데이터통신 오류 검출

한기대 박승철교수

강의 내용

- 오류 검출 개요
- 패리티 검사(parity check)
- 순환 중복 검사(cyclic redundancy check)
- 검사합(checksum)

오류 검출 개요

- **단일 비트(single bit) 오류**
  - 데이터 전송 단위(예, 바이트, 프레임)에서 1비트만 값이 변경되는 오류
- **버스트(burst) 오류**
  - 데이터 전송 단위에서 2비트 이상의 값이 변경되는 오류
- **버스트 길이(length of burst)**
  - 첫번째 손상 비트와 마지막 손상 비트간의 구간 크기(비트 수)
- **중복(redundancy)**
  - 오류 검출을 위해 별도로 데이터에 추가되는 일련의 비트

오류 탐지 개요

- **송신자**
  1) 데이터 단위로부터 오류 검출 코드를 생성
  2) 데이터 단위에 오류 검출 코드를 추가한 데이터 전송 단위를 전송
- **수신자**
  1) 수신된 데이터 단위로부터 오류 검출 코드 생성
  2) 생성된 오류 검출 코드와 수신된 오류 검출 코드가 동일한지 확인
  3) 동일하지 않으면 오류 발생

패리티 검사(parity check)

개념

- 데이터 단위에 오류 검출을 위해 마지막에 1비트(패리티 비트)를 추가하여 데이터 전송 단위 생성
- 전송 단위의 1의 개수가 짝수(또는 홀수)가 되도록 패리티 비트 값 설정

예

- data unit : 0101010
  → transmission unit : 01010101
- $p = d_6 + d_5 + d_4 + d_3 + d_2 + d_1 + d_0 \bmod 2$

패리티 검사(parity check)

장점

- 간단함
- 단일 비트 오류 포함 홀수 개의 버스트(burst) 오류 검출

단점

- 짝수 개의 비트 오류 검출 불가

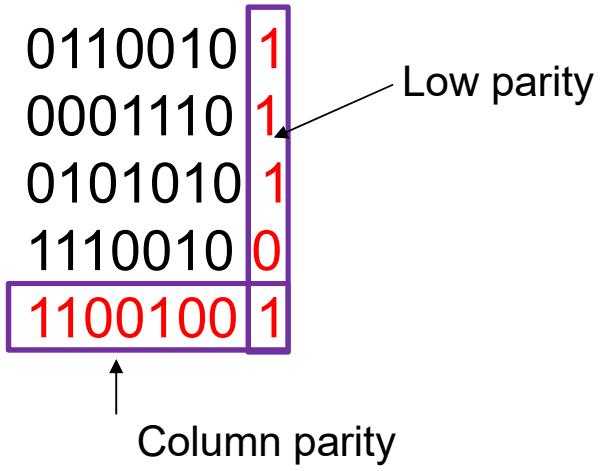
응용

- 저속의 시리얼 바이트 통신(serial byte communication)에 주로 사용

2차원 패리티 검사

개념

- Data : 0110010 0001110 0101010 1110010



- Transmission :
  01100101 00011101 01010101 11100100 11001001

2차원 패리티 검사

- 장점
  - 오류 검출 능력 향상
- 단점
  - 복잡성, 전송비트 증가
  - 서로 다른 2개의 데이터 단위에서 같은 위치의 2비트가 동시에 손상될 경우 검출 불가

순환 중복 검사(CRC)

❖ 송신자

1) n비트 데이터 단위를 왼쪽으로 k비트 이동시켜(shift) 임시 데이터 전송 단위 (n+k)비트 생성(오른쪽 k비트는 모두 0)
2) (n+k)비트 임시 데이터 전송 단위를 특수하게 제작된 (k+1)비트 생성자 비트열(generator bits)로 나누어 k비트 CRC 생성
3) 임시 데이터 전송 단위에 CRC를 더하여 최종 데이터 전송 단위 생성

❖ 수신자

1) 수신된 데이터 전송 단위를 (k+1)비트 생성자 비트열로 나누어 나머지 계산
2) 나머지가 0이 아니면 오류 발생

순환 중복 검사(CRC)

다항식(polynomial)

- n 비트열을 0과 1을 계수(coefficient)로 가지는 (n-1)차 다항식으로 표현
- (7 비트열) 1001101 = $x^6 + x^3 + x^2 + 1$(6차 다항식)

