Benchmarking of Java Verification Tools at the Software Verification Competition (SV-COMP)



The University of Manchester

Lucas Cordeiro Daniel Kroening <u>Peter Schrammel</u>





JPF Workshop 2018

What?

For me?

Annual comparative evaluation of fully automatic software verifiers

- Reflect state of the art w.r.t. effectiveness and efficiency
- Promote reproducibility and validity of experimental results
- Increase the visibility and credits for tool developers
- Establish set of benchmarks for software verification community

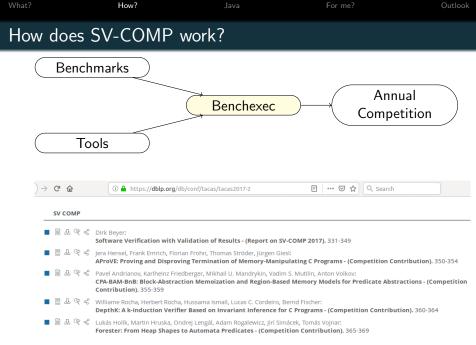
Started in 2012 for C programs, 8th edition in 2019:

- More than 10000 benchmarks
- More than 30 participants
- NEW: Java track

Outlook

What? How? Java For me? Outlook Goals of this talk

- Understand how SV-COMP works
- Know about the Java track at SV-COMP 2019
- Be able to use it for your own research
- Be able to contribute tools and benchmarks



What?	How?		For me?	Outlook
Benchmarks		https://github.c	com/sosy-lab/sv-	benchmarks

For each verification task (aka benchmark)

- Source files (open source license)
- Descriptor (.yml file)
 - File name is name of the benchmark
 - Reference to one or more properties (.prp files)
 - unreach-call: CHECK(init(main()), LTL(G !call(__VERIFIER_error())))
 - termination:

```
CHECK(init(main()), LTL(F end))
```

- no-overflow, valid-memsafety, ...
- Expected answer: true, false(property), unknown

What?	How?		For me?	Outlook
Benchmarks		https://github.c	com/sosy-lab/sv	-benchmarks

For each verification task (aka benchmark)

- Source files (open source license)
- Descriptor (.yml file)
 - File name is name of the benchmark
 - Reference to one or more properties (.prp files)
 - unreach-call: CHECK(init(main()), LTL(G !call(__VERIFIER_error())))
 - termination:

CHECK(init(main()), LTL(F end))

- no-overflow, valid-memsafety, ...
- Expected answer: true, false(property), unknown

Categories defined as subsets (.set files)

- ReachSafety, ConcurrencySafety, MemorySafety, NoOverflows, Termination, . . .
- There are sub-categories (loops, arrays, heap, ...).

What?	How?		For me?	Outlook		
Benchexec		https://git	hub.com/sosy-lat)/benchexec		

(Beyer et al SPIN'15)

- Implemented in Python 3
- Resource-limited execution (8 cores, 15GB, 900s CPU time)
- Interfaces to **competition candidates** (aka tools) via **tool-info** modules
 - Name, version
 - Build command line
 - Parse answer (true, false, unknown) from tool output

(Beyer et al SPIN'15)

- Implemented in Python 3
- Resource-limited execution (8 cores, 15GB, 900s CPU time)
- Interfaces to competition candidates (aka tools) via tool-info modules
 - Name, version
 - Build command line
 - Parse answer (true, false, unknown) from tool output
- Table-generator to generate HTML table of results

What?	How?		For me?	Outlook
Benchexec		https://git	hub.com/sosv-lal	o/benchexec

(Beyer et al SPIN'15)

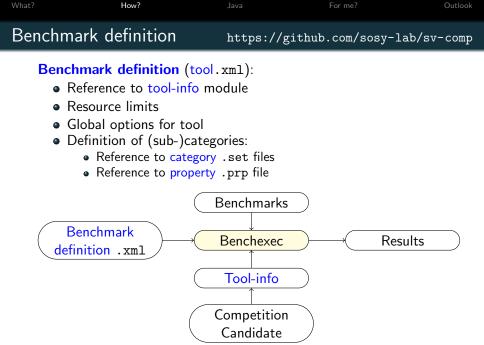
- Implemented in Python 3
- Resource-limited execution (8 cores, 15GB, 900s CPU time)
- Interfaces to competition candidates (aka tools) via tool-info modules
 - Name, version
 - Build command line
 - Parse answer (true, false, unknown) from tool output
- Table-generator to generate HTML table of results
- SV-COMP 2019 runs on Ubuntu 18.04

