Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Automatizálási és Alkalmazott Informatikai Tanszék

DESIGN AND PERFORMANCE EVALUATION OF CONTENT SHARING AND STREAMING TECHNIQUES IN DISTRIBUTED COMMUNICATION NETWORKS

TARTALOM MEGOSZTÁSI ÉS STREAMING TECHNOLÓGIÁK TERVEZÉSE ÉS TELJESÍTMÉNY VIZSGÁLATA ELOSZTOTT KOMMUNIKÁCIÓS HÁLÓZATOKBAN

Ph.D. tézisfüzet

Braun Patrik János

Konzulens: Dr. Ekler Péter
Témavezető: Prof. Frank H. P. Fitzek

Budapest
2020.

Ph.D. tézisfüzet

Braun Patrik János

Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Automatizálási és Alkalmazott Informatikai Tanszék

1117 Budapest, Magyar Tudósok körútja 2. QB-207.

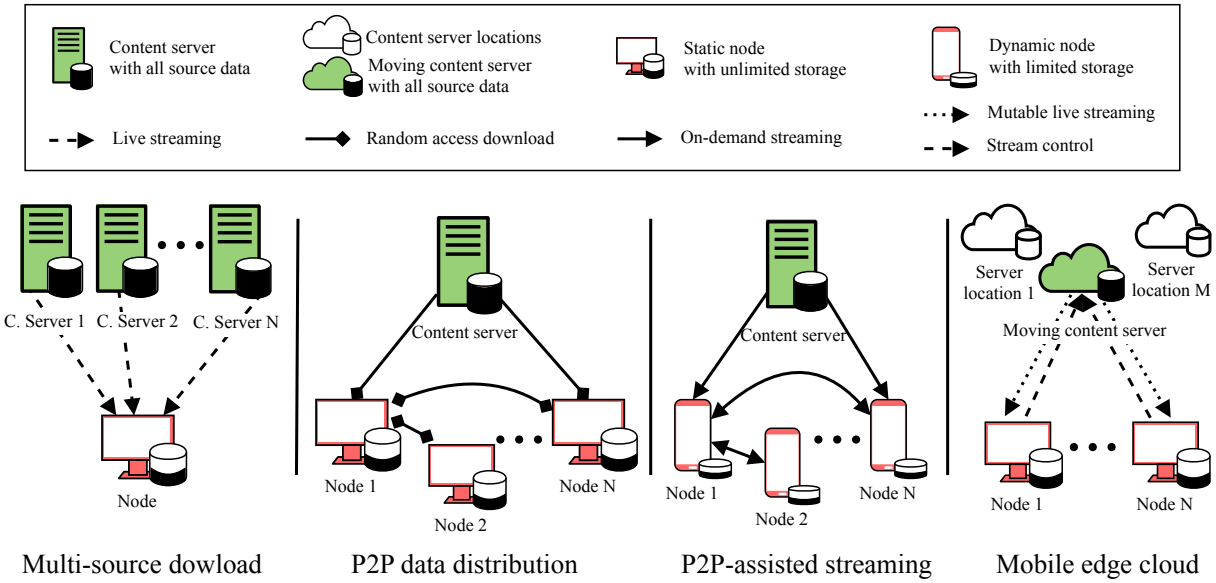
e-mail: braun.patrik@aut.bme.hu
tel: +36(1)4631668
fax: +36(1)4633478

Témavezető: Dr. Ekler Péter
Konzulens: Prof. Frank H. P. Fitzek

1. Bevezetés

A chip méretek zsugorodásával és a vezeték nélküli hálózatok fejlődésével, ma már több eszköz csatlakozózik hálózathoz, mint bármikor korábban. Ez az Ericsson bécslései is alátámasztják, amely szerint 8,3 milliárd mobil előfizetés várható 2024 végére [Eri18]. Ezen előfizetések szükségletei összetettek. Egyrészről a mobil előfizetések fele várhatóan Internet of Things (IoT) hálózatokban kerül felhasználásra. Az IoT hálózatok tervezésekor a kihívás nem a sávszélesség igényük, hanem a online csomópontok masszív mennyiségéből adódik. Másrészről a streaming szolgáltatások (pl.: YouTube, Netflix) népszerűségének folyamatos növekedésével a felhasználók már nemcsak otthon, hanem útközben is videókat töltenek le. Streaming szolgáltatások hatalmas hálózati terhelést hoznak létre. A 2018-as mobilos Internet forgalom 70%-t videó letöltés tette ki az Ericsson adatai szerint. Végül új alkalmazás típusok, mint például Vehicle-to-Vehicle (V2V) kommunikáció alacsony hálózati késleltetést igényelnek. A következő generáció mobil hálózat, az 5G célja, hogy megoldást nyújtson erre az igényre úgy, hogy 1 ms alatti garantált késleltetést célozza meg. Szubmilliszekundum késleltetést pusztán hardveres alapokon nem lehet elérni, mert egy csomag nem tud fénysebességnél gyorsabban utazni. Ezért okos szoftveres megoldások bevezetésére van szükség, amelyek csökkentik a szerver-kliens közti fizikai távolságot.

Munkám során ezeken kihívásokra javaslok megoldást elosztott hálózatokban network coding felhasználásával. Szerver-kliens hálózati topológiához képest, az elosztott hálózatoknak több előnyük is van. Egyrészről jobban skálázódnak, így több csomópontot tudnak kiszolgálni. Másrészről letöltéshez választhatnak a csomópontok közeli partner csomópontokat vagy szervereket, amely csökkenti a hálózati késleltetést. Végül network coding növeli a gyorsítótár hatékonyságát (cache hit ratio) és csökkenti a csomaghibákat.



1. ábra. Vizsgált hálózati topológiák áttekintése

Munkám során négy fő hálózati környezetre fókuszáltam: *Többforrásos letöltés* (Multi-source download), *Peer-to-Peer (P2P) adatelosztás*, *mobilos P2P-rel támogatott streamelés* és *hálózatperemi felhő* (mobile edge cloud), ahogy azt az 1. ábra is mutatja. A négy környezetre több kommunikációs protokollt és szoftver architektúrát terveztem. Valamint megvizsgáltam a network coding, pontosabban a Random Linear Network Coding (RLNC) [Ho+03] alkalmazásának a lehetőségét ezekben a környezetekben.

2. Alapvető definíciók

A munkám több csomópontból álló elosztott hálózatokra fókuszál, amelyet a következő definíciókkal határozok meg:

2.1. Definíció (Csomópont)**.** A csomópont a hálózat egy tárhellyel rendelkező entitása, amely képes más csomópontokhoz csatlakozni és tőlük adatot letölteni és felötleni.

Kapcsolódott csomópontok a forrásadatot továbbítják egymásnak:

2.2. Definíció (Forrásadat)**.** A forrás adat az eredeti, L csomagból álló adat, amelyet a csomópontok igyekeznek letölteni.

Négy fő csomópont-tulajdonságot különböztetek meg, amelyek kihatnak egy csomópont viselkedésére: Egy csomópont lehet *statikus* úgy, hogy soha nem hagyja el a hálózatot, vagy *dinamikus* ha elhagyja. Egy csomópontnak lehet *korlátozott* $A \leq L$, vagy *korlátlan* $A = L$ tárhelye. Egy csomópont begyűjtheti az L csomagot *véletlen eléréssel* (random access) vagy *streaming*en keresztül. Stremelés lehet *igény szerinti* (on-demand) vagy *fix* és *változtatható élőadás* (live streaming). Az utóbbi esetén a csomópontok módosíthatják a forrásadatot.

A további definíciók egyszerűsítésére, a következő fogalmakat vezetem be:

2.3. Definíció (Tartalomkiszolgáló)**.** Olyan statikus csomópont, ami az összes L forrás adattal rendelkezik.

2.4. Definíció (Mozgó tartalomkiszolgáló)**.** Olyan tartalomkiszolgáló, ami képes változtatni a helyét.

Az 1. ábra egy áttekintést ad arról a négy környezetről amelyekre a munkám során koncentráltam. Ezt a négy környezetet a következőképpen definiálom:

2.5. Definíció (Többforrásos hálózat)**.** Többforrásos hálózatok N tartalomkiszolgálót és egy további, korlátlan tárhellyel rendelkező statikus csomópontot tartalmaznak. A csomópont az összes tartalomkiszolgálóhoz csatlakozik egyszerre. A tartalomkiszolgálók és a csomópont között veszteséges előre irányú és veszteségmentes visszirányú kapcsolat van. A hálózat körbefordulási ideje (round trip time (RTT)) k időegységet tesz ki. A csomópont halmozott nyugtát (commutative acknowledgment) küld a kiszolgálóknak. A kiszolgálóknál mind az L csomag megvan. Ezt az L csomagot igyekszik a csomópont streameléssel letölteni, párhuzamosan az összes kiszolgálótól.

2.6. Definíció (Peer-to-Peer (P2P) adatelosztás)**.** P2P adatelosztás esetén a hálózat egy tartalomkiszolgálót és N statikus, korlátlan tárhellyel rendelkező csomópontot tartalmaz. A tartalomkiszolgáló átlagos csomópontként viselkedik, ezért a többi csomópont nem tudja, hogy melyik csomópont a tartalomkiszolgáló mielőtt csatlakozna hozzá. A tartalomkiszolgálónál mind az L csomag megvan, amit a többi csomópont *igény szerinti* letöltéssel igyekszik megszerezni.

2.7. Definíció (Mobilos P2P-rel támogatott streamelés)**.** A hálózat egy tartalomkiszolgálót és N dinamikus, korlátozott, A méretű tárhellyel rendelkező csomópontot tartalmaz. A kiszolgálóknál mind az L csomag megvan, amit a többi csomópont streaming alapú letöltéssel igyekszik megszerezni. A csomópontoknak folyamatos kapcsolatuk van a kiszolgálóhoz és további más csomóponthoz is képesek csatlakozni.

2.8. Definíció (Hálózatperemi felhő)**.** A hálózatperemi felhők (mobile edge cloud) N statikus csomópontokból, egy tartalomkiszolgálóból és több tartalomkiszolgáló helyekből állnak. A kiszolgáló az összes L csomagot birtokolja és képes a helyek között mozogni. A csomópontok *változtatható előadás* alapú letöltéssel gyűjtik be a csomagot a kiszolgálóktól. Továbbá a csomópontoktól a kiszolgálóhoz egy vezérlő csatorna is ki van építve, amelyen keresztül a csomópontok a streamelt tartalmat tudják módosítani. A módosítás az összes csomópontra kihat.

