



Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar

Baki Tamás

**IP FORGALOMTOVÁBBÍTÁSI
TÁBLÁK TÖMÖRÍTHETŐSÉGÉNEK
STATISZTIKAI VIZSGÁLATA**

KONZULENS

Dr. Rétvári Gábor

BUDAPEST, 2021

Tartalomjegyzék

Összefoglaló.....	5
Abstract.....	6
1 Bevezetés.....	7
1.1 Problémavezetés.....	8
1.2 Célkitűzés.....	11
1.3 A diplomaterv szerkezete.....	11
2 IP alapú útválasztás modellje.....	13
2.1 BGP általános felépítése.....	13
2.2 A RIB szerepe.....	16
2.3 A FIB szerepe.....	17
2.3.1 512K Day.....	19
2.4 Fogalmak.....	20
2.4.1 Specifikusabb prefix.....	20
2.4.2 Prefix fa.....	20
2.5 Matematikai definíciók.....	21
2.5.1 Tapasztalati Szórás.....	21
2.5.2 Korreláció.....	22
2.5.3 Kontingencia Táblázat és Peremeloszlás.....	24
2.5.4 Entrópia.....	24
2.5.5 Feltételes Entrópia és Kölcsönös Információ.....	25
2.6 Algoritmusok.....	27
2.6.1 Leaf-Push algoritmus.....	27
2.6.2 Futáshossz-kódolás.....	28
3 HBONE.....	29
3.1 Történeti áttekintés.....	29
3.2 HBONE IP hálózata.....	31
3.3 BGP Best External.....	34
4 Az útvonalválasztási és csomagtovábbítási táblák forrása.....	37
4.1 RIB állományok beszerzése.....	37
4.2 RIB formátum konverzió.....	38

4.3 FIB állományok.....	40
4.4 FIB-ek feldolgozása.....	40
4.5 Adatformázás, tisztítás.....	41
5 Prefix elemzés.....	42
5.1 Prefixek száma.....	42
5.2 Prefixek száma hosszonkénti bontásban.....	45
5.3 Specifikusabb prefixek száma.....	54
5.4 Specifikusabb prefixek száma hosszonkénti bontásban.....	57
5.5 Hirdetési tartomány.....	61
5.6 Szórások.....	63
5.7 Korrelációs elemzés.....	64
6 Next-hop elemzés.....	66
6.1 Leaf-Push megvalósítás.....	66
6.2 Futáshossz-kódolás.....	67
6.3 Kontingencia táblázatba rendezés.....	68
6.4 Entrópia elemzés.....	69
6.5 Feltételes Entrópia és Kölcsönös Információ.....	70
7 Összefoglalás.....	73
8 Irodalomjegyzék.....	75

HALLGATÓI NYILATKOZAT

Alulírott **Baki Tamás**, szigorló hallgató kijelentem, hogy ezt a diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző, cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy hitelesített felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2021. 05. 14.

.....
Baki Tamás

Összefoglaló

A Border Gateway Protocol (BGP) látja el az Internet hálózati tartományai közötti útvonalválasztást. A protokoll működését az egyes résztvevők gazdasági érdekei is befolyásolhatják, így nem tudható biztosággal a forgalom iránya. Az Internet gyors növekedése a rendszer üzemeltetői számára jelentős skálázási problémákat jelent, mert elfogy az eszközök fizikai kapacitása. A fő probléma a forgalomtovábbítási tábla mérete és ennek gyors növekedése.

Az elmúlt években egyre nagyobb szerepet kapott a forgalomtovábbítás vizsgálata, melyeket publikusan elérhető információk alapján végeznek. A publikus információk nem adnak teljesen pontos képet a valós Internet működéséről, így az ezek alapján végzett kutatások sem teljesen pontosak.

A diplomatervemben lehetőséget kaptam, hogy valós forgalomtovábbítási táblával dolgozhassak a HBONE (magyarországi akadémiai közösségi számítógép hálózata) jóvoltából. Arra szeretnék választ adni, hogy a publikusan elérhető útvonalválasztási táblák alapján mennyire jól közelíthető a valós működés, milyen mértékben lehet csomagtovábbítási táblák jellemzőire következtetni.

Az elemzésemben megvizsgáltam a forgalomtovábbítási táblákat külön, hogy egymáshoz mennyire hasonlóak az alap adatokban, majd hozzávettem a világ minden tájáról pár publikusan elérhető útvonalválasztási táblát és grafikonon ábrázoltam őket. A statisztikai jellemzőiket is megvizsgáltam a tartományokra és a továbbítási címekre, kibővítve időbeli vizsgálatokkal.

A diplomatervemben minden létrejött algoritmus és program publikusan elérhető a GitHub-on.

Abstract

The Border Gateway Protocol (BGP) is responsible for the routing between the network domains of the Internet. The routing decisions behind the protocol influenced by the economical interests of parties, therefore the route can not be known for certain. The rapid expanding of the Internet means a severe problem for the operators as the physical capacity of the devices is limited. The main problem that the size and the growing of the Forwarding Information Base.

In the past few years gain more and more attention the analysis of the packet forwarding, but these made on publicly available information. The public information not accurately reflect the real world operation, so the researches made on the can also be inaccurate.

I was granted the opportunity by the Hungarian Backbone (HBONE) to use their routing data. I want to give answer to what extent publicly available BGP routing table samples differ from real forwarding table samples from Internet routers and whether one can make assumptions regarding Internet scalability from the BGP traces.

In this Thesis Work I analysed the Forwarding Information Base to see what difference they hold to basic statistics then gathered routing informations all around the world and compare it against it. I used common statistical features on the range and the next hop to characterize them, then extended to time series analysis.

Every source code and algorithm I created in this work is publicly available on GitHub.

1 Bevezetés

A kommunikáció a minden napjai életünk szerves része. Az idő előrehaladtával egyre többen, egyre gyorsabban akarnak kommunikálni, nem csak a közvetlen környezettükkel, hanem a világ bármely részén élő rokonaikkal, barátaikkal vagy éppen munkatársaikkal. Ezekben a helyzetekben a kommunikációs csatorna nem más, mint az Internet. Többfélé minőségben beszélhetünk Internetes kommunikációról: lehet a saját helyi hálózatunk kommunikációja, vagy ennek az otthoni hálózatnak az internet szolgáltatóval való kommunikációja, de ide érhetjük a szolgáltatók közötti kommunikációt is.

Az Internetet éppen ezért gyakran szokás a „hálózatok hálózatának” hívni. Az elnevezése teljesen helytálló, mivel kezdetben és a mai napig is erről van szó. Az Amerikai Egyesült Államok kormánya az 1960-as években létrehozta a Fejlett Kutatási Projektek Ügynökségét (Advanced Research Projects Agency), az ARPA-t. 1969-re az ARPANET négy amerikai egyetemet foglal magában, név szerint az UCLA-t (University of California at Los Angeles), a Stanfordi Kutató Intézetet (Stanford Research Institute), az UCSB-t (University of California at Santa Barbara), és az University of Utah-t. [1] [2]

Az előzőekben említett négy egyetemi hálózatból mára 71 000 Autonóm Rendszer (Autonomous System – AS) lett. [3] Ezen hálózatok közötti útválasztást az erre a célra fejlesztett eszközök, a routerek végzik, különböző útválasztási protokollok segítségével. Speciális protokollüzenetekkel kommunikálnak egymással, ezáltal próbálnak egy mindig működőképest hálózatot fenntartani. Egy ilyen protokoll a Border Gateway Protocol (BGP).

Az AS-ek közötti útvonalválasztás jelentős részben inkább gazdasági alapon történik, egy-egy AS saját preferenciái alapján, nem pedig fizikailag a leggyorsabban, legrövidebb útvonalon vagy idő alatt valósul meg. Ezért az AS-ek közötti útvonalválasztási igényeket gyakran nevezük útvonalválasztási irányelveknek, vagy „policy routing”-nak.

A BGP tervezése során az azt végző mérnökök véig szem előtt tartották, hogy az illesfajta igények kielégíthetőek legyenek. Egy olyan protokollt próbáltak meg

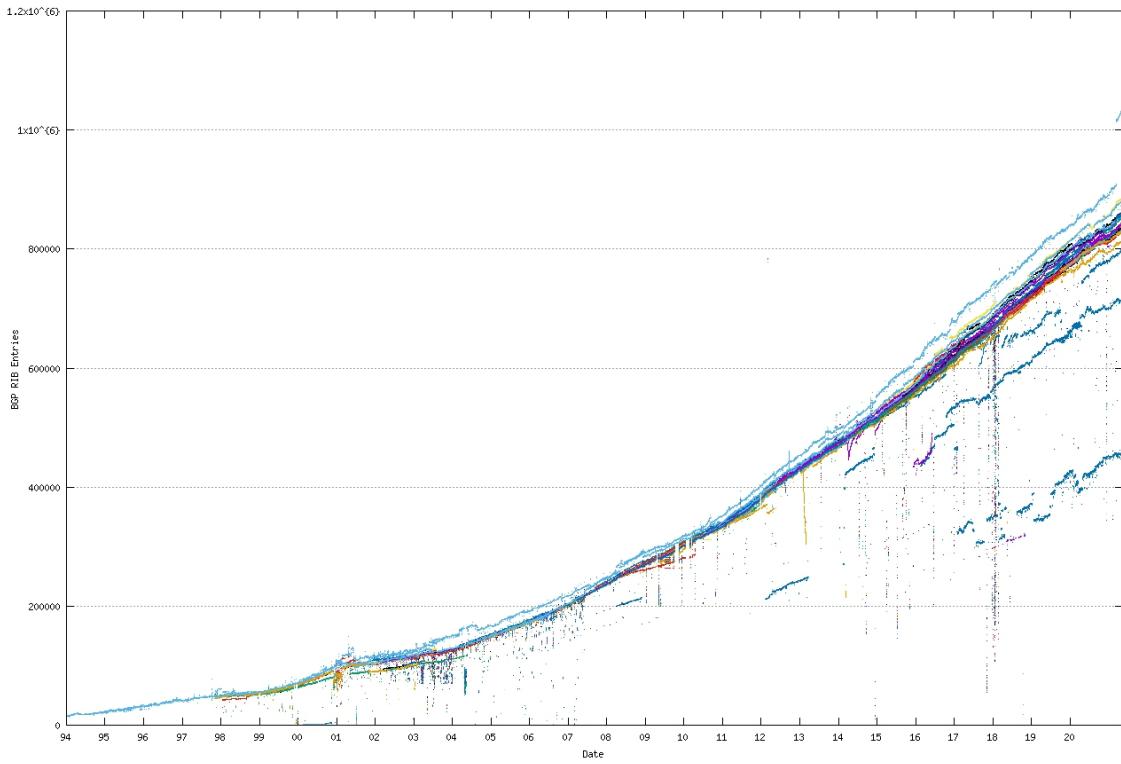
létrehozni, amely egy sokoldalú irányelv elfogadására képes. Így a BGP-t konfiguráló hálózati operátor egy kifejezetten nagy eszköztárú protokoll segítségével érvényesíteni tudja a szükséges útvonalválasztási preferenciákat.

A diplomatervemben az AS-ek közötti útvonalválasztás és ennek mérőszámai kerülnek előtérbe, oly módon, hogy a publikusan elérhető információkból hogyan és milyen mértékben lehet következtetni a BGP konkrét belső és alapvetően eltitkolt működésére. Mivel a BGP nagy szabadságfokú útvonalválasztást tesz lehetővé, ezzel az AS-eknek kedvezve, viszont a másik oldalról nézve annak a mérnöknek, aki hálózat elemzésével foglalkozik, erősen megnehezíti a dolgát, ugyanis ő több téren is nehézségekbe ütközik. Mégis nem ez okozza a legjelentősebb problémát, hanem az AS-ek titkolozása, mivel az AS-ekre nem jellemző, hogy nyilvánosságra hoznák, milyen irányelvek alapján kezelik a forgalmukat (mindezt teszik különböző titoktartási, gazdasági és vállalati döntések alapján), ennek folytán nehéz egy valós, jól használható képet kapni az Internet működéséről. Ugyanis pont, hogy az az alap hiányzik, amivel el lehetne kezdeni ennek készítését. [4]

Az egyetem és a tanszék jó kapcsolata miatt a KIFÜ HBONE (Kormányzati Informatikai Fejlesztési Ügynökség, Hungarian Backbone) rendelkezésünkre bocsátották a saját BGP routereik forgalomtovábbítási tábláit, ezáltal pontosan tudjuk, hogy milyen valós szabályok alapján történik a csomagtovábbítás. Valamint betekintést engedtek a belső működésre, amivel jobban értelmezni tudjuk a kapott adatokat. [5]

1.1 Problémafelvetés

Az elmúlt 15 év során a globális routing tábla méretek az alapértelmezett átjáró mentes zónában (Default Free Zone, DFZ) a lineárisnál jobban növekedtek. Az 2.3.1. ábra-n látható növekedést sikerült a „valós” (lásd 1.2 fejezet) saját adatokkal is reprodukálni az 5.1 fejezetben.



2.3.1. ábra A BGP bejegyzések növekedése [6]

Ennek a gyors növekedésnek több oka is van, melyek közül a néhány fontosabbat ki is fejtem. Ezek közül a legfontosabb ok az egyre több Internetre csatlakozó vállalat, és az egyre jobban elterjedő többhónáság (multihoming), vagyis egy végfelhasználói AS egyszerre több másik AS-től kap Internet hozzáférést. Ezáltal olyan új prefixeket kénytelen a szolgáltató AS meghirdetni, amelyek nem a saját címtartományukba esnek, ezzel egy-egy újabb bejegyzés keletkezik a globális routing táblákon. [7]

Másodsorban az IP deaggregáció is növeli a bejegyzések számát, amely az a folyamat, amikor egy nagyobb IP tartományt több kisebbre bontunk szét, és így az külön-külön kerül meghirdetésre.

Azok az új IP címek, akik újonnan kerültek használatba, tehát nem deaggregációval jöttek létre, az összes bejegyzés közel felét teszik ki. A multi-homed AS-ek az összes bejegyzés ötödét, a deaggregált prefixek pedig a negyedét teszi ki. [8]

A prefix deaggregációjának több célja is lehet: rendes üzemi működés mellett prefix hijack is megvalósítható, például 2008-ban a pakisztáni Telecom elterelte a Youtube forgalmát egy időre úgy, hogy a Youtube /22 prefixe helyett meghirdette sajátjaként a /24-es címeket az előfizetői számára. [9]

A gyors ütemű növekedés abból kifolyólag kellemetlen, mert az ISP-ek (Internet Service Provider) nem végeznek el hardverfrissítést nagy gyakorisággal, mert az nagyon költséges számukra és ez egy kiemelkedően kompetitív piac. Jelenleg a régi core routerek kapacitása szab határt az Internet növekedésének (lásd részletesen a 2.3 fejezetben). Ezen belül is egy speciális memória egység, ami sokáig 512 000 bejegyzést tudott kezelni. Ezt a Verizon amerikai telekommunikációs cég jóvoltából megtudtuk, hogy mi történik, ha átlépjük (lásd részletesen a 2.3.1 fejezetben).

Sajnos kényetlenek vagyunk ezt a hatalmas táblát minden routerben betöltve tartani, máskülönben az ezzel járó kommunikációs késleltetés miatt ellehetetlenedne a jelenlegi teljesítmény fenntarthatósága. A továbbítási elvárások miatt a hierarchikus vagy elosztott routing táblák nem megvalósíthatóak. Ezenfelül mivel a csomagtotvállalás pontról-pontra (hop-by-hop) történik, ebből rögtön következik, hogy szükséges a routing táblát mindegyik routerben helyileg eltárolni. [10]

Az előbbi problémákat elkerülendő, sokféle FIB (Forwarding Information Base, csomagtotvállalási tábla) tömörítési eljárást hoztak létre. De a tömörítési eljárásokat csak úgy lehet használni, ha az AS egyik routerére a saját routereitől és a többi AS-től függetlenül is implementálni lehet, ezáltal csak a memória problémákkal érintett routerekre külön is használható. Ezeknek az eljárásoknak együtt kell tudniuk működni a meglévő rendszerekkel átlátszó módon. Jelenleg is folynak az egyetemen kutatások ebben a témaában. [11]

Egyik ilyen módszer az „Aggregation with Increasing Scopes” (AIS), ahol a prefix aggregáció az egyre bővülő körben történik a helyi prefixektől, az AS-en belüli prefixekig és a globális szintig. [7]

Az elmúlt évtizedben sok hasonló témájú kutatás készült, ezek mindegyike a publikusan elérhető útvonalválasztási táblák alapján szűrte le a következtetéseket. Ezek a kutatások nem teljesen pontosak, ugyanis az útválasztás nem csak az előbbi táblák alapján zajlik. A FIB a routing táblának azokat a bejegyzéseit tartalmazza, amit a BGP kezelő által létrehozott szabályok alapján a protokoll kiválaszt (lásd részletesen a 2.2 fejezetben).

1.2 Célkitűzés

Az előző fejezetben felvetett problémák fényében azt tüztük ki célnak, hogy megválaszoljuk azt a kérdést, hogy az Interneten elérhető publikus RouteViews információk alapján mennyire és milyen mértékben lehet a BGP valós működésébe belelátni. Abban a kivételes helyzetben vagyunk, hogy az BME jó viszonyt ápol HBONE-al (Hungarian Backbone), akik az együtt működés keretében rendelkezésünkre bocsátották a saját BGP routereik forgalom irányítási tábláit az elmúlt több mint hét évben, ezáltal pontosan tudjuk, hogy milyen valós szabályok alapján történik és történt a csomagtovábbítás.

Az elemzést olyan módon kívánom elérni, hogy vettetem pár publikusan elérhető teljes BGP útvonalválasztási táblát és a HBONE-tól kapott csomagtovábbítási táblákat, majd kiválasztottunk néhány olyan jól megfigyelhető és érdekesnek tűnő metrikát, amit a továbbiakban egymáshoz lehet hasonlítani. Jelenleg az elemzéseket csak az IPv4 címeken végzem el, mivel skálázhatósági szempontból jelenleg ez a meghatározó. De az elkészült metodológia és program minimális módosítással IPv6-ra is át ültethető lesz.

Az első rész célja az IP tartományokon egy átfogó statisztikai alap létrehozása, egy olyan metodológia kidolgozása és program készítése, ami könnyen használható és új adatokkal, metrikákkal könnyen bővíthető legyen.

A második részben viszont az AS-ek közötti kapcsolat kerül megvizsgálásra, itt a next-hop (következő pont) információkból kinyert statisztikák lesznek bemutatva.

A vizsgált időszak 2013 novemberétől 2020 végéig terjed. Ebben a 2615 napból álló intervallumban végzem el az összehasonlító elemzéseket.

1.3 A diplomaterv szerkezete

A diplomatervben a RIB és FIB feldolgozását, adatkonverzióját, összehasonlító elemzését és a kapott eredmények vizualizációját fogom elvégezni. A vizualizáció és a köztes információk segítségével összegző elemzést fogok adni a dolgozat fő kérdéseire.

A második fejezetben bemutatom a teljes átláthatóság érdekében a BGP részletes működési elvét, amelynek keretében kitérek a fontos táblákra és ezek tartalmára, felépülési módjára. A továbbiakban a szükséges egyéb hálózati funkciókat ismertetem, illetve minden egyéb háttér tudást, ami szükséges a teljes megértéshez.

Valamint definiálom azokat a fogalmakat és bemutatom az algoritmusokat, amiket a későbbiekben használni fogok.

A harmadik fejezetben mutatom be a HBONE-t. Az alapításától kezdve egészen napjainkig egy rövid történeti áttekintést adok. majd átérek felépítésére és kapcsolataira. Végül ismertem a megértéshez szükséges technológiai hátteret.

A negyedik fejezetben részletesen leírom az adatfeldolgozás módjának menetét, kitérek a különböző adatforrásokra és az innen vett adatok beszerzésének módjára. Az adatkonverziókra, köztes állapotokra és az adattisztítás módjára és mennyiségrére.

Az ötödik fejezetben ismertem kiválasztott mérő metrikákat, melyek mentén az összehasonlítást elvégzem. Itt részletesen leírom a metrikák előállítását és ezek jelentőségét majd bemutatom a kapott eredményt. minden eredményt diagramokon ábrázoltam és ezekhez megjegyzéseket, észrevételeket és érdekességeket írtam.

A hatodik fejezetben a mélyebb statisztikai jellemzőket mutatom be. Valamint itt kaptak helyet a next-hop elemzések.

Végezetül, a hetedik fejezetben összegzem az elvégzett munkát, választ adok a szakdolgozat fő kérdéseire. Értékelem a kapott eredményeket és ismertem a további tervezeteket.

2 IP alapú útválasztás modellje

Ebben a fejezetben a BGP részletes felépítését, működési elvét, valamint ennek elemeit mutatom be, külön hangsúlyt fektetve a RIB és FIB táblákra. Itt kaptak még helyet a szükséges háttér információk és alapvető definíciók meghatározása is. A fejezet zárásaként pedig bemutatom a HBONE IP hálózatának felépítését és ismertetem a FIB-ek elemzésének szempontjából vett fontos tulajdonságait.

2.1 BGP általános felépítése

A BGP (Border Gateway Protocol) egy AS-ek (Autonomous System, autonóm rendszer) közötti forgalomirányító protokoll. Az AS egy olyan hálózati csoport, amely egy azonos közös forgalomirányítási és adminisztrációs szabályt valósít meg önállóan. Az AS belsejében természetesen vannak egyéb belső forgalomirányító protokollok, ilyen például az OSPF (Open Shortest Path First), RIP (Routing Information Protocol) vagy az EIGRP (Enhanced Interior Gateway Routing Protocol).

A BGP viszont egy külső forgalomirányító protokoll, mely policy routing-ot (útvonalválasztási irányelv) valósít meg, tehát figyelembe veszi az útvonalak fizikai jellemzőit, de végső soron az aktuális AS üzleti politikája lesz a döntő szempont. Az AS-ek közötti kapcsolat egy nagyon kompetitív piac: egy AS bármikor dönthet úgy, ha a piaci vagy politikai érdeke szolgálja, hogy eltereli vagy blokkolja egy másik AS forgalmát.

A tartományok közötti útválasztásnak kettő alapvető formája van BGP-t használva. Az első ilyen a *tranzit* szolgáltatás, amikor az AS egy nála nagyobb AS-től kapcsolatot vesz. Ebben az esetben a nagyobb AS-t hívjuk szolgáltatónak (provider), a másikat pedig előfizető (customer) AS-nek. A szolgáltató vállalja az előfizetője felé, hogy minden csomagját eljuttatja bármely, az Internetre csatlakozott hoszt számára, valamint minden, az előfizetőnek címzett csomagot eljuttatja neki. Az előfizető pedig ezért a szolgáltatásért forgalomfüggő díjat fizet.

A tranzit szolgáltató meghirdeti a hálózaton egy BGP hirdetéssel az előfizető IP tartományát. A külső címtartományokra vonatkozó hirdetéseket átadja az előfizetőnek, majd folyamatosan továbbítja az „előfizető ↔ Internet” forgalmát.

A másik féle útvonalválasztási forma a *peer* vagy *settlement-free interconnection* (SFI, költség nélküli kapcsolat) AS-AS kapcsolat, ezt két közel egyforma méretű AS szokta használni a leggyakrabban, akik ugyanannak az AS-nek az előfizetői. Ekkor ez a két AS ahelyett, hogy a közös szolgáltatójukat használnák kapocsnak, kiépítenek egy közvetlen kapcsolatot egymás között, ahol kicserélik egymás és saját előfizetőik forgalmát. Ezáltal mindenket költséget takarítanak meg és nem utolsó sorban csökken a kommunikációban résztvevő felek száma. [4] [12]

Tier 1-es az az AS, aminek nincsen szolgáltatója, vagyis kizárolag peering használatával biztosítja a teljes Internetelérést az előfizetői számára. Nincs természetesen egy hatóság, ami megmondja, hogy ki Tier 1 és ki nem. Egy népszerű definíció szerint: A Tier 1-es hálózat egy olyan hálózat, ami úgy éri el az összes többi hálózatot az Interneten, hogy ő maga nem vesz tranzit útvonalakat vagy fizet a peeringért. [13]

A forgalomtovábbítás alapvető szabálya, hogy a forgalom a „cash-flow” irányába haladjon, melyet mindenki igyekszik betartani, hisz ez a gazdasági érdeke. minden AS igyekszik azt elkerülni, hogy a peer-jei és a provider között zajló forgalom rajta keresztül menjen. Ezt valley-free (völgy mentes) routingnak nevezzük, ami abból áll, hogy ha egy csomagot több irányba is küldhetünk, akkor minden előfizetőnk felé próbáljuk küldeni. Abban az esetben viszont, ha arra mégsem lehet, akkor a peer felé, és csak végső esetben a szolgáltató felé. Ezt prefer customer szabálynak hívjuk.

A valley-free routing a nevét onnan kapta, hogyha felül képzeljük el a Tier 1 szolgáltatókat és legalul a végfelhasználói AS-eket, akkor bármely két AS között csak egy fel és egy lefelé menő út lehet, vagyis az út vizuális képében nem lehet völgy.

Ennek elérési módja, ha a szomszédjaink felé csak azokat az útvonalakat hirdetjük meg, amit szeretnénk, hogy használjanak. Ennek konkrét megvalósítása, ha peer-rel szemben a peer-peer és peer-szolgáltató utat tiltjuk és csak a peer-előfizető utat engedélyezzük. A szolgáltató felé hirdetett út még egyel szigorúbb, mivel neki csak az előfizetőink felé menő utakat hirdetjük meg. Természetesen az előfizetőinknek minden utat meghirdetünk, mert nekik mi szolgáltatjuk a globális hálózat elérését, amiért fizetnek nekünk, ezért feléjük nincs szűrés.

