

UNIVERSIDADE DO MINHO

MESTRADO INTEGRADO EM ENGENHARIA INFORMÁTICA

ARQUITECTURA E CÁLCULO

---

# Adventures in a Rope-Bridge

---



Lucas Pereira A68547



Ricardo Pereira A73577

May 18, 2020

# 1 Introdução

Este trabalho consistia na resolução de um problema em que têm de ser tomadas decisões dependentes do tempo. O caso em questão envolve quatro aventureiros e uma ponte que os mesmos têm que atravessar sempre acompanhados de uma lanterna, contudo existem algumas restrições que os mesmos têm que cumprir:

- a ponte deve ser atravessada no máximo por dois aventureiros;
- apenas existe uma lanterna;
- cada aventureiro demora um período diferente a atravessar a ponte (um, dois, cinco e dez minutos).
- quando dois aventureiros atravessam a ponte, o tempo dessa travessia é igual ao tempo de travessia do aventureiro mais lento;

Tendo em conta estes factos e condições, pretende-se saber qual o tempo total da travessia feita pelos quatro aventureiros. A solução deve ser dezassete minutos e menos do que isso deve ser impossível, algo que deve ser comprovado com a ajuda do *UPPAAL*. A parte interativa do *UPPAAL* também é algo bastante apelativo, na medida em que permite o utilizador visualizar e interagir com aquilo está a acontecer num determinado traço.

# 2 Resolução

Posto isto, no *UPPAAL* começou-se por declarar variáveis globais destacando-se a *time* (que indica o tempo total decorrido na travessia dos 4 aventureiros), a *flashSide* (que indica o lado em que a lanterna se encontra; tem valor 0 se a mesma estiver do lado inicial, caso contrário tem valor 1), a *carrierID* (que contém o identificador do aventureiro que carrega a lanterna ou o valor 0 caso nenhum a esteja a carregar), a *companion* (que indica se o portador da lanterna está ou não acompanhado; tem valor 0 caso esteja, caso contrário tem valor 1) e as variáveis do tipo *channel* cujas utilidades serão explicadas nos *templates* em que forem utilizadas.

Faz-se agora uma caracterização de cada *template*, tendo em conta que no ***Adventurer*** são utilizados os *templates* ***Companion*** e ***Flashlight***. O *template* ***Flashlight*** (figura 1) caracteriza o comportamento de um aventureiro pegar ou largar a lanterna com os *channels* *take* ou *release*, respetivamente, já o *template* ***Companion*** (figura 2) caracteriza o comportamento de um aventureiro se juntar ou separar do portador da lanterna com os *channels* *goalong* ou *leavecompanion*, respetivamente.

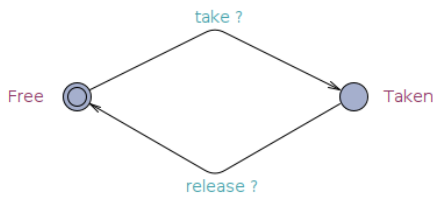


Figure 1: *Template Flashlight*.

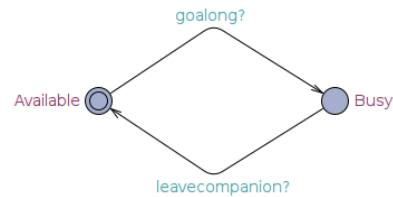


Figure 2: *Template Companion*.

Por sua vez, o *template* ***Adventurer*** (figura 7) é aquele que é mais complexo, caracterizando o comportamento de um aventureiro. Na criação de uma instância de *Adventurer* devem ser introduzidos o tempo de travessia (*delay*) e o *adventurerID* do aventureiro. Este *template* vai ser analisado por partes, cada uma com todas as opções que um aventureiro tem a partir de uma determinada localização. É ainda importante referir que cada instância de *Adventurer* possui uma variável *y*, um relógio que regula o tempo do aventureiro. Posto isto, o processo inicia-se na localização *Begin*, onde o *Adventurer* pode optar por dois caminhos:

