Virtual Memory Review Exercise

Consider a call to the following (delayed-branch MIPS) assembly function with the following initial page table. Assume that pages are 4KiB and that all page faults (but not protection faults) can be serviced by the OS without evicting pages, $sp is initially 0x6004, $ra is initially 0x1040, and $a0 is initially 0x1.

Assembly Function

(Virtual) Address	Instruction
0x2004	`Foo: addiu $sp, $sp, -4`
0x2008	`beq $a0, $zero, Skip`
0x200C	`sw $ra, 0($sp)  #delay slot`
0x2010	`jal Foo`
0x2014	`addiu $a0, $a0, -1 #delay slot`
0x2018	`Skip: lw $ra, 0($sp)`
0x201C	`jr $ra`
0x2020	`addiu $sp, $sp, 4  #delay slot`

Initial Page Table (excerpt)

Virtual Page Number (not stored)	Page Valid	Page Dirty	Permissions	Physical Page Number
0	0	0	none	4
1	1	0	read, exec	5
2	0	0	read, exec	1
3	0	0	none	1
4	0	0	read, write	12
5	1	0	read, write	3
6	1	0	read, write	2
…	…	…	…	...

What virtual address is $ra saved to in the first call to Foo? What physical address is this? What about in the second call (the recursive call) to Foo?

Where will page faults occur in the execution of this code?

After any changes in the TLB are flushed to the page table, what will have changed in the page table as a result of running the above code?

Suppose $a0 were initially 0xC00 instead of 0x1. What additional traps will occur?

MapReduce

Divide a large data set into many smaller pieces for independent parallel processing. Combine and process intermediate results to obtain final result. Execution goes as follows:

0) User program breaks the input files into small pieces and starts up many copies of the program on different machines. One copy is the *Master* and assigns work to the many *Workers*.
1) **Map Task:** Reads an input file fragment/shard and parses key/value pairs that are passed through the Map function and buffered in memory.
2) **Data Shuffle:** Buffered pairs are written to disk and partitioned. Locations of the partitioned regions are passed to the Master, who forwards these to Reduce workers.
3) **Reduce Task:** Reads buffered data and sorts by key. Key/value pairs are passed through Reduce function and saved to an output file.
4) When all tasks are completed, control returns to the user program.

Below is an example of the `map` and `reduce` functions for a word counter program. On the right, write the corresponding functions for a program that, given a set of numbers, outputs the sum of numbers with distinct modulo 5. (Assume any number module n is going to be a number from 0 and n-1)

`map(String key, String value):`
`  // key: document name`
`  // value: document contents`
`  for each word w in value:`
`  EmitIntermediate(w, "1");`
	`map(String key, int *values):`
	`  // key: document name`
	`  // values: document contents`

`reduce(String key, Iterator values):`
`  // key: a word`
`  // values: a list of counts`
`  int result = 0;`
`  for each v in values:`
`  result += ParseInt(v);`
`  Emit(key, AsString(result));`
	`Reduce(int key, Iterator values):`
	`  // key: original value mod 5`
	`  // values: original value`

In the boxes below, fill out how an execution of `sum_mod5` on the 4 3-number files shown would look like. Draw arrows from the Map machines to the Reduce machines, assuming Reduce1 handles numbers mod 5 is from 0 to 2, and Reduce2 handles numbers whose mod 5 is 3 or 4.

Distribute:

-1, -1, -2	-2, 0, 1	0, -1, 1	2, 1, 1
Map1	**Map2**	**Map3**	**Map4**

Shuffle:

Reduce1	**Reduce2**

Collect:

Warehouse Scale Computing

(Google's Datacenter Bill)[1] According to widely circulated 2006 estimate published in the New York Times,[2] Google has about 450,000 servers. More recent sources speculate that Google has over 1 million servers.[3] Assume each server draws an average of 200 watts and Google's datacenters have a PUE of 1.25. Assume that Google pays an average of about 6 cents per kilowatt-hour for datacenter electricity.

a) Estimate Google's annual power bill for its datacenters. Ignore the power cost of networking equipment. Assume 365.25 days in a year.

b) Assuming each server costs $2000 and server costs are amortized (paid for) over 3 years, estimate Google's annual costs for server hardware over this period.

c) Assume power and cooling infrastructure costs about $10/(peak server W)[4] and that average server power is 75% of peak power. Amortizing over 12 years, estimate Google's annual cost of power and cooling infrastructure.

d) How would reducing average server power utilization from 200 watts to 125 watts change your cost estimates? (Answer in words, not numbers)

e) Suppose Google had reduced the PUE in a 50,000 machine datacenter from 1.5 to 1.25 without significant power/cooling infrastructure changes and without decreasing the total power available for the datacenter. How much more power is available for servers? What's the total cost savings per server?

---

[1] Question inspired by from power calculation in A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs. "Cutting the electric bill for internet-scale systems." ACM SIGCOMM Computer Communication Review, 39(4):123–134, 2009.

[2] J. Markoff and S. Hansell, "Hiding in Plain Sight, Google Seeks an Expansion of Power," New York Times, June 2006.

[3] See, e.g., R. H. Katz, "Tech Titans Building Boom," IEEE Spectrum, February 2009 ("With reportedly more than 1 million servers…")

[4] The Warehouse-Scale Computing book estimates infrastructure costs at around $12-15/IT equipment W for large datacenters, with 80% of that cost going to power and cooling.