Assignment 1: The rope bridge - in MCRL2

Luís Albuquerque (A79010) e Rafaela Pinho (77293)

April 12, 2018

Contents

1 Exercicio 1

1.1 Parmetros do processo Aventureiro

O Aventureiro tem duas ações possveis.

1. Atravessar a ponte no sentido start para finish: forward_adventure
   Uma vez que o máximo de aventureiros a atravessar a ponte é 2, é lógico que sempre que atravessem neste sentido vão aos pares. Como a identificação dos aventureiros é pela sua velocidade, forward_adventure tem como parametro dois inteiros, que corresponde ás suas velocidades.

   ```
   forward_flashlight: Int # Int;
   ```

2. Atravessar a ponte no sentido finish para start: back_adventure
   O objetivo do problema é todos atraveçem a ponte, por isso o mais lógico será apenas um aventureiro regressar para o "start". Daí os paramentro seja apenas um inteiro, a sua velocidade.

   ```
   back_flashlight: Int;
   ```

1.2 Processo do Aventureiro

O processo Aventureiro recebe a sua posição, ou "start" ou "finish" e a velocidade, que é o seu identificador. E caso a posição seja "start" avançam dois aventureiros, então escolhemos avançar com mais rápido dos dois, e com o outro. ?? mal explicado
Caso a posiç seja "finish" pretendemos regressar ao "start", desta vez s vai um, por isso não há escolhas a fazer, apenas regressar com a velocidade "speed" que recebemos como paramentro e mudarmos o estado para "start".
(falta acabar de explicar)

```
proc Adventurer (pos:Position , speed:Int) =
        (pos == start ) ->
        sum s: Int .(s > speed)->
                foward_adventure(speed,s ).Adventure(finish ,speed)
                <>
                foward_adventure(s,speed).Adventure(finish ,speed)
        <>
        back_adventure(speed).Adventure(start ,speed);
```

2 Exercicio 2

2.1 Inicializar os 4 aventureiros e a lanterna

Os aventureiros começam todos do lado "start" e com a sua respetiva velocidade. Para os colucar em paralelo usamos " — —".

```
init
  Adventure(start,1) || Adventure(start,2) || Adventure(start,5) ||
  Adventure(start,10) || Flashligth(start)
  Flashligth(start) ;
```

2.2 Comunicação de processos

Para que os proçessos comuniquem usamos o comando "comm".

```
init
 comm (
 {
  forward_adventurer | foward_adventurer | forward_flashligth -> forward,
  back_adventurer | back_flashligth -> back
  },

  Adventure(start,1) || Adventure(start,2) || Adventure(start,5) ||
  Adventure(start,10) || Flashligth(start)
);
```

2.3 Garantir que apenas as ações "forward" e "back" possam ocorrer.

Para que apenas os processos "forward" e "back" possam ocorrer usa-se o comando "allow"

```
init
allow({ forward, back },
 comm (
 {
  forward_adventurer | foward_adventurer | forward_flashligth ->forward,
  back_adventurer | back_flashligth -> back
  },
  Adventure(start,1) || Adventure(start,2) || Adventure(start,5) ||
  Adventure(start,10) || Flashligth(start)
  )
);
```

3 Exercicio 3

3.1 mcrl22lps

3.2 lpsxsim

4 Exercicio 4

4.1 lps2lts

4.2 ltsgraph

4.3 ltsview

5 Exercicio 5