DEPARTMENT OF
NETWORKED SYSTEMS
AND SERVICES

Veszteségmentes forráskódolás

VIHIBB01 – Coding and IT Security, 2020

István Vajda

CrySyS Lab, BME
vajda@crysys.hu

www.crysys.hu

MŰEGYETEM 1782

Tartalom

- A forráskódolás célja
- Alkalmazási területek
- Veszteségmentes kódolás
  - Huffman-kódolás
  - LZW algoritmus

A forráskódolás célja

Cél az információ hatékony továbbítása vagy tárolása:

- több információ átvitele adott csatornán, több információ tárolása adott tárhelyen
- sávszélesség, tárkapacitás igény csökkentés

A forráskódolás két ága:

- veszteségmentes kódolás
- veszteséges kódolás

Alkalmazási területek (veszteségmentes kódolás)

Huffman kód (1952):

- GZIP
- PKZIP (winzip etc.)
- BZIP2
- JPEG rész-algoritmusa

LZW algoritmus (1984):

- UNIX *compress* algoritmusa
- GIF (Graphics Interchange Format) kódolásban
- PDF -ben

Alkalmazási területek (veszteséges kódolás)

Alkalmazások, amelyek nem léteznének forráskódolás nélkül:

- Digitális televízió (DVB-T)
- Internet video streaming (YouTube)

Alkalmazások, amelyek gazdaságossá váltak forráskódolás alkalmazásával:

- Distribution of digital images
- High definition television (HDTV) over IPTV

Számtalan alkalmazás használ forráskódolást:

- Software distributed in compressed form
- Audio data compression (MP3, AAC)
- Mobile audio players (IPod,...) and mobile phones
- Audio download (ITunes) and streaming services (Internet radio)
- Digital images are typically compressed (JPEG)
- Pictures on web sites are compressed
- Digital video data compression (MPEG-2, H.264/AVC)
- Output of video cameras, optical discs (DVD)
- Video streaming (Youtube, Internet TV)

- **Az Interneten a továbbított adatbitek kb. 70% -a forráskódolt video.**

Számpélda veszteséges forráskódolásra(*)

- File tömörítés (text file, office dokumentum, program kód, ...)
- Tipikus példa: 80 MByte tömörítése 20 Mbyte-ra (**25%-ra**)
- Audio tömörítés
- Sztereo: mintavételi frekvencia: 44,1 kHz
- 16 bit/minta
- =) Nyers adat sebesség: 44,1x16x2 = 1,41 Mbit/s
- =) Tipikus adatsebesség tömörítéssel: 64 kbit/s (**4.5%-ra**)
- Kép tömörítés
- Kép méret: 3000x2000 minta (6 MegaPixel)
- 3 színkomponens (vörös, zöld, kék) és 1 byte (8 bit) / minta
- =) Nyers file méret: 3000x2000x3 = 18 MByte
- =) Tipikus tömörített méret: 1 MByte (**5.6%-ra**)
- Video tömörítés
- Képméret 1920x1080 pixel és keret sebesség 50 Hz
- 8 bit/minta
- 3 színkomponens (vörös, zöld, kék)
- =) Nyers adat sebesség: 1920x1080x8x50x3 = 2,49 Gbit/s
- =) Tipikus adatsebesség tömörítéssel: 12 Mbit/s (**0.5%-ra**)

