微处理器热特性仿真的向导软件方案

冯凯 2023 年 11 月 8 日

1 项目介绍

本项目为已有的微处理器热特性仿真流程制作向导软件。

目前，我们已有的微处理器热特性仿真流程过程繁杂，没有操作界面，使用起来颇为不便。例如，完成某些操作时往往需要调用多个不同位置的项目文件，其输入和输出都需要手动在代码中更改等等。如果以后有新人接手该仿真流程，或者想要在此基础上拓展，将十分困难。因此，亟需一个能够简化仿真流程，并为用户提供一个友好的操作界面的向导软件。

实现一个向导软件，既需要制作用户友好的操作界面，也需要封装已有的仿真流程。黄同学将负责制作操作界面，操作界面引导用户一步步输入信息，并调用项目文件完成仿真。我将负责封装项目文件，尽量简化并使其易于调用。

本项目将一个完整的微处理器仿真流程拆解为三个步骤。第一步，引导用户输入微处理器的信息，调用性能模拟模块，仿真得到微处理器的性能数据；第二步，根据微处理器的性能数据，仿真得到功耗数据。第三步，引导用户输入微处理器中热传感器的位置，实时用图片反映热传感器的位置，并仿真得到温度数据。其中，第一步最为简单，本项目将优先实现第一步，看看效果。

第 2 节将介绍项目的准备工作，包括已有仿真流程的运行环境和项目文件；第 3 节将详细说明如何实现前言之“第一步”，即引导用户输入微处理器的信息并调用性能模拟模块。

2 准备工作

2.1 项目环境

经测试，已有的仿真流程能够在以下环境运行。我不确定其在其他环境下的表现，这里给出的信息仅供参考。

操作系统基本信息：

Linux ubuntu 4.15.0-142-generic #146~16.04.1-Ubuntu SMP Tue Apr 13 09:27:15 UTC 2021 x86_64 x86_64 x86_64 GNU/Linux

Linux 发行版详细信息：

Distributor ID: Ubuntu

Description: Ubuntu 16.04.7 LTS

Release: 16.04

Codename: xenial

CPU 相关信息：

Architecture: x86_64

CPU op-mode(s): 32-bit, 64-bit

Byte Order: Little Endian

CPU(s): 16

On-line CPU(s) list: 0-15

Thread(s) per core: 1

Core(s) per socket: 1

Socket(s): 16

NUMA node(s): 1

Vendor ID: GenuineIntel

CPU family: 6

Model: 165

Model name: Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz

Stepping: 5

CPU MHz: 2904.000

BogoMIPS: 5808.00

Hypervisor vendor: VMware

Virtualization type: full

L1d cache: 32K

L1i cache: 32K

L2 cache: 256K

L3 cache: 16384K

NUMA node0 CPU(s):0-15

2.2 项目文件

项目文件可通过我的 jbox 网盘下载：

https://jbox.sjtu.edu.cn/l/W1Tr2A

解压。如果你不知道如何解压，可以使用以下命令：

tar zxvf {要解压的文件}

压缩包内含有 4 个文件夹：

- full_system_images：仿真过程中用到的二进制文件和镜像文件
- TR-09-32-parsec-2.1-alpha-files：仿真脚本生成包
- gem5：微处理器性能仿真模块
- gem5_output：性能数据

我将这些文件夹所在的目录称为“实验目录”（下同）。实验目录下的文件都是已经封装好的可以直接调用的文件，你现在不需要对其做任何操作。

3 仿真性能数据

本小结将详细说明如何引导用户输入微处理器的信息并调用性能模拟模块。

3.1 安装前置软件

如果用户首次打开向导软件，需要先检查安装前置软件：

sudo apt install build-essential git m4 scons zlib1g zlib1g-dev libprotobuf-dev protobuf-compiler libprotoc-dev libgoogle-perftools-dev python-dev python python-pip -y

