BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2018
http://excel.fit.vutbr.cz

Monitorovanie peerov BitTorrent na základe informácií z distribuovanej hašovacej tabuľky

Martin Vaško*

Abstrakt
Cieľom tejto práce je monitorovanie uzlov vrámci siete BitTorrent. Záujem je zameraný hlavne na to aby bolo možné monitorovanie jednotlivca tak i celej skupiny ľudí, ktorý aktuálne sťahujú a tým pádom zdieľajú digitálny obsah medzi sebou. Pozornosť sa upiera na distribuované hašovacie tabuľky, ktoré majú byť základným kameňom monitorovania a vyhľadávania uzlov v sieti. V poslednej dekáde sa pozornosť upriamila aj na to, ako efektívne dokážeme v tejto sieti monitorovať. Existujúce algoritmy nijak nezaisťujú aktuálnu efektivitu a preto sa odporúča zaviesť odhad presnosti pomocou Bernoulliho procesu [1]. Takéto monitorovanie nie je až také zložité, pretože plánujeme prehľadávať do hĺbky. Z meraní vyplýva, že je prehľadávanie do hĺbky lepšie z krátkodobého hľadiska. Je to v dôsledku toho, že množstvo peerov sa nachádza blízko danému infohešu. Všetky získané informácie sú naviac skreslené tým, že mnoho uzlov je za NAT-om [2] resp. firewallom [3] a náš dopyt vôbec nedostane. Práca sa bude venovať efektivite monitorovania v čase, percentuálnemu podielu medzi uzlami a peerami v distruovanom systéme a odhadu chyby prehľadávania.

Kľúčové slová: BitTorrent — Monitorovanie — Distribuovaná Hešovacia Tabuľka

Priložené materiály: Demonstration Video — Downloadable PostProcess Code

*xvasko12@.stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Úvod

Problematika sťahovania nelegálneho digitálneho obsau alebo zaberanie väčšiny šírky pásma v sieti. Takéto chovanie na internete má za dopad preťaženie siete v dôsledku používania UDP nespojovanej služby, ktorá nekontroluje koľko šírky pásma zaberá. Samozrejme, že sa používaniu UDP služby zabrániť nedá, no chceme mať čo najlepší prehľad o tom koľko ľudí je bežne súčasťou BitTorrent sieti. Účastníci v sieti BitTorrent si medzi sebou zdieľajú obsah či už legálny alebo ilegálny. V tejto práci sa budem zaoberať BitTorrent špecifikáciou s rozšírením o DHT (Distribuovaná hešovacia tabuľka), ktorá ma mnoho kladných vlastností. Kademlia je konkrétna implementácia týchto DHT tabuliek, ktorá sa stala podstatou všetkých aplikácií pre sťahovanie torrentov.

Zmyslom práce je pochopiť princíp funkcie DHT tabuliek v BitTorrent sieti a ako ju využiť pre monitorovanie. Výsledkom práce bude monitorovanie siete nad rôznymi torrent súbormi s rôznými prístupmi ako je napr. použitie LIFO fronty namiesto FIFO alebo získanie podielu medzi peermi a uzlami.

V práci si najprv ujasníme ciele práce tj. čo práca chce autorovi povedať a čím sa budeme zaoberať. Následne v sekcií 2 si vysvetlíme základne pojmy a bude-

me sa venovať tomu, ako sa obsluhujú uzly v DHT. Zmienime aj existujúce prístupy a metódy pre vyhľadávanie v DHT sieti. Kapitola 8 sa bude zaoberať získanými výsledkami a diskusiou o nich. V závere si zhrnieme všetky dôležité pojmy a závery, ktoré táto práca prezentuje.

