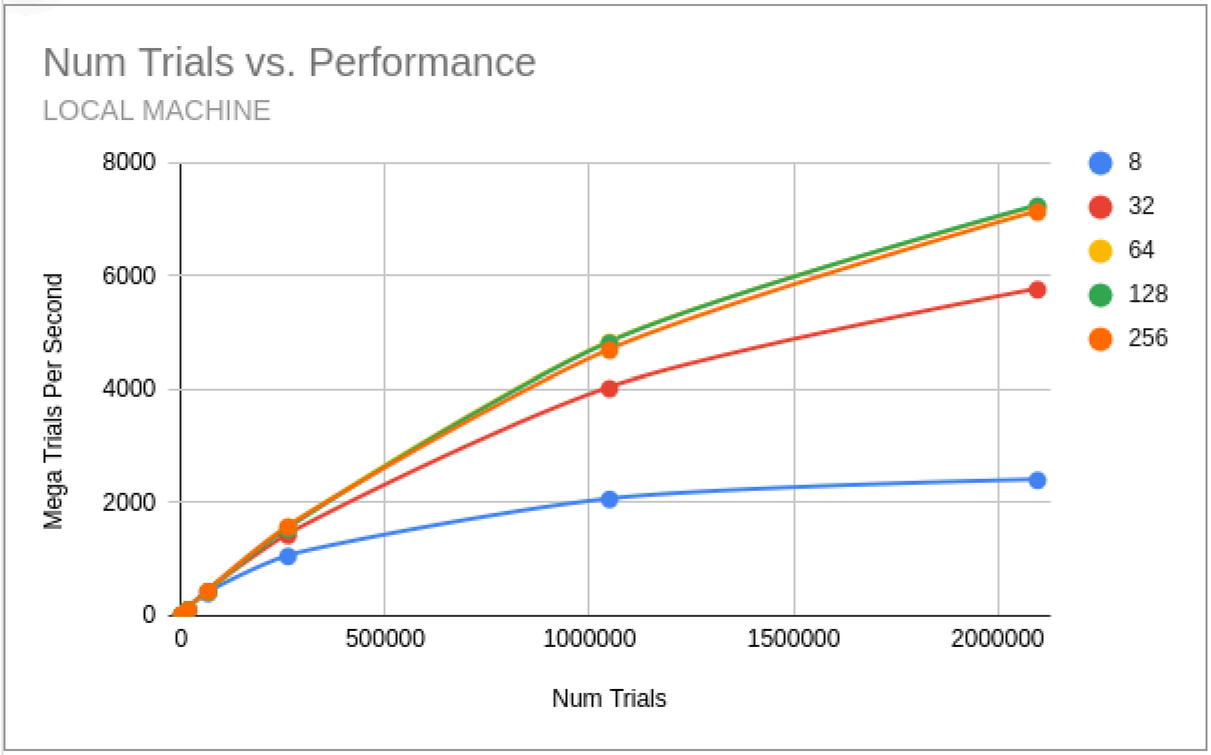
CS_475 Project #5

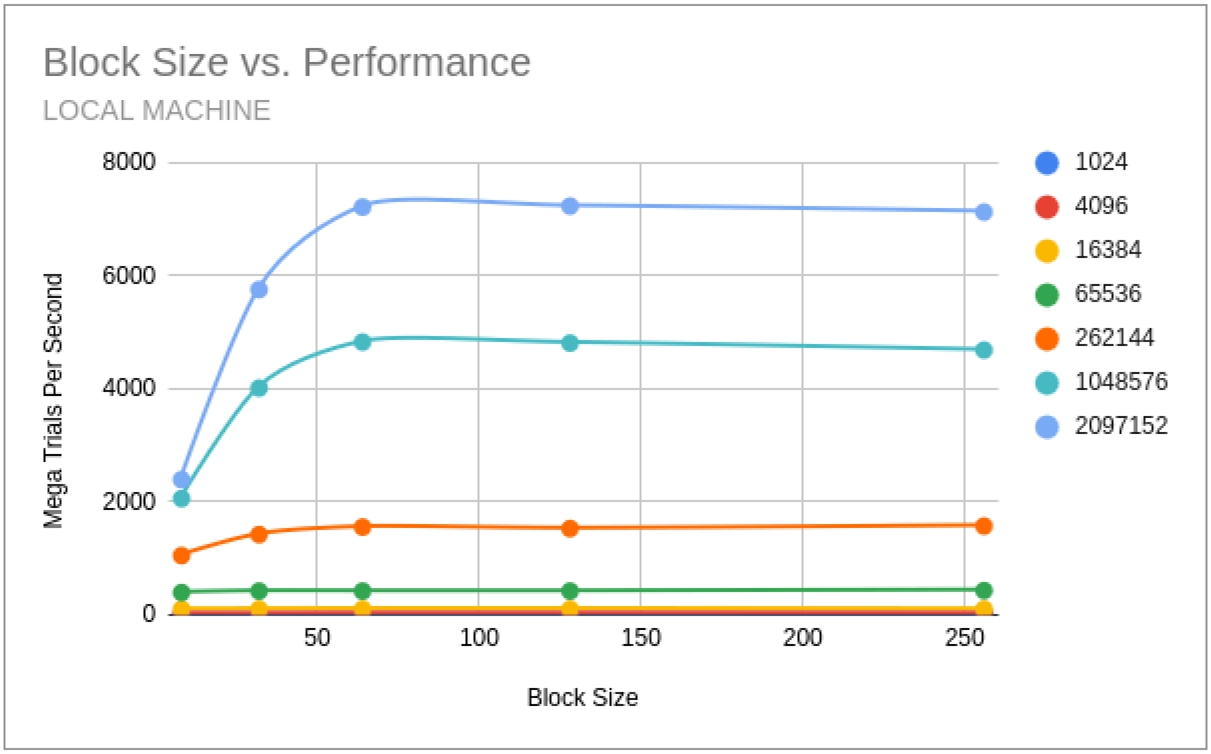
Benjamin Anderson II
anderbe2@oregonstate.edu

1. I ran this on both ‘rabbit’ and my personal machine. My personal machine has a 1660ti with cuda version 12.4.
2. I believe the new probability is around 83.75%. I got this by averaging the probabilities calculated by both my personal machine (83.753…) and rabbit (83.806…).
3. .

NUMTRIALS	BLOCKSIZE	megaTrialsPerSecond	Probability
1024	8	6.9025	82.52
1024	32	6.5293	81.84
1024	64	6.7568	85.45
1024	128	6.8817	84.18
1024	256	6.8143	84.08
4096	8	27.3797	82.74
4096	32	26.4901	83.76
4096	64	27.6936	83.45
4096	128	27.4149	83.64
4096	256	27.4325	84.62
16384	8	103.875	83.72
16384	32	105.3498	83.45
16384	64	108.1995	83.83
16384	128	107.8803	83.78
16384	256	105.2632	83.92
65536	8	396.515	83.87
65536	32	419.9303	83.82
65536	64	419.2426	84.11
65536	128	419.1568	83.77
65536	256	434.3584	83.65
262144	8	1050.9301	83.81
262144	32	1420.9888	83.72
262144	64	1553.2802	83.81
262144	128	1526.0805	83.79
262144	256	1572.6627	83.77
1048576	8	2057.3868	83.89
1048576	32	4018.1482	83.8
1048576	64	4828.0535	83.84
1048576	128	4813.8681	83.79

