BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM
SZÁMÍTÁSTUDOMÁNYI ÉS INFORMÁCIÓELMÉLETI TANSZÉK

MEGBÍZHATÓSÁGOT NÖVELŐ ÉS SZOLGÁLTATÁSMINŐSÉG-BIZTOSÍTÁST ELŐSEGÍTŐ MÓDSZEREK ETHERNET ÉS UTRAN HÁLÓZATOKBAN

Farkas János

Tézisfüzet

Tudományos vezetők:

Dr. Györfi László
Számítástudományi és Információelméleti Tanszék
Budapesti Műszaki és Gazdaságtudományi Egyetem
Dr. Antal Csaba
Ericsson Magyarország

Budapest
2011

1. Bevezetés

Az Ethernetet az egyszerűsége és az alacsony költségen nyújtott nagy sávszélesség tette vonzóvá különféle hálózati környezetben. Az 1970-es évekbeli feltalálása óta az Ethernet bizonyította, hogy képes alkalmazkodni a változó követelményekhez. Eredetileg helyi hálózatokon (Local Area Network – LAN) belüli összeköttetés nyújtására tervezték, és végül a vállalati hálózatok de facto szabványává vált. Manapság az Ethernet a vállalati hálózati szegmensből a szolgáltatói hálózatok irányába fejlődik. Azonban mint LAN technológia nem nyújtotta azt a hibatűrést, amit a szolgáltatói hálózatok a minőségbiztosítás érdekében igényelnek.

Az egyik legfontosabb szolgáltatói (carrier-grade) követelmény a hibatűrés. A szolgáltatók hozzászoktak a SONET/SDH hálózatok hibakezelési teljesítményéhez és robusztusságához, ezért hasonló teljesítményt várnak el a csomagkapcsolt hálózatoktól is. Az Ethernet hálózatokban először a Feszítőfa Protokoll (Spanning Tree Protocol – STP) nyújtott hibakezelést, ez azonban nem teljesítette a szolgáltatói követelményeket, mivel a konvergenciaideje tíz másodperces nagyságrendbe esett. A Gyors Feszítőfa Protokoll (Rapid Spanning Tree Protocol – RSTP) [1] és a Többszörös Feszítőfa Protokoll (Multiple Spanning Tree Protocol – MSTP) [2] lényegesen gyorsabb hibakezelési időt biztosít a protokoll tervezéséből adódóan, azonban ezek sem tudják garantálni az 50 milliszekundumos (ms) hibakezelést, amit a szolgáltatók elvárnak. Az RSTP és az MSTP távolság vektor alapú, ezért a végtelenbe számolás problémája [3] előfordulhat a hibakezelés során. További hibakezelési eljárásokat is specifikáltak Ethernet hálózatokra [4, 5, 6], amelyeket gyűrű topológiákra terveztek, ezért nem alkalmazhatók tetszőleges topológiájú hálózatokban. Egy további megközelítés, hogy a hibakezelést egy központi entitás végzi, ahogyan azt Sharama [7] javasolta. Ez azonban lelassíthatja a hibakezelést és egyszeres hibaponthoz vezethet. Ezért új algoritmusokra és protokollokra van szükség ahhoz, hogy az Ethernet hálózatok a szolgáltatói követelményeket akár nagyvárosi méretben teljesíteni tudják.

Az Ethernet hálózatok területén egy új fejlemény a kapcsolatállapot (link state) alapú vezérlő protokollok alkalmazása a korábbi távolságvektor alapú protokollok helyett. Ezáltal növelhető a hálózati kihasználtság és az átviteli kapacitás a feszítőfa protokollokhoz képest. Az ISO/OSI Intermediate System to Intermediate System (IS-IS) [8] útvonalválasztó protokoll egy alkalmas alap az Ethernet hálózatok új vezérlő protokolljának megalkotásához, mivel az IS-IS támogatja a MAC címek kezelését, illetve Típus, Hossz és Érték (Type, Length, Value – TLV) struktúrákon alapul. Két új szabvány is ezzel a problémakörrel foglalkozik: az IEEE 802.1aq Shortest Path Bridging (SPB) [9] és az IETF Transparent Interconnection of Lots of Links (TRILL) [10, 11]; mindkettő az IS-IS-en alapszik. Az SPB-t az IEEE 802.1 specifikálja, ezért az SPB megőrzi a 802.1 architektúrát, és kompatibilis a többi 802.1 szabvánnyal. Ezért az SPB szabványos 802 keretformátumot használ, rendelkezik a

802.1 által specifikált Üzemeltetési, Nyilvántartási és Karbantartási (Operations, Administration and Maintenance – OAM) eszközökkel, skálázhatóságot elősegítő, illetve adatközpontokat (data centre) támogató megoldásokkal. Ezzel szemben a TRILL új keretformátumot határoz meg, vagyis új adatsíkot vezet be, amely új hardvert tesz szükségessé. Továbbá a TRILL csak felhasználói Ethernet szolgáltatásokra alkalmazható [10], mivel az IEEE 802.1Q-2005 szabványt alapul véve specifikálták, ezért nem kompatibilis a [2] kiterjesztéseivel, vagyis az IEEE 802.1 által 2005 után készített szabványokkal. Ezért a TRILL nem rendelkezik OAM-el, skálázódási problémai vannak, és nem kompatibilis az új adatközpont szabványokkal, ahogyan azt Eastlake, a TRILL munkacsoport vezetője is összefoglalta [12]. A fenti különbségek miatt az SPB többféle hálózati környezetben alkalmazható, mint a TRILL. Azonban a kapcsolatállapot alapú vezérlés Ethernet hálózatokban történő bevezetése komoly problémákat vet fel, amelyeket meg kell oldani, mint például a hurokmegelőzés. Ezért az SPB-nek IS-IS kiterjesztéseket kell implementálnia ahhoz, hogy alkalmas legyen Ethernet hálózatok vezérlésére.

A kapcsolat nélküli (connectionless) hálózatok, mint például az Internet Protokoll (IP) és az Ethernet kapcsolat orientált (connection oriented) szolgáltatásokhoz való transzport technológiaként történő egyre elterjedtebb használata szükségessé teszi a szolgáltatásminőség-biztosítási (Quality of Service – QoS) megoldások alkalmazását. Az IP-t gyakran használják Rádiós Hozzáférési Hálózatok (Radio Access Network – RAN) transzportjaként. A valós idejű forgalmak QoS követelményei (késleltetés, csomagvesztés, dzsitter) szigorúak az Univerzális Mobil Távközlőrendszer (Universal Mobile Telecommunication System – UMTS) Földi Rádiós Hozzáférési Hálózatában (UMTS Terrestrial Radio Access Network – UTRAN). Például az UTRAN-ban a teljes késleltetés maximális értéke 7 ms lehet. Az UTRAN-nak kapcsolatengedélyezést (Connection Admission Control – CAC) kell alkalmaznia a QoS követelmények teljesítéséhez. A CAC-nak figyelembe kell vennie az UTRAN jellemzőit ahhoz, hogy el tudja dönteni, vajon egy új kapcsolat beengedhető-e a már bent lévő kapcsolatok minőségének romlása nélkül. Továbbá a CAC-nak alkalmazkodnia kell a transzport technológiához a hálózati erőforrások kihasználása érdekében.

2. Kutatási célok

A disszertáció célja olyan algoritmusok és protokollok definiálása, amelyek használhatóvá teszik az Ethernetet mint transzport technológiát nagyvárosi méretű hálózatokban is, azon előnyeinek megtartása mellett, amelyek vonzóvá tették vállalati és egyetemi hálózatokban. További cél olyan algoritmusok definiálása, amelyek elősegítik a szolgáltatásminőség-biztosítást IP transzportot alkalmazó UTRAN-ban. A következő lista a tézisek célkitűzését összegzi:

- Definiálni és kiértékelni olyan módszereket és algoritmusokat, amelyek biztosítják, hogy a magjában IEEE 802.1Q-2005 szerint szabványos hidakból álló hálózat teljesíteni tudja az 50 ms szolgáltatói hibakezelési követelményt. Tehát az új eljárások illetve algoritmusok csak a hálózat szélső csomópontjaiban vagy egy menedzsment rendszerben implementálhatók. A hálózat fizikai topológiája alapján a csomagtovábbítási utakat úgy kell meghatározni, hogy azok legalább egyszeres kapcsolati (link) vagy csomóponti (node) hibát tudjanak tolerálni. Továbbá a hibaesemények észrevételére és kezelésére is kell adni eljárást. (1. tézis)

- Javasolni és kiértékelni olyan algoritmusokat, amelyek meglévő kapcsolatállapot alapú protokollokat terjesztenek ki, ezáltal alkalmazhatóvá téve őket az Ethernet hálózatok vezérlésére. A legfontosabb cél a hurokmegelőzés biztosítása kapcsolatállapot alapú protokoll által vezérelt Ethernet hálózatban. (2. tézis)

- Definiálni olyan kapcsolatengedélyező algoritmust IP alapú UTRAN számára, amely ki tudja használni a hálózati csomópontok által implementált Wighted Fair Queuing (WFQ) ütemezés előnyeit. Továbbá, olyan CAC-ot definiálni, amely a forgalmi aggregátumoknál finomabb granularitást tud figyelembe venni, ezáltal képes szolgáltatásminőség garanciát nyújtani a beszédfolyamok számára. (3. tézis)

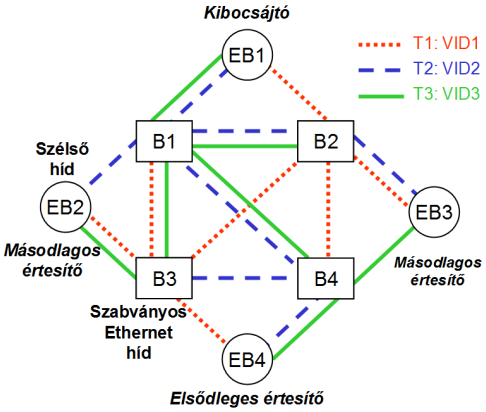
A kutatási célkitűzések mellett további célom volt, hogy az SPB-hez kapcsolódó kutatási eredményeimmel hozzájáruljak az IEEE 802.1-ben folyó szabványosításhoz.

3. Új eredmények

3.1. Hibatűrő Ethernet

A szolgáltatói hálózatok gyors hibakezelést követelnek, a hibakezelési időre vonatkozó előírás 50 ms. A Szolgáltatói Ethernet (Carrier Ethernet) elterjedése igényt támaszt az IEEE 802.1Q-2005 szerinti hidak fejlesztésére, mivel azok nem tudják garantálni az elvárt hibakezelési időt. Javasoltam és kiértékeltem továbbfejlesztési technikákat az 50 ms-os hibakezelés teljesítésére hidakból álló hálózatban.