1.1 Cieľ práce

Prvým milníkom, ktorý práca rieši je efektívnosť vyhľadávania. Tá sa nedá presne určiť iba za pomoci **Bernoulliho procesu**, ktorého podstatou sa budeme zaoberať v sekcií 2. Určenie presnej efektívnosti je nemožné. Vyplýva to z faktu, že nedokážeme povedať, koľko uzlov sme vrámci jedného priestoru vynechali, pretože priestor nemusí byť plne obsadený identifikátormi (identifikátor uzla môže byť veľmi blízko cieľovému identifikátoru). Konzistentné snímky systému sú kvôli neustálemu odpájaniu a pripájaniu uzlov, a kvôli krátkemu časovému kvantu, malé. Najväčšia časová jednotka pre monitorovanie je 15 minút, pretože po dlhšom prehľadávaní je snímok uzlov nutné skontaktovať, či uzol ešte žije. Preto je potrebné rozlišovať počet aktívnych uzlov v snímke a počet neaktívnych uzlov.

2. Vysvetlenie pojmov

V tejto práci je nutné vysvetliť dva pojmy aby sa predišlo nejasnostiam:

- **Peer** je klient/server načúvajúci na TCP/μTP porte, ktorý implementuje BitTorrent resp. Micro Transport protokol.
- **Uzol** (ang. node) je klient/server načúvajúci na UDP porte implementujúci chovanie distribuovanej hašovacej tabuľky.

2.1 Distribuovaná hešovacia tabuľka

BitTorrent používa DHT na uloženie kontaktných informácií o peerovi a uzloch pre *tracker-less torrent*. V tomto prípade sa každý peer stáva trackerom. Protokol je postavený na Kademlia [4] a je implementovaný cez UDP. Kademlia obsahuje vedrá, ktoré nám hovoria o tom do koľkých častí je rozdelená **smerovacia tabuľka uzlu**. Prázdna smerovacia tabuľka má 1 vedro. Každé takéto vedro zároveň dokáže udržať K uzlov. Kademlia vyhľadáva uzly s pomocou XOR metriky. Nad dvoma infohešmi spočíta XOR aby dokázala zistiť ich vzdialenosť. Takto dokáže určiť najkratšiu cestu k cieľu.

2.2 Infoheš

Každý uzol má globálne jedinečný identifikátor tiež známy ako identifikátor uzlu (node ID). Identifikátory uzlu sú vyberané náhodne z rovnakého 160-bitového priestoru ako BitTorrent **infoheš**. Tento infoheš je zhodný **s kľúčom Kademlia**(pre vyhľadávanie v DHT sa používa termín kľúč, pre BitTorrent infoheš). Pre pripájanie ku konkrétnym súborom je potrebné vedieť infoheš súboru, ktorý je ukrytý v súbore .torrent alebo v magnet-odkaze. Infoheš je tvorený pomocou **SHA1** hešovacím algoritmom s vysokou mierou entropie, aby bol zaistený **jedinečný infoheš** pre daný súbor.

2.3 Churn

Nezávislý príchod a odchod peerov sa nazýva v terminológii distribuovaných systémov churn (vír). Vír je hlavnou výzvou pre peer-to-peer systémy, pretože ovplyvňuje stabilitu celého systému a dostupnosť peerov a zdieľaných objektov. Taktiež vír sťažuje detegovanie peerov. Aby sme získali konzistentný snímok systému v danom okamihu, je potrebné vykonať danú detekciu v čo najmenšom čase [5].

2.4 Bernoulliho proces

Predpokladáme, že náš prehľadávač bude vždy vynechávať niektoré infoheše. V tom prípade môžeme modelovať vzorkovací proces ako Bernolliho proces[1]. To znamená že pre každý identifikátor v zóne máme dve možnosti:

1. ID je vybrané. Bernoulliho proces zvýši počet výskytov v zóne o 1.
2. ID chýba. Bernoulliho proces zaznamenal stratu v zóne. Nezvýši sa počet výskytov.

Pravdepodobnosť výberu označme ako p. Pravdepodobnosť chýb je potom $1-p$, takže všetko čo potrebujeme zistiť je presný odhad počtu uzlov v zóne, čiže pravdepodobnosť p. Ak máme lepší prehľad o pravdepodobnosti $1-p$, teda o chýbajúcich uzloch tak si vieme spätne vyjadriť pravdepodobnosť p. Tieto problémy sa týkajú hlavne prehľadávania celého priestoru DHT.