What?	How?		For me?	Outlook
Benchexec		https://git	hub.com/sosv-lal	o/benchexec

(Beyer et al SPIN'15)

- Implemented in Python 3
- Resource-limited execution (8 cores, 15GB, 900s CPU time)
- Interfaces to **competition candidates** (aka tools) via **tool-info** modules
 - Name, version
 - Build command line
 - Parse answer (true, false, unknown) from tool output
- Table-generator to generate HTML table of results
- SV-COMP 2019 runs on Ubuntu 18.04

E.g. run CBMC on sub-category ReachSafety-BitVectors: bin/benchexec cbmc.xml -t ReachSafety-BitVectors





Benchexec

Benchexec

Competition

Candidate

Witness

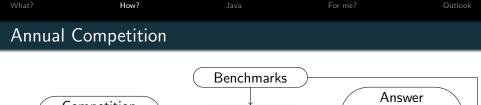
Validator

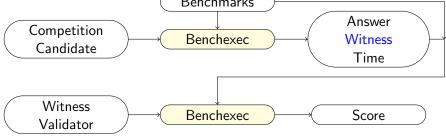
Answer

Witness

Time

Score





Witness Validation

(Beyer et al FSE'15, FSE'16)

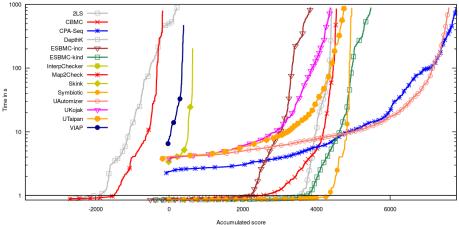
- Correctness and refutation witnesses (.graphml)
- Validated by witness validation tools

What?	How?	For me?	Outlook
(

\sim				
5	0	\sim	-	es
.)	ι.,	U		-5
-	-	-		~~

Points	Reported result	Description
0	UNKNOWN	Failure to compute verification result, out of resources, program crash.
+1	FALSE correct	The error in the program was found and a violation witness was confirmed.
-16	FALSE incorrect	An error is reported for a program that fulfills the specification (false alarm, incomplete analysis).
+2	TRUE correct	The program was analyzed to be free of errors and a correctness witness was confirmed.
+1	TRUE correct, witness unconfirmed	The program was analyzed to be free of errors but the correctness witness was not confirmed.
-32	TRUE incorrect	The program had an error but the competition candidate did not find it (missed bug, unsound analysis).





https://sv-comp.sosy-lab.org/2018/results/results-verified/

For me?