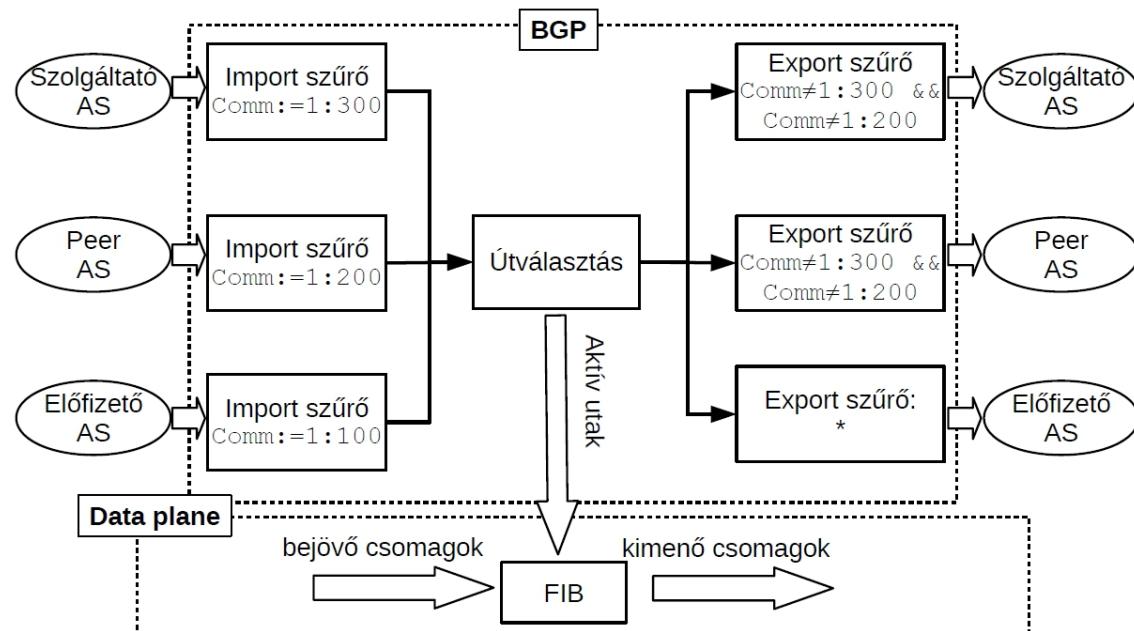
A valley-free routing konkrét megvalósítása a BGP-ben úgy néz ki, hogy import szűrőkkel felcímkezzük a bejövő BGP hirdetést, ez lesz a BGP COMMUNITY attribútum, ezáltal a hirdetésről bármikor el tudjuk dönteneni, hogy kitől kaptuk. [14]

Egy példa import címkézés lehet például:

- előfizetőtől kapott hirdetés 1:100
- peer-től kapott hirdetés 1:200
- szolgáltatótól kapott hirdetés 1:300
- saját hirdetést szűrés nélkül exportálunk

Az export szűrők pedig a community érték alapján szűrnak:

- peer és szolgáltató felé csak a 1:100-as címkéjű és a saját hirdetéseket küldjük ki,
- előfizető felé pedig az összes hirdetést átengedjük



2.3.1. ábra BGP community szűrője [4]

A 2.3.1. ábra-n az előbb említett szűrőbeállítást láthatjuk képen ábrázolva. Természetesen ettől jóval összetettebb community szűrőket is lehet a BGP felett definiálni, ha szükség van rá, de alapvetően a fent említett szabályok mindenkor érvényesek.

2.2 A RIB szerepe

A BGP routerek a szomszédjaiktól üzenetekben kapott útvonalakat úgynevezett útvonalválasztási táblákban, RIB-ekben (Routing Information Base) tárolják. A RIB a routerek control plane-jében (vezérlési sík) van, ebben a táblában vannak összegyűjtve az információk a szomszédos és a nem szomszédos AS-ekről.

A táblából különféle információk kiolvashatóak: a preferencia értékek, a community értékek és AS-PATH-ok, vagyis, hogy konkrétan mely AS-eken menne át a csomag, ha arra küldjük. Ha a router egy olyan route-ot kap meghirdetve, aminek az AS-PATH-jában már szerepel, azt természetesen eldobja, mintegy ezzel biztosítva a hurokmentességet.

Az útvonalválasztási tábla segítségével képezhető le a csomagtovábbítási tábla (FIB, Forwarding Information Base), ebben a táblában még az összes megkapott út szerepel, ebből kerül kiválasztásra a legjobb (amelyik az üzleti igényeknek megfelel és azon belül a legjobb) bejegyzés, amelyik alapján majd a csomagtovábbítás történik.

Ennek a kiválasztásnak a lépései sorrendben: [15]

1. Ha ez az egyetlen útvonal a cél felé, akkor minden további nélkül beszúrjuk a FIB-be.
2. LOCAL PREFERENCE érték alapján állítsuk csökkenő sorba.
3. Vegyük a legrövidebb AS-PATH-al rendelkező útvonalakat.
4. Vegyük a legalacsonyabb ORIGIN-ID-vel rendelkezőt. Vagyis, hogy melyik útvonal melyik protokolltól került be a táblába.
5. Vegyük a legalacsonyabb Multi-Exit-Discriminator (MED) attribútummal rendelkező útvonalakat.
6. Vegyük az eBGP útvonalakat előbb, mint az iBGP útvonalakat.
7. Vegyük a NEXT-HOP routerig legalacsonyabb útköltségű útvonalakat.
8. Vegyük a legalacsonyabb BGP-ID-val rendelkező útvonalat.

Fontos látni, hogy milyen sok minden műlik egy útvonalnak a FIB-be kerülése, amiből rögtön látható, hogy ez nem triviális értékelési folyamat, és az is jól látható, hogy ezt valós időben nagyon sokáig tartana eldönten minden bejövő csomagra. Érdekes még, hogy a 7. lépés fizikai paraméter, a 2. lépés pedig az AS saját érdeke.

A RIB-ben továbbá egyéb útvonalak is lehetnek, amiket a router különböző IGP (Internal Gateway Protocol) protokolloktól tanult meg. Az egyik ilyen elterjedt protokoll az OSPF (Open-Shortest Path First). Az OSPF párhuzamosan fut a BGP-vel, csak míg a BGP az AS-ek közötti utakkal foglalkozik, addig az OSPF az AS-en belüliekkel. Az AS belső routerei az OSPF segítségével kapják meg, hogy melyik AS „szélén” lévő routernek küldje a csomagot.

A RIB a control plane-ben kapott helyet és néhányszor akkora lehet, mint a valós csomagtovábbítási tábla, valamint általános célú memóriában van eltárolva. Ebből az okból kifolyólag elég lassú a RIB-ben való keresés (ahhoz legalább is biztosan, hogy a mai körülmények mellett csomagtovábbításra használni lehessen).

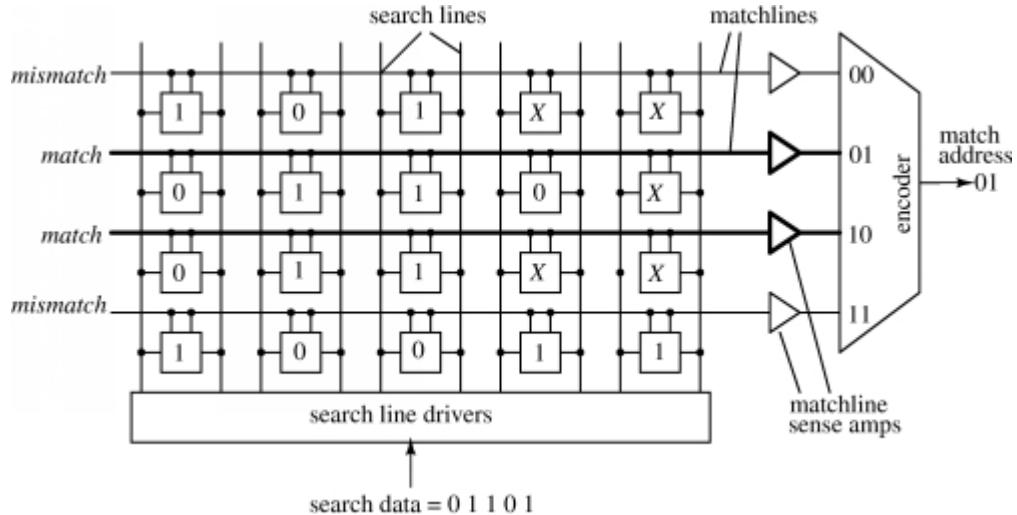
2.3 A FIB szerepe

A FIB (Forwarding Information Base), mely alapján a router a konkrét csomagtovábbítást végzi, ebben a táblázatban már csak a cél prefix szerepel és a next-hop router címe. A FIB-ben csak azok a prefix és next-hop párosok vannak, amelyek kielégítik az AS-ek minden kívánalmát. Itt már mindenkor a legspecifikusabb bejegyzést keressük, így a műveletnek nagyon gyorsnak kell lennie. Ennek megfelelően speciális hardware elemekkel vannak bizonyos funkciók megvalósítva, mint a FIB-ben való keresés.

Nem csak megtalálni kell egy illeszkedőt, hanem abból a legtovább illeszkedőt kell, vagyis a LPM (Longest Prefix Match), ennek a megvalósítása egyáltalán nem triviális. $O(n)$ a keresés komplexitása, ahol n a bejegyzések száma, de nagyságrendileg 820 000 bejegyzés van. Vegyük egy gyakorlati példát: 500 bájtos csomagokkal számolva egy 1Gbit/s kapcsolaton megérkezik 250 000 csomag egy másodperc alatt, a routernek van mondjuk 1 Ghz-es órajele. Ebből megkapható, hogy 4 000 órajel jut egy csomag feldolgozására, ami közelében sincs a 820 000 bejegyzésnek, amit minden egyes csomagnál át kell nézni ahhoz, hogy eldöntsük, merre kell tovább küldeni a csomagot.

A probléma megoldására egy tartalom címzhető memória (Content Addressable Memory, CAM) cél hardver van belerakva a routerekbe, ami kizártlag ezt az egy műveletet végzi. Működési elve pont a hagyományos memóriákkal ellentétes, vagyis megadunk neki egy adatot és a CAM visszaadja a memória címét (több találat esetén az elsőt) ahol a keresett adat található. Ennek kell egy speciális változata, amiben van egy

„don’t care” bit (x), így könnyű prefixet keresni benne. Ugyanis a prefixnek csak az alhálózati maszk által meghatározott első néhány eleme van megadva fixen, a többi már nem érdekes.



2.3.1. ábra TCAM egyszerűsített logikai ábrája [16]

A 2.3.1. ábra-n egy egyszerűsített 4 darab 5 hosszú szót tárolni képes háromállapotú CAM (Ternary CAM) logikai diagramja látható. Mindegyik blokkjában 0/1/x van tárolva és egy komparátor. A keresést függőlegesen nézi, vagyis annyi összehasonlítást végez el egyszerre, amennyi sora van, tehát a keresés ideje már csak a szó hosszától függ, ami IP címek esetén konstans: 32. A keresés végén egy kódoló összegzi az eredményt és a legkisebb címűt adja válaszul.

Problémát jelent ezáltal a prefixek sorrendje a TCAM-on belül, mert nem átgondolt sorrend esetén nem biztos, hogy a leghosszabb illeszkedőt fogom megkapni. Tehát prefix hossz szerint csökkenő sorrendben kell bejegyezni a prefixeket.

Miután a TCAM kiadta, hogy melyik memória címen van az a prefix, ami leghosszabb illeszkedik, szükséges még egy RAM is a FIB tárolására, amiben a next-hop címek vannak eltárolva oly módon, hogy az összetartozó cím-prefix párosnak ugyanaz a memória címe a TCAM-ben illetve a RAM-ban.

A FIB tábla ezáltal nagyon gyorsan képes a prefixekhez tartozó next-hop címeket megtalálni, de ennek jelentős ára van. Egyrészről a TCAM memória nagyon bonyolult, 16 tranzisztor/cella (SRAM: 6 DRAM:2 tranzisztor/cella), emiatt magas az előállítási költsége. Másrészről magas a fogyasztása, ebből kifolyólag nagyon gyorsan felmelegszik. A TCAM ezen okok miatt limitált erőforrás egy routerben, ugyanis véges

mennyiséggű hely van benne, 1024 000 bejegyzés fér bele a legtöbb gyártó routerébe, ami jelenleg is használatban van. Mint említettem, 820 000 körüli bejegyzésnek kell jelenleg beleférni a memóriába, a kapacitás fele az IPv6 számára van lefoglalva, ezzel 512 000 helyet hagyva az IPv4-es címeknek.

2.3.1 512K Day

Alapvetően jónak tartjuk az Internet növekedését, ám ennek gátat szabott a core BGP router-ekben lévő, fentebb említett memóriának a mérete. 512 000 IPv4 útvonal volt a limit. Régóta tudni lehetett és számítottak is rá, hogy előbb-utóbb elérik azt a számot, csak hamarabb történt meg, mint várták. Az egyik AS 2014 augusztus 12-én úgy döntött, hogy felszabdalja az egyik prefixét több kisebb tartományra és ezzel átlépték az 512 000-res korlátot. Egyrészt a várakozásoknak megfelelően részben használhatatlanná vált az Internet, mert egyes lekérések annyira lassúvá váltak, hogy egyszerűen eldobásra kerültek ezek a csomagok, így sok tranzakció szakadt meg, vagy volt lassú.

Másrészről viszont olyan hibák is előfordultak, hogy a frissen meghirdetett prefixek nem kerültek be a FIB-be. Ha mondjuk a Google pont ekkor frissítette a Youtube prefixét, akkor azok sem fértek vissza FIB-be. Tehát, így nagyon nehezen értelmezhető hibákat okozott.

Ezt a napot nevezik „512K Day”-nek, amely nem lett annyira felkapott, mint az Y2K. A kimaradások és zavarok főleg az amerikai térségben voltak tapasztalhatóak, több nagy Internetszolgáltatónál is érezhető volt. [17]

2.4 Fogalmak

2.4.1 Specifikusabb prefix

Egy prefixet akkor nevezünk egy másiknál specifikusabbraknak, ha az a prefix hosszabb, mint egy másik prefix és a közös szakaszon megegyeznek.

A:10.0.0.0	/8	bin:00001010
B:10.128.0.0	/9	bin:00001010 1
C:10.1.0.0	/16	bin:00001010 00000001
D:10.129.0.0	/16	bin:00001010 10000001

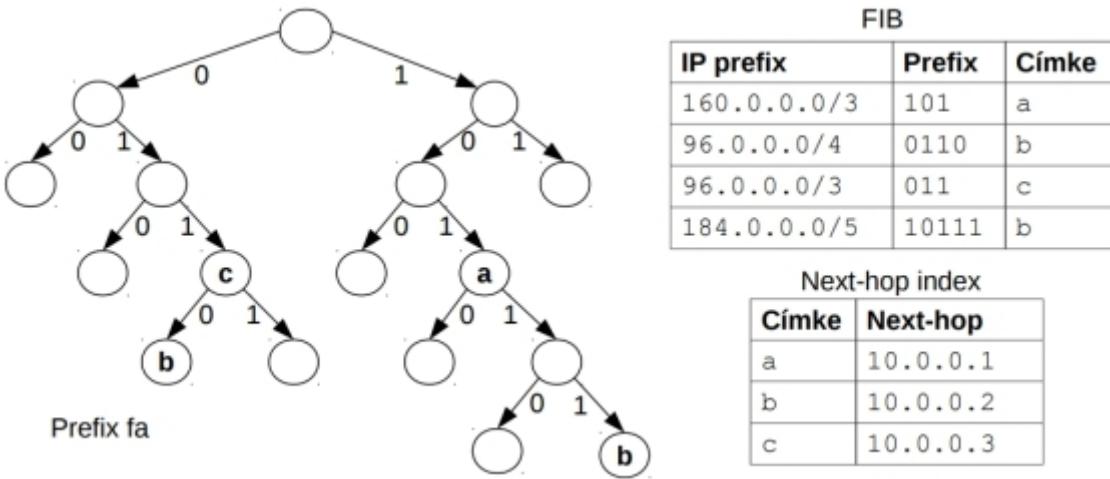
2.4.1 kódrészlet Példa specifikusabb prefixre

A 2.4.1 kódrészleten az látszik, hogy az „A” prefixnél specifikusabb minden a három másik prefix, mert az első 8 biten mindegyik ugyan azt az értéket veszi fel, ahogy a bináris kifejtésen látható. A „B” prefixnél viszont csak a „D” prefix lesz specifikusabb, mert az első 9 bitjük megegyezik. A „C” ugyan hosszabb, de bináris kifejtésén látszik, hogy a 9. bitje 0 értékű, míg a „B” prefixnek 1.

2.4.2 Prefix fa

A prefix fa célja, hogy a TCAM-ot ugyan nem megközelítő, de gyors sebességet érjünk el általános célú hardveren.

A prefix fa nem sokban különbözik bármelyik bináris fától, minden elemnek két gyereke van, tipikusan a balra eső a „0” és jobbra eső az „1” címkéjű élen van, így, ha az élek mentén összeolvassuk a címkéket a gyökértől a levélig, akkor megkapjuk a prefixet és a levélbe meg rögtön a next-hop lehet beírva, vagy egy arra utaló cím. Egy ilyen adatszerkezet csak prefix hossznyi összehasonlítást végez, ami jelentős javulást okoz egy lineáris kereséshez képest.



2.4.2. ábra Példa prefix fa [4]

Ha végig követjük a 2.4.2. ábra példáján, akkor láthatjuk, hogy a „bal-jobb-jobb-jobb” lépésekkel elérünk a „c” címkéjű csomóponthoz. A FIB-ben láthatjuk, hogy a „011” bináris alakú prefix tartozik hozzá, ami pont az út élcímkéjét adja ki összeolvasva.

2.5 Matematikai definíciók

2.5.1 Tapasztalati Szórás

A szórás tulajdonképpen a sokaság egyes értékeinek a számtani átlagától vett eltéréseinek négyzetes átlaga, vagyis, hogy az egyes értékek mennyivel térnek el az átlagtól a teljes sokaságra vetítve. Tapasztalati szórás:

$$\sigma = \sqrt{\frac{\sum_{i=1}^N (x_i - \bar{x})^2}{N}}$$

2.5.1. egyenlet Szórás képlete

Ahol:

N – az x adathalmaz elemeinek száma

x_i – az x adathalmaz i -edik eleme

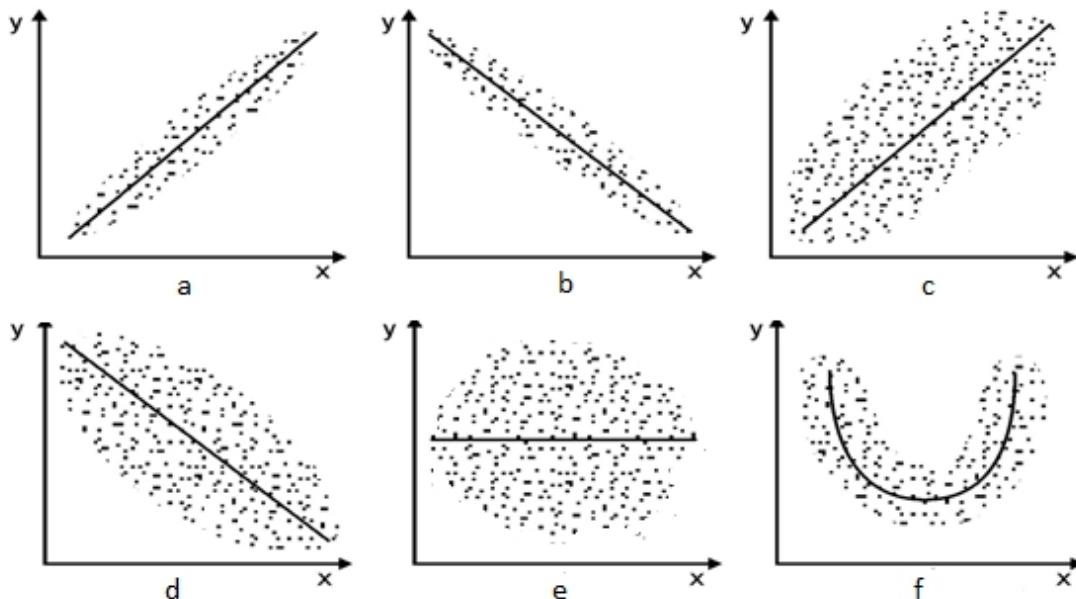
\bar{x} – az x adathalmaz elemeinek átlaga

2.5.2 Korreláció

Köznyelvi értelemben kölcsönös viszonyt, egymástól való függést jelent. Éppen ez utóbbi definíció adja a legtöbb félreértést a matematikai vagy statisztikai közegben használt korreláció fogalmával szemben. Amikor ugyanis tudományos szövegben találkozunk a korreláció szóval, ott szó sincs ok-okozati viszonyról. Ebből kifolyólag a korreláció szimmetrikus fogalom, vagyis megfordítható.

Tehát a korreláció pusztán a két adathalmaz közötti kapcsolatot írja le és megadja ezeknek a nagyságát és irányát is. Általános statisztikai jellemzés során megadja, hogy a két vizsgált érték nem független egymástól, vagyis hogy mennyire függ az egyik érték a másik értéktől.

Két adathalmaz sokféleképpen függhet egymástól. Lehet közöttük pozitív korreláció: ez azt jelenti, hogy ahogy az egyik nő, a másik érték is arányosan növekszik (2.5.1. ábra a, c rész). Lehet közöttük lineárisan negatív korreláció, vagyis ahogy az egyik érték nő a másik arányosan csökken (2.5.1. ábra b, d rész). Előfordulhat, hogy függetlenek egymástól (2.5.1. ábra e rész). Sőt, akár olyan is létezhet, hogy valamelyen függvényvel leírható kapcsolat van közöttük (2.5.1. ábra f rész).



2.5.1. ábra Korrelációs fajták [18]

Két adathalmaz kapcsolatát pont-diagram segítségével szokták ábrázolni. Ehhez valamelyen közös érték alapján párosítani kell őket. Mivel a két adathalmaz elemeit

párba állítjuk, ezért egyforma elemszámuk kell, hogy legyen. Ez előfeltétele a korreláció számításának.

A korreláció mértékét a korrelációs együtthatóval szoktuk jellemezni. A korrelációs együttható egy -1 és +1 közötti szám, melynek értéke jelzi a kapcsolat irányát és erősségét. [18]

- Ha a korrelációs együttható érteke +1-hez közel, akkor a két adathalmaz között erős pozitív kapcsolat van. Tehát, ahogy az egyik adathalmaz értéke változik, ugyan úgy változik hozzá arányosan a másik adathalmaz értéke is.
- Ha a korrelációs együttható érteke -1-hez közel, akkor a két adathalmaz között erős negatív kapcsolat van. Tehát, ahogy az egyik adathalmaz értéke változik, a másik adathalmaz értéke ellentétesen változik, de továbbra is hozzá arányosan.
- Ha a korrelációs együttható érteke 0-hoz közelít, akkor a két adathalmaz elemei között nincs kapcsolat.

A tapasztalati korreláció számításának képlete:

$$r_{xy} = \frac{1}{n-1} \sum_{i=1}^n \frac{(x_i - \bar{x})(y_i - \bar{y})}{s_x s_y}$$

2.5.2. egyenlet Korreláció képlete

Ahol:

r_{xy} – a korrelációs együttható

n – az x és y adathalmaz elemeinek száma

x_i – az x adathalmaz i-edik eleme

y_i – az y adathalmaz i-edik eleme

\bar{x} – az x adathalmaz elemeinek átlaga

\bar{y} – az y adathalmaz elemeinek átlaga

s_x – az x adathalmaz szórása

s_y – az y adathalmaz szórása

2.5.3 Kontingencia Táblázat és Peremeloszlás

A kontingencia táblázat a statisztikában egy olyan, mátrix formájú táblázat, amely a változók gyakorisági eloszlását mutatja. Az 2.5.1. 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

2.5.1. táblázat Példa Kontingencia táblázat

A soronként és oszloponként összesített gyakoriságokat nevezzük peremeloszlásnak. A peremeloszlás több valószínűségi változó közös eloszlásának jellemzője. Tehát a peremeloszlások a kontingencia táblázat peremén szereplő eloszlások. [19]

2.5.4 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.

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

2.5.3. egyenlet Entrópia

Ahol:

$H = \{h_1, h_2, h_3, \dots\}$ – a forrás jelkészlete

$P = \{p_1, p_2, p_3, \dots\}$ – 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 minden ugyanazt a jelet sugározza: ekkor a p_i valószínűségek egyike 1, a többi 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. [20]

2.5.5 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.

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

2.5.4. egyenlet Kölcsönös entrópia

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

2.5.5. egyenlet 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 átlaga.

- 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”.

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

$$I(B,A) = I(A,B)$$

2.5.6. egyenlet 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.

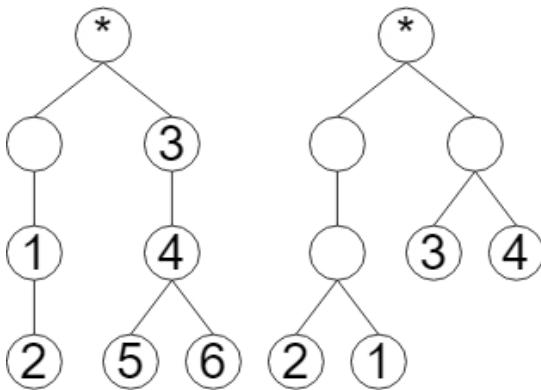
Ha viszont a B az A által **teljesen meghatározott**, akkor $H(B|A) = 0$, azaz $I(B,A) = H(B)$. [20] [21]

2.6 Algoritmusok

2.6.1 Leaf-Push algoritmus

Az 2.6.2. ábra Leaf-Push algoritmus példa 2.6.2. ábra-n látszó alakból egy prefix fát kell generálni. Generálás úgy zajlik, hogy a prefix oktális alakját át kell alakítani egy bináris karakterszorozattá. A karakterszorozaton végiglépelve kell építeni a fát, ha „0” jön, akkor a csomópont „bal” gyereke lesz, ha pedig „1”-es jön, akkor az adott csomópont „jobb” gyereke lesz. Az úton végigmenve 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.



