Diagnosis via Proofs of Unsatisfiability for First-Order Logic with Relational Objects

Nick Feng, **Lina Marsso**, and Marsha Chechik

ASE 2024

October 30, 2024

UNIVERSITY OF TORONTO

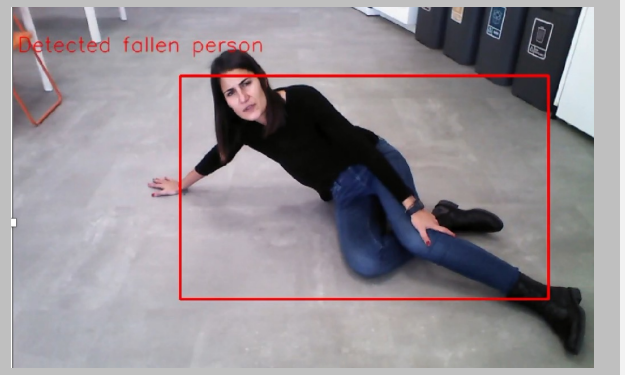
Context

Systems increasingly interacting with humans in various domains

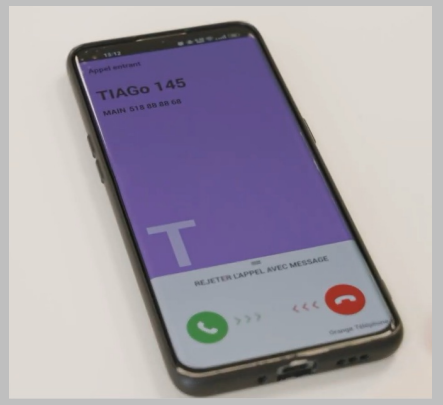
(transport, environment, health and social care)

ALMI: Assistive-care robotics

Helps with food preparation, dressing, fallen-user alert, etc.



Detect the user has fallen



Alert that the user has fallen

ALMI robot from RoboStar (University of York, UK)

Normative requirements

- Capture **s**ocial, **l**egal, **e**thical, **e**mpathetic, **c**ultural (**SLEEC**) aspects of systems

- Specified by stakeholders with non-technical expertise
  - Lawyer, regulators, ethicists, etc.

- Hard to get right
  - Stakeholders from different fields, different vocabularies
  - Their views are often conflicting or redundant
  - Stakeholders might not have sufficient technical background to reason about requirements
  - Requirements are complex: Allow constraints over data and time

Normative requirements well-formedness analysis

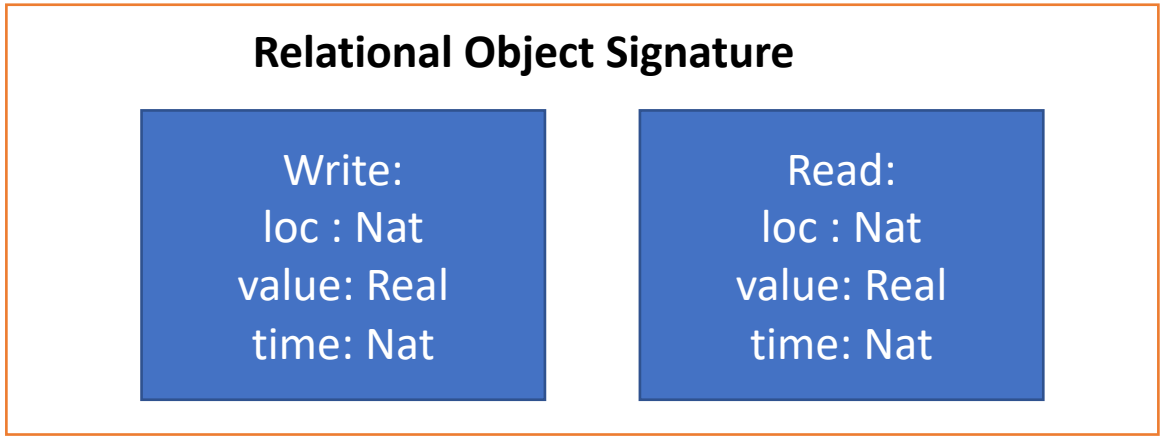
- SLEEC, a normative requirement DSL [FASE23]
- Translate SLEEC rules to **FOL***[ASE23]
- Well-formedness properties [ICSE24]:
  Conflicting, restrictive or Insufficient requirements, or unnecessary redundant
- LEGOS-SLEEC checks requirements well-formedness [ICSE24]
  (via FOL* satisfiability checking [CAV23])

