

Forgalom igény tudatos hálózat tervezés minimális torlódással és úthosszal

Tudáskezelő rendszerek II. labor összefoglaló

Szecsődi Imre

2019

Tartalomjegyzék

1. Bevezetés	2
1.1. Motiváció	2
1.1.1. Hálózat tervezési stratégiák	2
1.1.2. Adattárházak hálózati felépítése	3
1.1.3. Újrakonfigurálás megvalósítása	3
1.2. Labor célja	3
1.3. Laborban megvalósított munka	3
2. Modell	4
2.1. Forgalom igény tudatos hálózat tervezés probléma	4
2.2. Formális felírás	4
2.2.1. Torlódás	5
2.2.2. Úthossz	5
2.2.3. Skálázhatóság	5
2.2.4. Optimális torlódás	5
2.2.5. Optimális úthossz	5
2.3. cl-DAN hálózat tervezése	5
2.4. EgoTree	6
2.5. $EgoTree(s, \bar{p}, \Delta)$ algoritmus	6
2.5.1. Algoritmus elemzése	7
2.5.2. Longest Processing Time (LPT)	7
2.6. cl-DAN algoritmus	7
3. Megvalósítás	9
4. Teszt eredmények	10
5. Összefoglalás	11
6. Irodalomjegyzék	12

1. fejezet

Bevezetés

A labor munka a Demand-Aware Network Design with Minimal Congestion and Route Lengths [3] cikk alapján készült.

1.1. Motiváció

- A technika előrehaladásával egyre nagyobb lett a feldolgozandó adatok mennyisége
- Adattárházakban a szerverek közötti kommunikáció is ezáltal megnövekedett
- A jelenlegi hálózatok a legrosszabb esetre vannak tervezve, azaz, hogy majdnem teljes sávszélességű, kétirányú kapcsolat álljon fent bármelyik két szerver között
- A valós kommunikáció nem ezt a sémát követi, hanem túlnyomó részt megadott párok között történik a legtöbb kommunikáció

Microsoft Research ProjectORe [4].

- Nézzünk meg pár valós példát, Microsoft adattárházában 250 ezer szervert 5 production klaszterben elosztva

1.1.1. Hálózat tervezési stratégiák

- A technika fejlődésével elérhetővé váltak eszközök arra, hogy egy adott hálózatot újra konfiguráljunk, attól függően milyen terhelés éri
 - pl, korábbi kommunikációs minták alapján
- Két fő optimalizációs lehetőség van, legyen rövid az út (a) vagy legyen minimális a torlódás (b)

- A cikk bemutat egy módszert arra, hogy lehet mindkettőre majdnem optimális megoldást adni egyszerre (c)

1.1.2. Adattárházak hálózati felépítése

- Core switch
- Aggregation Swiches
- Top of Rack Switches
- In-Rack Switches

1.1.3. Újrakonfigurálás megvalósítása

- Átlag hálózatok statikusan vannak konfigurálva, nem sok lehetőséget adva annak, hogy változtassunk
 - pl. Ethernet switchek
- Optikai switchek már újra tudják konfigurálni magukat, de ezek "lassúak"
- Microsoft Research - ProjecToR[4], lézer segítségével kiváltani az optikai switcheket
 - 12 μs váltás idő (2500x gyorsabb mint egy optikai hálózati switch)

1.2. Labor célja

A labor célja a cikkben[3] bemutatott algoritmus implementálása, és annak alkalmazása különböző véletlenszerűen generált gráfokra. A kapott eredményeket össze lehet hasonlítani a megadott elméleti korlátokkal.

1.3. Laborban megvalósított munka

A labor ideje alatt elkészült egy keretrendszer, ami segítségével tesztelhető a szerzők által felvázolt algoritmus. A keretrendszer Python [2] nyelven íródott. Egy véletlen gráfok generálására egy külső csomag lett használva [1]

2. fejezet

Modell

2.1. Forgalom igény tudatos hálózat tervezés probléma

- Vegyünk egy hálózatot meghatározott számú csomóponttal
- A hálózathoz tartozik egy demand mátrix, ami leírja a valószínűségét annak, hogy i forrásból mekkora eséllyel lesz adat küldve j célba
- A cél, hogy ezen adatból egy olyan hálózati séma készítése, ami kis torlódást és rövid utakat eredményez, ez mellett még skálázható is

2.2. Formális felírás

- Adott N darab csúcspont $V = \{1, \dots, N\}$, és egy kommunikációs séma M_D ami egy $N \times N$ mátrix
- A mátrix (i, j) eleméhez tartozik egy $p(i, j)$ valószínűség, ahol i a forrás csomópont és j a cél
- A bemeneti mátrix ábrázolható egy irányított G_D gráfban, ahol az élsúlyok a két pont közötti kommunikációs valószínűség
- Az algoritmus feltétele, hogy a mátrix ritka legyen
- Egy N hálózatra a torlódást és az úthosszt útválasztási sémával fogjuk definiálni
- Egy útválasztási séma az N hálózatra $\Gamma(N)$, ami Γ_{uv} utak halmaza, ahol (u, v) párok különböző utakat jelölnek
- Γ_{uv} egy útsorozat, ami összeköti az u pontot v ponttal

