PEER-TO-PEER PŘÍSTUP K VYHLEDÁVÁNÍ NA INTERNETU

Adam Morávek
Ivan Jelínek

Katedra počítačů, FEL, ČVUT – České vysoké učení technické v Praze, Karlovo Náměstí 13, 121 35 Praha 2, morava3@fel.cvut.cz, jelinek@fel.cvut.cz

Abstrakt
Příspěvek se zabývá jednou z možností efektivního vyhledávání na Internetu. Popisuje výhody modelu Peer-to-Peer (P2P) nad modelem Client/Server a nastiňuje možnost použití kruhové topologie logické P2P sítě jako možnou a vhodnou topologii pro efektivní a spolehlivé vyhledávání. Dále je diskutován způsob propojení jednotlivých uzlů v kruhu a vzájemné propojení jednotlivých kruhů. Popisovaná metoda propojení uzlů v kruhu si klade za cíl minimalizovat dobu vyhledávání paralelizací propagace dotazu. Tento přístup je porovnán s přístupem systému Gnutella, který je nejrozšířenějším používaným systémem pro sdílení a vyhledávání dat.

1. Úvod

Problém vyhledávání v P2P sítích je v současné době více a více aktuální zejména z důvodu nedostatků současných centralizovaných vyhledávačů [1]. Výhody P2P sítí [2] tkví v jednotném pohledu na uzly, kdy každý uzel poskytuje (resp. je serverem) i získává (resp. je klientem) zároveň. Výhodou P2P sítě je, že mohou mít obecně libovolnou topologii, která se může dynamicky měnit. Volba vhodné topologie je základem efektivního vyhledávání. V současné době existuje několik fungujících P2P systémů pro vyhledávání sdílených dat. Jsou to zejména Gnutella [3], FreeNet [4], Napster [5], CAN [6] či Chord [7].

Topologie používaná těmito systémy je různá – distribuovaná (Gnutella), hybridní (Napster – kombinace centralizované a decentralizované topologie) či kruhová (Chord). Ovšem všechny tyto systémy mají některé podstatné nevýhody: Gnutella využívá k propagaci dotazů na vyhledání objektu tzv. *flooding* [8], což je obdoba broadcastu, který velmi zatěžuje síť [9]. U Napsteru (jelikož používá hybridní topologii) se do jisté míry objevuje centralizace, což je problém z hlediska stability – výpadek centrálního uzlu zapříčiní odpojení mnoha závislých uzlů.

Systém Chord byl vytvořen pro implementaci myšlenek distribuovaného systému souborů s využitím DHT (distribuované hash-tabulky [10]). Ovšem nepředpokládá, že se v síti objeví více stejných objektů. Možné řešení těchto nevýhod je nastíněno v dalším textu.

2. Vytyčené cíle výzkumu

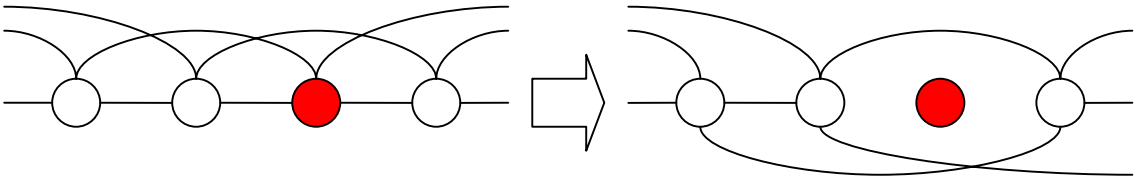
Tento příspěvek si klade za cíl představit takovou P2P topologii a takový systém směrování dotazů, který by řešil výše uvedené problémy a zároveň umožnil efektivní vyhledávání v P2P síti. Volba topologie P2P sítě je jednou ze stěžejních úloh při návrhu každého systému pro sdílení dat. Zásadními problémy jsou zejména přidávání a odebírání uzlů ze sítě, obsluha výpadku uzlu a propagace dotazu.

3. Vhodná P2P topologie

Vhodným řešením problémů při návrhu topologie sítě se jako vhodná, jeví topologie kruhová [8], protože:

- snadno řeší problémy přidávání a odebírání uzlů,
- je možno rozdělit zátěž,
- veškerá data obsažená v uzlech kruhu lze prohledat,
- je odolná proti výpadkům uzlů,
- lze snadno spravovat komunity (kruh = komunita).

