CS 422 (Park) Assignment III Due: Feb. 28 (Wed.), 2007

Submission instructions: Please type your answers and submit electronic copies using `turnin` *by 5pm on the due date. You may use any number of word processing software (e.g., Framemaker, Word, LaTeX), but the final output must be in pdf or ps format that uses standard fonts (a practical test is to check if the pdf/ps file prints on a CS Department printer without problem). For experiments and programming assignments that involve output to terminal, please use* `script` *to record the output and submit the output file. Use* `gnuplot` *to plot graphs. Use* `ps2gif` *to convert a eps/ps plot to gif format (e.g., for inclusion in Word) if there is a need.*

PROBLEM 1 (30 pts)

(a) Suppose you need to transmit 7-bit ASCII over a wireless communication channel where bit flips are common due to noise. Suppose you are allowed to use 21 bits per ASCII character (a 3-fold increase in redundancy given that 7 bits suffice when there are no bit flips) to achieve error correction under the following condition: in every block of 21 bits transmitted (i.e., a single ASCII character) at most one bit flip can occur, albeit at any position. Describe your method and give an argument as to why it works. How would you deal with 2 bit flips per ASCII character?

(b) We know that error correction, under k bit flips, requires that the code words used for two ASCII characters, have a Hamming distance of at least $2k + 1$. What is the smallest Hamming distance between a pair of code words in your 21-bit ASCII encoding scheme that corrects single bit flips? What is the largest Hamming distance? Based on the distances, what's your evaluation of how "good" your method is?

PROBLEM 2 (30 pts)

Write a program, `calc_shannon`, that takes a data file in ASCII as command-line input, calculates the relative frequency of all ASCII characters in the data file, outputs to *stdout* the relative frequency of the ASCII characters sorted in descending order (one per line), and prints the Shannon entropy of the data file which gives a lower bound on the average number of bits per ASCII character expended under optimal compression. Test your program on the data file /u/u9/park/pub/cs422/test-shannon.txt. Submit the output recorded by `script`. How much speed-up (percentage) in file transfer completion time can you gain by using Shannon compression before transmission compared to using fixed 7-bit ASCII encoding? Find a long write-up (e.g., report, essay) that you have written during your college days (don't go all the way back to high school) that is in electronic form, dump it to ASCII text, then analyze its compressibility using `calc_shannon`. Who writes more compressible text, you or the author of test-shannon.txt? When you get a chance, you may run `calc_shannon` on one of Hemingway's books who is known to have praised the virtue of brevity. Of course, brevity is not the same as compressibility.

PROBLEM 3 (50 pts)

Rewrite the iterative client/server application in Problem 5(a), Assignment II, such that the client and server use TCP sockets (in place of FIFO) to communicate over a network. From a programming perspective, the internals of how TCP (or IP) works is not needed to use it. Like FIFO, a *socket* is one of the 7 file types in UNIX-like operating systems that is created by a `socket()` system call that returns a file descriptor. TCP is selected by using the SOCK_STREAM option. 4-byte IP addresses are used to identify a destination machine (more precisely, a network interface on the machine if it's multihomed), and 2-byte port numbers are used to identify the process that data transferred by TCP is to be delivered. The details of necessary system calls, default options, and header files will be discussed during the PSOs. Benchmark the TCP client/server application by running a server on one machine and three clients on three separate machines. Submit the `script` output at the client and server machines.