1.0.0.0/24	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

2.6.1 FIB adatforrás példa

2.6.2. ábra Leaf-Push algoritmus példa

Az 2.6.2. ábra-n a baloldali a kiinduló fa, amiben az 1,3,4 –el jelölt csomópontok, azok a belső csomópontok, amik információt hordoznak. Ezeknek a csomópontoknak szeretnénk kivinni az információtartalmát a levelekbe.

Az algoritmus két nagy lépésből áll. Az első, hogy végigmegyünk az összes csomópontron, 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.

A második lépés valójában egy egyszerűsítés: a célja, hogy csökkentse a csomópontok számát. Ha két gyerek ugyanazt az információt hordozza, akkor egyesíteni lehet őket a szülő csomópontjukban. A 2.6.2. ábra 5,6-al jelölt csomópontokról tegyük

fel, hogy ugyan azt az információt hordozzák, akkor vissza lehet őket emelni a szülő csomópontjukba, ami a 4-es jelű lesz a jobb oldali fában. [11]

2.6.2 Futáshossz-kódolás

A futáshossz-kódolás egy nagyon elterjedt veszteségmentes tömörítési eljárás, melynek segítségével a hosszan 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 2.6.3 példát, ahol eltároljuk egy „H” betű képét.

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

2.6.3 példa Futáshossz-kódolás

Ezt a tömörítési eljárást könnyen lehet használni prefixek a hozzájuk tartozó next-hopok esetén. A prefix egy mennyiséget határozza meg, hogy hány címen át ismétlődik ugyanaz a next-hop.

3 HBONE

Egy rövid történeti áttekintést szeretnék adni, ami elhelyezi a világban a HBONE-t, így ebből kiderül, hogy honnan indult és hova jutott. A történelem után pedig bemutatom a jelenlegi IP hálózatot kiemelve azokat a részeket, amelyek az elemzések szempontjából fontosak.

3.1 Történeti áttekintés

1990. október: Ekkor jön létre az első bérelt vonalas kapcsolat az európai kutató hálózatokhoz az IIF SZTAKI (Információs Infrastruktúra Fejlesztési Program Számítástechnikai és Automatizálási Kutatóintézet) és a Linzi Egyetem között. Egy évvel később a sztaki.hu lesz az első magyarországi névszerver. 1991. októberétől Magyarország hivatalosan is „fent van” az Interneten.

1993. július 8. Létrejön a kapcsolat a BME, a KFKI (MTA Központi Fizikai Kutatóintézet) és az IIF Központ között (beleértve a SZTAKI-t). Ezzel létrejön a hazai kutatói Internet gerinchálózata, a HBONE. Ekkor a felhasználók már szinte minden mai felhasználási módot tudtak használni, ugyanúgy volt elektronikus levelezés, távoli fájlátvitel és web böngészés is. Természetesen a maihoz képest erősen limitált sávszélességgel.

Már ekkor, közel 30 éve is gondot okozott az elfogyó IPv4 tartomány, amit a NAT és az IPv6 segítségével részlegesen megoldottak. A routing tábla mérete a nagy csomóponti routerekben szintén problémát okozott és ma is ugyanazzal a problémával nézünk szemben, csak egy jóval nagyobb léptéken.

1996. februárjában kiépül a BIX (Budapest Internet Exchange). Először a Matáv telepén, majd két évvel később kerül a mai helyére, a Viktor Hugo utcába, innentől az ISZT (Internetszolgáltatók Tanácsa) működteti.

1997. május 5. Elkészül és üzembe áll a 30 Mbps sebességű maghálózat az ekkor már NIIF központ, Matáv központ és BME háromszögben. Az ezt követő két évben növelik a hazai, főleg vidéki (1 Mbps) és a nemzetközi Internetkapcsolatok (34 Mbps) sebességét. Ekkora valamennyi megyeszékhelyen létrejöttek a regionális

központok és hozzávetőleg minden 100. magyarnak volt Internethez való hozzáférése. [22]

A következő néhány évben minden megtesznek, hogy a regionális központok megfelelő szakmai támogatást tudjanak nyújtani a közeli csatlakozott intézményeknek. A növekvő igény kiszolgálásához szükségessé válik a gerinchálózat megbízhatóságának és a hálózat áteresztő képességének növelése.

2002 végén új 64 szálas, egyenként 10 Gbps sebességű optikai szálpárok kerülnek kiépítésre, de ezt nem tudják és nem is akarják kihasználni teljesen. Ennek az oka az, hogy amíg ez a kapacitás is elegendő a felhasználói igények lefedésére, addig érdemes a nagysárenddel olcsóbb gigabit Ethernet eszközöket használni az intézmények csatlakoztatására.

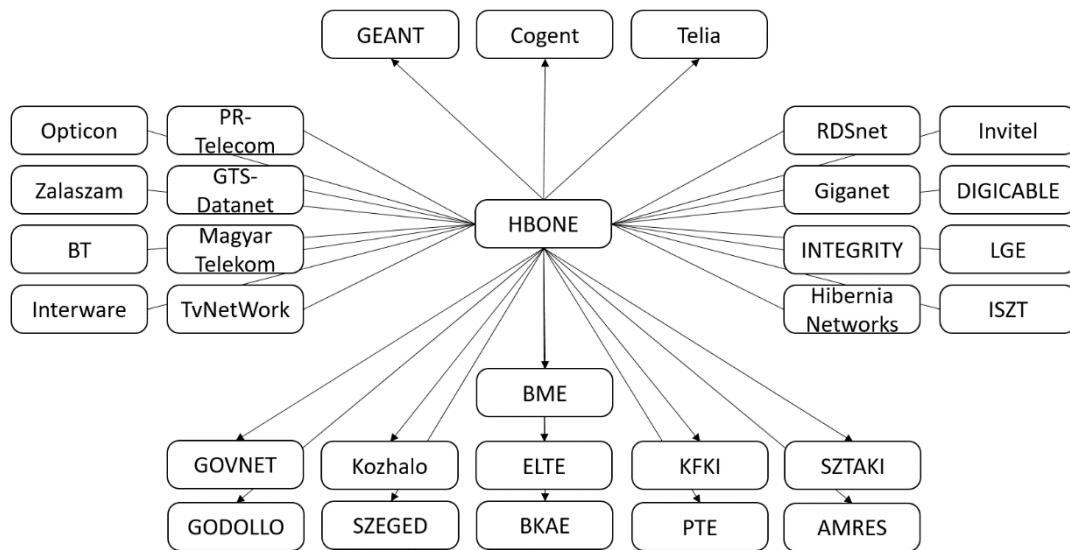
2005-től kezdve a régi technológiák párhuzamos használata az újak mellett, illetve a régiók nem egyenlő fejlettsége komoly problémákat okozott a HBONE mérnökeinek és komoly kompromisszumokat voltak kénytelenek kötni. A NIIF program alapelve, hogy világszínvonalú információs infrastruktúrával szolgálja ki az oktatási és kutatási szférát. A felméréseket és a nemzetközi trendet figyelembe véve az intézmények belső hálózatát, valamint a regionális központok kapacitását 1 Gbps adatkapcsolatok kezelésére növelték. Ennek módja vagy a kapcsolat sebességének növelése volt, vagy ahol lehetséges, kihasználni a redundáns kapcsolatokat. [23]

2012 HBONE+ projekt célja egyrészről a hálózati hatékonyság növelése, valamint a bővíthetőség és a megbízhatóság fejlesztése, másrészről pedig új szolgáltatások bevezetése volt. Ezáltal új lehetőségek nyíltak a kutatás és az egyetemek számára (teszt környezetek 100 Gbps kapacitás, CERN adatközpont). Az eduID/eduroam bevezetése és az ezeken alapuló szövetségi szolgáltatások lehetősége, mint: adattáróló, cloud szolgáltatások, e-learning, e-collaboration, labor hozzáférések. [24]

2016-től a HBONE a KIFÜ (Kormányzati Informatikai Fejlesztési Ügynökség) alá került és folytatta tovább a tevékenységeit, melyek két nagy csoportra bonthatók: uniós és hazai informatikai projektek vezetési, minőségbiztosítási feladatainak ellátása, valamint az informatikai infrastruktúra fejlesztése és üzemeltetése, az arra épülő szolgáltatások nyújtása a hazai közoktatási, felsőoktatási, kutatási intézmények, közgyűjtemények számára. [25]

3.2 HBONE IP hálózata

A KIFÜ gerinchálózata a HBONE, amely a hazai akadémiai közösségi számítógép-hálózata. A HBONE szolgálja ki a hazai felsőoktatást, közoktatást, kutatás-fejlesztést, könyvtárakat és közgyűjteményeket, valamint számos egyéb közintézményt is.



3.2.1 ábra HBONE AS gráfja [26]

A HBONE-nak három nagy szolgáltató AS-e van: a GÉANT a Cogent és a Telia.

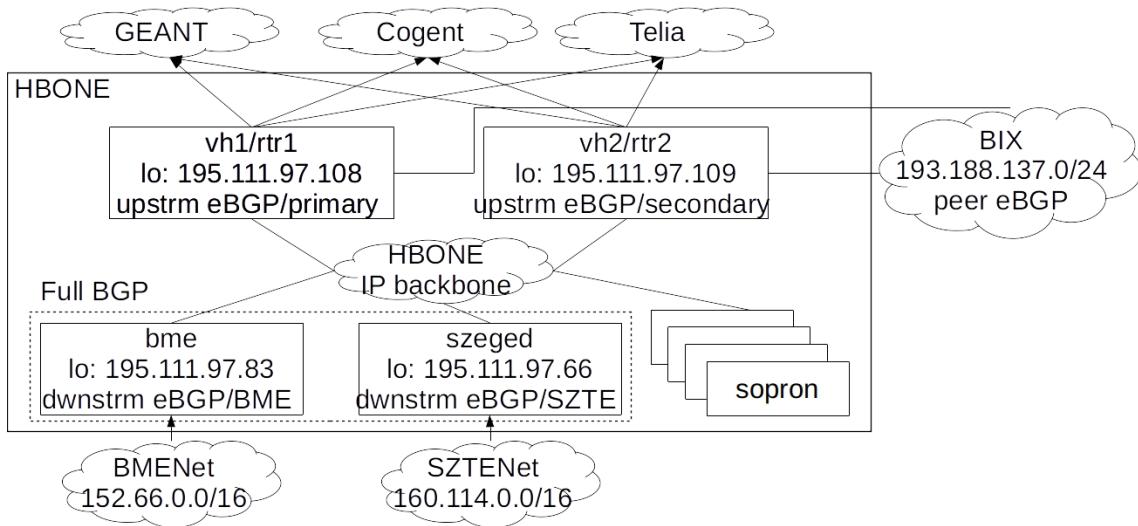
A GÉANT (Gigabit European Academic Network Technology) biztosítja a teljes hazai kutatási és oktatási közösséget, a teljes nemzetközi együttműködés (valamennyi tudományterület és valamennyi oktatási témakör) nemzetközi kapcsolatrendszerét. A KIFÜ a GÉANT közösségi tagjaként a hazai National Research and Education Network (NREN) üzemeltető/fejlesztő szerepét tölti be, biztosítva a környező országok többsége számára a nemzetközi adatforgalom lehetőségét.

A Telia svéd telekommunikációs szolgáltató a 2. legnagyobb AS, 2 200 közvetlen AS előfizetővel és több mint 35 000 AS-nek szolgáltat, ami az összes AS fele. A HBONE-al való viszonyban, mint elsődleges szolgáltató van jelen.

A Cogent a 4. legnagyobb AS, 6 200 közvetlen AS előfizetővel és több, mint 30 000 AS-nek szolgáltat. A HBONE-al való viszonyban, mint másodlagos szolgáltató van jelen.

A HBONE-nak peering megállapodása van minden magyar nagy távközlési szolgáltatóval, néhány kisebbel és pár külföldi szolgáltatóval is, mint a British Telecom, Liberty Global (angol-holland) vagy a Digi (román).

Az előfizetők között szerepelnek a nagyobb egyetemek, ideértve a BME, az SZTE, a PTE. A magyar tudományos intézmények, a SZTAKI és a KFKI, illetve a kormányzati szektor egy része, valamint a szerb akadémiai hálózat is.



3.2.2. ábra HBONE belső felépítése

A HBONE-nak két teljes BGP routere van, amelyek nemzetközi és belföldi kijáratot biztosítanak a HBONE számára, ez alatt azt kell érteni, hogy (közel) az összes meghirdetett IP tartományra van BGP bejegyzése eltárolva. Ezekre az eszközökre a diplomatervben a következőkben VH1 és VH2 néven fogok hivatkozni. Ez a két router azért kitüntetett, mert ők tanulják meg a peering partnerektől a teljes BGP táblát, majd terjesztik a HBONE-gerinc többi routere felé. Mindkét eszköz 2015-óta aktív készenléti tartalék módban működik a legkisebb adatvesztés érdekében, ezt a BGB Best External router opción segítségével érik el, amit a 3.3 fejezetben részletesen bemutatok.

A HBONE belső hálózata tartalmaz három redundáns *route reflector*-t vagy *útvonal-tükörzőt*. A route reflector-t tulajdonképpen az iBGP üzenetek mennyiségének csökkentésére használjuk. Az iBGP egy teljes háló-topológián működik, ahol minden eszköz minden másik eszköznek közvetlen szomszédja. Az útvonal-tükörző szerepe kettős, a segítségével egyszerűsít ezt a teljes háló topológiát egy csillag topológiára lehet csökkenteni. Tehát csak ennek az eszköznek kell kapcsolódnia minden másik eszközhöz, ezáltal csökken a fizikai komplexitása a rendszernek. A másik előnye pedig, hogy a

logikai komplexitás is csökken. Az útvonal-tüköröző az általa megkapott útvonalakat leegyszerűsíti, kiveszi a hurkokat és kiszámolja a legjobb útvonalat és csak azokat terjeszti szét a hálózaton. [27]

A VH1 az elsődleges határ-router, a VH2 pedig másodlagosként üzemel. A háromból kettő útvonal-tüköröző a VH1-et preferálja az összes meghirdetett prefixre, a saját lokális preferencia értékei alapján, a harmadik pedig a VH2-t. Ennek az az előnye, hogy így minden routerbe mind a két alapértelmezett lehirdetésre kerül, és ha valamiért megszűnik a VH1 elérhetősége (ami a preferált alapértelmezett irány), akkor a routereknek nem kell várni, amíg ezt a BGP visszavonja, és lehirdeti a másodlagos irányt (VH2), mert az már ott van a BGP táblájukban.

A határ-routerek nem használnak prefix összevonást (upstream aggregation), így valóban a FIB-ben a teljes BGP prefix tartomány látszik. A korábban említett szolgáltató AS-ek a Telia, a Cogent és a GÉANT közvetlen szomszédjai a VH1 és VH2 routereknek, így nincsen szükség rekurzív felderítésre. Ezáltal a címek közvetlenül a FIB-be kerülnek be.

A VH1-ben ahogy várható, mivel a default free zone része (alapértelmezett átjáró mentes zóna) nincs is benne alapértelmezett útvonal, míg a VH2 másodlagos routerben található. A VH2 alapértelmezett átjárója a VH1. A VH2 útvonalait a VH1-hez hasonlóan a szolgáltató az AS-ektől közvetlenül megkapja az eBGP kereszttü, valamint kap még az iBGP-n kereszttü a VH1-től.

A két routernak eltérő szolgáltatók vannak beállítva, mind az elsődleges útvonalnak, mind pedig a BGP Best External útnak. VH1 esetében ez úgy alakul, hogy:

- az elsődleges úton a meghirdetett tartományok a következőként alakulnak: 75% a Cogent felé megy, 13% GÉANT, és végül 10% a BIX (Budapest Internet Exchange) felé terelődik,
- a Best External utak kicsit máshogyan alakulnak, 90% a Telia felé megy, 8% a GÉANT és 2% a Cogent felé.

VH2 esetén sokkal egyszerűbb felépítésű:

- az elsődleges úton a meghirdetett tartomány majdnem teljes egészében a VH1 felé megy,
- a Best External utak a meghirdetett tartományok 90%-a GÉANT felé megy, a maradék 10 % a BIX felé.

A VH1-n és VH2-n kívül van még kettő teljes BGP forgalomtovábbítási táblával rendlekező router: az egyik a BME routere, a másik pedig a SZTE routere. A többi egyetemi router (debrecen, miskolc, pécs, sopron, szolnok, veszprém) pedig a routerek loopback címeit és a routerek közötti linkek tranzit tartományait kapják meg OSPFv2/v3 (Open Shortest Path First) protokollon keresztül, valamint van mindegyiknek egy alapértelmezett átjáró beállítva a VH2 felé. BGP protokollon tanulják a külső AS-ektől jövő prefixeket, a HBONE AS-e felől érkező belső címeket és a felhasználói prefixeket. Ezt a három iBGP route reflector valamelyikétől kapják, és csak a gerinc-hálózat részeit kapják meg, de ez csak egy részhalmaza a teljes BGP tábláknak.

A másik kettő teljes BGP-t kezelő router a BME és a SZEGED (SZTE), mindkét router a VH1/VH2-től tanulja meg a címeket a három route reflector segítségével.

3.3 BGP Best External

A BGP Best External egy olyan tulajdonság opció, ami a hálózat számára egy tartalék utat biztosít, ha az elsődleges út megszakadna. Megadja tartalékútnak a szomszédjai által előnyben részesített utat. Ez az eljárás aktív készenléti tartalékos topológiákban előnyös, ahol a szolgáltatók úgy állítják az útválasztási irányelveket, hogy a határ-router válassza az iBGP (Interior Border Gateway Protocol) munkameneten egy másik határ-routertől megkapott legjobb útvonal lehetőséget, akkor is, ha már van arra a prefixre megtanult útvonala. Az ilyenfajta aktív készenléti tartalékos topológiák elrejtik az eBGP szomszédjaik elől a tartalék útvonalukat, vagyis nem hirdetnek meg utat erre a prefixre.

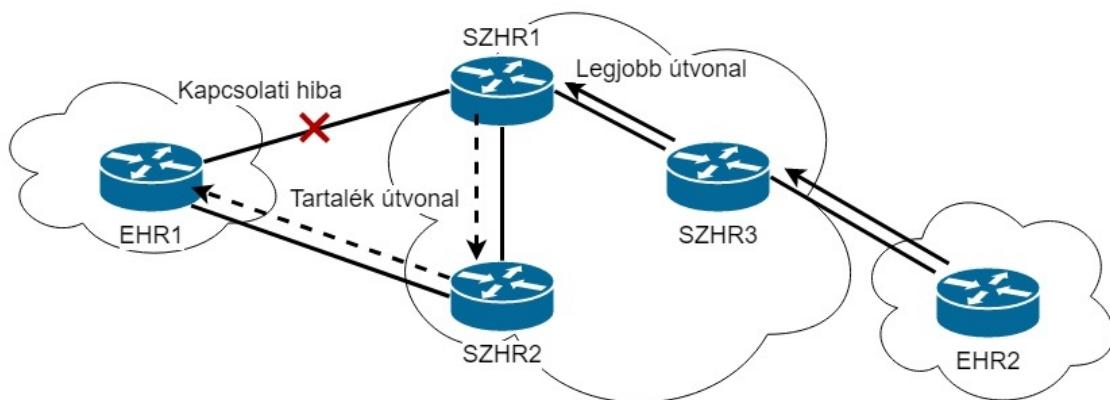
A BGP Best External használatához van pár előfeltétel. Gyorsan észlelni kell az adott kapcsolat hibáját. Többhónak (multihome) kell lennie az AS-nek, ahol használni szeretnénk. A tartalék útnak egy egyedi útnak kell lennie.

Az egyes routerek csak pontosan egy utat fognak meghirdetni, ami a legjobb külső (best external) út lesz. Ez a legjobb külső út viselkedés, azt fogja eredményezni, hogy minden cél felé két útvonalat tanul meg. Ha át kell váltania az iBGP útra, az eddigi utat nem vonja vissza a belső szomszédjai elől, csupán a tartalékot kezdi el használni.

A best external route (legjobb külső útvonal), mint készenléti tartalékként funkcionál, ami nem más, mint a külső szomszédjai által a leginkább előnyben részesített útvonal, amely kétféle lehet:

- Az AS-en belüli, de másik csoportban lévő routertől származó iBGP útvonal.
- Egy másik AS egy olyan routere, amivel közvetlen szomszédosak és eBGP kapcsolat van közöttük.

Ez az út lehet másik, mint a RIB-ben lévő legjobb útvonal, mert az lehet belső út is. Azzal, hogy megengedjük, hogy az elsődleges útvonal mellett legyen eltárolva és meghirdetve a legjobb külső útvonal, gyorsabb kapcsolati helyreállást eredményez, ha az elsődleges útvonal megszakad.



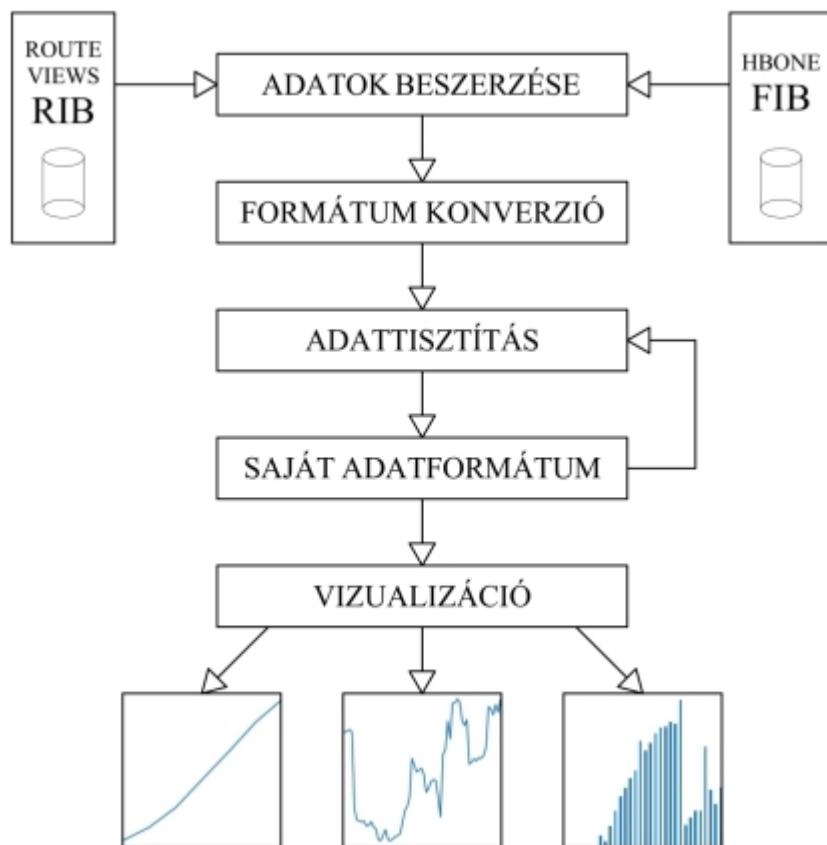
3.3.1. ábra BGP Best External kapcsolat

A 3.3.1. ábra BGP Best External kapcsolat3.3.1. ábra-n kiépült eBGP kapcsolat van szolgáltatói határ-routerek és az előfizető határ-routerek között. Az SZHR1 az elsődleges router és neki a legmagasabb a helyi preferenciája. A forgalom EHR2 felől EHR1 felé az elsődleges SZHR1 routert használja. SZHR1 routernek két útvonala van EHR1 felé, vagyis EHR1 többszörösen SZHR1 és SZHR2 routerrel. SZHR1 és SZHR2 BGP Best External funkció be van kapcsolva. SZHR2 mint tartalék útvonal lett felvéve, és meghirdetve. Amikor az SZHR1 – EHR1 kapcsolat megszakad, és ezt érzékelik a

routerek, a forgalom azonnal átirányításra kerül az SZHR2 tartalék útvonalára. Ezáltal az adatvesztés minimális lesz.

4 Az útvonalválasztási és csomagtovábbítási táblák forrása

A fejezetben az adatok forrásáról, azok beszerzési módjáról, adatkonverziókról és köztes fájlokról, az adatok egységesítéséről és tisztításáról lesz szó.



4. ábra Az adatok feldolgozása

4.1 RIB állományok beszerzése

Az oregoni egyetem Route views Project-jét [28] felhasználva, az általuk publikusan elérhetővé tett BGP RIB-eket szerettem volna letölteni az elemzés elvégzéséhez. A Route views Project-nek eredetileg az volt a célja, hogy az Internet üzemeltetőinek adjon valós idejű globális BGP információkat, különböző AS-eknek a szemszögéből. Az oldalon közel 30 helyről vannak gyűjtve a RIB állományok szerte a világból. A tanszéki rendszerben 2013 novembere óta vannak a FIB-ek gyűjtve, tehát olyan RIB szolgáltatókat kellett keresnem, amikből már volt adat 2013-ból is. Mivel volt rá lehetőségem, több kontinensről választottam egyet-egyet. Európából a Londoni

Internet Exchange (továbbiakban LINX) RIB-jét, az Amerikai Equinix Ashburn (továbbiakban EQIX), ami a Washington melletti IX (Internet Exchange), a Kenyai Internet Exchange-t (továbbiakban KIXP) és a Sydney Internet Exchange-t (továbbiakban SYDNEY). Ez a négy RIB szolgáltató megfelelőnek tűnt, mert volt nekik már 2013-tól kezdve RIB gyűjtve és nem volt jelentős hiány az adatkészletükben. Mindegyik esetben előfordult, hogy volt egy-egy nap vagy időszak, amikor nem volt RIB archiválva, ennek saját állításuk szerint az volt az oka, hogy pont a karbantartás időszakába esett bele a gyűjtés. Az EQIX esetében volt egy kieső időszak, egy fél évnyi adatot hiba miatt nem lehetett használni.

Az archív állományok tömörítve vannak feltöltve a weboldalra egy egyszerű könyvtárstruktúrában, a 4.1.1 kódrészleten leírt formában, így könnyen írni lehetett rá egy scriptet, ami ezeket letöltheti.