(*) heiko.schwarz@hhi.fraunhofer.de

Huffman-kód: karakter-gyakoriság tábla

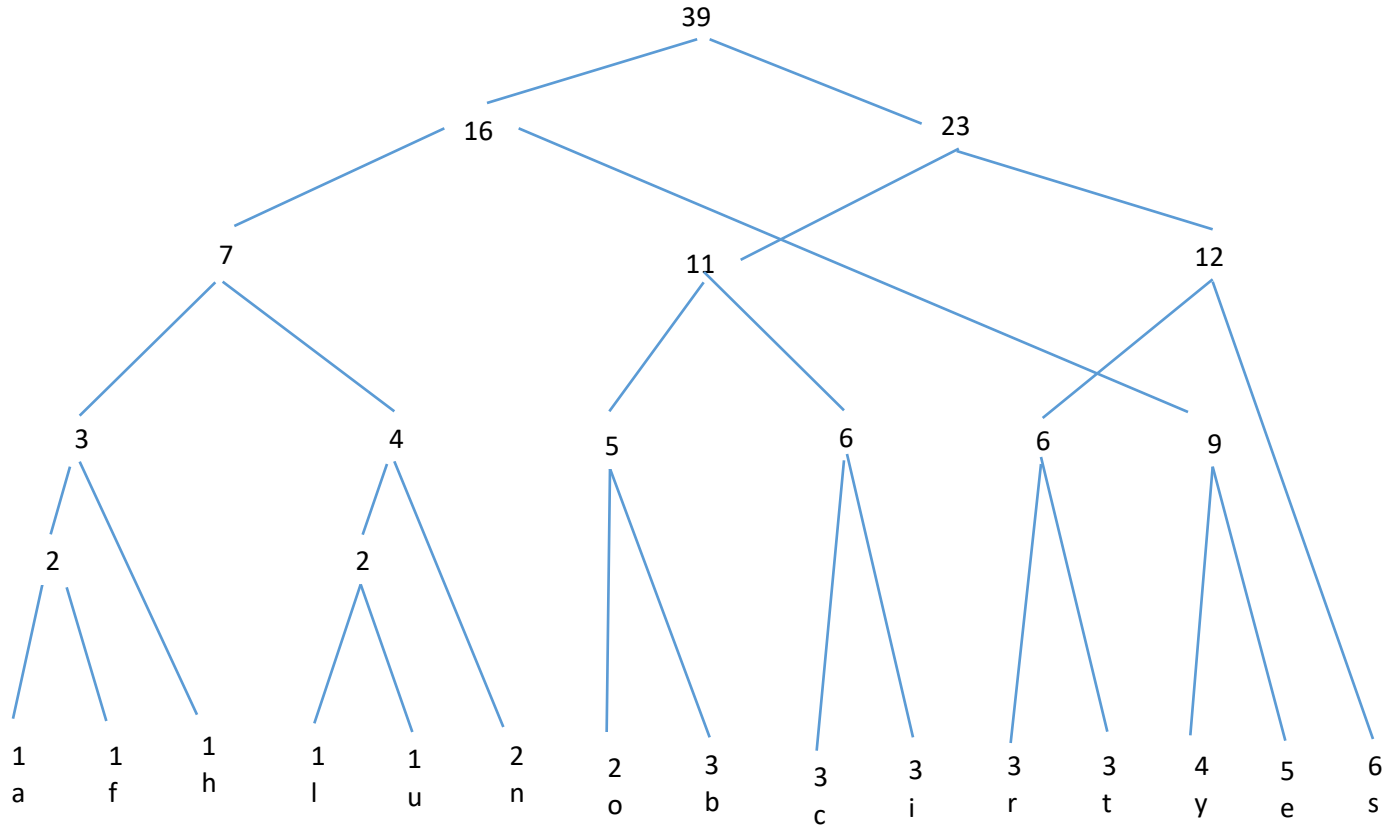
Tömörítendő szöveg:

„**Crysys is one of the best labs in cybersecurity**”

Karakter-gyakoriság tábla:

