UNIVERSITATEA "POLITEHNICA" DIN TIMIŞOARA

FACULTATEA DE ELECTRONICĂ ŞI TELECOMUNICAȚII

66 5

CONTRIBUȚII LA ANALIZA ŞI PROIECTAREA ALGORITMILOR DE PLANIFICARE A TRAFICULUI ÎN REȚELE CU COMUTARE DE PACHETE

TEZĂ DE DOCTORAT

Conducător ştiinţific:
Prof.dr.ing. Ioan Naforniță

Autor:
Ing. Hasan Harasis

1998

BUPT

CUPRINS

Capitolul 1 INTRODUCERE 1

1.1 Introducere 1
1.2 Tipuri de rețele 3
1.3 Congestia 4
1.4 Metode de control al congestiei 5
1.4.1 Controlul congestiei în buclă deschisă 6
1.4.2 Controlul congestiei în buclă închisă 7
1.4.3 Scheme hibride 8
1.5 Clepsidra sau perioadele de control a congestiei 8
1.6 Managementul congestiei în rețele de mare viteză 11
1.6.1 Legăturile şi traficul în rețelele de mare viteză 12
1.6.2 Controlul fluxului prin ferestre şi prin rată 13
1.6.3 Controlul cu reacție şi controlul în buclă deschisă 15
1.6.4 Controlul la sursă şi controlul în rutere 15
1.6.5 Contrapresiunea 17
1.6.6 Modul de lucru cu rezervare şi modul de lucru fără rezervare 18
1.6.7 Alegerea schemei de control a congestiei 19

Capitolul 2 CONTROLUL TRAFICULUI 21

2.1 Formarea traficului 21
2.1.1 Algoritmul găleții găurite 22
2.1.2 Algoritmul găleții cu jeton 23
2.2 Algoritm de control a admisiei 25
2.2.1 Algoritmul "suma simplă" 26
2.2.2 Algoritmul "suma măsurată" 26
2.2.3 Algoritmul regiunii de acceptanță 27
2.2.4 Algoritmul capacității echivalente 28
2.2.4.1 Estimarea capacității echivalente folosind distribuția normală 29
2.2.4.2 Estimarea capacității echivalente cu limita Hoeffding 30
2.2.4.3 Măsurarea ratei medii de sosire 31
2.2.5 Controlul probabilistic al admisiei 32
2.2.6 Controlul admisiei folosind rata de pierderi a celulelor 33
2.2.6.1 Algoritmul CAC-FA-CLR 34
2.2.6.2 Descrierea sistemului bazat pe reguli fuzzy 34

BIBLIOTECA CENTRALĂ
UNIVERSITATEA "POLITEHNICA"
TIMIŞOARA

Capitolul 3 CONTROLUL CONGESTIEI ŞI MANAGEMENTUL TRAFICULUI ÎN REȚELE ATM 39

3.1 Introducere 39
3.2 Parametrii traficului şi calitatea serviciului 39
3.2.1 Parametrii de trafic 39
3.2.2 Categorii de servicii 41
3.3 Metode de control a congestiei 42
3.4 Facilități pentru reacție în rețele ATM 43
3.5 Criterii de selecție 43

BUPT

3.5.1 Scalabilitatea 43
3.5.2 Optimalitatea 44
3.5.3 Indicele de imparţialitate 44
3.5.4 Robusteţea 45
3.5.5 Implementabilitatea 45
3.5.6 Configuraţii de simulare 45
3.6 Structura traficului 46
3.7 Scheme de tratare a congestiei 46
3.7.1 Rezervarea rapidă 46
3.7.2 Controlul ratei bazat pe întârziere 47
3.7.3 Notificarea explicită, înspre înapoi, a congestiei BECN 47
3.7.4 Distrugerea timpurie a pachetelor, EPD 48
3.7.5 Fereastra pe legătură şi controlul binar cap la cap al ratei 48
3.7.6 Alocare imparţială cu reacţie despre memorie şi rată 48
3.7.7 Abordarea bazată pe credite 49
3.7.8 Abordarea bazată pe rată 50
3.7.8.1 Schema MIT 51
3.7.8.2 Algoritm îmbunătăţit de control proporţional a ratei 52
3.7.8.3 Editarea congestiei prin algoritmul OSU 53
3.7.8.4 Evitarea congestiei folosind controlul proporţional 54
3.8 Surse şi destinaţii virtuale 55
3.9 Comparaţia între abordarea bazată pe credite şi cea bazată pe rată 56

Capitolul 4 PLANIFICAREA TRAFICULUI 58

4.1 Planificarea traficului în IBS 60
4.2 Planificarea traficului în OBS 62
4.2.1 Planificatoare reprezentative 64
4.2.1.1 Primul sosit primul servit, FCFS 64
4.2.1.2 Servirea imparţială, FQ 65
4.2.1.3 Servirea imparţială stohastică, SFQ 65
4.2.1.4 Ceasul virtual, VCL 66
4.2.1.5 Procesorul generalizat, GPS 66
4.2.1.6 Servirea imparţială cu autosincronizarea, SCFQ 68
4.2.1.7 Planificarea cu oprire şi pornire, SG 68
4.2.1.8 Planificarea carusel, RR 68
4.2.1.9 Planificarea carusel cu recuperarea deficitului, DRR 69
4.2.1.10 Planificarea carusel ierarhizată, HRR 70
4.2.1.11 Planificarea cu întârzierea datei planificată anterior, DEDD 70
4.2.1.12 Planificarea DEDD cu controlul jitterului, JEDD 71
4.3 Parametrii planificatoarelor 71
4.3.1 Garanţii asupra întârzierii cap la cap 73
4.3.2 Imparţialitate 74
4.3.3 Complexitatea implementării 75

Capitolul 5 ALGORITMI DE IMPLEMENTARE A REZERVĂRILOR DE BANDĂ ÎN COMUTATOARE CU MEMORARE LA INTRARE 77

5.1 Algoritm paralel de asociere probabilistică iterativă, PIM 77
5.2 Numărul de iteraţii al algoritmului PIM 80
5.3 Asocierea probabilistică cu filtrarea cererilor, APFC 83

BUPT

Capitolul 6 CLASIFICAREA GENERALĂ A PLANIFICATOARELOR UNIFICATE 89

6.1 Servere cu latența ratei, *LR* 89
6.1.1 Analiza unui singur server *LR* 92
6.1.2 Analiza unei rețele de servere *LR* 97
6.1.3 Limita întârzierii pentru surse formate cu o găleată dublă 105
6.2 Planificatoare din clasa *LR* 107
6.2.1 Servirea imparțială ponderată WFQ 110
6.2.2 Ceasul virtual, VCL 111
6.2.3 Servirea imparțială cu autosincronizare, SCFQ 111
6.2.4 Planificarea carusel cu recuperarea deficitului DRR 113
6.2.5 Planificarea carusel ponderată WRR 115
6.3 O margine îmbunătățită a întârzierii 117
6.4 Imparțialitatea serverelor *LR* 119
6.4.1 Imparțialitatea planificatorului PGPS 121
6.4.2 Imparțialitatea planificatorului SCFQ 121
6.4.3 Imparțialitatea planificatorului RR 122
6.5 Concluzii 124

Capitolul 7 SERVERE PROPORȚIONALE CU RATA 126

7.1 Funcțiile de potențial 126
7.2 Servere proporționale cu rata, SPR 127
7.2.1 Servere proporționale cu rata, pachet cu pachet 131
7.2.2 Analiza întârzierii 134
7.3 Imparțialitatea serverelor proporționale cu rata 134
7.4 Algoritm de planificare imparțială pentru rețele cu pachete 137
7.4.1 Metoda de menținere a potențialului sistemului 137
7.4.2 Planificarea imparțială bazată pe cadre 138

CONCLUZII
ANEXĂ
BIBLIOGRAFIE

BUPT

Capitolul 1

INTRODUCERE

1.1 Introducere

În prezent pe tot globul pământesc sunt răspândite rețelele de comunicații de date. Întrucât ele reprezintă o resursă costisitoare atât în ceea ce privește crearea lor cât și exploatarea lor, este deosebit de importantă optimizarea performanțelor lor, pentru obținerea de beneficii maxime la un preț minim.

Atunci când foarte multe pachete sunt prezente într-o subrețea, sau o parte a unei subrețele, performanțele se degradează, deoarece apare congestia. Congestia poate fi definită ca o diminuare a performanțelor rețelei la încărcare mare [GAP, J2, JK, KES]. Simptomele sale sunt prezentate în fig. 1.1. Când numărul de pachete emise în subrețea de calculatoarele gazdă nu depășește capacitatea de transport, ele sunt livrate în totalitate la destinație, mai puțin cele afectate de erori de transmisie. Numărul pachetelor livrate este proporțional cu numărul de pachete emise. Atunci când traficul crește prea mult, nodurile/ruterele încep să nu mai facă față și să piardă pachete. Situația se degradează progresiv, astfel că la un trafic foarte intens nici un pachet nu mai este livrat și se ajunge la blocarea rețelei, forma extremă a congestionării [DBPS, NMH, NM].

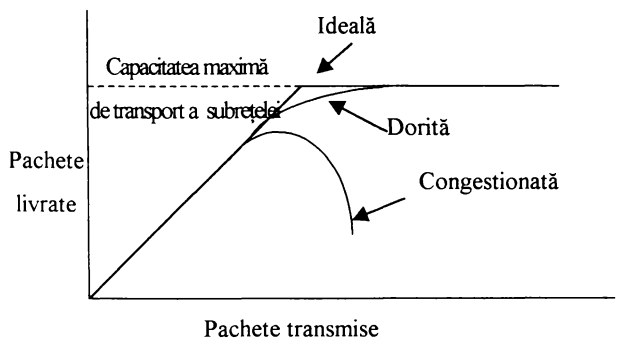


Fig. 1.2 Congestia apare când traficul este prea intens; performanțele se degradează

Majoritatea rețelelor se comportă bine la încărcare mică, dar pe măsură ce încărcarea cu trafic crește, apare congestia. Deoarece congestia poate cauza pierderi de date și întârzieri mari în transmiterea datelor, devine extrem de important controlul pentru evitarea congestiei.

Congestia poate fi produsă de mai mulți factori: memorie insuficientă în rutere, procesoare lente în rutere, linii neperformante, sau o incompatibilitate între părți ale sistemului. Datorită mecanismelor de retransmisie, prevăzute pentru a

BUPT

asigura transferul pachetelor prin subreţea până la destinaţie, congestia se autoalimentează şi se răspândeşte[Tane, AK, CG, Sch, J1, MG]. Se iau în consecinţă măsuri pentru evitarea congestiei care implică, printre altele o dirijare eficientă şi controlul fluxului în reţea. Aceste măsuri constituie subiectul lucrării de faţă. Domeniul este extrem de vast şi în consecinţă nici o abordare nu poate fi exhaustivă [J2, KES, JK, DS, GAP].

Primele reţele mari de comunicaţii au fost reţelele telefonice. Cum acestea transportă trafic de un singur tip şi cu un comportament bine cunoscut, e posibilă evitarea congestiei prin rezervarea de resurse suficiente în momentul stabilirii fiecărui apel [NM, THN, SYS, STA]. Limitând numărul total de utilizatori, se pot asigura suficiente resurse fiecărui apel acceptat, pentru asigurarea performanţei dorite şi astfel se poate evita congestia.

Ca la orice sistem cu rezervare însă, poate apărea o slabă utilizare a resurselor, dacă acestea sunt alocate unui apel care nu le utilizează şi astfel nu sunt disponibile pentru alte apeluri. Dirijarea folosită în reţele telefonice este o dirijare de tip ierarhic [Tane, TH, THN, SMV].

Primele reţele de comunicaţii de date, au fost reţelele de tip "memorează şi retransmite", SF (Store and Forward), fără rezervare de resurse, folosirea resurselor fiind făcută prin multiplexare statistică. Deoarece apariţia congestiei în aceste reţele era firească, nu erau constante nici numărul utilizatorilor nici sarcina oferită de aceştia. Avantajul obţinut prin multiplexarea statistică a resurselor reţelei a fost compensat de posibilitatea apariţiei congestiei. Au fost propuse în consecinţă diverse scheme pentru evitarea acesteia [STEV, BM, FF]. De asemenea au fost folosiţi o serie de algoritmi de dirijare, algoritmi bazaţi pe vectorul distanţă [CCS, ERN, EST, ZDESZ, WVTP, HMN1, HMN2, VB, STEE, BFG, BP] şi apoi algoritmi bazaţi pe starea legăturii.

În ultimii ani a apărut Internet-ul, ca o reuniune de subreţele locale, regionale şi continentale. Apariţia sa a fost posibilă datorită fuziunii dintre domeniul telecomunicaţiilor şi cel al calculatoarelor. Dimensiunile acestei interreţele depăşesc orice prognoză. Numărul mare de utilizatori, cu diferite tipuri şi volume de trafic, precum şi controlul complet distribuit al reţelei a resuscitat interesul cu privire la controlul congestiei. Dezvoltarea Internet-ului a dus la necesitatea adaptării şi a algoritmilor de dirijare atât la dimensiunile acestuia cât şi la politicile domeniilor tranzitate [GLA, DMB, KKMMR, KLO, LK, MS, ORS, VHNM, GLAM, GKT].

În plus, introducerea fibrelor optice pe liniile de trunchi a dus la creşterea lăţimii de bandă cu 4 ordine de mărime(600 Mbps/56Kbps). În reţelele WAN de mare viteză produsul întârziere-bandă al unui singur circuit e de ordinul a 30 Mb(600 Mbps x 50 ms timp de propagare peste Europa sau SUA). La o valoare atât de mare, o singură sursă poate genera o sarcină care să depăşească capacitatea tampoanelor de memorie din nodurile de comutare; acest lucru duce la pierderi de pachete şi întârzieri cap la cap excesive, pentru toate calculatoarele(hosturile) ataşate reţelei. Astfel şi în reţelele de mare viteză controlul congestiei e deosebit de important.

BUPT

1.2 Tipuri de rețele

În rețelele de transmisiuni de date, datele sunt transmise de la o sursă la o destinație, trecând prin nodurile de comutare, noduri de tip memorează şi retransmite. Sursele de date pot fi utilizatori umani, transferând pe durata sesiunii la distanță, fie fişiere, fie caractere. Din punct de vedere al rețelei, însă, toate nivelurile superioare OSI, 5, 6 sau 7 constituie surse de date. La destinație, un proces pereche cu cel al sursei, recepționează datele şi de obicei confirmă fiecare pachet primit. Nodurile de comutare, sau comutatoarele dirijează şi planifică pachetele pe liniile de ieşire, introducând datele în tampoanele de memorie de ieşire, dacă rata sosirilor depăşeşte rata serviciului. Un flux simplu de pachete dintre o sursă şi o destinație e denumit conversație.

Primul tip de rețea considerat, denumit rețea fără rezervare, e un model abstract pentru rețele ca Internet-ul. În astfel de rețele, nodurile intermediare pot rezerva tampoane de memorie (ceea ce nu afectează multiplexarea statistică a lățimii de bandă), dar nu pot face rezervări de lățime de bandă(ceea ce ar afecta multiplexarea statistică a lățimii de bandă). În acest tip de rețele se presupune că hosturile sunt ataşate direct la comutatoare noduri/rutere, care la rândul lor sunt conectate la alte comutatoare sau la hosturi. Un comutator poate fi un modul software rezident în host, sau un modul hardware separat.

Celălalt tip de rețele este cel al rețelelor orientate pe rezervare. În acestea, comutatoarele rezervă [DBPS, MG, PAX, FJ3, GF, GLAM, KVCP] atât lățimea de bandă cât şi tampoanele de memorie pentru circuitele virtuale(CV), ca de exemplu în Datakit. Presupunem că aceste rețele transportă două tipuri de trafic:

- trafic orientat pe performanță, respectiv trafic sincron sau de timp real(voce şi imagini în mişcare) care de obicei are cerințe asupra întârzierii de timp-real, lățimea de bandă;
- trafic de date, denumit şi [TMNL, VB] trafic de efort maxim(best-effort traffic) care nu are asemenea cerințe. Deoarece pentru traficul de efort maxim nu e necesară rezervarea lățimii de bandă, la rețelele orientate pe rezervare componenta de efort maxim poate fi modelată ca la rețelele fără rezervare. Astfel, schemele proiectate pentru rețelele fără rezervare, pot fi transferate, cu modificările necesare, rețelelor orientate pe rezervare sau orientate pe conexiuni/rezervare.

E de aşteptat că majoritatea viitoarelor rețele să fie orientate pe rezervare. Există însă suficiente argumente şi pentru studiul congestiei în rețelele fără rezervare. Primul motiv este că rețelele fără rezervare utilizează mai eficient banda disponibilă, decât cele orientate pe rezervare, datorită multiplexării statistice [GV, LEM]. Astfel, furnizorii de rețele care doresc să optimizeze costul vor continua să producă rețele fără rezervare. Apoi, tehnicile dezvoltate pentru controlul congestiei pot fi aplicate pentru controlul traficului de efort maxim în rețelele orientate pe rezervare. Şi, în final, rețelele fără rezervare sunt în prezent cele mai răspândite şi ele vor continua să existe în viitor, atât din cauza inerției cât şi a dorinței de a menține o tehnologie cunoscută şi deja verificată.

BUPT

1.3 Congestia

Actualele definiţii ale congestiei sunt legate de diferitele aspecte ale comportării reţelei în sarcină mare. Deoarece congestia apare la încărcări mari ale reţelei,

Considerăm o reţea fără rezervare, unde din anumite motive, pe termen scurt rata sosirii pachetelor la un ruter oarecare, depăşeşte rata serviciului. Rata serviciului e determinată atât de timpul de procesare per pachet cât şi de lăţimea de bandă a liniei de ieşire. Astfel strangularea (botlleneck) poate fi cauzată fie de procesorul comutatorului fie de linia de ieşire. În acest caz, pachetele sunt memorate şi vor suferi întârzieri. Întârzierile suplimentare pot să ducă la expirarea timpului (time-out) sursei, care va iniţia retransmisii, ceea ce evident va conduce la creşterea încărcării pe porţiunea strangulată [FF, FJ2, GBCHP, COV, KALV]. Această reacţie va conduce rapid la deteriorarea situaţiei, deoarece retransmisiile vor domina, diminuându-se astfel traficul util şi eficientă (througput) [HLG, KO, KKK, KR, LID]. În plus, dacă comutatoarele au prevăzut controlul fluxului (ca în ARPANET, TYMNET, DATAPAC, etc.), ele nu vor permite accesul noilor pachete, astfel că acestea vor fi memorate/întârziate în comutatorul anterior. Acest lucru duce la răspândirea congestionării şi în final la oprirea traficului, deci la blocare [LMJ, NI, MCS].

La apariţia congestiei creşte întârzierea pachetelor de date în cozile de aşteptare, pot apărea pierderi de pachete şi traficul e dominat de retransmisii, astfel că scade rata datelor efective. Definiţia standard a congestionării este: o reţea este congestionată dacă, datorită suprasarcinii, apare condiţia X, unde X este fie întârzierea excesivă în cozile de aşteptare, fie pierderea de pachete, fie scăderea traficului util [BG, STA, TAN, KL, KKK, KES].

Această definiţie nu e satisfăcătoare. Întârzierile şi pierderile sunt indici de performanţă ale căror modificări pot semnala şi alte fenomene. Cu excepţia cazului banal (al reţelelor deterministe, la care e bine definită curba sarcină-întârziere şi deci şi momentul apariţiei congestiei e bine definit), definiţia nu specifică momentul exact în care o reţea poate fi considerată congestionată. O reţea cu o întârziere medie în cozi, de (1-10) ori timpul de serviciu în fiecare comutator, sigur nu e congestionată, dar nu e clar dacă o reţea cu întârzierea medie în cozi de 1000 ori timpul de serviciu e sau nu congestionată. În plus o reţea care poate părea congestionată unui utilizator, ar putea părea necongestionată pentru alt utilizator. Dacă în reţea, rata de pierderi a celulelor e de 1-500, şi primul utilizator acceptă o rată de pierderi de 1-1000, iar al doilea doar de 1-100, doar al doilea va reclama că reţeaua e congestionată. Dar reţeaua poate fi considerată necongestionată doar dacă toţi utilizatorii sunt de acord cu asta.

Congestia reţelei depinde evident de perspectiva utilizatorului: utilizatorii nepretenţioşi pot accepta scăderi ale performanţei reţelei care pot părea inacceptabile utilizatorilor pretenţioşi. Deci problema este ce utilitate oferă

BUPT

rețeaua utilizatorilor şi cum descreşte această utilitate cu creşterea încărcării rețelei.

În concluzie o rețea e considerată congestionată din punct de vedere a unui utilizator, dacă utilitatea sa descreşte, datorită creşterii încărcării rețelei. Rețeaua poate fi congestionată din punctul de vedere al unui utilizator, şi necongestionată din punctul de vedere al altui utilizator. Ea este strict necongestionată dacă nici un utilizator nu constată vreo congestie. Utilitatea pentru un utilizator poate descreşte şi din alte cauze decât supraîncărcarea, dar utilizatorul nu poate face diferența. Este însă necesar să se determine cauza şi luate măsuri pentru evitarea şi/sau reducerea ei.

Conform acestei definiții, controlului congestiei îi revin sarcini suplimentare. O rețea care controlează congestia e responsabilă de utilitatea rețelei pentru fiecare utilizator şi trebuie să fie capabilă să-şi gestioneze resursele astfel încât să nu scadă utilitatea o dată cu creşterea congestiei. Rețeaua trebuie astfel să fie capabilă să facă diferența între conversații, şi să aloce priorități diferite conversațiilor, în funcție de calitatea serviciului QOS (quality of service) pretinsă de utilizator.

1.4 Metode de control a congestiei

Controlul congestiei este definit ca un set de mecanisme care previne sau reduce scăderea performanțelor rețelei şi anume a utilității pentru utilizator, datorată creşterii încărcării rețelei. Acest control ar trebui realizat pentru fiecare utilizator al rețelei. Dacă rețeaua nu poate preveni pierderea utilității pentru utilizator, atunci ar trebui să încerce măcar să reducă la minim posibil pierderea, şi să facă acest lucru în mod corect, sau imparțial adică la fel pentru toți utilizatorii. În rețelele fără rezervare, unde congestionarea e inevitabilă, trebuie detectat care utilitate se pierde şi luate măsuri atât pentru evitarea răspândirii congestionării, cât şi pentru ca pierderea utilității să fie egal distribuită utilizatorilor afectați. Cum în rețelele moderne pentru transmisii de date de tip ATM, s-a adoptat tehnica multiplexării statistice, e necesar să fie prevăzut un control al traficului, atât pentru evitarea congestionării în fiecare nod cât şi pentru obținerea QOS cerut de fiecare conexiune.

Multe dintre problemele care apar în rețelele de comunicații de date, care sunt sisteme complexe, pot fi privite din punct de vedere al teoriei controlului. Această abordare conduce la împărțirea tuturor soluțiilor, inclusiv a celor pentru controlul congestiei, în două categorii:

- controlul în buclă deschisă [Tane, AD], sau preventiv sau proactiv [Kes], care pot acționa fie asupra sursei fie asupra destinației, respectiv,
- controlul în buclă închisă sau reactiv cu reacție implicită sau explicită.

BUPT

1.4.1 Controlul congestiei în buclă deschisă

Soluțiile în buclă deschisă încearcă să rezolve problema printr-o proiectare atentă, în esență să se asigure că problema nu apare. După ce sistemul e pornit şi funcționează, nu se mai fac nici un fel de corecții. Instrumentele pentru realizarea controlului în buclă deschisă realizează planificarea deciziilor în diferite puncte din rețea: când să se accepte trafic nou, când să se distrugă pachete şi care să fie acestea, etc. Deciziile se iau fără să se țină cont de starea curentă a rețelei.

În cazul controlului în buclă deschisă se rezervă resursele necesare asigurării solicitărilor utilizatorului, exprimate în momentul conectării. Aceste resurse vor fi mereu disponibile utilizatorului. Controlul preventiv al congestiei este deci un control orientat pe rezervare. Resursele rezervate sunt disponibile pe toată durata conversației acceptate, fiind astfel garantată utilitatea solicitată. Dar acest avantaj este plătit cu prețul limitării numărului de utilizatori care pot lucra cu rețeaua la un moment dat ca şi în cazul multiplexării cu divizarea timpului TDM (Time Division Multiplexing) ceea ce poate duce la o utilizare slabă a rețelei [BG, STA, TANE].

Pentru prevenirea congestiei se folosesc o serie de politici, la diferite niveluri [TAN, BG, STA, LP, THP, TA]. Astfel la nivelul legăturii de date, congestia e influențată de politicile de retransmisie, de memorare a pachetelor în afara secvenței, de confirmare şi de control a fluxului. Politica de retransmisie stabileşte timpul de expirare a copiilor de siguranță. Dacă acest timp e prea mic, time-out-ul se produce rapid şi dacă vor fi retransmise ultimele *n* pachete în aşteptare, acest lucru va produce o mai mare încărcare cu trafic, decât în cazul alegerii unui timp mai mare şi a retransmiterii selective. Legată de aceasta este şi politica de memorare: dacă receptorul distruge pachetele în afara secvenței, acestea vor trebui retransmise ulterior, ducând la creşterea traficului. Politica de confirmare influențează şi ea congestia. Dacă fiecare pachet e confirmat separat creşte traficul, iar dacă confirmările sunt preluate de traficul de răspuns se pot produce expirări de timp şi retransmisii frecvente [AST, BFMMVZ, BG, DS]. La nivelul rețea, congestia e influențată de utilizarea circuitelor virtuale respectiv a datagramelor, deoarece cele două abordări implică algoritmi diferiți. Plasarea în cozi de aşteptare a pachetelor şi politicile de servire specifică modul de realizare al cozilor de aşteptare pentru liniile de intrare şi cele de ieşire, precum şi disciplinele de servire. Asociată cu acestea este politica de renunțare la anumite pachete atunci când nu mai există memorie disponibilă.

Tot în cadrul nivelului rețea, algoritmul de dirijare poate ajuta la atenuarea şi evitarea congestiei. Dacă o linie este deja congestionată, algoritmul de dirijare poate determina dirijarea unei părți din trafic pe linii mai puțin încărcate, chiar dacă ruta este suboptimală, [TNH, TANE]. În plus, e important şi timpul de viață alocat pachetelor. Dacă acesta e prea mic, pachetele vor fi distruse înainte de a ajunge la destinație, ceea ce va duce la retransmisii. Dacă timpul e prea mare, pachetele pierdute vor încurca prea mult activitatea.

BUPT

La nivelul transport se pun aceleaşi probleme ca la nivelul legătură de date, dar se impune observaţia că e mai dificil de determinat intervalul de expirare a timpului, deoarece timpul de tranzit prin subreţea e mai greu de estimat decât cel între două rutere. Un timp prea mic va conduce cum s-a văzut, la retransmisii suplimentare, deci la creşterea congestiei, iar un timp prea mare va duce la scăderea congestiei dar şi la creşterea timpului de răspuns în cazul pachetelor pierdute.

În reţelele ATM (Asynchronous Transfer Mode) o metodă de reducere a congestiei, folosită în buclă deschisă este formarea traficului (traffic shaping). Necesitatea ei a apărut datorită traficului în rafală (bursty traffic), [KA, KL], şi prevede impunerea unei rate previzibile cu care să fie transmise pachetele. Formarea traficului se ocupă de uniformizarea ratei de transmisie, spre deosebire de protocoalele cu fereastră glisantă care limitează volumul de date în tranzit la un moment dat. Formarea traficului este eficientă atunci când emiţătorul, receptorul şi subreţeaua sunt de acord, şi este deosebit de importantă pentru datele de timp real. Stabilirea formei traficului şi apoi urmărirea respectării ei (traffic policing) e mai uşor de făcut în subreţelele cu circuite virtuale decât în cele bazate pe datagrame.

1.4.2. Controlul congestiei în buclă închisă

Datorită constrângerilor de timp real, în reţelele de mare viteză, e preferat controlul preventiv [DM, GK, J1]. Soluţiile în buclă închisă, prin contrast cu cele în buclă deschisă, se bazează pe conceptul de reacţie inversă (feedback-loop).

Controlul congestiei se realizează în trei etape:

- supravegherea reţelei pentru detectarea locului şi momentului de apariţie a congestiei,
- trimiterea informaţiilor despre starea congestiei în locurile unde se pot lua decizii şi executa acţiuni,
- executarea de acţiuni pentru reducerea congestiei.

La supravegherea reţelei se pot urmări diferiţi parametri: procentul de pachete distruse din lipsă temporară de memorie, lungimea medie a cozilor de aşteptare, numărul de pachete retransmise din cauza expirării timpului de confirmare, întârzierea medie pe pachet, deviaţia standard a întârzierii medii, etc. Valorile mari a acestora indică creşterea congestiei.

Informaţiile despre starea congestiei pot fi transmise surselor/ calculatoarelor gazdă de către ruterele care o detectează, sau pot fi cerute de surse. În ambele cazuri apare o încărcare suplimentară a reţelei, deja congestionată. De aceea s-a propus şi marcarea unui bit/câmp din fiecare pachet de date, existente oricum în reţea, atunci când congestia depăşeşte un prag.

Pentru reducerea congestiei, prima măsură care se încearcă este creşterea capacităţii de transport a reţelei: lăţimea de bandă poate fi crescută temporar sau permanent prin folosirea unor linii suplimentare, despicarea traficului sau

BUPT

creşterea puterii de transmisie(în cazul reţelelor pe canal radio). În cazul unor congestii grave se pot folosi şi ruterele de rezervă, folosite doar pentru copii de siguranţă. Când nu mai e posibilă creşterea capacităţii, singura măsură care mai poate fi luată este reducerea încărcării, prin diferite măsuri: refuzarea sau degradarea serviciilor pentru unii sau toţi utilizatorii şi planificarea cererilor utilizatorilor.

Controlul reactiv se poate aplica în cazul reţelelor fără rezervare. Utilizatorilor li se permite să transmită date, fără rezervarea resurselor, dar există posibilitatea ca în cazul încărcării mari a reţelei, ei să primească un grad scăzut de utilitate din partea reţelei. Reţeaua trebuie să ofere utilizatorilor căi de detectare a schimbărilor de stare a reţelei şi să asigure un mecanism de adaptare a fluxului utilizatorilor la aceste schimbări. Cum de obicei reţeaua nu garantează utilizatorilor săi furnizarea unui anumit nivel de utilitate, resursele sale pot fi multiplexate statistic STDM (Statistical TDM), [BMM] cu avantajul posibilităţii creşterii numărului de utilizatori activi simultan. Rămâne însă, evident, posibilitatea ca traficul în rafale să supraîncarce reţeaua, ceea ce va duce la apariţia congestiei.

1.4.3 Scheme hibride

În cazul schemelor hibride se combină cele două abordări. Astfel, reţeaua poate să ofere statistic garanţii: de exemplu unui utilizator i se poate garanta o întârziere cap la cap mai mică de 10 secunde, cu o probabilitate de 0.9 [KES, JDSZ] Garantarea statistică permite administratorului de reţea să utilizeze la maxim resursele, dar fără să ofere performanţa maximă.

Altă schemă hibridă este cea a reţelelor care acceptă utilizatori cu serviciu garantat GS, (Guaranteed Service), şi utilizatori de efort maxim BE-(Best Effort). Pentru primii se garantează calitatea serviciului şi are loc rezervarea de resurse. Celorlalţi nu li se garantează calitatea şi pot folosi doar acele resurse neutilizate la primii utilizatori.

În sfârşit, serverul poate aloca, fiecărui utilizator, un minim de resurse, garantând implicit o utilitate minimă. Pe măsura încărcării reţelei utilizatorii concurează pentru accesarea la resursele nerezervate aflate într-un stoc comun [DJM, DK, LAN, BT, DOK, FJ3] Poate fi folosită multiplexarea statistică a acestora, fără pierderea completă a utilităţii.

