Gitterbasierte Kryptosysteme

SVP, CVP und SBP

Die Abkürzungen stehen für Shortest Vector, Closest Vector Problem und Smallest Basis Problem. Hierbei handelt es sich um folgende algorithmische Probleme.

1 Definition. Sei $\Lambda \subseteq \mathbb{Z}^n$ ein Gitter. Das SVP besteht darin, aus einer Basis von Λ einen Vektor $v \in \Lambda \backslash \{0\}$ mit $||v||$ minimal auszurechnen.

2 Definition. Sei $\Lambda \subseteq \mathbb{Z}^n$ ein Gitter. Das CVP besteht darin, aus einer Basis von Λ und einem $w \in \mathbb{Z}^n$ einen Vektor $v \in \Lambda$ mit $||v - w||$ minimal auszurechnen.

3 Definition. Sei $\Lambda \subseteq \mathbb{Z}^n$ ein Gitter. Das SBP besteht darin, aus einer Basis von Λ eine Basis von Λ zu berechnen, so daß das Produkt über die Längen der Basisvektoren minimal wird. Man kann anstelle des Produkts auch das Maximum der Längen der Basisvektoren betrachten.

Vom SVP und CVP gibt es auch Approximations- beziehungsweise Entscheidungsvarianten, ebenso für das SBP. Ist zusätzlich $d \in \mathbb{Z}$ gegeben, so kann man im SVP nach $v \in \Lambda \backslash \{0\}$ mit $||v|| \leq d$ und im CVP nach $v \in \Lambda$ mit $||v - w|| \leq d$ fragen. Der Quotient der Größe d und des jeweiligen möglichen Minimalwerts für $||v||$ beziehungsweise $||v - w||$ heißt dann Approximationsfaktor. Interessiert man sich nur für die Existenz geeigneter v, so handelt es sich um Entscheidungsprobleme. Das SVP und CVP kann darüber hinaus auch für andere Normen betrachtet werden.

Approximations- bzw. Entscheidungs-SVP, -CVP und -SBP liegen offenbar in NP, wenn man als Zeugen den gesuchten Vektor v bzw. die Basis verwendet. Das CVP is außerdem NP hart und das SVP NP hart unter randomisierten Reduktionen. Das SBP ist ebenfalls NP hart (evtl. unter randomisierten Reduktionen). Es ist daher naheliegend, zu fragen, ob diese Probleme für kryptographische Zwecke verwendet werden können.

NP und NP hart

Wir wollen kurz informell auf die Begriffe NP und NP hart eingehen.

Unter einem Problem verstehen wir die Aufgabe, zu einer Relation $\sim \subseteq \{0,1\}^* \times \{0,1\}^*$ und Eingabedaten x mögliche Ausgabedaten y mit $x \sim y$ zu berechnen. Ist $x, \sim$ gegeben, so spricht man von einer Instanz des Problems. Bei einem Entscheidungsproblem handelt es sich bei der Relation um eine Funktion nach $\{0,1\}$. Für Entscheidungsprobleme kann man $L_\sim = \{x \mid x \sim 1\}$ definieren. Dann nennt man $L_\sim$ eine Sprache und das Entscheidungsproblem für x besteht darin, $x \in L_\sim$ oder nicht zu entscheiden. Es stellt sich die Frage, ob und in welcher Zeit Probleme durch Algorithmen (Turingmaschinen) gelöst werden können.

P steht für polynomial time. Es handelt sich hierbei um die Klasse der Entscheidungsprobleme, welche durch eine deterministische Turingmaschine in Polynomzeit in der Länge der Eingabedaten des Problems gelöst werden können. Eine solche Maschine ist im wesentlichen ein Programm, welches in jedem Schritt genau eine mögliche Ausführung hat.

