MEMORY EFFICIENT IP LOOKUPS IN HIGH-SPEED NETWORKS

Martin Skačan

Master Degree Programme (2), FIT BUT

E-mail: xskaca00@stud.fit.vutbr.cz

Supervised by: Jan Kořenek

E-mail: korenek@fit.vutbr.cz

Abstract: The growth of Internet traffic and speed of network links has direct impact on the required performance of core routers. To handle network traffic with 100 Gbps throughput, routers have to process millions of packets per second. The most time-critical operation in core routers is the Longest Prefix Match (LPM). Many highly optimized algorithms and hardware architectures are able to process IPv4 traffic at very high speed, but all these algorithms and architectures have limited throughput or high memory requirements for IPv6 traffic. Therefore, we propose a new LPM algorithm which is able to provide very low memory demands for IPv4/IPv6 lookups. New representation of IP prefixes was designed in order to fit large routing tables to the on-chip memory. To the best of our knowledge, the proposed algorithm has the lowest memory requirements in comparison to existing LPM algorithms. Moreover, it can be implemented in hardware to achieve 100 Gbps throughput.

Keywords: IP lookup, Longest Prefix Match, IPv6, memory

1. ÚVOD

Rozvoj počítačových sietí a Internetu nezastaviteľne postupuje. Neustále sa zvyšujú prenosové rýchlosti liniek a objem dát, ktorý je prostredníctvom nich prenášaný. To kladie obrovské nároky na aktívne sieťové prvky, predovšetkým smerovače. Časovo kritickou operáciou v smerovačoch je vyhľadanie najdlhšieho zhodného prefixu (Longest Prefix Match – LPM). Jej úlohou je prehľadať celú smerovaciu tabuľku a vybrať z nej najdlhší prefix, ktorý odpovedá cieľovej IP adrese paketu. Na základe výsledku je potom paket preposlaný nasledujúcemu zariadeniu v sieti.

S rozvojom Internetu sa zvyšuje tiež celkový počet pripojených zariadení, a teda narastá počet záznamov v smerovacích tabuľkách, ktoré dosahujú až stovky tisíc položiek [1]. Pre zabezpečenie priepustnosti na úrovni 100 Gbps je nutné vykonať vyše 160 miliónov vyhľadaní za sekundu. Čas vyhradený na uskutočnenie jednej LPM operácie je potom približne 6 ns. Takéto výkonnostné hodnoty sme v súčasnosti schopní dosiahnuť jedine s využitím hardwarovej akcelerácie operácie LPM.

Ďalšou výzvou v tejto oblasti je prechod na protokol IPv6, ktorý prináša 4 násobné zvýšenie dĺžky IP adresy. Táto zmena spôsobuje ďalší nárast prehľadávaného priestoru, ktorý musíme spravovať. To výrazne zvyšuje nároky na výkon a potrebnú pamäť používaného algoritmu. V tomto článku preto popisujem návrh nového algoritmu, ktorý minimalizuje pamäťové nároky a dokáže efektívne pracovať s protokolom IPv6 ako aj s protokolom IPv4.

2. ANALÝZA SÚČASNÉHO STAVU

Väčšina známych LPM algoritmov je založená na štruktúre Trie, ktorá kóduje smerovaciu tabuľku do podoby binárneho stromu. Každý uzol stromu má 2 následníkov – ľavý pre bit 0, pravý pre bit 1. Uzol je označený ako prefixový, ak v ňom končí platný prefix zo vstupnej množiny. Operácia LPM potom znamená prechod stromom od koreňa do listového uzlu na základe bitov z IP adresy.

Výsledkom je posledný navštívený prefixový uzol. Táto dátová štruktúra je jednoduchá a ukazuje základnú myšlienku využitia stromovej štruktúry. Má však vysoké pamäťové nároky (spôsobené veľkým počtom ukazateľov) a mnoho prístupov do pamäti na 1 vyhľadanie (pre IPv6 až 128).

