8th Competition on Software Verification

Dirk Beyer

(Competition Chair)

Supported By:

Motivation - Goals

1. Community suffers from unreproducible results
   → Establish set of benchmarks
2. Publicity for tools that are available
   → Provide state-of-the-art overview
3. Support the development of verification tools
   → Give credits and visibility to developers
4. Establish standards
   → Specification language, Witnesses, Benchmark definitions, Validators

Schedule of Session

Session 1:

Competition Report, by organizer

System Presentations, 5 min by each team

Session 2:

Open Jury Meeting,Community Discussion,

moderated by organizer

Procedure – Time Line

Three Steps – Three Deadlines:

- Benchmark submission deadline
- System submission
- Notification of results (approved by teams)

Verification Problem

Input:

- C program → GNU/ANSI C standard

- Property
  - → Reachability of error label, of overflows
  - → Memory safety (inv-deref, inv-free, memleak)
  - → Termination

Output:

- TRUE + Witness (property holds)

- FALSE + Witness (property does not hold)

- UNKNOWN (failed to compute result)

Environment

Machines (1000 $ consumer machines):

- CPU: 3.4 GHz 64-bit Quad-Core CPU
- RAM: 33 GB
- OS: GNU/Linux (Ubuntu 18.04)

Resource limits:

- 15 GB memory
- 15 min CPU time (consumed 461 days)

Volume: 178 674 ver. runs, 517 175 val. runs

Scoring Schema (2019)

(from 2020 onwards: only confirmed results count)

Reported result	Points	Description
UNKNOWN	0	Failure, out of ressources
FALSE correct	+1	Error found and confirmed
FALSE incorrect	−16	False alarm (imprecise analysis)
TRUE correct	+2	Proof found and confirmed
TRUE unconfirmed	+1	Proof found but unconfirmed
TRUE incorrect	−32	Missed bug (unsound analysis)

Fair and Transparent

Jury:

- Team: one member of each participating candidate

- Term: one year (until next participants are determined)

Systems:

- All systems are available in open GitLab repo

- Configurations and Setup in GitHub repository

  → Integrity and reproducibility guaranteed

31 Competition Candidates

Qualification:

- 31 Qualified (out of 31 Submitted)
  1 verifier disqualified from several categories (rule viol.)
- One person can participate with different tools
- One tool can participate with several configurations
  (frameworks, no tool-name inflation)

Benchmark quality:

- Community effort, documented on GitHub

Role of organizer:

- Just service: Advice, Technical Help, Executing Runs

Benchmark Sets

- Everybody can submit benchmarks (conditions apply)
- Eight categories when closed (scores normalized):
  - Reachability: 3831 tasks
  - Memory Safety: 434 tasks
  - Concurrency: 1082 tasks
  - NoOverflows: 359 tasks
  - Termination: 2007 tasks
  - Software Systems: 2809 tasks
  - Overall: 10522 tasks
  - Java: 368 tasks

Replicability

- SV-Benchmarks:
  https://github.com/sosy-lab/sv-benchmarks
- SV-COMP Setup:

  https://github.com/sosy-lab/sv-comp
- Resource Measurement and Process Control

  https://github.com/sosy-lab/benchexec
- Archives

  https://gitlab.com/sosy-lab/sv-comp/archives-2019
- Witnesses

  https://sv-comp.sosy-lab.org/2017/results/results-verified/

Benchmark Definition

```
<?xml version="1.0"?>
<!DOCTYPE benchmark PUBLIC "+//IDN sosy-lab.org//DTD BenchExec benchmark 1.9//EN" "
http://www.sosy-lab.org/benchexec/benchmark-1.9.dtd">
<benchmark tool="cpachecker" timelimit="15 min" hardtimelimit="16 min" memlimit="15 GB"
cpuCores="8">
<require cpuModel="Intel Xeon E3-1230 v5 @ 3.40 GHz" cpuCores="8"/>
  <resultfiles>**.graphml</resultfiles>
  <option name="-svcomp19"/>
  <option name="-heap">10000M</option>
  <option name="-benchmark"/>
  <option name="-timelimit">900 s</option>
<rundefinition name="sv-comp19_prop-reachsafety">
  <tasks name="ReachSafety-Arrays">
    <includesfile>../sv-benchmarks/c/ReachSafety-Arrays.set</includesfile>
    <propertyfile>../sv-benchmarks/c/properties/unreach-call.prp</propertyfile>
  </tasks>
  <tasks name="ReachSafety-BitVectors">
    <includesfile>../sv-benchmarks/c/ReachSafety-BitVectors.set</includesfile>
    <propertyfile>../sv-benchmarks/c/properties/unreach-call.prp</propertyfile>
  </tasks>
```

Competition License

LICENSE FOR RESEARCH AND EVALUATION

[...]

The licensor grants to the user of this software the following rights, irrevocably:

1. Redistribution of exact copies of this archive.

2. Execution of this software for research and evaluation purposes.

3. Publication of the output and measurement results obtained from executing this software.

The licensor confirms that the licenses of all components contained in this archive are compatible with the above requirements.