pip install six

如果用户没有安装上面的软件，就帮助用户安装；如果已安装，就欢迎用户使用本软件，进入 3.2。

3.2 选择处理器架构

引导用户选择要使用的处理器架构。可选项为：

- Arm
- Mips
- X86
- Alpha

用户必须从中选择一个。目前仿真流程仅支持 X86，如果用户选择 X86，则进入 3.3；如果用户选择其他选项，则退出软件。

3.3 选择基准程序集

引导用户选择基准程序集。可选项为：

- SPEC 2017
- PARSEC

用户必须从中选择一个。如果选择 SPEC 2017，则进入 3.3.1；如果选择 PARSEC，则进入 3.3.2

3.3.1 选择 SPEC 2017 基准程序

如果用户在 3.3 中选择了 SPEC 2017，则进一步引导用户选择基准程序，可选项如下：

SPECrate®2017 Integer	SPECspeed®2017 Integer	Language [1]	KLOC [2]	Application Area
500.perlbench_r	600.perlbench_s	C	362	Perl interpreter
502.gcc_r	602.gcc_s	C	1,304	GNU C compiler
505.mcf_r	605.mcf_s	C	3	Route planning
520.omnetpp_r	620.omnetpp_s	C++	134	Discrete Event simulation - computer network
523.xalancbmk_r	623.xalancbmk_s	C++	520	XML to HTML conversion via XSLT
525.x264_r	625.x264_s	C	96	Video compression
531.deepsjeng_r	631.deepsjeng_s	C++	10	Artificial Intelligence: alpha-beta tree search (Chess)
541.leela_r	641.leela_s	C++	21	Artificial Intelligence: Monte Carlo tree search (Go)
548.exchange2_r	648.exchange2_s	Fortran	1	Artificial Intelligence: recursive solution generator (Sudoku)
557.xz_r	657.xz_s	C	33	General data compression

SPECrate®2017 Floating Point	SPECspeed®2017 Floating Point	Language [1]	KLOC [2]	Application Area
503.bwaves_r	603.bwaves_s	Fortran	1	Explosion modeling
507.cactuBSSN_r	607.cactuBSSN_s	C++, C, Fortran	257	Physics: relativity
508.namd_r		C++	8	Molecular dynamics
510.parest_r		C++	427	Biomedical imaging: optical tomography with finite elements
511.povray_r		C++, C	170	Ray tracing
519.lbm_r	619.lbm_s	C	1	Fluid dynamics
521.wrf_r	621.wrf_s	Fortran, C	991	Weather forecasting
526.blender_r		C++, C	1,577	3D rendering and animation
527.cam4_r	627.cam4_s	Fortran, C	407	Atmosphere modeling
	628.pop2_s	Fortran, C	338	Wide-scale ocean modeling (climate level)
538.imagick_r	638.imagick_s	C	259	Image manipulation
544.nab_r	644.nab_s	C	24	Molecular dynamics
549.fotonik3d_r	649.fotonik3d_s	Fortran	14	Computational Electromagnetics
554.roms_r	654.roms_s	Fortran	210	Regional ocean modeling

第一列和第二列中的基准程序名（蓝色字体）都是可选项，其余列则作为备注信息。

为了便于操作，请添加全选/全不选功能。用户点击全选后能选择全部基准程序，点击全不选后能放弃选择全部基准程序。

用户必须从中选择一个或多个基准程序。目前仿真流程尚不支持 SPEC 2017，退出软件。

3.3.2 选择 PARSEC 基准程序

如果用户在 3.3 中选择了 PARSEC，则进一步引导用户选择基准程序，可选项如下：

Program	Application Domain	Parallelization		Working Set	Data Usage	
		Model	Granularity		Sharing	Exchange
blackscholes	Financial Analysis	data-parallel	coarse	small	low	low
bodytrack	Computer Vision	data-parallel	medium	medium	high	medium
canneal	Engineering	unstructured	fine	unbounded	high	high
dedup	Enterprise Storage	pipeline	medium	unbounded	high	high
facesim	Animation	data-parallel	coarse	large	low	medium
ferret	Similarity Search	pipeline	medium	unbounded	high	high
fluidanimate	Animation	data-parallel	fine	large	low	medium
freqmine	Data Mining	data-parallel	medium	unbounded	high	medium
raytrace	Rendering	data-parallel	medium	unbounded	high	low
streamcluster	Data Mining	data-parallel	medium	medium	low	medium
swaptions	Financial Analysis	data-parallel	coarse	medium	low	low
vips	Media Processing	data-parallel	coarse	medium	low	medium
x264	Media Processing	pipeline	coarse	medium	high	high

第一列中的基准程序名都是可选项，其余列则作为备注信息。

为了便于操作，请添加全选/全不选功能。用户点击全选后能选择全部基准程序，点击全不选后能放弃选择全部基准程序。

用户必须从中选择一个或多个基准程序，然后进入 3.4。

3.4 选择测试集

引导用户选择一个测试集，可选项如下：

Input Set	Description	Time	Purpose
Test	Minimal execution time	N/A	Test & Development
Simdev	Best-effort code coverage of real inputs	N/A	
Simsmall	Small-scale experiments	≤ 1s	Simulations
Simmedium	Medium-scale experiments	≤ 4s	
Simlarge	Large-scale experiments	≤ 15s	
Native	Real-world behavior	≤ 15min	Native execution