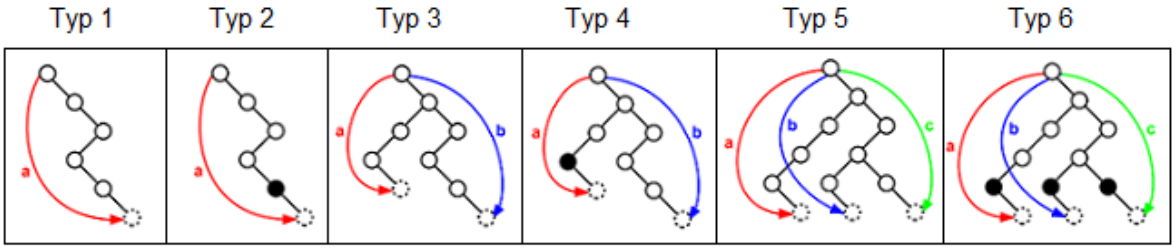
S cieľom zvýšiť efektivitu stromových algoritmov boli zavedené viacbitové metódy, ktoré v každom kroku spracujú *n* bitov adresy. Jednou z najlepších a najrozšírenejších sa stala metóda Tree Bitmap (TBM) [2]. Na základe nastaveného parametra pokrýva TBM uzol až *n* úrovní klasickej Trie a teda zapuzdruje až $2^n - 1$ jednobitových uzlov. Tým pádom sa počet potrebných krokov výrazne znižuje. Dátová štruktúra Tree Bitmap obsahuje 2 bitmapy – interná (pre popis platných prefixov) a externá (pre popis nasledovníkov) a 2 ukazatele. Táto efektívna reprezentácia šetrí priestor a umožňuje jednoduchú HW implementáciu. Objavuje sa však plytvanie alokovaným priestorom v riedko zaplnených častiach stromu, čo sa výrazne prejavuje práve u prefixov IPv6 adries.

Tento nedostatok sa pokúša odstrániť algoritmus Shape Shifting Trie (SST) [5], ktorý rozširuje predchádzajúcu dátovú štruktúru o tzv. tvarovú bitmapu. Tá popisuje tvar jednotlivých uzlov, ktoré sa dokážu meniť a dokonale prispôsobovať aktuálnej prefixovej množine. Pamäťové nároky sú potom extrémne nízke, avšak za cenu obrovskej výpočtovej zložitosti pri tvorbe stromu. Nutnosť dekódovať tvar každého uzlu zvyšuje tiež réžiu pri samotnej práci algoritmu. Navyše podľa dostupných informácií neexistuje pre túto dynamickú štruktúru žiadna vhodná HW realizácia.

S využitím algoritmov Trie a TBM som vykonal analýzu prefixových množín protokolov IPv4 a IPv6. Analyzované boli odpovedajúce stromové reprezentácie daných množín, ktoré ukázali niektoré typické charakteristiky. V analyzovaných stromoch sa vyskytuje veľký počet úsekov, ktoré neobsahujú žiadne prefixy a sú nevetvené (alebo s malým stupňom vetvenia). Ďalšiu výraznú skupinu tvoria listové TBM uzly, ktoré majú maximálne 4 prefixy. Tieto 2 charakteristické skupiny predstavujú hlavný priestor pre optimalizácie.

3. NÁVRH ALGORITMU

Návrh algoritmu vychádza z poznatkov predchádzajúcej analýzy reálnych prefixových množín. Zároveň sa snaží zachovať pozitíva algoritmov TBM a SST a súčasne eliminovať ich zápory. Na začiatku som navrhol sadu niekoľkých typov uzlov, ktoré sa budú v novom algoritme využívať. Grafická ukážka týchto uzlov je zobrazená na obrázku 1.



Obrázek 1: Návrh nových typov uzlov.

Tieto uzly predstavujú efektívne zakódovanie úsekov stromu s nízkym stupňom vetvenia. Existuje vždy varianta uzlu, ktorá umožňuje uloženie prefixu na konci každej vetvy (uzly typu 2, 4 a 6). Uzly typu 1 a 2 teda predstavujú dlhý nevetvený úsek. Uzly 3 a 4 povoľujú jedno vetvenie v rámci uzlu a uzly 5 a 6 povoľujú až dve vetvenia. Ďalším používaným typom je klasický TBM uzol (s parametrom n=3). Ten sa zvyčajne využíva v úsekoch s vysokým stupňom vetvenia alebo s vysokou hustotou prefixových uzlov. Uvedené uzly sa nakoniec ešte môžu vyskytovať vo variante listových uzlov, ktoré nemajú žiadnych nasledovníkov.

Dĺžky vetví sú variabilné a uložené pomocou dvojice (*cesta; dĺžka_cesty*). Existuje však maximálna dĺžka pre každý uzol (20, 20, 15, 12, 13 a 7 bitov postupne pre typy 1-6). Jednotlivé uzly teda nemajú uniformnú veľkosť ako je to u TBM. S ohľadom na budúcu HW implementáciu sa uvažuje zarovnanie uzlov na niekoľko pevných veľkostí tak, aby mohli byť jednoducho mapované do pa-

mäti. Každá cesta je v rámci uzlu zadaná binárne a môže mať ľubovoľný tvar. Tento návrh pripomína princíp a výhody SST, ktorý je však obmedzený len na niekoľko tvarových typov.

