Temporal Logic Assignment: Model Checking

Logic in Computer Science Course (ROKF),
Reykjavik University

1 Practical Information

- Due on Sunday 3 April at 23:59.
- Submission should be through Canvas.
- You must submit a PDF file with your report and a ZIP file with everything required to compile your Python program.
- This assignment is worth up to 10% of your grade ($\frac{1}{5}$ of your homework grade), but you can attempt a bonus 10% that can replace another of your assignments.

2 A Puzzle

A well-known puzzle is the following.

> A ferryman wants to transport three things across a river: a wolf, a goat, and a cabbage. He has a small ferry, but it can only fit himself and one more thing (the cabbage is obviously large). If he leaves the wolf alone with the goat on one side of the river, then the wolf will eat the goat. If he leaves the goat alone with the cabbage, then the goat will eat the cabbage. Both cases must be avoided. Can you find a schedule for the ferryman's trips, which starts with the ferryman, the ferry, the goat, wolf, and cabbage on one side, and ends with all these on the other side (without anything getting eaten)?

This is a well-known puzzle, and it appears in your book, where it is solved by LTL model-checking, using a model-checker called NuSMV. You should try to solve the puzzle, if you have not done that already, because it is interesting. The project is *not* to solve this, or a general version of puzzle, but it is somehow related.

3 Entrepreneurship and Ferries

After successfully solving the above puzzle and delivering the three items, the ferryman's reputation rose and he was able to expand his business of transferring items from one side to the other. He founded a company and upgraded his boat and storage facilities. With time, his granddaughter, the ferrywoman, has inherited the transportation company. Now, the company uses a variety of modern boats and hires several employees to transfer the goods and to take care of any conflicts between these goods. The transportation schedule is provided by a sophisticated machine learning algorithm. However, sometimes there are often unexpected issues with the schedule, which causes the company to lose revenue (and cabbages to perish due to unsupervised encounters with goats).

You are the newest hire of the company. Your job is to verify that the schedule that is provided every day by the algorithm satisfies certain necessary constraints, so that no unexpected problems occur. For this, you must

1. express the given properties in LTL;
2. develop a model-checker for LTL in Python;
3. use your model checker to see whether the given schedules satisfy the required properties; and
4. write a report with your findings.

4 The Format of the Schedule

To keep things from becoming too complicated, each schedule is a regular path — it has the form of a lasso, meaning that it has an initial part, I, and a loop L that is repeated for ever after that: $I\ L\ L\ L\ L \cdots$

Each schedule is given as a text file. The first line of the file has two integers. The first integer is the total length of the schedule ($|I| + |L|$), and the second one is the length of the loop (L). Each line after that gives the labeling of the states of the schedule in order. The labeling describes the current situation at each side of the river and what is being transported on the boat, if anything. For example:

```
10 4
sheep_right goat_left employee_left spinach_left
sheep_right goat_trans employee_trans spinach_trans popeye_left
sheep_right goat_right employee_right spinach_right popeye_left
sheep_trans goat_right employee_trans popeye_trans
wolf_left sheep_right goat_right employee_right popeye_right
wolf_left sheep_right employee_trans popeye_right
wolf_left wood_left employee_left
wolf_left wood_trans employee_trans
wolf_left wood_right employee_right
wolf_left wood_right employee_trans
```

The flow of goods is always from the left to the right. New items may appear on the left side of the river, as they are received for transportations (Popeye during the second state, wolf during the fifth state, wood occasionally), and some goods may disappear from the right side of the river, as they are delivered. The kinds of items that are transported is not fixed. You may consider `item_side` to be a propositional variable.

5 The Tools that You Can Use

A previous employee had started working on the problem and he left behind a parser for LTL that you can use, as well as a class Path that is initialized with a filename and generates a path. Rumors have it that he left the company to work for a rival. Others say that he was eaten by wolves during an unfortunate transportation job. The parser and the Path class is given to you as a ZIP file that contains certain Python files.

The parser reads a string and returns an Abstract Syntax Tree of the formula in the string. The Path class gives you path objects that can be traversed or accessed as a list of states. You can use both from the file interface.py, which has further instructions on how to use the parser, the Ast class for the formulas, and Path.

6 Desired Properties

One of your tasks is to rewrite the following schedule properties using LTL:

1. There are certain conflicts between items. Some combinations of items must never appear unsupervised on the same side. These combinations are:
   (a) wolf with goat
   (b) Popeye with spinach
   (c) Popeye with wine with computer (alcohol leads to bad decisions)

2. A wolf must never stay on the same side with a goat for four or more consecutive states before the employee visits that state again (and the wolf is there). Otherwise the wolf will get hungry and may eat the employee (again).

3. An employee can spend at most three consecutive states each time at a certain side. Otherwise, she is slacking off.

4. Popeye must not be delivered more than a round before spinach. Otherwise he may starve.

5. Goat transportation and sheep transportation must alternate. That is, after a goat is transfered, eventually a sheep must be transfered, and after that another goat, and then another sheep, and so on. We cannot have sheep transported twice with no goat transported in-between.

6. To ensure that the schedule is implementable from a Physics point of view: the employee can only switch sides by using the boat. That is, after the employee is on the left, before she appears on the right she must be on the boat, and so on.

Note: work on each formula separately! These properties are not supposed to be checked in combination. We are interested to see which properties hold in which paths. So, for example, in a certain path, property 1(a) may not hold, but property 5 may hold, and that is perfectly fine (except if you are a goat, of course).

7 The Model-Checker

You must implement in Python a model checker for LTL on paths. You may need to pre-process your formulas before you check them against the paths (or not). As it is mentioned above, parts of the full program are provided to you (methods to parse the formulas and the paths), and you have to fill in the model checking part of the program. You will submit the program, and you will explain in the report how you implemented it: What methods did you use? How was it different from general model checking for LTL? What syntax does your model checker allow and how did you handle the various operators?

This is not general LTL model checking! You will not have to use the full method we did in class. LTL model checking is simpler when it is done only for regular paths (i.e. lassos, or paths that end with one loop), as the schedules described above. As it is explained below, implementing general LTL model checking (on arbitrary finite transition systems) or parts of it may give you extra points.

Use `modelcheck` as the name of the process that performs the model checking, so that I can import it.

8 The Schedules

Some schedules are given to you as text files. You will need to read them and pass them to your model checker in an appropriate form. They are provided in a separate folder called `paths`. Your solution will also be tested against additional schedules (and against additional formulas). You are also provided with a generator of random paths, to help you evaluate your solution before submitting.

9 The Report

- There is no page limit for your report. Whether it is long or short will not affect your grade, but its contents and clarity will.
- You must describe appropriately how you designed your model checker.
- You must write the required formulas in LTL (see above) and present them in the report.
- You must write the results of checking these formulas against the given schedules.
- Your report must be legible! Either type it (ideally), or submit a clear scan that a human being (me) can read and understand.

10 Extra Credit

If you can implement a model checker for LTL on general transition systems, I will be impressed. Even implementing (well!) parts of the general process (and you must identify what these are) may give you some extra points — up to an additional 10% of your grade, to replace another assignment where you did worse.

11 On Submitting

You can talk to each other about the assignment, but your solution and report must be your own. We are here to help, if you encounter any difficulties, or if anything is unclear. Make sure to evaluate your solution with the given tools.