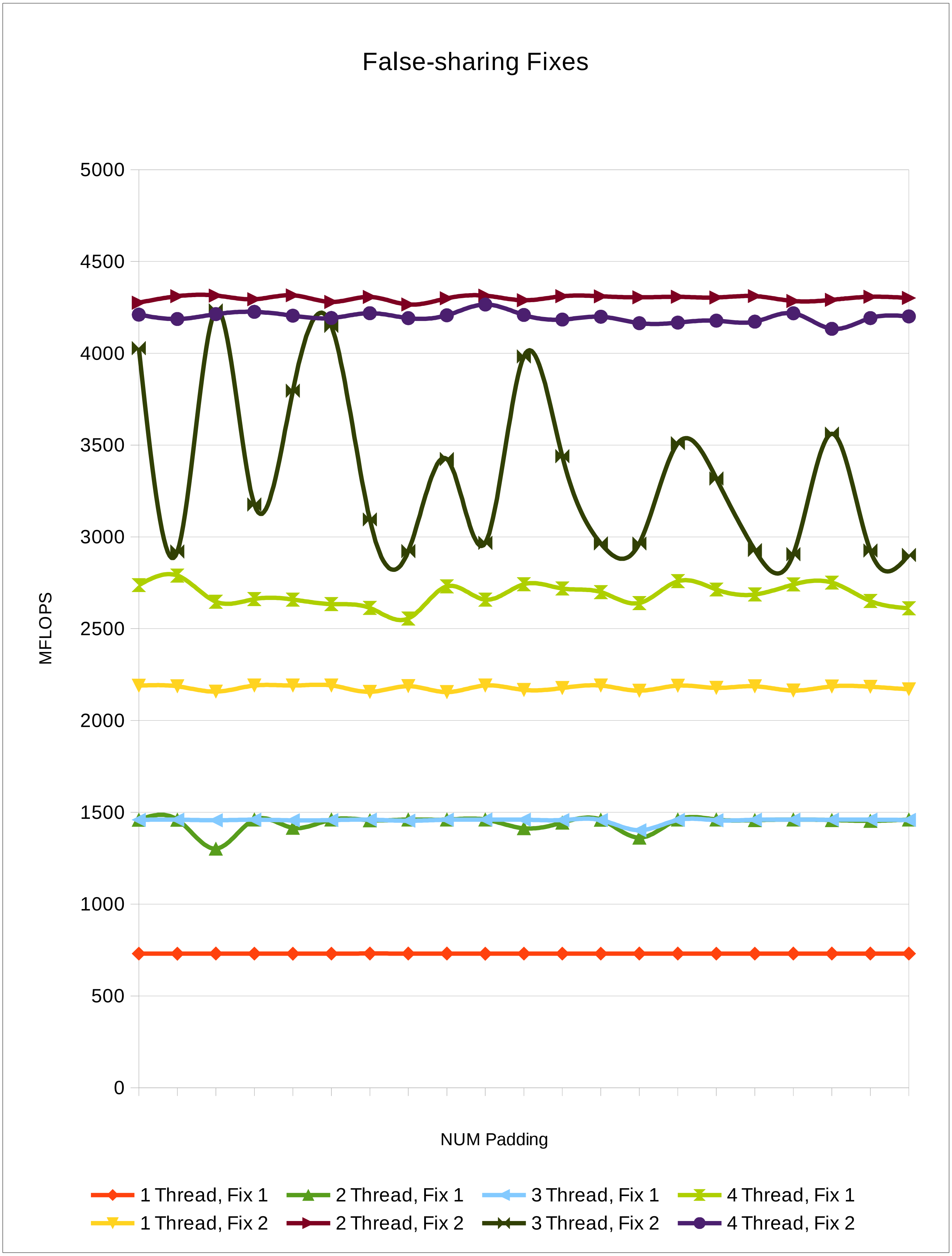
Geoffrey Corey
CS475 Project 3

Testing Environment:

- Personal Laptop:
  - Core i3-2330m @ Cores, 4 Threads
  - Ubuntu 14.04 with Unity
  - In order to allow the system to “settle down”, I added a sleep 2 command before the execution of each process.