3. Existujúce metódy

BitTorrent protokol s DHT je veľmi rozšírený a obsahuje viacero implementácií. Tieto implmentácie sa odlišujú v metódach výroby infohešu, prehľadávania stavového priestoru a iné. Pre túto prácu sa vybrala metóda MLDHT, ktorá má najviac uzlov aby sme dokázali overiť správnosť implementácie a funkciu DHT.

[1] https://math.tutorvista.com/statistics/bernoulli-process.html

4. Metóda pre detekciu peerov v systéme MLDHT

Metóda zmienená od pána Wang Liang [1], sa spolieha aj na to, že identifikátory sú rovnomerne rozložené v MLDHT vedrách. Principiálne MLDHT protokol by mal garantovať rovnomerné rozloženie ID uzlov [1]. Podstatou metódy je správne zvolenie zóny, v ktorej budeme detegovať peerov.

Pri monitorovaní malých sietí, by sme radi chceli dostať všetkých peerov zdieľajúcich torrenty. Takéto monitorovanie má tiež svoju nevýhodu a to je problém *missing node-issue* v každom prechode prehľadávača. Ak si zvolíme veľmi malú zónu (veľké K), mierne kolísanie spôsobené chýbajúcimi uzlami sa po zväčšení zóny zvýši, čo vedie k významnému (a náhodnému) rozdielu v počte peerov (oproti variante s nižším K). Obrovské kolísanie môže vyčerpať detekciu, ak máme veľkú sadu rovnakých vzoriek. Ďalšie nežiadúce účinky na kolísanie môžu byť firewally, preťaženie liniek, abnormálne správanie peerov a pod.

4.1 Súhrn MLDHT

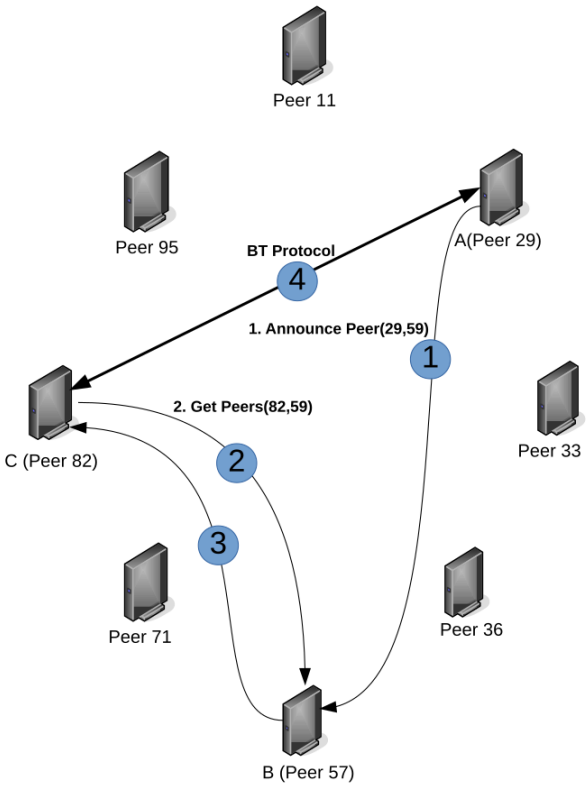
Ak chceme využiť čo najlepšie prehľadávanie v MLDHT, mali by sme použiť zónu o veľkosti 9 až 14 bitov [1]. Ak vyhľadávame konkrétnych peerov túto skutočnosť môžeme zanedbať. Pre určenie presnosti prehľadávania sa chceme vyhnúť problému s **missing node-issue** [1] . Tento missing-node pojem zahŕňa viacero faktorov.