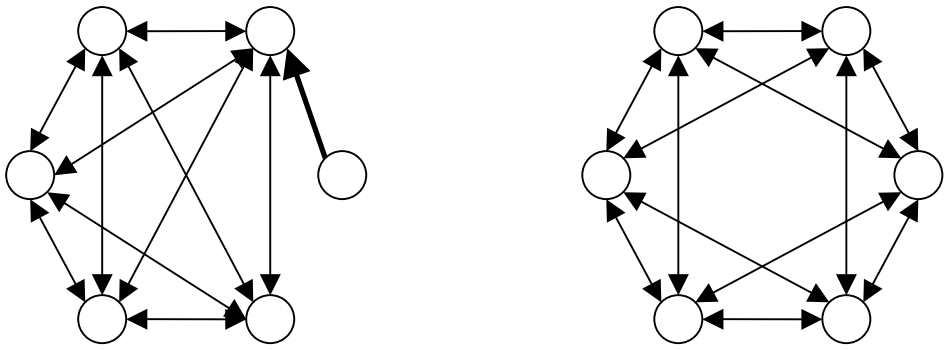
Každý uzel v kruhu uchovává vazby na dva bezprostřední uzly ve směru a dva bezprostřední uzly *v protisměru* hodinových ručiček. Tento způsob propojení zaručuje, že je možné ošetřit výpadek uzlu, který má oba funkční sousedy. Tak je tedy možné najednou ošetřit výpadek až (N div 2) - 1 uzlů najednou, kde N je počet uzlů u kruhu (viz obr. 1).



Obr. 1: Ošetření výpadků mezilehlých uzlů

3.1 Přidávání a odebírání uzlů

Kruhová topologie sítě umožňuje snadné přidání uzlu do sítě, a to pomocí vazeb na sousední uzly. Uzel, který chce být zařazen do kruhu, zašle zprávu jakémukoliv uzlu sítě, který jej zařadí jako svého bezprostředního souseda (viz obr. 2).

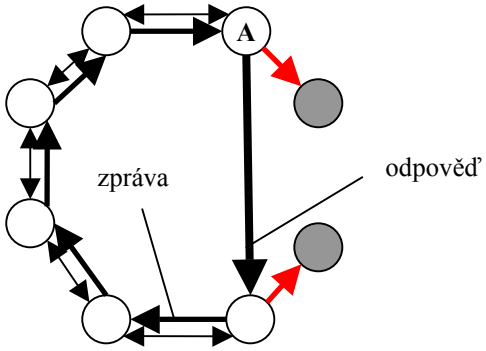


Obr. 2: Přidání uzlu do kruhu

3.2 Obsluha výpadků uzlů

Jak již bylo napsáno výše, je možné zachovat kruh při výpadku uzlů, které mají fungující sousedy – slouží k tomu vazby na sousední uzly. Pomocí těchto odkazů lze vytvořit „bypass“ přes nefungující uzel. Problém nastane, když selžou dva nebo více bezprostředně sousedních uzlů. Pokud tedy uzel **A** zjistí, že nemá v některém směru dva a více sousedů, vyšle po dostupné části kruhu zprávu obsahující vazbu na bezprostředního souseda uzlu A. Pokud zprávu zachytí uzel, který rovněž nemá bezprostředního souseda a nemá možnost provést

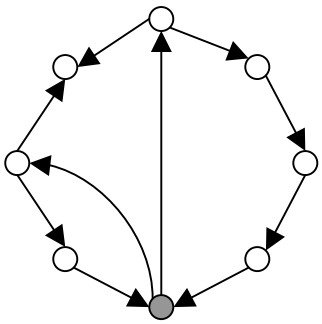
bypass, odpoví uzlu **A** a předá mu rovněž vazbu na svého bezprostředního souseda. Tím je dodán dostatek informací pro znovusestavení kruhu (viz obr. 3). Problém nastane, pokud se kruh přeruší na více místech najednou a nelze ho opravit pomocí odkazů na oba sousední uzly. V tom případě vznikne **N** kruhů, kde **N** je počet přerušených míst.



