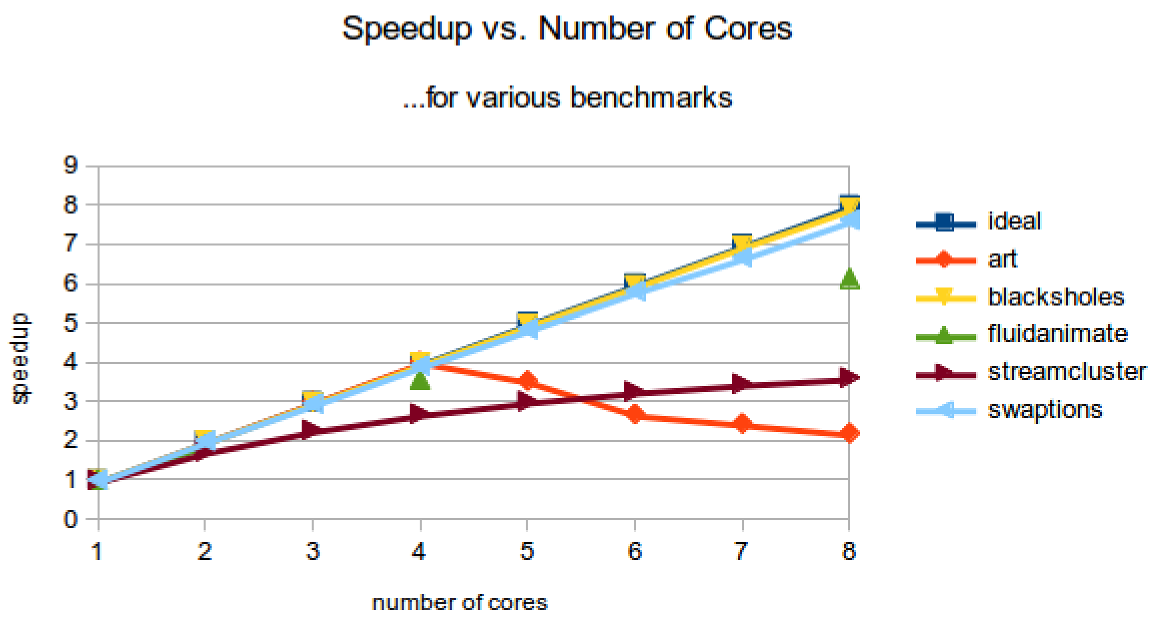
Andrew J. Dupree and Anthony Vivoli
Dr. Kozyrakis
EE282: Computer Systems Architecture
6/5/2013

PROGRAMMING ASSIGNMENT 2

PROBLEM 1

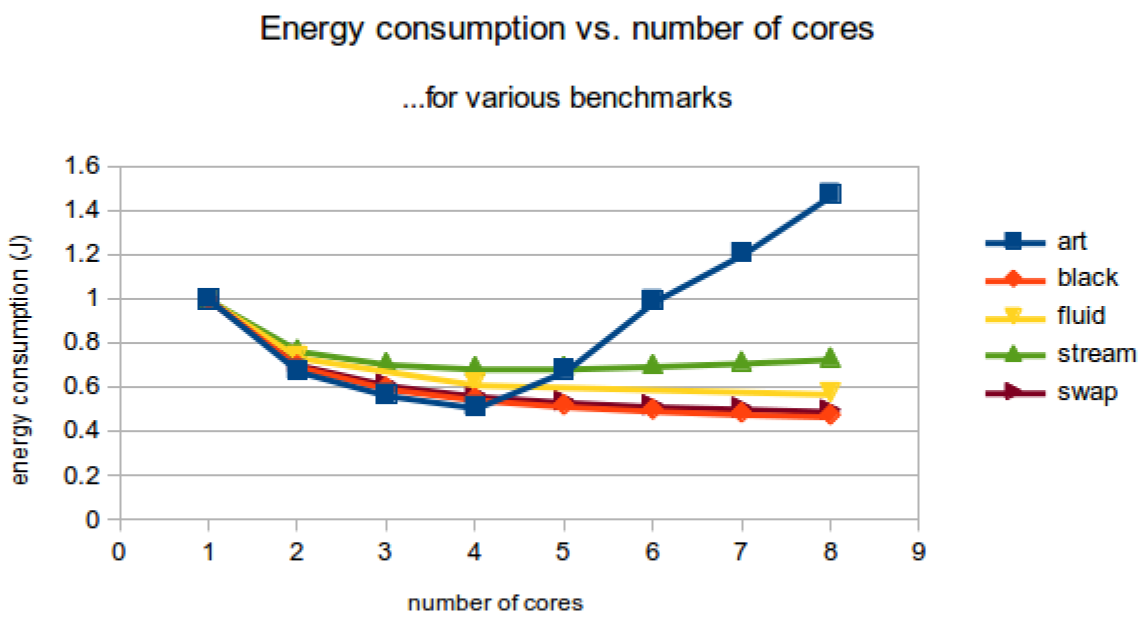
(a)

