PB165 – Grafy a sítě

10. Zajímavé grafy a peer to peer sítě

Obsah přednášky

- Náhodné grafy
- Náhodné geometrické grafy
- Power-law sítě
- Peer to peer sítě
  - Prohledávání v nestrukturovaných sítích
  - Strukturované sítě
  - Chord

Náhodné grafy (Erdös-Renyi)

- Pravděpodobnostní modely grafů

Definice

$G(n, p)$ model náhodného grafu: Mějme graf s n vrcholy, pak neorientovaná hrana vznikne mezi dvojicí vrcholů s pravděpodobností p, nezávisle na ostatních dvojicích uzlů.

-
- Pravděpodobnost, že graf G má k hran je dána výrazem

$$p^k(1-p)^{\binom{n}{2}-k}$$

- Pravděpodobnost, že graf $G \in G(n, p)$ obsahuje k hran má binomiální rozložení

$$\binom{\binom{n}{2}}{k} p^k(1-p)^{\binom{n}{2}-k}$$

Prahy náhodných grafů

Věta

Mějme model náhodného grafu $G(n,p)$ a nechť $p = c\frac{\log n}{n}$. Je-li $c > 1$, pak prakticky všechny grafy nemají izolované (nespojené) vrcholy. Je-li $c < 1$, pak prakticky všechny grafy mají alespoň jeden izolovaný vrchol.

Důkaz si pouze naznačíme:

- Nechť X je počet izolovaných vrcholů v grafu $G \in G(n,p)$.
- Pak *horní hranice* $E[X] = n(1-p)^{(n-1)} \approx n^{(1-c)}$
- Pravděpodobnost, že $X = 0$ je přibližně $\frac{1}{E[X]}$.
- $E[X]$ se blíží 0, pokud $c > 1$, tedy počet izolovaných vrcholů se blíží 0.
- Naopak, pro $c < 1$ se $E[X]$ blíží nekonečnu, a tedy pravděpodobnost, že graf G nemá izolovaný uzel, se blíží 0.

Náhodné geometrické grafy

Definice

Rozhoďme n bodů náhodně na jednotkovou síť. Náhodný geometrický graf $G(n, r)$ má n uzlů odpovídající n bodům a dva uzly jsou spojeny hranou pokud jsou ve vzdálenosti menší nebo rovné r. Vzdálenost mezi dvěma body $u = (x_1, y_1)$ a $v = (x_2, y_2)$ je definována jako
$d(u, v) = \max(|x_1 - x_2|, |y_1 - y_2|)$.

Jedná se o oblíbený model senzorových sítí.

Věta

Pokud $r \geq \sqrt{\frac{c \log n}{n}}$, *kde* c *je konstanta* > 4, *pak* $G(n, r)$ *je asymptoticky prakticky určitě souvislý graf, tj.* *Pravděpodobnost*($G(n, r)$ *je souvislý graf*) *se blíží 1 jak se* n *blíží nekonečnu.*

Důkaz je naznačen dále:

- Rozdělme jednotkovou síť na schránky velikosti $r/2 \times r/2$. Počet přihrádek je $4/r^2$ ($r < 1$).
- $G(n, r)$ je souvislé, pokud žádná přihrádka není prázdná.
- Nechť $r = \sqrt{\frac{c \log n}{n}}$, pak pravděpodobnost, že přihrádka je prázdná je menší nebo rovna $n^{-c/4}$.
- Nechť X je počet prázdných přihrádek. Pak $E[X] \leq n^{1-c/4}$, tj. $E[X]$ se blíží nule, pokud $c > 4$.
- Pravděpodobnost($X > 0$) $\leq E[X] \rightarrow 0$.

Power-law grafy

Definice

Power law je jakýkoliv polynomiální vztah, který nezávisí na měřítku. Pro dvě proměnné se nejčastěji vyjadřuje vztahem

$$f(x) = ax^k + o(x^k)$$

kde a a k jsou konstanty a $o(x^k)$ je asymptoticky malá (tj. zanedbatelná) funkce x.

Pro reálné sítě platí následující empirické tvrzení:
Distribuce stupně vrcholu sítě sleduje power law, tj. počet vrcholů stupně k je $ck^{-\beta}$.
Příklady:

- Internet a WWW: $\beta = 2,1$
- Sociální sítě (např. citační grafy): $\beta = 3$
- Biologické sítě (grafy interakce proteinů): $\beta = 2,5$

Většina grafů reprezentujících reálný svět má $\beta \in \langle 1, 4 \rangle$.

Prohledávání v nestrukturovaných sítích

Máme rozsáhlou síť (graf), každý uzel nese nějaká data. Jak najdeme hledaná data?

