CS475: Project 3

Trevor Bramwell

May 5, 2015

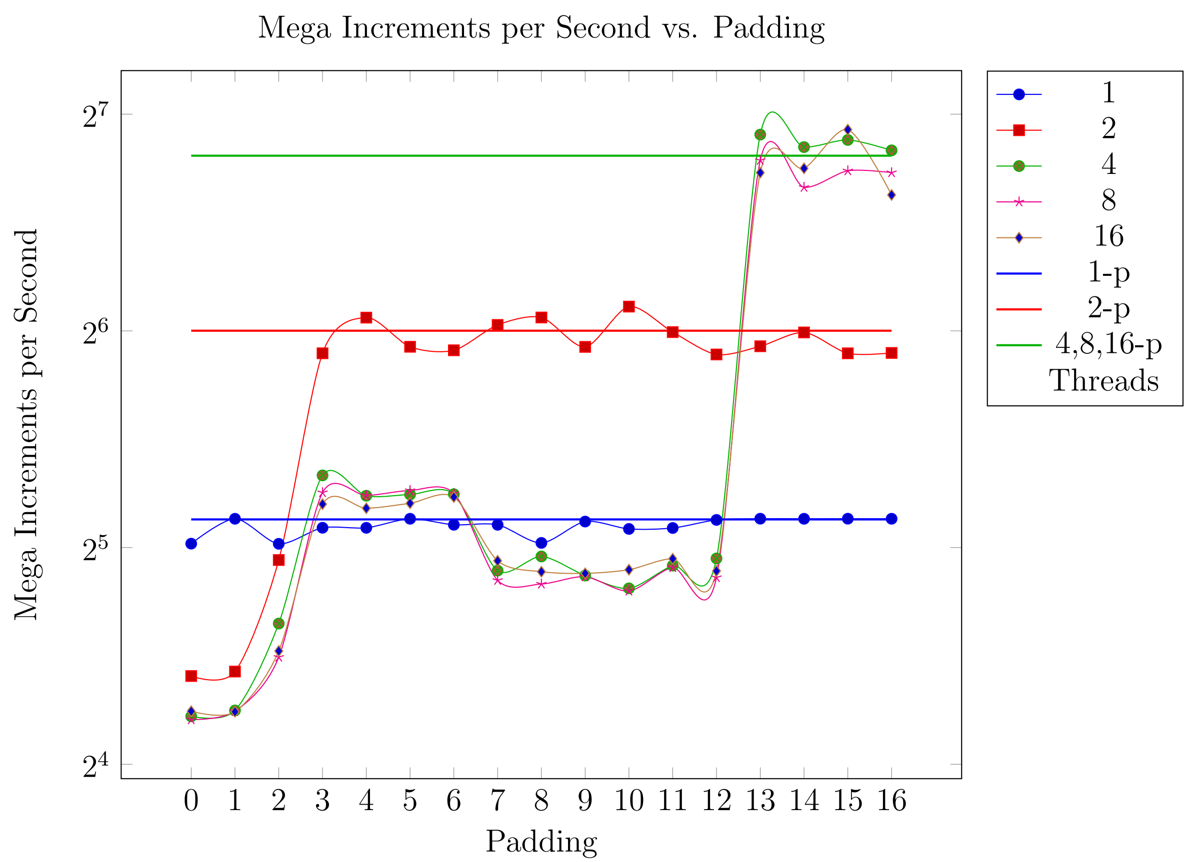
False Sharing

This project used OpenMP to show how false-sharing can affect the performance of parallel applications. These insights were gained by writing a program that performed computations on a C struct with variable end-padding. The program was compiled using *gcc* (without compiler optimizations) and ran on the *rabbit.engr.oregonstate.edu* Xeon processor with 32 CPUs (not the Xeon Phi). It used the following combination of parameters:

- Padding Size (in integers):
  - 0
  - 1
  - 2
  - 3
  - 4
  - 5
  - 6
  - 7
  - 8
  - 9
  - 10
  - 11
  - 12
  - 14
  - 15
  - 16
- Threads:
  - 1
  - 2
  - 4
  - 8
  - 16

Along with the combinations of the above resources, the program was also ran with a private variable maintained by each thread for accumulation. This put a stop to the false-sharing, and showed a more consistent improvement in performance when more threads were added.

