The University of Waikato
Department of Computer Science
Dr Robi Malik

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

COMPX552-19A

Model Checking

Assignment 3

Please write a Java program that checks
whether a system of finite-state machines is nonconflicting.

On the course homepage in Moodle, you will find an archive that contains the *Waters Automata Toolkit* that you are to extend and use. When extracted, this archive produces a directory named `waters552` with the following subdirectories.

`docs/` Javadoc documentation of the Waters Toolkit.

`examples/` A set finite-state machine models in Waters' file format and a `README` file with expected conflict check results, to test your implementation.

`jar/` Archives in JAR format containing the classes needed to run the Waters Toolkit.

`lib/` Waters library (needed for counterexample verification tool).

`src/` Some Java source files with sample code to be extended by you.

`tools/` Build tools for ANT and Eclipse.

The Waters Toolkit includes the definition of an abstract class called `ModelChecker` in package `net.sourceforge.waters.analysis.comp552`, which you have to extend from to provide a class for conflict checking. The `src/` subdirectory includes a sample source file called `ConflictChecker.java` as a template for this. Please use this template and implement your algorithm in a class with the same name and package.

The Waters Toolkit provides access to finite-state machine models, which are made available as Java objects. The key class representing a model of several finite-state machines is `net.sourceforge.waters.model.des.ProductDESProxy`. The `src/` subdirectory includes a sample main class `ConflictMain.java` that reads files with names taken from the command line and passes them to the `ConflictChecker` class.

Java 8 is required to compile and use the Waters Toolkit. To compile the sources, the provided JAR archives must be included in the classpath. To run the sample main program using the counterexample verification tool, the `lib/` directory must be included in the `java.library.path`. This can be achieved using the following commands from the directory named `waters552`:

```
mkdir classes
javac -classpath "jar/*" -d classes -sourcepath src \
  src/net/sourceforge/waters/analysis/comp552/*.java
java -classpath "classes:jar/*" -Djava.library.path=lib -enableassertions \
  -Xmx4096m net.sourceforge.waters.analysis.comp552.ConflictMain \
  examples/simple/*.wmod
```

(The backslash "\" indicates a long command wrapped over two lines. For Windows, please use semicolon ";" instead of colon ":".)

These commands compile and run the sample main class on all the models contained in the `simple` subdirectory of the provided example collection. For your convenience, an Eclipse project and an ANT build script to automate these build commands are included in the `tools/` directory.

The Task

Please complete all methods in the `ConflictChecker` class, in particular implementing the methods to determine whether the input model is nonconflicting, and to compute a counterexample if they are not. Use efficient data structures and algorithms so you can solve as many of the example problems as possible.

Remember to use good programming practises and write clear code.

In addition, please write a brief report of 1–2 pages that explains your program, and any steps you took to improve the efficiency of your implementation. Also please include some performance statistics for all the examples that you have solved.

Marking

This assignment is worth 30 marks in total.

Your program will be tested on 30 finite-state machine models of varying size and complexity. Ten of these models are nonconflicting, and five marks can be obtained by correctly classifying them as nonconflicting. 20 models are conflicting, and ten marks can be obtained by correctly classifying them as conflicting. (Assuming that your program genuinely tries to check the nonconflicting property—"super-clever" programs that always return true or false will not be tested.) Further ten marks are obtained by computing correct counterexamples for the conflicting models. All tests together will be given a total of 20 min to complete with 4 GiB of RAM allocated to the Java Virtual Machine.

Finally, five marks are allocated to your report and code readability.

Submission

Please hand in your solutions into the box marked COMPX552 in front of room G 1.15. Your complete submission should include

- a printout of your Java source files;
- your written report.

In addition, please submit your Java source files electronically in an archive through the assignment submission system in Moodle. Acceptable file formats are `.tar.gz` and `.zip`.

Submission deadline: Tuesday, 4 June 2019, 10:00 A.M.