Összeköttetés alapú kommunikációs protokollok állapottér-idő elemzése dinamikus idővetemítéssel

Dynamic Time Warping Based State Space-Time Analysis of the Connection Oriented Protocols

Dr. GÁL Zoltán

Szuperszámítógép (HPC) Központ vezető
Debreceni Egyetem Informatikai Kar, H-4028 Debrecen, Kassai út. 26., ZGal@inf.unideb.hu

Abstract

Processing of the data generated by the Internet of Things requires Big Data category services. High transmission capacity services are needed to provide efficiency of the interconnected high capacity processing and storage subsystems. There are considerable number of different congestion control mechanisms offered by the Internet technologies today. In the paper we investigate sixteen different connection oriented service types of the transport layer and we evaluate similarity measure based on Dynamic Time Warping algorithm. It was found that the TCP versions used in practice today can be grouped in three behaviour classes. Class belonging of these versions depend strongly on the number of parallel communication sessions running on the same path of data transfer.

Keywords: Internet of Things, transport layer protokols, time series analysis, Dynamic Time Warping, k-Mean clustering, Hiearchical clustering

Kivonat

A Tárgyak Internete által egyre nagyobb mennyiségben előállított szenzor adat feldolgozása Big Data kategóriába tartozó módszereket igényel. A hatékonyság miatt ezen adatok nagykapacitású feldolgozó, illetve tároló alrendszerekhez való továbbítása nagysebességű átvitelt feltételez. Mivel az Internet technológiák a szállítási rétegében sok féle torlódásvezérlési mechanizmus működtetnek, a dolgozatban ezek közül a gyakorlatban elterjedt tizenhat változat összehasonlítását végeztük el. Dinamikus idővetemítés módszerével végzett számszerűsítés alapján ezek három fajta viselkedési módot mutatnak, ami erőteljesen függ az azonos kommunikációs útvonalon egyidőben működő kapcsolatok számától.

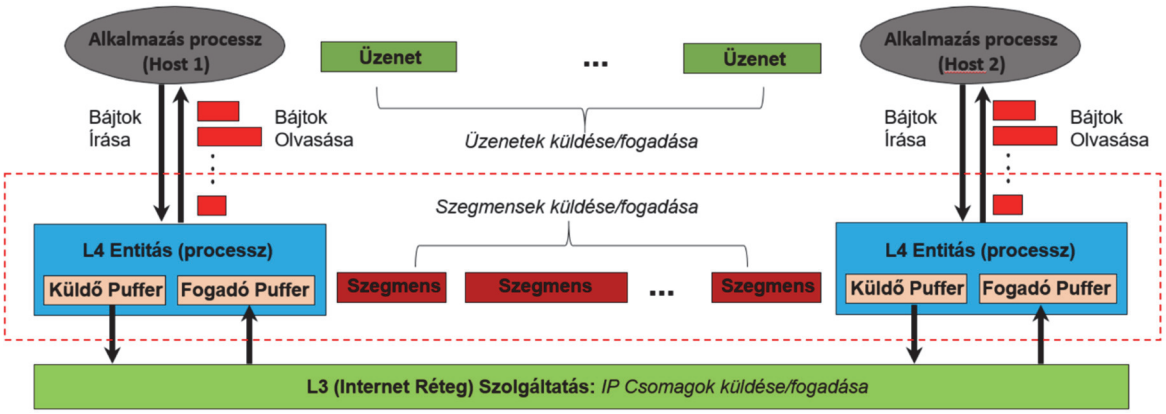
Kulcsszavak: Tárgyak Internete, szállítási réteg protokoll, idősoranalízis, idővetemítés, k-Mean algoritmus, Hierarchikus klaszter algoritmus

1. BEVEZETÉS

A Tárgyak Internete napjainkban egyre több adatot állít elő, ami a Big Data korszak idején ezeknek nem csak a feldolgozását és részbeni vagy teljes mértékben való tárolását, hanem továbbítását is szükségessé teszi. A nagy adatmennyiségek számítógépek közötti mozgatása különösen nagy kihívás napjainkban, amikor a lehetőségek széles palettájából kell kiválasztani adott esetben a célra legalkalmasabbat. Az IntServ és a DiffServ QoS mechanizmusok ugyan biztosítják a különböző típusú tartalmak (adat, hang, videó) legelőnyösebb továbbítását a közös és konvergált hálózati infrastruktúrán, de a nagymennyiségű adatok gyors továbbításának optimális kivitelezése korunk továbbra is egyik komoly műszaki kihívása marad [1]. IP technológiára épülő kliens-szerver rendszerekben az adatok átvitele jelentős időbe telik, ami miatt a Big Data feladatok elvégzése késlekedéssel jár [2]. Ugyanakkor az időérzékeny forgalmaknak ezen adatokkal azonos útvonalon történő továbbítása esetén minőségi romlás is

tapasztalható az egymástól független folyamok között megjelenő kölcsönös fékező hatás miatt. Részleteibe menően történt ezen forgalmak egymásra hatásának tárgyalása a [3] és [4] dolgozatokban.

Ebben a dolgozatban azonban a különböző hosszúságú idősorok közötti hasonlóság mértékét számszerűsíteni képes dinamikus idővetemítés módszerével megállapítjuk a különböző típusú torlódásvezérlési típusok viselkedése közötti különbségeket. Mivel nagy adathalmazok továbbításánál jelentkező torlódás esetén a szállítási réteg mechanizmus típusa erőteljesen befolyásolja az átviteli időt, ezért a legelőnyösebben alkalmazható változatot keressük. Ehhez röviden összefoglaljuk a jelenlegi Internet technológia esetén a negyedik logikai rétegben alkalmazott kommunikációs szolgáltatások működésének fontosabb elemeit (1. ábra).



