Name: _____ USC ID: _____

INF 551 – Spring 2016

Midterm (100 points)

3/2/2016

1. [**RAID, 25 points**] The following table shows the characteristic data of different RAID setups of N disks. The data for RAID-0 are already filled out for you. Note that Capacity = N means that the RAID has N times of the capacity of a single disk. S and R are sequential and random bandwidths of single disk respectively. D is delay or latency of a single request on a disk. For example, the latency is D for RAID-0, which means that the latency of RAID-0 is the same as that of a single disk. Note that for RAID, the latency accounts for the time spent in getting disk heads ready for operations.

   a. [15 points] Fill in the data for RAID-1, RAID-4, and RAID-5.

   Answer:

	RAID-0	RAID-1	RAID-4	RAID-5
Capacity	N	$N/2$	$N-1$	$N-1$
Reliability	0	1 (for sure) $\frac{N}{2}$ (if lucky)	1	1
Throughput				
Sequential Read	$N \cdot S$	$(N/2) \cdot S$	$(N-1) \cdot S$	$(N-1) \cdot S$
Sequential Write	$N \cdot S$	$(N/2) \cdot S$	$(N-1) \cdot S$	$(N-1) \cdot S$
Random Read	$N \cdot R$	$N \cdot R$	$(N-1) \cdot R$	$N \cdot R$
Random Write	$N \cdot R$	$(N/2) \cdot R$	$\frac{1}{2} \cdot R$	$\frac{N}{4} R$
Latency				
Read	D	D	D	D
Write	D	D	$2D$	$2D$

   b. [5 points] Explain the sequential read problem in RAID-1. That is, why does RAID-1 have low sequential read performance? Give an example for your explanation.

   Answer:

Disk 0	Disk 1
0	0
1	1
2	2
3	3

   For example, consider reading in sequence: blocks 0, 1, 2, 3.
   Block 0 goes to disk 0, 1-> disk1, 2-> disk 0, 3->disk1. Note that in this arrangement, 0 and 1 can be read in parallel, with the hope of doubling the bandwidth of a single disk.
   However, note that disk 0 receives requests for 0, 2, thus skipping block 1, while disk 1 receives 1 and 3, thus skipping block 2.
   So each disk receives request for every other block, half time wasted in rotating over the skipped block. This is the sequential read problem in RAID 1.

Name: _____ USC ID: _____

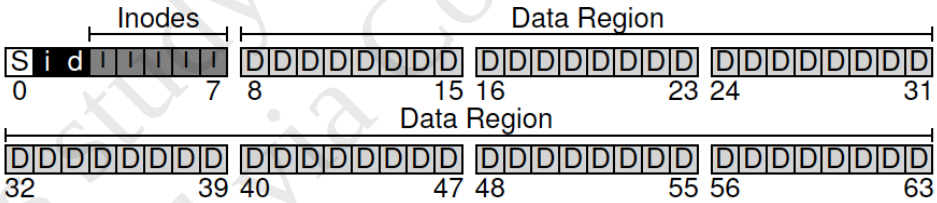
c. [5 points] Explain the small write problem in RAID-4 and how does RAID-5 addresses the problem? Also please use an example to help explain.

Answer:

Disk 0	Disk 1	Disk 2	Disk 3	Disk 4
0	1	2	3	P0
4	5	6	7	P1
8	9	10	11	P2
12	13	14	15	P3

Consider updating block 4 and 13:
• We will need to read and write parity disk (P1 and P3) for both updates.
Hence, the parity disk will become the bottleneck.
• All updates need to update the parity disk.
• To make it worse, the update has to be done in sequence, hence no parallelization.
Since, two I/O's per update (read followed by write), so throughput on random writes will be (R/2) MB/s. In other words, writing 10MB of data, will require 20MB of the bandwidth.
This is the small write problem in RAID-4.
In RAID-5, the problem is overcome by distributing the parity blocks among multiple disks. Hence, removing the blockneck caused by a single parity disk.

2. [**File System, 25 points**] Consider the file system similar to the one we have seen in class. The file system has the following parameters.



