

COMP 424/524-06A

Topics in Software Engineering

Assignment 3

Exercise 1 (15 marks)

Write a Java program that checks whether a system of finite-state automata is controllable.

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

`docs`. Javadoc documentation of the Waters Toolkit.

`examples`. A set automata models in VALID's file format to test your implementation.

`jar`. Two archives in JAR format containing the classes needed to run the Waters Toolkit.

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

The Waters Toolkit includes the definition of an abstract class called `ModelChecker` in package `net.sourceforge.waters.analysis`, which you have to extend from to provide a class for controllability checking. The `src` subdirectory includes a sample source file called `ControllabilityChecker.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 supports the reading and writing of finite-state machine models in different file formats, including VALID's. The models are made available as Java objects using the interfaces in a package called `net.sourceforge.waters.model.des`. The `src` subdirectory includes a sample main class `ControllabilityMain.java` that reads files with names taken from the command line and passes them to the `ControllabilityChecker` class.

From the directory named `waters`, you can use the following commands to compile and run the sample main class on all VALID models provided as examples.

```
mkdir classes
javac -classpath jar/waters424.jar:jar/waterslib.jar -d classes \
    -sourcepath src src/net/sourceforge/waters/analysis/*.java
java -classpath jar/waters424.jar:jar/waterslib.jar:classes \
    net.sourceforge.waters.analysis.ControllabilityMain \
    examples/*/*.vprj
```

Note. Java 1.5 is required to compile and use the Waters Toolkit.

Please complete all methods in the `ControllabilityChecker` class, in particular implementing the methods to determine whether the input automata are controllable, 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 practices and write clear code. Provide Javadoc-style documentation at least for all nontrivial methods that you implement, and for all auxiliary classes that you add. Finally, write a brief report of 1–2 pages that explains the methods and data structures you used, and any steps you took to improve the efficiency of your implementation. Also include some performance statistics for all the examples that you have solved.

Note. While the Waters Toolkit helps you to read automata models and access them through Java interfaces, it does *not* provide any further support for synchronous product computation. You are expected to design and implement the necessary data structures yourself.

Exercise 2 (5 marks)

Draw an optimal Ordered Binary Decision Diagram (OBDD) representing the state transition relation of a modulo-16 counter.

The state transition relation of a modulo-16 counter can be written as

$$x' = (x + 1) \bmod 16 ,$$

where x and x' are represented using four binary digits each, i.e.,

$$\begin{aligned} x &= 8x_3 + 4x_2 + 2x_1 + x_0 ; \\ x' &= 8x'_3 + 4x'_2 + 2x'_1 + x'_0 . \end{aligned}$$

Translate the transition relation into a Boolean formula, and draw an Ordered Binary Decision Diagram for it. Experiment to find an optimal ordering of the variables $x_3, \dots, x_0, x'_3, \dots, x'_0$. Hand in a diagram with the smallest number of nodes that you can produce.

Submission

Please hand in your solutions into the box marked COMP424A in front of room G 1.15. Your complete submission should include

- a) a printout of your Java source files for exercise 1;
- b) your written report as specified in exercise 1;
- c) a drawing of your OBDD for exercise 2.

In addition, please submit your Java source files for exercise 1 electronically through the assignment submission system at

<http://byerley.cs.waikato.ac.nz/~tonym/submit/> .

Please note that this system works only from our computing laboratories on campus.

Due date: Monday, 10th April 2006, 9:00 A.M.