[FASE23] S. Getir-Yaman, C. Burholt, M. Jones, R. Calinescu, and A. Cavalcanti. "Specification and Validation of Normative Rules for Autonomous Agents", FASE 2023.
[ASE23] N.Feng, L.Marsso, S. Yaman, B. Townsend, R. Calinescu, A. Cavalcanti, M. Chechik ``*Towards a Formal Framework for Normative Requirements Elicitation*". ASE/NIER'23
[CAV23] N. Feng, L. Marsso, M. Sabetzadeh, and M. Chechik. "*Early verification of legal compliance via bounded satisfiability checking*", CAV 2023.
[ICSE24] N. Feng, L. Marsso, S. Yaman, B. Townsend, Y. Baatartogtokh, R. Ayad, V. Mello, I. Standen, I. Stefanakos, C. Imrie, G. Rodrigues, A. Cavalcanti, R. Calinescu, and M. Chechik. "*Analyzing and Debugging Normative Requirements via Satisfiability Checking*", ICSE2024 – **ACM SIGSOFT Distinguished Paper Award**.

FOL*

First order logic with quantifiers over relational objects [CAV23]

Used to specify time- and data-sensitive declarative software requirements



R1: Every value written must be eventually read within 30 seconds.

R1 = ∀w: Write(loc, value, time) (∃ r:Read(loc, value, time)
r.loc = w.loc AND r.value = w.value AND
w.time < r.time <= w.time + 30s)

[CAV23] N. Feng, L. Marsso, M. Sabetzadeh, and M. Chechik. "Early verification of legal compliance via bounded satisfiability checking", CAV 2023.

Well-formedness analysis via FOL* satisfiability

Let Rules = {R1, R2, R3 … R5} **in FOL***

R5 is redundant if and only if

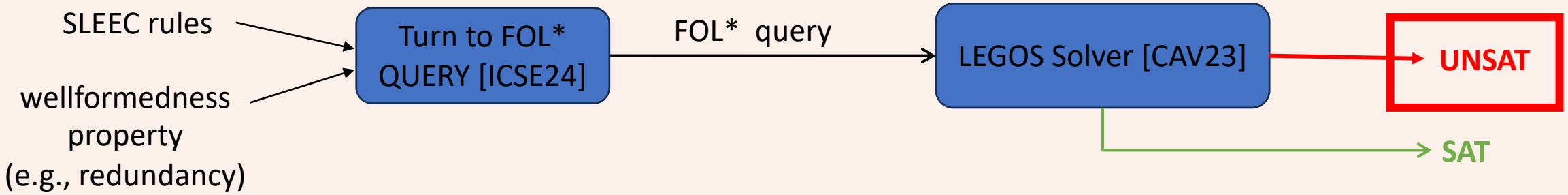
R5 is implied by the rest

FOL* Redundancy Query: (Rules \ R5) **AND NOT**(R5) is **UNSAT**

But which rules specifically are redundant with R5?

From SLEEC to FOL* UNSAT core

FOL* Redundancy Query: (Rules \ R5) **AND NOT**(R5) is **UNSAT**



LEGOS incrementally searches for a satisfying solution for the FOL* formulas *(over expanding domains of relational objects)* and grounds FOL* query

The solver produces a lot of info, in another language, with possibly buggy implementation

[CAV23] N. Feng, L. Marsso, M. Sabetzadeh, and M. Chechik. *"Early verification of legal compliance via bounded satisfiability checking"*, CAV 2023.

[ICSE24] N. Feng, L. Marsso, S. Yaman, B. Townsend, Y. Baatartogtokh, R. Ayad, V. Mello, I. Standen, I. Stefanakos, C. Imrie, G. Rodrigues, A. Cavalcanti, R. Calinescu, and M. Chechik. "*Analyzing and Debugging Normative Requirements via Satisfiability Checking*", ICSE2024 –

Diagnosis via Proofs of Unsatisfiability for First-Order Logic with Relational Objects

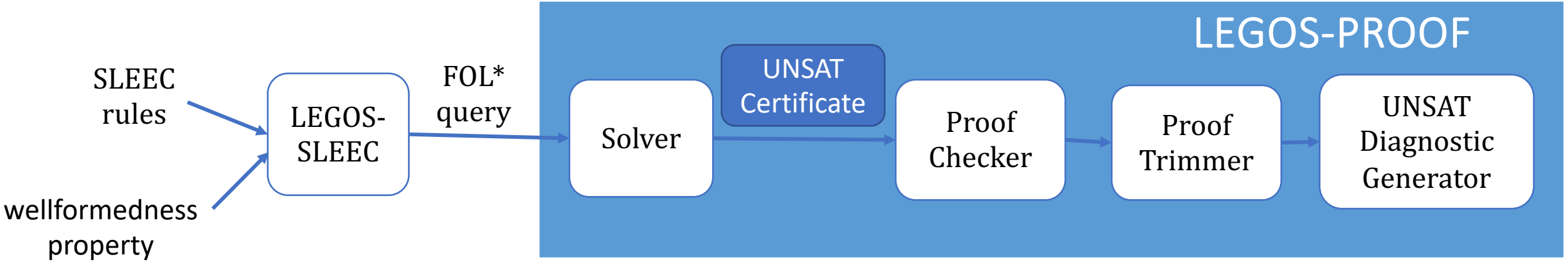
Challenges:

Legos solver produces a lot of info, in another language, with possibly buggy implementation

Our solution:

Derive and trim proof of UNSAT for FOL* to

- enable checking the UNSAT claim's correctness
- explain the unsatisfiability (diagnosis) in SLEEC



Goal: Diagnosis

Example: Why is R5 redundant?

