Static Race Detection and Mutex Safety and Liveness for Go Programs (Artifact)

Julia Gabet
Imperial College London, United Kingdom
j.gabet18@imperial.ac.uk

Nobuko Yoshida
Imperial College London, United Kingdom
n.yoshida@imperial.ac.uk

Abstract

This artifact contains a version of the Godel tool that checks MiGo$^+$ types – an extension of MiGo from [1] including GoL. Given the extracted MiGo$^+$ types, the tool can analyse them using the mCRL2 model checker to check several properties including liveness, safety and data race freedom as defined in our paper. The artifact also includes examples, shipped with both the source of the Godel tool and the benchmark repository. The latter also contains the Go source for the benchmark examples. We provide compiled binaries of the artifact in a Docker image, with instructions on how to use them. Finally, for convenience, the Docker image also contains a binary version of the migoinfer+ tool, developed as a fork from the original migoinfer by Nicholas Ng in [1]. This new version adds the ability to extract shared memory pointers as well as Mutex and RWMutex locks.

2012 ACM Subject Classification Software and its engineering → Concurrent programming languages; Software and its engineering → Model checking; Theory of computation → Process calculi

Keywords and phrases Go language, behavioural types, race detection, happens-before relation, safety, liveness

Digital Object Identifier 10.4230/DARTS.6.2.12

Funding The work is partially supported by VeTSS, EPSRC EP/K011715/1, EP/K034413/1, EP/L00058X/1, EP/N027833/1, EP/N028201/1, EP/T006544/1 and EP/T014709/1.

Acknowledgements The authors wish to thank Nicholas Ng for his initial collaboration on the project, David Castro-Perez and Fangyi Zhou for their comments and testing the artifact, and the anonymous reviewers for their comments and suggestions.

Related Article Julia Gabet and Nobuko Yoshida, "Static Race Detection and Mutex Safety and Liveness for Go Programs", in 34th European Conference on Object-Oriented Programming (ECOOP 2020), LIPIcs, Vol. 166, pp. 4:1–4:30, 2020. https://doi.org/10.4230/LIPIcs.ECOOP.2020.4

Related Conference 34th European Conference on Object-Oriented Programming (ECOOP 2020), November 15–17, 2020, Berlin, Germany (Virtual Conference)

1 Scope

The artifact presents a proof-of-concept model checking tool for Go programs to analyse their extracted behavioural types as defined in the companion paper, and check them for several properties, including those checked by the original tool presented in [1] and those defined in the companion paper.

The tool is a frontend to infer mCRL2 formulas and a model from a given MiGo$^+$ input, and feed them to the mCRL2 model checker in order to get the results for various properties. The tool is written in Haskell and serves as an example for how the properties in the companion paper can be checked in practice, though it is a proof-of-concept and would need more engineering to properly scale with heavy use of channels and `select` constructs.

The migoinfer+ addition to the Docker image is written in Go, like the original tool, and uses the `go/ssa` package as a first abstraction layer to extract the primitives we wish to analyse as MiGo$^+$ types. As this tool runs the program under analysis with instrumentation in order to perform such extraction, we require Go to be installed on the system.

2 Content

The artifact includes:

- The source code of the Godel tool, extended with support for Mutex and RWMutex locks, and shared variables, as well as checking for safety and liveness for said locks and data races for shared variables;
- The source code of our benchmark examples, and the script to run them;
- The source code of both migoinfer+ and the underlying MiGo$^+$ types library;
- A Docker image containing binaries of the updated Godel tool, that is able to run on the provided examples, and migoinfer+ to be able to extract MiGo$^+$ types from a given Go source code file;
- A step-by-step instruction on how to load the Docker image and run the different examples, along with warnings for the longer ones if running the automated script (the two longer ones can take *several hours* to complete on weaker configurations, all others finish within seconds to minutes).

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). Up to date versions of each of the included items, including compilation requirements and instructions, can be found at the following URLs:

- Godel2: `https://github.com/JujuYuki/godel2`
- The benchmark files and script used to test it, along with short explanations for each example and the expected results: `https://github.com/JujuYuki/godel2-benchmark`
- migoinfer+: `https://github.com/JujuYuki/gospal`,
- the underlying types library: `https://github.com/JujuYuki/migo`
- the Docker image containing pre-compiled versions of Godel2 and its dependencies, as well as a pre-compiled version of migoinfer+: `https://hub.docker.com/r/jgabet/godel2`

4 Tested platforms

This artifact is known to work on any platform able to run Docker with at least 2GB of free disk space and 2GB of free RAM, and should run with minimal RAM overhead on most smaller examples.

5 License

- Godel2 is released under the BSD 3-Clause License (`https://opensource.org/licenses/BSD-3-Clause`)
- migoinfer+ and the migo library are released under the Apache License version 2.0 (`http://www.apache.org/licenses/LICENSE-2.0.html`)

6 MD5 sum of the artifact

0965ca24dfe6c996566c1c25c65d8a07

7 Size of the artifact

The provided package is 470.6 MiB compressed, and 1.7 GiB after decompression.

References

1 Julien Lange, Nicholas Ng, Bernardo Toninho, and Nobuko Yoshida. A static verification framework for message passing in go using behavioural types. In *Proceedings of the 40th International Conference on Software Engineering*, ICSE '18, pages 1137–1148, New York, NY, USA, 2018. ACM. doi:10.1145/3180155.3180157.