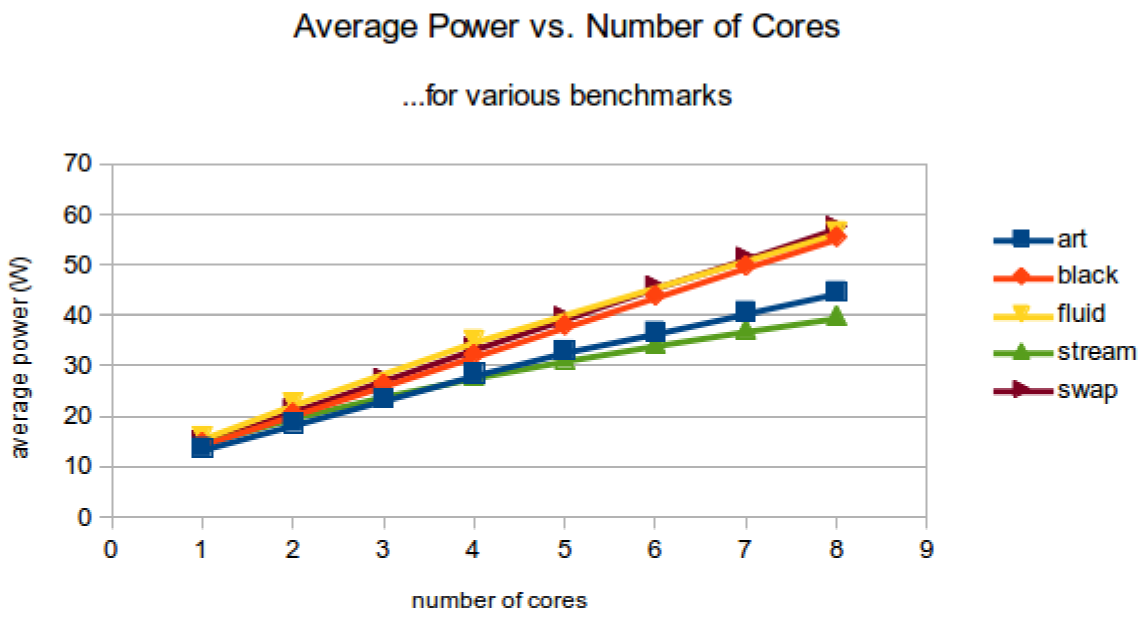


(b)
The cycle count and instruction count is much higher for core 0 than any of the other cores. This is likely because a large part of the algorithm must be done sequentially, so not all the work can be evenly distributed amongst cores. This is a common issue when trying to parallelize an algorithm.

(c)
As the number of cores increases past four, the number of L3 misses (and therefore memory accesses) dramatically increases. This is likely due to the fact that the L3 needs to be larger to accommodate the increase in cores, but since the L3 size is constant for these simulations, the different cores start thrashing each other's data in the L3. If the L3 is made to be larger, the performance decrease after the fourth core would likely be minimized or possible reversed.

(d)





Energy is the product of average power and execution time. Although power increases as the number of cores increases, the execution time decreases. As long as the execution time decreases more than the power increases, energy will be lower with more cores.

