Information Set Decoding

https://youtu.be/g-J9RzLKqzI

장경배

한성대학교 HANSUNG UNIVERSITY

CryptoCraft LAB
https://crypto.modoo.at

코드기반암호

- 코드기반암호의 안전성은 아래 **신드롬 디코딩** 문제에기반함

Challenge

$$C_0 = He^T$$

암호문과 공개키는 알려진 정보 : $C_0,\ H$

특정 무게 조건의 벡터 e 를 찾아내는 문제 → **NP-hard**

- H 생성에 대한 비밀정보(개인키)를 가지고 있다면 암호문 C_0로부터 원본 벡터 e 를 복구할 수 있음

코드기반암호

공개키를 생성행렬 G 를 사용하면 McEliece 버전 → $C = mG' + e$

패리티체크 행렬 H 를 사용하면 Niederreiter 버전 → $C = He$

Weight $(e) = t$

Information Set Decoding(ISD)

- 코드기반 암호에 대해 가장 효율적인 공격법

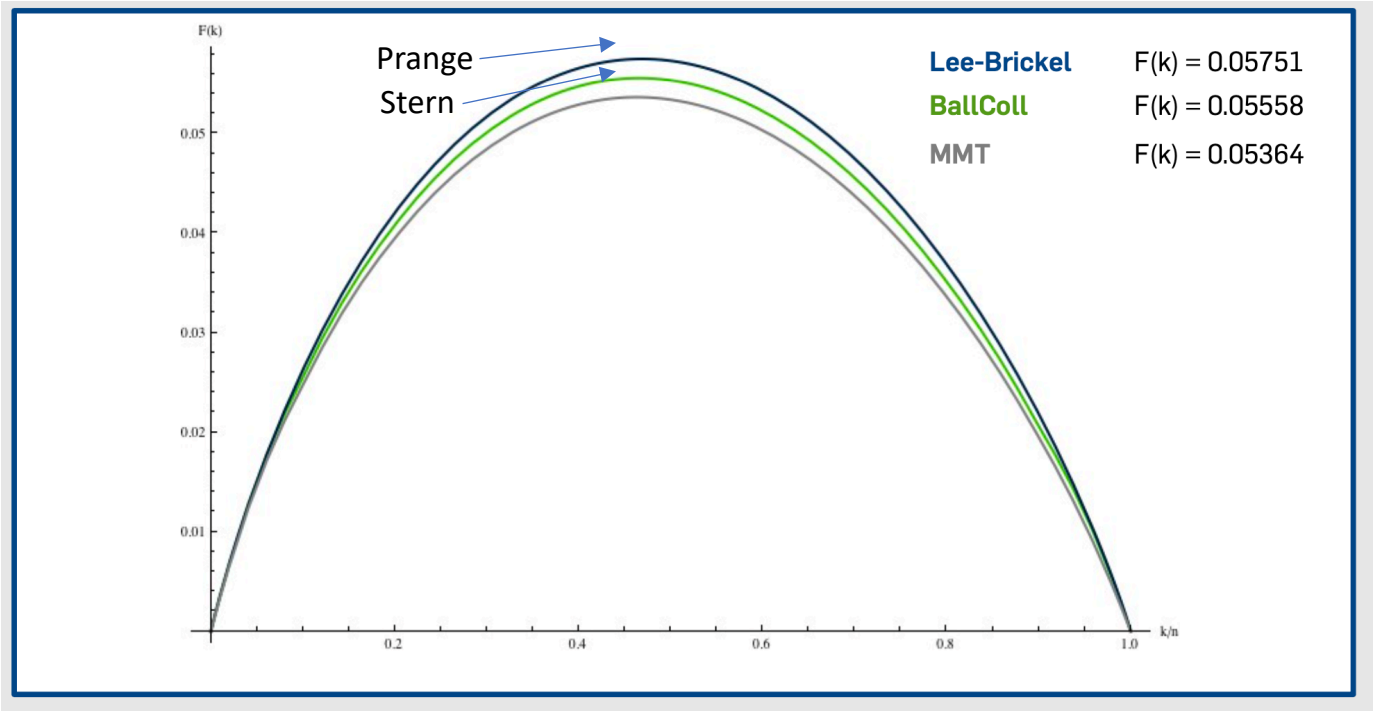
$$C_0 = He^T$$

  - 공개키 H 와 암호문 C_0 만을 가지고 원본 메세지를 복구함
    - 개인키를 찾아내진 않음

- 다른 공격법인 구조 공격(Structure attack)도 존재
  - 공개키 H 로부터 구조적 결함을 찾아 개인키를 복구하는 공격 방법
  - ISD 보다 성능이 좋지 않아 잘 연구되지는 않음

Information Set Decoding(ISD)

- Prange의 기본적인 ISD 부터 시작하여 이를 개선시키는 다양한 ISD 가 연구되고 있음
  - 획기적인 성능 향상을 보여주진 않으며 복잡도를 아주 조금씩 줄이는 정도



Information Set Decoding(ISD)

- Information Set Decoding
  - **Gaussian elimination** (1) + **Brute force** (2) 의 구조 → 1. Gaussian elimination을 통해 문제를 변형시킨 뒤,
    2. 해당 문제에서특정 조건을 만족하는 벡터를 찾을 때까지 반복
  - Quantum 버전은 **Brute force** (2)부분의 복잡도를 **절반**으로 줄임
    - Grover search 알고리즘 사용하여..

Prange's Algorithm (McEliece 버전)

- **Brute force** 만이 존재, Gaussian elimination **X**

$$C = mG' + e \ , \ \text{Weight}\ (e) = t$$

1. n – bit 의 길이 암호문 C 에서 k – bit 의 벡터 C_k 를 랜덤하게 선택 ($k \times n$ 행렬)

   → 오류 위치를 모르는 상태에서 자신이 선택한 k – bit 벡터에 오류가 포함되지 않아야 함

2. 선택한 열의 index에 맞춰 G' 으로부터 ${G'}_k$ 를 뽑아낸다. 이때, ${G'}_k$ 는 invertible

3. ${C_k\ {G'}_k}^{-1} = (mG' + e)_k\ {G'}_k^{-1} = m + e_k {G'}_k^{-1}$

   만약 e_k 가 0이라면 ((1번에서 조건), $e_k {G'}_k^{-1} = 0$ → 이에 대한 확인은 $\text{Weight}\ (C + \boxed{C_k {G'}_k^{-1}} G') <= t$

   찾고자 하는 m

   $(mG' + e)_k\ {G'}_k^{-1} = C_k {G'}_k^{-1}$ 이며,

   $mG' + e = C$ 이기 때문에, $C_k {G'}_k^{-1} = m$, 쉽게 원본 메시지를 찾을 수 있음

Information Set Decoding (ISD) (Niederreiter 버전)

Gaussian elimination + **Brute force**

- Gaussian elimination를 통해 패리티 체크 행렬을 아래와 같이 Systematic form으로 변경 가능함
  - Classic McEliece 의 경우, 이를 공개키로 사용

$$\hat{H} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}.$$

Gaussian elimination 적용

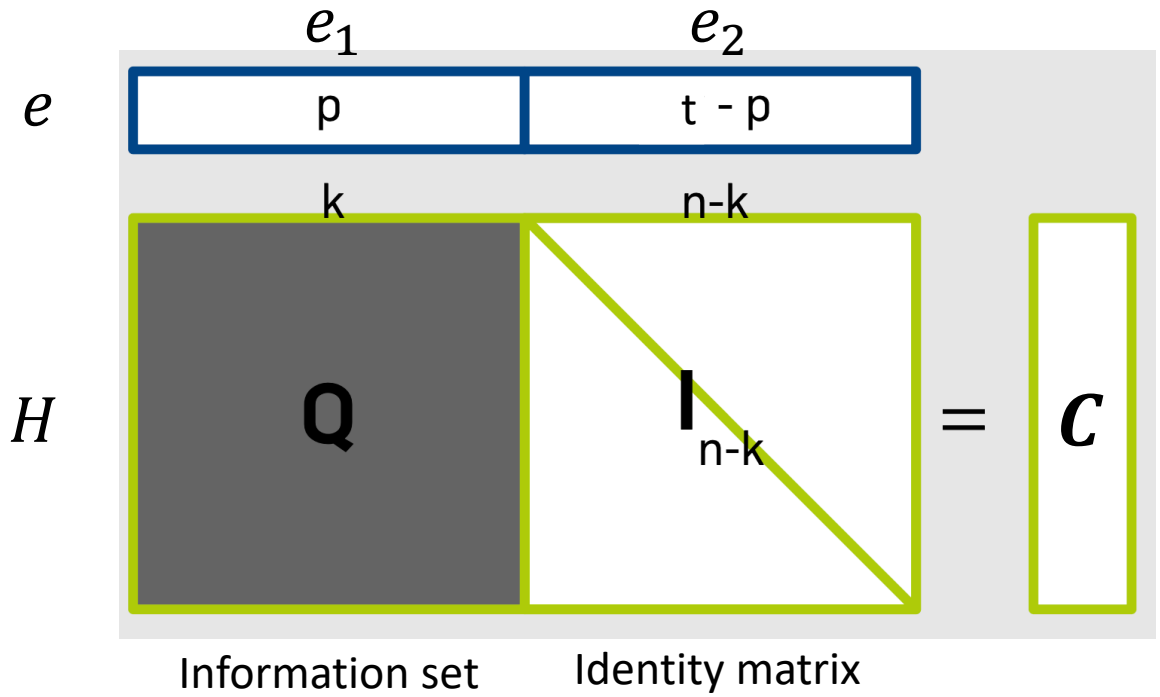
$$H = \left(\begin{array}{cccccccc|cccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{array}\right).$$

Identity matrix　　Information set

→ 오른쪽이 Identity matrix, 왼쪽을 information set으로도 가능

Lee-Brickell's Information Set Decoding (Niederreiter 버전)

- Lee-Brickell은 Prange의 알고리즘을 개선
  - Gaussian elimination을 적용, 아래와 같이 Systematic form을 구성
  - Information Set의 index에 해당하는 벡터 Weight(e_1) = p 를 허용, 최적의 파라미터 p는 행렬 크기에 따라 다름
  - 벡터 e의 무게 분포를 아래와 같이 설정, (information set 에는 p, identity matrix에는 t – p)

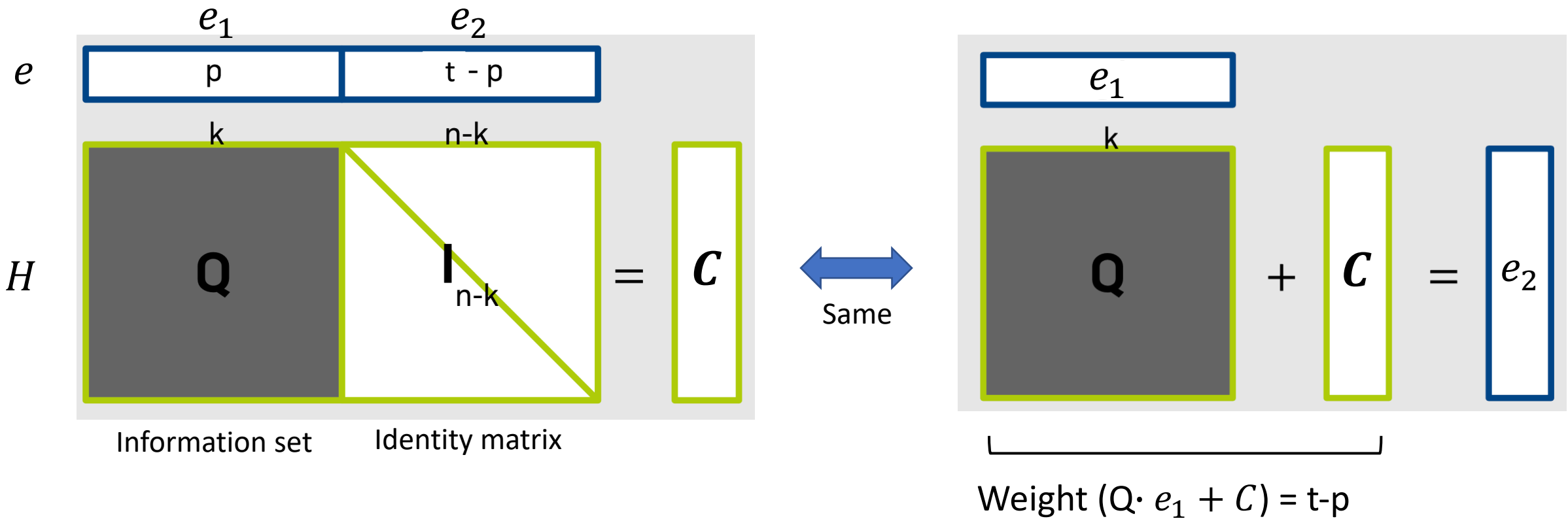


$$He = C$$

Weight(e) = t

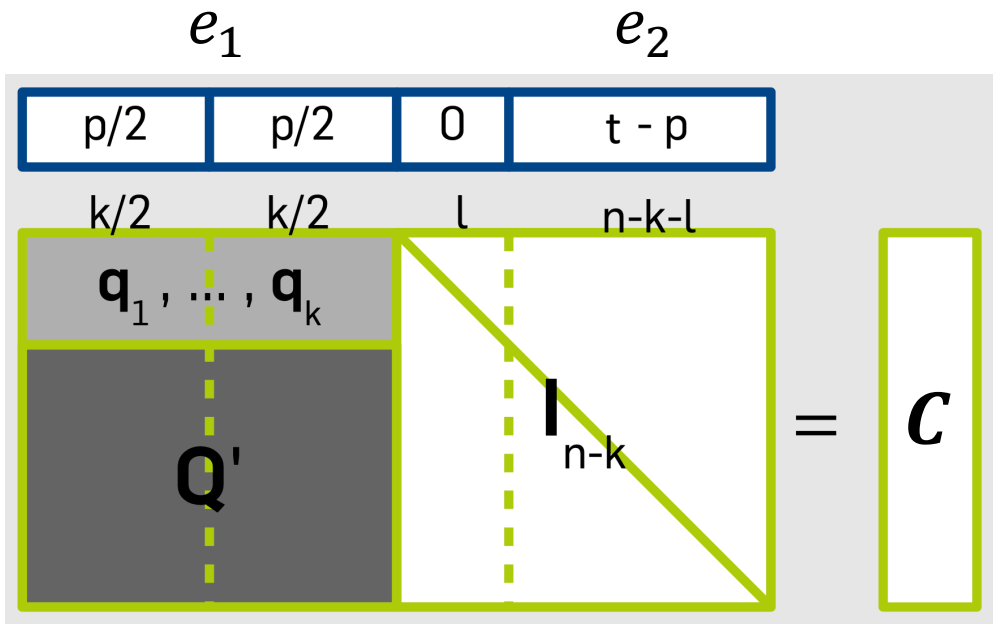
Lee-Brickell's Information Set Decoding (Niederreiter 버전)

- 이제 Q행렬에 대해 Brute force → Weight (Q· $e_1 + C$) = t-p 를 만족하는 e_1을 찾음
- 이를 만족한다면, e_2는 (Q· $e_1 + C$)값의 1 에 맞춰 벡터를 구성 가능, → 무게, 신드롬 값도 일치 (오른쪽 그림 참고)
- 만족하는 벡터 e_1 를 찾지 못했다면 처음부터(Gaussian elimination) 다시 반복



Stern's Information Set Decoding (Niederreiter 버전)

- Information set에서 MITM(Meet In the Middle) 적용
  - Gaussian elimination을 적용, 아래와 같이 Systematic form을 구성
  - 벡터 e의 무게 분포를 아래와 같이 설정



Information set　　Identity matrix

$$He = C$$

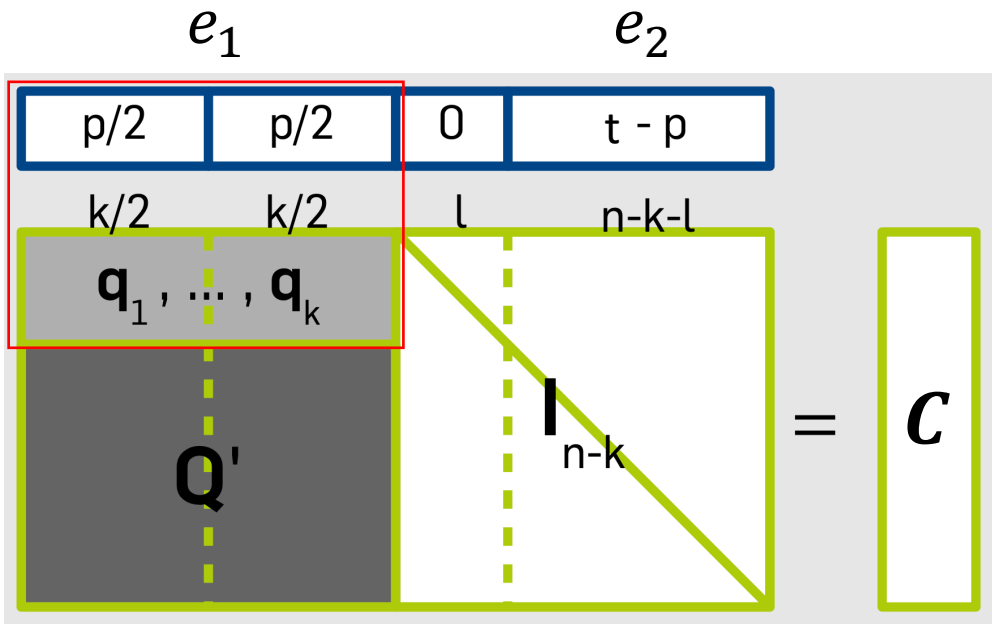
Weight(e) = t

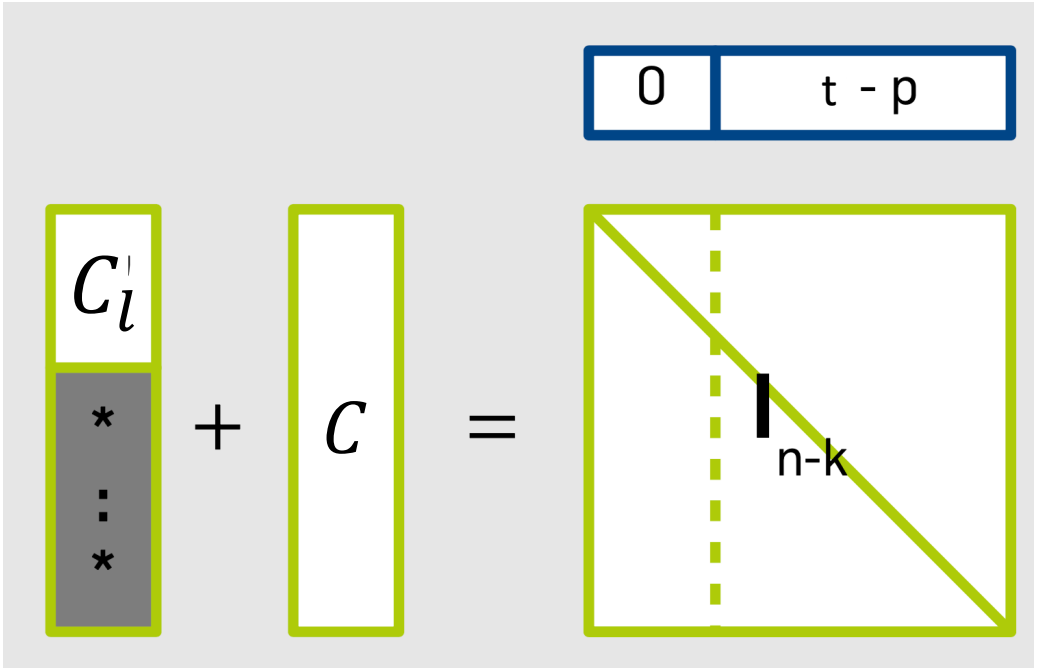
Stern's Information Set Decoding (Niederreiter 버전)

Step 1. $Q_l \cdot e_1$ = C_l인지 확인, C_l 은 C 의 l 크기 부분 (위로부터 l 만큼), l 또한 p와 마찬가지로 파라미터임

Step 2. Weight $(Q \cdot e_1 + C) = t - p$ 라면 e_2 를 결과 값에 맞춰 구성 → 신드롬 값, 무게 조건도 일치 (오른쪽 그림 참고)

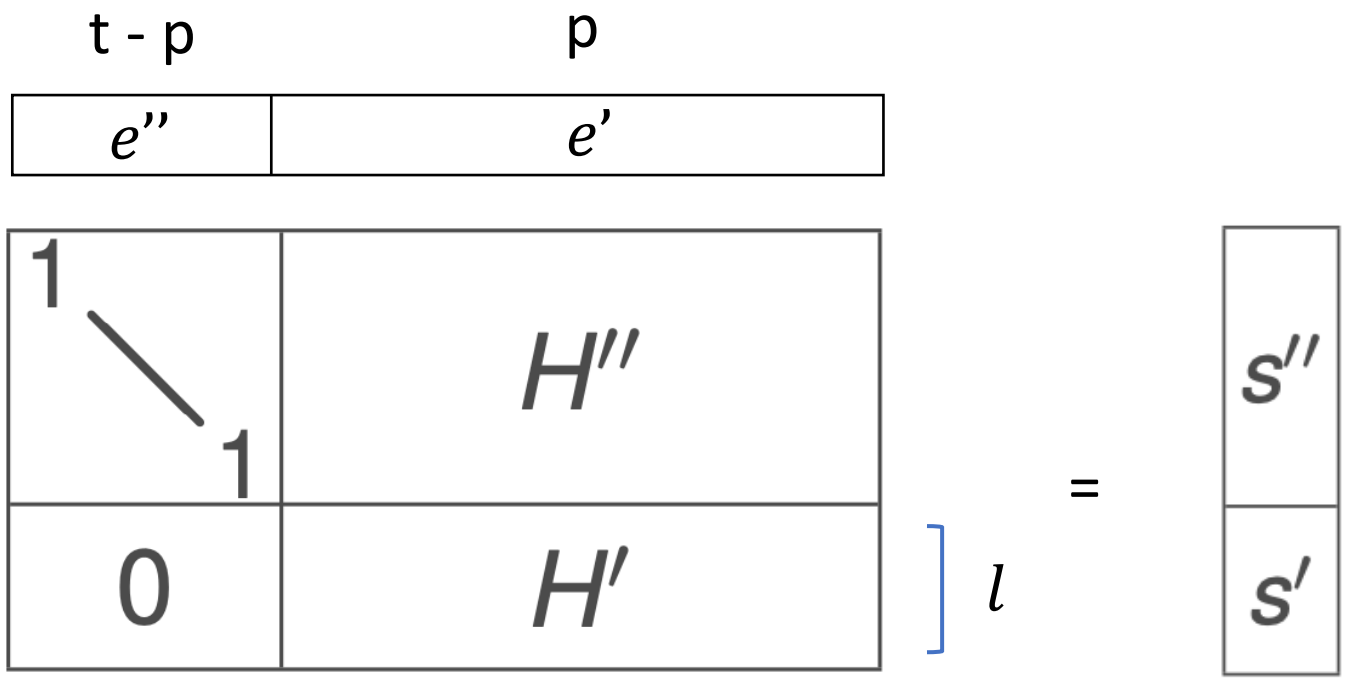
아니라면 Gaussian elimination 부터 다시 돌아가 반복





Dumer's Information Set Decoding (Niederreiter 버전)

- Gaussian elimination을 통해 아래와 같이 Systematic form을 구성, (이렇게 생기게도 만들수 있는진 모르겠는데, 이렇게 하고 bruteforce를 시작)
- Stern 의 알고리즘을 조금 개선
  - information set 에 해당하는 부분(e')을 반으로 나누지 않아도 되므로 더 효율적



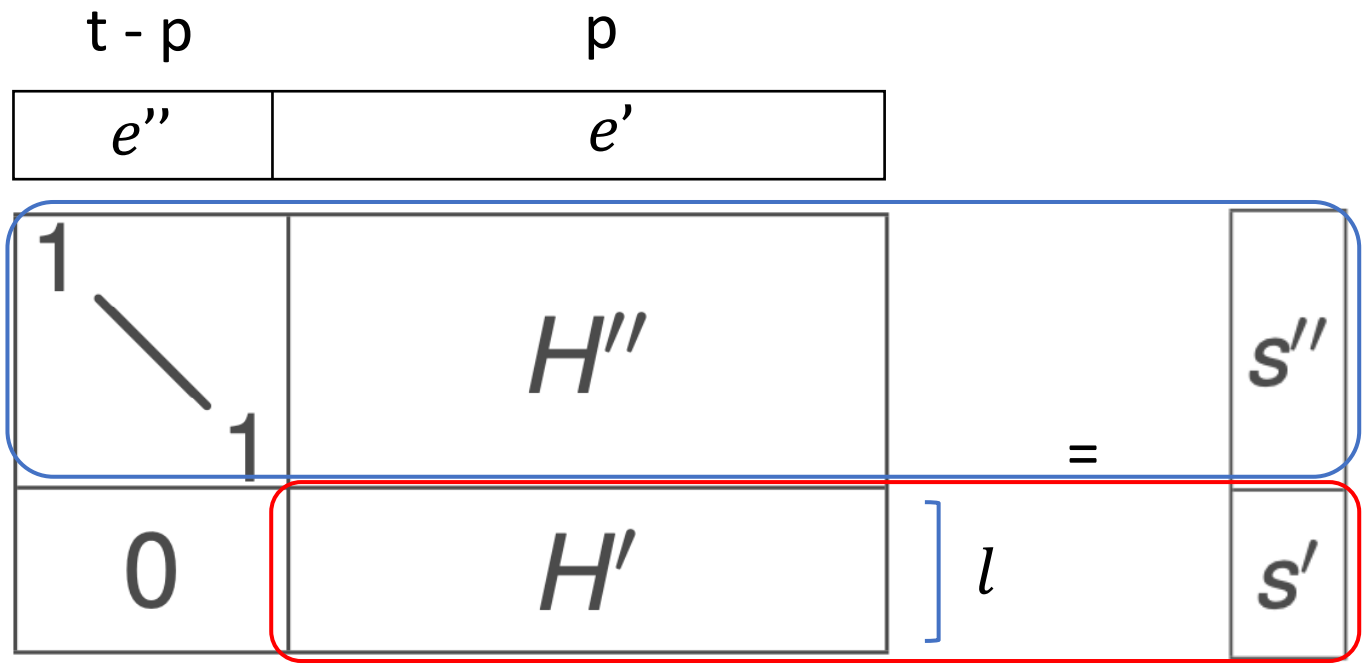
Dumer's Information Set Decoding (Niederreiter 버전)

Step 1. $e' \cdot H' = s'$ 을 만족하는 e'을 찾음

Step 2. Weight $(e' \cdot H'' + s'')$ 이 t - p 를 만족하는지 확인,

만족한다면 e'' 을 결과 값에 맞춰 구성 → 신드롬 값, 무게도 일치

찾지 못하면 처음부터 다시 반복



Quantum ISD

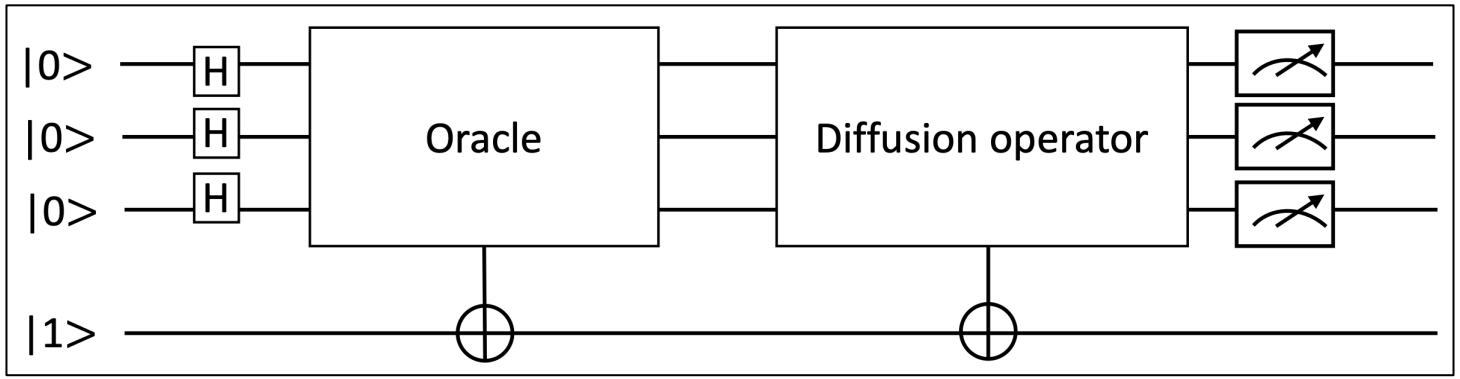
Quantum ISD

Challenge

- $C_0 = He^T$ 라는 신드롬 계산 식에서 C_0 와 H 가 주어진다 해도

  low – weight 벡터 e 를 찾아내기 매우 어려움→ Finding low-weight codeword problem

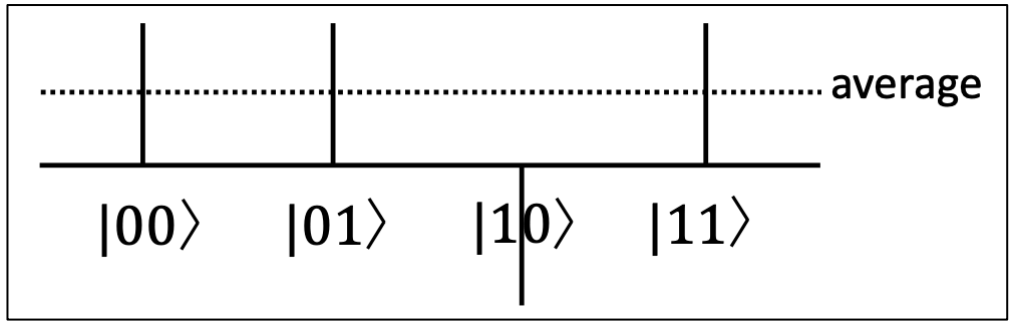
- Challenge에 대한 답은 Oracle에서 찾음, Diffusion operator에서는 해답 관측 확률 증폭



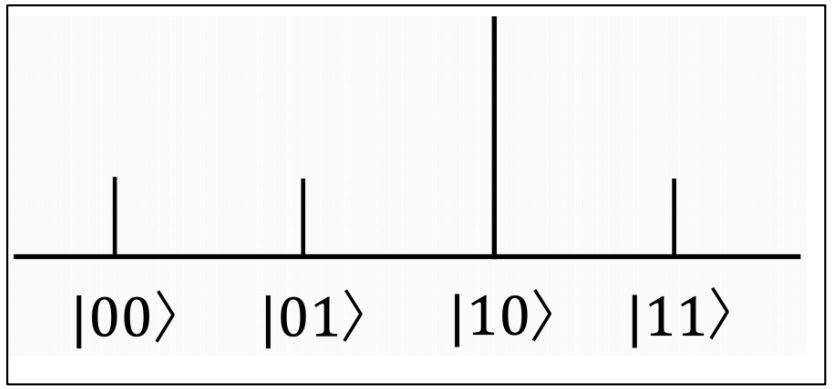
< Grover search >

Grover Search Algorithm

- Grover 알고리즘의 oracle에서는 정답인 큐비트 상태의 부호를 반전 (Z 게이트)



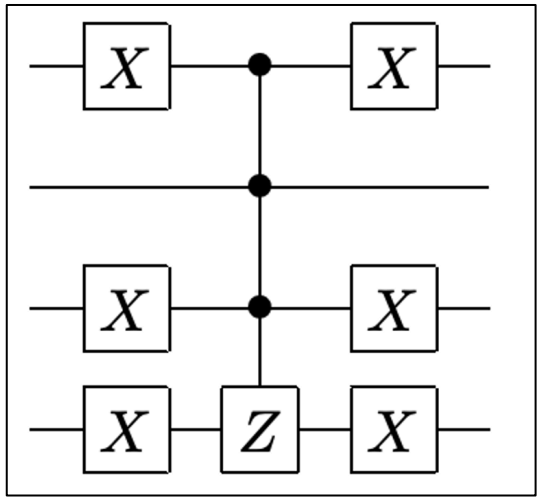
- Diffusion operator는 반전 된 큐비트 상태의 amplitude를 증폭
  - (평균 amplitude) – (해당 amplitude)



두 가지 과정을 반복한 뒤 관측

Grover Oracle Toy Example

- 간단한 Grover oracle
  - 4-Qubit Input일 때, 해답이 0 1 0 0 인 경우를 찾는 간단한 oracle



Step 1. 0 1 0 0 인 경우, X 게이트로 인해 1 1 1 1

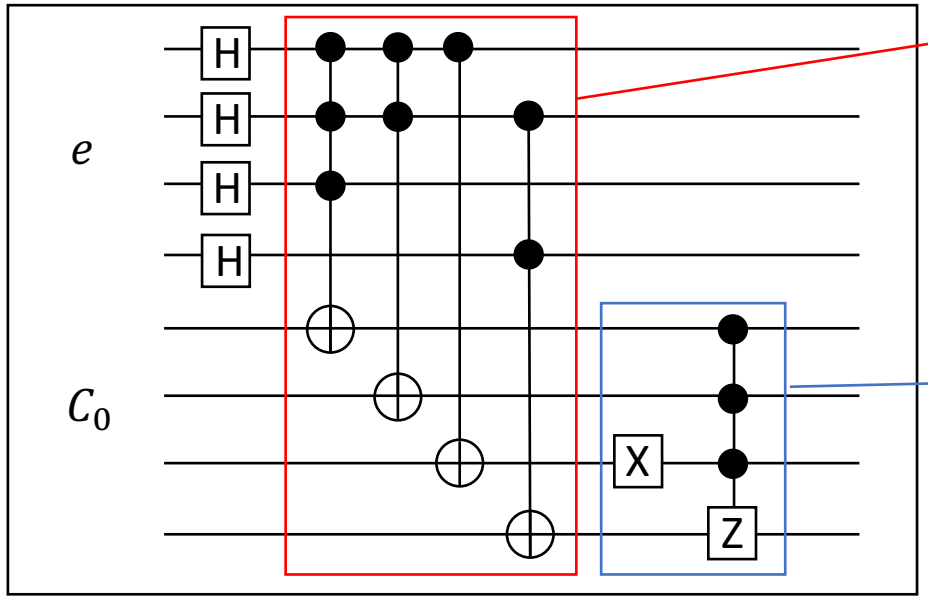
Step 2. 따라서 Input이 0 1 0 0 인 경우에만, Controlled - Z 게이트가 작동 → 부호 반전

Step 3. Reverse 연산을 수행하여 원래 상태로 되돌림 (1 1 1 1 → 0 1 0 0)

Quantum ISD (Brute Force Toy Example)

- $C_0 = He$ 를 만족하는 벡터 e를 찾는 oracle

$$H = \begin{matrix} 1\,1\,1\,0 \\ 1\,1\,0\,0 \\ 1\,0\,0\,0 \\ 0\,1\,0\,1 \end{matrix} \qquad C_0 = 1\,1\,0\,1 \qquad \text{Weight}(e) = 1$$

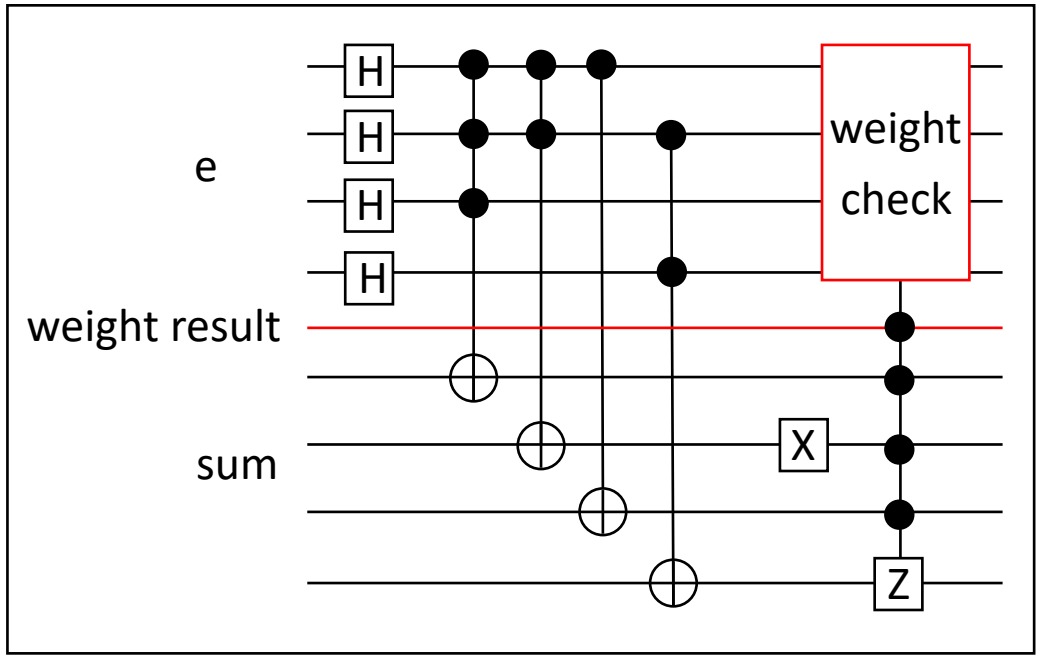


- Syndrome값 He 계산
  - e 는 중첩 상태
  - H 는 고정

- Syndrome 값이 1 1 0 1 인 경우, Controlled - Z 게이트 작동
- 벡터 e 의 Weight 확인 단계가 아직 남았음

Quantum ISD (Brute Force Toy Example)

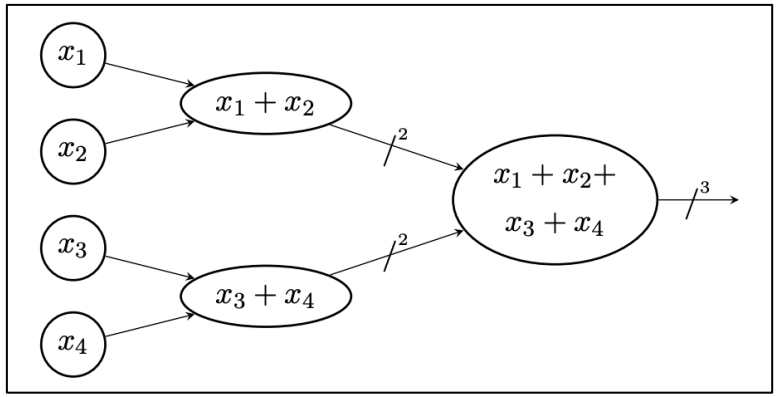
- $C_0 = He$ 에 대한 oracle 설계는 완료, 남은 건 Weight (e) = 1 인지 확인



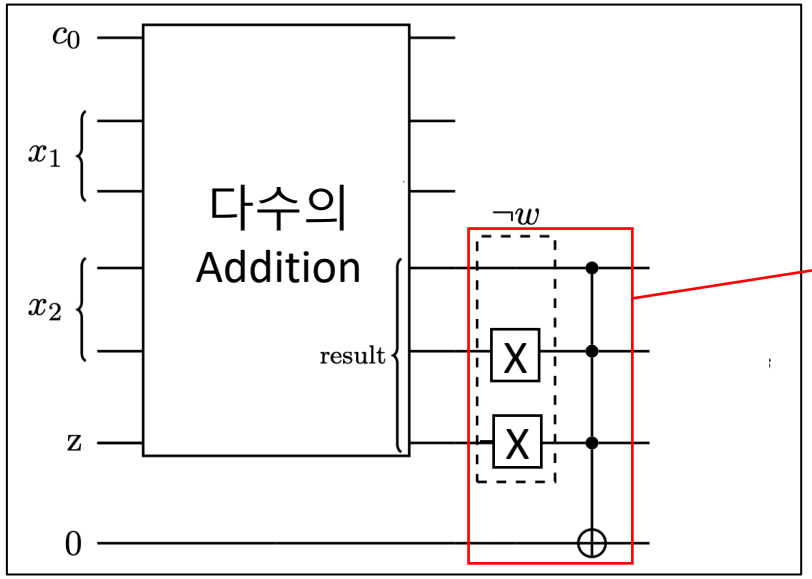
- Weight check에 가장 많은 양자 자원이 필요
  - 덧셈이 사용되며 Ripple-carry 회로 사용