Karakter	a	b	c	e	f	h	i	l	n	o	r	s	t	u	y
Gyakoriság	1	3	3	5	1	1	3	1	2	2	3	6	3	1	4

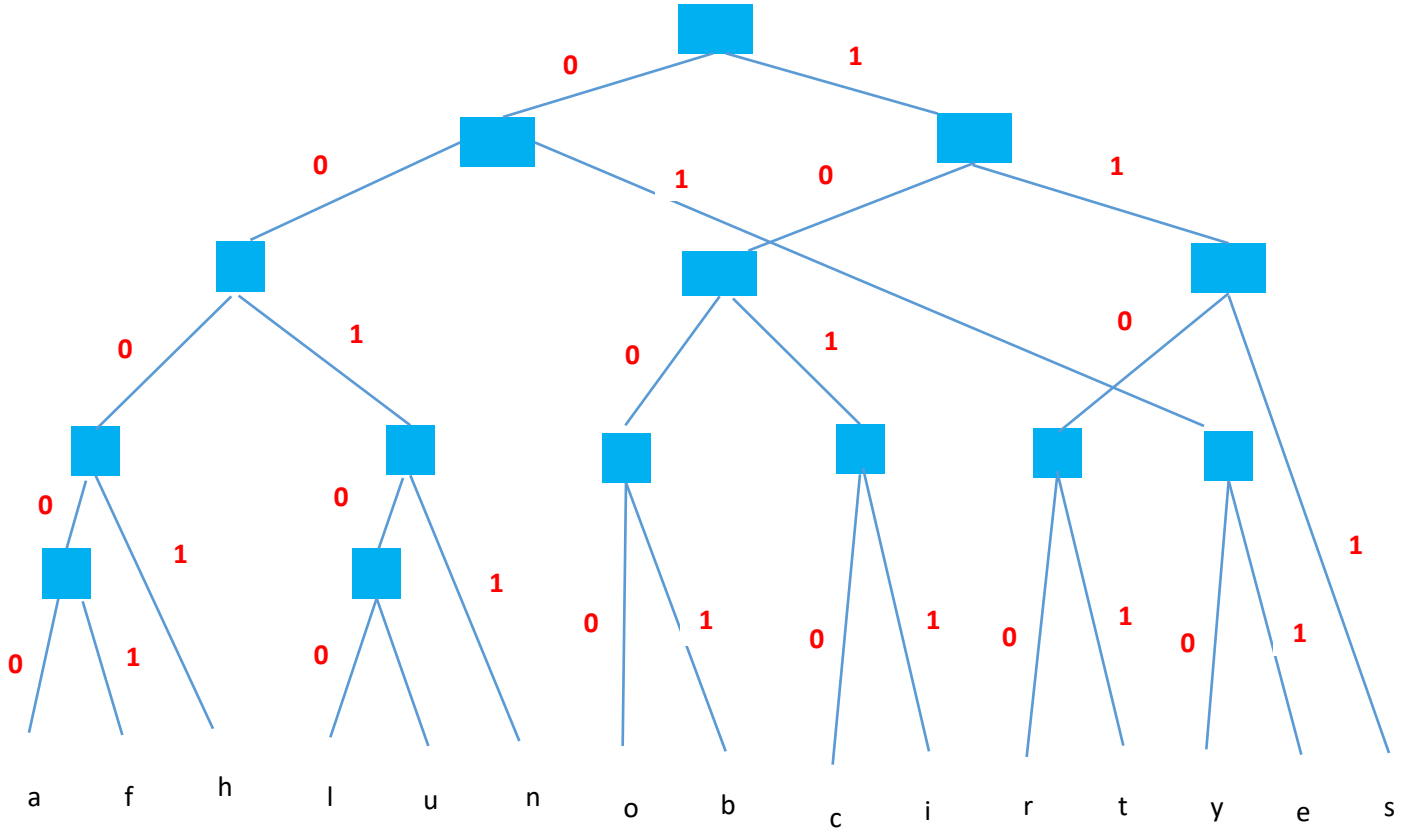
Huffman-kód: a gyakoriságok bináris fája



A fa generálása:

- a levelek a karakterekkel és gyakoriságukkal címkézve
- a gyökér felé (felfelé) haladva egy új csomópont úgy keletkezik, hogy összevonjuk azt a két csomópontot, amelyek frekvencia-összege a legkisebb (ha nem egyértelmű, egyik párt ezek közül)
- figyelembe vett csomópontot a további lépésekben már nem tekintünk

Huffman-kód: a kódszavak bináris fája



Éleket cimkézünk: egy él 0 (1) cimkéjű, ha bal oldali (jobb oldali) gyerekre mutat

Egy levélhez (~karakterhez) tartozó kódszó bitjei a gyökértől a levélig vezető út éleinek cimkéi.

Huffman-kód: kódszavak táblázata

a	f	h	l	u	n	o	b		
00000	00001	0001	00100	00101	0011	1000	1001		
c	**i**	**r**	**t**	**y**	**e**	**s**			
1010	1011	1100	1001	010	011	111			

A tömörítés ötlete: a rövidebb kódszavak tendenciájukban a nagyobb gyakoriságú karakterekhez tartoznak

Karakter:	a	b	c	e	f	h	i	l	n	o	r	s	t	u	y
Frekvencia:	**1**	3	3	5	1	1	3	1	2	2	3	**6**	3	1	4
Kódszóhossz:	**5**	4	4	3	5	4	4	5	4	4	4	**3**	4	5	3

Kódolás: a karaktereket a kódszavakkal helyettesítjük

Huffman-kód: tömörítési arány

Az input szöveg bitmérete: 39x5=195 bit

A kódolt szöveg bitmérete: 148 bit

Tömörítési arány: 148/195=0.76

LZW – algoritmus

- Veszteségmentes tömörítő algoritmus
- Szerzők: Lempel-Ziv-Welch (1984)
- Tulajdonsága: könnyen implementálható, HW implementációban nagyon gyors