Outlook

Annual Competition

Tool		JPF 32				JayH	om 5.1		JBMC 5.8			
Limits				ti	melimit: 180 s	s, memlimit:	15000 ME	, CPU core limi	t: 8			
Host						loc	alhost					
OS					Lin	ux 4.4.0-11	6-generic :	x86 64				
System	C	PU: Intel C	ore i7-67	00HQ CPU @	2.60GHz wit	h 8 cores, fi	equency:	3500 MHz, Turb	o Boost enat	oled; RAM:	16014056	5 kB
Date of execution	20	18-03-10 1	4:09:57 0	SMT	2	2018-03-10	11:39:56 0	SMT	20	018-03-10	16:54:00 0	SMT
Run set		jpf.sv-comp18				jayhorn.	v-comp18				-comp18	
/git-uv-benchmarka/jawa/	status	cputime	walltime	memUsage	status	cputime	walltime	memUsage	status	cputime	walltime	memUsage
jbmc-regression/ArithmeticException1_false-assert.jar	false(reach)	1.252s	0.492s	86929468	true	3.7485	1.352s	203776000	false(reach)	0.207s	0.224s	17719296
jbmc-regression/ArithmeticException5_true-assert.jar	false(reach)	1.2085	0.4795	85815296	true	3.564s	1.1695	184410112	true	0.2125	0.2265	12709888
jbmc-regression/ArithmeticException6_false-assert.jar	true	1.172s	0.452s	82345984	true	3.448s	1.164s	174071808	false(reach)	0.251s	0.264s	17477632
jbmc-regression/ArrayIndexOutOfBoundsException1_false-assert.jan	true	1.1645	0.475s	83734528	false(reach)	6.5165	1.9345	228200448		0.2565	0.2685	16728064
jbmc-regression/ArrayIndexOutOfBoundsException2_false-assert.jam	true	1.140s	0.443s	81854464	false(reach)	6.7485	2.8385	238465536	false(reach)	0.258s	0.257s	18628688
jbmc-regression/ArrayIndexOutOfBoundsException3_false-assert.jan	true	1.256s	0.501s	82014208	false(reach)	7.284s	2.0235	233783296	false(reach)	0.2235	0.2365	18378752
jbmc-regression/BufferedReaderReadLine_false-assert.jar	true	1.2005	0.464s	85873928	false(reach)	11.4845	2.9925	264642568	false(reach)	0.373s	0.381s	29315872
jbmc-regression/CharSequenceBug_false-assert.jar	true	1.1365	0.444s	79372288	false(reach)	7.336s	2.1915	233238528	false(reach)	1.407s	1.4225	26890240
jbmc-regression/CharSequenceToString_true-assert.jar	true	1.1365	0.501s	80220160	false(reach)	7.4125	2.1865	235985824	true	1.647s	1.059s	36552784
jbmc-regression/ClassCastException1_false-assert.jar	false(reach)	1.1725	0.462s	86487848	false(reach)	6.124s	1.898s	230752256	false(reach)	0.305s	0.319s	18565472
jbmc-regression/ClassCastException2_true-assert.jar	true	1.2165	0.5865	77561856	true	6.5125	1.9075	233218048	true	0.2335	0.2535	13275136
jbmc-regression/ClassCastException3_false-assert.jar	false(reach)	1.148s	0.441s	84291584	false(reach)	6.852s	2.8885	233496576	false(reach)	0.235s	0.242s	17874944
jbmc-regression/Class_method1_true-assert.jar	true	1.1605	0.457s	80973824	true	5.1485	1.6095	236470272	true	0.2365	0.2565	13553664
jbmc-regression/Inheritancel_true-assert.jar	true	1.180s	0.482s	83697664		7.240s	2.846s	238694912	true	0.268s	0.276s	13492224
jbmc-regression/NegativeArraySizeException1_false-assert.jar	false(reach)	1.2365	0.4925	87408448	true	3.7205	1.2305	185139200	false(reach)	0.2135	0.2255	16687184
jbmc-regression/NegativeArraySizeException2_false-assert.jar	false(reach)	1.176s	0.472s	83542016	true	3.988s	1.245s	185753688	false(reach)	0.241s	0.247s	18546688
jbmc-regression/NullPointerException1_false-assert.jar	false(reach)	1.1525	0.4485	84785280	unknown	1.8645	0.6865	136836352	false(reach)	0.2435	0.2455	16834560
jbmc-regression/NullPointerException2_false-assert.jar	false(reach)	1.2125	0.466s	85893120	unknown	1.740s	0.676s	134981768	false(reach)	0.244s	0.251s	16654336
jbmc-regression/NullPointerException3_false-assert.jar	false(reach)	1.3685	0.5325	88522752	unknown	1.6685	0.6165		false(reach)	0.2435	0.255s	16982816
jbmc-regression/NullPointerException4_false-assert.jar	false(reach)	1.2725	0.535s	77479936	unknown	1.6365	0.6265	128970752	false(reach)	8.2485	0.2485	18771968
jbmc-regression/RegexMatches01_true-assert.jar	true	1.300s	0.537s		false(reach)	8.344s	2.393s	246411264	true	1.2885	1.301s	39464960
jbmc-regression/RegexMatches02_false-assert.jar	true	1.1925	0.481s	85598268	false(reach)	9.8245	2.621s	254140416	false(reach)	1.033s	1.844s	64978944
jbmc-regression/RegexSubstitution01_true-assert.jar	true	1.252s	0.503s		false(reach)	14.268s	3.535s	394878976	true	72.774s	72.850s	757389448
jbmc-regression/RegexSubstitution02_false-assert.jar	true	1.1165	0.442s		false(reach)	17.1365	4.2845	398799368	true	48.973s	41.011s	526553888
jbmc-regression/RegexSubstitution03_true-assert.jar	true	1.232s	0.465s	86216704	false(reach)	12.689s	3.158s	338685952	true	1.754s	1.766s	46884992
jbmc-regression/StaticCharMethods01_true-assert.jar	true	1.1765	0.5025		false(reach)	6.7445	2.841s	228970496	true	0.261s	0.2765	14376960
jbmc-regression/StaticCharMethods02_false-assert.jar	true	1.196s	0.484s		false(reach)	8.524s	2.476s		false(reach)	0.829s	0.854s	50974720
jbmc-regression/StaticCharMethods03_false-assert.jar	true	1.1925	0.4865		false(reach)	10.1885	2.8885	243417088	false(reach)	0.8475	0.8585	50806784
jbmc-regression/StaticCharMethods04_false-assert.jar	true	1.196s	0.487s	84496384	false(reach)	8.664s	2.529s	231289648	false(reach)	0.821s	0.829s	51412992

