GENERIC HASH TREE BITMAP ALGORITHM FOR EFFICIENT IPV6 LOOKUP

František Sedlář
Master Degree Programme (2), FIT BUT
E-mail: xsedla76@stud.fit.vutbr.cz

Supervised by: Jiří Tobola
E-mail: itobola@fit.vutbr.cz

Abstract: As the speed of computer networks grows, all network devices have to work faster to keep up with services running in the network. LPM (longest prefix match) is one of the most time-consuming network operations which has to be done in every router. This article proposes a new Generic Hash Tree Bitmap LPM algorithm for fast longest prefix match for both IPv4 and IPv6 networks. The algorithm takes full advantage of using hash functions, which are used to jump over sparse parts of the IP prefix tree.

Keywords: Longest Prefix Match, LPM, Trie, Tree Bitmap, Hash Tree Bitmap, Generic Hash Tree Bitmap, Netbench

1 ÚVOD

Se zrychlováním počítačových sítí jsou kladeny stále větší nároky na veškeré síťové prvky. Ty musí pracovat rychleji a efektivněji, aby neomezovaly služby provozované v síti. Jednou z časově náročných operací, které je nutné provádět na každém směrovači, je vyhledávání nejdelších shodných prefixů.

Směrovač obsahuje směrovací tabulku, která se skládá z množiny prefixů IP adres. Ke každému prefixu je přiřazeno výstupní rozhraní. Pro jednotlivé pakety vstupující na rozhraní směrovače musí směrovač najít takový záznam, který je prefixem jejich cílové IP adresy. Platných prefixů pro konkrétní cílovou adresu může směrovač nalézt obecně mnoho. Z těchto musí vybrat ten nejdelší.

Pakliže (dle [1]) na vytížené 10 Gb/s lince mohou přicházet pakety každých 67 ns, je třeba v tomto čase projít směrovací tabulku (ta dnes může obsahovat stovky tisíc záznamů) a vyhledat nejdelší prefix. S nasazením protokolu IPv6 vzroste maximální délka každého prefixu z 32 bitů na 128 bitů, což tento úkon dále znesnadní.

Cílem vývoje algoritmů pro vyhledání nejdelšího shodného prefixu (LPM) je tedy zejména zvyšovat jejich rychlost. Na rychlejších linkách se proto využívá HW akcelerace. Tento článek představuje nový LPM algoritmus, který je vhodný i pro dlouhé adresy protokolu IPv6.

2 SOUČASNÉ ŘEŠENÍ

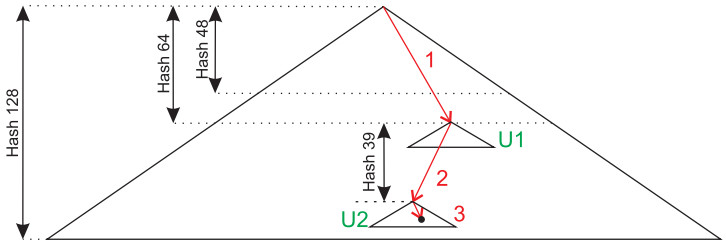
Algoritmus vychází z již existujícího a používaného algoritmu Tree Bitmap, který byl popsán v [2]. Tree Bitmap vytváří ze vstupní množiny prefixů stromovou strukturu. Zpracovává n bitů v jednom kroku, každý uzel uvnitř obsahuje $2^n - 1$ uzlů Trie. Trie je struktura podobná binárnímu stromu. Vyhledávání v ní probíhá dle jednotlivých bitů vstupního řetězce. Proto následníci libovolného uzlu mají shodný prefix. Uzel Tree Bitmap tedy může obsahovat až 2^n potomků. Počet bitů n zpracovávaných v jednom kroku lze zvolit (označuje se jako střída) a tím ovlivnit výšku generovaného stromu. U střídy větší než cca 8 již paměťové nároky na reprezentaci množiny prefixů neúměrně rostou. Při střídě 8 je

maximální výška stromu u IPv6 $128/8 = 16$ uzlů. U menších stříd (3–5) je zase vygenerovaný strom (zejména u IPv6) velmi vysoký a vyhledávání v něm vyžaduje mnoho kroků.

