CS 351 Exam 4, Fall 2012

Your name: __________________________________________

Rules
- You may use one handwritten 8.5 x 11” cheat sheet (front and back). This is the only resource you may consult during this exam.
- Include explanations and comments in your answers in order to maximize your partial credit. However, you will be penalized for giving extraneous incorrect information.
- You may use the backs of these pages if you need more space, but make it clear where to find your answer to each question.
- Unless otherwise specified, you do not need to work out the arithmetic on math problems. Just do enough algebra to set up an answer of the form: Answer = [arithmetic expression] [units]

Grade (instructor use only)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Your Score</th>
<th>Max Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem 1: Reliability and error correction</td>
<td></td>
<td>23</td>
</tr>
<tr>
<td>Problem 2: Disk performance and RAID</td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>Problem 3: I/O</td>
<td></td>
<td>15</td>
</tr>
<tr>
<td>Problem 4: Parallelism and optimizations</td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>100</td>
</tr>
</tbody>
</table>
Problem 1: Reliability and error correction (23 points)

a) [10 points] After painstaking data collection, you conclude that, over the past 500 days, your web hosting provider has been up and running 99% of the time. The average (mean) downtime is approximately 12 hours (half a day).

What are the MTTF, MTBF, MTTR, and availability of your hosting service based on this data?
b) [13 points] You have a RAID 5 array of 5 disks, including the parity information. Here is the value of the first byte of one of your RAID stripes on each disk:

<table>
<thead>
<tr>
<th>Disk</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10011110</td>
</tr>
<tr>
<td>2</td>
<td>01111100</td>
</tr>
<tr>
<td>3</td>
<td>01111010</td>
</tr>
<tr>
<td>4</td>
<td>11011111</td>
</tr>
<tr>
<td>5</td>
<td>01000111</td>
</tr>
</tbody>
</table>

- A space ray descends and flips the least significant (rightmost) bit of this byte on Disk 4 without leaving a trace.

Can you detect this error? Explain.

Can you reconstruct the information? Explain.

- A small child shakes Disk 4 vigorously until it fails to function at all.

Can you reconstruct Disk 4’s data? Explain.

- Which of these failure modes is more likely for hard disk drives?
Problem 2: Disk performance and RAID (30 points)

a) [10 points] You knew it was coming! It’s …drum roll… the mandatory disk performance question!!

Your disk has the following specs:

• Average seek time = 4 ms
• Controller overhead = 0.2 ms
• Rotation speed = 10,000 rpm
• Transfer rate = 150 MB/s

How long does it take this disk to handle a 1 MB request?

b) [20 points] You have a RAID 1 array with 8 mirrored pairs of disks (so 16 total). Each disk has latency \( L \) (in ms) and transfer rate \( T \) (in MB/s).

**Small reads:**
In terms of \( L \) and \( T \), how long does it take you to do a small (latency-dominated) read?

How many small reads can your array handle at once?
Small writes:
How long does it take you to do a small (latency-dominated) write?

How many small writes can your array handle at once?

Large reads:
How long does it take you to do a 2 MB read?

How many large reads can your array handle at once?

Large writes:
How long does it take you to do a 2 MB write?

How many large writes can your array handle at once?
**RAID 5:**
Pick any type of access (small read, small write, large read, large write), and repeat the two questions for a RAID 5 array with 8 disks’ worth of data and one disk’s worth of parity information.
**Problem 3: I/O (15 points)**

B. [10 points] Your company’s genius team of physicists has figured out a way to store 4 bits inside a single flash cell. They call this technology quad-level cell (QLC). Answer the following questions about QLC:

- How many threshold voltages would your QLC device need to distinguish among?

- How would you expect QLC’s **cost per bit** to compare to that of current flash technologies? Explain.

- How would you expect QLC’s **write performance** to compare to that of current flash technologies? Explain.

- How would you expect QLC’s **reliability** to compare to that of current flash technologies?
C. [5 points] Pick one of the following I/O-handling techniques and explain in detail the actions taken by the CPU with this technique:
   - Polling
   - DMA
   - Interrupts
Problem 4: Parallelism and optimizations (32 points)

A. [8 points] Pick one of the following code optimizations. Explain what it is, why it is useful, and when it is not useful:
   • Loop unrolling
   • Blocking (for matrix multiplication)
   • Software prefetching
   • Strassen’s algorithm (for matrix multiplication)

B. [8 points] Draw a dependency graph for the following code. Assume that the functions do not modify their inputs and do not have side effects.

   \[
   \begin{align*}
   x &= f_1(a, b); \quad \text{// Statement 1} \\
   y &= f_2(a, c); \quad \text{// Statement 2} \\
   z &= f_3(a, x); \quad \text{// Statement 3} \\
   a &= f_4(x, y); \quad \text{// Statement 4}
   \end{align*}
   \]

   How many threads can this code utilize, and which statements should each thread execute?
C. Consider the following C++ code, for some large value of $N$:

```cpp
for (int i=0; i < N; i++) {
    C[i] = ComplicatedFunction(A[i], B[i]);
}
```

Assume that ComplicatedFunction does not modify the arrays $A$ and $B$.

- [8 points] If you use OpenMP to parallelize this program on a dual-core CPU, and each CPU supports 2 hardware threads…
  - How many threads should OpenMP divide this program into?
  - Which values of $i$ should map to each thread? Be very specific in terms of $N$.
  - What is the maximum speedup you would expect from parallelizing this program? What might stop you from achieving this speedup?

- [8 points] If you use CUDA to parallelize this program on a GPU…
  - Write pseudocode for the kernel (the part of the program that will run on the GPU).
  - If the GPU supports 2000 hardware threads, what is the maximum speedup you would expect from parallelizing this program? What might stop you from achieving this speedup?
[Thanks to Zach, Nick, and Kyle for this image]

[Have a happy holiday full of magical elves and unicorns]