What? How? For me? Outlook

Competition Timeline

- September: Contribution of benchmarks
- October: Tool registration and qualification
- November: Tool submission
- December: Announcement of winners
- January: Tool paper submission
- April: SV-COMP session at ETAPS

Java Track at SV-COMP

Objectives:

- More languages in SV-COMP
- Standard benchmark set
- Comparability
- Reproducibility
- Re-use existing benchmarking infrastructure

Linix			-	1	metimit 190 r	L mentinit		CFU core time	1.9			
								1500 MHz, 3H				
Date at execution												
			angLi								-compLife	
	700.0	confirme.		mentioape	20010			recularge	200.0			nertilinge
ion: repression, living interdentificantial suprime. False aware, par	the state	1.560	4.470	accessed.		4.130	2.854			8.0164	0.368	10,75846
bec-repression/drosplashob/UPBismbilisory/Land_False-accent. (an	110	1.50%	0.4434	Katalah		0.101	2.8084		Service Heat	8.2584	0.213	200300
												303279
												20ADAT
				1040400							9.031	112962
											0.256	
				ENGOARD								248(12
two-repression/multiplanemooptiane_false-assert.jar		1,253					8.8091					2811399
	12.0	1.2284		KACARDO		17.1300	4,28%	280,790000	110			33500088
	100	1.033	0.401	10(15/19		17,000	2.5545		100	3,1941	1,996	40000
inc.repression/9541/DerNetholdel, true-accert. Jar	100	1.5%	4.5424	A Server		6.144	2.845			# 251s	+.214+	145.50
les: repression/Outsidur/Websidd0_false avant.jar	the state	3.564	0.484	and all 1		8.534	2.6%			8.800		SHC-022
two-repression/statistical@wthebdit_tate=acovit_jar	110	1.000	0.4804	KC04/20		20,2894	2.000		Seleng sech	8.80%	0.030	2408478

Java Track at SV-COMP

Objectives:

- More languages in SV-COMP
- Standard benchmark set
- Comparability
- Reproducibility
- Re-use existing benchmarking infrastructure

Current status:

• March 2018: Initial benchmarks collected, proof-of-concept tools integrated, potential participants contacted

								LCFU core tim				
				accessed.			3 856				0.368	
				and second							0.256	
	100	3.580	0.40	EARTH		T.38N	2.884	238NIN	\$100	8.2604	0.1704	1300000
			0.413	ENGOARD		2,720				8,252	0.1D1	200110
itec repression will represent out the ensert. jur		1,253	0.329	1147900		1,420	8.5051	23949120		1.001	0.3481	2011200
Jans, repression, Republisheski, Julian, assort, jur						9.434					1.644	
(Incorportation/Myrebalation/Incore) failed-accent. [ar	110	1.3284	0.443		bloop such	17.130	4,209	280,790000	110	41,872	10,003	33000000
	100	1,033	0.401	10(15/19		17,000	2,556		100	3,1941	1,996	410000
Jan. repression/StatisTerberholds. Instancert. (ar	100	1555		KORDON.		6.144	2.845		100	# 251s	+.214+	145,536
film: repression/OutscDarMetholid0_false assert.jar	and a second	3.566	0.484			4.134	2.6%			8.800		SHC-022
(Incompression/Instantion/Instantion) false-accent. (an	110	1.300	0.480	KC04/20		10.1894	2.000			1.10%		10080730