<http://archive.routeviews.org/route-views.eqix/bgpdata/2013.11/RIBS/rib.20131101.0000.bz2>

4.1.1. kódrészlet Egy példa egy konkrét RIB fájl linkjére (aláhúzással jelöltem a változtatandó részeket)

Pythonban írtam rá egy rövid scriptet, ami az `urllib` [29] könyvtárat felhasználva letöltheti őket. Ennek az volt az előnye, hogy a könyvtár már készen adta a link letöltés funkciót, nekem csak elő kellett állítani azokat. Szerencsére a linkek formalizáltak, mint a 4.1.1. kódrészleten látszik, így elég volt csak a dátumokat cserélni benne, és a forrás nevét. Miután letöltöttem, ki kellett írni háttértárra bináris formában.

Minden páros órában van egy ilyen RIB lementve, a közöttük lévő időközben pedig az `update` üzenetek vannak 20 percenként mentve. A saját FIB-jeink minden éjfélkor lettek lementve, ezért a RIB-ekből is az éjfélit töltöttem le. A link végén a négy darab „0000” az időt jelöli.

4.2 RIB formátum konverzió

A letöltött RIB-ek tömörítve vannak `bz2` fájl kiterjesztésben, amely a `bzip2` open-source tömörítő alkalmazás saját formátuma. A formátummal nem volt probléma, mivel a `bgpdump` [30] linuxos parancs pont ebben a formátumban szerette volna a bemenetét megkapni. A parancsot linux alatt futtatott python script adta ki az os könyvtár segítségével. Az `os.system` parancsnak egyszerű string összefűzéssel

legeneráltam a parancsokat, majd kiadtam a linux futtató környezetnek, mint a 4.2.1 kódrészleten látszik. A bgpdump parancs a „-O” kapcsolóval rögtön lehetővé teszi, hogy közvetlenül fájlba írjuk a végeredményt, ne pedig a konzol ablakba.

```
bgpdump rib.20190101.0000.bz2 -O 20190101.txt
```

4.2.1. kódrészlet A **bgpdump** használati parancsa

A forrásfájlok mérete nagyon változó, a várakozásoknak megfelelően folyamatosan nőttek, de a forrástól függően is különböztek, a legkisebb 3 MB körüli, a legnagyobb pedig 120 MB volt (a KIXP jellemzően kisebb volt, a legnagyobb is 8MB, míg a LINX-nél a legkisebb volt 40 MB). Ezek a fájlok olvasható szöveges formátumban a 18 GB méretet is elértek, így ez nyilvánvalóan kezelhetetlen hagyományos eszközökkel, szerencsére nem is akartam őket nagyon szabad szemmel nézegetni, de hogy lássuk mi volt benne, bemutatnék egy részletet:

```
TIME: 09/01/10 00:00:01
TYPE: TABLE_DUMP_V2/IPV4_UNICAST
PREFIX: 80.89.181.0/24
SEQUENCE: 78977
FROM: 202.167.228.44 AS10026
ORIGINATED: 02/17/70 16:30:33
ORIGIN: INCOMPLETE
ASPATH: 10026 174 30998 30998 30998 30998 35074 35074 35074
35074 35074 35074 35074 35074 35074 35074 35074
NEXT_HOP: 202.167.228.44
MULTI_EXIT_DISC: 10050
COMMUNITY: 10026:1230 10026:31840 10026:40913
```

4.2.2. kódrészlet **bgpdump** bejegyzés

A 4.2.2. kódrészleten látható, hogy egy prefixhez tartozó bejegyzés is 11 sorban van leírva és sok olyan információ található benne, ami a jelen diplomaterv szempontjából teljesen érdektelen. Ebből a 11 sorból csak a prefix értéke kell és a next-hop. Ezt a később bemutatott formátum szerint egy sorba rendeztem. Egy prefixhez több bejegyzés is tartozik, ezekből mindig az elsőt vettet ki. Ezért ezekből az esetenként hatalmas fájlokból kiszedtem az érdekes két értéket egyedi prefix bejegyzéseként és elmentettem egy új fájlba, ami immár megegyezik a FIB-ek formátumával, így ugyanaz a kód futtatható rajtuk, tehát egységesen lehet őket kezelni. Ezt követően a nagyméretű „köztes” állományt töröltem, mivel nincs rájuk ez követően szükség.

4.3 FIB állományok

Az egyetem kapcsolata miatt lehetőség volt a HBONE, a magyar akadémiai közösségi számítógép hálózatából a BGP routerek FIB-jeit megkapni. Nekik négy darab teljes BGP routerük van, ami azt jelenti, hogy ebben a négy routerben megvan a teljes BGP FIB tábla. Ezek a BME, Szegedi Tudományegyetem routere (a továbbiakban SZEGED), valamint a BIX-ben (Budapest Internet Exchange) a VH1 és VH2 router.

Ezek az állományok minden nap éjfélkor mentődnek le automatikusan egy `txt` fájlba és az összes ilyen `txt` fájl egy darab zipbe van csomagolva.

4.4 FIB-ek feldolgozása

A napi mentésű FIB-ek egy zip fájlba érkeznek az összes többivel összecsomagolva. Ezt ki kell tömöríteni és a megfelelő fájlokat, vagyis a négy szükségeset fel kell dolgozni. Egy FIB formátuma egy egyszerű `txt` fájl esetünkben, mely tartalmilag soronként egy prefixet tartalmaz és egy perjellel elválasztott alhálózati maszkot és egy next-hop IP címet egy tabulátorral elválasztva. A sorokban egymást követő prefixek szigorú monoton nőnek. Ennek formátuma az alábbi példán látható:

1.0.24.0/23	196.223.14.55
1.0.26.0/23	196.223.14.55
1.0.28.0/22	196.223.14.55
1.0.64.0/18	196.223.14.55
1.0.128.0/17	196.223.14.55
1.0.128.0/18	196.223.14.55
1.0.128.0/19	196.223.14.55

3.4.1. kód részlet FIB fájl tartalma

A FIB fájlok mérete is nő, ahogy telik az idő, hisz 2013 novemberében csak 480 000 bejegyzés volt benne, ez viszont 2020 év végére már 820 000 bejegyzésre dagadt. Ennek megfelelően a fájl mérete is 11 MB-ról 25 MB-ra nőtt.

Minden fájlnak a nevében benne van a dátuma és a pontos ideje, hogy mikor kerültek mentésre, ez jellemzően minden nap éjfél után történt.

A kitömörített fájlok felhasználása után a leszámolt információk egy egyesített `csv` fájlba kerülnek, amiből a diagramokat lehet készíteni. Ezt követően a kitömörített fájlok törlésre kerülnek.

4.5 Adatformázás, tisztítás

Az adatokat a Python Pandas [31] segítségével rendezem be az adatmodellbe. A Pandas egy gyors, hatékony, könnyen használható és teljesen nyílt forráskódú adatelemző és kezelő könyvtár a Pythonhoz. Az adatmodellt a Matplotlib [32] vizualizációs könyvtár felhasználásával készítettem. A Matplotlib teljesen testre szabható diagram-megjelenítést tesz lehetővé.

Először is megnéztem, hogy a Pandas milyen formátumban szeretné az adatokat megkapni. Ez egy sima csv (comma-separated values, veszővel elválasztott értékek) fájl. Az első sorában egy fejléc sor van, az utána lévő sorok pedig az egyes napokhoz tartozó értékek. A felépítése így néz ki egy sornak: dátum (yyyy-mm-dd formátumban), összes prefix száma, specifikusabb prefixek száma, majd ezek kibontása prefix hosszokra (pref_1-pref_32 és msp_1-msp_32), minden pozitív egész szám, hiszen darabszámokról van szó. A sor végén az IP fedési tartomány per nyolc ekvivalens értéke, ami egy tört szám 16 számjegy pontossággal. Ezek kiszámolási módja a következő fejezet témaköre.

Minden adatforrás (mind a nyolc) egy ilyen módon felépülő csv fájlban van eltárolva. Sajnos voltak olyanok a hiányzó napokon túl, hogy a mentő script nem tudott lefutni teljesen, és így csak egy részlete került lementésre. Amikor ebben az állapotban csináltam egy diagramot róla, vizuálisan is látható vált, hogy viszonylag sok ilyen „lefelé álló tüské” van a diagramban. Ezeket a napokat a csv fájlok ból kiszűrtetem, mivel a vizsgálat szempontjából kiugró értéknek minősül. Így, ahol a darabszám értéke az előző és következő $\pm 5\%$ -ban nem volt benne, azokat törltem a fájlból, ezt követően a tüskék eltűntek.

Az egyéb normális működésből adódóan, amit az eltűnő megjelenő prefixek okoznak, a vonal diagram nagyon vastag, ”szörős” volt. Ennek eltüntetését heti átlagolással szüntettem meg.

Ezekre a lépésekre a diagramok elkészítéséhez volt csak szükség, mivel a next-hop információkat közvetlenül a FIB fájlok ból kell kinyerni.

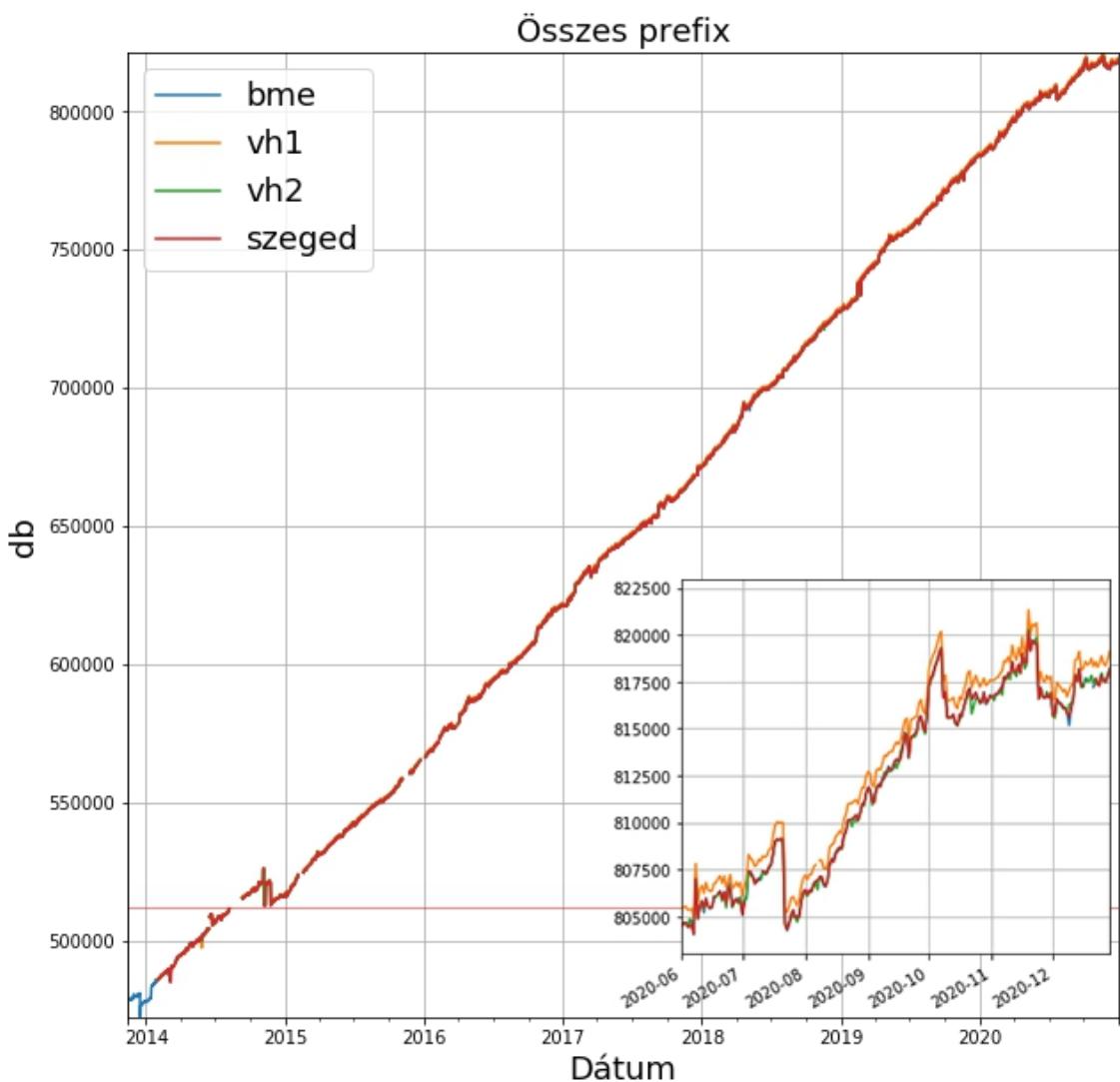
5 Prefix elemzés

A fejezetben olyan statisztikai információkat vetem figyelembe, amiket jelenleg is folyó más elemzések is használnak, például az Internet heti CIDR beszámolóiban. [3]. Így egyenként végigmegyek ezeknek a jellemzőknek az előállítási módján, ezt követően pedig a FIB mintákon kapott eredményeket bemutatom és elemzem, majd hozzá hasonlítom a publikus RIB-ből kapott adatokhoz. Az alfejezetek végén összegzem az eredményeket és következtetést vonok le.

5.1 Prefixek száma

A prefixek száma a legérdekesebb és a legkézenfekvőbb metrika, ami létezik. Ahogy említettem, a routerek tárkapacitása ilyen szempontból nagyon limitált. Itt tulajdonképpen a BGP bejegyzések számára vagyunk kíváncsiak, tehát nem kell mást tenni, mint megnézni, hogy hány sor van az adott táblában. A 4.4 fejezetben kifejtem, hogy egy ilyen bejegyzés egy sor a FIB-ben és a RIB-ből képzett egységesített fájlban is.

A FIB bejegyzések számában 71% növekedés volt megfigyelhető a vizsgált időszak hét évében. A bejegyzések száma 478 000-ről 818 000-re emelkedett. Ez naponta átlagosan 130 új bejegyzést jelent.

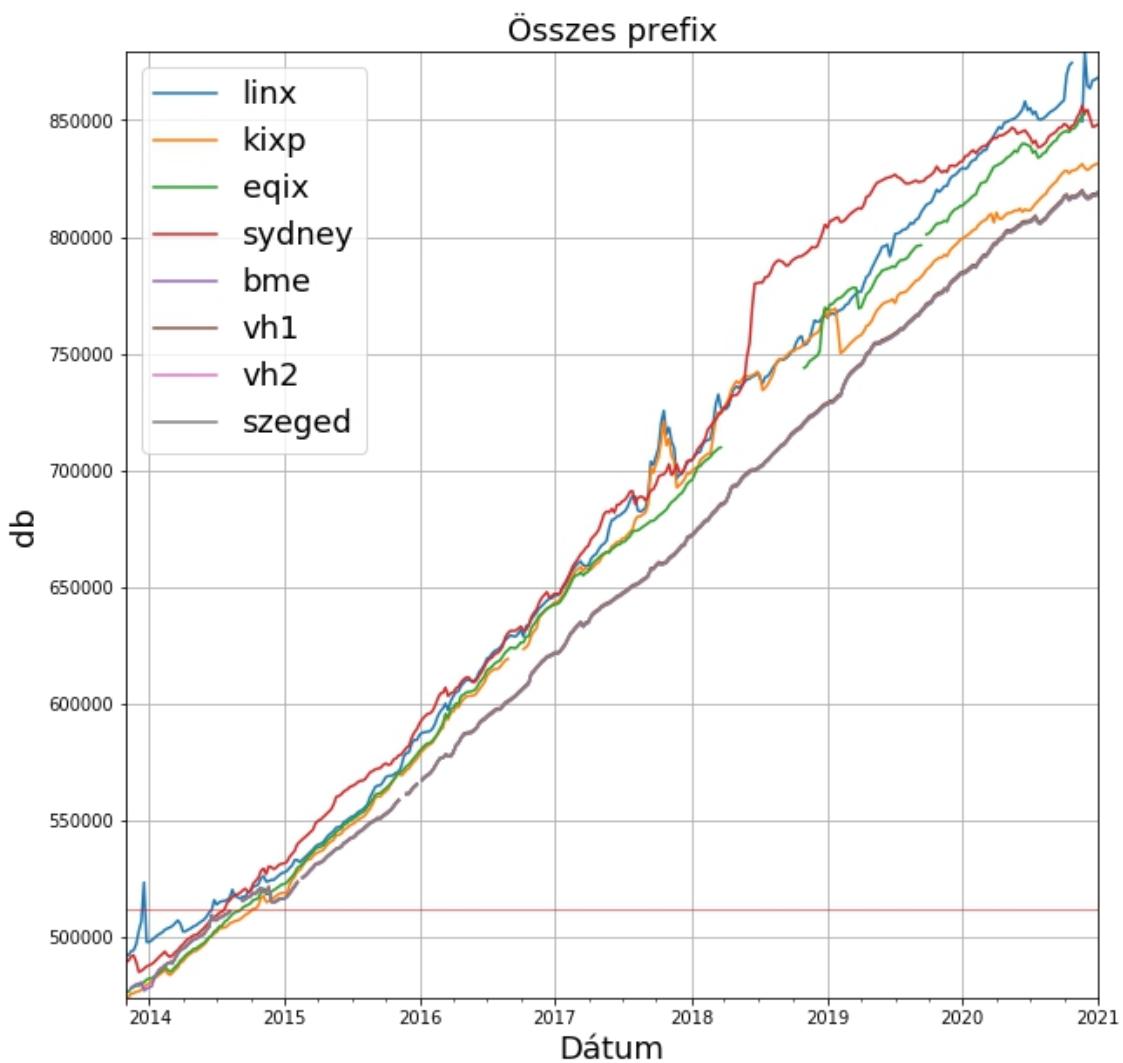


5.1.1. ábra FIB bejegyzések száma idősoros grafikonon

Az 5.1.1. ábra nem mutat nagy különbséget a 4 FIB tábla bejegyzéseinek számában, gyakorlatilag teljesen egymásra esnek. A vh1-ben van csak egy átlagosan 1000 bejegyzéssel több, és ezt az évek során tartja is.

Érdekes megfigyelni, hogy a grafikon folyamatosan nő néhány visszaeséstől eltekintve, de az utóbbi fél évben ellaposodik és egy plató szakaszra ér.

A RIB bejegyzéseket is rárakva a diagramra közel sem kapunk ilyen egybeeső grafikonokat, a nagy trend-jellegű változások itt is megfigyelhetőek mindegyik esetében, hogy együtt mozognak, de jóval nagyobb szórás mellett.



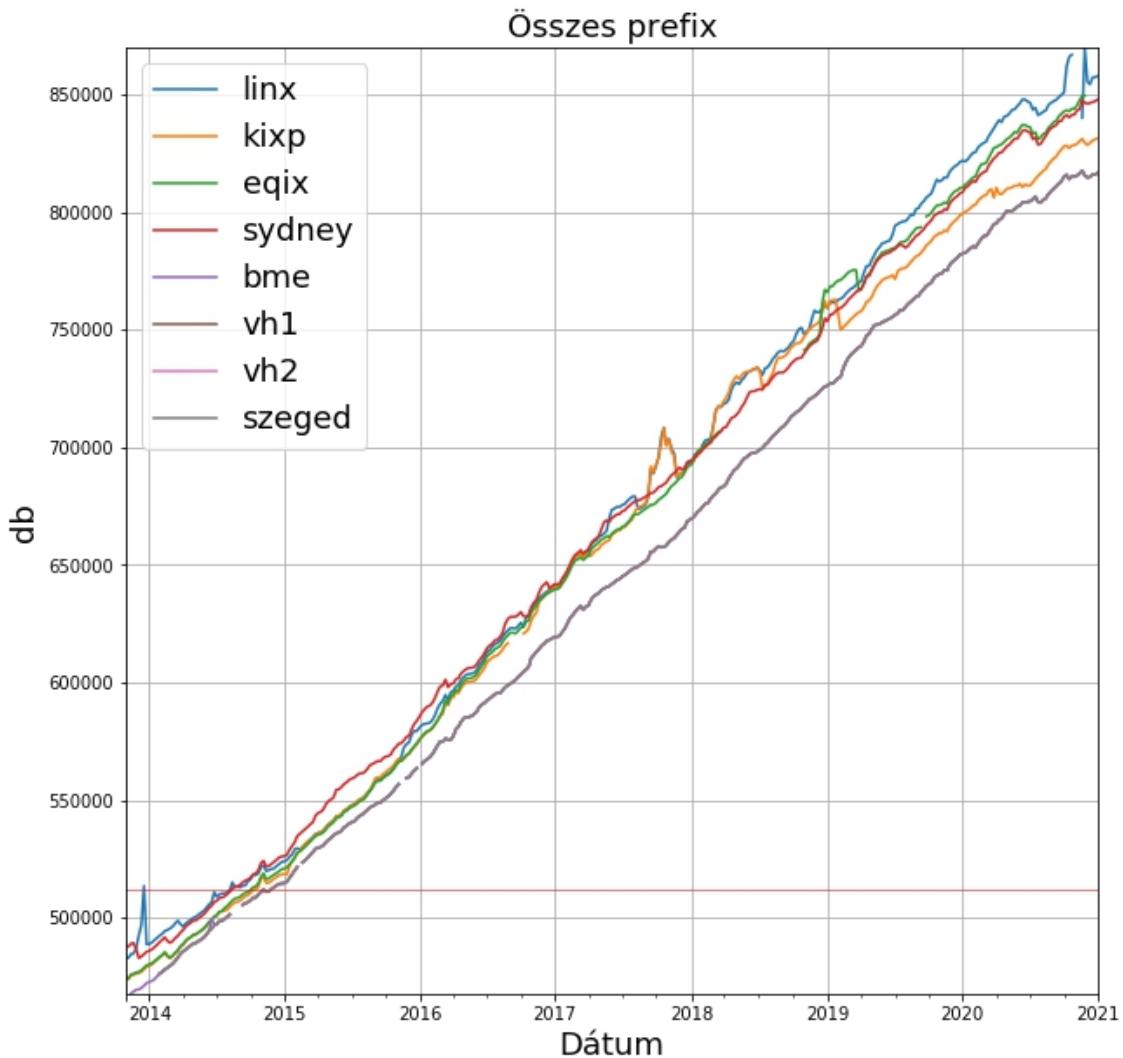
5.1.2. ábra FIB, RIB bejegyzések száma idősoros grafikonon

Az 5.1.2. ábra tanulsága az, hogy a RIB-ekben jóval több bejegyzés van, sőt, a RIB-ek többsége egyre jobban távolodik is a FIB-ek vonalától. 2014-ben elég közel voltak egymáshoz, volt ahol alá is esett, mára van olyan, ami 40 000 bejegyzéssel is többet tartalmaz.

Mindkét diagramra behúztam 512 000 értéknél egy vízszintes vonalat, ami a központi routerek maximum eltárolható bejegyzések számát jelentette egyidőben. Látszik, hogy ezen a ponton jelentősen túlszárnyáltunk az utóbbi hat évben.

Viszont az mindenkiépp pozitív, hogy el tudom mondani, hogy a RIB-ek jó felső becslést adnak a FIB-ekben található bejegyzések számára. Ráadásul, ha szerencsésen választjuk meg, akkor elég közeli értékkel is dolgozhatunk, mint a kixp esetén.

Kiszűrtem a 24-nél hosszabb prefixeket a vizualizációból, hogy azok, amik a BGP esetén nem használunk, ne zavarjanak be.



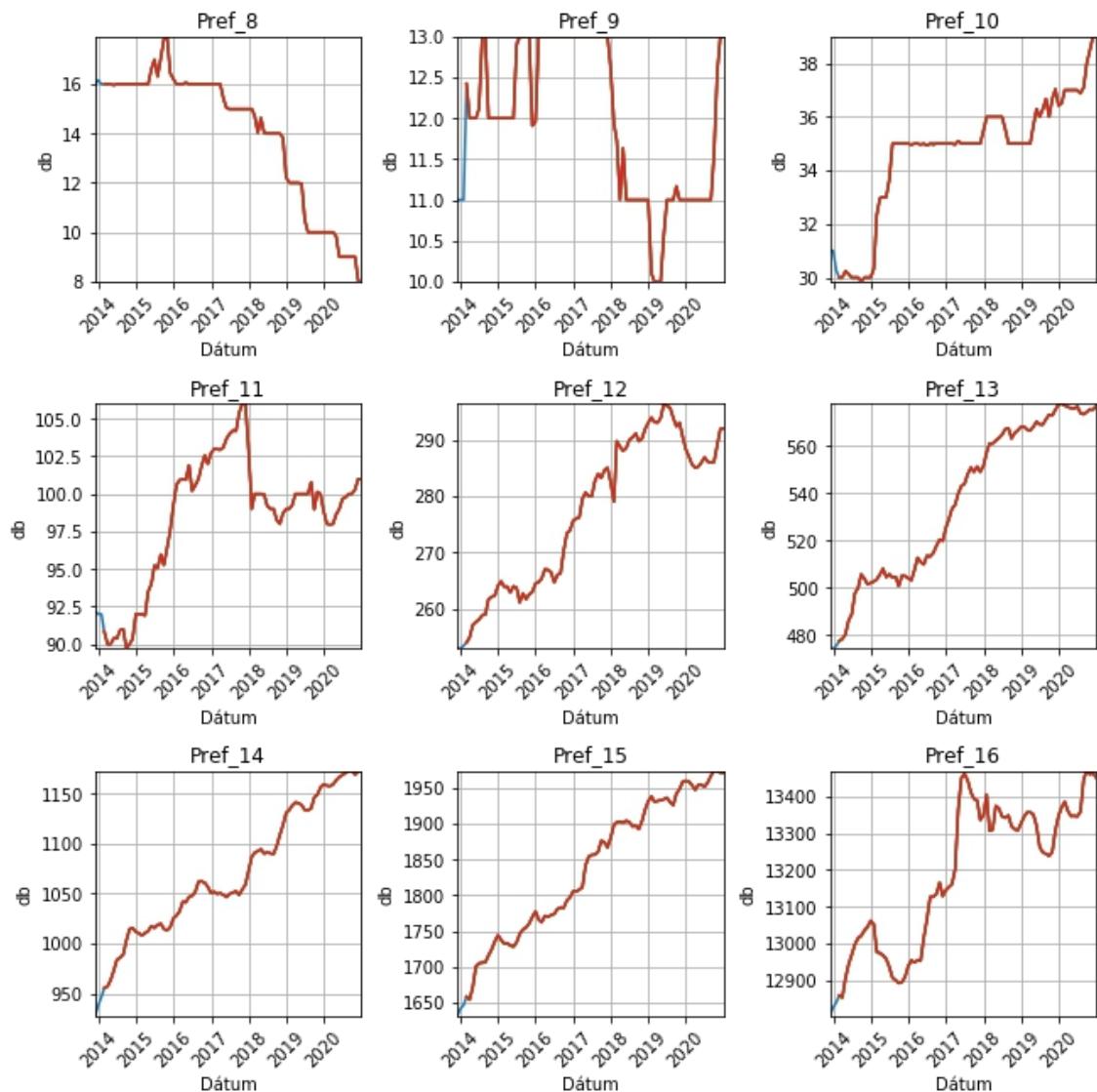
5.1.3. ábra FIB, RIB 25-nél rövidebb prefixek

Az 5.1.3. ábra így sokkal egységesebb képet alkot, a sydney is jobban belesimul a többi grafikon közé.

