SPARTA

SPARTA is a library of software components specially designed for building high-performance static analyzers based on the theory of Abstract Interpretation.

Abstract Interpretation

Abstract Interpretation is a theory of semantic approximation that provides a foundational framework for the design of static program analyzers. Static analyzers built following the methodology of Abstract Interpretation are mathematically sound, i.e., the semantic information they compute is guaranteed to hold in all possible execution contexts considered. Moreover, these analyzers are able to infer complex properties of programs, the expressiveness of which can be finely tuned in order to control the analysis time. Static analyzers based on Abstract Interpretation are routinely used to perform the formal verification of flight software in the aerospace industry, for example.

README.md

Building an industrial-grade static analysis tool based on Abstract Interpretation from scratch is a daunting task that requires the attention of experts in the field. The purpose of SPARTA is to drastically simplify the engineering of Abstract Interpretation by providing a set of software components that have a simple API, are highly performant and can be easily assembled to build a production-quality static analyzer. By encapsulating the complex implementation details of Abstract Interpretation, SPARTA lets the tool developer focus on the three fundamental axes in the design of an analysis:

- **Semantics:** the program properties to analyze (range of numerical variables, aliasing relation, etc.)
- **Partitioning:** the granularity of the properties analyzed (intraprocedural, interprocedural, context-sensitive, etc.)
- **Abstraction:** the representation of the program properties (sign, interval, alias graph, etc.)

SPARTA is an acronym that stands for **S**emantics, **PART**itioning and **A**bstraction.

Using SPARTA

A detailed documentation for SPARTA can be found in the code of the library itself. It includes code samples, typical use cases and references to the literature. Additionally, the unit tests are a good illustration of how to use the API. The unit test for the fixpoint iterator, in particular, implements a complete analysis (live variables) for a simple language.

SPARTA is the analytic engine that powers most optimization passes of the ReDex Android bytecode optimizer. The ReDex codebase contains multiple examples of analyses built with SPARTA that run at industrial scale. The interprocedural constant propagation illustrates how to assemble the building blocks provided by SPARTA in order to implement a sophisticated yet scalable analysis.

Dependencies

SPARTA requires Boost 1.71 or later.

macOS

You will need Xcode with the command line tools installed. To get the command line tools, please use:

```
xcode-select --install
```

Boost can be obtained via homebrew:

```
brew install boost
```

Ubuntu (64-bit)

On Ubuntu 16.04 or later, Boost can be installed through the package manager:

```
sudo apt-get install libboost-all-dev
```

For earlier versions of Ubuntu, we provide a script to install Boost:

```
sudo ./get_boost.sh
```

Setup

SPARTA is a header-only C++ library. You can just copy the contents of the `include` directory to any location in the include path of your compiler.

We also provide a quick setup using cmake that builds all the tests:

```
# Assume you are in the SPARTA directory
mkdir build-cmake
cd build-cmake
# .. is the root source directory of SPARTA
cmake ..
cmake --build .
```

To run the unit tests, please type:

```
./run_all_tests.sh
```

To copy the header files into `/usr/local/include/sparta` and set up a cmake library for SPARTA, you can use the following command:

```
sudo make install
```

Issues

Issues on GitHub are assigned priorities which reflect their urgency and how soon they are likely to be addressed.

- P0: Unbreak now! A serious issue which should have someone working on it right now.
- P1: High Priority. An important issue that someone should be actively working on.
- P2: Mid Priority. An important issue which is in the queue to be processed soon.
- P3: Low Priority. An important issue which may get dealt with at a later date.
- P4: Wishlist. An issue with merit but low priority which is up for grabs but likely to be pruned if not addressed after a reasonable period.

License

SPARTA is MIT-licensed, see the LICENSE file in the root directory of this source tree.

Releases

No releases published

Packages

No packages published

Contributors 15

+ 4 contributors

Languages

C++ 98.6% **CMake** 1.2% **Shell** 0.2%