



M Ű E G Y E T E M 1 7 8 2

Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Távközlési és Médiainformatikai Tanszék

Pálfi András

BGP HÁLÓZATI KONVERGENCIA AZ INTERNETEN

KONZULENS

Dr. Heszberger Zalán

BUDAPEST, 2011

Tartalomjegyzék

Tartalomjegyzék.....	2
HALLGATÓI NYILATKOZAT	4
Összefoglaló	5
Abstract	6
Bevezetés	7
1. A tartományközi útvonalválasztás felépítése.....	11
1.1 Az Internet útvonalválasztásának architektúrája.....	11
1.2 BGP alapok bemutatása	13
1.2.1 Általános tulajdonságok	13
1.2.2 BGP üzenettípusok.....	14
1.2.3 Update üzenetek küldése, kezelése	15
1.2.4 Az útvonalválasztás és üzenetkezelés folyamata	18
1.2.5 A BGP kiválasztási folyamat, útvonalfrissítések kezelése	19
1.2.6 Útvonal-visszavonások kezelése	22
1.2.7 Útvonalak kiválasztása – „breaking ties”	23
1.2.8 BGP útvonal-összevonás.....	24
1.3 Elterjedtebb BGP szimulátorok.....	26
2. BGP hálózati modell	28
2.1 A BGP egyszerűsített modellezése	28
2.1.1 AS-szintű leírás	28
2.1.2 A Gao-Rexford modell.....	30
2.1.3 A használt modell további sajátosságai.....	35
2.2 BGP konvergencia kérdések	38
3. Konvergencia-vizsgálatok	42
3.1 A készített szimulátor bemutatása.....	42
3.1.1 Általános tulajdonságok	42
3.1.2 Az idő kezelése	44
3.1.3 A szimulációk konvergencia-tulajdonságai	45
3.1.4 Implementált BGP üzenetek	47
3.2 Szimulációk, kapott eredmények	51

3.2.1 Előkészületek	51
3.2.2 Kapott eredmények	52
3.2.3 Tapasztalt jelenségek	58
3.2.4 Használt topológiák.....	63
4. Összefoglalás, tanulságok, kitekintés.....	64
4.1 Fontosabb tapasztalatok	64
4.2 A BGP konvergencia fejlesztésére tett javaslatok.....	65
4.3 Jövőbeli munkára vonatkozó kilátások	66
Irodalomjegyzék.....	67
Rövidítésjegyzék.....	69
Függelék: útmutató a szimulátor használatához.....	70

HALLGATÓI NYILATKOZAT

Alulírott **Pálfi András**, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot 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(k), 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 autentikált 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, 2011. 12. 08.

.....
Pálfi András

Összefoglaló

Az Internet hálózati tartományai között az útvonalválasztás feladatát a Border Gateway Protocol (BGP) végzi, mely protokoll sajátosságai lehetővé teszik, hogy a folyamatban résztvevő rendszerek útvonalválasztási döntéseiket saját gazdasági és politikai igényeik szerint hozzák meg, mintsem globális fizikai paraméterekből formált mérőszámok alapján - mindeközben nagyfokú rendelkezésre állást és globális összefüggőséget garantálva. A folyamat azonban jellegéből fakadóan korántsem biztos, hogy forgalomirányítási szempontból is optimális útvonalak kiépülését fogja eredményezni.

Az Internet szolgáltatásminőségének fenntartásához elengedhetetlen a BGP lehető leggyorsabb konvergenciájának biztosítása - ennek elérésére különböző elméletek eltérő javaslatokat tesznek. Jelen dolgozat célja néhány ilyen elképzelés alkalmazhatóságának szimuláció útján történő vizsgálata, melynek megvalósításához egy erre alkalmas szoftvereszközt készítettem. Az elkészült szimulátor képes a BGP kommunikáció és döntési folyamat számos tulajdonságának szimulálására változtatható hálózati- és protokoll-paraméterek mellett, így lehetőséget kínál a konvergencia alakulásának figyelemmel kísérésére. A szoftverrel végzett szimulációk eredményei érdekes tanulságokkal szolgálnak.

A program adottságai lehetővé teszik, hogy a forráskód kisebb módosításával a BGP mellett egyéb, kísérleti „Jövőbeli Internet” protokollok viselkedésének szimulációját is el lehessen vele végezni.

Abstract

The task of establishing datapaths between segments of a network is called *routing* and is performed by the Border Gateway Protocol (BGP) on today's Internet. Certain properties of the protocol allow the participating systems of making their routing decisions based on their own economical or political views, rather than system-wide parameters – while at the same time having to maintain global reachability and ensuring a high level of service availability. This way however, it is not guaranteed that the resulting paths will meet the network's physical optimum criteria.

Ensuring BGP's fast convergence is essential for securing a fair level of the Internet's Quality of Service, and various theories are in existence, suggesting different solutions of the problem. The goal of this thesis is to perform simulations in order to assess the usability of some of these suggestions. The making of a software tool that is designed for running such simulations is presented. The tool is capable of simulating BGP's behaviour by a diverse set of adjustable network settings, thus making it possible to examine the effect of these parameter changes on routing convergence. We also present the resulting simulated data, some of which provide surprising information.

The properties of the software make it useful for future work with only little or no modification of the source code. The types of supported further work may include research simulations of non-standard „BGP-like” routing protocols and Future Internet experiments.

Bevezetés

Az Internet használata napjaink szerves részévé vált, sokan már el sem tudnák képzelni az életüket a világháló szolgáltatásai nélkül. Naponta több millió banki tranzakció, több százezer VoIP- és videobeszélgetés zajlik le a világhálón, sőt, rengetegen már a munkájukat is otthonról végzik, hála az Internet nyújtotta kiváló lehetőségeknek. Az Internet folyamatos rendelkezésre állását, a szolgáltatás minőségének állandó szintjét sokan már tényként kezelik, épp ezért a hálózati mérnökök fontos feladata ennek biztosítása.

Az Internet jelenleg közel 40.000¹ (és évről évre egyre több), szerkezetileg egységes hálózatból, ún. Autonóm Rendszerből (Autonomous System – AS) tevődik össze - tulajdonképpen az Internet ezen hálózatok hálózata. Ezen hálózatokon belül, és a hálózatok között egyaránt útvonalválasztó eszközök (routerek) úgynevezett útvonalválasztási protokollok segítségével, speciális protokollüzenetekben, elosztott módon hozzák meg döntéseiket arról, hogy az adatsomagokat merre továbbítsák. Egy ilyen útvonalválasztó protokoll a Border Gateway Protocol (BGP) is, mellyel jelen dolgozat is foglalkozik.

Különbséget kell tennünk az autonóm rendszereken belüli (intra-AS) és az autonóm rendszerek közötti (inter-AS) útvonalválasztás között. Az útvonalválasztás ezen két kategóriája sokban különbözik egymástól mind a hálózat méretében, mind az útvonalválasztás alapelveiben - így rájuk más és más törvények és szabályszerűségek érvényesülnek. Általánosságban elmondható, hogy az AS-eken belüli útvonalválasztás célja a lehető legrövidebb út megtalálása a hálózat két pontja között (ahol egy út hossza érthetjük az útvonal átviteli sebességét, az útvonal legszűkebb sávszélességét, az útvonal megbízhatóságát, vagy akár ezek kombinációját is), és az erre kifejlesztett protokollok ezt a feladatot kiválóan el is látják. Az AS-ek közötti útvonalválasztás során azonban általában nem az utak fizikai paraméterei lesznek a mérvadók egy útvonal meghatározásánál, sokkal inkább a résztvevő AS-ek egyéni gazdasági vagy üzleti igényei, politikai meggyőződése. Ezeket, az AS-ek közti útvonalválasztási döntéseket irányító igényeket nevezzük *útvonalválasztási irányelveknek*, avagy „*routing policy*”-

¹ 2011 decemberében 39421 bejegyzett AS alkotta az Internetet. Forrás <http://www.cidr-report.org/as2.0/>

nek (sok helyen a szakirodalom az útvonalválasztás ezen formájára emiatt „policy routing”-ként is hivatkozik).

A BGP-t tervező mérnökök a protokollt célzottan a „policy routing” megvalósítására fejlesztették ki, és a BGP jelenleg az Internetet alkotó AS-ek közti útvonalválasztás egyetlen eszköze. Hatékonyságának és sokszínűségének köszönhetően megjelenését követően nem sokkal az Internet de-facto útvonalválasztási protokolljává vált, leváltva elődjét, az EGP-t (Exterior Gateway Protocol). A BGP-t konfiguráló hálózati operátor rendkívül széles eszköztárral rendelkezik az általa megfogalmazni kívánt útvonalválasztási irányelvek implementálására, mivel a BGP rendkívül sokszínű irányelv elfogadására képes, ám ezen konfigurációs technikák és a BGP pontos paraméterezésének tárgyalása a dolgozatnak nem célja².

Ez a nagyfokú diverzitás a hálózati operátornak öröm, a hálózatelemző mérnöknek azonban fájdalom, hiszen ez a sokszínűség kihívásokkal teli feladattá teszi az elosztott útvonalválasztási folyamat, a routerek által hozott döntések, illetve a döntések eredményeinek egyértelmű leírását. A legégetőbb probléma azonban, hogy az egyes AS-ek az esetek többségében azokat az irányelveket, melyek alapján az útvonalválasztást végzik, nem hozzák nyilvánosságra (teszik ezt gazdasági, titoktartási, vagy meggyőződési okokból) – így viszont épp az Interneten folyó útvonalválasztási folyamat alapját képező, a lokális döntések meghozásáért felelő algoritmusok és szűrési kritériumok azok, amik nem állnak a rendelkezésünkre. Mindezek tükrében meglehetősen nehéz feladatra vállalkozunk, ha az Internet BGP konvergenciájának leírását, vagy az Internet forgalmának feltérképezését tűzzük ki célul, hiszen pont a folyamat alappillérei hiányoznak a vizsgálatainkhoz.

Sok, jelenleg is folyamatban levő kutatási projekt kísérletezik az Interneten lezajló folyamatok univerzális leírásával – ezek a kutatások rendszerint olyan módszerekkel dolgoznak, melyek során az Internet forgalmáról nagymennyiségű adatot gyűjtenek, amely adatokból *következtetnek* az útvonalválasztók döntéseinek mozgatórugóit alkotó lokális irányelvekre. Ezek a kutatások folyamatosan javuló eredményekkel szolgálnak, ám az igazán áttörő eredmény még várat magára [1][2][3][4].

² Bővebb információkért lásd: <http://www.ciscopress.com/articles/article.asp?p=169556>

Mivel magán az Interneten értelemszerűen nem végezhetünk a BGP konvergenciájára vonatkozó méréseket (hiszen egyrészt a mérés során a hasznos forgalmat akadályoznánk, másrészt nincs sem fizikai hozzáférésünk, sem jogosultságunk az Internet minden eszközének ilyen célú használatához), ezért kézenfekvőnek tűnhet a dolog szoftveres szimulációja. Számos megoldás született a BGP viselkedésének szimulálására, melyek egymástól sokban különböznek. Jelen dolgozat is pontosan ezt tűzi ki céljául, ám az eddig ismert megoldásoktól valamelyest eltérő tulajdonságokkal, módszerrel.

A megvalósítás megtervezése során szembekerülünk azonban egy (újabb) akadállyal – mégpedig azzal a ténnyel, hogy az Internet mérete óriási. Napjaink számítástechnikai korlátai sajnos nem teszik lehetővé, hogy a szimulációt egy ilyen nagy kiterjedésű hálózaton egy ennyire komplex folyamatról belátható idő alatt lefuttathassuk. A probléma részleges megoldásaira különböző BGP szimulátorok készültek, melyekre az a jellemző, hogy a probléma csak bizonyos részeit képesek megoldani, hiszen a számítási komplexitás egyszerűsítése érdekében egyes esetekben komoly lényegi egyszerűsítéseket vagyunk kénytelenek alkalmazni. (Elterjedtebb BGP szimulátorok ismertetésével részletesebben az *1.3. fejezet* foglalkozik).

Egy 2011-ben napvilágot látott elmélet szerint a BGP protokoll egyik állítható időzítőjének bizonyos mértékű globális összehangolásával az Interneten a BGP konvergencia jelentős javulása lenne elérhető [5]. Ezen elgondolás egyelőre csak elméletben létezik - a jelenség teljes igazolásához vagy cáfolásához azonban egy, az Internettel azonos (vagy legalábbis összemérhető) méretű teszhálózaton végzett vizsgálatokra lenne szükség, ez pedig nyilvánvalóan nem kivitelezhető. Éppen ezért az elméletben említett jelenségek vizsgálatára a legjobb módszernek a szimuláció útján történő vizsgálat bizonyul.

Jelen dolgozat egyik célkitűzése, hogy ezen elmélet állításait szimuláció útján vizsgálja. (Az elmélet állításainak részletes ismertetésével az *2.2 fejezet* foglalkozik). Mivel nem találtam olyan szimulátort, amely az általam vizsgálni kívánt tulajdonságokat nagy hatékonysággal szimulálni tudná, ezért szakdolgozatom keretében magam készítettem egy szoftvereszközt, mely ugyan erős egyszerűsítésekkel él, és a BGP protokoll számos tulajdonságának megvalósításától eltekint - ám a szóban forgó elmélet állításainak vizsgálatára alkalmas.

Jelen szakdolgozat szerkezeti felépítése a következő: Az *1. fejezetben* ismertetjük az Internetes tartományközi útvonalválasztás folyamatának megértéséhez szükséges elemeket, így az Internet útvonalválasztási architektúráját és a BGP protokoll működésének alapjait. A fejezet végén bemutatásra kerülnek a jelenleg legelterjedtebb BGP-szimulátorok is. A *2. fejezet* a vizsgálatok elvégzéséhez választott absztrakt modellt mutatja be, valamint a fejezet második felében bizonyos, a BGP konvergenciájával kapcsolatban felmerülő kérdéseket, problémákat elemzünk. A *3. fejezet* a vizsgálatok megvalósítására készített BGP szimulátor főbb jellemzőit, valamint a vele elvégzett szimulációs vizsgálatokat és azok eredményeit ismerteti. A *4. fejezet* a dolgozat zárásaként a levonható konklúziókat ismerteti, a tapasztalatok alapján javaslatokat tesz a BGP konvergenciájának javítására, valamint kitekintést tesz a munka esetleges jövőbeli folytatásának lehetőségeiről. A dolgozat *függelék*e a szimulátor használatához tartalmaz egy rövid használati útmutatót.

1. A tartományközi útvonalválasztás felépítése

1.1 Az Internet útvonalválasztásának architektúrája

Mint ahogy az a bevezetésben már említésre került, az Internet szerkezetileg egységes tartományok (AS-ek) hálózata. AS-nek nevezzük hálózati elemeknek egy olyan halmazát, melyek egyazon szervezet adminisztratív felügyelete alatt állnak – így tehát éppúgy AS-nek tekinthető egy kisebb vállalkozás, vagy egy egyetem belső hálózata, mint a British Telecomé, vagy a Pentagoné. Az Interneten található AS-ek egyedi azonosítókkal, egy ún. „AS-számmal” rendelkeznek, melyeket az IANA³ bocsát az AS-ek rendelkezésére.

Ezen autonóm rendszerekben a hálózaton belüli útvonalválasztás feladatát ún. *Belső Átjáró Protokollok* (IGP – Interior Gateway Protocol) látják el. Az alkalmazott belső átjáró protokoll AS-enként eltérő lehet, és ennek kiválasztása az adott AS hálózati operátorának feladata. Ezek a kizárólag belső útvonalválasztásra használt protokollok arra lettek optimalizálva, hogy segítségükkel a routerek képesek legyenek a hálózat két adott pontja közötti legrövidebb utat megtalálni, mindezt a hálózati topológiáról és a hálózat fizikai tulajdonságai (linkek átviteli sebessége, sávszélesség stb.) szerzett információik alapján kialakított *mérőszámok* segítségével. A belső átjáró protokollok jellemzően nagy számításigényű algoritmusokat használnak a legjobb utak meghatározására, így ebből fakadóan rosszul skálázódnak nagyobb méretű (néhány száz routernél többet tartalmazó) hálózati topológiákhoz. Ilyen, gyakran használt belső átjáró protokoll pl. az *OSPF (Open Shortest Path First)*, az *IS-IS (Intermediate System To Intermediate System)*, a *RIP (Routing Information Protocol)*, illetve a Cisco által kifejlesztett *EIGRP (Enhanced Interior Gateway Protocol)*.

A belső átjáró protokollok rossz skálázódásából kifolyólag az Internet AS-ek közötti útvonalválasztását sajnos nem tudjuk ilyen protokollokra bízni, hiszen az Internetet alkotó AS-ek száma rendkívül nagy (közel 40.000). Ironikusan, az Internet globális hálózatán az ilyen rendszerszinten optimális utak megtalálása az egyes AS-eknek nem is célja – ugyanis az AS-ek egyes egyéni, „önző” érdekei a globális

³ Internet Assigned Numbers Authority

optimalitási szempontokat a legtöbb esetben felüldefiniálják. Moszkvából egy optimális útvonal hiába is vezetne a Pentagon hálózatán keresztül, az orosz kormány ezt érthető okokból elfogadhatatlannak tartaná.

Ezen eltérő igényekből és rossz skálázódási tulajdonságból kifolyólag ezt a feladatot egy egészen más koncepción alapuló protokollsaláddal, a *Külső Átjáró Protokollokkal* (Exterior Gateway Protocol, EGP) kell elvégeztetnünk, mely az Internet esetében egységesen a BGP. Mint ahogy azt bevezetőben is említettük, a BGP (és általában a külső átjáró protokollok) fontos tulajdonságai közé tartozik, hogy az utak megtalálására nem fizikai paramétereiből kialakított mérőszámokat használ, hanem az operátor által definiált irányelveket, forgalomirányítási preferenciákat. Ezen felül a külső átjáró protokollok kisebb számítási igényeiből kifolyólag sokkal jobban skálázódnak kiterjedt topológiákhoz.

Az Internethez a felhasználóknak, vagy akár más AS-eknek hozzáférést biztosító Internetszolgáltatók (*Internet Service Provider, ISP*) a versenyképesség megtartásához nem csak globális elérhetőséget kell garantálniuk, és egyre erősödő szolgáltatásminőségi (*Quality of Service, QoS*) elvárásoknak kell megfelelniük, hanem mindeközben saját gazdasági érdekeiket is érvényesíteniük kell. Mivel a szolgáltatók nagy része általában más szolgáltatóktól vásárolja az Internethez való hozzáférést, valamint a hálózati eszközeik üzemeltetéséből és fenntartásból is költségeik keletkeznek, ezért sokszor a szolgáltató számára létfontosságú, hogy a forgalmat olyan vonalon továbbítsa, amiből neki kevesebb költsége keletkezik – ebből kifolyólag amennyiben lehetőségük engedi, a forgalmat inkább vásárlóik felé, nem pedig szolgáltatóik felé irányítják (akiknek ugyanezért fizetniük kéne). Ez egy érthető magyarázatot ad arra, hogy az Interneten az AS-ek közti útvonalválasztás során a globális fizikai paraméterek alapján meghozott döntések jellemzően miért szorulnak háttérbe.

Pontosan ezeket a gazdasági igényeket figyelembe véve Lixin Gao [6] és Jennifer Rexford [7] megalkotott egy olyan modellt, mellyel olyan hálózatok alapvető tulajdonságait lehet leírni, melyeken a valóságban is rendkívül gyakran előforduló gazdasági kapcsolatok érvényesülnek. Jelen dolgozat a probléma megoldásához ezt a modellt használja fel alapul (A Gao-Rexford modell ismertetésével a 2.1 fejezet foglalkozik bővebben).

1.2 BGP alapok bemutatása

Jelen fejezet egy rövid összefoglalást nyújt a BGP protokoll lényegesebb tulajdonságairól, melyek a dolgozat további szakaszaiban tárgyalt fogalmak, jelenségek megértéséhez szükségesek. A fejezetnek nem célja, hogy a BGP-ről teljes körű ismertetést adjon, sokkal inkább csak a dolgozat által is mélyebben érintett részeket foglalja össze, magyarázza - így egyes részek tárgyalásától el is tekint. Azonban a dolgozat keretében készített (és a *3.1. fejezetben* bemutatásra kerülő) szoftvereszköz által implementált aspektusokra a fejezet részletesen kitér, így néhol az RFC-ben rögzített folyamatok, előírások pontos ismertetésére kerül sor⁴. A további részletek iránt érdeklődők az alábbi forrásokból tájékozódhatnak bővebben a BGP működéséről: [8].

1.2.1 Általános tulajdonságok

A Border Gateway Protocol az ún. „távolság-vektor” (distance-vector) útvonalválasztó protokollok egyik válfajához, az „út-vektor” protokollcsaládhoz tartozik. A távolság-vektor útvonalválasztás lényege, hogy az adott protokollt „beszélő” routerek a szomszédjaiknak küldött frissítési üzeneteikben bizonyos fizikai információkat közölnek a hálózatban található linkekről, valamint a hálózati topológiáról. Ezen információ felhasználásával minden router ki tudja számolni egy adott útvonal kedvezőségét, „jóságát”. Ha egy router egy adott célpont felé több lehetséges útvonalat is ismer, akkor az általa ténylegesen használt utat a fizikai paraméterekből számolt *mérőszámok* összevetésével fogja kiválasztani.

Az „út-vektor” útvonalválasztó protokollok (így a BGP is), a „távolság-vektor” koncepciótól annyiban különböznek, hogy a routerek egymásnak küldött üzenetei nem bizonyos fizikai link-adatokat, hanem a megjelölt célpont felé vezető *teljes útvonalat* tartalmazzák. A BGP esetében ezt az információt az üzenetekben foglalt „AS-path” attribútum hordozza, melyben az adott útvonal által érintett AS-ek azonosítószáma sorrendhelyesen szerepel (vagyis hogy egy, az adott úton kiküldött adatcsomag mely

⁴ A BGP-4 protokollt eredetileg az RFC1771 specifikálta, melyet 2006 Januárjában az RFC4271 írt felül. Jelen dolgozatban néhol a kettő közötti eltérésekről is említést teszünk.

AS-eken fog keresztülhaladni, mielőtt a célpontot elérné). Erre az „extra” információra azért van szükség, mert ennek a birtokában az útvonalválasztási hurkok⁵ kialakulását lényegesen egyszerűbb megelőzni, mint hagyományos távolság-vektor protokollok esetén - valamint a célpont felé vezető teljes útvonal ismerete az operátor számára kifinomultabb irányelvek megfogalmazására kínál lehetőséget (egy ilyen feltételre példa: „Ha a kapott útvonal keresztülhalad X hálózaton, akkor az útvonalat csak legvégső esetben használjuk adattovábbításra”).

A BGP protokollnak két alkategóriája létezik: az External BGP (eBGP) és az Internal BGP (iBGP). A két protokoll kevésben tér el egymástól, ám felhasználási területeik között van egy fontos különbség: ha két BGP router⁶ ugyanannak az AS-nek a tagja, akkor egymással iBGP protokollon keresztül kommunikálnak, két nem azonos AS-be tartozó router pedig eBGP-n keresztül vált üzeneteket. A két protokoll között lényegi különbségek vannak, ám mivel a dolgozat csak az eBGP vizsgálatával foglalkozik, így az iBGP részletes ismertetésére nem térünk ki.

