Stephen Sallas
sms0107

HW1

1.1. Based on overall SPEC performance, if I was considering every benchmark I would choose to purchase the Itanium 2 because the Opteron/Itanium Times ratios have a geometric mean of 1.30 which shows the Itanium 2 has overall better performance time. If I was only considering the three applications the company will be using I would choose the Opteron because the geometric mean of just these would be 0.888. (See calculations in Problem 1.2).

1.2. The weighted average of execution time ratios for this mix of application would be found by doing the calculation: $(0.6 * 0.92) + (0.2 * 1.03) + (0.2 * 0.65) = 0.888$. This calculation multiplies the amount of time the applications will be running with the Opteron/Itanium Times ratios of these applications. This means that the Opteron is actually the better choice considering applications the company will be using.

1.3. The speedup of the Opteron and the Itanium 2 is found by comparing the performance times of each machine. The performance time of the Opteron is found by the calculation: $(0.6 * 51.5) + (0.2 * 136.0) + (0.2 * 150.0) = 88.1$. The performance time of the Itanium 2 is found by the calculation: $(0.6 * 56.1) + (0.2 * 132.0) + (0.2 * 231.0) = 106.26$. Since the Opteron is faster, we will measure the speedup from the Itanium 2 to the Opteron. This is done by the calculation: $\frac{106.26}{88.1} = 1.2$. The speedup is 1.2.

Stephen Sallas
sms0107

2.1. The overall speedup would be found by the calculation: $\frac{1}{(1-.0.35)+\left(\frac{0.35}{3}\right)} = 1.3$. So the overall speedup is 1.3.

2.2. The overall speedup would now be found by the calculation: $\frac{1}{(1-(0.35+0.15))+\left(\frac{0.35}{3}\right)+(\frac{0.15}{.5})} = 1.09$. So the overall speedup is 1.09.

2.3. After we incorporate the fast floating point unit, the floating point instructions take up 11.67% of execution time due to the calculation: $\frac{35}{3} = 11.67$. The memory access instructions take up 30% of execution time due to the calculation: $\frac{15}{0.5} = 30$. The rest of the program originally took 50% of the execution time. With all this information we can determine that 12.73% of execution time is contributed to floating point instructions due to the calculation: $\frac{11.67}{50+11.67+30} = 0.1273$ and 32.73% of the execution time is contributed to memory access due to the calculation: $\frac{30}{50+11.67+30} = 0.3273$.