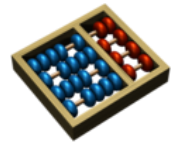




Universidade Estadual de Campinas  
Instituto de Computação  
Arquitetura de Computadores II – MO601  
Prof. Rodolfo Jardim de Azevedo



# **Projeto 3**

## **Experimental ferramentas e coletar dados**

**Rubens de Castro Pereira**

**RA 217146**

Campinas – SP

Maio de 2023

# Índice

1	Introdução.....	4
2	Ambiente de Experimentação .....	4
3	Ferramentas experimentadas .....	5
3.1	SPEC CPU 2017 benchmark * .....	5
3.2	Simulador multi-core Sniper * .....	7
3.3	Perf profiler * .....	8
3.4	PARSEC Benchmark Suite 3.0 * .....	14
3.5	Rodinia benchmark * .....	17
3.6	Intel Pin .....	21
3.7	Dinero cache simulator .....	22
4	Considerações sobre o aprendizado nesse projeto .....	22
5	Conclusões.....	22



# 1 Introdução

Repositório Github  <https://github.com/rubenscp/RCP-MO601-Project-03>

## 2 Ambiente de Experimentação

O computador utilizado em todos os experimentos está descrito conforme segue e será denominado “Laptop Rubens”:

- Notebook HP Pavilion dm4
- Memória RAM: 16 Gbytes
- SSD: 1 TBytes
- Sistema Operacional
  - Ubuntu 22.04.2 LTS (GNU/Linux 5.15.90.1-microsoft-standard-WSL2 x86\_64)
- CPU:
  - Model name: Intel(R) Core(TM) i7-2620M CPU @ 2.70GHz
  - Architecture: x86\_64
  - CPU op-mode(s): 32-bit, 64-bit
  - Address sizes: 36 bits physical, 48 bits virtual
  - Byte Order: Little Endian
  - CPU(s): 4
  - Vendor ID: GenuineIntel
  - CPU family: 6
  - Thread(s) per core: 2
  - Core(s) per socket: 2
  - L1d cache: 64 KiB (2 instances)
  - L1i cache: 64 KiB (2 instances)
  - L2 cache: 512 KiB (2 instances)
  - L3 cache: 4 MiB (1 instance)

### 3 Ferramentas experimentadas

As ferramentas utilizadas para avaliações em arquitetura de computadores foram SPEC CPU 2017, Simulador multi-core Sniper, Perf profiler, Parsec benchmark, Rodinia benchmark, Intel Pin e Dinero cache simulator. Os dados coletados para cada uma das ferramentas são apresentados nas próximas seções.

#### 3.1 SPEC CPU 2017 benchmark \*

O SPEC CPU 2017 é um pacote de benchmark que contém a próxima geração de SPECs (*Standard Performance Evaluation Corporation*), pacotes de processamento intensivo de CPU para medição e comparação de desempenho computacional, sobrecarregando o processador do sistema, memória e compilador. Esta ferramenta oferece 4 suites para benchmark considerando velocidade (*speed*) e throughput (*rate*) para números inteiros e em ponto flutuante: intspeed, fpspeed, intrate e fprate.

A Tabela 1 apresenta o resumo da experimentação do SPEC CPU 2017 no laptop Rubens com os parâmetros de execução como número de cópias, *threads*, número de iterações, tempo de execução e métrica final da execução (base).

Resultados da execução do SPEC CPU 2017						
Suíte	Cópias	Threads	Nº Iterações	Qtde de Benchmarks	Tempo de execução	Métrica Final (base)
intspeed	4	4	3	9	17993 s - 4,99 hs	<b>3,42</b>
intspeed	8	8	3	10	18438 s - 5,12 hs	<b>3,35</b>
intspeed	16	16	3	10	32523 s - 9,03 hs	<b>1,96</b>
intrate	4	4	3	10	38073 s - 10,57 hs	<b>5,32</b>
intrate	8	8	3	9	65121 s - 18,08 hs	<b>5,03</b>
fpspeed	4	4	3	9	79708 s - 22,14 hs	<b>3,11</b>
fprate	4	4	3	13	58396 s - 16,22 hs	<b>6,25</b>
Duração total das execuções					310252 s - 86.18 hs	

Tabela 1. Suites executadas na ferramenta SPEC CPU 2017 com seus parâmetros da execução, o tempo de execução e a métrica final da execução.

A Tabela 2 apresenta a comparação do computador utilizado no experimento (Laptop Rubens) e outros computadores selecionados da lista de resultados disponível no site do SPEC CPU 2017 (<https://www.spec.org/cpu2017/results/cpu2017.html>). Os computadores selecionados são aqueles que mais se aproximam das características do computador “Laptop Rubens” a fim de que as comparações das métricas finais possam ser equilibradas e justas.

