The MPI Bugs Initiative (MBI)
a Framework for MPI Verification Tools Evaluation

Mathieu Laurent[◇], Emmanuelle Saillard[‡], Martin Quinson[◇]

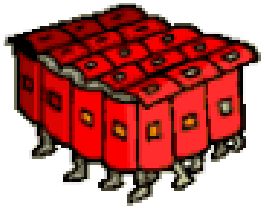
◇ Univ. Rennes – Inria – CNRS – IRISA (France)

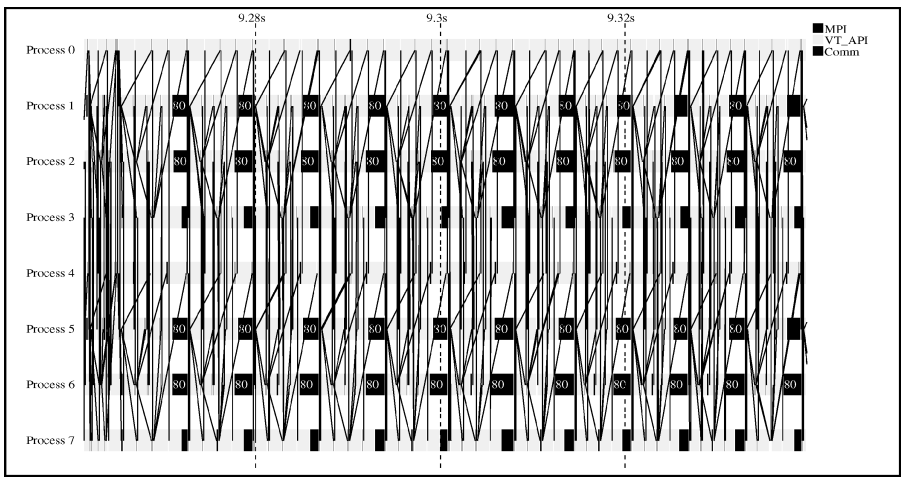
‡ Inria Bordeaux (France)

Correctness 2021
Saint-Louis, Missouri

Writting Correct MPI Applications

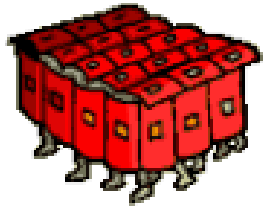
- Classical Solution: Proof of algorithms
- Pessimistic Solution: Lower performance expectations
- Optimistic Solution: Eventually Consistent
- HPC Solution: Rigid, Regular, Hand-tuned Communication Patterns

