MICHIGAN ENGINEERING
UNIVERSITY OF MICHIGAN

Installation Guide for repy_CUDD

Zephaniah Hill

The University of Michigan

Download

This installation is being done on a fresh virtual machine. The steps should be similar for any other 64 bit Ubuntu machine, but may be slightly different.

NOTE: To complete this install on a 32 bit machine you will need to use the 32 bit "make" files. To do this simply rename the 32 bit versions "Makefile"

First to install git use the command: **sudo apt install git**
To download the CUDD package with the python wrapper, navigate to a folder you would like to store the software and type:
git clone https://github.com/Zzephaniahh/repycudd

Dependencies

We also need to install make. We can get make with the command: **sudo apt install make**

We also need two compilers, gcc, and g++. To install gcc type: **sudo apt install gcc**

To get g++ type: **sudo apt install g++**

We also need python. which can be installed by typing: **sudo apt install python**

We also need python-dev to link the files later, install this with:
sudo apt install python-dev

We need swig, we can install swig with: **sudo apt install swig**

Finally, we need a fairly obscure program called libc6. To install this type: **sudo apt install libc6-dev-i386**

Build

Move into the repycudd directory by typing: **cd repycudd**

Move to the cudd folder by typing: **cd cudd-2.4.2**

Now type: **make**

To build the shared libraries we must now type: **make libso**

Now CUDD and the shared libraries should be compiled. We need to compile the repy_cudd wrapper next.

Return to the repycudd directory with:`cd ..`

Use the command: **make depend**

Finally, type: **make**

Test

This should install repycudd to the repycudd directory. You can now use CUDD in python using the wrapper. To test that the installation was successful lets use one of the examples.

Type: **cd examples**

Move one of the examples up to the repycudd folder with: **mv example1.py ..**

Then run it with python using: **python example1.py**

If the install was correct, you should see something like:

```
python example1.py
0----  1
10001  1
1010-  1
10111  1
110-1  1
1110-  1
11111  1
```

To use the EECS 598 libraries, you'll also need to install two python libraries.

Install bidirectional dictionaries with: **pip install bidict**

Install the in place file editing library with: **pip install in_place**

Using CUDD

Much of the CUDD documentation is built for the c++ CUDD package. Since we're using a python wrapper, our code will look somewhat different. Thankfully, the differences are fairly standardized. To find the wrapped function of a c++ CUDD function, simply omit the cudd prefix. additionally instead of passing the manager pointer it is used as an object usually in the form: `mgr.SomeFunction`.

For example, if the function is:
`Cudd_PrintMinterm( DdManager * manager, DdNode * node )`

This becomes: `mgr.PrintMinterms(DdNode)`

My documentation is available on my **website**.

For a list of functions look at **MIT's Index**.

Another good resource is **MIT's User Manual**.

Finally David Kebo wrote an excellent **tutorial**.