Suite	Threads	Métrica obtida do Laptop Rubens	Outros computadores	Métrica
intspeed	4	int_base: 3,42	SuperWorkstation 5039C-T (X11SCA , Intel Core i3-8100)	int_base: 7,58
intspeed	8	Int_base: 3,35	SuperWorkstation 5039C-T (X11SCA , Intel Core i7-9700K)	int_base: 10,6
intspeed	16	int_base: 1,96	Não localizado computador equivalente com thread = 16	---
intrate	4	int_base: 5,32	ASUS Z170M-PLUS Motherboard (Intel Core i7-6700K)	int_base: 23,5
intrate	8	int_base: 5,03	SuperWorkstation 5039C-T (X11SCA , Intel Core i7-9700K)	int_base: 44,8
fpspeed	4	fp_base: 3,11	SuperWorkstation 5039C-T (X11SCA , Intel Core i7-9700K)	fp_base: 32,2
fprate	4	fp_base: 6,25	SuperWorkstation 5039C-T (X11SCA , Intel Core i7-9700K)	fp_base: 42,6

Tabela 2. Comparação das métricas dos benchmarks executados no laptop Rubens e outros computadores.

A Tabela 3 apresenta os comandos com os parâmetros utilizados na execução de cada uma das suítes.

Suíte	Comando para execução do SPEC CPU 2017
intspeed	<code>runcpu --config=rubens-try1 --noreportable --iterations=3 600.perlbench_s 602.gcc_s 605.mcf_s 620.omnetpp_s 623.xalancbmk_s 625.x264_s 631.deepsjeng_s 641.leela_s 648.exchange2_s 998.specrand_is</code>
intrate	<code>runcpu --config=rubens-try1 --reportable --iterations=3 intrate</code>
fpspeed	<code>runcpu --config=rubens-try1 --noreportable --iterations=3 603.bwaves_s 607.cactuBSSN_s 619.lbm_s 621.wrf_s 628.pop2_s 638.imagick_s 644.nab_s 649.fotonik3d_s 654.roms_s 996.specrand_fs</code>
fprate	<code>runcpu --config=rubens-try1 --reportable --iterations=3 fprate</code>

Tabela 3. Comandos SPEC CPU 2017 executados para as suítes inspeed, intrate, fpspeed e fprate.

### 3.2 Simulador multi-core Sniper \*

.....

A Tabela 4 apresenta os comandos utilizados na execução dos programas bem como a indicação dos arquivos de resultados.

Comando para execução do programa	Resultado	Arquivo com o resultado da execução
make run > sniper-result-api.txt	sucesso	sniper-result-api.txt
make run > sniper-result-dvfs.txt	sucesso	sniper-result-dvfs.txt
make run > sniper-result-fft.txt	sucesso	sniper-result-fft.txt
make run > sniper-result-fft-dvfs.txt	sucesso	sniper-result-fft-dvfs.txt
make run > sniper-result-fft-hetero.txt	sucesso	sniper-result-fft-hetero.txt
make run > sniper-result-fft-hetero-cfg.txt	erro	sniper-result-fft-hetero-cfg.txt
make run > sniper-result-fft-marker.txt	erro	sniper-result-fft-marker.txt
make run > sniper-result-fork.txt	sucesso	sniper-result-fork.txt
make run > sniper-result-shared.txt	sem programa fonte	sniper-result-shared.txt
make run > sniper-result-signal.txt	erro	sniper-result-signal.txt
make run > sniper-result-smc.txt	erro	sniper-result-smc.txt
make run > sniper-result-sniper-in-sniper.txt	erro	sniper-result-sniper-in-sniper.txt
make run > sniper-result-spinloop.txt	sucesso	sniper-result-spinloop.txt
make run > sniper-result-true.txt	sucesso	sniper-result-true.txt
<b>Programas adicionais na pasta "extra_programs"</b>		
../run-sniper ./RADIX > sniper-result-radix.txt	sucesso	sniper-result-RADIX
../run-sniper ./CHOLESKY tk14.0 > sniper-result-cholesky.txt	sucesso	sniper-result-CHOLESKY

Tabela 4. Comandos Sniper executados nos benchmarks do experimento.

Programas selecionados para uso avaliação mais detalhada.

Programas selecionados	Tempo de execução no simulador Snipe "Total Time" (TSni)	Tempo de execução nativo "Total Time" (TNat)	Slowdown de simulação (TSni / TNat)
radix	2430 ms	2585 ms	0,940
cholesky	2946 ms	5084 ms	0,579
fft	248 ms	376 ms	0,659

- Apresentar algumas métricas de desempenho coletadas pelo simulador Sniper.

- o Start time : -1844408187
- o Initialization finish time : -1844371499
- o Overall finish time : -1844369069
- o Total time with initialization : 39118
- o Total time without initialization : 2430

Programas seleccionados	Sniper				
Radix – Sniper	PROCESS STATISTICS				
		Total	Rank	Sort	
	Proc	Time	Time	Time	
	0	2430	1115	1315	
	TIMING INFORMATION				
	Start time	:	-1844408187		
	Initialization finish time	:	-1844371499		
	Overall finish time	:	-1844369069		
	Total time with initialization	:	39118		
	Total time without initialization	:	2430		
	Radix – Native	PROCESS STATISTICS			
			Total	Rank	Sort
Proc		Time	Time	Time	
0		2585	742	1840	
TIMING INFORMATION					
Start time		:	1102732390		
Initialization finish time		:	1102761743		
Overall finish time		:	1102764328		
Total time with initialization		:	31938		
Total time without initialization		:	2585		
cholesky					
cholesky					



Fft – Sniper	PROCESS STATISTICS			
	Proc	Computation	Transpose	Transpose
		Time	Time	Fraction
	0	248	28	0.11290
	TIMING INFORMATION			
	Start time	:	-1844408306	
	Initialization finish time	:	-1844407925	
	Overall finish time	:	-1844407677	
	Total time with initialization	:	629	
	Total time without initialization	:	248	
Fft – Native	Overall transpose time	:	28	
	Overall transpose fraction	:	0.11290	
	PROCESS STATISTICS			
	Proc	Computation	Transpose	Transpose
		Time	Time	Fraction
	0	376	61	0.16223
	TIMING INFORMATION			
	Start time	:	-1988961673	
	Initialization finish time	:	-1988961469	
	Overall finish time	:	-1988961093	
Total time with initialization	:	580		
Total time without initialization	:	376		
Overall transpose time	:	61		
Overall transpose fraction	:	0.16223		

### 3.3 Perf profiler \*

Perf profiler é uma ferramenta Linux que coleta e analisa dados de desempenho de programas ou do sistema operacional.

Os programas selecionados para avaliação são: fft, fork, signal, smc e true.

A Tabela 5 apresenta os comandos com os parâmetros utilizados na execução de cada um dos programas selecionados.

Programa	Comando de execução
fft	perf stat -B ./fft
Resultado da Execução	
<pre>FFT with Blocking Transpose 1024 Complex Doubles 1 Processors 65536 Cache lines 16 Byte line size 4096 Bytes per page  PROCESS STATISTICS Computation      Transpose      Transpose Proc            Time             Time             Fraction 0                303              61               0.20132  TIMING INFORMATION Start time                :          695908542 Initialization finish time :          695908777 Overall finish time       :          695909080 Total time with initialization :          538 Total time without initialization :          303 Overall transpose time     :          61 Overall transpose fraction :          0.20132  Performance counter stats for './fft':  1.06 msec task-clock          #    0.746 CPUs utilized 0      context-switches      #    0.000 /sec 0      cpu-migrations        #    0.000 /sec 62     page-faults           #   58.701 K/sec 2540158      cycles          #    2.405 GHz 1814472      stalled-cycles-frontend #   71.43% frontend cycles idle 1042654      stalled-cycles-backend  #   41.05% backend cycles idle 2132716      instructions    #    0.84  insn per cycle #    0.85  stalled cycles per insn 195210      branches         #   184.823 M/sec 6000      branch-misses      #    3.07% of all branches  0.001415500 seconds time elapsed  0.001717000 seconds user 0.000000000 seconds sys</pre>	

Programa	Comando de execução
fft	perf stat -B ./fft if=/dev/zero of=/dev/null count=1000000
<b>Resultado da Execução</b>	
<pre> FFT with Blocking Transpose 1024 Complex Doubles 1 Processors 65536 Cache lines 16 Byte line size 4096 Bytes per page  PROCESS STATISTICS Computation      Transpose      Transpose Proc            Time              Time              Fraction 0                335                59                0.17612  TIMING INFORMATION Start time              :          760328360 Initialization finish time :          760328587 Overall finish time      :          760328922 Total time with initialization :          562 Total time without initialization :          335 Overall transpose time    :          59 Overall transpose fraction :          0.17612  Performance counter stats for './fft if=/dev/zero of=/dev/null count=1000000':  1.01 msec task-clock          #    0.731 CPUs utilized 0      context-switches      #    0.000 /sec 0      cpu-migrations         #    0.000 /sec 62     page-faults           #   61.198 K/sec 2561675 cycles                #    2.529 GHz 1839401 stalled-cycles-frontend #   71.80% frontend cycles idle 1031193 stalled-cycles-backend  #   40.25% backend cycles idle 2135867 instructions          #    0.83 insn per cycle #    0.86 stalled cycles per insn 195848 branches              #   193.316 M/sec 5650   branch-misses         #    2.88% of all branches  0.001386500 seconds time elapsed  0.001920000 seconds user 0.000000000 seconds sys </pre>	
Programa	Comando de execução
fork	perf stat -B ./fork
<b>Resultado da Execução</b>	
<pre> Hello world from parent Hello world from child  Performance counter stats for './fork':  0.94 msec task-clock          #    0.045 CPUs utilized 2      context-switches      #    2.139 K/sec 0      cpu-migrations         #    0.000 /sec 55     page-faults           #   58.811 K/sec 1820753 cycles                #    1.947 GHz 1528003 stalled-cycles-frontend #   83.92% frontend cycles idle 1222034 stalled-cycles-backend  #   67.12% backend cycles idle 578101 instructions          #    0.32 insn per cycle #    2.64 stalled cycles per insn 118406 branches              #   126.610 M/sec 6079   branch-misses         #    5.13% of all branches  0.020806800 seconds time elapsed </pre>	