- *take* - Representa a aquisição da lanterna, que necessita de estar no lado inicial (*flashside == 0*) e não pode ter qualquer portador (*carrierID == 0*), ficando o portador a ser o aventureiro em questão (*carrierID = adventurerID*).
- *goalong* - O aventureiro pode optar por este caminho se a lanterna estiver do lado inicial (*flashside == 0*), se já existir alguém a transportá-la (*carrierID != 0*) e se não existir

ninguém a acompanhar o portador da lanterna ( $companion == 0$ ), ficando o "lugar de acompanhante" do portador ocupado ( $companion = 1$ ).

Uma vez que ambas as ações são instantâneas, as mesmas não consomem qualquer tempo, sendo necessário indicar esse facto ao *UPPAAL* ( $y = 0$ ). Concluídas estas decisões, inicia-se a travessia da ponte.

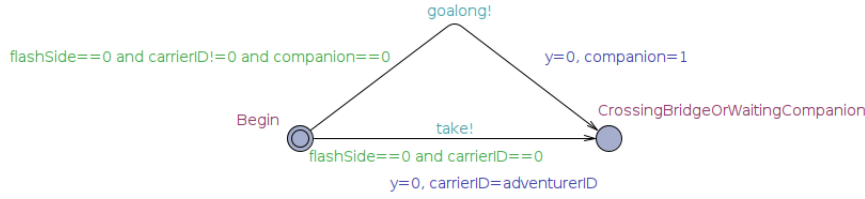


Figure 3: Channel *goalong* e *take*.

Partindo para outro ponto, *CrossingBridgeOrWaitingCompanion*, existem novamente duas opções que o aventureiro deve tomar, e as mesmas estão totalmente dependentes das opções que escolheu anteriormente. Estas são:

- *release* - Traduz a ação de largar a lanterna e deve impreterivelmente ser feita pelo aventureiro que anteriormente fez *take* ( $adventurerID == carrierID$ ), ficando a lanterna sem portador ( $carrierID = 0$ ) e do lado contrário ao inicial ( $flashSide = 1$ ).
- *leavecompanion* - O aventureiro escolhe este caminho se anteriormente optou por *goalong*, ou seja, se tiver sido apenas "acompanhante" durante a travessia ( $adventurerID != carrierID$ ), deixando posteriormente de o ser ( $companion = 0$ ) e ficando a lanterna do lado contrário ao inicial.

Pressupõem-se que qualquer um destes caminhos é tomado quando o tempo de travessia do aventureiro mais lento for atingido ( $y \geq delay$ ), isto é, quando a travessia da ponte for feita por ambos.

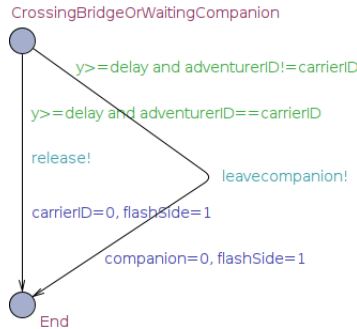


Figure 4: Channel *release* e *leavecompanion*.

Constatou-se anteriormente que a lanterna é largada no lado final da ponte ( $flashSide == 1$  and  $carrierID == 0$ ) e o acompanhante, existindo, deixava de o ser ( $companion == 0$ ); essas são as pré-condição para esta ação. Agora, um dos aventureiros que se encontra do lado final da ponte terá que voltar atrás com a lanterna ( $carrierID = adventurerID$ ) para ir buscar mais um aventureiro, pelo que se volta a repetir o *take*, que é instantâneo ( $y = 0$ ). O aventureiro em questão inicia assim a travessia da ponte no sentido inverso.

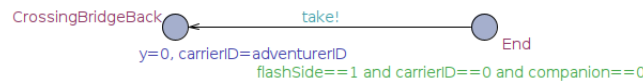
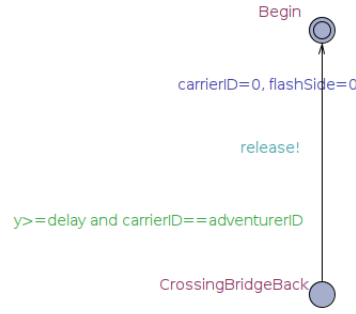
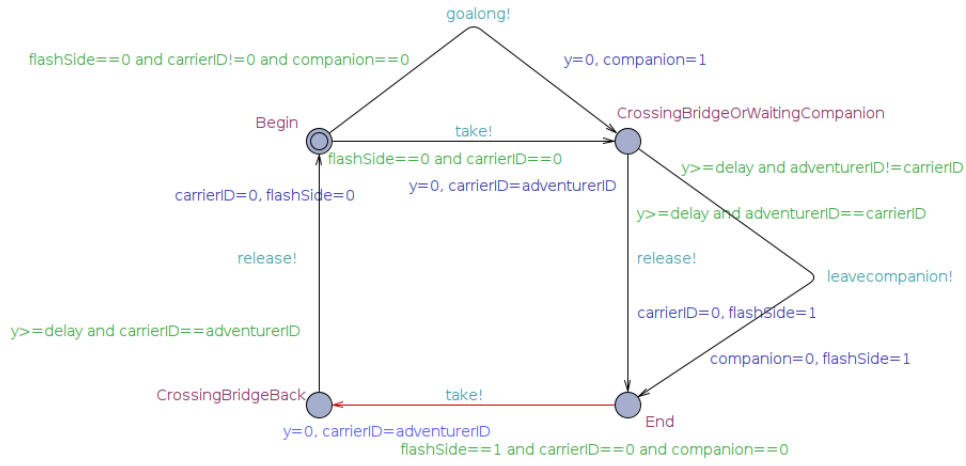


Figure 5: Channel *take*.

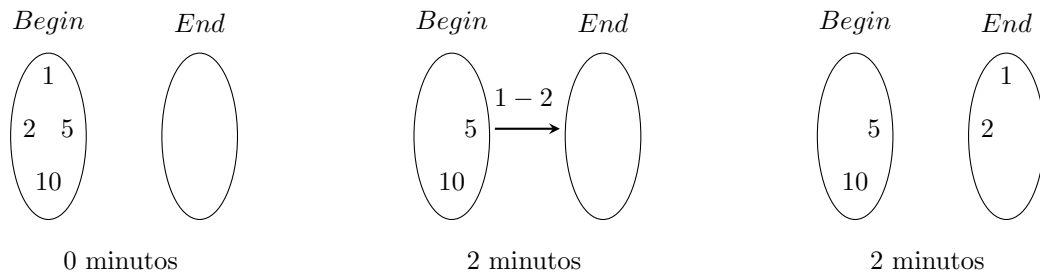
A execução do *release* acontece depois do aventureiro ter concluído com a lanterna ( $carrierID == adventurerID$ ) a travessia no sentido inverso (*CrossingBridgeBack*), que demora o tempo de travessia preestabelecido para o mesmo ( $y \geq delay$ ). A lanterna é novamente largada ( $carrierID = 0$ ) no lado inicial da ponte ( $flashSide = 0$ ).

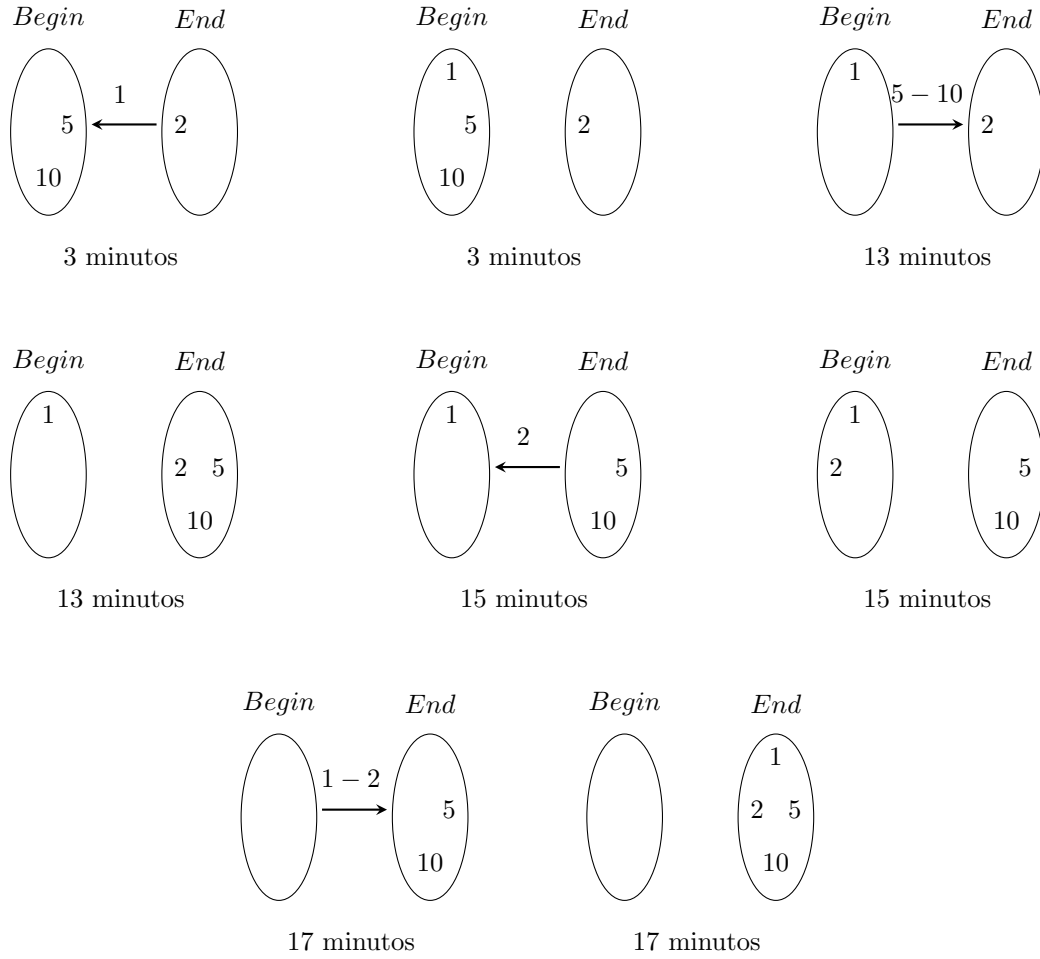
Figure 6: *Channel release*.

O processo volta a repetir-se mas é necessário utilizar o *Verifier* do *UPPAAL* para concluir o principal objetivo: os aventureiros não conseguem atravessar a ponte em menos de dezassete minutos. O *template* no seu todo pode ser observado na imagem seguinte.

Figure 7: *Template Adventurer*.

Para a execução do modelo ser feita de acordo com o problema proposto, é necessário criar quatro instâncias de *Adventurer*, atribuindo-lhes identificadores e tempos de travessia. Assim sendo, o traço ideal será aquele em que a travessia para o lado final da ponte é sempre feita com duas pessoas, a travessia para o lado inicial com uma e a minimização do tempo é uma prioridade. No *UPPAAL*, **não se consegue qualquer traço para *time* inferior a dezassete**, mas **consegue-se para *time* igual a dezassete**, tal como esperado. Assumamos que na demonstração a seguir, o identificador do aventureiro é igual ao valor do tempo que demora a fazer uma travessia da ponte:





Está identificado o traço ideal que corresponde a um tempo total de dezassete minutos.

### 3 Conclusão

O problema em questão resolvido em *UPPAAL* permite verificar a eficácia desta ferramenta em situações do género. De uma forma muito prática, conseguem-se estabelecer condições e fazer alterações de acordo com a ação que é feita, permitindo-se ainda verificar se determinados traços acontecem ou até se determinadas propriedades se verificam em determinados/todos os traços.