1. ábra. *Szállítási réteg mechanizmusok főbb elemei*

Nyilvánvaló okok miatt a szállítási réteg (L4) rövid ismertetése esetén is figyelembe kell vennünk a jelenlegi Internet technológiáknak a hibrid referencia modell szerinti szomszédos rétegei (L3 és L5) által nyújtott szolgáltatásokat, illetve elvárásokat. Mindhárom réteg entitásánál küldő és fogadó puffer létezik. Az alkalmazás processz (L5) üzenetek továbbítását a végzi, amik szegmensenkénti részleteit a szállítási processz továbbítja felhasználva az IP (L3) hálózati réteg nem megbízható datagram szolgáltatását. Az összeköttetés alapú (CO –Connection Oriented) L4 szolgáltatás adatfolyam vezérlést alkalmaz az alsóbb rétegek torlódásos eseményeinek kommunikációs korlátai miatt. Ez megbízható szegmens folyamot eredményez, de az átviteli ráta a lehetőséghez képes lényegesen kisebb. Összeköttetés mentes (CL – Connectionless) L4 szolgáltatás az L3 datagram funkciókat ismétli nagyon kis fejrész ráadással, cserében maximális továbbítási ráta lehetőséget biztosít az alkalmazási rétek számára. Komoly kihívás a különböző L4 szolgáltatások azonos kommunikációs útvonal mentén történő egyidejű működése esetén az L3 csatorna kapacitásának leghatékonyabb kihasználása, valamint L5 szolgáltatások számára pártatlan hozzáférés és maximális átviteli ráta biztosítása.

Jelen dolgozat további felépítése a következő: a második fejezetben az IP feletti szállítási réteg protokollok speciális tulajdonságait tárgyaljuk, kiemelve az összeköttetés alapú kommunikációs szolgáltatások manapság használatos, legismertebb változatainak jellemzőit. A harmadik fejezet bemutatja a tizenhat féle TCP (Transmission Control Protocol) forgalmát mérni képes rendszert és az azzal kinyert idősorokat. A negyedik fejezetben a különböző hosszúságú és tulajdonságú idősorok dinamikus idővetemítésen alapuló összehasonlítási módszerét, valamint a megállapításokat ismertetjük. Az utolsó fejezet összefoglalja a dolgozat eredményeit és a kutatási munka lehetséges folytatási irányait adja meg.

2. ÖSSZEKÖTTETÉS TÍPUSÚ SZÁLLÍTÁSI RÉTEG PROTOKOLLOK SPECIÁLIS JELLEMZŐI

A szállítási réteg protokolljai az adatfolyam szabályozási képesség típusától függő jelenléte vagy hiánya miatt meghatározó mértékben befolyásolják az alkalmazási réteg adatainak továbbítási képességét. Távoli gépek közötti átviteli ráta három kritikus tényezőtől függ: link-ek átviteli sebeségétől, végpontok közötti késleltetéstől, illetve a szállítási réteg hatékonyságától. Az Internet esetében úgy a

megbízható, mint a megbízhatatlan átviteli szolgáltatás létezik. Előbbi a kontroll síkban komplex vezérlő tevékenységet végez jelentős késleltetést okozva, míg utóbbi alacsony késést, de datagramok esetleges kihagyását okozhatja.

Jelen dolgozatban a TCP (Transmission Control Protocol) komplex szolgáltatásait tárgyaljuk, kiemelve azon sajátosságokat, amelyek alapján a különböző implementációk megkülönböztethetők egymástól. Az L4 szegmensek megbízható kézbesítése mellett a TCP az adatátviteli rátájának adaptív vezérléséhez szabályozható megoldásokat alkalmaz. Egyik ilyen legfontosabb a torlódásvezérlési algoritmus típusa, amely a kapcsolat kialakításakor választható. Ez a mechanizmus a köztes hálózatban esetlegesen megjelenő torlódás megjelenésekor a küldő entitás oldalon korlátozza az időegység alatt kiküldött adatmennyiséget, majd az átviteli lehetőség javulása esetén újból növeli a forgalmat. A közteshálózaton, az L4 entitáspárok között egyidőben létező különböző adatforgalmak egymásra hatással vannak, befolyásolva ezáltal minden TCP kapcsolaton a torlódásvezérlés aktuális állapotát, hogy a rendelkezésre álló összes sávszélesség a lehető legigazságosabban kerüljön felhasználásra. A torlódásvezérlési mechanizmus része a TCP-nek, így annak cseréje egyben a TCP változat cseréjét is jelenti. A továbbiakban a gyakorlatban használt változatok fontosabb tulajdonságait ábécé sorrendben foglaljuk össze [5-25].

1. BBR (Bottleneck Bandwidth and RTT): Ezt a modell alapú algoritmust a Google fejlesztette a YouTube számára, amely működés közben méri a torlódásos sávszélességet és a körbejárási időt (RTT – Round Trip Time). Adatfolyam küldésének kezdetén a küldő az átviteli rátát fokozatosan növeli, majd a fogadó nyugtájának (ACK) torlódás miatti kiesése esetén lecsökkenti a várakozási sor méretét és próbálja megállapítani jelen adatfolyam számára a létező sávszélességet. Ezt az RTT mindkét irányban való monitorozása segítségével éri el. A módszer hátránya a skálázhatóság hiánya.

2. BIC (Binary Increase Congestion): Gyors, nagytávolságú hálózatokra optimalizált megoldás. Az implementálás során kiemelt figyelmet fordítottak a skálázhatóságra, az RTT pártatlanságára (fairness), a TCP felhasználóbarát jellegére és a konvergenciára. A TCP window paraméter méretét bináris keresési algoritmussal határozza meg. A konvergencia finomhangolásához additív növelést, míg az RTT növekedése esetén multiplikatív csökkentést használ.

3. CDG (CAIA Delay-Gradient): A Linux operációs rendszerre kidolgozott változat a késleltetés gradiensre alapoz, mint torlódást mérő indikátorra. Az adatfolyam vezérlése az elveszett L4 szegmensek alapján történik. A torlódásra nagyon érzékeny, de elviseli az egyéb okok miatti csomagvesztést. Az RTT körbejárási idő minimális és maximális értékéhez tartozó gradiens mozgó átlagát számolja gördülő időablakban.

