KLASIFIKACIJA V HITRIH OMREŽJIH: KOMPONENTE IN SISTEMI

Mihael Kaiser, Žarko Čučej
Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Slovenija

Ključne besede: komunikacije, omrežja komunikacijska, omrežja hitra, prenosi paketni, klasifikacije paketov, prenos podatkov, arhitektura stikal, deli sestavni, prepustnost sistema, gradniki

Povzetek: V članku opisujemo problematiko klasifikacije paketov v hitrih telekomunikacijskih omrežjih (2,5 in 10 Gb/s). Zaradi vedno hitrejših fizičnih povezav in kompleksnosti komunikacijskih naprav je v vozliščih potrebne vedno več procesorske moči. Proces klasifikacije je ključni dejavnik, ki vpliva na skupno prepustnost sistema. Uvodoma uvrščamo temo s stališča telekomunikacijskih potreb in trendov, v članku samem pa se omejimo na arhitekture komunikacijskih naprav in analizo gradnikov. V splošnem so sodobna stikala v omrežnih vozliščih zgrajena iz procesorja, hitrega pomnilnika in vezij z implementirano logiko. Zahteva po vedno hitrejših in inteligentnih napravah je povzročila hiter razvoj arhitektur stikal. V članku je podan hiter pregled razvoja do pete generacije, kjer analiziramo različne kombinacije gradnikov v sistemu. Temu sledi analiza komponent za implementacijo algoritmov klasifikacije, tehnike za izboljšanje zmogljivosti ter ilustracija omejitev obstoječih sistemov. Članek zaključujemo z zahtevami in predlogi za novo generacijo stikal, kjer bo posebej izpostavljena potreba po večji fleksibilnosti komponent v smislu programljivosti. V zaključku predstavljamo še vlogo funkcijskega programiranja v teh okoljih.

Classification in High Speed Networks: Components and Systems

Key words: communications, communication networks, high speed networks, packet transmissions, packet classifications, data transmission, switch architecture, components, system throughput, building blocks

Abstract: The main topic in the paper is a problem of packet classification in high speed networks (2.5 and 10 Gbps). More processing power is needed in the nodes because of the ever increasing speed of physical links and the growing complexity of network devices. The process of classification is the essential factor impacting the throughput of whole system. It is the creator of traffic flows from users packets and requires multiple lookups per packet in different information tables (Fig. 1). A traffic flow is than elementary subject of further processing. Classical approach in building the modern high speed switches with the simple high over dimensioning of devices is no longer a promising solution for the economical use. We show that with such an approach the upper limit of the components used is reached. In the introduction the topic is classified from the point of view of telecommunication's requirements and trends. In the continuation paper describes architectures of modern communication devices and analyses the building blocks. A modern switch is typicaly built of a processor, a fast memory and integrated circuits with implemented logic. The main reason for fast evolution of the switch architectures lies in the requirement for ever faster and more intelligent devices in the network nodes. Service differentiation requires more intelligent treatment hundreds of traffic flows through the network. A preview of evolution path to the fifth generation of switch architectures is presented, with the analysis of different components which are appropriate for implementing the functionality of classification algorithms (Fig. 2). The evolution path is characterized mainly with the three bottlenecks. First, increasing speed of communication links requires intelligent interfaces. Secondly, after distribution of processing power among the main processor and intelligent interface cards a new bottleneck was evident on the interconnection path, that is a shared bus. Thirdly, when interconnection bottleneck is removed with the switch fabric the lack of processing power on interfaces became a bottleneck again. A solution has emerged in the form of the application specific integrated circuits (ASICs) which implement the time critical processing functionality in the hardware. Technics for improving the performance of classification algorithms are also shown with the illustration of the nowadays systems limitations. Among limitations the long accessing time of fast memories and the non-optimized instruction set of general processing units are the prevalent ones. The paper is concluded with requirements and suggestions for the new generation of switches, where the need for higher flexibility in the mean of programmability is exposed. Ever changing standards and market demands in the field with short time-to-market solutions require the use of highly programmable components for a fast adaptation to the new situation. A lot of expectation is put into the network processor, which combines the speed of application specific integrated circuits (ASICs) and the programmability of RISC processors. Therefore the functionality and the architecture of a network processor is described (Fig. 3), which is according to high investments possibly the hottest area in the processor industry today. The importance of functional programming in these environments is the closing theme.

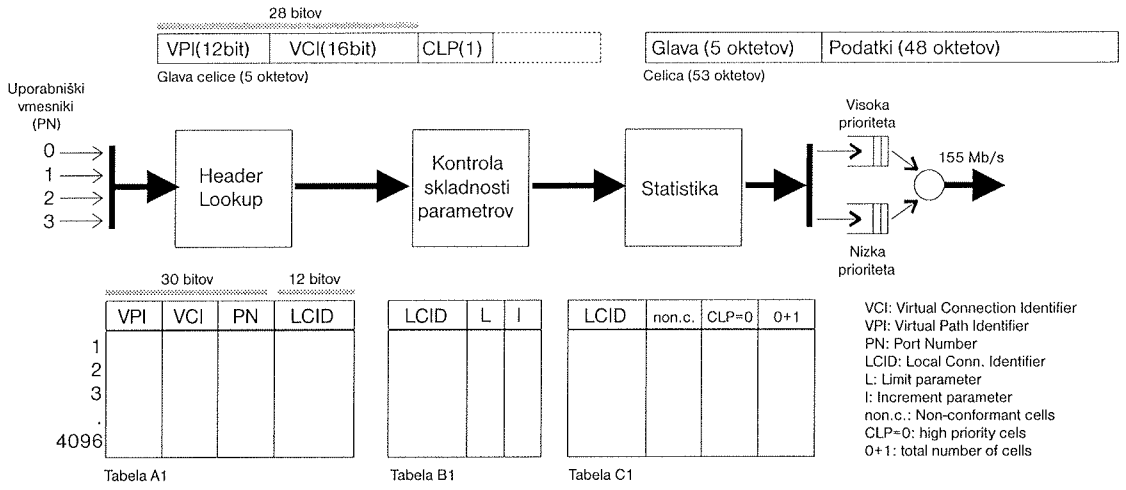
1. Uvod

Živimo v obdobju, ko število uporabnikov svetovnega omrežja še vedno eksponentno narašča. Brez dvoma je eden glavnih dejavnikov za takšno stanje komercializacija Interneta. Vedno več je podjetij, ki imajo svoje poslovanje delno ali pa kar v celoti organizirano čez svetovno omrežje. To nenazadnje pomeni, da so pripravljena plačevati zanesljivost storitev čez omrežje. Seveda pa si različna podjetja *zanesljivost* različno predstavljajo. Največkrat je zahtevana ustrezna prepustnost zvez, da bodo naročniki ob pregledovanju spletnih strani tega podjetja deležni konstantne zmogljivosti sistema (hitre strani). Pri drugih morda uporabljajo videokonferenčno sestankovanje kot alternativo za visoke potne stroške in kronično pomanjkanje časa. Zato je zanesljivost storitve v tem primeru ovrednotena v povezavi s kakovostjo prenašane video slike. Spet tretji potrebujejo le stalen dostop do Interneta. Povsem logično je, da zanesljivost storitev čez omrežje nima ene same cene; vpeljano je ti. razlikovanje med storitvami.

