The Subset-Sum Problem

Alexander May[1] Matthias Minihold[1,2]

[1]Horst Görtz Institute for IT Security, Ruhr-Universität Bochum, Germany
[2]ECRYPT-NET Early Stage Researcher

Motivation

Have you ever had a knapsack problem, i.e. when hiking or flying away? Informally the problem we try to solve is to pack a knapsack, too small to contain all items of a given set, with some of them, fulfilling a weight constraint.

More formally, suppose you are at the airport:

- Your luggage may weight S [kg] at check-in.
- Not squandering, your bag WILL weight exactly S [kg].
- You own n equally beautiful items of weight $a_1, a_2, \ldots a_n$.

The knapsack problem (also Subset-Sum problem) we want to solve:

$$\text{Given } n, S, a_1, a_2, \ldots a_n \in \mathbb{N}, \text{ find } I \subseteq [n] : \sum_{i \in I} a_i = S. \tag{1}$$

Historical Remarks

This problem was first studied in 1897 and was one of the first proven to be $\mathcal{NP}$-`complete` – worst-case instances are computationally intractable to tackle. The Subset-Sum problem appears on Karp's list of 21 $\mathcal{NP}$-`complete` problems. Unless $\mathcal{P} = \mathcal{NP}$, one cannot hope for a polynomial time Subset-Sum solver.

Algorithmic Evolution

In the following table one can see how the expected time/space requirements of algorithms solving (1) in hard cases evolved as the techniques were refined.

Algorithm (year)	Time	Space
Exhaustive Search	$2^{1.000n} \approx (2.000)^n$	$2^{0.000n} \approx (1.000)^n$
Horowitz-Sahni (1974)	$2^{0.500n} \approx (1.414)^n$	$2^{0.500n} \approx (1.414)^n$
Schröppel-Shamir (1979)	$2^{0.500n} \approx (1.414)^n$	$2^{0.250n} \approx (1.189)^n$
Howgrave-Graham-Joux 'representations' (2010)	$2^{0.337n} \approx (1.263)^n$	$2^{0.311n} \approx (1.241)^n$
Becker-Coron-Joux 'number-set' $\{-1,0,1\}$ (2011)	$2^{0.291n} \approx (1.223)^n$	$2^{0.291n} \approx (1.223)^n$
Bernstein-Jeffery-Lange-Meurer 'quantum algorithm' (2013)	$2^{0.241n} \approx (1.182)^n$	$2^{0.241n} \approx (1.182)^n$

Table 1: Expected time and space requirements of algorithms solving equation (1).

The currently best algorithm is a quantum algorithm, a lower bound is unknown.

Technique 1 - Meet in the Middle

Hard instances of the Subset-Sum problem are characterized by relatively large elements ($\log_2 a_i \approx n$) and a balanced solution, i.e. $|I| \approx \frac{n}{2}$ in Equation (1). Identifying subsets of $[n]$ with length n vectors x over the 'number-set' $\{0,1\}$ via $i \in I \Leftrightarrow x[i] = 1$ one constructs lists L_1, L_2 of pairs merged to a solution in L_0:

$$\boxed{L_0 \hat{=} \{(x_0, S_1 + S_2 = \textstyle\sum_i x_0[i]a_i = S)\}, wt(x_0) = \frac{n}{2}}$$

$$\boxed{L_1 \hat{=} \{(x_1, \textstyle\sum_i x_1[i]a_i)\}} \xrightarrow{\frac{n}{4}} L_0 \xleftarrow{\frac{n}{4}} \boxed{L_2 \hat{=} \{(x_2, S - \textstyle\sum_i x_2[i]a_i)\}}$$

$$\boxed{\{(x_{11}, S_{11})\}} \xrightarrow{\frac{n}{8}} L_1 \xleftarrow{\frac{n}{8}} \boxed{\{(x_{12}, S_{12})\}} \qquad \boxed{\{(x_{21}, S_{21})\}} \xrightarrow{\frac{n}{8}} L_2 \xleftarrow{\frac{n}{8}} \boxed{\{(x_{22}, S_{22})\}}$$

Figure 1: Schröppel-Shamir: Combining disjoint sub-problems of smaller weight.

Algorithms based on the birthday-paradox construct collisions in the second component of the sub-problems in the lists L_1, L_2 forcing any $x \in L_0$ to fulfill (1).

