CSE 207C: Lattices in Cryptology Winter 2002

Problem Set 1

Instructor: Daniele Micciancio Jan. 31, 2002

Due: Feb 14, 2002

In this problem set you will analyse the security of the subset sum function from various perspectives. Problem 1 asks you to analyze the impact of the choice of the modulus on the security of the modular subsetsum function. This is just a warm up question that can be solved with standard algorithmic techniques, without the use of lattices. In Problem 2 you are asked to analyze the efficiency of standard approaches to attack the subsetsum function. Problem 3 asks to analyze approaches to invert the subsetsum function based on lattices.

Problem 1

In this problem we consider the generalized subsetsum problem in finite groups. Let m be a (sufficiently large) positive integer, and let $G = Z/Zm$ be the set of all integers modulo m. The generalized subsetsum problem is the following: Given group elements $a_1, \ldots, a_n \in G$ and $b \in G$, find a subset of the indices $S \subset \{1, \ldots, n\}$ such that $\sum_{i \in S} a_i = b$ (sums computed in G) if such a set exists, or report failure otherwise.

Typical choices of m are

- Setting $m = 2^k$ to a power of 2. This has clear advantages from the efficiency point of view, as addition modulo a power of two can be executed very fast when numbers are represented in binary.
- Setting m to a prime number of k bits.

Using groups of prime order is usually the preferred choice in many cryptographic applications as it typically gives better security guarantees. Prove that the choice of the modulus has no impact on the security of the subsetsum problem, i.e., subsetsum modulo a power of two is not any easier than subsetsum modulo a prime of comparable bit size.

Problem 2

Consider the standard (integer) subsetsum problem with density 2: given n numbers, each $2n$ bits in length, and a target value b, find a subset that adds up to b. In this problem you should evaluate the hardness of solving this problem, using an exhaustive search attack, or the Schroeppel-Shamir's attack. Answer the following questions:

- How long would it take to invert this function when $n = 10$ using a typical computer (say a 1 GHz computer) using the first attack? What about the second attack?

- Say you want to break the function using moderately small resources (e.g., 100MB of memory and 1 hour of computing time). What is the largest value of the security parameter (n) for which you can break the function using the first attack? Using the second attack? Which resource (time or space) do you run out of first using the two attacks?

- Extrapolate the results from the previous questions, to evaluate which value of n should be used to get reasonable security. What value of n would you recommend if you were asked to suggest an appropriate choice for the security parameter?

In order to answer the questions, you can either do some calculations and estimate the cost of running the attack, or implement the attack and run some experiments.

Problem 3

In class we proved that if you are given access to an oracle that solves the shortest vector problem exactly, then you can solve most subsetsum problem with sufficiently low density. ($\delta < 0.9$). Unfortunately, we do not know how to solve SVP exactly, and known polynomial time algorithms only achieve exponential approximation factors. (e.g. LLL achieves $2^{O(n)}$.) Still, if the density of the subsetsum is sufficiently low ($\delta < O(1/n)$), then one can show that a 2^n approximation oracle is enough to break most subsetsum instances.

The LLL algorithm, and its more complicated variants, are often reported to perform much better in practice than the theoretical worst case analysis. (This is especially true when the algorithms are enhanced with various heuristics.) In this problem we take a pragmatic approach and try to determine the "practical" performance of lattice based attacks to subsetsum problems of density δ. The main question is: what is the highest value of the density (possibly a function of n) for which subset sum can be solved *in practice*?

You can approach this question using one of the two following methods. You are encouraged to to solve both parts of course, but you are only required to work on one of them for grading purposes:

- (Theoretical method) Assume that the approximation factor achieved in practice (with high probability) by known lattice reduction algorithms is polynomial in n (i.e., n^c for some constant c) instead of exponential. What is the highest value of the density for which the Lagarias-Odlyzko attack is expected to work? Considering also this lattice attack, which value of the security parameter (n) would you recommend in order to achieve reasonable security?

- (Practical approach) Experiment with the LLL algorithm. Up to what values of the security parameter the length doubling subsetsum function from Problem 2 can be broken using the LLL algorithm? Based on your experiments, what values of the security parameter would you recommend? Can you make a conjecture about the maximum density for which the LLL attack works when the security parameter n is large?