1. Tézis *Definiáltam egy hibatűrő Ethernet architektúrát olyan hálózatok számára, amelyek magja az IEEE 802.1Q-2005 szabvány szerinti hidakból áll, és megmutattam, hogy a javasolt architektúra teljesíti a szolgáltatók által támasztott 50 milliszekundumos hibakezelési követelményt.*



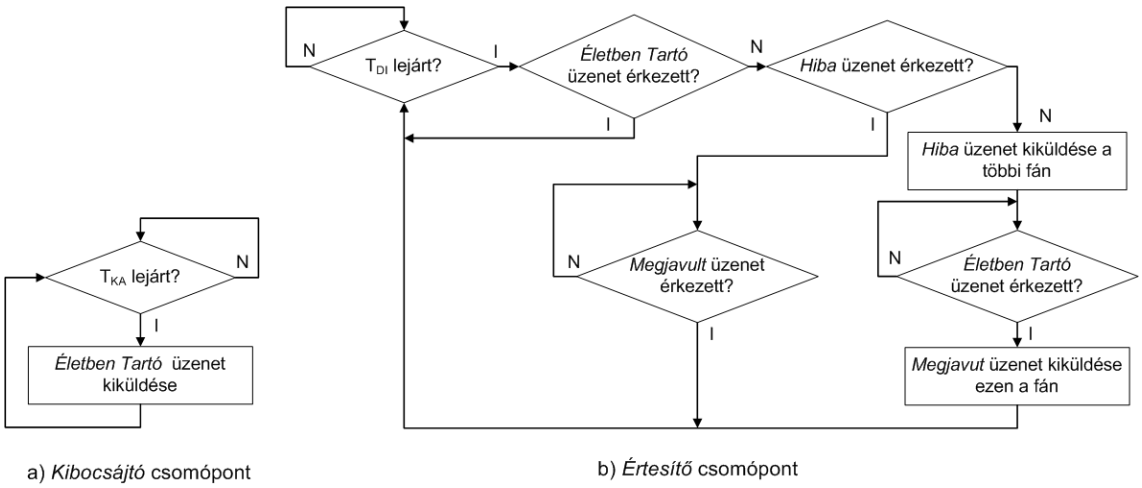
1. ábra. Hibatűrő Ethernet hálózat

Javasoltam az 1. ábrán illusztrált hibatűrő (resilient) Ethernet architektúrát, ahol a hálózat magja menedzselhető IEEE 802.1Q-2005 szerint szabványos Ethernet hidakból áll (B1-B4), amelyek támogatják a Virtuális LAN-okat (Virtual LAN – VLAN). Több előre meghatározott feszítőfa van statikusan beállítva a hálózatban, amelyek vagy elsődleges vagy másodlagos útként biztosítják az összeköttetést. Mindegyik fát (T1-T3) egy VLAN azonosító (VLAN Identifier – VID) azonosít. A fák hibatűrők, vagyis egy hálózati komponens elromlása ellenére biztosítják az összeköttetést. Hiba esetén a szélső csomópontok abbahagyják a keretek küldését az érintett fákon, és átirányítják a forgalmat a sértetlen fákra. Tehát új funkcionalitás csak a szélső hidakba (edge bridge) (EB1-EB4) kerül implementálásra.

A szélső csomópontoknak kell implementálniuk a hibakezelési mechanizmust, amely magában foglalja a fák elérhetőségének monitorozását és a forgalom átirányítását hiba esetén. A fákat úgy kell megtervezni, hogy legalább egy fa élje túl azon hibaeseményeket, amelyek ellen védeni akarjuk a hálózatot. Szükség van egy algoritmusra a hibatűrő fák meghatározásához, amelyhez a hálózat fizikai topológiájának pontos ismerete elengedhetetlen.

1.1. Tézis *Definiáltam egy elosztott hibakezelő protokollt, amelyet a hibatűrő Ethernet hálózat szélén lévő csomópontok futtatnak, és mérésekkel igazoltam, hogy a javasolt protokoll megfelel a szolgáltatók által támasztott 50 milliszekundumos hibakezelési követelménynek. [J3, C7, P16]*

A hálózat szélén lévő csomópontok implementálják az általam javasolt hibakezelő protokollt: Failure Handling Protocol (FHP), ahogy azt az 1. ábra illusztrálja. Az FHP néhány adatszórás (broadcast) üzenetet alkalmaz a hibák detektálására és a gyors reagálásra. A három adatszórás üzenet, illetve a szélső csomópontok szerepe az üzenetek kezelésében a következő:



2. ábra. Az FHP működése

- *Életben Tartó (Keep Alive – KA)* adatszórás üzeneteket küld egy vagy több úgynevezett *kibocsájtó (emitter)* csomópont az előre meghatározott T_{KA} periódus szerint minden VLAN-fán. Ha a *KA* üzeneteket az összes többi szélső csomópont megkapja, akkor a VLAN-fa üzemképes.
- *Hiba (Failure)* adatszórás üzenetet akkor küld egy *értesítő (notifier)* szereppel rendelkező szélső csomópont, amikor a *KA* üzenet nem érkezik meg egy VLAN-fán az előre meghatározott T_{DI} *detekciós intervallum*on belül. Ezáltal informálja az *értesítő* a többi szélső csomópontot az adott VLAN-fa meghibásodásáról.
- *Megjavult (Repaired)* adatszórás üzenetet küld a hibát detektáló *értesítő*, ha a *KA* üzenetek újra megérkeznek a korábban hibás VLAN-fán. Így az *értesítő* tájékoztatja a többi szélső csomópontot a sérült VLAN-fa kijavításáról.

Az 1. ábra mutat egy példát a szélső csomópontok szereposztására. Az adatszórási viharok (broadcast storm) elkerülése végett megkülönböztetünk *elsődleges (primary)* és *másodlagos (secondary) értesítőket*. A T_{DI} az *elsődleges értesítőkben* kisebb, mint a *másodlagos értesítőkben*, amely az egyetlen különbség a kétféle *értesítő* között. A protokoll működését a 2. ábrán látható folyamatábra mutatja be.

A hibakezelési idő fontos teljesítményjellemzője a hibatűrő megoldásoknak. A hibakezelési idő felső korlátja az általam javasolt architektúrában:

$$T_F \leq T_{KA} + T_{DI} + T_{tr} + T_{pr}, \tag{1}$$

ahol T_{tr} a legnagyobb átviteli késleltetés és T_{pr} a legnagyobb csomagfeldolgozási idő a hálózat egyik szélétől a másikig. Feltéve, hogy $RTT \sim 2(T_{tr} + T_{pr})$, és $T_{DI} = RTT$,

a hibakezelési idő: $T_F \leq T_{KA} + 1,5 \cdot RTT$, ahol az RTT a körülfordulási idő (Round Trip Time). Tehát a hibakezelési idő függ a hálózat méretétől. A hálózatra jellemző késleltetéseken kívül a hibakezelési idő csak T_{KA}-tól függ, melynek legkisebb gyakorlatban használt értéke 3 ms. Ezért a hibakezelési idő T_{KA} paraméter segítségével szabályozható. A protokoll konfigurálásával kapcsolatos további részletek a [D]-ben találhatók.

A protokoll helyes működésének ellenőrzése prototípus implementáció segítségével történt. A mérések megerősítették, hogy a hibakezelési idő 50 ms alatt tartható.

Ahhoz, hogy használni tudjuk az FHP-t, szükségünk van a megfelelő összeköttetési struktúrára, amelyet fa topológiák tudnak biztosítani Ethernet hálózatok esetében. Azonban, még a szükséges fák számának meghatározása is NP-teljes probléma, ahogy azt Čičić [13, 14] bizonyította.

1.2. Tézis *Megmutattam, hogy legalább* $k = \left\lceil \frac{l}{l-n+1} \right\rceil$ *feszítőfa szükséges ahhoz, hogy egy kapcsolat hibája ellen védelmet nyújtsunk egy olyan topológián, ami* n *csomópontból és* l *kapcsolatból áll. [C8]*

Észrevétel: Az alsó korlát közelebb van a feszítőfákat meghatározó algoritmusok eredményeihez, mint a felső korlát.

A hibatűrő architektúra a forgalom egyik fáról egy másikra történő átirányításán alapul, ha egy fa megsérül. Ahhoz, hogy kapcsolati hiba esetén végre tudjuk hajtani ezt a forgalom átirányítást, legalább egy fának túl kell élnie a hibát, hogy biztosítsa az összeköttetéseket. Tehát minden kapcsolathoz léteznie kell egy feszítőfának, amely nem tartalmazza az adott kapcsolatot. Ezen feltétel teljesítéséhez szükséges feszítőfák számának meghatározása NP-teljes probléma.

Jelölje n a csomópontok számát, l a kapcsolatok számát a $G = (N, L)$ topológiában, ahol $|N| = n$ és $|L| = l$. Az egyszeres kapcsolati hibák elleni védelemhez szükséges feszítőfák számának alsó korlátja:

$$k = \left\lceil \frac{l}{l-n+1} \right\rceil . \tag{2}$$

Čičić [15] később adott egy felső korlátot az egyszeres kapcsolati hibák kezeléséhez szükséges feszítőfák számára, továbbá egy heurisztikus algoritmussal nyert eredményeket is közölt. A 1. táblázat ad egy összehasonlítást a 2. egyenlet által adott alsó korlátra és Čičić eredményeire 16 csomópontból álló véletlen topológiákon. A táblázat az 1.3-1. algoritmus eredményeit is mutatja, amelyet a következő tézis részletez. Az algoritmusokhoz tartozó eredmények nem egész számok a táblázatban, mert minden érték több futtatás eredményeinek átlaga. A Čičić [15] által adott felső korlát a legnagyobb minimális kör (Largest Minimal Cycle – LMC) a topológiában; a táblázatban szereplő érték a különféle topológiákban lévő LMC átlaga.

1. táblázat. Kapcsolati hiba kezeléséhez szükséges feszítőfák száma 16 csomópontos hálózatokban

Átlagos fokszám	4	4,4	4,8
Čičić [15] felső korlátja	4,6	4,5	4,2
Čičić [15] heurisztikájának átlaga	2,6	2,5	2,2
Az 1.3-1 algoritmus eredményeinek átlaga	2,4	2,18	2,16
A 2. egyenlet által adott alsó korlát:	2	2	2

Ahogyan a táblázat mutatja, mindkét algoritmus eredménye közelebb áll az alsó korláthoz, mint a felső korláthoz. A Čičić [15] által közölt további eredmények még nagyobb eltérést mutatnak a felső korlát és a heurisztikus eredmények között. Tehát az alsó korlát pontosabb képet ad a topológia jellemzőire, mint a felső korlát.

Azon túl, hogy rendelkezünk egy becsléssel a szükséges feszítőfák számáról, magukat a feszítőfákat is meg kell határoznunk.

1.3. Tézis *Definiáltam egy algoritmust, ami egyszeres kapcsolati és csomóponti hibák ellen védelmet nyújtó feszítőfákat határoz meg a bemeneti topológiához. Az algoritmus heurisztikán alapul. Véletlen topológiákon futtatott kiterjedt szimulációk segítségével megmutattam, hogy az algoritmusom által az összeköttetések védelmére adott feszítőfák száma és az alsó korlát közötti eltérés 1, az 50 csomópontos hálózat méretig vizsgált topológiák legnagyobb részében. Továbbá definiáltam egy pontos fizikai topológia felderítő algoritmust heterogén Ethernet hálózatokhoz, amely a bemenetet szolgáltatja a feszítőfák meghatározásához, valamint mérésekkel vizsgáltam az algoritmus működését. [C3, C8, P11, P15]*

A kerettovábbításhoz használt VLAN-fáknak hibatűrőnek kell lenniük ahhoz, hogy kezelni tudjuk a hibaeseményeket. A cél az, hogy legyen legalább egy feszítőfa, amely teljes marad egy hálózati eszköz, vagyis egy kapcsolat vagy egy csomópont meghibásodása esetén. Ezért a kétféle hibát figyelembe véve a fákkal szemben támasztott követelmények a következők:

R1 *Kapcsolati hiba* – Minden kapcsolathoz kell lennie olyan feszítőfának, amely nem tartalmazza az adott kapcsolatot.

R2 *Csomópont hibája* – Minden csomóponthoz kell lennie egy olyan feszítőfának, amelyben az adott csomópont levél, vagyis egy a fokszáma.

Ha ezek a kritériumok teljesülnek, akkor minden hibához létezik egy fa, amelyet a hiba nem érint, ezáltal képes biztosítani az összeköttetést a hálózati csomópontok között.

Az általam javasolt algoritmus olyan VLAN-fákat határoz meg, amelyek teljesítik a fenti követelményeket. A VLAN-fák meghatározása két fázisra van osztva a kezelni kívánt kétféle hibatípus szerint. Az algoritmus megpróbál előrelátó lenni, amennyire lehetséges, úgy, hogy minimalizálja a kapcsolati ill. csomóponti hibák kezeléséhez használt fák számát. Annak ellenére, hogy az algoritmus az 1. fázisban a kapcsolati hibákat veszi célba, a fákat úgy alkotja meg, hogy azok lehetőleg a csomóponti hibákat is kezeljék, vagy legalább a csomópontok védelméhez szükséges fák meghatározását ne rontsa el. Továbbá, az egyes lépésekben a döntéseket a lehetséges további lépések figyelembevételével hozza meg. Az algoritmus az előrelátó működés érdekében számos attribútumot vesz figyelembe, amelyek részletes leírása a [D]-ben található. Az algoritmus lényege a következő:

1.3. algoritmus

1. fázis – 1.3-1. algoritmus: *Feszítőfák meghatározása a kapcsolatok védelmére*

1. lépés: A központi csomópont kiválasztása, amely a legmagasabb fokszámú csomópont.

