AltaRica au LaBRI : Travaux récents et projets

A. Griffault, G. Point and A. Vincent

$3^{èmes}$ journées AltaRica, Bordeaux.

27 & 28 novembre 2007

Plan

Plan

Les points communs entre AltaRica et AltaRicaDF

les automates à contrainte

- Un ensemble de variables d'états $\vec{s}$.
- Un ensemble de variables de flux $\vec{f}$.
- Un ensemble d'événements E.
- Un ensemble de transition :

$$G(\vec{s}, \vec{f}) \quad \xrightarrow{e} \quad \vec{s} := \sigma(\vec{s}, \vec{f})$$

$G(\vec{s}, \vec{f})$ est un(e) garde (c-à-d une expression booléenne) et $e \in E$.

Les différences entre AltaRica et AltaRicaDF

AltaRica

- Une assertion $A(\vec{s}, \vec{f})$.
- Des priorités (un ordre partiel) sur E.
- Non déterministe (plusieurs solutions possibles).

AltaRicaDF

- Des flux typés `in, out`.
- Pas de variable hors des feuilles de la hierarchie.
- Un ensemble d'affectations parallèles $\vec{f_o} := \sigma(\vec{s}, \vec{f_i})$.
- Des priorités (un ordre total) sur E.
- Déterministe.

Les différences entre AltaRica et AltaRicaDF

AltaRica : Un interrupteur

```
// AltaRica
node Interrupteur
  state  on : bool;
  flow   f1, f2 : bool;
  event  push;
  trans  true |- push -> on := ~on;
  assert on => (f1=f2)
edon
// Le contexte indique le sens de montage
node Circuit
  sub I : Interrupteur;
  assert f2 => on
edon
```

Les différences entre AltaRica et AltaRicaDF

AltaRicaDF : Un interrupteur

```
// AltaRica DF
// Attention au sens du montage
node InterrupteurDF
  state  on : bool;
  flow   f1 : bool:in, f2 : bool:out;
  event  push;
  trans  true |- push -> on := ~on;
  assert f2 = (f1 & on)
edon
```

Plan

Les projets ESACS et ISAAC : 2001-2006

ESACS : 2001-2003

- Définir une méthodologie pour améliorer les analyses de sûreté pour les systèmes complexes.
- Définir un atelier basé sur les outils supportant la méthodologie.
- Études de cas pour valider l'approche.

ISAAC : 2004-2006

- Dédié aéronautique.
- Suite d'ESACS.

Les projets ESACS et ISAAC : 2001-2006

Les résultats

- Une extension du langage AltaRica (l'anti-diffusion).
- Une variante AltaRica (flux typés, attributs de visibilité "public", "parent" et "private").
- Un compilateur AltaRica→lustre qui respecte la sémantique AltaRica, (graphes isomorphes sauf pour les états initiaux).
- Un compilateur lustre→AltaRica pour un sous ensemble de lustre.

Les projets ESACS et ISAAC : 2001-2006

AltaRica : Une lampe et un fil

```
node Lampe
  state  allume : bool : public;
  init   allume := true;
  event  on, off;
  trans  allume  |- off -> allume := false;
         ~allume |- on  -> allume := true;
edon
node Fil
  state  coupe : bool;      init   coupe := false;
  flow   f1, f2 : bool;
  event  failure, repair;
  trans  ~coupe |- failure -> coupe := true;
         coupe  |- repair  -> coupe := false;
  assert ~coupe => (f1=f2);
edon
```

Les projets ESACS et ISAAC : 2001-2006

AltaRica : Connexité dans un circuit

```
node Circuit
  sub L1, L2, L3, L4 : Lampe;
      FS1, FS2, F14, F23, F34 : Fil;
  assert FS1.f1 & FS2.f1;
         (FS1.f2 = L1.allume) & (L1.allume = F14.f1);
         (FS2.f2 = L2.allume) & (L2.allume = F23.f1);
         (F23.f2 = L3.allume) & (L3.allume = F34.f1);
         (F14.f2 = L4.allume) & (L4.allume = F34.f2);
  sync <FS1.failure, L1.off?, L2.off?, L3.off?, L4.off?>;
       <FS2.failure, L1.off?, L2.off?, L3.off?, L4.off?>;
       <F14.failure, L1.off?, L2.off?, L3.off?, L4.off?>;
       <F23.failure, L1.off?, L2.off?, L3.off?, L4.off?>;
       <F34.failure, L1.off?, L2.off?, L3.off?, L4.off?>;
       <FS1.repair, L1.on?, L2.on?, L3.on?, L4.on?>min;
       <FS2.repair, L1.on?, L2.on?, L3.on?, L4.on?>min;
       <F14.repair, L1.on?, L2.on?, L3.on?, L4.on?>min;
       <F23.repair, L1.on?, L2.on?, L3.on?, L4.on?>min;
```

Plan

L'ACI Persee : 2003-2006

Les objectifs

- Définir de nouvelles techniques pour la vérification de systèmes infinis hétérogènes (piles, files, horloges et compteurs).
- Développer un atelier pour ces techniques basé sur les outils existants TReX, FAST et BLAST.

Un choix important

- Utilisation du formalisme AltaRica (à la place de IF) comme langage de description des systèmes sous réserve d'extensions.

L'ACI Persee : 2003-2006

Les résultats

- Des extensions du langage AltaRica :
  - Les variables structurées.
  - Les tableaux de variables, d'événements et de composants.
  - Les paramètres.
  - Les types de données abstraits.
- Un outil ARsyntax.
- Une nouvelle implémentation des altatools $\rightarrow$ ARC.

L'ACI Persee : 2003-2006

AltaRica : Des nouveaux types

```
Domain
  message = Struct data : integer;
                   parity : bool
            Tcurts;

Sort FIFO;

Sig queue   : FIFO * message -> FIFO;
Sig dequeue : FIFO -> FIFO;
Sig head    : FIFO -> message;
Sig empty   : FIFO -> bool;
```

AltaRica : Une feuille partiellement définie

```
Const N = 4;

node Producer
  flow  m : message;
  event send
  trans true |- send ->;
edon

node Consumer
  state f : FIFO;
  flow  m : message;
  param id : integer;
  event receive
  trans (m.data mod N)=id |- receive -> f := queue(f,m);
edon
```

L'ACI Persee : 2003-2006

AltaRica : Un modèle partiellement défini

```
node Systeme
  state  f : FIFO;
  sub    P : Producer;        C : Consumer[N];
  event  in, out;
  trans  true |- in -> f := queue(f,P.m);
         ~empty(f) |- out -> f := dequeue(f);
  sync   <P.send, in>;
         <out, C[0].receive>;      <out, C[1].receive>;
         <out, C[2].receive>;      <out, C[3].receive>;
  assert empty(f) |
         (C[0].m = head(f) & C[1].m = head(f)&
          C[2].m = head(f) & C[3].m = head(f));
  param_set
    C[0].id:=0, C[1].id:=1, C[2].id:=2, C[3].id:=3;
edon
```

Plan

L'enseignement d'AltaRica

Le cours : Conceptions formelles

- Présentation du langage et des outils.
- Conception à l'aide de vérificateurs.
- Synthèse de contrôleurs.

Les étudiants

- Depuis 2003 : Master 2 informatique, spécialité "Ingénierie des Systèmes Complexes".
- Depuis 2003 : Enseirb 3, spécialité "Génie Logiciel".
- Depuis 2006 : Enseirb 3, spécialité "QFIAB".
- Depuis 2007 : Master 2 informatique au Vietnam, spécialité "Génie Logiciel".

L'enseignement d'AltaRica

Retombées outils : ajouts de fonctionnalités

- Sorties de résultats dans des formats graphiques (dot, gml).
- L'opérateur de projection pour la synthèse de contrôleurs AltaRica.

L'enseignement d'AltaRica

Les stages en entreprise (1/2)

- Airbus : Modélisation de la gouverne → Thèse.
- CEA Saclay : Ingénierie des modèles appliquée à la traduction de langage.
- CERT ONERA : Méthodes formelles pour l'analyse de propriétés d'indépendance en sûreté de fonctionnement → Thèse.
- CERT ONERA : Transformation de modèles AADL en modèles AltaRica.
- Clearsy (3) : AltaRica & B

L'enseignement d'AltaRica

Les stages en entreprise

- Eurogiciel : Vérification et Validation de logiciel de visualisation SMD88-NG de NH90.
- Siemens : Evaluation d'outils de vérification statique.
- Thales (Pessac) : Etude sur les outils de modèles exécutables.
- Thales (Toulouse) : Modélisation comportementale de fonctions avioniques.
- Trusted Logics : Développement d'un outil de cartographie des risques

Plan

Les doctorants

Laurent Sagaspe (ONERA)

- Allocation sûre dans les systèmes aéronautiques, modélisation, vérification et optimisation.
- Soutenance prévue début 2008.

Sophie Humbert (CIFRE Turbomeca)

- Déclinaison d'exigences de sécurité du système vers le logiciel, assistée par des modèles formels.

Hayssam Soueidan (LaBRI)

- BioRica : modèles formels pour la bioInformatique.

Les doctorants

Romain Bernard (CIFRE Airbus)

- Multi systèmes.
- Bisimulation, équivalences observationnelles et équivalences de traces pour calculer les séquences au lieu de les générer.

Fares Chucri (LaBRI)

- Approche CEGAR pour la vérification de modèles AltaRica.
- Abstraction de modèles (fini, infini) pour des propriétés de sûreté et de vivacité.
- Implémentation dans mecV.

Plan

mecV

Projet

- $\mu - calcul$.
- Algorithmes efficaces pour la synthèse de contrôleurs.
- Problème ouvert : algorithme polynomial pour les jeux de parité ?

Plan

L'ANR/RNTL SPaCIFY

Les partenaires

- CNES.
- Alcatel, Anyware, EADS Astrium, tni-software.
- Enst Bretagne, IRISA, IRIT, LaBRI.

Les objectifs

- Concevoir un atelier de développement et maintenance des logiciels de vol centraux pour astronefs.
- Généricité pour adaptation à d'autres métiers similaires.
- Logiciels libre.

L'ANR/RNTL SPaCIFY

Les moyens et techniques utilisés

- Définition d'un langage formel.
- Vérifications formelles :
  - Typage avancé.
  - Vérification de modèles.

Au LaBRI

- Réalisation d'un compilateur du langage (variante de signal) vers le langage AltaRica.
- Utilisation si possible des vérificateurs ARC et mecV.

Plan

Le projet AltaTina

Les objectifs

Techniques avancées d'analyses : Ordres partiels et dépliages, CEGAR et diagrammes de classes, Synthèse de contrôleurs, Raffinement, simulations et équivalences.

Études de cas et benchmarks : Comparaisons de performances des outils et approches sur des cas d'études communs.

Les outils du futur : Professionnalisation des outils et meilleure utilisation des évolutions technologiques.

- Impact des technologies multi-cœurs, des processeurs 64 bits, des grilles de calcul sur l'algorithmique de la vérification.
- Interopérabilité (sauvegarde de session en XML, traductions entre formalismes, intégration dans TOPCASED).

Le projet AltaTina

Etat du projet

Projet Inter-Carnot (Bernard Berthomieu et François Vernadat du LAAS) non accepté (depuis la semaine dernière), mais nouvelle soumission prévue.