2.2.1. Torlódás

1. Definíció. A torlódást egy $\Gamma(N)$ útválasztási sémán a D demand mátrix segítségével írjuk fel:

$$C(D, \Gamma(N)) = \max_{e \in \Gamma(N)} \sum_{e \in \Gamma(uv)} p(u, v)$$

2.2.2. Úthossz

2. Definíció. Az átlag súlyozott úthosszt egy $\Gamma(N)$ útválasztási sémán a D demand mátrix segítségével írjuk fel:

$$L(D, \Gamma(N)) = \sum_{(u,v) \in D} p(u, v) \cdot d_{\Gamma(N)}(u, v)$$

ahol a $d_{\Gamma(N)}(u, v)$ az útvonal hosszát jelöli

2.2.3. Skálázhatóság

- A hálózatot skálázhatóra kell tervezni, ezért meghatározunk egy Δ konstans fokszámot, ami a maximális csatlakozások számát fogja meghatározni egy adott csomóponthoz
- N_Δ jelölje az összes Δ fokszámú gráfot, és elvárjuk, hogy $N \in N_\Delta$

2.2.4. Optimális torlódás

Az optimális torlódást egy hálózatra, úgy határozzuk meg, hogy a csak a torlódást vesszük figyelembe számításakor

$$C^*(D, \Delta) = \min_{N \in N_\Delta, \Gamma(N)} C(D, \Gamma(N))$$

2.2.5. Optimális úthossz

Az optimális úthosszt egy hálózatra, úgy határozzuk meg, hogy a csak az úthosszt vesszük figyelembe számításakor

$$L^*(D, \Delta) = \min_{N \in N_\Delta, \Gamma(N)} L(D, \Gamma(N))$$

2.3. cl-DAN hálózat tervezése

3. Definíció. Adott egy D demand mátrix, és egy Δ maximális fokszám, az (α, β) -cl-DAN hálózat tervezési probléma:

- *Hogy tervezzünk egy olyan $N \in N_\Delta$ hálózatot, és egy hozzá tartozó $\Gamma(N)$ útválasztási sémát, ami közel optimális torlódásra és úthosszra is*

Az algoritmus egy felső korlátot tud adni arra, hogy mennyivel fog eltérni a megoldás az optimálistól.

- *Torlódásra: $C(D, \Gamma(N)) \leq \alpha \cdot C^*(D, \Delta) + \alpha'$*
- *Úthosszra: $L(D, \Gamma(N)) \leq \beta \cdot L^*(D, \Delta) + \beta'$*

Az alfa vessző és béta vesszők olyan tényezők aki amik függetlenek a problémától

2.4. EgoTree

- Az Egofa egy torlódásra és úthosszra optimalizált fa hálózat egy csomópontra nézve
- Az Egotree-t definiáljuk a következő módon,

$EgoTree(s, \bar{p}, \Delta)$:

- s a forrás csomópont
- \bar{p} a szomszédainak eloszlásai
- Δ fokszám

- Ez közel optimális megoldást ad torlódásra és úthosszra

1. Tétel. *Adott egy \bar{p} frekvencia eloszlás az s forrás ponthoz, és adott egy Δ fokszám, ekkor az $EgoTree(s, \bar{p}, \Delta)$ egy (α, β) -cl-DAN a következő paraméterekkel:*

- $\alpha = \frac{4}{3}$
- $\beta = \log^2(\Delta + 1)$

2.5. $EgoTree(s, \bar{p}, \Delta)$ algoritmus

1. s a gyökér elem, Δ fokszámmal, üres fa
2. Rendezzük sorba $\bar{p} = \{p_1, p_2, \dots, p_k\}$ valószínűségeket csökkenő sorrendben
3. Kezdjük rárakni a fára a csomópontokat, a gyökér elemre legfeljebb Δ levél kerülhet
4. Mikor elértük a Δ levelet, a következő csomópontokat mindig a legkisebb összeített súlyú levélre kapcsolok rá, itt már legfeljebb két levele lehet minden fának

2.5.1. Algoritmus elemzése

- A kapott eredményben látható, hogy a maximális torlódás a legnagyobb súlyú élen van
- Minimalizálni ezt, lényegében egy időzítés probléma, hogy osszuk ki a munkákat Δ processzornak, hogy minden leghamarabb kész legyen
- Erre az optimális algoritmus NP-nehéz, de van közelítő módszer

2.5.2. Longest Processing Time (LPT)

- Először sorba rendezzük a feladatokat hossz szerint csökkenő sorrendben
- Ha van szabad processzor, akkor ahhoz rendeli a leghosszabb munkát
- Ha nincs akkor ahhoz a processzorhoz rendeli, ahol a legkevesebb ideig tart a munka

