Take Final (40 Marks) **Nov 15th, 2019**

Dept. of Electrical Engineering, IIT Madras
EE4371 - Data Structures and Algorithms

- ▷ **Please write clear answers. Prefer a LYX or Latex file with well formatted maths equations.**
- ▷ **All code should have an algorithm, followed by pseudo code, following the style of the textbook (you can use C syntax)**
- ▷ **Code should be well commented and self-explanatory.**
- ▷ **Code should run!!**
- ▷ **All code will be checked for plagiarism. If instances of copying are found, I will turn over the cases to the disciplinary committee. In the midterm, there were a dozen cases of such copying. Please do not expect me to be kind during the final.**

1. **For this problem, please refer to https://en.wikipedia.org/wiki/Knapsack_problem in wikipedia. It discusses the knapsack problem and discusses both dynamic programming and greedy algorithmic approaches.**

   Given a set of N objects of positive integer weights $\{w_i\}$, find the subset of these objects that maximizes

   $$\sum_{i=1}^{N} x_i \log_{100} w_i$$

   subject to

   $$\sum_{i=i}^{N} \frac{w_i x_i}{\sqrt{k}} \leq 10000$$

   where

   $$x_i = \begin{cases} 0 & \textbf{if } w_i \textbf{ is not used} \\ 1 & \textbf{if } w_i \textbf{ is used} \end{cases}$$

   and k is the number of items selected.

   Note that $\log_{100} w_i$ is negative only for fractional w_i which is ruled out by the fact that they are positive integers. However zero values are not ruled out.

   . . . . [15]

   (a) **Write the algorithm to solve this problem. Convert the same to pseudocode.**

   (b) **Implement the same in C and use it to solve the case in *input1.txt***

   (c) **Print out the $\{w_i\}$ set that maximizes $\sum_{i=1}^{N} x_i \log_{10} w_i$ and print that value as well.**

   (d) **How many conditions had to be checked? What is the time complexity of the problem?**

.

2. We want to study the way a fluid approaches thermal equilibrium. We have a $2D$ region of size $2 \times 2m^2$ within which we have 10^8 fluid particles. When particles are more than 10^{-3} metres apart, they do not interact with each other. When they are closer, they experience a force given by

$$\vec{F} = \frac{x\hat{x} + y\hat{y}}{(x^2+y^2)^{3/2}}$$

The force is repulsive and both particles are repelled. At the boundaries, the particles bounce, i.e., the normal velocity at the walls changes sign and the particles move back into the region with the same speed.

Initially, half the particles are randomly placed and stationary. The other half are also randomly placed but have random velocity directions with unit velocity (in m/sec). We run the simulation with time steps of 10^{-4} seconds, and is run for 1 second.

(a) If we consider every particle interacting with every other particle, create an algorithm to solve this problem. Estimate the time required to complete the simulation on a CPU that will compute a force calculation and add all the forces on a particle and then move it, for each particle, for each time step, if the CPU can complete a floating point calculation (of any type) in 10nsec. Integer operations are zero cost (since they are done in parallel with the floating point calculations) .... **[5]**

(b) Construct a 4 way tree as follows. Every particle has an x and y coordinate. Depending on their sign, they can be placed in one of the subtrees. Eg. (-ve,-ve) would be one sub-branch, (-ve,+ve) would be another etc. Given a particle at (x, y), express each of x and y as signed fraction in binary. This is possible since x and y go from -1 to $+1$but never reach the boundaries. Now in each subtree, look at the particles there. Specifically, look at the first binary digit (the msb bit) we again get (x_1, y_1), where each can be 0 or 1. Break the region into four as above and place the particle. Continue for 10 bits. Repeat for every particle. .... **[5]**

For each particle, to find other particles within 10^{-3}metres of it, locate it from (x,y), and go to its parent. Find all the children of that parent. Those are the desired particles. There will be about 100 of them. Compute the force and move the particle. Repeat for all particles.

For each time step, you have to create the tree, then find the neighbours of particles, apply the force and move them.

Write pseudocode for this procedure. What will be the time complexity of this code counting tree operations, bit operations and the float calculations? Estimate the time to execute the code, again ignoring integer operations. Is the neglect of integer operations warranted?

(c) Actually code this problem and run the same. You may change the number of particles and the size of the region to get it to run. Vary your numbers, and then extimate the speed for the given problem, above. Does the execution time agree with what you estimated in (b)? Is (b) a better algorithm than (a)? For what size problem do you find speedup, if at all? Call *dist(*v)* to generate the distribution of velocities at the beginning and at the end. .... **[10]**

(d) Create a function *dist(*v)* to accept a pointer to the velocity array, compute the magnitude of velocity ($\sqrt{x^2+y^2}$) as a vector and bin it. This will be a huge vector of 100 million floats (if you did the full problem). Sort the values into 100 bins, equally spaced from negative maximum to positive maximum. Note that a full sort is not required. Which sort algorithm is most suitable (mention the L2 cache size of your CPU in your answer)? Write out the bin populations, and plot them in python or octave. .... **[5]**