0.001550000 seconds user 0.000000000 seconds sys	
Programa	Comando de execução
signal	perf stat -B ./signal
Resultado da Execução	
<pre> Installing signal handler Dereferencing NULL pointer Received signal 11  Performance counter stats for './signal':  0.45 msec task-clock           #    0.545 CPUs utilized 0      context-switches       #    0.000 /sec 0      cpu-migrations         #    0.000 /sec 30     page-faults            #   66.800 K/sec 978289      cycles            #    2.178 GHz 798099      stalled-cycles-frontend #   81.58% frontend cycles idle 634507      stalled-cycles-backend  #   64.86% backend cycles idle 350129      instructions       #    0.36  insn per cycle #      2.28  stalled cycles per insn 70957      branches           #   157.998 M/sec 3626      branch-misses       #    5.11% of all branches  0.000824699 seconds time elapsed  0.000910000 seconds user 0.000000000 seconds sys </pre>	
Programa	Comando de execução
smc	perf stat -B ./smc
Resultado da Execução	
<pre> Good morning!  Performance counter stats for './smc':  0.47 msec task-clock           #    0.588 CPUs utilized 0      context-switches       #    0.000 /sec 0      cpu-migrations         #    0.000 /sec 29     page-faults            #   61.259 K/sec 1070708      cycles            #    2.262 GHz 842947      stalled-cycles-frontend #   78.73% frontend cycles idle 621900      stalled-cycles-backend  #   58.08% backend cycles idle 493319      instructions       #    0.46  insn per cycle #      1.71  stalled cycles per insn 92454      branches           #   195.298 M/sec 3768      branch-misses       #    4.08% of all branches  0.000805700 seconds time elapsed  0.000894000 seconds user 0.000000000 seconds sys </pre>	
Programa	Comando de execução
true	perf stat -B ./true
Resultado da Execução	
<pre> Performance counter stats for './true':  0.48 msec task-clock           #    0.548 CPUs utilized 0      context-switches       #    0.000 /sec 0      cpu-migrations         #    0.000 /sec 28     page-faults            #   58.700 K/sec 855835      cycles            #    1.794 GHz </pre>	

```

697077      stalled-cycles-frontend      #    81.45% frontend cycles idle
536036      stalled-cycles-backend      #    62.63% backend cycles idle
323449      instructions                 #    0.38  insn per cycle
#    2.16  stalled cycles per insn
64571       branches                     #   135.369 M/sec
3028        branch-misses                #    4.69% of all branches

0.000870500 seconds time elapsed

0.000992000 seconds user
0.000000000 seconds sys

```

*Tabela 5. Comandos Perf profiler executados nos programas selecionados no experimento.*

- Incluir os programas RADIX e
- perf stat -B ./RADIX
- perf stat -B ./CHOLESKY tk14.O
- Extrair as mesmas métricas do Sniper de forma nativa
- Comparar as métricas do Perf com as do Sniper
- Justificar as diferenças

### 3.4 PARSEC Benchmark Suite 3.0 \*

O PARSEC (*Princeton Application Repository for Shared-Memory Computers*) é um conjunto de benchmark composto por programas *multithread* com o propósito de possibilitar estudos de desempenho em computadores com múltiplos processadores.

A Tabela 6 apresenta a compilação dos pacotes de benchmark oferecidos no PARSEC com o resultado indicando sucesso ou os erros apresentados no processo de compilação (build).

Pacote	Comando para compilação	Resultado
blackscholes	parsecmgmt -a build -p blackscholes	Compilou com sucesso.
bodytrack	parsecmgmt -a build -p bodytrack	Compilou com sucesso.
facesim	parsecmgmt -a build -p facesim	make[2]: *** [/usr/local/parsec-3.0/pkgs/apps/facesim/obj/amd64-linux.gcc/Public_Library/Makefile.common:407: obj/Collisions_And_Interactions/COLLISION_BODY_LIST_3D.o] Error 1 make[2]: Leaving directory '/usr/local/parsec-3.0/pkgs/apps/facesim/obj/amd64-linux.gcc/Public_Library' make[1]: *** No rule to make target '/usr/local/parsec-3.0/pkgs/apps/facesim/obj/amd64-linux.gcc/lib/libPhysBAM.a', needed by 'facesim'. Stop. make[1]: Leaving directory '/usr/local/parsec-3.0/pkgs/apps/facesim/obj/amd64-linux.gcc/Benchmarks/facesim' make: *** [Makefile:16: all] Error 2 [PARSEC] Error: 'env version=threads PHYSBAM=/usr/local/parsec-3.0/pkgs/apps/facesim/obj/amd64-linux.gcc CXXFLAGS=-O3 -g -funroll-loops -fprefetch-loop-arrays -fpermissive -fno-exceptions -std=c++11 -static-libgcc -Wl,--hash-style=both,--as-needed -DPARSEC_VERSION=3.0-beta-20150206 -fexceptions /usr/bin/make' failed.
ferret	parsecmgmt -a build -p ferret	make: *** [Makefile:108: /usr/local/parsec-3.0/pkgs/apps/ferret/obj/amd64-linux.gcc/parsec/obj/LSH_query.o] Error 1 [PARSEC] Error: 'env version=threads CFLAGS=-I/usr/local/parsec-3.0/pkgs/libs/gsl/inst/amd64-linux.gcc/include -I/usr/local/parsec-3.0/pkgs/libs/libjpeg/inst/amd64-linux.gcc/include -O3 -g -funroll-loops -fprefetch-loop-arrays -static-libgcc -Wl,--hash-style=both,--as-needed -DPARSEC_VERSION=3.0-beta-20150206 LDFLAGS=-L/usr/local/parsec-3.0/pkgs/libs/gsl/inst/amd64-linux.gcc/lib -L/usr/local/parsec-3.0/pkgs/libs/libjpeg/inst/amd64-linux.gcc/lib -L/usr/lib64 -L/usr/lib /usr/bin/make' failed.
fluidanimate	parsecmgmt -a build -p fluidanimate	Compilou com sucesso.
frequine	parsecmgmt -a build -p frequine	Compilou com sucesso.
raytrace	parsecmgmt -a build -p raytrace	No package 'xext' found  Consider adjusting the PKG_CONFIG_PATH environment variable if you installed software in a non-standard prefix.  Alternatively, you may set the environment variables XLIBGL_CFLAGS and XLIBGL_LIBS to avoid the need to call pkg-config. See the pkg-config man page for more details.  [PARSEC] Error: 'env ./configure --with-driver=xlib --enable-glut --enable-static --disable-shared --prefix=/usr/local/parsec-3.0/pkgs/libs/mesa/inst/amd64-linux.gcc' failed.
swaptions	parsecmgmt -a build -p swaptions	~~~~~ make[1]: *** [../build/Makefile.tbmalloc:70: proxy.o] Error 1 make[1]: Leaving directory '/usr/local/parsec-3.0/pkgs/libs/tbllib/obj/amd64-linux.gcc/build/linux_intel64_gcc_cc11.3.0_libc2.35_kernel5.15.90.1_debu g' make: *** [Makefile:49: tbmalloc] Error 2 [PARSEC] Error: 'env compiler=gcc PATH=/usr/bin:/usr/local/parsec-3.0/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bi

		n:/usr/local/parsec-3.0/bin CXXFLAGS=-O3 -g -funroll-loops -fprefetch-loop-arrays -fpermissive -fno-exceptions -static-libgcc -Wl,--hash-style=both,--as-needed -DPARSEC_VERSION=3.0-beta-20150206 -fexceptions /usr/bin/make' failed.
vips	parsecmgmt -a build -p vips	Compilou com sucesso.

Tabela 6. Resultado da compilação dos pacotes do PARSEC.

O PARSEC possibilita definir a região de interesse (ROI – *Region Of Interest*) baseada em seis tipos de entrada possíveis na execução dos benchmarks. As entradas são: test, simdev, simsmall, simmedium, simlarge e native.

Os testes realizados no experimento utilizaram todas as entradas nos benchmarks executados, cujos comandos de execução e indicação dos resultados são apresentados na Tabela 7 que segue. As saídas da execução estão armazenadas nos arquivos com extensão “txt”.

