

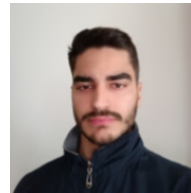


**Universidade do Minho**  
Escola de Engenharia

# Programação Ciber-Física 2023/2024

## Trabalho Prático 2

Ricardo Lopes Santos Silva :: pg54188; Afonso Xavier Cardoso Marques :: pg53601



### I. INTRODUÇÃO

Neste trabalho, abordaremos a modelação e análise de um sistema ciber-físico utilizando a linguagem de programação Haskell, com ênfase no uso de monads. Monads são uma poderosa abstração em Haskell que permitem tratar efeitos como estado, I/O, e não-determinismo de uma forma funcional e elegante. A tarefa proposta envolve a modelação de um cenário específico onde quatro aventureiros precisam de atravessar uma ponte de corda, respeitando restrições de segurança e tempo. Através desta modelação, pretendemos demonstrar a viabilidade de cumprir os requisitos temporais impostos, além de garantir que todas as regras de segurança sejam respeitadas.

O relatório elaborado explica o código desenvolvido e as conclusões obtidas durante a análise. Este exercício não só ilustra a aplicação prática das monads em Haskell, mas também destaca a importância da modelação precisa em sistemas ciber-físicos complexos.

#### A. *Contextualização*

No meio da noite, quatro aventureiros encontram uma ponte de corda velha que atravessa um desfiladeiro profundo. Por razões de segurança, decidem que não mais do que duas pessoas devem atravessar a ponte ao mesmo tempo e que uma lanterna precisa de ser carregada por um deles em cada travessia. Eles têm apenas uma lanterna. Os quatro aventureiros não são igualmente habilidosos: atravessar a ponte leva-lhes 1, 2, 5 e 10 minutos, respetivamente. Um par de aventureiros atravessa a ponte em um tempo igual ao do mais lento dos dois aventureiros.

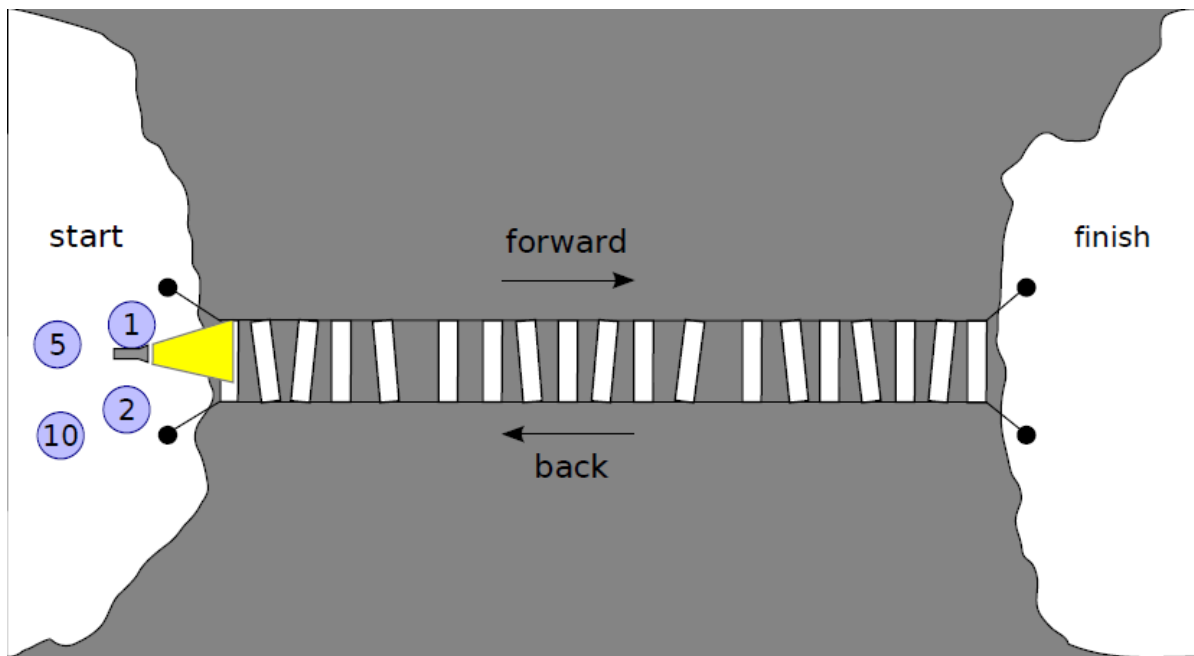


Fig. 1: Cenário em estudo

Um dos aventureiros afirma que eles não podem todos estar do outro lado em menos de 19 minutos. Um companheiro discorda e afirma que isso pode ser feito em 17 minutos.

## II. EXERCÍCIO 1

A primeira tarefa é verificar as afirmações seguintes utilizando Haskell. Especificamente, será preciso:

- 1) **modelar** o sistema acima usando **monads**, em particular as de duração e não-determinísticas;
- 2) mostrar que é realmente **possível** que todos os aventureiros estejam do outro lado em 17 minutos;
- 3) mostrar que é **impossível** que todos os aventureiros estejam do outro lado em menos de 17 minutos.

