Formal Verification 2023-24: Practical Exercise 2

SAT & SMT with the Z3 solver

1 Getting Started

The Z3 Wiki is the main place to start for information on Z3. For this exercise, you provide input to Z3 in the SMT-LIB format. A Z3 Tutorial guides you through the SMT-LIB commands that Z3 accepts, and allows you to try running them in your browser. To run you own examples in your browser, go to the Z3 Playground.

For Coursework 2, you will be asked to use a Python interface to Z3. The Z3 Wiki links to instructions on how to install Z3 and this interface. Depending on how you install this interface, Z3 also becomes available to run from the command line. For this exercise, you just need to run Z3 from the command line, if you are not running it in your browser.

On DICE Linux machines, you can use the following instructions to install Z3 and its Python interface in your DICE account file space.

1. Open a shell window and cd to your home directory.
2. Create a new Python virtual environment in directory `z3pyenv` by running the command

   ```
   python3 -m venv z3pyenv
   ```

   This directory will hold the Z3 binary and Python interface code.
3. To activate the virtual environment, enter

   ```
   source z3pyenv/bin/activate
   ```

   Your command line prompt will now have a `(z3pyenv)` prefix to indicate the environment is activated.
4. To install Z3 into this environment, enter

   ```
   pip install z3-solver
   ```
5. m The Z3 solver will be available as a command. For example,
   - `z3 -h` shows you command line options,
   - `z3 x.smt2` runs Z3 on file `x.smt2`,
   - `z3 -in` starts Z3 in interactive mode. When you then want to exists interactive mode and go back to the shell, enter the command `(exit)` or just type a *control-d*.
6. To deactivate the Z3 installation in a shell, enter the command `deactivate`.
7. You only have to install Z3 once. Subsequently, to activate Z3 again in any shell, just use again

   ```
   source z3pyenv/bin/activate
   ```

2 A constraint satisfaction puzzle

Consider 3 persons A, B and C who need to be seated in a row but

1. A does not want to sit next to C,
2. A does not want to sit in the left chair,
3. B does not want to sit to the right of C.

2.1 Solving the puzzle using SAT

Create a file `puzzle1.smt2` in which you set up the puzzle and its constraints as a propositional satisfiability problem. See the start of the *Propositional Logic* section of the Z3 Tutorial for the needed SMT-LIB statements.

Be sure to add a `(check-sat)` statement at the end, after setting up the puzzle. When you run Z3 on the file, you should find that it returns `unsat` to indicate that the problem is unsatisfiable. Remove constraint 3 and now observe Z3 returns `sat` to indicate the problem is satisfiable. Add a `(get-model)` statement after the `(check-sat)` statement to get Z3 to report the satisfying assignment it finds.

There are various ways you could encode the puzzle. For example, you may introduce 9 boolean constants, one for each of the three persons sitting in each of the three seats. Other encodings using fewer constants are possible.

2.2 Solving the puzzle using SMT

Create a file `puzzle2.smt2` in which you set up the puzzle and its constraints as an integer arithmetic satisfiability problem. See the *Theories > Arithmetic* section of the Z3 Tutorial for suitable statements.

For example, you might introduce 3 integer constants, each indicating the seat number of one of the persons.

Paul Jackson, 27th October 2023