SLEEC:

Rules

R1 **when** DressingStarted **then** DressingComplete **within** 120 **sec**
unless (roomTemperature < 19) **then** DressingComplete **within** 90 **sec**
unless (roomTemperature < 17) **then** DressingComplete **within** 60 **sec**

R2 **when** CurtainOpenRqt **then** CurtainsOpened **within** 60 **sec**

R3 **when** UserFallen **then** SupportCalled **unless** (**not** assentToSupportCalls)

R4 **when** DressingAbandoned **then** RetryAgreed **within** 30 **sec**

R5 when DressingStarted and (roomTemperature > 20) then DressingComplete within 120 sec

Diagnosis via Proofs of Unsatisfiability for First-Order Logic with Relational Objects

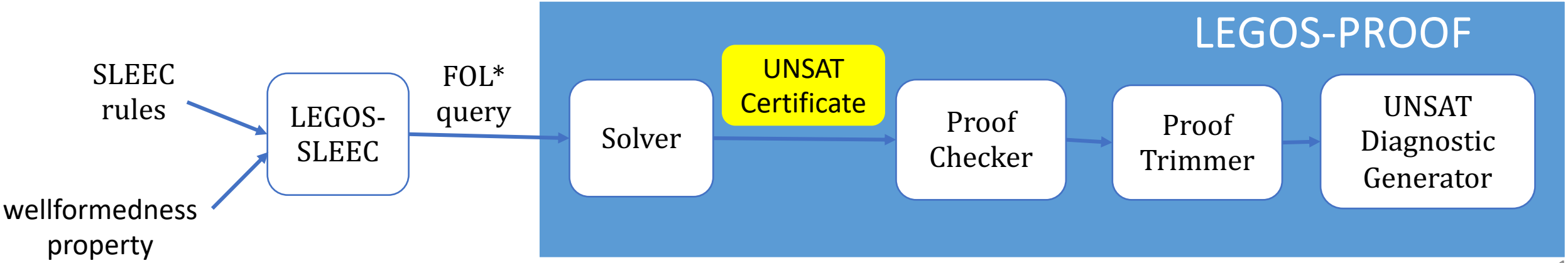
Challenges:

Legos solver produces a lot of info, in another language, with possibly buggy implementation

Our solution:

Derive and trim proof of UNSAT for FOL* to

- enable checking the UNSAT claim's correctness
- explain the unsatisfiability (diagnosis) in SLEEC



FOL* UNSAT proof certificate 1/2

Create UNSAT proof certificate by recording UNSAT derivation steps (using FOL* derivation rules)

FOL* Derivation rule example

ExistentialInst (EI): Instantiate an existential formula with a fresh new relational object outside of the domain D