1.5. Clepsidra sau perioadele de control a congestiei

Controlul congestiei trebuie realizat în perioadele de suprasarcină a reţelei. De exemplu, în reţelele telefonice, ca şi în cele de date de altfel, suprasarcina poate apare în anumite ore de vârf ale zilei, sau în anumite săptămâni ale anului. În reţelele de date poate apare, datorită traficului în rafale, suprasarcina pe durata

BUPT

câtorva milisecunde, deşi pe durata unei ore, traficul e cu mult mai mic decât cel nominal. Trebuie deci efectuat în primul caz un control la o scară de timp a orelor, ceea ce se realizează practic, prin controlul admisiei. În al doilea caz, dacă conversaţia e sensibilă la întârziere, controlul, realizat acum la o altă scară de timp, a milisecundelor, se face printr-o planificare, stabilind cine lucrează în intervalul respectiv, pentru a satisface cerinţa cu privire la întârziere.

Scara de timp la care trebuie controlată congestia este perioada de timp în care utilizatorul sesizează schimbări ale stării reţelei.

Controlul trebuie realizat simultan, la diferite scale de timp, iar procesele trebuie să coopereze între ele: planificarea fără controlul admisiei nu poate asigura garantarea întârzierii.

- Luna: în decursul unei luni sau a mai multora, poate apărea suprasarcina în reţea, ca de exemplu datorită creşterii numărului de abonaţi, etc. Efectul este supraîncărcarea liniilor de trunchi. Reacţia tipică la suprasarcina de durată este suplimentarea lăţimii de bandă disponibilă. Astfel vechile linii de 9600 bps vor fi înlocuite progresiv de linii de 64 kbps, 10 Mbps, 100 Mbps (FDDI) ş.a.m.d. La această scară, controlul congestiei constă în planificarea capacităţii [DJM, FF, GF, JDSZ, LSY]. Se construieşte o matrice de trafic, reprezentând volumul de trafic între toate punctele. Apoi se optimizează utilitatea oferită de reţea, utilizând tehnici de programare liniară, pentru a obţine o alocare optimă a capacităţii de-a lungul fiecărei căi. Este însă dificil de determinat matricea de trafic, iar dimensiunile sale pot fi foarte mari. În plus, nu există nici o garanţie că încărcarea cu trafic într-o perioadă ulterioară va fi la fel. Totuşi această metodă este mai bună decât o abordare la întâmplare, fiind şi în prezent folosită pentru reţele telefonice;

- Ziua: măsurările de trafic indică un comportament ciclic al reţelei, cu perioada de o zi [FF, SCH, SYS]. Reţeaua e foarte încărcată între orele 8 –17 respectiv e slab încărcată, sau deloc, în timpul nopţii. Schemele de control a congestiei trebuie să încerce să uniformizeze încărcarea reţelei pe parcursul întregii zile. Acest lucru se realizează prin metodele de taxare diferite: suprataxare a lucrului în orele de vârf şi tarife mai scăzute noaptea. Această soluţie duce la nivelarea traficului, dar rămân o serie de probleme de tratat, ca de exemplu: comportarea iraţională a unor utilizatori, traficul în rafale a altora(cu valori de vârf şi medii extrem de diferite), timpul relativ lung consumat pentru licitarea unei resurse şi de aici problema atingerii echilibrului sistemului la conectarea unor noi utilizatori, etc. Problema taxării fiind însă de mare interes pentru autoritatea administrativă, se fac în continuare studii în această direcţie [GF, GaP].

- Sesiune: în reţelele orientate pe conexiune sesiunea e intervalul de timp între stabilirea şi deconectarea apelului. La scara sesiunii controlul congestiei se face prin controlul admisiei. Dacă admiterea unei conversaţii ar degrada calitatea serviciului, atunci aceasta nu e admisă, ceea ce determină implicit utilizatorii să-şi declare corect solicitările, fără supraestimări. Bazele teoretice pentru controlul admisiei sunt date de teoria jocurilor şi anume de proiectarea mecanismelor

BUPT

[KKK, KES, SV1]. Se presupun însă cunoscute informaţiile despre structura sistemului, ca de exemplu funcţia utilităţii fiecărui utilizator. Dinamica sistemului este însă ignorată. Pentru controlul congestiei poate fi folosit în acest caz şi alegerea rutei în momentul stabilirii sesiunii. Dacă algoritmul de dirijare culege informaţii despre starea legăturilor de pe rută, el poate dirija noile circuite de-a lungul căilor mai puţin încărcate [DB, NM, BG, Tane]. Integrarea dirijării adaptive cu controlul fluxului este o problemă complexă, iar dinamica interacţiunii dintre ele nu este prea bine cunoscută. [MS] Au fost propuse modele simplificate pentru această problemă. [BBFG, BFM, MVZ, BP, CO, DMB, EST, KLO, FJ1].

- Timpul de transfer dus-întors (Round-Trip-Time): RTT este intervalul de timp scurs între momentul transmisiei unui pachet şi momentul recepţionării confirmării şi reprezintă constanta de timp utilizată în controlul cu reacţie al fluxului. RTT este timpul minim necesar sursei pentru a determina efectul ratei de transmisie în reţea. Schemele de control a congestiei care verifică starea reţelei, apelează la această scară de timp. Au fost propuse o serie de metode de control a fluxului, cu diferite ferestre de transmisie.

Bazele teoretice pentru această abordare sunt date de teoria cozilor de aşteptare şi teoria controlului. Teoria cozilor de aşteptare a fost larg studiată [DO, AK, KL, SCH] în ipotezele: sosiri Poisson de la toate sursele, distribuţia exponenţială a timpului de serviciu la toate serverele şi independenţa tipurilor de trafic. Aceste ipoteze sunt prea puţin îndeplinite în reţelele reale [PF, LiH, LTWW, LT, WTSW], astfel că rezultatele obţinute prin abordarea stohastică a cozilor de aşteptare nu sunt pe deplin satisfăcătoare. Abordarea are însă unele avantaje, astfel că este utilizată.

Teoria controlului a fost de asemenea pe larg studiată în literatură [PAB, RW, SSD, J1, J2], dar fie au fost studii informative, fără o demonstraţie formală, fie s-au făcut presupuneri restrictive asupra comportării traficului, ratelor de serviciu, etc. [KES].

Presupunând că fiecare pachet e confirmat, atunci în fiecare RTT pot fi recepţionate mai multe confirmări. Dacă informaţia despre starea reţelei e extrasă din fiecare confirmare, atunci schema de control a congestiei poate reacţiona la schimbări mai rapid decât o dată la un interval RTT.

Mai puţin de RTT dacă se ia în considerare un interval de timp mai mic decât RTT, la această scară controlul congestiei este identic cu planificarea datelor de la cozile de ieşire ale comutatoarelor. Scopul politicii de planificare este de a decide care unitate de date va fi transmisă la un moment dat pe legătură. Această alegere determină, pentru fiecare conversaţie, lăţimea de bandă, întârzierea şi jitterul [MSB] astfel că disciplina de planificare e o componentă critică în controlul congestiei. Exemple de discipline de planificare care nu-şi modifică alocările în funcţie de starea reţelei sunt: schema de ceas virtual (Virtual Clock), schemele Delay-EDD, Stop and Go, etc. (vezi capitolul 4).

Lăţimea de bandă a liniei este o resursă, deci serverul care implementează disciplina de planificare este un manager de resurse care alocă lăţimea de bandă,

BUPT

întârzieri şi întârzieri de jitter, fiecărei conversaţii. Astfel, pentru conversaţie, e foarte important să se transmită datele aşa încât să obţină maxim de utilitate de la server. Dar, având în vedere că toate conversaţiile au acest scop şi că un câştig pentru una poate reprezenta o pierdere pentru alta, ele trebuie tratate în forma jocurilor necooperante. Astfel fiecare conversaţie trebuie să-şi aleagă o strategie, rata de transmisie în cazul nostru, încât să maximizeze utilitatea. Dar, pentru teoria jocurilor, [DM, Ste] formularea teoretică a problemei pretinde atât ca utilitatea să fie definită pentru fiecare utilizator, cât şi alte informaţii şi presupuneri, la fel ca în abordarea economică, ceea ce o face prea dificil de folosit.

1.6 Managementul congestiei în reţele de mare viteză

Interesul pentru controlul congestiei a crescut odată cu apariţia liniilor si reţelelor de mare viteza, cu rate de transfer de ordinul gigabiţilor pe secunda (Gps). Acestea coexista cu vechile reţele de viteza mica cu care trebuie sa coopereze încă destul timp. Diferenţa dintre vitezele de transfer si de lucru ale acestora creează condiţii de apariţie a congestiei. Congestia apare atunci când rata, sau suma ratelor, de intrare intr-un nod/ruter este mai mare decât capacitatea disponibila a liniei. Cauza sa este eterogeneitatea reţelelor care compun Internet-ul.

Managementul congestiei cuprinde atât mecanismele de evitare a apariţiei congestiei, cat si controlul congestiei, adică mecanismele de refacere după congestie. Despre cauzele apariţiei congestiei si posibilităţile de evitare a apariţiei sale a apărut o literatura de specialitate extrem de bogată [JK, J1, J2, GOV, BC]. Multă vreme s-a crezut ca acestea sunt fie liniile lente, fie memoria insuficienta ,fie procesoarele lente, separat sau in combinaţie. In prezent insa au devenit accesibile liniile si procesoarele de mare viteza, iar preţul memoriilor a scăzut suficient astfel încât ele nu mai sunt o componenta prohibitiva. Problema congestiei nu s-a rezolvat automat, aşa cum se credea, ci a devenit şi mai acută.

Despre actualele reţele s-au făcut o serie de ipoteze, care au determinat o serie de acţiuni, prezentate în cele ce urmează:

i) S-a considerat ca în reţelele de mare viteza traficul va fi în principal trafic video, sau asemănător cu acesta, adică staţionar şi predictibil. În consecinţă serviciul de datagrama ar trebui înlocuit cu cel de rezervare a resurselor.

ii) Volumul mare de date existente pe canalele de mare viteză implică renunţarea la controlul cu reacţie şi apelarea la controlul în buclă deschisă.

iii) Controlul prin ferestre trebuie înlocuit cu controlul ratei.

iv) Deoarece schemele de control a sursei implică informarea sursei despre apariţia congestiei şi sunt prea lente, trebuie deci înlocuite prin scheme de control în rutere.

v) Schema ideală de control a congestiei în reţele de mare viteză este cea de contrapresiune (backpressure), deoarece ea produce o atenuare imediată a acesteia.

BUPT

vi) Este suficientă o singură schemă de management a congestiei.

Aceste ipoteze şi concluzii sunt însă adevărate doar în anumite condiţii şi false în alte condiţii.

1.6.1 Legăturile şi traficul în reţelele de mare viteză

Proiectarea şi alegerea unui algoritm depinde în principal de tipul de trafic, care la rândul sau depinde de aplicaţia în cauza.

Prima problema asupra căreia trebuie luata o decizie este modul de folosire a liniilor de mare viteza:

i) ca o coloană vertebrală (reţele backbone) pentru inter-conectarea reţelelor mici, generatoare de trafic sau,

ii) in interiorul subretelor mici. Apoi, interconectarea acestor reţele mici ar urma sa fie realizata cu linii lente - traficul spre exterior fiind mult mai redus ,fata de cel din interiorul subreţelei.

Actualmente în reţelele locale e folosita o tehnologie de mare viteza , LAN-urile fiind interconectate prin reţele lente de întindere mare, WAN-uri. Aceasta modalitate de interconectare va conduce evident la congestii severe.

Trasând o paralelă între traficul rutier şi cel din reţelele de comunicaţii, apare ca fondată şi cealaltă soluţie, în care traficul de viteză mică de la subreţele (oraşele generatoare de trafic) să fie dirijat pe liniile de mare viteză (şoselele interurbane) spre destinaţie, adică o alta subreţea (oraş). Apare ca justificată soluţia interconectării LAN-urilor prin reţele backbone de mare viteză.

Liniile de mare viteză sunt însă o resursă mult mai costisitoare decât cele de mică viteză, astfel încât e necesar ca ele sa fie partajate între un număr mare de noduri , din subreţelele de viteză mică. Creşterea gradului de partajare a liniilor de viteză mare are o serie de implicaţii. Prima implicaţie este faptul că debitul surselor individuale nu poate fi de ordinul gigabiţilor pe secundă, deşi punţile, ruterele, porţile sau celelalte resurse partajate trebuie să fie capabile să manevreze traficul de asemenea viteze. Apoi, ca un rezultat al partajării reţelei de către o mare varietate de aplicaţii, este faptul ca reţeaua va trebui să satisfacă o mulţime de criterii de performantă. Unele aplicaţii, de voce sau video, sunt sensibile la întârziere dar insensibile la pierderi, altele, ca transferul de fişiere, sunt insensibile la întârzieri dar sensibile la pierderi, respectiv aplicaţiile ca grafica interactivă sau calculul interactiv sunt sensibile atât la întârzieri cat si la pierderi. Fiecare dintre aceste aplicaţii trebuie tratata diferit, iar încercările de tratare imparţială a acestora în suprasarcină vor trebui să ţină cont de cerinţele specifice ale fiecărei surse. Traficul pe liniile de mare viteză va fi un amestec de date, voce, video, sau trafic multimedia.

Proiectarea si implementarea tehnicilor de management a congestiei trebuie să ţină cont de caracteristicile tuturor aplicaţiilor.

BUPT

1.6.2 Controlul fluxului prin ferestre şi prin rată

Controlul fluxului prin ferestre e folosit in majoritatea arhitecturilor de reţele de calculatoare , TCP/IP, DNA,OSI sau SNA [Spi, Tane, NN, SCH, SPT]. Alocarea resurselor bazată pe rata este folosită obişnuit în reţelele de telecomunicaţii, unde fiecare conexiune are asignată o bandă anumită. În ultimul timp au fost propuse şi pentru reţele de calculatoare o serie de protocoale pentru controlul fluxului bazate pe rată: la acestea, nodul destinaţie specifică rata maximă la care sursa poate transmite pachetele (numărul de pachete într-un timp dat).

Înlocuirea controlului prin ferestre, cu controlul bazat pe rata, s-ar justifica prin faptul că în curând, strangularea traficului nu va mai fi cauzată de lipsa de memorie. Strangularea va fi însă produsă de procesoare, linii, sau dispozitivele de memorare, care sunt limitate ca rată, în sensul că ele nu pot accepta sosirea pachetelor sau a biţilor la o rată depăşind capacitatea lor. Memoriile realizau strangularea deoarece nu puteau accepta un număr mai mare de pachete/biţi decât capacitatea lor, dar nu conta viteza de sosire a acestora. Controlul prin ferestre trebuia sa evite strangularea, adică depăşirea capacităţii memoriei. Alta problema a controlului prin ferestre o constituie transmiterea înlănţuită a tuturor pachetelor din fereastra (back to back), ceea ce conduce la rafale de trafic. Un alt argument in favoarea controlului prin rata este ca o parte din ce in ce mai mare a traficului de mare viteza este orientat pe curente/fluxuri (voce sau video) care cere o garantare a ratei, mai degrabă decât o contorizare a pachetelor, spre deosebire de traficul de date care pretinde respectiva contorizare.

Despre controlul prin rata exista insa si unele păreri greşite. În primul rând se neglijează faptul că rata λ rezultă din raportul a două mărimi, (n /T) adică numărul de pachete transmise într-un interval de timp. Combinarea acestor mărimi nu se poate face arbitrar. Astfel un procesor, capabil sa prelucreze un pachet la fiecare milisecundă, nu va putea face acest lucru dacă pachetele sosesc în rafală , 5 pachete la fiecare 5 milisecunde. De aceea, toate schemele bazate pe rata, inclusiv "găleata găurită" şi variantele sale [Tane, LLD, AST, BZ, DK], pretind specificarea dimensiunii rafalei si a intervalului dintre rafale. Modelele analitice ignora insa cei doi parametrii si lucrează doar cu unul singur, λ.

În plus, nu e clar pentru toţi că deoarece controlul bazat pe rată este un control nod la nod (hop by hop), toate sistemele din cale, punţi sau rutere, trebuie să fie conştiente care sunt parametrii referitori la rata şi trebuie să–i aplice. Acest control nu poate fi aplicat doar printr-un mecanism cap la cap. Fără forţarea nod la nod, câteva rafale ar putea fi combinate într-o singură rafală de către sistemele intermediare. Ca exemplu, o punte care nu e conştientă de parametrii referitori la rată, poate schimba un flux în care apare cate un pachet la fiecare milisecundă, într-un flux cu rafale de 5 pachete la 5 milisecunde. Destinaţia nu va putea sa proceseze sau să accepte fluxul alterat astfel.