- Pohybovanie siete: príchod a odchod uzlov. Tento problém sa sťažuje s dĺžkou prehliadania.
- Strata správ, kvôli preťaženiu siete.
- Rôzne protekčné mechanizmy: napr. čierna listina, zákaz komunikácie podozrivých uzlov, zahadzovanie deformovaných (malformed) správ, malá veľkosť vedra, malá veľkosť fronty.
- Neaktuálne informácie v smerovacej tabuľke, hlavne kvôli implementačným detailom Kademlia.
- Prehľadávač nie je dosť efektívny z hľadiska rýchlosti a výberovosti uzlov (musíme prehliadač obmedziť aby si nebral uzly, ktoré vyzerajú podozrivo).
- Problémy s firewallom.

Týmto faktorom sa musíme vyhnúť ak chceme prehľadávať iba bezpečné uzly. Cieľom práce je nezamedzovať týmto faktorom, aby sme vedeli aké robustné dokáže byť prehľadávanie bez obmedzení.

5. Obsluha uzlov pri zdieľaní

Obrázok 1 zobrazuje bežnú obsluhu v Kademlií (DHT). Predpokladáme, že máme 3 uzly A, B a C. A má súbor s infoheš hodnotou $x = 59$. Povedzme, že uzol B je zodpovedný za ukladanie hodnoty x, pretože je najbližšie infohešu 59. Uzol C chce stiahnuť tento súbor [6].



Obrázok 1. Bežná obsluha uzlov

1. A chce súbor uverejniť uložením infohešu x na uzol B. A v tomto prípade zavolá procedúru `GET_PEERS` opakovane dopytovaním uzlov zo súboru .torrent. Postupne sa tak dostane bližšie k B, až kým ho nedosiahne (vysvetlené v bode 5). Vtedy A použije procedúru `ANNOUNCE_PEER` aby ohlásil možnosť stiahnutia (zdieľania) súboru s infohešom x uzlu B. B si uloží kontaktné informácie o A do príslušnej peer množiny pre x. Keďže uzol A je jediný vydavateľ infohešu x v tejto množine sa nachádza sám.[6]
   Keď A pošle znova `GET_PEERS` správu, môžu sa vyskytnúť tieto dve rôzne situácie. Ak uzol dopytu už pozná daný infoheš a uloží peerov do korešpondujúcej množiny, odpoveď ou bude množina peerov.
2. Ak chce uzol C stiahnuť súbor musí najprv dostať x. Platí preň to isté, čo platilo pre A, a to použitie `GET_PEERS` aby sa dostalo k B.
3. Keďže B má už uloženú množinu peerov pre x, C môže získať počiatočnú množinu peerov od B.

4. *C* sa prihlási do roja užívateľov, nastaví TCP (dnes μTP) spojenie s peermi v množine a získa metadáta od ostatných peerov používajúcich BitTorrent [7, 8]. Týmto momentom a začína proces sťahovania [6].

(Bod 5) Ak uzol nepozná infoheš, odpovedá s *K* najbližšími uzlami k hľadanému infohešu z jeho smerovacej tabuľky. Takýmto spôsobom sa bod *X* dostane bližšie a bližšie k bodu *Y* až úplne dosiahne bod *Y* [6].

6. Riešenie a vlastná metóda

Pre monitorovanie peerov som po dohode s vedúcim práce vybral systém MLDHT (a jemu odvodené systémy). Jeho popularita a už sprostredkovaný výskum ponúkol možnosť monitorovať efektívne napriek nepriaznivým javom. Ak skombinujeme Bernoulliho proces ako bolo doporučené v kapitole o MLDHT 4, dokážeme monitorovať o niečo presnejšie ako doteraz bolo možné. Naším cieľom v tejto práci je načrtnúť vstupy a výstupy programu, jeho architektúru a testovaciu sadu. Predpokladom implementácie bude použitie schránok a implementovanie komunikácie DHT uzlov v BitTorrent sieti. Na túto implementáciu využijem jazyk python, ktorý má kvalitné knižnice pre zostavovanie správ protokolu Kademlia. XOR metriku využívame iba pri uzloch, ktoré sú veľmi blízko vyhľadávanému infohešu. Akonáhle sme pri ňom podstatne blízko je vyššia pravdepodobnosť, že väčšina peerov sa bude nachádzať pri ňom. Taktiež celý prístup pri prehľadávaní infohešov je kombinovaný s Bernoulliho procesom. Našou úlohou je monitorovanie peerov pomocou DHT a na to sme zvolili dopyt get_peers.