Tuto nevýhodu částečně eliminuje algoritmus Hash Tree Bitmap (představený v [3]), který strukturu vertikálně rozděluje na několik nižších stromů zadané výšky. Hash Tree Bitmap obsahuje navíc hashovací tabulky různých délek (například 32, 48, 64, 128 – dle typického histogramu délek prefixů) s odkazy na příslušné stromy. Při vyhledávání prefixu k IP adrese jsou z její binární reprezentace vytvořeny podřetězce daných délek a ty jsou následně paralelně vyhledány ve všech hashovacích tabulkách. Z nejdelšího nalezeného záznamu se načte adresa uzlu, který představuje kořen sníženého stromu. V tomto stromu se následně vyhledá zbytek původní IP adresy.

3 NÁVRH NOVÉHO ALGORITMU

Navržený algoritmus Generic Hash Tree Bitmap je optimalizací Tree Bitmap využívající efektivní a generické použití hashovacích funkcí. Nový algoritmus snižuje i podstromy, které tvoří Hash Tree Bitmap. Generic Hash Tree Bitmap zavádí možnost mít malé hashovací tabulky v každém uzlu stromu a použít tak hashování nejen v prvním kroku algoritmu. Princip vyhledávání znázorňuje obrázek 1. Ilustrovaný příklad obsahuje hashovací tabulky se záznamy délky 48, 64 a 128 bitů. Výsledkem hledání je prefix délky 105 bitů. Vyhledávání začíná stejně jako u Hash Tree Bitmap – nejprve jsou prohledány hashovací tabulky a dle nejdelšího nalezeného záznamu se dostáváme k Tree Bitmap uzlu U1 (krok 1). Dál už by algoritmus Hash Tree Bitmap pokračoval přes mnoho uzlů až k uzlu obsahujícímu nejdelší prefix (při střídě 3 by to bylo přes $\lceil (105-64)/3 \rceil = 15$ uzlů). Navržený algoritmus GHTB umožňuje, aby v uzlu U1 nalezeném v kroku 1 existovala další hashovací tabulka se záznamy zadané délky (v ilustrovaném případě 39 bitů). Přes záznam v této tabulce se lze rychle dostat blízko cílovému uzlu, případně (jako na obrázku 1) přímo k cílovému uzlu U2. Hashovací tabulky nemusí být přítomny ve všech uzlech. V uzlech bez tabulek se vyhledává dle algoritmu Tree Bitmap.



Obrázek 1: Vyhledávání navrženým algoritmem.

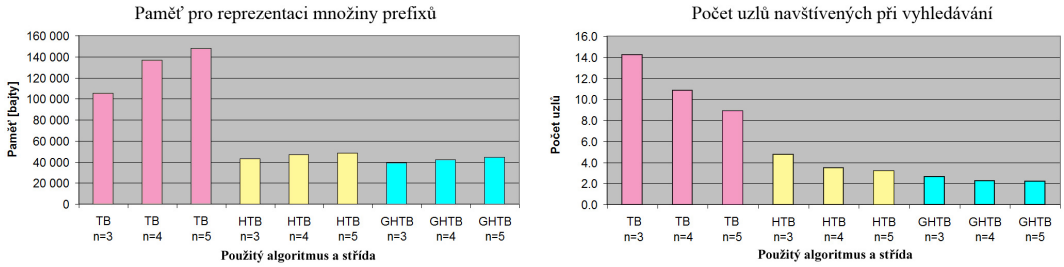
Implementována a otestována byla varianta, která ve stromu najde nevětvené cesty, uzly na této cestě odstraní a cestu přeskočí pomocí záznamu v hashovací tabulce prvního uzlu. Předpokládejme, že ve stromě je nevětvená cesta složená z uzlů $X_0, X_1, X_2, \ldots, X_k, X_{k+1}, \ldots, X_m$ a je zadáno, že uzly mohou obsahovat hashovací tabulku pro přeskočení k uzlů. Pak je do uzlu X_0 vložena hashovací tabulka délky kn (kde n je střída), obsahující odkaz na uzel X_k. Uzly $X_1, X_2, \ldots X_{k-1}$ mohou být odstraněny. Vyhledávání pak proběhne rychleji, protože není třeba procházet uzly $X_1, X_2, \ldots, X_{k-1}$.

