MTCS-102(P)

ADVANCED ARCHITECTURE

Assignment Report-1

Yogen Pradhan

Reg. No - 23566

Problem Description

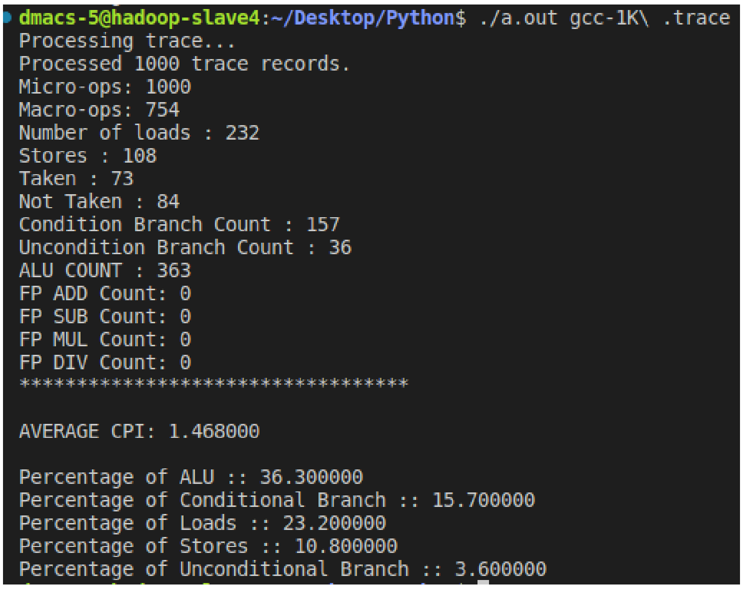
Given a few trace filese containing information about the instructions and a simulator which reads a trace file and counts the total number of micro-operations and macro-operations.

1. Find the instruction mix(category wise count of each machine instruction) of any two benchmark sets and compare the results. There are five main categories of instruction namely **Arithmetic and Logic(ALU)**,**Load and Store**, **Branch** , **FP and Others** instructions.
2. Plot a histogram of the percentage of each type of instruction (x-axis is labeled with each of these five types; the y-axis is percentage of all micro-operations).

Instruction mix: ***(Benchmark Used- Intel Silvermont)***

Si. No.	Category	Dataset1 (1K)	Dataset2 (50M)	Dataset3(100M)
1	ALU	363	19831439	27796023
2	Load	232	10500894	25272367
3	Store	108	5205971	10224954
4	Jumps	36	1931438	3914558
5	Conditional branch	157	7362160	12456836
6	Floating Point	0	0	0
	Average CPI	1.468000	1.420793	1.404344

Table1: Instruction mix for INTEL SILVERMONT

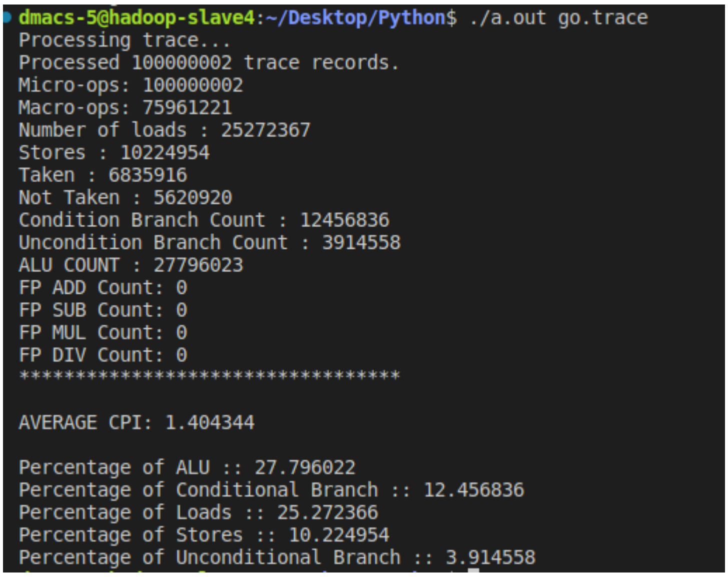


```
dmacs-5@hadoop-slave4:~/Desktop/Python$ ./a.out gcc-1K\ .trace
Processing trace...
Processed 1000 trace records.
Micro-ops: 1000
Macro-ops: 754
Number of loads : 232
Stores : 108
Taken : 73
Not Taken : 84
Condition Branch Count : 157
Uncondition Branch Count : 36
ALU COUNT : 363
FP ADD Count: 0
FP SUB Count: 0
FP MUL Count: 0
FP DIV Count: 0
**********************************

AVERAGE CPI: 1.468000

Percentage of ALU :: 36.300000
Percentage of Conditional Branch :: 15.700000
Percentage of Loads :: 23.200000
Percentage of Stores :: 10.800000
Percentage of Unconditional Branch :: 3.600000
```

