Módulo 07
Detección y Corrección de Errores (Pt. 1)

Organización de Computadoras
Depto. Cs. e Ing. de la Comp.
Universidad Nacional del Sur

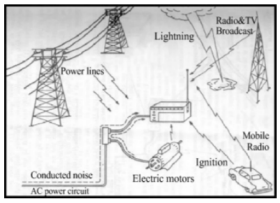
Copyright

Contenidos

- Concepto de error
- Mínima distancia de un código
- Mecanismos de detección de errores
- Paridad aplicada en los códigos **VRC** y **LRC**
- Generación y verificación de código **CRC**
- Mecanismos de corrección de errores
- Códigos correctores simples
- Hamming mínima distancia 3 y 4

Concepto de error

- Toda vez que una pieza de información es transmitida existe la posibilidad de que lo enviado no coincida con lo recibido
- El origen de estos errores suele depender del medio de transmisión utilizado:
  - Ruido y/o interferencia
  - Atenuación de la señal
  - Problemas de sincronización
  - Propagación multicamino



Concepto de error

- ¿Qué actividades se verán afectadas?
  - La comunicación de información sobre grandes distancias (por caso, Internet)
  - La comunicación de información sobre cortas distancias (por caso, comunicación entre el **CPU** y la memoria o los dispositivos)
  - El almacenamiento de información en dispositivos no confiables (por caso, disquettes)
  - El almacenamiento de información en dispositivos lábiles (por caso, **CD**, **DVD** y **BR**)

Posibles contramedidas

- Nosotros como humanos, ¿qué recaudos tomamos para contrarrestar los errores?
  - Al hablar por celular cuando entramos a un túnel o pasamos abajo de líneas de alta tensión, ¿qué hacemos si justo se corto lo que nos decían?
  - En un boliche con la música muy alta, al tratar de conversar con nuestra pareja de baile, ¿de qué manera hablamos?
  - Cuando un grupo de soldados solicita un ataque aéreo sobre posiciones enemigas en las cercanías, ¿de qué manera se transmiten las coordenadas?

Posibles contramedidas

- Para contrarrestar el efecto de los errores en la transmisión existen tres alternativas:
  - Aceptar que se produzcan: en ciertas circunstancias es posible que la información transmitida siga siendo relevante aún ante la presencia de un error (por caso, al transmitir en vivo un video)
  - Impedir que se produzcan: tomar todos los recaudos necesarios para asegurar que nunca se produzca un error. Esta alternativa suele tener un costo prohibitivo
  - Contemplar que se produzcan: incorporar mecanismo que permitan atemperar el impacto de estos errores

Posibles contramedidas

- En caso de adoptar la última alternativa la idea es incorporar un mecanismo que se encargue cancelar el impacto de los eventuales errores de transmisión
- Estos mecanismos, cuyo propósito es lidiar con los errores, se clasifican en dos categorías:
  - Códigos detectores de error
  - Códigos correctores de error

Detección vs. corrección

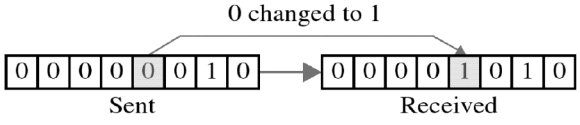
- Detección de errores:
  - Un código detector incorpora información adicional junto con los datos transmitidos de manera que se pueda determinar si se produjo o no un error durante la transmisión
- Corrección de errores:
  - Un código corrector incorpora más información que uno detector, ya que la idea es, además de detectar si se produjo un error, tener la certeza de en qué lugar se produjo a fin de poder corregirlo sin requerir la retransmisión del dato en cuestión

Tipos de error

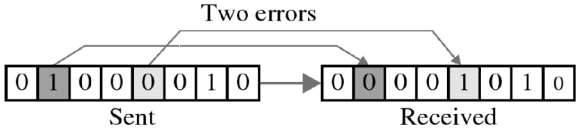
- Al trabajar con información binaria, el error se trata simplemente de un intercambio (toggle) del valor de uno o más bits
- Los errores se clasifican de la siguiente manera:
  - Error a nivel de bit: el error afecta a **n** bits del dato transmitido. En función de **n**, hablamos de error simple, error doble, etc.
  - Error en ráfaga: el error afecta a **m** bits consecutivos, estando el primero y el último en error (nótese que los bits entre medio pueden o no estar en error)

Error a nivel de bit

- Error simple:

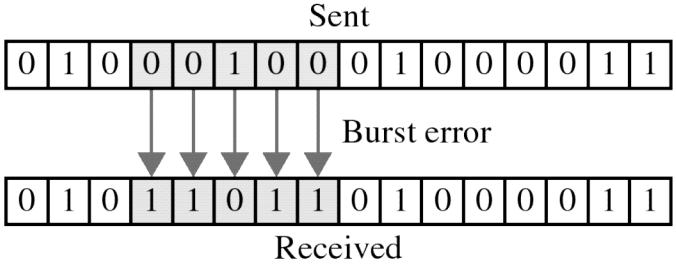


- Error doble:



