Önálló laboratórium beszámoló

Távközlési és Médiainformatikai Tanszék

Készítette:	**Baki Tamás**
Neptun-kód:	**FNARVP**
Ágazat:	**Internet architectúra és szolgáltatlások**
E-mail cím:	**bakitamas23@gmail.com**
Konzulens:	**Dr. Rétvári Gábor**
E-mail címe:	**retvari@tmit.bme.hu**

Téma címe: Forgalom irányítási táblák nexthop információinak elemzése

Feladat

A dolgozat témája a korábbi félévekben beszerzett FIB és RIB adattáblák további feldolgozása. A fő kérdés, hogy a publikusan elérhető RIB információk alapján következtethetünk-e a FIB belső felépítésére. A kérdést most abból a szemszögből fogom megvizsgálni, hogy a nexthop-ok vizsgálatával megfigyelhető-e valamilyen rendszer a FIB-ek és a RIB-ek között.

A statisztikai jellemzők a kiszámolásához először több algoritmustervezési problémát kell megvalósítani az adathalmazok méretéből kifolyólag. Szükség van egy olyan algoritmusra, ami a meglévő fának a belső elemeiből kitolja az információkat a levélelemekbe. Majd ezekből egy futáshossz-kódolással tömörített kezelhető méretű bemeneti adatot kell képezni. Az adathalmazokból ilyen módon képzett adatot páronként össze lehet hasonlítani, mátrixokba rendezni. Ezeken a mátrixokon az adott pár rendezettsége, entrópiája, kölcsönös információi kiszámíthatóak, és ezekből a mennyiségekből kvantitatívan jellemezni tudjuk majd a RIB-ekben és a FIB-ekben szereplő next-hop-ok közötti kapcsolat "erősségét".

2020/2021. 1. félév

A laboratóriumi munka környezetének ismertetése, a munka előzményei és kiindulási állapota

1.1 Bevezető

Az Internet kialakulása kezdetben négy amerikai egyetemnek az összekapcsolt hálózatából indult és mára már 70200 autonóm rendszerből (Autonom System - AS) áll. Ezen hálózatok közötti útvonalválasztásnál kiemelten fontos, hogy megfelelő gyorsaságú és megbízható legyen. A Border Gateway Protocol (BGP) valósítja ezt meg. A protokollt futtató routerek (útválasztó) az AS-ek „szélein" helyezkednek el és különböző protokoll üzentek segítségével tartják fenn a hálózat, illetve jelen esetben a teljes Internet működését. (1)

Az AS-ek közötti útválasztás nem csupán a fizikai értelemben vett legjobb útvonal alapján történik, hanem túlnyomóan gazdasági döntések szabják meg az egyes útvonalakat, ezért a BGP-t gyakran szokás „policy routing"-nak nevezni, mert útvonal választási irányelveket lehet benne definiálni.

Az Internet terjedése az elmúlt 15 évben olyan mértékben növekedett, hogy kifut a BGP adta lehetőségekből. A problémát az jelenti, hogy a gyors működéshez a BGP routereknek olyan táblákat kell fenntartani, amiben pontosan tudja, hogy melyik AS merre található. Kettő féle tábláról beszélhetünk a BGP esetében.

Az egyik ilyen a forgalomirányítási tábla (Routing Inforamtion Base - RIB), amelyben a szomszédos routerektől kapott információk kerülnek eltárolásra. Így a RIB-ben az összes útvonal szerepel és ezekhez különböző metrikák vannak meghatározva, mint a távolság, preferencia, melyik protokolltól származik (nem csak BGP üzentek lehetnek), útiköltség információk. Így egy cél felé akár több esetleges útvonal is létezhet. A RIB mérete nagyon nagy is lehet, és ebből a megfelelő szempontok alapján kell a legjobb útvonalat kiválasztani.