Vďaka tomuto návrhu máme k dispozícii pevnú sadu šablón, ktorú sa pokúsime vhodne namapovať na strom Trie. Mapovací algoritmus postupuje od koreňa smerom k listovým uzlom a hľadá čo najlepšie pokrytie prefixového stromu. V každom kroku skúsi algoritmus namapovať všetky uvedené typy uzlov a vyberie ten, ktorý má najlepšiu cenu. Cena predstavuje pomer pokrytých jednobitových uzlov a veľkosti namapovaného uzlu.

4. VÝSLEDKY

Navrhnutý algoritmus som implementoval v jazyku Python a uskutočnil som zhodnotenie pamäťovej náročnosti. Experimenty boli vykonané pomocou nástroja Netbench [4] a boli použité reálne smerovacie tabuľky protokolu IPv4 a IPv6 [1].Výsledky zachytáva tabuľka 1, ktorá ukazuje tiež porovnanie s algoritmom TBM a SST. Pri protokole IPv4 bola dosiahnutá pamäťovú úsporu vyše 40% oproti TBM a približne 20 % oproti SST. V kontexte protokolu IPv6 sú výsledky ešte priaznivejšie, kde bol algoritmus SST prekonaný až o takmer 30%.

Množina	Počet prefixov	Pamäťové nároky [kB]			Ušetrená pamäť	
		TBM	SST	Nový alg.	Oproti TBM	Oproti SST
Set1_IPv4	332118	1211,18	866,31	713,42	41,10%	17,65%
Set2_IPv4	220779	712,76	510,13	393,60	44,78%	22,84%
Set3_IPv6	10518	159,43	73,56	52,20	67,26%	29,04%
Set4_IPv6	10814	165,82	77,14	54,20	67,31%	29,74%

Tabulka 1: Porovnanie pamäťových nárokov uvedených algoritmov.

5. ZÁVER

Tento článok popisuje návrh nového algoritmu LPM s cieľom poskytnúť algoritmické riešenie pre protokol IPv6 a minimalizovať pamäťové požiadavky. Návrh je vhodný pre nasadenie vo vysokorýchlostných sieťach a zohľadňuje všetky aspekty budúcej HW realizácie. Vďaka nízkym pamäťovým nárokom nám vystačí úsporná a rýchla pamäť na čipe, čo zvýši rýchlostné limity. Algoritmus sa ukazuje ako veľmi perspektívny a svojou pamäťovou efektivitou prekonáva všetky dnes známe algoritmy. Uvedený návrh bol tiež prijatý ako publikácia na IEEE konferencii DDECS [3]. Pokračovaním je skutočná HW implementácia pre technológiu FPGA, na ktorej sa v súčasnosti pracuje.

REFERENCE

[1] BGP Routing Table Analysis Reports [online]. Dostupné na URL: http://bgp.potaroo.net/

[2] Eatherton, W.; Varghese, G.; Dittia, Z.: Tree Bitmap: Hardware/Software IP Lookups with Incremental Updates. *ACM SIGCOMM Computer Communication Review*. Apr. 2004, vol. 34, num. 2. pp. 97-122. ISSN 0146-4833.

[3] Matoušek, J.; Skačan, M.; Kořenek, J.: Towards Hardware Architecture for Memory Efficient IPv4/IPv6 Lookup in 100 Gbps Networks, In *IEEE Design and Diagnostics of Electronic Circuits and Systems DDECS'2013*, Karlovy Vary, CZ, IEEE, 2013.

[4] Puš, V.; Tobola, J; Košař, V; Kaštil, J.; Kořenek, J.: Netbench: Framework for Evaluation of Packet Processing Algorithms. In *Seventh ACM/IEEE Symposium on Architecture for Networking and Communi-cations Systems (ANCS'11)*. IEEE Computer Society, Oct. 2011, pp.95-96, ISBN 978-0-7695-4521-9.

[5] Song, H.; Turner, J.; Lockwood, J.: Shape Shifting Tries for Faster IP Route Lookup. In *Proceedings of the 13TH IEEE International Conference on Network Protocols*. Washington, USA: IEEE Computer Society, 2005. pp. 358-367. ISBN 0-7695-2437-0.