3. Elméleti eredmények

Tézis I: Többforrásos letöltési megoldások vizsgálata

Maximalizáltam a többforrásos protokollok áteresztő képességét úgy, hogy terveztem egy analitikus keretrendszert és megvizsgáltam különböző többforrásos protokollokat, gyengén koordinált hálózatban. Hat különböző protokollt terveztem, amelyek goodputját (hasznos áteresztőképességét) összehasonlítottam. A tervezett protokollok közé tartozik kódolatlan, RLNC-vel kódolt ráta nélküli és csúszóablakos protokoll is. Megmutattam, hogy az RLNC képes növelni a goodputot, míg a csúszóablakos RLNC eléri az elméleti maximum goodputot. Továbbá megmutattam, hogy az RLNC lényegesen növeli a hálózat áteresztőképességet valódi rendszerekben. Ehhez egy saját fejlesztésű rendszert használtam, amely YouTube videókat töltött le több forrásból. A rendszeren tizenegy hónapon keresztül végeztem méréseket, amely idő alatt összesen 1300000 naplófájlt gyűjtöttem be és elemeztem ki.

A tézishez tartozó publikációim: [1], [4], [5], [6], [7]
A tézishez tartozó szabadalmaim: [21]
A tézishez tartozó előadások és bemutatók: [22]

Altézis I.1: SR-ARQ alapú modell készítése többforrásos letöltéshez

A többforrásos letöltés megvizsgálásra egy módosított SR-ARQ alapú modellt terveztem, amely támogatja a veszteséges előre irányú és veszteség mentes visszirányú csatornákat. Az előre irányú csatorna állapotát rejtett Markov modellel modelleztem, hogy a megoldásomat általánosan lehessen alkalmazni. A csomópont státuszát szabadság fokokkal jellemzem (Degrees of Freedom (DoF)), hogy az analízis alkalmazható legyen mind kódolatlan, mind kódolt protokollokra.

Egy időréses modellt terveztem, ahol minden időrésben egy forrás (tartalomkiszolgáló) küld egy csomagot, amely vagy kézbesítésre kerül vagy elveszik a csatornán. Az i. csatornán történő csomagküldés eredménye egy Bernoulli véletlen változó, amelyet $\mathcal{X}^{(i)} = \{0, 1\}$ jelölök. A csatorna állapotát egy több állapotú Markov lánccal modellezem $S^{(i)} = \{1, \ldots, K^{(i)}\}$ aminek az átmeneti mátrixát $\boldsymbol{P}_{(i)}$ jelöli. Minden $S_t^{(i)} = j, j \in S^{(i)}$ állapothoz más hiba valószínűség tartozik: $\epsilon_j^{(i)}$. A csatorna hibák valószínűségének halmazát a $\epsilon^{(\mathbf{i})} = \{\epsilon_1^{(i)}, \ldots, \epsilon_{K^{(i)}}^{(i)}\}$ jelölöm. Az $S_t^{(i)}$-vel jelölt Markov folyamatot az $X_t^{(i)}$ rejtett Markov folyamat vezérli, amit a $\{S^{(i)}, \mathcal{X}^{(i)}, \boldsymbol{P}_{(i)}, \epsilon^{(\mathbf{i})}\}$ jellemez. Továbbá a csomagvesztés

és fogadás valószínűsége a $\boldsymbol{P}_{\mathrm{L},(i)} = \boldsymbol{P}_{(i)} \cdot \mathrm{diag}\{\epsilon^{(\mathbf{i})}\}$ és a $\boldsymbol{P}_{\mathrm{R},(i)} = \boldsymbol{P}_{(i)} \cdot \mathrm{diag}\{1-\epsilon^{(\mathbf{i})}\}$ mátrixok jellemzik.

A csomópont halmozott nyugtát küld minden forrásnak egy veszteségmentes csatornán. Ezért a források minden időpillanatban egy ACK-t vagy NACK-t kapnak, attól függően hogy a csomag elveszett vagy megérkezett. A nyugtákon kívül más információt nem tudnak a források egymásról, így a csomagütemezés független az egyes forrásoknál. Így ha újabb forrást adunk a rendszerhez, az nem befolyásolja a korábbi forrásokat. Ezért a rendszer elkerüli a csellengő problémát (straggler problem).

Altézis I.2: Általános analitikai megoldás

Különböző többforrásos protokollok megvizsgáláshoz alkottam egy analitikus keretrendszert. A N forrású hálózatok goodputjának kiszámításához egy matrix signal-flow gráfot és generátorfüggvényeket használtam. Megmutattam, hogy elég az utolsó k időbélyeg megvizsgálása a goodput kiszámításához. A keretrendszeremet felhasználva, összehasonlítottam különböző többforrásos protokollok teljesítményét és maximalizáltam a többforrásos hálózatok goodputját.

A sikeres és hasznos csomagtovábbítás valószínűségének kiszámítására egy matrix signal-flow gráfot használtam [AN07], amelyet felhasználva kifejeztem a csomagátviteli idő generátorfüggvényét (PGF):

i. csatorna csomagátviteli idejének generátorfüggvénye

$$\phi_{\tau(i)}(z) = \frac{1}{1-\epsilon_{\mathrm{F}}^{(i)}} \pi_{(i)} \mathcal{P}_{\mathrm{U},(i)} (\mathbf{I} - z\mathcal{P}_{\mathrm{F},(i)})^{-1} z \mathcal{P}_{\mathrm{U},(i)} \mathbf{1}, \tag{1}$$

ahol $\epsilon_{\mathrm{F}}^{(i)}$ a csomag hiba (csomagvesztés, vagy nem DoF-t növelő csomag) rátája, $\pi_{(i)}$ a $\boldsymbol{P}_{(i)}$ mátrix stacionárius eloszlás vektora. $\mathcal{P}_{\mathrm{U},(i)}$ és a $\mathcal{P}_{\mathrm{F},(i)}$ a csomaghasznosságnak és csomaghiba valószínűsége. $\mathbf{I}$ az egység mártix, míg a $\mathbf{1}$ egy csupa egyes oszlopvektor.

Az i. forrás átlagos csomagátviteli idejét, a $\overline{\tau}_{(i)}$-t úgy lehet megkapni, ha a PGF deriváltját a $\phi_{\tau(i)}(z)$-t a $z = 1$ helyen értékeljük ki. Az i. csatorna goodputja, a $\eta_{(i)}$ az átlagos csomagátviteli idő reciprokja: $\eta_{(i)} = 1/\overline{\tau}_{(i)}$.

Hasznos csomag $\mathcal{P}_{\mathbf{U},(i)}$ és csomaghiba $\mathcal{P}_{\mathbf{F},(i)}$ valószínűsége a t. időpontban

$$\begin{aligned} \mathcal{P}_{\mathrm{U}}(t) &= \sum_{v \in \{0,1\}^t} \mathcal{P}_{\mathrm{pU}}(\mathbf{v}) \mathcal{P}_{\mathrm{past}}(v_1 \dots v_{t-1}) \boldsymbol{P}_{\mathrm{R},(s(t))} v_t \\ \mathcal{P}_{\mathrm{F}}(t) &= \sum_{v \in \{0,1\}^t} \mathcal{P}_{\mathrm{pD}}(\mathbf{v}) \mathcal{P}_{\mathrm{past}}(v_1 \dots v_{t-1}) \boldsymbol{P}_{\mathrm{R},(s(t))} v_t + (1-v_t) \boldsymbol{P}_{\mathrm{L},(s(t))}, \end{aligned} \tag{2}$$

ahol a $\mathbf{v} = [v_1 \dots v_t]$ vektor a lehetséges események kimenetelei a $[1, t]$ idő intervallumban. A $\mathcal{P}_{\mathrm{pU}}(\mathbf{v})$ és a $\mathcal{P}_{\mathrm{pD}}(\mathbf{v})$ a valószínűségei annak, hogy egy csomag várhatóan hasznos vagy nem a t. időpontban.

$\mathcal{P}_{\mathbf{past}}(v_1 \dots v_{t-1})$ kifejezése a t. időpontban

$$\mathcal{P}_{\mathrm{past}}(v_1 \dots v_{t-1}) = \prod_{i=1}^{N} \prod_{\substack{l=i \\ (l-i \bmod N)=0}}^{t-1} \pi_{(i)} \boldsymbol{P}_{\mathrm{R},(i)}^{v_l} \boldsymbol{P}_{\mathrm{L},(i)}^{|1-v_l|} \mathbf{1}. \tag{3}$$

Várhatóan hasznos $\mathcal{P}_{\mathrm{pU}}(\mathbf{v})$ és várhatóan duplikátum $\mathcal{P}_{\mathrm{pD}}(\mathbf{v})$ csomag valószínűsége

$$\mathcal{P}_{\mathrm{pU}}(\mathbf{v}) = \sum_{u=0}^{k-\frac{k}{N}} \sum_{d=0}^{\frac{k}{N}} \sum_{m=0}^{\min(d,u)} \sum_{\substack{\sum a_j = u \\ \sum b_i = d \\ \sum c_i = m,\, c_i \leq b_i}} \mathcal{P}_{\mathrm{paired}}(\mathbf{a},\mathbf{b},\mathbf{c})\, \mathcal{P}_{\mathrm{sU}}(\mathbf{a},\mathbf{b},\mathbf{c}) \mathcal{P}_{\mathrm{outcome}}(\mathbf{v},\mathbf{a},\mathbf{b}) \tag{4}$$

ahol $\mathcal{P}_{\mathrm{paired}}(\mathbf{a},\mathbf{b},\mathbf{c})$ annak a valószínűsége, hogy az $\mathbf{a},\mathbf{b}$ és $\mathbf{c}$ vektorok az elmúlt k időegység eseményvektorai, míg $\mathcal{P}_{\mathrm{outcome}}(\mathbf{v},\mathbf{a},\mathbf{b})$ annak a valószínűsége, hogy a $\mathbf{a},\mathbf{b}$ és $\mathbf{c}$ vektorok események, feltéve $\mathbf{v}$.

A $\mathcal{P}_{\mathrm{sU}}(\mathbf{a},\mathbf{b},\mathbf{c})$ és a $\mathcal{P}_{\mathrm{sD}}(\mathbf{a},\mathbf{b},\mathbf{c})$ annak a valószínűsége, hogy a forrás hasznos vagy duplikált csomagot ütemez be küldésre. Ez a választott ütemezési stratégiától függ. $\mathcal{P}_{\mathrm{pD}}(\mathbf{v})$ a $\mathcal{P}_{\mathrm{pU}}(\mathbf{v})$-hoz hasonlóan ki lehet fejezni. Ehhez a $\mathcal{P}_{\mathrm{sD}}(\mathbf{a},\mathbf{b},\mathbf{c})$-t kell használni a $\mathcal{P}_{\mathrm{sU}}(\mathbf{a},\mathbf{b},\mathbf{c})$ helyett.