A RIB alapján kiválasztott útvonalakból egy másik tábla készül. Ezt nevezzük forgalom továbbítási táblának (Forwarding Information Base - FIB). A FIB tartalmazza azt a ma nagyságrendileg 833000 bejegyzést, ami alapján a BGP-t futtató router eldönti, hogy merre kell küldeni minden egyes csomagot. A táblában csak az AS-ek prefixei és az azokhoz tartozó next-hop párjai vannak. Nem csak az első illeszkedő prefixet kell megtalálni, hanem a legjobban illeszkedőt, és nem csak gyorsan, hanem azonnal. [1]

Vegyünk egy rövid számpéldát az „azonnal" szemléltetésére: 1 Gbit/s kapcsolaton érkezik 250000 csomag egy másodperc alatt, ha átlagosan 500 bájtos csomagmérettel számolunk. A router processzor sebessége mondjuk 3 Ghz. Ebben az esetben 12000 órajel jut egy csomag feldolgozására. Viszont a 833000 prefixből kell megkeresni azt, hogy merre küldjük tovább. Ez nyilvánvalóan nem lehetséges. Ezeket a problémákat egy cél hardver oldja meg, melynek részleteibe most nem mennék bele. Így a FIB terjedésének mértékét ez a memória korlátozza. [2]

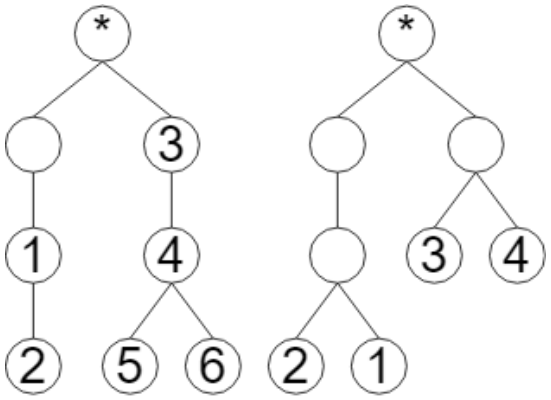
Minden olyan mérnök, aki hálózat elemzéssel foglalkozik, vagy Internet térképezéssel, ugyan azzal a problémával szembesült, hogy az AS-eknek az útválasztási preferenciája nem nyilvános, sőt kifejezetten titkolt. Így kénytelenek a publikusan elérhető RIB információk alapján megbecsülni. Az oregoni egyetem gyűjtésében vannak ilyen táblák a világ minden részéről. A tanszék kapcsolatai révén megkaptam a HBONE négy teljes FIB tábláját.[3][4]

1.2 Elméleti összefoglaló

1.2.1 Leaf-Push algoritmus

Az 1.2.1/b. ábrán látszó alakból egy prefix fa generálása úgy zajlik, hogy a prefix oktális alakját át kell alakítani egy bináris karaktersorozattá. A karaktersorozaton végig lépdelve kell építeni a fát ha 0 jön, akkor csomópont bal gyereke lesz ha pedig 1-es jön, akkor az adott csomópont jobb gyereke lesz. Az úton végig menve pedig kiolvasható, hogy az adott csomópont milyen prefixnek feleltethető meg.

A Leaf-Push algoritmust egy bináris fán hajtjuk végre. A célja, hogy a fa belső csomópontjaiból a levelekbe tolja az információt, így elég a leveleket vizsgálni.



1.2.1/a. ábra Leaf-Push algoritmus példa

1.0.0.0/24	195.111.97.108
1.0.4.0/24	195.111.97.108
1.0.5.0/24	195.111.97.108
1.0.6.0/24	195.111.97.108
1.0.7.0/24	195.111.97.108
1.0.20.0/23	195.111.97.108
1.0.22.0/23	195.111.97.108
1.0.24.0/23	195.111.97.108
1.0.26.0/23	195.111.97.108
1.0.28.0/22	195.111.97.108
1.0.64.0/18	195.111.97.108