Java Track at SV-COMP

Objectives:

- More languages in SV-COMP
- Standard benchmark set
- Comparability
- Reproducibility
- Re-use existing benchmarking infrastructure

- March 2018: Initial benchmarks collected, proof-of-concept tools integrated, potential participants contacted
- April 2018: Proposal accepted at Jury meeting at TACAS

Tot		J.	F 32				toen %1			30	C 6.8			
Linix				1	metimit 190 r	s mentioit		LCFU core tim	1.9					
									a Boast enabled, FMM, 35014055 MS					
Date of execution														
								menUsage						
jour repression/investigation/contribution/also priced. (also accord.) (at		1.564	4.475	accessed.		4.136	2.854	200200044		8.0164	0.368	201000		
per-repression/orrayEnderdorth?BrandsExceptionO_fator-accert. (an		1.509	0.4434	Kanasa	Data practic	0.105	2.8084			8.2584	0.2174	200300		
	114	1,254	0.312	525,4295	Machaech)	7,294	2,502			1,000	9,120	253717		
Incompression/Bellanderbooking.inc. faite-accent. jar	114	3.09	0.454	104/2001	March sech.	12,494	2,2903	3464[500		1.275	0.211	202.04		
											1.000	303001		
							2.8964							
				and second							0.256			
								238NIN			0.1704			
the opening highly wrighted and the meeting														
			0.31D			1,000	8.8584				0.000			
itec repression will represent out the ensert. jur		1,253	0.329	1147900		1,420	8.8091			1.001	0.3481	281126		
				and a second										
						14.305								
	12.0	1.32.84		KACARDO		17.130	4,28%	280,790000		41,872		3,300,000		
	100	3,033	0.401	10(15/14		12,000	2,506		100	3,041	1,996	40000		
Jac. repression/Statis/hetherholds. the assert, jar	100	1.5%		accesses.		6.144	2.845			4.2614		16550		
film: repression/OutscDarMetholid0_false assert.jar	the state	3.5%		and all 1		4.134	2.6%			8.8004	0.8544	Lancing of		
(Incompression/Instantion/Instantion) false-accent. (an	110	1.300		KC04/20		10.1894	2.000			1.10%		100007		

Java Track at SV-COMP

Objectives:

- More languages in SV-COMP
- Standard benchmark set
- Comparability
- Reproducibility
- Re-use existing benchmarking infrastructure

- March 2018: Initial benchmarks collected, proof-of-concept tools integrated, potential participants contacted
- April 2018: Proposal accepted at Jury meeting at TACAS
- July 2018: Announced at CAV

								30054				
102		timelinit: 180 s, mentinit: 16000 MB, CPU core limit &										
		logihost										
		Linux 4.4.0-118-genetic x86-64										
		CPU, Indi Cere IT 6700HQ CPU (# 2.800H2 with 8 cores, Requests). 200 MH2, Tube Boest erabled, RMR, 26524056 M										
		2018-03-03-05 L0 14-39-57 EMF 2118-03-03 11-39-58 GMT 2018-03-10-16-54-80 EMF										
										ibme.ev-compL8		
									Marganet.	8.0164	0.368	
two repression/loving@edeeder@?Boundstacoptized_failer-accent.lar		1.50%	0.4434			0.105	2.8084		Desegnant.	8.2584	0.2174	200300
											1.000	300000
						0.126				8.2854		204043
two repression to instantic optime the asset. (or	1.00	1.25	0.7991			4,312	3,990			8,000	9.031	
						6.140	3 4404				0.256	
								238NIN				
						2,720				8,252		200110
two regression weptions reactions registers follow-second dor-		2,379	0.4121			2,998	1,056	20,000		1.011	0.347	
		1,3404										
two-repression/multiplanemooptiane_false-assert.jar											0.3481	201100
her repression Reported works, true-resort dar		3,366		BOARD M		0.544	2,3906	24642254		3 (99)		
inc. repression, Republic health ful second to far	and a	1.565	0.4624			9.434	3.4264			3,8004	1.6444	640504
Inc-repression/hepstholititation/g_false-assert. [ar	12.0	1.2284	0.443			17.130	4,209	280,790000	110	41,872		
	100	1,033	0.401			12,000	2,509		100	3,041	1,996	
Incorpression/StatisThe/Nethologic true-accert. (ar	100	1.5%	4.562			6.144	2.845			# 261 s		
las: repression, Out in Due Nethologi, falles unverti, par	itse .	3.566				4.044	2.6%			8.3004	0.8524	Lucu?
bec-repression/statistica/Withold/0_fater-accert.jar	110	1.000				10.180	2.888		Support.	8.80%	0.8304	540887
			0.463			7,004			Marg such			