다항식 덧셈

- 동일 차수 항의 계수를 mod 2 덧셈
- $(x^6 + x^3 + x^2 + 1) + (x^6 + x^4 + x^3 + x + 1)$
  $= x^4 + x^2 + x$
  $= 0010110$

다항식 뺄셈

- mod 2 덧셈 역원을 더하는 연산
- 0과 1의 mod 2 덧셈 역원은 자기 자신 → 덧셈과 동일

순환 중복 검사(CRC)

❖ 다항식 곱셈

- 첫번째 다항식의 각 항을 두번째 다항식의 모든 항과 곱하여 구해진 모든 다항식을 더함
- $(x^6 + x^3 + x^2 + 1)(x^4 + x)$
  $= (x^{10} + x^7) + (x^7 + x^4) + (x^6 + x^3) + (x^4 + x)$
  $= x^{10} + x^6 + x^3 + x$
  $= 10001001010$

순환 중복 검사

❖ 다항식 나눗셈

1) 피제수(dividend)의 첫번째 항을 제수(divisor)의 첫번째 항으로 나누어 몫(quotient)의 첫번째 항 계산
2) 몫의 해당 항을 제수와 곱한 결과를 피제수에서 빼서 다음 피제수 계산
3) 다음 피제수의 차수가 제수의 차수보다 작아질 때까지 반복

❖ 다항식 나눗셈 예

$$
\begin{array}{r|l}
 & x^3 + x \\
\hline
x^3 + x + 1 & x^6 + \quad x^3 \\
 & x^6 + x^4 + x^3 \\
\hline
 & \quad x^4 \\
 & \quad x^3 + x^2 + x \\
\hline
 & \quad x^2 + x
\end{array}
$$

순환 중복 검사(CRC)

❖ 다항식으로 k비트 CRC 계산

1) n비트 데이터 단위 (n-1)차 다항식 d(x)에 x^k를 곱하여 임시 전송 단위 (n+k-1)차 다항식 t(x) 계산, $t(x) = x^k d(x)$
(n비트 데이터 단위를 k비트 좌측 shift 결과와 동일)
2) t(x)를 수학적으로 정교하게 설계된 k차 생성자 다항식 g(x)로 나누어 나머지 (k-1)차 다항식 r(x) 계산 (k비트 나머지)
3) T(x)에 r(x)를 더하여 최종 전송 데이터 단위 td(x)를 계산하여 대응되는 비트열 전송((n+k)비트 전송 단위)

$$\frac{x^k d(x)}{g(x)} = q(x) + \frac{r(x)}{g(x)}$$

$$td(x) = x^k d(x) + r(x)$$

순환 중복 검사(CRC)

❖ n비트열에서 k번째 비트 오류 발생(오른쪽부터 계산)

- k번째 비트가 변경(0 → 1 또는 1 → 0)
- (n-1)차 다항식에 (k-1)차 항의 계수가 1인 다항식 덧셈((0+1) mod 2 또는 (1+1) mod 2)

❖ 오류발생과 다항식 연산 예

송신 데이터 : 1 0 0 1 0 0 1 1 : $x^7 + x^4 + x + 1$

+

오류 : 0 1 0 1 0 0 0 0 : $x^6 + x^4$

=

수신 데이터 : 1 1 0 0 0 0 1 1 : $x^7 + x^6 + x + 1$

순환 중복 검사(CRC)

❖ 다항식으로 오류 확인

1) (n+k-1)차 데이터 다항식 rd(x) 수신
2) rd(x) = t(x) + e(x), e(x)는 오류 비트열
3) rd(x)를 생성자 다항식 g(x)로 나누어 나머지 계산
4) 나머지가 0이면(rd(x)=t(x)), 성공적으로 수신(e(x)≠0)

❖ 정상 수신 : rd(x) = t(x)

$$\frac{t(x)}{g(x)} = \frac{x^k d(x) + r(x)}{g(x)} = \frac{x^k d(x)}{g(x)} + \frac{r(x)}{g(x)}$$

$$= q(x) + \frac{r(x)}{g(x)} + \frac{r(x)}{g(x)}$$

$$= q(x), \text{ 나머지 } 0$$

순환 중복 검사(CRC)

