NC State University

Department of Electrical and Computer Engineering

ECE 463/563: Fall 2017

Project #3: Dynamic Instruction Scheduling

by

SOPAN PATRA

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

Student's electronic signature: **SOPAN PATRA**
(sign by typing your name)

Course number: **563** (463 or 563 ?)

1. GCC trace :

a. TABLES

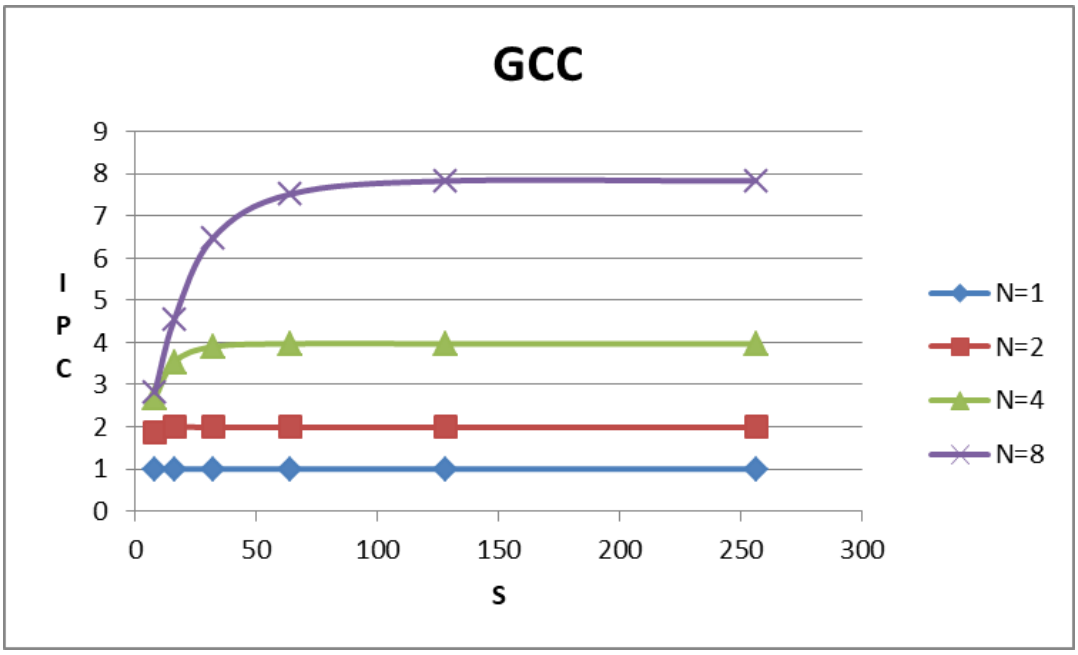
N	S	IPC
1	8	0.99
1	16	1
1	32	1
1	64	1
1	128	1
1	256	1

N	S	IPC
2	8	1.88
2	16	1.99
2	32	1.99
2	64	1.99
2	128	1.99
2	256	1.99

N	S	IPC
4	8	2.67
4	16	3.54
4	32	3.9
4	64	3.97
4	128	3.97
4	256	3.97

N	S	IPC
8	8	2.82
8	16	4.54
8	32	6.48
8	64	7.52
8	128	7.83
8	256	7.83

b. GRAPH



c. ANALYSIS:

1. We notice that till a certain S, the IPC increases with an increase in N.

   Reason: This is because as we make the processor more and more superscalar, it performs more and more operations each cycle which in effect increases the IPC
2. For lower N values, there isn't much improvement in the IPC(~1).

   Reason: This is because we restrict the bandwidth of each stage to 1, so no matter how much we increase S, the scalar nature of the processor will always restrict the IPC to be close to 1. This also suggests that having a big reservation station doesn't make sense if we have a scalar or low bandwidth superscalar processor
3. For higher values of N, we see an increase in the IPC with increasing S .

   Reason: This happens because if we are able to store more instructions in the schedule queue, it allows more instructions to be dispatched in the dispatch cycle. This also increases the probability of some low latency instruction to be dispatched and completed, which was earlier getting blocked in the dispatch stage due to a smaller reservation station size.
4. For higher N values, we also notice the IPC value saturating to N, with an increasing value of S.