V tehničnem smislu pomeni diferenciacija[1] med storitvami vpeljavo mehanizmov, s katerimi ima omrežje možnost krmiliti porabo skupnih omrežnih virov[2] za posameznega uporabnika oz. njegov prometni tok. Upravljanje prometa v omrežjih je skupno ime za nabor teh mehanizmov. Hiter pregled delovanja teh mehanizmov si oglejmo na primeru preprostega multiplekserja prometnih tokov v večstoritvenem omrežju ATM (Asynchronous Transfer Mode), ki je prikazan na sliki 1.

Prikazan multiplekser zbere štiri fizične prometne tokove z vhodnih vmesnikov v skupen izhodni vmesnik kapacitete 155 Mb/s. Komunikacija čez omrežje ATM poteka po navideznih kanalih (Virtual Channel Connection, VCC) in po navideznih poteh (Virtual Path Connection, VPC); le te tvori več navideznih kanalov (agregacija). Na sliki 1 je prikazan zbirni element, ki podpira 4096 navideznih kanalov s skupno prepustnostjo 155 Mb/s. Paketi so fiksne dolžine, ki znaša 53 oktetov. Komunikacija se začne po predhodni vzpostavitvi zveze iz navideznih povezav (VCC ali VPC) čez omrežje. Vsaka celica vsebuje glavo, dolgo 5 oktetov (glej sliko 1), v kateri so med drugim naslednja tri polja: polje VPI (Virtual Path Identifier) vsebuje identifikator navidezne poti, VCI (Virtual Channel Identifier) je identifikator navideznega kanala, po katerem potuje celica, CLP (Cell Loss Priority) je zastavica prioritete[3].

glave celice v tabeli A1 (ti. Header Lookup). V primeru, da iskane kombinacije ni, se celica takoj zavrže. Običajno pa zadetek vedno obstaja; tedaj se zgodi transformacija dolgega naslovnega polja (30 bitov) v krajši lokalni[4] identifikator (LCID, ki je za 4096 povezav dolg le 12 bitov). Smisel transformacije je krajši in enojen ključ, zato je v nadaljevanju iskanje po tabelah hitrejše. Sledi kontrola skladnosti parametrov prometa. Splošen promet v sodobnih omrežjih ima značilnosti rafala; pojavlja se neenakomerno, ko pa nastopi, zasede kanal v izbruhu. Zato s posebnim algoritmom[5] primerjamo trenutne vrednosti parametrov prometa z dogovorjenimi vrednostmi ob vzpostavljanju zveze. Algoritem skladnosti se najprej inicializira z vrednostima I in L iz tabele B1 (za aktualen LCID), nakar se meritev skladnosti tudi izvede. V primeru negativnega izida (prehitra celica in/ali predolg izbruh celic) se celico takoj zavrže. Sicer sledi faza vpisa statistike za izbran prometni tok. Odvisno od izida prejšnje faze se osvežijo polja v tabeli C1 za aktualen LCID. Temu sledi še razvrstitev celic v čakalne vrste. V obravnavanem primeru imamo samo dve čakalni vrsti (za celice z višjo prioriteto in preostale), nič nenavadnega pa ni, če jih je več deset do več sto (večja razdrobljenost, širši spekter storitev). Sledi še strežba celic iz čakalnih vrst, najprej tiste iz vrste z višjo prioriteto, ter njihovo posredovanje po paralelno serijski pretvorbi čez izhodni port naprej v omrežje.



Slika 1: Preprost zbirni element v večstoritvenem omrežju ATM.

Vsi štirje fizični tokovi se najprej pretvorijo iz serijskega v paralelen format in se zberejo na skupnem hitrem vodilu. Prične se procesiranje (glave) celice, ki poteka v več fazah. Najprej se poišče dana kombinacija VPI, VCI, PN polj iz

Na kratko, omrežje na osnovi informacije v glavi paketa in informacije iz podatkovne baze (ki jo vzdržujejo npr. krmilni protokoli) najprej prepozna prometni tok, tako identificiranemu paketu določi lokalen identifikator (LCID) za

1 Diferenciacija v splošnem pomenu. Diff. Serv., kot jo definira Internet Engineering Task Force (IETF), je le poseben primer modela storitev.

2 Najbolj značilna (dragocena) skupna omrežna vira sta pasovna širina in pomnilniški prostor.

3 Cell Loss Priority (CLP); če lokalno v omrežju nastopi zamašitev, se prične najprej z izločanjem celic s postavljeno zastavico CLP, šele nato po potrebi tudi prednostni promet (celice s CLP=0).

4 Lokalnost v smislu procesiranja celic znotraj ene naprave.

5 Leaky Bucket (počeno vedro) je standarden algoritem za kontrolo skladnosti parametrov prometnega toka. Parametra I (Increment) in L (Limit) določata minimalen časovni razmik med dvema zaporednima celicama in največjo dolžino izbruha za dan prometni tok.

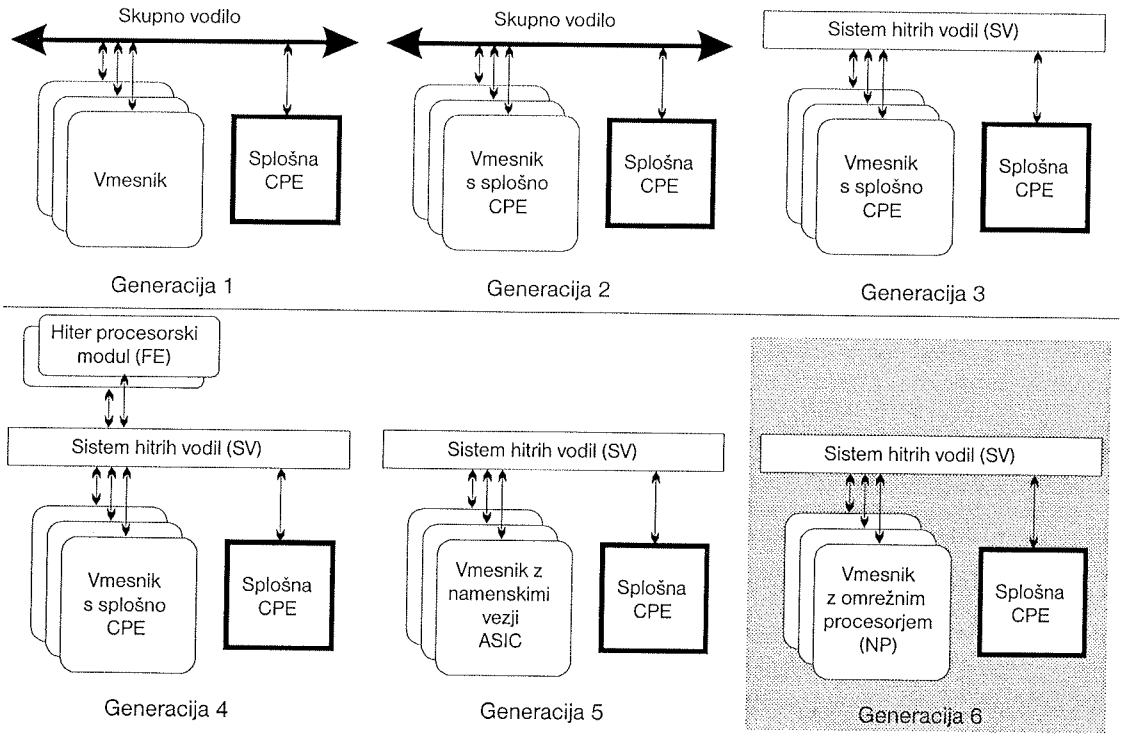
hitrejše iskanje zapisov v tabelah v kasnejših fazah procesiranja tega paketa.

Trend naraščanja števila uporabnikov Interneta smo že omenili. Dodajmo še ugotovitev, da poraba pasovne širine na aplikacijo/uporabnika prav tako narašča, obenem se v sodobnih omrežjih zahteva še diferencijacija med storitvami. Posledica tega so vedno hitrejša omrežja, kjer pa postaja problematično dejstvo, da je v takem omrežju potrebno v vedno krajšem času narediti vedno več. V tem članku opisujemo problematiko klasifikacije prihajajočih paketov v hitrih omrežjih, saj je le-ta ključni dejavnik, ki vpliva na skupno prepustnost sistema. V drugi sekciji je povzetek razvoja arhitektur hitrih stikal. V tretji sekciji analiziramo komponente in različne tehnike za izboljšanje algoritmov klasifikacije z ilustracijo omejitev. V četrti sekciji so zbrane zahteve za omrežni procesor in opis šeste generacije arhitektur.

2. Razvoj arhitektur hitrih stikal

Napredek v tehnologiji izdelovanja in uporabe optičnih vlaken je imel za posledico, da se prav vlakna množično uporabljajo kot najbolj primeren fizični medij v svetu sodobnih komunikacij. Še več, njihova teoretična pasovna širina za nekaj razredov presega sposobnosti obstoječih omrežnih stikal, kjer je v naslednjih dveh letih pričakovati naprave s prepustnostjo 2,5 oziroma 10 Gb/s /6/. Tako so ponovno naprave tiste, ki predstavljajo ozko grlo v sodobnih telekomunikacijskih sistemih. Potreba po vedno večji hitrosti procesiranja paketov pa ne prihaja samo zaradi naraščajočih hitrosti komunikacijskih linij (in posledično vmesnikov), temveč tudi zato, ker postajajo naprave v vozliščih vedno večje /1/. Zaradi velikega števila vmesnikov prihaja do problemov hitre strežbe v tako velikem sistemu. Pri načrtovanju sododnih stikal imamo na izbiro številne kombinacije gradnikov (vsebina 3. sekcije), v glavnem pa je potrebno paziti, da bo nastali izdelek zanesljiv in hiter. Prvo zahtevo izpolnimo s podvajanjem strojnih delov, težava pa nastopi pri hitrosti. Prav zahteva po vedno večji prepustnosti naprav je glavni motiv razvoja komunikacijskih stikal. V nadaljevanju podajamo kratek povzetek tega razvoja (slika 2).

Arhitektura stikal in usmerjevalnikov prve generacije je bila načrtovana s poudarkom na zanesljivosti vzpostavljanja in vzdrževanja zvez, hitrost sistema je bila drugotnega pomena. Sistem je sestavljen iz množice vhodno/izhodnih vmesnikov, ki so z vodilom povezani s centralno procesorsko enoto (CPE). Paket vstopi skozi vhodni vmesnik, po serijsko paralelni pretvorbi ga le-ta posreduje čez vodilo do CPE. Tam se določi naslov izhodnega vmesnika (ti. header lookup), odvisno od naslova končne destinacije, ki se nahaja v glavi paketa. CPE nato posreduje paket čez vodilo do izhodnega vmesnika in naprej v omrežje. Zmogljivost opisanega sistema je odvisna od prepustnosti vodila (vsak paket gre dvakrat čez vodilo) in hitrosti procesiranja paketa v CPE. V času nastanka te arhitekture je bilo zelo pomembno obdržati fleksibilnost sistema zaradi nenehno spreminjajočih se komunikacijskih protokolov. Zato je bil izbran izrazito centraliziran sistem s splošno CPE, saj zaradi stalnih sprememb ni bilo primerno naprav optimirati za določen protokol. To je obenem tudi čas intenzivnega razvoja večopravilnih operacijskih sistemov za delo v realnem času /9/



Slika 2: Razvoj arhitektur komunikacijskih stikal

Skupno vodilo je bilo najbolj izrazito ozko grlo, ki so ga skušali odpraviti s stikali druge generacije. Z inteligentnimi vmesniki se procesiranje paketov porazdeli med glavnim procesorskim modulom in vmesniki, ki so opremljeni z lastno CPE in predpomnilnikom. S tem se posredno sprosti tudi vodilo. Z bolj inteligentnim pristopom se da večino paketov posredovati z vhodnih vmesnikov do izhodnih kar lokalno, brez posredovanja glavne CPE. Pri načrtovanju te arhitekture je namreč upoštevano dejstvo, da pri večini zvez pride do prenosa velikega števila paketov z enakim končnim naslovom, zato je smiselno vzdrževati del podatkovne baze o aktualnih naslovih lokalno na vmesnikih v predpomnilniku. Tako pride do posredovanja paketa z vhodnega vmesnika čez vodilo do glavne CPE in nazaj čez vodilo do izhodnega vmesnika le pri prvem paketu neke zveze. Vrnjena informacija (številka izhodnega vmesnika s pripadajočim končnim naslovom) se vpiše še v lokalno bazo na vmesniku, zato ta vhodni vmesnik vse naslednje pakete z istim končnim naslovom pošlje čez vodilo neposredno do ustreznega izhodnega vmesnika. Večina paketov vodilo zaseda le enkrat, v predpomnilniku vmesnika pa je prostora za več 10 do več 100 naslovnih parov. Prav omejenost predpomnilnika pa povzroča, da je zmogljivost te arhitekture odvisna od narave prometa (glej sekcijo 3).

V stikalih tretje generacije je skupno vodilo zamenjal sistem hitrih vodil (ti. switch fabric), ki je popolnoma odpravil ozko grlo na notranjih povezavah med vmesniki in glavno CPE. Sistem vodil (SV) je sposoben prenašati pakete nekajkrat hitreje, kot jih je sposobna dostavljati (in obdelati) katerakoli CPE. Zato se je iz te arhitekture kmalu razvila nova generacija, ki bolje izkorišča veliko prepustnost SV.

V poskusih, da bi povečali prepustnost inteligentnih vmesnikov, je bilo potrebno nadomestiti klasično CPE. Vendar namestitev hitrih in zato dragih procesorjev na vsak vmesniški modul ne da komercialno zanimivega izdelka. Stikala so v splošnem drage naprave, zato je strošek na vmesnik pomemben indikator pri odločanju za široko uporabo. Zato so dodali v stikala četrte generacije sklop modulov s hitrimi procesorji za posredovanje pri procesiranju paketov (ti. forwarding engine, FE). Gre za implementacijo koncepta paralelnega procesiranja paketov; FE je večprocesorska enota hitrih procesorjev, ki je čez sistem hitrih vodil SV dostopna vsem vmesnikom. V primeru polne zasedenosti procesorske moči lokalno na vmesniku le-ta na novo prispelega paketa ne postavi v čakalno vrsto, temveč posreduje glavo paketa čez SV do FE in ta pridobi informacijo o izhodnem vmesniku, ter le-to vrne do vhodnega vmesnika, od koder je zahteva za pomoč prišla. Vhodni vmesnik nato na osnovi vrnjene informacije izvede posredovanje celotnega paketa do izhodnega vmesnika. Vmesniki s cenenimi CPE so še vedno sposobni lastnega procesiranja paketov (kot v tretji generaciji), vendar le za neko povprečno zasedenost kanala. Ko pa nastopi polna zasedenost (vršna bitna hitrost) pa lokalna CPE višek vhodnih paketov posreduje do FE. Arhitektura temelji na dejstvu, da vsi vmesniki ne bodo hkrati polno zasedeni, zato je draga inteligenca v obliki FE dana v souporabo. To je tudi eden izmed načinov za dosego nizkih stroškov na vmesnik.

V tej arhitekturi je prvič definirana počasna in hitra pot posredovanja paketov skozi stikalo. Vsak paket, ki se ga neposredno prenese z vhodnega na izhodni vmesnik, brez posredovanja CPE, je posredovan po hitri poti. Če pa posreduje CPE (npr. s strani FE) pravimo, da je šel paket po počasni poti. V interesu oblikovalcev arhitekture je, da zadržijo čimveč paketov v hitri poti in da je skupen procesorski modul FE čimveč v uporabi. Nastopi konfliktna situacija, saj vsaka uporaba FE pomeni prenos po počasni poti. S tem smo že napovedali potrebo po spremenjeni arhitekturi.

V peti generaciji komunikacijskih stikal klasičen CPE na vmesnikih zamenja mnogo hitrejše in cenejše namensko integrirano vezje s fiksno funkcionalnostjo (ASIC). S tem se iz hitre poti povsem izloči pomembna programljiva komponenta, namreč CPE. Vsi podatkovni paketi se prenašajo po hitri poti, po počasni poti gredo le krmilni paketi. Arhitekturo pete generacije ima večina današnjih stikal in usmerjevalnikov z ločenim delom za usmerjanje paketov (počasna pot) in delom za posredovanje (hitra pot).

3. Osnovne komponente in omejitve sistemov (diskusija)

V prejšnji sekciji smo prikazali pregled razvoja arhitektur omrežnih stikal, ki je posledica nenehnega prilagajanja vedno kompleksnejših naprav za delovanje v vedno hitrejših omrežjih. Izkaže se, da je najbolj kritično ozko grlo pri strežbi v hitrih omrežjih čas, potreben za določitev naslova naslednjega skoka /10/. Novejša stikala bodo podpirala poleg tega še diferencirane storitve, kar pomeni, da se pojavijo poleg klasičnega preiskovanja po bazi naslovov še dodatna preiskovanja zaradi potrebe po filtriranju in klasificiranju prihajajočih paketov (slika 1).

A. Proces klasifikacije

Iz primera v prvi sekciji (slika 1) je razvidno, da je proces posredovanja paketa (hitra pot) v začetni fazi sestavljen iz ti. Header Lookup-a (HL), ki mu sledi še klasifikacija paketa. Oba postopka sta si podobna, zato poglejmo najprej prvega. HL je postopek preiskovanja po tabeli naslovov destinacij (oziroma njihovih okrajšav, ti. prefiksov[6]) s ključem naslova z namenom najti naslov naslednjega skoka. Iskanje se konča pri tistem zapisu, ki je enak (se najbolje ujema) naslovu destinacije iz glave prispelega paketa. Tabela naslovov vsebuje več tisoč (lahko tudi več sto tisoč) zapisov. Ker pa so naslovi v tabeli zapisani v obliki prefiksov, je zadetkov v splošnem več. Veljaven je zapis iz tabele z najdaljšim prefiksom (Najboljše ujemanje). Dober pregled

6 Pojem prefiksa najenostavneje razložimo na analogiji s telefonskimi številkami, kjer so omrežne skupine prefiksi dolžine dve.

problematike HL v okolju s protokoli IP in ustreznih postopkov najdemo v /15/. Postopku HL sledi postopek filtriranja in klasifikacije paketov, ki je definiran kot identifikacija prometnega toka (na osnovi pregledovanja vsebine glave paketa). Za vsak prometni tok je nato v tabeli storitev definirana akcija, ki naj se izvede nad tem paketom, da bodo izpolnjene zahteve storitve, ki ji prepoznan podatkovni tok pripada. Ugotovimo, da gre pri obeh postopkih, HL in klasifikaciji paketov, za iskanje pravega zapisa v podatkovni bazi glede na vhodno vrednost ključa, ki se skriva v glavi prispelega paketa. Zato v nadaljevanju podajamo poenoten opis komponent za implementacijo kateregakoli od obeh postopkov.

