Model-Checking temporisé pour l'analyse de composants mémoires
Application à une portion de SPSMALL

E. André, E. Encrenaz, L. Fribourg

Projet ANR VALMEM - 13 mars 2008

Plan

- Model-checking temporisé
- Particularités des modèles issus de composants mémoire
- Démarche propre aux composants mémoires
- Application à SPSMALL

Model-checking temporisé

- ensemble d'automates temporisés : *horloges*
  - ⇝ variables sur $\mathbb{R}$
  - ⇝ invariants et gardes
  - ⇝ transitions d'action : remise à zéro d'ensembles d'horloges
  - ⇝ transitions d'écoulement du temps : progression des valeurs des horloges (avec la même dérivée)
- propriété (sûreté / logique temporisée)
- parcours du produit des automates temporisés
  - ⇝ analyse de graphes temporisés : état = locations + région temporelle

Deux types d'analyse

- Modèle instancié : déterminer si un temps de réponse est correct
- Modèle paramétré : synthétiser des contraintes entre les délais garantissant une plage de fonctionnement correct (i.e. un temps de réponse donné)

Particularités des modèles issus de composants mémoires

- chaque automate a 1 horloge distincte
- chaque automate est déterministe *en action et en temps*
- chaque location est d'une des deux formes suivantes :

  ATTENTE PASSIVE : invariant = TRUE, plusieurs transitions sortantes de la forme : garde = TRUE, synchronisation sur étiquette partagée

  ATTENTE ACTIVE : invariant = $x_i \leq p_i$, une transition sortante de la forme garde = $x_i = p_i$, synchronisation sur étiquette partagée, [et plusieurs transitions sortantes de la forme garde=TRUE, synchronisation sur étiquette partagée]

- chaque transition synchronisée a une unique garde non triviale

Conséquences

Le modèle est *quasi-déterministe* : pour une instanciation ponctuelle des paramètres, seules les transitions concurrentes provenant d'automates différents et franchissables au même instant peuvent provoquer de l'indéterminisme.

Pour une instanciation ponctuelle des paramètres, les sources d'indéterminisme sont très réduites.

Démarche propre aux composants mémoire

- Déterminer une zone *Bad*
- Sélectionner une trace T : un ordre d'occurrence des événements
- Déterminer l'instant d'occurence de chaque événement de T (parcours arrière des automates)
- Synthétiser le système de contraintes C autorisant T
- Vérifier que $Post^*(Init \cap C) \not\ni Bad$

Application à une portion de SPSMALL

- Modèle extrait par LIP6
- Transcription en automates temporisés
- Vérification temporisée
- Synthèse de contraintes autour d'un point d'instanciation

Modèle SPSMALL extrait par LIP6