The effect of an application of a derivation rule

Lemma: FOL* formula

Fact: quantifier-free clause in over-approximated query

Relational object in *D*

The name of the application rule being applied

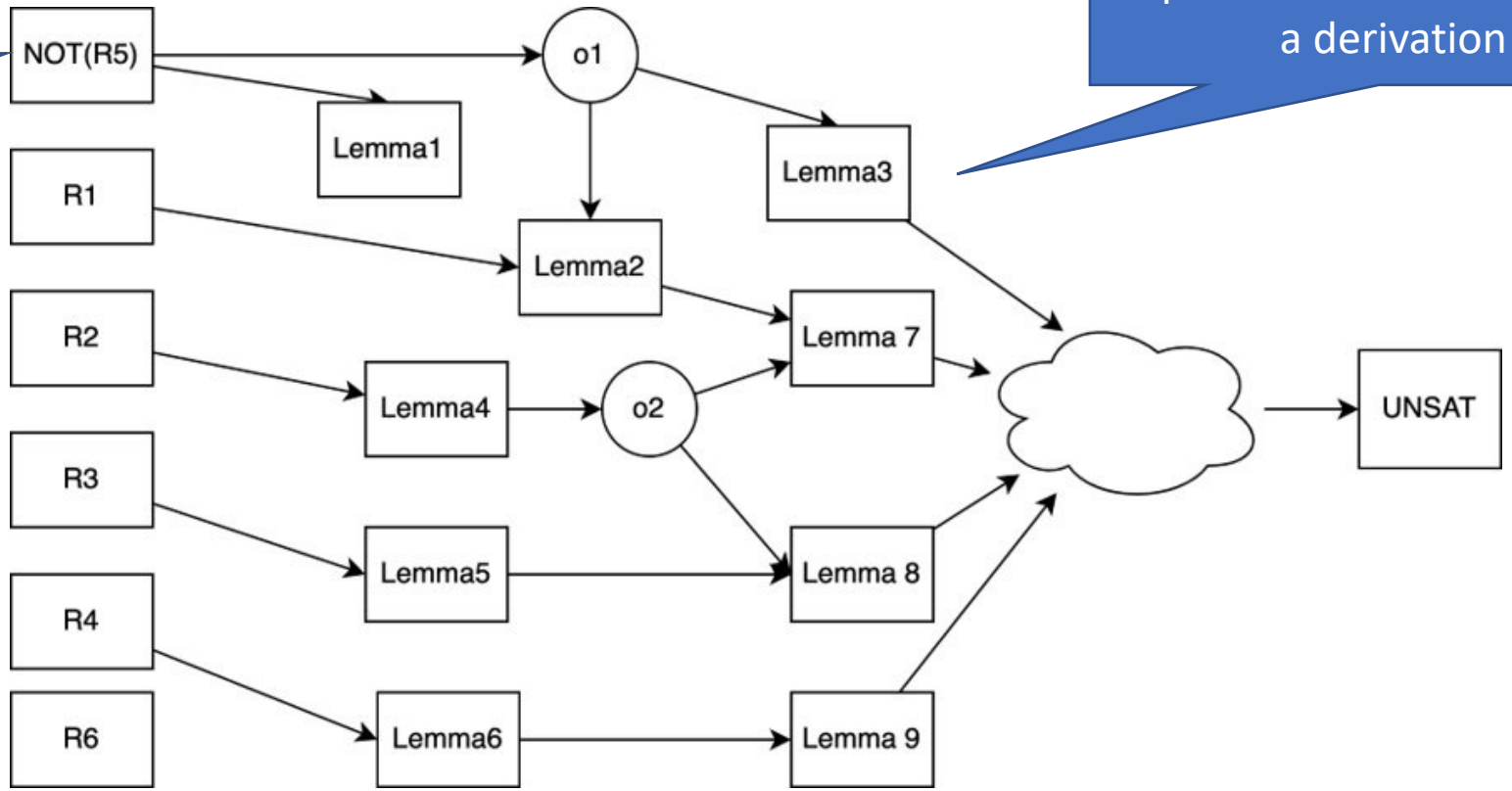
Step	Effects (lemma\fact\object)	Derivation rule	Dependency
1	$\exists\, r: Read(r.time > 5$ …)	Input	{}
2	$r_1.time > 5$ …	EI[$r \leftarrow r_1$]	{1}
…			
N	UNSAT	Implication	{2, 5, 7 …}

The dependent steps for enabling the derivation

FOL* UNSAT proof certificate 2/2

Each derived effect is a node

Incoming edges are the dependencies of an application of a derivation rule



Diagnosis via Proofs of Unsatisfiability for First-Order Logic with Relational Objects

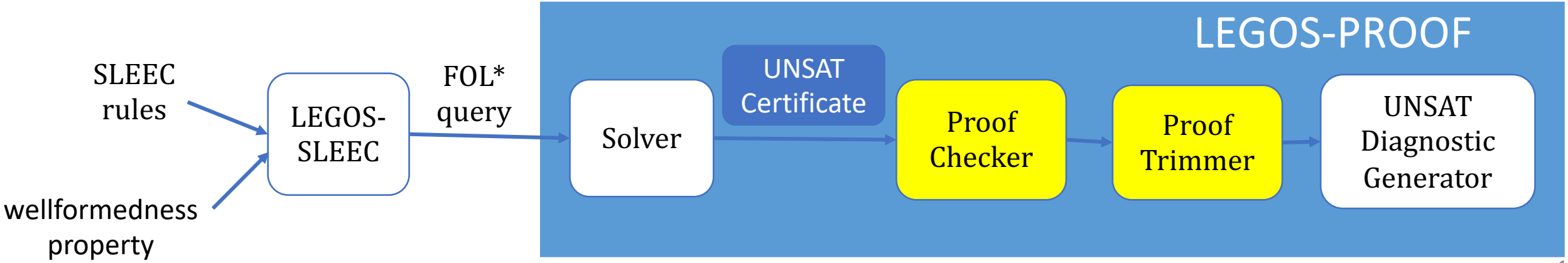
Challenges:

Legos solver produces a lot of info, in another language, with possibly buggy implementation

Our solution:

Derive and trim proof of UNSAT for FOL* to

- enable checking the UNSAT claim's correctness
- explain the unsatisfiability (diagnosis) in SLEEC