NP steht für non-deterministic polynomial time. Es handelt sich hierbei um die Klasse der Entscheidungsprobleme, welche durch eine nicht-deterministische Turingmaschine in Polynomzeit in der Länge der Eingabedaten des Problems gelöst werden können. Eine solche Maschine ist im wesentlichen ein Programm, welches in jedem Schritt mehrere Möglichkeiten der Ausführung hat. Dies liefert (theoretisch) einen Ausführungsbaum, und man fordert, daß die Entscheidung durch einen Weg durch den Baum in polynomiell vielen Schritten berechnet werden kann. Alternativ kann man auch eine deterministische Turingmaschine verwenden, welche mit Hilfe einer zusätzlichen polynomiell großen Eingabe die Entscheidung in Polynomzeit berechnen kann. Diese zusätzliche Eingabe wird Zeuge genannt und spezifiziert sozusagen den Weg durch den Ausführungsbaum der nicht-deterministischen Turingmaschine. Eine der großen Vermutungen ist P $\neq$ NP. Dies führt manchmal zu dem Irrtum, NP stünde für non-polynomial time.

Unter einer Reduktion eines Problems A auf ein Problem B versteht man im allgemeinen ein Verfahren, welches eine Lösung einer Instanz von A mit Hilfe von Lösungen von Instanzen von B berechnen kann. Die jeweils zulässigen Verfahren werden anhand der üblichen Eigenschaften (deterministisch, probabilistisch, Laufzeit, wann und wie oft wird eine Lösung einer Instanz von B benötigt, etc.) eingeteilt. Sind A und B Entscheidungsprobleme und gibt es eine Abbildung $f : \{0,1\}^* \to \{0,1\}^*$ mit $x \in L_A \Leftrightarrow f(x) \in L_B$, welche mit einer deterministischen Turingmaschine in Polynomzeit berechnet werden kann, spricht man von Karp Reduktion. Insbesondere wird hier keine Instanz von B gelöst. Allgemeiner spricht man von Cook Reduktion,

wenn das Verfahren eine deterministische Turingmaschine ist, welche in Polynomzeit läuft und Instanzen von B per Orakel löst. Meist wird wohl Karp Reduktion verwendet.

Man kann P und NP auch für allgemeinere Probleme als Entscheidungsprobleme definieren. Bei den Reduktionen muß man dann auch Verfahren für die Ausgaben berücksichtigen. Solche allgemeineren NP Probleme können häufig in Polynomzeit auf NP Entscheidungsprobleme reduziert werden (zum Beispiel binäre Suche nach dem kürzesten Gittervektor).

NP harte Entscheidungsprobleme sind Probleme, auf die sich jedes Entscheidungsproblem in NP Karp reduzieren läßt. Liegt ein NP hartes Problem in NP, so heißt es NP vollständig. Diese sind die „schwersten Probleme“ in NP.

GGH Kryptosystem

Die Abkürzung GGH steht für die Erfinder Goldreich, Goldwasser und Halevi. Wir wählen $n \in \mathbb{Z}^{\geq 1}$, unabhängige, kurze Vektoren $b_1, \ldots, b_n \in \mathbb{Z}^n$ und setzen $\Lambda = \mathbb{Z}b_1 + \cdots + \mathbb{Z}b_n$. Außerdem werden Parameter $p, q \in \mathbb{Z}$ gewählt.

Der geheime Schlüssel ist $B = (b_1, \ldots, b_n)$. Der öffentliche Schlüssel ist $V = (v_1, \ldots, v_n)$ mit einer Basis v_i von Λ, welche durch ausreichend viele, zufällige unimodulare Transformationen aus den b_i entsteht. Eine andere Wahl für die v_i besteht in der Hermite Normalform von B.

Verschlüsseln: Die Nachricht wird geeignet in einen Vektor $m \in \mathbb{Z}^n \cap [-q, \ldots, q]$ abgebildet. Ein geeignetes, kleines $e \in \mathbb{Z}^n \cap [-p, p]$ wird gewählt. Der Chiffretext ist $c = Vm + e$.