B. Komponente za implementacijo z omejitvami

Klasifikacija v praksi pomeni iskanje po tabeli z več komponentnim ključem; vrednosti posameznih komponent (polj) prinese paket v svoji glavi. Najde se identifikator akcije, ki se naj nad tem paketom izvede (npr. zavrži, znižaj prioriteto, označi, brez akcije...). V bistvu gre za preslikavo sestavljenih in dolgih ključev v enojne in krajše. Na sliki 1 je prikazan primer tri-komponentnega ključa (VPI, VCI, PN) v skupni dolžini 30 bitov, ki ga preslikamo v enojni ključ (LCID) dolžine 12 bitov. Da bo algoritem klasifikacije praktično uporaben v hitrih omrežjih, se zanj zahteva predvsem kratek iskalni čas in majhna velikost uporabljene podatkovne strukture za implementacijo tabele (oziroma v splošnem baze) /5/. Obe izpolnjeni zahtevi omogočata hranjenje tabel v hitrem pomnilniku. Hitrost pregledovalnega algoritma je odvisna od števila poskusov, preden najdemo iskan zadetek (branje vsebine pomnilnika), in hitrosti dostopa do hitrega pomnilnika. Med časovno kritične operacije spada poleg branja še operacija osveževanja zapisov tabele. Merjenja prometnih vzorcev v hitrih omrežjih kažejo, da je večina podatkovnih paketov TCP/IP narave (90% vseh paketov), od tega približno 45% takih, ki so dolgi le 40 oktetov (paketi za potrditev) /18/. /10/ navaja povprečen čas, v katerem pride do zahteve po osveževanju tabele naslovov okoli 2 minuti. Iz opisanega sledi, da je za implementacijo podatkovne baze (tabel) priporočljivo uporabiti zapletene podatkovne strukture, s katerimi optimiramo čas iskanja pri branju iz baze na račun časovno daljših operacij osveževanja. Standardna podatkovna struktura je drevo /15/. Posamezen zapis v tej strukturi ustreza poti od debla do enega od listov. Daljši kot je prefiks, bolj specifičen je naslov, iz večih vej je sestavljena pot. Struktura torej podpira komprimiran zapis, zato je varčna glede pomnilniškega prostora za hranjenje celotne baze. Vendar je takšna rešitev kljub varčnosti na račun večkratnih dostopov pri branju slabost za uporabo v sodobnih sistemih, saj cene hitrih pomnilnikov stalno upadajo.

V prvih generacijah stikal in v večini današnjih manjših sistemov naprave zaradi zahteve po nižji ceni izkoriščajo dejstvo, da se komunikacija v določenem časovnem obdobju odvija le na delu zvez. Uporaba predpomnilnika (ti. cache) omogoča, da se na vmesniku namesto celotne podatkovne baze vanj shranjujejo zapisi le za aktivne zveze. Na ta način dobimo visoko zmogljiv sistem z manj hitrega pomnilnika. Slaba stran te tehnike je, da je zmogljivost sistema odvisna od narave prometa; tehnika daje dobre rezultate za primere, ko so podatkovni tokovi stalni. Če pa prihaja le do kratkih komunikacij, je frekvenca pojavljanja novih tokov (in s tem novih zapisov v hitrem pomnilniku) velika, zato vedno pogosteje prihaja do počasnega osveževanja (prenašanja) duplikatov zapisov iz originalne baze v predpomnilnik in učinek predpomnilnika je močno zmanjšan.

Naslednja tehnika za izboljšanje zmogljivosti algoritmov klasifikacije je uporaba vsebinsko naslovljivega pomnilnika (ti. CAM). CAM je pomnilnik z integrirano logiko za hitro iskanje po pomnilniku. Najprej ga je potrebno napolniti, v fazi uporabe pa za iskan podatek vrne naslov lokacije. Preiskovanje se izvrši v enem ciklu, saj se iskan podatek išče hkrati po celotnem polju. V primeru, da je zadetkov več, prioritetni dekoder izbere en zadetek. Dobra lastnost teh elementov je, da podatkov ni potrebno sortirati ali indeksirati. Zaradi paralelnega iskanja imajo večjo porabo moči, kar je njihova največja ovira pred množično uporabo v velikih sistemih, kjer so tabele (in s tem pomnilniško polje) velike. /11/ navaja primer klasifikacije v 10Gb/s sistemu, kjer ob hitrosti 25 milijonov naslovov na sekundo za pregledovanje po tabeli, veliki 1 milijon zapisov, bi za rešitev s CAM ta porabil 50W moči, medtem ko z uporabo namenskih procesorjev dosežejo enako zmogljivost s porabljeno močjo pod 10W. Poleg navadnih CAM, ki pomnijo le binarno informacijo, se izdelujejo še ternarni CAM (ti. TCAM), kjer je možno pomnenje tudi nedefiniranega stanja /14/. Zato porabijo manj pomnilniškega prostora za pomnjenje prefiksov, saj ni potrebno shranjevati poleg naslova še maske. V primerjavi z SRAM so dražji; TCAM, 100MHz, 2Mb, 14W stane 70USD, medtem ko SRAM, 200MHz, 8Mb, 2W stane 30USD (marec 2001, /5/).

Zelo uporabna tehnika za hitro iskanje je tudi ti. razpršena tabela (Hash Table). Njen koncept opisujemo s pomočjo tabele A1 na sliki 1. Najkrajše iskanje po tabeli dobimo, če paket v svoji glavi prinese vrednost indeksa (ključa) v tabeli. Vendar takšno neposredno iskanje pride v upoštev le v primerih, ko dolžina N tabele približno ustreza številu vseh kombinacij n ključa; v tabeli A1 je dolžina ključa 30 bitov, torej je vseh možnih kombinacij ključa $n=2^{30}$. Tabela A1 pa je dolga samo $N=2^{12}$ (=4096) zapisov. Če bi uporabili neposredno indeksiranje, bi morala biti tudi tabela A1 dolga 2^{30} zapisov, od katerih je le 2^{12} (=0,0004%) dejansko koristnih. Če bi uporabili CAM dolg 4096 zapisov problem velike neizkoriščenosti pomnilnika odpade, saj pri CAM sortiranje ni potrebno. Razpršena tabela izkorišča podobno metodo, kjer s pomočjo posebne razpršitvene funkcije (Hash Function) vseh 2^{30} vhodnih kombinacij čimbolj enakomerno razdelimo med 2^{12} vrednosti. Pomembno pri tem je, da čim redkeje dve različni vhodni vrednosti preslikamo v isto izhodno vrednost (trk). Verjetnost zaporednih trkov je odvisna od izbrane razpršitvene funkcije. Verjetnost trka se še dodatno zmanjša, če uporabimo več zaporednih preslikav. Prav pojav trkov povzroča nedeterminizem iskalnega časa, kar omejuje uporabnost te softverske tehnike v hitrih sistemih.