6.1 Podrobnosti o metóde

Metóda spočíva v tom, že sa snažíme o overenie funkcie DHT. Začíname na verejne prístupnom uzle. Od tohto uzlu sa snažíme cyklicky posúvať v priestore a pýtať sa uzlov v ceste na daný torrent. Takto dokážeme overiť funkciu DHT. Túto metódu som implementoval s 2 prístupmi. Prvým z nich je prehľadávanie do šírky a druhým prehľadávanie do hĺbky.

Prehľadávanie do hĺbky sa snaží nájsť cieľový infoheš a potom sa postupne vynárať zo zanoreného priestoru. Opakom je prehľadávanie do šírky, ktorým prehľadávame všetky uzly, ktoré dostaneme ako odpoveď od uzlu v poradí ako nám ho poslal. Takto dostávame peknú snímku o sieti no nevýhodou je časová náročnosť.

Tieto dva prístupy sa v kapitole o testovaní budem snažiť porovnať a odôvodniť získané výsledky. Všetko sa koná navyše v súlade s Bernoulliho procesom.

7. Nevýhody implementácie aktívneho monitorovania

Problémom implementácie je pomerne veľká záťaž prenosového pásma, ktoré by mohlo byť využité inými prospešnými aplikáciami. Záťaž je hlavne z dôvodu neustáleho dopytovania sa, prijímania odpovede a rozčleňovania sekcií v odpovedi. Takouto réžiou je zaťažený procesor po celu dobu behu.

Pre túto réžiu som odľahčil nástroj od akejkoľvek kontroly blacklistu, filtrovanie malformed správ a podobných techník, čím som zamedzil akejkoľvek ďalšej réžií. Transformáciu IP adresy na informácie typu whois, geolokácie a dns reverzného záznamu sa transformujú pomocou externého nástroja, ktorý sme vytvorili k tomuto projektu. Celá táto transformácia je asi 2-krát taká dlhá ako prehľadávanie (20 minút). Preto sa celá transformácia vykonáva ako osobitný proces.

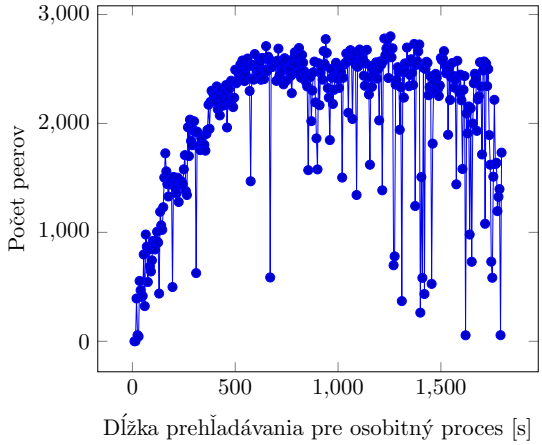
Nevýhodou je taktiež churn efekt. Neustále pripájanie a odpájanie uzlov sťažuje odhad dĺžky prehľadávania. Ak nástroj pustíme 100 krát a každý z týchto spustení bude prehľadávať o sekundu dlhšie ako predošlé spustenie, tak počet peerov je kolísavý. Čím novší torrent súbor máme na vstupe, tým fluktuácia peerov je vyššia. Naproti tomu pustením 100 behov tohto prehľadávača s jedným časovým obmedzením, ponúka lepšie výlsedky, no je výpočetne náročnejší na tvorbu grafu. Preto boli časové obmedzenia zmenšené na približne 140 sekúnd.