1.2.2 BGP üzenettípusok

A BGP protokoll 4 üzenettípust használ, ezek az *Open*, az *Update*, a *Keepalives*, és a *Notification* üzenettípusok.

Az *Open* üzenetek felelnek a BGP routerek közötti, TCP kapcsolat felett létrejövő szomszédságok kiépítéséért. Két router akkor tud BGP-n keresztül egymással kommunikálni, ha egy ilyen „BGP-szomszédság” kettejük között előzőleg sikeresen kiépült. A *Keepalives* üzenetek a szomszédságok aktív státuszban tartásáért felelnek: a BGP szomszédságban álló routerek egymásnak bizonyos időközönként⁷ keepalives üzenet küldenek, mellyel jelzik a szomszédjuknak, hogy a fizikai kapcsolat nem szakadt meg, illetve a BGP kommunikációt továbbra is folytatni szeretnék. Ha egy router a *Hold Timer* időzítőjében beállított ideig nem kap keepalives üzenetet egy szomszédjától, az adott szomszédot elveszettnek minősíti, és a BGP szomszédságot vele megszünteti. Az

⁵ Útvonalválasztási huroknak nevezünk egy olyan útvonalat, mely során az adatcsomagok körbe-körbe továbbítódnak routerek egy adott halmaza között, végcéljukat soha el nem érve.

⁶ BGP protokollal kommunikálni képes (BGP-t „beszélő”) router.

⁷ Tetszőleges érték, ajánlottan a Hold Timer értékének harmada, de nem gyakoribb, mint 1/sec.

Update üzenetek, vagy „*frissítési üzenetek*” a routerek útvonalválasztási információinak cseréjére szolgálnak. Segítségükkel a routerek egymással útvonalválasztással kapcsolatos információkat közölhetnek - illetve ilyen üzenetben értesítik egymást arról is, ha a forgalomirányításban valamilyen változás történt. A *Notification* üzenet protokollhibák jelzésre és javítására szolgál, illetve hibás, vagy tovább fenntartani nem kívánt kapcsolatok lezárásáért felel.

A négy üzenettípus közül bővebben csak az *Update* üzenetekkel foglalkozunk.

1.2.3 Update üzenetek küldése, kezelése

Közvetlenül egy BGP szomszédság felépülését követően a szomszédos routerek egymásnak átadják „tudásukat” – vagyis, sorozatos *update* üzenetküldésekkel egymásnak elküldik a saját maguk által legjobbnak ítélt útvonalakat. Ezt követően egy router további *update* üzeneteket csak akkor küld, ha egy adott célpont felé egy, az előzőtől eltérő útvonalat választ legjobbnak, illetve ha egy célpont felé teljesen elveszíti az elérhetőséget (törli a felé vezető útvonalat).

Egy *update* üzenet számos paramétert tartalmaz az üzenetben foglalt útvonalra vonatkozóan, ezek lehetnek *tranzitív* illetve *nem tranzitív* paraméterek. Ha egy paraméter *tranzitív*, az azt jelzi, hogy egy *update* üzenetben kapott útvonal továbbhirdetése esetén ezt a paramétert is mindenképpen tovább kell küldeni. Ha a helyi router egy szomszédjától kapott útvonalat továbbít, akkor először a fogadott üzenetben talált *tranzitív* paramétereket értelmezi, szükség esetén módosítja, és az útvonal továbbhirdetésekor az új *update* üzenetbe azt / azokat beilleszti. Ha a helyi router egy *tranzitív* paramétert nem tud értelmezni, akkor azt feldolgozás nélkül, de kötelezően továbbküldi. A *nem tranzitív* paramétereket nem szabad továbbküldeni. A helyi router a *nem tranzitív* paramétereket értelmezi és feldolgozza, de a kiküldendő *update* üzenetekbe nem helyezi bele. Ha a helyi router egy *nem tranzitív* paramétert nem tud értelmezni, akkor azt eldobja.

Az *update* üzenetek paraméterei közül az egyik legfontosabb a már korábban is tárgyalt *AS-Path* (AS-útvonal) attribútum, mely gyakorlatilag a célponthoz vezető úton keresztezett AS-ek sorozata. Ha egy router egy eBGP szomszédjának küld *update* üzenetet, akkor az *AS-Path* attribútum elejére fűzi a saját AS-ének számát – ily módon,

ahogy a frissítési üzenet a hálózaton keresztülhalad, miközben minden AS a saját számát az AS-Path attribútum elejére tűzi, kialakul a célpont felé vezető útvonal.

Egy, az update üzenetekben foglalt másik attribútum a *Next-Hop* (következő állomás) attribútum, ami gyakorlatilag annak a routernek az IP címét jelöli, melyre az adott útvonalra kiküldendő adatsomagot továbbítani kell. Fontos megjegyezni, hogy egy BGP routernek nem szükséges BGP szomszédságban állnia a next-hop címen található routerrel, ugyanis a BGP engedi „külső”, avagy „*third-party*” next-hop-ok használatát.

Egy további attribútum a *Local-Preference* attribútum, mely azt jelöli, hogy a küldő router az adott útvonalat mennyire preferálja a többi, ugyanarra a célpontra vezető útvonalhoz képest. A local preference attribútum értékének meghatározására egyénileg nyílik lehetőség, annak kiszámítási módja kötetlen (többek között egy AS helyi irányelveinek implementálása a local preference attribútum alkalmas megválasztásával lehetséges).

Ezen kívül az update üzenetek számos más attribútumot is hordoznak, mint például az *Origin ID*, a *Multi-Exit-Discriminator* és a *Community* attribútumok. Jelen dolgozat során ezekkel mélyebben nem foglalkozunk.

Egy BGP router az általa újonnan kiválasztott és forgalomtovábbításra használni kívánt útvonalait update üzenetekben hirdetheti a szomszédjainak – ekkor beszélünk *útvonal-frissítésről* („*route update*”). Továbbá, ha a router egy adott célpontot többé nem tud elérni (pl. vonalhiba, routerhiba miatt), akkor az oda vezető utakat update üzenetekben a többiektől visszavonhatja – ekkor beszélünk *útvonal-visszavonásról* („*route withdrawal*”). Egy update üzenet legfeljebb egy útvonal-frissítést, ám tetszőleges számú visszavonást tartalmazhat.

A BGP szabvány szerint az update üzenetekben útvonal-frissítések küldésének gyakoriságát egy állítható időzítővel korlátozni kell – ez az időzítő a *MinRouteAdvertiseInterval (MRI)* időzítő. Az update üzenetek által hordozott útvonal-visszavonásokra azonban ez a késleltetés nem kötelező⁸. Az elgondolás értelme az, hogy a topológia hirtelen megváltozása következtében a hálózat routerei a tranziens állapot fennállása alatt (a konvergencia közben) jellemzően nagy mennyiségű update

⁸ Érdekeség, hogy az eredeti RFC1771 előírását, miszerint a visszavonásokat *nem szabad* késleltetni, az új RFC4271 felülírja – az új ajánlás értelmében a visszavonásokat *is ajánlott* késleltetni, ám nem kötelező.

üzenetet küldenek – ez azonban kellő szabályozás hiányában a hálózaton nagyon nagy sávszélességet foglal, illetve a routerek processzoraira egy hirtelen terhelést jelent. Az útvonal-visszavonásokat ezzel szemben azért nem érdemes késleltetni, mert ezen üzenetek késleltetése ideiglenes fekete-lyukak⁹ kialakulásához vezethetne. Pl. ha egy adott célpont felé járhatatlanná válik egy útvonal, erről azonban néhány router a visszavonások késleltetése miatt nem értesül azonnal, úgy a visszavonás megérkezéséig még folyamatosan továbbítja a csomagokat az adott útvonalra, holott azok a célpontot már biztosan nem fogják elérni – ezáltal gyakorlatilag haszontalan forgalmat generál, egészen addig, amíg tudomást nem szerez a változásról.

A MRAI időzítők az IETF¹⁰ által ajánlott beállítási értéke 30 másodperc volt, ám ezt az ajánlást 2010-ben visszavonták [9], lehetővé téve hogy az operátorok tetszőleges értéket használjanak. Jelenleg a BGP szabványban a MRAI használatára vonatkozó előírás a következőképpen szól: *„Egy BGP routernek egyazon szomszédnak, egyazon célpontra mutató két frissítési üzenet elküldése között legalább MRAI idő elteltét kell biztosítania. Ennek értelmében azonban egy routernek minden szomszédjához, minden lehetséges célpont felé egy-egy külön időzítőt kellene használnia – ez viszont óriási overheaddel jár. Így bármilyen technika elfogadható, amely biztosítja, hogy két, ugyanarra a célpontra mutató útvonal elküldése között legalább MRAI idő eltelik, és ez az eltelt időintervallum felülről korlátos. A hosszan fennálló fekete lyukak kialakulásának elkerülése érdekében ez a megkötés nem vonatkozik az érvénytelenné vált útvonalakat expliciten visszavonó üzenetek küldésére.”* [10]. A MRAI tehát egy szomszédonként, és azon belül célpontonként külön-külön állítható időzítő - vagyis: a szabvány értelmében *[szomszédok száma x hirdetendő célpontok száma]* darab időzítőt kellene tudnunk kezelni – ám már maga a szabvány is utal rá, hogy ez a nagy erőforrásigény miatt valószínűleg nem mindig lesz kivitelezhető.

Látható tehát, hogy a szabvány a BGP implementálójára bízta a MRAI időzítő használatának pontos módját. Mivel az említett óriási overhead¹¹, melyet a potenciálisan óriási számú időzítő kezelése okoz, a hardveres korlátok miatt sokszor nem elfogadható, ezért néhány BGP implementáció a MRAI-t úgy valósítja meg, hogy a router *bármely*

⁹ Bármilyen olyan, az útvonalválasztásban bekövetkező hiba (pl. rossz helyre való továbbítás, útvonalválasztási hurok) mely eredményeképp az adatcsomag nem éri el a végcélját, és eldobásra kerül.

¹⁰ Internet Engineering Task Force

¹¹ Többletköltség (algoritmikus vagy számítási)

két kiküldött update üzenet között kivárja a megfelelő időt - függetlenül attól, hogy a szóban forgó két üzenet ugyanarra a célpontra mutat-e vagy sem. Ebben az esetben a router szomszédonként csak egyetlen MRAI időzítőt tart fenn, amely mindig újraindul egy útvonal-frissítési üzenet kiküldését követően. Ez azonban tekinthető akár a sávszélességgel való „túlzott spórolásnak”. Ráadásul - ahogy később megmutatjuk – ez akár az elosztott útvonalválasztási folyamat konzisztenciájának a rovására is mehet. A MRAI használatának erre a módjára mostantól „túlzó” („*crude*”) használatként fogunk utalni.

1.2.4 Az útvonalválasztás és üzenetkezelés folyamata

A BGP routerek a szomszédjaiktól update üzenetekben kapott útvonalakat ún. útvonaltáblákban (*routing table*) tárolják – ezeket más néven *RIB-eknek* (*Routing Information Base*) nevezzük. Egy routerben háromféle útvonaltábla található, ezek elnevezése: *bemenő útvonaltábla*, vagy *Adj-RIB-In* (*Adjacent Routing Information Base Inbound*), *helyi útvonaltábla*, vagy *Loc-RIB* (*Local Routing Information Base*) és *kimenő útvonaltábla*, vagy *Adj-RIB-Out* (*Adjacent Routing Information Base Outbound*). A router minden egyes szomszédjához fenntart egy bemenő és egy kimenő útvonaltáblát, viszont helyi útvonaltáblából csak egyetlen központi példányt. Ezek használatát az alábbiakban ismertetjük.

A BGP útvonalválasztás során egy újonnan fogadott útvonal a fogadás pillanatától a szomszédoknak való továbbküldés pillanatáig több fázison halad keresztül – ezt a többfázisos folyamatot *kiválasztási folyamatnak* nevezzük. A folyamatot a következőképpen foglalhatjuk össze röviden: Ha az üzenetben kapott új útvonal bizonyos feltételeknek eleget tesz, akkor először beépül a megfelelő (az üzenetet küldő szomszédhoz fenntartott) bemenő táblába. Ha az útvonal további feltételeknek eleget tesz, akkor beépül a helyi táblába, illetve ha újbóli feltételeket teljesít, akkor beépül a megfelelő kimenő útvonaltáblába - ahonnan egy update üzenetbe „csomagolva” a szomszédnak kiküldésre kerül. Az útvonalak szigorúan csak ilyen sorrendben haladhatnak végig a három táblatípuson, azokat „megkerülni” nem tudják - illetve adott feltételek nem teljesítése esetén egy adott fázisban „megakadhatnak”, vagyis a következő útvonaltábláig már nem jutnak el. A folyamat során az útvonal bizonyos

változtatásokon is keresztülmehet, vagyis a feldolgozás során (a fogadástól a kiküldésig) a router az útvonal egyes paramétereit módosíthatja. A kiválasztás során az útvonalak által érintett fázisokat szemlélteti az 1.1. ábra:



1.1. ábra: Az útvonalválasztás folyamatának fázisai egy BGP routerben

Forrás: [16]

1.2.5 A BGP kiválasztási folyamat, útvonalfrissítések kezelése

Mikor a router egy szomszédjától útvonalfrissítést tartalmazó update üzenetet fogad, először bizonyos kritériumokat megvizsgál, melyeket a kapott útvonalnak feltétlenül teljesítenie kell – ellenkező esetben az útvonal elvetésre kerül, és azt a felhasználásból ki kell zárni. Ezek a *kizárási feltételek* a következők:

- 1) Ha az útvonal hurkot tartalmaz, akkor az a forgalomirányításban nem alkalmazható. A szabvány szerint a hurokmentességet a következő módon kell ellenőrizni: ha a helyi router megtalálja az „AS-Path” attribútumban a saját AS-ének az azonosítósámát, akkor az útvonalat eldobja - amennyiben nem, akkor az útvonalat elfogadja.
- 2) Ha az útvonal „Next-Hop” attribútumában található IP címen lévő eszközt a router nem tudja elérni (vagyis az útvonalválasztási táblájában nincs arra vonatkozó bejegyzés, hogy az adott célcímre küldött csomagot melyik interfészen kell kiküldeni) - így az útvonalat ebben az esetben is eldobjuk.

Abban az esetben, ha az útvonal átesik a szűrésen (vagyis a fent leírt két feltételvizsgálat során nem zárjuk ki a használatból), úgy a router az útvonalat rögzíti az azt küldő szomszédjához fenntartott bemenő útvonaltáblában. Amennyiben azt vesszük észre, hogy ebben a táblában már jelen volt olyan útvonal, amely ugyanarra a célpontra

mutat, mint az imént kapott útvonal (vagyis: ugyanattól a szomszédától korábban már kaptunk egy másik, ugyanoda vezető útvonalat) úgy a router ezt az előző útvonalat kitörli a bemenő útvonaltáblából, majd ennek a helyére illeszti be az újonnan kapottat. Ez utóbbi művelet az *implicit útvonal-visszavonás*¹².

Azt követően, hogy az új útvonal az üzenetet küldő szomszédhoz tartozó bemenő útvonaltáblába beépült, a router azt kezdi vizsgálni, hogy az útvonal kiválasztandó-e helyi használatra. Amennyiben egy útvonal helyi használatra kiválasztásra kerül, úgy a router az útvonalat beépíti a helyi táblába, (a Loc-RIB-be), majd onnantól kezdve a forgalomirányítást e szerint a bejegyzés szerint fogja végezni (A helyi router a forgalomtovábbítást *minden esetben* a Loc-RIB útvonalbejegyzései alapján végzi). A helyi használatra való kiválasztás mikéntjét a szabvány rögzíti - ez pedig nem más, mint az útvonal számos attribútumának megadott sorrendben történő vizsgálata. A folyamat lényege, hogy a futásának végén semmiképpen se fordulhasson elő, hogy nem sikerült eldönteni, hogy útvonalak egy halmaza közül melyiket válasszuk ki használatra – vagyis a folyamat *pontosan egy* útvonalat fog a döntések eredményeképp kiválasztani (ezért a folyamatot úgy is nevezik a szabványban, hogy „*breaking ties*”, vagyis kicsit sután magyarra fordítva, a „holtverseny megdöntése”). A szabvány a feltételvizsgálatokat és azok pontos sorrendjét rögzíti, ettől a sorrendtől eltérni nem lehet (a kiválasztás pontos menetét a következő pont részletesen ismerteti) - azonban a BGP-t konfiguráló hálózati operátor ennek ellenére viszonylag nagy szabadsággal rendelkezik, ha a kiválasztás során hozott döntések eredményét befolyásolni szeretné. Ezt nem másképp teszi, mint a kiválasztás során vizsgált néhány *attribútum értékének* átkonfigurálásával. Így tehát elmondható, hogy maga a folyamat ugyan nem módosítható, azonban az egyes általa vizsgált értékek igen – így szerezhetünk kontrollt a felett, hogy a helyi router mely útvonalakat fogja helyi használatra kijelölni.

Amennyiben a kiválasztás során a „tie-breaking” folyamat értelmében az útvonal helyi használatra ki lett jelölve, úgy azt be kell illeszteni a helyi útvonaltáblába. Miután ez a beszúrás megtörtént, a router mérlegeli, hogy az újonnan kiválasztott útvonal beszúrásáról értesíteni akarja-e a szomszédjait (vagyis: küldünk-e nekik útvonal-frissítési üzenetet). Ezt a következőképpen teszi: A router minden egyes

¹² Az implicit visszavonás helyes implementálására különös gonddal kell odafigyelnünk, mert hiánya fekete lyukak kialakulását eredményezheti.

szomszédjára megvizsgálja, hogy a bekonfigurált irányelvek engedélyezik-e az imént beillesztett útvonal, az adott szomszéd felé való továbbítását. Amennyiben az irányelvek megtiltják az útvonal hirdetését, úgy semmilyen üzenet nem kerül kiküldésre. Ezt, az irányelvek által egy útvonal továbbhirdetésének megakadályozását *irányelvi szűrésnek*, vagy „*policy-filtering*”-nek hívjuk. Amennyiben a szűrési feltételek engedélyezik a továbbhirdetést, úgy a router beszúrja az útvonalat az adott szomszédjához tartozó kimenő útvonaltáblába. Ezekben a táblákban található útvonalak lesznek azok, amelyek kiküldésre kerülnek a hozzájuk tartozó szomszédhoz. Ezután a router a kimenő útvonaltáblákban levő útvonalakat update üzenetekbe csomagolja, az üzenet megfelelő paramétereit beállítja, majd a küldésre kész üzenetet beilleszti egy kimenő bufferbe, melyben az adott szomszédnak elküldendő üzenetek várnak.

Emlékezzünk azonban, hogy az útvonalfrissítő üzenetek kiküldését a MRAI időzítővel késleltetni kell: amennyiben a sor elején álló, kiküldésre váró update üzenet olyan célpontra vezet, melyhez tartozó MRAI időzítő lejárt (vagyis a kellő késleltetési idő letelt), úgy az üzenet kiküldésre kerül, és az *adott célponthoz tartozó* időzítő újraindul. Amennyiben az adott célponthoz tartozó MRAI még nem járt le, úgy addig ez az üzenet tovább várakozik a sorban - ám eközben a router megpróbálja a sorban következő üzenetet kiküldeni. Ha az ahhoz tartozó MRAI óra sem járt le, akkor haladunk a sorban következőre – és így tovább addig, amíg egy üzenetet ki nem tudunk küldeni, vagy a sor végére nem érünk¹³.

Emlékezzünk, hogy egy update üzenetben 0 vagy 1 útvonal *frissítését* lehet hirdetni, ám az egy update üzenetben elküldött útvonal-*visszavonások* száma tetszőleges. Abban az esetben, ha egy update üzenet tartalmaz útvonalfrissítést, akkor a fent leírtak értelmében az üzenet kiküldését MRAI időzítőkkel késleltetni kell – ettől fogva azonban az adott update üzenetben foglalt esetleges útvonal-visszavonások kiküldése is késleltetődik. Abban az esetben azonban, ha az update üzenet *kizárólag* útvonal-visszavonásokat tartalmaz, az üzenet kiküldését *ajánlott, de nem kötelező* MRAI időzítők használatával késleltetni.

Amennyiben a kimenő üzenetsorban egy kiküldendő üzenet várakozik (pl. MRAI késleltetés miatt), de még a kiküldés megtörténte előtt egy újabb update üzenet kerül a sorba (bufferbe), amely ugyanarra a célpontra mutató útvonalat tartalmaz, úgy az

¹³ Ez az RFC által ajánlott eljárás, de a küldés módja többféle lehet. Ennek tárgyalásával bővebben a 4. fejezetben foglalkozunk

új üzenettel felül kell írni az előzőt, és így megakadályozni, hogy a sorban várakozó előző üzenet (amely idő közben elavulttá vált) kiküldődjön.

1.2.6 Útvonal-visszavonások kezelése

Az előző pontban ismertetett üzenetkezelési folyamat az update üzenetben fogadott útvonal-frissítésekre vonatkozott. Az update üzenetekben foglalt útvonal-visszavonások kezelése ehhez képest némileg különbözik, ezt ismertetjük az alábbi pontban.

Ha egy router egy update üzenetben útvonal-visszavonást kap, abban az esetben az összes útvonalat, mely az adott célpontra mutat, ki kell törölni mind a küldőhöz fenntartott bemenő útvonaltáblából, mind a helyi útvonaltáblából. A megfelelő útvonalak kitörlését követően a router elkezd átnézni az összes bemenő útvonaltábláját olyan útvonalak után kutatva, melyek az imént kitörölt útvonallal megegyező célpont felé vezetnek. Ezt úgy nevezzük, hogy a router a kitörölt útvonal helyére *pót-útvonalakat*, avagy „*backup-route*”-okat keres. A keresés végén az összes megtalált pót-útvonal közül a már említett „*tie-breaking*” folyamat segítségével kiválasztjuk a számunkra legkedvezőbbet, amelyet beszúrunk a helyi útvonaltáblába, ezáltal használatba állítjuk. (Érdemes átgondolni, hogy ebben az esetben már nem szükséges az útvonalon a kizáró feltételeket vizsgálni, ugyanis ezeket az útvonalakat azelőtt a bemenő táblába egyszer már beszúrtuk, tehát a szűrésen egyszer már átestek).

Amennyiben találtunk megfelelő pót-útvonalat, a fentiek értelmében a legkedvezőbbet beszúrjuk a Loc-RIB-be, ezt követően pedig ugyanúgy járunk el, mint egy teljesen új Loc-RIB-be beszúrt útvonal esetén - vagyis megvizsgáljuk az irányelvi szűrés feltételei alapján, hogy kinek küldhetünk az eseményről útvonalfrissítést, majd a megfelelő szomszédoknak kiküldjük az update üzenetet. Jegyezzük meg, hogy az előző, felülírt útvonal kitörléséről nem kell explicit útvonal-visszavonást küldeni a szomszédoknak, ugyanis a kiküldött útvonal-frissítés hatására az érintett szomszédok az *implicit visszavonás* következtében a tábláikból az előző, már érvényét veszített útvonalat törölni fogják.

