Accumulation Analysis (Artifact)

Martin Kellogg
University of Washington, Seattle, WA, USA

Narges Shadab
University of California, Riverside, CA, USA

Manu Sridharan
University of California, Riverside, CA, USA

Michael D. Ernst
University of Washington, Seattle, WA, USA

Abstract

This artifact contains the data and analysis supporting the literature survey in section 4 of [4]. In our literature survey, we examined 187 papers from the literature that mention "typestate" and analyzed the typestate specifications they contained to determine whether or not they are accumulation typestate specifications.

Our purpose in doing this literature survey was to determine whether typestate FSMs were accumulation or not. However, we believe that the collection of typestate automata in `typestates.pdf` might be useful to anyone interested in the sort of typestate automata that appear in the literature. If we had had access to such a collection (gathered for a different purpose), our classification of whether these typestate automata were accumulation would have been much simpler. Anyone interested in properties of typestate automata can re-use our work.

2012 ACM Subject Classification Software and its engineering → Formal software verification

Keywords and phrases Typestate, finite-state property

Digital Object Identifier 10.4230/DARTS.8.2.22

Funding This research was supported in part by the National Science Foundation under grants CCF-2007024 and CCF-2005889, DARPA contract FA8750-20-C-0226, a gift from Oracle Labs, and a Google Research Award.

Related Article Martin Kellogg, Narges Shadab, Manu Sridharan, and Michael D. Ernst, "Accumulation Analysis", in 36th European Conference on Object-Oriented Programming (ECOOP 2022), LIPIcs, Vol. 222, pp. 10:1–10:30, 2022.
`https://doi.org/10.4230/LIPIcs.ECOOP.2022.10`

Related Conference 36th European Conference on Object-Oriented Programming (ECOOP 2022), June 6–10, 2022, Berlin, Germany

Evaluation Policy The artifact has been evaluated as described in the ECOOP 2022 Call for Artifacts and the ACM Artifact Review and Badging Policy.

1 Scope

This artifact supports the claims in section 4 about the research literature.

Table 1, in particular, is produced from this artifact. The correspondence between artifact components and each "dataset" row in Table 1 is the following:

- `papers.xlsx`: "Papers since 2000 with <20 TSAs"
- `icse99-notes.md`: "Dwyer et al. (1999) [18]"
- `ecoop11-notes.md`: "Beckman et al. (2011) [4]"

The list of conferences at the end of section 4.1 was computed using cells I1:L18 of `papers.xlsx`.

The summary statistics in the beginning of section 4.2.1 are computed using cells I30:J31 of `papers.xlsx`.

Section 4.2.1 groups typestate automata into typestate problems. File `typestates.pdf` contains the data underlying this grouping. The numbers of problems were counted using the largest subsection number in the accumulation and non-accumulation sections (e.g., there are 31 accumulation typestate problems, because the largest subsection number in section 2 of `typestates.pdf` is 2.31). Within `typestates.pdf`, the ordering of the problems is arbitrary.

Sections 4.2.1.1 and 4.2.1.2 give the most common typestate problems. These were chosen by the "count" appearing in the text describing each problem in `typestates.pdf`, which one author computed by hand from our notes. A second author double-checked all these computations.

We broke ties for the three examples in the paper by choosing the most interesting example. The problems chosen correspond to the following subsections in `typestates.pdf`, in the order in which they are presented in section 4.2.1:

- resource leaks (Figure 3): 2.1
- special initialization method (Figure 4): 2.5
- object construction (Figure 5): 2.13
- don't read/write a stream after it is closed (Figure 6): 3.31
- don't update a collection during iteration (Figure 7): 3.12
- classic file example (Figure 1): 3.1

The artifact also supports the claim about the causes of false positives in prior work [3]: that even with perfect aliasing information, the precision of an accumulation-based resource leak checker would only improve from 26% to 34%. Our reasoning for this assertion is based on the analysis of the false positives from that checker in `rlcfps.xlsx`.

2 Content

The artifact package includes:

- a spreadsheet listing each of the 187 papers that we examined closely, the number of typestate specifications in that paper, and the number of those specifications that are accumulation typestate specifications (`papers.xlsx`)
- our notes about the typestates in each paper so examined, ordered alphabetically by title (`notes.md`)
- a list of each typestate "problem", its corresponding finite-state machine, and citations to the papers that mention it, sorted by our judgment on whether they are accumulation (section 2) or are not accumulation (section 3) (`typestates.pdf`)
- our notes about Dwyer et al. (1999) [2], to support our claims in section 4.2.2.1 (`icse99-notes.md`)
- our notes about Beckman et al. (2011) [1], to support our claims in section 4.2.2.2 (`ecoop11-notes.md`)
- our analysis of the false positives issued by the Resource Leak Checker (RLC) in [3] and whether or not they could have been verified if the RLC had access to perfect aliasing information (`rlcfps.xlsx`)

3 Getting the artifact

The artifact endorsed by the Artifact Evaluation Committee is available free of charge on the Dagstuhl Research Online Publication Server (DROPS). The artifact is also available from `https://doi.org/10.5281/zenodo.5771196`.

4 Tested platforms

This artifact should be usable on any system with a PDF reader and a spreadsheet program capable of understanding the `.xlsx` format (this includes web-based platforms; we have tested Google Sheets).

5 License

6 MD5 sum of the artifact

429b8a9421df64399b328dfa1e075a9d

7 Size of the artifact

1.1M

References

1 Nels E Beckman, Duri Kim, and Jonathan Aldrich. An empirical study of object protocols in the wild. In *European Conference on Object-Oriented Programming*, pages 2–26. Springer, 2011.

2 Matthew B Dwyer, George S Avrunin, and James C Corbett. Patterns in property specifications for finite-state verification. In *International Conference on Software Engineering*, pages 411–420, 1999.

3 Martin Kellogg, Narges Shadab, Manu Sridharan, and Michael D. Ernst. Lightweight and modular resource leak verification. In *ESEC/FSE 2021: The ACM 29th joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*, 2021.

4 Martin Kellogg, Narges Shadab, Manu Sridharan, and Michael D. Ernst. Accumulation analysis. In *European Conference on Object-Oriented Programming (ECOOP)*, 2022.