How to use the framework

In this document describe how the framework is implemented and how it is used in practice. Our framework uses the μCRL toolset[1], LTSmin[2], and JFLAP[3].

The μCRL toolset allows manipulation of linear process equations (LPEs), state space generation, simulations, reduction tools, manipulation of labeled transition systems (LTS). A linear process equation is a process declaration that does not contain any parallel operators, encapsulations, or hidings. Using the LPE manipulation tools, we can declare the mid-point process as the parallel composition of the two end-points and the environment and compute the LPE corresponding to the mid-point process. Then, we use the state space generation tool to expand the mid-point's state space. LTSmin implements the branching bisimulation reduction algorithm, which we use to minimize the mid-point's state space. We implemented a script which transforms the LTS format output by the μCRL toolset into the format that can be input to the JFLAP tool. Using the JFLAP tool, we convert the mid-point to a deterministic finite state machine by applying a standard algorithm for transforming a non-deterministic finite automaton with silent steps to a deterministic finite automaton.

In this example, we assume that the protocol, the environment, and all data sort specifications are stored in a file example.mcrl. We use the mcrl tool to compute and transform the mid-point's specification into a linear process equation:

```
$mcrl -tbf example.mcrl
```

The mcrl program produces a file example.tbf, which contains the linear process equation of the mid-point. It also checks the syntax and the static semantics of the μCRL specifications stored in example.mcrl. Afterwards, the mid-point's state space is generated using the instantiator program as follows:

```
$instantiator example.tbf
```

The instantiator program outputs a file example.aut. This file describes the state space of the mid-point. If the mid-point has an infinite state space, one can use the msim simulator, which interactively simulates a system described by an LPE, i.e. the example.tbf file. After expanding the state space, it can be visualized using the CADP toolset[4]. To minimize the mid-point's state space,

[1] the μCRL toolset is available at `http://homepages.cwi.nl/~mcrl/`

[2] LTSmin is available at `http://fmt.cs.utwente.nl/tools/ltsmin/`

[3] The JFLAP tool is available at `http://www.cs.duke.edu/csed/jflap/`

[4] The CADP toolset can be downloaded at `www.inrialpes.fr/vasy/cadp`.

we use the LTSmin tool to apply a branching bisimulation reduction on the mid-point's state space.

```
$ltsmin -b -o example_min.aut example.aut
```

The minimized mid-point's state space is stored in the example_min.aut file. To open the mid-point's specification using the JFLAP tool, we need to transform the aut representation of the state space to the format accepted by JFLAP. The JFLAP tool uses a specific jff format as input. The aut to jff transformation is carried out by a aut2jff.pl perl script, which we implemented. The script is used as follows:

```
$perl aut2jff.pl example_min.aut example.jff
```

The example.jff file can be opened in JFLAP and the mid-point's state machine can be converted to a deterministic finite state machine using the "Convert to DFA" function as shown in Figure .

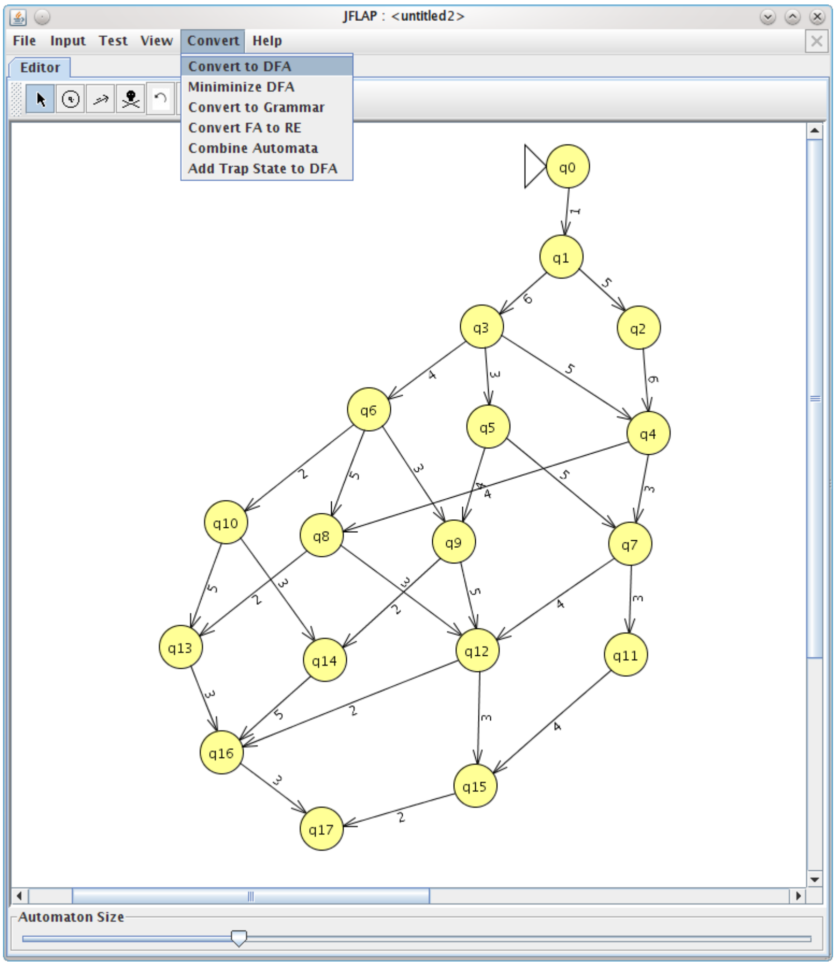

Figure 1: The JFLAP tool: converting the mid-point's state machine to a DFA