Para cumprir esta tarefa, completou-se o código no anexo (Adventurers.hs), ou seja, adicionou-se uma definição às funções que carecem de uma definição, seguindo os comentários presentes no código. Para auxílio à resolução, usou-se como inspiração o monad de duração dos slides e do código Haskell que foi previamente fornecido pelo docente. Também se analisou detalhadamente o código referente ao *Knight's quest* e, em particular, a monad LogList.

### 1) Verificação de Propriedades:

A resolução que se segue ...

codigo em haskell

## III. EXERCÍCIO 2

A segunda tarefa consiste em comparar ambas as abordagens (via UPPAAL e Haskell) para o problema dos aventureiros. Especificamente, o objetivo é fornecer os pontos fortes e fracos das duas abordagens: quais são as (des)vantagens do UPPAAL para este problema? E quanto ao Haskell?

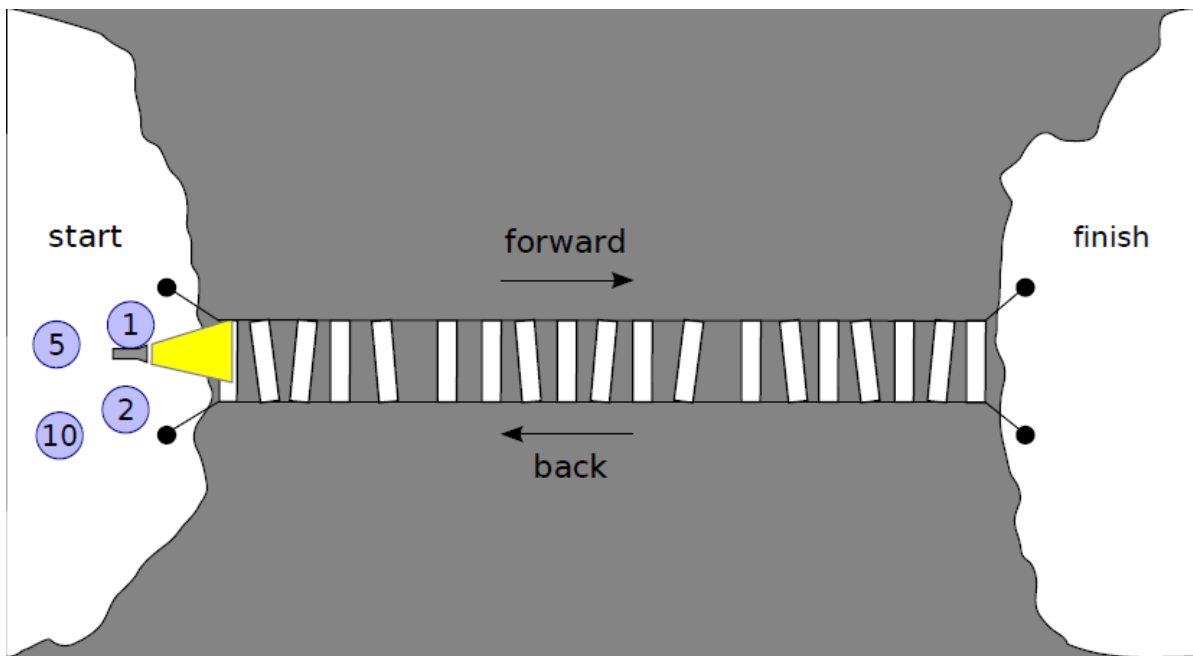


Fig. 2: Cenário em estudo no UPPAAL

#### IV. CONCLUSÃO

Com este projeto terminado, apresentamos uma breve conclusão que consideramos englobar todo o processo de aprendizagem despoletado por este trabalho. Em suma, consideramos que o trabalho desenvolvido e os resultados apurados são positivos, demonstrando uma clara capacidade em aplicar os conceitos das aulas de Programação Ciber-Física. Destacamos como pontos positivos as soluções encontradas para cada um dos cenários e em particular o cumprimento das regras de cada um. No entanto, reconhecemos que existem alguns pontos a melhorar, tais como aumentar a complexidade de alguns dos autómatos, permitindo assim que as propriedades que não se conseguiram provar nas duas fases exploradas passassem a ser satisfazíveis. Como trabalho futuro, seria também interessante introduzir cenários ainda mais complexos, que envolvessem, por exemplo, mais do que duas estradas, introduzindo assim um novo conjunto de desafios.

#### APPENDIX A

##### SIMULAÇÃO DO UPPAAL NA PARTE 1

`images/parte_1/P1_simulation.png`


#### APPENDIX B

##### RESULTADOS VERIFIER DO UPPAAL NA PARTE 1

`images/parte_1/P1_verifier.png`

## APPENDIX C


### SIMULAÇÃO DO UPPAAL NA PARTE 2



images/parte\_2/P2\_simulation.png

## APPENDIX D

### RESULTADOS VERIFIER DO UPPAAL NA PARTE 2



images/parte\_2/P2\_verifier.png

## REFERENCES

- [1] <https://uppaal.org/features/>
- [2] <https://haslab.github.io/MFP/PCF/2324/index>
- [3] <https://www.youtube.com/watch?v=7yDmGnA8Hw0>
- [4] <https://www.haskell.org/>
- [5] [https://en.wikipedia.org/wiki/Monad\\_\(functional\\_programming\)](https://en.wikipedia.org/wiki/Monad_(functional_programming))