Entschlüsseln: Die Idee ist, den zu c nächsten Vektor in Λ zu bestimmen. Wenn e klein genug ist, sollte aufgrund der Diskretheit von Λ der Vektor Vm herauskommen, und man kann nach m auflösen. Zur Berechnung des CVP verwendet man als Hintertür die Basis B wie folgt. Man stellt c in der Basis B dar, rundet die Koordinaten und hofft, so den zu c nächsten Vektor zu finden, welcher Vm sein sollte. Man berechnet also $m' = B^{-1}V\lfloor V^{-1}c \rceil$ als entschlüsselte Nachricht.

Es läßt sich zeigen, daß bei geeigneter Wahl der Parameter, insbesondere p, die Wahrscheinlichkeit für $m' = m$ ausreichend gut ist. Typische Parameter sind beispielsweise $n = 300$, $q = 127$ und $p = 3$.

Das GGH Kryptosystem kann mit der ElGamal Verschlüsselung wie folgt verglichen werden. Ein Chiffretext in ElGamal ist $c = (g^r, my^r)$, wobei $y = g^x$ der öffentliche und x der geheime Schlüssel ist. Die Nachricht m wird mit einem Fehler y^r perturbiert. Um diesen Fehler wieder wegrechnen zu können, schickt man noch g^r mit. Allerdings kann nach Annahme nur der Besitzer von x aus g^r auch $(g^r)^x = y^r$ ausrechnen.

Demgegenüber wird g^r im GGH System nicht benötigt. Die Nachricht m wird wieder durch einen Fehler e perturbiert. Durch das CVP kann man allein aus c den Fehler wieder wegrechnen, allerdings nach Annahme nur, wenn man im Besitz des privaten Schlüssels B ist.

Die Analogie zwischen ElGamal und GGH wird noch auffälliger, wenn man die Nachricht nach e und den Fehler nach m bringt, so daß der Chiffretext $c = Vm + e$ ist.

Sicherheit

Die Sicherheit des Verfahrens beruht auf zwei unterliegenden Problemen. Zum einen, daß man nicht B aus V berechnen kann. Dies entspricht im wesentlichen einer Instanz vom SBP. Zum anderen, daß man nicht Vm aus $c = Vm + e$ ausrechnen kann. Dies entspricht im wesentlichen einer Instanz des CVP.

Aufgrund der speziell erforderlichen Parameterwahlen konnte gezeigt werden, daß man $n \geq 400$ benötigt, um nicht eines dieser Probleme mit dem LLL Algorithmus und Varianten davon lösen zu können. Für solche n ist das Verfahren aber nicht mehr praktikabel.

GGH Signaturverfahren

Die Funktion $(m, e) \mapsto Vm + e$ verhält sich wie eine Einwegfunktion mit Falltür. Ähnlich wie bei RSA kann man daraus auch ein Signaturverfahren erhalten. Öffentlicher und geheimer Schlüssel sind wie beim Kryptosystem. Die Parameter werden teilweise anders gewählt.

Signieren: Die Nachricht wird in einen geeigneten Vektor $m \in \mathbb{Z}^n$ abgebildet. Mithilfe der geheimen Basis wird ein naheliegender Gitterpunkt $\sigma \in \Lambda$ als Linearkombination der öffentlichen Basis bestimmt. Dies ist die Unterschrift.

Verifizieren: Es wird überprüft, ob $\sigma \in \Lambda$ gilt und $||\sigma - m||$ klein genug ist.

Sicherheit

Dieses Verfahren ist ebenfalls nicht sicher, denn ein adaptiver chosen-message Angriff ist möglich. Durch eine Anzahl von Signaturanfragen kann der Angreifer ein Teilgitter von Λ definieren und mit dem Teilgitter Unterschriften fälschen.

Warum gitterbasierte Kryptographie?