1.2.1/b. ábra FIB adatforrás példa

Az 1.2.1/a. ábrán a baloldali a kiinduló fa, amiben az 1,3,4 –el jelölt csomópontok hordoznak információt, amelyek belső csomópontok, ezeket szeretnénk kivinni a levelekbe. Ezt úgy érjük el, hogy végig megyünk az összes csomóponton, és ha egy gyereke van, mint az 1,3 csomópontnak, akkor az ott tárolt információkat egy szinttel lejjebb toljuk a gyereke mellé. Viszont ha 2 gyereke van, akkor az a csomópont, mint a 4-essel jelölt, nem hordoz új információt, így eldobjuk. [5]

1.2.2 Futáshossz-kódolás

A futáshossz-kódolás egy nagyon elterjedt veszteségmentes tömörítési eljárás, melynek segítségével a hosszasan ismétlődő karaktereket lehet egyetlen karakterrel és egy ismétlődés számmal helyettesíteni. Ezt gyakran egyszerűbb képek esetén, mint vonalrajzok, ikonok, alacsony színtelítettségű képek esetén használják.

Például nézzük meg a 1.2.2. példát, ahol eltároljuk egy „**H**” betű képét.[6]

```
Tegyük fel, hogy van egy fekete-fehér képem, aminek a pixel
információi sorra:
BBBWWWBBBBBBBBBBBBBBBWWWBBB
Ez a jelen esetben 27 karakter, ha futáshossz-kódolással tárolom el,
akkor ez úgy változik, hogy:
B3W3B15W3B3
Így viszont 11 karakterrel le tudom írni.
```

1.2.2. példa Futáshossz-kódolás

1.2.3 Kontingencia táblázat és peremeloszlás

A kontingencia táblázat a statisztikában egy olyan, mátrix formájú táblázat, amely a változók gyakorisági eloszlását mutatja. Az 1.2.3. táblázat egy példa, amiben tegyük fel, hogy van két változó: a nem és a kezesség. Megkérdeztünk 50 nőt és 50 férfit, hogy melyik a domináns keze és ezt szemléltetni készítettünk egy kontingencia táblázatot, hogy egyes diszjunkt csoportokban hány fő található.

	Jobbkezes	Balkezes	Összes
Férfi	42	8	50
Nő	46	4	50
Összes	88	12	100

1.2.3 táblázat Kontingencia táblázat példa

A soronként és oszloponként összesített gyakoriságokat nevezzük peremeloszlásnak. A peremeloszlások több valószínűségi változó közös eloszlásának jellemzője. Tehát a peremeloszlások a kontingencia táblázat peremén szereplő eloszlások.[7]

1.2.4 Entrópia

Az entrópia az informatikában is a rendszer rendezetlenségét fejezi ki, pont úgy, mint ahogy a XIX. században a hőtanban is megfogalmazták.

Az entrópia tulajdonképpen azt adja meg, hogy mekkora valószínűségekkel kell súlyozni az összes jel információtartalmát.

$$H(S) = -\sum_{i=1}^{n} p_i * log_2 * p_i$$

1.2.4. képlet Entrópia

Ahol:

H ={h1,h2,h3,…} – a forrás jelkészlete
P = {p1,p2,p3,…} – az állapotvalószínűségek
n – a H elemszáma
H(S) – A rendszer entrópiája

A rendszer entrópiája a következő értéket veheti fel: $0 \leq H(S) \leq log_2 n$, ahol n továbbra is a H elemszáma.

- Az entrópia akkor a legkisebb (0), ha a jelforrás biztosan mindig ugyanazt a jelet sugározza: ekkor a p_i valószínűségek egyike 1, a többié pedig 0
- Az entrópia akkor a legnagyobb ($log_2 n$), ha az összes jel valószínűsége egyenlő ($p_i = \frac{1}{n}$). Ekkor a bizonytalanságunk a legnagyobb, hiszen bármelyik jel ugyanakkora valószínűséggel érkezhet.[8]

1.2.5 Feltételes entrópia és kölcsönös információ

Ezeket a fogalmakat két vagy több forrás együttes, átlagos információtartalmának a jellemzésére használjuk. A feltételes entrópia megadja, hogy az A forrás ismeretében, ahhoz képest átlagosan mennyivel tér el a B forrás.