(e)
Five cores gives the lowest energy consumption for streamcluster. As cores are increased further, the energy consumed increases slightly. This is because the power consumption scales linearly with

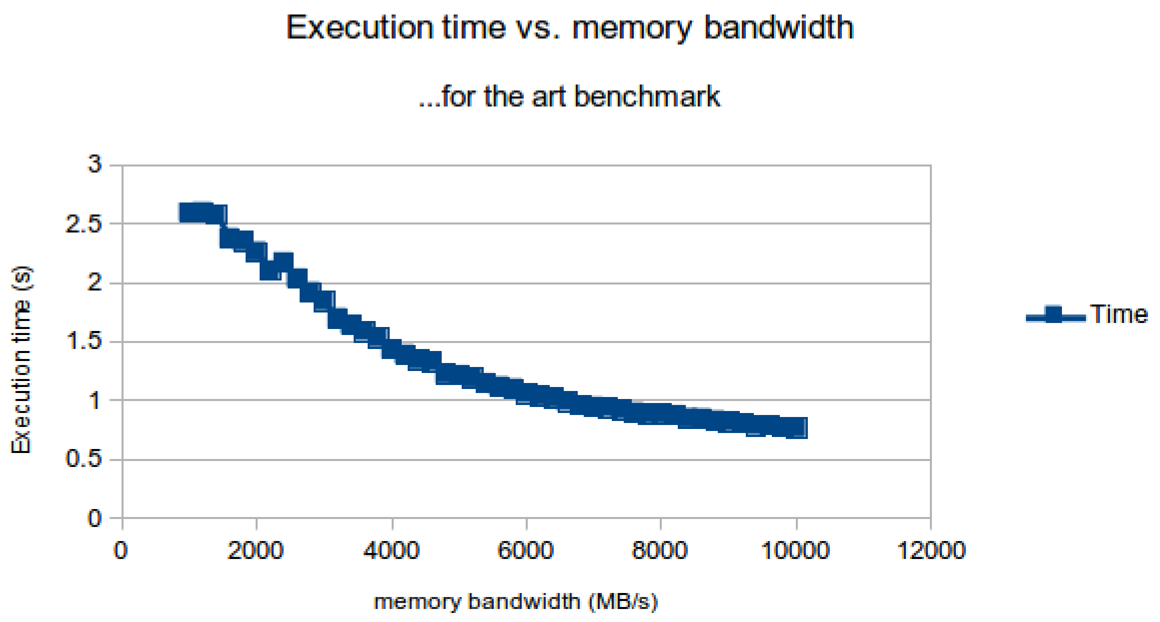
the number of cores, but the execution time has diminishing returns and scales sub-linearly. This means that although adding extra cores will yield higher performance, it comes at the cost of slightly higher energy (a wise man once said "There ain't no such thing as a free lunch").