5.2 Prefixek száma hosszonkénti bontásban

Ez alatt azt kell érteni, hogy az IP címből hány bit tartozik a hálózat azonosítóhoz, tehát hogy milyen hosszú az adott prefix. A prefix hosszonkénti darabszámot az előzőhöz nagyon hasonlóan meglehet kapni, csak itt nem egy összeg lesz a vége, hanem 32 összeg-érték minden hosszra egy-egy.

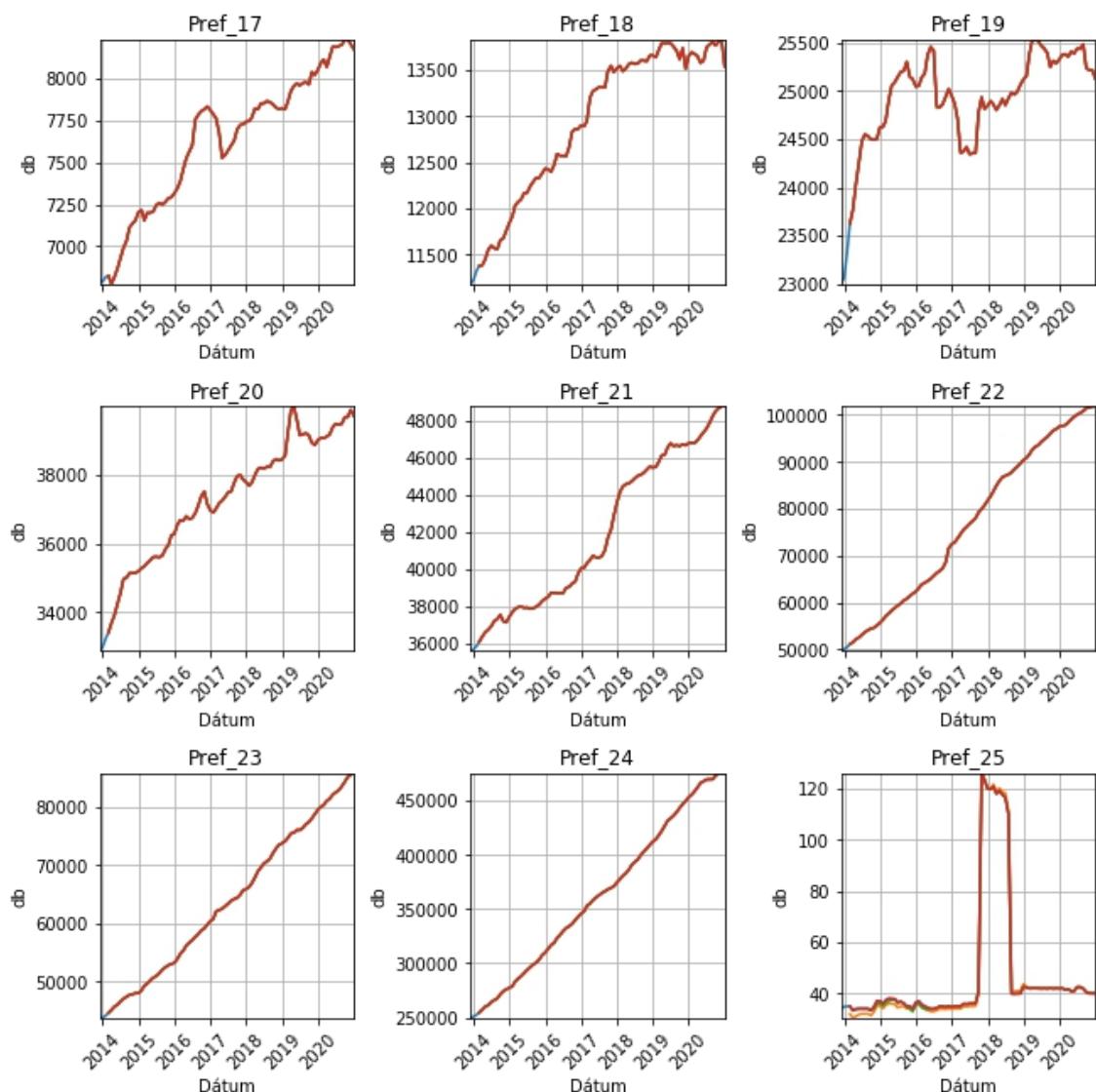
Azt vártuk, hogy nyolcnál rövidebb prefixek nem lesznek a táblákban (a BGP működéséből adódóan) és ez minden esetben igaz is volt. Valamint, hogy a rövidebb prefixek, amik nagyobb tartományokat fednek le, ezek száma csökkenjen, a rövidebb prefixek, amik kis tartományokat fednek le, pedig nőjön. Vagyis, hogy a nagy tartományok feldarabolódjanak. A növekedés egészen 24-ig tart, majd ott hirtelen csökken a számuk, és nem is lesznek teljesen egyformák.



**5.2.1. ábra A 8-16 hosszú FIB bejegyzések száma hosszonkénti bontásban, idősoros grafikonon
jelmagyarázat az 5.2.3. ábra-n**

A prefixek darabszáma az 5.2.1. ábra-n továbbra is együtt marad kivétel nélkül. Az a feltétezés is beigazolódott, hogy a nagyobb prefixek száma csökken, ugyanis a 8 hosszú prefixek a felére csökkentek, míg a többi nőtt a vizsgált időtartományban. Egyes

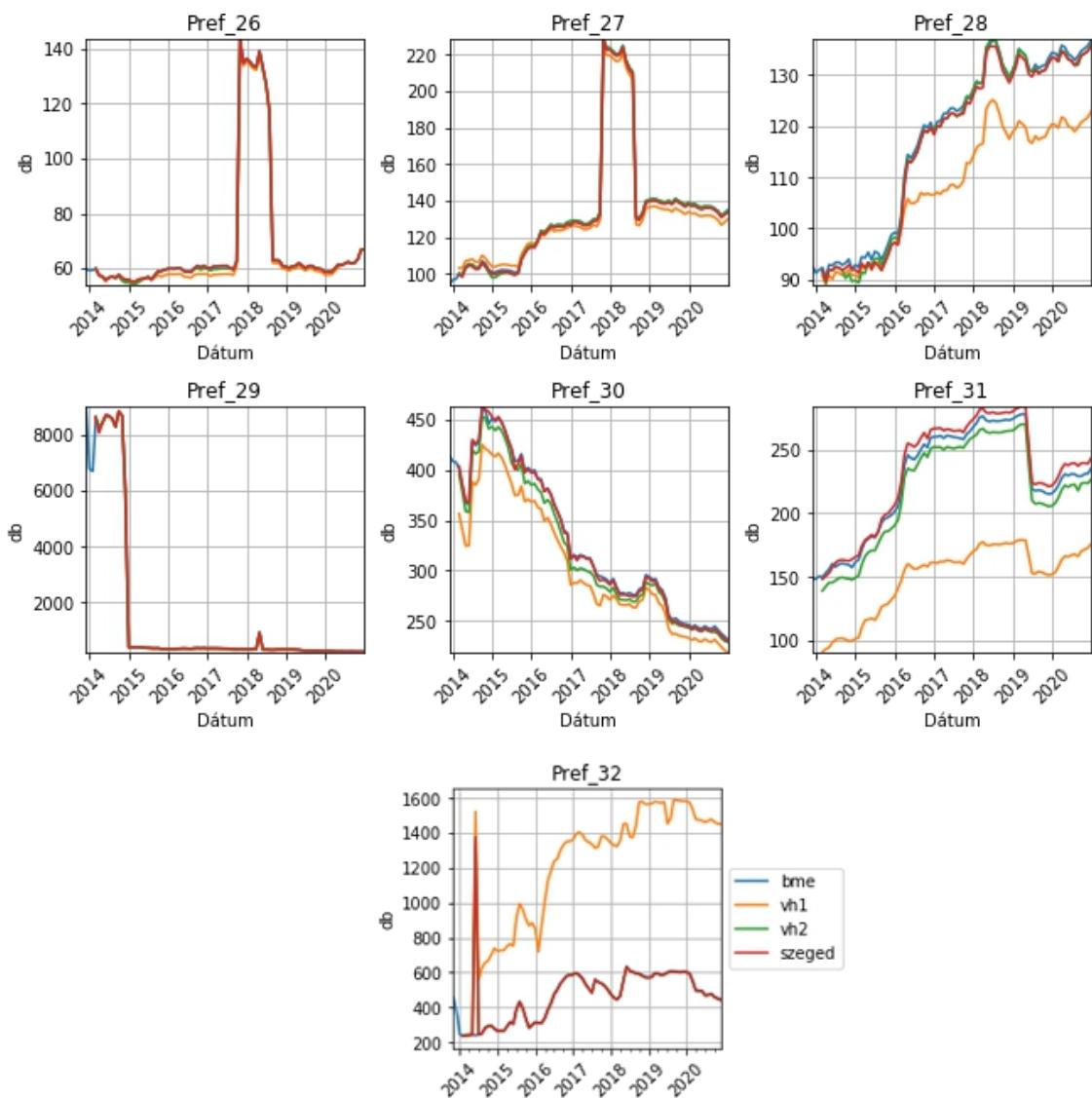
hosszok bejegyzési darabszáma stagnál, vagy ellaposodik a 2018-as évtől kezdve, de ez csak a középső tartományban figyelhető meg inkább.



**5.2.2. ábra A 17-25 hosszú FIB bejegyzések száma hosszonkénti bontásban, idősoros grafikonon
jelmagyarázat az 5.2.3. ábra-n**

Az 5.2.2. ábra-n jól kivehető, hogy a közel 350 000-res növekedés jelentős részét a 24 hosszú prefixek alkotják, aminek közel megduplázódott a száma a vizsgált időtartományban.

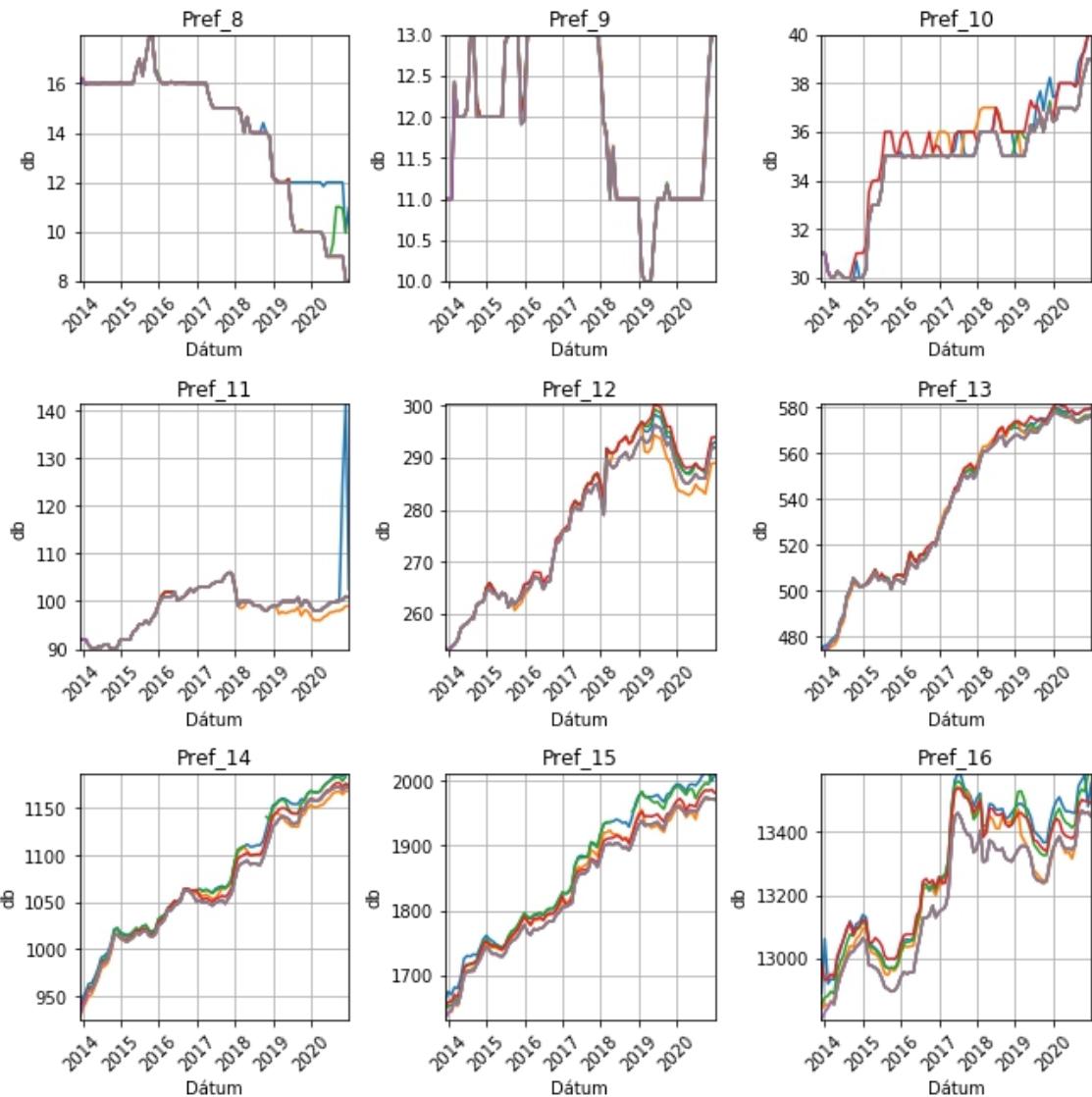
A várakozásnak megfelelően a 24-nél hosszabb prefixek száma jelentősen kevesebb, például a 25 hosszúakból csak 40 darab van, míg a 24 hosszúból 476 000.



5.2.3. ábra A 26-32 hosszú FIB bejegyzések száma hosszonkénti bontásban idősoros grafikonon

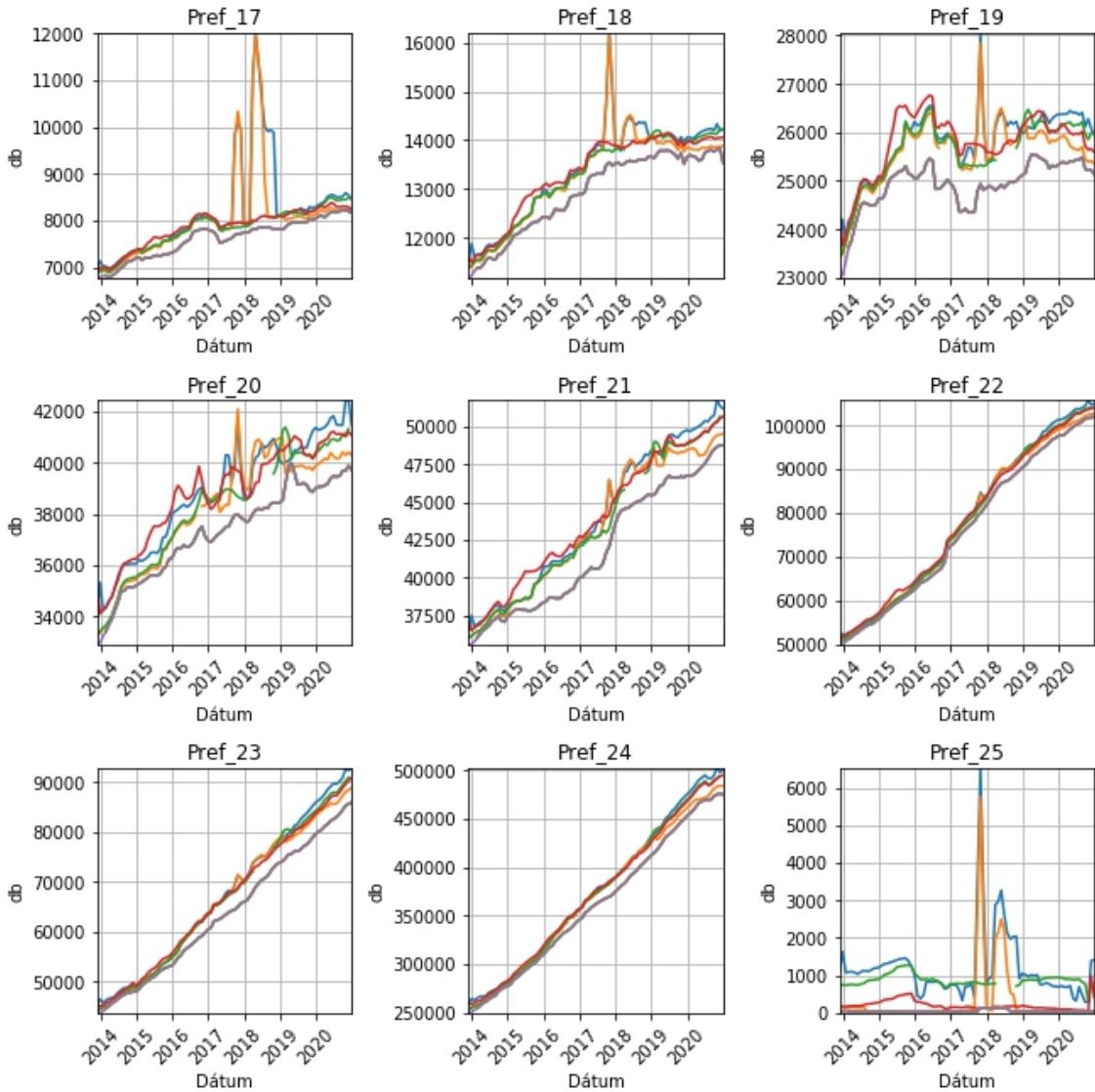
Az 5.2.3. ábra-n jól látható, hogy a nagyon kicsi, vagy konkrét IP címek darabszáma már jobban szétugrik egymástól. Itt jól lehet látni, hogy az összes prefixben látott vh1 eltérést a 32 hosszú „prefixek” okozzák.

A RIB-eknél is ugyanazokat az előfeltételeket állítom, mint a FIB-ek esetén volt, hogy a 8 hosszú prefixek száma csökken, a többi javarészében nő, a 24 hosszúakból lesz a legtöbb és a nála hosszabbak nem fognak érdemi információt hordozni, és teljesen zavaros lesz.



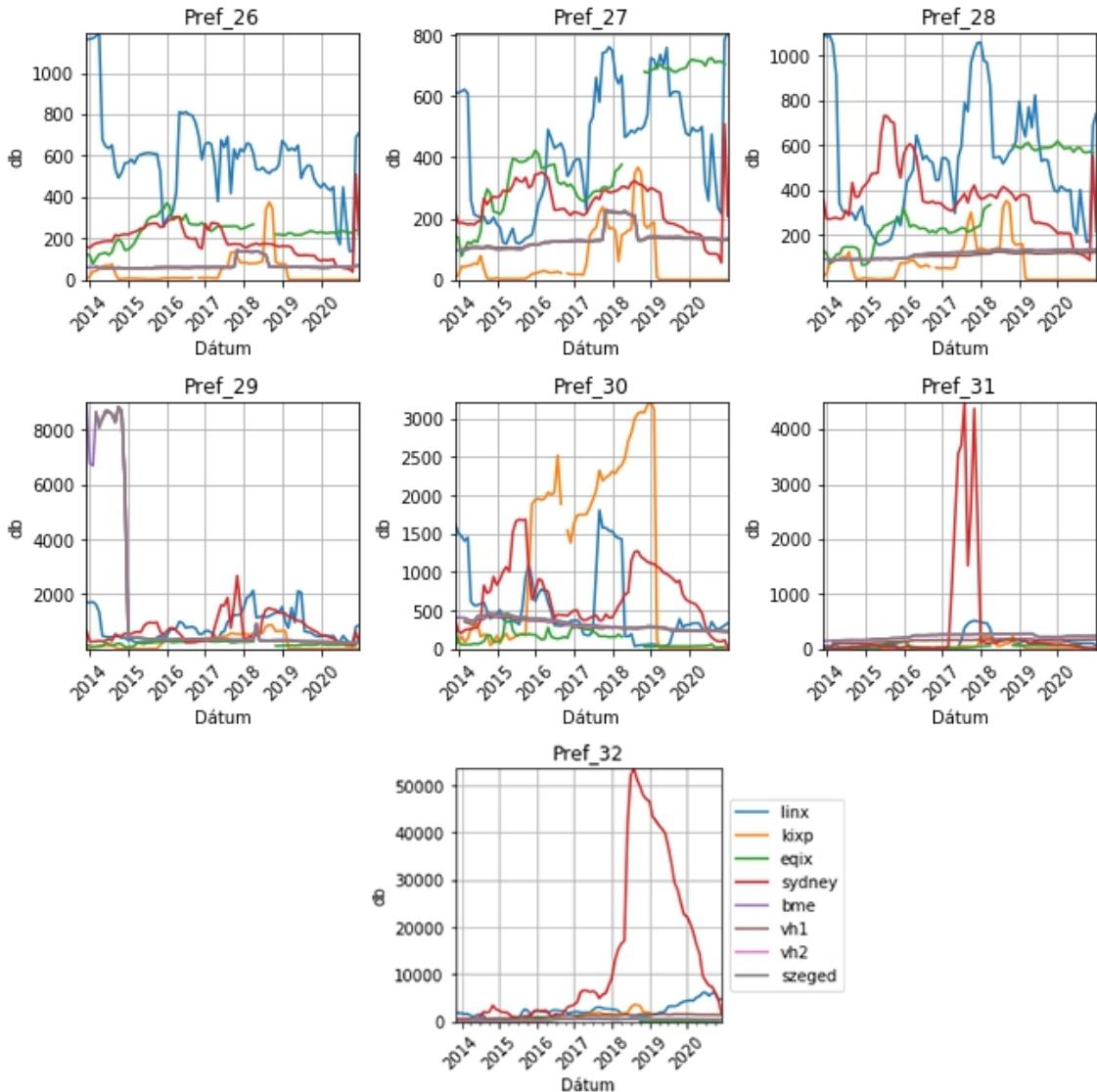
5.2.4. ábra A 8-16 hosszú FIB, RIB bejegyzések száma hosszonkénti bontásban, idősoros grafikonon, jelmagyarázat az 5.2.6. ábra-n

Az 5.2.4. ábra-t megvizsgálva azt láthatjuk, hogy a 8 hosszú prefixek 2019 második feléig szépen együtt változnak a RIB és FIB táblákban, de utána kettőben nem csökken le 8-ra, mint a FIB-eknél, hanem 10 darabra csökken csak. Viszont a 9-es hossznál teljesen megegyeznek.



5.2.5. ábra A 17-25 hosszú FIB, RIB bejegyzések száma hosszonkénti bontásban, idősoros grafikonon, jelmagyarázat az 5.2.6. ábra-n

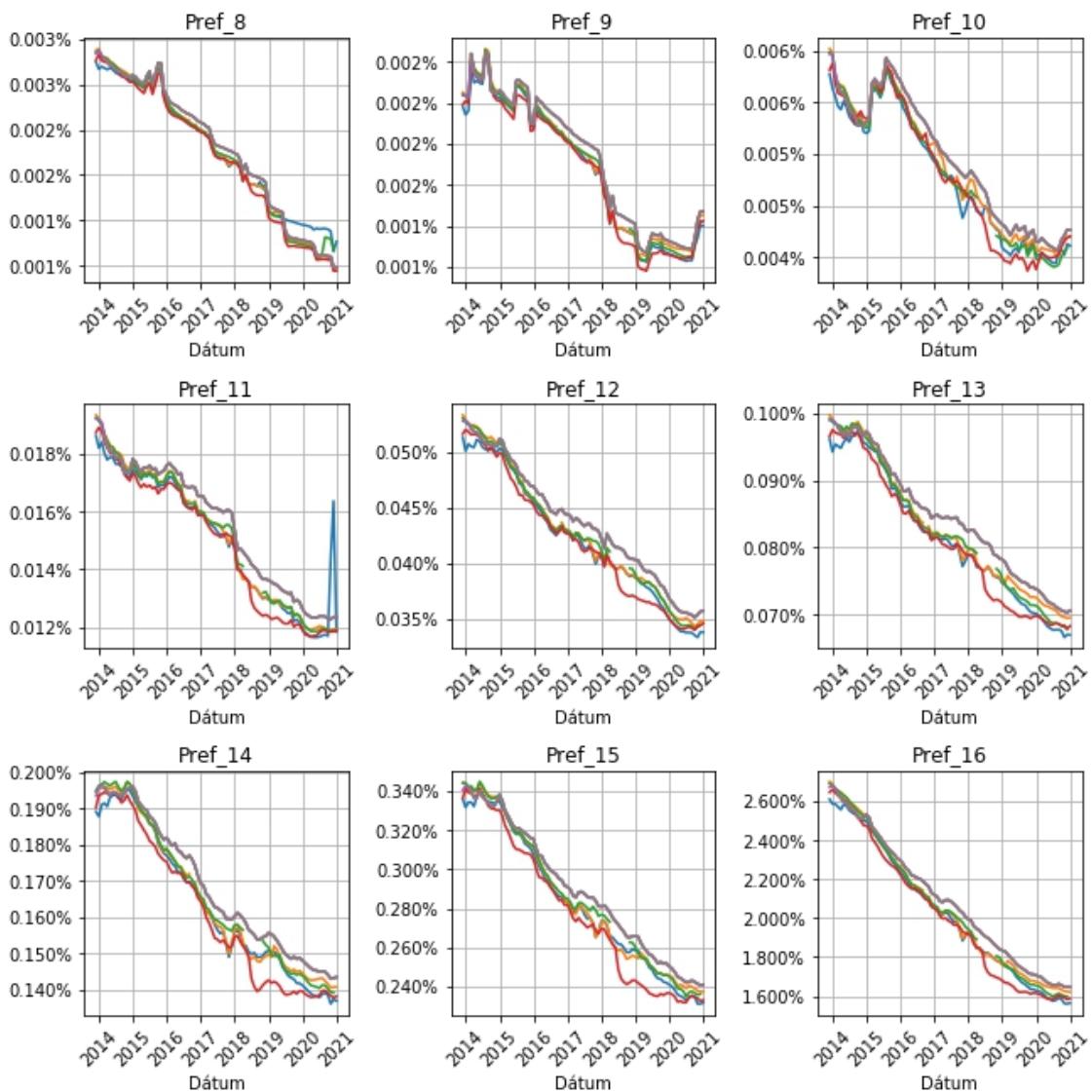
Az 5.2.5. ábra-n látható, hogy a grafikonok továbbra is nagyon hasonlóak egymáshoz, de egyre növekvő különbséggel. Érdekes még, hogy a 17, 18, 19 hosszú prefixeknél a linx és kixp-nél is egy nagyobb tüske látható 2017-2018-ban. Pedig másik kontinensen vannak, de a másik kettő RIB-ben vagy a FIB-ekben egyáltalán nem látható. Itt is tartja a korábban megfigyelt trendet, hogy a 24 hosszú prefixek adják a teljes RIB méretének a jelentős részét.