Checking UNSAT proof correctness and trimming 1/2

Validate the soundness of LEGOS implementation

Trim UNSAT FOL* **proof certificates** to only contain **relevant information**

Step 3: Recursively check all rules in the dependency set in the reverse step order

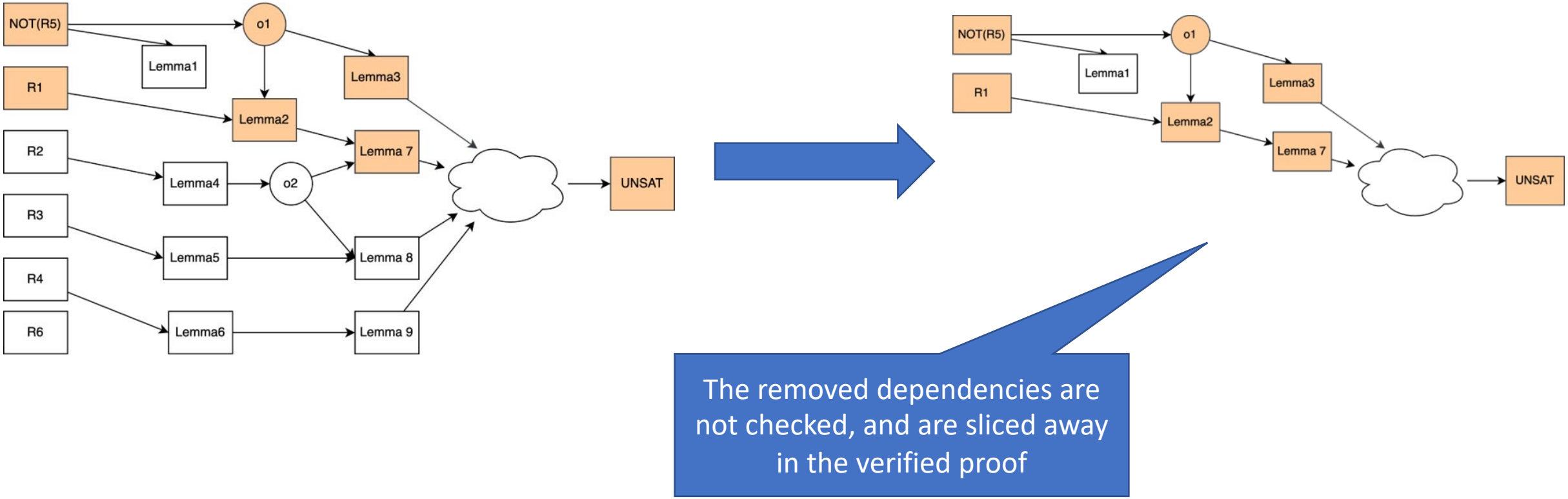
Step	Effects (lemma\fact\object)	Derivation rule	Dependency
1	$\exists\, r\colon Read(r.time > 5$ …)	Input	{}
2	$r_1.time > 5$ …	EI[$r \leftarrow r_1$]	{1}
…			
N	UNSAT	Implication	{2, 5, 7 …}

Start from the derivation of UNSAT

Step 1: Check the validity of the application of the rule

Step 2: minimize the dependency set

Checking UNSAT proof correctness and trimming 2/2



Diagnosis via Proofs of Unsatisfiability for First-Order Logic with Relational Objects

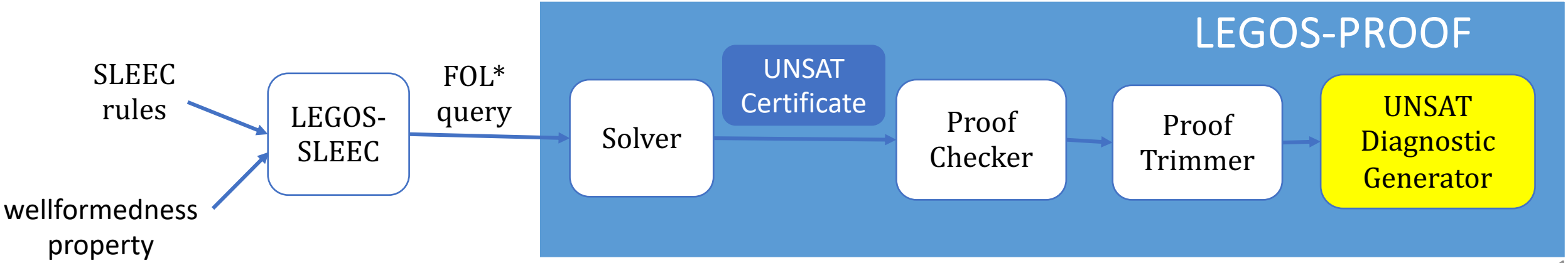
Challenges:

Legos solver produces a lot of info, in another language, with possibly buggy implementation

Our solution:

Derive and trim proof of UNSAT for FOL* to

- enable checking the UNSAT claim's correctness
- explain the unsatisfiability (diagnosis) in SLEEC



Proof-based diagnosis 1/3

- Refine the UNSAT core by projecting the derivation steps onto the core and removing atoms that are irrelevant for the derivation
- Obtain a subset of input clauses for deriving UNSAT (UNSAT core)

Example: Why is R5 redundant?

