Program Instrumentation with QEMU

RCFC'4
2011/12/04

christophe.guillon@st.com
cedric.vincent@st.com

Outline

Context

Motivations

Program instrumentation

Dynamic binary translation

Execution time and translation time instrumentation

Emulation and instrumentation overhead

Contributions and future work

Context

QEMU, a versatile opensource tool used for:

Platform emulation	(Google SDK,…)
Devices emulation	(VirtualBox,…)
Program emulation	(Scratchbox,…)

Our context:

Emulation of Linux programs (ARM, SH4, ST200, x86)

Our focus:

TCG (Tiny Code Generator), the QEMU compiler

Motivations

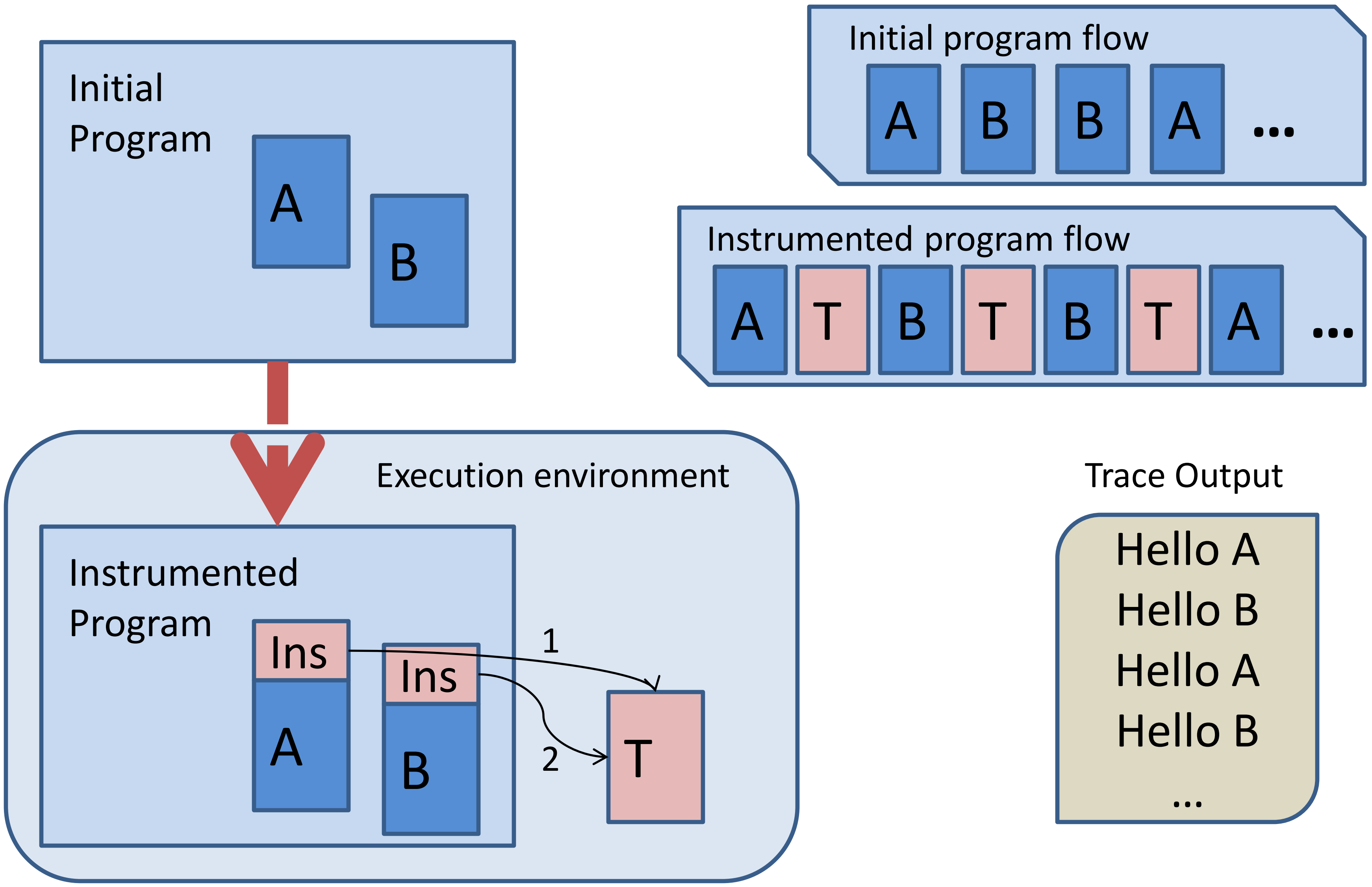
Initial motivations:

- Program performance analysis
  - *cycles count, profiles, call graphs, …*
- Profile driven compiler optimizations
  - *I-cache placement, edge profiling, data dep. estimation, …*

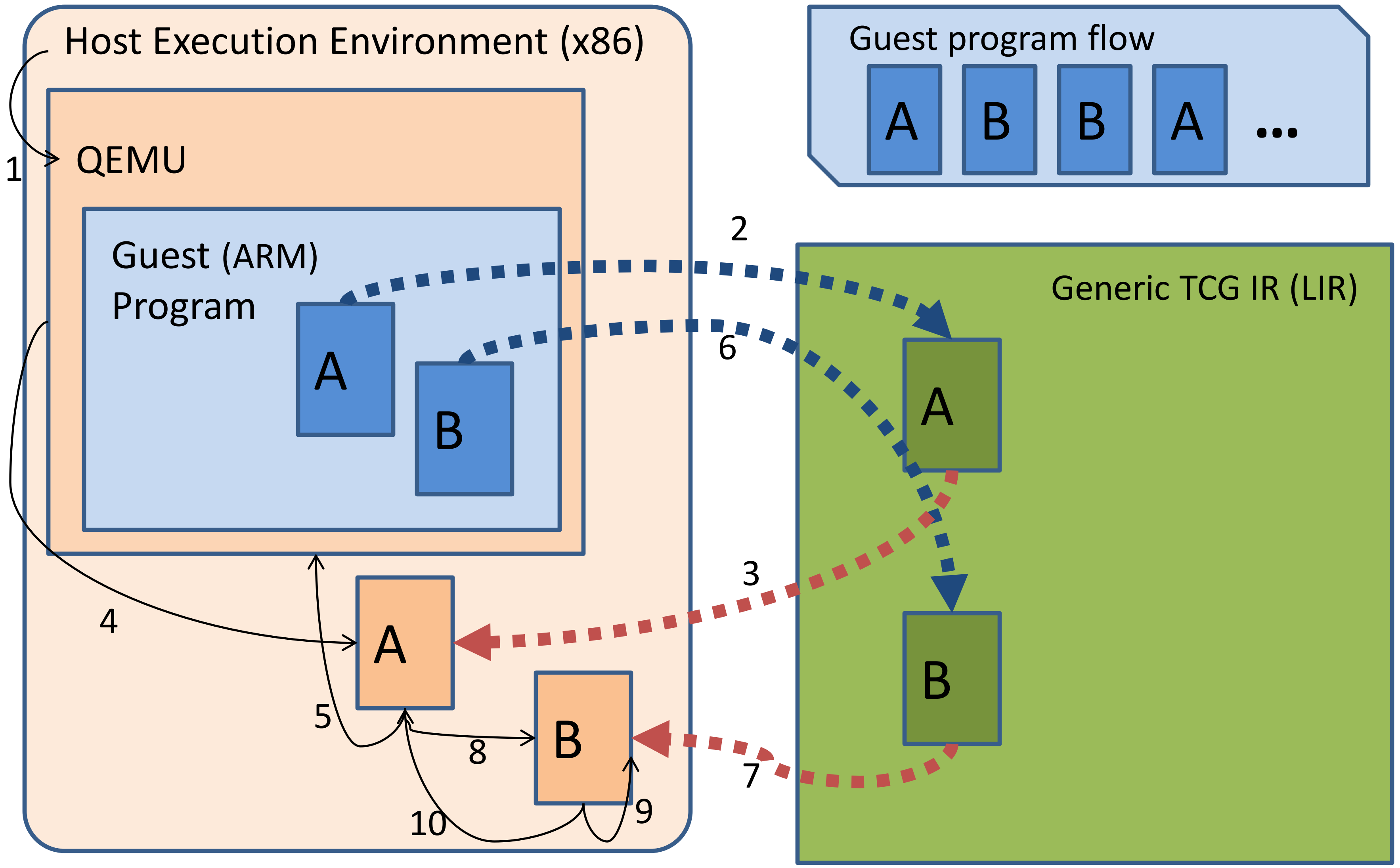
Other usages:

- Program debugging
  - *call traces, syscall traces, memory checks, …*
- Processor architecture analysis
  - *instructions usage, cache behavior, …*

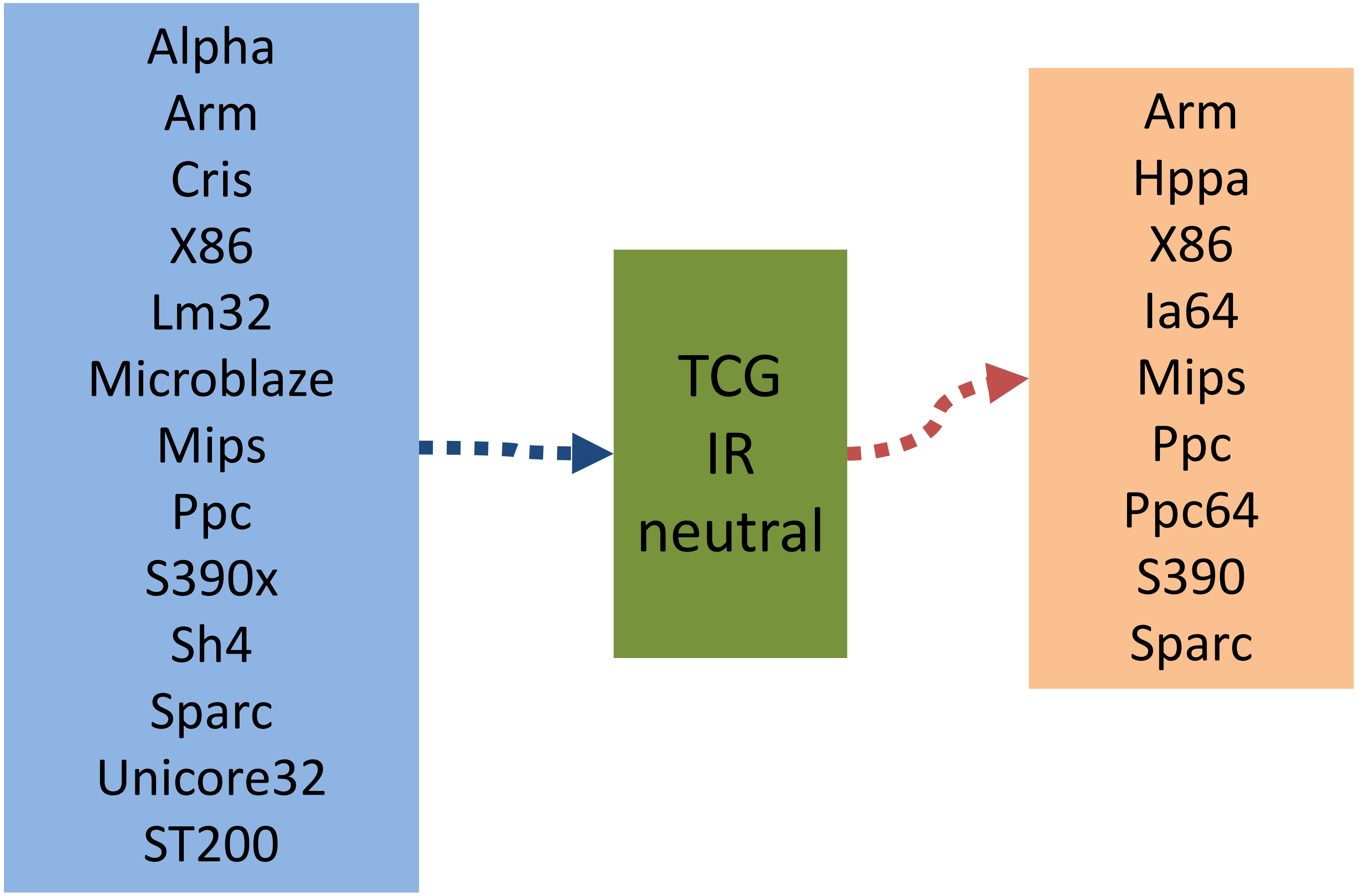
Program Instrumentation



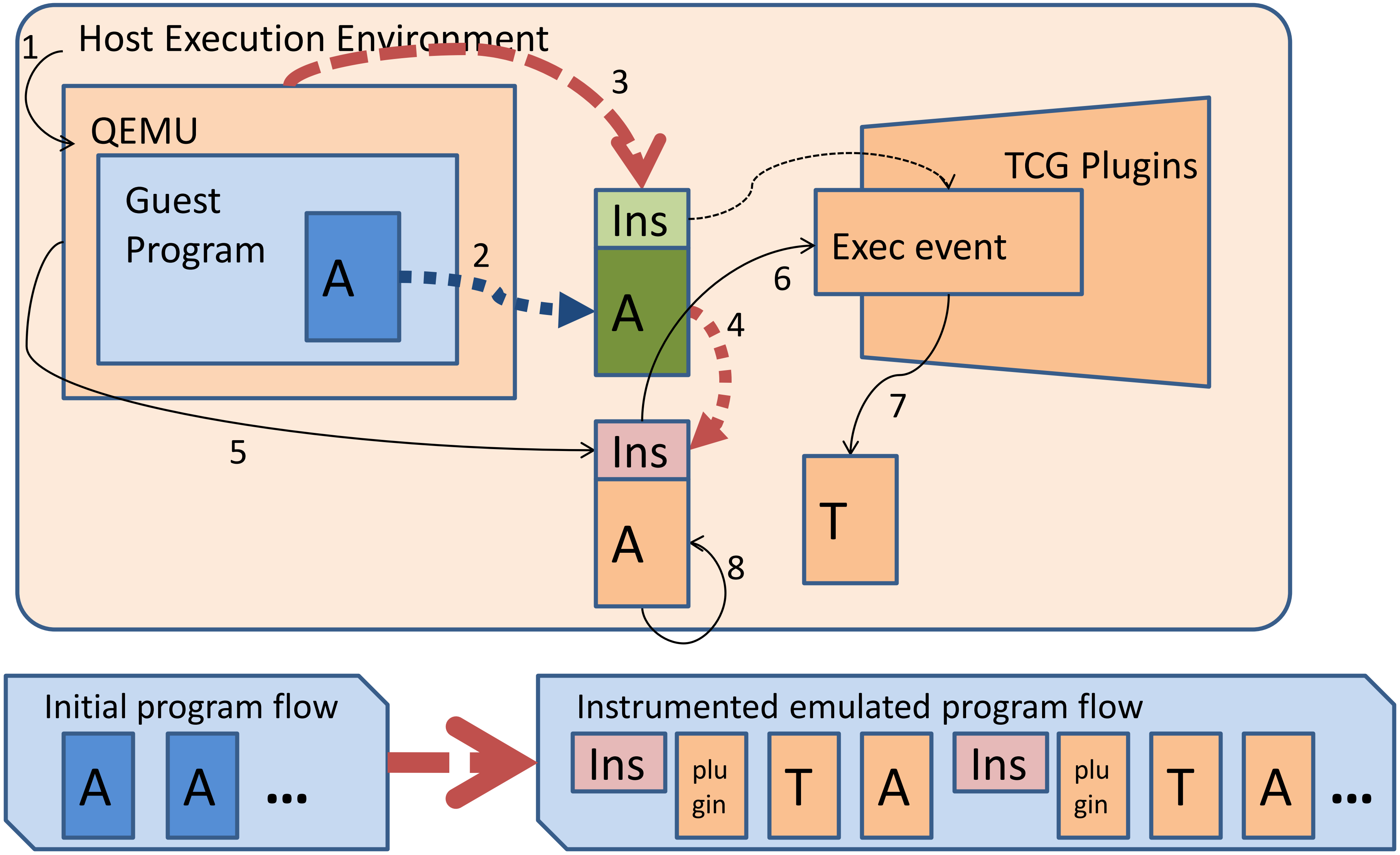
Binary Translation (i.e. QEMU ARM -> x86)