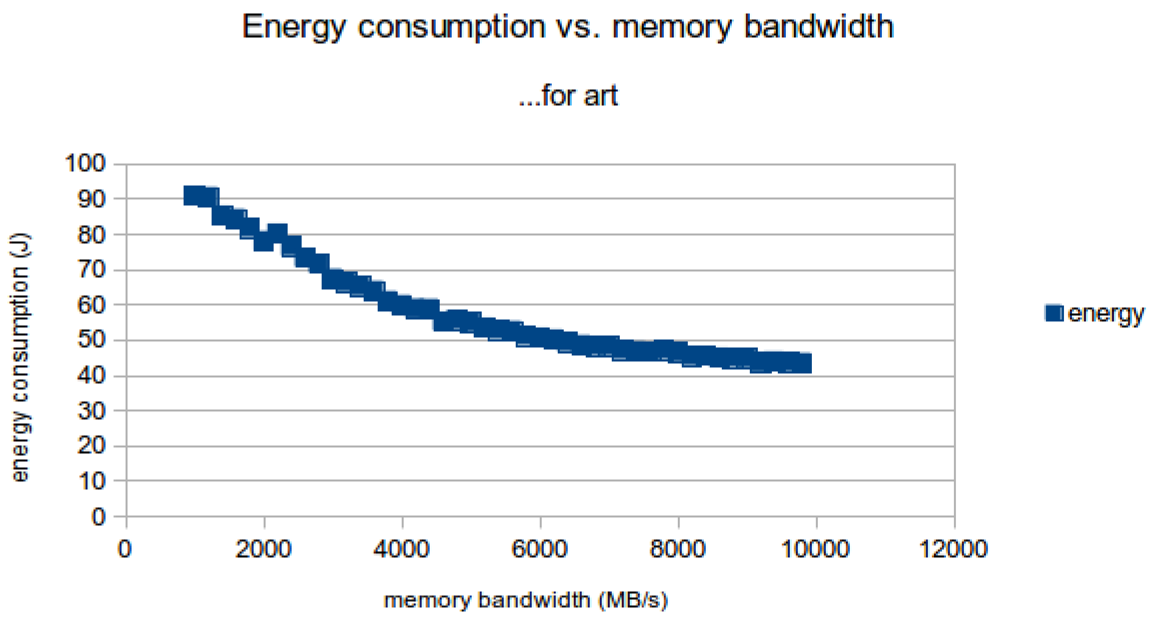
PROBLEM 2

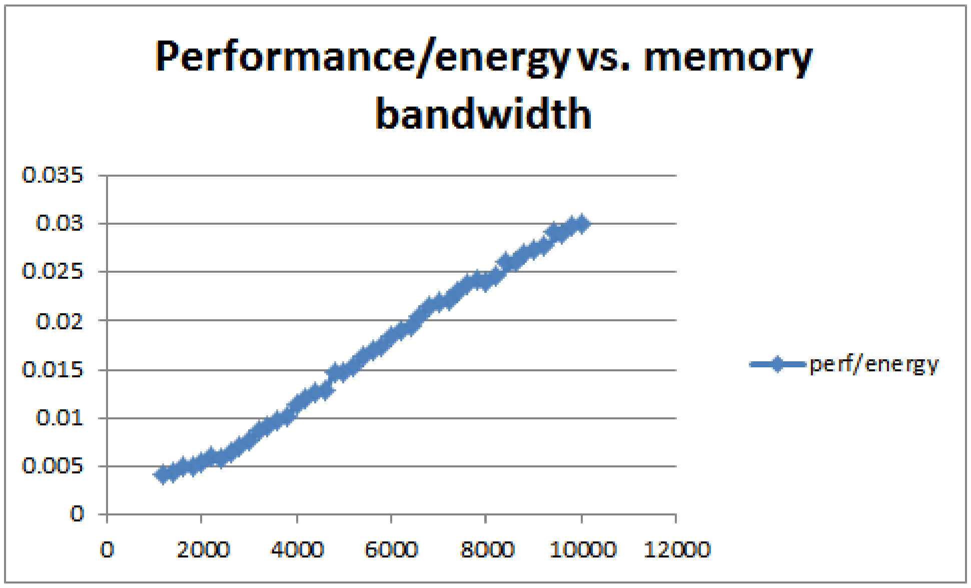
(a)

The dynamic energy remains rather constant, but the static energy begins to increase rapidly above four cores. A possible explanation is that the amount of work being done remains constant as the number of cores increases, so the dynamic energy shouldn't change that much, but because the execution time and area are both increasing the static energy will increase. Static energy consumed is a function of execution time and chip area, so increasing both will increase static energy.

(b)





