UNIVERSITY OF CAGLIARI, DEPARTMENT OF MATHEMATICS AND COMPUTER SCIENCE

Tutorial for the TST Online Tool

Massimo Bartoletti, Sebastian Podda, Livio Pompianu

This brief tutorial will introduce you to the online validation tool for *Timed Session Types*. You can found that tool at: http://co2.unica.it. The web interface allows you to type string contracts, check their syntax and verify their compliance.

String syntax

Timed session types can be expressed in a nicely, human-readable string format. Note that spaces and new lines are ignored by the tool. Here is a brief syntax explaination:

EMPTY INTERNAL ACTION

```
!action
```

Empty braces can be omitted (branch labels must contains only lowercase a-z letters).

INTERNAL ACTION WITH GUARDS AND WITHOUT RESETS

```
!action{x<5,y>4}
```

Clock labels must contains only lowercase a-z letters.

INTERNAL ACTION WITH GUARDS AND RESETS

```
!action{x<5,y>4;x,y}
```

INTERNAL ACTION WITH RESETS AND WITHOUT GUARDS

```
!action{;x,y}
```

EMPTY EXTERNAL ACTION

```
?action{}
```

Same as internal action, but preceded by ?.

INTERNAL CHOICE BETWEEN TWO BRANCHES

```
!first{x>4;x} + !second{x<4;x}
```

EXTERNAL CHOICE BETWEEN THREE BRANCHES

```
?first{} & ?second{;z} & ?third{x>20;y}
```

SEQUENCE OF ACTIONS (DOT OPERATOR HAVE PRIORITY TO CHOICE OPERATORS)

```
!first{;x,y} . ?second{x<4,y>2}
```

RECURSION WITH CALL (REC* AND CALL OPERATOR)

```
REC 'x' [ ?a.'x' & ?b ]
```

Recursive branch must be enterely closed by [ ] brackets. Recursive variables must be enclosed in single quotes (variable names must contains only lowercase a-z letters).

XML syntax

TST contracts can also be expressed in XML format, using the following syntax. Currently, the web interface doesn't accept the XML format (but APIs accept XML syntax only). Each XML contract must start and terminate with a `contract` tag:

```
<contract>
     . . .
</contract>
```

Other elements are internal actions (`intaction`), external actions (`extaction`), sequences of actions (`sequence`), recursive sequence (`rec`), internal choices (`intchoice`) and external choices (`extchoice`). You can also "call" a recursive sequence (`call`).

Internal and external actions can have guards containers (`guards`), each containing single guards (`guard`), and resets containers (`resets`), each containing single resets (`reset`). Guards may specify operators and integer values.

In addition, internal and external actions, guards, resets, recursive sequences and calls must have a mandatory `id` attribute, which will contain the label of the action/sequence. Here is a brief example:

```
<contract>
     <sequence>
          <!-- First action, with guards and resets -->
          <intaction id="pay">
               <guards>
                    <guard id="x" op="great" value="5" />
               </guards>
               <resets>
                    <reset id="x" />
               </resets>
          </intaction>
          <!-- Second action, with empty guard -->
          <extaction id="receive" />
     </sequence>
</contract>
```

Here is another example with recursive sequence:

```
<contract>
```

```
    <sequence>
        <intaction id="pay" />
        <!-- Recursive sequence -->
        <rec name="y" />
            <intaction id="wait" />
            <call name="y" />
        </rec>
    </sequence>
</contract>
```

There's an important restriction when using some tags: if you want to express a sequence (or a choice) of multiple elements, you cannot use nested tags. For example:

```
 <contract>
    <sequence>
        <intaction id="x" />
        <intaction id="y" />
        <intaction id="z" />

        <!-- THE FOLLOWING IS NOT ALLOWED -->
        <sequence>
            <intaction id="p" />
            <intaction id="q" />
        </sequence>
    </sequence>
</contract>
```