Alkalmazások:

- UNIX *compress* algoritmusa
- GIF kódolásban
- PDF -ben

LZW: kódolás

Kódolás:

1. A könyvtár inicializálása: az input („szöveg”) karakterei (indexekkel)
2. Azon leghosszabb W string megkeresése a könyvtárban, amely illeszkedik a pillanatnyi inputra.
3. Outputként küldjük W könyvtári indexét, valamint eltávolítjuk a W-nek megfelelő részt az inputból.
4. Bővítjük a könyvtárat a következő stringgel: (W,q) ahol q a következő karakter az inputban
5. Ugorjunk a 2. lépésre.

Intuitív magyarázat:

Ahogy egyre beljebb haladunk az inputban, egyre hosszabb stringek kerülnek a könyvtárba, így tendenciájában az input egyre hosszabb szeleteit tömöríthetjük indexbe.

Az algoritmus jobban tömöríti azon inputokat, amelyek ismétlődő mintázatokat (rész-stringeket) tartalmaznak. Az input kezdeti szakaszán kicsi a tömörítés.

LZW: példa

- Input: m = abbababacbaabcbe
- ABC: a,b,c,(d),e
- Könyvtár inicializálás:

  a=1

  b=2

  c=3

  e=4

LZW: példa

m= **ab**babababacbaabcbe

c=1

1=a
2=b
3=c
4=e
5=ab

LZW: példa

m=**ab**babababacbaabcbe

c=12

1=a
2=b
3=c
4=e
5=ab
6=bb

LZW: példa

m=**abba**bababacbaabcbe

c=12 2

1=a
2=b
3=c
4=e
5=ab
6=bb
7=ba

LZW: példa

m=**abbaba**bacbaabcbe

c=12 2 5

1=a
2=b
3=c
4=e
5=ab
6=bb
7=ba
8=aba

LZW: példa

m=**abbababac**baabcbe

c=12 2 5 8

1=a
2=b
3=c
4=e
5=ab
6=bb
7=ba
8=aba
9=abac

LZW: példa

m=**abbababacb**aabcbe

c=12 2 5 8 3

1=a
2=b
3=c
4=e
5=ab
6=bb
7=ba
8=aba
9=abac
10=cb

LZW: példa

m=**abbababacba**abcbe

c=12 2 5 8 37

1=a
2=b
3=c
4=e
5=ab
6=bb
7=ba
8=aba
9=abac
10=cb
11=baa

LZW: példa

T=**abbababacbaabc**be

c=12 2 5 8 37 5

1=a
2=b
3=c
4=e
5=ab
6=bb
7=ba
8=aba
9=abac
10=cb
11=baa
12=abc

LZW: példa

T=**abbababacbaabcbe**

c=12 2 5 8 37 5 10

1=a
2=b
3=c
4=e
5=ab
6=bb
7=ba
8=aba
9=abac
10=cb
11=baa
12=abc
13=cbe

LZW: példa

m=**abbababacbaabcbe**

c= 12 2 5 8 37 5 10 4

1=a
2=b
3=c
4=e
5=ab
6=bb
7=ba
8=aba
9=abac
10=cb
11=baa
12=abc
13=cbe

LZW: példa

m=abbababacbaabcbe

c= (1,2,2,5,8,3,7,5,10,4)

Egy kiegészítő egyszerű optimalizálási lépés:

1 2 3 4 5 7 8 10

0 1 2 3 4 5 6 7

Igy az eredmény:

c'= (*0,1,1,4,6,2,5,4,7,3*)

Tömörítés:

Input bináris hossza (m): 16·2=32 bit

Kódszó bináris hossza (*c'*): 10·3=30 bit

(emlékeztető: rövid szövegen kisebb a tömörítési lehetőség)

Ellenőrző kérdések

- Mi a forráskódolás célja?
- Miért hívunk veszteségmentesnek egy kódolást?
- Nevezze meg a Huffmann-kódolás néhány alkalmazását!
- Nevezze meg az LZW-kódolás néhány alkalmazását!
- Hogyan generáljuk a Huffman kódolás frekvencia tábláját?
- Mi a Huffman-kódolás tömörítés ötlete?
- Mi az alaplépés Huffman kódfa építésekor?
- Mi az alaplépés LZW könytár bővítésekor?
- Mi az LZW-kódolás tömörítés ötlete?
- Milyen inputokra a leghatékonyabb az LZW tömörítés?