BUPT

Deci, controlul bazat pe rata necesita o abordare orientata pe conexiune, deoarece parametrii *n* şi T trebuie cunoscuţi şi acceptaţi de toate sistemele din cale. Implementarea controlului bazat pe rata în reţele fără conexiune e dificilă. Pe de alta parte, controlul cu ferestre poate fi aplicat cap la cap, nod la nod, sau în ambele versiuni [LT, MKT, LX]. În versiunea cap la cap, sistemele intermediare nu trebuie informate despre dimensiunea ferestrei stabilita de destinaţie [KR, JDSZ, KalV].

După cum rezultă din cele prezentate, controlul admisiei la intrarea în reţea bazat pe rată nu este suficient singur. Controlul ratei trebuie forţat în fiecare nod al reţelei. Majoritatea discuţiilor despre controlul bazat pe rată, ca de exemplu găleata găurită şi variantele sale, s-au limitat la controlul admisiei. Acest lucru poate fi valabil doar în reţele cu un număr mic de noduri, dar in reţelele cu număr mare de noduri controlului admisiei trebuie suplimentat cu mecanisme implementate în noduri. Recent, Golestani [SV, J1] a propus o schemă cu oprire şi pornire, de tip "stop-and-go", pentru urmărirea serviciului, care permite forţarea ratei şi în nodurile intermediare.

În al treilea rând, cu un control dinamic bazat pe rata, există posibilitatea unei creşteri importante a lungimii cozilor de pachete atunci când rata totală de intrare se apropie de capacitatea maximă [BC, ABB, ARZ, BM GBCHP]. Controlul bazat pe rată conduce la întârzieri totale mai mari , incluzând atât întârzierea în coada sursei cât şi în cele din reţea. Rata de pierderi este mai mare decât la controlul prin ferestre, la încărcări depăşind 60% din capacitate. Creşterea lungimii cozilor depinde de controlul întârzierii şi de reacţie, lungimea putând ajunge la câteva mii de pachete. Acest lucru se poate întâmpla când reacţia e pierdută sau întârziată. Chiar dacă exista memorie suficientă, memorarea unui număr excesiv de pachete duce la întârzieri inacceptabil de mari. La controlul prin ferestre intrările in reţea se opresc automat când dimensiunea ferestrei e atinsă din cauza creşterii întârzierii reacţiei. Cu alte cuvinte schemele cu ferestre lucrează inerent în buclă închisă, iar cele bazate pe rata lucrează în buclă deschisă.

Menţinerea cozilor în anumite limite rezonabile implică transformarea controlului bazat pe rată într-un control în buclă închisă. O cale ar fi suplimentarea controlului prin rată cu o fereastră de dimensiuni mari, care devine operativă doar când lungimea cozilor creşte, din cauza unei combinaţii inacceptabile a diferitelor tipuri de trafic. În acest caz, sursele, constatând că s-a ajuns la limita superioară a ferestrei vor opri intrarea traficului in reţea, până când o noua confirmare de la destinaţie va redeschide fereastra. Dimensiunea uzuală a ferestrei este acum de 1 până la 32 de pachete, dar dacă mecanismul ferestrei se utilizează împreună cu controlul prin rată, dimensiunea ferestrei va trebui crescută cu una sau doua ordine de mărime. [KES, BC]. În tabelul 1.1 sunt prezentate sintetic cele doua mecanisme de control, prin rata si prin fereastra.

BUPT

Tabelul 1.1 Controlul prin fereastra si prin rata

Tip de control	*Cu fereastră*	*Bazat pe rată*
Control	Fereastra(W)	Număr de pachete (n) şi Interval de timp (T)
Rata efectiva	Fereastră / Întârziere dus-întors	n/T
Necesar dacă strangularea e produsă de:	Memorie	Procesoare, legaturi sau alte dispozitive
Lungimea maxima a cozii	Limita sumei ferestrelor	Nelimitată
Rafale	Rezulta trafic în rafale	Nu sunt rafale la sursă
Întinderea controlului	Cap la cap, nod la nod, sau ambele	Nod la nod
Nivelul reţea	Fără conexiune sau orientat pe conexiune	Orientat pe conexiune

1.6.3 Controlul cu reacţie şi controlul în buclă deschisă

În reţelele actuale de mare viteză (Gps) şi la dimensiunea de pachete folosită, întârzierea de propagare (timpul necesar primului bit al pachetului să traverseze reţeaua e considerabil mai mare decât timpul de transmisie a pachetului (timpul dintre primul şi ultimul bit al pachetului). Pe măsura creşterii vitezei liniilor, numărul de biţi aflaţi pe canal (pipe), creşte, şi deci creşte şi numărul de pachete: astfel, pe o legătură pe fibră de 5.000 km, la un debit de 1 Gps, pot exista 25 Mb de date. La o dimensiune medie de 512 octeţi pe pachet, aproximativ 6.000 de pachete se vor afla pe canal (dimensiunea actuală uzuală a pachetelor este însă de 128-256 octeţi).

Multe din vechile scheme de control a congestiei sunt scheme în buclă închisă, în sensul că resursele congestionate transmit semnale de reacţie surselor de trafic, care îşi ajustează în acest caz volumul traficului. S-a argumentat ca aceste scheme sunt însă prea lente, deoarece până ce sursa primeşte reacţia şi reacţionează, se pot pierde mii de pachete. Acest fapt a dus la apariţia câtorva abordări în buclă deschisă, care să nu necesite reacţie, cum sunt: controlul în rutere, rezervările apriori, sau contrapresiunea, ale căror proprietăţi vor fi prezentate în cele ce urmează.

1.6.4 Controlul la sursă şi controlul în rutere

Reţeaua de comunicaţii e formata din rutere, înţelegând prin aceasta punţi, multiplexoare, rutere sau porţi, si sursele de date, adică toate sistemele terminale

BUPT

ataşate reţelei, interconectate prin diverse linii sau medii. Concepţia trecută despre schemele de control a congestiei prevedea că ruterele să transmită semnale de reacţie surselor, care să poată astfel iniţia măsuri de remediere a congestiei. Acesta este controlul la sursa, folosit prin pornirea lenta de la TCP/IP, la debit, etc. [DS]. Exista o serie de argumente împotriva acestor scheme. Primul este întârzierea mare dintre momentul sesizării congestiei şi momentul în care se iau masuri de remediere. Dacă durata sesiunii e redusa, reacţia poate ajunge după încheierea acesteia. În al doilea rând, sursele pot să coopereze sau nu, la primirea reacţiei. Ca exemplu, e posibil ca o sursă să-şi reducă intervalul dintre retransmisii, chiar în condiţiile unor pierderi mari prin reţea, obţinând o rată mare de succes. Iar în al treilea rând, anumite reacţii implică generarea unor pachete suplimentare în reţea.

Controlul la rutere nu suferă de aceste probleme, deoarece ruterele îşi distribuie resursele independent de surse. Shenker a demonstrat [DKS] ca acest tip de control este necesar pentru imparţialitate deoarece controlul la sursa deşi este eficient nu este întotdeauna imparţial. Politica de distrugere aleatoare a pachetelor [BC], servirea imparţială (fair queuing) [KKK], şi contrapresiunea fac parte din controlul la rutere.

Controlul la sursa a fost aplicat cu succes atât în reţelele private, unde atât sursele cât şi reţeaua aparţin aceleiaşi organizaţii, dar în reţelele publice de date, unde doar ruterele aparţin companiilor de telecomunicaţii şi sursele nu pot fi controlate, el trebuie suplimentat cu controlul în rutere. Controlul la sursa foloseşte în general protocolul de nivel reţea pentru transmiterea reacţiei şi protocolul de nivel transport pentru răspândirea reacţiei pe toate caile. De aceea, controlul in rutere este bun pentru realizarea imparţialităţii la supraîncarcare de scurtă durată, iar controlul la sursă e necesar pentru supraîncarcarea de durata.

Câteva din propunerile recente [BC, STEV, FF] au adus argumente în favoarea distrugerii pachetelor, ca o soluţie de dorit pentru controlul congestiei în suprasarcină, la viteză mare. Pentru unele aplicaţii, voce/video, pierderea unor pachete este acceptabilă. Pentru majoritatea aplicaţiilor de date însă, fiecare pachet este important. Aici, fiecare pierdere de pachete trebuie recuperată printr-o retransmisie, atât a pachetului respectiv cât şi a altora, operaţie costisitoare la orice viteză s-ar lucra. În reţelele de mare viteză numărul de pachete de retransmis este mai mare, datorită dimensiunilor crescute ale canalului. De aceea înainte de a se ajunge la distrugerea pachetelor, e necesar sa fie aplicate metode de evitare a congestiei.

Indisciplina surselor din reţelele publice poate fi depăşită, plasând controlul in punctul de acces în reţea (DCE-uri). Ruterele din interiorul reţelei pot să transmită reacţia ruterelor din punctele de acces, care vor putea sa accepte sau sa respingă traficul de intrare de la diferitele surse, folosind o schema de alocare imparţială.

În concluzie, controlul in rutere e necesar pentru imparţialitate şi lucrul în suparsarcina de scurta durata, iar controlul la sursa e necesar pentru suprasarcina de lunga durata. Dacă scopul este viteza de procesare, ruterele trebuie făcute cât

BUPT

mai simple, iar dacă scopul este rezolvarea indisciplinei surselor, atunci trebuie folosit controlul în rutere, în detrimentul vitezei rețelei (vezi tabelul 3.2).

Tabelul 3.2 Comparație între controlul la sursă și în rutere

Tipul controlului	*În rutere*	*La sursă*
Exemple	Distrugere aleatoare Tratare imparțială a cozilor Presiune înapoi	Ferestre dinamice Pornire lenta DEC bit
Întârziere	Nu are	Întârzierea reacției
Antetele necesare reacției	Nu are	Biți sau mesaje de reacție
Antete în	Rutere	La surse
Necesar dacă	Nu exista control asupra surselor	Supraîncărcări de durata
Imparțialitate	Realizabila	Nu e garantata

1.6.5 Contrapresiunea

Contrapresiunea a fost considerata în unele rapoarte ca principalul, sau chiar unicul, mecanism de tratare a congestiei [J1, Tane, BG, STA]. Aceasta este o forma de control on/off a fluxului de tip nod la nod. Ruterele congestionate transmit un semnal de oprire a transmisiei (X-Off) ruterelor sau surselor învecinate si nu mai accepta pachete până ce lungimea cozilor lor nu scade. Când sarcina a scăzut ele transmit un semnal "X-On" şi fluxul de pachete reîncepe.

Contrapresiunea e un mecanism de nivel legătură de date, ceea ce îl face sa aibă o bucla de reacție mult mai redusă decât mecanismul de nivel transport. De aceea acționează foarte bine dacă durata suprasarcinii este mică. Pe acest interval se folosesc resursele ruterelor vecine pentru a suporta suprasarcina. Contrapresinea are acelaşi efect ca si creşterea numărului de tampoane de memorie. Dar dacă supraîncărcarea se prelungeşte contrapresiunea poate să oprească toată rețeaua, din cauza opririi traficului de intrare în rețea.

Pentru suprasarcini de lungă durată, contrapresiunea este mai eficientă în rețelele mici decât în rețelele mari, deoarece în cele mici, sursele fiind mai aproape de rutere primesc mai rapid semnalele de încetinire. Această constatare trebuie avută în vedere la simulări, unde rezultatele obținute pentru rețele mici nu pot fi extrapolate la rețele mari.

Contrapresiunea nu este imparțială, în sensul că traficul care nu traversează resursele congestionate este defavorabil influențat, datorită răspândirii congestionării. Partajarea liniilor de către un număr mare de surse duce la

622109
3661
BUPT

nivelarea traficului [AST, CC, DK, KA], ceea ce face ca suprasarcina de scurtă durată să fie puțin probabilă în rețele de mare viteză.

În concluzie, contrapresiunea poate fi utilizată doar pentru suprasarcini de durată mică după care ea trebuie să înceteze. Pentru suprasarcini de durată mare, contrapresiunea trebuie însoțită de un control la nivelul rețea sau transport.

1.6.6 Modul de lucru cu rezervare şi modul de lucru fără rezervare

Rezervarea înseamnă că sursa trebuie să rezerve resursele în etapa de stabilire a conexiunii. Rezervarea este necesară în cazul traficului staționar, cum este traficul de voce sau video [CT, DK, ZDESZ, HL]. Rezervarea este necesară şi în cazul traficului de date. Cum acesta are o durată foarte mică, dacă s-ar folosi reacția, cel mai probabil este că, atunci când reacția ajunge la sursă, aceasta şi-a încetat deja transmisia. De fapt acesta nu este altceva decât efectul raportului dintre durata transmisiei pachetului/mesajului față de durata de propagare: la rate de Gbps, transmiterea unui facsimil poate dura câteva milisecunde, în timp ce reacției îi trebuie câteva zeci de milisecunde să parcurgă rețeaua. Garantarea calității serviciilor, ca lățimea de bandă sau întârzierea, e dificil de realizat fără rezervare. Rezervarea facilitează administrarea resurselor, deoarece se cunosc în avans nu numai resursele ci şi cererile. In absența rezervării, administrarea este dinamică şi dificilă.

Principalul dezavantaj al rezervării este risipa resurselor care, odată rezervate de către un utilizator nu mai pot fi folosite de altul, nici dacă primul nu le foloseşte.

Rezervarea reprezintă o soluție foarte bună pentru traficul staționar, dar dezavantajoasa pentru traficul in rafale [GF, GP, HL]. Sistemele distribuite au nevoie de o comunicare extrem de rapidă între procesoare, sau între procesoare si dispozitivele de memorie, ceea ce se traduce printr-o rată de transfer de ordinul Gbps pe distanțe de doar câțiva km. Deşi distanțele vor creşte, traficul va rămâne tot în rafale, deci rezervarea nu e recomandabilă.

Alt dezavantaj al rezervării îl constituie antetele implicate, care nu se justifică pentru sesiunile de durată mică.

În concluzie, rezervarea nu este recomandabilă pentru medii dinamice, cu trafic în rafale, ci doar pentru traficul staționar, de lungă durată. Alegerea între soluția cu rezervare sau fără nu depinde de viteza rețelei (vezi tabelul 3.3). Pentru rețelele viitoare e de aşteptat ca traficul să fie de ambele tipuri, ceea ce conduce la consecința logică a necesității adoptării şi prezenței ambelor tipuri de administrare a resurselor, atât cu cât şi fără rezervare. De altfel au apărut deja o serie de soluții hibride, ca de exemplu rezervarea resurselor, de către primul pachet dintr-o secvență, anularea rezervării având loc la sfârşitul secvenței [GAP, JSD, KL, KMR, KWC]. Pentru a evita pierderea primului pachet din

BUPT

secvența, din cauza resurselor ocupate, rezervarea se poate face şi printr-un pachet special.

Tabelul 3.3 Comparație între modul de lucru cu rezervare şi fără rezervare

Mod de lucru	*Cu rezervare*	*Fără rezervare*
Garanții despre	Lățimea de bandă şi /sau întârzierea	Lățimea de banda sau întârzierea sunt variabile
Administrarea resurselor	Facilă	Dificilă
Resursele neutilizate	Sunt pierdute	Pot fi folosite de celelalte surse
Aplicabil la	Trafic staționar (voce / video)	Trafic în rafale (date)
Stabilirea conexiunii	E necesară (recomandabilă pentru sesiuni lungi)	Nu este necesară (recomandabilă pentru sesiuni scurte)
Stare	Dinamică redusă	Dinamică mare

1.6.7 Alegerea schemei de control a congestiei

În final întrebarea care se pune este asupra criteriului de alegere a schemei de control a congestiei. Din cele expuse rezultă că alegerea depinde, de fapt, de durata suprasarcinii. Cu cât durata congestiei creşte, cu atât nivelul la care trebuie exercitat controlul e mai ridicat (vezi figura 1.2).