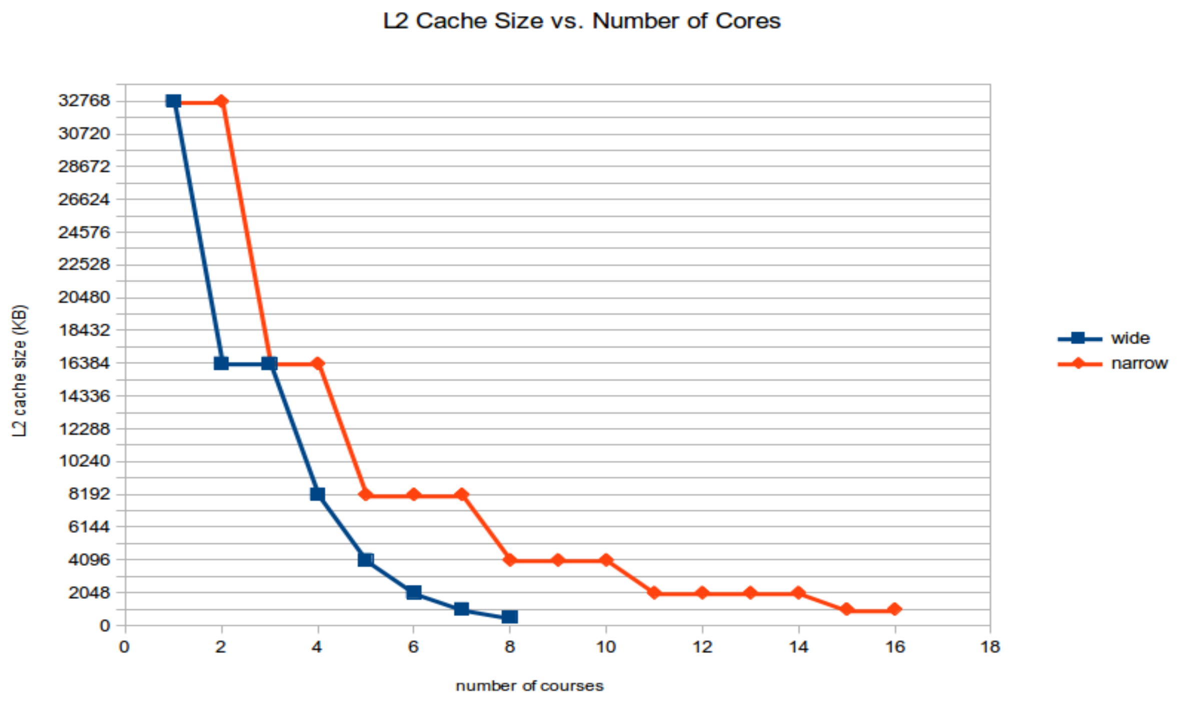


Performance, energy consumption, and performance/energy all improve with higher memory bandwidth (A maximum memory bandwidth of 10 GB/s is best for all three scenarios).

In looking at the graphs, we can see that at each increase in memory bandwidth, the improvement is less than the previous step. As such, while increasing memory bandwidth always improves performance, energy, and performance/energy, it also always has diminishing returns.

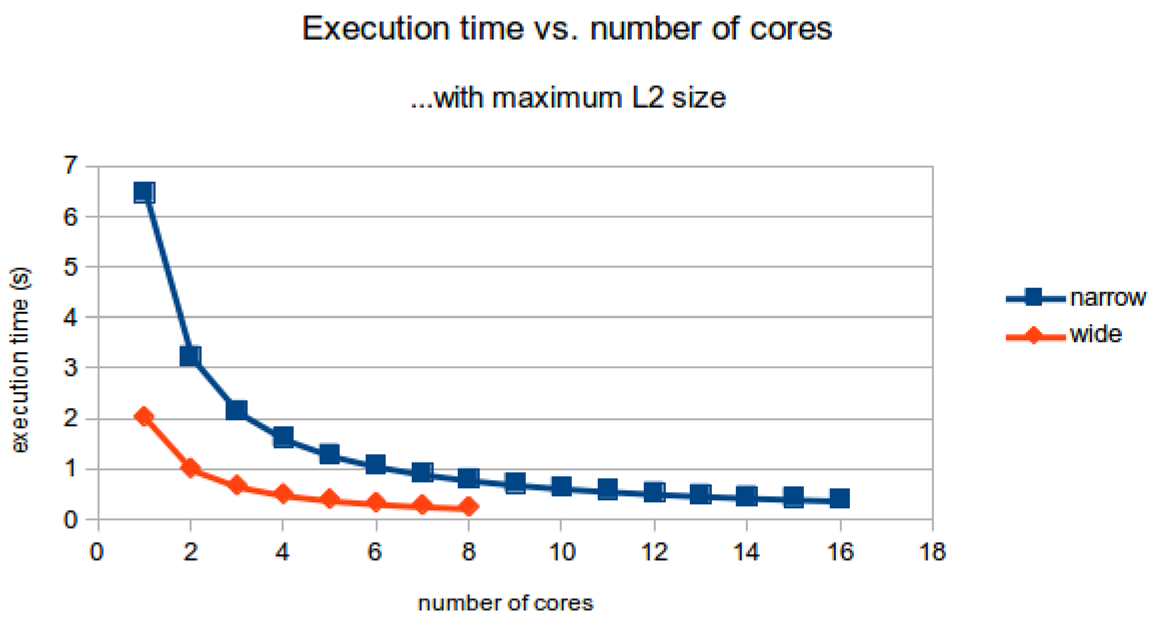
PROBLEM 3

(a)