Java Track at SV-COMP

Objectives:

- More languages in SV-COMP
- Standard benchmark set
- Comparability
- Reproducibility
- Re-use existing benchmarking infrastructure

- March 2018: Initial benchmarks collected, proof-of-concept tools integrated, potential participants contacted
- April 2018: Proposal accepted at Jury meeting at TACAS
- July 2018: Announced at CAV
- Since September: detailed discussions with 4 registered participants (JPF, SPF, JayHorn, JBMC)

			F 32				oin \$1				IC 6.1	
		Emelimit: 180 s, reendinit: 15000 MB, CFU core limit 8										
		logihost										
		Linux 4.40.115-periori off5.64										
		CPU, Intel Care (7.6700HQ CPU (8.2.800Hz with 3 cons, Traggeray, 3500 MHz, Turbe Boest enabled, IVML 35514255 M3										
Date of execution		2018-03-02-04-02-03-02-03-02-03-02-03-02-03-02-03-02-03-02-03-02-03-02-02-02-02-02-02-02-02-02-02-02-02-02-										
Run tet.		infers canalify infers av complify infers av complify										
	200.0			mentioape	20010			menUsage	200.0			nertilinge
Jac repressor/wstMetsdsogtland_fator-accert.jar							1,300					
											0.368	
					Selection (Children (Child					8.2584	0.213	
						7.201						
two repression to an extension the enset, for		1,2250	0.790	1040400			3,990				9.031	112962
percompression/CanolaetExoptian0. False-ansert. (ar		3,545		8456504		4.452				 266 		2,00,000
											0.256	
	100	3.580		EARTH		T.38N	2.884		a sue	8.2604		1340000
			0.413	ENGOARD		2,720				8,252	0.ID	200110
the repression want internet out reption. for re-reserving			0.412	004030		2,998	1,056	20,000		1.011	0.347	20464
itec repression will represent out the ensert. jur		1,253	0.329	1147900		1,420	8.5051	23949120		1.001	0.348	2811249
Incorporation/Augustationality and an and a second second		3,000		BOARD M		0.544	2,3906	24642254		3 (99)		2946406
				#0.04004		9.434				2,000	1.444.	
	110	1.3284				17.130	4,209	280,790000		41,872		33500088
	1100	1,033	0.401	10(15/14		12,000	2,509		100	3,041	1,996	400400
Jac. repression/Statis/hetherholds. the assert, jar	100	1.5%		accesses.		6.144	2.845		100	4.2614	0.236	145516
film: repression/OutscOurMetholid0_Estar awart.jar	and a second	3.566	0.484	anetel 11		4.134	2.6%			8.8004		SHC-022
(Incompression/Instantion/Instantion) false-accent. (ar	110	1.300	0.480	KC04/20		10.1894	2.000			1.10%		2408479

Java Track at SV-COMP

Objectives:

- More languages in SV-COMP
- Standard benchmark set
- Comparability
- Reproducibility
- Re-use existing benchmarking infrastructure

- March 2018: Initial benchmarks collected, proof-of-concept tools integrated, potential participants contacted
- April 2018: Proposal accepted at Jury meeting at TACAS
- July 2018: Announced at CAV
- Since September: detailed discussions with 4 registered participants (JPF, SPF, JayHorn, JBMC)
- Planned start of competition runs: 20 November