2. lépés: Az első fa megalkotása a központi csomópontból kiindulva az összes lehetséges kapcsolat felhasználásával, de minden csomópontnál egy kapcsolatot fenntartva a második fa számára, amennyiben ez lehetséges. Így az első fa a központi csomópontból kiinduló csillagszerű topológia lesz, amely lehetővé teszi egy diszjunkt fa megalkotását, amennyiben ezt a topológia megengedi.

3. lépés: A további fák megalkotása, amíg a kapcsolati hibák lekezeléséhez szükséges R1 kritérium nem teljesül. A további fák szintén a központi csomópontból kiindulva kerülnek meghatározásra. Azon csomópontok, amelyek még nem levelek egyik korábbi fában sem, lehetőleg csak egy kapcsolattal kerülnek bekötésre a fába, ezzel segítve a csomópontok védelméhez szükséges fák meghatározását. Az algoritmus fő célja ebben a lépésben az, hogy elkerülje olyan kapcsolat hozzáadását a fához, amely minden korábbi fában szerepel. Ha ez nem lehetséges, akkor további fára van szükség.

2. fázis – 1.3-2. algoritmus: *Feszítőfák meghatározása a csomópontok védelmére*

4. lépés: További fák megalkotása, amíg a kapcsolati hibák lekezeléséhez szükséges R1 kritérium nem teljesül. Az aktuális fa úgy kerül meghatározásra, hogy az elágazási pontok olyan csomópontok legyenek, amelyek levelek egy korábbi fában. Azon csomópontok, amelyek még nem levelek egy fában sem, csupán egy kapcsolattal kerülnek bekötésre az aktuális fába. Amennyiben ez nem lehetséges – vagyis egy még nem levél csomópontnak elágazási pontnak kell lennie ahhoz, hogy feszítőfát kapjunk – akkor további fára van szükség.

Az algoritmus fontos tulajdonsága, hogy minimalizálja a hibakezeléshez használt fák számát, amit az 1. táblázat is illusztrál. Az 1.3-1. algoritmus mindig a táblázatban

szereplő átlagérték felfelé vagy lefelé kerekített értékét adta eredményül. Továbbá az 1.3-1. algoritmust kiértékeltem véletlen topológiákon végzett kimerítő szimulációk segítségével, ahol a topológia mérete 5-től 50 csomópontig változott, a csomópontok átlagos fokszáma pedig 2,5 és 5 közötti értékeket vett fel. Az eredmények azt mutatták, hogy az 1.3-1. algoritmus legfeljebb egyel több fát eredményezett, mint az alsó korlát, ha az átlagos fokszám legalább 2,8 volt.

A fizikai topológia pontos ismerete elengedhetetlen a VLAN-fák meghatározásához. Ezért definiáltam egy algoritmust, amely képes felderíteni a teljes fizikai topológiát több gyártó hídjait tartalmazó Ethernet hálózatokban, abban az esetben is, ha a hidak nem implementálják a topológia-felderítést elősegítő szabványt. Az algoritmus feltételezi, hogy a menedzselhető hidak implementálnak néhány alapvető szabványt: STP vagy RSTP, VLAN, SNMP [16], Bridge MIB [17], MIB-II [18] és Interface MIB [19]. Az algoritmus a következő lépésekből áll:

1.4. algoritmus

1. lépés – LLDP felderítés: Azon szegmens topológiájának felderítése, amely implementálja a Link Layer Discovery Protocol (LLDP) [20] szabványt, amely a topológia felderítést elősegítő szabvány.

2. lépés – Csomópont felderítés: Az LLDP felderítés után a hálózati menedzsment rendszer (Network Management System – NMS) – amely a topológia-felderítő algoritmust implementálja – kiküld egy adatszórásos ping üzenetet az alhálózat adatszórás címére, majd várja a válaszokat azon csomópontok felderítéséhez, amelyek nem támogatják a szabványos topológia felderítést.

3. lépés – Feszítőfa felderítés: A ping üzenet és az arra érkezett válaszok alapján az NMS meghatározza azokat a kapcsolatokat, amelyek részt vesznek a nem LLDP-képes hidak közti feszítőfában, amelyet a menedzsment forgalom is használ.

4. lépés – Inaktív kapcsolatok felderítése: Végül azon nem LLDP hidak közti kapcsolatok kerülnek felderítésre, amelyek nem részei a feszítőfának. Ehhez az NMS belép a nem LLDP-képes hidakra, és lekapcsolja azokat a portokat, amelyek még nem részei a topológia adatbázisnak. Emiatt az NMS SNMP üzenetet kap az inaktív link mindkét végéről. Az inaktív kapcsolatokkal együtt a topológia adatbázis teljes lesz. Megjegyzés: a feszítőfán kívüli kapcsolatok le- és felkapcsolása nem okoz változásokat a feszítőfa protokollban, és a felhasználói forgalmat sem zavarja.

Az öt gyártó hídjaiból álló, különféle, 12 csomópontig terjedő szövevényes (mesh) topológiákat alkotó teszt hálózaton végzett mérések igazolták, hogy az algoritmus pontos. Az algoritmus részletes leírása és a méréssel kapcsolatos további részletek a [D]-ben találhatók.

3.2. Kiterjesztések az SPB-hez

A feszítőfa protokollok által vezérelt Ethernet hálózatokban a kereteket gyakran kerülő úton továbbítják. Ezért az IEEE 802.1aq Shortest Path Bridging (SPB) [9] szabvány bevezeti a kapcsolatállapot alapú vezérlést Ethernet hálózatokban, ami a kerettovábbítás hatékonyabbá tételét célozza a legrövidebb út használatával. A többesküldés (multicast) támogatása miatt az SPB a legrövidebb úton való továbbításra a forrásnál gyökerező (source rooted) legrövidebb utakból álló fákat (Shortest Path Tree – SPT) alkalmazza, vagyis minden hídnak saját fája van a keret küldéshez. Azonban a létező kapcsolatállapot alapú protokollok kiterjesztésre szorulnak ahhoz, hogy alkalmazhatóak legyenek az SPB-ben, pl. azért, mert nem tartalmaznak hurokmegelőzést, ami pedig kritikus az Ethernet hálózatokban.

2. Tézis *Megmutattam, hogy hurokmegelőzési mechanizmus alkalmazása szükséges a kapcsolatállapot elven vezérelt Ethernet hálózatokban. Definiáltam hurokmegelőzési algoritmusokat, amelyek az IS-IS protokoll kiterjesztéseként implementálhatók, így beépíthetők az IEEE 802.1aq SPB architektúra vezérlő protokolljába. Bebizonyítottam, hogy a javasolt algoritmusok megelőzik a hurkok kialakulását. Realisztikus topológiákon végzett kiterjedt szimulációkkal vizsgáltam a javasolt algoritmusok hatását a hálózati konvergenciaidőre.*

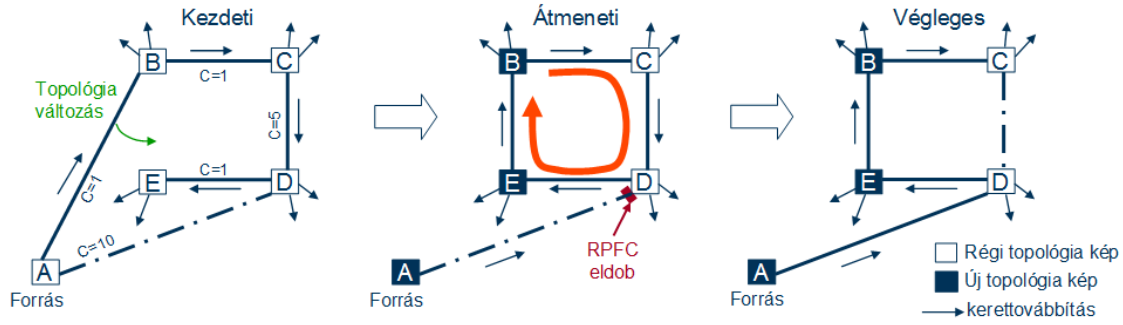
A minden időpillanatbeli hurokmentes működés alapkövetelmény Ethernet hálózatokban. Ezért az SPB-nek magában kell foglalnia egy mechanizmust, amely meggátolja a hurkok kialakulását függetlenül a folyamatban lévő kapcsolatállapot üzenetek számától, azok érkezési sorrendjétől, ill. a kapcsolatállapot alapú számításokban való részvételüktől. Voltak hurokmegelőző javaslatok az IP alapú gyors hibajavításhoz (IP Fast Re-Route – IPFRR), azonban nem volt olyan vezérlő protokoll javaslat, amely képes több topológia változást is kezelni elfogadható időn belül, ahogyan azt Shand [21] összefoglalta. Ezért nem alkalmazhatók SPB-ben.

Hurokenyhítő mechanizmusok alkalmazását javasolták SPB-ben az átmeneti hurkok kezelésére. A visszafelé út ellenőrzése (Reverse Path Forwarding Check – RPFC) használható annak vizsgálatára, hogy egy keret érkezési portja a forrás felé vezető legrövidebb úton van-e. Az RPFC-re belépési ellenőrzésként hivatkozik az SPB specifikáció [9].

2.1. Tézis *Megmutattam, hogy a visszafelé út ellenőrzése nem elégséges a hurkok megelőzéséhez. [S7]*

Észrevétel: Az általam adott ellenpélda Farkas hurokként szerepel az idegen hivatkozásokban, pl. Allan [22].

Minden stabil csomagtovábbítási topológia hurokmentes, mind Ethernet, mind IP



3. ábra. Farkas hurok

hálózatokban, azonban a topológia változás során kialakulhatnak hurkok. A 3. ábra mutat egy példát ilyen topológia változásra, ahol az RPFC (ingress check) ellenére hurok keletkezik.

Az ábra csak egy részét mutatja a topológiának, további csomópontok lehetnek a B, C, D vagy E csomópontokhoz kötve. Az ábra az összes kapcsolatot és azok költségét mutatja az ábrán szereplő csomópontok között. A folytonos vonallal jelölt kapcsolatok aktívak A csomópont SPT-je szempontjából, a pontvonallal jelölt kapcsolatok inaktívak, pl. az RPFC általi csomagdobás miatt. A nyilak a kerettovábbítás irányát mutatják A SPT-jén.

A kezdeti topológia megváltozik a példában: az A és B közötti fizikai kapcsolat megszakad, ugyanakkor egy új fizikai kapcsolat jön létre B és E között. A kezdeti és a végleges topológia hurokmentes, ahogy azt az ábra is mutatja. Az A és D közötti kapcsolat használaton kívül van a kezdeti topológiában, míg a C és D közötti kapcsolat inaktív a végleges topológiában. Az átmenet során azonban hurok keletkezik, ha az A, B és E csomópontok tudnak a változásról és pontos képük van a topológiáról, de C és D csomópontok nem tud a változásról, ezért elavult képük van a topológiáról. A hurok annak ellenére jelentkezik, hogy az RPFC-t használjuk. Többesküldés ill. adatszórás esetén egy keretről több másolat is a hálózat többi részébe kerül a B, C, D és E csomópontokon keresztül a hurok következményeként, ahogyan azt a nyilak mutatják. Érdemes megjegyezni, hogy az IP csomagokban alkalmazott hátralévő élettartam (Time To Live – TTL) mező egy gyengébb hurokkezelési technika, mint az RPFC, mert a TTL nem akadályozza meg a hurkok kezelését, hanem egy eszközt ad a velük való együttélésre.

Mivel a létező metódusok nem adnak megfelelő megoldást hurokmegelőzésre kapcsolatállapot alapon vezérelt Ethernet hálózatokban, szükség van egy új, hatékony mechanizmusra.