$$H(A,B) = \sum_{i,j} p_{i,j} * log \frac{1}{p_{i,j}}$$

1.2.5/a. képlet Kölcsönös entrópia

$$H(B|A) = \sum_{i,j} p_{i,j} * log \frac{p_{i,j}}{p_i}$$

1.2.5/b. képlet Feltételes entrópia

Ahol:

A, B – két jelforrás
$p_{i,j}$ – annak a valószínűsége, hogy A az i-edik B pedig a j-edik jelet tartalmazza
H(B|A) – a B forrásnak az A forrásra vonatkozó feltételes entrópiája

A képletből látható, hogy a *H(B|A)* feltételes entrópia nem más, mint a B forrásnak az A forrás egyes elemi eseményeihez tartozó entrópiáiból képzett súlyozott átlag.

- B forrás **teljesen független** A-tól: akkor a *H(A,B)* együttes entrópia a két forrás külön-külön vett entrópiájának összegével egyenlő. Ekkor tehát $H(A,B) = H(A) + H(B)$.
- B forrás az A által **teljesen meghatározott:** akkor nem várunk új információt a B-ből. Azaz: $H(A,B) = H(A)$ és $H(B|A) = 0$ adódik.

Összegzésképpen megállapíthatjuk, hogy $0 \leq H(B|A) \leq H(B)$.

Az A és B közti kölcsönös információ, *I(A,B)* fogalmát úgy definiáljuk, mint „a B forrás átlagos információtartalmából az a rész, amely az A által meghatározott”.

$$I(B,A) = H(B) - H(B|A)$$
$$I(B,A) = I(A,B)$$

1.2.5/c. képlet Kölcsönös információ

Ha a két forrás **teljesen független** egymástól, akkor a $H(B|A)$ feltételes entrópia H(B)-vé alakul, azaz a kölcsönös információ 0.

Ha viszont a B az A által **teljesen meghatározott**, akkor $H(B|A) = 0$, azaz $I(B,A) = H(B)$. [8][9][10]

1.3 A munka állapota, készültségi foka a félév elején

Az önálló laboratórium során a BSC szakdolgozatomat, és a korábbi önálló laboratóriumi munkámat folytatom. A szakdolgozatban a nyers FIB-ek feldolgozását végeztem, belőlük kinyertem a statisztikai számításokhoz használható alap metrikákat. Ezt követően a publikusan elérhető RIB információkból kiszűrtem ugyanazokat a részeket, amik a FIB-ekben is szerepelnek, majd a FIB-ekhez hasonló módon leszámoltam a metrikákat. A szakdolgozat továbbá ezeknek az adatoknak a vizualizációja és elemzése is volt, de most csak az azok által is felhasznált adatokat fogom felhasználni.

A négy FIB tábla a HBONE négy teljes BGP routeréből származik: BME, a Szegedi Tudományi Egyetem (szeged), valamint a BIX-ben lévő vh1 és vh2. A négy RIB pedig a világ négy sarkából: a Londoni Internet Exchange (linx) RIB, az Amerikai Equinix Ashburn (eqix), ami a Washington melletti IX (Internet Exchange), a Japán Internet Exchange-t (jinx) és a Sydney Internet Exchange-t (sydney).

A szakdolgozat végére megvoltak a négy FIB és négy RIB leszámolt értékei, melyek a naponkénti bejegyzések darabszáma, a specifikusabb prefixek száma, valamint ezek prefix hosszonkénti bontásban. Mindegyik napra ki van számolva a teljes IPv4 tartomány fedése per nyolc ekvivalens formában.

Az előző félévben végzett munka egy teljesen másik aspektusból vizsgálja az adatokat, így azok nem befolyásolják jelen önálló laboratóriumot.

