CS 575

Project #3

Name: Chi Wen
ID: 933-276-677
Email: wench@oregonstate.edu

1. **Tell what machine you ran this on**

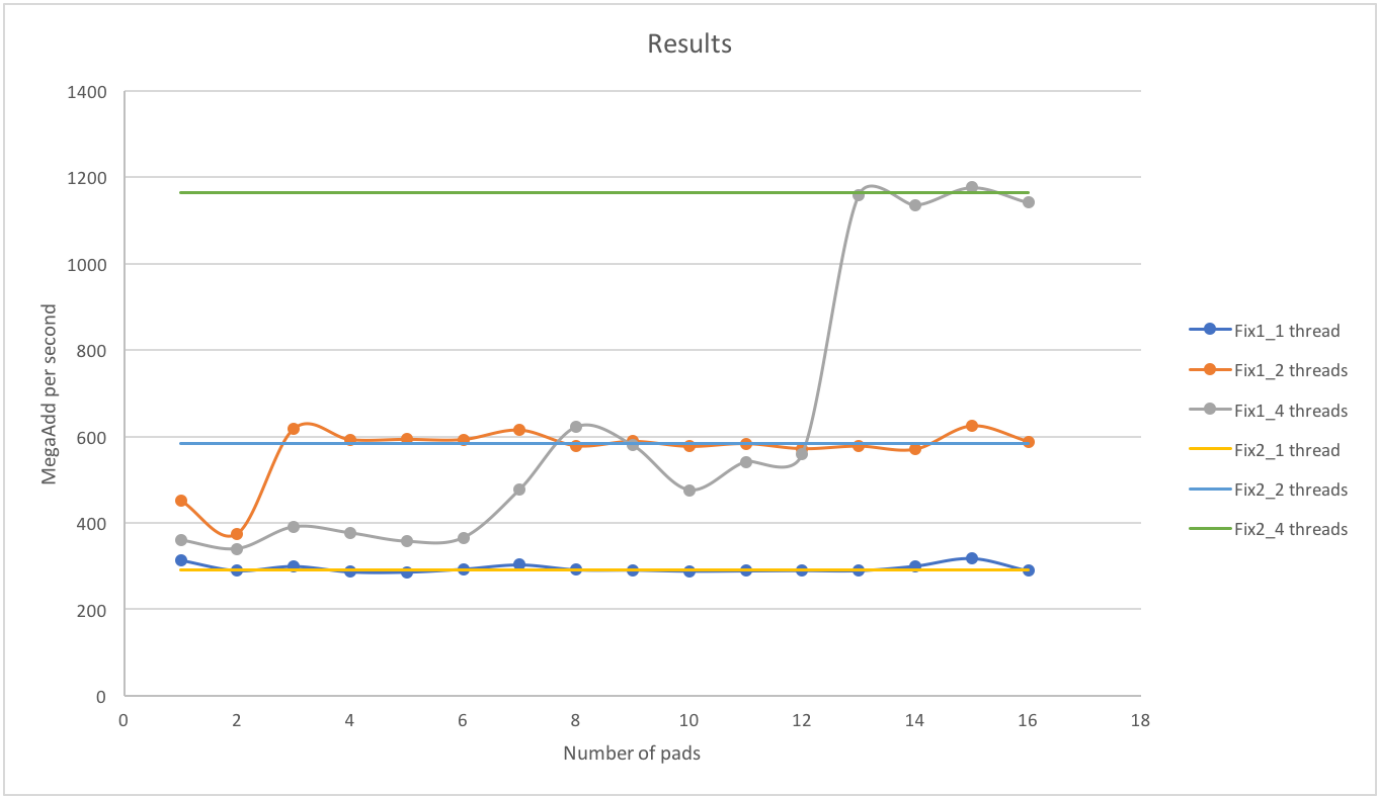
   School server, flip1.

2. **Create a table with your results.**

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Fix1_1 thread	314.3	290.03	299.89	286.92	286.43	293.35	303.44	291.96	291.18	288.71	289.35	290.11	289.94	300.48	318.4	289.28
Fix1_2 threads	451.13	374	617.52	592.56	593.13	592.74	614.66	578.3	588.49	576.77	583.23	571.44	577.35	570.23	624.3	588.26
Fix1_4 threads	359.83	339.99	390.48	376.28	356.81	365.2	476.97	622	579.36	476.04	541.21	558.62	1158.65	1135.85	1176.26	114.53
Fix2_1 thread	290.82	290.82	290.82	290.82	290.82	290.82	290.82	290.82	290.82	290.82	290.82	290.82	290.82	290.82	290.82	290.82
Fix2_2 threads	583.21	583.21	583.21	583.21	583.21	583.21	583.21	583.21	583.21	583.21	583.21	583.21	583.21	583.21	583.21	583.21
Fix2_4 threads	1163.39	1163.39	1163.39	1163.39	1163.39	1163.39	1163.39	1163.39	1163.39	1163.39	1163.39	1163.39	1163.39	1163.39	1163.39	116.39

3. **Draw a graph. The X axis will be NUM, i.e., the amount of integers used to pad the structure. The Y axis will be the performance in whatever units you sensibly choose. There should be at least 6 curves shown together on those axes:**

   1-3: Using padding with 1, 2, and 4 threads.
   4-6: Using a private variable with 1, 2, and 4 threads



4. **What patterns are you seeing in the performance?**

   The three curves indicate Fix#1's 1, 2 and 4 threads, and the yellow, light blue and green straight lines are from Fix#2 program. 1 thread's performance is similar to the fix#2's 1 thread. However, we can see that the other two curves all have a really low performance at the beginning, but as the padding increases, they went up quickly at one point. For 2 threads, it quickly raised up as the pads increased to 3, and it remained its performance, similar to fix#2's 2 threads. When using 4 threads, it went up quickly when the padding is 13; however, 15 pads will get the highest performance.

5. **Why do you think it is behaving this way?**

   1 thread doesn't need to share the cache line with others. Because there was no false sharing, its performance is close to the fix#2's 1 thread's performance. As for 2 and 4 threads, it is because I ran this on flip, which will have a lot students using and running programs at that time, the cache line might not be empty, so it might have some effect on the results.