A matrix-flow gráf lapaú megközelítésem csak akkor használható a goodput kifejezésére, ha $\lim_{t\to\infty} \mathcal{P}_{\mathrm{U}}(t)$ és a $\lim_{t\to\infty} \mathcal{P}_{\mathrm{F}}(t)$ határértékek léteznek.

Altézis I.3: Többforrásos protokollok tervezése

Alsó és felső korlátot adtam többforrásos protokollokhoz úgy, hogy terveztem egy elegendő orákulum és egy kódolatlan szekvenciális protokollt. Hogy maximalizáljam a hálózatok goodputját további négy többforrásos protokollt terveztem, többek között RLNC alapú protokollokat is. Korlátoztam a csomag késletetetést úgy, hogy két feltöltési ablak mechanizmust javasoltam a forrás oldalon. Megmutattam, hogy a ráta nélküli RLNC megoldások javítják a hálózat goodputját, míg a csúszóablakos RLNC eléri az elméleti maximum teljesítményt is.

Két feltöltési ablakmegoldás terveztem többforrású adat továbbításra:

Ritka mozgó ablak w csomagot tartalmaz. Amint egy csomag sikeresen továbbításra került az ablakból, azt eltávolítja a kiszolgáló és az L forrásadat következő csomagját adja hozzá az ablakhoz.

Szigorú mozgó ablak a $\mathcal{W}^{(i)}(t)$ halmazzal írható le:

$$\begin{aligned} \mathcal{W}^{(i)}(t) &= \{l \in \mathcal{L} \mid l_{\mathrm{dMin}} < l \leq l_{\mathrm{dMin}} + w\} \\ \mathcal{L}^{(i)}_{\mathrm{down}}(t) &= \{l \in \mathcal{L} \mid l.\text{ csomag a } t.\text{ időpontig megérkezett a klienshez és az nyugtázta is.}\}, \end{aligned} \tag{5}$$

ahol $\mathcal{L} = \{1,\ldots,L\}$ a forrásadat halmaza. A $l_{\mathrm{dMin}} = \min(\mathcal{L} \backslash \mathcal{L}^{(i)}_{\mathrm{down}}(t))$ olyan legkisebb sorszámmal rendelkező csomag, amely nem került nyugtázásra a $t.$ időegységig.

hat különböző többforrásos protokollt terveztem ehhez a két ablakos megoldáshoz:

3.1. Protokoll (Elegendő orákulum)**.** Csak olyan csomagokat küld a hálózaton, amik garantáltan növelik a DoF-t a fogadó csomópontnál.

A csomagok elveszhetnek a csatornánk, ezért beszélünk csak *elegendő* orákulumról. Az orákulum tökéletes ütemezést nyújt az adott hálózati feltételek mellett.

$\mathcal{P}_{\mathbf{sU}}(\mathbf{a},\mathbf{b},\mathbf{c})$ **kiszámítása elegendő orákulum esetén**

$$\mathcal{P}_{\mathrm{sU}}(\mathbf{a},\mathbf{b},\mathbf{c}) = 1 - \mathcal{P}_{\mathrm{sD}}(\mathbf{a},\mathbf{b},\mathbf{c}) = 1. \tag{6}$$

Az eredmény független az alkalmazott feltöltési ablaktól.

3.2. Protokoll (Kódolatlan szekvenciális)**.** A csomagokat sorfolytonosan, first-in first-out (FIFO) jelleggel ütemezi.

$\mathcal{P}_{\mathbf{sU}}(\mathbf{a},\mathbf{b},\mathbf{c})$ **kiszámítása kódolatlan szekvenciális protokoll esetén**

$$\mathcal{P}_{\mathrm{sU}}(\mathbf{a},\mathbf{b},\mathbf{c}) = 1 - \mathcal{P}_{\mathrm{sD}}(\mathbf{a},\mathbf{b},\mathbf{c}) = \left|(1 - \min(\sum_{j=k-s(t)-1}^{k} a_j, 1))\right|. \tag{7}$$

Az eredmény független az alkalmazott feltöltési ablaktól.

3.3. Protokoll (Kódolatlan véletlen)**.** A protokoll véletlenszerűen ütemez egy csomagot az ablakból.

$\mathcal{P}_{\mathbf{sU}}(\mathbf{a},\mathbf{b},\mathbf{c})$ **kódolatlan véletlen protokoll esetén** ritka mozgó ablakkal:

$$\mathcal{P}_{\mathrm{sU}}(\mathbf{a},\mathbf{b},\mathbf{c}) = 1 - \mathcal{P}_{\mathrm{sD}}(\mathbf{a},\mathbf{b},\mathbf{c}) = \frac{w - (\frac{k}{N} - (d-m)) - (u-m)}{w - (\frac{k}{N} - (d-m))}, \tag{8}$$

ahol $u = \sum_{j=1}^{k} a_j$, $d = \sum_{i=1}^{k} b_i$, $m = \sum_{i=1}^{k} c_i$.

$\mathcal{P}_{\mathbf{sU}}(\mathbf{a},\mathbf{b},\mathbf{c})$ **kódolatlan véletlen protokoll esetén** szigorú mozgó ablakkal:

$$\begin{aligned} \mathcal{P}_{\mathrm{sU}}(\mathbf{a},\mathbf{b},\mathbf{c}) &= \sum_{w_a=w_{min}}^{w} P(\mathcal{W}=w_a)\frac{w_a - (\frac{k}{N} - (d-m)) - (u-m)}{\sum_{w'_a=w_{min}}^{w} P(\mathcal{W}=w'_a)(w_a - (\frac{k}{N} - (d-m)))} \\ w_{min} &= max\left(1, u, (\frac{k}{N} - (d-m)) + (u-m)\right) \end{aligned} \tag{9}$$

3.4. Protokoll (Ráta nélküli RLNC-vel kódolt protokoll, véletlenszerű generáció választással)**.** A protokoll g méretű generációkba csoportosítja a csomagokat amire RLNC-t alkalmaz. A kódolt csomagküldéshez véletlenszerűen választja ki a generációt.

3.5. Protokoll (Ráta nélküli RLNC-vel kódolt protokoll, legritkább generáció választással)**.** A protokoll g méretű generációkba csoportosítja a csomagokat amire RLNC-t alkalmaz. A kódolt csomagküldéshez azt a generációt használja, amelyiknél a legkisebb a DoF a kliens oldalon.

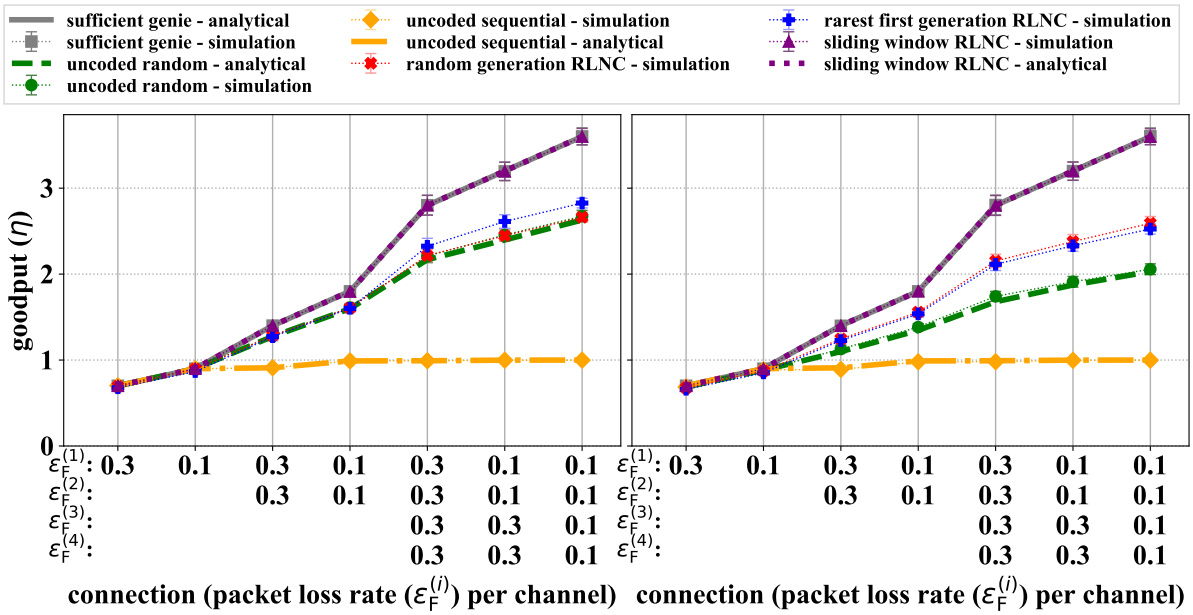
3.6. Protokoll (Csúszóablakos RLNC protokoll)**.** A protokoll az ablakban lévő összes csomagot egyszerre kódolja RLNC-vel. Ezek a kódolt csomagok utaznak a hálózaton.

$\mathcal{P}_{\mathbf{sU}}(\mathbf{a},\mathbf{b},\mathbf{c})$ **csúszóablakos RLNC protokoll esetén**

$$\mathcal{P}_{\mathrm{sU}}(\mathbf{a},\mathbf{b},\mathbf{c}) = 1 - \mathcal{P}_{\mathrm{sD}}(\mathbf{a},\mathbf{b},\mathbf{c}) = \begin{cases} 1 & \text{ha } (t \bmod k) < w \\ 0 & \text{egyébként} \end{cases}. \tag{10}$$

Az eredmény független az alkalmazott feltöltési ablaktól.

Ahogy az 2. ábrán is látható, a ráta nélküli RLNC növeli a goodputot, míg a csúszóablakos RLNC eléri elméleti maximum goodputot. Munkám során továbbá azt is megmutattam, hogy a goodput jelentősen csökken ha a nyugták is elveszhetnek a csatornán. Végül pedig beláttam, hogy a többforrásos protokolljaim elkerülik a csellengő problémát.



2. ábra. Goodput ritka (bal) és szigorú (jobb) mozgó ablak esetén, $N \in \{1, 2, 4\}$ forrással, $\kappa_c = 3$ RTT-vel, $r = 0.3$ burst rátával, $w = 24$ ablak és generáció size $g = 12$ mérettel.

