CDS 270 (Fall 09) - Lecture 1 (Oct 2, 09)

Contact information:

- Andy Lamperski, andyl@cds.caltech.edu, Steele 19
- Ufuk Topcu, utopcu@cds.caltech.edu, Steele 126

Syllabus Tentative list of topics to be covered in CDS 270. Remember that this course is an "experimental" course that tries to merge ideas on formal verification from two ends of the spectrum (CS and controls). This interaction is subject to current research and there is neither a course that covers this material nor a well-established unification of methods and tools. The goal of this course is to lay out the advances in both directions as well as attempts and deficiencies toward such a unification. Therefore, the list of topics is subject to change as we progress in the course.

1. Preliminaries for the "controls-type" verification ($\sim$ 2 weeks)
   - Convex optimization: Linear programming, positive semi-definite matrices, linear matrix inequalities, semidefinite programming (SDP), tools for solving semidefinite programs.
   - Sum-of-squares programming: Positive semidefinite polynomials $\rightarrow$ sum-of-squares (SOS) polynomials, SOS $\leftrightarrow$ SDP, SOS programming, tools.
   - S-procedure type ideas: Sufficiency of S-procedure for set containments, advanced versions, integration with SOS programming, applications.
2. System analysis questions ($\sim$ 2 weeks)
   - Global & local stability (Lyapunov functions)
   - Safety (Barrier certificates)
   - Input-to-state, input-to-output properties, reachability (Storage functions)
   - Application of a "standard" procedure to the problem above and demonstration on simple and not-so-simple examples
   - Supporting ideas for improving the efficiency of algorithms
3. "Computer Science" verification ($\sim$ 3 weeks (a small, but useful, subset))
   - Transition systems, Promela modeling
   - Finite automata, Buchi automata, regular languages
   - Model checking regular properties
   - Linear temporal logic (LTL), LTL $\rightarrow$ Buchi automata

- Examples (using Spin)

4. Abstractions and applications of the finite state verification tools to abstractions of continuous systems.

5. (As time allows) Hybrid system verification: simulation, hybrid Lyapunov techniques, further reachability concepts/tools+ Hytec/Phaver.

At least the first half of the course will heavily rely on solving certain convex optimization problems and we will use standard packages to attack most of these problems. The following two packages (YALMIP and SeDuMi) will be very useful to this end. The following a few pages explain how to get these to packages running on your computers. Please feel free to bring your laptops to lectures as we will try to have real time demonstrations and you may find useful to try out these examples as we demonstrate.

Installing SeDuMi

Warning: Pay attention to download the right version depending on your system being 32-bit or 64-bit.

- Download SeDuMi
  - Go to `http://sedumi.mcmaster.ca/`
  - Check the left of the page and find "Register" in the "Login Form." Click on "Register." Create an account. It requires `Name, Username, E-mail,` and `Password`. Complete those fields and click "Send Registration."
  - When you see "Registration Complete!" go to "Login Form" at the left, put the account info you just created, and click on "Login."
  - Click on "Downloads" in the "Main Mune" on the top left panel.
  - Click on "SeDuMi Archive"
  - Click on "SeDuMi 1.1R2 hot!
  - Scroll down the page, choose "I agree" (if you agree - I always agreed so far and haven't had any problems) and click on "Click here to proceed".
  - This will download the zip file of the software. Save it somewhere you can find later.
  - Go and extract the contents of that zip file.
  - Start Matlab (if it is not already running).
  - Go to `File>>Set Path ...>>Add with Subfolders`. Now browse and find the SeDuMi folder you just extracted. Choose the SeDuMi folder and click on "OK."

    Example 1. *Say you saved, the SeDuMi folder at*

    `XXX\SeDuMi_1_1,`
  - Click on "Save" and "Close"
  - In the command window, type `help SeDuMi`. If you see meaningful information about SeDuMi, you probably have got SeDuMi running on your computer.

Installing YALMIP

This note will explain how to install YALMIP assuming that SeDuMi is already installed on your machine. If not, please see the notes on installing SeDuMi.

From YALMIP web page `http://control.ee.ethz.ch/~joloef/wiki/pmwiki.php`:

YALMIP is a modeling language for defining and solving advanced optimization problems. It is implemented as a free toolbox for MATLAB. The main motivation for using YALMIP is rapid algorithm development.

Recall from last week that we needed the SOSTOOLS as an interface between the user and the optimization solver SeDuMi. YALMIP does the same for a much larger family of optimization problems (including the conversion from SOS questions to corresponding optimization problems as a special case).

How to install YALMIP?

Very similar to what we did last week for SOSTOOLS and SeDuMi.

1. Go to `http://control.ee.ethz.ch/~joloef/wiki/pmwiki.php?n=Main.Download`
2. Locate the latest release (YALMIP R20080415) and click on it. This will download the compressed folder which includes the code for YALMIP.
3. Extract its contents and save somewhere on the hard disk.
4. Start Matlab.
5. Go to `File>>Set Path ...>>Add with Subfolders`. Now browse and find the YALMIP folder you just extracted. Choose the YALMIP folder and click on "OK."
6. You should see the following folders in your path:
   - /yalmip
   - /yalmip/extras
   - /yalmip/demos
   - /yalmip/solvers
   - /yalmip/modules
   - /yalmip/modules/parametric
   - /yalmip/modules/moment
   - /yalmip/modules/global
   - /yalmip/modules/sos
   - /yalmip/operators

7. If not, something is wrong. Re-try, read the YALMIP installation guide for more details, find us and ask.

8. Run `yalmiptest` (type it in the command window and hit enter). It should ask you to hit a key once and then run bunch of tests. If everything alright, then you should not see any complaint or errors.