- Centrální index
  - Napster
  - Snadné, ale neškáluje (Google má $>$ 100.000 serverů)
- Decentralizované
  - Gnutella: Primitivní hledání: záplava
  - Chytřejší algoritmy

Základ výzkumu v oblasti *peer to peer sítí*

Gnutella

- Protokol: záplava dotazů
  - Ping/Pong
    - Uzel se hlásí při připojení do sítě a dále průběžně
    - Připojení: najdi nějakého člena, pošli *ping* zprávu a sbírej *pong* zprávy
  - Dotaz
    - Údaje o úspěchu se vrací cestou, kterou putoval dotaz

Gnutella – charakter sítě

- Power law distribuce uzlů
  - Je zcela nezávislá na distribucí uzlů v Internetu
- Prohledávání založeno na záplavě
  - Nastavena délka prohledávání (zpravidla 7)
  - Vysoce neefektivní
    - Obrovská spotřeba pásma (kapacity hran)
    - Obrovský počet redundantních zpráv
    - Paralelní aktivita uživatelů: lokální zátěž roste lineárně s velikostí sítě

Decentralizace

- Co můžeme ovlivnit:
  - Topologii peer to peer sítě
  - Umístění objektů (replikací)
- Power-law sítě
  - Základní principy:
    - Je-li distribuce stupně vrcholu p_k, pak distribuce stupně sousedů je úměrná kp_k.
    - Uzly mohou snadno uchovávat indexy objektů sousedů
  - Důsledky
    - Uzly s vysokým stupněm je možné snadno najít
    - Tyto uzly drží údaje (index) o velké části sítě

Prohledávání v power-law sítích

- Náhodná procházka (RW, Random Walk)
  - Nevracet se do naposled navštíveného uzlu
- Náhodná procházka ovlivněná stupněm vrcholu (DS)
  - Začni v ještě nenavštíveném vrcholu s nejvyšším stupněm
  - Následně sestupuj na uzly s nižším stupněm
  - Dokazatelně nejlepší pokrytí

Prohledávání v power-law sítích – shrnutí

Maximální flexibilita (libovolný algoritmus shody – např. full textové prohledávání) vykoupeno nepříliš vysokou efektivitou

- Výhody
  - Sublineární časová složitost
- Nevýhody
  - Problém s nalezením vzácných objektů
  - Velmi vysoká zátěž uzlů s vysokým stupněm

Modifikace algoritmů

- Expanze rozsahu
  - Záplava s postupně narůstajícím TTL (dokud se objekt nenajde nebo nedosáhne hranice)
  - Smyslem je nezaplavit příliš velkou část sítě okamžitě
- K-walker
  - K nezávislých náhodných procházek, žádná záplava
  - Různé varianty
    - Např. kontrola: po každých 4 krocích dotaz, zda druzí nenašli cíl

Další vlastnosti

- Distribuce dotazu
  - Podíl dotazů na jednotlivé objekty
    - Uniformní: všechny objekty jsou stejně populární
    - Power-law: malý počet objektů je mimořádně populární
- Replikace
  - Kopie rozeslány po síti: populárnější objekty se snáze naleznou (a jejich hledání méně zatíží síť)
  - Replikace zpravidla úměrná popularitě
    - Optimální replikace je úměrná odmocnině frekvence dotazů

Motivace dalších rozšíření

- Výhody nestrukturovaných sítí
  - Robustnost a fault tolerance
  - Nemají omezení na typ dotazů
- Záplava je velmi neefektivní
- Náhodné procházky
  - Lepší než záplava
  - Stále nepříliš efektivní bez dodatečné podpory (např. algoritmus DS uvedený výše)
  - Hlavním problémem je rozložení zátěže (příliš mnoho práce dělají uzly s vysokým stupněm)

Nový přístup

- Replikace přes sousedy
- Adaptace topologie
  - Zajistit, aby většina uzlů byla blízko uzlům s vysokým stupněm
  - Dynamická adaptace
    - Každý uzel se snaží zlepšit své sousedy
    - Podle posledních dotazů žádá konkrétní uzly, zda jej nevezmou za své sousedy
- Řízení toku
- Modifikace protokolu prohledávání
  - Náhodná procházka přes uzly s vysokou kapacitou (ne stupněm)
  - Neexistuje souvislost mezi kapacitou a stupněm uzlu

Řízení toku

- Výběr uzlu ovlivněn existencí *tokenu*
- Tokeny přidělovány podle kapacity uzlů
  - Důvěryhodnost: sděluje uzel korektně svou kapacitu?
- Při prohledávání se soustřeď na sousedy s největší kapacitou a volným tokenem
- Tímto způsobem přispíváme k dynamickému rozložení zátěže

Shrnutí

- Hlavní komponenty úspěšného prohledávání
  - Vlastní algoritmus prohledávání
  - Topologie (překryvová síť)
  - Strategie tvorby replik
  - Řízení toku
- Všechny tyto komponenty lze ovlivnit
- Topologie a replikace jsou ovlivněny agregovaným chováním uživatelů
- Stále nedostatečné pro řídce se vyskytující objekty