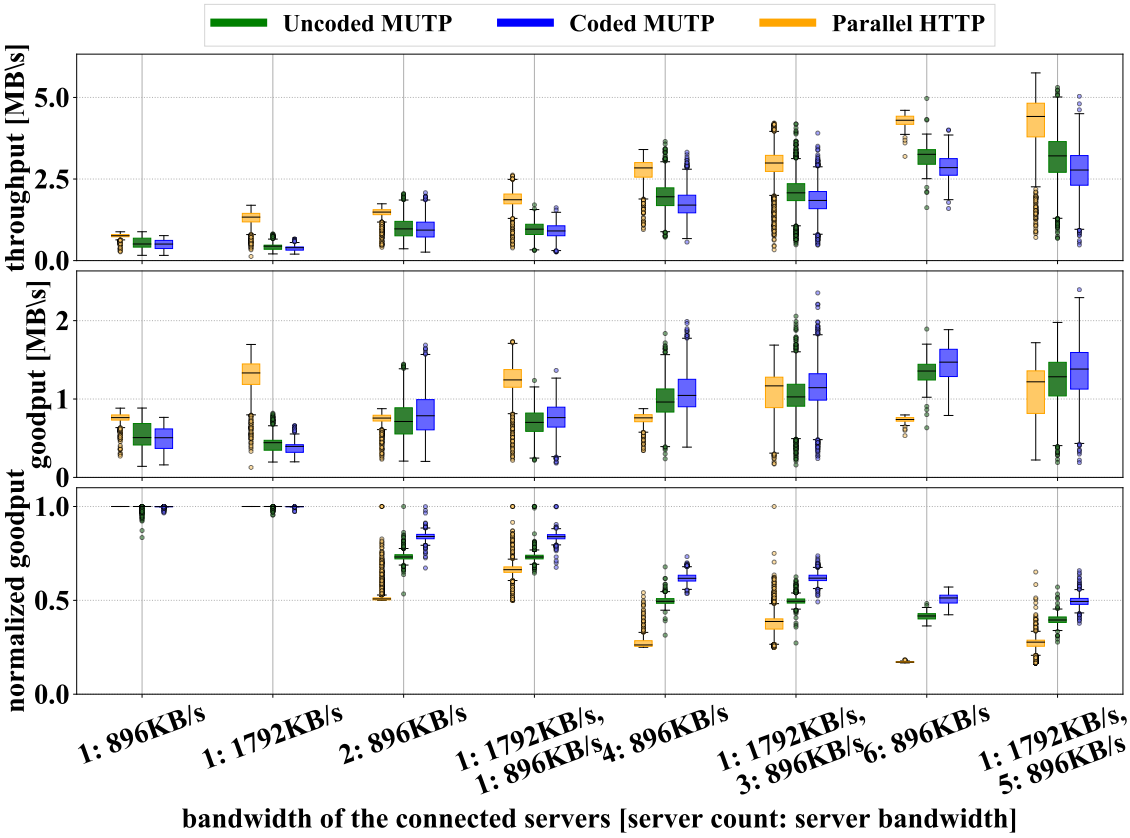
Altézis I.4: Többforrásos rendszer tervezése

Megmutattam a gyakorlati alkalmazhatóságát a többforrásos letöltésnek úgy, hogy terveztem és implementáltam egy rendszert, amely többforrásos protokollokat használ adatletöltésre. Megterveztem a kódolatlan és kódolt MUTP-t, amelyek a kódolatlan véletlenszerű és a ráta nélküli RLNC protokollokat implementálják. Tizenegy hónapon keresztül végzett mérések eredményein keresztül megmutattam, hogy az MUTP protokollok teljesítménye akár háromszorosan is felülmúlja HTTP teljesítményét.

3.7. Protokoll (MUlti-source Transmission Protocol (MUTP))**.** MUTP az eredeti adatot L darab 1100 byte-os csomaggá szeleteli. A források egy *in-window*, és egy *in-transit* listát tartanak nyilván, míg a nyelő csomópont egy *received* csomaglistát. A csomópont halmozott nyugtát küld a forrásoknak a *received* listája alapján. A nyugták és az *in-transit* lista alapján a források létrehozzák a *sendable* listát, amely küldhető csomagokat tartalmazza. A *sendable* listájukból a források véletlenszerűen választanak csomagot, amelyet elküldenek a nyelőcsomópontnak.

3.8. Protokoll (Coded MUTP)**.** Kódolt MUTP az L forráscsomagot g méretű generációkba csoportosítja. A generációkban lévő csomagokat RLNC-vel kódolja. Kódolt MUTP a legritkább generáció alapú ütemezést használja a megfelelő generáció kiválasztásához.

Terveztem és implementáltam egy rendszert, amely elkapja a YouTube videó letöltéseket és több proxy szerveren keresztül tölti azokat le. A letöltéshez az MUTP protokolljaimat és párhuzamos HTTP kapcsolatokat használ. Egy több mint tizenegy hónapon át tartó mérési kampányt futtattam, amely megmutatta, hogy négy vagy több forrás esetén az MUTP protokolljaim teljesítménye akár háromszorosan felülmúlja a párhuzamos HTTP kapcsolatok teljesítményét. Az eredményeim a 3. ábrán láthatóak.



3. ábra. 1-2MB-os adatletöltése $N \in \{1, 2, 4, 6\}$ forrásból, 896 KB/s and 1 792 KB/s sebességű forrásokkal, $w = 240$ ablak és $g = 24$ generáció mérettel.

Tézis II: Peer-to-Peer adat elosztás

Megmutattam, hogy RLNC képes a hálózati összefüggőséget növelni és az adatelosztási időt csökkenteni P2P hálózatok kezdetei szakaszában. Az eredményeim igazolására terveztem és implementáltam egy BitTorrent kiegészítést, amely lehetővé teszi RLNC-val kódolt csomagok küldését úgy, hogy megtartja a kompatibilitást az eredeti BitTorrent protokollal. Több metrikát is javasoltam a P2P hálózatok összehasonlítására és több algoritmust terveztem az RLNC alapú P2P hálózatokban keletkező lineárisan összefüggő csomagok kezelésére.

A tézishez tartozó publikációim: [2], [8], [9], [10]

Ebben a tézisben a hálózat kezdeti állapotára fókuszáltam, amikor csak a tartalomkiszolgálónál van meg az összes L csomag.

Altézis II.1: Metrikák ajánlása P2P hálózatokhoz

A P2P hálózatok legfőbb karakterisztikájának a hálózati összefüggőséget definiáltam. A hálózati összefüggőség és az adatelosztási idő objektív összehasonlítására definiáltam a hálózat egészsége és hálózat teljesítménye metrikákat. Metrikáimmal megmutattam, hogy az RLNC alapú P2P protokollok növelik a hálózat áteresztőképességét a hálózat kezdeti szakaszában.

3.9. Definíció (Hálózati összefüggűség)**.** Hálózati összefüggűség $i \leq N$, ha bármelyik i darab csomópont eltávolítása után mind az L csomag legalább egyszer megtalálható még a hálózatban.

Hálózati összefüggőség mérésére az aggregált $\mathbf{B}$ csomag vektort kell megvizsgálni: $\mathbf{b_i} = \{p_{i1}, p_{i2} \dots p_{iL}\}$ reprezentálja a letöltött cosmagokat az $i.$ csomópontnál, ahol $p_{i,l} = 1$ ha az $l.$ csomagot már letöltötte a csomópont, egyébként 0. Ekkor:

$$\mathbf{B} = \{B_1, B_2, \cdots, B_N\} = \sum_{i=1}^{N} \mathbf{b_i} \tag{11}$$

$\mathbf{B}$ vektort felhasználva a hálózat egészsége és teljesítménye a következőképpen fejezhető ki:

3.10. Definíció (H_k, a hálózat egészsége a $k.$ időegységben)**.**

$$\begin{aligned} p_{\mathrm{c}} &= |\{l \in \mathbf{B} \,|\, l > p_{\min}\}| \\ H_k &= p_{\min} + \frac{p_{\mathrm{c}}}{L}, \end{aligned} \tag{12}$$

ahol $p_{\min} = \min(\mathbf{B})$.

3.11. Definíció (E, a hálózat teljesítménye)**.**

$$E = \sum_{k=0}^{k_{\mathrm{end}}} H_k, \tag{13}$$

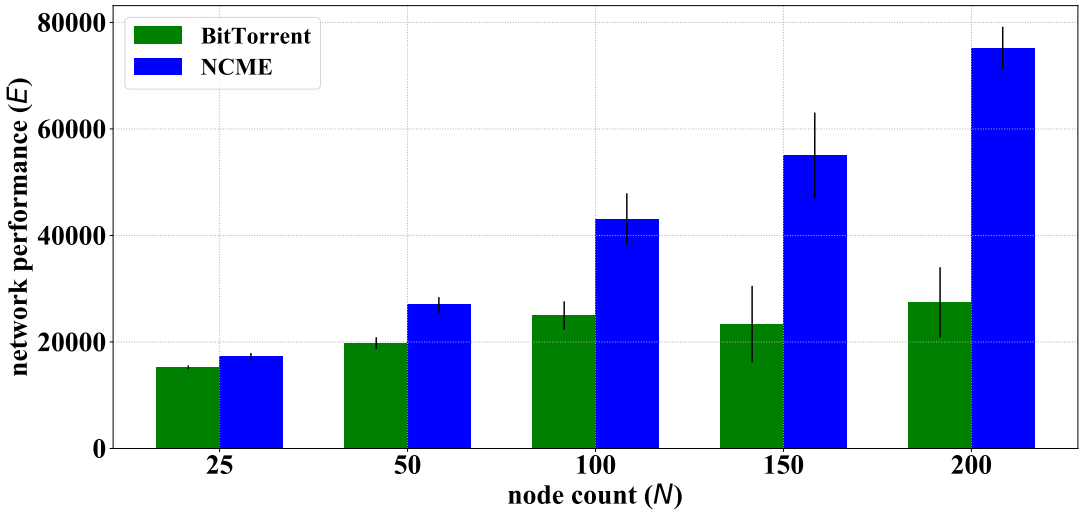
ahol k_{end} az az időegység amikor az összes csomópont begyűjtötte az összes L csomagot.

Altézis II.2: RLNC alapú BitTorrent protokollkiegészítés tervezése

Megmutatta az RLNC alapú protokollok P2P hálózatra gyakorolt potenciális hasznát úgy, hogy terveztem és implementáltam az Network Coding Messaging Extension (NCME) BitTorrent protokoll kiegészítést. Az eredményeim validálásához több mérést is végeztem, ahol összehasonlítottam az NCME és az alap BitTorrent protokoll teljesítményét. Megmutattam, hogy az RLNC alapú protokollom akár 20%-os teljesítmény növekedést is képes elérni.