Amennyiben a keresés során egyetlen pót-útvonalat sem találtunk a kitörölt útvonal helyére, úgy nekünk a szomszédjainkat a „veszteségről” értesíteni kell. Ebben

az esetben egy útvonal-visszavonást kell küldeni a kitörölt útvonalról, *minden* szomszédnak.

1.2.7 Útvonalak kiválasztása – „breaking ties”

Az előbbi pontban többször említést tettünk arról a folyamatról, mely útvonalak egy halmazából pontosan egyet jelöl ki helyi használatra. A folyamatot mindig olyan útvonalak halmazára futtatjuk, amelyek ugyanarra a célpontra mutatnak, és nekünk ezek közül kell a legkedvezőbbet kiválasztani. Az előbbi pontokban leírtak értelmében a „tie-breaking” folyamat az alábbi esetekben fut le:

- a) Ha a router útvonal-frissítési üzenetben egy új útvonalat kap egy szomszédjától. Ekkor vesszük az új útvonalat és azt az útvonalat, amelyet a router jelenleg az adott célpont felé való továbbításra használ (tehát a helyi útvonaltáblában található bejegyzést az adott célpontra - amennyiben van ilyen), és megvizsgáljuk, hogy a kiválasztási folyamat feltételvizsgálatai értelmében az új útvonal kedvezőbb-e mint az előző.
- b) Ha a router útvonal-visszavonási üzenet fogadása következtében egy útvonalat kitöröl a helyi útvonaltáblából. Ebben az esetben a folyamat veszi az összes, a kitörölt route-tal egyező célpont felé vezető útvonalat az összes bemenő táblából, és ezek közül kiválasztja a legkedvezőbbet, mely helyi használatra a Loc-RIB-be beszúrásra kerül.

Mint korábban említettük, a folyamatban a lépések sorrendje szigorúan rögzített, azokat felcserélni, vagy megkerülni nem lehet – ám az operátor a lépések által vizsgált értékek némelyikének átkonfigurálásával „érvényesítheti az akaratát”.

A folyamat logikája a következő: minden lépésben egy adott szempontnak a legjobban eleget tevő útvonalat választjuk ki, a többit kizárjuk a „versenyből”. Amennyiben több ilyen „legjobb” útvonal van, úgy ezekkel továbbhaladunk a következő lépésre. Ha a következő lépés végén is több mint egy útvonal marad, akkor haladunk velük tovább az azt követőre - és így tovább. A folyamat végére pontosan egy útvonal kerül ki, mint a verseny „abszolút győztese”. Ezt az útvonalat fogjuk a helyi táblában rögzíteni.

A BGP kiválasztási folyamat a szabvány által előírt lépései, sorrendben:

- 1:** Ha az adott útvonal az egyetlen általunk ismert útvonal erre a célpontra, akkor minden további vizsgálatot mellőzve beszúrjuk a Loc-RIB-be.
- 2:** Vegyük a legmagasabb *LOCAL PREFERENCE* értékkel rendelkező útvonala(ka)t.
- 3:** Vegyük a legkevesebb elemből álló (legrövidebb) *AS-PATH* attribútummal rendelkező útvonala(ka)t.
- 4:** Vegyük a legalacsonyabb *ORIGIN-ID* paraméterrel rendelkező útvonala(ka)t .
- 5:** Vegyük a legalacsonyabb *Multi-Exit-Discriminator (MED)* attribútummal rendelkező útvonala(ka)t.
- 6:** Ha az útvonalak közül valamely(ek) eBGP-n, mások iBGP-n keresztül érkeztek, preferáljuk az utóbbi(aka)t.
- 7:** Vegyük a *NEXT-HOP* routerig legalacsonyabb útköltségű útvonala(ka)t („*nearest next-hop*” szabály¹⁴).
- 8:** Ha még mindig több mint egy útvonalunk van, válasszuk azt, amelyik a legkisebb *BGP-ID*-val rendelkező szomszédtól érkezett.

Ezen lépések közül néhányal jelen dolgozat nem foglalkozik – ugyanis az általam készített (és a *3.1 fejezetben* bemutatásra kerülő) szimulátor a lépések során vizsgált számos attribútumot nem is implementálja. Ilyen attribútum az *Origin ID*, a *Multi-Exit Discriminator*, eBGP és iBGP protokollok, nexthop-költség. Így a szimulátorban is megvalósított lépések az *első* (egyetlen ismert útvonal), a *második* (legmagasabb local preference), a *harmadik* (legrövidebb AS-Path), és a *nyolcadik* (legalacsonyabb BGP-ID-val rendelkező szomszéd) lépésekre korlátozódnak.

1.2.8 BGP útvonal-összevonás

A BGP protokoll lehetőséget nyújt bizonyos, valamilyen szempont alapján hasonlóknak ítélt útvonalak összevonására. Ezáltal egy BGP routernek lehetősége van arra, hogy az általa nagyon hasonlóan vagy teljesen azonos módon kezelt útvonalakat

¹⁴ Az útköltség („path cost”) egy fizikai paraméterekből alkotott mérőszám

„összeolvassza”, és a szomszédoknak való továbbhirdetés során már egy ilyen összegzett útvonalat küldjön el. A módszer értelme, hogy segítségével számítási költséget, memórafoglaltságot és sávszélességet takaríthatunk meg – hiszen az amúgy is egységesen kezelt útvonalakra nem érdemes külön táblabejegyzéseket tárolni, róluk külön update üzeneteket küldeni – hiszen így ennyivel is rövidebb ideig tart az útvonaltáblában egy bejegyzés megkeresése, illetve ennyivel kevesebb „felesleges” üzenetet küldenénk ki a vonalra szomszédjainknak.

Az *útvonal-összevonás (route aggregation)* lényegét legegyszerűbben egy példával lehet szemléltetni: Egy Németországban található AS-nek fontos, hogy ismerje a közvetlen környezetében található AS-eket, hiszen ez alapján tudja a forgalmát hatékonyan továbbítani, illetve ez alapján tud rájuk eltérő irányelveket megfogalmazni. Semmi szüksége azonban külön információt eltárolni a Shanghai környékén elhelyezkedő összes AS-ről, hiszen mindegyikről elég annyit tudnia, hogy Kínában vannak - vagyis a nekik szánt csomagot kelet felé kell továbbítani. Egy hasonló példa a hétköznapi életből: egy fővárosi ember jellemzően legjobban Budapestet (esetleg csak a saját kerületét) ismeri, hiszen így fogja tudni, hogy hogyan lehet a leggyorsabban eljutni a boltba, vagy a fodrászhoz - illetve tudja, hogy a város bizonyos részeit tanácsosabb elkerülni. Azonban semmi szüksége arra, hogy Olaszország összes falvának utcahálózatát fejből ismerje – ha esetleg egyszer oda akar majd utazni, bőven elég, ha tudja, hogy a kerületben hol árulják az adott város térképét.

Az útvonal-összevonás IP címeken végez műveleteket a hálózati maszk hosszának változtatásával, és a BGP szabvány az egyes routereken alkalmazott útvonal-összevonás mértékét és módját a hálózati operátorra bízta. A folyamat menetének, illetve következményeinek vizsgálata meglehetősen összetett témakör, ezzel jelen dolgozat bővebben nem foglalkozik. A téma után mélyebben érdeklődők figyelmébe ajánljuk a [11] weboldalt.

Mivel az általam írt szimulátor nem használ IP címeket, így a BGP ezen fontos tulajdonságának megvalósításától eltekint – ennek a szimulációs eredményekre gyakorolt hatását a 3.2 fejezet tárgyalja részletesebben.

1.3 Elterjedtebb BGP szimulátorok

Számos szoftver létezik, mely alkalmas a BGP működésének részleges vagy teljes szimulálására. Ezek a szimulátorok sokban különböznek egymástól, és jellemzően mindegyikük más-más célkitűzéssel lett megalkotva. Az alábbi fejezet az ismertebb BGP-szimulátorok tulajdonságait foglalja össze röviden.

A BGP működését legrészletesebben szimulálni képes szoftverek a teljes, vagy közel-teljes BGP implementációk. Ezek a szoftverek jellemzően a BGP szinte minden aspektusát megvalósítják, beleértve a statikus és dinamikus részeket – így pl. az összes időzítő, a szomszédságok kiépítéséhez használt véges automata, minden szóba jöhető attribútum implementálásra került. Ezen implementációk nagy előnye, hogy a BGP minden tulajdonságát szimulálni képesek, szinte már éles környezetben is használhatóak lennének – azonban nagy hátrányuk, hogy még kifinomult virtualizációs technikákkal is csak nagyon kevés példány futtatható belőlük egy-egy munkaállomáson, nagy topológiákhoz emiatt rosszul skálázódnak. Nagy hálózatok folyamatainak vizsgálatára ebből kifolyólag nem alkalmasak. Egy ilyen BGP-implementáció a *Quagga* [12].

A BGP-szimulátorok egy másik családját *DES (Discrete Event Simulation)* alapú szimulátorok alkotják. A lényegi különbségük a teljes BGP-implementációkhoz képest az, hogy nem valós időben szimulálják a viselkedést, így pl. lehetőség van arra, hogy hosszabb időintervallumokat „átugorjanak”, mikor a rendszerben amúgy sem történne semmi. A működés alapja, hogy a szimulátor egy globális eseménysorba gyűjti az egyes BGP-routerek által generált üzeneteket, majd ezeket időrendi sorrendben feldolgozza, és ez alapján módosítja a hálózat globális állapotát – vagyis a routerek útvonaltábláit és üzenet-buffereit. A DES alapú szimulátorok előnye, hogy jóval gyorsabb futási idővel rendelkeznek a teljes implementációknál (a kisebb számítási igényből és a diszkrét időkezelésből kifolyólag), de emellett a BGP dinamikus viselkedésének szimulálására is ugyanúgy alkalmasak. Hátrányuk azonban, hogy számítási igényük még így is viszonylag nagy, ezért pár száznál több routerből álló topológiákat már nem képesek kezelni. Ilyen DES alapú, széles körben elterjedt szimulátor az *SSFNet* [13].

A szimulátoroknak egy harmadik osztályát a BGP emulátorok alkotják, melyeknek jelen pillanatban egyetlen kiforrott implementációja a Nick Feamster [14]

által készített *BGP Emulator*. A szoftver előnye, hogy a DES alapú szimulátoroknál gyorsabb, mivel egy speciális módszert használva üzenetváltások modellezése nélkül is képes a BGP döntési folyamat emulálására – hátránya azonban, hogy a szoftver az AT&T cég¹⁵ tulajdonát képezi, azt csak belső kutatási célokra használják, így nem nyilvánosan hozzáférhető.

Egy ezektől eltérő felépítéssel rendelkező szimulátor-típus a TES (*Timeless Event Scheduling*) alapú szimulátorok családja, melyek közül a legismertebb, és legszélesebb körben elterjedt szimulátor a *C-BGP* [15]. A TES módszeren alapuló szimulátorok az előzőekben említett szoftverek által (a nagy komplexitás miatt) nem elvégezhető vizsgálatokat célozzák meg – vagyis céljuk, hogy minél nagyobb topológiákra is jól skálázódjanak. Azonban ennek eléréséhez a szimuláció számítási igényének radikális csökkentésére van szükség, így ezek a szimulátorok komoly „áldozatokat” hoznak annak érdekében, hogy futási idejük javulhasson. Ebből kifolyólag a BGP nagyon sok tulajdonsága nem kerül bennük megvalósításra – pusztán azokat a részeket hagyják meg, amelyek a döntési folyamat vizsgálatához elengedhetetlenül szükségesek. Így például a BGP állapotgép, az időzítők, és a szállítási rétegbeli TCP protokoll implementálása kimarad belőlük, melyből kifolyólag azonban számos tulajdonság vizsgálatára nem lesznek képesek – ilyen tulajdonságok többek között a BGP dinamikus tulajdonságai, mint pl. a konvergenciaidő, vagy a „*route-flapping*”¹⁶ kezelése.

Könnyen látható tehát, hogy minden szimulátor-család eltérő célkitűzésekkel rendelkezik, és létjogosultságukat más-más tulajdonságaiknak köszönhetik. Ebből kifolyólag azonban mindegyikük csak a feladatok egy szűk körét képes elvégezni. Nekünk, mint felhasználónak ezért először mérlegelnünk kell, hogy milyen szimulátort válasszunk, melytől azt reméljük, hogy céljainknak megfelel – a választás sokszor nem egyszerű, mert sokszor „mindenből egy kevés” jellegű igénnyel találjuk magunkat szemben. Így jártam magam is a dolgozat készítésekor, így egy meglévő szimulátor használata helyett úgy döntöttem, egy saját szoftvereszközt készítek, mely célzottan az általam elvégezni kívánt feladatra lesz használható – miközben minden egyéb részlettől eltekint.

¹⁵ American Telephone and Telegraph Company

¹⁶ Olyan hibás működés, mikor egy router egy útvonal elérhetetlenségét, majd újbóli elérhetőségét sűrű egymásutánban, felváltva hirdeti.

2. BGP hálózati modell

2.1 A BGP egyszerűsített modellezése

Jelen dolgozat a BGP kommunikáció folyamatának szimuláció útján történő vizsgálatát tűzte ki céljául. Egy szimuláció során azonban minden esetben szükség van egy alkalmasan megválasztott *modellre*, mely a valóság eseményeit minél jobban reprodukálni képes. Jelen dolgozat esetében egy komplex folyamat nagy topológiákon történő viselkedésének szimulációjára van szükség. Az ennek elvégzéséhez választott modell megválasztása kritikus, hiszen a számítási teljesítmény jelenlegi korlátai miatt nem kivitelezhető, hogy a szimulálni kívánt folyamat egyszerre nagy méreteket öltson, és egyben komplex is legyen. Emiatt tehát a hatékonyság és a részletesség közötti megfelelő egyensúlyra kell törekednünk, melynek eléréséhez bizonyos egyszerűsítésekkel kell élnünk - a hatékonyság hasznára, ám a pontosság rovására. Jelen fejezet az általam választott, a BGP hálózatok leírását célzó modell tulajdonságait ismerteti.

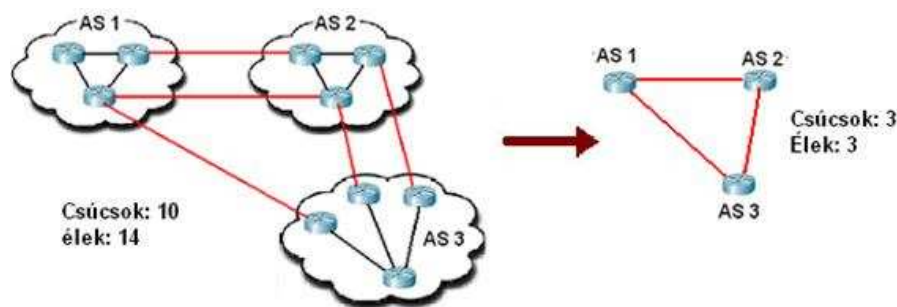
2.1.1 AS-szintű leírás

Mint már korábban említettük, a BGP a külső átjáró protokollok családjába tartozik, így kizárólag hálózatok közötti útvonalválasztásban használják. Szintén említésre került korábban, hogy az AS-ek közti útvonalválasztás alapját nem rendszerszintű fizikai paraméterek, hanem a résztvevő AS-ek egyéni igényei képezik. Ezzel szemben az egyes AS-eken belül használt belső átjáró protokollok két végpont közötti legrövidebb út megtalálására használatosak - így ezek modellezhetők bármilyen, legrövidebb utat megtalálni képes algoritmussal (pl: Dijkstra, Bellmann-Ford). Az AS-eken belüli útvonalválasztás modellezésével jelen dolgozat nem foglalkozik, helyette kizárólag a „policy routing” viselkedésére koncentrálok.

Egy BGP hálózat routereinek útvonalválasztási döntéseit az egyéni igények alapozzák – egyszerűen fogalmazva, egy BGP hálózat forgalmát „gazdasági és politikai

erők irányítják”. Ebből a szempontból nézve viszont tekinthetjük az AS-eket ezen gazdasági és politikai igények „alapegységeinek” – hiszen az előbb tárgyaltak szerint egy AS-en belüli útvonalválasztás egészen más alapokon nyugszik. Azzal a feltételezéssel élünk tehát, hogy az AS-ek közti irányelvalapú útvonalválasztás leírásakor a szimuláció során kapott eredmény és a hálózaton tapasztalt viselkedés független lesz az AS-eken belüli belső útvonalválasztási folyamatok eredményeitől. (Megjegyzés: ahogy azt [16] megmutatja, ez a feltételezésünk sajnos nem teljesen helyénvaló, ugyanis igenis lesz eltérés a kapott eredmények között, ha a belső folyamatokat elhanyagolhatónak tekintjük – ám ez az eltérés elegendően kicsi ahhoz, hogy az AS-szinten való modellezést a valós hálózatok viselkedésének egy elegendően jó közelítésének tekinthessük.)

E megállapítástól vezérelve a választott modellem egy ún. *AS-szintű gráfon* alapul, mely a következőt takarja: a hálózat gráffal való modellezése során - mivel az egy AS-en belül történő folyamatoktól eltekintünk - a gráf egy csúcsának egy-egy AS-t választunk, a gráf éleinek pedig az AS-ek között kiépített BGP szomszédságokat. A modell gyakorlatilag az AS-eket egyetlen BGP routerként kezeli. Meg kell azonban jegyezni, hogy ez az absztrakció egy óriási méretbeli és bonyolultságbeli csökkenést eredményez egy pusztán routerszintű modellhez képest – hiszen egy AS esetenként akár több száz routert is tartalmazhat. Erre az egyszerűsítésre mutat egy szemléletes példát a 2.1. ábra:



2.1. ábra: Az AS-szintű modell egyszerűsítései

Mivel a BGP első és legfontosabb felhasználási területe az Internet, így – bár jelen dolgozatnak ugyan nem - a BGP vizsgálásának valódi célja az Interneten folyó folyamatok megértése, leírása. Azonban ahogy az imént említettük, az AS-szintű vizsgálat meglehetősen sokat egyszerűsít, így ez a fajta egyszerűsített vizsgálat természetéből fakadóan nem fog pontos eredményeket szolgáltatni. Az Internet

vizsgálata során azonban sajnos be kell érünk ezzel a szinttel – egészen egyszerűen abból az okból, hogy az Internetről az AS-szintnél részletesebb adatok nem állnak a rendelkezésünkre. Ehhez ugyanis ismerni kéne az Internetet alkotó AS-ek teljes belső struktúráját is, amely információ nyilvánvalóan nem hozzáférhető.

AS-szintű modellek alkotásával és vizsgálatával sokan foglalkoztak az elmúlt évtizedben – így pl. az Internet AS-szintű leírásával a CAIDA¹⁷ szervezet kísérletezik. Holnapjukon egy időről időre frissülő, és az Internet AS-szintű topológiájáról egyre helyesebb képet adó AS-szintű gráf éllistája bárki számára hozzáférhető [17]. Mivel az AS szintű gráffal való modellezés a BGP viselkedését vizsgáló kutatók hasznos alapeszközévé vált, így velük kapcsolatban számos megállapítás, elmélet született. A témával mélyen foglalkozó Lixin Gao [6] és Jennifer Rexford [7] megalkotta a vezetékeveikről elkeresztelt *Gao-Rexford modellt*, melynek azonosságaira és meglátásaira a jelen dolgozatban használt modell is épít. A következő pont ismerteti ezt a modellt, valamint a hozzá szorosan kapcsolódó tételeket és összefüggéseket.

2.1.2 A Gao-Rexford modell

Mint ahogy azt [18] is meg mutatja, egy tetszőleges BGP hálózatban megadhatók a résztvevő AS-ek útvonalválasztási irányelveinek olyan kombinációi, melyek esetén egy topológia-változást követő BGP kommunikáció soha nem fog konvergálni (az üzenetek küldése és a RIB-ek folyamatos frissítése sosem fejeződik be, nem alakul ki stabil végállapot). Az Internet AS-einek irányelvei nem publikusak, így tehát nincs lehetőségünk annak a kivizsgálásra, hogy az Interneten jelen lévő irányelvek együttese a hálózatot „divergenciára ítéli-e”. A problémát tovább nehezíti, hogy még abban az esetben, ha ismernénk is az összes irányelvet, akkor sem tudnánk belátható időn belül nyilatkozni a konvergenciáról – ugyanis ahogy [19] megmutatja, már csak annak a ténynek a vizsgálata, hogy egy BGP hálózatban egyáltalán létezik-e stabil végállapot, NP-nehez probléma. Ahogy azt szintén ebből a cikkből megtudhatjuk, amennyiben irányelvek egy olyan kombinációját vesszük, melyekre igaz, hogy nem lesz

¹⁷ The Cooperative Association for Internet Data Analysis, <http://www.caida.org/home/>

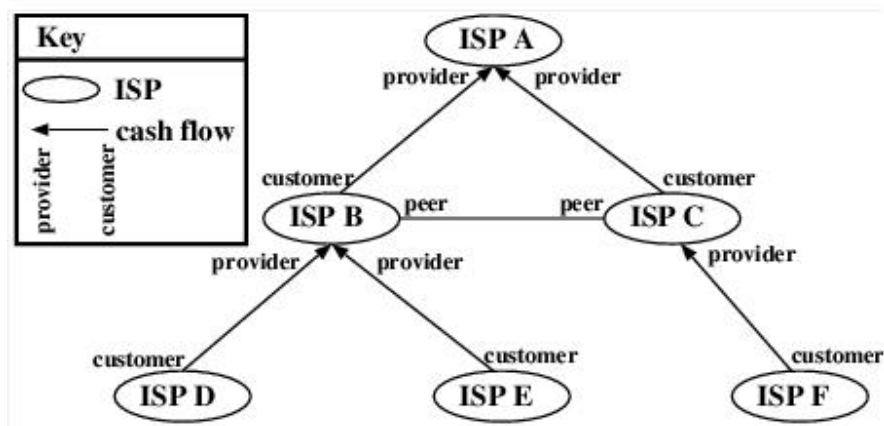
a hálózatban ún. „vitakör” (*dispute-wheel*)¹⁸, úgy a hálózatra a BGP konvergenciája *garantált* - amennyiben azonban ilyen vitakör jelen van a hálózatban, úgy a BGP kommunikáció *garantáltan* nem lesz konvergens. Ezek értelmében, mivel a BGP protokoll konvergenciájának szükséges és elégséges feltétele a hálózat vitakörmentessége, ezért a konvergencia vagy divergencia kinyilvánításának problémáját ilyen vitakörök keresésének problémájára módosíthatjuk - azonban ahogy említettük, az Internet AS-einek irányelveit nem ismerve nem tudjuk (és enélkül az információ nélkül nem is tudhatjuk), hogy vajon az Interneten jelen van-e ilyen vitakör, azaz, hogy fennáll-e az irányelvek ilyenfajta „konfliktusa”.

