Traçage de systèmes Linux multi-coeurs en temps réel

Raphaël Beamonte

Laboratoire DORSAL
Département de génie informatique

POLYTECHNIQUE MONTRÉAL
AFFILIÉE À L'UNIVERSITÉ DE MONTRÉAL

Hiver 2012

Outils pour la vérification de la capacité temps réel d'un système

- **La suite rt-tests**
  - **Cyclictest**
    - **génère un processus RT périodique dans l'userspace**
    - **lui attribue une priorité et une période**
    - **vérifie que la période est respectée**
  - **hwlatdetect (module noyau *hwlat_detector*)**
    - **utilise stop_machine() pour désactiver tout autre processus, toute interruption et s'attribuer tous les processeurs**
    - **fait un nombre de cycles choisi pour en calculer la latence**

Outils pour la vérification de la capacité temps réel d'un système

- **Coté préemption ?**
  - **preempt-test**
    - **exécuter un processus de priorité p**
    - **exécuter n processus fils du précédents ayant pour priorité 1..p dans l'ordre croissant de priorité**
    - **vérifier que chaque processus préempte le précédent**
    - **calculer la latence engendrée par la préemption réalisée**

Linux temps réel

xaf@station11-64:~/git/rt-tests$ sudo ./cyclictest -S -p 99
/dev/cpu_dma_latency set to 0us
policy: fifo: loadavg: 0.77 0.39 0.15 1/286 3285

T: 0 (3282) P:99 I:1000 C: 10386 Min: 1 Act: 2 Avg: 2 Max: 16
T: 1 (3283) P:99 I:1500 C: 6923 Min: 1 Act: 2 Avg: 2 Max: 15
T: 2 (3284) P:99 I:2000 C: 5193 Min: 1 Act: 3 Avg: 2 Max: 25
T: 3 (3285) P:99 I:2500 C: 4154 Min: 1 Act: 2 Avg: 2 Max: 753

xaf@station11-64:~/git/rt-tests$ uname -a
Linux station11-64 3.2.0-3-amd64 #1 SMP Thu Jun 28 09:07:26 UTC 2012 x86_64 GNU/Linux

Linux temps réel

- **Patch PREEMPT_RT**
  - **CONFIG_PREEMPT / CONFIG_PREEMPT_RCU / …**
  - **Intégré petit à petit dans le noyau mainline**
  - **Améliore les capacités de préemption du système, évite certaines inversions de priorités**

Linux temps réel

xaf@station11-64:~/git/rt-tests$ sudo ./cyclictest -S -p 99
/dev/cpu_dma_latency set to 0us
policy: fifo: loadavg: 0.33 0.33 0.14 1/318 3397

T: 0 (3394) P:99 I:1000 C: 10351 Min: 1 Act: 3 Avg: 2 Max: 7
T: 1 (3395) P:99 I:1500 C: 6900 Min: 1 Act: 3 Avg: 2 Max: 7
T: 2 (3396) P:99 I:2000 C: 5175 Min: 1 Act: 3 Avg: 3 Max: 7
T: 3 (3397) P:99 I:2500 C: 4140 Min: 1 Act: 2 Avg: 2 Max: 9

xaf@station11-64:~/git/rt-tests$ uname -a
Linux station11-64 3.2.0-3-rt-amd64 #1 SMP PREEMPT RT Mon Jul 23 03:37:45 UTC 2012 x86_64 GNU/Linux

