CHC COMP

2025 Extended Report

https://chc-comp.github.io/

presented at
HCVS 2025, July 22, Zagreb, Croatia

Gidon Ernst, LMU Munich, Germany
Jose F. Morales, IMDEA Software Institute, Spain
Levente Bajczi, Budapest University of Technology and Economics

Goals & Overview

- CHC-COMP: friendly but competitive evaluation of constrained Horn-clause solvers, since 2018
https://chc-comp.github.io/
- common task format (subset of SMT-LIB)
https://chc-comp.github.io/format.html
- public benchmark repository (please submit!)
https://github.com/chc-comp
- Timeline: Jan—May, results presented (at SPIN and) HCVS

Setup and Updates in 2025

SoSy-Lab
Software Systems

- Move from StarExec to LMU cluster (SV-COMP infrastructure)
  - Intel Xeon E3-1230 v5 @ 3.40 GHz, 8 cores, 30 GB memory, 1800s
  - https://github.com/chc-comp/chc-comp25-scripts

- Add **BV, LRA-Lin** tracks, evaluate **all feasible benchmarks**
  - https://github.com/chc-comp/chc-comp25-benchmarks
  - overall CPU time: ~half a year (including some trial&error)

- Model validation done for HCVS@CAV 2025 (**thanks to Levente**)
- **thanks to Dirk Beyer and members of SoSy Lab for support**

2025: Participants

1. CHC2C 1.0 (Mihály Dobos-Kovács, Levente Bajczi, András Vörös), Note: meta-solver
2. ChocoCatalia (Hiroyuki Katsura, Naoki Kobayashi, Ryosuke Sato)
3. Eldarica 2.2 (Hossein Hojjat, Philipp Ruemmer)
4. Golem 0.7.1 (Martin Blicha)
5. LoAT (Florian Frohn, Jürgen Giesl)
6. MuCyc (Kazuki Uehara, Hiroshi Unno)
7. PCSat (Takuma Monma, Hiroshi Unno)
8. ThetaCHC 6.13.2 (Levente Bajczi, Mihály Dobos-Kovács, Márk Somorjai, András Vörös)
9. Ultimate Tree Automizer (Matthias Heizmann, Max Barth, Daniel Dietsch, Dominik Klumpp)
10. Ultimate Unihorn (Matthias Heizmann, Max Barth, Daniel Dietsch, Dominik Klumpp)

	LIA-Lin	LIA	LIA-Lin-Arrays	LIA-Arrays	ADT-LIA	ADT-LIA-Arrays	BV	LRA-Lin
	1312	1266	139	1728	3585	1045	559	274
CHC2C (meta)	✓	✓						
ChocoCatalia					✓			
MuCyC	✓	(*)						
PCSat	✓	✓	✓	✓	✓	✓	✓	✓
Eldarica	✓	✓	✓	✓	✓	✓	✓	✓
Golem	✓	✓	✓					✓
LoAT	✓							
Theta	✓	✓	✓	✓			✓	✓
U. Tree Automizer	✓	✓	✓	✓				
Ultimate Unihorn	✓	✓	✓	✓				

Z3/Spacer was not submitted

(*) I somehow missed running this (apologies!)

2025: Winners*

LIA-Lin	LIA	LIA-Lin-Arrays**	LIA-Arrays	ADT-LIA	ADT-LIA-Arrays	BV	LRA-Lin
Golem	Golem	Eldarica	Eldarica	Catalia	Eldarica	Eldarica	Golem
MuCyc	Eldarica	Unihorn	PCSat	Eldarica	PCSat	Theta	Eldarica
LoAT	PCSat	PCSat	Unihorn	PCSat	--	PCSat	Theta

*accepting all results as correct even though there are definitely inconsistencies

**ranking changed after cleanup of results

2025: Infrastructure

https://gitlab.com/sosy-lab/software/benchcloud

benchexec: resource control, measurements [Beyer+ STTT19]

- Python toolinfo modules as API for tools (provided by organizer)
  Examples: **chc.py** and **eldarica.py**
- XML benchmark definition files to set up experiments (provided)
  Example: **golem.xml**

Should establish community process to maintain these s part of **FM tools**|
(→ akin to SV-COMP but keep it light-weight!)

benchcloud: cluster management, job scheduling [Beyer+ ASE24]

- deployed at LMU and also on a Hungarian cluster (this helped!)

Pitfall: duplicate benchmark names not supported (→ design decision)

2025: Scripts

https://github.com/chc-comp/chc-comp25-scripts

Tool-chain accumulated over past years

- **`format.py`** ensure CHC-COMP requirements, obfuscation
  (has several problems, should be replaced next year)
- **`classify.py`** determine benchmark's categories
  (new! replaces previous slow/inflexible `format-checker`)
- scripts to set up benchexec metadata `.yml` files
  → cross-check result, found and fixed inconsistencies (bugs)

- **Lacking: fully documented pipeline**

2025: Benchmarks

https://github.com/chc-comp/chc-comp25-benchmarks

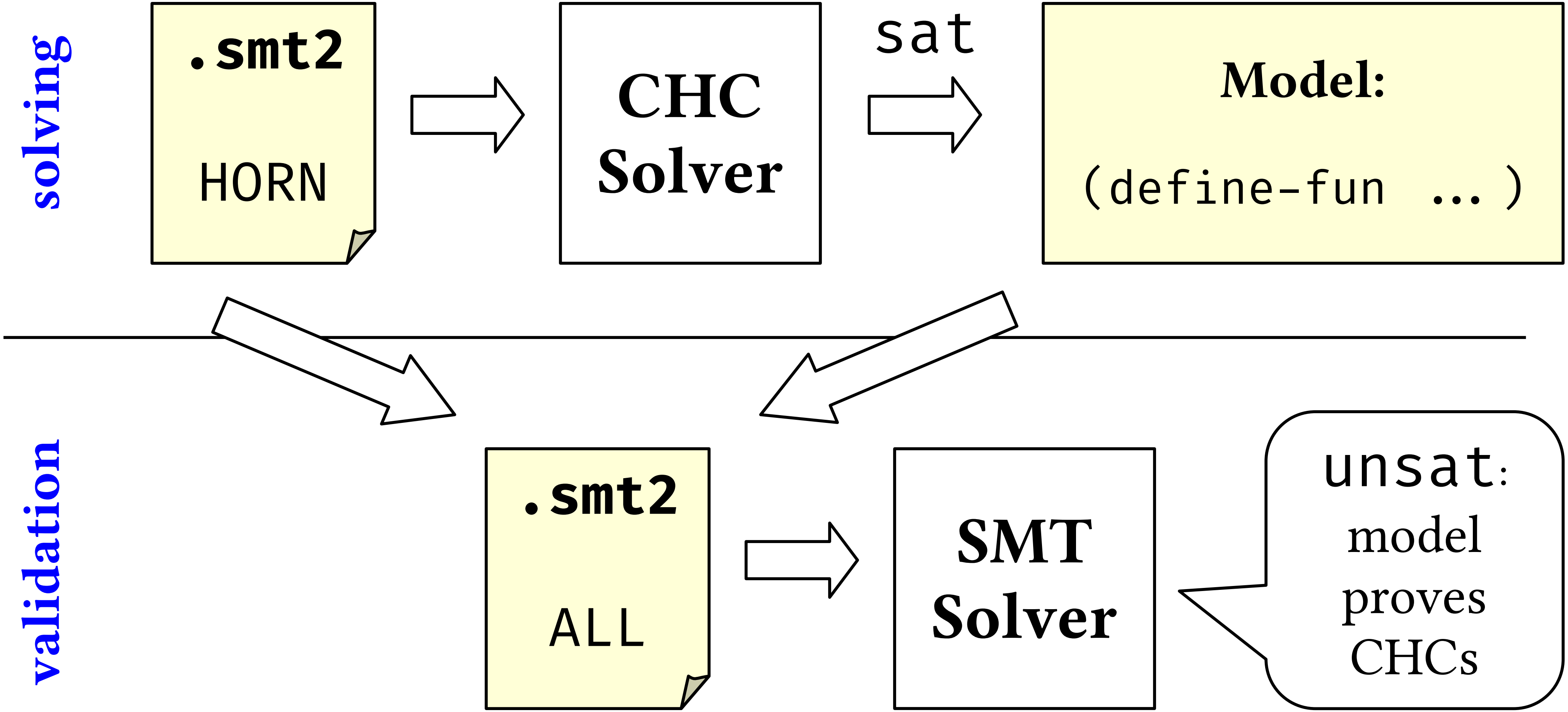
Benchmarks accumulated over past years

- fairly diverse in features and in difficulty
- organization into individual sub-repositories with “raw” sources

- not “ready to use”: format inconsistencies, lack of ground truth
- unclear which benchmarks are “interesting”
- preprocessing pipeline fails on 2/3 of files (⁉)

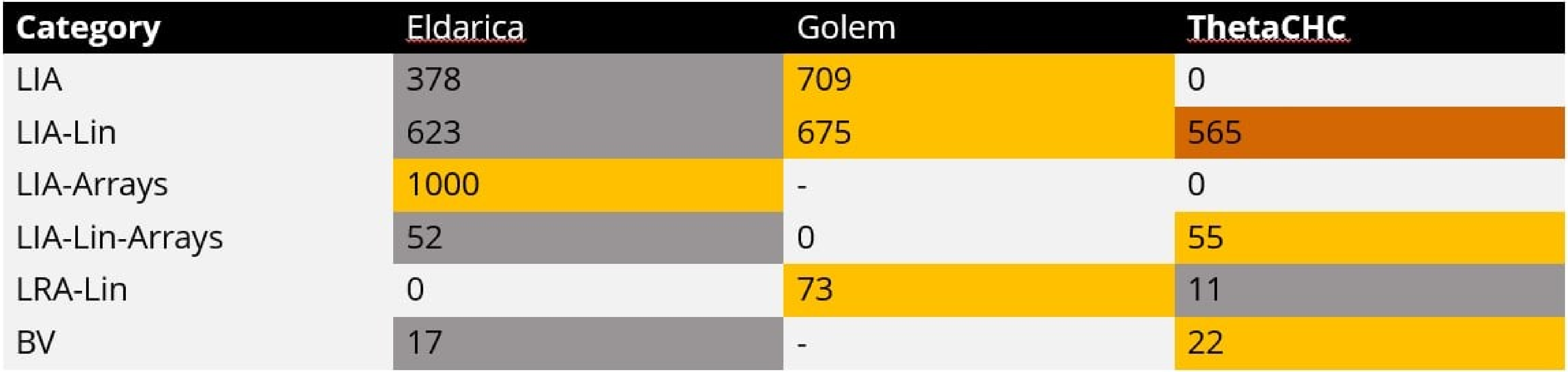
Please contribute benchmarks!

2025: Validation (`validate-model.py`)



