Teorija informacij in sistemov, predavanje 3

Uroš Lotrič

Univerza v Ljubljani,
Fakulteta za računalništvo in informatiko

3.3 Huffmanov kod 1

- Danes predstavlja osnovo mnogih tehnik za stiskanje
- Fano: od zgoraj navzdol, Huffman: od spodaj navzgor
- Dve fazi: združevanje in razdruževanje
- Združevanje
  - poišči r najmanj verjetnih znakov in jih združi v sestavljeni znak, katerega verjetnost je vsota verjetnosti vseh znakov
  - preostale znake skupaj z novo sestavljenim znakom spet razvrsti
  - postopek ponavljaj dokler ne ostane samo r znakov
- Razdruževanje
  - vsakemu od preostalih znakov priredi po en znak kodirne abecede
  - vsak sestavljeni znak razstavi in mu priredi po en znak kodirne abecede
  - ko zmanjka sestavljenih znakov, je postopek zaključen

3.3 Huffmanov kod 2

- Primer 1 $\{0.5, 0.25, 0.125, 0.125\}$, $H = L = 7/4, \eta = 1$
- Primer 2: $\{0.39, 0.19, 0.16, 0.13, 0.13\}$, $L = 2.22, H = 2.17, \eta = 0.98$
- pokaži še gradnjo drevesa brez razvrščanja

3.3 Je Huffmanov kod res optimalen?

- Optimalen kod ne sme imeti neizrabljenih vej - če bi jih imel, bi obstajale krajše kodne zamenjave, obstajal bi bolj gospodaren kod.
- Zaradi zgornjega morajo najdaljše kodne zamenjave nastopati v parih. Par se razlikuje samo v zadnjem znaku.
- Najdaljše kodne zamenjave ustrezajo najmanj verjetnim znakom - če ne bi, bi povprečno dolžino kodnih zamenjav zmanjšali, če bi namesto njih postavili manj verjetne znake.
- Če ima osnovni kod povprečno dolžino L, ima kod, v katerem sta najmanj verjetna znaka združena, v najboljšem primeru povprečno dolžino $L - 1 \cdot (p_q + p_{q-1})$. Tak je Huffmanov kod.
- Ko nam pri postopku združevanja ostaneta samo dva sestavljena znaka, za njiju velja $L = 1$. Če bi obstajal bolj optimalen kod, bi imel $L < 1$. To pa ni mogoče.

3.3 Je vsak optimalen kod Huffmanov?

- Vsak optimalen enoznačen kod ni Huffmanov. Dokaz s primerom:
  - $P = \{1/2, 1/4, 1/8, 1/8\}$,
  - $C_H = \{0, 10, 110, 111\}$
  - $C_? = \{0, 01, 011, 111\}$
- Vsak optimalen trenuten kod ni Huffmanov. Dokaz:
  - $P = \{0.3, 0.3, 0.2, 0.2\}$
  - $C_H = \{10, 11, 00, 01\}$
  - $C_? = \{00, 10, 01, 11\}$

3.3 Nedoločenost Huffmanovega koda

- Huffmanov kod ni en sam
  - Zamenjava ničel in enic (povsod, po posameznih vejah)
  - Gradnja dreves odvisna od razvrščanja simbolov z enakimi verjetnostmi
- Postopek ponudi več enakovrednih rešitev
- Primer 3:

x_i	p_i	Kod 1	Kod 2
x_1	1/3	1	00
x_2	1/3	00	01
x_3	2/9	010	10
x_4	1/9	011	11

$L = 2, H = 1.89, \eta = 0.99$

3.3 Huffmanov kod za r > 2

- Primer 4 (r=3): ($H_3 = 1.54$)

x_i	p_i	Kod (? na koncu)	Kod (? na začetku)
x_1	1/3	00	1
x_2	1/6	01	00
x_3	1/6	02	01
x_4	1/9	10	02
x_5	1/9	11	20
x_6	1/9	12	21
?	0	2	22
		$L = 2, \eta = 0.77$	$L = 1.67, \eta = 0.93$

- najprej narobe, potem prav
- kodiranje začnemo, če za število znakov velja $n = r + k(r-1), k \geq 0$

3.3 Problem Huffmanovega koda

- če je entropija vira majhna, je odstopanje med entropijo in L veliko
- problem dodatnega znaka
- primer: $\mathrm{P} = 0.999, 0.001$, $H = 0.081$
  - Shannon: $\lceil -\log_2 0.999 \rceil = 1$, $\lceil -\log_2 0.001 \rceil = 10$, $L_S = 1.009$
  - Huffman: $L_H = 1$
  - $H \le L_H \le L_S < H + 1$
- Rešitev nam predlaga prvi Shannonov teorem

3.3 Kodiranje z razširitvijo abecede

- Več osnovnih znakov združujemo v sestavljene znake
- S kodiranjem abecede zgrajene iz sestavljenih znakov dobimo bolj učinkovite kode
- Primer, Huffmanov kod
  $X = \{A, B\}$, $P = \{0.75, 0.25\}$, $L = 1$
  $X^2 = \{AA, \ldots, BB\}$, $L/2 = 0.8438$
  $X^3 = \{AAA, \ldots, BBB\}$, $L/3 = 0.8229$
- Problem: kombinacijska eksplozija

3.4 Aritmetični kod: uvod

- state-of-the art pri brezizgubnem stiskanju podatkov
- je **blizu optimalnemu** kodu in je **hiter**; ni tako učinkovit kot Huffmanov kod, vendar ni težav s kombinacijsko eksplozijo
- Vsak niz je predstavljen kot realno število $0 \leq R < 1$
- Daljši kot je niz, bolj natančno mora biti podano realno število R