Graph



Patterns

Private Variables The lines 1-p, 2-p, and 4,8,16-p are the average performance per thread using private variables. They are baselines for the best average-performance for those numbers of threads.

One Thread With a single thread there is no performance gain as it is being ran on a single CPU and can't take advantage of spatial coherence.

Two Threads With two threads, performance stays constant after a padding of 3-4. I believe this is because the number of cache lines is greater than 2, and being limited to two threads, the program can't take advantage of the other cache lines.

4,8,16 Threads The following is an explanation of the performance of the program while padding is increased each step of the way.

0: Without any padding, performance was abysmal (worse than a single thread) even with 16 threads. This is the most important data point since it helps characterize the rest.

1: There is the slightest increase in performance.

2: Performance is continuing to increase, though the greatest comes with two threads. Remember that the Xeon has 2 CPU sockets. This same performance can be seen in the 4,8, and 16 threads, but at a lower gain because of the cache-miss overhead.

3-6: Performance peaks and begins to stabilize.

7: There is a distinct drop in performance as false-sharing has started. With 4 or more threads each cache line is reloaded with the same data multiple times.

8-12: There is little change in performance.

13-16: The cache lines are now all being taken advantage of, false-sharing has stopped, and the best performance has been reached.

There is difference in the amount of padding needed before false-sharing stops, between this graph and the one in the notes. The performance increases at 13, while the notes show an increase at 15. This is because of the Xeon most likely has a slightly smaller 4-Way L1 cache than the machine the notes used.

Tables

NUM	NUMT	PERF	TIME
0	1	32.3884	12.3501s
1	1	35.07	11.4058s
2	1	32.3727	12.3561s
3	1	34.0742	11.7391s
4	1	34.0726	11.7396s
5	1	35.0725	11.405s
6	1	34.4015	11.6274s
7	1	34.401	11.6276s
8	1	32.4631	12.3217s
9	1	34.7607	11.5072s
10	1	33.9478	11.7828s
11	1	34.0513	11.747s
12	1	34.94	11.4482s
13	1	35.0815	11.402s
14	1	35.0714	11.4053s
15	1	35.0785	11.403s
16	1	35.0736	11.4046s

Table 1: Padding - 1 Thread

NUM	NUMT	PERF	TIME
0	2	21.2069	18.8618s
1	2	21.5146	18.592s
2	2	30.7336	13.0151s
3	2	59.5413	6.71803s
4	2	66.7354	5.99382s
5	2	60.7943	6.57957s
6	2	60.1138	6.65404s
7	2	65.1904	6.13587s
8	2	66.7568	5.9919s
9	2	60.7737	6.58179s
10	2	69.1342	5.78585s
11	2	63.7171	6.27775s
12	2	59.3136	6.74382s
13	2	60.8943	6.56876s
14	2	63.647	6.28467s
15	2	59.5327	6.719s
16	2	59.6032	6.71105s

Table 2: Padding - 2 Threads

NUM	NUMT	PERF	TIME
0	4	18.643	21.4558s
1	4	18.9978	21.0551s
2	4	25.0836	15.9467s
3	4	40.2898	9.92808s
4	4	37.7572	10.594s
5	4	37.9033	10.5532s
6	4	37.9384	10.5434s
7	4	29.7119	13.4626s
8	4	31.0881	12.8666s
9	4	29.2485	13.6759s
10	4	28.0729	14.2486s
11	4	30.1886	13.25s
12	4	30.8928	12.948s
13	4	119.824	3.33823s
14	4	115.119	3.47465s
15	4	117.79	3.39588s
16	4	113.938	3.51067s

Table 3: Padding - 4 Threads

NUM	NUMT	PERF	TIME
0	8	18.4491	21.6813s
1	8	18.9462	21.1124s
2	8	22.5273	17.7562s
3	8	38.1387	10.488s
4	8	37.782	10.5871s
5	8	38.392	10.4188s
6	8	38.0468	10.5134s
7	8	28.8034	13.8873s
8	8	28.4657	14.052s
9	8	29.1355	13.729s
10	8	27.8339	14.371s
11	8	30.006	13.3307s
12	8	29.0607	13.7643s
13	8	110.353	3.62473s
14	8	101.234	3.95126s
15	8	106.736	3.74758s
16	8	106.101	3.76999s