Technique 2 - Enlarge Number Set

The idea in Howgrave-Graham-Joux (2010) and Becker-Coron-Joux (2011) was to allow multiple representations, which at the same time enlarges the number-set

$$x_0[i] = x_1[i] + x_2[i] \notin \{0,1\}.$$

Although introducing a non-trivial filtering step to remove 'inconsistent solutions' when merging L_1 and L_2, overall speed-ups were achieved for $x_0[i] \in \{-1,0,1\}$.

$$\boxed{L_0 \hat{=} \{(x_0, \textstyle\sum_i x_0[i]a_i = S_1 + S_2 = S)\}, wt(x_0) = \frac{n}{2}}$$

$$=$$

$$\boxed{L_1 \hat{=} \{(x_1, \textstyle\sum_i x_1[i]a_i)\}, wt(x_1) = \frac{n}{4}}$$

$$+$$

$$\boxed{L_2 \hat{=} \{(x_2, S - \textstyle\sum_i x_2[i]a_i)\}, wt(x_2) = \frac{n}{4}}$$

Figure 2: Becker-Coron-Joux: Adding length n solutions of sub-problems increases the number-set.

Our Approach: Gaussian Sampling

The techniques reviewed above are:

- tricky to analyze,
- somewhat hard to generalize,
- produce exponentially many inconsistent solutions,
- thus require a non-negligible amount of intermediate filtering.

Instead of approaching an instance of Subset-Sum with combinatorial methods or quantum algorithms, we want to solve (1) with a classical algorithm using probabilistic tools. Gaussian sampling is a possible approach to overcome the combinatoric ad-hoc analysis while allowing any number-set in theory. The figure shows how sampling from a Gaussian distribution, $x[i] = X \sim \mathcal{N}(\mu = \frac{1}{2}, \sigma = \frac{8}{10})$, naturally leads to a number-set exceeding $\{0,1\}$. This happens with a certain probability.

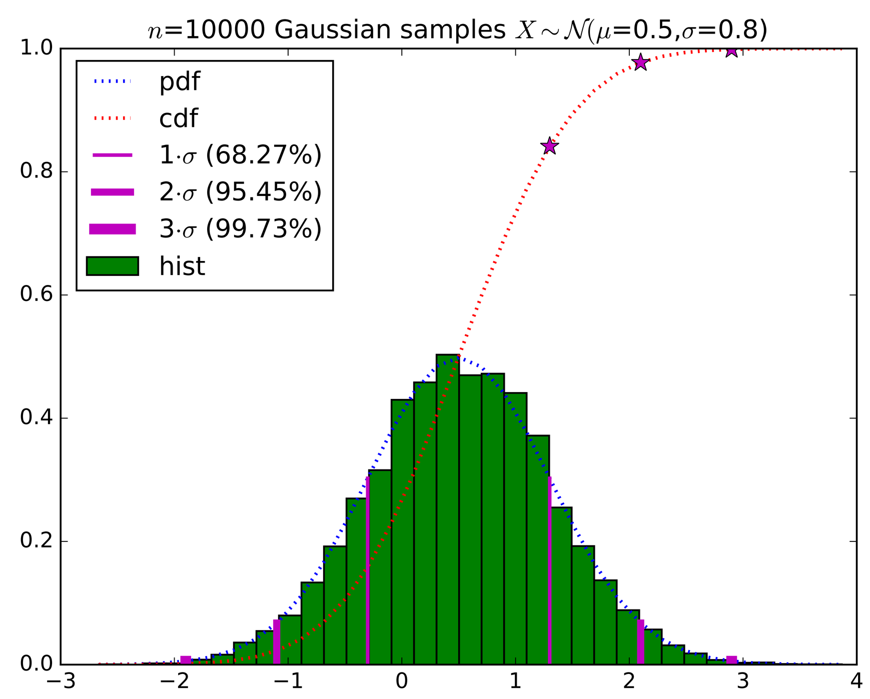

Figure 3: Histogram of samples $X \sim \mathcal{N}(\mu = \frac{1}{2}, \sigma = \frac{8}{10})$.

We strive for algorithmic speed-ups by relaxing the constrained number-set, thus accepting components $x[i]$ with a certain probability $P[x[i] \notin \{0,1\}]$ while ultimately ensuring a valid solution of Equation (1).

Applications

The cryptanalytic methods for structurally approaching the Subset-Sum problem are valuable algorithmic meta-techniques also applicable to other $\mathcal{NP}$-`complete` problems like lattice- or code-based problems.

Acknowledgements

This project has received funding from the European Union's Framework Programme for Research and Innovation Horizon 2020 (2014-2020) under the Marie Skłodowska-Curie Grant Agreement No. 643161.