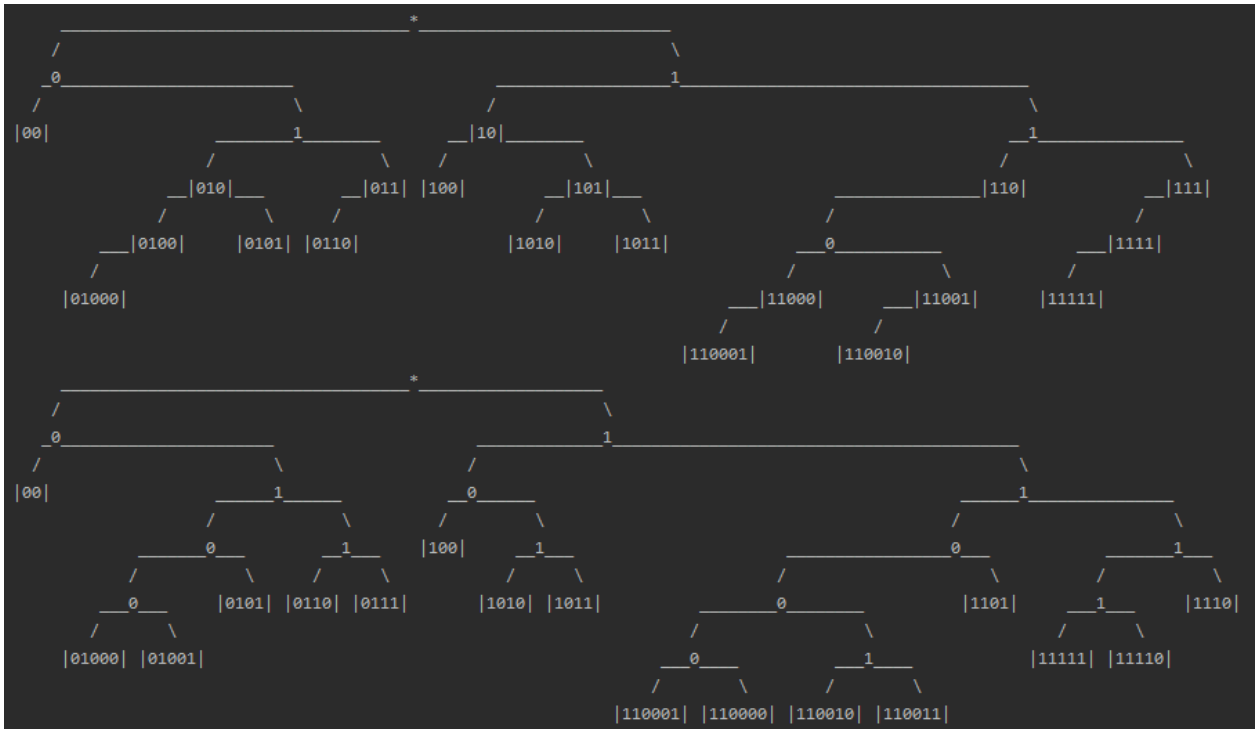
2 Az elvégzett munka és eredmények ismertetése

2.1 Leaf-Push megvalósítás

A kiindulási alap egy olyan nem teljes bináris fa, aminek nem minden csomópontja hordoz információt. Ez a bináris fa tulajdonképpen a korábbi félévben létrehozott prefix fa. Ebben a prefix fának egy-egy gyökér-levél útvonala maga a prefix, amit eltároltam benne. A csomópontoknak csak akkor van információtartalma, ha az egy létező prefix végét jelöli. Mivel a prefixek változó hosszúságúak és létezik olyan prefix, amelynél van specifikusabb, tehát, hogy egy bizonyos szakaszon a prefix eleje megegyezik, majd az egyik tovább folytatódik. Így fordulhat elő olyan csomópont, ami hordoz hasznos információt, de nem levélben szerepel.

A Leaf-Push algoritmus lényege, hogy megszüntesse ezeket az információt tartalmazó belső csomópontokat és kitolja őket egy levél csomópontba. Ezt úgy valósítottam meg, ha létezik az adott csomópont és nincs neki gyereke, akkor természetesen az a csomópont levél volt. Ha viszont csak egy gyereke van, akkor belső csomópont és lejjebb kell tolni a fában.

