NC State University

Department of Electrical and Computer Engineering

ECE 463/563: Fall 2019

Project #3: Dynamic Instruction Scheduling

by

DAKSH KANDPAL

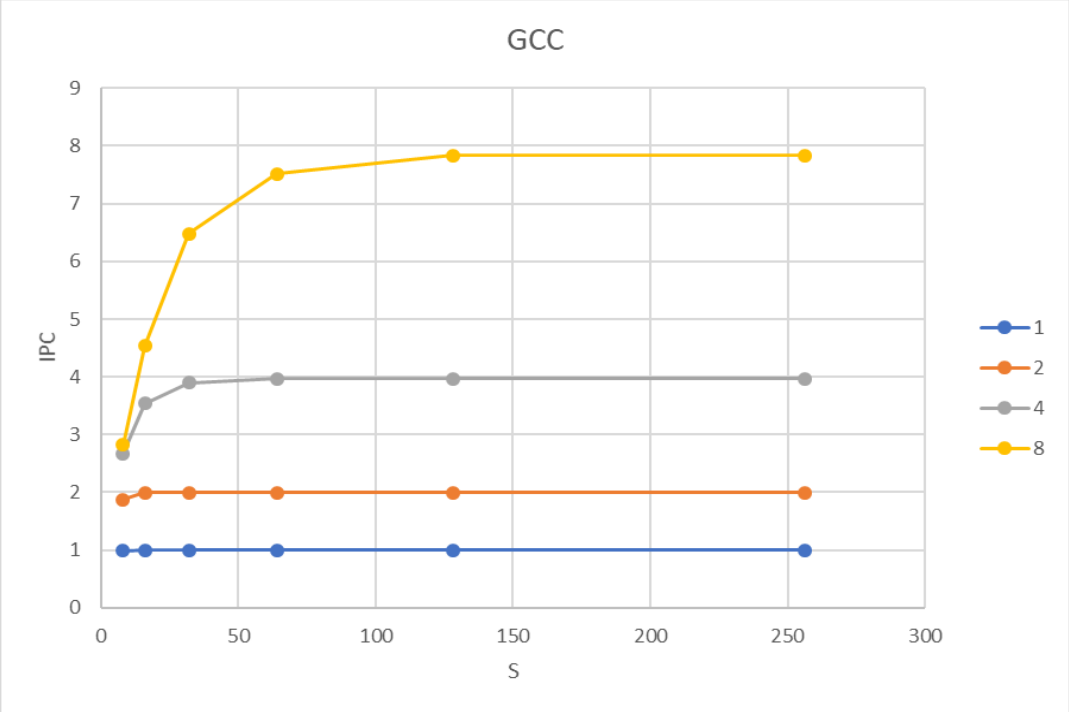
NCSU Honor Pledge: "I have neither given nor received unauthorized aid on this test or assignment."

Student's electronic signature: _____Daksh Kandpal_____
(sign by typing your name)

Course number: _____563_____
(463 or 563 ?)

