Toffoli Depth Optimized Quantum Multiplication

장경배

https://youtu.be/35jFwt5cnmA

한성대학교 HANSUNG UNIVERSITY

CryptoCraft LAB

D. Maslov et al.

ON THE DESIGN AND OPTIMIZATION OF A QUANTUM POLYNOMIAL-TIME ATTACK ON ELLIPTIC CURVE CRYPTOGRAPHY[a]

DMITRI MASLOV
Department of Combinatorics and Optimization, University of Waterloo
Waterloo, Ontario, N2L 3G1, Canada

JIMSON MATHEW
Department of Computer Science, University of Bristol
Bristol, BS8 1UB, UK

DONNY CHEUNG
Department of Computer Science, University of Calgary
Calgary, Alberta, T2N 1N4, Canada

DHIRAJ K. PRADHAN
Department of Computer Science, University of Bristol
Bristol, BS8 1UB, UK

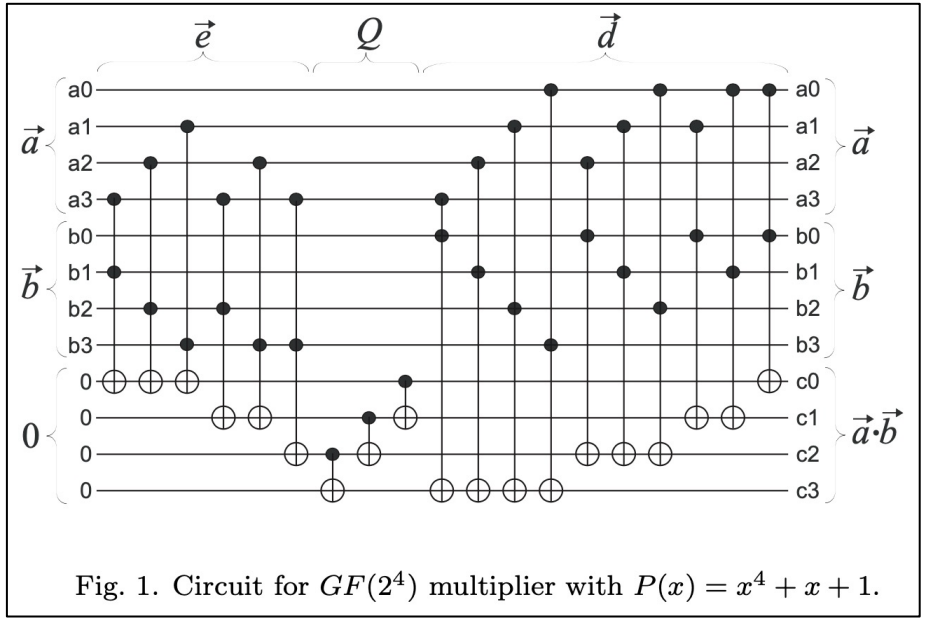


Fig. 1. Circuit for $GF(2^4)$ multiplier with $P(x) = x^4 + x + 1$.

- **일반적인 Schoolbook 곱셈**
- $h = f \cdot g$ $(size = n)$의 곱셈에 **$3n$개의 큐비트 사용**
- **Modular 연산 고려, 상위 곱셈부터 수행**
- **Toffoli 게이트 : n^2**

S. Kepley et al.

Quantum Inf Process (2015) 14:2373–2386
DOI 10.1007/s11128-015-0993-1

CrossMark

Quantum circuits for $\mathbb{F}_{2^n}$-multiplication with subquadratic gate count

Shane Kepley[1] · Rainer Steinwandt[1]

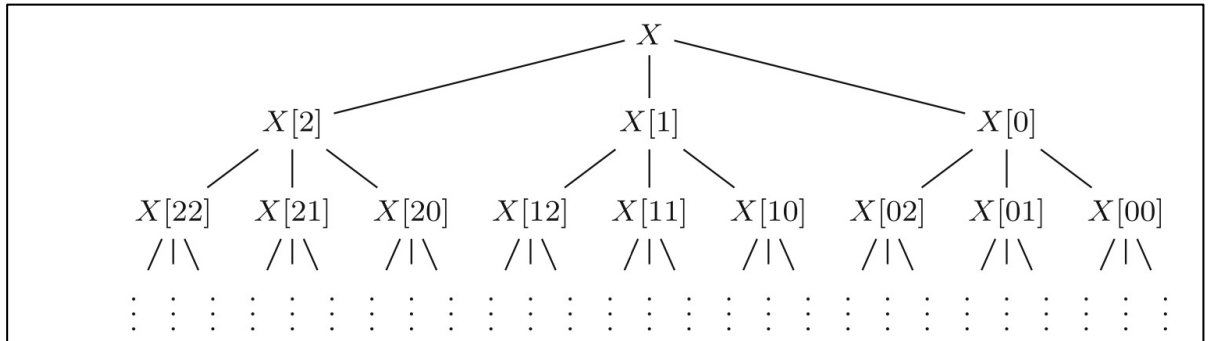


Fig. 1 Indexing in a ternary tree

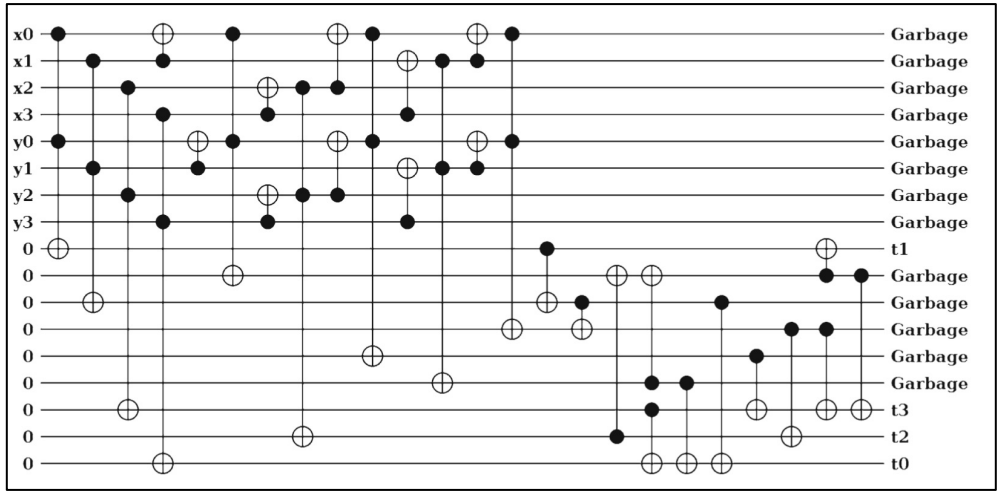
- **Karatsuba 곱셈**
- $h = f \cdot g$ $(size = n)$의 곱셈에 **Karatsuba 알고리즘을 재귀적으로 적용**
- **곱셈 시 CNOT 게이트가 추가적으로 사용되지만, Toffoli 게이트를 최대한 감소시킬 수 있음**
- **큐비트를 추가 사용 → (2n + 사용 된 Toffoli 게이트 수)**
- **Toffoli 게이트 :** $n^2 \cdot \frac{3}{4}^{\log_2 n}$

S. Kepley et al.

Example 3 For $\mathbb{F}_2[t]/(f)$ with $f = t^4 + t + 1$, before reduction we can form the appropriate coefficients in the product as follows:

j	z_j
0	C_0
1	$C_0 + C_1 + C_2$
2	$C_0 + \quad C_2 + C_3 + \quad C_6$
3	$C_0 + C_1 + C_2 + C_3 + C_4 + C_5 + C_6 + C_7 + C_8$
4	$\quad C_2 + \quad C_5 + C_6 + \quad C_8$
5	$\quad C_6 + C_7 + C_8$
6	$\quad C_8$

j	z_j
0	$C_0 + \quad C_2 + \quad C_5 + C_6 + \quad C_8$
1	$C_0 + C_1 + \quad + \quad C_5 + \quad C_7$
2	$C_0 + \quad C_2 + C_3 + \quad C_7$
3	$C_0 + C_1 + C_2 + C_3 + C_4 + C_5 + C_6 + C_7$



I. van Hoof

Space-efficient quantum multiplication of polynomials for binary finite fields with sub-quadratic Toffoli gate count

Iggy van Hoof

Technische Universiteit Eindhoven
i.v.hoof@student.tue.nl

→

Concrete quantum cryptanalysis of binary elliptic curves

Gustavo Banegas[1], Daniel J. Bernstein[2,3], Iggy van Hoof[4] and Tanja Lange[4]

[1] Chalmers University of Technology, Gothenburg, Sweden gustavo@cryptme.in
[2] University of Illinois at Chicago, Chicago, USA djb@cr.yp.to
[3] Ruhr University Bochum, Bochum, Germany
[4] Eindhoven University of Technology, Eindhoven, The Netherlands
i.v.hoof@tue.nl,tanja@hyperelliptic.org

- S. Kepley et al.와 동일하게 Karatsuba 알고리즘을 재귀적으로 적용하여 Toffoli 게이트를 최대한 줄임
- **차이점은 큐비트 수**
  - **LUP 분해를 활용하여**, Schoolbook 곱셈과 동일하게 **$3n$의 큐비트만을 사용**
  - CNOT 게이트를 조금 더 사용하긴 하지만, 그렇게 큰 단점은 아님

- Banegas, Bernstein의 **Binary ECC에 대한 Shor 알고리즘 적용 논문에서** 해당 곱셈 기법을 사용
  - S. Kepley의 곱셈보다 Toffoli Depth와 Full depth는 더 높을 것으로 보임

Ours

- **기존 연구들은 Depth를 신경 쓰지 않음**
- **이번 곱셈은 Toffoli depth 최적화 → Full depth 또한 감소**
- I. van Hoof의 곱셈(LUP 분해)을 사용하는 것은 X
  - Toffoli depth 최적화가 불가능
- 동일하게 Karatsuba 알고리즘을 재귀적으로 적용
  - 곱셈이 나눠지기 때문에 이것 만으로도 효과는 있음
  - 하지만 여전히 **종속성이 존재**
- **Karatsuba를 적용할 때 마다 생기는 종속성을 다 제거**
  - **추가 큐비트**
  - 끝까지 Karatsuba를 적용하고, 한 번에 곱셈
- 모든 Field size 2^n에 대해 **Toffoli depth 1** 곱셈 가능

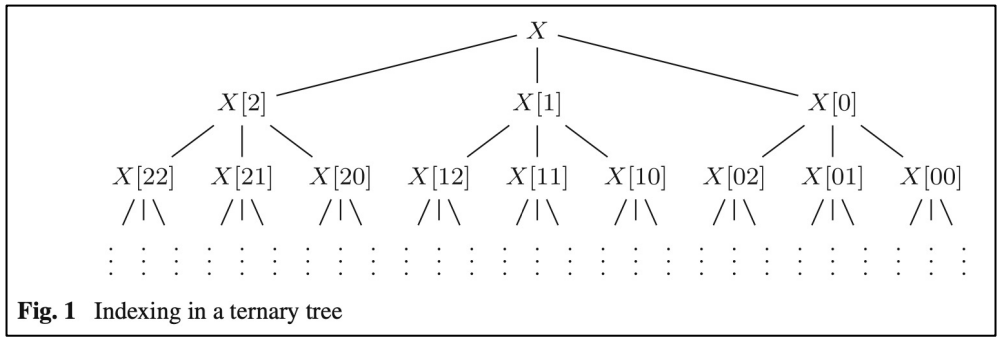
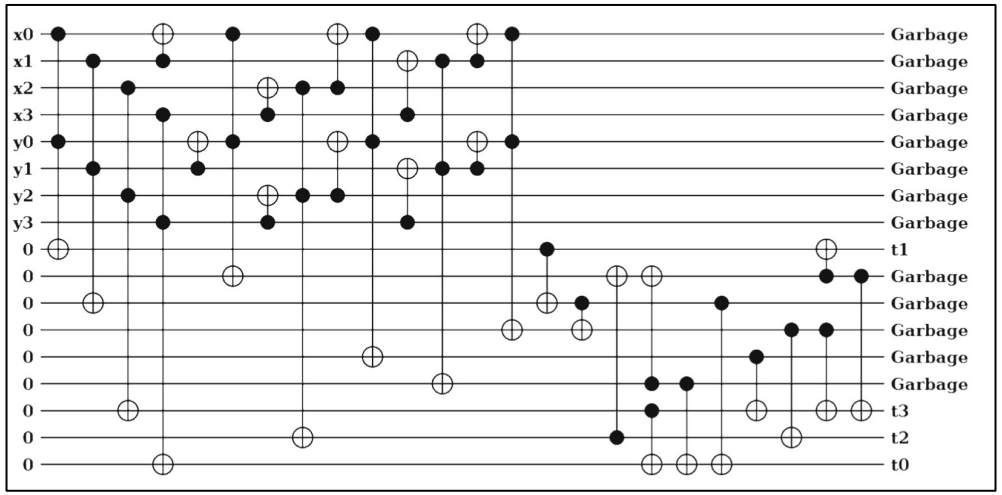


Fig. 1 Indexing in a ternary tree



S. Kepley et al.

Ours

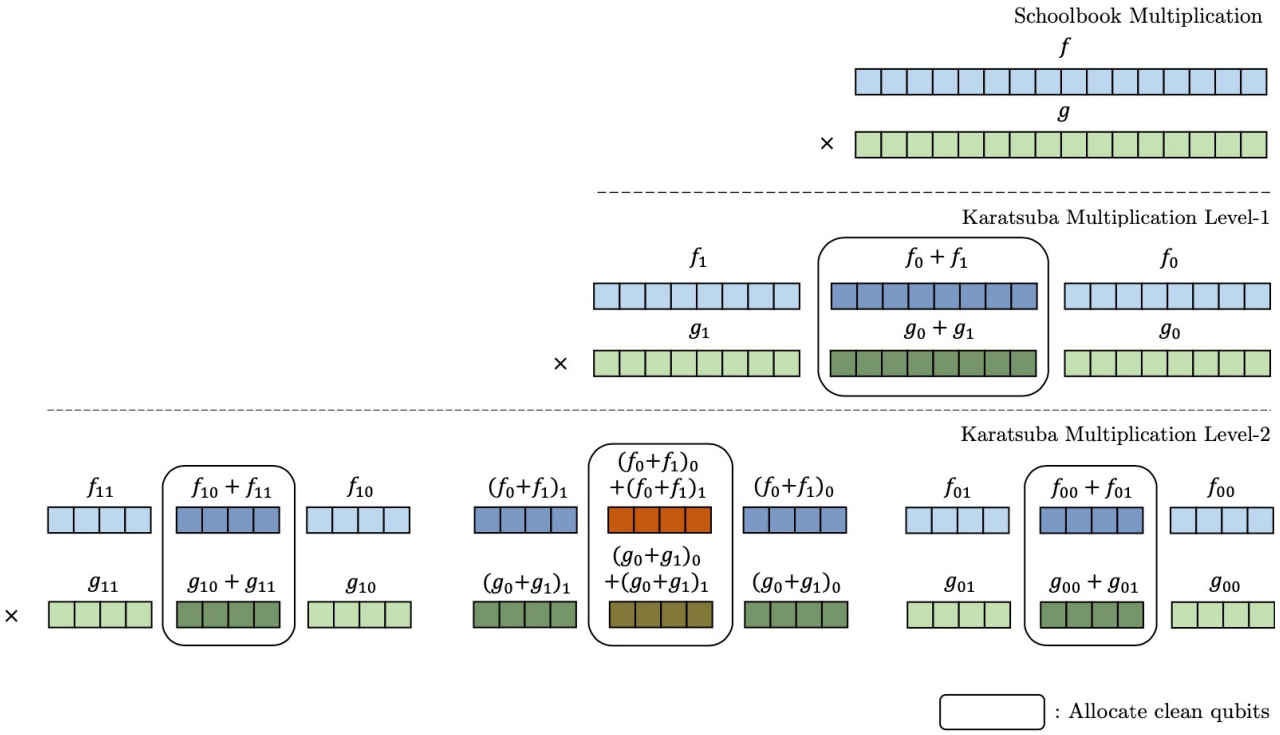


Table 1. Quantum resources required for $f \cdot g$ with Karatsuba Level-1

Field size 2^n	#CNOT	#Toffoli	Toffoli depth	#qubits	Full depth
Schoolbook	·	n^2	n^2	$4n-1$	$8n^2$
Karatsuba Level-1	$5n-4$	$3 \cdot (n/2)^2$	$(n/2)^2$	$3 \cdot (2n-1)$	$8 \cdot (n/2)^2 + 5$
Karatsuba Level-2	$(5n-4)+$ $3 \cdot (5n/2-4)$	$3^2 \cdot (n/2^2)^2$	$(n/2^2)^2$	$3^2 \cdot (n-1)$	$8 \cdot (n/4)^2 + 10$
Karatsuba Level-3	$(5n-4)+3 \cdot (5n/2-4)$ $+9 \cdot (5n/4-4)$	$3^3 \cdot (n/2^3)^2$	$(n/2^3)^2$	$3^3 \cdot (n/2-1)$	$8 \cdot (n/8)^2 + 15$

(Before Modular)

Ours

```python
def Karatsuba_Toffoli_Depth_1(eng) :

    n = 8
    a = eng.allocate_qureg(n)
    b = eng.allocate_qureg(n)
    r_a = eng.allocate_qureg(n)
    r_b = eng.allocate_qureg(n)
    c = eng.allocate_qureg(27)

    if (not resource_check):
        Round_constant_XOR(eng, a, 0xff, n)
        Round_constant_XOR(eng, b, 0xff, n)

    new_a = []
    new_b = []
    new_r = []

    for i in range(int(n/2)):
        CNOT | (a[i], r_a[i])
        CNOT | (a[int(n/2)+i], r_a[i])
        CNOT | (b[i], r_b[i])
        CNOT | (b[int(n/2) + i], r_b[i])

    new_a = karatsuba_mul_4bit(eng, a[0:4], b[0:4], c[0:9])
    new_b = karatsuba_mul_4bit(eng, a[4:8], b[4:8], c[18:27])
    new_r = karatsuba_mul_4bit(eng, r_a, r_b, c[9:18])

    # Combine
    result = []
    result = combine(eng, new_a, new_b, new_r, n)

    # modular

    if(not resource_check):
        print_state(eng, result, 2*n-1)
```

- Performance
  - Field 2^8 곱셈 (T + Clifford Level)

```
/Users/kb/PycharmProjects/
Result : 101010101010101

Gate counts:
    Allocate : 81
    CX : 300
    Deallocate : 81
    H : 54
    T : 81
    T^\dagger : 108

Depth : 23.
```

감사합니다