Figure 1a :   1k Trace

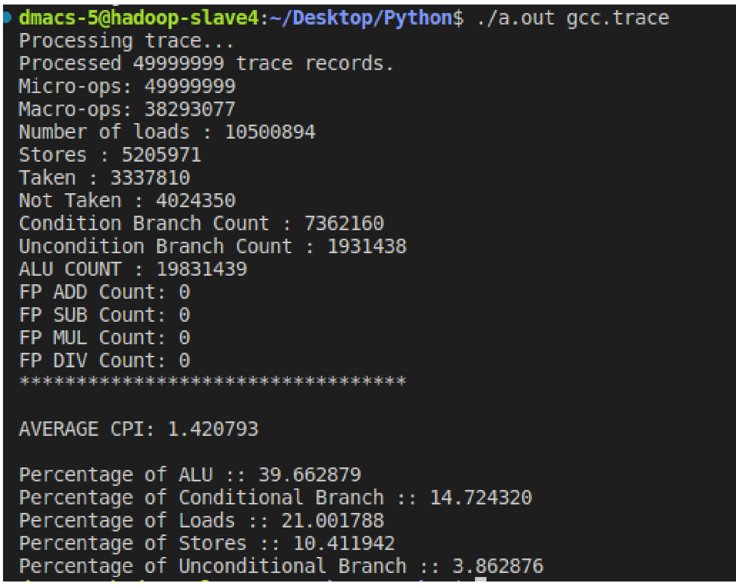


```
dmacs-5@hadoop-slave4:~/Desktop/Python$ ./a.out go.trace
Processing trace...
Processed 100000002 trace records.
Micro-ops: 100000002
Macro-ops: 75961221
Number of loads : 25272367
Stores : 10224954
Taken : 6835916
Not Taken : 5620920
Condition Branch Count : 12456836
Uncondition Branch Count : 3914558
ALU COUNT : 27796023
FP ADD Count: 0
FP SUB Count: 0
FP MUL Count: 0
FP DIV Count: 0
**********************************

AVERAGE CPI: 1.404344

Percentage of ALU :: 27.796022
Percentage of Conditional Branch :: 12.456836
Percentage of Loads :: 25.272366
Percentage of Stores :: 10.224954
Percentage of Unconditional Branch :: 3.914558
```

Figure 1b : 100M trace



```
dmacs-5@hadoop-slave4:~/Desktop/Python$ ./a.out gcc.trace
Processing trace...
Processed 49999999 trace records.
Micro-ops: 49999999
Macro-ops: 38293077
Number of loads : 10500894
Stores : 5205971
Taken : 3337810
Not Taken : 4024350
Condition Branch Count : 7362160
Uncondition Branch Count : 1931438
ALU COUNT : 19831439
FP ADD Count: 0
FP SUB Count: 0
FP MUL Count: 0
FP DIV Count: 0
**********************************

AVERAGE CPI: 1.420793

Percentage of ALU :: 39.662879
Percentage of Conditional Branch :: 14.724320
Percentage of Loads :: 21.001788
Percentage of Stores :: 10.411942
Percentage of Unconditional Branch :: 3.862876
```

Figure 1c : 50M trace

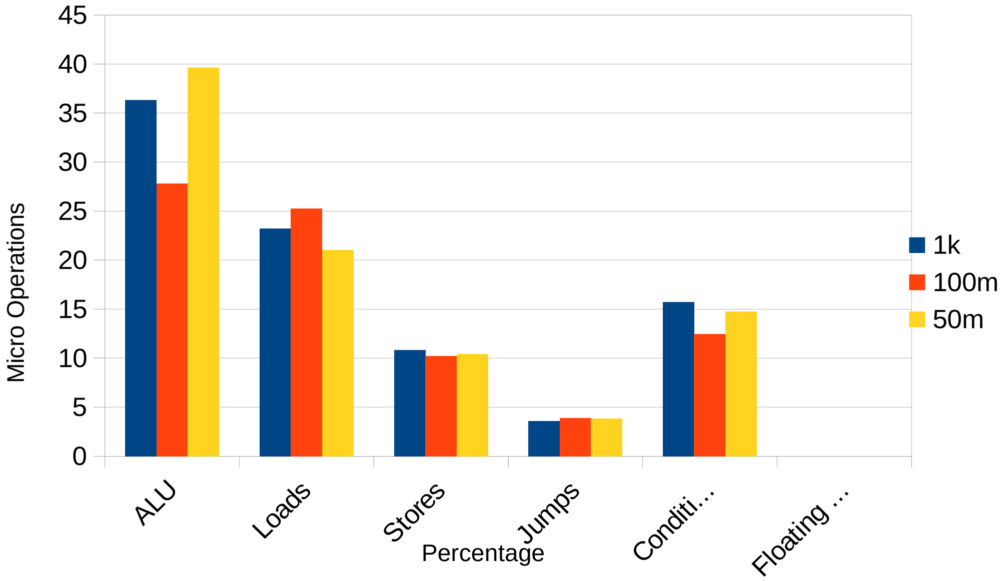


Figure 2 : Histogram of given instructions

For doing this experiment, we followed certain steps to find the instruction mix category wise for each machine instruction.Here are the steps:

1. Prepared the trace files for benchmark sets where each file recorded the sequence of executed instructions along with their relevant information.
2. We set the simulator up to read the trace files and count the total number of micro-operations and macro-operations for each benchmark set. Necessary information was extracted to calculate the instruction mix from the simulator.
3. Decided the categories which I used for instruction mix analysis.
4. Wrote a program to read and parse the trace files generated by the simulator.And then we analyzed each instruction in trace files and classified them into each categories.
5. Kept track of each micro-operations and macro-operations performed and then counted the occurrences of each instruction category.
6. We calculated the percentage of each instruction category relative to total number of micro-operations and macro-operations for each benchmark set.
7. Compared the instruction mix between these three benchmark sets.

Conclusion :

- Dataset3 has higher computational workload than other two because of highest number of ALU instructions.
- For data loading or more memory access requirements ,dataset3 is the most required.
- More complex control flow and branch behaviour for dataset3.
- Floating point operations not included in these benchmarks.

Overally, the Dataset3 have the most significant worload and complexity compared to other two datasets. So in worload complexity ,memory intensiveness and control flow ,Dataset3 exhibits more.