3.12. Protokoll (Network Coding Messaging Extension (NCME))**.** NCME az L csomagból álló forrásadatot g méretű generációkba csoportosítja és RLNC-vel kódolja. Ezek a kódolt cosmagok utaznak a hálózaton. Miután egy csomópont g lineárisan független, egy generációhoz tartozó csomagot megkapott, azokat képes dekódolni és továbbosztani más csomópontoknak, beleértve az NCME-t nem implementáló csomópontokat. NCME veszteségmentes kapcsolat felett működik. Az üzenetek felépítése megegyezik a BitTorrent üzenetek felépítésével, de csomagazonosítók helyett generáció azonosítókat tartalmaznak.

Kiegészítettem a BitTorrent protokollt implementációt NCME-vel és több mérést végeztem emulált hálózaton, különböző paraméterekre fókuszálva, mint például csomópont (N) és csomag (L) szám. Megmutattam, hogy az NCME akár 20%-kal felülmúlja az alap BitTorrent teljesítményét. Ez a teljesítménynövekedés magasabb több csomópont vagy csomag esetén ahogy az 4. ábra is szemlélteti.



4. ábra. Csomópontok számának hatása a hálózat teljesítményére.

Altézis II.3: Algoritmusok tervezése lineárisan összefüggő csomagok kezeléséhez

Azonosítottam azokat a tipikus P2P hálózati mintákat, amelyek növelhetik a lineárisan összefüggő csomagok keletkezésének esélyét. A problémára megoldására három algoritmust terveztem és megmutattam, hogy az algoritmusaim képesek lényegesen csökkenteni a lineárisan összefüggő csomagok számát.

Terveztem három algoritmust a lineárisan összefüggő csomagok kezelésére:

3.1. Algoritmus (Lineárisan összefüggő csomagok figyelmen kívül hagyása)**.** *Ha egy csomópont kap egy lineárisan összefüggő csomagot, azt figyelmen kívül hagyja és folytatja a letöltést.*

3.2. Algoritmus (Futásidejű szűrés)**.** *Ha az A csomópont egy lineárisan összefüggő csomagot kap B partnerének i. generációjából, akkor A nem kér több csomagot onnan, amíg B nem jelzi, hogy az i. generációjához újabb csomag érkezett.*

3.3. Algoritmus (Szonda alapú szűrés)**.** *Mielőtt az A csomópont egy csomagot kérne B i. generációjából, egy szonda üzenetet küld, amelyre válaszul B csak az együtthatóvektort küldi vissza A-nak. A a vektor alapján eldönti, hogy B tud-e lineárisan független csomagot generálni. Ha igen, akkor A kér egy adatcsomagot B i. generációjából.*

NCME-vel kiegészített BitTorrent implementációmon több mérést is végeztem, hogy összehasonlítsam a három algoritmus képességét. Megmutattam, hogy a *lineárisan összefüggő csomagok figyelmen kívül hagyása* hatalmas plusz terhelést tesz a hálózatra. Bizonyos esetekben a *futásidejű szűrés* küldi az adatot a leggyorsabban, de *szonda alapú szűrés* használja a legkevesebb sávszélességet. Ezért amennyiben a sávszélesség egy kritikus erőforrás, a *szonda alapú szűrés* alkalmazása javasolt.

Tézis III: P2P-rel támogatott streamelés

Csökkentettem a szerverterhelést a P2P-rel támogatott streamelő rendszerekben úgy, hogy terveztem egy analitikus keretrendszer és több gyakorlati megoldást is javasol-

tam. Kiértékeltem a szerverterhelést különböző hálózati környezetben. Megmutattam az RLNC terheléscsökkentő hatását úgy, hogy az RLNC alapú megoldásommal együtt négy gyorsítótárazási stratégiát javasoltam limitált erőforrással rendelkező csomópontok számára. Megmutattam a P2P-vel támogatott streamelés gyakorlati hasznát úgy, hogy terveztem és implementáltam egy webes rendszert, amely véletlenszerű és RLNC-vel támogatott kliensoldali gyorsítótározást használ. Méréseken keresztül megmutattam, hogy a véletlenszerűhöz képest a RLNC-vel támogatott gyorsítótárazásnak 50%-al kevesebb csomag is elegendő a nullához közeli szerverterhelés eléréshez.

A tézishez tartozó publikációim: [3], [11], [12], [13],[14], [15], [16],[17]
A tézishez tartozó előadások és bemutatók: [23], [24], [25]

Ez a tézis a szerverterhelés minimalitására fókuszál, amelyet a következőképpen definiálok:

3.13. Definíció (Szerverterhelés)**.** A szerverterhelés az a minimális és szükséges szerver feltöltési ráta, amely elegendő miden csomópont kiszolgálására úgy, hogy azok streamelésében nincs megszakítás.

Altézis III.1: P2P-rel támogatott streamelés hálózati modellje

Formalizáltam a tézis problémáját úgy, hogy egy időosztásos modellt terveztem, amely mobilos P2P-rel támogatott streamelés megkötéseit is magába foglalja. A modell továbbá magában foglalja csomópontok életciklusát, kommunikációs kapcsolatok létrejöttét, letöltések ütemezését és a csomópontok korlátozott tárhelyét.

P2P-rel támogatott streamelő hálózatok N_k csomópontot tartalmaznak a k. időpontban, akik Poisson folyamat alapján csatlakoznak a hálózathoz és skew-normal eloszlás szerint hagyják el a hálózatot.

A csomópontoknak egy friss listájuk van más, hálózatban lévő csomópontokról. A k. időpontban, a csomópontok az elérhető $N_k - 1$ csomópontok egy maximálisan C elemű részhalmazához kapcsolódnak. Az i. csomópont aktív kapcsolatait a k. időpontban a $\mathbf{c}^{(i)}(k)$ vektorral jelölöm:

$$\mathbf{c}^{(i)}(k) \in \{0,1\}^{N_k},\ i = 1, \ldots, N_k,\ w(\mathbf{c}^{(i)}(k)) \leq C, \tag{14}$$

ahol $w(\mathbf{c}^{(i)}(k))$ az aktív kapcsolatok számát jelöli.

A csomópontok szekvenciálisan töltik le a L csomagot. Ehhez egy letöltési ablakot definiálok:

3.14. Definíció (Letöltési ablak)**.** A $\mathcal{W}^{(i)}(k)$ azon csomagok halmaza amit az i. csomópont a k. időpontban le szeretne tölteni:

$$\mathcal{W}^{(i)}(k) = \{l \in \mathcal{L} \mid \pi_{\mathrm{p}}^{(i)}(k) < l \leq \pi_{\mathrm{p}}^{(i)}(k) + w\}, \tag{15}$$

ahol $\mathcal{L} = \{1, \ldots, L\}$ az összes csomag halmaza és a $\pi_{\mathrm{p}}^{(i)}(k)$ a csomagalapú letöltési állapot, amelyet a következőképpen definiálok:

3.15. Definíció (Csomagalapú letöltési állapot)**.** A $\pi_{\mathrm{p}}^{(i)}(k) \in 0, \ldots, L$ a csomagalapú letöltési állapot, amely annak a legnagyobb azonosítójú csomagnak az azonosítója, amelyet meg nem töltött le az $i.$ csomópont:

$$\begin{aligned} \pi_{\mathrm{p}}^{(i)}(k) &= max(\mathcal{L}_{\mathrm{down}}^{(i)}(k)) \\ \mathcal{L}_{\mathrm{down}}^{(i)}(k) &= \{l \in \mathcal{L} \mid l. \text{ csomag letöltődott a } k. \text{ időpontig}\} \end{aligned} \tag{16}$$

A csomópontok egy A méretű gyorsítótárral is rendelkezik, ahol a letöltött csomagokat tudják tárolni és később másokkal megosztani. A csomópontok egypróbás gyorsítótárat használnak amit a következőképpen definiálok:

3.16. Definíció (Egypróbás gyorsítótár)**.** Egypróbás egy gyorsítótár ha az $\mathcal{A}_{\mathrm{pot}}^{(i)}(k)$ halmaz azon csomagokat jelenti, amelyeket tárolni lehetne, az $\mathcal{A}^{(i)}(k)$ halmaz a gyorsítótárban lévő csomagokat jelenti a $k.$ időpontban és

$$\begin{aligned} \mathcal{A}_{\mathrm{pot}}^{(i)}(k) &\subset \{p \in \mathcal{W}^{(i)}(k) \,|\, p \text{ is downloaded at time } k\} \\ \mathcal{A}^{(i)}(k) &\subset \mathcal{L}_{\mathrm{down}}^{(i)}(k),\, w(\mathcal{A}^{(i)}(k)) \leq A, \end{aligned} \tag{17}$$

ahol a $w(\mathcal{A}^{(i)}(k))$ az $\mathcal{A}^{(i)}(k)$ halmaz számosságát jelenti.

A bemutatott rendszerben $\overline{r}$-t, a szerverterhelést minimalizálom:

$$\begin{aligned} \overline{r} &= \lim_{T\to\infty} \sum_{t=1}^{T} \frac{1}{T} r(k) \\ r(k) &= \frac{\text{a } k. \text{ időpontig a szervertől letölött csomagok száma}}{\text{a } k. \text{ időpontig letöltött összes csomag száma}}. \end{aligned} \tag{18}$$

Altézis III.2: Analitikus keretrendszer tervezése

Megmutattam a különböző gyorsítótárazási megoldások szerverterhelést csökkentő képességeit P2P-rel támogatott streamelő rendszerekben úgy, hogy terveztem egy analitikus keretrendszert és javasoltam négy egypróbás gyorsítótár megoldást. A keretrendszer segítségével elemeztem ezen gyorsítótárak hatását a szerverterhelésre és megmutattam, hogy RLNC képes lényegesen csökkenteni a terhelés mértékét.

Az analízis egyszerűsítése érdekében feltételeztem, hogy létezik egy legvalószínűbb csomópont konfiguráció, amelyet csomag alapú letöltés állapottal jellemzek. Ezt a Ψ transzformációs függvény és a csomópont kora p.d.f segítségével fejezek ki:

3.17. Definíció (Transzformációs függvény)**.** $\Psi : \pi_{\mathrm{t}} \to \pi_{\delta}$ a transzformációs függvény ami egy $\pi_{\delta} \in [0, 1]$ letöltési állapotot rendel minden csomóponthoz a hálózatban töltött idejük alapján.

3.18. Definíció ($f_{\mathrm{alive}}(t)$, a csomópont kora sűrűségfüggvény (p.d.f.))**.**

$$f_{\mathrm{alive}}(t)\Delta t = \frac{\Delta t(1 - F_{\mathrm{skew}}(t))}{\int_0^{\infty}(1 - F_{\mathrm{skew}}(t))\mathrm{d}t}, t \geq 0, \tag{19}$$

ahol a $F_{\mathrm{skew}}(t)$ a csomópont kilépési skew-normal eloszlásfüggvénye.