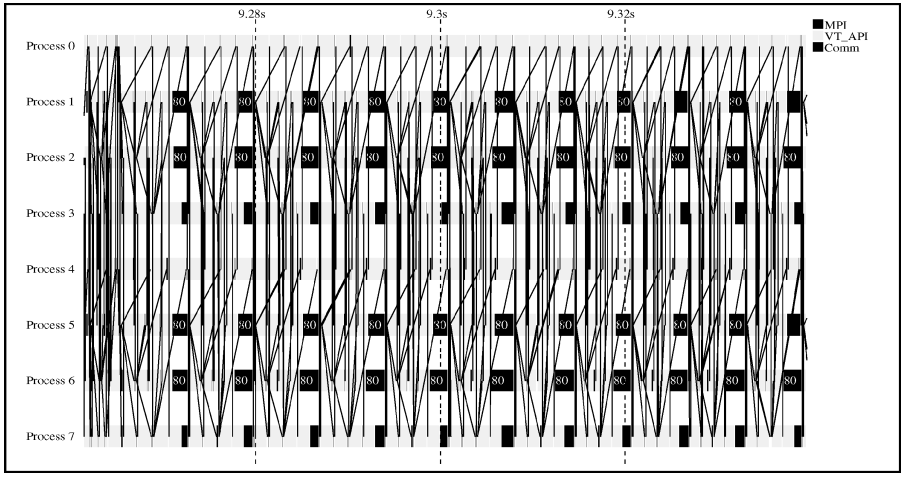


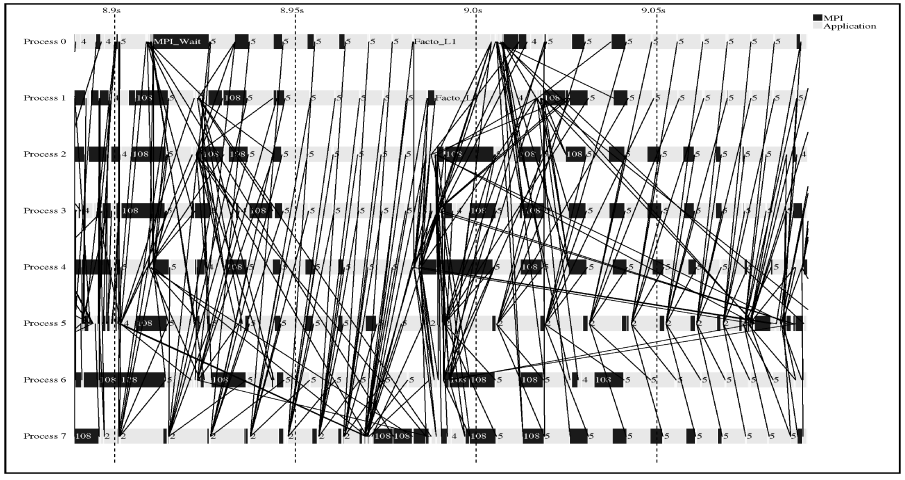


Writting Correct MPI Applications

- Classical Solution: Proof of algorithms
- Pessimistic Solution: Lower performance expectations
- Optimistic Solution: Eventually Consistent
- HPC Solution: Rigid, Regular, Hand-tuned Communication Patterns
- Large-Scale Hybrid Machines: Dynamic, Irregular (task-based?)







Need for good **bug finding tools**

MPI bugs and tools

MPI is feature-rich and intended for performance

- Semantic "specified" in plain text, with room for interpretation
- Some features may introduce subtle bugs, resulting in many different errors
- Not all features are equally important in MPI

Bug finding approaches: static, dynamic or symbolic analysis

- Many MPI bug finding tools, combining these approaches
- Not all tools cover the whole MPI standard
- Some tools specialized in some classes of bug
- Tool comparison difficult for users (and authors!)

MBI: systematic and practical evaluation of bug finding tools

- New classification of MPI root cause errors
- Codes generated to cover: MPI features $\times$ error classes
- Eval: Aislinn, CIVL, ISP, ITAC, McSimGrid, MPI-SV, MUST, PARCOACH
- Extensive tooling to evaluate future versions and tools

Systematically generating MPI codes

Simple classification of MPI features

MPI feature label		Codes with the label	
		# Incorrect	# Correct
P2P	base calls	132	14
	nonblocking	114	13
	persistent	65	10
COLL	base calls	542	468
	nonblocking	529	480
	communicators, datatypes, etc	102	32
Remote Memory Access		42	3

- We generate correct and incorrect codes covering the target features
- More features should be added in the future (MPI-IO)

MPI error labels, categorized by scope

Errors in single call

- Invalid parameter: invalid root, buffer size, communicator, datatype, etc
- The MPI implementations catch many of these errors

Errors local to a process

- Inconsistency between local context and function parameters
- Resource leak: datatype, request, communicator (not malloc)
- Request lifecycle: missing *wait* or *start* calls
- Local concurrency: RMA, or use during asynchronous comm

Multi-processes errors

- Message race: Message ordering issues
- Parameter matching: Sending bytes and receiving floats, or similar
- Call ordering: local orders of collectives is not the same on all ranks
- Global concurrency: conflicting Put in RMA

Errors classification

		Point-to-point			Collective			RMA	Unique files
		base calls	nblocking	persistent	base calls	nblocking	tools		
Single call	Invalid Param.	48	36	36	33	33	55	12	154
Single-Process	Resource leak	0	1	2	0	0	14	0	16
	Request lifecycle	0	4	5	0	12	0	0	18
	Local concur.	2	3	3	0	11	0	18	37
Multi-processes	Param. matching	27	19	19	29	29	33	0	97
	Message Race	3	3	0	0	0	0	0	4
	Call ordering	52	48	0	480	444	0	0	648
	Global concur.	0	0	0	0	0	0	12	12
Correct codes		14	13	10	468	480	32	3	682
Total		146	127	75	1010	1009	134	45	1668