Competition Guarantee

GUARANTEE OF RIGHTS FOR RESEARCH AND EVALUATION

The licensor guarantees that the licenses of all components contained in this archive grant the following perpetual, worldwide, no-charge, royalty-free, irrevocable rights to everybody:

1. Redistribution of exact copies of this archive.

2. Execution of this software for research and evaluation purposes.

3. Publication of the output and measurement results obtained from executing this software.

For all parts of the software for which the licensor holds the copyright, the licensor grants to the user of this software the above rights. This takes precedence over any contradicting clauses in accompanying licenses.

Results Details at: sv-comp.sosy-lab.org

Select Columns | Filter Rows | Quantile Plot | Scatter Plot | Shrink Header | Generated with BenchExec

2LS 0.5.0

timelimit: 900 s, memlimit: 15000 MB, CPU core limit: 8

apollon*

Linux 4.4.0-57-generic

CPU: Intel Xeon E3-1230 v5 @ 3.40 GHz, cores: 8, frequency: 3.8 GHz, Turbo Boost: disabled; RAM: 335

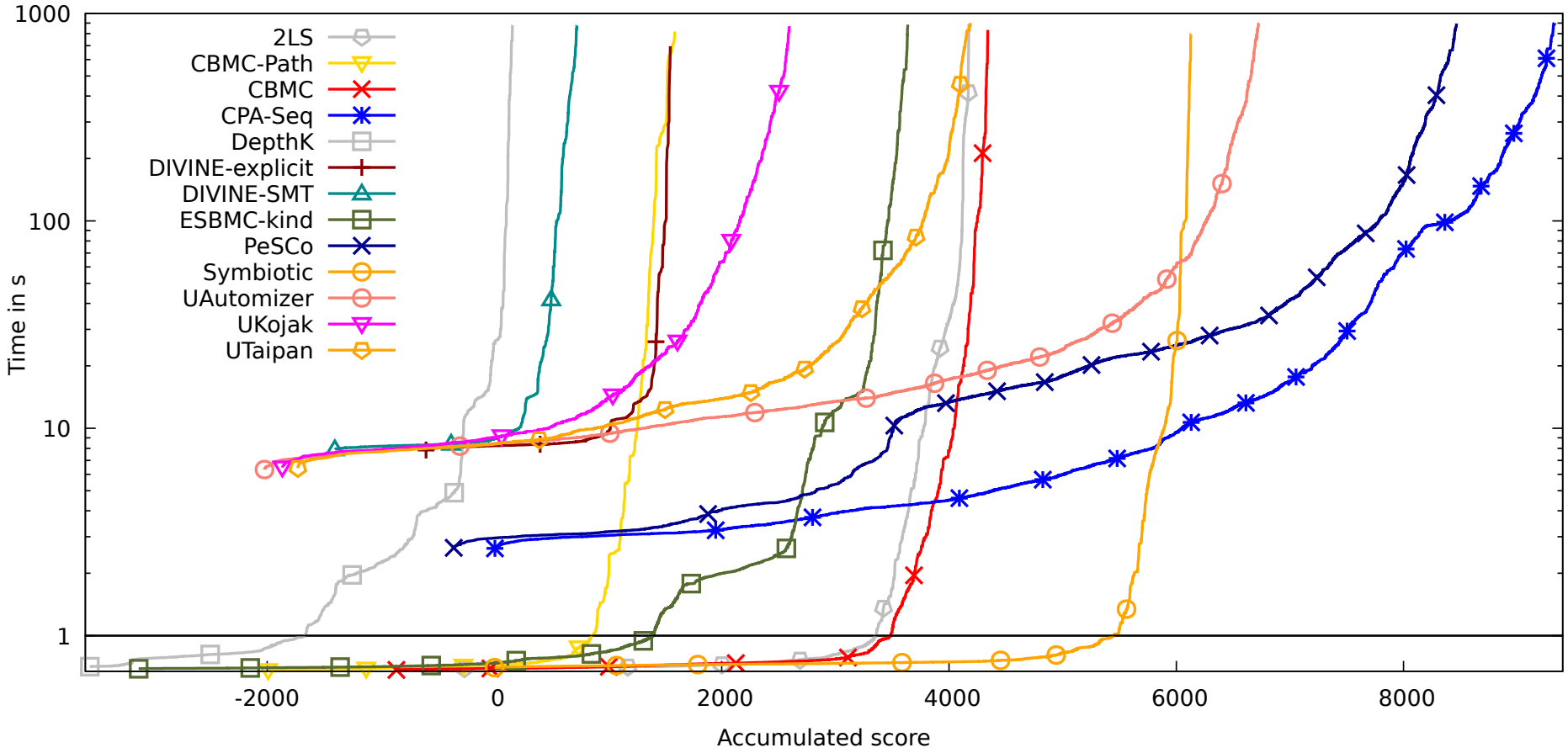
2017-01-10 17:21:21 CET [[2017-01-14 18:00:17 CET]] [[2017-01-14 20:02:31 CET]] [[2017-01-14 18:18:08 CET]] [[20