   Reason: This happens because the instruction bandwidth is restricted to N per cycle. So no matter how much we increase S, once the instruction bandwidth is reached, the processor will only be able to process N instructions per cycle and not able to fully utilize the large space available in the reservation station

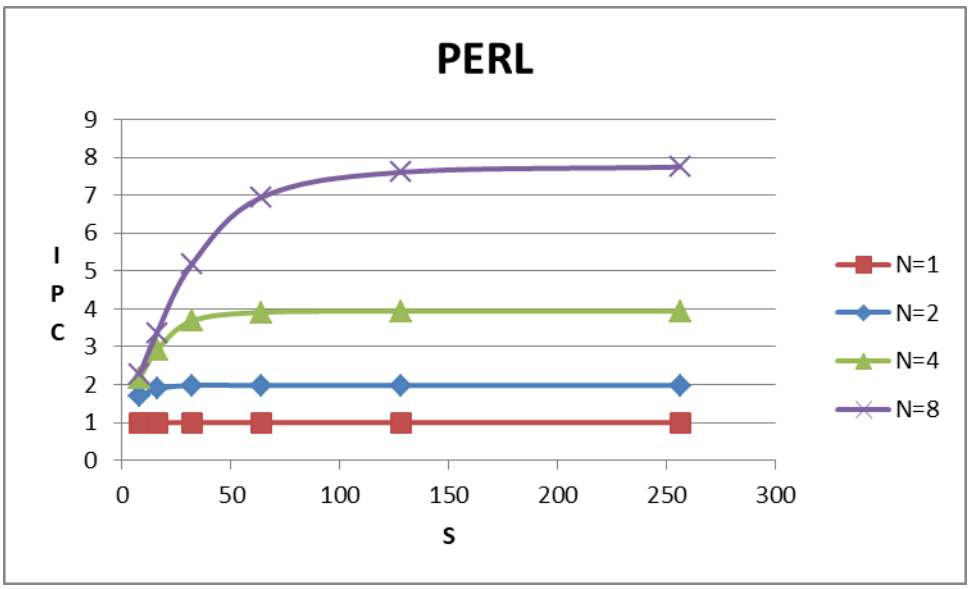
2. PERL trace:

N	S	IPC
1	8	0.98
1	16	1
1	32	1
1	64	1
1	128	1
1	256	1

N	S	IPC
2	8	1.68
2	16	1.89
2	32	1.98
2	64	1.98
2	128	1.98
2	256	1.98

N	S	IPC
4	8	2.18
4	16	2.91
4	32	3.68
4	64	3.91
4	128	3.94
4	256	3.94

N	S	IPC
8	8	2.28
8	16	3.37
8	32	5.18
8	64	6.95
8	128	7.61
8	256	7.75



ANALYSIS:

1. We notice that for a certain S, the IPC increases with an increase in N.

   Reason: This is because as we make the processor more and more superscalar, it performs more and more operations each cycle which in effect increases the IPC
2. For lower N values or a nearly scalar processor(N~1), there isn't much improvement in the IPC which stays approximately close to 1.

   Reason: This is because we restrict the bandwidth of each stage to 1, so no matter how much we increase S, the scalar nature of the processor will always restrict the IPC to be close to 1. This also suggests that having a big reservation station doesn't make sense if we have a scalar or low bandwidth superscalar processor
3. For higher values of N, we do see an increase in the IPC with increasing S .

   Reason: If we are able to store more instructions in the schedule queue, it allows more instructions to be dispatched in the dispatch cycle. This also increases the probability of some low latency instruction to be dispatched and completed, which was earlier getting blocked in the dispatch stage due to a smaller reservation station size.
4. For higher N values, we also notice the IPC value asymptotically saturating to N, with an increasing value of schedule queue size.

   Reason: This happens because the instruction bandwidth is restricted to N per cycle. So no matter how much we increase S, once the instruction bandwidth is reached, the processor will only be able to process N instructions per cycle and not able to fully utilize the large space available in the reservation station

CONCLUSION:

1. For lower values of S, there is not much increase in IPC with an increasing N, since a smaller issue queue restricts the number of independent instructions that can be buffered and issued out of order. So here S is the dominant parameter governing the value of IPC
2. Similarly the IPC saturates to the value of N with increasing issue queue size as N becomes the dominant parameter when S is very large
3. **Benchmark comparison:** We notice a slightly higher IPC for the GCC trace when compared to PERL trace. This is because the instructions in the PERL trace have more true dependences compared to the GCC stage. Due to this, if we have a sufficiently large reservation station and instruction bandwidth, we can execute more instructions in the GCC trace compared to the PERL trace as they are independent.