5.2.6. ábra A 26-32 hosszú FIB, RIB bejegyzések száma hosszonkénti bontásban, idősoros grafikonon

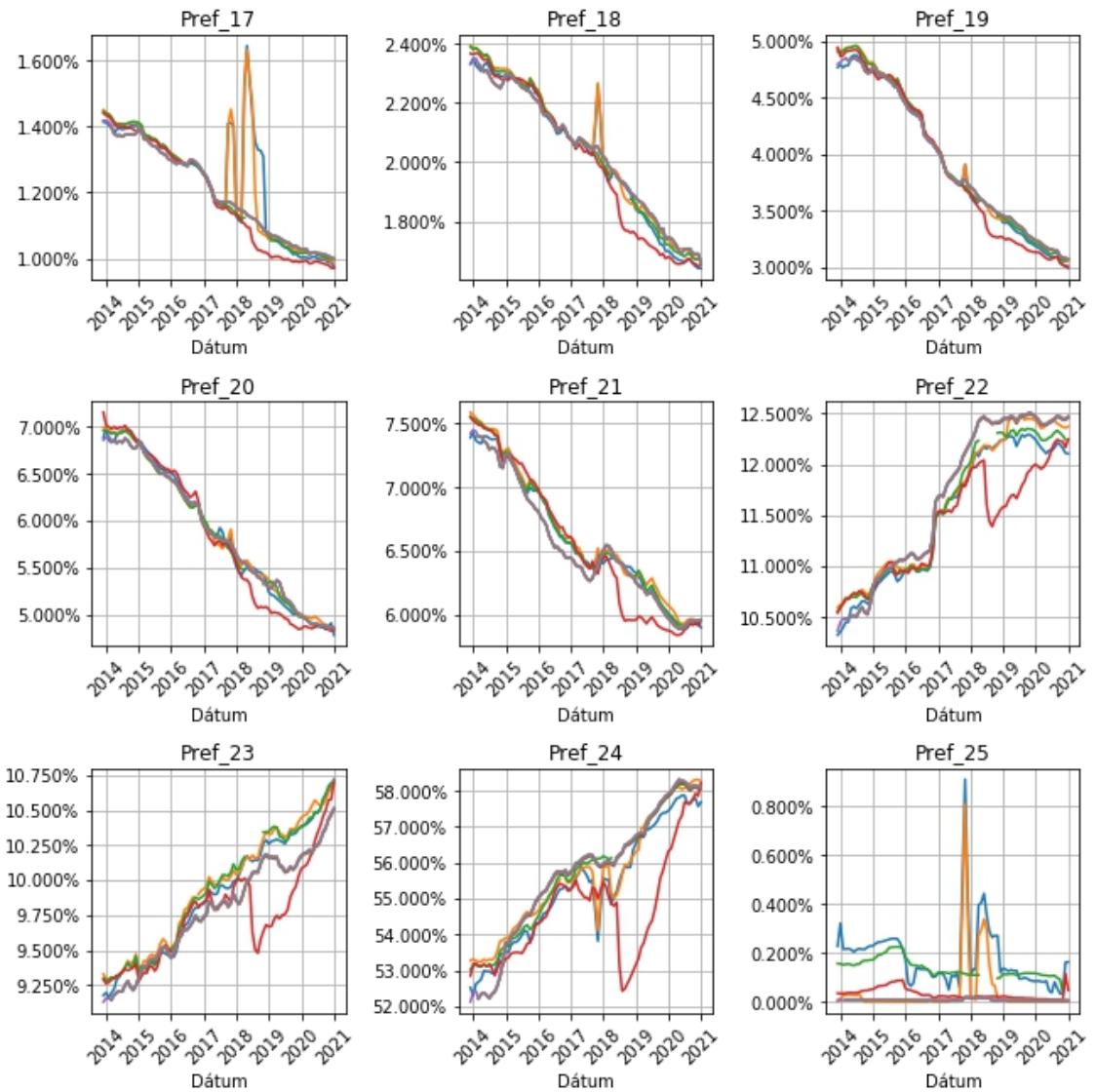
Az 5.2.6. ábra pontosan azt mutatja, amit vártunk, teljesen össze-vissza van és semmilyen hasonlóság nem figyelhető meg. Viszont itt látszik, hogy a sydney eltérést 2018-tól kezdve az az 50 000 meghirdetett 32 hosszú bejegyzés adja.

Ezeket az értékeket lenormalizáltam a teljes darabszámról, mert kíváncsiak voltunk a változásokra az egészhez képest. Így ezek egyértelműen alátámasztják a kezdeti feltételezésünket. Az előzőekből ugyanis nem lehetett látni, hogy a hosszú prefixek mennyisége csökkenne darabszámról.



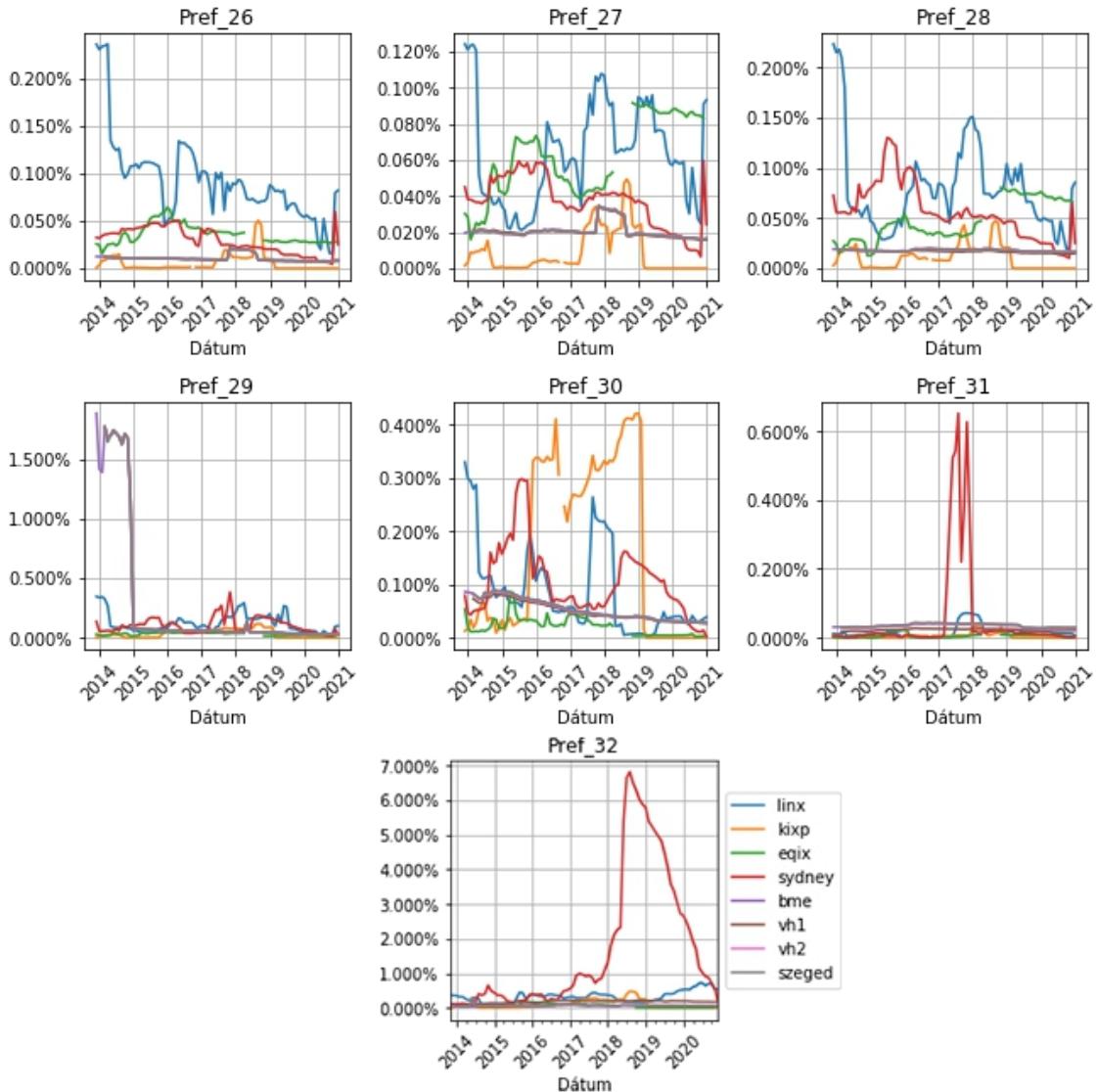
5.2.7. ábra A 8-16 hosszú FIB, RIB bejegyzések változása hosszonkénti bontásban, normalizált idősrögzítés grafikonon, jelmagyarázat az 5.2.9. ábra-n

Az 5.2.7. ábra ténylegesen bizonyítja, hogy a rövid prefixek mennyisége csökken az összes prefixek mennyiségéhez képest, akár harmadára is. Ami érdekes, hogy most látszólag több van a FIB-ekben, mint a RIB-ekben, de az valójában azt jelenti, hogy nagyobb részét teszi ki egy FIB-ben az adott hossz, mint a RIB esetében, mivel a FIB kevesebb bejegyzést tartalmaz.



5.2.8. ábra A 17-25 hosszú FIB, RIB bejegyzések változása hosszonkénti bontásban, normalizált idősortos grafikonon, jelmagyarázat az 5.2.9. ábra-n

Az 5.2.8. ábra-n lehet látni, hogy a 22, 23, 24 hosszú prefixek mutatnak csak növekvő tendenciát. Ez a három hossz az összes prefix közel 80%-át teszi ki.



5.2.9. ábra A 26-32 hosszú FIB, RIB bejegyzések változása hosszonkénti bontásban, normalizált idősoros grafikonon

5.3 Specifikusabb prefixek száma

A fogalmaknál a 2.4.1 fejezetben van a jelentése leírva a specifikusabb prefixeknek. A keresésük lineáris módszerekkel természetesen nem megvalósítható, ezért a 2.4.2 fejezetben bemutatott fába kellett rendezni az adatokat.

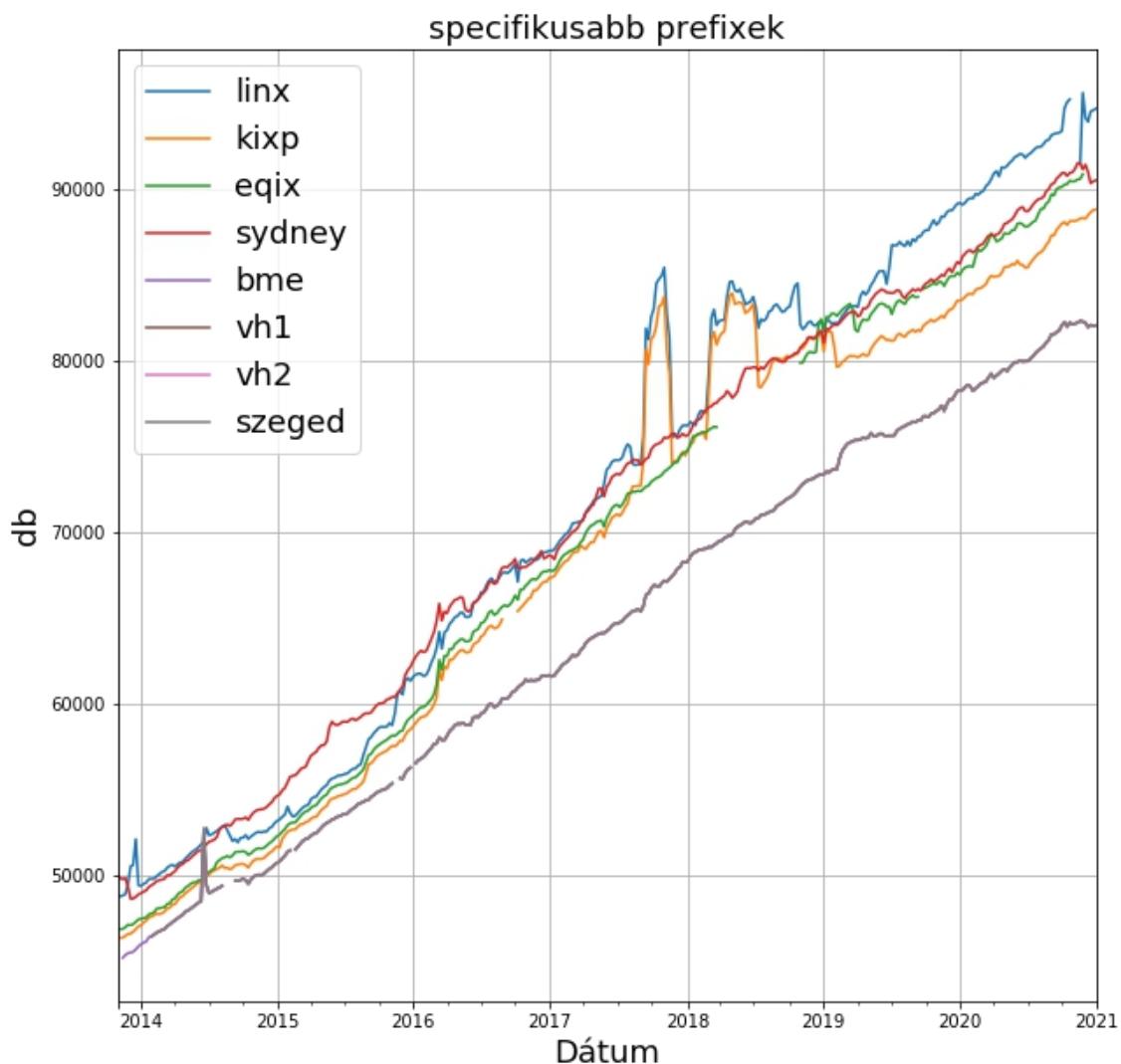
Az alap fa struktúrát annyival kell kiegészíteni, hogy tudja magáról hogy a fa egy adott csomópontja, hogy létező prefix-e vagy sem. Egy üres gyökér elemből indultam ki és rögtön be is lehet szűrni bele az adatokat, nincs szükség előre generálni a fát. A fa így már feltöltés közben azonnal tudja számolni a specifikusabb prefixeket, ugyanis egy új elem a fába való beszúrásakor rögtön meg is vizsgálom, hogy egy létező csomópontron megyek-e épp a beszúrás közben. Amennyiben igen, akkor abban a

csomópontban eltárolom, hogy egyel több létező csomópont őse, a csomópont tudja majd, hogy hány létező prefix van még alatta.

Végül egy pre-order fa bejárással összeadtam az egyes csomópontokban eltárolt gyermeket számát, és így megkaptam, hogy hány darab specifikusabb prefix van összesen az adott táblában.

A FIB-ben lévő bejegyzések közül a specifikusabb prefixek száma 45 000-ről 82 000-re nőtt a vizsgált időszakban, ami 82%-os növekedés hét év alatt.

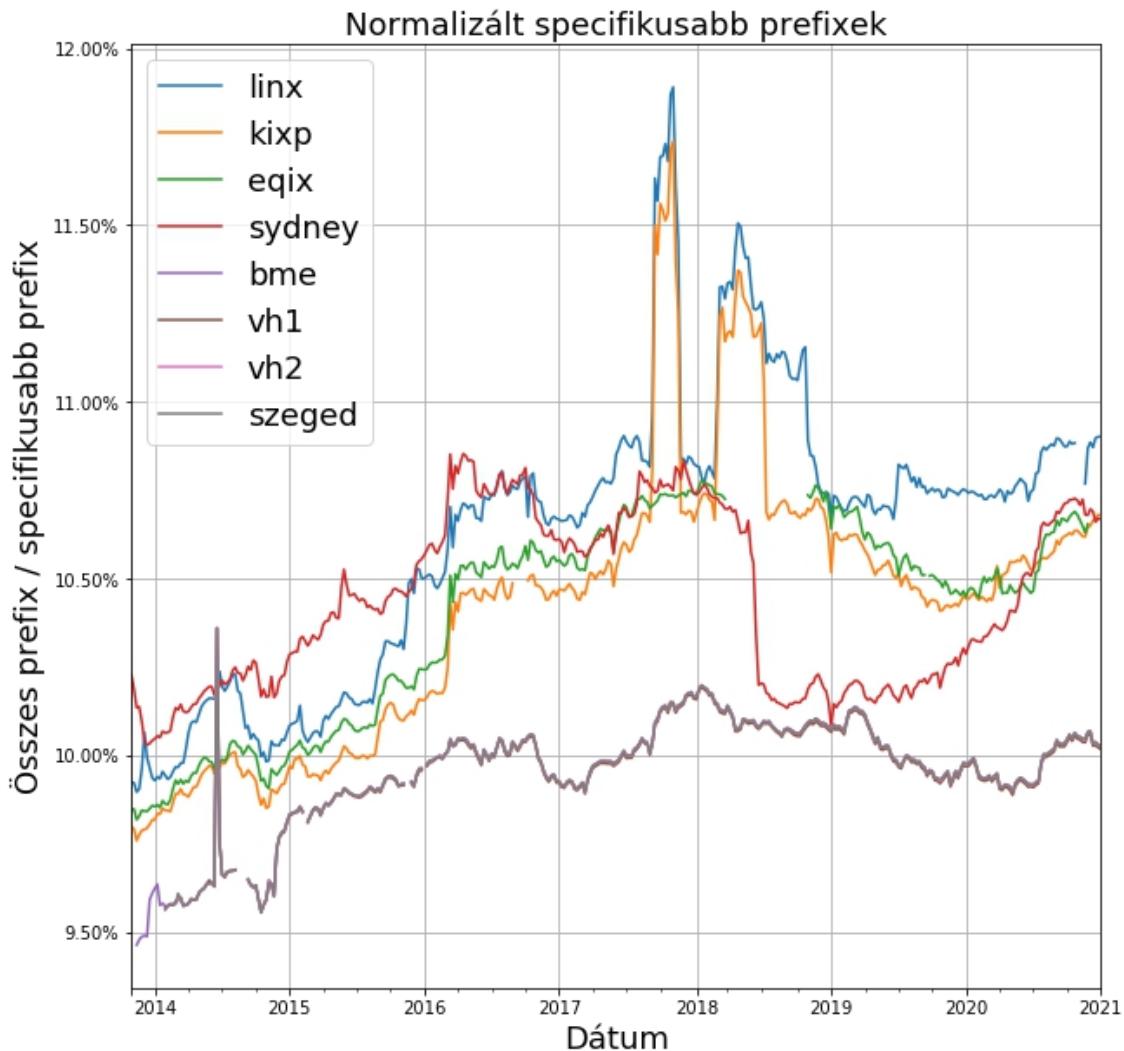
A RIB-ben lévő bejegyzések közül a specifikusabb prefixek száma 46 000-ről és 50 000-ről 89 000-re és 94 000-re nőtt a vizsgált időszakban, ami 93, 88%-os növekedésnek felel meg hét év alatt.



5.3.1. ábra Specifikusabb prefixek száma idősoros grafikonon

Az 5.3.1. ábra két dolgot azonnal elárul a számunkra. Egyszerűen a FIB grafikon ugyan olyan egymásra illő mint a bejegyzés szám esetében, valamint ugyan úgy meredeken emelkedik és enyhén lehajlik a vége. Másrészt pedig a RIB-ek ugyan úgy viszonyulnak hozzá, vagyis felette vannak és egyre jobban távolodnak. Itt is meglehet figyelni, hogy a bizonyos változások minden a 8 adatforrásnál meglátszanak.

A csv fájlban észrevettem, hogy az összes prefix száma és a specifikusabb prefixek száma ugyan az az érték csak egy nullával kisebb ezért ezt is lenormalizáltam a teljes darabszámmal.



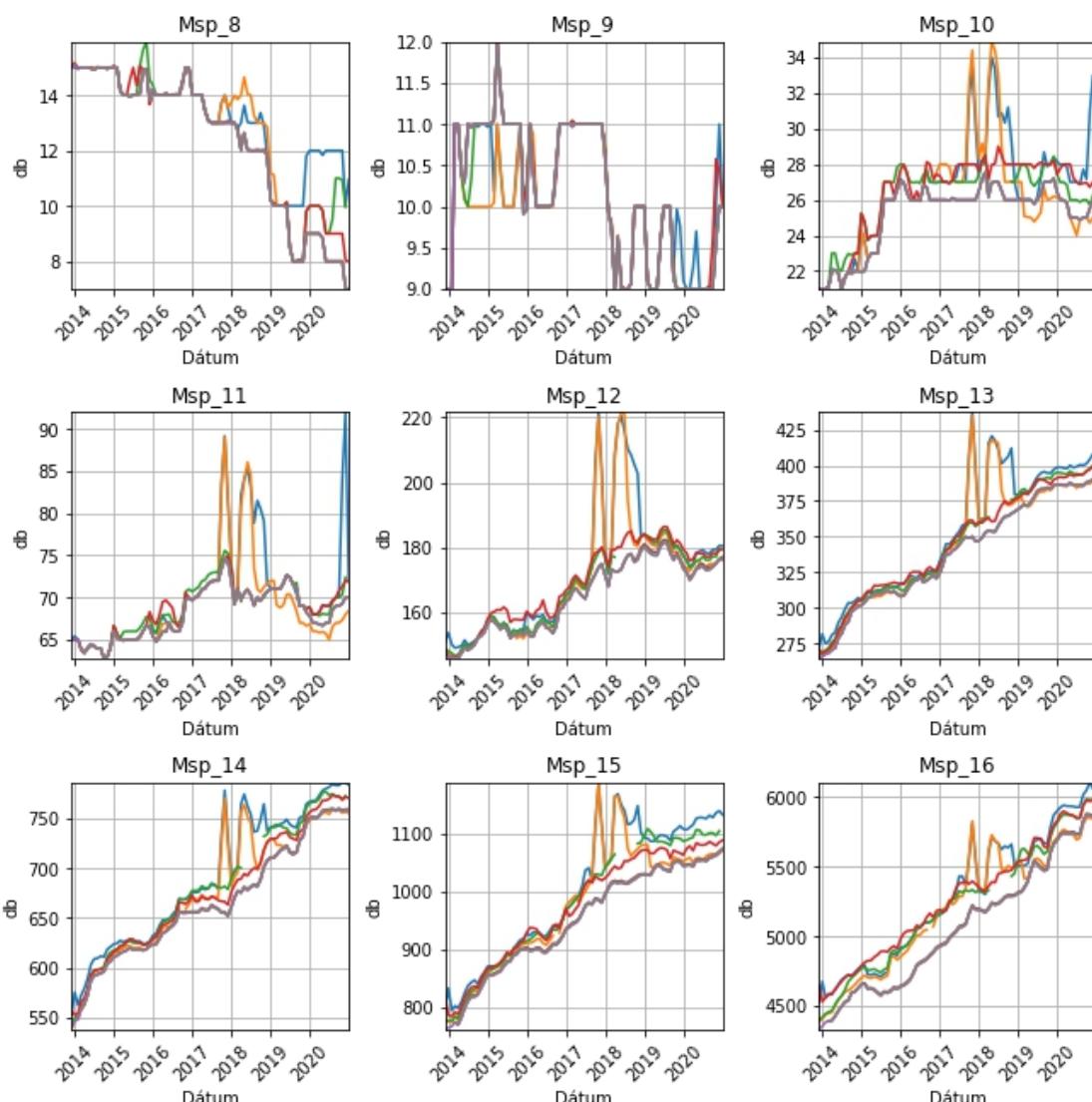
5.3.2. ábra Normalizált, specifikusabb prefixek ábrázolása idősoros grafikonon

Az 5.3.2. ábra a normalizált értéket mutatja, amin jól megfigyelhető a 10% körüli érték. Ugyan akkor jelentős változások vannak a RIB-ek némelyikében, de ezek csak időszakosak voltak.