Durata congestiei	Mecanismul de control a congestiei
Mare ↓ Redusă	Proiectarea rețelei şi planificarea capacității Controlul admisiei conexiunii Dirijarea dinamică Compresia dinamică Reacția cap la cap Reacția nod la nod Asigurarea unei memorii suficiente

Fig. 1.2 Abordări recente ale controlului congestiei în rețele ATM

Dacă congestia e permanentă, atunci trebuie instalate linii suplimentare de mare viteză. Când congestia se manifestă pe intervale apropriate de durata unei sesiuni, cel mai potrivit control este cel de nivel sesiune, ca de exemplu semnalul de ocupat.

Dacă durata congestiei este de câteva ori mai mare decât întârzierea de propagare dus-întors, atunci e necesară o schemă de control la nivelul transport,

BUPT

cu reacție de la nivelul rețea. Dacă durata congestiei e mai redusă, trebuie folosit un control de nivel rețea (variante de control în rutere sau control a admisiei), sau de nivel legătura de date (contrapresiunea). Cum în rețelele reale congestia se poate manifesta la oricare din aceste scale de timp, rezultă că trebuie realizat un control combinat, la toate nivelurile. Nici o schemă nu poate singura să rezolve toate problemele legate de congestie.

O propunere ar fi să se utilizeze un control al admisiei de tip găleată găurită pentru operarea normală, un control la sursă pentru pierderi de pachete, respectiv, refuzul sesiunii pentru congestia pe termen lung [JSD, SSL]. Alta propunere este să fie permisă producătorului, alegerea uneia sau nici uneia din schemele de control. Această soluție conduce însă, în cele mai multe cazuri, la o comportare părtinitoare şi necontrolabilă a rețelei. De aceea este mult mai bine să existe în această parte a proiectării şi construcției rețelei, o singură regulă, decât să fie permisă alegerea regulii proprii de către fiecare participant, având în vedere că aici are loc o partajare a resurselor.

BUPT

Capitolul 2

CONTROLUL TRAFICULUI

Mecanismele de control a traficului şi de control a congestiei au ca scop să menţină performanţele reţelei atunci când în reţea sunt prezente diferite tipuri de trafic. În B-ISDN, congestia e definită ca o stare a elementelor reţelei – comutatoare, concentratoare, linii de transmisie – în care reţeaua nu mai e capabilă să asigure transferul datelor la parametrii negociaţi pentru conexiunile deja existente şi/sau pentru noile cereri de conectare [DK]. Congestia poate apărea din cauza unor fluctuaţii care nu pot fi prevăzute, ale traficului fluxurilor şi/sau unor defecţiuni în reţea. Trebuie făcută distincţia între controlul traficului şi controlul congestiei. Controlul traficului se referă la o serie de acţiuni întreprinse de reţea pentru a evita situaţiile de congestie, el fiind un control preventiv. Controlul congestiei se referă la acţiunile întreprinse de reţea atunci când congestia s-a instalat deja, pentru reducerea la minimum posibil a intensităţii, răspândirii şi duratei congestiei, el fiind un control reactiv.

În acest capitol vor fi prezentate câteva scheme de control a traficului, scopul acestora fiind de a proteja atât utilizatorul cât şi reţeaua pentru a obţine performanţele dorite. Cu aceste scheme se monitorizează şi controlează traficul oferit la fiecare interfaţă, pentru a proteja resursele reţelei de traficul surselor "răutăcioase" [DK, BM, JDSZ] sau a excesului neintenţionat de trafic care afectează Q_oS al conversaţiilor deja existente, detectând abaterile faţă de parametrii negociaţi şi luând măsuri adecvate. Traficul în exces, neconform cu parametrii negociaţi va fi respins pentru a minimiza efectul său asupra celorlalte conversaţii. Conversaţiile cu trafic de rată variabilă, VBR, trebuie să specifice rata medie a datelor, mediată pe intervale mari de timp, care trebuie monitorizată şi controlată. Identificarea traficului neconform, pe baza ratei medii este o problemă dificilă. Mecanismul de control poate, fie să rejecteze pur şi simplu acest trafic suplimentar, fie să-l admită, marcând eventual pachetele respective care dacă e necesar vor fi primele distruse.

2.1 Formarea traficului

Traficul în rafală constituie una din principalele cauze ale congestiei. [Tane, LCH]. Dacă s-ar putea impune calculatoarelor gazdă să transmită cu o rată uniformă, congestia ar apărea mult mai rar. Gestionarea congestiei prin impunerea unei rate previzibile cu care să fie transmise pachetele se numeşte formarea traficului (traffic shapping) şi este larg răspândită în reţelele ATM. Ea este o metodă în buclă deschisă sau o metodă de control preventiv: formarea traficului se ocupă cu uniformizarea ratei medii de transmisie a datelor, deci cu atenuarea rafalelor, şi nu cu volumul de date aflat la un moment dat în tranzit, cu

BUPT

care se ocupă protocoalele cu fereastră glisantă.

2.1.1 Algoritmul găleții găurite

Funcționarea algoritmului este similară cu a unei găleți găurite, (leaky-bucket) care indiferent de rata cu care este umplută, lasă să iasă apa la o rată constantă [Tane]. Pentru pachetele de date, găleata este o coadă cu capacitate finită şi rată de transmisie constantă [NG, DK]. Atunci când coada este plină noile pachete care sosesc vor fi distruse. Dacă există pachete în coadă calculatorul gazdă livrează în rețea câte un pachet la fiecare tact al ceasului, astfel că fluxul neregulat de pachete sosite de la procesele sale se transformă într-un flux uniform, netezind rafalele. Dacă pachetele au toate aceeaşi dimensiune, de exemplu cele ATM, algoritmul poate fi folosit aşa cum a fost descris. Dacă pachetele au lungimi variabile, e preferabil ca la fiecare tact să se transmită un număr fix de octeți. Astfel, dacă regula este 1024 de biți la fiecare tact, atunci se pot transmite fie un pachet de 1024 de biți, fie două pachete de 512 biți, patru de 256 biți etc. Chiar dacă numărul rezidual de octeți este scăzut, următorul pachet va trebui să aştepte următorul tact.

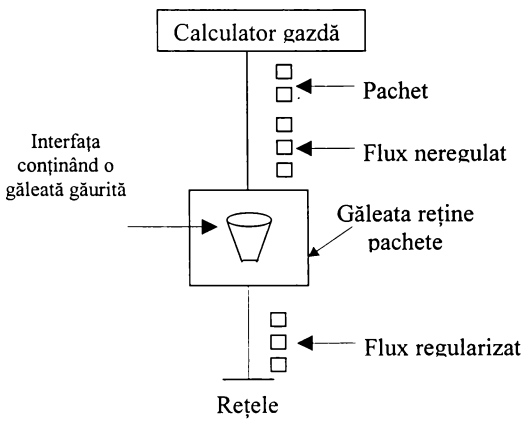


Fig. 2.1 Algoritmul găleții găurite

Algoritmul găleții folosind contorizarea octeților este implementat aproximativ în acelaşi fel. La fiecare tact un contor este inițializat la n. Dacă primul pachet din coadă are k octeți, cu $k < n$, pachetul este transmis şi contorul devine n-k. Mai pot fi transmise şi alte pachete, atâta timp cât contorul rămâne mai mare decât lungimea primului pachet din coadă. Când contorul scade sub această lungime, transmisia încetează până la următorul tact, când contorul este reinițializat şi vechea valoare este pierdută. După cum se observă în acest caz poate apărea cazul în care o parte din intervalul de timp dintre două tacte este nefolosit pentru transmisie.

Ca un exemplu de găleată găurită, considerăm un calculator care produce

BUPT

date cu rata $R = 200Mbps = 25MBps$ şi o reţea de aceeaşi viteză. Dar ruterele pot să gestioneze date până la această viteză doar pentru un scurt interval de timp, pentru intervale mai mari de timp lucrează optim pentru rate care nu depăşesc $2MBps$. Rezultă atunci că rata de transmisie ρ a găleţii este $\rho=2MBps$. Dacă notăm cu T_{1r} timpul de transmisie a rafalei de către calculator, cu T_{2r} timpul de transmisie a rafalei de către găleată şi cu C = capacitatea găleţii rezultă:

$$T_{1r} = \frac{C}{R} \tag{2.1a}$$

$$T_{2r} = \frac{C}{\rho} \tag{2.1b}$$

Rezultă că la o capacitate a găleţii $C=1MB$, calculatorul poate transmite rafale de $T_{1r}=40msec$ fără pierderi de date. Aceste rafale vor fi împrăştiate de-a lungul a $500msec$; indiferent de cât de repede sosesc. Dacă rafalele sunt mai lungi de $40msec$, sau 1 milion de octeţi la fiecare secundă pentru exemplul considerat, capacitatea găleţii este depăşită şi datele vor fi pierdute. Traficul total pe durata unui interval T va fi mărginit de $RT+C$.

2.1.2 Algoritmul găleţii cu jeton

Algoritmul găleţii găurite impune un model rigid al ieşirii, din punct de vedere al ratei medii, indiferent de cum arată traficul. Pentru numeroase aplicaţii este mai convenabil să se permită a creşterii a vitezei de ieşire la apariţia unor rafale mari, astfel încât este necesar un algoritm mai flexibil de preferat unul care nu pierde date. Un astfel de algoritm este algoritmul găleţii cu jeton [JDSZ, Tane, PG1, DK] (token bucket). În acest algoritm, găleata găurită păstrează jetoanele, generate de un ceas cu rata de 1 jeton la fiecare ΔT secunde. Pentru ca un pachet să poată fi transmis el trebuie să capteze şi să distrugă apoi un jeton. Dacă n pachete aşteaptă să fie transmise şi găleata are m jetoane, $m \geq n$, vor fi

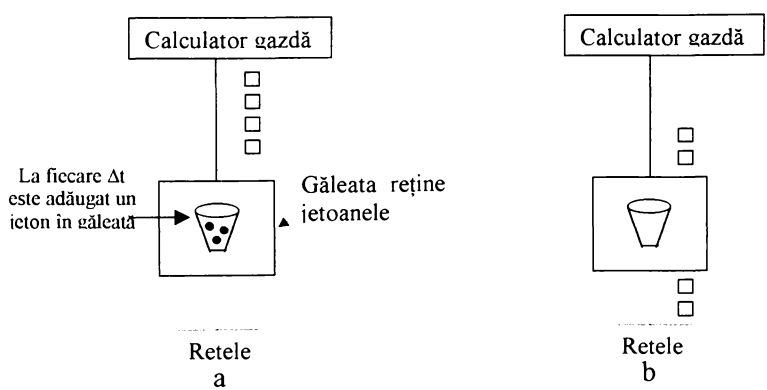


Fig. 2.2 Algoritmul găleţii cu jeton
a) înainte **b) după**

BUPT

transmise doar m pachete iar restul de *n-m* pachete aşteaptă să fie generate alte *n-m* jetoane.

Algoritmul găleții cu jetoane asigură o formare diferită a traficului comparativ cu algoritmul găleții găurite. Algoritmul găleții găurite nu permite calculatoarelor gazdă inactive să acumuleze permisiuni de trecere pentru a trimite o rafală mai mare, ulterior. Algoritmul găleții cu jeton permite această acumulare, mergând până la dimensiunea maximă a găleții N. Astfel pot fi transmise rafale de până la N pachete şi se asigură un răspuns mai rapid la apariția bruscă a unor rafale de intrare. În plus, nu apar pierderi de date, fiindcă algoritmul găleții cu jeton aruncă jetoanele la umplerea găleții, dar nu distruge pachetele. O variantă a algoritmului este situația în care un jeton reprezintă dreptul de a transmite k biți, nu un pachet. Un pachet va putea fi transmis doar dacă există suficiente jetoane pentru a-i acoperi lungimea în biți.

Ambii algoritmi sunt folosiți pentru a regla traficul de ieşire al calculatoarelor gazdă. Pentru reglajul traficului între rutere aplicarea lor poate duce la pierderi de date.

Dacă T_{1r}, este lungimea rafalei în secunde, C capacitatea găleții în octeți, $r=1\ \Delta T$ rata de sosire a jetoanelor şi ρ rata maximă de ieşire în oct/sec., atunci rafala de ieşire conține maxim:

$$C + r\,T_{1r} = \rho\,T_{1r} \tag{2.2a}$$

octeți. De aici se poate obține durata maximă a rafalei:

$$T_{i,r} = \frac{C}{\rho - r} \tag{2.2b}$$

Durata rafalei poate fi astfel reglată prin selectarea atentă a valorilor r şi ρ.

Algoritmul permite apariția unor rafale mari. De multe ori este nevoie să se reducă valoarea de vârf, dar fără a reveni la valorile scăzute permise de algoritmul original, al găleții găurite. O cale de a obține un trafic mai uniform este de a pune o găleată cu jeton, de rată r, urmată de o găleată găurită de rată ρ cu r ρ R_{max} a rețelei.

Propunerile curente referitoare la serviciul de control a încărcării în Internet specifică faptul că pachetele, destinate să treacă prin mecanismul de control a încărcării, şi care depăşesc parametrii găleții cu jetoane pentru acel flux să fie transmise mai departe conform cu regulile pentru efortul maxim, dacă resursele permit acest lucru şi ca ruterele să implementeze mecanismul de efort maxim în cozile de aşteptare ale căilor, pentru a proteja traficul de efort maxim neelastice. Aceste cerințe sunt ortogonale cu cerințele impuse de algoritmii de control a admisiei.

BUPT

2.2 Algoritm de control a admisiei

Propunerile curente referitoare la serviciul de control a încărcării în Internet specifică faptul că pachetele, destinate să treacă prin mecanismul de control a încărcării, şi care depăşesc parametrii găleţii cu jetoane pentru acel flux, să fie transmise mai departe conform cu regulile pentru efortul maxim, dacă resursele permit acest lucru şi ca ruterele să implementeze mecanismul de efort maxim în cozile de aşteptare ale căilor, pentru a proteja traficul de efort maxim elastic de fluxurile de efort maxim neelastic. Aceste cerinţe sunt ortogonale cu cerinţele impuse de algoritmii de control a admisiei.

Scopul acestora este de a asigura ca admisia unui nou flux în reţeaua cu resurse limitate să nu ducă la încălcarea angajamentelor de serviciu ale reţelei faţă de fluxurile deja admise. Se controlează volumul datelor în tranzit prin reţea şi nu rata la care se transmit acestea. Algoritmii pentru controlul încărcării prin controlul admisiei sunt de două tipuri [GK, JSD, FJ1, FJ2]: algoritmii bazaţi pe parametrii în care se asigură ca suma resurselor rezervate plus noile rezervări să nu depăşească capacitatea reţelei, respectiv algoritmii bazaţi pe măsurări care folosesc măsurările pentru a determina încărcarea dată de traficul existent şi cel nou sosit, pentru a lua decizia de admisie. Aceştia din urmă folosesc măsurarea lăţimii de bandă, a lăţimii de bandă echivalente, respectiv a regiunii de acceptantă. Într-un mediu cu trafic majoritar de efort maxim, scopul secundar al algoritmilor de control a admisiei poate fi pur şi simplu controlul fracţiei de bandă alocată traficului de timp real, mai degrabă decât maximizarea acestuia.

Criteriul principal pentru evaluarea oricărui algoritm de control a admisiei este faptul cât de bine asigură îndeplinirea angajamentelor de serviciu, mai precis ca aceste angajamente să nu fie încălcate. Cea mai simplă cale de a asigura îndeplinirea completă a angajamentelor este de a aloca resurse suficiente pentru a face faţă cazului cel mai defavorabil al fiecărui flux. Această soluţie duce însă, pentru traficul în rafale, la o utilizare scăzută a reţelei. S-a ajuns astfel la al doilea criteriu de evaluare şi anume ce grad de utilizare a reţelei se poate obţine pentru un anumit algoritm de control a admisiei, în condiţia respectării angajamentului de serviciu. Al treilea criteriu de evaluare este costul implementării şi operării algoritmului. Deoarece controlul admisiei se face la nivelul transport şi nu la nivelul sesiune, e de aşteptat ca implementarea şi operarea mecanismului de control a admisiei să nu fie un factor prohibitiv.

