Számítógépes Hálózatok GY – 5.hét

Laki Sándor
ELTE-Ericsson Kommunikációs Hálózatok Laboratórium
ELTE IK - Információs Rendszerek Tanszék
lakis@elte.hu
http://lakis.web.elte.hu

Teszt – 10 kérdés 10 perc

canvas.elte.hu

Kód:

Kódolások, Hamming-távolság…

1. Feladat

Egyetlen paritásbit által nyújtottnál nagyobb biztonságot akarunk elérni, így olyan hibaészlelő sémát alkalmazunk, amelyben két paritásbit van: az egyik a páros, a másik a páratlan bitek ellenőrzésére.

- Mekkora e kód Hamming-távolsága?
- Mennyi egyszerű és milyen hosszú burst-ös hibát képes kezelni?

2. Feladat

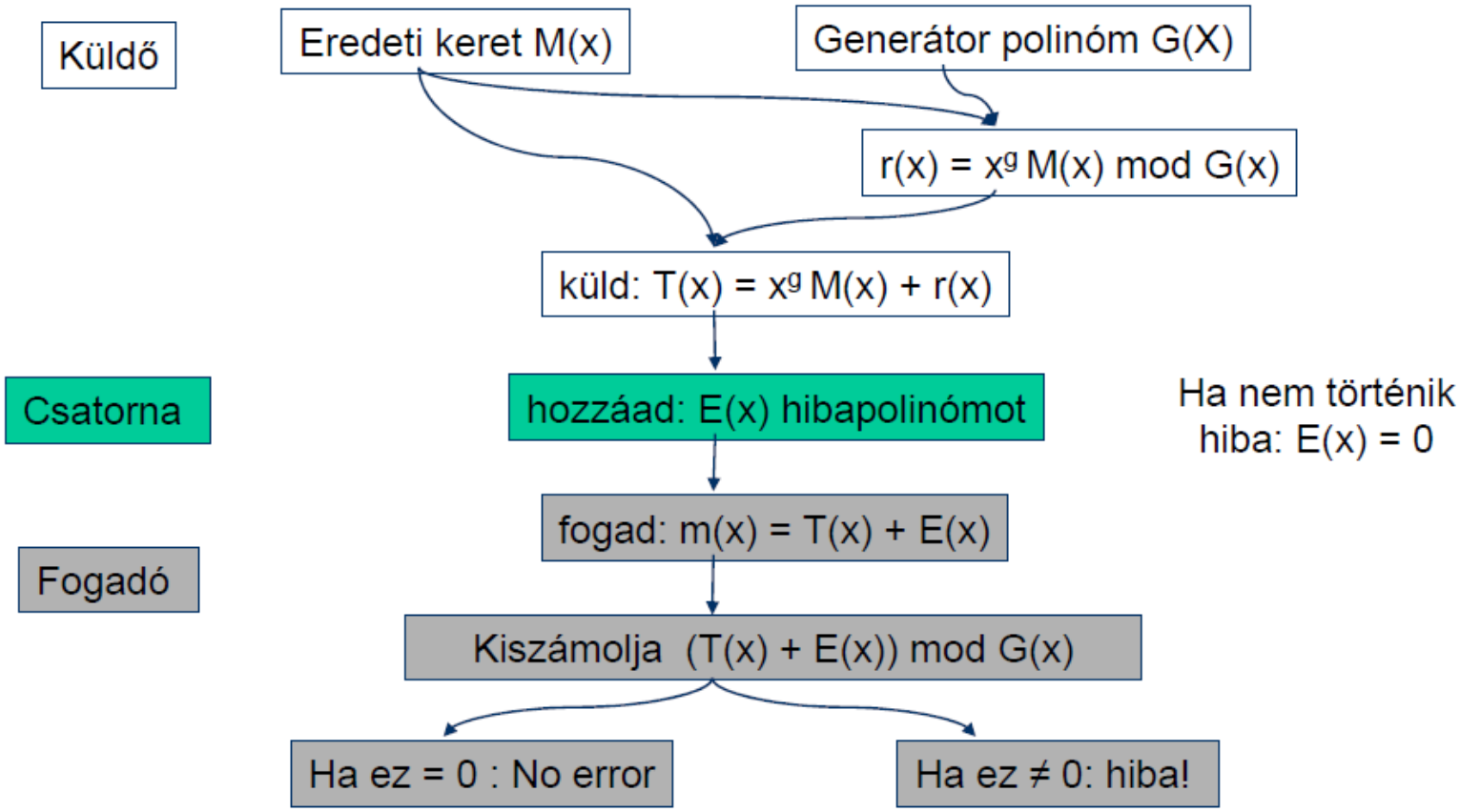
2. feladat: Tekintsük a következő paritás-technikát:

- Tekintsük az n küldendő adatbitet mint egy $k \times \ell$ bit-mátrix.
- Minden oszlophoz számoljon ki egy paritás-bitet (pl. odd parity) és egészítse ki a mátrixot egy új sorral, mely ezeket a paritás-biteket tartalmazza.
- Küldje el az adatokat soronként.

1. Adjon egy példát $k = 3$, $\ell = 4$ esetén.
2. Hogy viselkedik ez a módszer egyszerű bit-hibák és löketszerű (burst) bit-hibák esetén? Milyen hosszú lehet egy bitsorozat, melynek minden bitje hibás (burst), hogy a hibát felismerjük?
3. Egészítse ki a mátrixot egy új oszloppal is, amely minden sorhoz paritás-bitet tartalmaz (két dimenziós paritás technika). Hogyan használható ez a módszer 1-bit-hiba javítására? Mi a helyzet több bithibával és burst-hibákkal.

CRC

CRC áttekintés



3. Feladat

Számolja ki a **0101.1011.1101** inputhoz a 4-bit-CRC kontrollösszeget, ha a generátor polinom $\mathbf{x^4 + x^2 + 1}$!

Adjon egy olyan inputot, amely 1-gyel kezdődik és ugyanezt a kontrollösszeget eredményezi!

4.Feladat

Történt-e hiba az átvitel során, ha a vevő a következő üzenetet kapja: **1011 0001 1101 1111**

A generátor polinom $x^4 + x + 1$

CRC/MD5 Pythonban

32 bites CRC

```
def create_table(poly):
    a = []
    for i in range(256):
        k = i
        for j in range(8):
            if k & 1:
                k ^= poly
            k >>= 1
        a.append(k)
    return a

def crc32(buf,crc_table):
    crc = 0xffffffff
    buf = bytearray(buf)
    for k in buf:
        crc = (crc >> 8) ^ crc_table[(crc & 0xff) ^ k]
    return crc ^ 0xffffffff


gen_poly = 0x1db710640
crctable = create_table(gen_poly)

test_string = 'FeketeRetek a rettenetes'

print( "Input: {}".format(test_string) )
print( "crc32: {}".format( hex(crc32(test_string, crctable) % (1<<32) )))
```

binascii, zlib

```
import binascii
import zlib

test_string = 'FeketeRetek a rettenetes'

print( "Input: {}".format(test_string) )
print( "binascii.crc32: {}".format(hex(binascii.crc32(bytearray(test_string)) % (1<<32) )) )
print( "zlib.crc32: {}".format(hex(zlib.crc32(test_string) % (1 << 32) )) )
```

md5 - hashlib

```
import hashlib

m = hashlib.md5()
m.update("Nobody inspects")
m.update(" the spammish repetition")
print( "Digest: {}".format(m.digest()))
print( "Digest in hex: 0x{}".format(m.hexdigest()))
print( "Digest size: {}".format(m.digest_size))
```

Házi feladat

Házi feladat
netcopy alkalmazás 4 pont

Készíts egy netcopy kliens/szerver alkalmazást, mely egy fájl átvitelét és az átvitt adat ellenőrzését teszi lehetővé CRC vagy MD5 ellenőrzőösszeg segítségével!

Részletek a hét folyamán…

Beadás: BE-AD rendszeren keresztül

Határidő: 2019.03.31. 23:59

Vége
Köszönöm a figyelmet!