Linix				1	metimit 190 r	s menticit		LCFU core tim	1.9			
		looihot										
		Linux 4.4.0-118-genetic x86-64										
		CPU, Indi Cere 17 6700HQ CPU @ 2,800H2 with 8 cores, linguistics, 3000 MHz, Turbe Boest enabled, RMM, 35524055 MB										
		2018-03-10 14 39-57 EMI 2018-02-03 11 3956 GMT 2018-03-10 16 54 30 EMI										
									bmc.av.compile			
ion: repression, ServeyEntenbeth/Moundal suprised. False arounts (an	the state	1.564	4.475	accessed.		4.136	2.854		Marganet.	8.064	0.368	201000
per-repression/orrayEndedor@?BoundsExceptionO_fator-accert. (an	11.0	1.509	0.4434	Kanasa		0.105	2.8084		Desegnant.	8.2504	0.2174	20030
	114	1,04	0.312	525,4295		7,294	2,502		Margaret,	1.023	9,120	2017
							2.186+	220444004			1.000	
			0.463			0.126						
			0.790	1040400							9.031	132756
				and second							0.256	
				ENGOARD				2012/09/09			0.1D1	20015
												20140
		1.36								1,200		
hec repression the Union to optione failed assert. (an		1,253		1147900		1,420			Marg tech	1.041	0.3481	28712
	12.0	1.108		KACARDO		17.130	4,28%	280,790000		46,8726		3,300.00
	100	3,033	0.401	10(15/14		12,000	2,506		100	3,1941	1,996	4000
incorrepression/Statistical International Transport. Jan	100	1.5%				6.144	2.845			4.2614	0.236	36559
les: repression/OutsinDarWeihold2_false avant.jar	the state	3,506		and all 1		4.134	2.6%			8.300		Lance -
Bec-repression/Statistice/Withold/States-accent.jar	110	1.00	0.480	KC04/20		10.1894	2.000			1.10		100007



368 benchmarks (40LOC on average, 250LOC max)

• jayhorn-recursive, jbmc-regression, jpf-regression, MinePump

Java 1.8

Calls to Java standard library (java.*, javax.*) allowed; sources of other dependencies must be part of the benchmark.

1 category for violation of asserts ("ReachSafety")
Property: CHECK(init(Main.main), LTL(G assert))

Benchmarks and Properties

```
iava/ibmc-regression/StringStartEnd02/Main.java
                                                    import org.sosy_lab.sv_benchmarks.Verifier;
java/jbmc-regression/StringStartEnd02.yml
                                                    public class Main {
                                                      public static void main(String[] args) {
                                                        String[] strings = new String[4];
                                                        strings[0] = Verifier.nondetString();
                                                        strings[1] = Verifier.nondetString();
                                                        strings[2] = Verifier.nondetString();
                                                        strings[3] = Verifier.nondetString();
                                                        int i = 0:
                                                        for (String string : strings) {
format version: "0.1"
                                                          if (string.startsWith("te"))
input files:
                                                            ++i:
  - ../common/
  - StringStartEnd02/
                                                        assert i == 1;
properties:
  - property file: ../properties/assert.prp
    expected_verdict: false
```

If a tool requires class files as input it is responsible for compiling the benchmark.

A benchmark must be compilable by passing all .java files within the directories listed in input_files to javac.

Rules for Nondeterminism

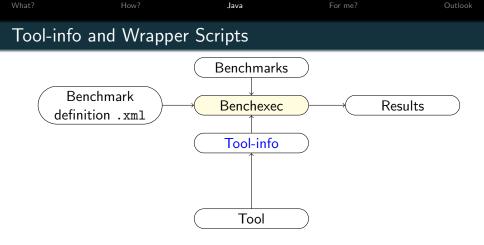
```
Only source of nondeterminism:
return values of methods defined in
org.sosy_lab.sv_benchmarks.
Verifier class.
```

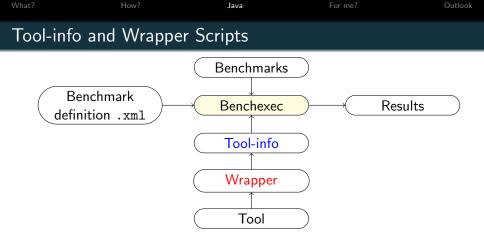
Must not be used:

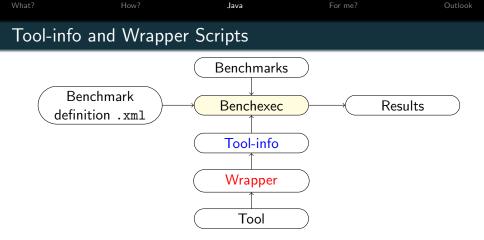
- Arguments of main
- Library methods that make system calls

```
package org.sosy lab.sv benchmarks;
import java.util.Random;
public final class Verifier
  public static void assume(boolean condition)
    if(!condition) {
      Runtime.getRuntime().halt(1);
  public static boolean nondetBoolean()
    return new Random().nextBoolean();
  public static byte nondetByte()
    return (byte)(new Random().nextInt());
```

. . .