Table 4: Padding - 8 Threads

NUM	NUMT	PERF	TIME
0	16	18.9503	21.1078s
1	16	18.923	21.1383s
2	16	22.9774	17.4084s
3	16	36.7524	10.8837s
4	16	36.2713	11.028s
5	16	36.8519	10.8543s
6	16	37.5977	10.6389s
7	16	30.6643	13.0445s
8	16	29.6074	13.5101s
9	16	29.4533	13.5808s
10	16	29.7997	13.423s
11	16	30.8653	12.9595s
12	16	29.6794	13.4774s
13	16	106.074	3.77093s
14	16	107.565	3.71869s
15	16	121.744	3.28559s
16	16	98.7983	4.04865s

Table 5: Padding - 16 Threads

Tables - Private Sum

NUM	NUMT	PERF	TIME
0	1	34.3947	11.6297s
1	1	31.4988	12.6989s
2	1	35.0762	11.4037s
3	1	35.0786	11.403s
4	1	35.0789	11.4029s
5	1	33.0348	12.1085s
6	1	34.0649	11.7423s
7	1	33.61	11.9012s
8	1	32.1739	12.4324s
9	1	35.0788	11.4029s
10	1	34.4016	11.6274s
11	1	35.0792	11.4028s
12	1	35.0787	11.4029s
13	1	35.0808	11.4022s
14	1	34.401	11.6276s
15	1	34.3965	11.6291s
16	1	34.0274	11.7552s

Table 6: Padding - 1 Thread - Private Sum

NUM	NUMT	PERF	TIME
0	2	60.2753	6.63622s
1	2	60.6648	6.59361s
2	2	59.7313	6.69665s
3	2	66.6845	5.9984s
4	2	63.5273	6.29651s
5	2	59.4386	6.72964s
6	2	64.7578	6.17687s
7	2	65.1093	6.14352s
8	2	63.9279	6.25705s
9	2	63.5375	6.29549s
10	2	66.7765	5.99013s
11	2	61.2848	6.5269s
12	2	63.084	6.34075s
13	2	60.0052	6.66608s
14	2	60.7519	6.58416s
15	2	59.6553	6.70518s
16	2	60.3798	6.62473s

Table 7: Padding - 2 Threads - Private Sum

NUM	NUMT	PERF	TIME
0	4	103.685	3.85782s
1	4	91.8025	4.35718s
2	4	101.258	3.95029s
3	4	103.1	3.87975s
4	4	102.234	3.9126s
5	4	101.976	3.9225s
6	4	103.385	3.86904s
7	4	102.916	3.88665s
8	4	103.089	3.88014s
9	4	103.197	3.87607s
10	4	107.438	3.72306s
11	4	103.235	3.87466s
12	4	107.463	3.72221s
13	4	124.427	3.21473s
14	4	107.747	3.71238s
15	4	125.768	3.18045s
16	4	119.878	3.33672s

Table 8: Padding - 4 Threads - Private Sum

NUM	NUMT	PERF	TIME
0	8	95.0553	4.20808s
1	8	88.3615	4.52686s
2	8	102.719	3.89412s
3	8	102.557	3.90027s
4	8	101.912	3.92497s
5	8	102.063	3.91915s
6	8	84.0855	4.75706s
7	8	107.811	3.71019s
8	8	104.796	3.81693s
9	8	107.466	3.72212s
10	8	90.1571	4.4367s
11	8	101.511	3.94045s
12	8	102.968	3.88469s
13	8	102.28	3.91085s
14	8	103.273	3.87321s
15	8	106.845	3.74376s
16	8	106.548	3.75418s

Table 9: Padding - 8 Threads - Private Sum

NUM	NUMT	PERF	TIME
0	16	108.223	3.69608s
1	16	103.515	3.86418s
2	16	95.166	4.20318s
3	16	103.62	3.86025s
4	16	107.856	3.70864s
5	16	102.225	3.91293s
6	16	104.835	3.81551s
7	16	105.167	3.80348s
8	16	103.976	3.84705s
9	16	105.246	3.80061s
10	16	102.479	3.90323s
11	16	101.987	3.92207s
12	16	102.094	3.91795s
13	16	107.091	3.73513s
14	16	120.715	3.31359s
15	16	124.035	3.22489s
16	16	102.44	3.90473s

Table 10: Padding - 16 Threads - Private Sum