Az Internetet alkotó AS-ek kommunikációja azonban konvergens – ezt a mindennapjainkban tapasztaljuk, hiszen másképp nem volna lehetséges az Internet szolgáltatásminőségének elfogadható szintű biztosítása. Ez viszont a [19] -ban foglaltak szerint nem mást jelent, mint hogy az Interneten nincs jelen vitakör – aminek kialakulásának azonban már egészen kisméretű hálózatok esetében is meglepően nagy az esélye. Hogyan lehetséges mégis ez? Erre ad egy elfogadható magyarázatot a Gao és Rexford által megalkotott egyszerű gazdasági modell. A *Gao-Rexford modell* az Internetet alkotó AS-ek gazdasági megfontolásainak (és az ezekből megalkotott irányelveinek) egy általános megragadására alapul – és ahogy [20] is megmutatja, minden, a Gao-Rexford modell azonosságainak eleget tevő hálózat egyben vitakörmentes is lesz, azaz az imént tárgyaltak szerint bennük a BGP konvergenciája garantált.

A modell alapkoncepciója a következőkön alapul: az Interneten jellemző elrendezési forma, hogy egy AS az Internethez való hozzáférését más AS-ektől, mint szolgáltatóktól vásárolja - illetve további AS-eknek az Internethez való hozzáférést biztosít. Ez alapján beszélhetünk az AS-ek közötti szolgáltató-vásárló (*provider-to-customer*, „*p2c*”) kapcsolatról, melyben értelemszerűen a vásárló a szolgáltatójának pénzt fizet. Egy másik gyakran előforduló kapcsolattípus, a társ-társ (*peer-to-peer*, „*p2p*”) kapcsolat, melyben a két összekapcsolt AS egy dedikált vonalat épít ki közös használatra, melynek fenntartási költségeit együttesen állják – ám egymásnak a *használatért* nem fizetnek (hiszen a vonal a kettejük közös tulajdonát képezi). A társ-társ linkek kiépítésének népszerűsége abban rejlik, hogy a forgalomnak

¹⁸ BGP irányelvek olyan együttese, melyek hatására bizonyos routerek egy adott célpont felé a közvetlen útvonalak helyett egymáson keresztülhaladó, közvetett útvonalakat preferálnak, ily módon útvonalválasztási hurkok kialakulását eredményezve.

egy ilyen linkre való továbbítása az AS-eknek anyagilag jobban megéri, mint ugyanannak a forgalomnak egy közös szolgáltató felé való továbbítása – ugyanis ekkor a szolgáltatóknak mindkettejüknek fizetni kellene. A Gao-Rexford modell koncepcióját és az általa feltételezett gazdasági elrendezéseket szemlélteti a 2.2. ábra:



2.2. ábra: A Gao-Rexford modell gazdasági elrendezései.

Forrás: www.caida.org

Összegezve tehát, a Gao-Rexford modell világában egy AS a szomszédjaival a fent említett három kapcsolattípus egyikével fog kapcsolódni, ami tehát lehet szolgáltató-vásárló (p2c) kapcsolat, vásárló-szolgáltató (c2p) kapcsolat, illetve társ-társ (p2p) kapcsolat.

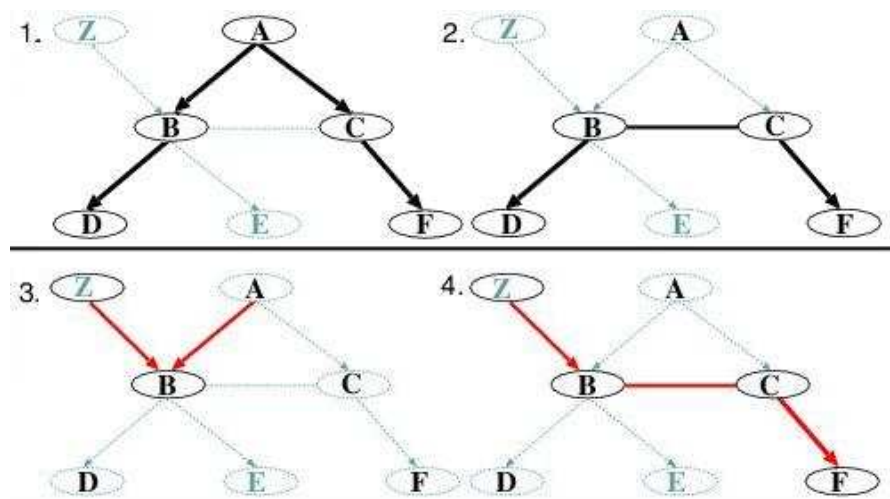
Amit fontos észrevenni, hogy egy racionálisan gondolkodó AS szemszögéből nézve a három kapcsolattípus között egy egyértelmű preferencia-sorrend állítható: egy AS legszívesebben a forgalmát vásárlói felé továbbítja, hiszen ezért a forgalomért a vásárlóinak nem kell pénzt fizetnie. A társai felé kevésbé szívesen továbbítja forgalmát, hiszen ugyan a link fenntartási költségét a szomszédjával közösen kell állniuk – ám ez még mindig kevesebb kiadással jár, mint a szolgáltató felé továbbítani. Legvégső esetben a forgalom a szolgáltató felé továbbítódik, mely esetben azonban a vonal használatáért is fizetni kell. Mindezek értelmében, ha $p()$ az AS-ek egy preferenciafüggvénye, akkor az AS-ek racionális gondolkodása esetén a három linktípusra egyértelműen felállítható a

$$p(p2c) > p(p2p) > p(c2p) \quad (4.1)$$

preferencia-sorrend. A Gao-Rexford modell pontosan ezt a jellegzetes „gazdasági elrendezést” veszi alapul. Fontos azonban megjegyezni, hogy ez a

valóságnak csak egy igen komoly egyszerűsítése – hiszen ez az elrendezés korántsem tükrözi az AS-ek közti esetleges egyéb (pl: politikai, üzleti) eseti igényeket, preferenciákat (pl: Oroszország az USA felé nem irányít forgalmat, függetlenül a velejáró veszteségtől; vagy pl. a Microsoft titokfélési okokból semmiképp nem továbbít az Apple felé). Mint azt már korábban említettük, az Internetet alkotó AS-ek irányelvei nem publikusak – így ennek a hiánynak az „általános ésszerűséggel” való pótlását célozza meg a Gao-Rexford modell, mely ilyen egyszerű, ám „gazdaságilag racionális” irányelveket feltételez minden AS-től. Ezen feltételezésre alapozott modellen azonban néhány megállapítás tehető:

Amennyiben a modell által felételezett racionalitásnak minden AS eleget tesz, úgy a kialakult topológiában egyes útvonalak szükségszerűen nem alakulhatnak ki. Az ilyen útvonalakat *érvénytelen útvonalaknak* nevezzük, míg azokat, melyeket a szabályok betartásával kiépíthetünk, *érvényes útvonalnak* hívunk. Az érvénytelen út elgondolása onnan ered, hogy a racionalitásból fakadóan egyetlen AS sem fog olyan irányban forgalmat továbbítani, melyből fedezetlen költsége származik – vagyis: egy AS egy szolgáltatótól származó forgalmát nem fogja egy másik szolgáltató felé irányítani, hiszen ennek során ő két szolgáltatónak is fizet, ám ezt a költségét senki nem fedezi. A szolgáltatótól származó forgalmat társvonalon szintén nem továbbítjuk, hiszen ebben az esetben is fedezetlen költség keletkezik. Hasonlóképpen, egy társ felől érkező forgalmat sem szolgáltató, sem egy újabb társ felé nem továbbítjuk, hiszen ebben az esetben sem fog „senki fizetni azért, hogy a forgalmat továbbítottuk” (ráadásul a társszerződések legtöbbször a résztvevőknek megtiltják, hogy a vonalat „idegen”, vagyis más társ felől érkező forgalom továbbítására használják). Ezen érvényes útvonalakban tehát a forgalom által keresztezett linkek sorrendjét tekintve p2p és c2p vonal kizárólag p2c vonalat követhet, illetve egy p2p vagy egy c2p vonalat szükségképpen p2c vonalnak kell követnie. A 2.3. ábra ilyen érvényes és érvénytelen útvonalakat szemléltet.



2.3. ábra: Érvényes (1,2) és érvénytelen (3,4) útvonalak Gao-Rexford hálózatokban

Forrás: www.caida.org

A modell feltételezéseiből következő és a BGP konvergenciáját garantáló feltételek a jelen dolgozat számára is nagy fontossággal bíró *Gao-Rexford tételben* foglalhatók össze:

Tétel: Ha egy Gao-Rexford modellen alapuló hálózatban teljesül az alábbi három feltétel, akkor a hálózat vitakör-mentes lesz, következésképp a hálózatban a BGP protokoll konvergenciája *minden esetben* garantált. Ez a három feltétel a következő:

- 1: *Topológiára vonatkozó megkötés* - Nincsen a hálózatban p2c élekből álló irányított kör – vagyis egyszerűbben: egyetlen csomópont sem lehet áttételes saját magának a vásárlója.
- 2: *Üzleti preferenciákra vonatkozó megkötés* - Minden csomópont lokális irányelve szerint a forgalmát szívesebben irányítja a társvonalak felé, mint a szolgáltatói felé, illetve szívesebben továbbít forgalmat a vásárlói felé, mint a társai felé. Vagyis, ha az üzleti preferenciafüggvény $p()$, akkor teljesül a (4.1) összefüggés.
- 3: *Stratégiára vonatkozó megkötés* - A hálózat minden tagja „gazdaságilag racionális”, vagyis minden csomópont egyénileg gondoskodik arról, hogy rajta keresztül *érvénytelen útvonal* ne húzódhasson.

Vegyük észre, hogy valós hálózatokon az első feltétel minden esetben teljesül, hiszen értelemszerűen senki nem lehet önmagának az áttételes vásárlója.

A második feltétel a Gao-Rexford modell által feltételezett gazdasági racionalitás első eleme, vagyis hogy mindenki arra szeret továbbítani, ami neki kevesebbe kerül. A harmadik feltétel a gazdasági racionalitás másik eleme, vagyis, hogy senki nem továbbít forgalmat olyan módon, hogy neki a forgalomtovábbításból vesztesége keletkezne. (Ahogy azonban [21] megmutatja, nem lehetünk abban biztosak, hogy a harmadik feltétel „játékszabályainak” minden AS eleget tesz – vagyis nem lehetünk biztosak abban, hogy minden AS valóban csak a vásárlóinak biztosít tranzit szolgáltatást¹⁹.

2.1.3 A használt modell további sajátosságai

Az általam választott, és a szimulátorban megvalósított modell tehát a Gao-Rexford koncepción alapul. A modell egy gráf, melyben minden csúcspont egy AS-t reprezentál, amelyek egy-egy önálló BGP routerrel vannak modellezve. A szimulátorban a vizsgált topológia esetén megadható az egyes vonalak típusa (p2c, p2p vagy c2p). Lehetőség van arra, hogy a routereken bizonyos alapértelmezett kimenő szűrők segítségével biztosított legyen a Gao-Rexford tételben említett 2. és 3. (az üzleti preferenciára és a stratégiára vonatkozó) feltétel. Lehetőség van a kimenő szűrők kiiktatásával arra is, hogy egy router ne állítson fel a linkek között preferenciasorrendet, illetve engedélyezzen „érvénytelen” átmenő forgalmat – így a szimulációk futtathatók általános (nem Gao-Rexfordi) hálózatokra is – azonban emlékezzünk, hogy az ilyen esetben esetlegesen kialakuló vitakörök miatt a konvergencia nincsen garantálva!

A szimulátorban minden egyes gráf-csomóponton implementálva vannak a MRAI időzítők. Mint ahogy az előző fejezetben említettük, a szabvány nem tesz szoros megköttést a MRAI alkalmazásának módjára, így sokan a szabvány ajánlásával ellentétben *minden* egymást követő frissítési üzenetet késleltetve üzemeltetik a MRAI-kat - míg mások szabványszerűen, kizárólag az azonos célpontra vezető útvonalakat hirdető, egymást követő üzeneteket késleltetve használják az időzítőket. A

¹⁹ Vegyük észre, hogy a tétel a valós BGP konvergenciájáról nyilatkozik – ám könnyen lehet, hogy egyes BGP szimulátorokban a hatékonyság érdekében eszközölt egyszerűsítések miatt még e három feltétel együttlállása esetén sem garantálható a konvergencia.

szimulátorban lehetőség van arra, hogy a MRAI használatának módját *globálisan* állítsuk, ezáltal vizsgálhatók a használati mód különbözőségének a konvergenciára gyakorolt hatásai.

A szimulátorban az egyes AS-ek (vagyis a modell szintjén az egyes routerek) kizárólag AS azonosító számaikkal vannak azonosítva, nem pedig IP címeikkel – így azonban a szimuláció folyamata a valós BGP egyik fontos aspektusától, az *útvonal-összevonástól* eltekint. Az egyszerűsítésre több okból kifolyólag került sor: Egyrészt a szimulátor kizárólag a kiépülő útvonalak AS-Path attribútumának (vagyis a keresztezett AS-ek sorrendjének) alakulását hivatott szimulálni – ezek pedig pontosan az AS azonosítókból tevődnek össze. Másfelől, az Internetről a jelenleg rendelkezésre álló (és bárki számára elérhető, publikus) adathalmazok kizárólag az AS-azonosítókat teszik nyilvánossá, az egyes AS-ek pontos IP-cím kiosztásáról nem áll rendelkezésre pontos adat²⁰. Az IP címtartományok egyes AS-ekhez történő *pontos* kiosztásának ismerete nélkül viszont nem modellezhető az útvonal-összevonás. A harmadik ok, amiért az útvonal-összevonás modellezésétől eltekintetünk az, hogy a BGP által használatos útvonal-összevonás módszere egy nagyon összetett témakör, ráadásul az egyes AS-ek által végzett tényleges útvonal-összevonás pontos módjának és mértékének meghatározása az operátor feladata, mivel az összevonás mértékét a szabvány nem rögzíti. A szimulátor ezen „hiányossága” azonban ahhoz vezet, hogy a szimulátorban kialakuló útvonaltáblák mérete, illetve az elküldött üzenetek száma nem lesz egyező a valós hálózatokban tapasztalt értékekkel – hiszen itt *minden* célpont felé vezető útvonalat külön kezelünk, azokat össze soha nem vonjuk. Ezek a kapott értékek tehát a szimulátorban kizárólag tájékoztató jellegűek, nem adnak abszolút egyezést a valós protokollban tapasztalt számokkal – azonban a futtatott szimulációk során ezen számok *változási arányát* jól fogjuk tudni vizsgálni az útvonal-összevonási procedúra mellőzése mellett is.

Mivel a szimulátor kizárólag AS-ek közti útvonalválasztás vizsgálatára készült (emiatt használ AS-szintű modellt), így az AS-eken belül végbemenő belső útvonalválasztási folyamatoktól eltekint. Ebből következően azonban az iBGP sajátosságai nincsenek megvalósítva a szimulátorban. Így viszont az útvonalválasztási döntések során alkalmazott kiválasztási folyamat is némileg módosul, ugyanis némelyik

²⁰ az IP címek kiosztását és nyilvántartását a világon található 5 Regionális Internet Nyilvántartó hivatal (Regional Internet Registry, RIR) végzi

feltétel a nem implementált paraméterek miatt nem is kerül kiértékelésre - ebből kifolyólag néhány lépést a folyamat nem valósít meg. Mindezek következtében a szimulátorban ténylegesen használt kiválasztási folyamat lépései az alábbiakra korlátozódnak:

Amennyiben az útvonal hurkot tartalmaz, az útvonalat a folyamatból kizárjuk.
(kizárási kritérium)

- 1: Ha az adott útvonal az egyetlen általunk ismert útvonal erre a célpontra, akkor minden további vizsgálatot mellőzve beszúrjuk a Loc-RIB-be.
- 2: Vegyük a legmagasabb LOCAL PREFERENCE értékkel rendelkező útvonala(ka)t.
- 3: Vegyük a legkevesebb elemből álló (legrövidebb) AS-Path attribútummal rendelkező útvonala(ka)t.
- 4: Ha még mindig több mint egy útvonalunk van, válasszuk azt, amelyik a legkisebb BGP-ID-val rendelkező szomszédtól érkezett.

(Megjegyzendő, hogy a valós BGP protokollban egy router BGP ID-ja egy IP cím. Mivel a szimulátor IP címeket nem használ, itt így egy router BGP ID-ja egyenlő lesz az AS-azonosítójával).

A szoftverben ténylegesen kiértékelésre kerülő 4 lépés által vizsgált paraméterek közül az AS-Path és a BGP ID nem konfigurálható - viszont a Local Preference attribútum tetszőleges képlet alapján számítható. Ennek a szoftverben alapértelmezésként használt képlete a (4.1) –es képlet, mely csak a linkek típusa (p2c, p2p, c2p) között tesz preferenciabeli különbségeket – ám kis módosítással a programba ettől eltérő preferenciafüggvény is beleilleszthető. Egy ilyen egyszerűen eszközölhető módosítással lehetőség nyílik változatos irányelvek definiálására, így a szimulátor alkalmassá tehető a szabványos BGP-n felül bármilyen „BGP-szerű” vagy „Future Internet” EGP protokollok viselkedésének vizsgálataira is.

2.2 BGP konvergencia kérdések

Az alábbiakban ismertetjük a dolgozat által vizsgált, az IEEE 2011-es shanghai INFOCOM konferenciáján egyik bemutatásra kerülő cikk, a „*There’s something about MRAI: Timing diversity can exponentially worsen BGP convergence*”[5] által állított feltételezéseket. Az ezután következő fejezetben az elkészült szimulátorról esik szó, mely alkalmas lesz a cikkben foglaltak ellenőrzésére.

A MRAI időzítők használatának célja, hogy egy topológia-változás során kiküldésre kerülő update üzenetek frekvenciáját korlátozza, ezáltal sávszélességet spórolva és a routerek processzorainak hirtelen leterhelését megelőzve - ám egyúttal a konvergencia sebességét rontva. Mivel az Interneten működő alkalmazások egyre erősebb sebesség-követelményeket állítanak, így a BGP konvergenciáját gyorsítandó, az IETF szervezet a MRAI beállításának az RFC 4271 [8] által ajánlott (eBGP-re vonatkozó) 30 másodperces értékét *érvénytelenítette*, és az új ajánlás szerint a MRAI beállítás értéke *tetszőleges* lehet [9]. Ennek hatására - a konvergencia gyorsításának reményében - egyes AS-ek operátorai időről időre új, alacsonyabb értékeket állítanak be routereiken, azonban más AS-ekben a változtatások nem, vagy csak késve kerülnek beállításra. Ebből kifolyólag azonban az Interneten használt MRAI beállítások nagyon eltérőek lesznek – ráadásul mivel az ajánlott alapértelmezett érték érvényét veszítette, így megközelítőleg sem tudhatjuk, hogy milyen értékek vannak használatban.

Az vizsgált elmélet állítása szerint mindez a konvergencia várt javulása helyett ironikusan pont a helyzet romlását eredményezi – a szerzők szerint minél heterogénebb MRAI beállítások vannak érvényben egy BGP hálózaton, annál „rosszabbak” lesznek a konvergencia-tulajdonságok. Ez érdekes állítás, hiszen szöges ellentétben áll a józanész által diktált feltételezéssel – vagyis hogy minél többen használnak rövidebb késleltetési időket, annál gyorsabb lesz a konvergencia. Az elmélet szerint, amíg a beállítás csökkentését *mindenki* egységesen el nem végzi, addig a helyzet nem hogy javulni nem fog, de még romlik is.

A MRAI heterogenitás, illetve a konvergencia „jóságának” számszerűsítésére a cikk bevezet néhány mérőszámot, melyek az alábbiak:

Heterogenitás mérőszámai:

- a. *MRAI eltérőség (diszparitás)*: A rendszerben érvényben levő legmagasabb, és legalacsonyabb MRAI beállítás értékének hányadosa. Jelölése: r
- b. *MRAI sokszínűség (diverzitás)*: A rendszerben érvényben levő különböző MRAI beállítások száma („hány féle MRAI érték van érvényben”). Jelölése: v

A konvergencia minőségének mérőszámai:

- c. *Konvergencaidő*: A topológia megváltozásától az új stabil állapot eléréséig eltelt idő. Jelölése: T
- d. *Konvergenca-forgalom*: A konvergencia során a rendszerben összesen kiküldött update üzenetek száma. Jelölése: R

A cikk külön összefüggéseket állít Gao-Rexford hálózatokra, illetve nem Gao-Rexfordi, de vitakör-mentes elrendezésekre (utóbbi feltétel a konvergencia garantáltsága miatt kell) – ez utóbbi esettel a dolgozatban nem foglalkozunk. Az elmélet szerint Gao-Rexfordi hálózatokon a

$$\begin{aligned} T &= \Theta(\alpha t_*) = \Theta(\alpha t^*) \\ R &= \Theta(\alpha m r) \\ R &= \left(\frac{\alpha}{v} \right)^{\Omega(v)} \end{aligned} \tag{5.1}$$

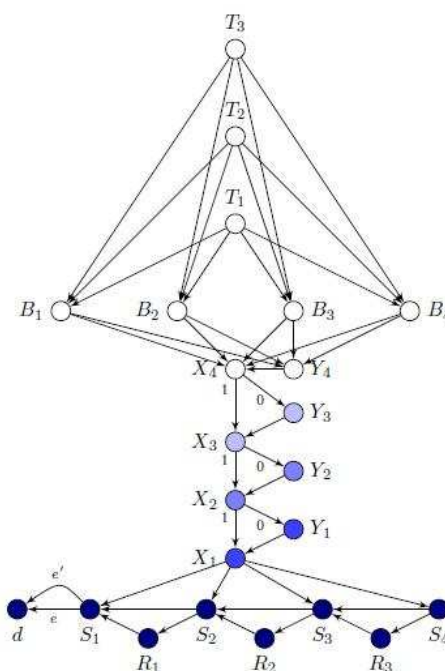
összefüggések teljesülnek, melyekben „ α ” a hálózat szintjeinek száma, „ m ” a hálózatban található linkek száma, „ t_* ” a hálózatban használt legalacsonyabb, míg „ t^* ” a legmagasabb MRAI érték. Ezen összefüggéseket a következők szerint értelmezhetjük:

- 1) A konvergencaidő azonos méretű (megegyező szintszámú) topológiákon a legmagasabb MRAI értékkel lesz arányos, függetlenül az összes többi beállítás értékétől – vagyis amíg a beállítások csökkentését az összes résztvevő nem végzi el, addig a konvergencia nem fog javulni.
- 2) Az elküldésre kerülő üzenetek mennyisége arányos lesz a MRAI diszparitással – vagyis amíg a legmagasabb MRAI érték nem kerül csökkentésre, addig az üzenetek száma sem csökken.

- 3) Az elküldött üzenetek száma a MRAI diverzitás növekedésével exponenciálisan nő - vagyis, azonos MRAI diszparitás mellett minél *többféle* érték kerül alkalmazásra t^* és t^* között, annál több üzenet kerül elküldésre.