E.g. command line produced for JPF:

./jpf-sv-comp

--graphml-witness witness.graphml

--propertyfile ../sv-benchmarks/java/properties/assert.prp

 $../{\tt sv-benchmarks/java/common/org/sosy_lab/sv_benchmarks/Verifier.java}$

../sv-benchmarks/java/jbmc-regression/StringStartEnd02/Main.java

17 / 23



Re-use existing benchmarking infrastructure

- Stop writing benchmarking scripts
- Use it for running your tests

'Standard' benchmark set

- Use it for running comparisons
- Contribute your benchmarks

Compare with the 'best' configuration of a tool

• Take the most recent competition candidate: Download from https://sv-comp.sosy-lab.org/2018/systems.php
 What?
 How?
 Java
 For me?
 Outlook

 How can l use it?

Reproduce the competition results:

• Download:

git clone https://github.com/sosy-lab/sv-benchmarks
git clone https://github.com/sosy-lab/benchexec
git clone https://github.com/sosy-lab/sv-comp
git clone https://gitlab.com/sosy-lab/sv-comp/archives-2019

• Run:

```
cd benchexec
for tool in jpf spf jayhorn jbmc
do
    unzip ../archives-2019/2019/$tool.zip; mv $tool/* .
    bin/benchexec ../sv-comp/benchmark-defs/$tool.xml
done
bin/table-generator results/*.xml.bz2
```

Currently (5 Nov 2018) only runs with benchexec's yaml branch.



Run JPF on your own benchmarks:

- Add descriptor yml file for each benchmark
- List descriptor files in MyBenchmarks.set
- Add MyBenchmarks.set to benchmark definition jpf.xml: <tasks name="MyBench"> <includesfile> ../sv-benchmarks/java/MyBenchmarks.set</includesfile> <propertyfile>

../sv-benchmarks/java/properties/assert.prp</propertyfile>
</tasks>

• Run with -t MyBench

bin/benchexec ../sv-comp/benchmark-defs/jpf.xml -t MyBench bin/table-generator results/*.xml.bz2



 More benchmarks: Fork https://github.com/sosy-lab/sv-benchmarks and create PR with your benchmarks



- More benchmarks: Fork https://github.com/sosy-lab/sv-benchmarks and create PR with your benchmarks
- Witness validators

Witness Validation for Java

Refutation witnesses (for ReachSafety property):

- Witness contains counterexample trace annotated with evaluated assignments and conditionals → Check whether counterexample trace is feasible and violates the property
- Proposed implementation: generate harness, compile and execute

Correctness witnesses (for ReachSafety property):

- Witness contains dynamic CFG annotated with invariants \rightarrow Check whether invariants hold and imply properties
- ???

What?	How?	For me?	Outlook
Outlook			

• More benchmarks:

Fork https://github.com/sosy-lab/sv-benchmarks and create PR with your benchmarks

Witness validators

What?	How?	For me?	Outlook
Outlook			

• More benchmarks:

Fork https://github.com/sosy-lab/sv-benchmarks and create PR with your benchmarks

- Witness validators
- Encourage participation of more tools and tool variants

Subscribe to sv-comp@googlegroups.com

Up-to-date version of paper: http://arxiv.org/abs/1809.03739

What?	How?	For me?	Outlook
Outlook			

• More benchmarks:

Fork https://github.com/sosy-lab/sv-benchmarks and create PR with your benchmarks

- Witness validators
- Encourage participation of more tools and tool variants

Subscribe to sv-comp@googlegroups.com

Up-to-date version of paper: http://arxiv.org/abs/1809.03739

www.diffblue.com

 Jobs in program analysis, verification and machine learning!

