MPI-RCDD: A Framework for MPI Runtime Communication Deadlock Detection

Hongmei Wei[1], Jian Gao[2*], Peng Qing[2] , Kang Yu[2] , Yanfei Fang[2] and Minglu Li[1]

(1. Department of Computer Science & Engineering, Shanghai jiaotong University, Shanghai 200240, China)
(2. Jiangnan Institute of Computing Technology, Wuxi 214083, China)

Wei HM, Gao J, Qing P et al. MPI-RCDD: A framework for MPI runtime communication deadlock detection. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 35(2): 395–411 Mar. 2020. DOI 10.1007/s11390-020-9701-4

Research Background

- The Message Passing Interface (MPI) has become a de facto standard for programming models of high performance computing (HPC)
- Establishing communication deadlock-freedom in MPI programs is known to be a challenging exercise
- MPI communication deadlock makes it difficult for programs to guarantee their correctness, which seriously affects availability of HPC system
- Due to the significant uncertainty and complexity of the execution of the MPI process, communication deadlock detection becomes extremely difficult.
- Although many scholars have conducted a lot of research on the problem of MPI communication deadlock detection, **a scalable solution remains elusive**

Contributions

- MPI-RCDD: A framework for MPI runtime communication deadlock detection is proposed
  - The first time to design and optimize in the MPI runtime library to solve the MPI deadlock problem
  - Many processes are involved in the deadlock detection of the program, with strong capability and scalability
- A new MPI communication deadlock detection algorithm based on AND⊕OR wait graph model is proposed
  - Using the asynchronous processing thread provided by the MPI runtime environment to transparently implement the dependency transfer between processes
- Multiple typical benchmarks were used to evaluate the effectiveness of the MPI-RCDD
  - Capability：Umpire Test Suit
  - Scalability：NPB benchmarks

Experiments

- **Capability**
  - MPI-RCDD always detects deadlocks within valid time and does not generate false positives

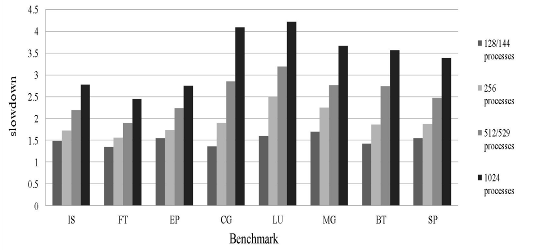
Results on the X86 cluster

Test Program	Deadlock Type	Process Size	Tool	Detection Time (s)
2D-Diffusion	potential	4	MPI-RCDD	0.284
			MUST	0.283
Heat Error	potential	32	MPI-RCDD	0.276
			MUST	0.369
basic-deadlock	deterministic	4	MPI-RCDD	0.232
			MUST	0.226
basic-deadlock2	deterministic	8	MPI-RCDD	0.185
			MUST	0.265
irecv-deadlock	deterministic	16	MPI-RCDD	0.313
			MUST	0.299
wait-deadlock	deterministic	32	MPI-RCDD	0.34
			MUST	0.448
waitall-deadlock	deterministic	4	MPI-RCDD	0.226
			MUST	0.224
waitany-deadlock	deterministic	8	MPI-RCDD	0.213
			MUST	0.212
any_src-can-deadlock7	potential	16	MPI-RCDD	0.265
			MUST	0.298
any_src-can-deadlock9	potential	32	MPI-RCDD	0.275
			MUST	0.303
any_src-wait-deadlock	potential	4	MPI-RCDD	0.212
			MUST	0.207
any_src-waitall-deadlock	potential	8	MPI-RCDD	0.354
			MUST	0.372
any_src-waitany-deadlock	potential	16	MPI-RCDD	0.511
			MUST	0.578
bcast-deadlock	deterministic	32	MPI-RCDD	0.243
			MUST	0.286
collective-misorder	deterministic	4	MPI-RCDD	0.335
			MUST	0.323
collective-misorder2	deterministic	8	MPI-RCDD	0.338
			MUST	0.402
collective-misorder-allreduce	deterministic	16	MPI-RCDD	0.244
			MUST	0.317