Az első két állítás értelmében, amíg nem csökkenti a beállításokat mindenki, addig a konvergencia nem javul – ám a harmadik állítás mondanivalójában rejlik a lényeg, miszerint a helyzet nem csak hogy nem javul, hanem *romlik*.

Amit fontos megjegyezni, hogy ezek az összefüggések legrosszabb esetekre („*worst case*”) lettek definiálva, ezen összefüggések teljesülésének valós esélyéről a cikk nem nyilatkozik. A másik, a szerzők által is „hiányossággént” említett tulajdonság, hogy az összefüggések meglehetősen valószerűtlen elrendezéseken végzett számításokra lettek alapozva – olyan topológiákra, melyek az Interneten (vagy bármilyen valós BGP hálózaton) nem túl sok eséllyel vannak jelen. Egy ilyen elrendezést, a „karácsonyfa elrendezést” láthatjuk a 2.4. ábrán:



2.4. ábra: A „Karácsonyfa-elrendezés” [5].

Pontosan ezen hiányosságok miatt merül fel a cikk szerzőiben is az igény a feltételezett jelenségek szimuláció útján történő vizsgálatára. Fontos azt is megjegyezni, hogy a cikkírók az elmélet megalkotása során minden résztvevőtől „túlzó” („*crude*”) MRAI használatot feltételeztek – vagyis azt az esetet, melyben a MRAI használója az ajánlást be nem tartva, *minden* két update üzenet kiküldése között kivárja az adott időt.

Ez a feltételezés jogos, hiszen az Interneten található AS-ek legtöbbje is pontosan így használja a MRAI időzítőket – azonban érdemes lesz megvizsgálni az összefüggéseket ettől eltérő MRAI használati mód esetén is.

3. Konvergencia-vizsgálatok

3.1 A készített szimulátor bemutatása

Mint korábban említésre került, vizsgálataim elvégzéséhez egy meglévő szimulátor használata helyett egy saját, egyszerűsített BGP-szimulátort készítettem. Ezt a döntést több okból kifolyólag hoztam meg: Első sorban, olyan általános szimulátort nem találtam, mely az én céljaimnak maradéktalanul eleget tenne, és mindemellett nagy hatékonysággal dolgozik. A vizsgálataim középpontjában a 2.2 *fejezetben* ismertetett, a MRAI időzítővel kapcsolatos összefüggések vizsgálata állt - azok a szimulátorok azonban, melyekben a BGP MRAI időzítői is implementálásra kerültek, túlságosan sok egyéb részletet tartalmaztak ahhoz, hogy velük a szimulációt nagyméretű topológiákra, egyetlen munkaállomáson, nagy hatékonysággal futtatni lehessen. Egy másik szimulátor, a C-BGP (mely a nagy topológiákra való jó skálázódási tulajdonságairól közzismert) viszont túlságosan sok egyszerűsítést tartalmazott ahhoz, hogy a céljaimnak megfelelően – ugyanis többek között pont a MRAI időzítők implementálásától tekintett el a nagyobb hatékonyság elérése érdekében. Nekem tehát egy olyan szimulátorra volt szükségem, mely lehetőleg a C-BGP-hez hasonlóan minél több, a vizsgálataimhoz „irreleváns” részletet mellőz, ám képes a MRAI időzítők kezelésére. Mivel a feladat célkitűzése túlságosan specifikus, így érthető, hogy ennek eleget tevő szimulátorral nem találkoztam - ezért döntöttem úgy, hogy magam írok egyet. Az alábbi fejezet ismerteti az elkészült szimulátor tulajdonságait, a benne végbemenő folyamatokat és a megvalósításra került BGP elemeket. A függelékben található a program használatához egy részletes útmutató is.

3.1.1 Általános tulajdonságok

A program C++ nyelven íródott, a Bloodshed cég Dev-C++ fejlesztőeszközének segítségével. A kód fordítása a fejlesztőeszköz beépített compilerével, a MinGW-vel történt. A program által kezelt adatok C++ Standard Template Library (STL) dinamikus tárolókban (mint pl. az STL list és az STL vector) kerültek eltárolásra.

A program parancssorból vezérelhető, a vizsgált topológia számos paramétere a szimulációt megelőzően egy-egy paranccsal megváltoztatható – így pl. a linkek típusa, a linkeken használt MRAI időzítők végértéke, kimenő szűrők alkalmazása, stb. A vizsgálni kívánt topológia a program indulása után parancssorból manuálisan felépíthető, vagy megadott formátumú topológia-fájlból betölthető. Hasonlóképpen, a betöltött topológián a MRAI időzítők értékei kézzel állíthatók, vagy pedig egy megadott formájú MRAI konfigurációs file-ból utólag betölthetők. A szimulátor által szolgáltatott eredmények fájlba menthetők, vagy a képernyőre kiírathatók.

Mivel a szimulátor elsősorban a MRAI időzítőknek a BGP konvergenciájára gyakorolt hatását vizsgálja, ezért az *1.2. fejezetben* már említett MRAI használati mód is parancssorból állítható. A program 3 lehetséges módot kínál a MRAI időzítők beállítására, ezek az alábbiak:

- 1) *Dedicated, Non-blocking*: Ez a szabvány által is ajánlott eljárás, mely szerint minden router²¹ minden linkjén minden célpont felé külön időzítőt tart fenn („dedicated”). Mikor a kimenő üzenetsorban (buffer) több üzenet várakozik, és a sor elején álló üzenetet a MRAI le nem telte miatt késleltetni kell, úgy a végrehajtás a sorban következő üzenetre ugrik. Ha ezt az üzenetet sem sikerül kiküldeni a MRAI le nem járta miatt, akkor ugrunk a következőre stb. mindezt addig, amíg egy üzenetet ki nem küldünk, vagy a sor végére nem érünk („non-blocking”).
- 2) *Dedicated, Blocking*: Ez a használati mód egy köztes átmenet a szabvány szerinti, és a „túlzó” megoldás között. Hasonlít az előző módhoz, vagyis minden linken minden célpontra külön órát tartunk fenn („dedicated”) ám ha a sor elején álló üzenetet nem küldjük ki, akkor nem vizsgáljuk a további üzenetek kiküldhetőségét – így a többi üzenetnek meg kell várnia a sor elején álló üzenet elküldését (effektíve az elől álló üzenet „bedugítja” a sort – innen a „blocking” elnevezés).
- 3) *Crude* – Ez a megoldás a sokak által a valóságban is alkalmazott „túlzó” megoldás, vagyis mikor minden linken csak egyetlen MRAI órát használunk,

²¹ Mivel a szimulátor AS-szintű modellre épít, így néhány helyen mostantól a „router” és „AS” szavakat rokonértelműként fogjuk használni

és ez *minden* két frissítési üzenet kiküldése között az adott idő elteltét kivárja.

Érdemes megjegyezni, hogy az 1-es beállítás használata esetén a szimuláció futásának ideje már kisebb topológiák esetén is észrevehető mértékben megnő a jóval több MRAI időzítő által okozott extra komplexitásnak köszönhetően. Ugyanígy, mikor a MRAI használati módjának a konvergencia-folyamatra gyakorolt hatását vizsgáljuk, érdekes eredménybeli különbségeket tapasztalhatunk. Ezeket az eredményeket a 3.2 *fejezet* részletesen ismerteti.

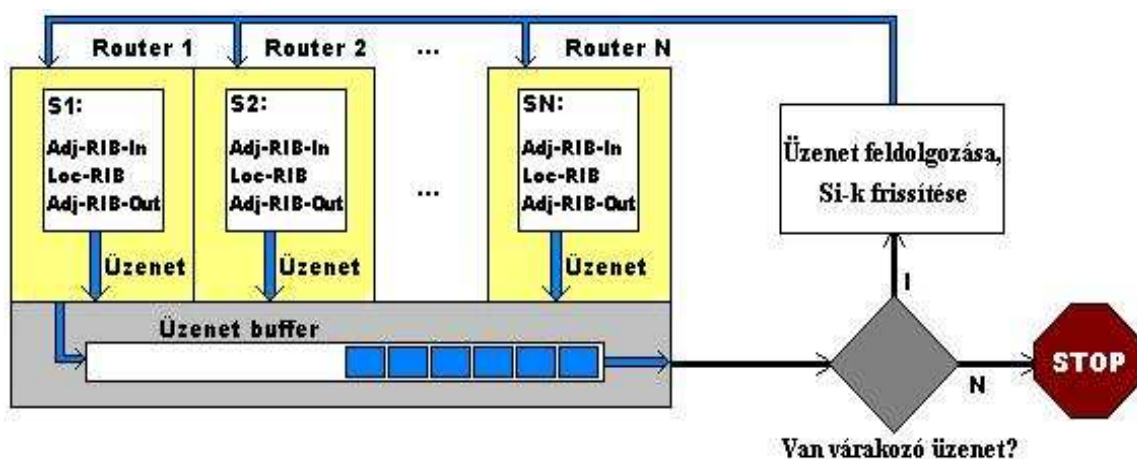
A szimulátor mellőzi a BGP szomszédságok kiépítésére használt állapotgépet (*Finite State Machine, FSM*), a BGP által a szállítási rétegben használt TCP protokollt, a csupán az iBGP-re jellemző vonásokat, az IP címek használatát (így az útvonal-összevonást), valamint a protokollüzenetek pontos, szabványos formátumának betartását. Azonban a kiválasztási folyamat vonatkozó lépéseit, az üzenetek generálását és kiküldését, valamint a MRAI időzítők használatát teljesen a szabvánnyal konform módon valósítja meg.

3.1.2 Az idő kezelése

A vizsgált topológiában a résztvevő routerek üzeneteket fogadnak, dolgoznak fel, majd további üzeneteket generálnak – szükséges, hogy ezen nagyszámú esemény lefolyását valamilyen módon sorrendezzük, koordináljuk. Az elkészült szimulátor a bekövetkező eseményeket, és azok lekezelését nem valós időben szimulálja, sokkal inkább az 1.3. *fejezetben* említett TES (Timeless Event Scheduling) módszerhez nagyban hasonló módon kezeli az eltelt időt. Ennek a módszernek a segítségével pl. megoldható, hogy olyan időintervallumokat „átugorjunk” amikor semmilyen változás nem történik a globális állapotban. Az alkalmazott módszer lényege, hogy a routerek által generált üzenetek a routerek kimenő soraiban folyamatosan gyűlnek - ám a vezérlés körkörös kiosztás, „*round-robin*” jelleggel vált routerről routerre, mindegyiknek pontosan 1 beérkező üzenet feldolgozására, és pontosan 1 kimenő üzenet kiküldésére adva futási jogot. Ilyen formán tehát minden „körben” a routerek csak egy-egy (illetve ha nincs kiküldendő üzenet, akkor egy sem) üzenetet küldenek ki a szomszédjaiknak – illetve a bemenő sorukból minden „körben” pontosan egy üzenetet dolgoznak csak fel – melyet adott esetben beépítenek a helyi útvonaltáblájukba,

lefuttatják rájuk a kiválasztási folyamatot, és esetleg hatásukra új, kiküldendő üzenetet generálnak.

A vezérlés minden körben az éppen futási jogot kapott routerrel „elhiteti”, hogy az előző kör óta egy bizonyos, fix időtartam telt el²² – így a szimuláció végéig eltelt összes ilyen „lépésköz” összege fogja kiadni a szimulált időtartamot. A routerek ezt az eltelt időt használják fel arra is, hogy MRAI időzítőiket léptessék – így adott esetben, ha egy router MRAI le nem telte miatt nem tud üzenetet kiküldeni, úgy a futási jogot visszaadja a vezérlésnek. Látható tehát, hogy a szimuláció futási ideje, és a szimulált időtartam teljesen függetlenek egymástól – így lesz lehetséges az is, hogy ugyanaz a szimuláció két, egymástól eltérő hardveres képességű munkaállomáson ugyanazt az eredményt fogja szolgáltatni, holott a szimulációk futási ideje valószínűleg lényegesen eltér. A futási folyamat szemléltetésére szolgál a 3.1. ábra:



3.1. ábra: A szimulátorban megvalósított futási folyamat

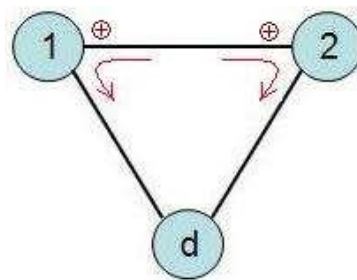
3.1.3 A szimulációk konvergencia-tulajdonságai

A szimulátor által használt Gao-Rexford modellhez kapcsolódó tétel bizonyos feltételek teljesülése esetén a BGP garantált konvergenciájáról nyilatkozik – ám mivel a szimulátor a BGP számos tulajdonságát nem teljesen, vagy egyáltalán nem valósítja meg, így lehetségesek olyan esetek, amikor ugyan a valós BGP konvergenciája garantált, a szimulátor mégis divergálni fog – vagyis az üzenetváltások folyamata soha

²² Ez a fix időtartam a szimulátorban 1ms és 100ms között állítható, és mostantól szimulációs lépésközként hivatkozunk rá.

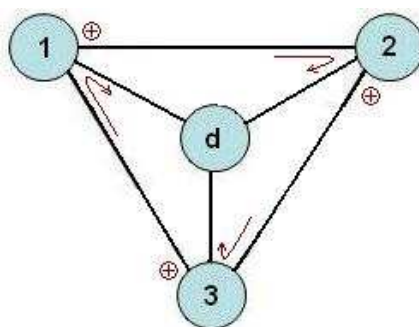
nem fejeződik be. Ezek olyan esetben fordulnak elő, amikor a valós BGP döntések kimenetele biztosan létrejön (vagyis a folyamat konvergens) ám az eredmény a kommunikáció aszinkronitása miatt nem lesz egyértelmű – a tényleges végállapot a pontos időzítési körülményektől függ. Azonban a szimulátorban az időkezelés megvalósításának módja miatt a kommunikáció *szinkron* – hiszen amint fentebb ismertettük, minden router adott fix időközönként kap „futási jogot”, mikor az üzeneteket elküldheti. Alapértelmezett esetben minden routerrel „elhitetjük”, hogy a legutóbbi futása óta 100ms telt el (ez a lépésköz állítható), így a globális időskálát tekintve *minden* router a 0ms, 100ms, 200ms stb. időpillanatokban fogja az üzeneteket elküldeni – innen a tulajdonképpeni szinkronitás.

Két egyszerű példa olyan esetre, amikor a valóságban az aszinkronitás fog dönteni, és a szimulátorban pedig divergenciát tapasztalunk, az 3.2. és a 3.3. *ábra* mutat.



3.2. *ábra*: A „huncut elrendezés”

Az 3.2. *ábrán* az úgynevezett „*huncut elrendezés*” (*naughty-gadget*) látható. Egy olyan esetet láthatunk, melyben az 1-es és 2-es router is ismeri a közvetlen kapcsolatot a „d” router felé, de emellett mindketten tudnak a másikon keresztülhaladó közvetett útvonalról is, és mindketten a másikon keresztül való továbbítást preferálják (magasabb local preference értéket határoztak meg a közvetett úthoz, mint a közvetlenhez - az *ábrákon* a „+” jelek a routerek legmagasabb preferenciaértékeit jelölik). Egy ilyen elrendezés esetén a valóságban a kialakuló végállapot attól fog függeni, hogy a két router közül melyik küldi el előbb az update üzenetet a másiknak a saját útvonaláról – vagyis, hogy ki szerez tudomást előbb a másikon keresztülhaladó közvetett útról. Azonban mivel a szimulátorban ezek az üzenetek „egyszerre” küldődnek el, így a kettejüket összekötő vonalon egymást váltó útvonal-frissítő és útvonal-visszavonó üzenetküldések véget nem érő sorozatát fogjuk tapasztalni.



3.3. ábra: A „rossz elrendezés”

A 3.3. ábrán az úgynevezett „rossz elrendezést” (*bad gadget*) láthatjuk. Tulajdonképpen ez az előző eset kiterjesztése, melyben a probléma gyökere szintén az, hogy mindenki a saját, közvetlen útja helyett egy valaki máson keresztül vezető közvetett utat preferál.

3.1.4 Implementált BGP üzenetek

Mivel a szimulátor minél több, a vizsgálatunk által nem érintett részlet mellőzésével él, így a BGP üzenetek közül az Open, Keepalives és Notification üzenetek használatától is eltekint. Mivel a szimulátorban a BGP szomszédságok kiépítésekor használatos véges automata nem került megvalósításra, így az ennek vezérlését végző Open üzenetekre sincs szükség. Mivel alsóbb rétegbeli protokollok és egyéb, nem közvetlenül a BGP-hez kapcsolódó hálózati tulajdonságok sem kerültek implementálásra, a szimulátorban kiküldött üzenetek minden esetben hiba nélkül érkeznek meg a címzetthez. Emellett, mivel a szimulált üzenetek pontos formátuma sem egyezik a szabványban rögzítettel (lásd később), így protokollhibákról sincs értelme beszélni – emiatt a hibajavításért felelős Notification üzenetektől is eltekintünk.

A kapcsolatok aktív állapotban tartásáért felelős Keepalives üzenetek is kimaradtak a szoftverből. A valós BGP kommunikációban egy router akkor tekinti elveszettnek egy szomszédját, ha tőle bizonyos ideig nem érkezik keepalives üzenet – ebben az esetben a router megkezdi a topológiaváltozáskor szükséges lépések végrehajtását. A szimulátorban egy szomszéd „elvesztésének” a mechanizmusa lényegesen egyszerűbb – mivel a konvergált hálózaton a topológiaváltozást (pl. linkszakadás, router hiba) a felhasználó a parancssorból kezdeményezi, így ekkor az

érintett routereken a megfelelő „szomszéd-elvesztő” függvény meghívódik. Ezután kezdődik meg a routerek topologiaváltozást követő egyezkedése.

Az egyetlen üzenettípus tehát, mely a szimulátorban megvalósításra került, az update üzenet. Mint ahogy az *1.2. fejezetben* is láthattuk, egy update üzenet meglehetősen sok (kötelező vagy opcionális) paramétert tartalmazhat. Mivel ezek közül sokat a szimulátorban nem is használunk (hiszen pl. az őt használó folyamat meg sincs valósítva – pl: MED attribútum és iBGP) így az update üzenetből számos paraméter ki lett hagyva. Ezen felül – mivel nem célunk valós BGP routerekkel kommunikálni – az üzenet pontos formátumának betartását is mellőztük, így a fejlécek, flag-ek, bitminták is a valós protokolltól eltérnek²³. Így tehát a szimulátorban megvalósított update üzenet egy egyszerű „Message” névre hallgató C++ osztályként került megvalósításra, mely az alábbi paramétereket, mint tagváltozókat tartalmazza:

- 1) Küldő router azonosítója (AS-száma)
- 2) A vonal Gao-Rexford típusa, melyen át az üzenet érkezett (p2c, p2p, c2p)
- 3) Az üzenet által hordozott útvonal AS-Path attribútuma.

Láthatjuk, hogy ez a valós update üzenet tartalmánál nem csak kevesebb, de valamivel el is tér tőle (hiszen a Gao-Rexford típus természetesen a valós BGP-ben nem jelenik meg paraméterként). Azonban a vizsgálatainkhoz ennyi paraméter éppen elegendő.

A szimulátorban a routerek az útvonalak közötti különbségtétel során (a „tie-breaking” folyamatban) a korábban ismertetett egyszerűsített, néglépéses folyamatot alkalmazzák – melynek során összesen három attribútum vizsgálatára kerül sor. Az első lépésben, mikor azt vizsgáljuk, hogy az útvonal tartalmaz-e hurkot, az AS-Path-t vizsgáljuk (a szabvány szerint, ha egy router megtalálja az AS-számát az AS-Path attribútumban, akkor az útvonal hurkot tartalmaz). Ugyanígy az AS-Path attribútumot vizsgálja a 3. lépés – mely az AS-Path hossza szerint priorizálja az útvonalakat. A negyedik (és egyben az utolsó) lépés, ha más alapján eddig nem tudtunk dönteni, a küldő router azonosítója alapján fogja kimondani a kiválasztási folyamat végső ítéletét.

²³ Jegyezzük meg, hogy az *1.2. fejezetben* épp emiatt a protokollüzenetek formátumáról nem is ejtettünk szót

Felmerül a kérdés, hogy az üzenetet küldő router felé vezető link Gao-Rexford típusának elküldésére miért van szükség. A magyarázat a következő: a kiválasztási folyamat második lépésében a Local Preference attribútumot vizsgáljuk – azonban mivel a szimulátorban kizárólag eBGP jellemzők kerültek megvalósításra, így a Local Preference nem az iBGP által is használt tranzitív, az üzenettel együtt kapott attribútum, hanem egy helyileg számított paraméter lesz. A szimulátorban a Local Preference számításának az alapértelmezett módja, a Gao-Rexford tétel 2. feltételének legegyszerűbb módon való betartása – vagyis, hogy a c2p, p2p és p2c élek között a megkövetelt preferencia-sorrend biztosított legyen. Így azonban látjuk, hogy a számított preferencia kizárólag az adott éltípus függvénye lesz – vagyis az üzenetben kapott Gao-Rexford paraméterből a local preference egyenesen számolható. Azonban ahogy korábban már említettük, a local preference számításának módja a programkód minimális megváltoztatásával módosítható, ezáltal a szimulátorba változatos irányelvek bekonfigurálására is megteremthető a lehetőség. Megjegyzendő továbbá, hogy a program lehetőséget kínál minden fizikai linkhez sávszélesség és késés (latency) érték hozzárendelésére, mely tulajdonságokat ugyan a program jelen állapotában nem használ – ám későbbi, esetleges „*Future Internet*” vizsgálatok esetén ezen paraméterekből számolandó Local Preference attribútum segítségével akár rendhagyó, kísérleti jellegű irányelvek is megalkothatók, így módosított, „BGP-szerű” protokollok viselkedésének vizsgálata is lehetségessé válik.

Vegyük észre, hogy a döntések meghozásának során többször is használt Next-Hop attribútum az üzenetekből kimaradt. Azonban emlékezzünk, hogy a Next-Hop attribútum a valós BGP-ben a next-hop router IP címe - a szimulátor azonban IP címeket nem használ, helyette a routereket AS-számuk alapján azonosítja. Másrészt, a szimulátor eltekint a szabvány által engedélyezett külső, „third-party” next-hopok használatától. Ezekből kifolyólag a szimulációban a next-hop router AS-száma nem lesz más, mint az AS-Path attribútum első eleme, hiszen a next-hop AS az útvonal során keresztezett AS-ek közül az első lesz.

Az szimulátor által megvalósított üzenetküldési forma némileg eltér a szabványostól. Ennek okát a következőkben ismertetjük.

A szabvány az update üzenetekben elküldhető útvonal-frissítésekre és útvonal-visszavonásokra a következőket írja elő: *„Egy update üzenet legfeljebb egy útvonalat hirdethet, melyet számos, az üzenethez csatolt attribútummal írhat le. [...] Egy update üzenetben több útvonal felsorolására van lehetőség, melyet a helyi BGP router a*

szolgáltatból vissza szeretne vonni. [...] Egy update üzenetben lehetőség van kizárólag visszavonandó útvonalak hirdetésére, mely esetében az üzenet nem fogja tartalmazni a frissítendő útvonalakra vonatkozó információs mezőket. Ellenben lehetőség van kizárólag útvonalfrissítés elküldésére is, mely esetében a visszavont útvonalak attribútumait leíró mezők maradnak üresen.” [8]

Vegyük észre, hogy az útvonalfrissítésekre vonatkozó szövegrész egyértelműen kimondja, hogy egy update üzenetben legalább 0, de legfeljebb 1 útvonalfrissítés kiküldésére van lehetőség – ám a szöveg nem tesz megkötést az egy update üzenetben elküldhető útvonal-visszavonások maximális számára - mindössze annyit tudunk, hogy legalább 0, legfeljebb tetszőleges számú visszavonást csomagolhatunk egy üzenetbe.

Ismét egy olyan pontot találtunk a szabványban, mely a részleteket az implementálóra bízta – azonban nekem a szimulátor írásakor el kellett döntenem, hogy az üzenetek küldése során hogyan kezeljem ezt a kérdést. Az egyik lehetőségem az volt, hogy minden körben, mikor egy router futási (és ezzel üzenetküldési) jogot kap a vezérléstől, úgy kiküld egy (vagy ha nincs a kimenő sorában ilyen, akkor nulla) útvonalfrissítést, illetve az összes, a kimenő sorában várakozó visszavonást (ami ez által lehet nulla, de legfeljebb akárhány). Egy másik lehetőség, hogy egy routernek minden körben 0 vagy 1 frissítés, illetve csak 0 vagy 1 visszavonás kiküldését engedélyezzük. A program végső változatában a kérdés eldöntése a felhasználóra van bízva, ugyanis a program lehetőséget teremt a visszavonások kiküldési módjának parancssorból való megváltoztatására. A két lehetőség, mely megvalósításra került az alábbi:

- 1) *Single (egyenkénti)* – Minden router minden körben legfeljebb egy visszavonást küldhet el.
- 2) *Bulk (köteget)* – Minden router minden körben a kimenő bufferéből adott számú visszavonást küld el (message bulk) – köteg mérete globálisan állítható (tehát minden routernek ugyanakkora lesz a „kötegmérete”). Amennyiben egy routernek a kötegméretnél kevesebb várakozó üzenete van, úgy az összeset elküldi.

Ugyanazon topológiákon az üzenetküldési mód változtatásával, de minden egyéb paraméter változatlanul hagyásával futtatott szimulációk eredménye között érdekes különbségeket tapasztalhatunk. Ezeket az eredményeket a 3.2. *fejezet* ismerteti.

3.2 Szimulációk, kapott eredmények

3.2.1 Előkészületek

A szimulátor megírását követően elkészítettem azokat a minta-topológiákat, melyeken a 2.2 fejezetben ismertetett feltételezéseket ellenőrző szimulációkat végeztem. Mint említettük, a cikk szerzői maguk is negatívumként említik, hogy elméleteiket viszonylag „valószerűtlen” elrendezésekre alapozva alkották meg. Mivel a Gao-Rexford modell pontosan az általános ésszerűségekre alapozva modellezi a BGP hálózatokat, így szimulációimat én is ilyen Gao-Rexfordi topológiákon futtattam. Két ilyen elrendezés rajza, melyekre a legtöbb szimulációt futtattam, jelen fejezet utolsó oldalain látható.

Ezek a topológiák különböző MRAI értékek konfigurálásával *teszteseteket* hoztam létre, melyek tulajdonképpen ugyanazon topológián, ugyanannak a topológia-változásnak különböző MRAI konfigurációk melletti hatását vizsgálják. Próbáltam olyan teszteseteket megalkotni, melyekben jól elkülönülnek az egyes, az elmélet által definiált tulajdonságok - így a MRAI diszparitás és a MRAI diverzitás. Mivel a MRAI időzítő az IETF által ajánlott beállítási értéke annak érvénytelenítése előtt 30 másodperc volt, ezért csak néhány olyan „szélsőséges” esetet vizsgáltam, melyben a routerek 30 másodpercnél magasabb MRAI értéket használnak – ugyanis azzal a feltételezéssel éltem, hogy az ajánlás megszűnése óta az egyes AS-ek a gyorsabb konvergencia érdekében csak csökkentették, és nem növelték a MRAI beállításait.

A szimulációk futtatása során igen meglepő jelenségeket is tapasztaltam. Korábban említést tettünk arról, hogy a szabvány megengedő a MRAI időzítők használatának módját illetően – így az ajánlott használattal ellentétben sokan „túlzó” módon használják a MRAI-kat, vagyis minden két üzenet küldése között várakoznak. A szimulációk rávilágítottak arra, hogy az ajánlás be nem tartása súlyos inkonzisztenciákat okozhat a BGP döntési folyamatában.

Ahogy szintén említettük, a szabvány az egy update üzenetben maximálisan küldhető visszavonások számára sem tesz megkötést (erre ajánlást sem tesz). Megmutatjuk, hogy az egy update üzenetben kiküldött visszavonások számának megválasztása is nagyban befolyásolja a konvergencia-folyamat kimenetelét.

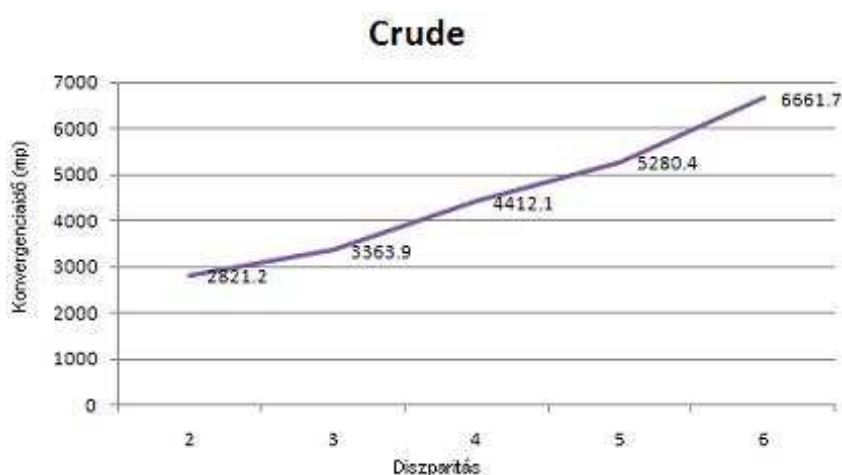
Végül amint arról már szintén szó esett, az IETF legújabb ajánlása értelmében az útvonal-visszavonásokat is ajánlott MRAI-val korlátozni, ám nem kötelező. Bemutatjuk, hogy a visszavonások késleltetése, illetve a késleltetés alól való felmentése esetén is számolnunk kell bizonyos problémákkal.

3.2.2 Kapott eredmények

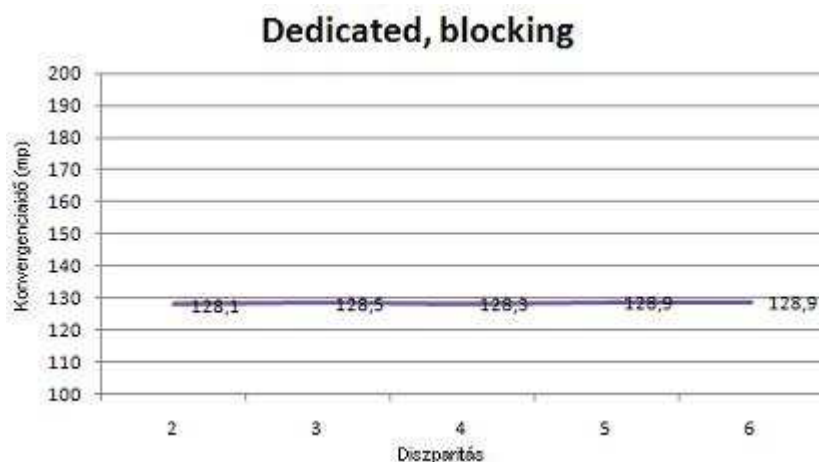
Jelen pont ismerteti az egyes teszteseteken futtatott szimulációk eredményeit. A tesztestek lefuttatásával a szimulátor által szolgáltatott adatokat táblázatba gyűjtöttem, majd a kapott számokból diagramokat készítettem. Nem minden diagram került bele a fejezetbe, és nem is minden adatból készült diagram – a teljes adathalmaz a dolgozat mellékletében található.

3.2.2.1 MRAI diszparitás hatása a konvergenciaidőre:

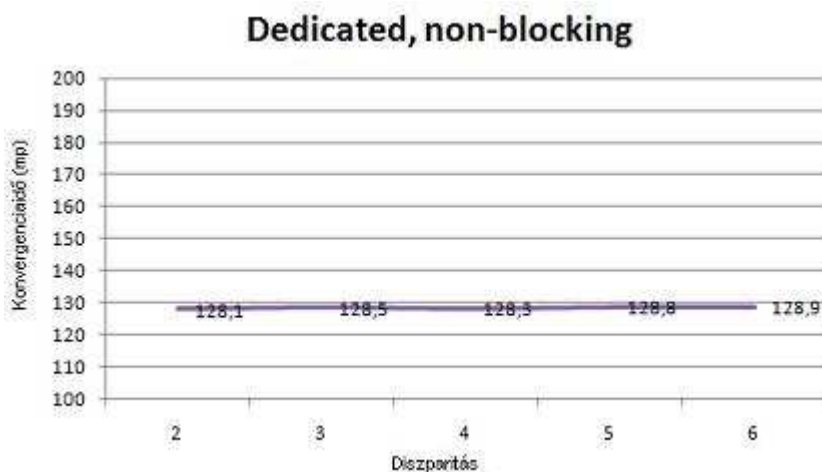
A vizsgálatot elvégeztem a szimulátor által támogatott mindhárom MRAI használati mód („*crude*” – *túlzó*, „*dedicated-blocking*” – *köztes*, illetve „*dedicated non-blocking*”- *szabványos*, ezek magyarázatát lásd a 3.1. fejezetben) mellett. A kapott eredmények meglepetésre adnak okot:



3.4. diagram: A konvergenciaidő és a MRAI diszparitás függése túlzó használat esetén



3.5. diagram: A konvergenciaidő és a MRAI diszparitás függése köztes használat esetén



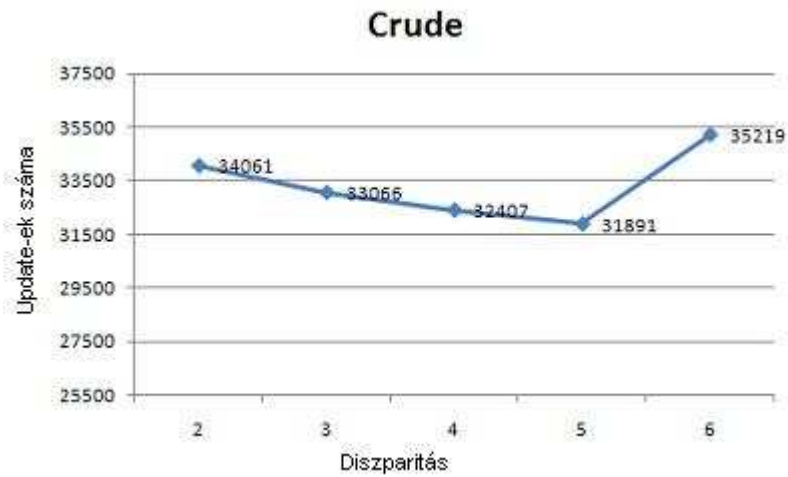
3.6. diagram: A konvergenciaidő és a MRAI diszparitás függése szabványos használat esetén

Az első és legszembetűnőbb dolog, hogy túlzó használat esetén a konvergenciaidőben a többi használati módhoz képest nagyságrendi különbség van. A másik, ami egyértelműen látszik: A diszparitás kizárólag túlzó használat esetén fogja növelni a konvergenciaidőt, a többi használati mód esetén nincsen rá hatással. Ez gyakorlatilag azt jelenti, hogy bármilyen hosszú is a hálózatban alkalmazott leghosszabb késleltetés – a MRAI szabványos használata esetén ez a konvergencia idejére *nem lesz befolyással*²⁴.

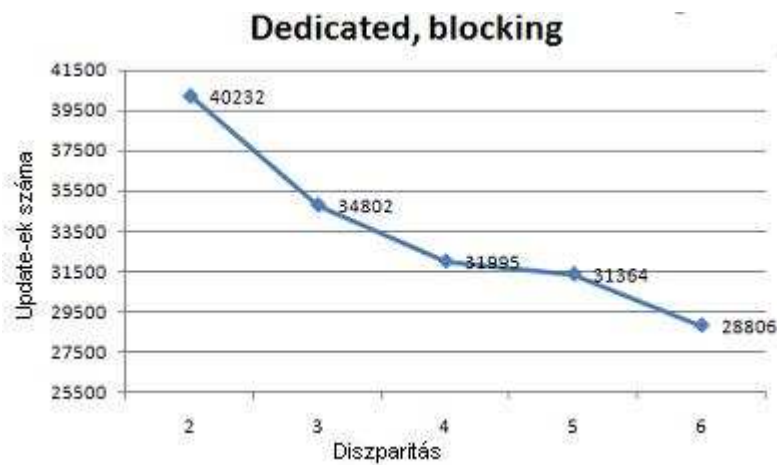
²⁴ Emlékezzünk, hogy a szimulátor által szolgáltatott konvergenciaidő-értékek tájékoztató jellegűek – a szimuláció során a routerekkel „elhitetjük” hogy két futási jog megadása közt adott idő (alapértelmezett lépésköz: 100ms) telt el – a végeredmény ezen lépésközök összege lesz. Az itt bemutatásra kerülő eredményeket 100ms-os lépésközű szimulációkkal kaptam.

3.2.2.2 MRAI diszparitás hatása az elküldött üzenetek számára:

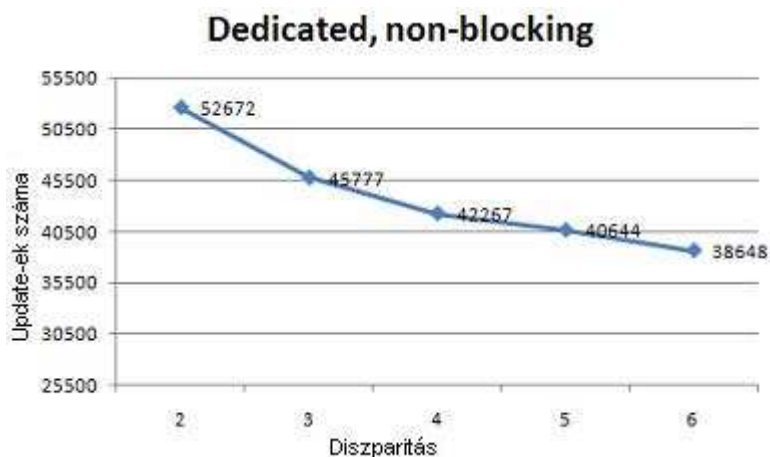
A vizsgált elmélet a diszparitás növekedésével a küldött üzenetek számának lineáris növekedését jósolja. A kapott eredmények azonban ennek ellent mondanak:



3.7. diagram: A küldött üzenetek száma és a MRAI diszparitás függése túlzó használat esetén



3.8. diagram: A küldött üzenetek száma és a MRAI diszparitás függése köztes használat esetén



3.9. diagram: A küldött üzenetek száma és a MRAI diszparitás függése szabványos használat esetén

Az üzenetek számának növekedését a három eset közül egyikben sem tapasztaljuk - sőt, a második két használati mód esetén egyértelmű csökkenés figyelhető meg. Ugyan a cikk írói minden résztvevőtől túlzó használati módot feltételeztek, ám a jóslott növekedés ebben az esetben is elmarad.

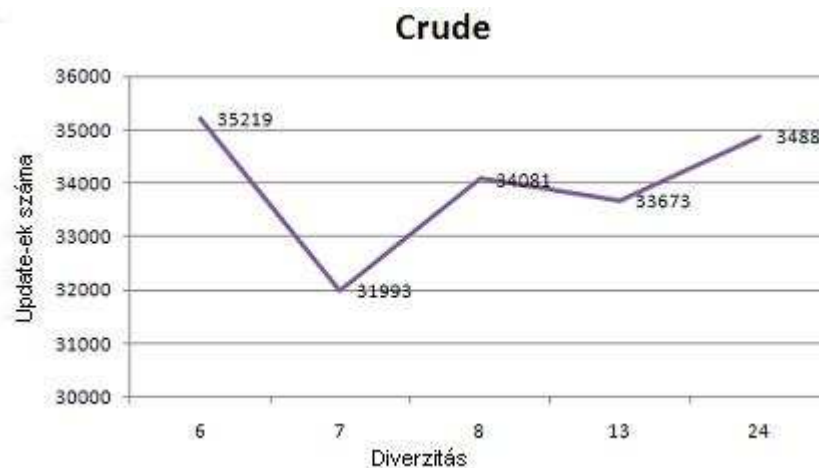
Ami még szembetűnő, hogy túlzó használati mód esetén az elküldött üzenetek száma jóval kisebb, mint köztes, illetve szabványos használat esetén. Erre a következő a magyarázat: Mivel a szabvány előírja, hogy amennyiben a kimenő bufferben egy frissítési üzenet MRAI időzítő lejáráására várakozik, ám eközben érkezik a bufferbe egy újabb, ugyanarra a célpontra mutató frissítés, úgy az első üzenetet nem szabad kiküldeni, hiszen az közben idejét múlttá vált. Mivel túlzó használati mód esetén egy üzenet átlagosan jóval több időt tölt a bufferben várakozva, mint a többi esetben, így többször kerül sor ilyen elavult üzenetek felülírására. Mivel átlagosan a legkevesebb üzenetvárakoztatás a szabvány szerinti módban történik, úgy ott ezek az üzenetek kiküldésre kerülnek, még mielőtt kiderülne róluk, hogy elavultak. Ez tekinthető a sávszélességgel való pazarlásnak.

Másfelől, a szimulációk futtatása során egyértelműen érződött a számításigénybeli eltérés a három mód között. A rengeteg MRAI időzítő menedzseléséből adódó megnövekedett erőforrásigény miatt a szimuláció futási ideje gyakran akár másfélszer hosszabb is volt a szabványos módszer esetén, mint túlzó esetben. Kijelenthetjük tehát, hogy a túlzó módszer alkalmazása sokkal lassabb konvergenciához vezet, mint a szabványos használat, ám jóval kevesebb sávszélességet, memóriát és számítást igényel. A köztes megoldás gyakorlatilag pontosan ugyanannyi

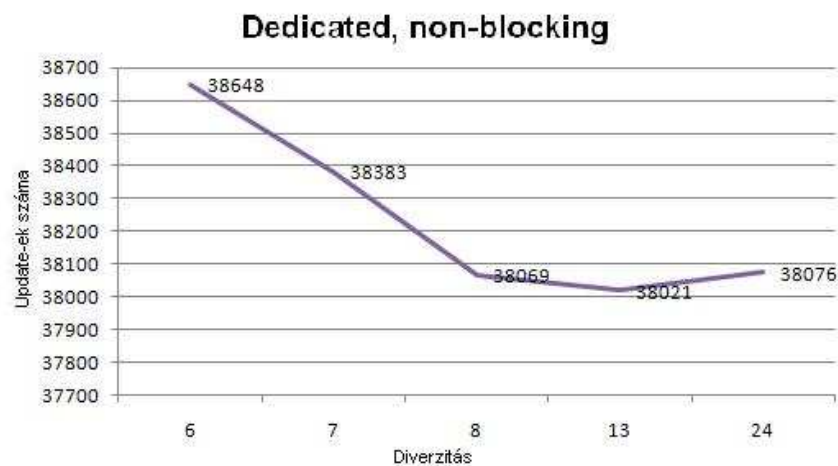
memóriát és számítást igényel, mint a szabványos eljárás, ám valamivel kisebb a sávszélesség-igénye.

3.2.2.3 MRAI diverzitás hatása az elküldött üzenetek számára:

Ez a jelenség az elmélet által állított „legerősebb” állítás – vagyis ez az a pont, amiben egy heterogén hálózat valóban *rosszabb* lesz, mint ugyanaz a hálózat homogén beállítások mellett. A feltételezések a diverzitásban exponenciális üzenetszámnövekedést jósolnak. Nagy megkönnyebbülésünkre azonban, a szimulációk során nem ezt tapasztaljuk:

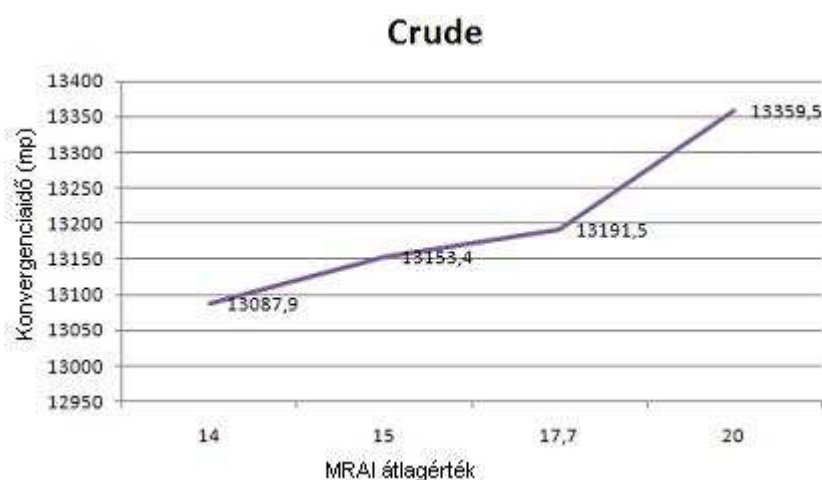


3.10. diagram: A küldött üzenetek száma és a MRAI diverzitás függése túlzó használat esetén



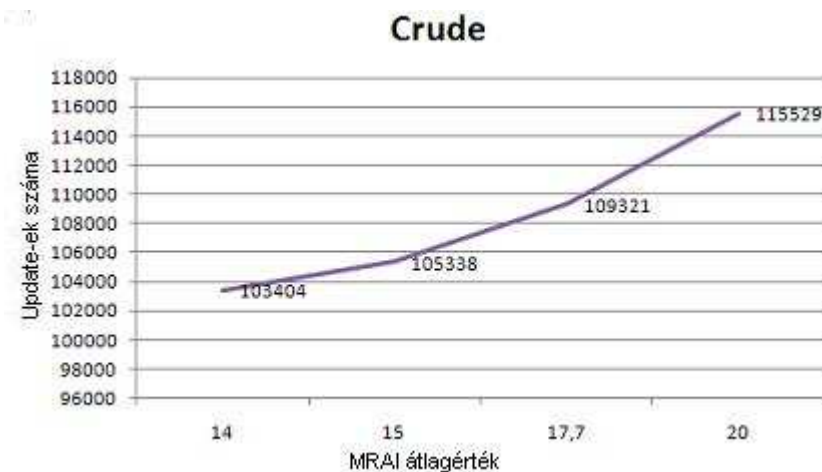
3.11. diagram: A küldött üzenetek száma és a MRAI diverzitás függése szabványos használat esetén

Túlzó esetben tulajdonképpen semmilyen összefüggés nem vehető észre a két érték között, szabványos esetben pedig még egy csökkenés is tapasztalható. Kijelenthető tehát, hogy csupán a használt értékek halmazának számossága nem fogja önmagában befolyásolni a kiküldésre kerülő üzenetek számát. Azonban a szimulációk során tettem egy észrevételt – a MRAI értékek halmazának *számosságától* az üzenetszám ugyan független, azonban a használt MRAI időzítők *értékeitől* igenis függ mind a konvergenciaidő, mint az üzenetek száma. Egy diagramon ábrázoltam ezek alakulását a használt MRAI-k átlagos értékének függvényében:



3.12. diagram: A konvergenciaidő és a használt MRAI értékek átlagának függése túlzó módban.

Igazából pontosan ez az, amit a józanész is diktál – a rendszerben átlagosan minél tovább késleltetünk egy üzenetet, annál lassabb lesz a konvergencia. Ami azonban meglepő lehet, hogy nem csak a konvergencia lesz lassabb, de több lesz az elküldött üzenet is:



3.13. diagram: A küldött üzenetek száma és a használt MRAI értékek átlagának függése túlzó módban.

Ami érdekes, hogy szabványos használat esetén a konvergencia ideje, és az üzenetszám is *független lesz* a MRAI-k átlagos értékétől (ehhez a fejezetben mellékelt ábra nincs, a dolgozat mellékletében található adatokból azonban kiolvasható). Így tehát igaz az, hogy *szabványos* használat esetén kizárólag a leglassabb MRAI érték fogja meghatározni a teljes rendszer konvergencia-tulajdonságait. Túlzó esetben ez nem teljesül – ott a rendszer átlagos MRAI-értéke lesz meghatározó.

Érdekes, és tulajdonképpen megnyugtató eredményeket kaptunk. A cikk íróinak állítása - miszerint amíg a rendszerben a leglassabb MRAI beállítást nem veszik gyorsabbra, addig a többi beállítás értékétől függetlenül a konvergencia nem javul – csak szabványos használat esetén teljesül. Azonban mivel az Internetet alkotó AS-ek közül legtöbbször nem a szabvány szerint használják a MRAI időzítőket, úgy a szimulált eredmények értelmében a MRAI időzítők inkrementális átkonfigurálása is meghozza a kívánt javulást – így tehát nincs okunk aggodalomra, hiszen a konvergencia javulásához nem kell „megvárni” amíg a leglassabb beállítás is alkalmazkodik a többiekhez.

3.2.3 Tapasztalt jelenségek

A szimulációk futtatása során az elmélet legtöbb állításának nem teljesülése mellett igen érdekes jelenségeket is tapasztaltam. Az alábbi pont ezeket ismerteti.

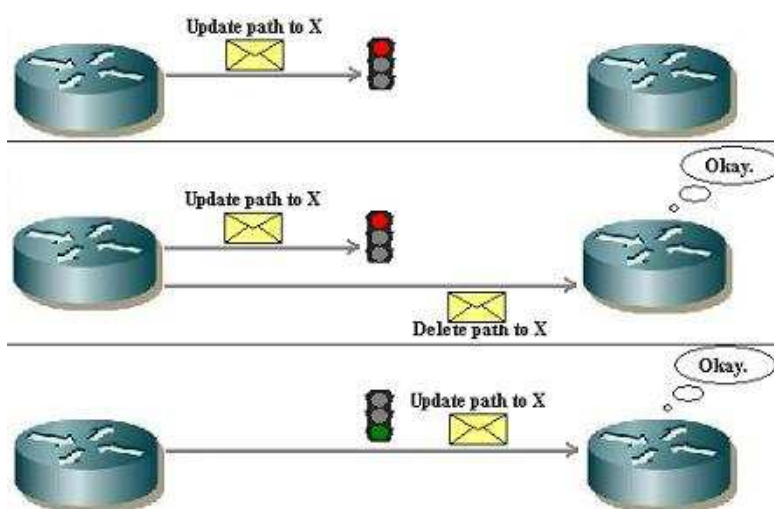
Mivel a szimulátor fel lett készítve az RFC által nem teljesen specifikált szempontok kezelésére is (eltérő MRAI használat, visszavonások kiküldésének üteme, visszavonások késleltetése), így arra vonatkozóan is futtattam szimulációkat, hogy az ezekre vonatkozó eltérő megvalósítások mennyiben befolyásolják a konvergencia folyamatát. A tapasztalt jelenségek meglepetésre adnak okot, ugyanis bizonyos viselkedési módok esetén a konvergencia *hibás végállapot* kialakulására vezet.

Észrevettem, hogy sok esetben számos router útvonaltáblájában „bennragadnak” bizonyos útvonalak, amiknek a konvergencia során ki kellett volna törlniük. Egy ilyen példa: a topológiából eltávolítunk egy routert, ám sokaknak mégis marad néhány olyan bejegyzése, amely útvonalak továbbra is ezen az eltávolított routeren vezetnek keresztül. Észrevettem azt is, hogy ez a jelenség annál nagyobb eséllyel fordul elő,

minél távolabb (minél több ugrásra) vagyunk az adott topológia-változástól. Továbbá, a jelenség legnagyobb számban túlzó MRAI használat esetén fordul elő, szabványos üzem esetén pedig szinte soha.

A jelenségre a magyarázatot hosszas hibakeresés után találtam meg, mely a következő: mivel a jelenséget olyan szimulációk futtatásakor tapasztaltam, melyben az útvonal-visszavonásokat *nem késleltetjük* MRAI időzítőkkal (vagyis, az eredeti RFC 1771-nek megfelelő módon járunk el), így ebben az esetben az alábbi hibás működést tapasztaljuk:

Egy topológia-változás során az érintett routerek nagyszámú útvonal-frissítés kiküldésébe kezdenek. A döntési folyamat során egy adott célpont felé többször is megváltozhat a használt útvonal, ezek mindegyikéről egy üzenetfrissítés kerül kiküldésre – azonban mikor az adott célpont felé „elfogytak az ismert útvonalak”, úgy azt a helyi útvonaltáblából töröljük, és erről az eseményről útvonal-visszavonást küldünk a szomszédjainknak. Azonban mivel a frissítési üzeneteket késleltetjük, a visszavonásokat viszont késleltetés nélkül azonnal kiküldjük, ezért kialakulhat olyan eset, mikor egy router az ugyanarra a célpontra vonatkozó visszavonási üzenetet előbb kapja meg, mint a rá vonatkozó frissítési üzenetet. Ezt a fogadó router a következőképpen értelmezi: *„Érkezett X felé egy visszavonás, így a felé vezető utat töröltem – azonban a hiba idő közben helyreállhatott, ugyanis nem sokkal ezután kaptam felé egy frissítési üzenetet is. Ezért az útvonalat a táblámba beszúrom használatra”*. A jelenséget a 3.14. ábra szemlélteti:



3.14. ábra: A frissítések késleltetése hibás döntéshez vezethet.

Tovább rontja a helyzetet, ha az útvonal-visszavonásokat nem egyesével, hanem kötegelve („bulk”) küldjük (a kötegelt küldés leírását lásd a 3.1 fejezetben) – ennek pedig a szabvány értelmében nincsen semmilyen akadálya. A probléma abból fakad, hogy amennyiben egy üzenetben egynél több visszavonást is kiküldünk, úgy a visszavonások küldésének üteme még gyorsabb lesz, így még nagyobb arra az esély, hogy túl korán fognak célba érkezni.

A 3.1. táblázat az egyes esetek szimulálása során kapott hibás, „bennragadt” útvonalbejegyzések számát mutatja. Vegyük észre, hogy túlzó esetben már a topológia-változástól mindössze három ugrásra is óriási mennyiségű hibás bejegyzésünk keletkezik! Ennek oka az, hogy túlzó esetben a frissítési üzenetek túlzott mértékben vannak késleltetve, így tehát míg egy frissítési üzenet a sorban várakozik, sokkal több visszavonási üzenet küldődik el, mely a fent leírt probléma extrém esetét adja. Ami érdekes, hogy szabványos használat esetén ez a probléma *nem jelentkezik* – de ekkor is csak abban az esetben, ha a küldés nem kötegelve történik (ezt az eset a táblázatban zöld színezéssel jelöltük). Már kicsi, mindössze 2 visszavonási üzenetből álló kötegek használata esetén ismét jelentkezik a probléma, és a kötegméret növelésével egyre erősödik - míg bizonyos kötegméret felett már a túlzó használat esetén kapott óriási hibaarányt kapjuk.

Hibás RIB bejegyzések száma (visszavonások nincsenek késleltetve)						
Távolság a Topológia-változástól (ugrásszám)	Crude	Dedicated Blocking	Dedicated non-blocking kötegméret:			
			1	2	3	10
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	112	0	0	0	42	112
4	112	0	0	0	42	112
5	112	0	0	50	112	112
6	112	0	0	63	112	112
7	112	2	0	112	112	112
8	112	2	0	112	112	112

3.1. táblázat: Hibás RIB bejegyzések száma eltérő MRAI használatok esetén

Megjegyzendő, hogy a kötegek méretének változása nagy valószínűséggel a szimulátorban lényegesen nagyobb hatással van a konvergencia-folyamat kimenetelére, mint valós BGP esetén. Ennek a szimulátor által használt időkezelési módszer az oka –

mivel a routerek csak diszkrét időközönként kapnak jogot a buffereikben felgyülemlt üzenetek elküldésére, így a kötegméret n-szerezésével gyakorlatilag egyben a visszavonási üzenetek kiküldésének gyakoriságát is n-szeresére növeljük. Nyitott marad azonban a kérdés, hogy ezt a valós BGP implementációkban hogyan volna érdemes megvalósítani – valóban egyesével küldjük ki a visszavonásokat? Esetleg „várjunk be” n darab visszavonást, és egyszerre küldjük ki őket? Vagy netán várjunk adott ideig, és az addig összegyűlt visszavonásokat küldjük egyszerre? Mivel az RFC nem szabályozza ezt a kérdést, valószínűleg ennek megvalósítására a valóságban sokféle eltérő módszer létezik, melyeknek - a szimuláció eredményeit alapul véve - mind-mind megvan a maga előnye és hátránya.

Mivel azt tapasztaltuk, hogy a visszavonási üzenetek túlságosan gyors kiküldése az említett problémára vezet, értelmesnek tűnik a gondolat, hogy a jelentkezett hibát a visszavonási üzenetek késleltetése megoldhatja – ugyanis a hiba forrásának a visszavonási üzenetek túl korai kiküldését hisszük. Annak ellenőrzésére, hogy a probléma megszűnik-e a visszavonási üzenetek késleltetésével, újabb szimulációkat futtattam. Az meglepő eredmény a 3.2. táblázatban látható:

Hibás RIB bejegyzések száma (visszavonások késleltetve, kötegméret fixen 1)				
Távolság a Topológia-változástól (ugrásszám)	Crude	Dedicated Blocking	Dedicated non-blocking	
1	0	0	0	0
2	0	0	0	0
3	112	0	0	0
4	112	0	0	0
5	112	0	0	0
6	112	89	33	33
7	112	89	89	89
8	112	89	89	89

3.2. táblázat: Hibás RIB bejegyzések száma a visszavonások késleltetése mellett

Vegyük észre, hogy a helyzet *rosszabb*, mint volt – az előző esetben legjobb konstrukció (nem kötegelt küldés, szabványos MRAI használat – a táblázatban zöld színnel jelölve) is hibás eredményeket ad a visszavonások késleltetése esetén!

Jogos a felmerülő kérdés – hol is lehet akkor a hiba? A probléma gyökerének hitt, a visszavonások túl korai kiküldését ugyanis megszüntettük. Azonban ez a

megoldás a hiba kiküszöbölése helyett egy új hibát generál: Mivel most már egy visszavonási üzenet kiküldésekor *is újraindul* a MRAI időzítő, így amíg mi a visszavonási üzeneteket késleltetjük, mellékhatásként a frissítési üzenetek *még tovább* lesznek késleltetve. Így tehát a problémától nem szabadultunk meg, cserébe viszont lényegesen lassabb konvergenciával kell számolnunk (hiszen most már minden üzenet kiküldése késik).

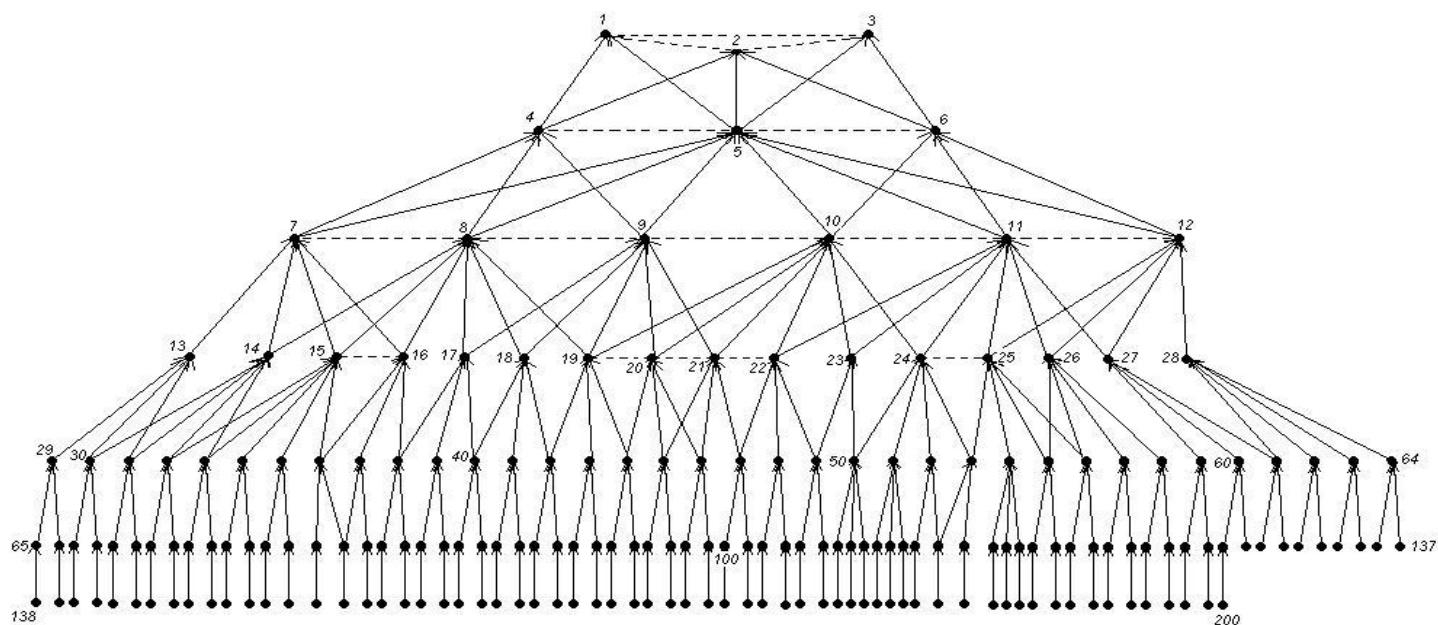
Az a gondolatunk támad, hogy ha külön MRAI időzítőket tárolnánk a frissítési és a visszavonási üzenetekre, úgy megkerülhetnénk azt a helyzetet, hogy a visszavonások és a frissítések „mellékhatásként” egymást is késleltessék. Ebben az esetben azonban kétszer annyi MRAI időzítőt kellene kezelnünk, mint eddig – és emlékezzünk, hogy a szabványos megoldás esetében már a kiindulási erőforrásigény is elfogadhatatlanul magas volt.

Túlzó esetben ugyan csak szomszédonként kettő időzítőt kellene fenntartanunk (egyet a frissítéseknek, egyet a visszavonásoknak), azonban akad egy probléma: ezt önkényesen nem tehetjük meg, ugyanis a szabványban ilyesmiről szó sem esik – az RFC 4271 ugyanis csak annyit ír, hogy az elavult RFC 1771 ajánlása helyett mostantól a MRAI időzítőket ne csak a frissítések, hanem a visszavonások késleltetésére is használjuk. Külön időzítőkről szó sincs. Érdekes tapasztalat, hogy a legkevesebb hibára vezető konstrukció pontosan az eredeti RFC 1771 által javasolt megoldás – szabványos MRAI használat a visszavonások késleltetése nélkül – az eredményen a szabvány tömeges be nem tartása, illetve az új RFC javaslatok csak rontanak.

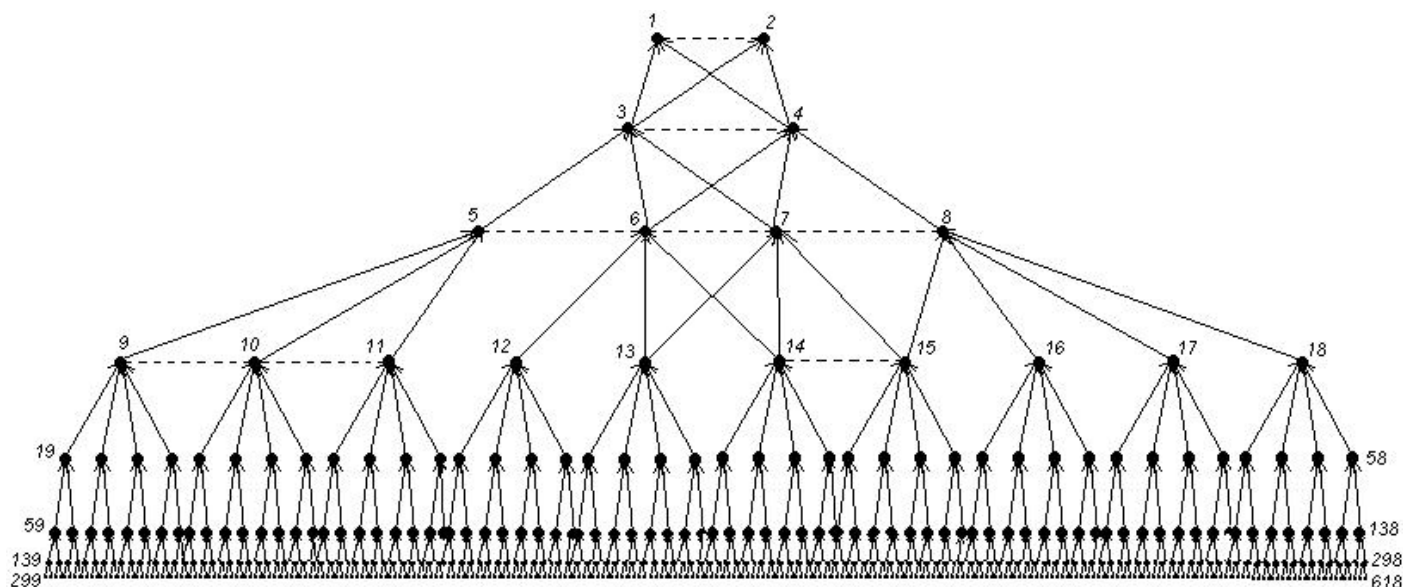
A fejezet elején örömteli hírekkel lehattunk gazdagabbak, melyek szerint az elmélet pesszimista jóslatai nem bizonyulnak igaznak – azonban a fejezet második felében ismertetett eredmények kisebb aggodalomra adnak okot. Amennyiben ugyanis a konvergencia végeredménye egy hibás állapot, úgy a topológia-változást követő tranziens állapot hiába lesz gyors és sávszélesség-kímélő – ha a kialakuló végállapot hibás forgalomtovábbítást eredményez, úgy stabilan fennálló problémával kell számolnunk.

3.2.4 Használt topológiák

Végezetül a két topológia rajza következik, melyekre a legtöbb szimulációt futtattam:



3.15. ábra: 200 routerből álló, szintezett topológia (topo4)



3.16. ábra: 618 routerből álló, szintezett topológia (tree4)

4. Összefoglalás, tanulságok, kitekintés

4.1 Fontosabb tapasztalatok

Összefoglalás gyanánt tegyünk említést néhány fontos dologról, melyeket a szimulációk során tapasztaltunk.

Megismertünk egy elméletet, melynek állításai közül a szimulált eredmények csak keveset támasztanak alá. Ez természetesen nem jelenti azt, hogy a vizsgált elmélet helytelen volna – ugyanis a cikk kizárólag a legrosszabb esetek tulajdonságait („*worst case scenario*”) írja le. Az, hogy bizonyos összefüggések nem teljesültek az általunk szimulált teszteseteken, mindössze annyit jelent, hogy ezek a legrosszabb esetre felállított azonosságok általános hálózati elrendezéseken nem minden esetben jelentkeznek – sőt, néhány esetben pontosan a jósolt tendenciák ellenkezőjét tapasztaltuk. Mindenképpen igaz azonban, hogy ilyen, viszonylag kisszámú szimuláció eredményeire nem lehet általános érvényű kinyilatkoztatásokat alapozni – a viselkedés pontosabb megértéséhez feltétlenül szükséges, hogy ennél lényegesen nagyobb számú szimulációt végezzünk. Minden esetre örömteli hírnek tekintjük, hogy az elmélet egyes baljós feltételezései által felírt összefüggések túlnyomó része a szimulációk során nem mutatkozott meg.

Rossz hír azonban, hogy a munka során bizonyos anomáliák váratlanul felütöttek a fejüket. Általános tanulságként elmondható, hogy a szabvány által nem teljesen tisztázott, nyitott kérdések nagy része különböző jellegű és súlyosságú hibák forrása lehet. Láttuk azt is, hogy a legtöbb ilyen „nyitott kérdés” tetszőleges értelmezése mellett sem feltétlenül garantált a konvergencia-folyamat hibátlan végállapota. Megismertük a MRAI időzítők eltérő használati módjainak előnyeit és hátrányait – nevezetesen, hogy a szabványszerű használat rövidebb konvergenciát és a kialakuló végállapot aránylag kevés hibáját biztosítja, míg az általánosan használt üzemeltetési mód, a túlzó mód lassabb konvergenciát és sok hibát eredményez, ám sávszélesség-igénye alacsonyabb, és a résztvevők erőforrásait is kíméli.

Természetesen az, hogy az általam készített szimulátor a látott eredményeket szolgáltatja, korántsem jelenti azt, hogy ezek a jelenségek a valóságban is ugyanígy jelen lennének – hiszen ahogy említettük is, a szimulátor a BGP számos részletének

implementációjától eltekint. Ha a valós BGP konvergenciája is az általunk tapasztalt óriási mennyiségű hibát építené be a routerek tábláiba, akkor valószínűleg az Internet nem is működne úgy, ahogy mi azt megszokhattuk – így azonban felmerül a kétség a kapott adatok hitelességéről. Teljes mértékben megbízható eredményeket természetesen csak valós teszhálózatokon történő mérésekkel lehetne kapni, ám a dolgozat első fejezeteiben említett okokból ennek a megvalósítása meglehetősen körülményes (gyakorlatilag kivitelezhetetlen). Annyi tanulságot azonban mindenképp leszűrhetünk, hogy minél több a szabvány által nem rögzített, az implementálóra bízott részlet, a folyamat kimenetele annál bizonytalanabb, megjósolhatatlanabb lesz.