Error en ráfaga

- Error en ráfaga (el ejemplo asume que la ráfaga alteró a la totalidad de los bits alcanzados):



Definiciones

- Denominaremos código a un conjunto finito de patrones de bits de longitud fija
- Sean **p** y **q** dos patrones de bits, en este contexto denominaremos distancia entre **p** y **q**, notado **d(p,q)**, a la cantidad de posiciones de bits en los cuales los patrones **p** y **q** difieren
- Sea **p** un patrón de bits, denominaremos peso de la palabra **p**, notado **w(p)**, al número de bits puestos a **1** dentro de esa palabra

Ejemplo

- Sean **p = 01001010** y **q = 10001011**
- En este contexto, calcular los siguientes pesos:
  - **w(p) = ¿?**
  - **w(q) = ¿?**
- Finalmente, determinar qué distancia existe entre los patrones **p** y **q**:
  - **d(p,q) = ¿?**

Mínima distancia

- Sea **C** un código compuesto de patrones de bits de longitud fija. Denominaremos mínima distancia del código **C** a la menor distancia que se verifique entre cualesquiera dos patrones de bits no idénticos pertenecientes a **C**
  - Tener en cuenta que para determinar la mínima distancia de un cierto código es necesario calcular un gran número de distancias, ya que se debe analizar la distancia entre cada patrón con respecto a los restantes patrones del código

Ejemplo

- Supongamos que un cierto código se compone de los siguientes patrones de bits los que a su vez codifican a cuatro caracteres:

A: 0 0 0 0 0
B: 1 1 1 0 0
C: 0 0 1 1 1
D: 1 1 0 1 1

- ¿Cuál es la mínima distancia de este código?

Ejemplo

- Supongamos que el emisor envía el carácter **D** usando el código recién considerado (es decir, envía el patrón **11011**), pero el receptor en cambio recibe otro patrón, el **11000**
  - ¿Cuántos errores a nivel de bit se produjeron?
  - El patrón **11000** no será confundido con ningún otro patrón válido, es decir, este código permitió detectar el error
  - ¿Hasta cuántos bits en error detectará con certeza un código cuya mínima distancia sea **m**?

Análisis

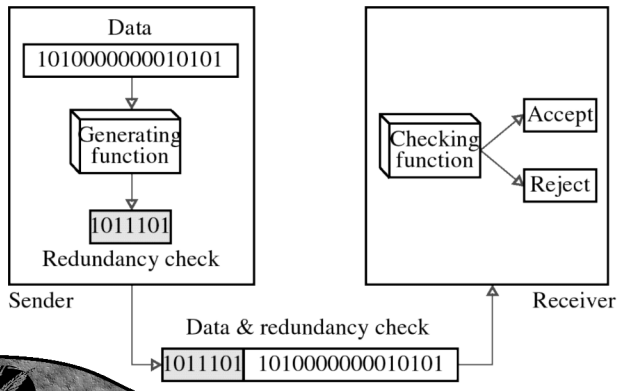
- En general, podemos concluir lo siguiente:
  - Toda vez que se produzca una cantidad menor de errores que la mínima distancia del código adoptado, el error será siempre detectado
  - Pero, si se produce una cantidad igual o mayor de errores, a veces se detectará pero a veces no
  - Por caso, usando el código antes visto, tres errores pueden convertir una ***A*** en una **B**, o bien pueden convertir la ***A*** en un patrón inválido:

A: 0 0 0 0 0 → **B: 1 1 1 0 0**
A: 0 0 0 0 0 → **1 1 0 0 1**

Detección de errores

- Para detectar errores se debe incorporar alguna forma de redundancia al dato transmitido, a fin de que se pueda determinar si la información se recibió correctamente
- Se han ensayado distintas alternativas:
  - **VRC**: verificación de redundancia vertical
  - **LRC**: verificación de redundancia longitudinal
  - **CRC**: verificación de redundancia cíclica
  - Checksum: cálculo de una suma de comprobación

Detección de errores



Bit de paridad

- Una de las formas más elementales de detección de errores consiste en incorporar un bit de paridad al dato transmitido
  - El bit de paridad es un bit que toma el valor **0** ó **1** con el objeto de satisfacer una cierta restricción sobre la paridad de un determinado patrón de bits
- Es posible implementar un esquema de paridad par o impar, a saber:
  - Un patrón de bits se dice tener paridad par o impar si, y sólo si, tiene un peso par o impar respectivamente