4. CUBIC (Cubic BIC - Binary Increase Congestion): A Linux 2.6.19 kerneltől alkalmazott megoldás. A TCP ablakméret növelését agresszíven növeli, így az optimális ablakméretet gyorsabban el tudja érni, mint a régebbi változatok. Nagysebességű hálózaton rövid tartalom küldésekor az egyéb változatok lassabb konvergenciája miatt az optimális ablakméret elérése előtt már befejeződik az átvitel, gyengébb hatásfok mellett. A BIC-hez hasonlóan, alacsony sebességű hálózatnál a TCP ablak méretének gyors növelése nem jelent előnyt. Párhuzamos adatfolyamok azonos útvonalon történő továbbítása esetén jó hatásfokot ad.

5. DCTCP (DataCenter TCP): Ez a RENO algoritmusnak adatközpontok számára továbbfejlesztett változata, amely erőteljes adatlöketek (burst) mellet alacsony késleltetést és nagy adatátviteli rátát képes nyújtani. A DCTCP a torlódásvezérlésen túlmenően a modern útválasztók RED (Random Early Detection) megjelölés algoritmusának megfelelően aktív várakozási sor menedzsmentet is nyújt. Az ACK nyugtacsomagok helyett CE (Congestion Experienced) kódponttal ellátott visszirányú ECN (Explicite Congestion Notification) kontroll csomagokat küld a várakozási sorok túlterhelési állapotának gyors jelzése érdekében.

6. High Speed TCP (HSTCP): Sally Floyd (RFC 3649) által létrehozott mechanizmus, amely a TCP ablak paraméter (window) méretét a csomagok előnyösebb eldobási rátájához igazítja. Az additív növelés és multiplikatív csökkentés módszer paramétereit a TCP aktuális ablakméretének függvényében változtatja az adatátviteli ráta nagy értékének gyors elérése érdekében. Más TCP implementációkkal való kompatibilitás céljából a saját módszerét csak akkor alkalmazza, ha a csomagvesztési ráta kisebb egy megadott küszöbértéknél, egyébként a hagyományos módon működik.

7. HTCP (H-TCP): Nagysebességű és távoli csomópontok közötti átvitelre tervezték, amely az üzemi sávszélesség becsléshez a minimális, illetve maximális ráta függvényében módosítja a TCP ablakméret paramétert. Kis ráta esetén alacsony ablakmérettel, nagy sebességénél és nagy távolságnál nagy ablakmérettel dolgozik.

8. Hybla: Heterogén, nagyobb meghibásodási aránnyal és nagyobb körbejárási idővel működő környezetre (pl. műholdas kapcsolat) kidolgozott torlódásvezérlési változat. Analitikus elemzés alapján szabályozza a TCP ablakméretet. SACK opció és időbélyeg segítségével teszi hatékonnyá a szegmens sorozat átvitelét.

9. Illinois: Nagysebességű hálózatoknál javasolt változat, amely az elvesztett csomagok száma alapján dönt a TCP ablakméret növeléséről, illetve csökkentéséről. Legtöbb implementáció esetén az additív növelés és multiplikatív csökkentés mechanizmus nem elég hatékony a nagysebességű átvitel számára, mivel sok időbe telik a torlódásos állapot utáni sebesség adaptációja. Az Illinois fenntartja a pártatlanságot, a stabilitást és az útválasztók közötti kölcsönhatást. Az additív növelést más értékkel végezi attól függően, hogy mennyire távol vagy közel tart a folyamat a torlódáshoz. Távoli állapot esetén nagy additív lépésközt, míg torlódáshoz közeledve csökkenti az ablakméret növelési lépésközt. A multiplikatív csökkentésnél a szorzófaktor kicsi, ha a folyamat távol van a torlódástól és nagyobb, ha torlódástól csak kevésbé távolodott el. Egyébként megtartja a NewReno változat egyéb tulajdonságait.

10. LP (Low Priority): Amíg a legtöbb TCP torlódás vezérlési algoritmus a legnagyobb adatátviteli rátára törekszik a pártatlanság megőrzése mellett, addig az LP a pártatlanságot részesíti előnyben az átviteli rátával szemben. Ezzel az azonos útvonalon működő többi logikai kapcsolatot nem zavarja. A TCP ablakméret additív növelése és multiplikatív csökkentése mechanizmust következtetés fázissal egészíti ki és módosított visszacsatolás szabályt alkalmaz.

11. NV (TCP New Vegas): A TCP-Vegas optimalizált változata, amelyet adatközpontokra fejlesztettek. A torlódást a csomagok elvesztése előtt képes érzékelni. Csak olyan egyidejű, nagysebességű összeköttetések esetén javasolt használni, ahol az összes TCP változat azonos.

12. Reno/New Reno: A Reno a legrégebbi TCP torlódásvezérlési mechanizmusok egyike, amit a New Reno váltott fel annak ellenére, hogy Linux kernelekben még létezik. A Reno olyan esetben teljesít jól, ha a csomagvesztés ritka, mivel csak egyedi csomagok kiesését képes érzékelni. A New Reno több csomag kiesését is érzékelni tudja, ezáltal hatékonyabb elődjénél.

13. Scalable: A hagyományos TCP torlódásvezérlési mechanizmushoz képest ezen egyszerű változtatást végeztek, amit a TCP ablakméretet meghatározó függvény módosításával értek el. Ezáltal nagysebességű hálózatokban gyorsabban éri el a rendelkezésre álló sávszélességet.

14. Vegas: Ez a Reno torlódásvezérlési mechanizmus módosított változata. Képes érzékelni a csomagvesztést, de a torlódás kialakulását még a csomagok elvesztése előtt felismeri. Ennek köszönhetően reaktív helyett proaktív stratégiát alkalmaz. A lassú kezdés („slow-start") TCP mechanizmust módosítja azáltal, hogy minden egyes RTT körbejárási idő után exponenciálisan növeli az ablakméret értékét.

16. Veno: Ez a TCP Reno vezetéknélküli hálózatokra optimalizált változata. A Reno multiplikatív csökkentés mechanizmusát módosítja úgy, hogy a hálózatnak a tapasztalt torlódási szintjéhez igazítja. A Veno algoritmus becsli az összeköttetés állapotát és csomagvesztés esetén osztályozza az állapotot az alapján, hogy hálózat torlódás vagy véletlenszerű ok miatt történt a csomag elvesztése.

