An Improved Multi-Set Algorithm for the Dense Subset Sum Problem

Andrew Shallue

University of Calgary

May 19, 2008

Outline

Outline

Problem Statement

Modular Subset Sum (MSS) :
Given $a_1, \dots, a_n, t \in \mathbb{Z}/m\mathbb{Z}$, find $x_i \in \{0, 1\}$ such that

$$\sum_{i=1}^{n} a_i x_i = t \mod m$$

Random Modular Subset Sum (RMSS) :
n, t, m fixed, a_i chosen uniformly at random from $\mathbb{Z}/m\mathbb{Z}$

Density

Definition

The *density* of an instance of MSS is given by $\frac{n}{\log m}$

Intuitively, the map $\mathbf{x} = (x_1, \ldots, x_n) \mapsto \sum_{i=1}^{n} a_i x_i \mod m$ is

$$1 - 1 \text{ if density less than } 1$$

$$\text{onto if density greater than } 1$$

We will focus on dense instances of RMSS, i.e. $m < 2^n$

These instances should have many solutions

Birthday Problems

Definition (k-Set Birthday Problem)

Given k lists $L_1, \ldots, L_k$ of elements drawn uniformly and independently from $\mathbb{Z}/m\mathbb{Z}$, find $\ell_i \in L_i$ for $1 \leq i \leq k$ such that

$$\ell_1 + \ell_2 + \cdots + \ell_k = 0 \mod m \ .$$

Expect solution to exist if $|L_i| = m^{1/k}$

Wagner (2002): heuristic algorithm that expects to find solution if $|L_i| = m^{1/\log k}$

Previous Solutions

General solutions:
time-space tradeoff: $O(2^{n/2})$ time and space

dynamic-programming: $O(n \cdot m)$ time and space

Schroeppel-Shamir (1981): $O(2^{n/2})$ time, $O(2^{n/4})$ space

Sparse case:
Lagarias-Odlyzko (1985): for almost all problems of density $d < 0.645$, reduces to shortest vector problem

for almost all problems of density $d < \frac{1}{n}$, poly time using LLL

Coster et. al. (1992): bound improved to $d < 0.98$

Previous Solutions

Dense case:

Chaimovich (1999) : $n = (m \log m)^{1/2}$, time $O(n^{7/4} / \log^{3/4} n)$

Flaxman-Przydatek (2005) : $m = 2^{O(\log n)^2}$, time $O(n^{3/2})$

Lyubashevsky (2005) : $m = 2^{n^{\epsilon}}$, $\epsilon < 1$, time and space $2^{O(n^{\epsilon} / \log n)}$, by solving birthday problem in $\widetilde{O}(m^{2/\log k})$.

New Results

Theorem (S, 2007)

Let lists $L_1, \ldots, L_k$ each contain $\alpha m^{1/\log k}$ elements drawn independently and uniformly from $\mathbb{Z}/m\mathbb{Z}$. Assume that $\alpha > \max\{1024, k\}$ and $\log m > 7 \log \alpha \log k$. Then Wagner's algorithm has complexity $\widetilde{O}(k\alpha \cdot m^{1/\log k})$ time and space and outputs a solution with probability greater than $1 - m^{1/\log k} e^{-\Omega(\alpha)}$.

Corollary

Let $m = 2^{n^\epsilon}$, $\epsilon < 1$ and assume that $n^\epsilon = \Omega((\log n)^2)$. Then there is an algorithm for RMSS that runs using time and space $\widetilde{O}(2^{\frac{n^\epsilon}{(1-\epsilon)\log n}})$ and finds a solution with probability greater than $1 - 2^{-\Omega(n^\epsilon)}$.

Outline

Algorithm ListMerge

Input: parameter $p < 1$, lists L_1, L_2 of integers in interval $[-\frac{mp^\lambda}{2}, \frac{mp^\lambda}{2})$
Output: list $L_{12} \subset L_1 + L_2$ of integers in interval $[-\frac{mp^{\lambda+1}}{2}, \frac{mp^{\lambda+1}}{2})$, at most one element per $b \in L_1$

1. sort L_1, L_2
2. **for** $b \in L_1$ **do**
3. **if** there exists $c \in L_2$ in interval $[-b - \frac{mp^{\lambda+1}}{2}, -b + \frac{mp^{\lambda+1}}{2})$ **then** add $b + c$ to L_{12}

