GPU Parallelization of Cryptographic Primitives using Multivariate Quadratic Polynomials and its Security Evaluation

田中, 哲士

https://doi.org/10.15017/1500750

氏　　名　：田中哲士

論文題名　：GPU Parallelization of Cryptographic Primitives
using Multivariate Quadratic Polynomials and its Security Evaluation
（多変数二次多項式を用いた暗号プリミティブと
その安全性評価のGPU並列化）

区　　分　：甲

論　文　内　容　の　要　旨

ストリーム暗号は，平文と乱数生成された鍵ストリームとの排他的論理和を取る事で，暗号化を行う手法である．多くのストリーム暗号では，擬似乱数生成器（Pseudo-Random Number Generator, PRNG）に線形フィードバックシフトレジスタや攪拌アルゴリズムを利用しており，計算量の少ないままデータの攪拌が可能である．一方で，使用するPRNGの安全性を数学問題の解決困難性に帰着させ，公開鍵暗号と同等の安全性証明を持たせる種類のストリーム暗号の研究が行われている．多変数二次多項式の付値を暗号プリミティブとするQUADは，その種のストリーム暗号の一つであり，安全性の根拠として有限体上の連立二次方程式の求解問題 (Multivariate Quadratics, MQ)を利用しているため信頼性が高いが，反面，QUADは鍵ストリームの生成に数万回以上の計算を必要とし計算コストが高い．この為，QUADを実用化するためには計算の効率化が研究課題となる．

計算を効率化する手法の一つとして並列処理がある．一般に，多項式は各々独立に付値の計算が可能であり，多項式内の項も各々独立である．従って，多項式の付値は独立に計算可能な部分が多く，並列計算に適している．並列計算の実現手法として，GPU(Graphics Processing Units)を利用した手法が注目されている．現在のGPUは1,000個を超えるコアを有しており，並列計算に基づいた高い処理能力を有している．反面，GPU実装はスレッドの実行タイミング，メモリローディング等，動作上においてCPU実装とは異なる制約があり，GPUに適した実装を行う事が課題となる．

本学位論文は以下のように構成される．

第1章では，本研究の背景と目的を述べる．また，本研究の主要な挑戦課題と貢献についても論じる．

第2章では，QUADの基礎となっている有限体上の多項式，及び，既存の多変数公開鍵暗号の手法について説明を行う．また，GPU上で用いる実装環境について説明し，本論文のGPU実装で用いるCUDA(Compute Unified Device Architecture) APIの説明を行い，暗号における並列実装手法及びGPUを用いた既存の実装成果についても説明する．

第3章では，QUADで用いるGF(2)上の多変数二次多項式の付値に対する並列化を行った．具体的には，GF(2)上の多項式の付値は，非零な項の係数の加算のみで計算可能なことに着目し，付値の手順を(i)多項式内の非零な変数の判定，(ii)非零な変数と対応する多項式の係数の読み取り，(iii)係数の総和の3ステップに分割した．このとき，(i)は判定が逐次的であるため，逐次処理に強いCPUで，並

列処理可能な(ii),(iii)はGPU上で実装した．また，(iii)では並列リダクションと呼ばれる手法を利用した. 更に，ビットスライス戦略により，一度に32の多項式を取り扱う事で計算の回数を削減した．最終的に，本章の実装手法を用いる事で，GF(2)上の160変数320多項式のQUADにおいて約12Mbpsの速度を達成した．これは，CPU実装と比較して約18倍高速であった．

第4章では，拡大体上における多変数二次多項式の付値に対する並列化を行った．拡大体を利用する事で，多変数多項式のパラメータである変数の数を小さくとる事が可能である．本章では，拡大体上の多項式に対する付値を(i)多項式間で共通する項の乗算，(ii)多項式内の項と係数の乗算，(iii)多項式内の項の総和の計算，という3ステップに分割した．各ステップは並列処理可能でありGPU上で実装した．(iii)は第３章と同様に，並列リダクションを利用した．また，(i),(ii)では拡大体上の乗算が必要となるため，効率的な乗算手法について比較を行った．GF(2^32)の乗算手法について6つの手法を比較した結果，ビットスライス戦略を用いたものが最速であった．更に，GPUのメモリローディングに適したデータ構造を提案した．最終的に，本章の実装手法を用いる事で，GF(2^32)上で48変数96多項式のQUADにおいて約25Mbpsの速度を達成した．これは，CPU実装と比較して約90倍高速であった．

第5章では，多変数二次多項式の付値に線形回帰数列を用いたPetzoldtの手法に対する並列化を行った．Petzoldtは多項式の係数に関係性を持たせ，複数の乗算を同時に処理する事で効率化を行っている．しかし，彼の手法は逐次的なステップが多く，単純な並列化では効率化ができない．そこで，複数の暗号を同時に実行する事により，１回の処理で生成される鍵ストリームの数を増やすマルチストリーム手法を導入した. 更に, 第４章のデータ構造を拡張し, マルチストリームを用いたPetzoldtの手法に適したデータ構造を提案した．最終的に，GF(2^32)上の32変数64多項式のQUADを256個のストリームで同時に実行する事により，約190Mbpsの実装速度を達成した．

第6章では，QUADの安全性の根拠となるMQ問題を解決する拡張線形化アルゴリズム（eXtended Linearization, XL）の並列化を行った．XLは解決可能な線型方程式を構成とその線型方程式の解決の，2つのステップで構成される．線型方程式の解決には，方程式の係数行列の逆行列を求める手法が知られているが，XLが構成する行列は疎であるため，疎行列に適したアルゴリズムを用いることによる効率化が期待できる．疎行列に適したアルゴリズムの中でも，Wiedemannアルゴリズムは並列性が高く, 8PCクラスタを用いた並列実装によるMQ問題の解決結果が知られている. 本研究では，Wiedemannアルゴリズムの主要な3ステップ(i)数列の生成，(ii)数列に対する最小多項式の導出，(iii)解ベクトルの導出について，並列性の高い(i),(iii)をGPU，逐次性の強い(ii)をCPUで実装した．最終的に，GF(7)上24変数の48連立二次方程式を約10時間で解決した．これは，CPU実装と比較して約38倍の高速化に相当する．

第7章では，本研究の結論と今後の研究課題について述べる．