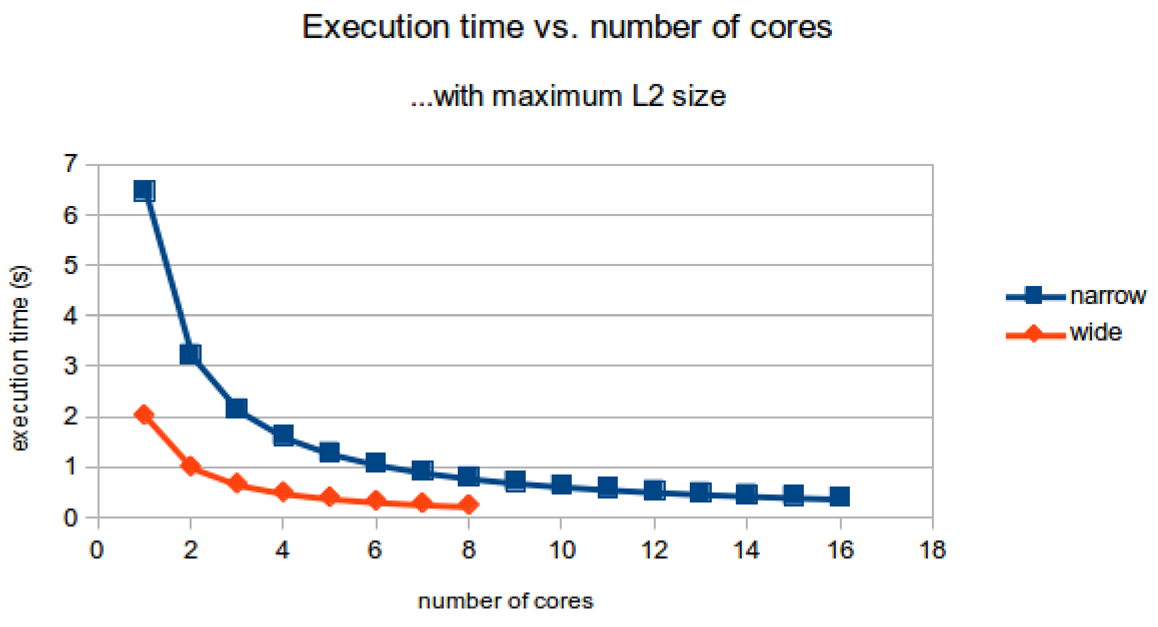


cores	wide	narrow
1	32768	32768
2	16384	32768
3	16384	16384
4	8192	16384
5	4096	8192
6	2048	8192
7	1024	8192
8	512	4096
9		4096
10		4096
11		2048
12		2048
13		2048
14		2048
15		1024
16		1024

(b)