Assume $|L_1| = |L_2| = \frac{\alpha}{p}$. Then resource usage is $O(\frac{\alpha}{p} \log \frac{\alpha}{p})$ time and space

k-set Birthday Algorithm

assume $t = 0$ (easy modifications for general t)

Input: parameter $k < n$, $p = m^{-1/\log k}$, $\alpha = O(n)$
Lists $L_1, \ldots, L_k$ of size α/p of elements from $\mathbb{Z}/m\mathbb{Z}$

Output: $\ell_i \in L_i$ such that $\ell_1 + \cdots + \ell_k = 0 \mod m$

1. treat list elements as integers in $[-\frac{m}{2}, \frac{m}{2})$
2. **for** level $\lambda = 0$ to $\log k - 1$ **do**
3. apply ListMerge to pairs of lists (keep track of partial sums)
4. **if** remaining list after level $\log k - 1$ is nonempty
5. **then** output $(\ell_1, \ldots, \ell_k)$ else output "No Solution"

Running Time

Final list has integers in the range $[-\frac{mp^{\log k}}{2}, \frac{mp^{\log k}}{2}) = [-\frac{1}{2}, \frac{1}{2})$

Generating and storing initial k lists costs $\widetilde{O}(\alpha/p)$ time and space

Applying ListMerge $2k$ times costs $\widetilde{O}(k \cdot \alpha/p)$ time and space

Total running time is $\widetilde{O}(k\alpha \cdot m^{1/\log k})$

Correctness: enough to show there is a $c \in L$ in interval $[-b - \frac{mp^{\lambda+1}}{2}, -b + \frac{mp^{\lambda+1}}{2})$

Row Distinct

Let L_1, L_2 be lists at some level of the algorithm.

Suppose we organize elements of $L_1 + L_2$ into a table, so that $\ell = b + c$ is in row corresponding to b and column corresponding to c.

Call $\ell_1, \dots, \ell_N$ *row-distinct* if each appears in a different row.

Correctness Proof Sketch

1. Show the distributions of elements of L_1, L_2 at level λ are close to uniform.
2. Show that elements of L_2 are close to independent, assuming they were row distinct at the previous level.
3. Apply a martingale tail bound theorem to show that for fixed $b \in L_1$, there exists with high probability a $c \in L_2$ so that $b + c \in [-\frac{mp^{\lambda+1}}{2}, \frac{mp^{\lambda+1}}{2})$.
4. Apply the union bound to prove that with high probability each row has at least one element in the restricted interval.

Outline

Close to Uniform

Let U be the uniform distribution on $\mathbb{Z}/m\mathbb{Z}$

Let X be the distribution given by $X(\mathbf{x}) = \sum_{i=1}^{n} a_i x_i \mod m$

The statistical difference is defined by

$$\Delta(X, U) = \frac{1}{2} \sum_{a \in \mathbb{Z}/m\mathbb{Z}} |\Pr[X = a] - \Pr[U = a]|$$

Let $m = 2^{cn}$, $c < 1$. Call $\mathbf{a} = (a_1, \ldots, a_n)$ *well-distributed* if $\Delta(X, U) \leq 2^{-\frac{(1-c)n}{4}}$

Theorem (Impagliazzo, Naor)

The probability that $\mathbf{a}$ *is not well-distributed is less than* $2^{-\frac{(1-c)n}{4}}$.

k-set Algorithm for RMSS

Input: $a_1, \ldots, a_n$ from $\mathbb{Z}/m\mathbb{Z}$, target 0

Output: $\mathbf{x} \in \{0,1\}^n$ such that $\sum_{i=1}^n a_i x_i = 0 \mod m$

1. Partition indices $\{1, \ldots, n\}$ into k sets $I_1, \ldots, I_k$
2. Generate lists $L_1, \ldots, L_k$ of $n \cdot m^{1/\log k}$ elements. For each element of L_j, generate random bits x_i and store $\sum_{i \in I_j} a_i x_i$ along with bits
3. Apply k-set birthday algorithm to $L_1, \ldots, L_k$

Choosing $m = 2^{n^\epsilon}$, $k = \frac{1}{2} n^{1-\epsilon}$ gives corollary

Outline

Applications

1. Wagner's list of cryptographic applications has theoretical foundation

2. New message attacks on knapsack cryptosystems

3. Finding Carmichael numbers with large number of prime factors