- Incorrect codes only contain one error each

Machine-readable headers

- Features: present/missing; Expected outcome: OK, or which error
- command line to launch the test

Evaluated tools

Tool	Static analysis	Model checking	Symbolic execution	Testing
Aislinn	-	-	yes	-
ISP	-	yes	-	-
CIVL	-	yes	yes	-
MPI-SV	-	yes	yes	-
ITAC	-	-	-	yes
MUST	-	-	-	yes
Mc SimGrid	-	yes	-	-
Parcoach	yes	-	-	-

- M. Quinson author of Mc SimGrid; E. Saillard author of Parcoach
- Aislinn technically built upon ISP
- MPI-SV built upon Cloud9/Klee and Azequia MPI (MPI 2)
- Some tools can be challenging to install
- ITAC is closed source

Results

Outcome of each test with each tool

- Failure: Compilation error (CE), Runtime error (RE), 5 minutes timeout (TO)
- Success: True Positive (TP) or True Negative (TN)
- Error: False Negative (FN) or False Positive (FP)
- Timing not considered: simplistic codes; 5 mn analysis to find a bug is short

Results

Outcome of each test with each tool

- Failure: Compilation error (CE), Runtime error (RE), 5 minutes timeout (TO)
- Success: True Positive (TP) or True Negative (TN)
- Error: False Negative (FN) or False Positive (FP)
- Timing not considered: simplistic codes; 5 mn analysis to find a bug is short

Best tools per error category

Error	**Best tool**	**Second best**
Invalid param	ITAC (TP: 152, FN: 2)	MUST (TP: 150, RE: 4)
Resource leak	MUST (TP: 16)	Mc SimGrid (TP: 14, FN: 2)
Req lifecycle	McSimGrid (TP: 16, FN: 2)	MUST/ITAC (TP: 14, FN: 4)
Loc concurrence	ISP (TP: 16, FN: 21)	ITAC (TP: 13, FN: 24)
Param match	MUST (TP: 97)	ITAC (TP: 94, FN: 3)
Message race	ISP/Mc SimGrid (TP: 4)	ITAC (TP: 3, FN: 1)
Call ordering	ITAC (TP: 648)	MUST (TP: 643, RE: 3)
Glob concurrence	(best score: TP = 0)	
Correct code	ITAC (TN: 682)	MUST (TN: 681, FP: 1)

- All details are in the paper (1668 tests × 8 tools)

Agregated results

Tool	Errors			Results				
	CE	TO	RE	TP	TN	FP	FN	
ITAC	**0**	**0**	**0**	**926**	**682**	**0**	60	
MUST	**0**	4	5	**926**	681	1	51	
Mc SimGrid	**0**	**0**	53	657	675	**0**	283	
ISP	**0**	7	1	893	202	479	86	
Aislinn	860	**0**	15	449	297	**0**	47	Some silly bugs
MPI-SV	1043	**0**	53	122	181	4	265	MPI-2 because of AzequiaMPI
CIVL	1296	**0**	88	144	133	1	**6**	Parse errors and coverage
Parcoach	944	**0**	**0**	217	59	182	266	Coverage, too sensitive
Ideal tool	*0*	*0*	*0*	*986*	*682*	*0*	*0*	

Agregated results

Tool	Errors			Results				Robustness	
	CE	TO	RE	TP	TN	FP	FN	Coverage	Conclusiveness
ITAC	**0**	**0**	**0**	**926**	**682**	**0**	60	**1**	**1**
MUST	**0**	4	5	**926**	681	1	51	**1**	0.99
Mc SimGrid	**0**	**0**	53	657	675	**0**	283	**1**	0.97
ISP	**0**	7	1	893	202	479	86	**1**	0.99
Aislinn	860	**0**	15	449	297	**0**	47	0.48	0.48
MPI-SV	1043	**0**	53	122	181	4	265	0.37	0.34
CIVL	1296	**0**	88	144	133	1	**6**	0.22	0.17
Parcoach	944	**0**	**0**	217	59	182	266	0.43	0.43
Ideal tool	*0*	*0*	*0*	*986*	*682*	*0*	*0*	*1*	*1*

- Coverage: Ability to compile codes. $Cov = 1 - \frac{CE}{total}$
- Conclusiveness: Ability to draw a diagnostic. $Cc = 1 - \frac{CE+RE+TO}{total}$

Agregated results

Tool	Errors			Results				Robustness		Usefulness			
	CE	TO	RE	TP	TN	FP	FN	Coverage	Conclusiveness	Specificity	Recall	Precision	F1 Score
ITAC	**0**	**0**	**0**	**926**	**682**	**0**	60	**1**	**1**	**1**	0.94	**1**	0.83
MUST	**0**	4	5	**926**	681	1	51	**1**	0.99	0.99	0.95	0.99	0.83
Mc SimGrid	**0**	**0**	53	657	675	**0**	283	**1**	0.97	**1**	0.70	**1**	0.83
ISP	**0**	7	1	893	202	479	86	**1**	0.99	0.30	0.91	0.65	0.25
Aislinn	860	**0**	15	449	297	**0**	47	0.48	0.48	**1**	0.91	**1**	0.76
MPI-SV	1043	**0**	53	122	181	4	265	0.37	0.34	0.98	0.32	0.97	0.52
CIVL	1296	**0**	88	144	133	1	**6**	0.22	0.17	0.99	**0.96**	0.99	**0.94**
Parcoach	944	**0**	**0**	217	59	182	266	0.43	0.43	0.24	0.45	0.54	0.22
Ideal tool	*0*	*0*	*0*	*986*	*682*	*0*	*0*	*1*	*1*	*1*	*1*	*1*	*1*

- Specificity: Don't find errors in correct codes. $S = \frac{TN}{TN+FP}$
- Recall: Find existing errors. $R = \frac{TP}{TP+FN}$
- Precision: Confidence when a code is reported correct. $P = \frac{TP}{TP+FP}$
- F1 score: Overall quality of diagnostics. $F1 = \frac{2 \times P \times R}{P+R}$

