Ring - Specification Environment on Top of Maude (Manual)

Ha Thi Thu Doan* Adrián Riesco† Kazuhiro Ogata*

*School of Information Science, JAIST
†Universidad Complutense de Madrid
{doanha,ogata}@jaist.ac.jp
ariesco@fdi.ucm.es

Contents

1 Introduction

This manual gives an introduction and the guide to use the ring - specification environment. This environment is designed to specify *ring* topology. The environment is built on top of Maude, a powerful rewriting logic-based programming and specification language. We extend Maude [1] to add a `ring` attribute that allows us to define the ring data structure.

A specifier can specify a *ring* topology by using the `ring` attribute. The `ring` attribute allows users to indicate that a given constructor behaves as a ring topology. The specification environment is built as depicted in Fig. 1. A specification in the environment is considered as a user specification. The main player in the system is Transformer that takes a user specification and transfers it into an ordinary Maude specification, which can be executed by Maude. The two following important factors are guaranteed:

- The way to declare the `ring` attribute is the same as any other Maude attribute.
- The transformation is transparent to specifiers.

The environment guarantees that all Maude facilities, such as the LTL model checker, can be directly used for user specifications.

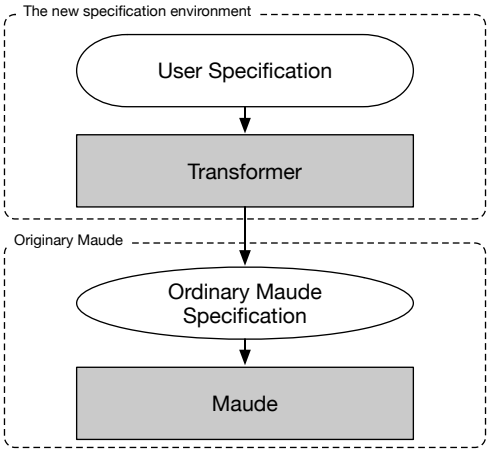

Figure 1: Architecture of the new specification environment.

2 Ring - Specification Environment

2.1 Using Ring - Specification Environment

2.1.1 Getting Ring - Specification Environment

The Ring - Specification Environment is available at the link:

`https://github.com/doanha86/PODC-2018.`

There you can find also the manual of the environment and the specification of mobile robot exploration on a ring shape network as a case study.

2.1.2 Running Ring - Specification Environment

The `maude-ring.maude` file should be located in the same directory with Maude (full-maude version). The environment can be loaded after starting Maude section.

```
~ ./maude.darwin64
                         \||||||||||||||||||/
                       --- Welcome to Maude ---
                         /||||||||||||||||||\
                Maude 2.7 built: Mar  3 2014 18:07:27
                Copyright 1997-2014 SRI International
                        Thu Feb  5 16:39:35 2018


Maude> load maude-ring.maude

                            Full Maude 2.7 March 10th 2015

                      Ring attributes available(October 25th,2017)

Maude>
```

The environment is now ready to accept modules and commands.

The environment is built on top of Maude by extending FULL-MAUDE [1]. A module, therefore, can be entered in the environment by enclosing it in parentheses. For example, a module SIMPLE-RING can be entered as follows:

```
Maude> (mod SIMPLE-RING is
         sort Seq .
         sort Config .

         op emp : -> Seq [ctor] .
         op __ : Seq Seq -> Seq [ctor assoc id: emp] .
         op <_> : Seq -> Config [ctor ring].
       endm)

Introduced module SIMPLE-RING
```

We can also input one or several modules by saving in a file (noting that a module is need to be enclosed in parentheses) and then entering the file with `in` or `load` commands. For example, the file `my-ring.maude` contains the module `SIMPLE-RING` above, we can do the following to enter it:

```
Maude> load my-ring.maude
Introduced module SIMPLE-RING
```

2.2 Syntax

There are two kind of rings: *oriented* rings in which the orientation of a ring such as clockwise order and anti-clockwise order is taken into account, and *disoriented* rings in which there is no orientation of a ring.

A *disoriented* ring can be defined by using the `ring` attribute. The ring is constructed by a sequence of elements. The sort for sequences of elements is `Seq`. The configurations of a system as a ring could be defined as follows:

```
op emp : -> Seq [ctor] .
op __ : Seq Seq -> Seq [ctor assoc id: emp] .
op <_> : Seq -> Config [ctor ring].
```

For *oriented* rings, the environment provides the `r-ring` attribute that could be considered as a sub-class attribute of the `ring` attribute.

```
op emp : -> Seq [ctor] .
op __ : Seq Seq -> Seq [ctor assoc id: emp] .
op <_> : Seq -> Config [ctor r-ring].
```

3 Case Studies

We have formalized and specified a mobile robot algorithm on the ring in the environment as a case study.

3.1 Specification and Model Checking of Mobile Robot Exploration on Ring

We consider the problem of exploring with stop an disoriented ring by a group of identical mobile robots. The robots are oblivious. About timing assumption, the ASYNC model is considered. In addition, there may be more than one robot located at the same node. Each robot can distinguish whether a node is empty, occupied by one robot, or more than one robot. When there is more than one robot, the node is called a multiplicity (or a tower). The problem of exploring with stop requires that regardless of the initial placement of the robots, each node must be visited by at least one robot and the robots must be in a configuration in which they all remain idle. The mobile robot exploration algorithm [2] is specified and model checked in the environment. The specification and model checking is available in the file `exploration.maude`.

References

[1] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. Talcott. *All About Maude*, volume 4350 of *LNCS*. Springer, 2007.

[2] P. Flocchini, D. Ilcinkas, A. Pelc, and N. Santoro. Computing without communicating: Ring exploration by asynchronous oblivious robots. *Algorithmica*, 65(3):562–583, 2013.