Du traçage en temps réel

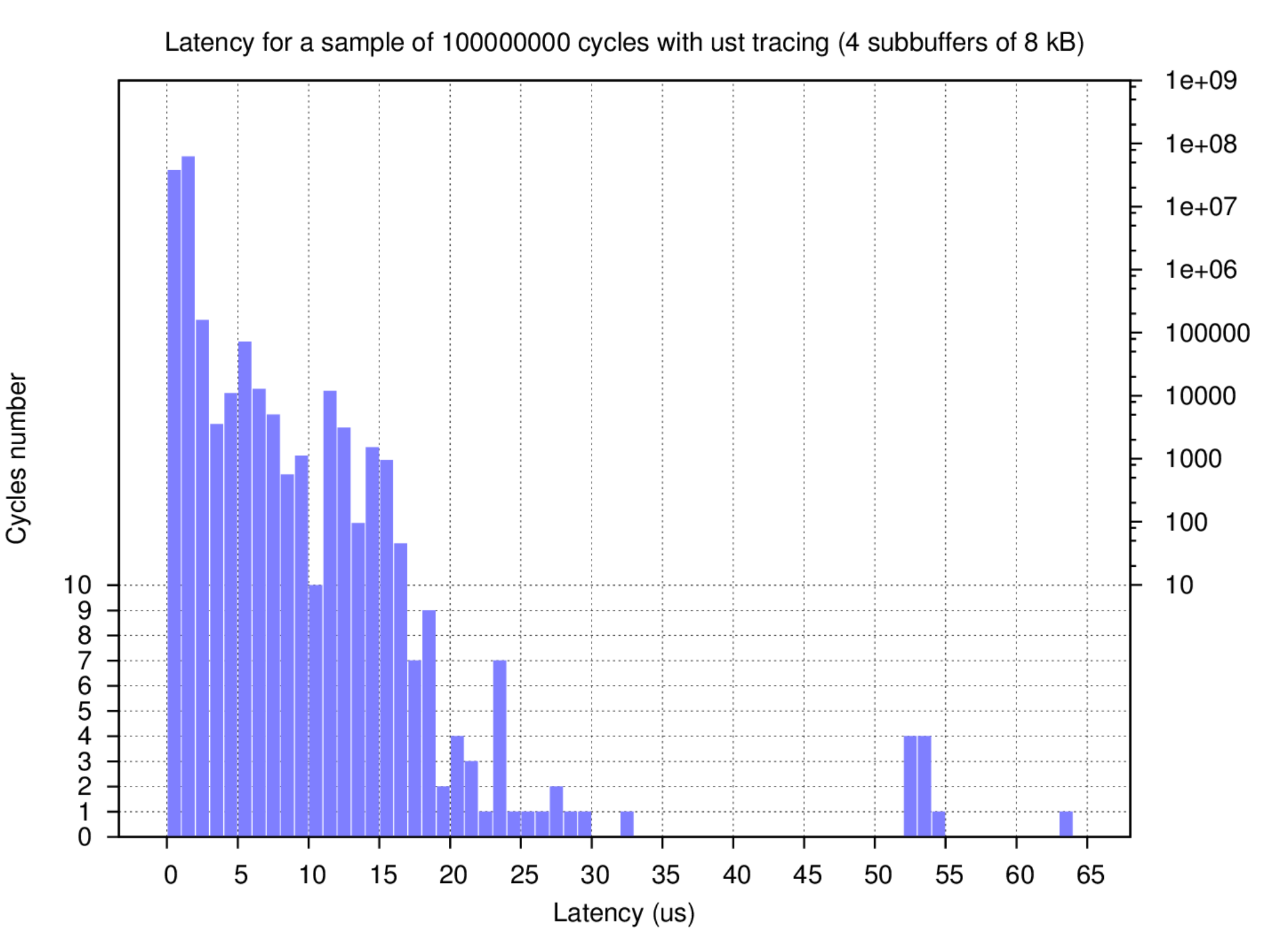
- **Pourquoi tracer ?**
  - **Analyse en profondeur d'une anomalie temps réel**
  - **Identification des sources internes et/ou externes de cette anomalie**

- **Problème**
  - **Temps réel strict = déterminismes forts**
  - **Traçage = ajout de latences non déterministes**
  - **Ajout de latences = noyer l'anomalie ou la faire disparaître**

Projet

- **But : rendre LTTng apte à tracer des applications temps réel strict**

- **Première étape :**
  - **Identifier une situation adéquate : une boucle d'exécution s'exécutant en haute priorité avec très peu d'opérations sur un processeur isolé**
  - **Appliquer du traçage kernel, userspace, puis les deux**
  - **Analyser les résultats**


Latency for a sample of 100000000 cycles with ust tracing (4 subbuffers of 8 kB)
Cycles number
Latency (us)
0
1
2
3
4
5
6
7
8
9
10
0
5
10
15
20
25
30
35
40
45
50
55
60
65
10
100
1000
10000
100000
1e+06
1e+07
1e+08
1e+09

Travaux en cours et futurs

- **LTTng :**
  - **UST provoque un échange entre l'application et le démon LTTng : limiter cet échange aux buffers**
  - **Selon les résultats :**
    - **Trouver un moyen de les améliorer encore**
    - **Appliquer la solution obtenue**
- **Outils pour l'analyse des systèmes RT :**
  - **Développer un outils complet intégrant les traceurs UST et kernel de LTTng comme outil complémentaire à la suite rt-tests**