Literatura navaja še številne hibridne tehnike (zgoraj omenjenih). Dober pregled slednjih navaja /5/.

C. Ilustracija omejitev

V tabeli 1 so zbrani časi dostopa (enkratno branje) za različne hitre pomnilnike, ki se uporabljajo v današnjih sistemih /17/. Če imamo linijo s kapaciteto 10 Gb/s in želimo realizirati posredovanje kratkih paketov (40 zlogov) pri polni hitrosti, potem je izračunan strežni čas enak 10Gbps / 320b, kar je približno 32 ns. Z ozirom na tabelo 1 bi morali z uporabo pomnilnika SRAM že v nekaj poskusih najti pravi zapis, saj vsota dostopnih časov ne sme preseč izračunanih 32 ns. Praktično bi bil uporaben le vsebinsko naslovljiv pomnilnik (CAM).

pomnilnik	čas enkratnega dostopa [ns]
DRAM	60-100
SRAM	10-20
on-chip DRAM	10
on-chip SRAM	1-5
CAM	15

Tabela 1: različni bralni časi (enkraten dostop) za različne hitre pomnilnike /17/.

Za še boljšo ilustracijo omejenosti obstoječih sistemov podajamo informativni izračun idealiziranega sistema za posredovanje kratkih paketov z eno samo vhodno/izhodno linijo (vmesnikom) kapacitete 1Gb/s, hitrim pomnilnikom in CPE, ki so povezani na hitro vodilo.

širina hitrega vodila:	64 bitov
frekvenca vodila:	50 Mhz
=> pasovna širina vodila	=50Mhz x 64b = 3.2Gb/s

Pasovna širina hitrega vodila je 3.2 Gb/s, od tega je 1 Gb/s porabi vhodni paketni tok, enako izhodni. Tako ostane proste pasovne širine za komunikacijo CPE s hitrim pomnilnikom (preiskovanje in klasifikacija) le še ena tretjina, to je 1.2 Gb/s.

velikost kratkega paketa:	40B = 320 bitov
pas. širina vh./izh. vmesnika:	1 Gb/s
=> prosta p.š. vodila	= 3.2–1–1 = 1.2Gb/s

Če preračunamo kapaciteto vhodnega vmesnika (za 320 bitne pakete) in največjo paketno hitrost na 64 bitov širokem paralelnem vodilu, izraženo v paketih, ugotovimo, da ima CPE na voljo največ 6 operacij čez vodilo do hitrega pomnilnika na vsak paket. Če odmislimo osveževanje tabele to pomeni, da je potrebno v nekaj 1000 vrstic dolgi tabeli v najslabšem primeru v šestih poskusih najti pravi zapis, kar je iluzorno.

vh. paketna hitrost:	=1Gbps/320b=3.125M pk./s
max. pak. hitrost na vodilu:	=1.2Gbps/64b=18.75Mop./s
=> max. št. operacij / paket	= 18.75M / 3.125M = 6

Če je uporabljena CPE 200 MHz procesor, ki ima instrukcijski cikel enak strojnemu ciklu za cel nabor instrukcij, potem je največje število instrukcij, ki jih procesor sme porabiti pri obdelavi kratkega paketa za strežbo pri polni hitrosti linije, enako 64. Tudi tukaj je nesmiselno pričakovati asemblersko kodo za delo z drevesno strukturo, dolgo vsega 64 instrukcij.

sistemska frekvenca CPE:	200MHz (1 instrukcija / cikel)
=> max. št. instr. / paket	=200M / 3.125M = 64

Iz prikazanega sledi, da je v hitri arhitekturi poleg hitrega procesorja najdragocenejši skupni vir še prosta pasovna širina notranjega vodila za komunikacijo CPE s pomnilnikom. Raziskave kažejo, da je za strežbo linije s kapaciteto 10 Gb/s potrebna prosta pasovna širina za komunikacijo s hitrim pomnilnikom okoli 500 Gb/s, pri tem je potrebnega še vsaj 128 Mb hitrega pomnilnika /4/. Zato je tehnološki trend čimmanj krat dostopati do pomnilnika. Rešitve se nakazujejo v obliki zelo širokih vodil za dostop (360 bitov), oziroma zelo širokih on-chip pomnilnikov (16 Mb, 512b širine, /4/).

4. Zahteve in predlogi novih sistemov

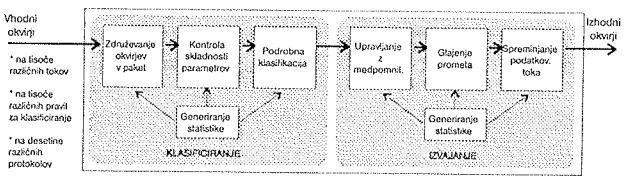
Načrtovalci sodobnih komunikacijskih naprav za uporabo v hitrih omrežjih imajo pri svojem delu na izbiro veliko možnosti, da z novim sistemom optimalno zadostijo postavljenim zahtevam. Izziv je čimbolje in za čim daljši čas zadostiti prisotnim kompromisom. V drugi sekciji smo prikazali razvoj arhitektur stikal, ki ga je povzročila zahteva po vedno hitrejši strežbi. Kar pet generacij se je zvrstilo v komaj dobrem desetletju, zato se samo po sebi postavlja vprašanje, kako dober je koncept slednje in kako dolgo bo uspešno kljubovala.

A. Hitrost ali programljivost

Po eni strani se je okoli ideje o uporabi strojno implementirane funkcionalnosti v časovno najbolj kritičnih delih sistemov razvila zelo močna industrija integriranih vezij s fiksno funkcionalnostjo (ASIC). Prav z uporabo gradnikov z nespremenljivo funkcionalnostjo pa tvegamo zastarelost sistema še pred njegovo široko proizvodnjo. Vezja ASIC sicer podpirajo zelo hitro procesiranje paketov (ti. wire-speed performance) in so zelo ekonomična v smislu porabljene silicijeve površine na funkcionalnost in porabe električne energije, sam razvoj takšnega vezja pa traja od 12 do 18 mesecev, zato ne omogočajo hitrega prilagajanja novim zahtevam na tržišču. Obenem zaradi dolgega razvojnega obdobja veliko proizvajalcev komunikacijskih sistemov uporablja v svojih izdelkih vezja ASIC, ki jih kupijo na trgu. S tem se zelo zmanjša možnost konkurenčnega boja, saj je iz enakih gradnikov fiksne funkcionalnosti težko narediti izdelek, ki se bo bistveno bolje obnašal od konkurentovih.