3.19. Definíció ($\varrho = \{\varrho_1, \ldots, \varrho_L\}$, a csomag alapú csomópont sűrűség).

$$\varrho_l = \int_{\frac{l}{L+1}}^{\frac{l+1}{L+1}} f_{\text{prg}}(n)\mathrm{d}t,\ l = 0, \ldots, L-1, \tag{20}$$

ahol $f_{\text{prg}}(n) = f_{\text{alive}}(\Psi^{-1}(n))$.

A Sűrűségfüggvény felhasználásával kifejeztem a gyorsítótár hiba arányát egy P2P kapcsolat esetén:

Gyorsítótár hiba $C = 1$ esetén

$$s(n) = \sum_{m=0, m\neq n}^{L} (1 - \frac{\mathbb{E}(\delta_{nm})}{\min\{w, L-n\}})\varrho_n w_p(n,m), \tag{21}$$

ahol a $w_p(n,m)$ annak a valószínűsége, hogy az i. csomópont $\pi_{\text{p}}^{(i)} = n$-vel kapcsolódik egy j csomóponthoz, ahol $\pi_{\text{p}}^{(j)} = m$. Az $\mathbb{E}(\delta_{nm})$ a várható értéke annak, hogy a i hány csomagot kap j-től.

Ezt felhasználva, kifejeztem a gyorsítótár hiba arányát több kapcsolat esetén:

Gyorsítótár hiba $C >= 1$ esetén Feltesszük, hogy a csomópontok létrejötte független esemény, akkor a gyorsítótár hiba az i. csomópontnál ($\pi_{\text{p}}^{(i)} = n$) a következőképpen adható meg:

$$\mathbf{S}(n) = \sum_{j=0}^{N} (1 - (1 - s(n))^j) w_c(j), \tag{22}$$

ahol a $w_c(j)$ annak a valószínűsége, hogy az i. csomópont ($\pi_{\text{p}}^{(i)} = n$) j kapcsolattal rendelkezik.

A gyorsítótár hiba felhasználva az átlagos szerverterhelés a következőképpen fejezhető ki:

Átlagos szerverterhelés

$$\overline{r} = \frac{1}{L} \sum_{n=0}^{L} (\mathbf{S}(n))\varrho_n. \tag{23}$$

Altézis III.3: Egypróbás gyorsítótár tervezése

Megmutattam a gyorsítótárak szerverterhelés csökkentő hatását a P2P-rel támogatott streamelő rendszerekre úgy, hogy terveztem négy egypróbás gyorsítótárat: korlátlan, FIFO, véletlenszerű és RLNC-vel kódolt gyorsítótárat. Megmutattam, hogy a véletlenszerű módszer teljesítménye felülmúlja a FIFO megoldást a szerverterhelés tekintetében, míg az RLNC tovább csökkenti a szerverterhelést.

Terveztem egy korlátlan gyorsítótáras megoldást:

3.20. Definíció (Korlátlan gyorsítótár)**.** A csomópontok minden letöltött csomagot képesek tárolni: $\mathcal{A}^{(i)}(k) = \mathcal{L}_{\text{down}}^{(i)}(k)$

Ez a megoldás az adott kapcsolatfelépítési és csomagletöltési megkötések mellett az optimális megoldást nyújtja.

Az $i.$ csomópont ($\pi_{\mathrm{p}}^{(i)} = n$), a $j.$ csomóponttól ($\pi_{\mathrm{p}}^{(j)} = m$, $m \geq n$) letölthető csomagok számának várható értéke, korlátlan gyorsítótár esetén a következőképpen fejezhető ki:

$\mathbb{E}(\delta_{nm})$ **korlátlan gyorsítótár esetén**

$$\mathbb{E}(\delta_{nm}) = \min\{w, m - n\} \tag{24}$$

3.21. Definíció (FIFO gyorsítótár)**.** FIFO gyorsítótár az utolsó A letöltött csomagot tárolja: $\mathcal{A}^{(i)}(k) \in \{p \,|\, \max\{\pi_{\mathrm{p}}^{(i)}(k) - A, 0\} < p \leq \pi_{\mathrm{p}}^{(i)}(k)\}$

$\mathbb{E}(\delta_{nm})$ **FIFO gyorsítótár esetén**

$$\mathbb{E}(\delta_{nm}) = \max\{0, \min\{A_{\mathrm{end}}^{(m)} - w_{\mathrm{start}}^{(n)}, w_{\mathrm{end}}^{(n)} - A_{\mathrm{start}}^{(m)}\}\}, \tag{25}$$

ahol az $A_{\mathrm{start}}^{(m)} = \max\{0, m - A\}$ és az $A_{\mathrm{end}}^{(m)} = m$ a gyorsítótár elejét és végét jelentik a $j.$ csomópontnál. A $w_{\mathrm{start}}^{(n)} = n$ és a $w_{\mathrm{end}}^{(n)} = \min\{n + w, L\}$ pedig a $i.$ csomópont ablakának elejét és végét jelentik.

3.22. Definíció (Véletlenszerű gyorsítótár)**.** A véletlenszerű gyorsítótár a letöltött csomagokat véletlenszerűen tárolja úgy, hogy annak a valószínűsége, hogy egy csomag benne van a gyorsítótárban az összes letöltött csomagra egyenlő:

$$P(p_1 \in \mathcal{A}^{(i)}(k)) = P(p_2 \in \mathcal{A}^{(i)}(k)),\ p_1, p_2 \in \mathcal{L}_{\mathrm{down}}^{(i)}(k), \tag{26}$$

ahol P a valószínűségfüggvény.

$\mathbb{E}(\delta_{nm})$ **véletlenszerű gyorsítótár esetén**

$$\mathbb{E}(\delta_{nm}) = \sum_{q=0}^{\min\{A,w,m-n\}} \frac{\binom{\min\{w,m-n\}}{q}\binom{\max\{n,m-w\}}{A-q}}{\binom{m}{A}} q. \tag{27}$$

3.23. Definíció (RLNC gyorsítótár)**.** RLNC gyorsítótár a L csomagot g méretű generációkba csoportosítja és RLNC-vel kódolja azokat.

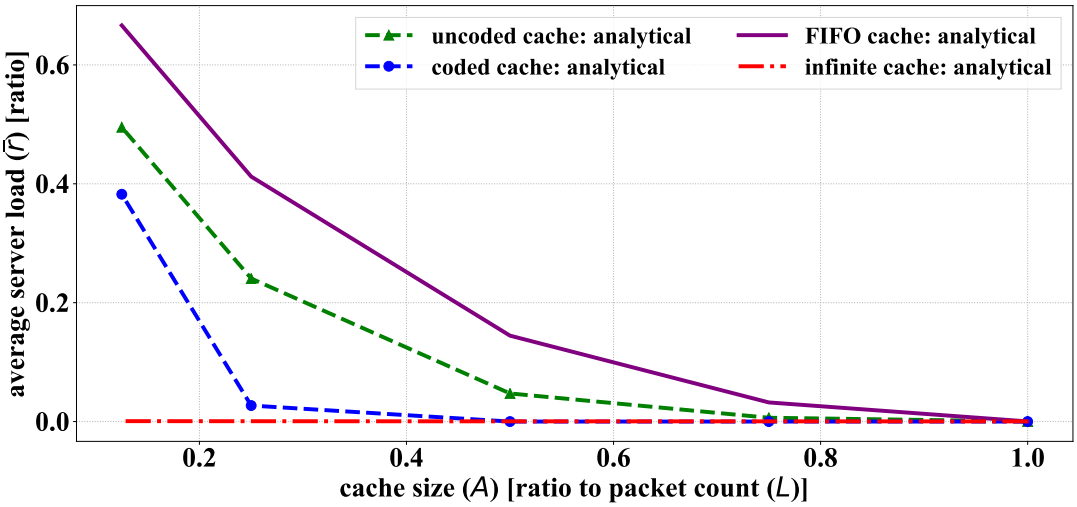
Az RLNC generációkkal dolgozik csomagok helyett, ezért javasoltam egy generáció alapú analízist, amelyet "′"-vel jelölök:

$\mathbb{E}'(\delta_{n'm'})$ **RLNC gyorsítótár estén**

$$\mathbb{E}'(\delta_{n'm'}) = \begin{cases} \min\left\{\frac{A}{n'}, w\right\} \text{ ha } g \geq w \\ \min\{w', m' - n'\} g_{\min}^{m'} + \mathbb{E}(P'_{n'm'}) \text{ egyébként,} \end{cases} \tag{28}$$

ahol

$$\mathbb{E}(P'_{n'm'}) = \sum_{q=0}^{\min\{(A \bmod m'), w', m'-n'\}} \frac{\binom{\min\{w', m'-n'\}}{q}\binom{\max\{n', m'-w'\}}{(A \bmod m')-q}}{\binom{m'}{(A \bmod m')}} q. \tag{29}$$



5. ábra. Szerverterhelés $N = 18$ csomópont, $g = 8$ generáció és $L = 633$ csomagméret esetén.

Ahogy az 5. ábra is mutatja, véletlenszerű gyorsítótárazás esetén 40%-al kevesebb csomagot szükséges tárolni ugyanolyan teljesítmény eléréséhez, mint FIFO estén. RLNC ezt tovább csökkenti. RLNC esetén 50%-kal kevesebb csomag szükséges a nullához közeli szerverterhelés eléréshez mint a véletlenszerű megoldás esetén. Továbbá elég a csomagok felét tárolni RLNC esetén, hogy a szerverterhelés nullára essen, amikor a csomópontok képesek az újonnan jövő csomópontokat önállóan kiszolgálni.

Altézis III.4: Protokollok és algoritmusok tervezése P2P-rel támogatott streameléshez

Megmutattam a P2P-rel támogatott streamelés gyakorlati hasznát úgy, hogy terveztem és implementáltam a Peer-to-Peer Assisted Streaming Network-t (PasNet) és több mint 100 táblagépen futtattam azt. PasNet-hez több algoritmust és protokollt terveztem a négy egypróbás gyorsítótáraimat felhasználva. A rendszeren több mérést végeztem, aminek az eredményei hasonló tendenciát mutattak mint az analitikus keretrendszerem eredményei.