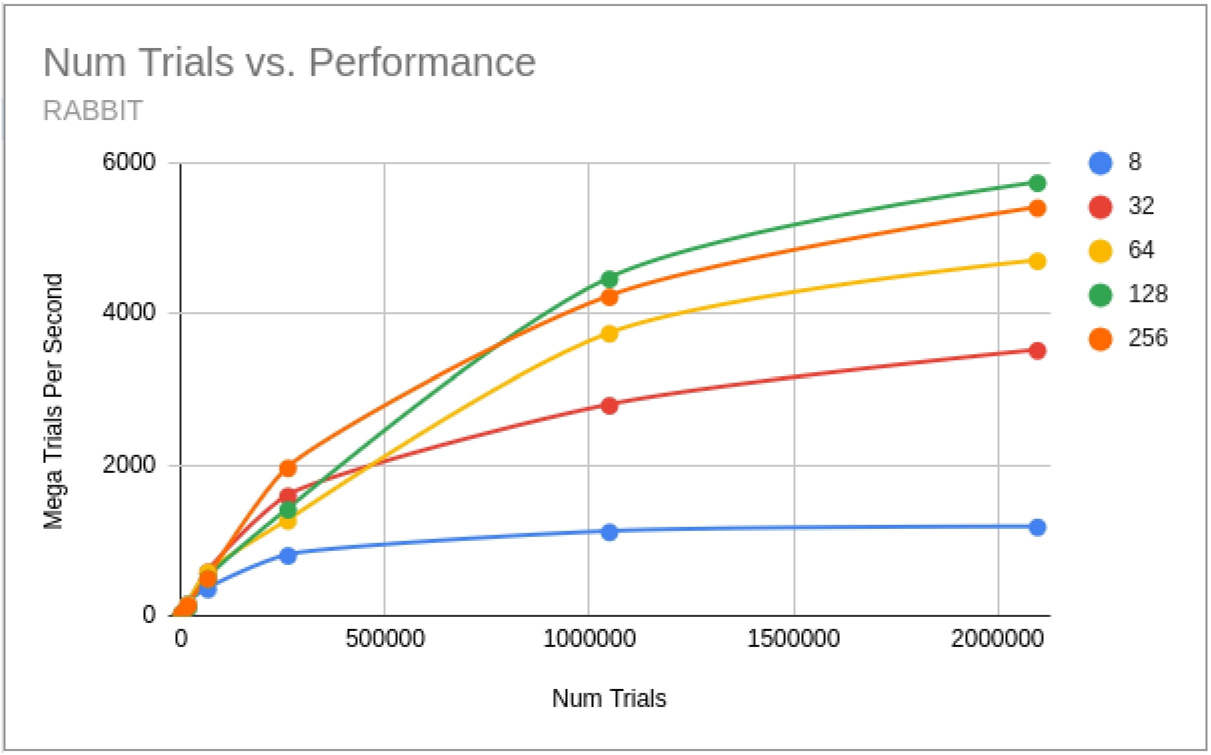
1048576	256	4685.1586	83.84
2097152	8	2398.4775	83.82
2097152	32	5754.8296	83.81
2097152	64	7220.802	83.82
2097152	128	7235.1516	83.85
2097152	256	7133.5584	83.81

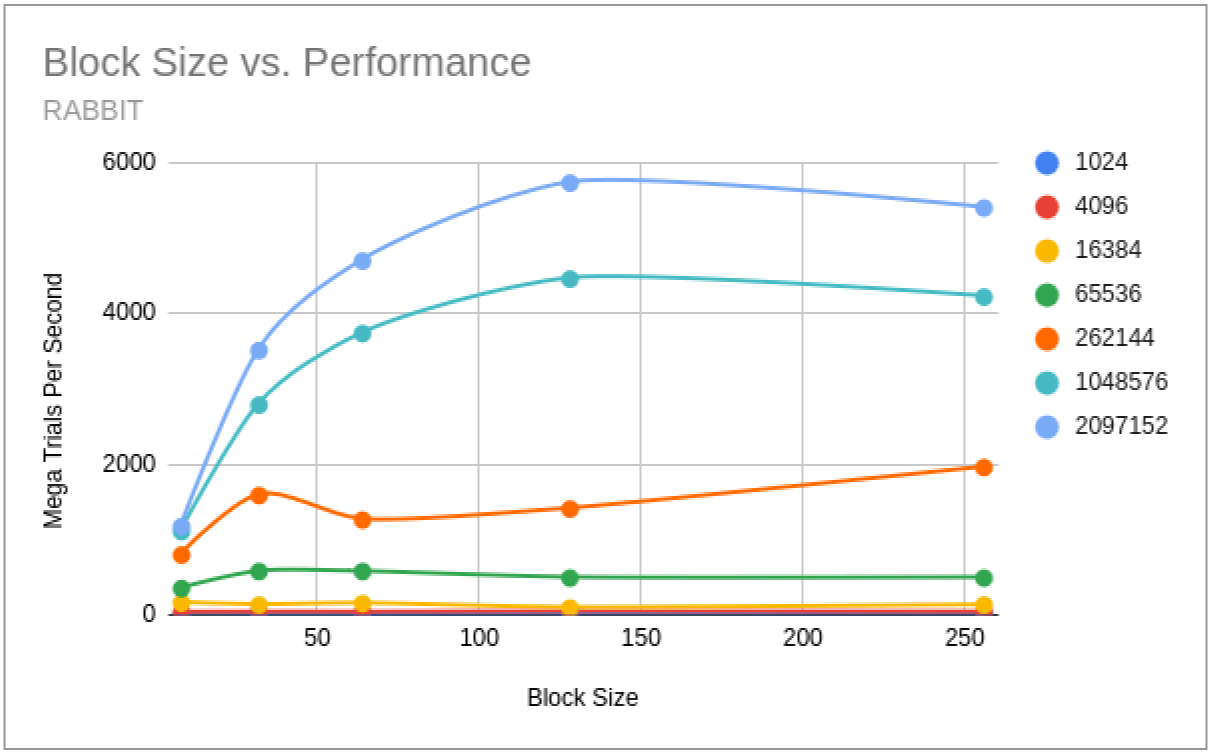




NUMTRIALS	BLOCKSIZE	megaTrialsPerSecond	Probability
1024	8	8.45	83.89
1024	32	6.1776	83.79
1024	64	2.2286	82.62
1024	128	11.1459	85.55
1024	256	8.3595	83.4
4096	8	33.8266	83.37
4096	32	35.7442	83.72
4096	64	35.0493	83.94
4096	128	40.9469	83.2
4096	256	37.1877	83.98
16384	8	159.601	84.05
16384	32	132.2997	83.75
16384	64	149.3582	83.72
16384	128	93.0571	84.14
16384	256	128.3852	83.87
65536	8	350.8652	83.77
65536	32	573.991	83.82
65536	64	572.7069	83.76

65536	128	492.4261	83.85
65536	256	491.9529	83.78
262144	8	797.9739	83.78
262144	32	1586.3671	83.91
262144	64	1261.0837	83.88
262144	128	1403.4607	83.84
262144	256	1953.2667	83.73
1048576	8	1110.2151	83.8
1048576	32	2787.1055	83.84
1048576	64	3732.5438	83.76
1048576	128	4460.6588	83.84
1048576	256	4222.1362	83.76
2097152	8	1171.6666	83.79
2097152	32	3507.0369	83.84
2097152	64	4700.2798	83.8
2097152	128	5731.1761	83.85
2097152	256	5399.2422	83.84





4. .
   a. On my personal machine:
      i. I see that as the number of trials increases, as well as the size of each block, the performance only increases.
      ii. The block sizes above and at 64 all seem to perform the same.
      iii. The performance for two billion trials is always better than any other number of trials.
   b. On rabbit:
      i. Similar to my personal machine, as the number of trials increases the performance only increases. This indicates that we are likely still not getting maximum performance out of the program.
      ii. The block size of 128 seems to have the best average performance for the higher trial counts, but the lower number of trials seem to be less deterministic. This might be due to others using the server at the same time or any other number of things.
      iii. The performance for two billion trials is always better than any other number of trials.
5. .
   a. Local machine: I would say the patterns are like this just because the block size and number of trials haven't hit max performance yet. The lines on the graph are only going up. As for the clumping of performance for block sizes of 64 and up, I would say this is probably due to the complexity of the task more than anything. The 2 warps are likely enough to get the job done without any more swapping in.
   b. For the rabbit machine there is likely more traffic, so increasing the number of warps actually does help. I don't know why the 256 would be worse than the 128

though. Maybe, there were more people using the server, so trying to switch into a space that hasn’t necessarily been fully guaranteed to me? I don’t know.

6. For a block size of 8 you’re only using a 4th of a warp, which is inefficient–to put it lightly.
7. The performance for my project 1 topped out at around 600 to 800 mega trials per second, while this one tops out at around 6,000 to 8,000 mega trials per second, meaning there was roughly a 10x increase. Reason being the way GPUs are put together. Since the threads all just run a copy of the instruction, rather than running their own mini processes they can perform simple instructions considerably quicker than a CPU thread.
8. This, put simply, means I can probably 10x my performance if I just parallelize using my GPU rather than my CU, which is pretty amazing.