QEMU guests & hosts



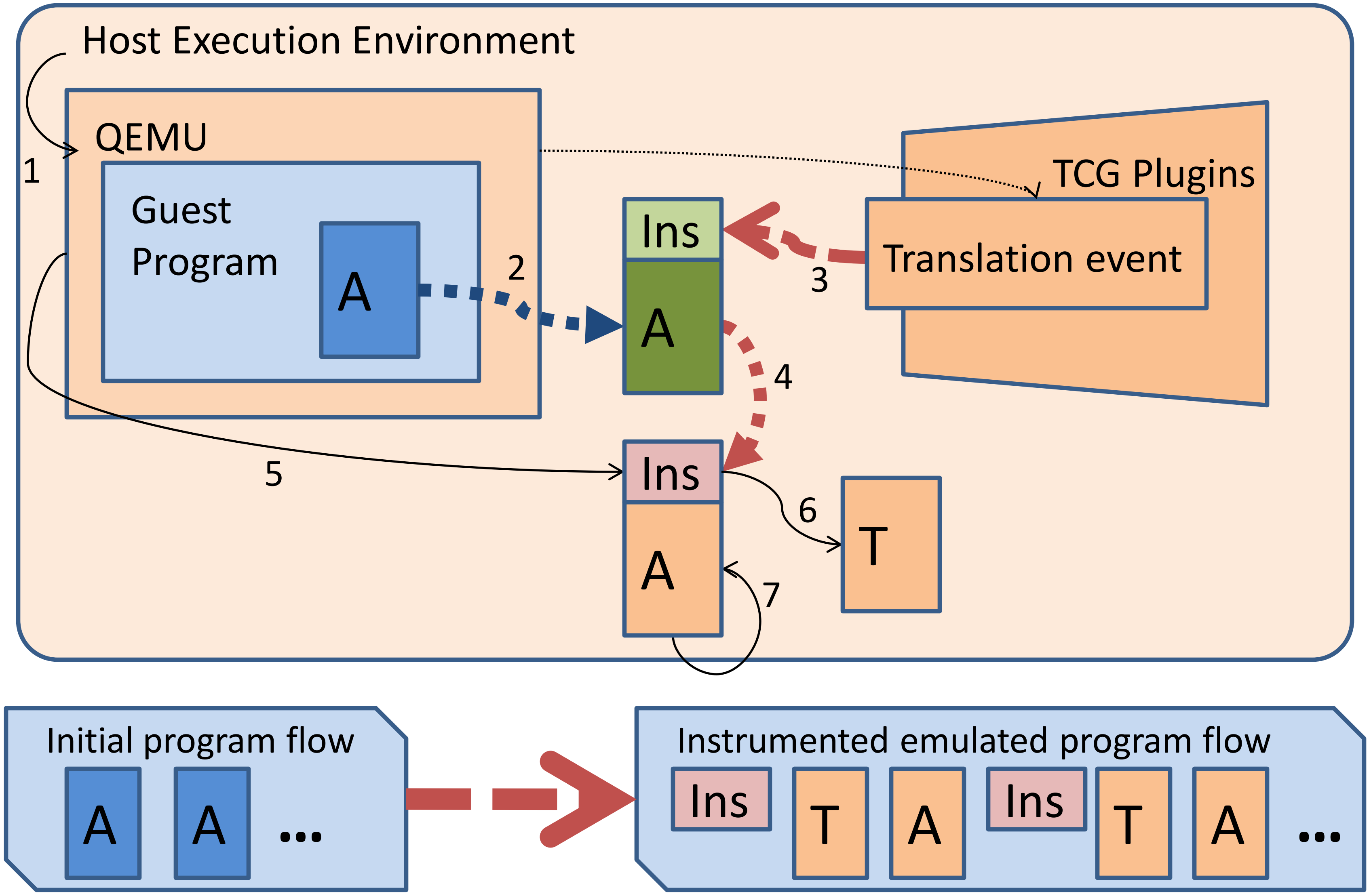
Execution Time Instrumentation



Instruction Count Plugin

```
/* A simple plugin for counting executed guest instructions.
 * Usage : cc shared -o icount.so icount.c
 * qemu-i386 -tcg-plugin ./icount.so the_program
 */
#include <stdint.h>
#include <stdio.h>
#include « qemu_tpi.h »

uint64_t total_icount;

/* Instruction count updated at each block execution. */
void qemu_tpi_block_execution_event(qemu_tpi_t *tpi) {
   total_icount += TPI_tb_icount(tpi);
}
void qemu_tpi_fini(qemu_tpi_t *tpi) {
   printf("Instructions: %"PRIu64"\n", total_icount);
}
```

Translation Time Instrumentation



Inlined Instruction Count Plugin

```
/* The translation event based code for the icount.c plugin. */

void qemu_tpi_block_translation_event(qemu_tpi_t *tpi) {
   /* Instruction count update inlined at each block translation.
    * Code is generated into the TCG IR buffer directly. */
   TPIv_ptr ptr = TPI_const_ptr(tpi, &total_icount);
   TPIv_i64 total = TPI_temp_new_i64(tpi);
   TPI_gen_ld_i64(tpi, total, ptr, 0);
   TPIv_i64 icount = TPI_const_i64(tpi, TPI_tb_icount(tpi));
   TPI_gen_add_i64(tpi, total, total, icount);
   TPI_temp_free_i64(tpi, icount);
   TPI_gen_st_i64(tpi, total, ptr, 0);
   TPI_temp_free_i64(tpi, total);
   TPI_temp_free_ptr(tpi, ptr);
}
```