Im wesentlichen basiert die Kryptographie mit öffentlichem Schlüssel auf der Schwierigkeit des DLP und der Faktorisierung. Zur Absicherung wäre es wünschenswert, auch auf andere schwierige Probleme zurückgreifen zu können. Die beiden ersten Probleme können auf einem Quantencomputer beispielsweise in Polynomzeit gelöst werden, wohingegen dies für SVP und CVP bisher nicht bekannt ist.

Ein weiterer Grund ergibt sich aus der Effizienz. Für den Sicherheitsparameter n ergibt sich folgende Tabelle. Die Sicherheit in der Tabelle ist natürlich

System	Schlüsselgröße	Laufzeit	Sicherheit
RSA	$O(n)$	$O(n^3)$	$L_n(1/3)$
DLP endl. Körper	$O(n)$	$O(n^3)$	$L_n(1/3)$
DLP ell. Kurve	$O(n)$	$O(n^3)$	$L_n(1)$
GGH	$O(n^2)$	$O(n^2)$	$L_n(1)$?
NTRU	$O(n)$	$O(n^2)$	$L_n(1)$?

in keinem Fall bewiesen. Jedoch erscheinen die Abschätzungen für RSA und DLP eher sicher als die für GGH und NTRU, da diese Probleme schon länger untersucht werden und sich auch nicht soviele „böse“ Überraschungen eingestellt haben. In gewisser Weise hat sich die gitterbasierte Kryptographie (siehe auch Merkle-Hellmann) als sehr instabil bezüglich der erforderlichen Parameterwahl gezeigt und ist auch sehr von statistischen Zusammenhängen abhängig.

Auf der anderen Seite sieht es so aus, aus würden Gittermethoden häufig zu sehr effizienten Verfahren führen, wenn sie denn sicher wären. Die Schlüsselgröße bei GGH ist allerdings in besonderem Maß dafür verantwortlich, daß GGH nicht praktikabel ist (Challenge mit $n = 400$ liefert 1.8MB Schlüssel).

NTRU

NTRU steht (inoffiziell) für „Number Theorists aRe Us“, da die Erfinder und Firmengründer Pipher, Hoffstein und J. Silverman Zahlentheoretiker sind (J. Silverman ist insbesondere Experte für elliptische Kurven).

Bei NTRU handelt es sich um das wohl einzige zur Zeit sichere und dabei sehr effiziente Kryptosystem basierend auf speziellen Gittern. Entwicklungsversuche für ein Signaturverfahren basierend auf diesen speziellen NTRU Gittern sind bisher fehlgeschlagen, jede Version wurde in relativ kurzer Zeit gebrochen.

Im Vergleich zum GGH Kryptosystem kann man folgendes festhalten. Die Verschlüsselung erfolgt ähnlich wie beim GGH Kryptosystem, eine Nachricht wird mit einem Fehlerterm perturbiert. Die Entschlüsselung basiert jedoch nicht auf dem Lösen eines CVP mittels einer kurzen, geheimen Basis als Falltür, sondern auf gewissen modulo q und modulo p Rechnungen unter Verwendung eines sehr kurzen, geheimen Vektors des NTRU Gitters als Falltür. Die Sicherheit des Verfahrens hängt daher (mindestens) von der Schwierigkeit des SVP im NTRU Gitter ab.

Die Schlüssel (die öffentliche und geheime Gitterbasen) im GGH Kryptosystem sind sehr speicheraufwendig. Im NTRU Gitter benötigt man jedoch aufgrund der speziellen rotationssymmetrischen Gestalt des Gitters nur je einen einzigen Vektor, um eine vollständige Basis (öffentlich und geheim) zu beschreiben. Daraus ergeben sich besonders speichereffiziente öffentliche und geheime Schlüssel.

Weitere Details des NTRU Verfahrens und Gitters werden in der Übung besprochen.

Ajtai-Dwork Kryptosystem

Dieses Kryptosystem ist nicht praktikabel (z.B. bitweise Verschlüsselung), ist aber von theoretischem Interesse, da es auf einer worst-case/average-case Verbindung von SVP Varianten beruht.

In der Kryptographie benötigt man Probleme (mit Falltür), deren zufällige Instanzen nur schwer zu lösen sind, also Probleme, die im Durchschnitt bzw. average-case schwer zu lösen sind. Im Gegensatz dazu weiß man von vielen Problemen wie SVP nur, daß es Instanzen gibt, welche schwer zu lösen sind. Dies sind also Probleme, welche im worst-case nur schwer zu lösen sind. Eine worst-case/average-case Reduktion von worst-case Instanzen eines schwierigen Problems A auf average-case Instanzen eines Problems B ist daher von großem Interesse. Mit einer solchen Reduktion könnte man worst-case Instanzen von A lösen, wenn man nur zufällige Instanzen von B lösen kann. Da A schwierig im worst-case ist, muß B schwierig im average-case sein.

Eine solche Reduktion wurde 1996 erstmalig von Ajtai gefunden. Beim average-case Problem B handelt es sich hier um ein approximiertes SVP in einer besonderen Klasse von Gittern (das NTRU Gitter weist gewisse Ähnlichkeiten zu diesem Gitter auf). Das worst-case Problem A ist eines der drei folgenden Probleme.

$A1$ Das SVP mit in der Dimension polynomiellem Approximationsfaktor,

$A2$ Eine Variante vom SBP in Dimension n, wo $\max_{i=1}^{n} ||b_i||$ mit in n polynomiellen Approximationsfaktor zu minimieren ist,

$A3$ Das SVP in Gittern Λ der Dimension n mit $\lambda_2(\Lambda) \geq n^c \lambda_1(\Lambda)$ und c eine ausreichend große, von n unabhängige Konstante.

Das Ajtai-Dwork Kryptosystem zeichnet sich durch die Eigenschaft aus: Wenn eine Verschlüsselung von 0 von einer Verschlüsselung von 1 in einer zufälligen Instanz des Systems mit von 1/2 signifikant abweichender Wahrscheinlichkeit unterschieden werden kann, dann läßt sich Problem $A3$ effizient lösen. Es gibt also im wesentlichen keine unsicheren Instanzen bzw. unsicheren Schlüssel dieses Kryptosystems. Das Problem, einen Chiffretext zu entschlüsseln, ist aber vermutlich nicht NP hart.

Aus praktischer Sicht gilt die Aussage über die unsicheren Schlüssel auch für RSA und DLP basierte Systeme.

Aussagen über das SVP und CVP

Hier sind ein paar Aussagen über die worst-case Schwierigkeit des SVP und CVP.

- Das SVP ist NP hart bezüglich randomisierter Reduktionen und konstantem Approximationsfaktor.
- Das CVP in Dimension n ist NP hart bezüglich subpolynomiellem Approximationsfaktor $n^{c/\log(\log(n))}$ für eine absolute Konstante $c > 0$.
- Das SVP läßt sich effizient auf das CVP in der gleichen Dimension und bezüglich des gleichen, beliebigen Approximationsfaktors reduzieren.
- Das SVP und CVP in der Dimension n mit Approximationsfaktor $n^{1/2}$ ist wahrscheinlich nicht NP hart.
- Das SVP und CVP können mit exponentiellem Approximationsfaktor $(1+\varepsilon)^n$ für festes $\varepsilon > 0$ in Polynomzeit in n gelöst werden (Schnorr's LLL Variante). Häufig sind die erreichten Approximationsfaktoren deutlich besser als exponentiell.

Für andere Normen sind weitere Aussagen bekannt, das CVP ist beispielsweise für jede p-Norm $|| \cdot ||_p$ NP hart. Das SVP scheint theoretisch etwas unzugänglicher als das CVP zu sein.