Test Results:

[redacted]	Threads							
Padding	1 Thread, Fix 1	1 Thread, Fix 2	2 Thread, Fix 1	2 Thread, Fix 2	3 Thread, Fix 1	3 Thread, Fix 2	4 Thread, Fix 1	4 Thread, Fix 2
0	730.4037690295	2189.8153876138	1459.7868857126	4275.6146730544	1458.7578166626	4028.0148432347	2737.3973647076	4210.3534696997
1	730.1771062826	2186.2989020407	1459.8721298002	4311.027716675	1459.9664426713	2920.2430518292	2789.4100049163	4186.6886328264
2	730.5773277746	2157.4786841106	1301.9540703153	4314.4547176405	1456.6668912362	4233.319398657	2647.2079236228	4213.2582811592
3	730.4787320204	2190.4965308011	1460.4467214432	4294.4120037404	1459.8486866836	3175.8710024716	2661.3422080225	4225.6050538237
4	730.1144655206	2190.3849765875	1415.6059943836	4315.7581276515	1456.2320360126	3795.8262991927	2658.844346599	4204.7370567682
5	730.4709949143	2190.9356609834	1460.2942346854	4279.635674615	1457.6462656925	4149.9243157553	2633.8454406819	4192.0372252906
6	730.9059945255	2156.0889839485	1456.9220918196	4306.6507607699	1459.8774578862	3095.0409706049	2612.8864949042	4218.2523780398
7	730.7123897766	2186.9001401256	1460.4851147357	4265.4473174602	1454.0235922598	2922.6861824165	2555.443541947	4191.3300242783
8	730.5920060084	2155.0830057158	1460.2889035567	4299.041313787	1459.7772963757	3423.9569129262	2730.5245337629	4206.8464322261
9	730.0570320553	2190.7328713443	1460.5149775811	4313.8126122939	1460.2633146809	2967.6078187563	2658.1923494566	4265.4154781264
10	730.4419155328	2166.6430141471	1412.9866900186	4288.6566104817	1459.8167200108	3982.4295209536	2742.1488849737	4208.6612143461
11	730.4205743407	2176.6966125703	1443.9640857253	4310.7489603012	1458.0150000583	3439.7742820116	2718.717417871	4183.5796590173
12	730.5004768342	2190.1798904253	1460.0426478457	4310.4423699243	1457.7843864003	2964.3067820375	2699.3995860471	4199.003366551
13	730.7544400183	2162.7080998284	1361.969189533	4304.5513097135	1402.4596337056	2963.9487503622	2639.7726627783	4163.9514233427
14	730.7294762752	2189.8333701043	1460.3624765703	4307.3371180469	1458.1457489583	3510.1790805612	2758.443768779	4167.0399639968
15	730.4159061211	2177.7880859747	1460.4584525128	4303.7222894081	1456.7735234326	3318.2436204693	2713.5348404306	4177.6722219263
16	730.3029533999	2186.4387231417	1457.3908113695	4311.4087420279	1459.3762552916	2928.3906004519	2686.2746809881	4172.173065911
17	730.6783526491	2163.2028381221	1460.2723773052	4284.4213085694	1460.0676960387	2905.6379546052	2740.720946645	4217.8476112747
18	730.5846668178	2185.6645540397	1457.8619579669	4291.104219121	1459.5967207241	3561.967096329	2751.1886854614	4133.1622204174
19	730.6612685412	2183.9103679507	1453.6410255292	4307.5087415506	1459.6723546449	2925.4671422501	2650.7515543344	4191.5803725058



Note: I don't know why the graph dropped the x-axis values. In my excel sheet version, the x-axis numbers are there. The range is [0-20].

Patterns:

From the data gathered, there does not seem to be much of a performance difference as the NUM padding increases but rather the performance seems to stay mostly constant overall. The difference seems to be dependent upon the number of threads used.

One noticeable anomaly is when NUM=3 in Fix 2. The results are very sporadic. This is most likely due to the architecture of the CPU in my laptop and having a single thread be idle for general OS use that can cause unneeded pre-emption on the working threads.

A noticeable pattern is that the MFLOPS of Fix 2 are consistently higher than the MFLOPS for Fix 1 when compared to the same thread counts.

Another interesting note is that 2 Threads with Fix 2 out performed 4 Threads with Fix 2. This could be due to the architecture of my CPU in that it has 2 physical cores and each core has Hyper-threading which presents to the system as 2 “virtual” cores. This hyper-threading can effect the outcome of benchmarks like this with the actual components that the physical cores has (like the number of FPUs the core has).

Theory:

One reason the difference in NUM padding might not be noticeable is that I have noticed that using Ubuntu Unity on my laptop instead of a different Desktop Environment, the background usage tends to be relatively noisy. This causes more pre-emption scheduling than in other environments and can effect the outcome of this test.

Another issue is that I change the the sum variable to be volatile so that the compiler did not optimize out the for loop since the end result of the loop is not used. Without this, there was a large anomaly for NUM=1 and NUMT=1 where the MFLOPS was somewhere in the range of 800 TFLOPS. Another way to fix the compiler optimization is to change the optimization level from -O{2,3} to -O0. However, this caused the timing of the 2nd fix to go from roughly 1.6s (with the volatile keyword) to over 16s.