Fastest: eight wide cores, L2 cache size of 512 KB, time = 0.255s



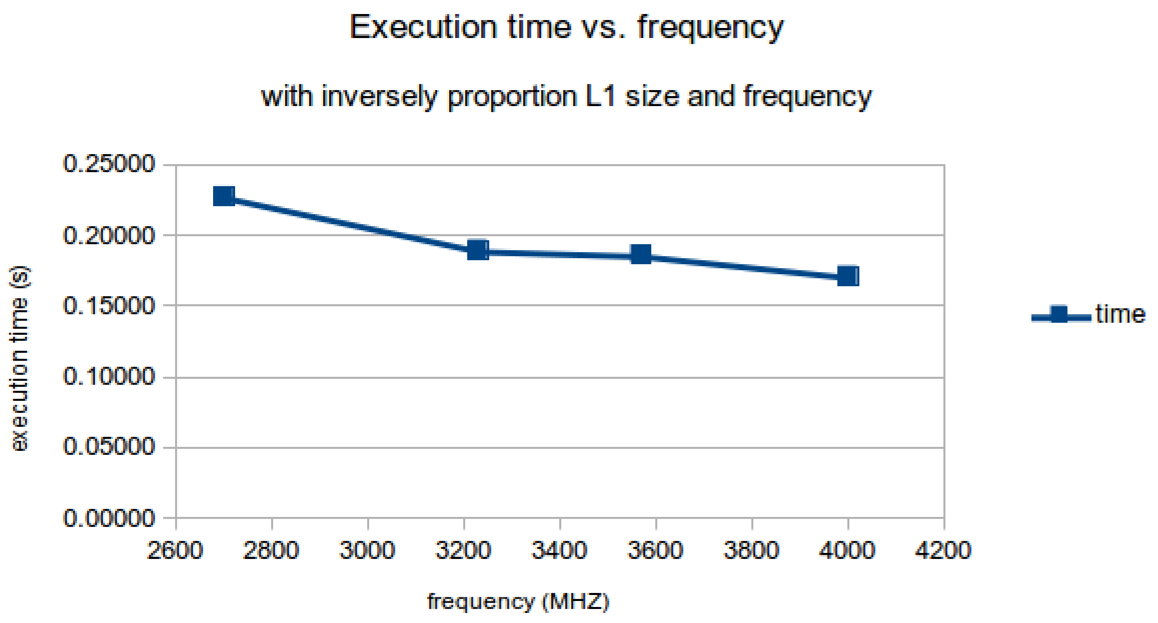
Lowest energy: eight wide cores, L2 cache size = 512 KB, energy = 14.237 J

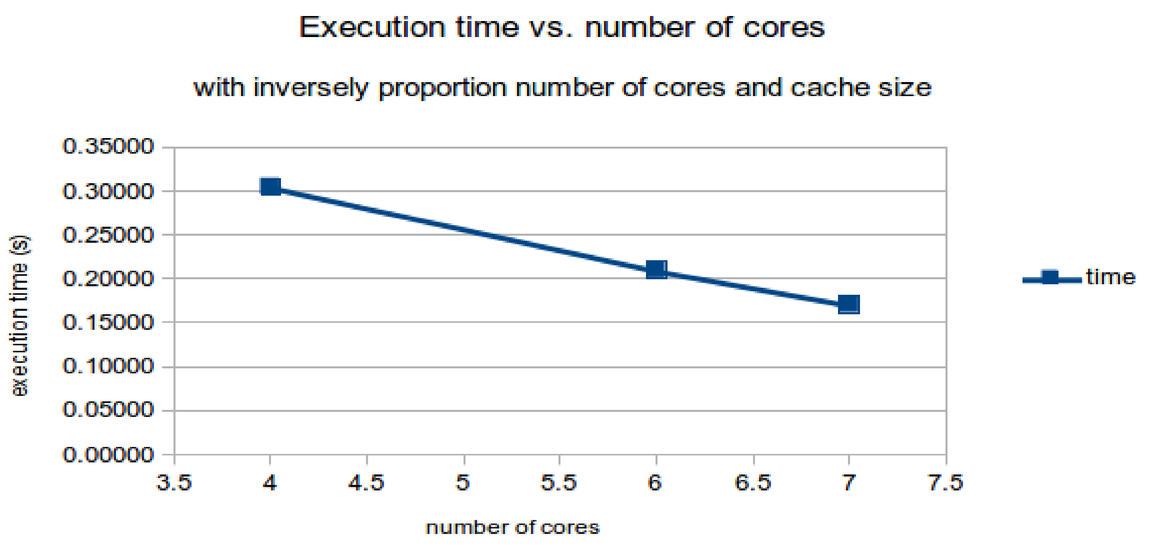
PROBLEM 4

(a)

We began our exploration of the high-performance design space by considering the factors that most influence performance. We identified three: core type, core frequency, cache size/configuration, and number of cores. For the first, a few simulations confirmed what is obvious - wide cores are much faster than narrow cores. The next three are more challenging. Number of cores conflicts with cache size due to area restrictions. Core frequency conflicts with L1 cache size in the zsim simulator.