Aceşti algoritmi sunt folosiţi de serviciul de control a încărcării, care e proiectat pentru aplicaţii adaptive de timp real, care pot tolera variaţii în întârzierea pachetelor. Modelul de serviciu cu controlul încărcării e definit [JSD] că: aproximează bine comportarea vizibilă aplicaţiilor care necesită serviciul de efort maxim, în condiţii de sarcină mică, pe aceeaşi cale. În plus aplicaţiile care cer serviciul de control a încărcării pot presupune că rata lor de pierdere de pachete e determinată de rata erorii de transmisie şi că întârzierea sa tipică e cauzată de întârzierea de propagare pe cale [JSD, DFM, DK, DM, EM, Fl]. Mai precis întârzierea medie a pachetelor în cozi nu poate fi mai mare decât timpul

BUPT

de rafală a fluxului şi rata de pierderi, mediată pe un timp mai mare decât timpul de rafală, ar trebui să fie minimă. Timpul de rafală, s-a văzut în paragraful anterior că, e timpul necesar pentru a servi rafala maximă de flux la rata rezervată fluxului.

Controlul admisiei se realizează la nivelul apelului. Deşi specificaţiile serviciului pentru controlul încărcării nu indică anumite valori pentru QOS, ca de exemplu limita întârzierii sau rata de pierderi, tocmai aceşti parametri se iau în considerare pentru evaluarea algoritmilor de control a încărcării.

2.2.1 Algoritmul "suma simplă"

Algoritmul denumit suma simplă [JSD, Fl, MS] asigură pur şi simplu ca suma resurselor solicitate să nu depăşească capacitatea liniei. Se notează cu $\sum_{i=1}^{n} r_i = r$ – suma ratelor rezervate, C - lăţimea de bandă a liniei, $n+1$ - numele fluxului cerând admisia, r_{n+1}- rata solicitată de fluxul aceasta. Algoritmul va accepta un flux nou doar dacă:

$$r + r_{n+1} < C \tag{2.3}$$

Acesta e cel mai simplu algoritm de control al admisiei şi deci cel mai des folosit de producătorii de rutere. Uneori pentru a asigura întârzieri mici în cozi, cerute de serviciul de control a încărcării, se foloseşte şi o aproximare a disciplinei de planificare de servire a cozii, denumită planificare imparţială ponderată, WFQ (Weighted Fair Queuing), împreună cu acest algoritm de control a admisiei. WFQ asigură fiecărui flux o coadă proprie, servită la rata proprie rezervată, izolând astfel fiecare flux de rafalele celorlalte fluxuri.

2.2.2 Algoritmul "Suma măsurată"

În timp ce algoritmul "suma simplă" asigură ca suma rezervărilor existente plus noile rezervări care se cer să nu depăşească capacitatea reţelei, algoritmul "suma măsurată" foloseşte măsurarea pentru a estima încărcarea dată de traficul existent [JSD, Flo, KR]. Acest algoritm admite fluxuri noi dacă e verificat testul:

$$\hat{r} + r_{n+1} < \rho C \tag{2.4}$$

Unde: $\hat{r} = \sum_{i=1}^{n} r_i$ este încărcarea măsurată date de traficul existent;

r_{n+1} este rata cerută de fluxul $n+1$;

BUPT

ρ este utilizarea dorită, definită de utilizator;
C este lățimea de bandă a liniei.

Pe măsura admiterii unui flux nou, încărcarea estimată e crescută, folosind relația:

$$\hat{r}' = \hat{r} + r_{n+1} \tag{2.5}$$

În cazul cozii simple de tip *M/M/1* lungimea cozii creşte foarte mult, pe măsură ce sistemul se apropie de utilizarea completă, ($\rho \to 1$). [SCH]. Algoritmul "suma măsurată" va eşua când variația întârzierii este foarte mare, situație care apare când utilizarea este foarte mare. Este deci necesar să se stabilească utilizarea dorită şi să se mențină prin algoritmul de control, utilizarea sub această valoare ($\rho = 0,9$).

Măsurarea încărcării se face într-o fereastră de timp T, în care eşantionarea se face la intervale S. la sfârşitul ferestrei de măsurare T, valoarea estimată a încărcării pentru următoarea fereastră T se alege egală cu cea mai mare medie din fereastră. Pentru a avea un număr static semnificativ de eşantioane $T/S \geq 10$. O valoare mică pentru S duce la valori medii mari, iar o valoare mare pentru T duce la medii mai "netede", şi luarea în considerare a unei istorii mai lungi. Ambele fac algoritmul de control al admisiei mai conservativ.

2.2.3 Algoritmul regiunii de acceptanță

Acest algoritm, bazat pe măsurare [J1, FF, KR], calculează o regiune de acceptanță, care maximizează utilizarea cu prețul pierderii de pachete. Având date lățimea de bandă a liniei, dimensiunea tampoanelor de memorie din comutator, parametrii găleții cu jetoane pentru filtrarea fluxului, lungimea rafalei fluxului şi probabilitatea dorită cu care încărcarea actuală să depăşească limita se poate calcula pentru un tip dat de fluxuri o regiune de acceptanță, peste care nici un flux de tipul respectiv să nu mai fie admis. Calculul regiunii de acceptanță implică procese Poisson de sosire a apelurilor şi timpi de servire a apelurilor independenți şi exponențial distribuiți [JSD, Fl]. Varianta bazată pe măsurare a acestui algoritm asigură ca, încărcarea instantanee măsurată plus rata de vârf a noului flux, să se afle sub regiunea de acceptanță. Încărcarea măsurată nu este ajustată artificial la admiterea unui nou flux. Pentru un flux descris de un filtru de tip găleată cu jetoane (r, b) (unde r este rata de sosire a jetoanelor şi b este capacitatea găleții) şi nu de rata lor de vârf, se poate deduce rata lor de vârf ($\hat{p}$) din parametrii găleții cu jetoane, folosind ecuația:

$$\hat{p} = r + b \cdot T \tag{2.6}$$

unde T este perioada de mediere definită de utilizator, în secunde. Intervalul T de interes nu este în mod necesar un parametru global, dar poate fi specificat

BUPT

pentru un anumit ruter. Ecuația (2.6) implică faptul că pentru procedurile de control admisiei bazate pe rata de vârf a fluxurilor, o cerere de admisie să fie acceptată cu o probabilitate mai mare, dacă parametrii găleții cu jetoane sunt asigurate astfel: b la o valoare mai mică, iar r la rata de vârf a conexiunii. Dimensiunea redusă a găleții introduce variații mici ale întârzierii pachetelor, întârzieri care se cumulează pe parcursul rețelei. Trebuie menționat însă că procedura de control a admisiei nu pretinde obligatoriu o dimensiune redusă a găleții cu jetoane, deoarece limita superioară a ratei de vârf pe un interval T poate fi calculată pentru orice valoare a parametrilor găleții.

Dacă un flux este respins, algoritmul de control a admisiei nu va admite un alt flux până când unul din fluxurile existente nu părăseşte rețeaua. Termenii de utilizare dorită şi pragul utilizării se folosesc cu acelaşi sens cu regiunea de acceptanță. Eşantioanele se iau la perioadele de eşantionare S'.

2.2.4 Algoritmul capacității echivalente

Prin capacitate echivalentă, a unei clase de trafic compus, se înțelege valoarea $C(\varepsilon)$, determinată astfel încât rata sosirilor clasei în regim staționar să nu poată depăşi $C(\varepsilon)$, decât cel mult cu probabilitatea ε [Fl, JSD]. Un flux nou (sau o conexiune) dintr-o clasă va fi admis doar dacă prin acceptarea sa, capacitatea echivalentă a clasei va fi mai mică decât lățimea de bandă alocată clasei respective.

Procedura de control a admisiei descrisă în cele ce urmează presupune că parametrii, specificând traficul noului flux, includ parametrii găleții cu jetoane, rata permiselor şi dimensionarea găleții, şi că aceşti parametrii sunt administrați de ruter. Traficul noilor fluxuri, care depăşesc aceşti parametri ai găleții cu jetoane este transmis înainte conform cu principiul efortului maxim, dar nu e luat în considerare de procedura de control a admisiei. Parametrii găleții cu jetoane, pot fi folosiți (vezi ec. 2.6.) pentru a deduce rata de vârf a fiecărui flux, într-o perioadă de timp fixă; probabilitatea ca un flux nou să fie admis creşte dacă rata permiselor e apropiată de rata de vârf a fluxului. Dimensiunea găleții e de dorit să fie suficient de mică pentru scăderea întârzierii pachetelor ce se transmit în rețea.

Estimările se fac pe baza măsurărilor ratei medii a tuturor sosirilor şi a ratelor de vârf a fluxurilor individuale. Rata sosirilor într-o clasă poate fi modelată ca o variabilă aleatoare cu distribuție normală [Fl]; această abordare este cea mai potrivită pentru multe clase cu trafic de timp real, care are sute de conexiuni acceptate, cu rate de vârf asemănătoare. Dacă însă numărul de conexiuni e redus sau mediu, respectiv ratele de vârf ale diferitelor fluxuri variază foarte mult, distribuția normală nu mai este potrivită pentru calcul.

În cele ce urmează vom folosi notațiile:

- $r_{i,T}$, rata instantanee de sosire, la momentul T, a fluxului/conexiunii i. Dacă $i \neq j$, $r_{i,T}$ şi $r_{j,T}$ sunt independente. Nu e necesară ipoteza ca $r_{i,T}$ şi $r_{i,T-\delta}$ să fie

BUPT

independente;

- $S_T = \sum_{i=1}^{n} r_{i,T}$, rata instantanee a sosirilor la momentul T, a fluxurilor agregate ale clasei respective;
- $\hat{\mu}_s$, rata medie estimată a sosirilor, obținută prin măsurarea ratei sosirilor a tuturor fluxurilor (fluxurilor agregate) ale clasei respective. Rata medie a sosirilor poate fi aproximată prin încărcarea medie măsurată [JSD].

2.2.4.1 Estimarea capacității echivalente folosind distribuția normală

Ipoteza că S_T, rata instantanee a sosirilor tuturor fluxurilor, sau fluxuri agregate, poate fi modelată printr-o variabilă cu distribuție normală, e valabilă doar pentru clase de dimensiuni mari. Valoarea medie μ_s, respectiv dispersia σ^2 a ratei S a sosirilor clasei, se pot deduce din parametrii galeților cu jetoane a conexiunilor individuale, sau se pot estima din măsurări. Capacitatea echivalentă $\hat{C}_N$, folosind distribuția normală, e dată de [Fl]:

$$\hat{C}_N(\mu_S, \sigma^2, \varepsilon) = \mu_S + \alpha\sigma \tag{2.7}$$

pentru $\alpha = \sqrt{2ln\frac{1}{\varepsilon} + ln\frac{1}{2\pi}}$.

Astfel dacă rata S a sosirilor are o distribuție normală, atunci la un moment de timp viitor t, rata S_t a sosirilor va putea depăşi capacitatea $\hat{C}_N$ cu probabilitatea maximă ε. Pentru controlul admisiei pe baza capacității echivalente, parametrul ε reprezintă o limită superioară estimată a probabilității cu care ratele instantanee ale sosirilor pot depăşi debitul alocat clasei. Parametrul ε a fost denumit rata de pierderi. Totuşi în mediile în care o mare parte a traficului este traficul de efort maxim, traficul de timp real a cărui rată depăşeşte capacitatea sa echivalentă nu e pierdut, dar deranjează traficul de efort maxim. Cu cât parametrul ε al unei clase e mai mic, cu atât algoritmul de control a admisiei este mai conservativ pentru acea clasă [EM].

Cererea de admitere a unui nou flux va fi acceptată dacă rata de vârf a noului flux însumată capacității echivalente este mai mică decât capacitatea echivalentă alocată clasei respective.

După cum s-a mai afirmat, deoarece formula pentru $\hat{C}_N(\mu_S, \sigma^2, \varepsilon)$ presupune o distribuție normală pentru ratele sosirilor dintr-o clasă, această abordare nu este adecvată pentru clase de dimensiuni mici sau medii. Pentru clase mici, estimarea capacității echivalente folosind distribuția normală, poate subestima capacitatea echivalentă actuală [Fl]. În continuare, va fi prezentată o altă formulă pentru capacitatea echivalentă, folosind limita Hoeffding, care poate fi folosită şi în cazul claselor de dimensiune medie.

BUPT

2.2.4.2 Estimarea capacității echivalente cu limita Hoeffding

Pentru o conexiune admisă i, cu rata instantanee de sosire r_i, pe un interval de timp dat, poate fi dedusă din parametrii găleții cu jetoane (vezi ec. 2.6). Se poate deduce o limită superioară pentru capacitatea echivalentă $\hat{C}_H$, din limitele superioare ale ratelor de vârf ale fiecărei conexiuni admise, din rata măsurată a sosirilor agregate şi folosind rezultatele lui Hoeffding, [Fl, DJM].

Teorema lui Hoeffding afirmă că, având variabilele independente $X_1, X_2, ..., X_n$ şi $0 \le x_i \le p_i$, atunci, pentru $t > 0$,

$$Prob\left[S \ge \mu_S + nt\right] \le e^{-2n^2t^2 / \sum_{i=1}^{n} (p_i)^2} \tag{2.8}$$

Această teoremă este asemănătoare cu cele de tip Cernov, [EM, Fl, JDSZ, DJM], unde pentru o sumă de variabile aleatoare se determină o limită superioară a distribuției. Ecuația (2.8) arată probabilitatea ca suma a n variabile aleatoare să fie mai mare decât media sumei plus nt.

Astfel, pentru cazul nostru:

$$Pr\left[S_T \ge \mu_S + nt\right] \le \varepsilon, \tag{2.9}$$

e satisfăcută dacă:

$$e^{-2n^2t^2 \sum_{i=1}^{n} (p_i)^2} \le \varepsilon, \tag{2.10}$$

unde prin alegerea parametrului ε se determină gradul de risc al procedurii de control al admisiei. Condiția (2.10) este satisfăcută dacă:

$$t \le \frac{1}{n} \sqrt{\frac{ln(1/\varepsilon) \sum_{i=1}^{n} (p_i)^2}{2}}. \tag{2.11}$$

Astfel rezultă capacitatea echivalentă $\hat{C}_H$, obținută prin urmărirea ratelor de vârf:

$$\hat{C}_H + (\mu_S, \{p_i\}_{1,\dots,n}, \varepsilon) = \mu_S + \sqrt{\frac{ln(1/\varepsilon) \sum_{i=1}^{n} (p_i)^2}{2}}, \tag{2.12}$$

unde, din nou, μ_s e rata medie de sosire a traficului din clasa respectivă şi ratele de vârf ale fluxurilor clasei, iar ε este probabilitatea ca rata de intrare să depăşească capacitatea liniei. Algoritmul de control a admisiei verifică dacă un nou flux, $n+1$, cerând admisia, respectă condiția:

$$\hat{C}_{H} + p_{n+1} \le C, \tag{2.13}$$

BUPT

unde C este capacitatea sau lățimea de bandă alocată clasei respective. După ce un flux este admis, rata sa de vârf p_{n+1} este adăugată la $\hat{\mu}_S$, rata medie estimată a clasei. Dacă estimarea ratei de sosire a fluxului compus se face folosind o mediere exponențială, atunci noua estimare va reflecta progresiv noul trafic şi nu valoarea de vârf p_{n+1} .

Dacă fluxului nou i se interzice admisia, atunci nici unui alt flux de acelaşi tip nu i se mai permite intrarea, până când un alt flux nu iese.

Capacitatea echivalentă estimată $\hat{C}_H$ este determinată în ipoteza că ratele de sosire ale diferitelor fluxuri sunt independente, că procedura de control cunoaşte μ_S, media ratelor sosirilor agregate şi că aceasta este limitată superior.

2.2.4.3 Măsurarea ratei medii de sosire

O problemă importantă a procedurilor de control a admisiei bazate pe măsurări este modul de estimare a ratei medii de sosire pentru clasa respectivă în locul lățimii de bandă instantanee, pentru a putea lua decizia de admisie a noului flux.