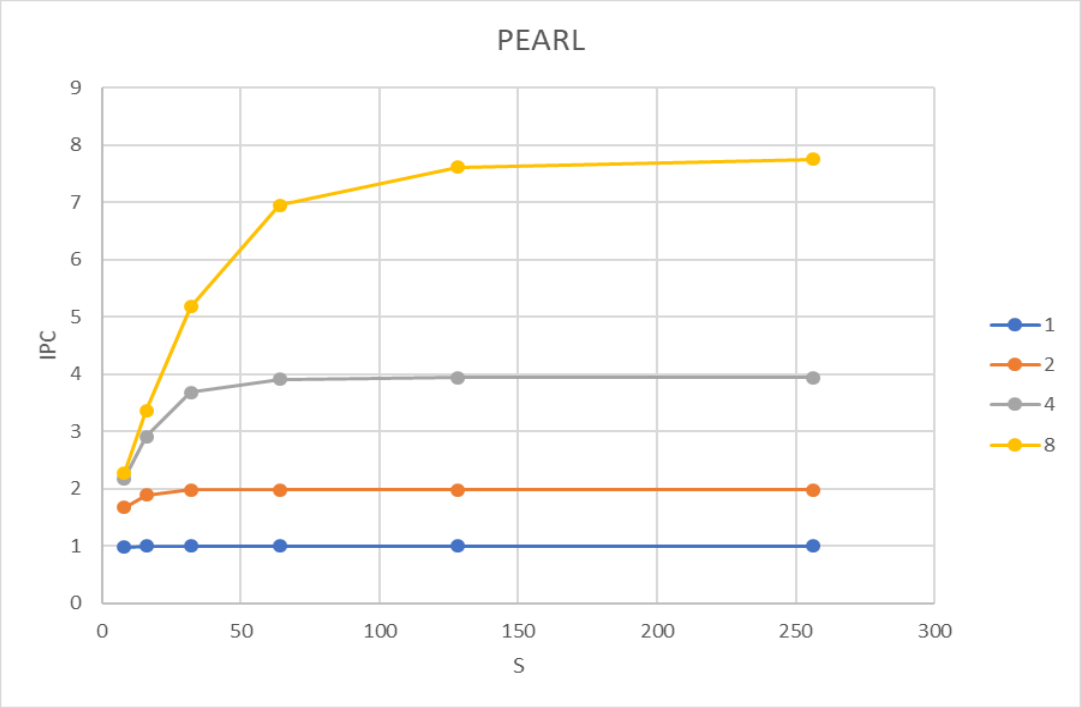
Graphs for the performed experiments:

GCC trace:



	N=1	N=2	N=4	N=8
	IPC	IPC	IPC	IPC
S=8	0.99	1.88	2.67	2.82
S=16	1	1.99	3.54	4.54
S=32	1	1.99	3.90	6.48
S=64	1	1.99	3.97	7.52
S=128	1	1.99	3.97	7.83
S=256	1	1.99	3.97	7.83

PEARL trace:



	N=1	N=2	N=4	N=8
	IPC	IPC	IPC	IPC
S=8	0.98	1.68	2.18	2.82
S=16	1	1.89	2.91	3.37
S=32	1	1.98	3.68	5.18
S=64	1	1.98	3.91	6.95
S=128	1	1.98	3.94	7.61
S=256	1	1.98	3.94	7.75

IPC vs Scheduling Size:

It is evident from the above two graphs that for the lower values of N, IPC saturates very quickly. It can also be observed that the performance of the processor in terms of the number of cycles executed improves greatly when S and N both increase. In order to achieve an optimal IPC, the dispatch bandwidth (N) needs to be above a certain threshold.This is because we need a large window of instructions to find independent instructions. A small N means that this search window will be small. As dependent instructions are placed close to each other, for a small window it'll be less likely to find independent instructions.

Relationship between scheduling queue size (S) and peak issue rate (N):

It can be inferred from the graphs above that IPC increases as S increases, but it is also influenced by N because the processor cannot execute more instructions than what it can fetch. IPC also increases on increasing N, but it is not entirely independent too and depends on S, which is quite clear from the above graphs.

Benchmark Performance:

For the same microarchitecture configuration, the IPC values are different for the given gcc and pearl benchmarks, with gcc showing slightly better performance. This could be attributed to the instruction distance between interdependent instructions. If iB is dependent on iA and iC is dependent on iB, the time iA takes to finish execution directly impacts the time iB takes and indirectly impacts the wait time for iC. If there are more multi-level interdependent instructions, the performance of the architecture might degrade.

Conclusion:

The key takeaway of this project and the accompanying project report is the understanding of out of order instruction execution using Tomasulo's algorithm for dynamic instruction scheduling. The project and the experimental runs were very instrumental in understanding the performance characteristics of the instruction scheduler and the factors affecting it.