17. Westwood/Westwood+: Küldő oldalon módosítja a TCP protokollt. Végpontok közötti sávszélességet állapít meg azáltal, hogy felderíti az esetleges csomagvesztés okát. Valós előnye az átviteli sebesség gyors visszaállítása útján jelenik meg vezetékes és vezeték nélküli vegyes hálózatokban.

18. Yeah (YEt Another Highspead): Ez abba a TCP torlódásvezérlési családba tartozik, amely nagy rendelkezésre álló sávszélességnél használatos. Nagy átviteli ráta mellett fenntartja a pártatlanságot is. Legalább annyira barátságos, mint a Reno, ugyanakkor Reno típusú áramkörökkel is barátságosan működik.

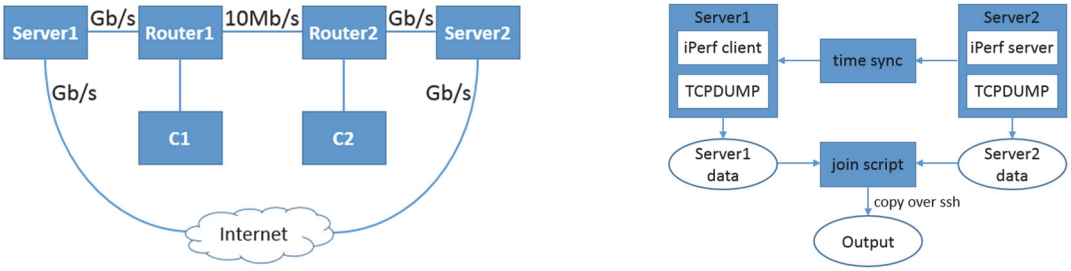
A fentiekben felsorolt TCP változatok esetén joggal fogalmazódik meg a kérdés, hogy egymáshoz képest ezek milyen viszonyban állnak. A válasz megadása nem egyszerű feladat, mivel az objektív számszerűsítés a viszonylag nagyszámú változat által egymástól eltérő tulajdonságokkal rendelkező idősorok megmérése után egy közös metrika meghatározását és annak értelmezését feltételezi.

3. ELEMZÉSI KÖRNYEZET, MÉRÉSI ADATOK

Változtatható számú, egyidejű és azonos kommunikációs útvonalon működő független TCP kapcsolat egymásra hatásának méréséhez egy „dumbbell" (súlyzó) topológiát alkalmaztunk (ld. 2. ábra). A két szerver között hét mérésben k = 1, 2, 5, 10, 20, 50, 100 darab egyidejű TCP kapcsolattal egyetlen

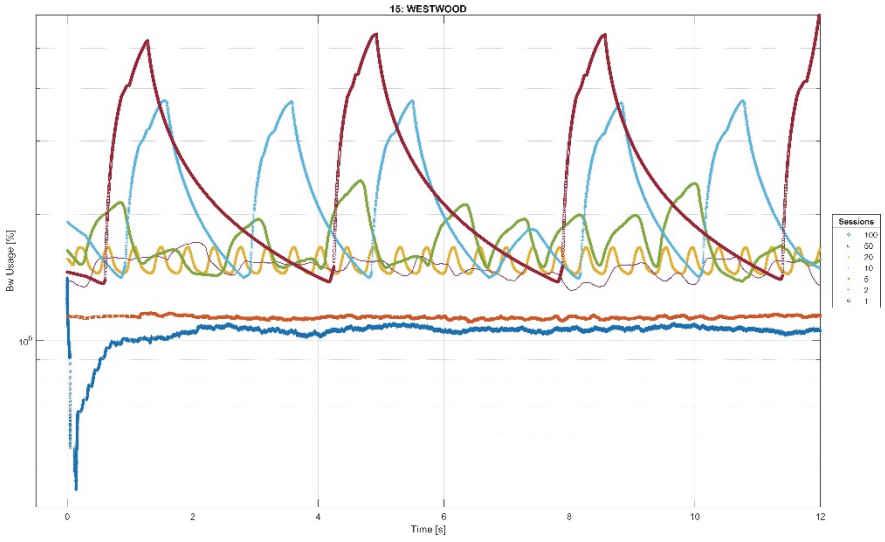
adatfájl letöltési processzét futtattuk. A hét mérés összesen N = 178 elemi adatsort eredményezett, amelyet állapottér idősorként kezeltünk. *tcpdump* applikációra ráfejlesztett szoftverrel úgy a küldőnél, mint a fogadónál Ethernet keretenként mértük a küldési, illetve beérkezési időközöket és az adatkapcsolati elemek méretét. Minden keretet külön válogattunk TCP kapcsolatonként, így az elemi szegmens folyamok is azonosíthatók voltak. Jelen dolgozatban csak a vételi oldalon mért idősorokkal dolgozunk. Adott mérési esetben az egyidejű TCP kapcsolatok azonos implementáció szerintiek.

A mérések során M = 16 darab TCP változatra végeztük el az Ethernet keretek idősorának mintavételezését: 1:BBC, 2:BIC, 3:CDG, 4:CUBIC, 5:DCTCP, 6:HIGHSPEED, 7:HTCP, 8:HYBLA, 9:ILLINOIS, 10:LP, 11:NV, 12:RENO, 13:VEGAS, 14:VENO, 15:WESTWOOD, 16:YEAH. Ez TCP típusonként N = 178 idősor rögzítését jelentette, azaz 16N = 2848 idősort mértünk meg és tároltunk le további feldolgozás céljából. Egyrészt a párhuzamos processzek száma, másrészt az egyes TCP típusok eltérő hatékonysága miatt az elemi TCP kapcsolatok idősorainak hosszúsága egymástól lényegesen különbözik. A neurális hálózatok tanításánál szokásos nulla értékekkel való kitöltés (padding) a bemeneti adatvektorok azonos hosszúságra való hozásához nem alkalmazható jelen esetben, mivel éppen az időbeni mintázatok alapján akarjuk az egyes idősorokat összehasonlítani. A mintázatok közötti hasonlóságokat az egyes TCP implementációk által nyújtott kommunikációs szolgáltatások közötti hasonlóságként értelmezzük.