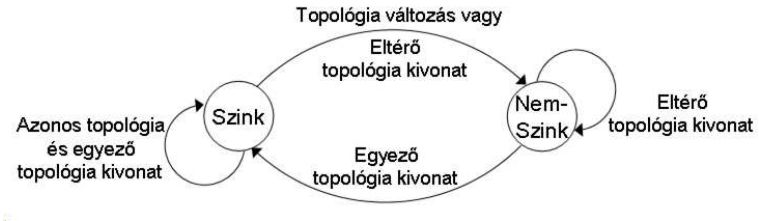
2.2. Tézis *Definiáltam a Szomszédok Szinkronizálása hurokmegelőzési algoritmust, és megmutattam, hogy az algoritmus hurokmentes működést biztosít a kapcsolatállapot elven vezérelt hálózatokban, amennyiben azon szomszédos csomópontok, amelyeknek a topológiáról alkotott képük nem egyezik, eldobják a csomagokat ahelyett, hogy továbbítanák azokat egymásnak. [J2, S6, P7]*

Észrevétel: A Szomszédok Szinkronizálása algoritmus egy egyszerű kiterjesztés hurokmegelőzésre a meglévő kapcsolatállapot alapú protokollokhoz.

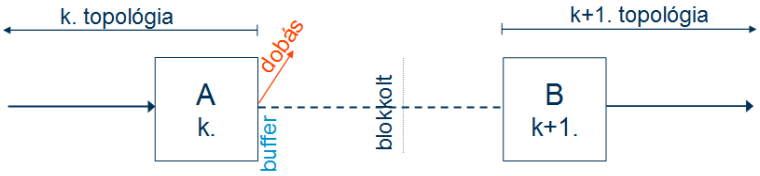
Ha kapcsolatállapot alapú protokollt használunk a hálózat vezérlésére, akkor átmeneti hurkok keletkezhetnek amiatt, hogy a csomópontok fizikai topológiáról alkotott képe különbözhet a topológia változás miatt. Ezért a hurkok kialakulásának meggátolására a Szomszédok Szinkronizálása algoritmust javasoltam, ahol a szomszédos csomópontok kézfogásos technikával biztosítják, hogy a topológiáról alkotott képük megegyezik. Ha a topológiáról alkotott képük eltér, akkor a szomszédok nem küldenek egymásnak adatcsomagokat. A Szomszédok Szinkronizálása nagyon jól illeszkedik a szabványos kapcsolatállapot alapú protokollokhoz, pl. az IS-IS-hez. Az egyező topológia adatbázis ellenőrzése történhet a topológia adatbázis kivonatának szomszédok közti cseréjével.

A topológia adatbázis szinkronizálása közvetlen szomszédok között valósul meg, és független egy másik csomópont pár között lezajló szinkronizálástól. Ezért egy csomópont különböző portjainak a szinkronizációs állapota különböző lehet. Egy port állapotait egy két állapotú állapotgéppel lehet leírni, ahogyan azt a 4. ábra mutatja. Az állapotok neve azt tükrözi, hogy a csomópont szinkronban van-e vagy sem az adott porthoz kapcsolódó szomszédjával.

A port szinkronban van (Szink), ha a szomszéd topológia kivonata megegyezik a saját topológia kivonattal, egyébként nincs szinkronban (Nem-Szink). A port Szinkben marad, amíg nem értesül topológiaváltozásról, illetve amíg a szomszédtól nem kap a sajátjától eltérő topológia kivonatot. Ha a kettő közül valamelyik bekövetkezik, akkor a port állapot Nem-Szink-re változik. A port Nem-Szink-ben marad, amíg a két szomszédnak eltérő a topológia kivonata. Amint a két topológia kivonat ismét megegyezik, a port állapota Szink-re változik. Ha a port állapota Nem-Szink, akkor



4. ábra. Állapotgép a csomópontok portjain



5. ábra. Topológia szeparáció

a port blokkolja az adatkommunikációt az adott porthoz kapcsolódó szomszédjával. Az adatkommunikáció csak akkor működik, ha a port Szink állapotban van.

Indirekt bizonyítással megmutattam, hogy a Szomszédok Szinkronizálása megakadályozza a hurkok kialakulását – a bizonyítás részletei a [D]-ben találhatók. A hurokmegelőzés alapja az, hogy a Szomszédok Szinkronizálása biztosítja, hogy nincs átjárás a különböző topológiák között, amint azt az 5. ábra is illusztrálja. Az A csomópont a k. topológiához tartozik, mert csak a k. topológiát leíró kapcsolatállapot üzeneteket kapta meg. Ezzel szemben a B csomópont a k+1. topológiához tartozik, mivel kapott olyan kapcsolatállapot üzeneteket, amelyek különböznek a k. topológiától. A Szomszédok Szinkronizálása blokkolja az A és B közti kapcsolatot, mert a két csomópont topológia képe különböző.

Ha A egy csomagot kap a k. topológián, amelyet B-nek kellene továbbítania, akkor A két műveletet hajthat végre a csomagon. A vagy eldobja a csomagot, vagy eltárolja egy bufferben, és csak akkor továbbítja B-nek, ha A és B ismét ugyanahhoz a topológiához tartozik. Ha bufferelést alkalmazunk, akkor a csomag végrehajthat topológia átlépést, vagyis átléphet a k. topológiáról a k+1.-re.

Ha egy csomag átléphet egyik topológiáról egy másikra, akkor a csomag többször is áthaladhat ugyanazon a csomóponton.

2.3. Tézis *Megmutattam, hogy a Szomszédok Szinkronizálása még akkor is optimális hurokmegelőzési algoritmus, amikor bufferelést alkalmazunk csomagdobás helyett, mivel biztosítja, hogy minden csomag úgy jut célba, hogy egy csomóponton legfeljebb k+1-szer halad át, ha legfeljebb k topológia változás volt a hálózatban.*

Ha a csomagokat eldobás helyett bufferelјük, amíg a szomszédok nincsenek szinkronban, és továbbítjuk, amint a szomszédoknak ismét azonos a topológia képük, akkor ugyanaz a csomag többször áthaladhat ugyanazon a csomóponton az események nagyon valószínűtlen sorozata ill. együttállása folytán, amire egy példa található [D]-ben. Egy optimális hurokmegelőzési algoritmus minimalizálja egy csomag egy csomóponton történő áthaladásainak számát még ilyen valószínűtlen esetekben is.

2.1. Definíció. Egy, a célja felé tartó csomag állapotát (X, D) határozza meg, ahol – X a hálózatban történt topológia változások számának és a csomag topológia

átlépési számának különbsége, továbbá
– D a célig hátralévő ugrások száma az adott topológián belül.

Az állapotokra lexikografikus sorba rendezést lehet alkalmazni:

$$(X_1, D_1) > (X_2, D_2) \equiv (X_1 > X_2) \vee (X_1 = X_2 \wedge D_1 > D_2). \quad (3)$$

Az 1-es állapot nagyobb, mint a 2-es, ha a 2-es állapotban lévő csomag több topológia átlépést hajtott végre, mint az 1-es állapotban lévő csomag, vagy ha a két különböző állapotban lévő csomag ugyanazon a topológián belül van, de a 2-es állapotban a céltól való távolság kisebb, mint az 1-es állapotban.

A Szomszédok Szinkronizálása biztosítja, hogy a csomag állapota mindig szigorúan csökkenő: X vagy D csökken a csomag állapotában a korábbihoz képest, egyébként a két állapot megegyezik. Ennek az oka, hogy különböző topológiákban lévő csomópontok nem küldenek egymásnak adatcsomagot, ahogyan azt az 5. ábra is mutatja. A csomópontok csak akkor küldenek egymásnak csomagot, ha azonos a topológia képük, vagyis ugyanannak a topológiának a tagjai. Egy adott topológián belül minden csomag egy fa mentén továbbítódik úgy, hogy egy csomóponton csak egyszer halad át, ezáltal a céltól való távolsága minden ugrással csökken.

Bufferelés miatt egy csomag átléphet egy régebbi topológiáról egy újabbra, ha a csomagot tároló csomópont átkerül az új topológiára. Ha k topológia változás van a hálózatban, akkor a csomag legfeljebb k-szor léphet át egy újabb topológiára. Így a csomag legfeljebb k+1 topológián továbbítódhat. Ezért tehát a Szomszédok Szinkronizálása biztosítja, hogy egy csomagot egy csomópont legfeljebb k+1-szer továbbíthat, így a Szomszédok Szinkronizálása optimális hurokmegelőzést biztosít még az események valószínűtlen konstellációja esetén is.

A csomagtovábbítás felfüggesztése a szomszédok között növelheti a topológia változások utáni hálózati konvergenciaidőt.

2.4. Tézis *Valós és mesterséges topológiákon végzett kiterjedt szimulációkkal megmutattam, hogy a Szomszédok Szinkronizálásán alapuló hurokmegelőzési algoritmus csupán milliszekundumokkal növeli a hálózati konvergencia időt. Prototípus implementáción végzett mérésekkel igazoltam, hogy a Szomszédok Szinkronizálásának hatása elhanyagolható a hálózat konvergencia idejéhez képest, szabványos protokoll paraméterek alkalmazása esetén. [S1]*

A Szomszédok Szinkronizálása algoritmust az OMNeT++ 4.0-hoz fejlesztett szimulátor segítségével analizáltam, amely egy C++ alapú, diszkrét eseményű szimulátor. A híd architektúra az OMNeT++ INET keretrendszerében került implementálásra az IEEE 802.1Q [2] specifikáció szerint. Erre került rá az IS-IS mint egy felsőbb szintű entitás (Higher Layer Entity) a híd architektúrában. A Szomszédok Szinkronizálásához szükséges kézfogást IS-IS Hello PDU-k valósítják meg.

A szimulációs analízishez hat topológiát használtam: a 22 csomópontos AT&T [24], a 37 csomópontos COST266 [25] referencia topológiákat, egy 50 csomópontos német gerinchálózati topológiát (Német50 [24]), egy mesterséges, több gyűrűből álló, ezért Gyűrűknek nevezett topológiát, továbbá 100 ill. 150 csomópontos véletlen topológiákat (R100 és R 150). Az IS-IS paramétereket úgy állítottam be, hogy kiküszöböljem a mesterséges késleltetéseket, pl. a topológia változásról való értesülés és a Dijkstra számítás közé beiktatott késleltetést. Így lehetséges volt a Szomszédok Szinkronizálása algoritmus hatásának vizsgálata.

A 2. táblázat száz szimulációs eredmény átlagát mutatja a vizsgált esetek mindegyikében. A Szomszédok Szinkronizálása nélküli (Sz. Szink. nélkül) illetve az azt alkalmazó (Sz. Szink.) eredményeket összevetve látható, hogy a különbség körülbelül 1 ms kapcsolati hiba esetén. A különbség jobban változik csomóponti hiba esetén, ekkor 1 ms és 10 ms közötti értéket vesz fel. Az eredmények alapján megállapítható, hogy a Szomszédok Szinkronizálása nem rontja el a hálózati konvergenciát.

A Szomszédok Szinkronizálásának működését mérésekkel is kiértékeltem egy hat Debian GNU/Linux PC-ből álló prototípus hálózatban. A prototípus a *quagga* nyílt forráskódú készlet *isisd* útvonalválasztó démonját használja az IS-IS megvalósítására. A Szomszédok Szinkronizálása az *isisd* démonban került implementálásra. Az IS-IS paramétereket a szabvány [8] által megengedett legkisebb értékre állítottam a mérések során.

A konvergenciaidőt a Szomszédok Szinkronizálása hurokmegelőző algoritmussal illetve anélkül is vizsgáltam. A konvergenciaidőt a kapcsolati hibáról szóló első értesítés megérkezésétől a topológiaváltozáshoz kapcsolódó utolsó csomagtovábbítási bejegyzés (FIB) változásáig mértem.

A húsz mérésből származó átlagos konvergenciaidő a hurokmegelőző algoritmus alkalmazása nélkül 2,03 másodperc volt. A Szomszédok Szinkronizálása az átlagos konvergenciaidőt 2,1 másodpercre emelte, vagyis a különbség két nagyságrenddel kisebb,

2. táblázat. Hibaesemény utáni átlagos konvergencia idő [ms]

Kapcsolati hiba

	AT&T	COST266	Német50	Gyűrűk	R100	R150
Sz. Szink. nélkül	7,980	18,632	35,432	77,331	163,648	394,811
Sz. Szink.	9,008	19,615	36,019	78,336	164,734	395,758

Csomóponti hiba

	AT&T	COST266	Német50	Gyűrűk	R100	R150
Sz. Szink. nélkül	9,339	21,044	37,629	86,904	164,490	395,140
Sz. Szink.	10,202	29,319	44,809	95,342	165,333	396,746

mint maga a konvergenciaidő.