We swept these two curves to determine trends.





We also ran several simulations with different cache ways, and determined an associativity of 16 to be ideal.

The performance curves tended to favor high frequency and high number of cores at the expense of L3 cache size and L1 size/associativity. We explored this space, and our best result was:

cores: 7, wide

frequency: 4000 Mhz

L1 size/associativity: 4 KB / 1

L2 size/associativity: 512 KB / 16

L3 size/associativity: 16384 KB / 16

execution time: 0.17110 s

normalized execution time: 0.307

(b)
Exploring the low energy space was more complicated, primarily because the selection of core type is less clear. Theoretically, one would expect narrow cores to solve a problem using less energy. Smaller cores are designed to do a task somewhat slower, but with significantly less energy consumption - leading to a net energy savings. However, a few simulations quickly showed that this is not the case in zsim. Narrow cores are *much* slower than their wider counterparts, leading to higher total energy use.

A few more simulations confirmed what the narrow core space suggested - that solving the task quickly with higher resources generally results in a lower energy use than solving the task more slowly with fewer resources. However, energy consumption increases sharply at the highest performance cases due to high frequencies requiring higher supply voltages (recall power scales quadratically with voltage). As such, we expect to find a local minimum of energy consumption as we scale up performance.

We explored the relatively high performance space: cores from 4-7 and frequency from 2300-4000 Mhz. Our minimum energy was at 2500 Mhz and 4 wide cores, for an energy consumption of 19.203 J. In comparing this answer to the default case we realized our exploration had been incomplete, as the value of the zsim default results in a lower energy. This then became our minimum energy.

parameters: default

energy consumption: 17.301 J

normalized energy consumption: 1

(c)
Maximizing performance per area is another problem involving the performance vs. number of cores and performance vs. cache size curves. Increasing cores and cache size increases performance, but at some point one starts to see diminishing returns from allocating more resources (due to how parallelizable the workload is and how it uses cache).

Based on 1A, we expect the art benchmark to have a maximum performance per area value with four wide cores (and likely a corresponding middle-ground cache value). Simulations in this space confirm this prediction, with a resulting minimum of:

cores: 4, wide

frequency: 4000 Mhz

L1 size/associativity: 4 KB / 1

L2 size/associativity: 512 KB / 16

L3 size/associativity: 8192 KB / 16

performance/energy: 0.12423

performance/area: 0.01736

normalized performance/area: 1.828

(d)
Maximizing performance/energy is a similar problem, except that energy spikes sharply at the higher values of frequency (due to increasing supply voltage) as opposed to number of cores and cache size. As such, we expect our maximum performance/energy to be at some high number of cores and middling frequency. Dynamic power is relatively unimportant due to the low execution time in the higher performance echelons, so we expect cache size to remain high.

Simulation in this space bears out this analysis, with the maximum performance/area as follows:

cores: 7, wide

frequency: 3230 Mhz

L1 size/associativity: 8 KB / 2

L2 size/associativity: 512 KB / 8

L3 size/associativity: 16384 KB / 8

performance/area: 0.01537

performance/energy: 0.23683

normalized performance/area: 2.28220

(e)
The performance per unit energy and performance per unit area winners are the not the same. We expect this because the tradeoffs are different - cores and cache size vs. performance is more critical for performance/area, while core frequency vs. performance is more critical for performance/energy.

In this case, the performance/energy winner is “better.” The performance per area of the 7-core setup is 88% of the winning 4-core setup, while the performance per energy of the 4-core setup is only 52% of the ideal 7-core setup.

(f)
Using the data we collected for (a) and (b), there was actually no configuration within 20% of the fastest speed and lowest energy.

The fastest case within 20% of the energy minimum (4 cores at 3.23 GHz) is 48% of the maximum speed, and the lowest energy case within 20% of the maximum speed (a 6 core 4 GHz setup) is 217% of the minimum energy consumption.

Both winning cases are highly specialized - the performance case with an energy guzzling high frequency/supply voltage, and the energy case with a low frequency. As such, it is hard to find a single scenario which does both near-optimally. Perhaps this is why specialization is rising to the forefront of computer architecture research.