Rules

R1 **when** DressingStarted **then** DressingComplete **within** 120 **sec**
unless (roomTemperature < 19) **then** DressingComplete **within** 90 **sec**
unless (roomTemperature < 17) **then** DressingComplete **within** 60 **sec**

R2 **when** CurtainOpenRqt **then** CurtainsOpened **within** 60 **sec**

R3 **when** UserFallen **then** SupportCalled **unless** (**not** assentToSupportCalls)

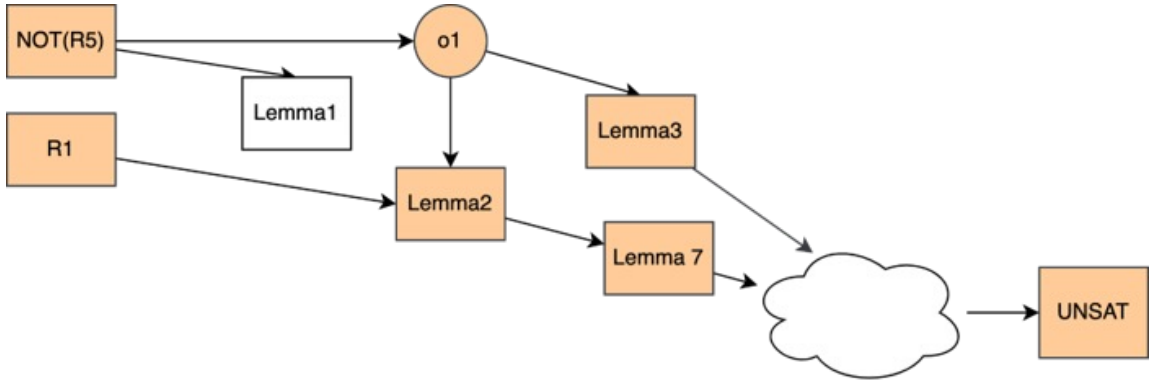
R4 **when** DressingAbandoned **then** RetryAgreed **within** 30 **sec**

R5 when DressingStarted and (roomTemperature > 20) then DressingComplete within 120 sec

Proof-based diagnosis 2/3

Identify input rules that cause redundancy

Example: Why is R5 redundant?



R5 when DressingStarted and (roomTemperature > 20) then DressingComplete within 120 sec

Diagnosis

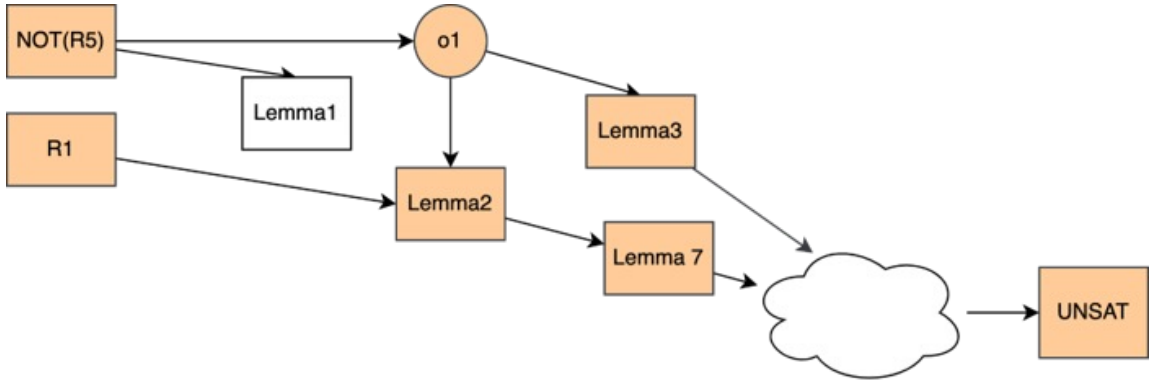
R5 is redundant because of R1

R1 **when** DressingStarted **then** DressingComplete **within** 120 **sec**
unless (roomTemperature < 19) **then** DressingComplete **within** 90 **sec**
unless (roomTemperature < 17) **then** DressingComplete **within** 60 **sec**

Proof-based diagnosis 3/3

Highlight the reason of a derivation

Example: Why is R5 redundant?