Megállapítható tehát, hogy a Szomszédok Szinkronizálása nem növelte jelentősen a konvergenciaidőt szabványos IS-IS paraméter beállítások mellett. Mind a szimulációs, mind a méréses vizsgálatok igazolták, hogy a Szomszédok Szinkronizálása megakadályozza a hurkok kialakulását, amelyek hurokmegelőzés nélkül jelentkeznének. Továbbá a Szomszédok Szinkronizálása nem rontja el a hálózati konvergenciát.

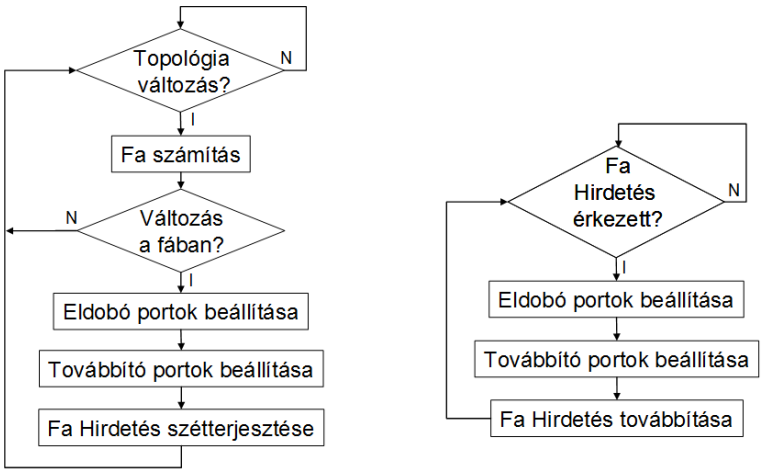
2.5. Tézis *Definiáltam a Gyökér Vezérelt Hidakat (Root Controlled Bridging – RCB) az SPB vezérléséhez, és megmutattam, hogy az RCB megakadályozza a hurkok kialakulását. Továbbá az RCB a számítási komplexitást* $\mathcal{O}(|L| + |N| \cdot \log|N|)$*-ra csökkenti az alternatív megoldások* $\mathcal{O}(|N|(|L| + |N| \cdot \log|N|))$ *komplexitásához képest a* $G(N, L)$ *topológiákon, amelyek* $|N|$ *csomópontból és* $|L|$ *kapcsolatból állnak. [C2, S7, P9, P10]*

Észrevétel: Az RCB egy olyan kiterjesztés a meglévő kapcsolatállapot alapú protokollokhoz, amely a számítási komplexitáson is javít a hurokmegelőzés biztosítása mellett.

Az SPB megoldást, amely szabványos IS-IS-t alkalmaz speciális útvonal számítási kiterjesztések nélkül, Alap IS-IS-nek (Basic IS-IS) nevezzük a továbbiakban. Az Alap IS-IS-ben minden csomópontnak ki kell számolnia az összes csomópont SPT-jét, ami a helyes kerettovábbítás megvalósításához szükséges; tehát egy összes pár közti legrövidebb út (All Pairs Shortest Path) számítást kell elvégezniük. Ezért a számításigény az Alap IS-IS-ben $\mathcal{O}(|N|(|L| + |N| \cdot \log|N|))$.

Javasoltam egy új megközelítést az SPB hálózatok vezérlésére, amelyet Gyökér Vezérelt Hidaknak (Root Controlled Bridging – RCB) nevezünk, és az IS-IS kiterjesztéseként implementálható. Az RCB-ben a hidak ugyanúgy fenntartják a kapcsolatállapot adatbázist, ahogyan azt az IS-IS meghatározza, azonban minden híd csupán a saját fáját számolja ki, és vezérli annak fenntartását, frissítését, tehát minden fát annak gyökér hídja kontrollálja. Így az architektúra elosztott és robusztus, mivel az SPT-k egymástól függetlenül vannak vezérelve. Azonban minden fa vezérlése centralizált, aminek jelentős előnyei vannak a számításigény lecsökkentése szempontjából. Mivel minden RCB híd csak egyetlen SPT-t számol, a számításigény lecsökken: $\mathcal{O}(|L| + |N| \cdot \log|N|)$. Ha egy híd meghibásodik, akkor annak SPT-je feleslegessé válik.

Az RCB működése csak a fák kiszámításában és azoknak a hálózatban történő beállításában tér el a szabványos IS-IS-től. Ennek a két folyamatnak a működését írják le a 6. ábrán látható folyamatábrák. Ahogyan azt a 6(a) ábra mutatja, a gyökér híd új SPT-t számol, ha változás történt a topológiában. Ha az SPT is megváltozott, akkor azt be kell állítani a hálózatban. Ebben az esetben a gyökér híd beállítja az eldobó (discarding) portjait, majd a továbbító (forwarding) portokat. Az eldobó port



(a) Fa számítás (b) Fa beállítás

6. ábra. Fa számítás és beállítás

minden beérkező keretet eldob. Ezután a gyökér híd Fa Hirdetés (Tree Advertisement – TA) üzenetek segítségével behirdeti az SPT-jét a többi híd számára. A TA üzenetek egy új TLV-ben implementálhatók, így a TA üzenet egy szabványos kiterjesztés az IS-IS-hez.

A 6(b) ábra mutatja egy fa beállítását vagy frissítését a TA üzenet vétele után egy hídban, amely nem a gyökér. A TA üzenet gyökértől a levelek felé történõ terjesztésének egy fontos tulajdonsága, hogy csak azon fa mentén továbbítódik, amelyet önmaga ír le, vagyis nem kerül elárasztásra. Az SPT frissítési folyamat további kulcsfontosságú tulajdonsága, hogy az eldobó portok beállításra kerülnek, mielőtt a továbbító portok beállításra kerülnének, ill. a TA üzenet továbbítódna, ahogyan azt a 6. ábra is mutatja. Ez biztosítja a hurokmentes működést, ezért az RCB-ben nincs szükség további mechanizmusra a hurkok elkerülésére.

Egy indirekt bizonyítással megmutattam, hogy az RCB megakadályozza a hurkok kialakulását. A bizonyítás a [D]-ben található.

Az RCB csökkenti a számítási komplexitást, ehhez azonban új üzeneteket használ, amely befolyásolhatja a hálózati konvergenciaidőt.

2.6. Tézis *Különféle topológiákon és paraméter beállítások mellett végzett kiterjedt szimulációk segítségével megmutattam, hogy az RCB gyorsabb hálózati konvergenciát biztosít a 200 csomópontnál nagyobb hálózati topológiákon, mint az alternatív kapcsolatállapot alapú megoldások. [C1]*

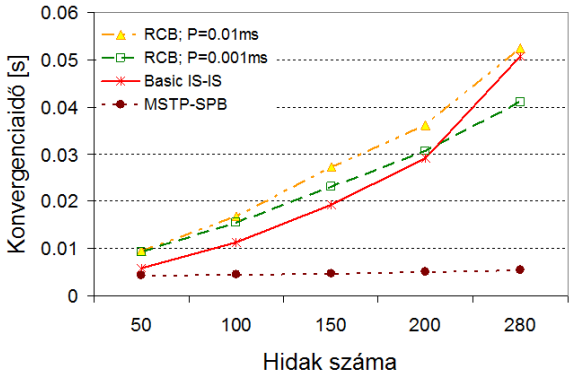
Kiértékeltem és összehasonlítottam az Alap IS-IS és az RCB kapcsolati és csomóponti hibák utáni konvergencia idejét különféle típusú és méretű topológiákon. Továb-

bá egy MSTP alapú SPB vezérlés (MSTP-SPB) is a vizsgálatok részét képezte, amely a legelső javaslat volt a szabványosítás kezdetekor. Ezáltal az eredmények kapcsolatállapot ill. távolságvektor alapú működésre is adnak egy összehasonlítást. Háromféle topológia került kiértékelésre. A Gyűrűk topológia egy középső gyűrűhöz kapcsolódó további gyűrűkből áll. Ezen kívül enyhén ill. sűrűn összekötött szövevényes topológiákat alkalmaztam. A hálózatot alkotó csomópontok száma 50-től 280-ig változott. A 7. ábra mutatja a kapcsolati hiba utáni konvergenciaidőt enyhén szövevényes topológián. A további eredmények és a paraméter beállítások részletes leírása a [D]-ben található.

A ritka topológiák, mint pl. a gyűrűk, nem olyan kedvezőek a vezérlő üzeneteket intenzíven használó megoldások számára, mint pl. az MSTP-SPB vagy az RCB, mivel a vezérlő információnak hosszú utat kell megtennie. Ezért a Gyűrűk topológián az Alap IS-IS konvergenciaideje kisebb, mint az RCB konvergenciaideje.

Az SPB-t azonban szövevényesebb topológiákon célszerű alkalmazni, ahol a legrövidebb út használata javítja a csomagtovábbítás hatékonyságát. Az Alap IS-IS konvergenciaidejét jelentősen befolyásolja a hálózat mérete, mivel a csomópontok és a kapcsolatok számának is hatása van a számítási komplexitásra. Ezért szövevényes topológiákon egy bizonyos méret felett az RCB jobban teljesít, mint az Alap IS-IS. Enyhén szövevényes topológián belüli kapcsolati hiba esetén a legrealisztikusabb TA üzenetfeldolgozási beállítás mellett (P = 0,001 ms) az RCB konvergenciaideje kisebb, mint az Alap IS-IS konvergenciaideje, amennyiben a hálózat legalább 200 csomópontot tartalmaz, ahogyan a 7. ábra is illusztrálja.

A teljesítményanalízis megmutatta, hogy az Alap IS-IS konvergenciaideje érzékeny a hálózat méretére szövevényes topológiák esetén, a számítási komplexitás miatt. Az RCB üzenetküldések árán csökkenti a számítási komplexitást, és gyorsabban konvergál, mint az Alap IS-IS, ahogy a szövevényes hálózat mérete növekszik. Minél



7. ábra. Kapcsolati hiba enyhén szövevényes topológiában

sűrűbben összekötött és nagyobb a topológia, az RCB annál gyorsabban konvergál az alternatív Alap IS-IS-hez képest.

3.3. Kapcsolatengedélyezés UTRAN-ban

Az UTRAN-ban kapcsolatengedélyező (Connection Admission Control – CAC) algoritmust kell implementálni ahhoz, hogy a beengedett kapcsolatok teljesíteni tudják a szigorú QoS követelményeket. Másrészt a CAC-nak lehetővé kell tennie az elérhető hálózati erőforrások kihasználását.

3. Tézis *Olyan kapcsolatengedélyező algoritmusokat definiáltam az UTRAN Iub interfészéhez, amely figyelembe tudja venni a folyam szintű tulajdonságokat, illetve a hálózati csomópontokban implementált WFQ ütemezés előnyeit kihasználva lehetővé teszi a jobb hálózati kihasználtság elérését.*

Az UTRAN Iub interfészének forgalma független ON-OFF modulált periodikus forrásokkal modellezhető. Az i forgalmi osztályba tartozó forgalmi források a $\{T_i, b_i, \alpha_i\}$ paraméter halmazzal írhatók le. Az adási időközt (Transmission Time Interval – TTI) T_i jelöli, ami a csomagok érkezése közötti determinisztikus idő, amennyiben a forrás ON állapotban van. A csomagméretet b_i jelöli, és α_i az aktivitás faktor, ami az ON-OFF viselkedést írja le. Az i osztály QoS követelményeit statisztikusan írjuk le a $\{d_i, \varepsilon_i\}$ paraméterekkel, ahol d_i a késleltetési követelmény és ε_i a megengedett csomagvesztési ráta. Az UTRAN rendszer K forgalmi osztályból áll, és az N_i változó írja le az i osztályban lévő kapcsolatok aktuális számát.

A QoS követelmény sérül, ha a buffer *túlterhelt*, vagyis a bemeneti rátája nagyobb, mint a kiszolgálási rátája. A túlterhelés okozta QoS sértést *börszt szintű QoS sértés*nek nevezünk, mivel ezt az ON állapotban lévő források börsztje okozza. A QoS követelmény nagy csomag*késleltetés* miatt is sérülhet, ami egy átmeneti csomagtorlódás miatt annak ellenére is előfordulhat, hogy a buffer bemeneti rátája kisebb, mint a kiszolgálási rátája, ezért *csomagszintű QoS sértés*ként, ill. *késleltetés sértés*ként hivatkozunk rá. Malomsoky [27, 28] javasolt egy modellt, amely ezt a kétféle QoS sértést külön kezeli. Tehát a csomagdobási követelmény kettéválasztható:
$\varepsilon_i = \varepsilon_i^{börszt} + \varepsilon_i^{csomag}$.

