

# Temps-réel en multi-cœurs : problème de la contention mémoire

Louisa BESSAD & Roberto MEDINA

April 25, 2014

# Chapter 1

## Problème de concurrence d'accès

### 1.1 Introduction

- Description du sujet
- Problématique: gestion de l'accès concurrent aux caches et à la mémoire entre une tâche temps-réel et des tâches attaquant en sachant qu'on ne peut faire que des mesures globales dans la mémoire et pas pour chaque coeur
- Objectif: proposer une solution en mode user en utilisant une librairie particulière (PAPI) pour les mesures
- Description de l'architecture: intel Core I3-330M Processor(2.13GHz, 3MB L3 Cache), approximation des accès mémoire par les MISS sur le L3 + schéma de la différence entre les machines qui ont fait les tests.

### 1.2 Les différentes tâches (attaquantes, temps réel)

- Description des 2 types de tâches et leur impact sur la mémoire:
  - on a une tâche temps-réelle non périodique et sans contrainte temporelle, on doit uniquement être sûre qu'elle se termine en un temps fini
  - les tâches attaquant doivent effectuer un maximum d'accès mémoire afin de saturer le bus mémoire et les caches partagés entre les différents coeurs
- Evolution des choix face au prefetching, régime transitoire et permanent, les printf

### 1.3 Comment mesurer?

- Utilisation de PAPI (fonction pour les mesures, évènements initialisés et utilisés)
- Mesures du nombre de MISS sur le cache partagé L3 pour approximer le nombre d'accès mémoire puisqu'on ne peut mesurer les accès mémoire sur notre machine
- Fonctionnement du wrapper

### 1.4 Résultats et analyse des courbes

- Evolution des résultats avec les évolutions de code, en fonction du gouverneur du CPU (si on teste sur plusieurs machines), 2 attaquants au lieu de 3 (garder un coeur pour l'OS), désactivation de l'hyperviseur (pas d'autres scheduler sur les coeurs utilisés)
- Différences des résultats avec pré-fetching
- Résultats sur le temps d'exécution ou le nombre d'octets lu pendant la période de mesures par chaque coeur (C1 puis C1 & 2 puis C1,2 & 3)

### 1.5 Conclusion

- conclure sur le problème d'accès mémoire à une tâche temps-réel face à des attaquants
- ouvrir sur une solution en mode user qui sera la sous-réservation de BP mémoire

## Chapter 2

# Problème de la sous-réservation de BP mémoire:

### 2.1 Introduction

- Explication du problème de sous-réservation de BP mémoire
- Les problèmes que cette solution peut générer: comment gérer les accès mémoire tout en gardant une certaine concurrence
- Une solution proposée: Création d'un hyperviseur envoyant des signaux aux différentes tâches lorsqu'elles ont consommées toute la BP accordé et que la tâche temps-réel n'a pas encore utilisé sa BP et remise à zéros des différents compteurs de BP dans ce cas

### 2.2 L'hyperviseur, la tâche temps-réel et les tâches attaquantes

- Evolution du fonctionnement de la tâche temps-réel et des tâches attaquantes face à l'insertion de l'envoi de signaux
- Description du fonctionnement de l'hyperviseur (mesure de la consommation de BP, gestion des compteurs mesurant cette consommation et envoi de signaux aux différentes tâches), problèmes rencontrés ou pas
- Comment utiliser PAPI dans cette situation (fonctionnalités rajoutées)

## **2.3 Mesures, résultats et analyse**

- Changement de la méthode de mesure
- analyse des courbes obtenues avec la solution proposée

## **2.4 Conclusion**

- La solution résout-elle le problème de contention mémoire, si oui pourquoi?

## Chapter 3

# Conclusion face au sujet proposé:

- Avantages et inconvénients de l'utilisation d'un hyperviseur et comparaison avec la première partie
- Ouverture: Existe-t-il une solution plus optimale?