927	514	325	472	310
pothesis and key estion	AR	28	28	514	373	28	28
3MA framework	LR	30	30	514	325	30	30
PSUITS enchmarks			Throu	ighput (s	system page	es/sec)	

Experiments

- A Aggressive
- L Lazy
- I In-memory
- R On disk



Synthetic benchmark

SBMA

Experiments

		Read (x == y)		Write $(x = y)$		$Read/Write (x \mathrel{+}= y)$	
API		OS	SBMA	OS	SBMA	OS	SBMA
rview MA	AI	1195	1194	514	373	472	352
ivation	LI	1195	927	514	325	472	310
othesis and key tion	AR	28	28	514	373	28	28
1A framework ults	LR	30	30	514	325	30	30

Throughput (system pages/sec)

- A Aggressive
- L Lazy
- I In-memory
- R On disk



Synthetic benchmark

SBMA

eliminaries DMPI		Read(OS	x == y) SBMA	Write OS	(x = y) SBMA	Read/Wi OS	rite (x += y) SBMA
verview RMA	AI	1195	1194	514	373	472	352
otivation	LI	1195	927	514	325	472	310
pothesis and key estion	AR	28	28	514	373	28	28
3MA framework	LR	30	30	514	325	30	30
SUITS enchmarks			Throu	ughput (s	system page	es/sec)	

Experimental setup

Experiments

- A Aggressive
- L Lazy
- I In-memory
- R On disk



Real world benchmarks





Conclusions

SBMA

What we've learned

- Preliminaries
- BDMPI
- Overview

SBMA

- Motivation Hypothesis and key question SBMA framework
- Results
- Benchmarks
- Experimental setup
- Experiments
- Conclusions

- Possible to implement a user space virtual memory manager with less a $2\times$ slowdown in memory throughput
- Exploiting BDMPI's execution and memory models improves performance over OS VMM with speedups from $2\times$ to $12\times$



Conclusions

SBMA

What we've learned

- Preliminaries
- BDMPI
- Overview

SBMA

- Motivation Hypothesis and key question SBMA framework
- Results
- Benchmarks
- Experimental setu
- Experiments
- Conclusions

- Possible to implement a user space virtual memory manager with less a $2\times$ slowdown in memory throughput
- Exploiting BDMPI's execution and memory models improves performance over OS VMM with speedups from $2\times$ to $12\times$

Moving forward

- Add support for MPI+X
- Allow more than one process to run simultaneously on each compute node so long as memory constraint is not violated



Thank you

SBMA	
BDMPI	
Overview	
	Questions?
Motivation	Questions.
Hypothesis and key question	
SBMA framework	jiverson@cs.umn.edu
	5
Benchmarks	
Experimental setup	http://glaros.dtc.umn.edu/gkhome/bdmpi/download
Experiments	
Conclusions	