Estimarea ratei medii de sosire prin măsurări prezintă unele dificultăți. Astfel chiar având un număr fix de conexiuni admise, ipoteza existenței mediei "stării staționare" nu e întotdeauna exactă. Apoi, chiar dacă aceasta există, e dificil de determinat rata medie a sosirii prin măsurări, din cauza posibilelor dependențe de gamă largă, LRD Long-Range Dependence, a traficului compus [ENW, TWW]. Astfel rata medie de pe interval ar putea să nu fie un predicator bun pentru intervalul care va urma şi deci s-ar putea să nu fie folosită întreaga capacitate a liniei de transmisii, utilizarea integrală care reprezintă chiar scopul acestor proceduri. Dar cum serviciul de timp real are loc cu interfețele unde o mare parte din trafic este de efort maxim a adaptat la rată, procedura de control a admisiei va accepta o valoare medie mai puțin precisă.

O procedură simplă de estimare a ratei medii de sosire constă din măsurarea ratei de sosire r_i la fiecare s secunde. Rata medie va putea fi apoi calculată folosind o mediere exponențială cu ponderea w:

$$med = (1 - w) \cdot med + w \cdot r_i \tag{2.14}$$

Constanta de timp pentru aceasta este:

$$T = -1/ln\,(1 - w) \cdot S, \quad \text{secunde} \tag{2.15}$$

De aici, pentru

$$-1/ln\,(1 - w) \cdot S \le T, \tag{2.16}$$

rezultă ponderea

$$w \ge 1 - e^{-S/T}. \tag{2.17}$$

BUPT

Media ratei de sosire este calculată folosind relația (2.14) astfel:

$$\hat{\mu}' = (1 - w)\hat{\mu} + w\,\hat{\mu}_S \qquad (2.18)$$

unde $\hat{\mu}_S$ rata medie de sosire este măsurată la fiecare S secundă.

Dacă traficul variază brusc de la 0 la 1 şi rămâne la 1, cu o pondere de $w = 2e\text{-}3$, în 10 perioade se ajunge la 75% din valoarea 1 a noii rate.

O valoare mare a ponderii face procesul de mediere mai adaptiv la schimbările sarcinii, iar o valoare mică conduce la medii mai netede care ţin cont de o istorie mai lungă.

În concluzie, folosind marginea Hoeffding este necesar să se urmărească ratele de vârf ale conexiunilor admise precum şi rata agregată de sosire. Dacă se foloseşte marginea Cernov, [GK, ABB] se obţine o procedură de control a admisiei bazată pe măsurarea ratei agregate de sosire şi a unui parametru p de rafală pentru toate conexiunile admise, unde p este probabilitatea ca un flux să fie activ la un moment dat [Fl]. Se dezvoltă o procedură de control a admisiei bazată tot pe măsurări, folosind teoria deciziei, care nu necesită cunoştinţe mportarea asimptotică a CLR când dimensiunea sistemului e foarte mare, de exemplu tampoane infinite sau capacitate suficient de mare de servire.

Fie p(m), m = 1,2,..., CLR rata de pierderi a celulelor, pentru dimensiunea Sm a sistemu flux, abordarea prin teoria deciziei foloseşte "recompense" pentru pachetele livrate cu succes, respectiv "penalizări" pentru fiecare pachet distrus. Rezultatele acestei proceduri de admisie sunt încurajatoare, dar în prezent se cunosc relativ puţine lucruri despre cum se va mixa traficul viitoarelor servicii de tip real ale Internet.

2.2.5 Controlul probabilistic al admisiei

În [MS] se propune un control probabilistic al admisiei. În funcţie de capacitatea legăturii disponibilă la momentul sosirii apelului, acesta va fi acceptat (dar nu şi conectat încă) cu o anumită probabilitate. Dacă apelul nu e acceptat atunci e blocat şi şters. Dacă apelul este acceptat în faza de control a admisiei, atunci el e trecut procedurii de dirijare. Noţiunea de unitate a lăţimii de bandă de bază BBU (Basic Bandwith Unit) se referă la rata sau debitul de bază şi se presupune că toate serviciile lucrează cu multiplii ai acestei rate de bază. De exemplu dacă considerăm BBU ca fiind de 64 Kbps, atunci serviciul de transmitere a vocii având rata de vârf de 64 Kbps necesită o BBU, iar serviciul de transmitere a imaginilor video cu rata de vârf de 384 Kbps necesită 6 BBU.

Controlul admisiei este dat pentru un serviciu $s \in S$ al traficului pereche $[i,j]$, prin următoarea funcţie de acceptanţă:

$$\alpha^s_{[i,j]} = p^s_{[i,j]} \text{ dacă } L_{[i,j]} = t_{(i,j)} - 0_{(i,j)} < U_{(i,j)}$$
$$\text{sau } \alpha^s_{[i,j]} = 0 \text{ în caz contrar} \qquad (2.19)$$

BUPT

unde $L_{[i,j]}$ şi $U_{(i,j)}$ sunt marginile inferioară respectiv superioară a unității de lățime de bandă de bază BBU, iar $0 \le p^{s}_{[i,j]} \le 1$. Dacă $p^{s}_{[i,j]} = 1$, $s \in S$ pentru orice trafic pereche, atunci nu există control al admisiei, situație care este denumită şi partajare completă. Controlul admisiei este local, nu este bazat pe informații despre întreaga rețea, ci depinde doar de gradul de ocupare al legăturii directe la momentul sosirii apelului.

2.2.6 Controlul admisiei folosind rata de pierderi a celulelor

Rata de pierdere a celulelor CLR (Cell Loss Ratio) este un parametru important al proiectării şi controlului rețelelor ATM, fiind folosit în multe din funcțiile principale ale rețelelor ca de exemplu controlul admisiei apelurilor CAC (Call Admission Control), alocarea benzii şi altele. Pentru a lua decizia admiterii sau respingerii unui apel pe baza CLR, este necesar ca CLR să fie calculat rapid, fie pe baza capacității legăturii fie pe baza capacității serviciului pretinse de respectivul apel, pentru a obține o valoare anumită a CLR. Determinarea capacității serviciului în vederea obținerii unei anumite CLR este însă foarte dificil de făcut, din cauza interacțiunii celor doi factori, CLR şi capacitatea serviciului, chiar având specificate caracteristicile traficului şi dimensiunea bufferelor [BLHT, BLCT, CT].

Majoritatea algoritmilor prezentați în literatură [KWC, KA, LM, SSD, ABB, JSD, IDSZ, PAB, MS] se bazează pe conceptul de lățime de bandă efectivă, calculul capacității de serviciu necesară fiind făcut prin aproximarea cu o margine superioară. Conceptul de lățime de bandă efectivă este atractiv din cauza proprietății sale că lățimea de bandă efectivă a unui set de surse este suma lățimilor de bandă individuale. Acest lucru reduce complexitatea calculului. Dar, lucrând în ipoteza tamponului infinit, nu e luat în considerare multiplexarea statistică a surselor care partajează tamponului şi uneori, când încărcarea e scăzută, apar supraestimări exagerate ale capacității de serviciu necesare. Când numărul de surse e mare aproximarea gaussiană e de asemenea propusă, pentru a beneficia de avantajul multiplexării statistice. Totuşi şi aceasta conduce la supraestimări ale lățimii de bandă agregată necesară, ceea ce duce la subutilizarea resurselor scumpe ale rețelei: lățimea de bandă şi memoria. În plus, dacă traficul e mai puțin în rafală decât Poisson, ca de exemplu la ieşirea filtrelor de tip găleată, aproximarea cu lățimea de bandă efectivă nu mai este conservativă şi duce la o subestimare a capacității de serviciu necesare. Aproximările mai noi, bazate pe estimările deviațiilor mari ale distribuțiilor lungimilor cozilor de aşteptare din multiplexoare, conduc la o estimare mai precisă a marginii superioare a CLR decât cea bazată pe conceptul de bandă efectivă [PAB, JSD, FF]. Toate aceste abordări necesită însă cunoştințe complete despre procesul de sosire în multiplexor.

În cele ce urmează se va prezenta un algoritm de control al admisiei bazat pe măsurări, care foloseşte un algoritm de aproximare fuzzy FA (Fuzzy Approximation) pentru estimarea benzii agregate necesare. Mecanismul este

BUPT

destinat în principal serviciului de rată variabilă VBR (Variable Bit Rate) şi serviciilor cu alocare statică a benzii (adică acolo unde nu e permisă renegocierea lăţimii de bandă pe durata apelului) dar poate fi extins şi pentru serviciul RCBR (Renegociation Constant Bit Rate).

2.2.6.1 Algoritmul CAC-FA-CLR

Algoritmul de control a admisiei apelurilor bazat pe logica fuzzy pentru estimarea ratei de pierdere a celulelor, conduce la următoarea structură, figura 2.1 [BLHT, CFPP, CC].

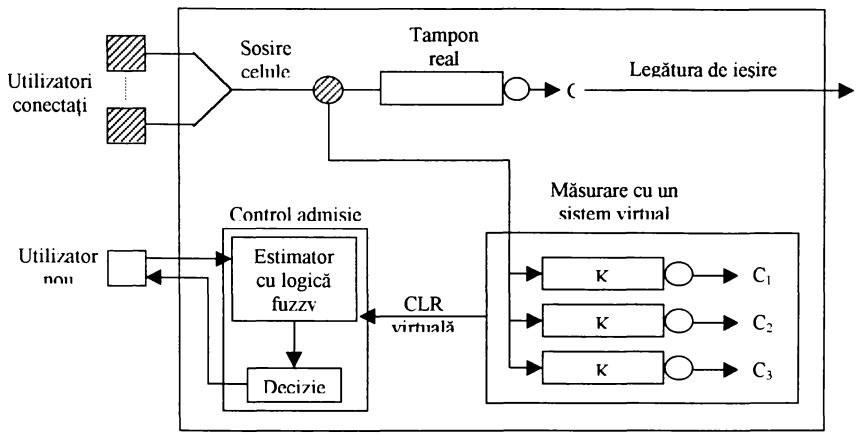


Figura 2.3 Mecanism de control a admisiei cu logică fuzzy

Mecanismul are două părţi: prima parte constă dintr-un set de tampoane virtuale (contoare), cu o capacitate de serviciu redusă cu care se observă pierderile foarte mari de celule, cu o variaţie mică într-un interval scurt de măsurare. În funcţie de configuraţia hardware a multiplexorului, traficul poate fi integrat într-un tampon comun, partajat sau segregat în mai multe tampoane partajând aceeaşi linie, caz în care sunt necesare atâtea contoare câte tampoane sunt. A doua parte a mecanismului constă din algoritmul FA şi procesul de decizie. Algoritmul FA e folosit pentru a determina lăţimea de bandă necesară pentru apelurile care intră, după care procesul de decizie hotărăşte dacă un apel e acceptat sau nu, în funcţie de ieşirea dată de algoritmul FA, lăţimea de bandă disponibilă şi lăţimea de bandă solicitată de noul apel.

2.2.6.2. Descrierea sistemului bazat pe reguli fuzzy

Sistemele fuzzy se bazează pe reguli matematice severe. Ele manevrează un gen de informaţie incompletă, vagă sau fuzzy, dificil de formulat matematic,

BUPT

inima lor fiind un sistem bazat pe reguli sau pe cunoaştere, constând dintr-o serie de reguli de tipul "dacă – atunci" (if–then).

Schema bloc a unui sistem fuzzy este prezentată în figura 2.4. El este format dintr-un fuzificator, un defuzificator şi un nucleu de inferenţă, precum şi o bază de reguli. În [BLHT] s-a optat pentru un fuzificator de tip "unic" (Singleton), un defuzificator de tip "medie centrată" (center average) şi un nucleu de inferenţă de tip "produs".

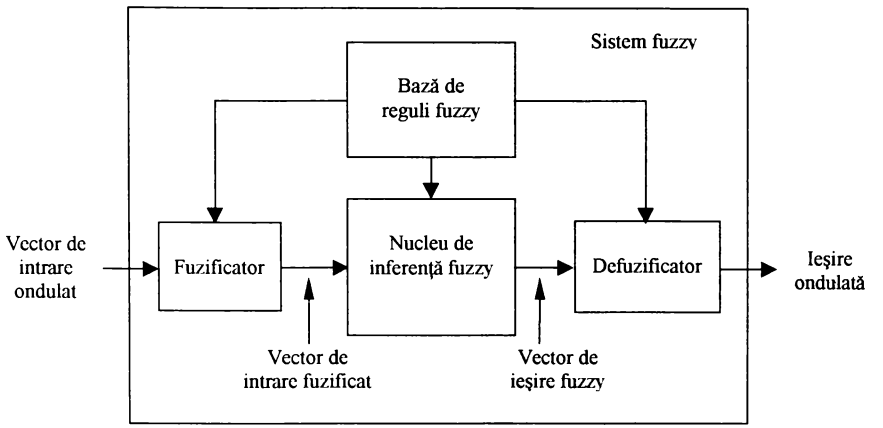


Figura 2.4 Schema bloc a unui sistem fuzzy

Se caută modelarea comportării cozii de aşteptare a unui multiplicator ATM, ale cărui performanţe depind de traficul de intrare, care nu este predictibil.

Presupunem cunoscute N seturi de perechi de intrare-ieşire $(X_0^j; y_0^j)$, $X_0^j = (x_{01}^j, ..., x_{0n}^j)^T \in R^n$, N fiind un număr mic. Se arată că sistemul fuzzy, pentru fuzificatorul de tip unic, nucleul de inferenţă de tip produs şi defuzificator de tip medie centrată se reduce la o formă compactă:

$$f(X) = \frac{\sum_{j=1}^{N} y_0^j \prod_{i=1}^{n} \mu_i^j(x_i)}{\sum_{j=1}^{N} \prod_{i=1}^{n} \mu_i^j(x_i)} \qquad (2.20)$$

unde $X = (x_1, ..., x_n)^T \in R^n$ este intrarea (ondulată) a sistemului fuzzy,
$f(X) \in R$ este ieşirea sistemului fuzzy,
μ_i^j sunt funcţiile de apartenenţă ale sistemului fuzzy de intrare.

Dacă în plus, funcţiile de apartenenţă sunt gaussiene, adică:

BUPT

$$\mu_i^j(x_i) = exp\left\{-\left(x_i - x_{0i}^j\right)^2 / \left(\sigma_i^j\right)^2\right\}, \tag{2.21}$$

$$j = 1,\ldots,N, \quad i = 1,\ldots,n,$$

atunci (2.20) are capacitatea de aproximare universală. Aceasta înseamnă că pentru orice funcţie reală continuă $g(x)$ şi un $\varepsilon > 0$ ales arbitrar, există un sistem fuzzy astfel încât

$$sup_{x \in R^n} |f(X) - g(X)| < \varepsilon. \tag{2.22}$$

Dacă se foloseşte (2.20) pentru interpolarea funcţiilor, cu cât valorile lui σ_i^j sunt mai mici, cu atât aproximarea este mai bună. Pentru extrapolarea funcţiilor, care este scopul nostru, eroarea de aproximare nu poate fi teoretic controlată. În acest caz valorile lui σ_i^j vor fi astfel alese încât funcţiile de apartenenţă corespunzătoare să acopere uniform toate cele *n* componente ale fiecărei intrări *j*, pentru a asigura o influenţă suficient de semnificativă a fiecăreia din cele *n* componente.

Căutăm să estimăm CLR în multiplexoarele ATM, cu ajutorul unui sistem fuzzy ca cel de mai sus. CLR este o funcţie necunoscută de variabilă reală, variabilă care poate fi dimensiunea tamponului multiplexorului, numărul de conexiuni sau capacitatea de serviciu. Algoritmul nu are nevoie să facă vreo presupunere despre parametrii traficului, dar trebuie să cunoască:

- CLR al multiplexorului când dimensiunea sistemului e redusă, de exemplu tampoane mici sau capacitate redusă de serviciu. În acest caz CLR este relativ mare şi poate fi uşor calculat sau măsurat,
- comportarea asimptotică a CLR când dimensiunea sistemului e foarte mare, de exemplu tampoane infinite sau capacitate suficient de mare de servire.

Fie $p(m)$, $m = 1,2,\ldots$, CLR rata de pierderi a celulelor, pentru dimensiunea S_m a sistemului. S_m poate fi dimensiunea tamponului, numărul de utilizatori sau capacitatea serviciului. Setul de valori (S_m) este o funcţie liniar crescătoare,

$$S_m = S_0 + m\delta, \tag{2.23}$$

unde S_0 este valoarea iniţială a dimensiunii S, $\delta > 0$ este dimensiunea pasului.