A lejjebb tolt csomópont természetesen fele akkora tartományt fed majd le a címtérben, és ennek megfelelően kell majd kezelni, amikor számbavételre kerül.



2.1.1. ábra Teszt prefix és Leaf-Pushed fa

A 2.1.1 ábrán egy kis példa fa látható, amit az algoritmusom fejlesztése közben is használtam a könnyebb átláthatóság miatt. Az első fa a kezdeti állapotot mutatja a „*” jelű gyökér csomóponttal, az egy 0-ból vagy 1-esből álló információ nélküli csomópontokkal és a „|” közötti prefix értékekkel, amik a szükséges információkat tartalmazzák.

Két példát kiemelnék: a „|101|” jelzett csomópontnak két létező gyereke van, nem hordoz hasznos információt, vagyis az útválasztásban nem szerepel, tehát eldobható. Így az algoritmus helyesen egy sima információ nélküli csomópontot csinált belőle. A másik az „|010|” jelzett

csomópont, aminek egy gyereke van. Így az algoritmus egy szinttel lejjebb tolta és ennek megfelelően megváltoztatta az új perfixet, amivel elérhető.

2.2 Futáshossz-kódolás

Erre azért van szükség, mert az IPv4 es címek 32 bit hosszúak, tehát 4,3 milliárd különböző értéket vehetnek fel, ennek egyesével való feldolgozása rengeteg időbe kerülne. A futáshossz-kódolás IP tartományokra kifejezetten hasznos. Hiszen gondoljunk bele, hogy az IP tartomány elején van 16 millió használatlan cím, mivel ezek a localhost számára vannak fenntartva. A végén pedig ~530 millió, amik különböző multicast-ra és jövőbeni használatra vannak lefoglalva. Ezeket a tartományokat egy-egy next-hop cím és tartományméret bejegyzéssel össze lehet foglalni. Ha viszont a használt tartományokat nézzük, még úgy is, hogy csak 256-osával lehet össze fogni a többségét, így is körülbelül a 0,0125%-ára csökken a mérete.[11]

A konkrét megvalósításban egy preorder fa bejárással végignézem a Leaf-Pushed prefix fát. A preorder bejárás szükséges, hogy növekvő sorrendben kapjam meg a tartományokat. A tartományok mérete pedig a fa gyökerétől számított távolsága alapján számolható úgy, hogy: $r_i = 2^{32-h_i}$.

A 2.2.1 ábra alapján, a példa kedvéért tegyük fel, hogy 8 hosszúak lehetnek maximum az IP címek. Így az „101” csomópont a 160/3-as IP cím egy 32 hosszú tartományt jelentene, míg az „110010” csomópont a 200/6-os IP cím pedig egy 4 hosszú tartomány lenne.

2.3 Kontingencia táblázatba rendezés

Két adatforrást ezután párba kell állítanom, amiket össze szeretnék hasonlítani. Legyenek a táblázat sorai a FIB next-hop-jai, az oszlopai pedig egy RIB next-hop-jai. Az első sor/oszlopába a nem routolt tartományok kerülnek, a többibe pedig sorra a next-hop-ok előfordulás szerinti sorrendben. Egy p_{ij} cellába akkor kerül érték, ha van olyan tartomány, hogy a FIB-ben az i-edik nexthoppal van bejegyezve, a RIB-ben pedig a j-edik next-hop-pal. Ezzel egy ritka táblázatot kapok, ami tele van nulla értékekkel és néhány cella pedig nagy értékeket tartalmaz.