8. Priebeh merania a overenie implementácie

Meranie prebiehalo nad rôznymi torrent súbormi, z ktorých sme vytiahli informácie iba o infoheši. Chceli sme zistiť nakoľko dobre funguje DHT, ktorá nás má postupne previesť celým priestorom k správnemu infohešu. Monitorovanie som začal z verejne prístupného uzla dht.transmissionbt.com, na ktorý som poslal prvý dopyt typu get_peers.

Celý priebeh bol spustený z Kolejnetu na infoheš súboru *The Greatest Showman 2017 720p BluRay HEVC x265-RMTeam*. Dňa 23.4.2018 o 17:10 mal niečo okolo 2800 peerov, ktoré ho zdieľali. Takto som sa postupne v celom priestore snažil dohľadať čo najviac uzlov, ktoré obsahujú informácie o peeroch. V poslednom kroku som všetkých získaných peerov skontaktoval kvôli faktu, že sa neustále odpájajú a pripájajú aby som našiel všetkých aktívnych peerov v daný moment. v grafoch sú zobrazené osobitné behy aplikácie. Každý nový beh je s oneskorením 5 sekúnd. Začína sa na 10 sekundách, pretože príliš krátke behy nám neprinášajú veľa výsledkov. V grafoch sú behy medzi 10 až 1800 sekundou. V nich sa zameriavam na

čas, ktorý by mal byť optimálny na beh aby sme našli aspoň 90% peerov.



Obrázok 2. Torrent The Greatest Showman 2017 720p BluRay HEVC x265-RMTeam, LIFO fronta, štart 17:10, fluktuácia peerov.

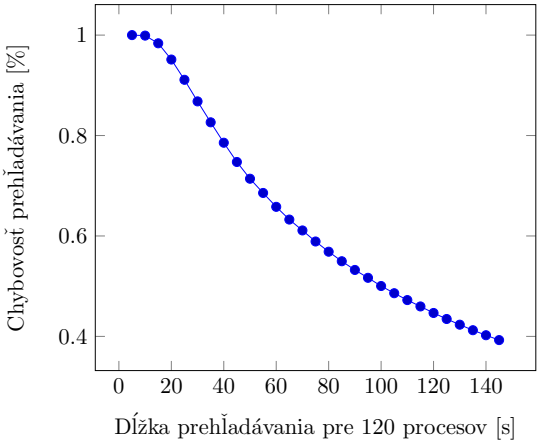
Z grafu 2 je vidno počiatočnú fluktuáciu na novom obľúbenom torrente. Časom sa táto fluktuácia ustáli a počet nových peerov stále pribúda. Neustálym pripájaním a odpájaním sa v grafe ťažko odhaduje ďalší priebeh. Preto je vhodné monitorovať jeden torrent dlhšiu časovú dobu. Z takého grafu je potom vidno akú mal torrent obľúbenosť, koľko ľudí ho priemerne cez týždeň sťahovalo a podobne. Pre porovnanie sem prikladám graf 4 na ten istý torrent v hodinovom rozmedzí ale s použitím FIFO fronty. Je vidno ako sa postupným prehliadaváním priestoru dostávame k požadovanému výsledku.

Graf na obrázku 3 znázorňuje odhad chyby prehliadávania. Začíname pri 100% chybovosti prehľadávania, kde sa postupným dopytovaním dostávame k menšej chybovosti v pomerne krátkom časovom horizonte. Pomocou LIFO fronty prehľadávame priestor do hĺbky a preto máme veľmi priaznivé výsledky pre vyhľadávanie peerov za krátky čas. Oproti grafu 5 je vidno účinnosť prehľadávania pri použití LIFO fronty.