Block size	4KB
Number of blocks on disk	64
Inode size	512B
Number of inode blocks	5 (blocks #3 to #7)

Note that Superblock 'S' stores the location of imap, dmap, and the inode table. It will need to be read first into the main memory. Suppose that the inumber of root directory is 0.
Consider **WRITING** a file "/foo/more/bar.txt", assuming that this is a new file. The size of the file is 10KB, occupying blocks #16, 17, and 18. Suppose the inumbers of "foo", "more", and "bar.txt" are 1, 8, and 10 respectively. Assume that NO caching and write buffering are available.

a. [15 points] In the table below, fill in either **read** or **write** operations in the order of their execution.

Answer:

	supernode	data bitmap	inode bitmap	root inode	foo inode	more inode	bar.txt inode	root data	foo data	more data	bar.txt data[0]	bar.txt data[1]	bar.txt data[2]
create()	read												
				read									
								read					
					read*								
									read				
						read							
										read			
			read										
			write										
										write			
							write						
						write							
							read						
write()		read											
		write											
											write		
							write						
							read						
write()		read											
		write											
												write	
							write						
							read						
write()		read											
		write											
													write
							write						

b. [5 points] How many block I/O's are needed for create()? Explain your answer.

Answer:

Block size=4KB

Inode size=512B

Number of inodes per block=4KB/512B=8

The inumber of 'root', 'foo', 'more' and 'bar.txt' are 0,1,8,10 respectively.

'root' and 'foo' are in one block and 'more' and 'bar.txt' in other block.

Assume that NO caching and write buffering are available. 11 block I/O's are needed. (include superblock)

c. [5 points] How many block I/O's are needed for each write()? Explain your answer.

Answer:
5 block I/O's are needed.
1. read bar.txt inode data
2. read the data bitmap
3. write/update the data bitmap (to remember which block has been allocated for bar.txt4. write data of bar.txt5. write/update inode of inode.txt (with new modified time, for example)

3. [**XML, 15 points**] Consider the following XML document.

```
<menu>
 <popup>
  <menuitem value="New directory" default=”true” />
  <menuitem value="New file" />
  <menuitem value="Close" />
```

```
    </popup>
  </menu>
```

a. [10 points] Convert it into a JSON document.

Answer:

```
{
  "menu":{
    "popup":{
      "menuitem":[
        {
          "value": "New directory",
          "default": "true"
        },
        {
          "value": "New file"
        },
        {
          "value": "Close"
        }
      ]
    }
}
```

b. [2 points] Write an XPath expression to find menuitems whose defaults values are "true". What is the result of evaluating the expression?

Answer:

/menu/popup/menuitem[@default="true"]

<menuitem value="New directory" default="true"/>

c. [3 points] Write an XPath expression to find menuitems whose values contain "file". What is the result of evaluating the expression?

Answer:

/menu/popup/menuitem[contains(@value,"file")]

<menuitem value="New file"/>

4. [**Unicode, 15 points**] Fill in the table below with binary code point, binary UTF-8, and hexadecimal UTF-8 for the 3 characters shown in the first column.

Answer:

Character		Binary code point	Binary UTF-8	Hexadecimal UTF-8
$	U+0024	0100100	00100100	24
¢	U+00A2	00010100010	11000010 10100010	C2 A2
€	U+20AC	0010000010101100	11100010 10000010 10101100	E2 82 AC

5. [**Data modeling, 20 points**] Consider a relational model R that consists of the following relations:
   Laptop(SerialNo, OperatingSystem, HardDrive)

Product(SerialNo, Brand, Model, Price)
Store(ID, Name, City)
Purchase(BuyerName, SellerName, StoreID, SerialNo, Date)
Person(Name, PhoneNumber, City)

Note the following:

- Underlined attributes are key attributes.
- Laptop also has a brand, a model, and a price.

We have learned how to convert ER diagrams into relational models. In this problem, you are asked to do the opposite conversion: reverse engineer the above relational model R into an ER diagram E such that when we translate E using the approach given in class, we obtain the model R.

Here are the requirements:

- Properly depict entity sets, relationships, and attributes in your ER diagram.
- Indicate the multiplicity of relationships. Explain your answer.
- Indicate key attributes of entity sets.

Answer: