CS252 HOMEWORK 5

Due Tues. 12/04/07

Problem 1:

Suppose a processor sends 10 disk I/Os per second, these requests are exponentially distributed, and the average service time of an older disk is 20 ms. Answer the following questions:

a) On average, how utilized is the disk?

b) Calculate the average length of the queue.

c) Calculate the average length of the system.

d) What is the average time spent in the queue?

e) What is the average response time for a disk request, including the queuing time and disk service time?

f) Suppose that we get a new, faster disk. Recalculate the answers to the questions a-e above, assuming the disk service time is 10 ms.

g) Suppose that instead of a new, faster disk, we add a second slow disk, and duplicate the data so that reads can be serviced by either disk. Let's assume that the requests are all reads. Recalculate the answers to questions a-e, this time using an M/M/m queue. Note that the equation for the probability that there are as many or more tasks than there are servers in the book has a typo; in the book, it looks like Utilization is raised to the power of the Number of servers, whereas it should be the quantity (Number of servers times Utilization) that is raised to that power.

Problem 2:

Some memory systems handle TLB misses in software (as an exception), while others use hardware for TLB misses.

a) What are the trade-offs between these two methods for handling TLB misses?

b) Will TLB miss handling in software always be slower than TLB miss handling in hardware? Explain.

c) Use the data from Figure 5.45 to calculate the penalty to CPI for I-TLB misses on the following workloads assuming hardware TLB handlers require 10 cycles per miss and software TLB handlers take 30 cycles per miss:

Workload 1: 50% gcc, 25% perl, 25% ijpeg

Workload 2: 35% swim, 35% wave5, 20% hydro2d, 10% gcc

d) TLB miss rates for floating-point programs are generally higher than those for integer programs. Why?

Program	CPI	Cache misses per 1000 instructions		TLB misses per 1000 instructions
		I-cache	L2 cache	I-TLB
TPC-C-like	2.23	11.15	7.30	1.21
go	0.58	0.53	0.00	0.00
m88ksim	0.38	0.16	0.04	0.01
gcc	0.63	3.43	0.25	0.30
compress	0.70	0.00	0.40	0.00
li	0.49	0.07	0.00	0.01
ijpeg	0.49	0.03	0.02	0.01
perl	0.56	1.66	0.09	0.26
vortex	0.58	1.19	0.63	1.98
Average SPECint95	0.55	0.88	0.18	0.03
tomcatv	0.52	0.01	5.16	0.12
swim	0.40	0.00	5.99	0.10
su2cor	0.59	0.03	1.64	0.11
hydro2d	0.64	0.01	0.46	0.19
mgrid	0.44	0.02	0.05	0.10
applu	0.94	0.01	10.20	0.18
turb3d	0.44	0.01	1.60	0.10
apsi	0.67	0.05	0.01	0.04
fpppp	0.52	0.13	0.00	0.00
wave5	0.74	0.07	1.72	0.89
Average SPECfp95	0.59	0.03	2.68	0.09

Figure 5.45 CPI and misses per 1000 instructions for running a TPC-C-like database workload and the SPEC95 benchmarks (see Chapter 1) on the Alpha 21264 in the Compaq ES40. In addition to the worse miss rates shown here, the TPC-C-like benchmark also has a branch misprediction rate of about 19 per 1000 instructions retired. This rate is 1.7 times worse than the average SPECint95 program and 25 times worse than the average SPECfp95. Since the 21264 uses an out-of-order instruction execution, the statistics are calculated as the number of misses per 1000 instructions successfully committed. Cvetanovic and Kessler [2000] collected these data, but unfortunately did not include miss rates of the L1 data cache or data TLB. Note that their hardware performance monitor could not isolate the benefits of successful hardware prefetching to the instruction cache. Hence, compulsory misses are likely very low.