UNIVERSITÉ PARIS 13

Institut Galilée, Université Paris 13 — 2017–2018
Master 2 PLS

SITH

TP 2

Étienne André

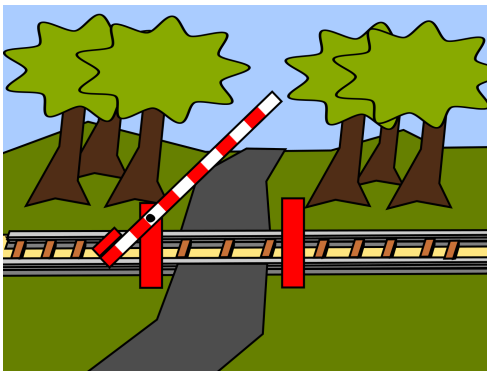
Version du sujet : 23 février 2018

Ce TP s'effectuera individuellement. Vous rédigerez un compte-rendu qui contiendra toutes les commandes et toutes les réponses aux questions ; il sera envoyé à l'enseignant à l'issue du TP (avant 11h45). Ce compte-rendu prendra la forme d'un document mis en forme avec LibreOffice et exporté en PDF, ainsi que le fichier `train.imi`
Nom du fichier du compte-rendu : `SITH-TP2-nom-prenom.pdf`
Objet du courriel : `[M2 PLS] SITH TP n°2`
Adresse électronique : `SITH(arobase)lipn13.fr`

IMITATOR et les fichiers de modèles sont téléchargeables sur
https://www.lipn.fr/~andre/enseignement/SITH/

Exercice 1 : Spécification et vérification d'un passage à niveau

On considère un système de contrôle de passage à niveau.



Source : https://commons.wikimedia.org/wiki/File:Schranke_fast_oben.svg (contrat cc by)

Un capteur est installé à une certaine distance en amont du passage à niveau ; le temps entre le moment où le train active le capteur (action approche) et celui où il arrive au passage à niveau (action passe) est d_{arrive}. Une fois que le train active le capteur, cela va enclencher la baisse de la barrière (action baisse) ; le temps entre le moment où le train active le capteur et celui où la barrière commence à descendre est $d_{activeBarrière}$. Le temps entre le moment où la barrière commence à descendre et celui où la barrière est entièrement baissée (action enBas) est $d_{descend}$.

Si le train passe quand la barrière est déjà baissée, le train s'éloigne instantanément, et la barrière met 5 unités de temps à remonter, puis le système repart dans son état initial.

On suppose que des voitures, piétons ou vélos peuvent toujours passer tant que la barrière n'est pas complètement baissée. Si le train passe alors que la barrière n'est pas complètement baissée, une collision se produit, et le système part dans un état crash. Par souci de simplification, on suppose que le train suivant ne peut pas passer pendant que la barrière remonte.

Question 1 : Rédiger un (unique) automate temporisé paramétré au format IMITATOR dans un fichier appelé `train.imi` et qui modélise ce système. On considère qu'il n'y a qu'une seule voie, que les trains ne passent que dans un seul sens et que, même si plusieurs trains peuvent se suivre, un second train n'arrive qu'une fois que le premier a quitté le passage à niveau et que la barrière est remontée.

La syntaxe de votre automate devra être cohérente avec celle vue en cours : une contrainte (dans une garde ou un invariant) est une conjonction de comparaisons de la forme $x \sim p$ ou $x \sim c$.

Question 2 : Afin de visualiser votre travail, générer l'automate sous forme graphique avec la commande suivante :

```
$ ./imitator train.imi -PTA2JPG
```

Question 3 : Quelle est la propriété de sûreté de ce passage à niveau ? L'exprimer en CTL.

Question 4 : En utilisant l'algorithme EF-synthesis, déterminer automatiquement toutes les valeurs des paramètres qui garantissent l'absence de collision.

Exercice 2 : Vérification d'un système matériel

Question 1 : Télécharger le modèle `AndOr.imi` ; le générer sous forme graphique (automatiquement), et l'étudier. De quel type de système matériel s'agit-il ?

Question 2 : Générer le graphe des états symboliques. Que constate-t-on ? Pourquoi ?

Question 3 : Ouvrir le fichier `AndOr.pi0`. Quelle est l'instance des paramètres définie dans ce fichier ?

Question 4 : Appliquer la méthode inverse à partir de `AndOr.imi` et `AndOr.pi0`. Que constate-t-on quant à la terminaison ? Quelle est la contrainte générée ?

Question 5 : Nous allons chercher à minimiser ou maximiser la valeur de `dAnd_u` définie dans `AndOr.pi0`, sans toucher aux autres valeurs. Quelle est la borne minimale acceptable pour `dAnd_u` telle que l'on ne change pas le comportement du système ? Et sa borne maximale ?

Exercice 3 (complémentaire) : Amélioration du train

Question 1 : On suppose désormais que, pendant que la barrière remonte, un nouveau train peut commencer à approcher. Modifier votre modèle pour prendre en compte ce cas. Il est possible qu'il faille ajouter plusieurs états supplémentaires.

Question 2 : Améliorer votre modèle de passage à niveau pour que les 3 temps considérés soient désormais des intervalles. Par exemple, le temps entre le moment où le train active le capteur et celui où il arrivera au passage à niveau devient un intervalle $[d_{arrive}{}^{-}, d_{arrive}{}^{+}]$.

Question 3 : Calculer l'ensemble des valeurs des paramètres pour lesquelles aucune collision ne peut se produire.

Question 4 : Améliorer votre modèle pour qu'un second train puisse arriver alors que le premier n'était pas encore sorti du passage à niveau. Peut-il y avoir en plus une collision entre les deux trains ? Dans quelle situation ?

Question 5 : Même question pour 3 trains. Que remarque-t-on sur la complexité du modèle ? Peut-on facilement généraliser à n trains ?