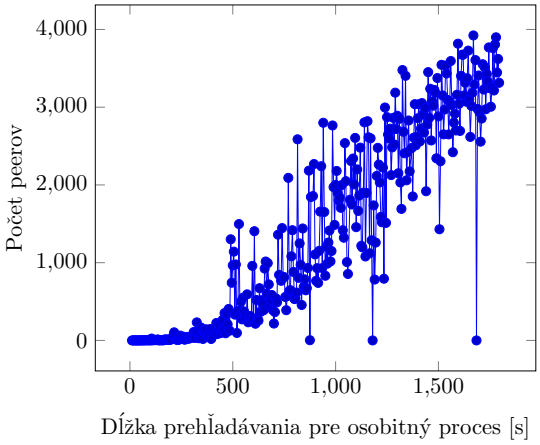
V posledných grafoch 6 a 7 budeme porovnávať percentuálny podiel medzi peermi a nájdenými uzlami aby sme videli rozdiel medzi prehľadávaním do hĺbky a prehľadávaním do šírky. Použil som agregované dáta z predošlých grafov aby nebol graf príliš nečitateľný. Jasne vidíme, že počet peerov pre LIFO je v začiatku až 8 až 16 krát vyšší než počet uzlov.

9. Zhrnutie

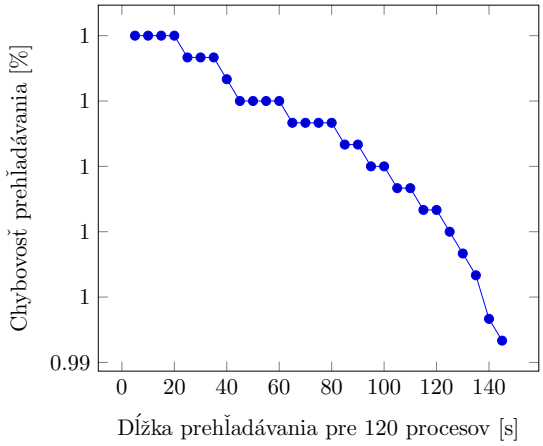
Táto práca sa venovala efektivite a metódam pre vyhľadávanie peerov pre konkrétny torrent. Cieľom bolo popísanie problémov pri prehľadávaní, pochopenie XOR metriky, ktorá je podstatou rýchleho vyhľadavania



Obrázok 3. Torrent The Greatest Showman 2017 720p BluRay HEVC x265-RMTeam, LIFO fronta, štart 18:30, chybovosť prehliadania v časovom horizonte.

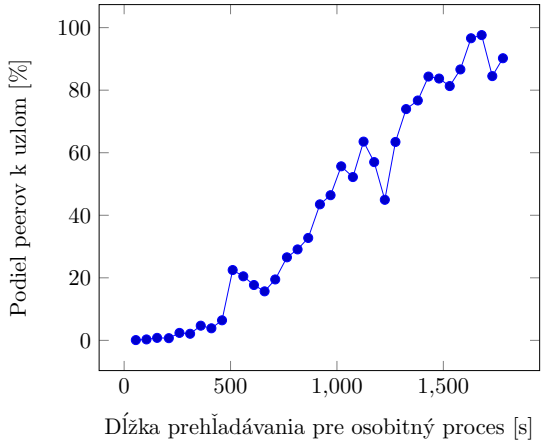


Obrázok 4. Torrent The Greatest Showman 2017 720p BluRay HEVC x265-RMTeam, FIFO fronta, štart 17:50, počet peerov.

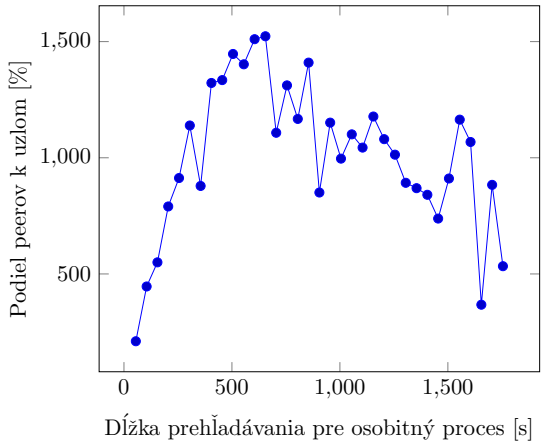


Obrázok 5. Torrent The Greatest Showman 2017 720p BluRay HEVC x265-RMTeam, FIFO fronta, štart 18:30, chybovosť prehliadania v časovom horizonte.