Obr. 3: Znovusestavení kruhu

3.2 Propagace dotazu

V kruhové topologii je propagace dotazu velmi jednoduchá a je zaručeno, že skončí po konečném počtu kroků – dotazující uzel pošle dotaz v obou směrech po kruhu. Dotaz tedy projde celým kruhem, tj. **N** uzly. Aby se zpracování dotazu urychlilo, každý uzel si ke každému hledanému výrazu uchová seznam uzlů, které na hledaný výraz odpověděly. Při příštím hledání stejného výrazu se dotaz propaguje jak od hledajícího uzlu po kruhu, tak i přímo na uzly z tabulky, které pak propagují dotaz dále. Tak se hledání do jisté míry paralelizuje (viz obr. 4).



Obr. 4: Částečně paralelizovaná propagace dotazu

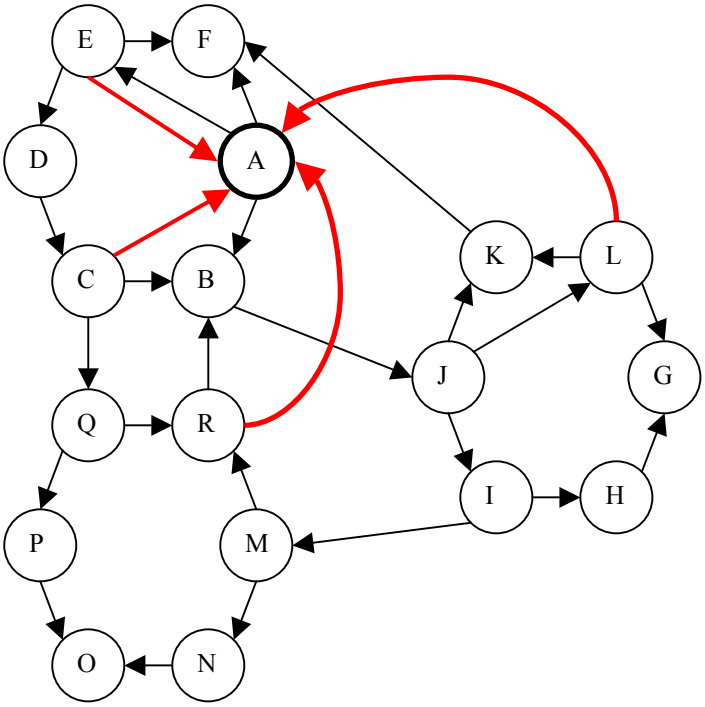
Tímto způsobem se prohledá každý uzel v kruhu. Ovšem každý uzel může obsahovat i vazby na uzly v jiných kruzích. Tyto uzly jsou při každém hledání kontaktovány a hledání se tím pádem dostává i do „jejich“ kruhů. Je zřejmé, že takto může dotaz procházet různými kruhy donekonečna nebo se např. vrátit do kruhu, ze kterého byl propagován jiným uzlem. Proto je s každým dotazem spjat jedinečný identifikátor, pomocí kterého uzel pozná, že jím již byl zpracován – každý uzel si vytváří tabulku identifikátorů jím zpracovaných dotazů.

Dotaz je tak propagován neustále do míst, kde ještě nebyl zpracován a pokračuje, dokud má kam. V situaci, kdy jsou uzly v kruhu přetíženy, příchozí dotaz mohou odmítnout, čímž jeho propagaci násilně ukončí. Celý proces propagace je ilustrován na obr. 5. (pozn.: uzel A již má v tabulce předem nastaven uzel E).

