Silent stores optimization: user guide
(version 1.0)

August 2017

Authors: Rabab Bouziane, Erven Rohou and Abdoulaye Gamatie

Contents

1

About this guide

This document provides all the steps for applying the silent stores optimization, developed in the framework of the French ANR CONTINUUM project (http://www.lirmm.fr/continuum-project). It is assumed that the user already installed the corresponding software.

2

How to compile a program and apply the optimisation

Before Compiling, you have to prepare the compilation environment, and you must be in the root of your downloaded directory. Start by running the *insertInclude* script as follows:

```
$ ./insertInclude.sh
```

In the context of this project, we mainly focused on ARM architecture. The following steps are the cross-compilation commands for ARM. If you compile for x86, do not use the option *--sysroot*. The compilation process is composed of two steps: profiling and transforming. Please note that while compiling, three files will be created (counter1, counter2 and counter3). Do not delete them before finishing the compilation. Moreover, you have to specify an environment variable called *CPT* where the counters files will be generated.

```
$ export CPT=/pathToCompilationDirectory
```

To compile a program *A*, you have to first compile it for the profiling process.

2.1 Profiling

```
$ arm-linux-gnueabi-gcc -c ss.c
$ echo 0 >$CPT/counter1
$ echo 0 >$CPT/counter2
$ echo 0 >$CPT/counter3
$ clang --sysroot=/usr -I/usr/arm-linux-gnueabi/include -target armv7-linux-gnueabi -c -emit-llvm -o A.bc A.c
$ opt -load=/PATH/libSkeletonPass.so -SSDetect -o A_opt.bc A.bc
$ llc -mtriple=arm-linux-gnueabi A_opt.bc
$ clang –sysroot=/usr -I/usr/arm-linux-gnueabi/include -target armv7-linux-gnueabi A_opt.s -o A /PathTo/ss.o
```

After executing these commands, please do run your program with an input of your choice. The result will be the profiling of the program with the chosen input. A file will

be generated called "ss_stores2.data". Note that this file contains all stores that are at least silent one time. If you wish to select the stores that are silent above a certain percentage, you can filter this file.

2.2 Transforming

```
$ clang --sysroot=/usr -I/usr/arm-linux-gnueabi/include -target armv7-linux-gnueabi -c -emit-llvm -o A.bc A.c
$ opt -load=/PATH/libSkeletonPass.so -PrePass -o A_opt.bc A.bc
$ opt -load=/PATH/libSkeletonPass.so -SSOpt -o A_opt2.bc A_opt.bc
$ llc -mtriple=arm-linux-gnueabi A_opt2.bc
$ clang --sysroot=/usr -I/usr/arm-linux-gnueabi/include -target armv7-linux-gnueabi A_opt2.s -o A
```

The above commands compile a C program. Note that if you want to compile a C++ program, you have to use **clang++** or if you want to compile a Fortran program, you have to use the **flang**.