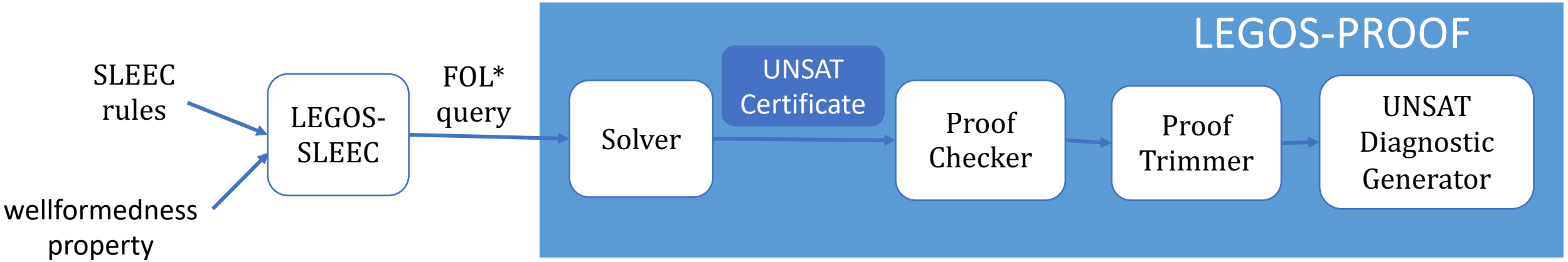
R5 when DressingStarted and (roomTemperature > 20) then DressingComplete within 120 sec

Diagnosis

R5 is redundant because of R1

R1 **when** DressingStarted **then** DressingComplete **within** 120 **sec**
unless (roomTemperature < 19) **then** DressingComplete **within** 90 **sec**
unless (roomTemperature < 17) **then** DressingComplete **within** 60 **sec**

LEGOS-PROOF Framework



- LEGOS [CAV23] incrementally searches for a satisfying solution for the FOL∗ formulas (over expanding domains of relational objects)
- Extend LEGOS with support for **proof of UNSAT for FOL***
- **Create UNSAT proof certificate** by recording UNSAT derivation steps (using FOL* derivation rules)
- Developed technique to **validate the soundness** and **trim** UNSAT FOL* **proof certificates**
- Derived **`proof-based diagnoses** to explain the cause of UNSAT in SLEEC

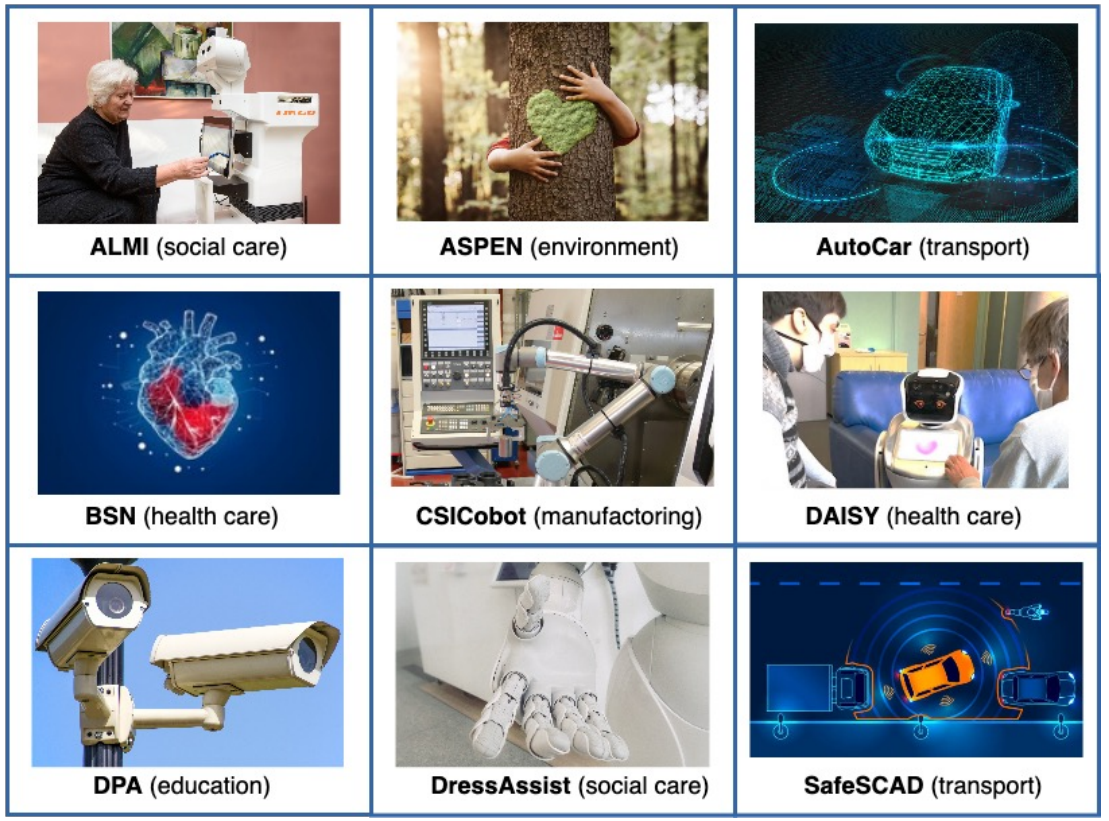
LEGOS-PROOF tool: https://github.com/NickF0211/LEGOS-Proof-Artifact/

[CAV23] N. Feng, L. Marsso, M. Sabetzadeh, and M. Chechik. "Early verification of legal compliance via bounded satisfiability checking", CAV 2023.