其中，第一列都是可选项，其余列为备注信息。

用户必须从中选择一个测试集，然后进入 3.5。

3.5 输入线程数

引导用户输入一个线程数，线程数必须是一个正整数。

用户输入一个线程数后，进入 3.6。

3.6 生成脚本

进入实验目录下的 TR-09-32-parsec-2.1-alpha-files 文件夹，在该文件夹目录下

调用 writescripts.pl，具体命令为：

./writescripts.pl {基准程序名} {线程数}

其中，{基准程序名}为用户在 3.3.2 中选择的基准程序的名字，线程数为用户在 3.5 中输入的线程数。

基准程序名一次只能输入一个，如果用户选择了多个基准程序，则需要反复调用 writescripts.pl。例如：用户在 3.3.2 中选择了 blackscholes 和 bodytrack 基准程序，在 3.5 中输入的线程数为 1，那么你需要调用两次 writescripts.pl：

./writescripts.pl blackscholes 1

./writescripts.pl bodytrack 1

又例如，如果用户选择了 10 个基准程序，那么你需要调用 10 次 writescripts.pl，每次输入 10 个基准程序中的 1 个。

完成后，会得到许多.rcS 文件，例如“blackscholes_1c_simsmall.rcS”。文件名中，blackscholes 和 1c 表示你在调用 writescripts.pl 时输入的基准程序名和线程数；simsmall 表示测试集，除了 simsmall，还会生成其他测试集对应的 rcS 文件。

不要做任何操作，进入 3.7。

3.7 仿真性能数据

清空实验目录下的 gem5_output 文件夹。

进入实验目录下的 gem5 文件夹，在该目录下调用性能仿真模块，具体命令为：

M5_PATH=../full_system_images/ ./build/X86/gem5.opt configs/example/fs.py //

--script=../TR-09-32-parsec-2.1-alpha-files/{基准程序名}_{线程数}c_{测试集}.rcS //

--disk-image=x86root-parsec.img //

--kernel=x86_64-vmlinux-2.6.28.4-smp --caches //

--l2cache --cpu-type 'DerivO3CPU' --maxtime=10

其中，{基准程序名}_{线程数}c_{测试集}.rcS 就是在 3.6 中生成的 rcS 文件。{基准程序名}和{线程数}在 3.6 中已经解释过了，{测试集}指的是用户在 3.4 中选择的测试集。

一次仿真完成后，会在 m5out 文件夹下得到仿真结果，其包含：

- stats.txt：微处理器的性能数据
- config.json：微处理器的硬件信息
- 其他文件

在实验目录下的 gem5_output 文件夹下创建新文件夹，文件夹名为{基准程序名}。将上述 m5out 中的文件全部复制到刚才创建的{基准程序名}文件夹内。

重复此过程，对用户选择的每一个基准程序都完成仿真，退出软件。

为了便于说明，我举个例子。假如用户选择了 blackscholes 和 bodytrack 基准程序，线程数为 1，测试集为 simsmall。那么你需要做的是：

- 进入/gem5_output 文件夹，清空所有内容，然后创建 blackscholes 文件夹和 bodytrack 文件夹
- 进入/gem5 文件夹，调用性能仿真模块，输入 blackscholes_1c_simsmall.rcS
- 将/gem5/m5out 文件夹内的内容复制到/gem5_output/blackscholes 中
- 调用性能仿真模块，输入 bodytrack_1c_simsmall.rcS
- 将/gem5/m5out 文件夹内的内容复制到/gem5_output/bodytrack 中

4 其他

本文为制作微处理器热仿真向导软件提供了一个方案。方案目前仅包含仿真微处理器性能数据的步骤，我们先将这一部分工作做好，然后看情况做后续步骤。

本文提出的方案仅为参考，如果黄同学有更好的解决方案，或者有任何问题，请随时与我沟通。黄同学也可以大胆发挥，在保证软件具有本方案要求功能的前提下，怎么做可以让软件更好用、更合理？

虽然热仿真流程已经过测试，但是我很难保证它在其他环境下的表现。如果在调用项目文件时遇到了 bug，黄同学可先将用户界面做好。