2. ábra. *A „dumbbell" mérési környezet eszközei (bal), illetve kommunikációs funkciói (jobb)*

Példaként a 3. ábra a Weswood TCP változatnak a különböző kapcsolatszám szerinti idősorait mutatja egyetlen grafikonon. Megfigyelhető, hogy a különböző idősorok eltérő mennyiségű elemet tartalmaznak, de alakra hasonlítanak egymásra. A kapcsolatszám növelésével az átviteli ráta csökken és időben torzul, hiszen az elemi L4 processzre jutó hálózati erőforrás is csökken, miközben a közös útvonal egyidejű használata miatt erősödik ezek egymásra irányuló lassító hatása is.



3. ábra. *Westwood TCP torlódásvezérlési mechanizmus függése az egyidejű kapcsolatok számától*

Nagyszámú parallel kapcsolat esetén az átviteli ráta munkapontja a TCP előzetes „slow-start” mechanizmusa miatt csak két másodperc után áll be (ld. alsó görbe k = 100 kapcsolatra). TCP változattól függően a sávszélesség görbék bizonyos mértékben hasonlíthatnak egymásra, de léteznek nagyon különbözőek is.

4. ELEMZÉSI MÓDSZER ÉS MEGÁLLAPÍTÁSOK

Az egymástól eltérő hosszúságú idősorok mintázatának hasonlóságát dinamikus idővetemítés (DTW - Dynamic Time Warping) módszerével számszerűsítjük. A DTW két tetszőleges hosszúságú idősor optimális módon való illeszkedésének mértékét határozza meg. Ehhez az algoritmus a következő lépéseket alkalmazza:

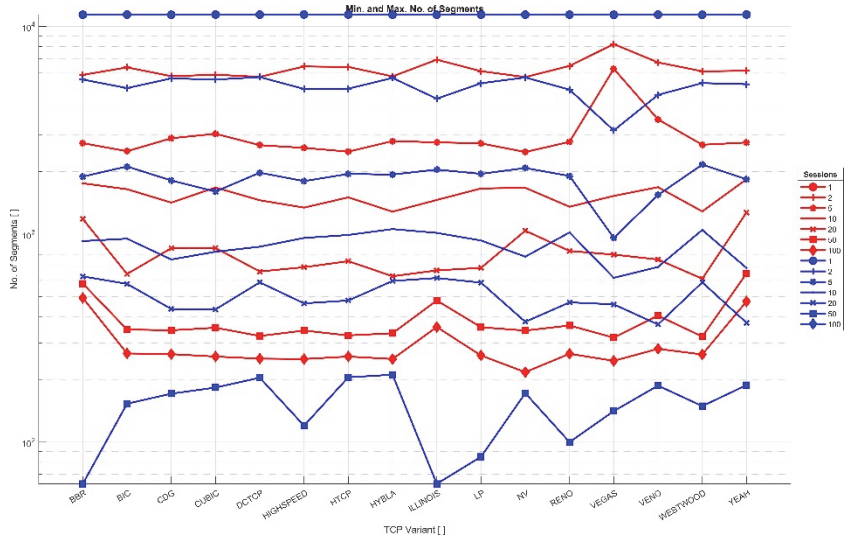
a) Az első sorozat minden egyes indexét a második sorozat egy vagy több indexéhez illeszti, és visszirányban hasonlóan illeszt;
b) A két sorozat első indexei egymásra kötelezően illeszkednek, miközben ezek másik sorozat más indexekre is illeszkedhetnek;
c) A két sorozat utolsó indexei egymásra kötelezően illeszkednek, miközben ezek másik sorozat más indexekre is illeszkedhetnek;
d) Az egyik sorozat indexeinek másik sorozat indexeire illesztése monoton módon történik, azaz ha j > i az első sorozat indexei, akkor a másik sorozatban nem lehet m > n másik két index, amelyeknél i illeszkedik m-re és j illeszkedik n-re. Ugyanez a szabály visszirányban is kötelező.

Optimális illeszkedés azon illesztések eredménye, amikor az illesztett indexekhez tartozó értékek közötti abszolút távolságok összköltsége minimális. Ezzel az illeszkedéssel az időnek bizonyos intervallumonként nemlineáris zsugorítása, valamint megnyújtása történik meg, amit közösen vetemítésnek nevezünk. A DTW két tetszőleges (i, j) idősor közötti hasonlóság megállapítására távolság metrikát kínál fel: dtw(i,j). Ez ugyan teljesíti a szimmetria szabályt de nem teljesíti a háromszög szabályt, azaz:

$$dtw(i,j) = dtw(j,i) \text{ teljesül} \quad (1)$$

$$dtw(i,j) + dtw(j,k) \geq dtw(i,k) \text{ nem teljesül} \quad (2)$$

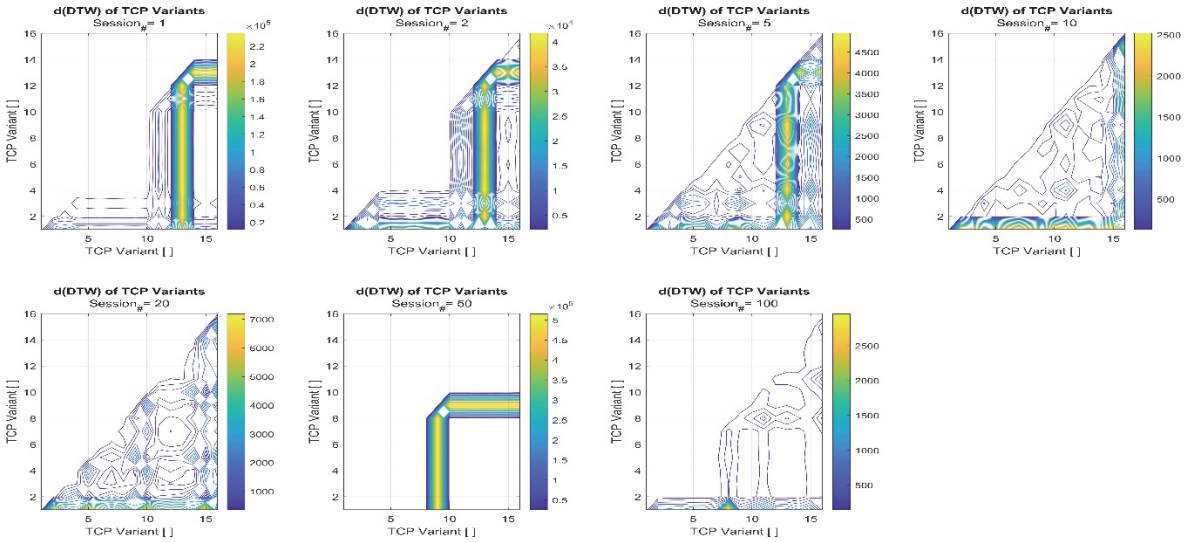
Ugyanazon adatfájlnak különböző egyidejű TCP kapcsolaton a szerverről való letöltését mind a tizenhat féle TCP változaton elvégeztük. Az egyes idősorok elemei számának tartományát a 4. ábra mutatja.



4. ábra. *Elemi TCP idősorok hosszának (min, max) tartománya kapcsolatszám (k) függvényében*

A DTW képes megállapítani az idősor párok közötti hasonlóság mértékét. Ez az egyidejű kapcsolatok számától erőteljesen függ, amit az 5. ábra szemléltet. Azonos kapcsolatszámhoz tartozó, ám

különböző típusú TCP változat által létrehozott idősor párok közötti DTW távolság értékét színkóddal jelöltük. Kék szín alacsony étéket, sárga szín magas értéket képvisel. Fontos megjegyeznünk, hogy a különböző kapcsolatszámok esetén a DTW metrika tartománya jelentős nagyságrendi különbségeket mutat: míg k = 1 esetén 10^5, addig k = 100 esetén csak 10^3 nagyságrendről beszélhetünk.

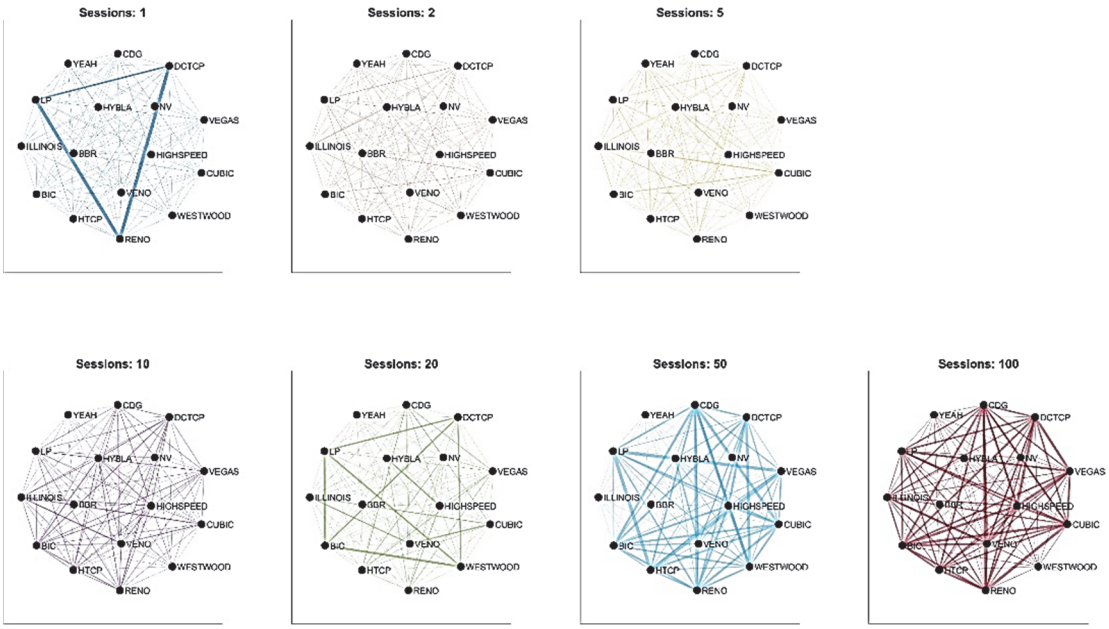


5. ábra. *Elemi TCP idősorok DTW távolságának függése a kapcsolatok számától (k)*

Emiatt a különböző kapcsolatszámokhoz tartozó idősorok közötti hasonlóság számszerűsítéséhez normalizáltuk a DTW távolságokat és annak reciprokát tekintjük különbözőségi metrikának:

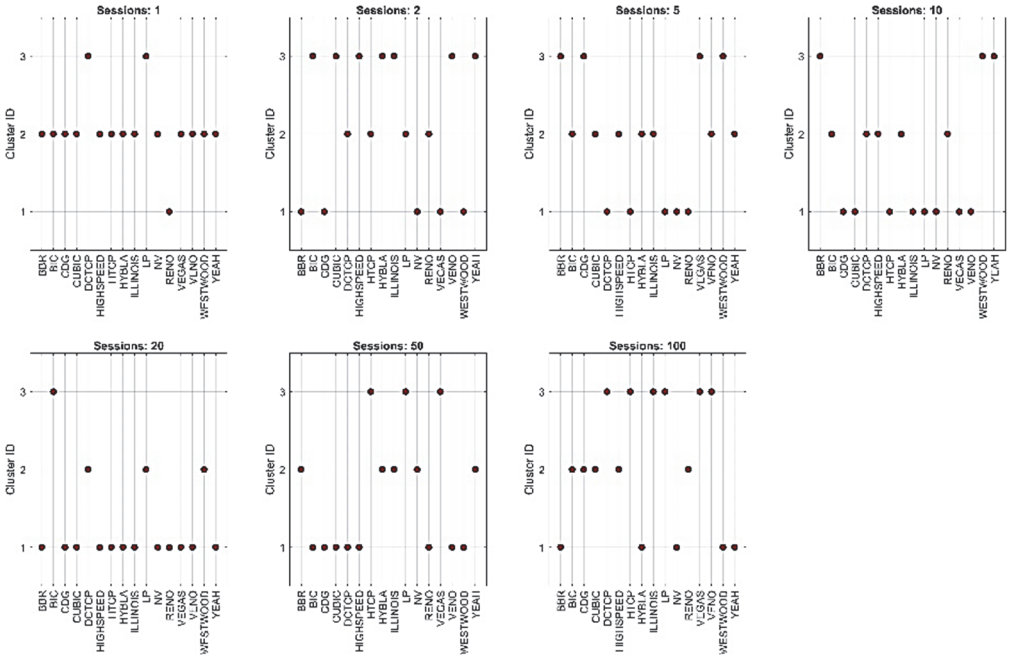
$$D(m,n) = min(dtw(i,j)), \quad i = m_1, m_2, \dots, m_7, \quad j = \text{i} = n_1, n_2, \dots, n_7 \tag{3}$$