3.24. Protokoll (WebPeer)**.** WebPeer egy kommunikációs protokoll, amely a véletlenszerű gyorsítótáras megoldást implementálja. A támogatott üzeneteket az 1. táblázat foglalja össze. Minden üzenet csomagazonosítót tartalmaz. Amikor két csomópont kap-

1. táblázat. WebPeer üzenetek

Elérhetőségi vektor	BitTorrent bitfield-jéhez hasonló vektor, a csomópont gyorsítótárában lévő csomagot jelzi.
Megvan	Jelzi, ha új csomag elérhető a csomópontnál.
Elveszett	Jelzi, ha egy csomag törlődött a csomópontnál.
Kérés	Adatcsomagkérés.
Mégse	Kérés törlése.
Adat	Adatcsomagot tartalmaz.

csolódik egymáshoz, kicserélik az *elérhetőségi vektorukat*. Később csak *elveszett* és *megvan*

üzenettel jelzik a emphelérhetőségi vektorukban történő változásokat. Külön kézfogás mechanizmust nem definiál a protokoll, az alsóbb rétegek kézfogásaira támaszkodik.

3.25. Protokoll (CodedWebPeer)**.** CodedWebPeer a WebPeer-t egészíti ki RLNC gyorsítótáras megoldással a következőképpen: *megvan* üzenet az adott generáció rangját tartalmazza. *Elveszett* üzenet nem támogatott, mert a *megvan* üzenet ezt helyettesíti, ha alacsonyabb ranggal küldjük a generációt. A *kérés* és a *mégse* üzenetek a generáció azonosító mellett a kért csomag mennyiségét is tartalmazzák.

3.4. Algoritmus (Kapcsolatkezelés)**.** *A csomópontok minden bejövő kapcsolatot elfogadnak, míg új kapcsolatot csak akkor létesítenek, ha C-nél kevesebb nyitott kapcsolatuk van. Minden kapcsolat kétirányú és legalább T_{con} ideig nyitva maradnak. Ha egy csomópont több mint $C-1$ kapcsolattal rendelkezik, akkor a leglassabb kapcsolatot bezárja.*

3.5. Algoritmus (Letöltés ütemezés)**.** *Csomópontok csak akkor ütemeznek egy csomagot a szerverhez letöltésre, ha semelyik más P2P kapcsolatánál nincs meg az adott csomag. Az egyes P2P kapcsolatokhoz ütemezett csomagok mennyisége egyenesen arányos az adott kapcsolat átviteli rátájával. Ha egy csomag nem érkezik meg a kérés elküldése után T_{down} időn belül, akkor az adott csomagot újra ütemezi a csomópont.*

A protokolljaim és algoritmusaim felhasználásával terveztem és implementáltam a *Peer-to-Peer Assisted Streaming Network-t* (PasNet), amelyet több mint 100 táblagépen is futtattam, hogy megmutassam a gyakorlati potenciálját. A *PasNet*-n végzett méréseim eredményeit felhasználva validáltam az analitikus keretrendszeremet. A két rendszer eredményei hasonló tendenciákat mutattak és az eredmények között ≤ 0.149 négyzetes eltérés volt.

Tézis IV: Hálózatperemi felhő

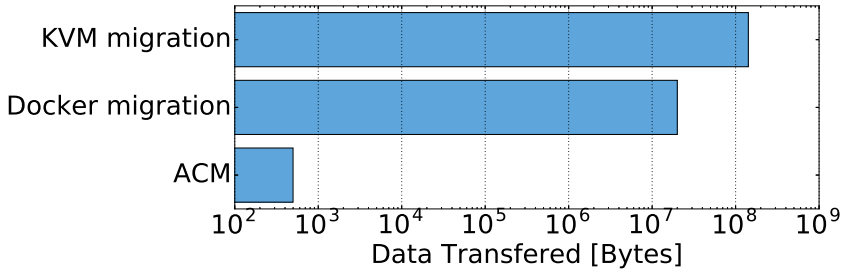
Minimalizáltam a felhő migrálás idejét úgy, hogy tanulmányoztam több migrációs megoldást és megterveztem az Agile Cloud Migration-t (ACM). Megmutattam az ACM képességeit azáltal, hogy terveztem és implementáltam egy ACM alapú rendszert, amely képes a felhő alapú szervereket valós időben migrálni, akár a hálózat peremére is mozgatni. Mérések segítségével megmutattam, hogy a megoldásom teljesítménye lényegesen felülmúlja a korszerű KVM és Docker alapú megoldásokat.

A tézishez tartozó publikációim: [19], [18], [20]
A tézishez tartozó előadások és bemutatók: [26], [27], [28], [29], [30], [31], [32], [33], [34]

Altézis IV.1: Szoftverarchitektúra tervezése ACM-hez

A gyors felhő migráláshoz terveztem egy három állapotú szoftver architektúrát, amely külön kezeli a hálózati kommunikációt, az alkalmazás feldolgozó egységét és az állapotát. Megmutattam, hogy a megoldásom teljesítménye lényegesen felülmúlja a korszerű KVM és Docker alapú megoldásokat.

Ebben a tézisben a tartalomkiszolgálóknak, amelyeket *motoroknak* is nevezek, három állapota van. A motor architektúrája három egységre osztható fel: 1) *Feldolgozó*, amely képes módosítani az 2) *Állapotot* és a *Hálózat*, amely kezeli a bejövő üzenteket és továbbítja



6. ábra. Migrációs módszerek összehasonlítása.

a módosított *Állapotot* a kliensek felé. A *Hálózat* modul három csatornával rendelkezik: 1) parancs csatorna, amely a kliensektől érkező parancsokat továbbítja a feldolgozónak. 2) A kliens-állapot csatorna, amely aktuális állapotot elküldi a klienseknek. 3) A motor csatorna, amelyen keresztül bonyolítja le a migrálást a két motor.

Az architektúrának három állapota van: 1) aktív, amikor a motor fogadja az üzeneteket. 2) Migráció alatt, amikor a motor átadja a vezérlést egy másik motornak. 3) Készenléti, amikor várja, hogy egy másik motor átadja neki a vezérlést.

Méréseken keresztül megmutattam, hogy a megoldásomat alkalmazva a migrációhoz szükséges adat továbbítása 143 MB-ról pár KB-ra csökkenthető.

Altézis IV.2: Kommunikációs protokoll tervezése ACM-hez

Terveztem egy kommunikációs protokoll a gyors felhő migráláshoz, amely a felhasználók számára átlátszó módon képes mozgatni a szervereket. Hogy megmutassam az ACM képességeit terveztem és implementáltam egy böngésző alapú többszereplős játékot, amely képes volt a felhőben lévő számításokat helyszínek között mozgatni minimális késleltetéssel. A szerver a hálózat peremére is mozgatható volt, hogy a szerver-kliens késleltetés minimálisra csökkenjen.

Terveztem egy öt lépésből álló migrációs protokollt:

3.26. Protokoll (Migrációs protokoll ACM-hez)**.** 1) A motor (tartalomkiszolgáló) kapcsolódik az új pozíción lévő motorhoz. 2) A régi motor *migráció alatt* állapotba lép. Átküldi az *Állapotot* az új motornak, valamint minden bejövő parancsot is továbbít neki. 3) Az új motor megkapja az *Állapotot* és aktív állapotba lép. 4) A régi motor készenléti állapotba lép és jelzi a csomópontoknak, hogy kapcsolódjanak fel az új motorhoz. 5) A csomópontok kapcsolódnak az új motorhoz.

A protokollom migrációs idejének kiszámítására a következő formula hasznát javasoltam:

t, migrációs idő kiszámítása

$$t = L_{\mathrm{ctrl,dst}} + C_{\mathrm{dst,src}} + L_{\mathrm{dst,src}} + T_{\mathrm{state}} + L_{\mathrm{src,dst}} + L_{\mathrm{dst,src}} + L_{\mathrm{src,cl}} + C_{\mathrm{cl,dst}} + L_{\mathrm{cl,dst}}, \quad (30)$$

ahol az $L_{\alpha,\beta}$ az α és a β közötti késleltetés ideje. A $C_{\gamma,\delta}$ a γ és δ motorok közötti kapcsolat felépítésének ideje. A T_{state} az *Állapot* átvitelének ideje.

A megoldásom teljesítménye lényegesen felülmúlja a korszerű KVM és Docker alapú migrációs megoldásokat, ahogy azt a 6. ábra is mutatja. Méréseken keresztül megmutattam, hogy a migrációs idő pár milliszekundumomra is lecsökkenthető, ami a jövőbeli 5G-s hálózatok egy célja is.

4. Eredmények gyakorlati alkalmazása

Mind a négy bemutatott tézisem az elméleti eredmények mellett gyakorlati alkalmazásokat is tartalmaz.

Az I. tézisben a többforrásos letöltéshez terveztem egy rendszert, amely több Amazon Web Service-ben (AWS) futó tartalomkiszolgálót tartalmazott, valamint Chrome és Firefox böngészőkiegészítő alapú klienset. A böngészőkiegészítők elkapták a YouTube videóletöltéseket és több tartalomkiszolgálótól töltötték le azokat párhuzamosan. Rendszeren tizenegy hónapon keresztül végeztem méréseket, amelyet több mint 75 felhasználó használt, akik 1300000 db naplóbejegyzés generáltak.

A II. tézisemben egy BitTorrent kiegészítőt implementáltam egy Java alapú BitTerrent klienshez. Létrehoztam egy kontrollált hálózattal rendelkező mérési környeztet és több mérést végeztem a rendszeren, hogy igazoljam a hipotéziseimet.

A III. tézisemben megterveztem a *PasNet*-t, amely kódolatlan és kódolt MUTP protokoll implementációkat használta, hogy P2P-rel támogatott rendszerben adatot streameljen. *PasNet*-t több mint 100 táblagépen is futtattam, hogy demonstráljam a gyakorlati potenciálját. Ezenkívül több konferencián és kiállításon is bemutattam az alkalmazást, többek között: 5G Summit-on Drezdában, CCNC-n és CES-n Las Vegasban, ahol demonstrációs díjat is nyert.

A IV. tézisemben egy többszereplős játékot terveztem és implementáltam ami az ACM megoldásomat használja felhő migrálásra. A rendszeren több mérést is végeztem és megmutattam, hogy a saját megoldásom teljesítménye lényegesen felülmúlja a korszerű KVM és Docker alapú migrációs megoldásokat. A játékalkalmazást több konferencián és kiállításon is bemutattam, többek között a 5G Summit-on Drezdában, az IFA-n Berlinben és a CCNC-n és a CES-n Las Vegasban.

Tudományos publikációk

Folyóirat cikkek

[1] **Patrik J. Braun**, M. Médard és P. Ekler. "Practical Evaluation of Multi-source Coded Downloads". *IEEE Access* 7 (2019). **IF*: 4.098**, 120304–120314. old. DOI: 10.1109/ACCESS.2019.2936638.