Po drugi strani je programljivost komponent najboljše jamstvo za fleksibilnost sistema. Z vključitvijo RISC procesorjev v sisteme pridobimo (navidezno) univerzalnost. S tem

pa se žrtvuje toliko na hitrosti sistema, da je le-ta zastarel že v konceptu. To jasno kaže tudi razvoj arhitekture stikal, ki smo ga opisali v drugi sekciji. Zato se veliko pričakuje od novega gradnika, ki bo bistveno pripomogel v naslednji (šesti) generaciji arhitektur stikal. Glavne zahteve za nov gradnik izvirajo iz izkušenj in omejitev prejšnjih generacij in so naslednje /12/: (1) nov gradnik mora podpirati osnovne funkcije za posredovanje paketov; (2) njegovo mesto je v hitri poti (zamenjava za ASIC, glej sliko 2), zato mora delovati "at-wire-speed"; (3) biti mora enostavno programljiv in visoko programljiv; (4) njegova funkcionalnost naj bo razširljiva za uporabo v velikih sistemih tudi čez čas. Nov gradnik, s katerim vodilna svetovna podjetja procesorskih vezij napovedujejo rešitev kompromisa hitrost-fleksibilnost, se imenuje omrežni procesor (Network Processor, NP). NP podpira implementacijo časovno kritične funkcionalnosti pri posredovanju paketov v hitri poti v obliki programov, ki se izvajajo na namensko razvitem procesorskem sistemu /1/. Nov gradnik omogoča dodajanje, razširitev in spreminjanje funkcij tretje do sedme plasti OSI modela v obliki programov, s čemer pridobimo na fleksibilnosti v primerjavi z dolgotrajnim in dragim ponovnim oblikovanjem novega ASIC vezja. NP podpira polno funkcionalnost posredovanja paketov v hitri poti s hitrostjo ASIC vezij in s programljivostjo RISC vezij /12/.



Slika 3: Funkcijska shema omrežnega procesorja

B. Osnovne funkcije omrežnega procesorja

Slika 3 prikazuje razporeditev osnovnih funkcij NP. Najprej se zgodi klasifikacija, kjer NP na osnovi informacije v glavi paketa in ustrezne podatkovne baze določi, kako bo paket obdelan in posredovan naprej. Te aktivnosti se nato zgodijo v fazi izvršitve. Bistveno pri tem je, da je funkcionalnost programljiva (3. zahteva) in to opisujemo v nadaljevanju. Večina sodobnih omrežnih tehnologij razdeli daljše pakete pred prenosom po fizičnem mediju v manjše okvirje, zato je prva naloga klasifikacije, da okvirje pravilno sestavi v prvoten paket, kot je določeno v programu. Nato se preveri skladnost parametrov paketa (najmanjša dovoljena zakasnitev, največje dovoljeno trepetanje zakasnitve, največja dolžina izbruha) s predhodno dogovorjenimi vrednostmi, z algoritmom, ki je trenutno sprogramiran. Temu sledi podrobna klasifikacija, kjer na osnovi parametrov kvalitete storitev razvrstimo paket v ustrezen razred storitve (čakalna vrsta z ustrezno prioriteto in hitrostjo strežbe). Tukaj sprogramiramo lokacijo posameznih atributov (polj) znotraj paketa, njihovo število in seveda sam klasifikator, ki je v splošnem tabela pravil.

V fazi izvršitve se izvede strežba paketov iz vrst skladno s ti. statističnim multipleksiranjem, kjer se alocira le srednja vrednost potrebovane pasovne širine, presežki paketov pa se začasno zadržijo v medpomnilniku. Zato sta za to fazo značilni funkciji upravljanja z medpomnilnikom in glajenje prometnega toka, s katerim popravimo časovne parametre prometnega toka, preden le-ta zapusti stikalo. S funkcijo modifikacije prometnega toka spreminjamo vsebino določenih krmilnih polj v paketu, dodajamo oziroma odvzemamo dele paketa (pomožna glava) in razstavimo dolg paket v krajše okvirje. V obeh sklopih (klasifikacija in izvršitev) je prisotna še funkcija zbiranja statistike na posamezen prometni tok, ki igra ključno vlogo pri trženju storitve in alarmiranju v sistemu. Ugotovljeno je, da so funkcije izvršitve podobno kot klasifikacija predmet številnih sprememb in dopolnitev znotraj standardov ter novih potreb tržišča in dognanj znanosti, zato je visoka programljivost teh delov izrednega pomena za proizvajalca komunikacijskih naprav.

C. Arhitektura omrežnega procesorja

Arhitektura NP je značilno deljena v dva dela. Za strežbo v počasni poti je namenjena splošna CPE, ki komunicira s krmilnimi protokoli in vzpostavlja ter osvežuje podatkovne strukture. Za posredovanje paketov v hitri poti pa je na voljo procesorski kompleks (PK) in stroj za hitro pregledovanje in klasifikacijo (SPK) ter hiter medpomnilnik za pakete (MP) /1/. SPK je ključni element v hitri poti. Predstavlja implementacijo najsodobnejših iskalnih algoritmov za neposredno preiskovanje, preiskovanje prefiksov in preiskovanje do pet-dimenzionalnih območij. Za ilustracijo navajamo nekaj številk /1/: sposobnost klasificiranja do 26 milijonov paketov na sekundo, preiskovanje po tabelah, dolgih 100.000 prefiksov, obvladuje pet-dimenzionalne klasifikatorje na do 6.000 prioritetnih nivojih. Algoritmi so optimirani za strojno implementacijo in za delo s komprimiranimi tabelami, zato so tudi varčni glede porabe pomnilniškega prostora in omogočajo zelo kratke osveževalne čase. Skratka, 3D optimirani algoritmi: velika hitrost delovanja, majhen potreben pomnilnik za hranjenje podatkovnih struktur in hitro osveževanje le-teh.

PK je večprocesorski sistem (do 16 jeder) z optimiranim naborom instrukcij za delo v hitri poti (podobno kot so DSP-ji optimirani za obdelavo signalov). Pri delu s klasičnimi postopki (izračun zaščitne kode, dekodiranja) mu pomaga več koprocesorjev (ti. Hardware Assist). Pri tem je PK brez čakalnih ciklov (izvajanje obdelave večih paketov sočasno v več nitkah, ali pa cevna arhitektura). Današnji PK-ji imajo procesorsko moč ene miljarde instrukcij na sekundo /1/, kar zadostuje za strežbo v sistemih s prepustnostjo 2,5 Gb/s.

