Dist-AI in TLA^{+*}

Xiaosong Gu
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing, China
xxx@smail.nju.edu.cn

Jiacheng Zhao
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing, China

Wenjun Cai
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing, China
xxx@smail.nju.edu.cn

Hengfeng Wei*
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing, China
hfwei@nju.edu.cn

Yu Huang
State Key Laboratory for Novel Software Technology
Nanjing University
Nanjing, China
yuhuang@nju.edu.cn

...

...

ABSTRACT

PVLDB Reference Format:
Xiaosong Gu, Jiacheng Zhao, Wenjun Cai, Hengfeng Wei*, Yu Huang, . . . , and . . . . Dist-AI in TLA$^+$. PVLDB, 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at URL_TO_YOUR_ARTIFACTS.

1 INTRODUCTION

TLA$^+$, TLC, and TLAPS.

Automatic invariant inference.

Overview.

- **TLA$^+$traces sampling**
  - **Counter-example Guided**
  - **Coverage (e.g., minimal spanning)**
- **invariants space enumeration (exploration)**
  - **using Apalache: VARIABLES to relations (in Ivy), which are used as items in invariants**
  - **convert invariants in terms of relations back to those in terms of TLA$^+$ variables**
- **Validation (utilizing Apalache)**
  - **on finite models; for any steps**
- **Refinement**
  - **Counter-example Guided**
- **Generalization to any models (for any steps)**
  - **How to validate it? (find some SMT???)**

Table 1: Details of the TPCommit Variables

Name	Type
rmState	**(Str -> Str)**
tmState	**Str**
tmPrepared	**Set(Str)**
msgs	**Set([rm: Str, type: Str])**

Our Contributions.

-
-
-

2 OVERVIEW

2.1 Sampling TLA$^+$ Traces

2.2 Enumerating Invariants

- **directed by syntax of TLA$^+$**
- **restricting terms, operations,** . . .

2.3 Validating Inductive Invariants

- **using Apalache (modified for validating fols with quantifiers)**
- **using [?]**

3 CASE STUDY

3.1 Lock Server

3.2 Two-Phase Commit

We use Two-Phase Commit as an example. First, we need to extract the main variables of this protocol. Their types are listed as follows. They are marked on the top of the TLA+ code, so when the type check work is done, they can easily be extracted.

*Corresponding author. Hengfeng Wei is also with Software Institute at Nanjing University.

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

The next step is to calculate the possible values of these variables, and to generate relations for them. Variable rmState, tmState **and** tmPrepared **have clear value ranges, which have been listed in the** TPTypeOK **section. Some of these relations are listed below.**

tmState(x) : tmState = x**, in which** x **can be either**"init", "commited" **or** "aborted"

rmState(x,y) : rmState(x) = y**, in which** x **can be all RMs,** y **can be the four strings specified by** TPTypeOK **section.**

tmPrepared(x) : x ∈ tmPrepared**, in which** x **can be all RMs.**

Relation generated from msgs **is relatively difficult. First we need to calculate the maximum number of elements in** msgs**, which is 3.(The way to calculate this number is still unknown.) Each possible element (or record) in** msgs **contains two properties, which are** type **and** rm**. Thus we need to define seven relations as below.**

msgs(x) : x ∈ msgs

msgs.msg1.type(y) : x.type = y

msgs.msg1.rm(y) : x.rm = y

msgs.msg2.type(y) : x.type = y

msgs.msg2.rm(y) : x.rm = y

msgs.msg3.type(y) : x.type = y

msgs.msg3.rm(y) : x.rm = y

It's worth noting that all the x **occurred in these relations are unnamed variables. In practice they are given random names without repetition, such as** msgsv1**, to indicate that it is the first element of the set** msgs**. Using** msgsv1 **here doesn't mean that we are sorting elements in a set, instead it is only a way to maintain the consistency among all these three relations. We are able to depict the states of** msgs **with these three relations.**

10 relations are defined above, and 31 predicates will be generated assuming that RM is a 2-element set. DistAI will use these predicates to handle further processes.

There is another way to depict msgs**. The six relations used to describe elements within msgs can be synthesized. Thus we can replace the 8 relations with the 2 relations below. This method reduces the amount of relations, but increases the amount of predicates. In the 2-element case, we would have 5 relations and 34 predicates.**

msgs(x) : x ∈ msgs

msgs.type(x,y,z) : x.rm = y ∧ x.type = z

3.3 Paxos

4 RELATED WORK

DistAI

SWISS

Ivy

I4: inductive invariants for finite models (utilizing Averroes), and then generalize them to general models

Apalache

5 CONCLUSION

@inproceedingsProofAutomation:PhDThesis2014, title=Proof automation and type synthesis for set theory in the context of TLA+, author=Hernán Vanzetto, year=2014