Problem Sheet 3

Deadline: Dec 01 (Thu) 2:15pm

In this problem we are going to study an NP-complete problem: *the subset sum problem*. This is a classic problem in computer science, and we are interested in its connections to statistical mechanics of disordered systems.

The *subset sum problem* is the following: suppose we have N integers $a_1, a_2, \ldots a_N$, each in the range $1 \le a_j \le 2^M$. The task is to tell whether there is a subset whose sum is three seventh of the total sum.
The *size* of the problem is NM (this corresponds to the number of bits necessary to describe an instance with parameters N and M). As this problem is **NP**-complete, there is no known algorithm that can solve this problem in time polynomial in NM; but if we have a guess for the subset, it can be checked in polynomial time.

(a) First we start brute force: write a function that takes a list S of N (possibly repeated) integers, does an exhaustive search over the 2^N subsets S', and returns the minimum value of

$$C = 7\left|\sum_{j\in S'} a_j - \frac{3}{7}\sum_{j\in S} a_j\right| = \left|4\sum_{j\in S'} a_j - 3\sum_{j\in \overline{S'}} a_j\right| .$$

We will call this *cost*; if the cost is zero, S' is a three-seventh-sum subset. Test your fuction on these examples:
$S_1 = [13, 38, 24, 16, 28]$,
$S_2 = [16, 28, 25, 28, 18, 16, 16]$,
$S_3 = [94, 21, 147, 132, 399, 48, 126, 279]$, and
$S_4 = [2482, 1114, 1388, 3058, 506, 2774, 208, 734]$.
Verify that S_1 has a three-seventh-sum subset, and S_2 has minimum cost 7.

(b) Write a function that generates N integers randomly chosen from $\{1, \ldots, 2^M\}$ (rejecting lists whose sum cannot be divided by 7 – these cannot have a three-seventh-sum subset). Write another function that generates T trials of these lists, calls the minimum-cost calculating routine you wrote in (a) for each trial, and determines p, the fraction of the trials which have a three-seventh-sum subset.

Plot p against $r := M/N$ for a few values of N (eg. 3, 5, 7, 9) and M (eg. all integers M for which $0 < r < 2$), using at least $T = 100$ trials each case. Do you see signs of a phase transition for large systems, where the probability that a three-seventh-sum subset exists changes from quite certain to quite unlikely? At what value of r does the transition occur? (We will call this r_c.)

(c) Now we look for analogies with statistical physical systems. We can look at the problem as assigning $s = +4$ or -3 to each number, and asking whether $\sum_j a_j s_j$ is zero. Recall, we defined the cost as $C = \left|\sum_j a_j s_j\right|$. In analogy with a spin glass system, we will call the signed sum $E = \sum_j a_j s_j$ as "energy". The energies are distributed between $[-3N2^M, 4N2^M]$, but for most partitions there will be cancellations and E will be closer to zero.

First estimate the mean and variance of a single term $a_j s_j$ in the sum, averaging over different partitioning (changing s_j) and over different number-partitioning problems (changing a_j over its range). Use the formula $\sum_1^K k^2 = K^3/3 + K^2/2 + K/6$, and keep only the most important term in M (M is large).

(d) Use the central limit theorem to calculate the ensemble-averaged probability distribution $P(E)$ for large N. Watch out for the fact that $P(E) = 0$ if E is not dividable by 7, so the normalisation is multiplied by 7: $P(E) \approx (7/\sqrt{2\pi\sigma^2}) \exp\{-(E - E_0)^2/2\sigma^2\}$.

(e) If we assume that the energies are randomly selected from the distribution $P(E)$ (the "random-energy" approximation), what is the probability that the energy of a given partition is zero? (N and M are large.) Show that

$$p = 1 - (1 - P(0))^{2^N} .$$

Recall that p was the probability that a three-seventh-sum subset exists. In the random energy approximation we assume that the energies are uncorrelated. Using the approximation $(1 - A/K)^K \approx \exp(-A)$ for large K, show that

$$p \approx 1 - \exp\left[-\sqrt{\frac{B}{N}}\, 2^{-N(r-r_c)} \right] .$$

What is B and r_c? Does r_c you just calculated agree with the estimate in (b) ?

(f) At this point redo the simulations of part (b) with increasing the range of N and/or T such that it would still take less than 10 minutes to run on the queue. What N and T did you use? How long did it take?

(g) We can test whether the empirical values for p in (f) follow the formula you derived in part (e). If we suitably rescale our coordinates, all curves for different N's should fall on top of each other ("data collapse"). We will change variables to $x = N(r - r_c) + (1/2)\log_2 N$. What is $p(x)$ from part (e)? Plot both this function (the theory) and your numerical results in (f) against x. It might be useful to connect data points of equal N and varying r. Do you get a good data collapse?

If you did, than you can be more confident that there were no mistakes in the calculations (in particular for the value of r_c), and that the approximations we used (random energy, large N) were good.

(Just for fun, not for credit: try plotting your numerical values (f) against x with a wrong value of r_c: no data collapse!)

Note: We will start working on part (a) and (b) in class in groups, so start with (c) in what you hand in. These, however, including (f) should be individual work. Please include the printout of your code as well. (Only you, and not your group partners are responsible for mistakes in the final version of your code!)