Az így létrehozott metrika képes megmutatni látványos formában az egyes TCP változatok közötti különbségeket nem csak azonos kapcsolatszám esetén, hanem különböző kapcsolatszámok között is (ld. 6. ábra). Megfigyelhető, hogy jelentős különbség létezik k = 1 esetén bizonyos TCP változatok között, míg k = 2, 5 esetekre elenyésző a különbség. k = 10, 20 eseteknél egyre több TCP változat különbözik egymástól, míg k = 50, 100 esetekben a különbözőség jelentős.



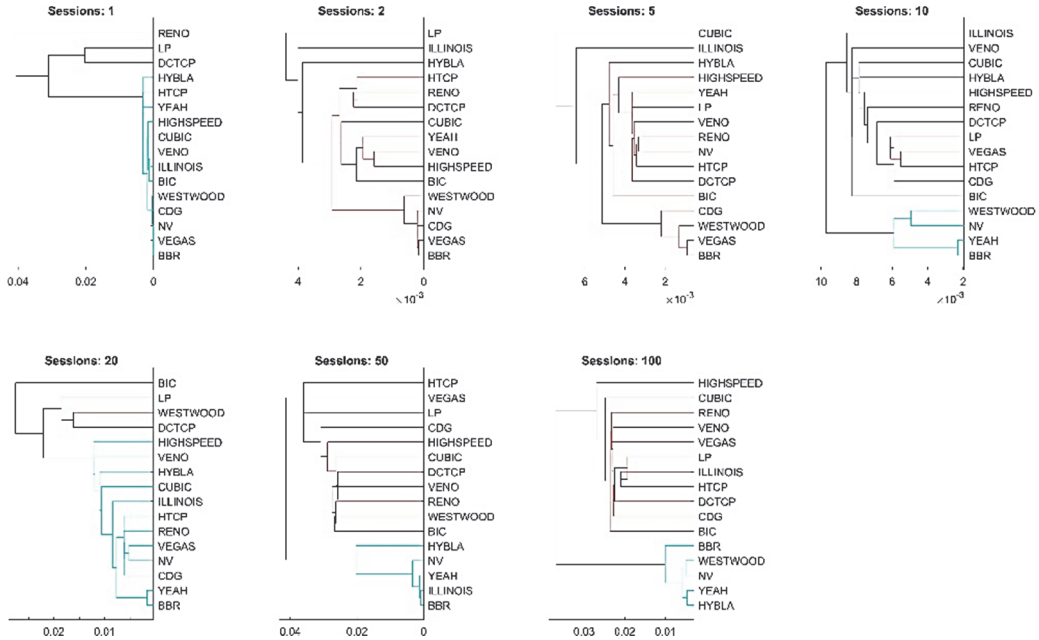
6. ábra. *Elemi TCP idősorok DTW távolságának függése a kapcsolatok számától (k)*

Ahhoz, hogy a sokféle TCP változat sokféle kapcsolatszám közötti hasonlóságokat jobban megértsük, k-Mean algoritmussal klaszterezést alkalmaztunk (ld. 7. ábra). Megállapítjuk, hogy az M = 16 féle TCP változat alapvetően három féle működést mutat, de a hasonló viselkedés erőteljesen függ az egyidejű kapcsolatok számától, azaz k-tól.



7. ábra. *TCP változatok klaszteresődése a kapcsolatok számától (k) függően*

A hasonlóság mértékének módosulását Hierarchikus klaszter algoritmussal faszerkezetben ábrázoltuk a 8. ábrán. Ez alapján kijelenthető, hogy az M = 16 féle TCP változat k = 1 esetén többnyire hasonlóan működik, csupán a Reno, LP, DCTCP tér el a többitől. Féltucat körüli egyidejű kapcsolatszám esetén az Illinois és a Cubic is másként kezeli a torlódást.



8. ábra. *TCP változatok klaszteresődése a kapcsolatok számától (k) függően*

Tíz kapcsolatnál egyértelmű a TCP változatok három fő csoportba való tömörülése: A (Illinois, Veno, Cubic, Hybla, HighSpeed, Reno, DCTCP), B (LP, Vegas, HTCP, CDG) és C (Westwood, NV, Yeah, BBR). Húsz kapcsolat esetén ismét több változat hasonlóan működik, kivéve a BIC, LP,

Westwood és DCTCP változatokat. Ötven kapcsolatnál a három csoport számossága jobban kiegyenlített. Száz egyidejű kapcsolatnál a Highspeed egyéni megoldást alkalmaz, a többi változat két csoportba sorolódik 10:5 = 2:1 számosság arányban.

