D4: Fast Concurrency Debugging with Parallel Differential Analysis

A. Artifact description

A.1 Abstract

This artifact contains the source code of D4, the executables of ECHO along with 14 benchmarks to demonstrate the performance improvement by the new techniques described in paper *D4: Fast Concurrency Debugging with Parallel Differential Analysis*. We provide the required benchmarks and settings to reproduce the performance results presented in the paper, e.g., the same content presented in Table 4, 5, 6 and 7. Due to the large amount of time to complete the full evaluation (i.e., around 10 days), we also provide a short version of evaluation to show the performance advantage of D4.

A.2 Description

A.2.1 Check-list (artifact meta information)

- **Algorithm: Parallel incremental pointer analysis, parallel incremental happens-before analysis, a client-server architecture design**
- **Program: WALA, Akka**
- **Compilation: Java 1.7**
- **Executable: Java runnable jars**
- **Benchmark: Dacapo-9.12**
- **Run-time environment: Eclipse Mars installed on UNIX OS**
- **Hardware: Memory capacity of no less than 15GB**
- **Output: All results are shown in the console or terminal with the same metrics in our PLDI'18 paper**
- **Experiment workflow: git clone; build by Eclipse; generate a runnable jar for the remote server; run the evaluation; evaluate performance results.**
- **Publicly available?: Yes**

A.2.2 How delivered

The artifact is publicly available on Github at:
https://github.com/parasol-aser/D4.

A.2.3 Hardware dependencies

The artifact evaluation requires two machine connected by stable network connection: one machine acts as a client and the other acts as a server. To perform the full evaluation, we require a memory with more than 15 GB. To evaluate the performance of the parallel algorithms, we recommend to use a server that can support at least 48 threads, which is performed in the paper.

A.2.4 Software dependencies

The artifact is tested on a client machine with Eclipse Mars running on Mac OS X, and a server machine with CentOS Linux 7. We recommend to setup the maximum size of the memory allocation pool with `-Xmx15g` for each benchmark to guarantee enough run-time heap space while reproducing the full data in the paper.

A.2.5 Benchmarks

The evaluated benchmark is Dacapo-9.12 including 14 large, real-world Java programs, in which 13 of them are multi-threaded. We have built and included these benchmarks in the artifact.

A.3 Installation

After cloning the repository, import the projects into Eclipse, compile and build them with Java 1.7. To generate the executable for the server evaluation, update the `master.conf` (in `/edu.tamu.cse.aser.d4/src/master.conf`) and `worker.conf` (in `/edu.tamu.cse.aser.d4remote/src/worker.conf`) with the local machine and remote server hostnames as indicated in the `.conf` files. Then, export `edu.tamu.cse.aser.d4remote` as a runnable jar with the main method `/edu.tamu.cse.aser.d4remote/src/edu/tamu/aser/tide/dist/remote/remote/BackendStart.java` and the option "Copy required libraries into a sub-folder next to the generated JAR". Transfer the generated jar, sub-folder libraries and `/edu.tamu.cse.aser.d4remote/data` to your remote server.

A.4 Experiment workflow

To reproduce the full results of D4-1 and D4-48 in the paper, run the main method in `ReproduceBenchmarks.java` with program argument `all` in Run Configration. To reproduce individual benchmark data, please run the main method with the benchmark name as the program argument.

To reproduce the full data of ECHO (including the Reset-Recompute algorithm, Reachability-based algorithm and its data race detetion), run the runnable jar `echo.jar` (in `/ECHO_jars/`) with argument `all`.

Reproducing the full results requires around 10 days, which is too long for reviewer evaluation. So, we provide a shorter version of evaluation with a reasonable running time

to compare the performance. To evaluate D4-1 and D4-48, run `ReproduceBenchmarks.java` with program argument `all_short`. For individual benchmark, please run the main with the benchmark name with `_short`, for example, `avrora_short`. To evaluate ECHO, run `echo.jar` with argument `all_short`.

A.5 Evaluation and expected result

All the evaluation results will be shown in the Eclipse console or the server terminal. We don't expect absolute values to match with the paper, due to the hardware difference and the stability of the wireless connection. But we expect a similar trend of D4 performance metrics as presented in the paper.