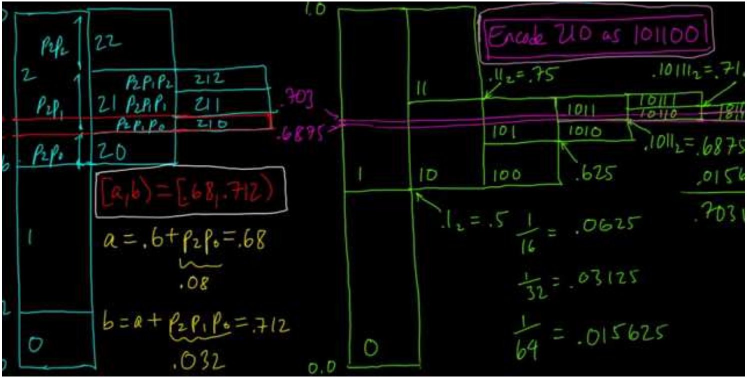
3.4 Aritmetični kod: postopek kodiranja

- **Znakov nam ni treba razvrstiti!!!**
- Začnemo z intervalom $[0, 1)$.
- Izbrani interval razdelimo na n podintervalov, ki se ne prekrivajo. Širine podintervalov ustrezajo verjetnostim znakov. Vsak podinterval predstavlja en znak.
- Izberemo podinterval, ki ustreza iskanemu znaku.
- Če niz še ni končan, izbrani podinterval ponovno razdelimo.
- Niz lahko predstavimo s poljubnim realnim številom v zadnjem podintervalu.

3.4 Aritmetični kod: primer z razlago 1

Primer 1: $A = \{0, 1, 2\}, P = \{0.2, 0.4, 0.4\}$. Kodiranje niza

- najprej decimalno, potem še binarno



- Ko pridemo do 101, se je treba odločiti za 1010 ali 1011. Odločimo se za drugega, ker je 0.6875 bolj stran od 0.712 kot od 0.68.

3.4 Aritmetični kod: primer z razlago 2

- Dekodiranje
  - Enak postopek, pokaži kar na obstoječi sliki
  - Kdaj se ustavimo?
    - Vemo koliko znakov
    - **Imamo stop znak** - naj bo to 0: veljavni nizi: 210, 0, 10, 2120, ...

3.4 Aritmetični kod: primer z razlago 3

- Zakaj interval ne pa samo številka v intervalu?
- 1011 je v tem intervalu. Zakaj še dve ničli?
- Naš primer: Vzemimo, da imamo niz 1011|0111....
  - zamenjava 1011 –> 0.6875 je v [0.68, 0.712) –> 210 –> se ustavimo
  - zamenjava 1011 definira širši interval [0.6875, 0.75), ki vsebuje neskončno drugih kodnih zamenjav.
  - zamenjava 10110111 –> gre v interval [0.71484375, 0.71875] –> 2110
- številka v intervalu lahko, če povemo število znakov. Dodaten strošek.
- če je tudi zgornja meja v intervalu, nadaljnji znaki ne morejo povzročiti preskoka v drug interval.
- **binarni interval mora biti vsebovan v decimalnem intervalu!**

3.4 Aritmetični kod: primer z razlago 4

- hitrejše računanje binarnega koda
- pretvarjanje dec –> bin z upoštevanjem mej intervalov
- zaradi lažjega risanja namesto manjšanja binarnih intervalov množimo ostanke v decimalnih intervalih
- na podoben način delajo tudi algoritmi v končni natančnosti

3.4 Aritmetični kod in entropija

- Komentar: Huffman (+1/znak), bločni Huffman (+1/blok) in aritmetični (+2/niz)
- Imamo $X = \{0, 1, \dots, n\}$, $P = \{p_0, \dots, p_n\}$, $0 =$ stop.
- Za poljuben niz $\tilde{x}_i = x_{\pi(1)} \cdots x_{\pi(m)} 0$ velja $\tilde{p}_i = p_{\pi(1)} \cdots p_{\pi(m)} \cdot p_0 = b - a$.
- Vsak dodaten znak binarne kodne abecede prepolovi interval. Na prvo žogo mora veljati $2^{-l_i} \leq b - a$.
- To ni dovolj zaradi diskretne delitve intervalov. Vedno pa se bo našel pol ožji interval, ki bo znotraj [a, b). Torej: $\frac{1}{2^{l_i}} \leq \frac{b-a}{2} = \frac{\tilde{p}_i}{2} \rightarrow l_i \geq -\log \tilde{p}_i + 1 \rightarrow l_i = \lceil -\log \tilde{p}_i + 1 \rceil$.
- Pri kodiranju celotnega niza porabimo le dva znaka preveč: $L = \sum_i \tilde{p}_i l_i \leq \sum_i \tilde{p}_i(-\log \tilde{p}_i + 2) = \tilde{H} + 2$
- Pri kodiranju v končni aritmetiki je še nekaj več kot 2

3.4 Aritmetični kod: primer 2

- $A = \{a, b\}, P = \{0.9, 0.1\}$
- Kodiranje niza 6 "a-jev: "aaaaaa"da kodno zamenjavo 0
- Kodiranje niza 13 "a-jev: "aaaaaaaaaaaaa"da kodno zamenjavo 00
- Več je znakov v nizu, krajši je kod.