Ezáltal a börszt szintű ill. a csomagszintű működést külön lehet vizsgálni. Tehát a CAC algoritmus egymástól függetlenül ellenőrizheti a börszt ill. csomagszintű QoS sértést egy új kapcsolat beengedése előtt.

A CAC algoritmus kihasználhatja a rendszerről és a beérkező forgalom jellemzőiről rendelkezésre álló ismereteket. A fent leírt modell alkalmazható az UTRAN Iub interfészére. A Roberts [29] által részletesen leírt $n \cdot D/D/1$ sorbanállási rendszer kiterjeszthető az UTRAN Iub működésének leírásához. Tehát modell alapú CAC

algoritmust alkalmazhatunk az UTRAN Iub interfészén.

A börszt szintű működés modellezéséhez buffermentes multiplexálást alkalmazhatunk, mivel a késleltetési követelmények kicsik a börszt szintű működési dinamizmushoz képest. Ezzel szemben, ha a késleltetési követelmény csomagszintű működés miatt sérül, akkor feltételezhetjük, hogy a buffer nem csordul túl. Ezért a csomagszintű működés vizsgálatakor a rendszert úgy modellezhetjük, mintha a buffer végtelen nagy lenne.

Egy modell alapú CAC algoritmust tárgyalunk a továbbiakban.

3.1. Tézis *Definiáltam egy olyan kapcsolatengedélyező algoritmust, amely kihasználja az IP UTRAN hálózatban alkalmazott WFQ ütemezés nyújtotta előnyöket. Szimulációk segítségével megmutattam, hogy az algoritmusom javítja a kis kapacitású kapcsolatok sávszélesség kihasználtságát, pl.* $2 \cdot E1$ *sávszélesség esetén* 46%*-kal. [C10, P18]*

A csomagszintű QoS sértés ellenőrzéséhez a CB-WFQ rendszer szigorú prioritásos (Strict Priority – SP) rendszerek halmazával történő közelítését javasoltam. Ezt a közelítést Szeparált Szigorú Prioritásnak (Separated Strict Priority) hívják. A közelítés alkalmazására a CB-WFQ rendszer bonyolultsága miatt van szükség.

Egy forgalmi mix $\underline{N} = N_1, N_2, \ldots, N_K$, amely a K osztályból álló rendszerben minden osztályra megadja az éppen bent lévő kapcsolatok számát, akkor nem okoz csomag szintű QoS sértést, ha $\underline{N}$ a csomagszintű beengedési felület alatt található a különböző osztályokban lévő kapcsolatok száma által kifeszített térben.

A javasolt Szeparált Szigorú Prioritásos modellt alkalmazva, egy CB-WFQ rendszerben lévő forgalmi osztály komplex csomagszintű beengedési felülete hipersíkok egy halmazával közelíthető.

Egy beengedési kérelem esetén a CAC algoritmus az esetleges QoS sértést le tudja ellenőrizni az új forgalmi mixet $\underline{N}_{új}$ alkalmazva, amit úgy kapunk, mintha az új kapcsolatot beengednénk. Az új forgalmi mix $\underline{N}_{új}$ nem okoz késleltetés sértést az i osztályban, ha $\underline{N}_{új}$ az i osztály valamelyik hipersíkja alatt van, amit a következőképp ellenőrizhetünk:

$$\max_{Y} \left(F_{ii}^{Y} + 1 - \sum_{j \in Y} \frac{F_{jj}^{Y}}{F_{ji}^{Y}} \cdot N_j \right) > 0, \tag{4}$$

ahol Y a bufferek indexének halmaza, F_{ji}^{Y}, F_{ii}^{Y} és F_{jj}^{Y} pedig a hipersíkot adja meg, amelyek kiszámítása a továbbiakban kerül ismertetésre.

Ha az új forgalmi mix $\underline{N}_{új}$ minden forgalmi osztályra átmegy az 4. egyenlet által leírt ellenőrzésen, akkor az új igény beengedése nem okoz csomagszintű QoS sértést.

A fentiek alapján a cél tehát az, hogy olyan, az eredeti CB-WFQ-val egyenértékű rendszereket kapjunk, amely lehetővé teszi a k sorban kiszolgált i osztály csomagszintű

beengedési tartományának meghatározását. A 8. ábra illusztrálja a javasolt közelítő modellt egy három sort tartalmazó rendszerben. Háromféle sort különböztetünk meg: a megfigyelés alatt álló k-val jelölt sort, továbbá telített és enyhén terhelt sorokat. Vagyis a sorokat három halmazra osztjuk: $\{k\} \cup A \cup B$, ahol A az enyhén terhelt, B pedig a telített sorokat tartalmazza. A 8. ábrán mutatott példában az 1-es osztály a megfigyelt, a 2-es osztály $\in A$, és a 3-as osztály $\in B$.

A k sor S_k kiszolgálási rátájára a következő alsó korlát adható WFQ rendszerben:

$$S_k \geq \frac{c_k}{c_k + \sum_{j \in B} c_j} \left(C - \sum_{j \in A} R_j \right), \tag{5}$$

ahol R_j a j sor bemeneti rátája.

Ahhoz, hogy egy egyenértékű rendszert kapjunk, a telített buffereket (B index halmaz) szeparáljuk az ütemezőből. Így a csökkentett rendszer a vizsgált k buffert és az A index halmazban lévő buffereket tartalmazza. A kiszolgálási ráta a csökkentett rendszerben:

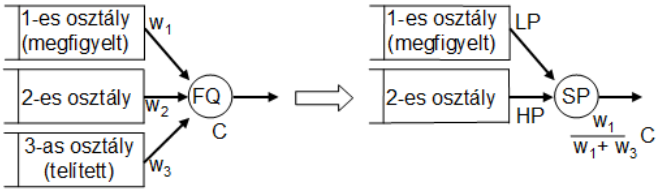
$$C' = \frac{c_k}{c_k + \sum_{j \in B} c_j} C, \tag{6}$$

amely a k sor kiszolgálási rátájára egy felső korlát.

A csomag-kiszolgálási sorrend a minimális sávszélesség (c_i) kiosztásától függ. Ahhoz, hogy a legrosszabb sorbanállási késleltetést vegyük figyelembe a k bufferben, azt feltételezzük, hogy az A-beli bufferekben lévő csomagok prioritása magasabb, mint a k bufferben lévő csomagoké.

A csomagméretet szintén módosítani kell az A-ban lévő bufferekben ahhoz, hogy a csökkentett rendszer helyesen működjön. Ezeket a csomagokat az eredeti rendszerben a teljes rátával szolgálják ki, ami egy k bufferben lévő csomag számára a csökkentett rendszerben úgy tűnik, mintha az A-ban lévő bufferek csomagméretét lecsökkentenénk:

$$b'_i = \frac{c_k}{c_k + \sum_{j \in B} c_j} b_i; \ \forall i \in A. \tag{7}$$



8. ábra. Szeparált Szigorú Prioritásos modell

Ennek eredményeképp a csökkentett rendszer úgy működik, mint egy szigorú prioritásos rendszer C' és b' paraméterekkel. Annak függvényében, hogy mely buffereket tekintjük telítettnek, 2^{L-1} csökkentett rendszert különböztethetünk meg, ahol L az UTRAN Iub valós idejű buffereinek száma. A különféle csökkentett rendszerek különböző forgalmi feltételek esetén adnak jó közelítést, a kombinációjuk pedig konzervatív közelítést ad a kiszolgálási rátára bármilyen forgalmi mix esetén. Ezért a modellből származó sorbanállási késleltetés egy felső korlát az eredeti rendszerbeli sorbanállási késleltetésre.

A Szeparált Szigorú Prioritásos modellt alkalmazva a CB-WFQ rendszer csomagszintű beengedési felületének problémája visszaesik az alacsony prioritású (LP) osztályok csomagszintű beengedési felületének közelítésére több osztályos szigorú prioritásos rendszerekben. Ahogyan azt Malomsoky [28] megmutatta, egy LP osztály késleltetési korlátjának felülete egyetlen hipersíkkal közelíthető, és ez a közelítés konzervatív.

Jelölje Y a sorok indexének halmazát, K_{SP} pedig az osztályok számát az SP rendszerben. F_{ji}^{Y} a maximális j osztályú kapcsolatok száma az SP rendszerben úgy, hogy az i osztálybeli egyetlen kapcsolat késleltetési követelménye teljesül, és az összes többi osztály üres. Formalizálva:

$$F_{ji}^{Y} = \max \left\{ N_j \mid \sum_{n_j=0}^{N_j} \mathbb{P}\left[\mathcal{D}_i^{n_j} > d_i\right] \cdot \Pi(n_j) \leq \varepsilon_i^{csomag} \right\}, \tag{8}$$

ahol $\mathbb{P}\left[\mathcal{D}_i^{n_j} > d\right]$ egy i osztályú kapcsolat késleltetésének eloszlása, ha a rendszerben lévő j osztályú kapcsolatok száma n_j. $\Pi(n_j)$ pedig annak a valószínűsége, hogy n_j kapcsolat aktív, amelyet a többdimenziós binomiális eloszlás segítségével határozhatunk meg.

Az i osztály késleltetési korlátját közelítő hipersík egyenlete a következőképpen írható fel:

$$\sum_{j \in Y} \frac{F_{jj}^{Y}}{F_{ji}^{Y}} \cdot N_j = F_{ii}^{Y} + 1. \tag{9}$$

A csomagszintű követelmény akkor teljesül, ha a 4. egyenlet minden forgalmi osztályra érvényesül.

A börszt szintű QoS sértést is ellenőrizni kell, amihez egy gaussi közelítést javasoltam. A centrális határeloszlás tételt alkalmazva, az egy bufferben lévő aktív kapcsolatok száma normális eloszlással közelíthető, ha a források száma növekszik. Ezért a börszt szintű QoS sértés ellenőrzése elvégezhető a következő közelítő formula segítségével:

$$1 - \Phi\left(C, \tilde{R}_k, \tilde{V}_k\right) + \frac{\tilde{V}_k}{\tilde{R}_k} \cdot \varphi\left(C, \tilde{R}_k, \tilde{V}_k\right) \leq \varepsilon_k^{börszt}, \tag{10}$$

ahol $\varphi(x, \mu, \sigma^2)$ a sűrűségfüggvénye, és $\Phi(x, \mu, \sigma^2)$ az eloszlásfüggvénye a μ várható értékkel és σ^2 szórásnégyzettel rendelkező normális eloszlásnak. Továbbá $\tilde{R}_k$ a várható értéke, és $\tilde{V}_k$ a szórásnégyzete a k buffer bemeneti rátájának a CB-WFQ működése szerint korrigálva, ahogyan az a [D]-ben részletesen szerepel.

Szimulációk segítségével kiértékeltem a javasolt CAC algoritmus teljesítményét, és összevetettem a sávszélesség kihasználtságát a Szeparált FIFO algoritmuséval, amely az egyetlen alternatívája volt a Szeparált Szigorú Prioritásnak. A Szeparált FIFO modell FIFO rendszerrel közelíti a WFQ-t, vagyis a kapacitás szét van osztva a súly beállítás szerint, így a garantált ráta mindig fenn van tartva minden sor számára. Kis kapacitásoknál, pl. 2·E1, a Szeparált FIFO kapacitásigénye 46%-kal nagyobb, mint a Szeparált Szigorú Prioritás kapacitásigénye. A különbség a kapacitás növekedésével csökken, mert a csomagszintű működés kevésbé meghatározóvá válik.

3.2. Tézis *Definiáltam egy olyan kapcsolatengedélyező algoritmust, amely folyam szinten garantálja a QoS követelményeket úgy, hogy gaussi közelítést alkalmaz a buffer nélküli multiplexálási modellben; és megmutattam, hogy a csomagvesztési követelmény megsértésének valószínűsége a normális eloszlás kvantilisével közelíthető. [C9]*

Ha a QoS-t folyam szinten akarjuk garantálni aggregátumok helyett, akkor a [C9]-ben javasolt QoS definíciót kell alkalmaznunk. Az aggregátumokra, pl. egy forgalmi osztályra vonatkozó QoS követelményt tipikusan a d késleltetési követelmény és a ε csomagdobási követelmény határoz meg. Az eldobott csomagok aránya azonban a beengedett kapcsolatok száma mellett a bent lévő kapcsolatok α aktivitás faktorától is függ. A [C9]-ben javasolt folyam szintű QoS követelmény az, hogy a ε csomagdobási követelmény megsértésének δ valószínűségét kell egy előírt érték alatt tartani.

A börszt szintű működés kiértékelésére buffermentes multiplexálás alkalmazható akkor is, ha a folyamszintű QoS követelményt vesszük figyelembe. A multiplexer $Z = C \cdot T/b$ csomagot szolgál ki egy periódus alatt, és N ON-OFF forrást multiplexál, amelyek aktivitás faktorai független azonos eloszlású valószínűségi változók.

A QoS követelmény teljesülése attól függ, hogy mennyi forrás van ON állapotban egy időben, amely Bernoulli valószínűségi változók összege. A centrális határeloszlás tételt alkalmazva a δ QoS mértékre azt kapjuk, hogy:

$$\delta = \mathbb{P}\left[\frac{Z - \sum_{i=1}^{N} \alpha_i}{\sqrt{\sum_{i=1}^{N} \alpha_i(1 - \alpha_i)}} \leq \Phi^{-1}(1 - \varepsilon)\right], \tag{11}$$

ahol $\delta = 1$ QoS sértést jelent.[1]

A δ kiszámítása még kis N értékek esetén is nagyon komplikált, ami egy pontos és gyors közelítés alkalmazását teszi szükségessé.

Általános eloszlású aktivitás faktor esetén a következő algoritmust javaslom a δ kiszámítására, amelyhez a következő hányadost kell meghatároznunk:

$$\mathcal{Y} = \frac{Z - \sum_{i=1}^{N} \alpha_i}{\sqrt{\sum_{i=1}^{N} \alpha_i (1 - \alpha_i)}}. \tag{12}$$

A következő lépéseket alkalmaztam δ meghatározásához:

1. Megmutattam, hogy $\mathcal{Y}$ normális eloszlású, függetlenül az aktivitás faktorok eloszlásától;
2. Meghatároztam $\mathcal{Y}$ várható értékét és szórásnégyzetét az aktivitás faktor első momentumainak segítségével. Taylor sorba fejtést alkalmazva az $\mathcal{Y}$ első és második momentuma a következőképp számolható:

$$\begin{aligned}
\mathbb{E}[\mathcal{Y}] &= A(\mu_1) + C(\mu_1) N\sigma^2 + E(\mu_1) N\mathbb{E}\left[(\alpha - \mu_1)^3\right] + \dots \\
\mathbb{E}\left[\mathcal{Y}^2\right] &= A^2(\mu_1) + N\sigma^2 \times \\
&\quad \left[B^2(\mu_1) + 2A(\mu_1)C(\mu_1)\right] + N\mathbb{E}\left[(\alpha - \mu_1)^3\right] \times \\
&\quad [2A(\mu_1)E(\mu_1) + 2B(\mu_1)C(\mu_1)] + \dots
\end{aligned}$$

ahol μ_i és σ^2 az aktivitás faktor i.-ik momentumát, ill. szórásnégyzetét jelöli, és

$$\begin{aligned}
A(a) &= \frac{Z - Na}{\sqrt{Na(1-a)}} \\
B(a) &= -\frac{1}{2}\frac{(Z - Na)(1 - 2a)}{[Na(1-a)]^{3/2}} - \frac{1}{\sqrt{Na(1-a)}} \\
C(a) &= \frac{1}{2}\frac{Z - Na}{[Na(1-a)]^{3/2}} + \frac{3}{8}\frac{(Z - Na)\,(1 - 2a)^2}{[Na(1-a)]^{5/2}} + \\
&\quad \frac{1}{2}\frac{1 - 2a}{[Na(1-a)]^{3/2}} \\
E(a) &= -\frac{9}{4}\frac{(1 - 2a)^2}{(Na(1-a))^{5/2}} - \frac{15}{8}\frac{(Z - Na)(1 - 2a)^3}{(Na(1-a))^{7/2}} - \\
&\quad 3\frac{1}{(Na(1-a))^{3/2}} - \frac{9}{2}\frac{(Z - Na)(1 - 2a)}{(Na(1-a))^{5/2}}.
\end{aligned}$$

[1] $\Phi(x)$ és $\Phi^{-1}(x)$ a standard normális eloszlás eloszlásfüggvényét, ill. annak inverzét jelöli.

3. Ezt követően kiszámoltam a $\delta = \mathbb{P}\left[\mathcal{Y} \geq \Phi^{-1}(1-\varepsilon)\right]$-t.

A fenti eredmények olyan kapcsolatengedélyező algoritmusban alkalmazhatók, amely figyelembe veszi a folyam szintű jellemzőket, ahogyan azt a [C9] részletesen leírja. A δ_N csomagdobás sértési valószínűség N folyamra a fenti eredmények alapján a következőképpen számolható:

$$\delta_N = \Phi\left(\Phi^{-1}\left(1-\varepsilon\right);\ \mu = \mathbb{E}\left[\mathcal{Y}\right],\ \sigma^2 = \mathbb{E}\left[\mathcal{Y}^2\right] - \mathbb{E}\left[\mathcal{Y}\right]^2\right), \tag{13}$$

ahol az $\mathcal{Y}$ várható értékét és szórásnégyzetét a fenti 2. lépés alapján kell meghatározni. Ha $\delta_N < \delta$, akkor a CAC döntés "Beengedés", egyébként pedig "Visszautasítás".

A javasolt algoritmus tulajdonságait egyenletes, ill. a Westholm [30] által mért GSM beszéd szerinti aktivitás faktor eloszlásokra vizsgáltam. A kiértékelés azt mutatta, hogy a javasolt algoritmus pontos. A 13. egyenlet bizonyítása és a részletes kiértékelés a [D]-ben találhatók.

4. Módszertan

A létező, ill. már specifikált távközlő hálózati rendszereket általában mérések, szimulációk és analitikus formalizmus segítségével értékelik ki, amely a hálózatok részletes megértését segíti. Ezen kiértékelési módszerek azonban nem feltétlenül elégségesek a távközlési hálózatok fejlődésében fontos szerepet játszó innováció definiálására. A fejlődés új architektúrák, protokollok és algoritmusok specifikációját igényli, amelyeket azután ki lehet értékelni a széles körben alkalmazott módszerekkel.

Az 1. tézisben a hibatűrési követelmények teljesítéséhez egy új Ethernet architektúrát definiáltam, amely magában foglalja a szükséges protokoll komponensek és algoritmusok definiálását is. Ezen protokollok és algoritmusok teljesítményét szimulációk és prototípus implementáción végzett mérések segítségével ellenőriztem és értékeltem ki. Az architektúra teljesítményét mérések segítségével vizsgáltam.

A 2. tézisben protokoll kiterjesztéseket javasoltam ahhoz, hogy a bevezetés alatt álló kapcsolatállapot alapú vezérlést hozzáigazítsuk az Ethernet hálózatokhoz. A Szomszédok Szinkronizálása hurokmegelőző algoritmust prototípus implementáción végzett mérések segítségével ellenőriztem. Továbbá a javaslataim jellemzőit csomag szintű szimulátorral végzett szimulációs analízis segítségével határoztam meg.

A 3. tézisben új algoritmusokat definiáltam kapcsolatengedélyezésre az UTRAN hálózatok Iub interfészéhez. Ezek az algoritmusok analitikus eszközöket alkalmaznak, és szimulációk segítségével ellenőriztem őket.

5. Az eredményeim alkalmazhatósága

Pont-pont és pont-többpont Ethernet szolgáltatások védelme megoldható a PBB-TE [31] szabvány segítségével, amely hasonló alapelvekre épül, mint az 1. tézisben definiált hibatűrő architektúra. Továbbá a többpont (multipoint) szolgáltatások védelme SPB hálózatokban szintén megvalósítható az 1. tézisben leírt alapelvek szerint, amint azt [P1] részletesen taglalja. A [P16]-os szabadalmi bejelentés az 1.1. tézisben szereplő hibakezelő protokollt írja le. Az 1.3. tézisben leírt feszítőfákat meghatározó algoritmust a [P15]-ös szabadalmi bejelentés védi, és az algoritmus alkalmazható a Menth [32] és Čičić [13] által IP hálózatok védelmére javasolt több topológiás útvonalválasztásra is. A [P11]-es szabadalmi bejelentés az 1.3. tézisben szereplő topológia felderítő algoritmust írja le. Ezenkívül, az 1. tézis prototípus implementációja volt az első helyezett egy 2006-os Ericsson szintű prototípus versenyen.

Az SPB [9] által szabványosított hurokmegelőző megoldás a 2.2. tézisen alapul. Továbbá a 2.2. tézis alkalmazható hurokmegelőzésre IP alapú gyors hibajavítás (IPFRR) esetén is. A [P7]-es szabadalmi bejelentés a 2.2. tézist védi. A [P5, P8, P9, P10] szabadalmi bejelentések a 2.5. tézishez kapcsolódnak.

A 3.1. tézisben szereplő CAC algoritmus, amelyet a [P12]-es szabadalmi bejelentés véd, IP UTRAN hálózatokban alkalmazható. Egy dimenzionáló algoritmus, mint pl. a [P13]-ban leírt algoritmus, szintén figyelembe veheti az átviteli hálózat csomópontjaiban alkalmazott CAC algoritmust.

Köszönetnyilvánítás

Szeretném kifejezni köszönetemet mindenkinek, aki hozzájárult ehhez a munkához.
Elsősorban köszönöm konzulenseimnek Dr. Györfi Lászlónak és Dr. Antal Csabának a Ph. D. tanulmányaim és a kutatómunkám során nyújtott segítségüket.
Köszönöm a Traffic Lab-os kollégáimnak, különösen Dr. Antal Csabának az inspiratív környezetet és a közös munkát. Köszönettel tartozom minden szerzőtársamnak az együtt végzett kutatómunkáért.
Köszönöm Dr. Császár Andrásnak, Dr. Rácz Sándornak és Dr. Rétvári Gábornak a konstruktív javaslataikat.
Köszönöm továbbá az Ericsson Traffic Lab és a HSN Lab által nyújtott támogatást.
Legfőképpen Beatrixnek és szüleimnek köszönöm a szeretetet és a támogatást.

Hivatkozások

[1] IEEE Std. 802.1D-2004, "IEEE Standard for local and metropolitan area networks: Media Access Control (MAC) bridges," 2004.

[2] IEEE Std. 802.1Q-2005, "IEEE standard for local and metropolitan area networks: Virtual bridged local area networks," 2005.

[3] A. Myers, T. S. E. Ng and H. Zhang, "Rethinking the service model: scaling Ethernet to a million nodes," *Third Workshop on Hot Topics in Networks (HotNets-III)*, San Diego, November, 2004.

[4] ITU-T Std. G.8032, "Ethernet ring protection switching," March 2010.

[5] IEEE Std. 802.17, "Resilient packet ring," 2004.

[6] IETF RFC 3619, "Ethernet automatic protection switching," October 2003.

[7] S. Sharama, K. Gopalan, S. Nanda, and T. Chiueh, "Viking: A multi-spanning-tree Ethernet architecture for metropolitan area and cluster networks," in *Proceedings of IEEE InfoCom 2004*, March 2004.

[8] ISO/IEC 10589, "Information technology – Telecommunications and information exchange between systems – Intermediate system to intermediate system intra-domain routing information exchange protocol for use in conjunction with the protocol for providing the connectionless-mode network service (ISO 8473)," 2nd ed., 2002.

[9] IEEE Draft Std. 802.1aq D3.6, "IEEE draft standard for local and metropolitan area networks: Virtual bridged local area networks – Amendment 9: Shortest path bridging," February 2011.

[10] IETF RFC 5556, "Transparent Interconnection of Lots of Links (TRILL): Problem and applicability statement," May 2009.