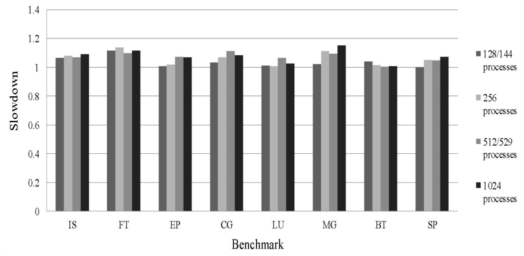
Results on the Sunway TaihuLight

Test Program	Deadlock Type	Process Size	Tool	Detection Time (s)
Ex-basic-deadlock	deterministic	1024	MPI-RCDD	0.752
			MUST	1.319
		2048	MPI-RCDD	0.909
			MUST	1.767
		4096	MPI-RCDD	1.114
			MUST	2.404
Ex-irecv-deadlock	deterministic	1024	MPI-RCDD	0.677
			MUST	1.224
		2048	MPI-RCDD	0.856
			MUST	1.821
		4096	MPI-RCDD	1.073
			MUST	2.676
Ex-any_src-can-deadlock9	potential	1024	MPI-RCDD	0.754
			MUST	1.365
		2048	MPI-RCDD	0.968
			MUST	2.29
		4096	MPI-RCDD	1.133
			MUST	2.986
Ex-waitany-deadlock	potential	1024	MPI-RCDD	0.841
			MUST	1.506
		2048	MPI-RCDD	0.935
			MUST	2.098
		4096	MPI-RCDD	1.173
			MUST	3.33
Ex-collective-misorder	deterministic	1024	MPI-RCDD	0.664
			MUST	1.158
		2048	MPI-RCDD	0.725
			MUST	1.658
		4096	MPI-RCDD	0.962
			MUST	2.326

- **Scalability**
  - MPI-RCDD's deadlock detection work subtly bypasses the strong correlation with program size, and it has strong scalability, reaching the expected goal of processing large-scale parallel applications

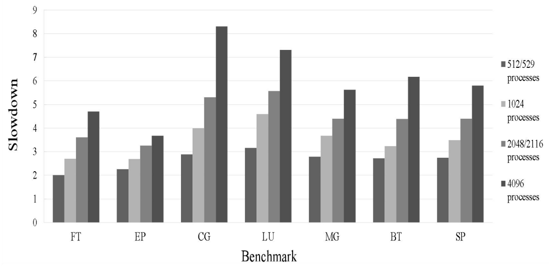


Results for MUST on the X86 cluster

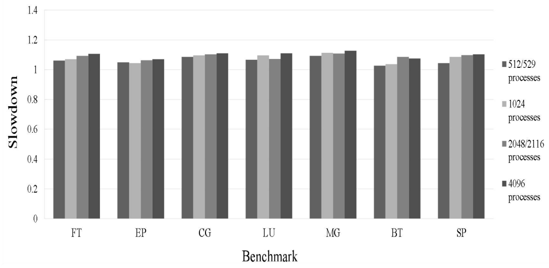


Results for MPI-RCDD on the X86 cluster

Results of the NPB benchmark on X86 clusters (C scale)



Results for MUST on the Sunway TaihuLight



Results for MPI-RCDD on the Sunway TaihuLight

Results of the NPB benchmark on Sunway TaihuLight (D scale)

Conclusions

- MPI-RCDD is not limited to a specific system structure and does not rely on additional components by closely linking the deadlock detection problem to the MPI runtime library
- The AODA algorithm we proposed combines two common deadlock analysis methods, message timeout and process dependency, to ensure that no false positives are generated, and the root cause of deadlock can be accurately located, alleviating the performance bottleneck of centralized analysis
- The capability and scalability of the MPI-RCDD was verified using a number of typical benchmarks with satisfactory results