A kontingencia táblázatba valószínűségi értékeknek kell lennie, így minden elemet leosztok 2^{32}-el, ezáltal minden elem nulla és egy között lesz. Ebből egy cellát nézve az lesz leolvasható, hogy mennyi annak az esélye, hogy egy FIB-nek egy tartománya i-edik next-hop-ra irányítódik, akkor az a RIB-nek a j-edik next-hop-jára mutat. Ha egy sort/oszlopot összeadunk, akkor annak a konkrét next-hop-nak a peremeloszlását kapjuk meg, vagyis a konkrét előfordulási valószínűségét. Ha a teljes mátrixot összeadjuk, akkor pedig természetesen egyet kell kapni.

A perem eloszlásokat a 2.3.1. táblázatban össze írtam egy rövid példát az egyik pár alapján.

FIB		RIB	
nexthop	eloszlás	nexthop	eloszlás
195.111.97.108	0.5117939379314688235	206.126.236.21	0.00000884226057205
195.111.103.206	0.0018862153484770691	206.126.236.26	0.10578348848443073
195.111.97.1	0.0002870532597856228	206.126.236.37	0.50600993075679362
195.111.97.13	0.0001318061115799954	206.126.236.19	0.02197287529062274
193.224.58.254	0.0000003931452734752	206.126.236.76	0.32854204055589903
195.111.97.109	0.0005926231121430691	206.126.236.12	0.16743501729096173
195.111.97.84	0.0000031810972806304	206.126.236.88	0.04629687946217756
195.111.97.78	0.0000009733260471505	206.126.236.154	0.07520369714967666

2.3.1. táblázat BME-EQIX peremeloszlás részlet

2.4 Entrópia

Az entrópiát a mátrix alapján könnyen ki tudtam számolni, a 2.4.1 táblázatban látszanak a kapott értékek.

	Entrópia	Entrópia Maximuma
bme	1.0013607940261027	6.459431618637297
szeged	1.0013161919863975	6.392317422778761
vh1	1.851280638786121	7.727920454563199
vh2	1.0013561074544708	6.6293566200796095
eqix	2.0513936552028604	6.832890014164741
jinx	1.021311003277895	5.044394119358453
linx	2.123874218030973	8.055282435501189
sydney	1.584219173863598	6.491853096329675

2.4.1. táblázat Adatforrások entrópiája és maximuma

Az entrópia értékei viszonylag kicsik a várakozásnak megfelelően. A sok nem route-olt tartomány nagyon torzítja az értékeket.

2.5 Feltételes entrópia

A feltételes entrópiát az 1.2.5/b. képlet alapján kiszámoltam minden pára úgy, hogy a RIB ismeretében, ahhoz képest átlagosan mennyivel tér el a FIB. Azt várjuk a feltételes entrópiáktól, hogy legyenek a H(FIB) értékekhez közeliek, vagyis legyen teljesen független.

	eqix	jinx	linx	sydney
bme	0.0574	0.061723293	0.106969	0.12483865691916
szeged	0.0295	0.027257399	0.008389	0.00000736900124
vh1	0.0295	0.027287768	0.008428	0.00007520059419
vh2	0.0295	0.027287768	0.008428	0.00007822365520

2.5.1. táblázat *H(FIB|RIB)* feltétes entrópiák

A 2.5.1. táblázat tanulsága pont nem az, amit vártunk, az értékek inkább azt támasztják alá, hogy függnek egymástól. Ezt a hatást a sok nem route-olt címek okozzák.

2.6 Kölcsönös információ

A kölcsönös információ kiszámítása az 1.2.5/c. képlet alapján megint nem azt az eredményt hozta, amit vártunk tőle. Ugyanis ezek elég magas értékek mi viszont 0 szerettünk volna kapni. Ez azt jelenti, hogy a FIB jelentős mértékben meghatározott a RIB által.

	eqix	jinx	linx	sydney
bme	0.979265976	0.976956	0.9613	0.955608
szeged	1.007199751	1.011422	1.05988	1.080439
vh1	1.007169957	1.011392	1.059841	1.080368
vh2	1.007169957	1.011392	1.059841	1.080368

2.6.1. táblázat I(FIB,RIB) kölcsönös információ

A 2.6.1. táblázat jól mutatja, hogy a kölcsönös információk közel vannak a maximum értékükhöz ami a H(FIB), a FIB-ek entrópiáji. Ez a sok nem route-olt tartomány miatt van.