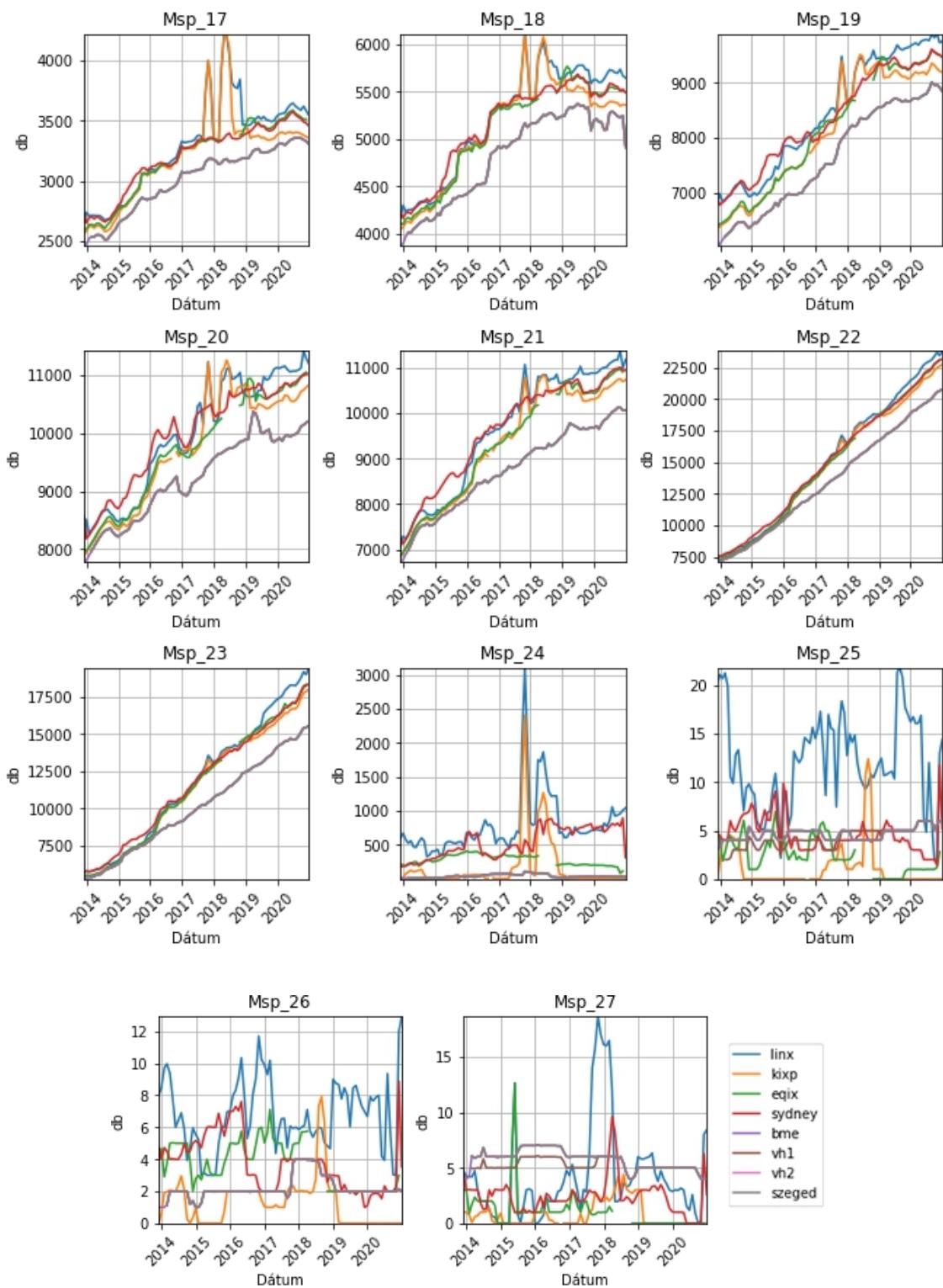
A specifikusabb prefixek mennyiségről is elmondható, hogy jó felső becslést ad a FIB-ek re nézve. És a vizsgálat alapján egy fél százalékos eltéréssel korrigálva még jobb eredményeket kaphatunk.

5.4 Specifikusabb prefixek száma hosszonkénti bontásban

Itt az előző feladatot kellett egy minimális eltéréssel végrehajtanom, a fa felépítése teljesen azonos az összeszámolásban van egy apró különbség. A végén a fa bejárásnál nem egy összegre van szükség, hanem a szintenkénti összegre. A fa tulajdonságai miatt minden egyforma hosszú prefix ugyan arra a mélységi szintre kerül így, könnyen összetudtam szedni, hogy egy adott szinten, prefix hossznál hány specifikusabb bejegyzés található.



5.4.1. ábra A 8-16 hosszú specifikusabb prefixek száma idősoros grafikonon jelmagyarázat az 5.4.2. ábra-n

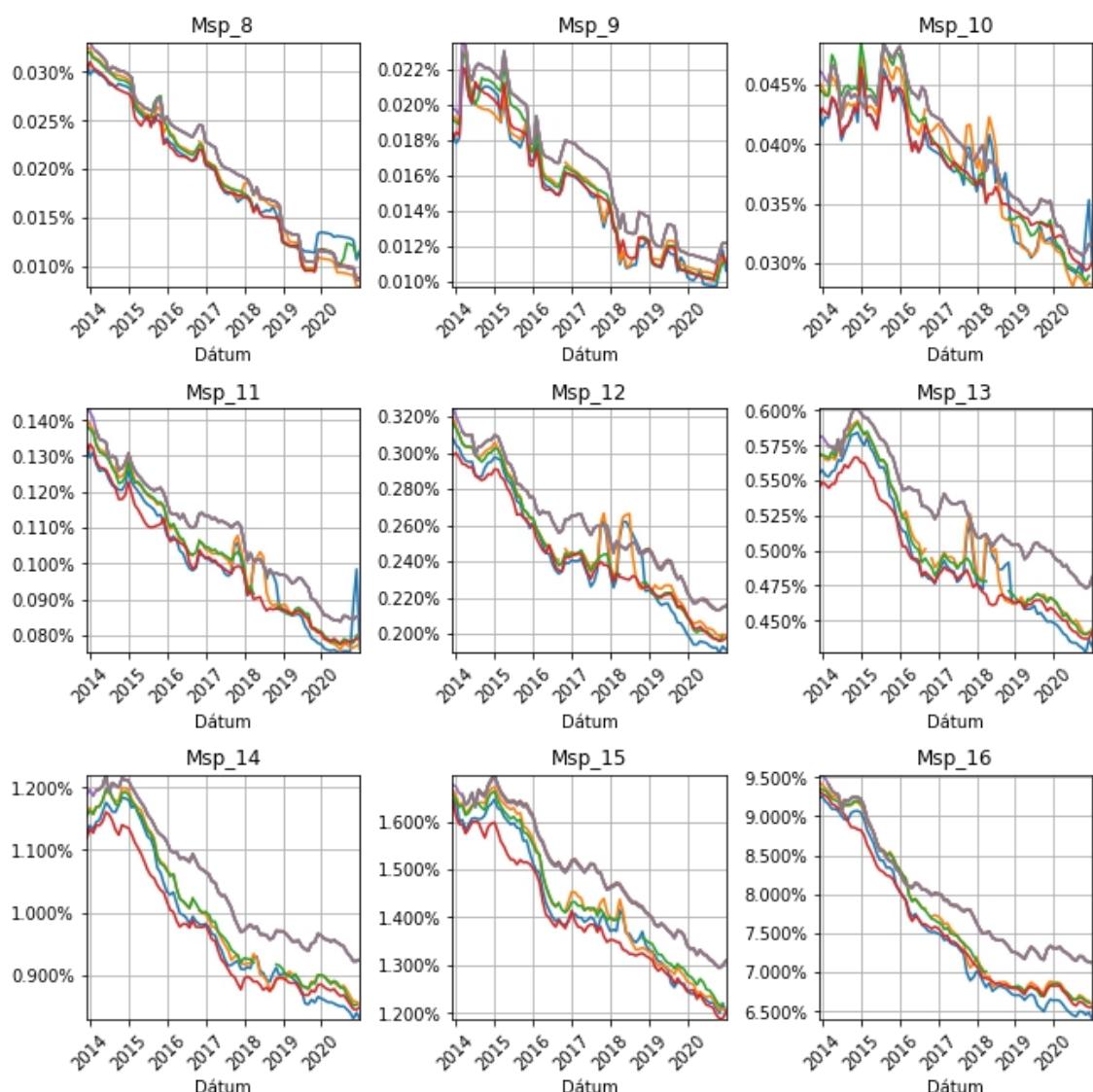


5.4.2. ábra A 17-27 hosszú specifikusabb prefixek száma idősoros grafikonon

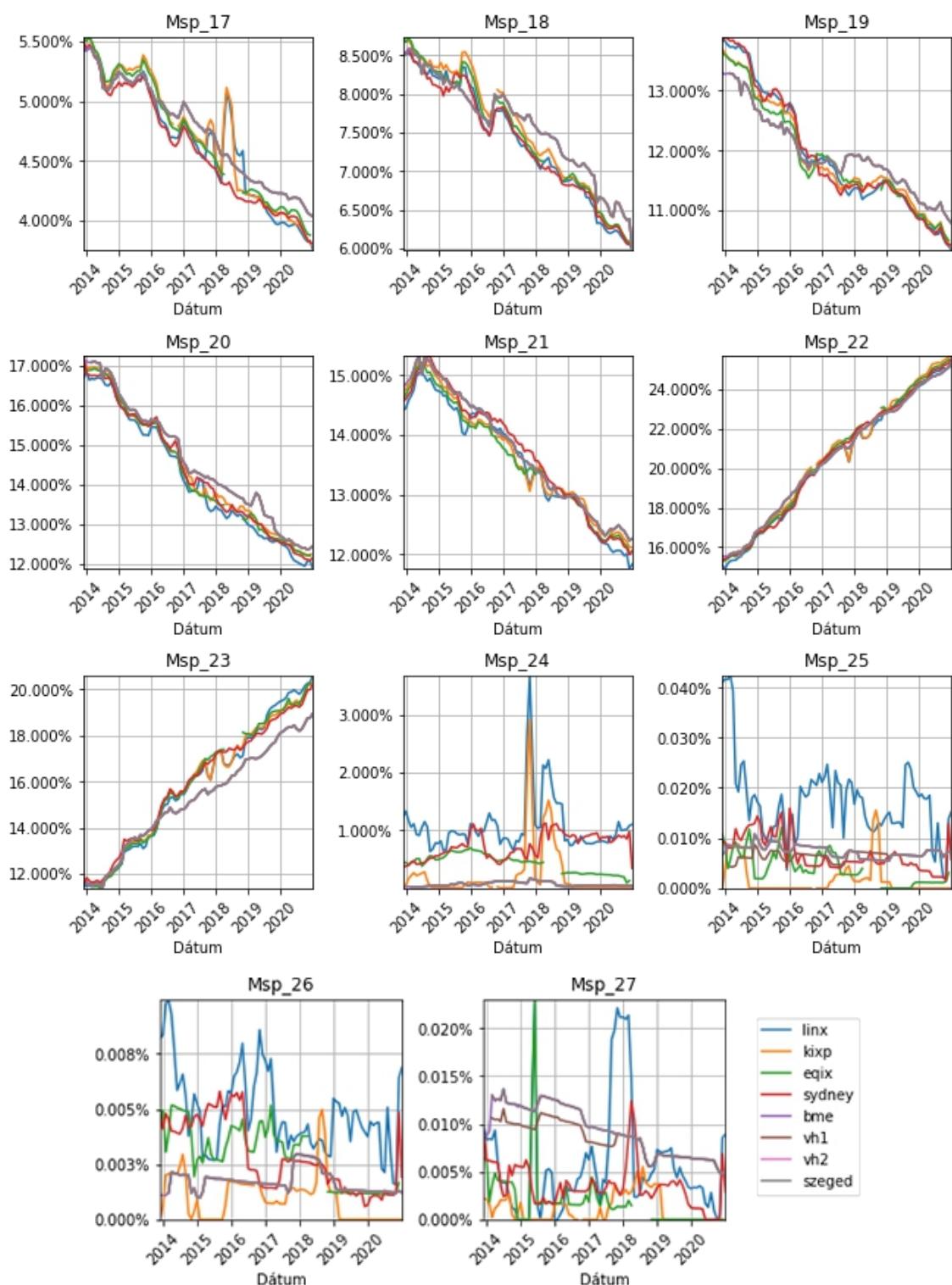
Érdekes megfigyelni az 5.4.1. ábra 5.4.2. ábra jól mutatja, hogy nem a nagy tartományt lefedő prefixeknek van sok specifikusabb prefixük hanem a jóval hosszabb

prefixeknek van inkább több. Ebbe természetesen az is közre játszik, hogy azokból sokkal több is van. Ami nem meglepő, hogy itt nem a 24 hosszú prefixek vezetnek, hanem a nála valamivel rövidebbek inkább a 18-23 tartomány. Itt is megfigyelhető ugyan az a FIB-RIB tendencia, hogy egyre jobban távolodnak egymástól. Csak 27 hosszig rajzoltam ki a grafikonokat mivel utána specifikusabb prefix gyakorlatilag nem volt.

Ezeket a diagramokat is lenormalizáltam az összes specifikusabb prefix számával, hogy lássam, milyen mennyiségen járulnak hozzá a teljes melyik hosszok, és hogy hogyan változnak.



**5.4.3. ábra A 8-16 hosszú specifikusabb prefixek normalizált változása idősoros grafikonon
jelmagyarázat az 5.4.4. ábra-n**



5.4.4. ábra A 17-27 hosszú specifikusabb prefixek normalizált változása időosoros grafikonon

Az 5.4.4. ábra tanulsága, hogy a 18-23 hosszúak adják a specifikusabb prefixek 69%-ról a 86%-át mégis, csak a 22 és a 23 hosszúak nőttek az egészhez képest.

A FIB-ek így nézve is teljesen egyformák a RIB-ek különbségei pedig itt is megfigyelhetők. A rövid prefixek, amiből kevesebb van, jobban egybeesnek, mint a hosszabbak, ahogy az korábban is látható volt.

A RIB-ek darabszámai alapján itt is jól becsülhető felülről a FIB-ek specifikusabb prefixek darabszámai.

5.5 Hirdetési tartomány

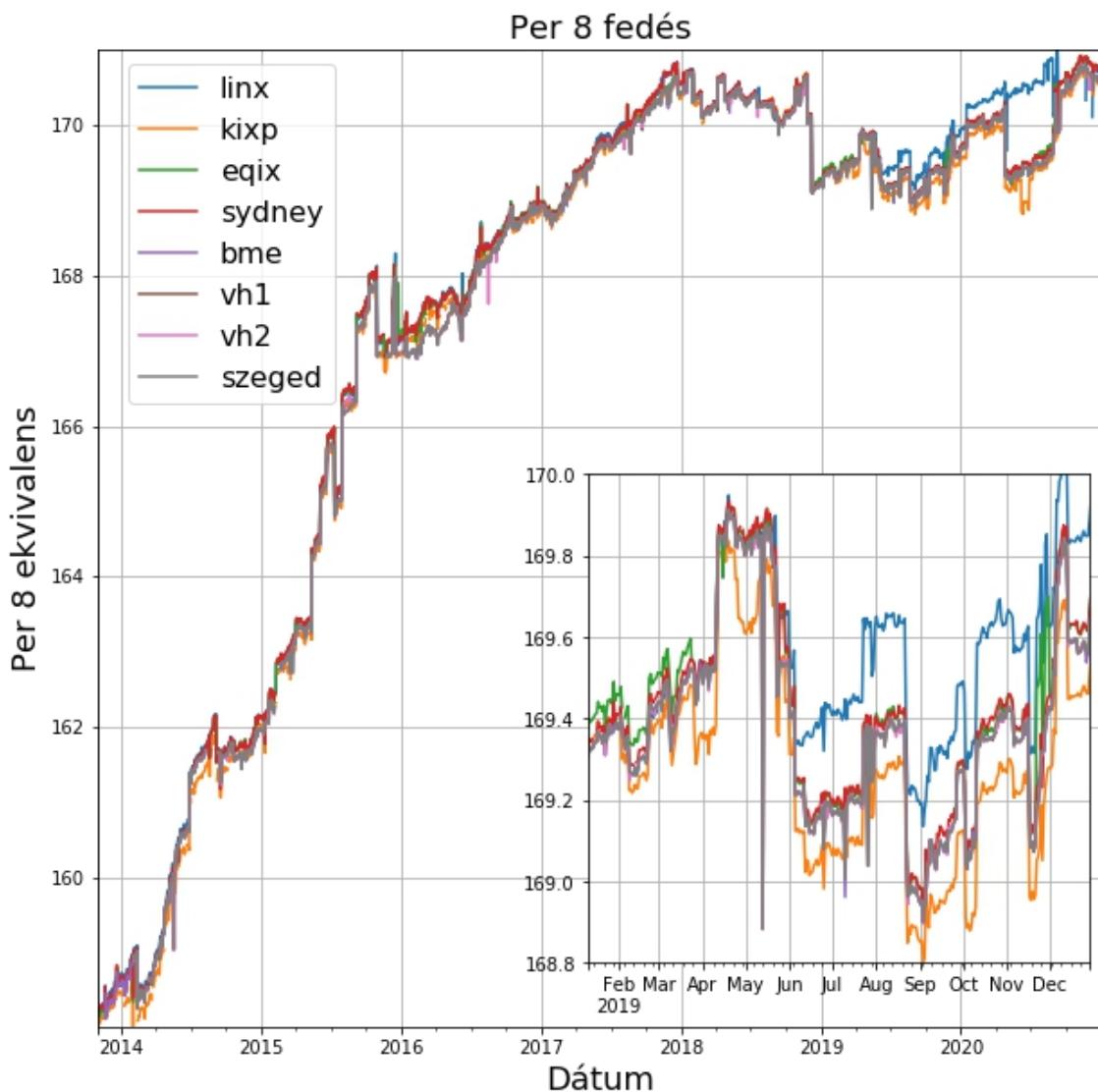
A hirdetési tartomány méretének megszámolása nem triviális feladat, több nehezítő körülmény is van. Egyrészről a táblákban különböző méretű bejegyzések vannak, amik nem ugyan akkora tartományt fednek le pl. a 8 hosszú prefixek 16 millió címet takar, míg egy 24 hosszú prefix csak 255 címet. Másrészről az előbb össze számolt specifikusabb prefixek a teljes tartomány lefedéséhez nem adnak hozzá így azokat ki kell szűrni.

A hirdetési tartományokat az elfogadott módszer szerint „/8 ekvivalens” formában szokás megadni, vagyis hogy hány darab „/8” hosszúságú prefixnek felel meg. Az összegben a 8 hosszú prefix az 1 értékkel szerepel a 9 hosszú $\frac{1}{2}$, a 10 hosszú pedig $\frac{1}{4}$ értékkel járul hozzá az összeghez.

A hirdetési tartományról tudjuk, hogy nem a teljes IPv4 tartomány címezhető a különböző privát címtartományok, a local host tartománya, vagy a nem kiosztott, tesztelési célokra fenntartott tartományok. Ezek a tartományok legalább 35 db „/8 ekvivalens”-nek felelnek meg, tehát a hirdetési tartomány legfeljebb 220 lehet. [33] [34]

A hirdetési tartomány kiszámítására is a prefix fát használtam. Itt meg kellett úgy számolnom a prefixeket hogy az átfedő tartományok csak egy szer szerepeljenek és a megfelelő prefix mérteket rendlejem hozzájuk. Ehhez egy módosított mélységi bejárást használtam fel, ami valójában nem a fa aljára ment le, hanem csak az első létező prefixet tartalmazó csomópontig. Így az átfedő tartományok kiszűrésre kerültek. A mérteket pedig hozzá rendelni az adott csomópont mélységével tudtam hozzá venni.

A hirdetési tartomány 158-ról 170,8-re nőtt a vizsgált időszakban, ami 8%-os növekedés öt és fél év alatt. Ez 12 „/8 ekvivalens”, ami 214,4 millió új IP cím.



**5.5.1. ábra IPv4 hirdetési tartomány fedés ábrázolva idősoros grafikonon kiegészítő ábrán
2019-es év kiemelve**

Az 5.5.1. ábra-n látható, hogy a FIB-ek nagyon együtt vannak, mint az összes diagramon láthattuk. A linx (kék) teljesen elszakadt a többitől és a trendet is se nagyon követte, de 2020 második felére vissza ért a többihez. Érdekes, hogy többször is előfordult, hogy csökkent a meghirdetett tartomány, például 2019 május és június között 8 millió IP címmel volt kevesebb meghirdetve de 2020 közepén is vissza esett közel 16 millió címmel.

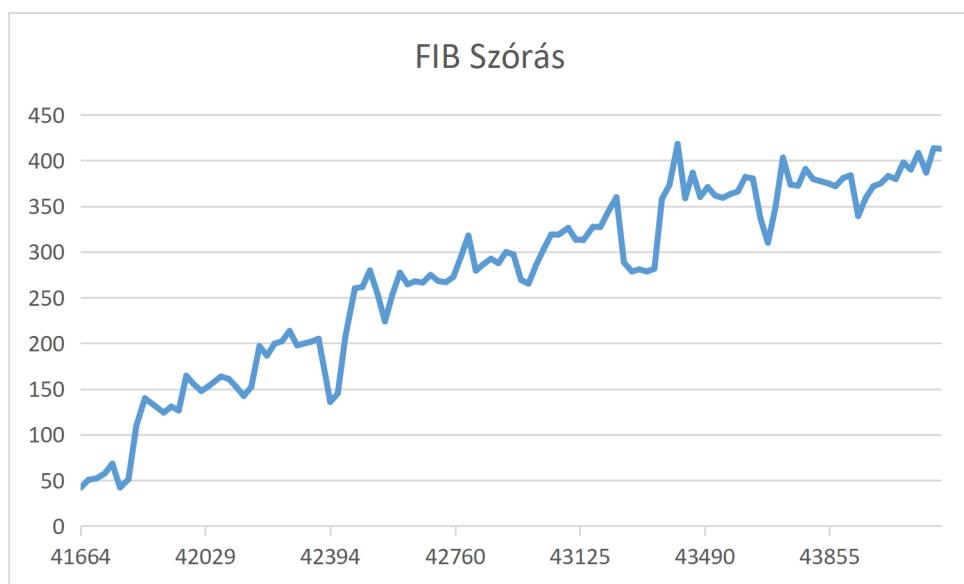
Jól látható, hogy 2016 után lelassult a növekedés és 2018-ban meg is állt azóta nagyjából ugyanannyi cím van meghirdetve. 2014-16 között új 164 millió IP cím került meghirdetésre, ami a teljes mennyiséget 75%-a majd az ezt követő 2 évben még a maradék 50 millió cím.

Ezek alapján azt látjuk, hogy a 2018 után hozzáadott prefixek nem új címek, hanem a meglévőek feldarabolódása, újra hirdetése.

Az 5.5.1. ábra kinagyított részén látható, hogy a nagy mozgások egyszerre történnek bennük bár a felvett különbséget megtartják. Érdekes, hogy a két földrajzilag legtávolabbi (eqix, sydney) ix jobban közelít a FIB-ekhez mint a linx vagy a kixp. A fejezet befejezésében megfogalmaztam, hogy az elvi maximum 220 „/8 ekvivalens” ebből csak 171 van jelenleg használva, ami még lehetőséget biztosít további növekedésre a jövőben.

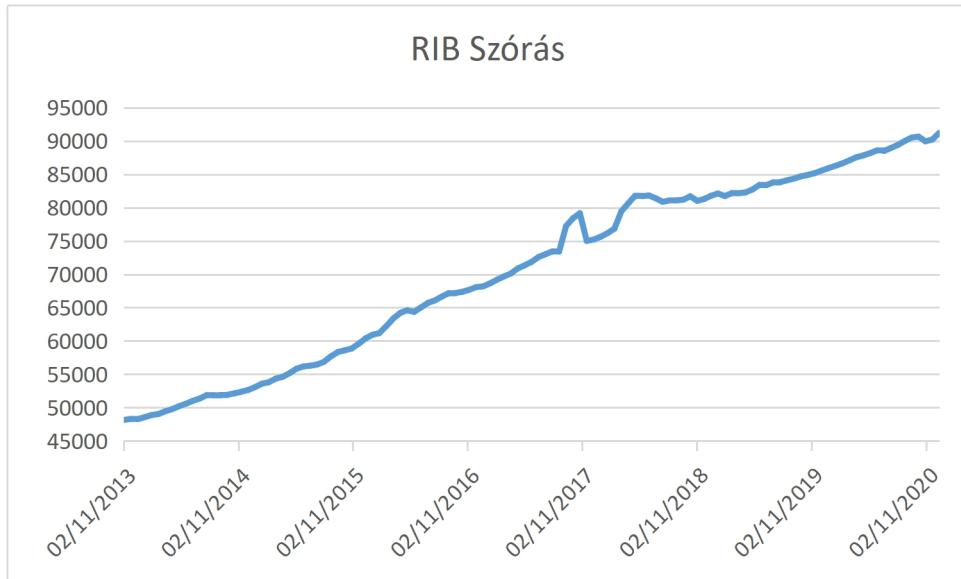
5.6 Szórások

A tapasztalati szórás kiszámításhoz a 2.5.1 fejezetben ismertetett képletet használtam. Itt külön-külön számoltam a szórásokat a FIB illetve a RIB mintákra. Egy másra illesztettem a mintákat és napok alapján vettettem a szórását, majd kapott értékekből kiszűrtem a kiugró elemeket.



5.6.1. ábra FIB teljes bejegyzés szám szórása

Az 5.6.1. ábra-n érdemes megfigyelni, hogy ugyan az a trend látható, mint amit a korábbi fejezetekben is meg lehetett figyelni. Három jól elkülönülő részre oszlik egy meredekebb emelkedő szakasz 2016-ig egy enyhén emelkedő szakasz 2016-2018 között és egy plató szakasz 2018-tól kezdve. A négy darab teljes BGP router szórása a várakozásoknak megfélően nem jelentős a teljes bejegyzés számhoz viszonyítva.



5.6.2. ábra RIB teljes bejegyzés szám szórása

Az 5.6.2. ábra-n a szórás már a vizsgált időszak elején is két nagyságrenddel több, mint a FIB-ek esetén volt. Itt a korábban megfigyelt hármas felosztás nem látható.