5. ÖSSZEFOGLALÁS

A dolgozatban a tizenhat féle TCP torlódásvezérlési mechanizmust hasonlítottunk össze alacsony, közepes, illetve nagyszámú és egyidejű kapcsolat esetén, amelyek azonos útvonalon továbbítják az adatforgalmat. Bemutattuk a gyakorlatban leginkább használt TCP változatok speciális tulajdonságait. A hasonlóságok és különbségek számszerű megjelenítéséhez a dinamikus idővetemítés módszerére támaszkodtunk, mivel a 2848 mért idősor egymástól eltérő hosszúságú. Megállapítottuk, hogy a tanulmányozott tizenhat féle TCP mechanizmus erőteljesen függ az egyidejű kapcsolatok számától és alapvetően három féle viselkedésmód szerint működnek. Ezen viselkedésmódok változnak a kapcsolatszám alapján.

A dolgozatban azonos összeköttetés típusú kapcsolatok egymásra gyakorolt hatását vizsgáltuk, amit folytatásként ki lehet bővíteni összeköttetés nélküli kapcsolatokat is működtető heterogén rendszerek vizsgálatával is.

KÖSZÖNETNYILVÁNÍTÁS

Jelen munkát egyrészt a FIKP-20428- 3/2018/FEKUTSTRAT projekt, másrészt az Európai Unió és az Európai Szociális Alap által finanszírozott EFOP-3.6.3-VEKOP-16-2017-00002 projekt támogatta. A mérésekhez a technológiai hátteret a QoS-HPC-IoT Labor biztosította.

IRODALMI HIVATKOZÁSOK

[1] Elgendy N, Elragal A. *Big data analytics: a literature review paper*, In: Perner P, editor. Advances in data mining. applications and theoretical aspects. ICDM 2014. Lecture Notes in Computer Science. vol. 8557. Cham: Springer; 2014.

[2] Lee R, Luo T, Huai Y, Wang F, He Y, Zhang X. *Ysmart: Yet another SQL-to-mapreduce translator*. IEEE International conference on distributed computing systems (ICDCS). 2011. p. 25–36.

[3] Gal Z, Varga I, Tajti T, Kocsis G, Langmajer Z, Kosa M, Panovics J. *Performance evaluation of massively parallel communication sessions*. In: Iványi P, Topping BHV, editors. Proceedings of the sixth international conference on parallel, distributed, GPU and cloud computing for engineering. Stirlingshire, UK: Civil-Comp Press; 2019. p. 34. https://doi.org/10.4203/ccp.112.34.

[4] Zoltan Gal, Gergely Kocsis, Tibor Tajti, Robert Tornai. *Performance evaluation of massively parallel and high speed connectionless vs. connection oriented communication sessions*. Advances in Engineering Software, Elsevier, Advances in Engineering Software 157-158 (2021) 103010, 2021.

[4] Bagnulo M. *Threat analysis for TCP extensions for multipath operation with multiple addresses,* RFC 6181. INTERNET STANDARD; 2011.

[5] Ford A. *Architectural guidelines for multipath TCP development*. RFC 6182. INTERNET STANDARD; 2011.

[6] Huston G. *TCP and BBR*, RIPE 76 meeting. 2018. https://ripe76.ripe.net/presen tations/10-2018-05-15-bbr.pdf(last visited 08.11.2019).

[7] Cardwell N, Cheng Y, Gunn CS, Yeganeh SH, Jacobson V. *BBR: congestion-based congestion control - measuring bottleneck bandwidth and round-trip propagation time*. ACMQueue Netw 2016;14:5.

[8] Allman M, Paxson V, Blanton E. *TCP congestion control*. RFC 5681. 2009.

[9] Floyd S. *Congestion control principle*. ser RFC2914. Internet Engineering TaskForce (IETF); 2000.

[10] Xu L, Harfoush K, Rhee I. *Binary increase congestion control for fast, long distance networks*. IEEE INFOCOM. 2004.

[11] Hayes DA, Armitage G. *Revisiting TCP congestion control using delay gradients*. IFIP Networking. Springer; 2011. p. 328–41.

[12] CUBIC T. *A transport protocol for improving the performance of TCP in long distance high bandwidth cyber-physical systems*. IEEE International Conference on Communications workshops (ICC Workshops). 2018.
[13] Alizadeh M, Greenberg A, Maltz DA, Padhye J, Patel P, Prabhakar B, Sengupta S, Sridharan M. *Data center TCP (DCTCP)*. Proc. ACM SIGCOMM, New Delhi. Data Center Networks session; 2010.
[14] Alizadeh M, Javanmard A, Prabhakar B. *Analysis of DCTCP: stability, convergence, and fairness*. Proc ACM SIGMETRICS, San Jose. 2011.
[15] Floyd S. *Highspeed TCP for large congestion windows*. RFC. INTERNET STANDARD; 2003.
[16] Floyd S, Ratnasamy S, Shenker S. *Modifying TCP's congestion control for high speeds*. Technical note. 2002.
[17] Shorten RN, Leith DJ. *H-TCP: TCP for high-speed and long-distance networks*. Proc. PFLDnet, Argonne. 2004.
[18] Caini C, Firrincieli R. *TCP-hybla: a TCP enhancement for heterogeneous networks*. Int J Satellite Communication 2004.
[19] Liu S, Basar T, Srikant R. *TCP-illinois: a loss and delay-based congestion control algorithm for high-speed networks*. ScienceDirect Performance Evaluation, 2008;65:417–40.
[20] Kuzmanovic A, Knightly EW. *TCP-LP: a distributed algorithm for low priority data transfer*. IEEE INFOCOM. 2003.
[21] Fu CP, Liew SC. *TCP veno: TCP enhancement for transmission over wireless access networks*. IEEE Journal Selected Areas of Communication 2003.
[22] Mascolo S, Casetti CE, Gerla M, Sanadidi MY, Wang R. *TCP westwood: Bandwidth estimation for enhanced transport over wireless links*. MobiCom; 2001.
[23] Baiocchi A, Castellani AP, Vacirca F. *YeAH-TCP: Yet another highspeed TCP*. CiteSeerX; 2008.
[24] Meister BW, Janson PA, Svobodova L. *Connection-oriented versus connectionless protocols: a performance study*. IEEE Transactions on Computers 1985:1164–73.C34/12
[25] Postel J. *ISI, user datagram protocol*. RFC 768. INTERNET STANDARD; 1980.