2.7 Összefoglalás

A fő kérdés az volt, hogy a publikusan elérhető RIB információk alapján következtethetünk-e a FIB belső felépítésére. A kérdést abból a szemszögből néztem meg, hogy a nexthop-ok vizsgálatával megfigyelhető-e valamilyen rendszer a FIB-ek és a RIB-ek között.

A kérdésre most nem kaptam kielégítő választ, és nem is volt cél. Ennek a válasznak a felderítése egy későbbi félév feladata lesz, most az volt a cél, hogy előállítsam a szükséges adatokat, elkészüljön a szükséges metodika, ami lehetővé teszi ugyan ennek a vizsgálatnak az elvégzését újabb adatokon.

Ennek lépései elsőként a korábbi félévek munkájára alapozva kiválasztok egy olyan időpontot ahol mind a 8 (4 FIB, 4 RIB) elérhető ezek lesznek a jelenlegi munka adatforrásai. A prefix fán elvégzett Leaf-Push algoritmussal megkaptam a tényleges IP tartományhoz tartozó nexthop címeket. Az IP tartomány és nexthop párokból egy futáshossz-kódolás tömörítési eljárással kezelhető méretű bementeti adatot tudtam előállítani.

A most már használható formájú adatot össze tudom hasonlítani páronként. Az összehasonlításhoz egy mátrixot hozok létre, aminek a sorai az egyik adathalmaz az oszlopai pedig a másik így előáll az adott pár kontingencia táblázata.

Kontingencia táblázaton a peremeloszlás kiszámolható könnyen kiszámolható, ami szükséges lesz az adott pár rendezettségének, entrópiájának kiszámolásához és a kölcsönös információjának kiszámolásához.

Eredményül összeségében elmondható, hogy a várakozásoknak megfelelően hibás értékeket kaptam mindenhol, de tudjuk mi az oka ennek az eltérésnek. Viszont az eredmények igazolták, hogy az algoritmusok és számítások megfelelően futnak le, mindezt elfogadható válaszidő melett.

A következő lépés arra irányul, hogy a nem route-olt tartományokat kiszűrjük a vizsgálatból, és csak azokat hagyjuk benne, ami mindkettőben route-olva van.

Irodalom, és csatlakozó dokumentumok jegyzéke

[1]. CIDR REPORT for 30 Nov 20. [Online] [Hivatkozva: 2020. november 30.] https://www.cidr-report.org/as2.0/#General_Status

[2.] Pagiamtzis, Kostas. Content-Addressable Memory Introduction. [Online] 2007. [Hivatkozva: 2020. november 30.] https://www.pagiamtzis.com/cam/camintro/

[3.] KIFÜ gerinchálózat HBONE. [Online] [Hivatkozva: 2020. nvember 30.] https://kifu.gov.hu/szolgaltatasok/ikt/halozati/hbone

[4.] University of Oregon Route Views Project. [Online] 2020. [Hivatkozva: 2020. november 30.] http://www.routeviews.org/routeviews/

[5.] ***Compressing IP Forwarding Tables:Towards Entropy Bounds and Beyond.*** **G. Rétvári, J. Tapolcai, A. Kőrösi, A. Majdán, and Z. Heszberger. ACM SIGCOMM 2013.**

[6.] Run-length_encoding. [Online] [Hivatkozva: 2020. november 30.] https://hu.wikipedia.org/wiki/Run-length_encoding

[7.] István, Fazekas. ***Valószínűségszámítás és statisztika.*** **2009.**

[8.] Szilvia, Nagy. ***Információelmélet.*** **2006.**

[9.] István, Dr. Vassányi. ***Információelmélet.*** **2003.**

[10.] Bertalan, Komenczi. ***Információelmélet.*** **Eger : 2011.**

[11.] iana.org. [Online] [Hivatkozva: 2020. november 30.] https://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml