

# **CRISS : Contrôle de Ressources et d'Interférence dans les Systèmes Synchrones**

Projet ACI Sécurité Informatique 2003-2006

Coordinateur : Roberto AMADIO, Université Aix-Marseille

<http://www.cmi.univ-mrs.fr/~amadio/Criss/criss.html>

## Participants

Site	Responsable
INRIA-Sophia	G. BOUDOL
LIF, Marseille	S. COUPET-GRIMAL
LIPN, Villetaneuse	P. BAILLOT
LORIA, Nancy	J.-Y. MARION

# Survol

**Motivations :** problèmes de sécurité soulevées par le *code mobile*.

Contextes applicatifs : réseaux programmables, jeux en réseau, cartes à puces ...

## Vérification de deux classes de propriétés :

- *Contrôle de ressources* (e.g., éviter le déni de service).  
**NB** Les bornes sur les ressources sont des fonctions pas des constantes.
- *Contrôle du flux d'information* (e.g., éviter la fuite de données confidentielles).

## Technique

- Vérification *statique* autant que possible (fiabilité et efficacité).
- Ingrédients :
  - *Réécriture* (interprétations polynômiales, RPO),
  - *Typage* (logique linéaire, types et effets, ...)

## Modèle de calcul :

- *GALS* : Globalement *asynchrone*, localement *synchrone*.
- Focalisation sur les propriétés d'*un noeud* du réseaux.
- Threads *synchrones* et *coopératives*.

**NB** Cf. projet ALEDICS...

## Expérimentation :

- *Machine virtuelle* : formalisation et implémentation.
- *Génération de certificats* pour le code octet.
- *Validation en Coq* des algorithmes de certification.

# Articulation du projet

1. Contrôle de ressources pour programmes fonctionnels du premier ordre.
2. Contrôle de ressources à l'ordre supérieur et logique linéaire.
3. Modèle de threads synchrones et coopératives.
4. Non-interférence pour systèmes synchrones.
5. Contrôle de ressources pour systèmes synchrones.
6. Génération de certificats et validation de la machine virtuelle.

**NB** 1 thèse en cours/thème.

## Contrôle de ressources pour programmes fonctionnels du premier ordre

**Approche** Contrôle de ressources = terminaison+borne sur la taille des données (cf. théorème de Cobham).

### Contribution

- *Quasi-interprétation* : méthode modulaire pour déterminer la borne.
- *Petits polynômes* souvent suffisants.
- *Combinaison avec RPO* produit des bornes polynômiales.

**Expérimentation** : Mise en oeuvre heuristique pour la synthèse d'interprétations max-plus.

**Objectif** Extension à d'autres méthodes de terminaison (*dependency pair, size-change*).

## Références

- R.M. Amadio. *Synthesis of max-plus quasi-interpretations*. Fundamenta Informaticae, 2004 (à paraître).
- G. Bonfante, J.-Y. Marion, J.-Y. Moyen. *Quasi-interpretations*. TCS (en révision). 2004.

**Thèse en cours** : R. PECUCHET, *Automates, logique et contrôle de ressources* (2004–), MENRT Nancy.

## Contrôle de ressources à l'ordre supérieur et logique linéaire.

**Approche** La *logique linéaire* est le bon cadre pour contrôler la complexité de fonctions d'*ordre-supérieur*.

### Contributions

- DLAL (*dual light affine logic*) (variation sur LAL, *light affine logic*).
- DLAL assure une borne polynômiale en temps par rapport à une variété de stratégies de réduction (par opposition à LAL).
- L'inférence de type pour DLAL semble plus simple (comparable à celle d'EAL, *elementary affine logic*).

**Objectif** Cacher la complexité du système sous une interface à la ML.

## Références

- P. Baillot, V. Mogbil. *Soft lambda-calculus : a language for polynomial time computation*. FOSSACS 2004.
- P. Baillot, K. Terui. *Light types for polynomial time computation in lambda-calculus*. LICS 2004.

**Thèse en cours** : D. DE CARVALHO, *Logique linéaire et complexité* (2003–), ACI-Interfaces Math. Villetaneuse.

## Modèle de threads synchrones et coopératives

**Approche** Le code mobile doit pouvoir réagir à l'absence d'un événement par le biais d'une hypothèse de synchronie (locale) (cf. 'programmation réactive' de F. BOUSSINOT).

**Contributions** Définition d'un noyau de langage (ULM : ultra-leger motorisé) :  $\lambda$ -calcul plus primitives pour la synchronisation (locale) et la migration (globale).

**Expérimentation** Prototype machine ULM sous SCHEME.

**Objectif** Mise en oeuvre et expérimentation du modèle dans un environnement de programmation réaliste (SCHEME).

## Référence

- G. Boudol. *ULM, a core programming model for global computing*. ESOP04.

**Thèse en cours** S. EPARDAUD, *Mise en oeuvre d'un langage fonctionnel réactif et mobile* (2004–), MENRT Sophia.

## Non-interférence pour systèmes synchrones

**Approche** Assurer par *typage* que des résultats publiques ne dépendent pas de données privées (cf. Volpano-Smith).

### Contribution

- Formulation de la non-interférence dans le cadre d'une sémantique du *parallélisme* (bisimulation, traces, ...)
- Raffinement du système de types. Si  $\vdash S : (\tau, \sigma)$  alors :
  1.  $\tau$  est une *borne inférieure* au niveau des variables affectées dans  $S$ .
  2.  $\sigma$  est une *borne supérieure* au niveau des gardes dans  $S$ .

**Objectif** Intégrer la notion de *déclassification* afin d'augmenter la flexibilité du système.

## Références

- G. Boudol, I. Castellani, A. Matos. *Typing non-interference for reactive programs*. Foundations of Computer Security Workshop 2004.

**Thèse en cours** : A. MATOS, *Non-interférence pour programmes synchrones* (2002–), EGIDE Sophia.

## Contrôle de ressources pour systèmes synchrones

**Approche** Extension des méthodes développées dans le cadre fonctionnel du premier ordre.

**Contribution** Analyse de flot *modulaire* qui permet d'assurer :

1. la *terminaison* de chaque instant et
2. des *bornes dans un instant*.

**Objectif** Améliorer la qualité des bornes de complexité.

## Références

- R.M. Amadio, S. Dal Zilio. *Resource Control for Synchronous Cooperative Threads*. CONCUR 2004.

**Thèse en cours :** F. DABROWSKI, *Contrôle de ressources pour systèmes synchrones* (2003–), ACI-SI, Sophia-Marseille.

## Génération de certificats et validation de la machine virtuelle

**Approche** TAL : code octet avec annotations.

**Contribution** Analyse de flot qui permet de vérifier les certificats pour la taille et la terminaison au niveau du code octet.

**Expérimentation** Mise en oeuvre de la machine virtuelle et de l'analyse de flot. Validation en COQ.

**Objectif** Généralisation de l'analyse à un code octet 'non-fonctionnel'.

**Référence** R.M. Amadio, S. Coupet-Grimal, S. Dal Zilio, L. Jakubiec. *A functional scenario for bytecode verification of resource bounds*. CSL 2004.

**Thèse en cours** W. DELOBEL, *Certification de machines virtuelles en COQ*, MENRT Marseille.