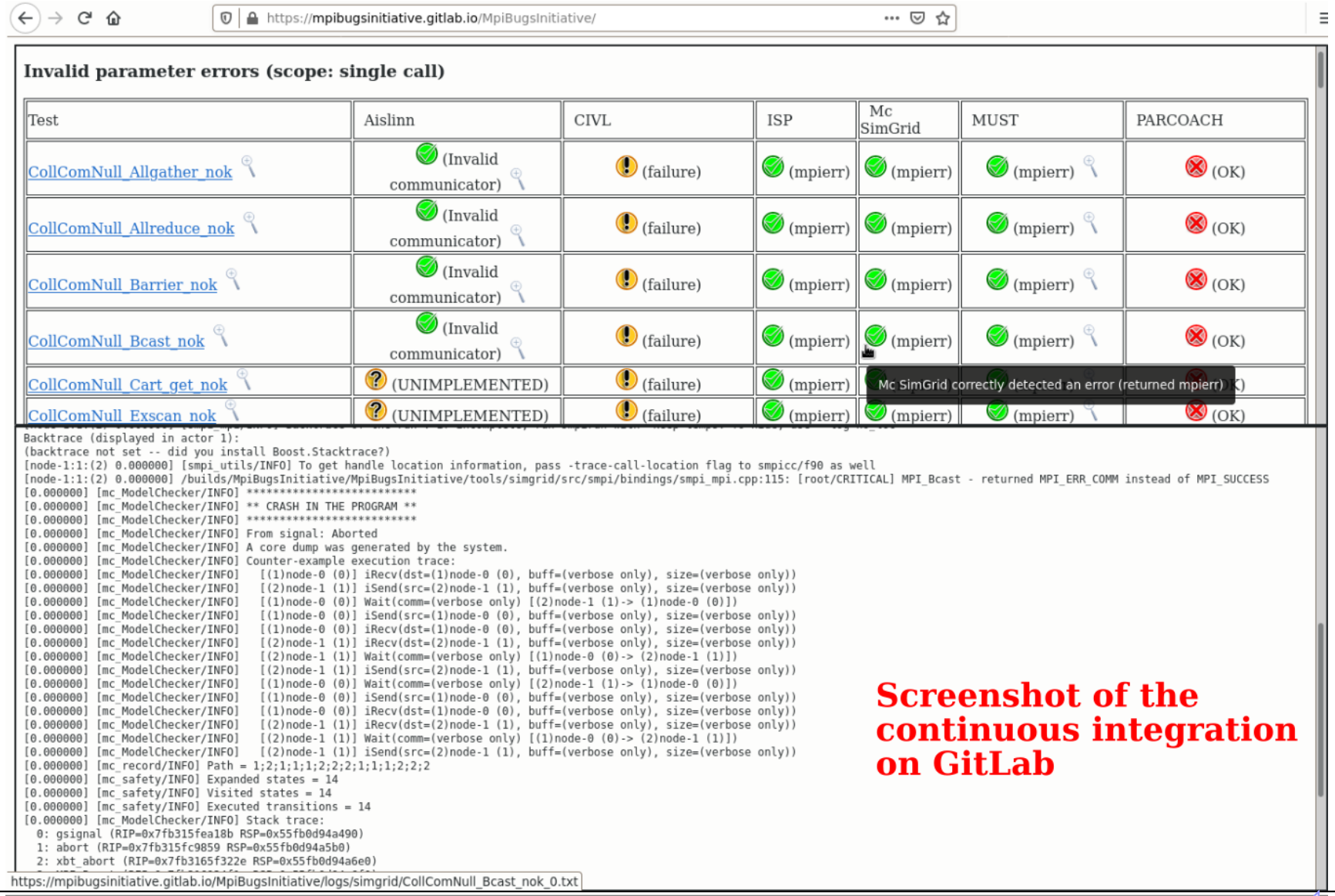
Agregated results

Tool	Errors			Results				Robustness		Usefulness				Overall
	CE	TO	RE	TP	TN	FP	FN	Coverage	Conclusiveness	Specificity	Recall	Precision	F1 Score	accuracy
ITAC	**0**	**0**	**0**	**926**	**682**	**0**	60	**1**	**1**	**1**	0.94	**1**	0.83	**0.96**
MUST	**0**	4	5	**926**	681	1	51	**1**	0.99	0.99	0.95	0.99	0.83	0.96
Mc SimGrid	**0**	**0**	53	657	675	**0**	283	**1**	0.97	**1**	0.70	**1**	0.83	0.80
ISP	**0**	7	1	893	202	479	86	**1**	0.99	0.30	0.91	0.65	0.25	0.67
Aislinn	860	**0**	15	449	297	**0**	47	0.48	0.48	**1**	0.91	**1**	0.76	0.45
MPI-SV	1043	**0**	53	122	181	4	265	0.37	0.34	0.98	0.32	0.97	0.52	0.18
CIVL	1296	**0**	88	144	133	1	**6**	0.22	0.17	0.99	**0.96**	0.99	**0.94**	0.17
Parcoach	944	**0**	**0**	217	59	182	266	0.43	0.43	0.24	0.45	0.54	0.22	0.17
Ideal tool	*0*	*0*	*0*	*986*	*682*	*0*	*0*	*1*	*1*	*1*	*1*	*1*	*1*	*1*

- F1 score: Overall quality of diagnostics. $F1 = \frac{2 \times P \times R}{P+R}$
- Accuracy: Correct diagnostics over all codes. $A = \frac{TP+TN}{total}$

MBI tooling

- MBI fully dockerized. Simple script for data provenance and analysis
- Web dashboard to explore all logs + caching, as needed by tool authors



Screenshot of the continuous integration on GitLab

Conclusion

MBI aims at fair comparison of the many bug finding tools for MPI

- Classification of MPI root cause errors
- Codes generated to cover: MPI features × error classes
- We evaluated our benchmark on 8 sota tools
- Tooling is provided for authors to test and improve their tools
  - Bugs opened after the study: Aislinn:4, CIVL:1, MUST:1, Hermes:2, MPIch:1. SimGrid & Parcoach:many.

Future work

- Cover MPI-IO and others; Introduce more code variants for the same errors
- Better testing of transient errors and system hazards
- Larger codes for scalability testing
- Go for an annual bug finding competition at Correctness!