Strukturované sítě/prohledávání

- Distributed hash tables (distribuované hash tabulky)
  - Řešení pro účinné hledání i řídce se vyskytujících objektů
- Hash tabulky
  - Rychlé a levné nalezení dat indexovaných podle klíče
  - Výše jsme neomezovali způsob vyhledání dat
  - Pokud použijeme hash tabulky, pak ztrátu obecnosti musíme nějak kompenzovat (ale často i vyhledání podle klíče – např. název objektu – plně stačí)
- Obětujeme obecnost za efektivitu a rychlost

Distribuované hash tabulky

- Hash tabulky v p2p sítích: hledáme data podle klíče
- Předpokládáme, že data jsou uložena v distribuovaných uzlech (ne v poli)
- Důsledky:
  - Použijeme překryvovou síť která spojuje uzly pro nás vhodným způsobem
  - Počet uzlů neznáme a navíc se mění v čase
  - Pracujeme s idealizovanými vlastnostmi
    - Hash funkce mapuje na idealizovaný prostor
    - Samotný prostor je mapován na uzly dynamicky

Hlavní principy

- Přiřadíme klíči jedinečný uzel
- Nalezneme tento uzel rychle a snadno v překryvové síti
- Zavedeme replikaci (více uzlů pro jeden objekt/klíč)
- Rozložení zátěže může dynamicky měnit přiřazení klíče

Nejrozšířenější jednoduchá implementace: **Chord**

- Velmi populární (více jak 3000 citací).
- Překryvová síť: orientovaná kružnice

Chord: principy

- Uzly spojeny do orientované kružnice
  - Každý uzel zná svého předchůdce a následníka
- Klíče i uzly hashovány SHA-1 (160 bitové ID), identifikátory jsou rovnoměrně rozloženy po celém prostoru
- Klíče jsou přiřazeny uzlům s použitím konzistentního hashování
  - Náhodnost: Každý uzel dostane zhruba stejný počet objektů
  - Lokalita: Přidání či ubrání uzlu znamená, že $\mathcal{O}(1/N)$ klíčů získá nové přiřazení uzlům
- Vlastní vyhledávání:
  - Složitost $\mathcal{O}(\log N)$ vyměněných zpráv je možné dosáhnout při znalosti jen $\mathcal{O}(\log N)$ uzlů (plus předchůdce a následníka uzlu, který iniciuje hledání)

Vyhledávání v Chordu

- Naivní:
  - Pošlu dotaz předchůdci nebo následníku (podle identifikátoru); rekurzivně pokračuji
- Možnost optimalizace:
- Každý uzel má tzv. *finger table*
  - Obsahuje $\log_2 n - 1$ položek (n je počet uzlů v Chord síti)
  - Každá položka nese číslo uzlu, na němž je $k + 2^{i-1}$-tý klíč (k je identifikátor aktuálního uzlu).
- Vyhledávání
  - Nalezni v tabulce uzel, který buď obsahuje klíč nebo předchází uzlu, který klíč obsahuje.
  - Přejdi na tento uzel a pokračuj stejným způsobem
  - Celkem $\mathcal{O}(\log N)$ přechodů (zpráv)

Chord – přidání uzlu

- Striktní algoritmus
  - Nový uzel musí nalézt svého předchůdce, následníka a naplnit finger tabulku
  - Musí rovněž oznámit svou existenci všem uzlům, kteří by na něj měli ukazovat ze své finger tabulky
  - Zvládnutelné v $\mathcal{O}(\log N)$ čase, ale velmi složitým protokolem.
- Relaxovaný algoritmus
  - Využívá toho, že údaje ve finger tabulce nemusí být přesné
    - V nejhorším případě se prohledávání redukuje na naivní algoritmus
  - Stabilizační protokol: přepočtení finger tabulky (periodicky každý uzel)
  - Přidání uzlu: nalezne následníka a spustí stabilizační protokol

Chord – reakce na výpadky uzlů

- Replikací
  - Místo jednoho následníka si držíme r následníků
- Alternativní cesty
  - Pokud uzel nepřebere hledání, využij předchozí uzel z finger tabulky nebo uzel s nejbližší replikou

Sdílení souborů

Hybridní přístup

- DHT pro méně časté objekty
- Náhodná procházka pro populární objekty

Gnutella a vzácné objekty:

- 41 % dotazů nalezne jen méně jak 10 objektů
- 18 % dotazů nenalezne žádný objekt (přitom jen 6 % dotazů vedlo na neexistující objekty)

Nutno identifikovat vzácné objekty a ty zpřístupnit přes DHT

- Také možné časové omezení:
- Jestli náhodná procházka nenalezne objekt do 30 s, přepni na DHT