TCG Plugins Command Line

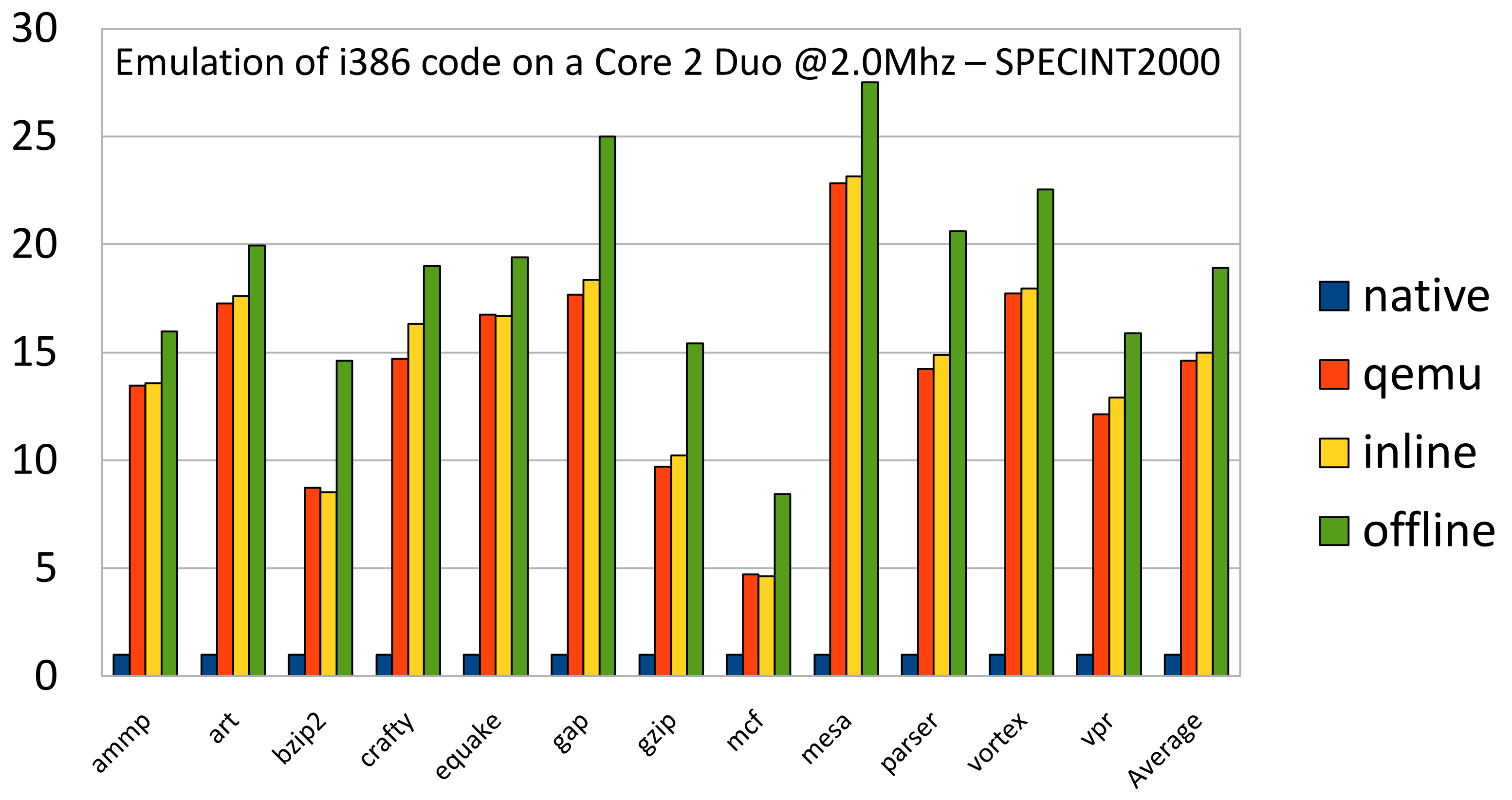
```
$ qemu-i386 –tcg-plugin icount.so sha1-i386.exe
SHA1=15dd99a1991e0b3826fede3deffc1feba42278e6
Instructions: 107270808

$ proot –Q qemu-sh4 -tcg-plugin icount.so /root-sh4/ sha1-sh4.exe
SHA1=15dd99a1991e0b3826fede3deffc1feba42278e6
Instructions: 166854038

$ proot –Q qemu-arm -tcg-plugin icount.so /root-arm/ sha1-arm.exe
SHA1=15dd99a1991e0b3826fede3deffc1feba42278e6
Instructions: 95419722
```

proot: a sandboxing tool developed at STM

Profile Plugin Example

```
$ qemu-i386 –tcg-plugin profile.so sha1-386.exe
SHA1=15dd99a1991e0b3826fede3deffc1feba42278e6
Instrs          bytes           blocks          symbol
106497664       364229691       1792028         SHA1Transform
571297          1815640         133175          SHA1Update
961             3399            168             SHA1Final
23590           85199           6141            main
18              40              3               __libc_csu_init
16              43              5               .init
2093            12552           2087            .plt
55              146             10              .text
12              28              3               .fini
175102          504418          50840           unknown/libc.so.6
```

Instrumentation Overhead



Emulation cost is x15 (qemu)
Instrumentation overhead is 30% (offline) for a I-count plugin
Overhead reduced to 3% with translation time interface (inline)

Contributions

Plugin interface for program instrumentation with QEMU

Per code block and per instruction instrumentation

Execution and translation time events

Access to TCG code generator API

Program and libraries symbol table access

Several plugins

- *I/D-Cache, I-Count, Profile, PC-sample, IO-mem*

Companion tool

- *PRoot sandboxing tool for easy use in Linux user-mode emulation*

Future work

Push to QEMU mainstream

Easier access to debug information

Memory checker plugin

Delinquant load detection (compiler feedback)

Gcov like edge-profile (compiler feedback)

Other compilation oriented usage…