3.1 TESTOVÁNÍ ALGORITMU

Nový algoritmus Generic Hash Tree Bitmap se stal součástí knihovny Netbench výzkumné skupiny ANT na FIT VUT v Brně. Tato knihovna obsahuje implementaci mnoha dalších LPM algoritmů a umožňuje tyto algoritmy porovnávat.

Algoritmus byl testován na reálné množině IPv6 prefixů získané z [4], která obsahuje téměř 15000 záznamů. Obrázek 2 vlevo ukazuje paměťové nároky pro reprezentaci testovací množiny při použití

algoritmu Tree Bitmap (TB), Hash Tree Bitmap (HTB) a Generic Hash Tree Bitmap (GHTB) při různých střídách. Z obrázku je patrné, že paměťové nároky GHTB jsou lehce nižší než nároky HTB. To je dáno tím, že bylo odstraněno mnoho uzlů – například u střídy 3 obsahovala stromová struktura vytvořená algoritmem HTB 3072 uzlů, zatímco struktura algoritmu GHTB pouze 2123 uzlů.



Obrázek 2: Výsledky testování.

Další test se zaměřuje na rychlost vyhledávání jednotlivých algoritmů (v podobě počtu uzlů, kterými musí algoritmus při vyhledávání projít). Pro tento test bylo vybráno 150 IPv6 adres, které byly vyhledány v testovací množině prefixů. Průměrný počet uzlů navštívených během vyhledávání nejdelších shodných prefixů ukazuje obrázek 2 vpravo.

Nově implementovaný algoritmus dle očekávání umožňuje vyhledávat rychleji a efektivněji, přičemž paměťové nároky oproti Hash Tree Bitmap nestoupají. Můžeme si například všimnout, že HTB se střídou 3 vyžaduje 43 kB paměti, což je srovnatelné s GHTB se střídou 4 (42 kB paměti). GHTB se střídou 4 však při vyhledávání provedl průměrně pouze 2,3 náhodných přístupů do paměti, zatímco srovnávaný HTB dvojnásobek (4,7 přístupů).

4 ZÁVĚR

Tato práce představuje LPM algoritmus Generic Hash Tree Bitmap pro vyhledávání IPv6 prefixů. Svými parametry předčí i v současné době používaný Tree Bitmap. Algoritmus navíc umožňuje nastavením parametrů do značné míry upravovat paměťové a časové nároky a rychlost vyhledávání. Ve vývoji algoritmu budu pokračovat i v rámci diplomové práce – ta by měla navrhnout další metody pro výběr uzlů vkládaných do hashovacích tabulek a následné úpravy Tree Bitmap stromů.

PODĚKOVÁNÍ

Tento příspěvek vznikl za podpory grantu FIT-S-11-1 a výzkumného záměru MSM 0021630528.

REFERENCE

[1] Tobola, J.: Vyhledání nejdelšího prefixu. [Pojednání k tématu dizertační práce]. FIT VUT v Brně, 2009

[2] Eatherton, W.; Varghese, G.; Dittia, Z.: Tree bitmap: hardware/software IP lookups with incremental updates. SIGCOMM Computer Communication Review, Volume 34, Issue 2, New York: ACM, 2004, s. 97–122, ISSN 0146-4833

[3] Tobola, J.; Kořenek, J.: Effective Hash-based IPv6 Longest Prefix Match. In: IEEE Design and Diagnostics of Electronic Circuits and Systems DDECS'2011, Cottbus, DE: IEEE Computer Society, 2011, s. 325–328, ISBN 978-1-4244-9753-9

[4] BGP table data. Dostupné na URL http://bgp.potaroo.net/ (únor 2013)