2025: Model Validation Results

https://leventebajczi.com/chc-model-artifact/

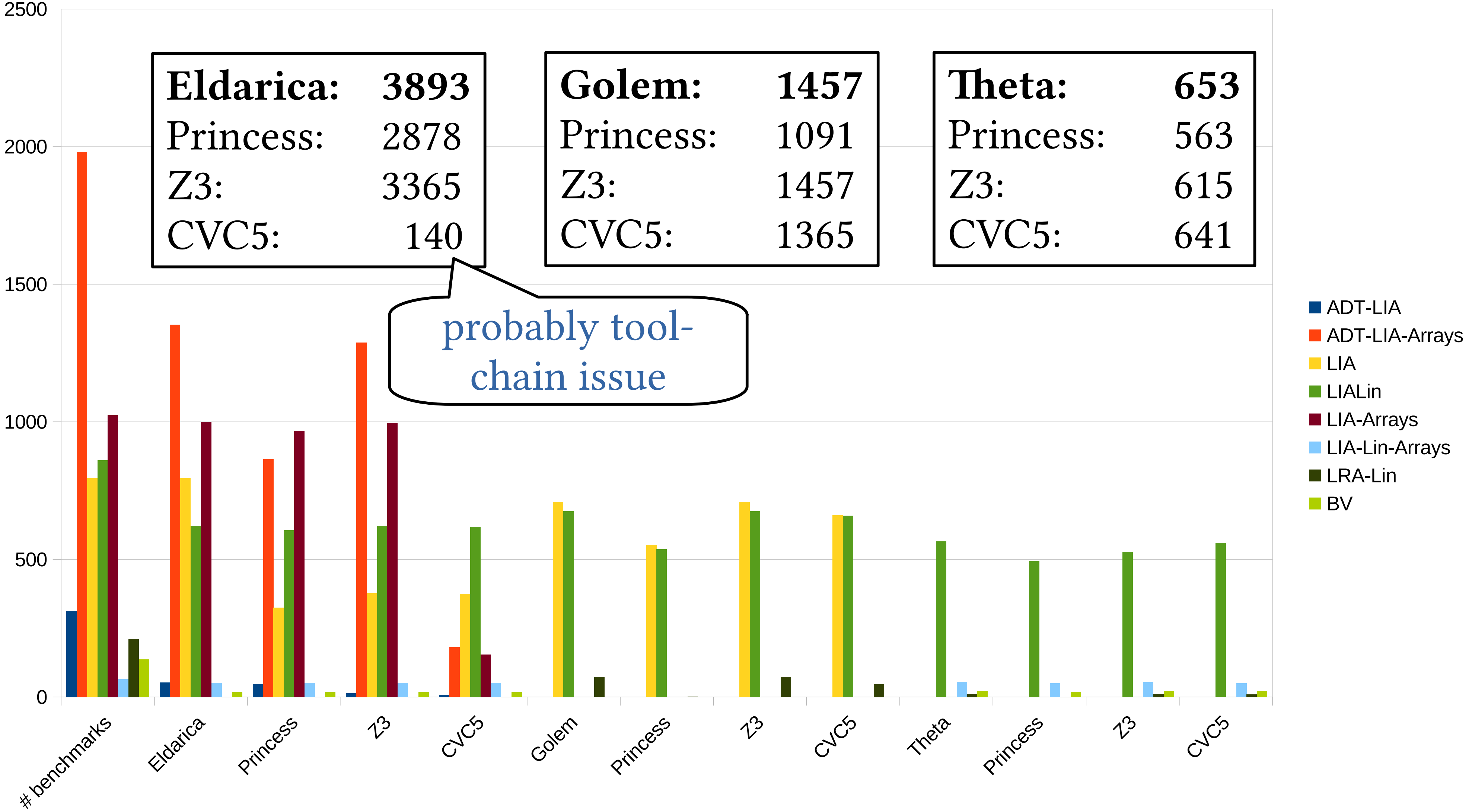


Category	Eldarica	Golem	ThetaCHC
LIA	378	709	0
LIA-Lin	623	675	565
LIA-Arrays	1000	-	0
LIA-Lin-Arrays	52	0	55
LRA-Lin	0	73	11
BV	17	-	22

points are awarded for correct results (wrt. ground-truth) for which at least one SMT solver confirms the model

2025: Model confirmation (36%-100%)

https://leventebajczi.com/chc-model-artifact/



Eldarica:	**3893**
Princess:	2878
Z3:	3365
CVC5:	140

Golem:	**1457**
Princess:	1091
Z3:	1457
CVC5:	1365

Theta:	**653**
Princess:	563
Z3:	615
CVC5:	641

probably tool-chain issue

Summary and Outlook

CHC COMP

2025 Achievements (modulo small hick-ups and lessons learned)

- infrastructure switch `StarExec` → `benchcloud`
- new benchmarks (LRA, BV)
- first steps towards model validation

Follow-up

- document scripts and pipeline, write a report (with fixed results?)
- discuss benchmark repository (ground truth, pre-processing, file names)
- counterexample proof format and validation
- massively parallel track? “best-effort” track without goal clauses?

Partitipants: publish your archive on zenodo, make an entry in **FM-Tools**

Organizers of the next edition? I’m happy to help with onboarding :)