L2C Documentation

Seyed Mehran Kazemi
Department of Computer Science
University of British Columbia
Vancouver, BC, Canada
smkazemi@cs.ubc.ca

David Poole
Department of Computer Science
University of British Columbia
Vancouver, BC, Canada
poole@cs.ubc.ca

1 L2C

L2C (or LRC2CPP) compiles lifted inference into C++ programs. Please refer to [1, 3] for the details on how the compilation works. This documentation describes how the compiler can be used on MacOSX. The software is available on GitHub at `https://github.com/Mehran-k/L2C/`. The current version takes as input a file containing a theory in *.wmc*[1] format and returns the weighted first-order model count (WFOMC) of the theory.

In order to use L2C, you need to install Ruby Version Manager (RVM) and Ruby v-2.3.0 or higher. To do so, use the following command[2]:

```
\curl -sSL https://get.rvm.io | bash -s stable --ruby
```

Check the version of your default ruby using:

```
ruby -v
```

If the default version is not v-2.3.0 or higher, use the following command to change the default version:

```
rvm --default use 2.3.0
```

Also make sure the g++ compiler is installed on your machine. You can do so by observing the output for the following command:

```
g++ -v
```

Now cd to the L2C directory:

```
cd ~/<path to the L2C folder>/L2C
```

There are *10* example theories in the *examples* folder. Use the following command to run L2C for the first example (or similarly for any other example):

[1] .wmc format is described in detail in wfomc manual at `https://dtai.cs.kuleuven.be/software/wfomc`

[2] Refer to `https://rvm.io/rvm/install` for more detail.

ruby L2C.rb -f examples/example1.wmc

The *-f* flag specifies the name and the path to the input *.wmc* file. Below is a list of the other flags and what they do:

-h: Specifies the heuristic to be used for finding the branching (elimination) order of the parameterized random variables. Currently, the possible values for the heuristic are *MTS* for MinTableSize [2] and *MNL* for MinNestedLoops [3]. *MNL* is the default heuristic.

-k: If *MNL* is selected as the heuristic, the number of stochastic local search iterations can be set using *-k* flag. The default value is 25.

-r: Specifies if the C++ code generated by the compiler must be readable or not. The possible values are *true* and *false*. The default value is *false*.

As an example, in order to run L2C for the first example using *MNL* heuristic with 10 stochastic local search iterations and produce readable C++ code, use the following:

ruby L2C.rb -f examples/example1.wmc -h MNL -k 10 -r true

2 *.wmc* Syntax

The *.wmc* files consist of three parts: 1- domains, 2- predicates, and 3- clauses.

2.1 Domains

Below is an example of a domain definition:

domain movie 5 {Titanic, Her, Zootopia, Hangover, Inception }

The *domain* at the beginning of the line indicates that this line defines a domain, *movie* is the name of the domain, 5 is the size of the domain, and then come the names of the 5 movies in the domain.

2.2 Predicates

Below is an example of a predicate definition:

predicate Likes(person, movie) 2.12 1

The *predicate* at the beginning of the line indicates that this line defines a predicate, *Likes* is the name of the predicate, *person* and *movie* are its input types (the types of the logical variables), and 2.12 and 1 represent the weight associated with $Likes$ being true and false respectively. If the weights are not specified, both weights will be considered 1.

2.3 Clause

Below is an example of a clause definition:

!Smokes(x) ∨ *!Friend(x, y)* ∨ *Smokes(y), x* != *y*

! is used to define negated literals, v is used to define the logical OR, and the constraints are separated from the clause using a comma. Currently, *logvar != logvar* is the only type of allowed constraint. If there are more than one constraints for a clause, they must be separated using ^. Lines starting with // are considered as comment lines. The *examples* folder contains 10 example theories in *.wmc* format.

3 Contact

Report any issues to:

Seyed Mehran Kazemi
Computer Science Department
The University of British Columbia
568-2366 Main Mall, Vancouver, BC, Canada (V6T 1Z4)
Webpage: `http://www.cs.ubc.ca/~smkazemi/`
Email: smkazemi@cs.ubc.ca

4 License

Licensed under the GNU General Public License Version 3.0.
`https://www.gnu.org/licenses/gpl-3.0.en.html`
Copyright (c) 2016, The University of British Columbia. All rights reserved.

References

[1] Seyed Mehran Kazemi and David Poole David. Why is compiling lifted inference into a low-level language so effective? *IJCAI-16 Statistical Relational AI Workshop*, 2016.

[2] Seyed Mehran Kazemi and David Poole. Elimination ordering in first-order probabilistic inference. In *Proc. of Association for the Advancements of Artificial Intelligence (AAAI)*, 2014.

[3] Seyed Mehran Kazemi and David Poole. Knowledge compilation for lifted probabilistic inference: Compiling to a low-level program. In *Proc. 15th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 2016.