A

RIB-ek közötti szórás viszont jelentős.

5.7 Korrelációs elemzés

A korreláció definícióját a 2.5.2 fejezetben leírtam. A megvalósítás során nem írtam meg a korreláció számítást, hanem a *python numpy* [35] könyvtárral oldottam meg. A korrelációval arra voltunk kíváncsiak, hogy melyik RIB hasonlít legjobban a FIB-ükre. A kiszámításához párba állítottam minden egyik FIB-et minden egyik RIB-el és így kaptam 4x4 értéket minden a 67 vizsgált metrikára. Az elkészült táblázatokat szín szerint három részre osztottam zöld, ami nagyon hasonlít, sárga, ami közepesen hasonlít és piros, ami legkevésbé hasonlít. A színeket relatív rendeltem hozzájuk nem adtam meg neki határértéket, hogy honnantól esik melyik osztályba.

Bejegyzés szám korrelációs mátrixa MSP szám korrelációs mátrixa

	linx	kixp	eqix	sydney		linx	kixp	eqix	sydney
bme	0,9314	0,8988	0,9512	0,9512	bme	0,9489	0,8955	0,9530	0,9530
szeged	0,9405	0,9252	0,9754	0,9666	szeged	0,9430	0,9157	0,9757	0,9535
vh1	0,9350	0,9319	0,9800	0,9642	vh1	0,9391	0,9212	0,9803	0,9498
vh2	0,9396	0,9280	0,9764	0,9663	vh2	0,9426	0,9181	0,9768	0,9531

5.7.1. táblázat Bejegyzés és MSP darabszám korrelációs mátrixa

Ahogy azt az 5.7.1. táblázat-ról leolvashatjuk az eqix közelíti meg legjobban a FIB-eket a vizsgált RIB-ek közül. Bár korábban azt láttuk, hogy ezekre az értékekre a kixp van a legközelebb a FIB-ekhez mégis azt lehet látni, hogy azzal hogy a mennyiségi változásokat jobban követi akár egy nagyobb különbséggel az korrelációs szemszögből jobban hasonlít mintha kisebb különbséggel, de lazábban követné. Érdekes, hogy a vh1 és az eqix hasonlít a legjobban egymásra 0,98-as korrelációs együtthatóval minden vizsgált esetben. A linx ami pedig földrajzilag a legközelebb van kevésbé egyezik.

Kiválasztottam 8,16,24 hosszú prefixeket, hogy azoknak is nézzük meg korrelációs tábláit. Azt várjuk tőlük, hogy a 8,24 hosszúak jobban hasonlítanak egymásra majd, de más ok miatt. A 8 hosszúak, mert kevés van belőle és azok is ugyan akkor változnak, 24 hosszúak, viszont mert nagyon sok van belőlük és egy-egy kis különbség nem fog akkor súlyval szerepelni benne. A 25 hosszú prefixetől viszont azt várjuk, hogy nulla közeli értéket adjanak.

8 hosszú prefixek

	linx	kixp	eqix	sydney
bme	0,9129	0,7765	0,8571	0,8571
szeged	0,8953	0,8463	0,9096	0,8559
vh1	0,8871	0,8552	0,9227	0,8447
vh2	0,8930	0,8498	0,9140	0,8536

16 hosszú prefixek

	linx	kixp	eqix	sydney
bme	0,8959	0,8579	0,8895	0,8895
szeged	0,8393	0,8835	0,8683	0,8457
vh1	0,8291	0,8934	0,8633	0,8358
vh2	0,8347	0,8890	0,8662	0,8401

24 hosszú prefixek

	linx	kixp	eqix	sydney
bme	0,9642	0,9009	0,9498	0,9498
szeged	0,9352	0,9283	0,9745	0,9461
vh1	0,9289	0,9350	0,9792	0,9413
vh2	0,9340	0,9312	0,9756	0,9452

25 hosszú prefixek

	linx	kixp	eqix	sydney
bme	0,3426	0,1050	-0,2091	-0,2091
szeged	0,2440	0,2852	-0,2157	-0,1674
vh1	0,2334	0,2768	-0,2121	-0,1780
vh2	0,2351	0,2843	-0,2183	-0,1740

5.7.2. táblázat A 8,16,24,25 hosszú prefixek korrelációs táblázatai

Az 5.7.2. táblázat várakozásoknak megfelelően alakul, valóban a 8 és 24 hosszúak jobban hasonlítanak, mint a 16 hosszú. Ezek esetében is megfigyelhető, hogy az eqix-el vannak a legközelebb egymáshoz. A 25 hosszúknál, ahogy korábban mutattam szinte véletlenszerű, hogy mennyi prefix van és azt a korreláció is jól vissza adja a 0,3 alatti értékekkel, ami az eqix és sydney esetén negatívba is fordul.

6 Next-hop elemzés

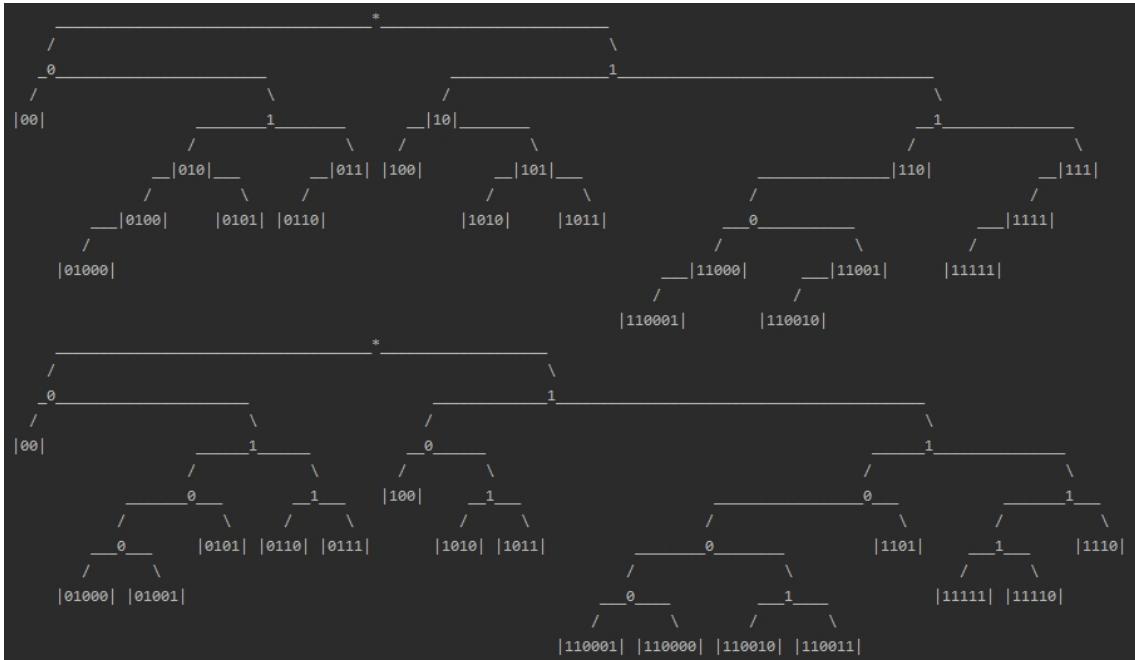
Ebben a fejezetben egy teljesen másik szemszögből fogom megvizsgálni, mint eddig, itt a next-hop-okon lesz a hangsúly az eddigi prefixekkel ellentében. Két új algoritmust vezetek be és másik fajta szemlélet móddal fogom megvizsgálni.

6.1 Leaf-Push megvalósítás

A kiindulási alap a prefix fa, ami egy olyan nem teljes bináris fa, aminek nem minden csomópontja hordoz információt. 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 elő fordulhat 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 tolta csomópont természetesen fele akkora tartományt fed majd le a címtérben, és ennek megfelelően kell majd kezelni, amikor számbavételre kerül.



6.1.1. ábra Teszt prefix és Leaf-Pushed fa

A 6.1.1. ábra egy kis teszt fa látható, amit az algoritmus fejlesztése közben használtam az átláthatóság miatt. Szerepel benne az összes érdekes csomópont kombináció. 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 „|110|” jelzett csomópont, aminek egy gyereke van. Így az algoritmus egy szinttel lejebb tolta és ennek megfelelően megváltoztatta az új prefixet, amivel elérhető.

6.2 Futáshossz-kódolás

A futáshossz-kódolásra nagy 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 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. [34]

```
[ '0', 16777216]
[ '195.66.224.175', 256] 1.0.0.0/24 195.66.224.175
[ '0', 768]
[ '195.66.226.39', 256] 1.0.4.0/24 195.66.226.39
[ '195.66.226.39', 256] 1.0.5.0/24 195.66.226.39
[ '195.66.226.39', 256] 1.0.6.0/24 195.66.226.39
[ '195.66.226.39', 256] 1.0.7.0/24 195.66.226.39
[ '0', 2048]
```

6.2.1 kódrészlet balra a futáshossz kódolt jobbra az eredeti

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 6.1.1. 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.

6.3 Kontingencia táblázatba rendezés

Minden két adatforrást a futáshossz-kódolás párba állítottam, hogy össze hasonlítsam őket. 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 táblázatban 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.

Perem eloszlás			
bme		szeged	
0.3352627	195.111.97.108	0.3352662	195.111.97.108
0.0000951	195.111.97.49	0.0000886	195.111.97.49
0.0000167	195.111.97.8	0.0000167	195.111.97.8
0.0000078	195.111.97.13	0.0000078	195.111.97.13
vh1		vh2	
0.2330814	80.239.195.56	0.3352498	195.111.97.108
0.0671979	193.188.137.175	0.0000926	195.111.97.49
0.0149223	149.11.10.9	0.0000167	195.191.97.254
0.0090127	83.97.88.81	0.0000078	195.111.97.8
eqix		kixp	
0.1511515	206.126.236.234	0.3356595	196.223.21.74
0.0877766	206.126.237.175	0.0000196	196.223.21.11
0.0266520	206.126.238.92	0.0000176	196.223.21.100
0.0233977	206.126.236.47	0.0000104	196.223.21.109
linx		sydney	
0.1256231	195.66.224.175	0.1120348	45.127.172.20
0.0363429	195.66.226.39	0.0855215	45.127.172.78
0.0244270	195.66.226.99	0.0381954	45.127.172.196
0.0219800	195.66.226.33	0.0253805	45.127.173.24

6.3.1. táblázat Peremeloszlások a hozzájuk tartozó next-hop címekkel

A 6.3.1. táblázaton jól megfigyelhető, hogy a bme, szeged, vh2 és kixp esetén az összes IP cím harmada ugyanarra az egy cím felé továbbítódik. Ez nem meglepő, mivel a felépítés miatt hasonló eredményre számítottunk a FIB-ek esetén. A vh1 viszont jobban szét osztja a forgalmat a kapcsolatai között a RIB-ekben látottakhoz hasonlóan.

6.4 Entrópia elemzés

Egy adott FIB vagy RIB entrópiája a kontingencia táblázat segítségével könnyen kiszámolható a 2.5.4 fejezetben ismertetett módon. Az összes táblának kiszámoltam négy év entrópiját és maximum entrópiját. A vizsgálattól azt várjuk, hogy az entrópia értéke nem lesz túl nagy, mivel sok a nem routolt cím és a routolt címek nagy része is ugyan arra címre továbbítódik.

	Entrópia				Maximum entrópia			
	2017	2018	2019	2020	2017	2018	2019	2020
bme	0.9037	0.9110	0.9158	0.9226	6.7415	6.7944	6.8580	6.9069
szeged	0.9036	0.9110	0.9156	0.9226	6.7004	6.7549	6.8202	6.8455
vh1	1.3675	1.2802	1.3429	1.4007	8.2192	7.8704	7.8074	7.8329
vh2	0.9036	0.9112	0.9158	0.9227	6.9542	6.9773	7.0553	7.0768
eqix	1.7768	1.9917	1.8897	1.7251	7.3837	7.4263	7.5078	6.9773
kixp	1.1575	1.2778	0.9272	0.9216	5.4919	4.6439	5.2095	4.2479
linx	2.2054	2.4964	2.0155	2.1137	9.1085	8.9658	8.7142	8.7715
sydney	1.4948	1.7160	1.6747	1.8395	7.1189	7.3663	7.4094	7.4838

6.4.1. táblázat Entrópia értékek

A 2.5.1. táblázatban megfigyelhetjük, hogy az entrópia értéke valóban alacsony a FIB-eknél, de a RIB-eknél sem sokkal magasabb. A vizsgált időszakba azt látjuk, hogy a FIB-ek entrópiái folyamatosan nőnek, tehát nő a rendszer rendezetlensége. A RIB-ek esetében nem mindegyiknél figyelhető meg egy jellemző trend, ezek az értékek nagyon változékonyak lehetnek. Például sydney folyamatosan nő, addig a kixp, eqix egy fel-le változást csinál.

6.5 Feltételes Entrópia és Kölcsönös Információ

A feltételes entrópiát kiszámoltam minden párra a 2.5.5. egyenlet alapján négy évre visszamenőleg. Innen kivettem a nem routolt címtartományokat. Ezt azért lehet megtenni, mert ha az egyik táblában nem routolható akkor a másikban sem lesz az. A feltételes entrópia számolásának az a célja, hogy megmondja, hogy ha az egyik az IP tartomány x felé küldi, akkor mennyire következtethetünk arra, hogy a másik pedig y felé küldi.

	Feltételes entrópia								
	2017				2018				
	eqix	kixp	linx	sydney		eqix	kixp	linx	sydney
bme	0.0371	0.1818	0.1760	0.0480	bme	0.0353	0.0693	0.0585	0.0459
szeged	0.0370	0.1817	0.1759	0.0480	szeged	0.0353	0.0692	0.0584	0.0458
vh1	0.4796	0.6271	0.6003	0.5062	vh1	0.3779	0.4350	0.3929	0.4037
vh2	0.0370	0.1817	0.1759	0.0479	vh2	0.0354	0.0694	0.0585	0.0460

	2019					2020			
	eqix	kixp	linx	sydney		eqix	kixp	linx	sydney
bme	0.0408	0.0421	0.0508	0.0818	bme	0.0591	0.0349	0.1018	0.0525
szeged	0.0409	0.0420	0.0508	0.0816	szeged	0.0590	0.0348	0.1018	0.0525
vh1	0.4478	0.4663	0.4540	0.4903	vh1	0.5238	0.5112	0.5439	0.5003
vh2	0.0408	0.0420	0.0506	0.0817	vh2	0.0593	0.0349	0.1019	0.0526

6.5.1. táblázat Feltételes entrópia

	H(FIB)			
	2017	2018	2019	2020
bme	0.9036	0.9110	0.9157	0.9036
szeged	0.9036	0.9109	0.9155	0.9036
vh1	1.3675	1.2801	1.3429	1.3675
vh2	0.9036	0.9111	0.9157	0.9036

6.5.2. táblázat H(FIB) Feltételes Entrópia maximum értéke

A 6.5.1. táblázatban azt láthatjuk, hogy az értékek meglehetősen közel esnek a nullához tehát a RIB-ek alapján egész jól lehet a FIB-ek next-hop csoportjaira következtetni. Kivéve a vh1 esetén ahol is a kölcsönös entrópia értéke a legmagasabb, mivel az az egy olyan tábla, ami ténylegesen forgalom továbbítást végez a szolgáltatói felé.

A feltételes entrópia és a kölcsönös információ valójában ugyan azt jelent csak a másik oldalról vizsgálva.

	Kölcsönös Információ								
	2017				2018				
	eqix	kixp	linx	sydney		eqix	kixp	linx	sydney
bme	0.8666	0.7219	0.7277	0.8557	bme	0.8757	0.8417	0.8525	0.8651
szeged	0.8666	0.7219	0.7277	0.8557	szeged	0.8757	0.8417	0.8525	0.8651
vh1	0.8879	0.7404	0.7672	0.8613	vh1	0.9023	0.8451	0.8873	0.8765
vh2	0.8666	0.7219	0.7277	0.8557	vh2	0.8758	0.8418	0.8526	0.8652

	2019					2020			
	eqix	kixp	linx	sydney		eqix	kixp	linx	sydney
bme	0.8749	0.8737	0.8650	0.8340	bme	0.8635	0.8877	0.8208	0.8701
szeged	0.8747	0.8735	0.8648	0.8340	szeged	0.8635	0.8877	0.8208	0.8701
vh1	0.8951	0.8767	0.8889	0.8526	vh1	0.8769	0.8895	0.8567	0.9004
vh2	0.8750	0.8738	0.8652	0.8340	vh2	0.8634	0.8878	0.8208	0.8701

6.5.3. táblázat Kölcsönös információ

A 6.5.3. táblázatban láthatjuk, hogy az értékek elég közel vannak a maximumukhoz, tehát a RIB jelentősen meghatározza a FIB értékeit. Itt is szintén a vh1 kivételével.

7 Összefoglalás

A diplomaterv a fő kérdésre, hogy az Interneten nyilvánosan is elérhető RIB-ek statisztikai elemzése alapján mennyire megbízhatóan következtethetünk a FIB-ek egyes statisztikai jellemzőire? Válaszként azt tudom adni, hogy a vártnál jobban lehet RIB-ek alapján a FIB-ekre következtetni. Persze ezt csak, úgy ha egy kicsit ügyesen állunk neki. Mint azt láttuk a sydney esetében, hogy egyszer csak meghirdettek egy halom 32 hosszú prefixet, ami a BGP szempontjából teljesen érdektelen, de ha csak a növekedést akarnánk vizsgálni, akkor bizony megtévesztő lehet.

A földrajzi lokáció alapján vett legközelebbi RIB sem biztos, hogy a legcélravezetőbb, a londoni IX által biztosított adathalmaz tért el sok esetben, a legjobban a FIB-ektől.

A next-hop-ok hasonlósága teljesen megdöbbentő volt elsőre, persze az alaposabb vizsgálat után érthető lett miért kaptam rá nagyon hasonló eredményt.

Ettől függetlenül minden esetben érdemes lehet több RIB-en párhuzamosan futtatni a kutatást egy optimálisabb végeredmény érdekében. Hisz láthatóvá vált, hogy egy szerencsétlenül vagy épp szerencsésen választott RIB adathalmaz milyen jelentős mértékben tudja befolyásolni a kutatás eredményét.

A diplomatervem készítése közben sok tapasztalatot gyűjtöttem, hogyan nem szabad a Big Data jelegű munkának neki állni. Volt a diplomaterv készítése folyamán egy közel két hetes idő intervallum ahol a számítógépem konkrétan megállás nélkül számolt, adatot formázott. De a két hét végére sikeresen előálltak a szükséges fájlok szóval megérte kivární.

Persze hasznosabb tapasztalatokkal is gazdagabb lettem, sokat mélyült a statisztikai tudásom, érdekes kihívás volt néhány statisztikai képletet papírról programba vinni, még ha használtam is kész matematikai könyvtárak elkészített függvényeit.

A diagramok vizualizációja okozta talán a legnagyobb problémát a diplomaterv készítése során. A matplotlib könyvtárat használtam a diagramok elkészítéséhez a, ami egy nagyon jó könyvtár abból a szempontból, hogy szép letisztult ábrákat lehet vele készíteni és teljesen testre lehet szabni. A szlogenjével viszont teljesen egyet értek, ha

nem akar sok minden tőle az ember, akkor nagyon valóban nagyon könnyű használni, de ha egy kicsit bonyolultabb összetettebb dolgot akar megvalósítani, akkor már kevésbé egyszerű, viszont biztos, hogy van rá mód.

Az eredmények, amiket kaptam szerintem szép összefüggésekre világít rá a témaiban. Érdekes, hogy a RIB-ek között is meg lehetett néhány kulcs dolgot figyelni, ami nagyon érdekes lehet a témaiban érdeklődőknek. Innentől további metrikákat behozni vizsgált területre, már nagyon könnyen lehet. A további kutatásnak a témaiban mindenképp gépi tanuláson alapuló predikciós eljárásnak lehetne tovább menni.

Jelenleg is zajlik ebben a témaiban egy másik jellegű kutatás, ami a táblák tömöríthetőségét vizsgálja, ami szintén bíztató eredményekkel kecsegtet. Az évek alatt egy nagy, összetett feladattá nőtte ki magát.

8 Irodalomjegyzék

- [1] D. B. Péter, „Internet az információs társadalom hajtóereje,” BME, 2017.
- [2] W. Howe, „A Brief History of the Internet,” 2016. [Online]. Available: <http://www.walthowe.com/navnet/history.html>. [Hozzáférés dátuma: 26 Április 2021].
- [3] „CIDR REPORT for 26 Apr 21,” [Online]. Available: <https://www.cidr-report.org/as2.0/>. [Hozzáférés dátuma: 26 április 2021].
- [4] „Az Internet ökoszisztemája és evolúciója,” [Online]. Available: <https://www.tmit.bme.hu/vitmma00>. [Hozzáférés dátuma: 26 április 2021].
- [5] „KIFÜ HBONE,” [Online]. Available: <https://kifu.gov.hu/szolgaltatasok/ikt/halozati/hbone>. [Hozzáférés dátuma: 26 április 2021].
- [6] „BGP Routing Table Analysis Reports,” [Online]. Available: <https://bgp.potaroo.net/>. [Hozzáférés dátuma: 26 április 2021].
- [7] D. J. X. Z. Y. L. D. M. L. W. B. Z. a. L. Z. V. Khare, „Evolution towards global routing scalability,” *IEEE JSAC*, %1. kötet28(8), p. 1363–1375, 2010.
- [8] W. M. S. U. R. B. P. F. O. M. Luca Cittadini, „Evolution of Internet Address Space Deaggregation:Myths and Reality,” *IEEE JSAC*, %1. kötetVOL. 28, %1. számNO. 8, 2010.
- [9] „Pakistan hijacks Youtube,” [Online]. Available: <https://blogs.oracle.com/internetintelligence/pakistan-hijacks-youtube>. [Hozzáférés dátuma: 26 április 2021].
- [10] D. J. P. a. J. S. X. Zhao, „ Routing scalability: an operator’s view.,” *IEEE JSAC*, %1. kötet28(8), pp. 126-1270, 2010.
- [11] J. T. A. K. A. M. a. Z. H. G. Rétvári, „Compressing IP Forwarding Tables:Towards

- Entropy Bounds and Beyond,” ACM SIGCOMM, 2013.
- [12] W. B. Norton, The Internet Peering Playbook, Palo Alto, California: DrPeering Press, 2012.
- [13] G. Huston, „The Death of Transit?,” 2016. [Online]. Available: <https://labs.ripe.net/author/gih/the-death-of-transit/>. [Hozzáférés dátuma: 26 április 2021].
- [14] B. O. Benjamin Wijchers, „Quantitative Analysis of BGP Route Leaks,” Ripe, 2014.
- [15] „BGP Best Path Selection Algorithm,” [Online]. Available: <https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13753-25.html>. [Hozzáférés dátuma: 26 április 2021].
- [16] „Content-Addressable Memory Introduction,” [Online]. Available: <https://www.pagiamtzis.com/cam/camintro/>. [Hozzáférés dátuma: 26 április 2021].
- [17] P. Cooper, „Could 512k be the new Y2K?,” 2014. [Online]. Available: <https://www.itproportal.com/2014/08/13/ebay-amazon-and-linkedin-taken-down-by-huge-worldwide-internet-outage-ipv4-ipv6-bgp-routers-address-exhaustion/>. [Hozzáférés dátuma: 26 április 2021].
- [18] K. A. M. M. S. Z. T. M. Havasy György, Általános statisztika I, Nemzeti Tankönyvkiadó Rt, 1996.
- [19] I. Fazekas, Valószínűségszámítás és statisztika, 2009.
- [20] N. Szilvia, Információelmélet, 2006.
- [21] K. Bertalan, Információelmélet, Eger, 2011.
- [22] M. Balázs, „Az akadémiai internethálózat (HBONE) születése,” Informatikatörténeti Fórum, 2017.
- [23] V. György, „A HBONE története,” 2009.
- [24] M. János, „A HBONE+ projekt áttekintés,” Budapest, 2012.
- [25] „KIFÜ Bemutatkozás,” [Online]. Available: <https://www.kifu.hu/bemutatkozasi.html>.

<https://kifu.gov.hu/rolunk/bemutatkozas>. [Hozzáférés dátuma: 26 április 2021].

- [26] „RIPE Database Query,” [Online]. Available: <https://apps.db.ripe.net/db-web-ui/query?searchtext=as1955>. [Hozzáférés dátuma: 26 április 2021].
- [27] K. Downie, „Mi az az útvonal-tüköröző?,” 2021. [Online]. Available: <https://docs.microsoft.com/hu-hu/azure-stack/hci/concepts/route-reflector-overview>. [Hozzáférés dátuma: 27 április 2021].
- [28] „University of Oregon Route Views Project,” [Online]. Available: <http://www.routeviews.org/routeviews/>. [Hozzáférés dátuma: 26 április 2021].
- [29] „Python urllib dokumentációja,” [Online]. Available: <https://docs.python.org/3/library/urllib.request.html#module-urllib.request>. [Hozzáférés dátuma: 26 április 2021].
- [30] „bgpdump manual oldal,” [Online]. Available: <https://manpages.debian.org/testing/bgpdump/bgpdump.1.en.html>. [Hozzáférés dátuma: 26 április 2021].
- [31] „python pandas,” [Online]. Available: <https://pandas.pydata.org/>. [Hozzáférés dátuma: 9 május 2021].
- [32] „Matplotlib: Visualization with Python,” [Online]. Available: <https://matplotlib.org/>. [Hozzáférés dátuma: 9 május 2021].
- [33] „IPv4 Speciális tartományok,” [Online]. Available: https://en.wikipedia.org/wiki/IPv4#Special-use_addresses. [Hozzáférés dátuma: 9 május 2021].
- [34] „IANA IPv4 Address Space Registry,” 2021. [Online]. Available: <https://www.iana.org/assignments/ipv4-address-space/ipv4-address-space.xhtml>. [Hozzáférés dátuma: 9 május 2020].
- [35] „numpy,” [Online]. Available: <https://numpy.org/>. [Hozzáférés dátuma: 9 május 2021].