Quantum ISD (Brute Force Toy Example)

- 4-Qubit의 Weight를 확인하는 법, (덧셈 시, 캐리 값을 저장하기 위한 추가 큐비트 필요)



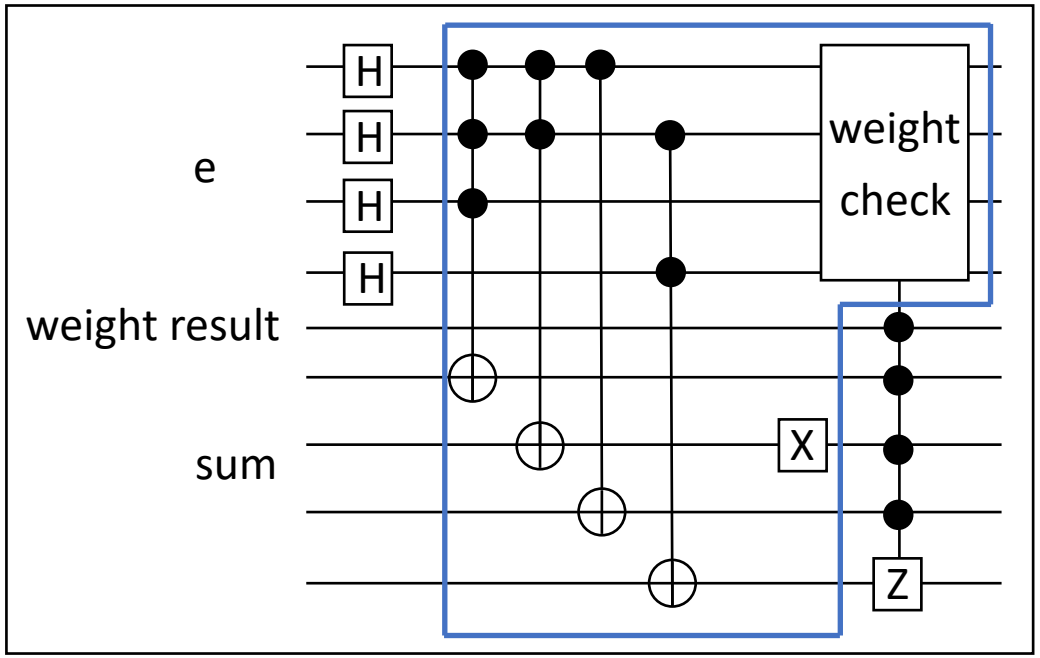
- 4-Qubit의 최대 Weight = 4 (1 0 0)



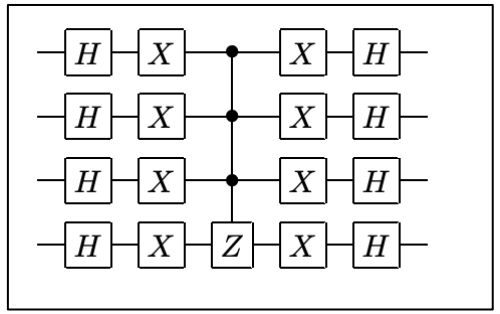
Weight 가 1인경우(0 0 1), weight_result = 1

Quantum ISD (Brute Force Toy Example)

- 최종 Quantum ISD circuit



+ Reverse (파란색 부분) +



Diffusion operator

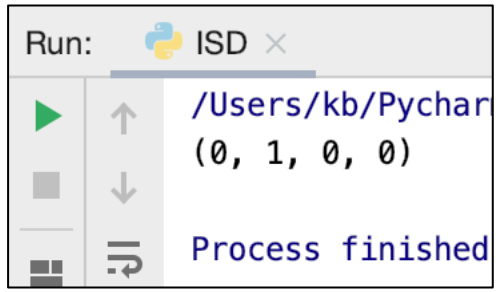
n번 반복하여 관측 확률 증가

Quantum ISD (Brute Force Toy Example)

- $C_0 = He$ 를 만족하는 벡터 e를 찾는 Quantum ISD

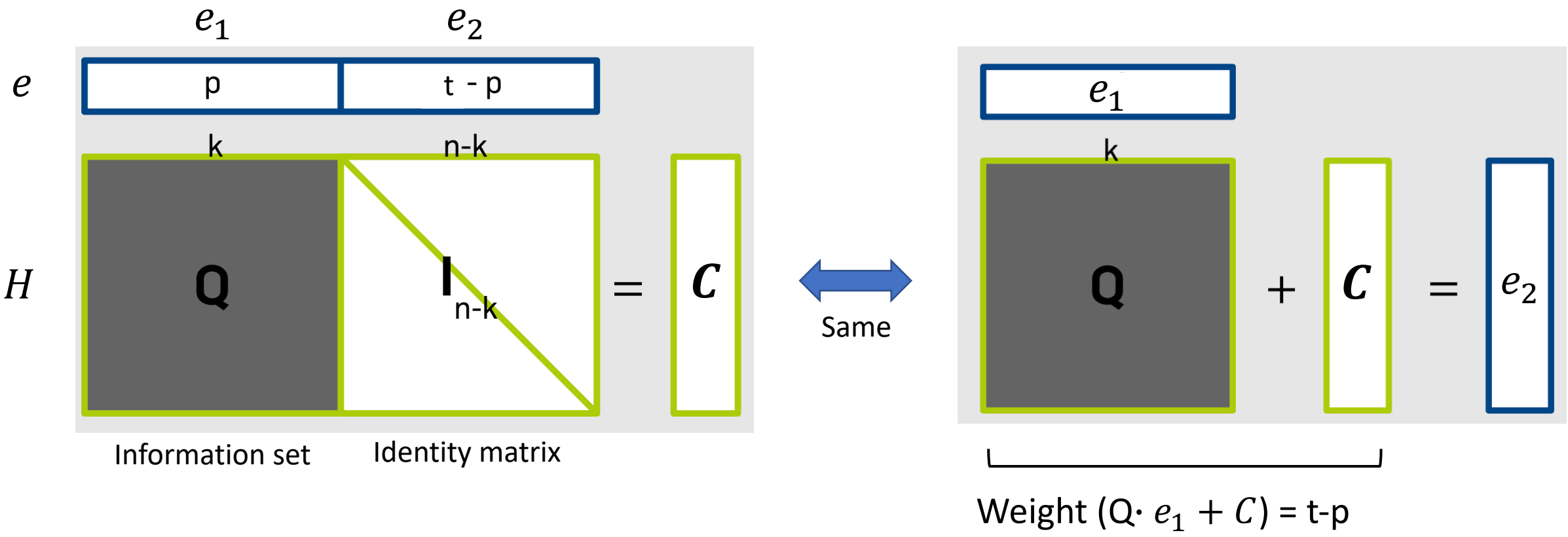
$$H = \begin{matrix} 1\,1\,1\,0 \\ 1\,1\,0\,0 \\ 1\,0\,0\,0 \\ 0\,1\,0\,1 \end{matrix} \qquad C_0 = 1\,1\,0\,1 \qquad \text{Weight}(e) = 1$$

- 3번 반복 뒤, 관측 결과



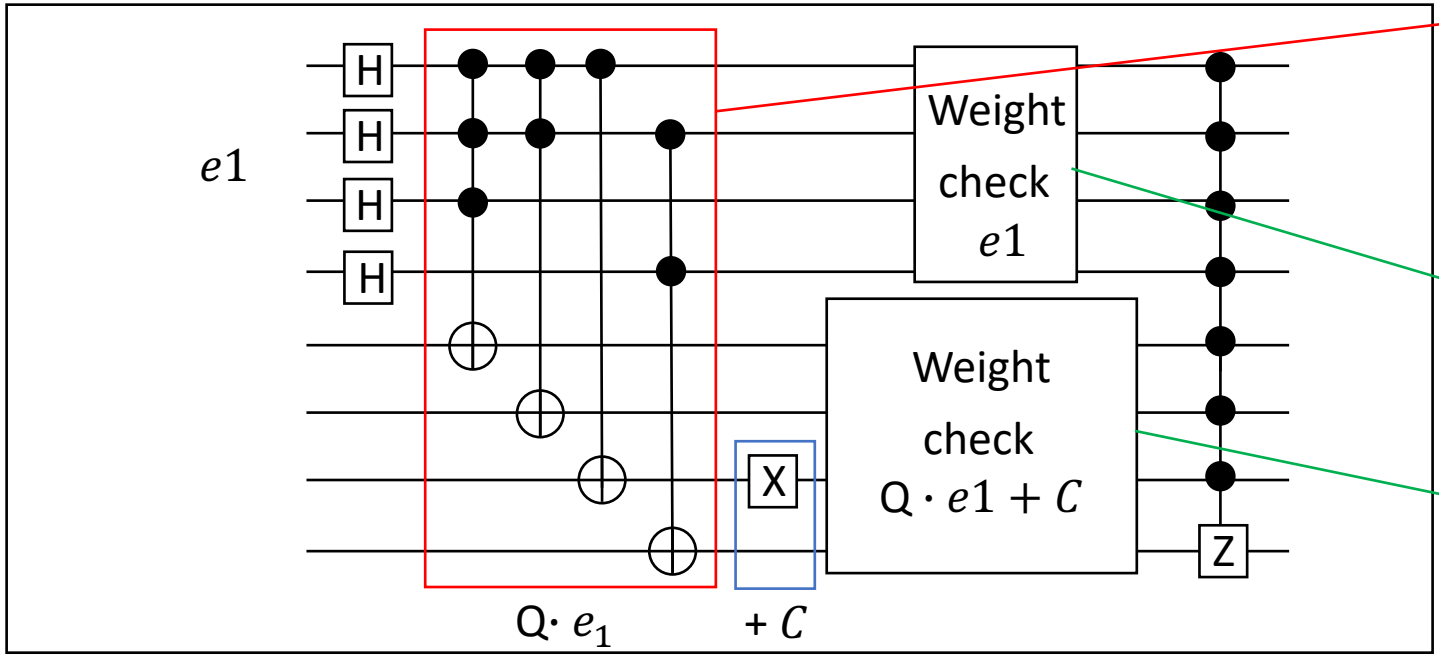
Quantum ISD (Lee-Brickell)

- 앞서 Quantum ISD 와는 다른 점, Weight 확인이 2번 수행 됨, → Q· $e_1 + C$ 그리고 e_1
- 회로에서 H에 대한 행렬을 구성할 때, Q 만 구성하면 됨, $\mathrm{I}_{\mathrm{n-k}}$ 는 제외 → 큐비트 감소



Quantum ISD (Lee-Brickell)

- Lee-Brickell Quantum ISD circuit (미니어처), oracle 부분
  - 앞선, Circuit과 크게 다르지 않음, (앞에 예제 회로로 설명)
  - Q· $e_1 + C$ 의 Weight를 확인해야 함
    - C 가 만약 0 0 1 0 이라 하면, 3번째 큐빗 라인에 X 게이트를 수행해주면 됨



- Q· e_1 계산
  - e_1는 중첩 상태
  - Q는 고정

weight 가 p 인지 확인

weight 가 t-p 인지 확인

진행 사항

- 실제 Goppa 코드에서 생성한 (8 x 16) 행렬 H에 대한 $C_0 = He$ 문제에 Quantum ISD 설계 (Lee-Brickell 버전으로)
  - 32 큐비트 사용 → 시뮬레이션 되지 않음 (약 5 큐빗 정도만 덜 썻어도..??), 더 작은 행렬로 시도해야 함
  - 사용 양자 자원 (반복횟수 n = 5)

Lee_brickell(CM_8x16)

```
/Users/kb/PycharmProjects/proj
Gate class counts:
    AllocateQubitGate : 32
    CCCCCCCCXGate : 10
    CCCCCCCCZGate : 5
    CCCCCCCZGate : 5
    CCXGate : 360
    CXGate : 1070
    DeallocateQubitGate : 32
    HGate : 88
    MeasureGate : 8
    XGate : 170
```

진행 사항

- Classic McEliece 가장 작은 파라미터 mceliece348864 를 타겟으로 Quantum ISD (Lee-Brickell) 구현
  - mceliece348864에서 사용하는 (768 X 3488) 의 행렬 H가 타겟
  - Weight를 확인하는 부분이 메인 → 대부분의 양자 자원이 여기서 사용
    - Weight 확인 모듈이 제대로 동작하는지 여부는 확인했지만(Classical simulator)
      - $e1$ → 2720-qubit 벡터 Weight 측정
      - $Q \cdot e1 + C$ → 768-qubit 벡터 Weight 측정
    - 실제 시뮬레이션은 불가, 양자 자원 측정 (반복 횟수 n=2로 확인) →