sv-comp17.ReachSafety-ControlFlow

```
--graphml-witness witness.graphml
[[
-witnessValidation
-setprop witness.checkProgramHash=false
-disable-java-assertions
-heap 10000m
-witness ../../results-verified/2ls.2017-01-10_1721.logfiles/sv-comp17.${inputfile_name}.files/witness.graphml
]]
[[
--validate ../../results-verified/2ls.2017-01-10_1721.logfiles/sv-comp17.${inputfile_name}.files/witness.graphml
]]
[[
-witnessValidation
-setprop witness.checkProgramHash=false
-disable-java-assertions
-heap 10000m
-witness ../../results-verified/2ls.2017-01-10_1721.logfiles/sv-comp17.${inputfile_name}.files/witness.graphml
]]
[[
--validate ../../results-verified/2ls.2017-01-10_1721.logfiles/sv-comp17.${inputfile_name}.files/witness.graphml
]]
```

	verifier status	score	witness	inspect witness	cpu (s)	wall (s)	energy (J)	mem (MB)	blkio-w (MB)	blkio-r (MB)	validator cpachecker violation t<90s status	cpu (s)	wall (s)	energy (J)	mem (MB)	validator uautomizer violation t<90s status	cpu (s)	wall (s)	energy (J)	mem (MB)	c
ination.cil.c	false(unreach-call)	1	wit	inspect	1.3	1.3	13	370	.0041	0	false(unreach-call)	8.2	4.4	120	320	false(unreach-call)	17	9.1	320	520	
nation.cil.c	false(unreach-call)	1	wit	inspect	.35	.34	3.3	60	.0041	0	false(unreach-call)	8.1	4.3	170	310	false(unreach-call)	13	6.6	240	450	
nation.cil.c	false(unreach-call)	1	wit	inspect	.55	.53	4.9	120	.0041	0	false(unreach-call)	8.3	4.4	100	330	false(unreach-call)	12	6.6	210	500	
ination.cil.c	false(unreach-call)	1	wit	inspect	.29	.28	2.4	39	.0041	0	false(unreach-call)	7.8	4.2	80	380	false(unreach-call)	13	6.7	180	410	
nation.cil.c	true	2	wit	inspect	1.4	1.4	14	410	.0041	12	-	–	–	–	–	-	–	–	–	–	
ination.cil.c	true	2	wit	inspect	.53	.53	4.9	110	.0041	0	-	–	–	–	–	-	–	–	–	–	
ation.cil.c	true	2	wit	inspect	.36	.35	3.3	67	.0041	0	-	–	–	–	–	-	–	–	–	–	
ation.cil.c	true	2	wit	inspect	.59	.58	5.6	130	.0041	0	-	–	–	–	–	-	–	–	–	–	
nation.cil.c	true	2	wit	inspect	.19	.19	1.4	26	.0041	0	-	–	–	–	–	-	–	–	–	–	
nation.cil.c	true	2	wit	inspect	.29	.29	2.3	45	.0041	0	-	–	–	–	–	-	–	–	–	–	
	false(unreach-call)	1	wit	inspect	21	21	200	200	.0041	0	false(unreach-call)	7.4	3.9	170	310	false(unreach-call)	12	6.8	210	460	
	false(unreach-call)	1	wit	inspect	23	23	200	200	.0041	0	false(unreach-call)	7.4	3.9	130	310	false(unreach-call)	14	7.1	200	480	

Results – Example: Overall



'Value' of result is defined by Scoring Schema

Impact / Achievements

- Large benchmark set of verification tasks
  → established and used in many papers
  for experimental evaluation

- Good overview over state-of-the art
  → covers model checking and program analysis

- Participants have an archived track record
  of their achievements

- Infrastructure and technology for
  controlling the benchmark runs (cf. StarExec)

[Competition Report and System Descriptions
are archived in Proceedings TACAS '17]

Witness-Based Result Validation

Table 8: Confirmation rate of witnesses

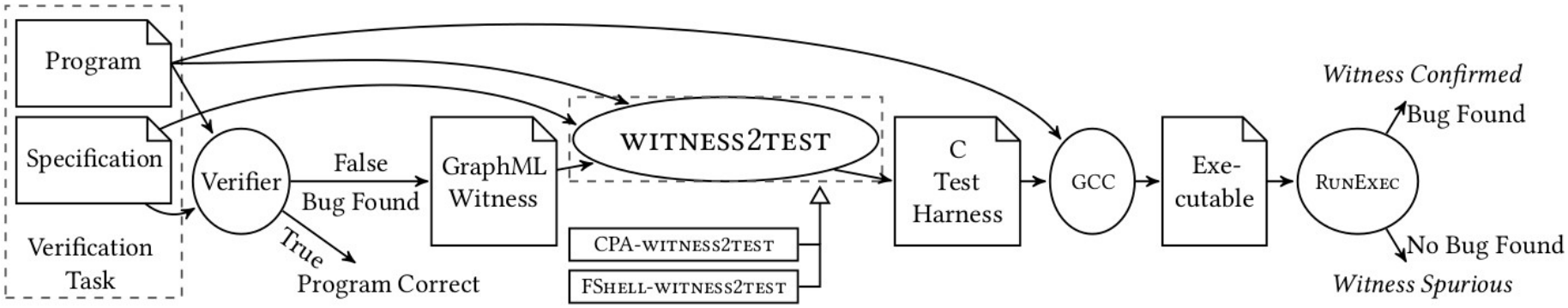
Result	TRUE			FALSE		
	Total	Confirmed	Unconfirmed	Total	Confirmed	Unconfirmed
UAUTOMIZER	3 558	3 481	77	1 173	1 121	52
SMACK	2 947	2 695	252	1 929	1 768	161
CPA-SEQ	3 357	3 078	279	2 342	2 315	27