Bit de paridad

- A partir de la paridad par o impar es posible definir sendos códigos de detección de errores:
  - Código de paridad par: a partir de un cierto dato se incorpora un bit adicional el cual adoptará el valor necesario para que el patrón dato + bit de paridad tenga paridad par
  - Código de paridad impar: definido de manera análoga, con la salvedad de que el patrón dato + bit de paridad tenga ahora paridad impar
- Por convención, el bit de paridad lo indicaremos a la izquierda del patrón de bits original

Ejemplo

- Supongamos que nuestro dato original es **p = 0111001**. El patrón de bits que resulta al incorporar un bit de paridad dependerá de la paridad que estemos usando:
  - **0 0111001**, al usar paridad par (pues **w(p)** es par)
  - **1 0111001**, al usar paridad impar (pues **w(p)** es par)
- Al recibir un cierto patrón de bits, se puede verificar si respeta la paridad adoptada:
  - Por caso, usando paridad par se acepta el patrón de bits **0 00000000**, pero se rechaza **1 11111111**

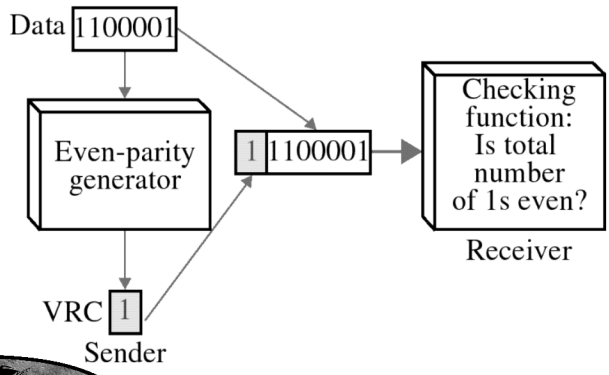
Análisis

- Paridad es un código con mínima distancia **2**, ¿qué capacidad de detección manifestará?
- Esta capacidad de detección, ¿depende del esquema de paridad elegido?
- ¿Será capaz de detectar errores en ráfaga?
- Este código no está en condiciones de corregir los errores detectados... ¿por qué razón?
- Resulta muy fácil de implementar en hardware... ¿de qué manera?

Código VRC

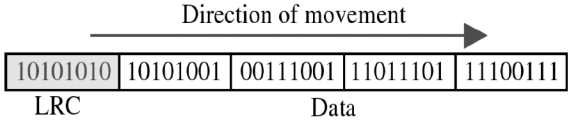
- El código **VRC** (Vertical Redundancy Check) añade un bit de paridad a cada unidad de datos de manera que la cantidad de bits en **1** sea par o impar (en función de la paridad adoptada)
  - También se lo conoce como código **TRC** (Transverse Redundancy Check)
  - La idea es que se use un bit de paridad para cada unidad de datos, en vez de usar un único bit para la totalidad del mensaje
  - Al igual que paridad, detecta la totalidad de los errores simples a nivel de unidad de datos

Código VRC



Código LRC

- El código **LRC** (Longitudinal Redundancy Check) aplica la misma idea que el código **VRC**, pero computa la paridad en sentido longitudinal:



  - La idea es que un bloque de bits se divida en filas, para luego añadir una fila de bits de redundancia
  - La intención es permitir la detección de errores en ráfaga

Ejemplo

- Supongamos que se desea transmitir usando el código **LRC**, paridad par, al mensaje ***HOLA*** el cual está codificado en **ASCII** extendido:

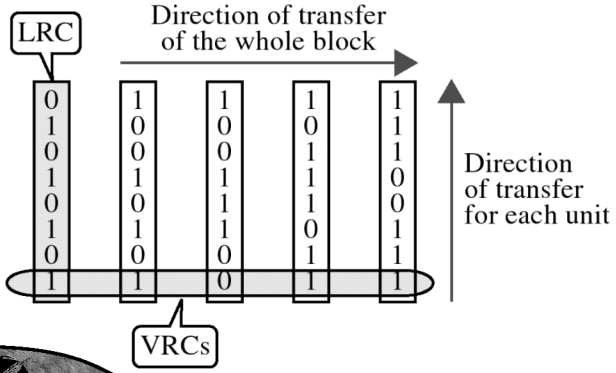
```
H: 0 1 0 0 1 0 0 0
O: 0 1 0 0 1 1 1 1
L: 0 1 0 0 1 1 0 0
A: 0 1 0 0 0 0 0 1
   ---------------
   0 0 0 0 1 0 1 0
```

- ¿En qué orden se deben transmitir estos bits?

Análisis

- El código **LRC** resulta más complicado de analizar producto de la manera en la que ordenan los bits del mensaje a ser transmitido
- En relación a los errores a nivel de bit, este código sigue detectando correctamente los errores simples
- El principal beneficio es que ahora estamos en condiciones de detectar errores en ráfaga
  - ¿Hasta qué tamaño de ráfaga podremos detectar correctamente?

Combinando VRC y LRC



¿Preguntas?