[11] R. Perlman et al. , "RBridges: base protocol specification," Internet Draft, March 2010.
https://tools.ietf.org/html/draft-ietf-trill-rbridge-protocol-16

[12] D. Eastlake, "Future work for TRILL 2," Technical presentation at IETF79, November 2010.
http://www6.ietf.org/proceedings/79/slides/trill-7/trill-7_files/trill-7.pptx

[13] T. Cicic, "On basic properties of fault-tolerant multi-topology routing," *Computer Networks Journal*, Elsevier, Vol. 52, pp. 3325-3341, 2008.

[14] T. Cicic et al., "Relaxed multiple routing configurations: IP fast reroute for single and correlated failures," *IEEE Transactions on Network and Service Management*, Vol. 6/1, pp. 1-14, March 2009.

[15] T. Cicic, "An upper bound on the state requirements of link-fault tolerant multi-topology routing," in *Proceedings of ICC 2006: IEEE International Conference on Communications*, Vol. 2, pp. 1026-1031, Istanbul, June 2006.

[16] IETF RFC 1157, "A Simple Network Management Protocol (SNMP)," May 1990.

[17] IETF RFC 1493, "Definitions of Managed Objects for Bridges," July 1993.

[18] IETF RFC 1213, "Management Information Base for Network Management of TCP/IP based internets: MIB-II," March 1991.

[19] IETF RFC 2233, "The Interfaces Group MIB using SMIv2," November 1997.

[20] IEEE Std. 802.1AB, "IEEE standard for local and metropolitan area networks: Station and media access control connectivity discovery," 2005.

[21] M. Shand and S. Bryant, "A Framework for Loop-free convergence," Internet Draft, October 2009.
http://tools.ietf.org/html/draft-ietf-rtgwg-lf-conv-frmwk-07

[22] D. Allan, N. Bragg, J. Chiabaut and D. Fedyk, "802.1aq link state protocol, SPBB multicast loop prevention," Technical presentation, July 2008.
http://www.ieee802.org/1/files/public/docs2008/aq-fedyk-Loop-Prevention-0708-v01.pdf

[23] OMNeT++
http://www.omnetpp.org

[24] Survivable fixed telecommunication Network Design library (SNDlib),
http://sndlib.zib.de

[25] M. L. Garcia-Osma, "TID scenarios for advanced resilience," Technical Report of The NOBEL Project, Work Package 2, Activity A.2.1, Advanced Resilience Study Group, September 2005.

[26] GNU Quagga routing software
http://www.quagga.net

[27] S. Malomsoky, S. Rácz and S. Nádas, "Connection admission control in UMTS radio access networks," *Computer Communications – Special Issue on 3G Wireless and Beyond for Computer Communication*, Vol. 26, pp. 1907-1917, November, 2003.

[28] S. Malomsoky, "Resource management problems in packet switched networks," *Ph.D. dissertation*, Budapest University of Technology and Economics, 2003.

[29] J. W. Roberts eds., "Methods for the performance evaluation and design of broadband multiservice networks," Part III, Traffic models and queuing analysis, The COST 242 Final Report,1996.

[30] T. Westholm and B. Olin, "A model for GSM speech," in *Proceedings of 2000 Symposium on Performance Evaluation of Computer and Telecommunication Systems*, pp. 458-62, July 2001.

[31] IEEE Std. 802.1Qay, "IEEE standard for local and metropolitan area networks: Virtual bridged local area networks – Amendment 10: Provider backbone bridge - traffic engineering," 2009.

[32] M. Menth and R. Martin, "Network resilience through multi-topology routing," in *Proceedings of DRCN 2005: Design of Reliable Communication Networks*, pp. 515-522, Ischia, October 2005.

Publikációk

[D] J. Farkas "Resilience and quality of service assurance methods in access and metro networks," *Ph.D. dissertation*, submitted to Budapest University of Technology and Economics, 2011.

Folyóirat cikkek

[J1] D Allan, **J. Farkas** and S. Mansfield, "Intelligent load balancing for shortest path bridging," *IEEE Communications Magazine*, 2011.

[J2] D. Allan, P. Ashwood-Smith, N. Bragg, **J. Farkas**, D. Fedyk, M. Ouellete, M. Seaman and P. Unbehagen, "Shortest path bridging: Efficient control of larger Ethernet networks," *IEEE Communications Magazine*, October 2010.

[J3] **J. Farkas**, A. Paradisi and C. Antal, "Low-cost survivable Ethernet architecture over fiber," *Journal of Optical Networking*, Vol. 5, Issue 5, pp. 398-409, April 2006.

[J4] Kumli T. és **Farkas J.**, "Közcélú kapcsolt távbeszélő hálózatok (PSTN) valós idejű szimulációja," *Híradástechnika*, Vol. XLIX, Budapest, November 1998.

[J5] B. P. Gerő, **J. Farkas**, S. Kini, P. Saltsidis and A. Takács, "Upgrading the metro Ethernet network," *submitted for review to IEEE Communications Magazine*

Konferencia cikkek

[C1] **J. Farkas** and Z. Arató, "Performance analysis of shortest path bridging control protocols," in *Proceedings of GlobeCom 2009: IEEE Global Communications Conference*, Honolulu, December 2009.

[C2] **J. Farkas** and R. Pallos, "Root controlled bridging: A scalable control protocol for shortest path bridging," in *Proceedings of Networks 2008: 13th International Telecommunications Network Strategy and Planning Symposium*, Budapest, September 2008.

[C3] **J. Farkas**, V.G. Oliviera, M.R. Salvador and G.C. Santos, "Automatic discovery of physical topology in heterogeneous multi-vendor Ethernet networks," in *Proceedings of ICC 2008: IEEE International Conference on Communications*, pp. 2055-2060, Beijing, May 2008.

[C4] **J. Farkas**, V.G. Oliviera, M.R. Salvador and G.C. Santos, "Automatic discovery of physical topology in Ethernet networks," in *Proceedings of AINA 2008: Advanced Information Networking and Applications*, pp. 848-854, Okinawa, March 2008.

[C5] R. Pallos, **J. Farkas**, I. Moldován and C. Lukovszki, "Performance of rapid spanning tree protocol in access and metro networks," in *Proceedings of AccessNets 2007: Second International Conference on Access Networks*, Ottawa, August 2007.

[C6] C. Lukovszki, I. Moldován, A. Kern, **J. Farkas**, W. Zhao and Z. Ghebretensaé, "Standard-based physical and active topology discovery in Ethernet-based aggregation networks," in *Proceedings of NOC 2007: 12th European Conference on Networks and Optical Communications*, Stockholm, June 2007.

[C7] **J. Farkas**, C. Antal, L. Westberg, A. Paradisi, T.R. Tronco and V.G. Oliviera, "Fast failure handling in Ethernet networks," in *Proceedings of ICC 2006: IEEE International Conference on Communications*, Vol. 2, pp. 841-846, Istanbul, June 2006.

[C8] **J. Farkas**, C. Antal, G. Tóth and L. Westberg, "Distributed resilient architecture for Ethernet networks," in *Proceedings of DRCN 2005: Design of Reliable Communication Networks*, pp. 515-522, Ischia, October 2005.

[C9] S. Rácz, T. Jakabfy, **J. Farkas** and C. Antal, "Connection admission control for flow level QoS in bufferless models," in *Proceedings of InfoCom 2005: 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 1273-1282, Miami, March 2005.

[C10] G. Mátéfi, **J. Farkas** and C. Antal, "Towards efficient call admission control in IP UTRAN," in *Proceedings of ITC 18: 18th International Teletraffic Congress*, pp. 238-253, Berlin, September 2003.

Szabadalmi bejelentések

[P1] **J. Farkas**, "Method and arrangement for multipoint service protection in Ethernet networks," International Patent Application, WO/2011/038750, 2011.

[P2] **J. Farkas**, "System, network management system and method for avoiding a count-to-infinity problem," International Patent Application, WO/2011/058450, 2011.

[P3] D. Jocha and **J. Farkas**, "Loss measurement for multicast data delivery," International Patent Application, WO/2011/003478, 2011.

[P4] **J. Farkas**, R. Pallos, G. Kapitány, S. Plósz and D. Horváth "Port table flushing in Ethernet networks," International Patent Application, WO/2010/086022, 2010.

[P5] **J. Farkas**, C. Antal and A. Takács "Multiple tree registration protocol," International Patent Application, WO/2010/007467, 2010.

[P6] **J. Farkas**, C. Antal and A. Takács "Method and apparatus for Ethernet protection with local re-routing," International Patent Application, WO/2009/115480, 2009.

[P7] **J. Farkas**, "Method and apparatus for link-state handshake for loop prevention," International Patent Application, WO/2009/112929, 2009.

[P8] **J. Farkas**, C. Antal, A. Takács and P. Saltsidis, "Ethernet spanning tree provision," International Patent Application, WO/2008/125144, 2008.

[P9] **J. Farkas**, C. Antal, A. Takács and P. Saltsidis, "Method and apparatus for network tree management," International Patent Application, WO/2008/087547, 2008.

[P10] **J. Farkas**, C. Antal, A. Takács and P. Saltsidis, "Method, bridge and computer network for calculating a spanning tree based on link state advertisements," International Patent Application, WO/2008/087543, 2008.

[P11] **J. Farkas**, V.G. Oliveira and M.R. Salvador, "Method of discovering physical topology of a telecommunications network," International Patent Application, WO/2008/076052, 2008.

[P12] **J. Farkas**, W. Zhao, "Method for fault localisation in multiple spanning tree based architectures," International Patent Application, WO/2008/095538, 2008.

[P13] **J. Farkas**, S. Nádas, C. Antal, S. Rácz, S. Malomsoky and U. Rosberg, "Dimensioning link capacity in a packet switched telecommunications network," International Patent Application, WO/2008/120077, 2008.

[P14] P. Lundh, C. Faronius, **J. Farkas**, S. Rácz and S. Nádas, "Enhanced flow control in a cellular telephony system," International Patent Application, WO/2008/066430, 2008.

[P15] **J. Farkas**, and G. Tóth, "Centralised calculation of minimum number of multiple spanning trees," International Patent Application, WO/2007/043919, 2007.

[P16] **J. Farkas**, C. Antal and L. Westberg, "Method and arrangement for failure handling in a network," International Patent Application, WO/2006/135282, 2006.

[P17] G. Mátéfi, **J. Farkas** and T. Éltető, "Method and device for audience monitoring on multicast capable networks," United States Patent Application, US 2006/0294259 A1, 2006.

[P18] G. Mátéfi, **J. Farkas** and C. Antal, "Connection admission control system and method for interpreting signalling messages and controlling traffic load in Internet protocol differentiated services networks," International Patent Application, WO/2005/022851, 2005.

Szabványosítási kontribúciók

[S1] **J. Farkas**, "Notes on IS-IS network convergence," Technical presentation, November 2010.
http://www.ieee802.org/1/files/public/docs2010/new-farkas-convergence-1110.pdf

[S2] Clause 28.9 of the IEEE Std. 802.1aq D3.2, "IEEE Draft Standard for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks - Amendment 9: Shortest Path Bridging," October 2010.

[S3] **J. Farkas**, "CFM in 802.1aq SPB," Technical presentation, July 2010.
http://www.ieee802.org/1/files/public/docs2008/aq-farkas-CFM-in-802.1aq-0908.pdf,
http://www.ieee802.org/1/files/public/docs2008/aq-farkas-proposal-for-CFM-in-SPB.pdf

[S4] Clauses 28.7 and 28.8 of the IEEE Std. 802.1aq D3.0, "IEEE Draft Standard for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks - Amendment 9: Shortest Path Bridging," June 2010.

[S5] Clause 27.7 of the IEEE Std. 802.1aq D2.0, "IEEE Draft Standard for Local and Metropolitan Area Networks: Virtual Bridged Local Area Networks - Amendment 9: Shortest Path Bridging," June 2009.

[S6] **J. Farkas**, "802.1aq: Link-state handshake for loop prevention," Technical presentation, March 2008.
http://www.ieee802.org/1/files/public/docs2008/aq-farkas-link-state-handshake-0308.pdf

[S7] **J. Farkas**, "802.1aq: link-state protocol and loop prevention," Technical presentation, November 2007.
http://www.ieee802.org/1/files/public/docs2007/aq-farkas-loop-prevention-1107-v02.pdf