Verifiable Witnesses. For SV-COMP, it is not sufficient to answer with just TRUE or FALSE: each answer must be accompanied by a verification witness. For correctness witnesses, an unconfirmed answer TRUE was still accepted, but was assigned only 1 point instead of 2 (cf. Table 2). All verifiers in categories that required witness validation support the common exchange format for violation and correctness witnesses. We used the two independently developed witness validators that are integrated in CPACHECKER and UAUTOMIZER [7, 8].

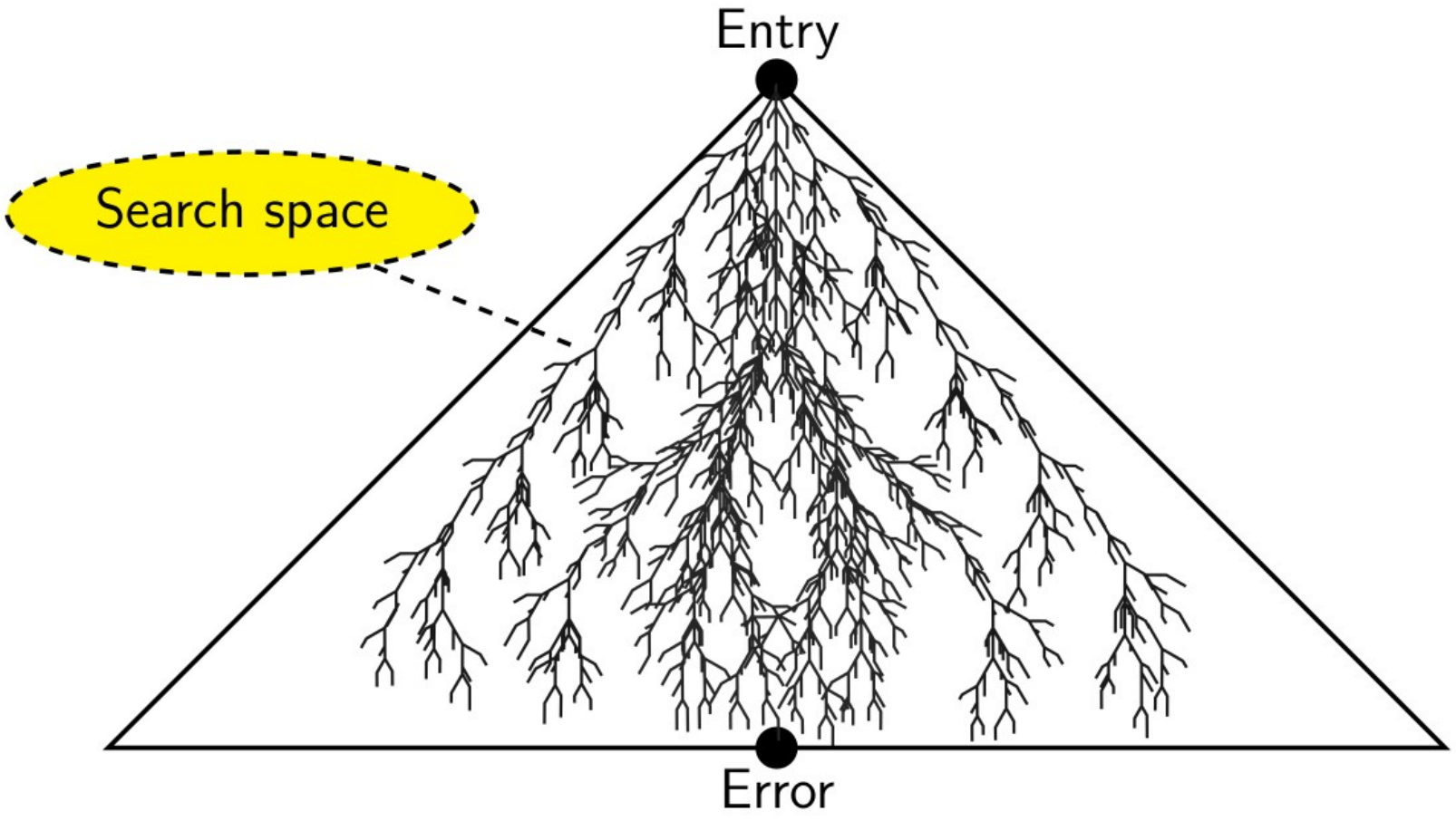
SV-COMP 2020

- License
- More programs
- LTL properties
- Eliminate pre-processing
- Undefined behavior of C programs
- Witnesses in all categories
- Tests as Witnesses

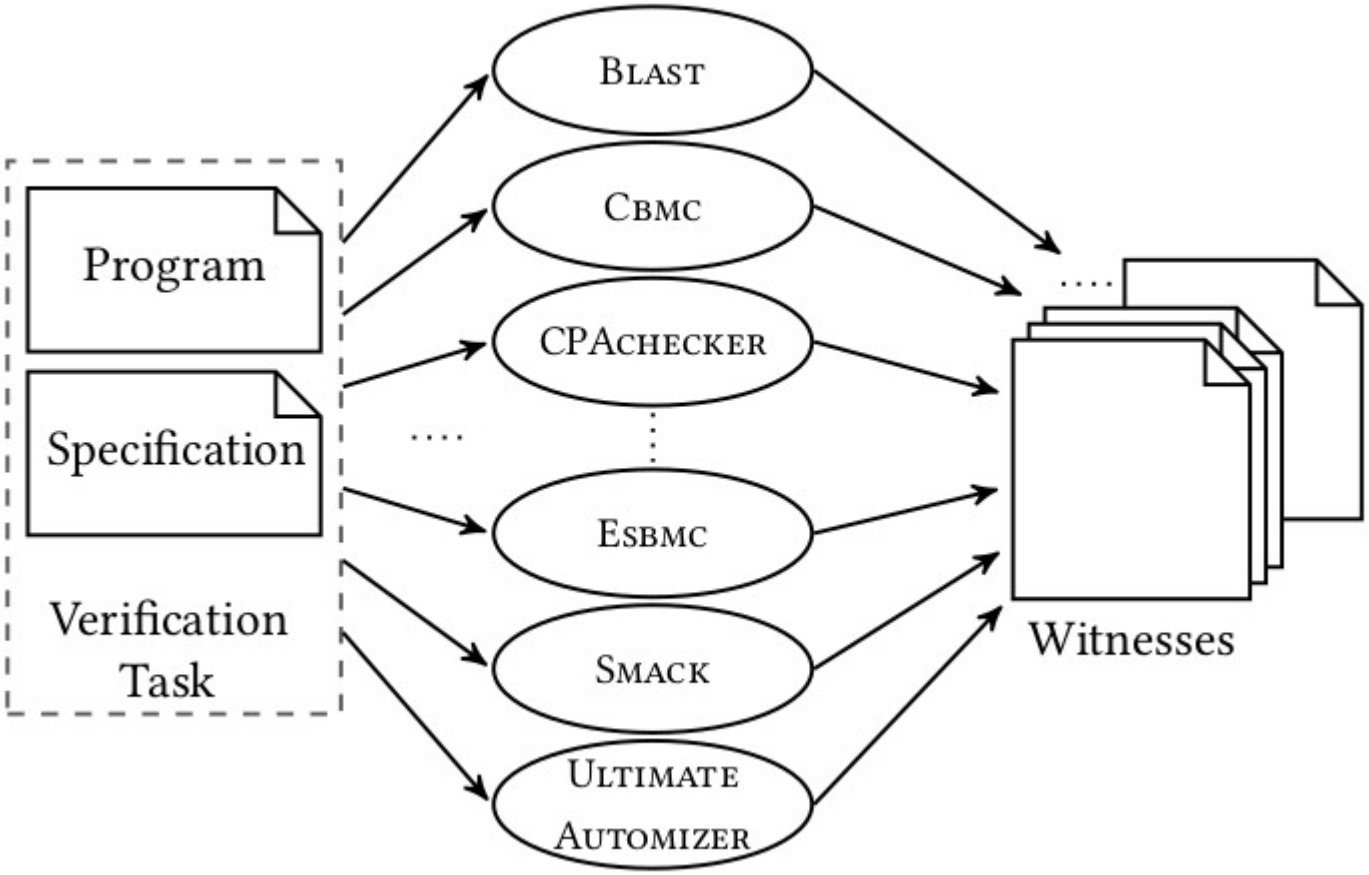
Practical Impact: Get Tests from Verification Tools



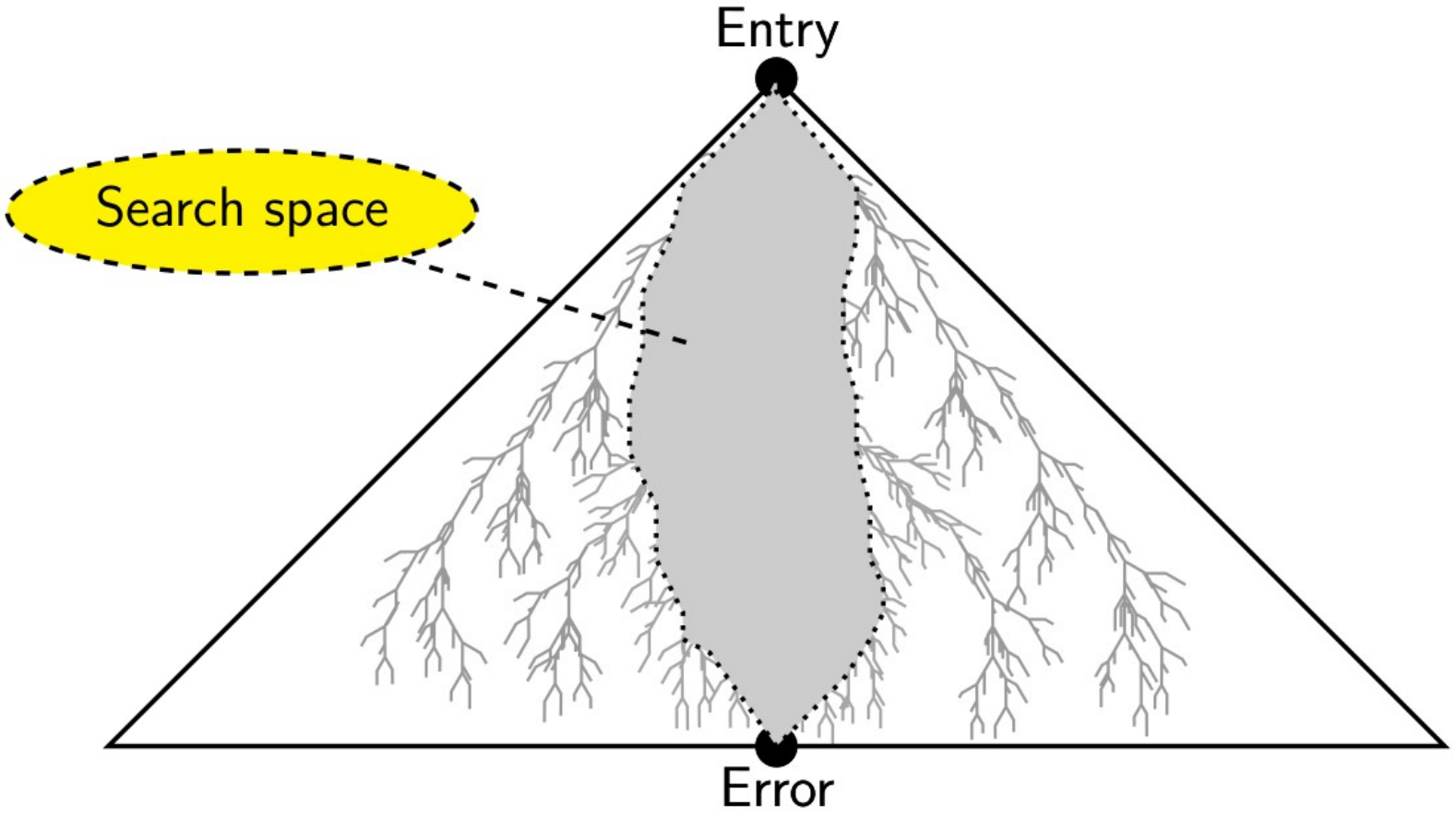
Search-Space Reduction for Stepwise Testification



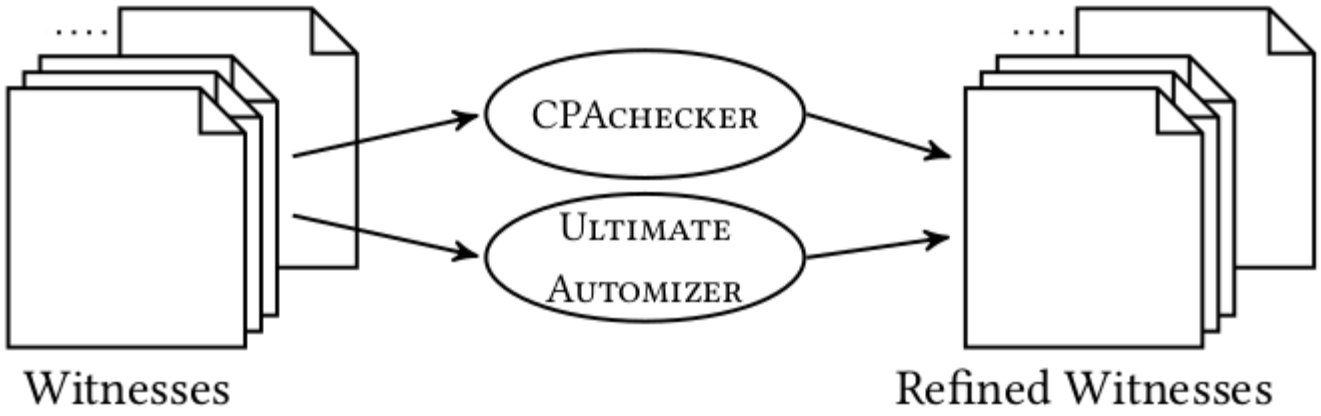
Produce Witnesses



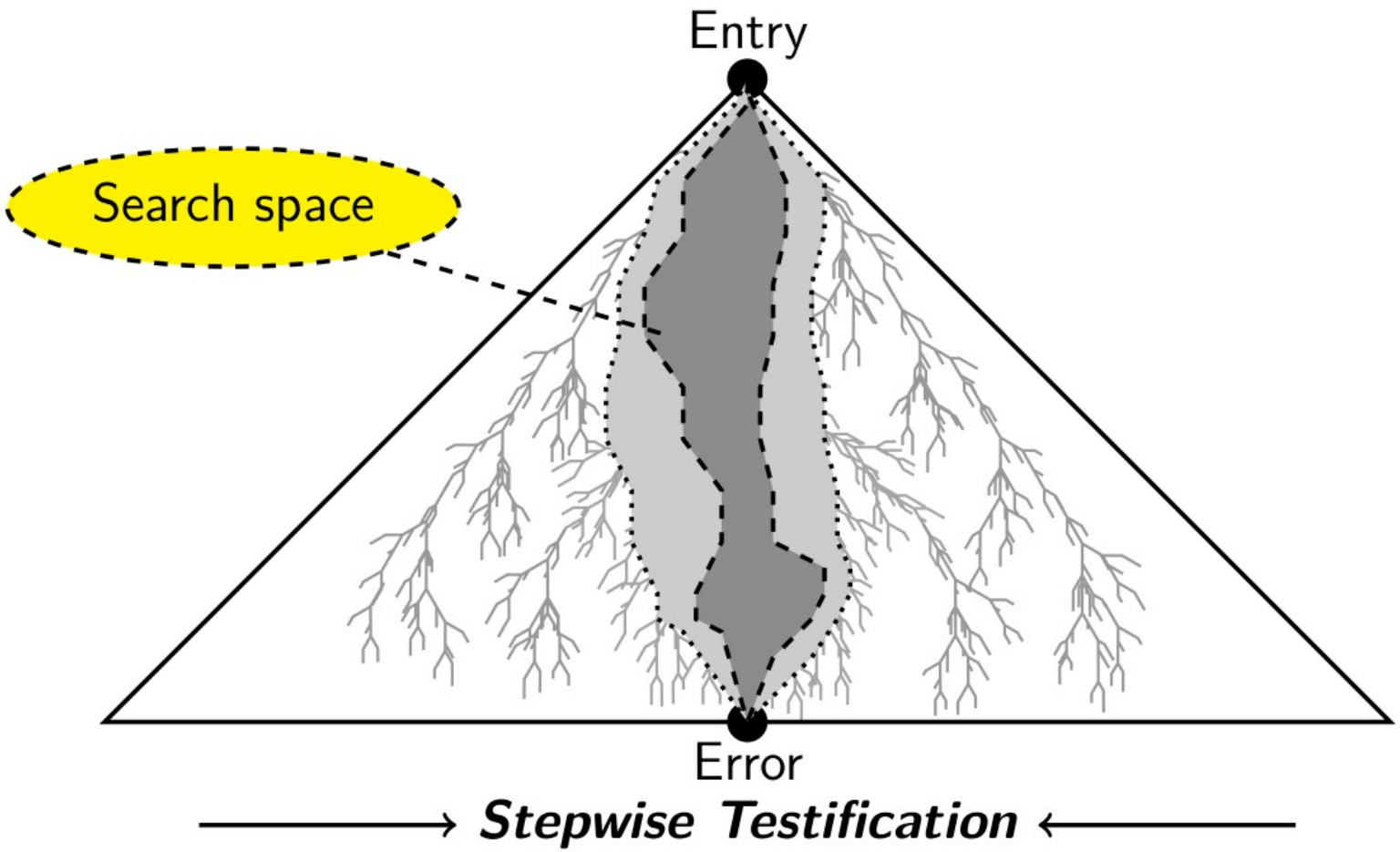
Search-Space Reduction for Stepwise Testification



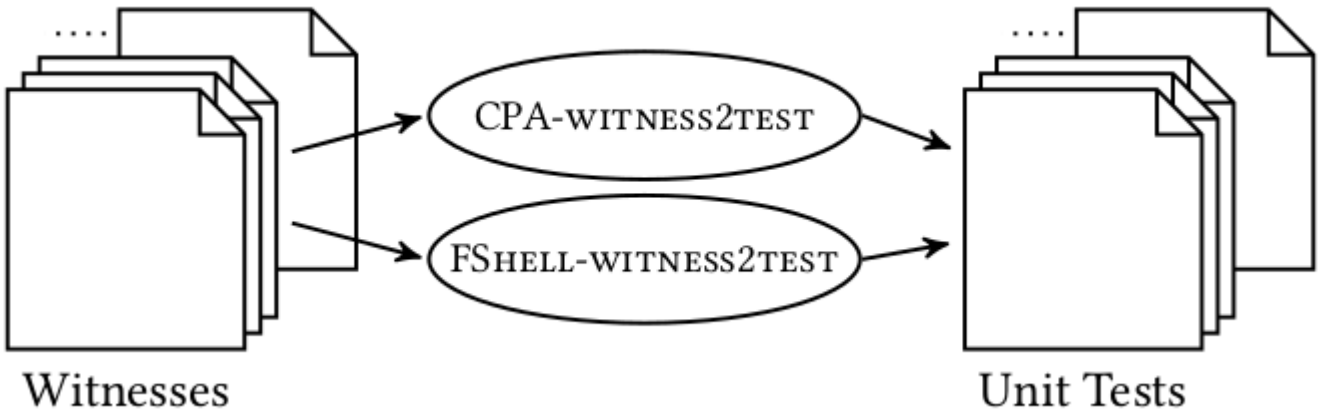
Refine Witnesses



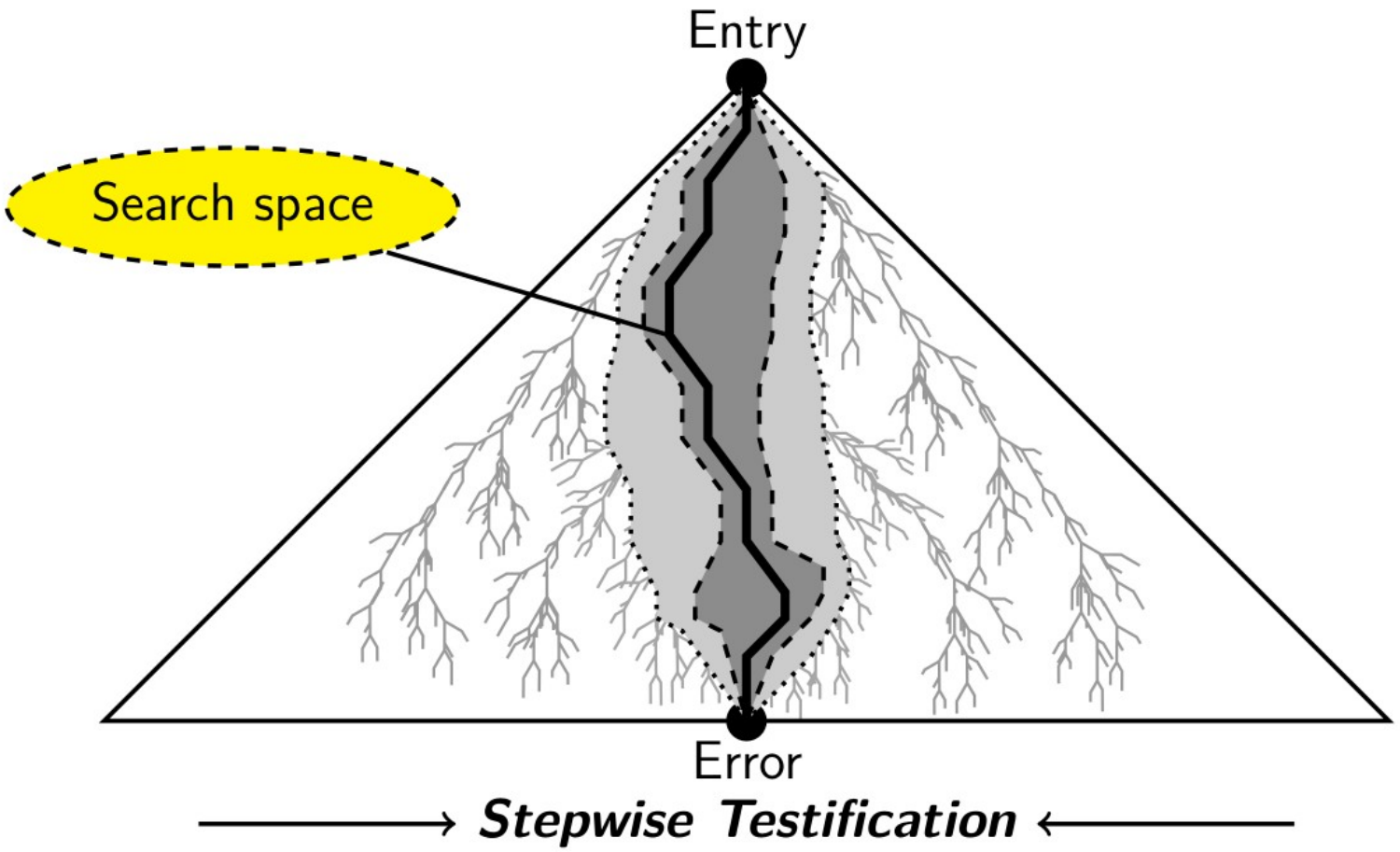
Search-Space Reduction for Stepwise Testification



Produce Unit Tests From Witnesses



Search-Space Reduction for Stepwise Testification



Thanks to:

- TACAS (PC Chairs + TACAS SC, thanks!)
- Jury (32 people)
- Participants (177 people)
- Sponsors: Amazon Web Services

  LMU Munich