Evaluation

RESERVE:
repository of 9 real-world case studies

- **Domains:** transport, environment, manufacturing health and social care.
- **Different stages:** ranging from the design phase to deployed systems
- **Non-technical stakeholders:** an ethicist, a lawyer, a philosopher, and a psychologist
- **Technical stakeholders:** a safety analyst, and 3 engineers
- **Normative requirements**: 233 N-NFRs in total



RESERVE: http://www.cs.toronto.edu/~sleec/

Research questions

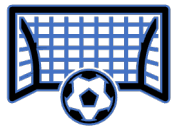
- **How efficient is LEGOS-PROOF for generating/checking the proof for redundancies?**
- **How effective is LEGOS-PROOF for trimming the proof?**
- **How the diagnosis assist users in debugging identified inconsistencies?**

How the diagnosis assist users in debugging identified inconsistencies?

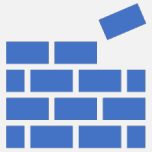
case studies	well-formedness issues										
	conflict		s-conflicts				redundancy		restrictiveness		
	nRu (Ru)	nCu (nC)	nRu (nR)	nCu (nC)	nSEu (nSE)	nSMu (nSM)	nRu (nR)	nCu (nC)	nC (nCu)	nEu (nE)	nMu (nM)
ALMI	NA		2 (39)	4 (7)	2 (3)	1 (2)	NA			NA	
ASPEN	NA		2 (15)	6 (6)	2 (3)	2 (3)	3 (15)	7 (7)		NA	
Casper	NA				NA		2 (26)	8 (8)		NA	
DAISY	NA		2 (26)	6 (6)	2 (5)	5 (5)	1 (1)	2 (2)		NA	
CSI-Cobot	NA				NA		2 (20)	4 (4)		NA	
SafeSCAD	NA		2 (28)	4 (6)	1 (5)	2 (3)	2 (2)	4 (4)	4 (4)	2 (2)	1 (1)
Tabiat	4 (28)	1 (3)	2 (28)	5 (6)	2 (4)	3 (3)	4 (28)	1 (4)	2 (3)	2 (4)	0 (0)

- LEGOS-PROOF identified small core of 2-4 rules causing the issue out of 15/19 rules
- All the highlighted clauses in the redundancy diagnosis were used for understanding and resolving them

Conclusion



Goal: Produce a diagnosis to explain the cause of UNSAT



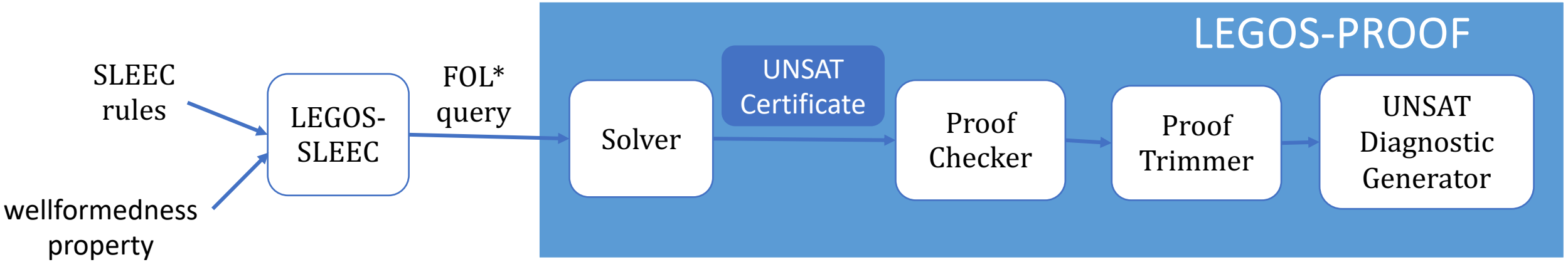
Our contributions:

- Support **proof of UNSAT for FOL***
- Developed technique to **validate and trim** UNSAT FOL* **proofs**
- Derived `**proof-based diagnoses** to explain the cause of UNSAT

Next step:

Extend the proof-based diagnosis to other SE activities (e.g., test coverage)

Outcome: An effective engagement with a formal reasoning tool for non-technical stakeholders!



Thank you! ☺ *Questions?*

LEGOS-PROOF:
https://github.com/NickF0211/LEGOS-Proof-Artifact/