1. Evaluation des besoins pour la conversion des modèles ANIMO/UPPAAL vers IMITATOR.

Ce document présente les améliorations à apporter au model checker IMITATOR pour qu’il puisse supporter une conversion des modèles des outils ANIMO/UPPAAL.

Le problème de la transformation des modèles ANIMO vers les modèles IMITATOR est celui de passer d’un modèle(ANIMO) modéliser avec une dynamique continue vers un modèle IMITATOR plutôt hybride(états discrets et temps continu).

Dans le modèle checker UPPAAL, les concepteurs permettent de l’exécution des fonctions au niveau des actions du modèle. Ce qui permet d’évaluer à chaque fois la dynamique et de choisir l’action à effectuer. Ainsi, il est donc possible de convertir les modèles ANIMO vers les UPPAAL.

Nous souhaitons donc partic des modèles UPPAAL pour obtenir des modèles IMITATOR. Ce qui implique des augmentations au niveau de la syntaxe d’IMITATOR. Nous présentons dans la suite quelques possibles modifications à effectuer pour permettre à IMITATOR de

Les possibles modifs:

* Au niveau des déclarations:

< declarations > ::

var < var\_lists >; function <function\_lists>

| function <function\_lists>;var <var\_lists>

| var <var\_lists>

| function <function\_list>

| $\epsilon$

//var comme dans le manuel

function <function\_lists> :: <function\_list>; <function\_lists>

| $\epsilon$

<function\_list> :: //comme fonction ocaml, ou des functions C

* Au niveau des mises à jour

< update > ::

<name> ' = < linear\_expression > | <function\_list>

II. Les potentialités des autres models checkers

1. UPPAAL

* Une riche syntaxe fortement (voir équivalente) inspirée du C qui prend en compte

les opérateurs du C, y compris ceux de post et pré incrémentation.

* Les tableaux, les arrays, les tableaux d’horloges, les canaux.
* Les expressions *forall* et *exists.*
* Les priorités sur les canaux, les processus.
* Les fonctions spécifiées comme en C.

2. Roméo

Je n’ai pas encore trouvé une documentation ou manuel utilisateur.

3. PAT

En cours d’analyse...