[2] Márton Sipos, **Patrik J. Braun**, D. Lucani, F. H. P. Fitzek és H. Charaf. "On the Effectiveness of Recoding-based Repair in Network Coded Distributed Storage". *Periodica Polytechnica Electrical Engineering and Computer Science* 61.1 (2017), 12–21. old. DOI: https://doi.org/10.3311/PPee.9377. URL: https://pp.bme.hu/eecs/article/view/9377.

[3] **Patrik J. Braun**, Á. Budai, J. Levendovszky, M. Sipos, P. Ekler és F. H. P. Fitzek. "Mobile Peer-to-Peer Assisted Coded Streaming". *IEEE Access* 7 (2019). **IF*: 4.098**, 159332–159346. old. DOI: 10.1109/ACCESS.2019.2950800.

Előkészített folyóirat cikkek

[4] **Patrik J. Braun**, D. Malak, M. Médard és P. Ekler. "Goodput Analysis for Multi-Source Coded Downloads". *IEEE Transactions on Wireless Communications* (2019). **IF*: 6.394**.

Konferencia cikkek

[5] Dániel Pásztor és **Patrik J. Braun**. "Improving content delivery systems with Network Coding over WebRTC". *2017 Automation and Applied Computer Science Workshop (AACS)*. Budapest, Hungary, 2017. jún.

[6] **Patrik J. Braun**, D. Malak, Muriel Médard és P. Ekler. "Multi-Source Coded Downloads". *IEEE International Conference on Communications (ICC)*. Shanghai, China, 2019. máj., 1–7. old.

[7] **Patrik J. Braun**, D. Malak, M. Médard és P. Ekler. "Enabling Multi-source Coded Downloads". *IEEE International Conference on Edge Computing (EDGE)*. Milan, Italy, 2019. júl.

[8] **Patrik J. Braun**, M. Sipos, P. Ekler és H. Charaf. "Increasing data distribution in BitTorrent networks by using network coding techniques". *Proceedings of 21th European Wireless Conference*. Budapest, Hungary, 2015. máj., 1–6. old.

[9] **Patrik J. Braun** és M. Sipos. "Reducing linear dependencies in network coding assisted BitTorrent networks". *Automation and Applied Computer Science Workshop (AACS)*. Budapest, Hungary, 2015. jún.

[10] **Patrik J. Braun** és M. Sipos. "Analysis of power usage on android platform". *Automation and Applied Computer Science Workshop (AACS)*. Budapest, Hungary, 2014. jún.

[11] **Patrik J. Braun**. "Peer-to-peer streaming using Typescript and Network Coding". *Automation and Applied Computer Science Workshop (AACS)*. Budapest, Hungary, 2016. jún.

[12] **Patrik J. Braun** és P. Ekler. "Improving QoS in web-based distributed streaming services with applied network coding". *The 10th Jubilee Conference of PhD Students in Computer Science*. **Best Talk Award**. Szeged, Hugnary, 2016, 48. old.

[13] **Patrik J. Braun**, P. Ekler és F. H. P. Fitzek. "Network coding enhanced browser based Peer-to-Peer streaming". *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. Budapest, Hungary, 2016. okt.

[14] **Patrik J. Braun**, M. Sipos, P. Ekler és F. H. P. Fitzek. "On the Performance Boost for Peer to Peer WebRTC-based Video Streaming with Network Coding". *IEEE International Conference on Communications (ICC)*. Paris, France, 2017. máj., 1–6. old.

[15] **Patrik J. Braun**, Péter Ekler és F. H. P. Fitzek. "Demonstration of a P2P assisted video streaming with WebRTC and Network Coding". *14th IEEE Annual Consumer Communications Networking Conference (CCNC)*. **Demonstration Award Runner-Up**. Las Vegas, USA, 2017. jan.

[16] **Patrik J. Braun**. "Hot content analysis in a P2P-assisted VoD streaming system with network coding". *Automation and Applied Computer Science Workshop (AACS)*. **Best Paper of the Section Award**. Budapest, Hungary, 2017. jún.

[17] Khalid Kahloot, **Patrik J. Braun** és Péter Ekler. "Behavior Analysis for WebRTC Peer-to-Peer Streaming with Dynamic Topology". *13th International Conference on Wireless and Mobile Communications (ICWMC)*. 2017, 43. old. ISBN: 978-1-61208-572-2.

[18] **Patrik J. Braun**, S. Pandi, R. Schmoll és F. H. P. Fitzek. "On the study and deployment of mobile edge cloud for tactile Internet using a 5G gaming application". *14th IEEE Annual Consumer Communications Networking Conference (CCNC)*. Las Vegas, USA, 2017. jan., 154–159. old. DOI: 10.1109/CCNC.2017.7983098.

[19] S. Pandi, R. S. Schmoll, **Patrik J. Braun** és F. H. P. Fitzek. "Demonstration of mobile edge cloud for tactile Internet using a 5G gaming application". *14th IEEE Annual Consumer Communications Networking Conference (CCNC)*. Las Vegas, USA, 2017. jan., 607–608. old.

[20] R. Schmoll, S. Pandi, **Patrik J. Braun** és F. H. P. Fitzek. "Demonstration of VR / AR offloading to Mobile Edge Cloud for low latency 5G gaming application". *15th IEEE Annual Consumer Communications Networking Conference (CCNC)*. Las Vegas, USA, 2018. jan., 1–3. old. DOI: 10.1109/CCNC.2018.8319323.

Benyújtott szabadalom

[21] M. Médard, J. László és **Patrik J. Braun**. *System and Technique for Generating, Transmitting and Receiving Network Coded (NC) QUICK UDP Internet Connections (QUIC) Packets*. PCT/US18/59340. 2018. szept.

Fontosabb előadások és bemutatók

[22] **Patrik J. Braun**, D. Malak, M. Médard és P. Ekler. "Keynote talk about Multi-Source Coded Downloads for Future Networks". *IEEE 5G Summit Dresden 2019*. **Presenter: Patrik J. Braun**. Dresden, Germany, 2019. szept.

[23] **Patrik J. Braun**, P. Ekler és F. H. P. Fitzek. "Demonstration of a Network Coding Enhanced Browser-based Peer-to-Peer Streaming system". *5G Lab Germany Industry Day 2015*. **Presenter: Patrik J. Braun**. Dresden, Germany, 2015. szept.

[24] **Patrik J. Braun**, P. Ekler és F. H. P. Fitzek. "Demonstration of a P2P-assisted Data Distribution with WebRTC". *IEEE 5G Summit Dresden 2016*. **Presenter: Patrik J. Braun**. Dresden, Germany, 2016. szept.

[25] **Patrik J. Braun**, P. Ekler és F. H. P. Fitzek. "Demonstration of a P2P-assisted video streaming system with WebRTC and Network Coding". *2017 Consumer Electronics Show (CES'17)*. **Presenter: Patrik J. Braun**. Las Vegas, USA, 2017. jan.

[26] **Patrik J. Braun**, S. Pandi, R. Schmoll és F. H. P. Fitzek. "Demonstration of Mobile Edge Cloud for Tactile Internet using a 5G Gaming Application". *International Funkausstellung (IFA) 2016*. Presenter: Deutsche Telekom. Berlin, Germany, 2016. szept.

[27] **Patrik J. Braun**, S. Pandi, R. Schmoll és F. H. P. Fitzek. "Demonstration of Mobile Edge Cloud for Tactile Internet using a 5G Gaming Application". *IEEE 5G Summit Dresden 2016*. Presenter: Sreekrishna Pandi. Dresden, Germany, 2016. szept.

[28] **Patrik J. Braun**, S. Pandi, R. Schmoll és F. H. P. Fitzek. "Demonstration of Mobile Edge Cloud for Tactile Internet using a 5G Gaming Application". *Tag der Deutschen Einheit 2016*. Presenter: Sreekrishna Pandi. Dresden, Germany, 2016. okt.

[29] **Patrik J. Braun**, S. Pandi, R. Schmoll és F. H. P. Fitzek. "Demonstration of Mobile Edge Cloud for Tactile Internet using a 5G Gaming Application". *2017 Consumer Electronics Show (CES'17)*. Presenter: Sreekrishna Pandi. Las Vegas, USA, 2017. jan.

[30] R. Schmoll, S. Pandi, **Patrik J. Braun** és F. H. P. Fitzek. "Demonstration of VR / AR offloading to Mobile Edge Cloud for low latency 5G gaming application". *Centrum für Büroautomation, Informationstechnologie und Telekommunikation (CeBit) 2017*. Presenter: R. Schmoll. Hanover, Germany, 2017. márc.

[31] R. Schmoll, S. Pandi, **Patrik J. Braun** és F. H. P. Fitzek. "Demonstration of VR / AR offloading to Mobile Edge Cloud for low latency 5G gaming application". *2018 Consumer Electronics Show (CES'18)*. Presenter: R. Schmoll. Las Vegas, USA, 2018. jan.

[32] R. Schmoll, S. Pandi, **Patrik J. Braun** és F. H. P. Fitzek. "Demonstration of VR / AR offloading to Mobile Edge Cloud for low latency 5G gaming application". *International Funkausstellung (IFA) 2018*. Presenter: Deutsche Telekom. Berlin, Germany, 2018. szept.

[33] R. Schmoll, S. Pandi, **Patrik J. Braun** és F. H. P. Fitzek. "Demonstration of VR / AR offloading to Mobile Edge Cloud for low latency 5G gaming application". *IEEE 5G Summit Dresden 2018*. Dresden, Germany, 2018. szept.

[34] R. Schmoll, S. Pandi, **Patrik J. Braun** és F. H. P. Fitzek. "Demonstration of VR / AR offloading to Mobile Edge Cloud for low latency 5G gaming application". *5G Experience Hub 2018*. Warsaw, Poland, 2018. dec.

Hivatkozások

[Eri18] Ericsson. *Ericsson Mobility Report*. Ericsson. 2018. nov. URL: https://www.ericsson.com/assets/local/mobility-report/documents/2018/ericsson-mobility-report-november-2018.pdf.

[Ho+03] T. Ho, R. Koetter, M. Medard, D. R. Karger és M. Effros. "The benefits of coding over routing in a randomized setting". *IEEE International Symposium on Information Theory, 2003. Proceedings.* 2003. jún., 442–. old. DOI: 10.1109/ISIT.2003.1228459.

[AN07] K. Ausavapattanakun és A. Nosratinia. "Analysis of Selective-Repeat ARQ via Matrix Signal-Flow Graphs". *IEEE Transactions on Communications* 55.1 (2007. jan.), 198–204. old. ISSN: 0090-6778. DOI: 10.1109/TCOMM.2006.885092.