2. Tétel. Legyen ω_L a maximum idő, mielőtt egy processzor befejezi az összes munkát a mohó LPT algoritmus szerint, és ω_0 az optimális, ekkor

$$\frac{\omega_L}{\omega_0} \leq \frac{4}{3} - \frac{1}{3\Delta}$$

Ez az algoritmus polinom időben lefut

1. Lemma. Az $EgoTree(s, \bar{p}, \Delta)$ ad egy $\frac{4}{3}$ szorzóval nagyobb közelítést a minimális torlódásra az optimális Δ fokú fához képest, ami kiszolgál \bar{p} frekvencia eloszlást egy adott s forrás csomóponttra

2. Lemma. Az $EgoTree(s, \bar{p}, \Delta)$ ad egy $\log^2(\Delta + 1)$ szorzóval nagyobb közelítést a minimális úthosszra az optimális Δ fokú fához képest, ami kiszolgál \bar{p} frekvencia eloszlást egy adott s forrás csomóponttra

2.6. cl-DAN algoritmus

3. Tétel. Legyen D egy szimmetrikus kommunikáció kérelmoszlás, ahol az átlag csúcs fokszáma ρ , (azaz az élek száma $\rho \cdot \frac{n}{2}$). Ekkor a maximum fokszám $\Delta = 12\rho$, ehhez lehetséges generálni egy (α, β) -cl-DAN hálózatot, ahol:

- $\alpha = 1 + (\frac{8}{9})\Delta$
- $\beta = 1 + 4\log^2(\Delta + 1)$

Konstans ρ esetén ez konstans közelítést ad a minimális torlódásra és az optimális úthosszra

1. Felosszuk a hálózat csúcsait két halmazra, H - magas és L - alacsony fokszámúakra fele-fele arányban
 - Az alacsony fokszámú csúcsok fokszáma legfeljebb 2ρ
2. Megkeressük az összes olyan (u, v) élt, ahol u és v is a magas fokszámú halmazba tartozik
3. Az ilyen éleket a gráfban kiegészítjük egy segítő csomóponttal, $l \in L$, az eredeti csomópontok között megszüntetjük az élt, és felveszünk két új élt (u, l) és (v, l)
 - Minden segítő l csúcs választásakor egy még nem felhasználtat válasszunk az L halmazból
4. Meghatározzunk egy mátrixot, ami első lépésben az eredeti
 - Ahol segítő csomópontot vettünk fel, ott az útvonal hosszúhoz hozzá kell még adni az l -el való áthaladást is, és törölni kell az eredeti pontok közti élt
 - Ezután elkészítjük a magas halmaz csúcsaira a T_u fát, ahol a valószínűségeket a mátrixból kiolvassuk, $\Delta = 12\rho$ fokszámmal, ez közel optimális megoldást ad mindkét fel
5. Mivel u és v pontok közt egy l segítő csomópont van használva ezért T_u és T_v módosításra szorul. Alakítsuk át először T_u -t T'_u -ra
 - Ha $l \notin T_u$, $(p(u, l) = 0)$, akkor l átveszi v helyét T'_u -ban
 - Ha $l \in T_u$, $(p(u, l) > 0)$, akkor két lehetőségünk van:
 - Ha $(p(u, l) > (p(u, v)))$, akkor töröljük v -t a fából
 - Ha $(p(u, l) \leq (p(u, v)))$, akkor l átveszi v helyét T'_u -ban
 - T'_v hasonlóan számítjuk ki, ezzel garantálva, hogy T'_u és T'_v közötti kommunikáció az l csomóponton keresztül fog áthaladni
6. Konstruáljuk meg az új N hálózatot, vegyük az előbb készített egofákat és vegyük az uniójukat, azaz húzzuk be az összes olyan élet amik szerepeltek a fákban
 - De mivel nem csak magas fokú csomópontok közt történhetett adatforgalom, ezért még vegyük hozzá az N hálózathoz azokat az éleket is, ahol mindkét csomópont alacsony fokszámú volt

3. fejezet

Megvalósítás

4. fejezet

Teszt eredmények

5. fejezet

Összefoglalás

6. fejezet

Irodalomjegyzék

- [1] NetworkX - <http://networkx.github.io/>.
- [2] Python - [Python.org](http://python.org).
- [3] C. Avin, K. Mondal, and S. Schmid. Demand-Aware Network Design with Minimal Congestion and Route Lengths. page 9.
- [4] M. Ghobadi, D. Kilper, R. Mahajan, A. Phanishayee, N. Devanur, J. Kulkarni, G. Ranade, P.-A. Blanche, H. Rastegarfar, and M. Glick. ProjecToR: Agile Reconfigurable Data Center Interconnect. In *Proceedings of the 2016 conference on ACM SIGCOMM 2016 Conference - SIGCOMM '16*, pages 216–229, Florianopolis, Brazil, 2016. ACM Press.