## EXPLORAR O PARALELISMO –N

parsecmgmt -a run -p fluidanimate -i native -n 16

parsecmgmt -a run -p blackscholes -i native -n 16

parsecmgmt -a run -p freqmine -i native -n 16

parsecmgmt -a run -p vips -i native -n 16

parsecmgmt -a run -p fluidanimate -i native -n 8

parsecmgmt -a run -p blackscholes -i native -n 8

parsecmgmt -a run -p freqmine -i native -n 8

parsecmgmt -a run -p vips -i native -n 8

parsecmgmt -a run -p fluidanimate -i native -n 4

parsecmgmt -a run -p blackscholes -i native -n 4

parsecmgmt -a run -p freqmine -i native -n 4

parsecmgmt -a run -p vips -i native -n 4

parsecmgmt -a run -p fluidanimate -i native -n 2

parsecmgmt -a run -p blackscholes -i native -n 2

parsecmgmt -a run -p freqmine -i native -n 2

parsecmgmt -a run -p vips -i native -n 2

parsecmgmt -a run -p fluidanimate -i native

parsecmgmt -a run -p blackscholes -i native

parsecmgmt -a run -p freqmine -i native

parsecmgmt -a run -p vips -i native

Núm. da Execução	Pacote	Entrada	Comando de execução do pacote de Benchmark
001	blackscholes	test	parsecmgmt -a run -p blackscholes -i test > result/exec-001-blackscholes-test.txt
002	blackscholes	simdev	parsecmgmt -a run -p blackscholes -i simdev > result/exec-002-blackscholes-simdev.txt
003	blackscholes	simsmall	parsecmgmt -a run -p blackscholes -i simsmall > result/exec-003-blackscholes-simsmall.txt
004	blackscholes	simlarge	parsecmgmt -a run -p blackscholes -i simlarge > result/exec-004-blackscholes-simlarge.txt
005	blackscholes	native	parsecmgmt -a run -p blackscholes -i native > result/exec-005-blackscholes-native.txt
006	vips	test	parsecmgmt -a run -p vips -i test > result/exec-006-vips-test.txt
007	vips	simdev	parsecmgmt -a run -p vips -i simdev > result/exec-007-vips-simdev.txt
008	vips	simsmall	parsecmgmt -a run -p vips -i simsmall > result/exec-008-vips-simsmall.txt
009	vips	simlarge	parsecmgmt -a run -p vips -i simlarge > result/exec-009-vips-simlarge.txt
010	vips	native	parsecmgmt -a run -p vips -i native > result/exec-010-vips-native.txt
011	bodytrack	test	parsecmgmt -a run -p bodytrack -i test > result/exec-011-bodytrack-test.txt
012	bodytrack	simdev	parsecmgmt -a run -p bodytrack -i simdev > result/exec-012-bodytrack-simdev.txt
013	bodytrack	simsmall	parsecmgmt -a run -p bodytrack -i simsmall > result/exec-013-bodytrack-simsmall.txt
014	bodytrack	simlarge	parsecmgmt -a run -p bodytrack -i simlarge > result/exec-014-bodytrack-simlarge.txt
015	bodytrack	native	parsecmgmt -a run -p bodytrack -i native > result/exec-015-bodytrack-native.txt
016	fluidanimate	test	parsecmgmt -a run -p fluidanimate -i test > result/exec-016-fluidanimate-test.txt
017	fluidanimate	simdev	parsecmgmt -a run -p fluidanimate -i simdev > result/exec-017-fluidanimate-simdev.txt
018	fluidanimate	simsmall	parsecmgmt -a run -p fluidanimate -i simsmall > result/exec-018-fluidanimate-simsmall.txt
019	fluidanimate	simlarge	parsecmgmt -a run -p fluidanimate -i simlarge > result/exec-019-fluidanimate-simlarge.txt
020	fluidanimate	native	parsecmgmt -a run -p fluidanimate -i native > result/exec-020-fluidanimate-native.txt



021	freqmine	test	parsecmgmt -a run -p freqmine -i test > result/exec-021-freqmine-test.txt
022	freqmine	simdev	parsecmgmt -a run -p freqmine -i simdev > result/exec-022-freqmine-simdev.txt
023	freqmine	simsmall	parsecmgmt -a run -p freqmine -i simsmall > result/exec-023-freqmine-simsmall.txt
024	freqmine	simlarge	parsecmgmt -a run -p freqmine -i simlarge > result/exec-024-freqmine-simlarge.txt
025	freqmine	native	parsecmgmt -a run -p freqmine -i native > result/exec-025-freqmine-native.txt
026	splash2	test	parsecmgmt -a run -p splash2 -i test > result/exec-026-splash2-test.txt
027	splash2	simdev	parsecmgmt -a run -p splash2 -i simdev > result/exec-027-splash2-simdev.txt
028	splash2	simsmall	parsecmgmt -a run -p splash2 -i simsmall > result/exec-028-splash2-simsmall.txt
029	splash2	simlarge	parsecmgmt -a run -p splash2 -i simlarge > result/exec-029-splash2-simlarge.txt
030	splash2	native	parsecmgmt -a run -p splash2 -i native > result/exec-030-splash2-native.txt
031	splash2x	test	parsecmgmt -a run -p splash2x -i test > result/exec-031-splash2x-test.txt
032	splash2x	simdev	parsecmgmt -a run -p splash2x -i simdev > result/exec-032-splash2x-simdev.txt
033	splash2x	simsmall	parsecmgmt -a run -p splash2x -i simsmall > result/exec-033-splash2x-simsmall.txt
034	splash2x	simlarge	parsecmgmt -a run -p splash2x -i simlarge > result/exec-034-splash2x-simlarge.txt
035	splash2x	native	parsecmgmt -a run -p splash2x -i native > result/exec-035-splash2x-native.txt

Tabela 7. Comandos PARSEC para execução dos benchmarks com as entradas possíveis.

Fazer um gráfico quatro aplicações, variando o valor de N no eixo X e o tempo real no Y. Usar N=1 a referencia (baseline)

### 3.5 Rodinia benchmark \*

O Rodinia Benchmark é uma ferramenta destinada a infraestrutura de computação heterogênea com implementações com OpenMP, OpenCL e CUDA.

A Tabela 8 apresenta a lista dos programas que foram compilados com sucesso em cada implementação.