D. Funkcijsko programiranje

Izkušeni programerji vedo, da kvalitetno programiranje časovno kritičnih postopkov zahteva optimiranje kode na arhitekturo procesorja. Zaskrbljujoč je že hiter pregled opisa arhitekture omrežnega procesorja, kaj šele pisanje programov in nadgrajevanje ter testiranje. V tretji sekciji smo navedli definicijo klasifikacije, ki v povzetku pravi, da je to postopek, kjer pregledujemo paket, deloma ali v celoti, in

na osnovi tega določimo eno ali več možnih akcij, ki naj se nato izvedejo. Vidimo, da je klasifikacija idealna za opisovanje na način "kaj se naj zgodi", v članku pa smo prikazali problem "kako specificirati" klasifikacijo. Pri funkcionalnem programiranju navedemo spremenljivke in določimo njihov medsebojni odnos, prevajalnik pa nato sam uporabi niz fiksnih (visoko optimiranih) algoritmov za razrešitev navedenih relacij. Zato ima takšno programiranje številne prednosti pred proceduralnim. Programi v hitri poti imajo z uporabo funkcijskega programiranja do 20-krat manj kode /19//20/, zato se kodo enostavneje vzdržuje, odkrivanje napak je bolj enostavno, krajši je čas, potreben da zamisel implementiramo. Najbolj značilno področje, kjer se funkcijsko programiranje uspešno uporablja že desetletja, je delo s podatkovnimi bazami (jezik SQL).

5. Zaključki

Osnovna težava gigabitnih omrežij je v tem, da uporabne komunikacije ni mogoče več zagotoviti na tehnično preprost način, to je z velikim predimenzioniranjem komunikacijskih naprav v vozliščih, saj dosežemo zgornjo mejo zmogljivosti obstoječih gradnikov. Trend globalizacije in zlivanje telekomunikacijskih storitev v skupno infrastrukturo prav tako neugodno vpliva na uporabnost obstoječih rešitev. Namreč, zahteva s strani uporabnikov po večji pasovni širini je primerljiva zahtevi po diferenciranih storitvah na istem omrežju, kar problem hitrih naprav še dodatno zaplete. Velika pasovna širina več ne omogoča ekonomičnega prenosa čez paketno omrežje, temveč je zato potrebna upravljana pasovna širina. Šele z napravami, ki obravnavajo prometne tokove na inteligenten način, je mogoče realizirati zagotovljeno kakovost plačljivih storitev. Proizvajalci telekomunikacijskih naprav ugotavljajo, da je podpora širokemu spektru svojih naprav z različnimi integriranimi vezji s fiksno funkcionalnostjo neekonomična rešitev. Že sama standardizacija na področju telekomunikacij se hitro spreminja, pri izdelovanju inteligentnih naprav pa je fleksibilnost še toliko bolj pomembna za učinkovit nastop na tržišču. Zato je področje programljivih procesorskih vezij za uporabo v hitrih telekomunikacijskih sistemih trenutno ena najbolj vročih zadev v svetovni industriji procesorskih vezij.

Klasifikacija prometnih tokov v omrežju je mehanizem, ki omogoča meritev različnih prometnih parametrov, ki so v končni fazi potrebni pri trženju storitev. Dejstvo, da predstavlja opis postopka klasifikacije v 95% opis ustreznega protokola, narekuje potrebo po visoko programljivem omrežnem procesorju. Poleg visoko optimirane arhitekture za delo v paketnem omrežju je pomemben tudi način, kako je opis klasifikacije sprogramiran. Tukaj je zaznati trend funkcijskega programiranja.

6. Literatura

/1/ W. Bux, W. Denzel et al., "Technologies and Building Blocks for Fast Packet Forwarding," IEEE Commun. Mag., Jan. 2001, pp. 70-77.

/2/ T. Chu, " Network Processors or Co-processors," Integr. Commun. Design, Apr. 2001, www.icd.com.

/3/ A. Deb, " Building a Network-processor-based System," Integr. Commun. Design, Dec. 2000, www.icd.com.

/4/ L. Geppert, " The New Chips on the Block," IEEE Spectrum, Jan. 2001, pp. 66-68.

/5/ P. Gupta, N. McKeown, " Algorithms for Packet Classification," IEEE Network, March 2001, pp. 24-32.

/6/ L. Gwennap, " Net processor Makers race toward 10-Gbit/s Goal," EE Times, June 2000.

/7/ H. Higuma, M. Won, " Building configurable Network Processors," Integr. Commun. Design, Sept. 2000, www.icd.com.

/8/ –, " Network Processor Hardware," IBM Zurich Research Lab., May 2001, www.zurich.ibm.com.

/9/ V. Kumar, T. Lakshman, D. Stiliadis, " Beyond Best Effort: Router Architectures for the Diff. Serv. of Tomorrow's Internet," IEEE Commun. Mag., May 1998, pp. 152-164.

/10/ S. Keshav, R. Sharma, " Issues and Trends in Router Design," IEEE Commun. Mag., May 1998, pp. 144-151.

/11/ C. Matsumoto, " Danish Company tackles Classification," EE Times, April 2001.

/12/ D. Nix, " Using the Network Processor to mitigate Speed vs. Programability Tradeoff," Integr. Commun. Design, April 2001, www.icd.com.

/13/ E. Rothfus, " The Case for a Classification Language," White Paper, Agere Inc., Sept. 1999, www.agere.com.

/14/ A. Shubat, K. Balachandran, " CAMs improve the Internet's Performance," Integr. Commun. Design, Dec. 2000, www.icd.com.

/15/ M. A. Sanchez, E. Biersack, W. Dabbous, " Survey and Taxonomy of IP Address Lookup Algorithms," IEEE Network, March 2001, pp. 8-23.

/16/ K. Shiomoto, M. Uga et al., " Scalable Multi-QoS IP+ATM Switch Router Architecture," IEEE Commun. Magazine, Dec. 2000, pp. 86-92.

/17/ S. Sikka, G. Varghese, " Memory-Efficient State Lookups with Fast Updates," SIGCOM 2000, Stockholm, Sweden, Sept. 2000.

/18/ K. Thompson, G. Miller, R. Wilder, " Wide-Area Internet Traffic Paterns and Characteristics," IEEE Network, November 1997, pp. 10-22.

/19/ –, " The Challenge for Next Generation Network Processors," White Paper, Agere Inc., Sept. 1999, www.agere.com.

/20/ –, " Building Next Generation Network Processors," White Paper, Agere Inc., Sept. 1999, www.agere.com.

Mihael Kaiser, univ.dipl.inž.el.,
izr.prof.dr. Žarko Čučej, dipl.inž.el.,
Univerza v Mariboru, Fakulteta za elektrotrehniko, računalništvo in informatiko,
Smetanova 17, 2000 Maribor, Slovenija.

Prispelo (Arrived): 27.07.2001 *Sprejeto (Accepted): 20.08.2001*