uzlov v sieti. Pri prehľadávaní je každý prechod ovplyvnený vírom. Problemom je to ak je na vstupe torrent s vysokým počtom peerov. V tom prípade je



Obrázok 6. Torrent The Greatest Showman 2017 720p BluRay HEVC x265-RMTeam, FIFO, Percentuálny podiel peerov k uzlom v čase 10-1800 s agregáciou po 40 sekundách



Obrázok 7. Torrent The Greatest Showman 2017 720p BluRay HEVC x265-RMTeam, LIFO, Percentuálny podiel peerov k uzlom v čase 10-1800 s agregáciou po 40 sekundách

to náročnejšie z hľadiska získavania konzistentných snímkov. Pri prehľadávaní musíme vždy vediet konkrétny infoheš torrentu aby sme boli schopní vyhľadať peerov. Bez konkrétneho infohešu nedokážeme vopred určiť peerov, no dokážeme vyhľadávať uzly. Bernoulliho proces bol nápomocný v lepšom prehľadávaní stavového priestoru uzlov. Zisťoval nám koľko percent infohešov sme vynechali.

9.1 Výsledky práce

Z práce vyplýva, že je vhodnejšie použiť LIFO frontu (graf 2), pretože potrebujeme menej času, čo nám zaisťuje v krátkom čase konzistentnosť snímky. Percentuálny pomer získaný z predošlej sekcie jasne popisuje výhodu LIFO fronty v krátkom časovom horizonte. Monitorovanie v takto veľkej sieti je náročne, pretože neustále odpájanie a pripájanie sťažuje toto vyhľadávanie. Podstatou je to, že snímok sa dá považovať za relevantný iba v danom časovom okamihu. Často sa stáva že sa do roja pripoja nový užívatelia a graf už pre nás nemá význam. Preto sa snažia výsledky porovnávať prístupy a zisťovať optimálne nastavenie monitorovania.

Nevýhoda práce s LIFO frontou na torrente, ktorý ma veľa peerov je to, že sa prehliadanie môže vyčerpať, pretože na všetky dotazy typu get_peers dostávame iba peerov a nie uzly. Riešením je zaviesť pravdepodobne ešte dotaz typu find_node na vyhľadanie uzlov pri vyčerpaní, alebo novú frontu, ktorá bude obsahovať peerov.

Čím je vír väčší, tým je ťažší odhad optimálnej dĺžky prehľadávania. Z dosiahnutých výsledkov vidíme ze s LIFO frontou dosahujeme oveľa rýchlejšiu konvergenciu k výsledku, avšak pomocou FIFO fronty lepší prehľad o DHT priestore a o tom v akej hĺbke sa nachádzajú informácie.

Literatúra

[1] Wang Liang and Kangasharju Jussi. *Measuring Large-Scale Distributed Systems: Case of BitTorrent Mainline DHT*. University of Helskinki, Finland, 2013.

[2] M. Holdrege P. Srisuresh. IP Network Address Translator (NAT) Terminology and Considerations, 1999. [Online; navštevené 3.4.2018].

[3] N. Freed. Behavior of and Requirements for Internet Firewalls, 2000. [Online; navštevené 3.4.2018].

[4] Andrew Loewenstern and Avrid Norberg. Bep 5: DHT protocol, 2017. [Online; navštevené 6.10.2017].

[5] D. Stutzbach and R. Rejaie. Understanding churn in peer-to-peer networks. [Online; navštevené 12.12.2017].

[6] Liang Wang and Jussi Kangasharju. Real-world sybil attacks in BitTorrent mainline DHT. In *Global Communications Conference (GLOBECOM), 2012 IEEE*, pages 826–832. IEEE, 2012.

[7] A. N. Greg Hazel. Bep 9: Extension for peers to send metadata files, 2008. [Online; navštevené 6.10.2017].

[8] Ludvig Strigeus G. H. Arvid Norberg. Bep 10: Extension protocol, 2008. [Online; navštevené 6.10.2017].