CUDA (make CUDA)	OPENMP (make OMP)	OPENCL (make OPENCL)
<u>backprop</u>	backprop	OCL_particlefilter_double

<u>bfs</u>	bfs	OCL_particlefilter_naive
<u>dwt2d</u>	euler3d_cpu	OCL_particlefilter_single
<u>gaussian</u>	euler3d_cpu_double	backprop
<u>heartwall</u>	heartwall	gaussian
<u>hotspot</u>	hotspot	heartwall
<u>kmeans</u>	kmeans	hotspot
<u>leukocyte</u>	lavaMD	kmeans
<u>needle</u>	leukocyte	lavaMD
<u>nn</u>	lud_omp	leukocyte
<u>pathfinder</u>	needle	lud
<u>sc_gpu</u>	nn	nn
<u>srad_v1</u>	particle_filter	nw
<u>srad_v2</u>	pathfinder	srad
	pre_euler3d_cpu	
	pre_euler3d_cpu_double	
	sc_omp	
	srad_v1	
	srad_v2	

Tabela 8. Lista de programas que foram compilados com sucesso no ambiente da ferramenta Rodinia Benchmark.

A Tabela 9 apresenta a execução de benchmarks com alguns resultados detalhados ou o nome do arquivo de resultado devido ao seu tamanho excessivo.

<b>Implementação</b>	<b>Benchmark</b>	<b>Resultado</b>
OPENMP	bfs	result.txt
OPENMP	cf (euler3d)	409.637 segundos root@NotebookRubens:/usr/local/rodinia_3.1/openmp/cfd# ./run Starting... Compute time: 409.637 Saving solution... Saved solution... Cleaning up... Done...
OPENMP	heartwall	result.txt
OPENMP	hotspot	output.out
OPENMP	kmeans	root@NotebookRubens:/usr/local/rodinia_3.1/openmp/kmeans# ./run ./run: line 1: ./kmeans_serial/kmeans: No such file or directory I/O completed num of threads = 4 number of Clusters 5 number of Attributes 34  Time for process: 4.266001
OPENMP	lavaMD	root@NotebookRubens:/usr/local/rodinia_3.1/openmp/lavaMD# ./run Configuration used: cores = 4, boxes1d = 10 Time spent in different stages of CPU/MCPU KERNEL: 0.000000000000 s, 0.000000000000 % : CPU/MCPU: VARIABLES 0.000014000000 s, 0.000279933040 % : MCPU: SET DEVICE 0.000000000000 s, 0.000000000000 % : CPU/MCPU: INPUTS 5.001182079315 s, 99.999717712402 % : CPU/MCPU: KERNEL Total time: 5.001195907593 s
OPENMP	leukocyte	root@NotebookRubens:/usr/local/rodinia_3.1/openmp/leukocyte# ./run Num of threads: 4 Detecting cells in frame 0

		<p>Cells detected: 36</p> <p>Detection runtime -----</p> <p>GICOV computation: 0.52551 seconds GICOV dilation: 0.21413 seconds Total: 0.79247 seconds</p> <p>Tracking cells across 5 frames Processing frame 5 / 5</p> <p>Tracking runtime (average per frame): -----</p> <p>MGVF computation: 14.68158 seconds Snake evolution: 0.02456 seconds Total: 4.09308 seconds</p> <p>Total application run time: 21.25787 seconds</p>
OPENMP	nn	<p>root@NotebookRubens:/usr/local/rodinia_3.1/openmp/nn# ./run</p> <p>The 5 nearest neighbors are:</p> <p>1974 12 22 18 24 JOYCE 30.6 89.9 80 593 --&gt; 0.608276 2003 8 27 12 10 TONY 29.9 89.4 160 286 --&gt; 0.608275 1997 11 14 12 24 HELENE 30.5 89.8 134 529 --&gt; 0.538515 1980 10 22 18 3 ISAAC 30.1 90.4 110 778 --&gt; 0.412312 1988 12 27 0 18 TONY 30.0 89.8 113 39 --&gt; 0.199997 total time : 0.527607023716 s</p>
OPENMP	particle_filter	Result.txt
OPENMP	pathfinder	o.out
OPENMP	srاد_v1	<p>root@NotebookRubens:/usr/local/rodinia_3.1/openmp/srad/srad_v1# ./run</p> <p>Time spent in different stages of the application:</p> <p>0.000000000000 s, 0.000000000000 % : SETUP VARIABLES 0.000021000000 s, 0.001276622177 % : READ COMMAND LINE PARAMETERS 0.131821006536 s, 8.013600349426 % : READ IMAGE FROM FILE 0.002430000110 s, 0.147723421454 % : RESIZE IMAGE 0.000082999999 s, 0.005045697093 % : SETUP, MEMORY ALLOCATION 0.016366999596 s, 0.994974911213 % : EXTRACT IMAGE 1.328287959099 s, 80.748657226562 % : COMPUTE 0.005131000187 s, 0.311921358109 % : COMPRESS IMAGE 0.160110995173 s, 9.733392715454 % : SAVE IMAGE INTO FILE 0.000714000023 s, 0.043405152857 % : FREE MEMORY Total time: 1.644966006279 s</p>
OPENMP	srاد_v2	<p>root@NotebookRubens:/usr/local/rodinia_3.1/openmp/srad/srad_v2# ./run</p> <p>Randomizing the input matrix Start the SRAD main loop Computation Done</p>

Tabela 9. Benchmarks executados com os resultados.

A Tabela 10 apresenta comparações de alguns benchmarks que foram executados nas três implementações CUDA, OpenMP e OpenCL.

### Comparação de Benchmarks entre as Implementações CUDA, OpenMP e OpenCL

Hotspot		
CUDA	OpenMP	OpenCL
root@NotebookRubens:/usr/local/rodinia_3.1/cuda/hotspot# ./run WG size of kernel = 16 X 16 pyramidHeight: 2 gridSize: [512, 512] border:[2, 2] blockGrid:[43, 43] targetBlock:[12, 12] Start computing the transient temperature Ending simulation	root@NotebookRubens:/usr/local/rodinia_3.1/openmp/hotspot# ./run Start computing the transient temperature Ending simulation Total time: 0.045 seconds	Erro de execução
BFS		
CUDA	OpenMP	OpenCL
root@NotebookRubens:/usr/local/rodinia_3.1/cuda/bfs# ./run Reading File Read File Copied Everything to GPU memory Start traversing the tree Kernel Executed 1 times Result stored in result.txt >> 1.000.000 lines	root@NotebookRubens:/usr/local/rodinia_3.1/openmp/bfs# ./run Reading File Start traversing the tree Compute time: 0.633491 Result stored in result.txt >> 1.000.000 lines	Erro de compilação
HeartWall		
CUDA	OpenMP	OpenCL
root@NotebookRubens:/usr/local/rodinia_3.1/cuda/heartwall# ./run WG size of kernel = 256 frame progress: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19  Resultados no arquivo result.txt	root@NotebookRubens:/usr/local/rodinia_3.1/openmp/heartwall# ./run num of threads: 4 frame progress: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19  Resultados no arquivo result.txt	Erro de execução
particlefilter		
CUDA	OpenMP	OpenCL
Erro de compilação	video sequence took 0.043539 time to get neighbors took: 0.000005 time to get weightstook: 0.014813 time to set arrays took: 0.000106 time to set error took: 0.000682 time to get likelihoods took: 0.002394 time to get exp took: 0.000109 time to sum weights took: 0.000008 time to normalize weights took: 0.000004 time to move object took: 0.000008 xe: 64.523185 ye: 64.469547 0.702991  ... time to calc cum sum took: 0.000033 time to calc u took: 0.011697 time to calc new array x and y took: 0.061382 time to reset weights took: 0.000047 time to set error took: 0.006374 time to get likelihoods took: 0.008245 time to get exp took: 0.011114 time to sum weights took: 0.011051 time to normalize weights took: 0.008430 time to move object took: 0.016451 xe: 48.546698 ye: 72.385056 17.581630 time to calc cum sum took: 0.000034 time to calc u took: 0.013806 time to calc new array x and y took: 0.053608 time to reset weights took: 0.000045 particle filter took 0.937339	root@notebookrubens:/usr/local/rodinia_3.1/opencl/particlefilter# ./run video sequence took 0.063222 error: clgetplatformids(1,*,0) failed particle filter took 0.694592 entire program took 0.757814 video sequence took 0.031961 error: clgetplatformids(1,*,0) failed particle filter took 0.632983 entire program took 0.664944

	entire program took 0.980878	
--	------------------------------	--

*Tabela 10. Comparação de benchmarks nas três implementações CUDA, OpenMP e OpenCL.*

- Se tiver hardware suficiente, rodar as múltiplas versões do programa e comparar o desempenho no mesmo computador.
- Para múltiplas configurações do mesmo programa, indicar as diferenças de desempenho.
- Executar o Rodinia nos três programas abaixo e comparar o desempenho:
  - RADIX
  - CHOLESKY
  - FFT

### 3.6 Intel Pin

- ➔ Aguardando definição dos 3 programas para experimentos posteriores
- ➔ Executar o Pin nos três programas abaixo e comparar o desempenho utilizando a ferramenta do “PinTools” (opcodemix) aplicado aos três programas abaixo
  - RADIX
  - CHOLESKY
  - FFT

### 3.7 Dinero cache simulator

A ferramenta Dinero é um simulador de cache de 4ª geração de simuladores.

Comando de execução:

```
/dinerolV -l1-dsize 2K -l1-isize 2K -l1-ibsize 16 -l1-dbsize 8 < test.din
```

```
./dinerolV -l1-dsize 2K -l1-isize 2K -l1-ibsize 16 -l1-dbsize 8 -informat p < RADIX
```

```
./dinerolV -l1-isize 16k -l1-dsize 16k -l1-ibsize 16 -l1-dbsize 16 -l1-iassoc 8 -l1-dassoc 8 -l2-usize 2m -l2-ubsize 16 -l2-uassoc 8 -l3-usize 16m -l3-ubsize 16 -l3-uassoc 8 -informat p < RADIX
```


## 4 Considerações sobre o aprendizado nesse projeto

## 5 Conclusões