4.2 A BGP konvergencia fejlesztésére tett javaslatok

A szimulációk során tapasztalt jelenségek biztosabb elkerülésére, valamint a konvergencia-tulajdonságok javítására az alábbi javaslatokat tesszük:

1 - A BGP protokoll időbélyegzéssel való kiegészítése: Mivel a tapasztaltak alapján a MRAI használatának előírását akárhogy „bonyolítjuk”, a hibamentesség nem lesz garantált - ugyanis az üzenetek helytelen sorrendben való megérkezését egyik módszer sem biztosítja, legfeljebb az előfordulási valószínűségét csökkenti. Időbélyegzés használatával azonban a sorrendezés helyessége biztosíthatóvá válna, így az ezzel kapcsolatos hibák kialakulása megelőzhető lenne. Ilyen módon nem is volna feltétlenül szükség a visszavonási üzenetek fizikai késleltetésére, elég lehetne az eredeti RFC 1771 által is javasolt eljárás – vagyis, hogy a MRAI időzítők kizárólag a frissítési üzenetek kiküldését szabályozzák. Amennyiben egy időben korábban keletkezett frissítési üzenet később érkezik meg a fogadóhoz, úgy erről a tényről az időbélyegek vizsgálatával a fogadó is tudomást szerez, így a későn érkező update üzenetet elvetheti.

2 – Tegyük lehetővé, hogy a MRAI használatának módja a konvergencia során adaptívan kerülhessen megállapításra: Láttuk, hogy a szabványos módszer alkalmazása rengeteg előnnyel jár, miközben egyetlen hátránya a nagyszámú MRAI időzítő menedzseléséből fakadó túlzott teher. Ez azonban nem minden esetben fog elfogadhatatlan mértékű leterheltséget jelenteni, csak abban az esetben, ha az adott

router valóban nagyszámú (néhány száz) BGP szomszédsággal rendelkezik, és az *aktuális* topológia-változás jellege olyan, hogy valóban nagyszámú célpont felé kell frissítési üzenetet küldeni (ekkor ugyanis a $[célpont \times szomszédság]$ számú MRAI időzítő ténylegesen nagy terhelést fog jelenteni). Azonban egy kisebb topológia-változás eredményezheti azt, hogy mindössze néhányszor tíz célpont felé kell csak frissítést küldeni - ennyi időzítőt pedig gond nélkül fenn tudunk tartani. Javaslatunk tehát, hogy amennyiben a konvergencia során az időzítők kezeléséhez szükséges erőforrásigény egy adott küszöbszintet túllép, úgy a router *ne vegyen fel több időzítőt*. Ezzel elérhető lenne az, hogy egy „jelentéktelenebb” topológia-változást követő konvergencia-folyamat lefolyásának ideje rövidüljön – míg komolyabb változások esetén a folyamat marad a régi, és ebben az esetben a hosszúra nyúló konvergenciát továbbra sem ússzuk meg. Minden esetre még egy ilyen nagyobb topológia-változás esetén sem *romlik* a konvergenciaidő, legfeljebb nem tapasztalunk javulást.

4.3 Jövőbeli munkára vonatkozó kilátások

Mint említettük, az elvégzett szimulációk száma túlságosan kicsi ahhoz, hogy a kapott adatokból általános következtetéseket vonhassunk le. A folyamatok megértéséhez mindenképpen további vizsgálatok szükségesek.

A munka során elkészült szimulátor azonban rugalmasságából adódóan a dolgozat konkrét célkitűzésein felül egyéb vizsgálatok elvégzésére is alkalmassá tehető. Napjainkban is folynak kutatások a „*Jövő BGP-je*”, illetve a „*Jövő Internetje*” témákban - ilyen jellegű vizsgálatok elvégzésére viszont a szimulátor könnyűszerrel rábírható. A kód kismértékű átírásával kedvünk szerint alakíthatjuk a döntési folyamat által vizsgált BGP paramétereket (pl. Local Preference), a kimenő szűrőket, valamint akár új vizsgálati szempontokat is definiálhatunk – ilyen lehet például, hogy a döntési folyamat az egyéni irányelveken felül bizonyos fizikai paramétereket is mérlegeljen (természetesen ezeket a paramétereket a helyi elveknél alacsonyabb „prioritással” kezelve).

Irodalomjegyzék

- [1] A CAIDA szervezet kutatásainak weboldala: www.caida.org/research (2011.nov)
- [2] A DIMES project hivatalos weboldala: www.netdimes.org (2011.nov)
- [3] A RouteViews project hivatalos weboldala: www.routeviews.org (2011.nov)
- [4] A RIPE Regionális Internet Nyilvántartó Hivatal (RIR) hivatalos weboldala: www.ripe.net (2011.nov)
- [5] Alex Fabrikant, Umar Syed, Jennifer Rexford, „There’s something about MRAI: Timing diversity can exponentially worsen BGP convergence”, *INFOCOM, 2011 Proceedings IEEE, 2011.április, pp. 2975 - 2983*
- [6] Lixin Gao hivatalos honlapja: <http://www-unix.ecs.umass.edu/~lgao> (2011.nov)
- [7] Jennifer Rexford hivatalos honlapja: <http://www.cs.princeton.edu/~jrex> (2011.nov)
- [8] RFC 4271, „A Border Gateway Protocol 4 (BGP-4)”, webes elérés: <http://www.ietf.org/rfc/rfc4271.txt> (2011.nov)
- [9] Revisions to the BGP 'Minimum Route Advertisement Interval', *IETF munkadokumentum*, webes elérés: <http://tools.ietf.org/html/draft-ietf-idr-mrai-dep-04> (2011.nov)
- [10] RFC 1771, „A Border Gateway Protocol 4 (BGP-4)”, webes elérés: <http://www.ietf.org/rfc/rfc1771.txt> (2011.nov)
- [11] Understanding Route Aggregation in BGP, *Cisco Technology Support dokumentum, Document ID: 5441*, webes elérés: http://www.cisco.com/en/US/tech/tk365/technologies_tech_note09186a0080094826.shtml (2011.nov)
- [12] A Quagga szimulátor hivatalos weboldala: <http://www.quagga.net/> (2011.nov)
- [13] Az SSFNet szimulátor hivatalos weboldala: <http://ssfnet.org/> (2011.nov)
- [14] Nick Feamster hivatalos honlapja: <http://www.cc.gatech.edu/~feamster/> (2011.nov)
- [15] A C-BGP szimulátor hivatalos weboldala: <http://c-bgp.sourceforge.net/> (2011.nov)
- [16] Wolfgang Mühlbauer, „Towards Reproducing Inter-Domain AS Paths”, *Diplomamunka, Technische Universität München, 2005. november*
- [17] A CAIDA szervezet által az Internetről készített AS-szintű gráf éllistája – „The CAIDA AS Relationships Dataset”, webes elérés: <http://www.caida.org/data/active/as-relationships/> (2011.nov)
- [18] Timothy G. Griffin, Gordon Wilfong, „A Safe Path Vector Protocol”, *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, 2000.március, pp. 490 - 499*

- [19] Timothy G. Griffin, F. Bruce Shepherd, Gordon Wilfong, „Policy Disputes in Path-Vector Protocols”, *Network Protocols*, 1999. (ICNP '99) *Proceedings. Seventh International Conference on*, 1999.november, pp. 21 - 30
- [20] Lixin Gao, Jennifer Rexford, „Stable Internet Routing Without Global Coordination”, *Networking, IEEE/ACM Transactions on*, 2001.december, vol. 9 issue 6, pp. 681 - 692
- [21] Sophie Y. Qiu, Patrick D. McDaniel, Fabian Monrose, „Toward Valley-Free Interdomain Routing”, *Communications*, 2007. ICC '07. *IEEE International Conference on*, 2007.június, pp. 2009 - 2016

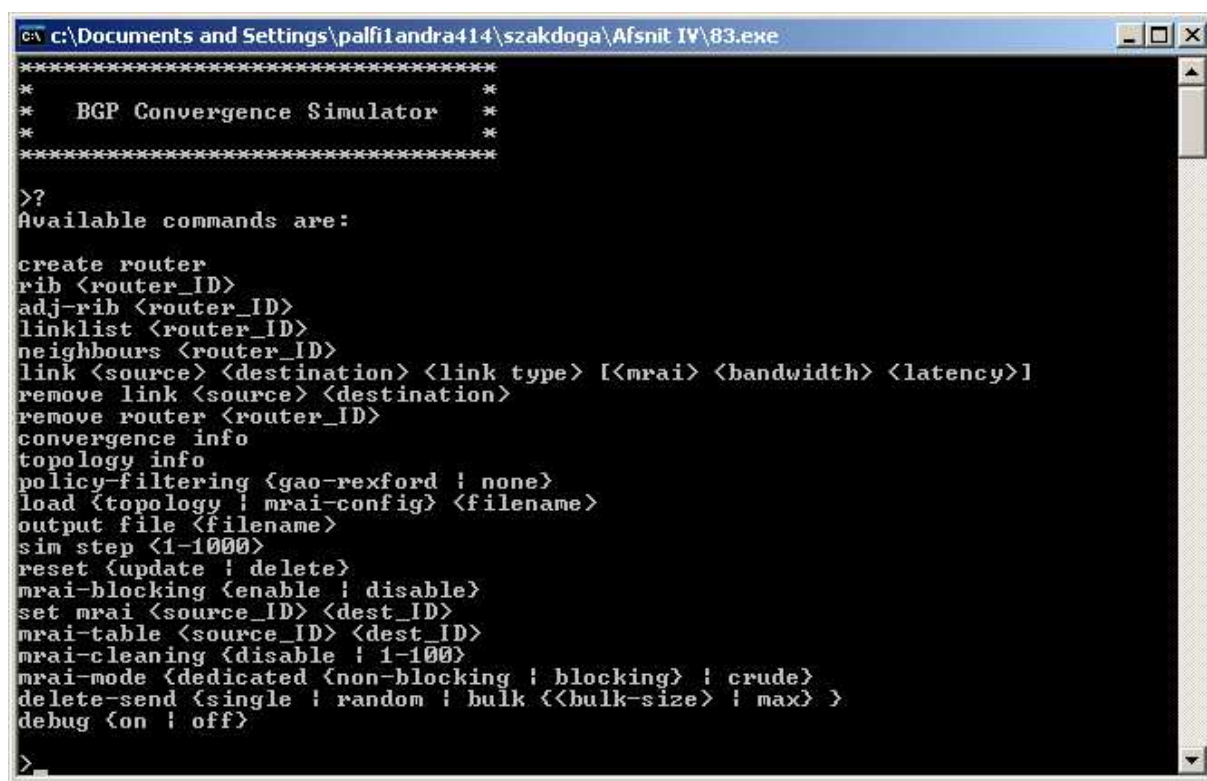
Rövidítésjegyzék

Adj-RIB-In:	Adjacent Routing Information Base Inbound
Adj-RIB-Out:	Adjacent Routing Information Base Outbound
AS:	Autonomous System
AT&T:	American Telephone and Telegraph Company
BGP:	Border Gateway Protocol
c2p:	Customer-to-Provider
CAIDA:	The Cooperative Association for Internet Data Analysis
eBGP:	External Border Gateway Protocol
EGP:	Exterior Gateway Protocol
EIGRP:	Enhanced Interior Gateway Routing Protocol
FSM:	Finite State Machine
IANA:	Internet Assigned Numbers Authority
iBGP:	Internal Border Gateway Protocol
IETF:	Internet Engineering Task Force
IGP:	Interior Gateway Protocol
IP:	Internet Protocol
ISP:	Internet Service Provider
Loc-RIB:	Local Routing Information Base
MED:	Multiple Exit Discriminator
MRAI:	Minimal Route Advertisement Interval
OSPF:	Open Shortest Path First
p2c:	Provider-to-Customer
p2p:	Peer-to-Peer
PIB:	Policy Information Base
QoS:	Quality of Service
RIB:	Routing Information Base
RIP:	Routing Information Protocol
RIR:	Regional Internet Registry
TCP:	Transmission Control Protocol
VoIP:	Voice over IP

Függelék: útmutató a szimulátor használatához

Jelen függelék az elkészült szimulátor használatához nyújt segítséget. A program parancssorból vezérelhető, és a végzett vizsgálatokhoz megadott formátumú topológia- és MRAI konfigurációs fájlokat használ – ezeket a parancsokat, illetve formátumokat ismertetjük a fejezet során.

A program indulásakor a megnyíló ablakban megjelenik a fejléc, valamint a prompt, ahova a program a parancsokat várja. A „?” karakter begépelésével, majd az *Enter* billentyű megnyomásával egy tájékoztatást kaphatunk az értelmezett parancsokról:

The image shows a Windows command prompt window titled "c:\Documents and Settings\palfi1andra414\szakdoga\Afsnit IV\83.exe". The window contains the following text:

```
*****
*                               *
*   BGP Convergence Simulator   *
*                               *
*****

>?
Available commands are:

create router
rib <router_ID>
adj-rib <router_ID>
linklist <router_ID>
neighbours <router_ID>
link <source> <destination> <link type> [<mrai> <bandwidth> <latency>]
remove link <source> <destination>
remove router <router_ID>
convergence info
topology info
policy-filtering <gao-rexford ! none>
load <topology ! mrai-config> <filename>
output file <filename>
sim step <1-1000>
reset <update ! delete>
mrai-blocking <enable ! disable>
set mrai <source_ID> <dest_ID>
mrai-table <source_ID> <dest_ID>
mrai-cleaning <disable ! 1-100>
mrai-mode <dedicated <non-blocking ! blocking> ! crude>
delete-send <single ! random ! bulk <<bulk-size> ! max> >
debug <on ! off>

>
```

F.1. ábra: A szimulátor által értelmezett parancsok

Parancsok topológia kézzel való megadásához, módosításához:

create router: Egy új routert hoz létre a topológiában. A router által használt BGP ID-kat (AS számokat) a program 1-től indulva növekvő sorrendben rendeli hozzá a létrehozott routerekhez (tehát ez a paraméter nem a felhasználó által kerül beállításra).

remove router <router_ID>: A paraméterben megadott BGP ID-val rendelkező routert (amennyiben van ilyen) a topológiából eltávolítja (minden BGP szomszédságát megszakítja).

link <source> <dest> <linktype> [<mrai><bandwidth><latency>]: A source és destination paraméterekben megadott routerek között egy linket hoz létre. A link egyes paraméterei külön specifikálhatók, ezek közül a link típusa (Gao-Rexford típus) kötelező, a linken használt MRAI időzítő értéke, illetve a program által jelenleg nem használt sávszélesség és késés pedig opcionálisak. A link típusa -1 (provider-customer), 0 (peer-peer) vagy 1 (customer-provider) lehet.

Példa: a **link 1 2 -1 25 1000 50** parancs egy provider-customer élet hoz létre az 1-es és 2-es router közt (az 1-es lesz a provider) 25 másodperces MRAI időzítővel, 1000Mbit/sec sávszélességgel és 50ms késéssel.

remove link <source_ID> <dest_ID>: A paraméterekben megadott két router közötti BGP szomszédságot (amennyiben létezik) megszakítja.

Parancsok fájlból olvasáshoz, fájlba kiíratáshoz:

load {topology |mrai-config} <filename> : Egy előre elkészített topológia-fájlt, vagy egy MRAI-konfigurációs fájlt tölt be a topológiába. Egy topológia-fájl egy szöveges file, melynek első sorában a topológiában található routerek száma (egy egész szám) áll. A második sortól kezdve hármassával állnak a számok, melyek a topológiában kiépített linkeket jelképezik <forrás router ID> <cél router ID> <link típusa> formátumban. Az előbbi példában létrehozott linket pl. az „1 2 -1” számsorral lehet leírni. Az egyes linkeket jelentő számhármassok soronként helyezkednek el, és minden sort egy sortörés karakter választ el egymástól. *Nagyon fontos*, hogy a fájlban található utolsó link után is legyen egy sortörés – így tehát a fájl ezzel a sortöréssel fog végződni. Szintén nagyon fontos, hogy minden linknek szerepelnie kell a file-ban *mindkét irányban* a helyes működéshez. A fenti példa esetében tehát a fájlban szükségszerűen szerepelnie kell valahol a „2 1 1” sornak, ellenkező esetben a program helytelen

működést fog szimulálni (az egyik fél oldaláról nem tekintődik BGP szomszédnak a másik fél, így neki nem fog üzeneteket küldeni).

A MRAI konfigurációs fájl formátuma szintén ilyen számhármassokból áll, azzal a különbséggel, hogy a harmadik szám nem a link típusát, hanem a linken alkalmazott MRAI értékét fogja jelenteni. Különbség még, hogy ennek a fájlnek az elején nem kell feltüntetni a topológiában található routerek számát – vagyis a fájl tartalma egyből az első linkkel kezdődik. *Fontos*, hogy a MRAI konfigurációs fájl csak olyan linkekre hivatkozzon, amelyek előzőleg a topológiában már létre lettek hozva, azonban nem szükséges, hogy a topológia összes linkje megjelenjen a fájlban, illetve hogy a linkek feltétlenül megjelenjenek mindkét irányban – ám a fel nem tüntetett linkek esetében ilyenkor a linken használt MRAI érték az alapértelmezett 10 másodperces érték marad.

Egy egyszerű topológiafájltra, és a rajta alkalmazható MRAI konfigurációs fájlra mutat egy példát az *F.2. ábra*:



F.2. ábra: Példák egyszerű topológia- és MRAI konfigurációs fájlokra

output file <filename>: A parancs kiadását követően a program a kimeneti adatokat a képernyőre íráson kívül a paraméterben megadott szöveges fájlba is elmenti. Ettől fogva a lekérdező parancsok (pl RIB, Adj-RIB, linklista, konvergencia-adatok kiírása) kimenete is ebbe a fájlba kerül mentésre.

Parancsok szimuláció futtatásához:

build: Miután egy topológia a programba betöltődött (kézzel létrehoztuk, vagy a program fájlból beolvasta), de még mielőtt a szimulációt le tudnánk futtatni, először a

szimulált hálózat kezdeti alapállapotát ki kell építeni – erre szolgál a *build* parancs. Ennek során a routerek nagyszámú update üzenet segítségével megegyeznek az útvonalválasztáson, és a routerek táblái feltöltődnek a kezdeti bejegyzésekkel.

run: Miután az alapállapotot a *build* paranccsal kiépítettük, és az általunk vizsgálni kívánt topológia-változtatást az alapállapot kialakulását követően elvégeztük (pl. egy *remove router* vagy *remove link* paranccsal), valamint esetleg a kívánt MRAI konfigurációt a topológián alkalmaztuk (kézzel, vagy a *load mrai-config* paranccsal) úgy a topológiaváltozást követő eseménysorozat elindítása a *run* paranccsal történik. Ennek során a routerek update (frissítés) és delete (visszavonás) üzenetekkel megegyeznek a módosult útvonalválasztáson.

Parancsok adatok, információk kinyeréséhez:

Megjegyzés: Ebben a pontban ismertetett parancsok kimenetei az előzetesen kiadott *output file* <filenév> parancs hatására kimeneti fájlba is mentésre kerülnek.

rib <router_ID>: A paraméterben megadott BGP ID-val rendelkező router helyi útvonaltáblájának (Loc-RIB) tartalmát írja ki a képernyőre és a kimeneti fájlba (amennyiben az meg lett adva az *output file* paranccsal).

adj-rib <router_ID>: A paraméterben megadott BGP ID-val rendelkező router összes bemenő útvonaltáblájának (Adj-RIB-In) uniójának tartalmát írja ki a képernyőre és a kimeneti fájlba (amennyiben az meg lett adva az *output file* paranccsal).

Megjegyzés: A szimulátorban a bemenő és kimenő útvonaltáblák nem szeparált tárolóegységek formájában kerültek implementálásra. Ehelyett minden router egy helyi táblát (RIB) tárol, illetve egy olyan táblát, melyekben az összes, a szomszédjainktól beérkezett útvonalat tároljuk (Adj-RIB) – ez a parancs tehát ez utóbbi tartalmát fogja kilistázni²⁵.

²⁵ A programban az kimenő útvonaltáblák (Adj-RIB-Out) nincsenek is megvalósítva, az útvonalak egyből a kimenő bufferekbe kerülnek a megfelelő üzenetekbe csomagolva.

linklist <router_ID>: A paraméterben megadott BGP ID-val rendelkező router linkjeit (BGP szomszédságait) listázza ki – a linkeken található szomszédok ID-jával, Gao-Rexford típusukkal, a használt MRAI értékkel, és a fizikai jellemzőkkel.

topology info: A betöltött topológiáról, illetve a szimuláció egyes globális beállításairól kaphatunk hasznos információkat, mint pl. a topológiában található linkek és routerek száma, a MRAI diszparitás és diverzitás, az alkalmazott MRAI használati mód, várakozó üzenetek száma, kimenő szűrők beállításai.

convergence info: A lefuttatott szimuláció konvergencia-tulajdonságait kaphatunk információt ezzel a paranccsal – így pl. a szimulált konvergenciaidő, illetve a szimuláció indulása óta eltelt idő (valós és modellezett idő), elküldött összes update és delete üzenetek száma, a szimuláció lépésköze.

debug on: A szimuláció során számos eseményről jelentést ad, így pl. üzenetek elküldéséről, MRAI időzítők lejárásáról, stb.

Megjegyzés: a hibakeresés bekapcsolásával egy nagyobb topológia esetén óriási mennyiségű adat keletkezhet (erre a program figyelmeztet is), mely sokszor már-már szüségtelenül sok, kezelhetetlen mennyiségű információval szolgál. A funkció elsősorban a fejlesztés közbeni hibakeresésre szolgált, ám a végső változatban is megmaradt.

Parancsok globális paraméterek átállításához:

policy [gao-rexfor | none]: A szimuláció során használt kimenő szűrőket lehet vele ki- vagy bekapcsolni. Az egyetlen, a program által alapértelmezetten használt szűrési stratégia a Gao-Rexford tétel 2. feltételének legegyszerűbben megfelelő összefüggés (4.1-es képlet)²⁶.

Megjegyzés: A kimenő szűrők kikapcsolásával a Gao-Rexford tétel feltételei nem teljesülnek, így a szimuláció konvergenciája nem garantált!

²⁶ A programkód kisebb módosításával egyéb kimenő szűrők is könnyen implementálhatók

mrai-blocking [enable | disable]: A MRAI időzítők használatát lehet vele ki- vagy bekapcsolni (pl. abban az esetben, ha bennünket csak a konvergencia statikus tulajdonságai érdekelnek, úgy lehetőség van az időzítőket kikapcsolni).

mrai-mode [crude | dedicated [non-blocking | blocking]]: Az alkalmazott MRAI módot választhatjuk ki ezzel a paranccsal.

delete-send {single | bulk [<bulk-size>] }: A visszavonási üzenetek küldésének módját választhatjuk ki vele, kötegelt üzenetküldés esetén a köteg méretét is megadhatjuk.