❖ 다항식으로 오류 확인

1) (n+k-1)차 수신 데이터 다항식 rd(x) 수신
2) rd(x) = t(x) + e(x), e(x)는 오류 비트열
3) rd(x)를 생성자 다항식 g(x)로 나누어 나머지 계산
4) 나머지가 0이 아니면(rd(x)≠t(x)), 오류 발생(e(x)≠0)

$$\frac{t(x)+e(x)}{g(x)} = \frac{t(x)}{g(x)} + \frac{e(x)}{g(x)}$$

e(x)≠0이면서
나머지가 0이면
미검출 오류 발생

나머지 0

가능하면
나머지가 0이 되지 않도록
수학적으로 g(x) 설계

순환 중복 검사(CRC)

표준 다항식

- LAN : $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
- HDLC : $x^{16} + x^{12} + x^5 + 1$

CRC 장점

- 높은 오류 검출 능력(모든 단일 비트 오류, 모든 2비트 오류, 모든 홀수 비트 오류, k보다 작은 길이의 모든 버스트 오류, 높은 확률로 그 외의 버스트 오류 검출 등)
- 하드웨어 구현 용이

단점

- 복잡성

응용

- 주로 하드웨어 구현이 용이한 데이터링크 계층(LAN)에 적용

검사합(checksum)

❖송신자

1) 송신하는 메시지를 정해진 길이의 데이터 단위로 나눔
2) 모든 데이터 단위를 1의 보수(1's complement) 연산으로 더하여 합(sum)을 구함
3) 합의 1의 보수를 검사합(checksum)으로 생성하고, 검사합을 추가하여 메시지 전송

검사합(checksum)

❖ 수신자

1) 수신된 메시지를 정해진 길이의 데이터 단위로 나눔(검사합 포함)
2) 모든 데이터 단위를 1의 보수(1's complement) 연산으로 더하여 합(sum)을 구함
3) 합이 0이면 성공적인 수신, 아니면 오류 발생

검사합(checksum)

1의 보수 연산

- 이진수 1의 보수 : 각 비트 값 0과 1을 서로 바꾼 값(원래 비트 값과 더하여 1이 되는 값)
- +0 : 모든 비트가 0, -0 : 모든 비트가 1

m비트 이진수의 1의 보수 덧셈

- m비트 결과 값 생성
- 최종 비트 올림(carry)이 생길 경우 캐리 값을 m비트 결과에 더하여 최종 m비트 결과값 생성

검사합

❖ 1의 보수 덧셈(8비트 이진수)

```
  10011010
         +
  10111001
------------
 101010011
         +
end-round carry
         1
------------
  01010100
```

검사합

❖ 8비트 검사합 계산

- 데이터 : 00111010 10011000 00111010

```
  00111010
         +
  10011000
------------
  11010010  ←  중간합
         +
  00111010
------------
  00001100
         +
         1  ←  carry
------------
  00001101  ──→  검사합 : 11110010
  1의 보수
```

검사합

❖ 오류 확인(데이터 : 00111010 10011000 00111010)

- 수신 데이터 : 00111010 10011000 00111010 11110010 ← 검사합

```
  00111010                    00001101
         +                           +
  10011000                    11110010
------------                ------------
  11010010 ← 중간합          11111111 ← -0
         +
  00111010
------------
  00001100
         +
         1 ← carry
------------
  00001101 ← 중간합
```

검사합

장점

- 높은 오류 검출 능력
- 단순함
- 소프트웨어 구현에 적합

단점

- 각 데이터 단위에서 발생하는 오류의 합이 0이 되는 오류 검출 불가
- 체섬을 변경하지 않는 오류 검출 불가

검사합

❖ 오류 검출 불가(데이터 : **00111010 10011000 00111010**)

- 수신 데이터 : 00111000 10011010 00111010 11110010

```
  00111000                    00001101
         +                           +
  10011010                    11110010
------------                ------------
  11010010 ← 중간합          11111111 ← -0
         +
  00111010
------------
  00001100
         +
         1 ← carry
------------
  00001101 ← 중간합
```

검사합

❖ 오류 검출 불가

- 수신 데이터 : 00111000 10011001 00111011 11110010

```
  00111000                    00001101
         +                           +
  10011001                    11110010
------------                ------------
  11010001 ← 중간합          11111111 ← -0
         +
  00111011
------------
  00001100
         +
         1 ← carry
------------
  00001101 ← 중간합 ──→ 검사합 : 11110010(변경 없음)
```

요약

- 패리티 검사(parity check)
  - 시리얼 바이트 통신
- 순환 중복 검사(cyclic redundancy check)
  - 데이터링크 계층(LAN) 통신
  - 하드웨어 구현
- 검사합(checksum)
  - 상위 계층(IP, TCP)
  - 소프트웨어 구현