CDA 4205

Project 1

David Decker

Overview

The purpose this project is to run a provided selection of Benchmarks and find the distribution of the instructions they execute. The software used for the analysis is provided by SimpleScalar. A total of 12 Benchmarks have been provided by Professor Sah and analyzed using tools provided by SimpleScalar. The analysis will include the total number of instructions executed for each Benchmark and a percentage of the individual instructions. A series of questions are answered about each set of Benchmarks. The metric of evaluation is abbreviated. The following is a key for the abbreviations.

# of instruction = total number of instructions

Load% = overall percentage of load instructions executed

Store% = overall percentage of load instructions executed

Uncond branch% = overall percentage of unconditional branch instructions executed

Condit branch% = overall percentage of conditional branch instructions executed

Integer Comp% = overall percentage of integer computational instructions executed

Float Comp% = overall percentage of float computational instructions executed

INSTRUCTION SETS

Benchmark	# of instruction	Load%	Store%	Uncond branch %	Condit branch%	Integer Comp%	Float Comp%
anagram.alpha	25593315	25.36	9.93	4.46	10.3	44.63	5.31
go.alpha	545812421	30.62	8.17	2.58	10.96	47.64	0.03
compress.alpha	88214	1.62	79.16	0.19	5.74	13.28	0
gcc.alpha	759132	21.28	14.17	4.74	11.18	48.59	0.03

For each set of benchmarks, the following questions are answered. The questions are labeled b, c and d. Each Benchmark is answered separately.

b) Is the benchmark memory intensive of computation intensive?
c) Is the benchmark mainly using integer of floating point computations?

d) What percentage of the instructions executed are conditional branches? Given this percentage, how many instructions on average does the processor execute between each pair of conditional branch instructions? (do not include the conditional branch instructions)

For anagram.alpha:
b) 44.63% of the time running the Benchmark is spent on integer computation and an additional 5.31% is spent calculating float values. This makes the benchmark computation intensive.
c) Mainly integer computation
d) 10.3% Which averages to the execution of one pair of branch instructions per 9.71 instructions.

For go.alpha:
b) Computation intensive, but memory instructions are high
c) Integer computation
d) 10.96%, one pair per 9.12 instructions

For compress.alpha:
b) Memory intensive
c) The benchmark uses only integer computations
d) 5.74%, one pair per 17.42 instructions

For gcc.alpha
b) Computation intensive
c) Integer computation
d) 11.18%, one pair per 8.94 instructions

The following are ALPHA benchmarks

Benchmark	# of instruction	Load%	Store%	Uncond branch %	Condit branch%	Integer Comp%	Float Comp%
test-math	49481	17.2	10.4	3.93	11.11	55.33	1.88
test-famth	19570	17.78	12.47	4.68	11.36	53.12	0.42
test-llong	10698	17.91	14.5	5.38	12.55	49.37	0.1
test-printf	983544	17.99	10.73	4.82	11.39	54.84	0.09

For test-math:

b) Computation intensive

c) Integer computation

d)11.11%, one pair per 9.00 instructions

For test-fmath:

b) Computation intensive

c) Integer computations

d) 11.36%, one pair per 8.80 instructions

For test-llong:

b) Computation intensive

c) Integer computations

d) 12.55%, one pair per 7.97 instructions

For test-printf:

b) Computation intensive

c) Integer computations

d) 11.39%, one pair per 8.78 instructions

The following are PISA Benchmarks

Benchmark	# of instruction	Load%	Store%	Uncond branch %	Condit branch%	Integer Comp%	Float Comp%
test-math	213553	15.96	10.67	4.22	13.84	54.42	0.88
test-fmath	53312	16.17	14.47	4.24	15.08	49.9	0.11
test-llong	29495	16.38	18.11	4.37	15.4	45.7	0
test-printf	1813745	19.22	9.28	5.13	17.01	49.33	0.01

For test-math:

b) Computation intensive

c) Integer computations

d)13.84%, one in 7.22 instructions

For test-fmath:

b) Computation intensive

c) Integer computations

d) 15.08%, one in 6.63 instructions

For test-llong:

b) Computation intensive

c) Integer computations

d) 15.4%, one in 6.49 instructions

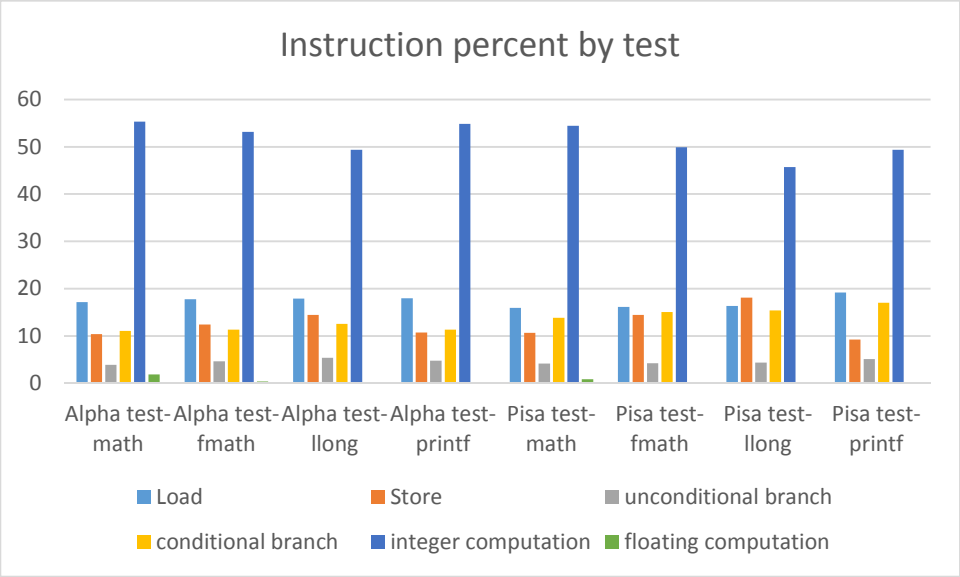
For test-printf:

b) Computation intensive

c) Integer computations

d) 17.01%, one in 5.87 instructions

Chart of average instructions for ALPHA and PISA instructions



CONCLUSIONS

The graph provides a visual illustration of the instruction distribution for the 4 test administered for both Alpha and Pisa instruction set architecture (ISA). The distribution among the six instruction types is illustrated by the color-coded vertical bars in the graph. Upon casual observation it becomes clear that the instruction sets perform very similar frequencies of the six types of instructions. In each test and for both ISA's, integer computation is responsible for the majority of instructions executed. Floating point computations consume very little of the overall execution percentage. I believe that the conclusion can be drawn that although the ALPHA and PISA ISA's are different, they perform the same operations at very similar frequencies when running the same Benchmarks.