Tabulka uzlů uzlu A

výraz	uzly
‘pokus’	E, **C, L, M**

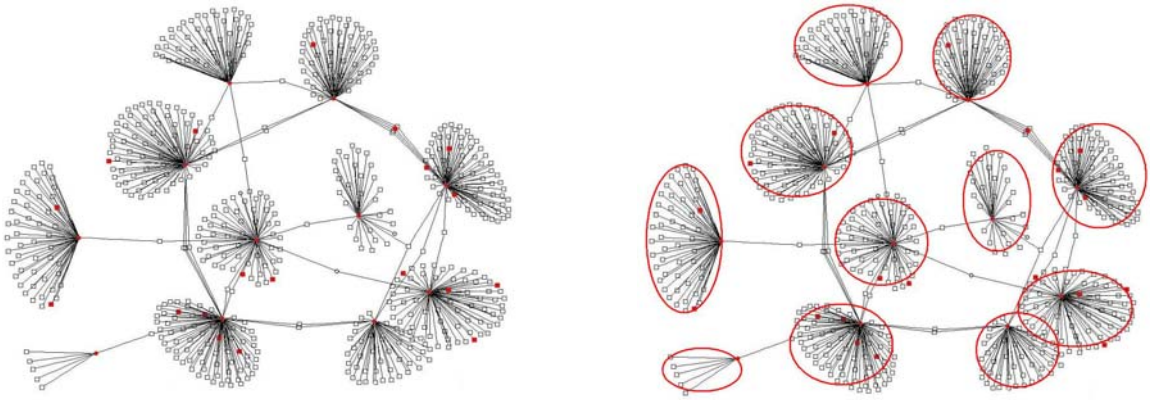
Uzel **A** hledá dokument obsahující slovo ‘pokus’



Obr. 5: Propagace dotazu

4. Porovnání se systémem Gnutella

Každý uzel systému Gnutella obsahuje vazby na několik dalších uzlů, čímž jsou při propagaci dotazu znevýhodněny uzly s větším počtem vazeb (shluky) neboť jej propagují pomocí všech jejich vazeb. V případě kruhové topologie je zátěž uzlů při propagaci téměř rovnoměrná. Nahradíme-li shluky v Gnutelle kruhy, maximální hodnota zatížení uzlů se radikálně zlepší (viz obr 6):



Gnutella Nahrazení shluků kruhy

Obr. 6.: Možné vylepšení zatížení uzlů v Gnutelle

Po nahrazení se ovšem sníží rychlost propagace dotazu neboť ten se musí propagovat po kruhu a ne přímo k uzlům. To se ovšem týká situace před natrénováním kruhu, kdy každý uzel obsahuje pouze vazby na sousedy. Po natrénování uzlů se rychlost propagace zvýší. Pro zjištění přesných výsledků je ovšem třeba vyvinout simulátor celého procesu.

4. Závěr

V současné době probíhá implementace simulátoru celého systému a v době prezentace tohoto příspěvku budou k dispozici první výsledky experimentální simulační verze navrhovaného systému. Očekává se, že popsaný přístup bude efektivnější zejména z hlediska zatížení uzlů a důmyslnější z hlediska tvorby komunit. Další výzkum bude zaměřen na hledání efektivního způsobu směrování dotazů pro urychlení jejich propagace.

Lliteratura:

1. Andrzej Goscinski, Wanlei Zhou. The Client-Server Model and Systems. Wiley Encyclopedia of Electrical and Electronics Engineering, Volume 3. Ed. J. G. Webster, John Wiley & Sons, Inc., New York, pp. 431-451, 1999
2. Manoj Parameswaran, Anjana Susarla, Andrew B. Whinston. P2P Networking: An Information-Sharing Alternative. IEEE Computer 34(7): 31-38 (2001)
3. gnutella.com. http://www.gnutella.com/
4. The Free Network Project. http://freenet.sourceforge.net/
5. Napster. http://www.napster.com/
6. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable and Content-Adressable Network. Proceedings of ACM SIGCOMM, San Diego, CA, USA, Aug. 2001
7. Chord. http://www.pdos.lcs.mit.edu/chord/
8. Yunhao Liu, Zhenyun Zhuang, Li Xiao and Lionel M.Ni. AOTO: Adaptive Overlay Topology Optimization in Unstructured P2P Systems, Proceedings of IEEE Globecom 2003
9. Doval, D., O'Mahony, D. Overlay Networks: A Scalable Alternative for P2P, IEEE Internet Computing, Vol. 7, N0. 3, June/July 2003, pp 2-5
10. Ion Stoica, et. al, Chord: A scalable Peer-To-Peer lookup service for internet applications, Proceedings of the 2001 ACM SIGCOMM Conferrence pages 149–160, 2001