Cu aceste informaţii algoritmul FA se construieşte plecând de la cele *M* valori cunoscute ale CLR, $P(k-M+1)$, $P(k-M+2)$,..., $P(k)$, unde *k* este indicele stării curente S_k.

Pasul 1, definirea perechilor de intrare ieşire: cele $M-2$ perechi de intrare-ieşire sunt alese după cum urmează:

$$\begin{array}{c} \{[P(k-2), P(k-1)]\ ;\ P(k) - P(k-1)\} \\ \{[P(k-3), P(k-2)]\ ;\ P(k-1) - P(k-2)\} \\ \vdots \qquad\qquad\qquad \vdots \\ \{[P(k-M+1), P(k-M+2)]\ ;\ [P(k-M+3), P(k-M+2)];\ \} \end{array} \tag{2.24}$$

BUPT

unde: $[P(k-j-1), P(k-j)]$ este vectorul de intrare,
$[P(k-j+1)-P(k-j)]$ este vectorul de ieşire,
$j = 1,\ldots,M-2$.

Cum CLR este o funcție monotonă de variabilă aleasă (lungimea sistemului, capacitatea serviciului sau numărul de apeluri), sistemul fuzzy poate fi construit cu puține valori. Intrările reprezintă valorile CLR pentru starea precedentă şi starea curentă. Ieşirea este diferența dintre valoarea următoare şi cea curentă a CLR. Perechile de intrare ieşire sunt actualizate în timp real, când k e incrementat la valoarea dorită.

Pasul 2, funcțiile de apartenență. Se defineşte:

$$\Delta_{k,j} = P(k-1) - P(k-j-1). \tag{2.25}$$

Pentru $j = 1,\ldots,M-2$, funcțiile de apartenență ale seturilor de intrări fuzzy sunt definite:

$$\mu_1^j[P(k-1)] = exp\left\{-\frac{\Delta_{k,j}^2}{(\sigma_1^j)^2}\right\},$$
$$\mu_2^j[P(k)] = exp\left\{-\frac{\Delta_{k,-1}^2}{(\sigma_2^j)^2}\right\}, \tag{2.26}$$

cu:

$$\sigma_1^j = \frac{\max_{j'} P(k-j'-1) - \min_{j'} P(k-j'-1)}{M-2},$$
$$\sigma_2^j = \frac{\max_{j'} P(k-j') - \min_{j'} P(k-j')}{M-2}, \tag{2.27}$$

cu $j' = 1,\ldots,M-2$.

Funcțiile de apartenență definesc gradul de influență a primei, respectiv a celei dea doua componente a vectorului de intrare j, asupra valorii de ieşire. Parametrii σ_1^j şi σ_2^j, $j = 1,\ldots,M-2$ parametri liberi care determină precizia aproximării, s-au ales ca mai sus pentru ca funcțiile de apartenență să acopere uniform domeniul vectorilor de intrare. Altfel spus valorile precedente ale CLR influențează uniform următoarea valoare prezisă.

Pasul 3: construirea sistemului fuzzy: Sistemul s-a obținut [BLCT] pentru $n=2$ şi $N=M-2$. După cum s-a văzut în pasul 1, ieşirea sistemului fuzzy e definită ca incrementul (sau decrementul) lui CLR față de valoarea precedentă. La început, valorile variabilei de stare, pentru care CLR, e cunoscută sunt indexate de la 0 la $M-1$. Ecuația (2.20) este apoi folosită pentru estimarea CLR pentru următoarea variabilă de stare (de exemplu pentru S_M). După ce s-a obținut CLR, S_0 e şters şi se transmite înapoi sistemului valoarea estimată a noului punct. Altfel spus stările indexate cu $1, \ldots, M$ sunt folosite pentru

BUPT

construirea perechilor de intrare-ieşire necesare pentru definirea noului sistem fuzzy pentru estimarea CLR pentru S_{M+1}. Acest proces continuă până la obținerea CLR dorită, sau a CLR corespunzătoare dimensiunii dorite a sistemului. În general, algoritmul FA furnizează o predicție bună dacă numărul de paşi necesari pentru predicția CLR dorite nu e prea mare. Totuşi atingerea unei CLR de 10^{-10}, necesită uneori foarte mulți paşi în detrimentul preciziei, care scade din cauza însumării erorilor mici de predicție. Pentru îmbunătățirea estimării, se iau în considerare comportarea asimptotică a CLR la creşterea S_m a dimensiunii sistemului. Astfel dacă $\hat{P}(m)$ este valoarea estimată a CLR, iar $\tilde{P}(m)$ este valoarea obținută ținând cont de comportarea asimptotică a CLR funcție de dimensiunea S_M a sistemului, în loc de a transmite ca reacție algoritmului FA, mărimea obținută în pasul 3, se transmite:

$$P(m) = [1 - \lambda(m)]\hat{P}(m) + \lambda(m)\tilde{P}_2(m), \tag{2.28}$$

unde

$$\lambda(m) = \begin{cases} exp\left[-\dfrac{(S_m - S_\infty)^2}{\sigma^2}\right], & daca\ S_m \le S_\infty \\ 1, & in\ rest \end{cases}, \tag{2.29}$$

şi $\lambda \in [0,1]$, iar S_∞ este orice stare la care se ia în considerare comportarea asimptotică şi σ este un parametru controlând precizia aproximării. Funcția λ funcționează ca un filtru trece sus. Ea permite influența comportării asimptotice doar când dimensiunea S_m a sistemului se apropie de S_∞.

Alegerea lui S_∞ şi σ influențează precizia predicției. Experimental s-a constatat că pentru CLR funcție de lungimea tamponului, S_∞ trebuie ales de zece ori lungimea medie a rafalei la încărcarea redusă cu trafic, respectiv, este suficient, 1-2 ori lungimea medie a rafalei la încărcare mare cu trafic. Dacă CLR se estimează funcție de capacitate, S_∞ se alege egal cu rata de vârf agregată. Alegerea lui σ se face 60% din S_∞ când se lucrează cu lungimea tampoanelor, 48% din S_∞ când se lucrează cu număr de utilizatori, sau 28% din S_∞ când se lucrează cu capacitatea sistemului. Alegerea pasului δ a fost $0,1$ ori lungimea medie a rafalei, iar cu $M=5$, cinci valori inițiale, s-a lucrat doar cu trei perechi de intrare-ieşire.

În concluzie sistemul adaptiv bazat pe logica fuzzy, este folosit pentru a prezice eficient CLR în sisteme de dimensiuni mari, bazându-se pe un volum redus de informații despre sisteme. Aceste informații pot fi obținute fie prin măsurări în timp real când caracteristicile traficului nu sunt cunoscute, fie din orice alt model analitic când caracteristicile traficului sunt date. Valoare prezisă a CLR este apoi rafinată, cunoscând comportarea asimptotică a CLR. Avantajul metodei este simplitatea implementării, eficiența calculului şi precizia bună.

BUPT

Capitolul 3

CONTROLUL CONGESTIEI ŞI MANAGEMENTUL TRAFICULUI ÎN REŢELE ATM

3.1 Introducere

Reţelele de mare viteză din viitor vor folosi, foarte probabil modul de transfer asincron ATM, cu transmiterea de celule de dimensiune fixă. Acest lucru reduce mult din dispersia întârzierii şi permite integrarea în aceeaşi reţea a traficului de voce, video şi date. Pentru toate aceste tipuri de trafic va trebui asigurată o calitate a serviciilor, lucru mult mai dificil decât în reţelele actuale. Un management adecvat al traficului asigură o operare eficientă şi imparţială a reţelei, în condiţiile unor cereri variabile. Acest lucru este important mai ales pentru traficul de date care este foarte puţin predictibil şi deci nu permite rezervarea resurselor în avans, ca în cazul reţelelor de telecomunicaţii pentru voce.

Managementul traficului asigură utilizatorilor calitatea dorită a serviciilor. Problema este extrem de dificilă în cazul supraîncărcării reţelei sau în cazul când cererile traficului nu pot fi anticipate. De aceea controlul congestiei este partea cea mai importantă a managementului traficului [YL, J1, J2, KA, KMR, KR, KMR, EPD, LSY, EPDJ].

3.2 Parametrii traficului şi calitatea serviciului.

În momentul stabilirii conexiunii prin reţeaua ATM, utilizatorul trebuie să specifice următorii parametrii referitori la trafic şi calitatea serviciului [RB, Ne, J1].

3.2.1 Parametrii de trafic

1) Rata de vârf a celulelor, PCR (Peack Cell Rate)– rata maximă instantanee la care va transmite utilizatorul;
2) Rata medie a celulelor, SCR (Sustained Rate Cell) – rata de transmitere a celulelor mediată pe un interval larg de măsurare;
3) Rata de pierdere a celulelor, CLR (Cell Loss Ratio) – procentul celulelor pierdute în reţea din cauza erorilor şi a congestiei, care nu vor fi livrate la destinaţie:

BUPT

$$CLR = \frac{Numar\ de\ celule\ pierdute}{Numar\ de\ celule\ transmise} \tag{3.1}$$

Fiecare celulă ATM are în antet un bit CLP (Cell Loss Priority). Dacă apare congestia, primele distruse vor fi celulele cu CLP=1. Deoarece celulele cu CLP = 0 sunt mai importante pentru aplicație, se poate specifica separat CLR pentru celulele cu CLP = 0 şi pentru cele cu CLP = 1.

4) Întârzierea de transfer a celulelor, CTD (Cell Transfer Delay) – întârzierea suferită de celulă între punctele de intrare şi de ieşire din rețea. CTD include întârzierea de propagare, întârzierea în cozile nodurilor şi timpii de servire a celulei în fiecare coadă.
5) Variația întârzierii celulelor, CDV (Cell Delay Variation) este o măsură a variației întârzierii CDT. O variație mare este generată de tampoane mari de memorie şi este importantă pentru traficul sensibil la întârzieri ca vocea sau video.
6) Variația admisibilă a întârzierii celulelor sau toleranța acesteia CDVT (Cell Delay Variation Tolerance) şi toleranța rafalei BT (Burst Tolerance): pentru orice sursă transmițând la o rată oarecare este admisă o uşoară variație a intervalului între celule.

De exemplu, o sursă care transmite la o rată de vârf PCR = 10000 celule/sec va transmite teoretic o celulă la fiecare 100 μsec. Folosirea unei găleți găurite va asigura ca variația intervalelor între celule să fie acceptabilă. Acest algoritm are doi parametri. Primul este intervalul nominal dintre celule (inversul ratei) şi al doilea este variația permisă a acestui interval. Astfel o găleată găurită cu parametrii (100 μs, 10 μs) nu va permite celulelor o abatere mai mare de 10 μs de la intervalul nominal de 100 μsec. În acest exemplu CDV = 100 μs şi CDVT = 10 μs. Astfel, al doilea parametru al găleții, CDVT este folosit pentru a forța PCR. Pentru a forța SCR este folosit BT.

7) Dimensiunea maximă a rafalei MBS (Maximum Burst Size) – este numărul maxim de celule înlănțuite care pot fi transmise la rata de vârf fără a afecta SCR.

$$BT = (MBS - 1)\left(\frac{1}{SCR} - \frac{1}{PCR}\right)$$

Deoarece MBS e mai intuitiv decât BT , în semnalizare se foloseşte MBS, iar BT se calculează ca mai sus.

Se observă că PCR, SCR, CDVT, BT şi MBS sunt parametri ai traficului de intrare şi sunt dictați de rețea la intrarea sa, iar CLR, CTD şi CDV sunt indicatori ai calității serviciului furnizat de rețea şi se măsoară la punctul de ieşire din rețea.

8) Rata minimă a celulelor MCR (Minimum Cell Rate) este rata minimă dorită de utilizator.

BUPT

Aceşti parametri au fost specificaţi în interfaţa utilizator reţea UNI (User Network Interface) a reţelelor ATM [GaP].

3.2.2 Categorii de servicii

1) Serviciul cu rata de bit constantă CBR (Constant Bit Rate), este folosit pentru emularea circuitelor comutate. Rata celulelor este constantă. Pentru celulele cu CLP = 0 se specifică rata de pierderi CLR, iar pentru celulele cu CLP = 1 această rată poate să fie specificată sau nu. Se aplică pentru serviciul telefonic, videoconferinţe şi televiziune.
2) Serviciul cu rata de bit variabilă VBR (Variable Bit Rate) permite utilizatorilor să transmită la rate variabile. Se foloseşte multiplexarea statistică şi deci pot are pierderi foarte mici, aleatoare. În funcţie de aplicaţie, sensibilă sau nu la întârziere, această categorie se împarte în două subcategorii: VBR de timp real şi VBR nu de timp real. Pentru prima categorie se specifică întârzierea maximă şi variaţia vârf la vârf a întârzierii CDV, iar pentru a doua categorie se specifică doar întârzierea. Traficul video interactiv comprimat e un exemplu de trafic VBR de timp real, iar poşta electronică multimedia este un exemplu de trafic VBR nu de timp real.
3) Serviciul cu rata de bit disponibilă ABR (Available Bit Rate) este proiectat pentru traficul normal de date, ca transferul de fişiere sau poşta electronică. Deşi standardul nu pretinde garantarea sau minimizarea CTD şi CLR e mai bine ca nodurile să încerce minimizarea întârzierilor şi a pierderilor. În cazul congestiei, reţelei, sursei i se poate cere să-şi controleze rate de transmisie. În acest caz, reţeaua trebuie să garanteze MCR, rata minimă declarată de sursă. Majoritatea circuitelor virtuale vor solicita MCR = 0. Dacă nu există suficientă lăţime de bandă vor fi rejectate conexiunile cu cel mai mare MCR.
4) Serviciul cu rata de bit nespecificată UBR (Unspecified Bit Rate) a fost proiectat pentru aplicaţiile insensibile la pierderi sau întârzieri, capabile să folosească orice capacitate în exces rămasă. Aceste conexiuni nu sunt rejectate din cauza crizei de lăţime de bandă, pentru că nu se face controlul admisiei pe conexiune, şi nici nu li se controlează comportarea. În timpul congestiei, celulele sunt pierdute, dar nu se aşteaptă ca sursele să-şi reducă rata. Dar aceste aplicaţii pot avea un mecanism propriu de nivel înalt, pentru retransmisii şi recuperarea în cazul pierderii de celulele. Ca exemple de folosire a acestui serviciu sunt poşta electronică, transferul de fişiere, ştirile, etc., care evident pot folosi şi serviciul ABR.

Ceea ce trebuie menţionat este faptul că doar serviciul ABR răspunde la reacţia care soseşte din partea reţelei în caz de congestie, deci toate metode de control a congestiei se referă la acest tip de trafic.

BUPT

3.3 Metode de control a congestiei

Un management adecvat al traficului asigură o operare eficientă a rețelei, chiar în condițiile unor cereri variabile [J1, J2, JK, KA, KMR, KR]. Managementul traficului are ca scop să asigure utilizatorilor, de către rețea, servicii de calitatea solicitată. Problema e deosebit de dificilă în perioadele de supraîncărcare a rețelei, mai ales dacă cererile nu pot fi estimate în avans. De aceea controlul congestiei este partea cea mai importantă a managementului congestiei.

Congestia apare când suma ratelor de intrare e mai mare decât capacitatea disponibilă a liniilor de ieşire.

$$\sum \textit{ratelor de intrare} > \textit{capacitatea disponibila a liniilor de iesire} \qquad (3.2)$$

Majoritatea schemelor de control procedează la ajustarea ratelor de intrare la rata de ieşire disponibilă. O metodă de clasificare a schemelor de control a congestiei este după nivelul OSI – ISO la care se aplică ele (vezi 1.6). De obicei se foloseşte o combinație între aceste scheme, în funcție de durata şi gradul congestiei.

Pentru congestii sporadice, o metodă este dirijarea în funcție de gradul de încărcare a legăturii şi respingerea noilor conexiuni atunci când toate legăturile sunt foarte încărcate, aplicându-se un algoritm de control a admisiei. Aceşti algoritmi sunt eficienți pentru congestii de durată medie, deoarece odată ce un apel a fost admis congestia se poate menține pe toată durata apelului. Pentru congestii mai scurte decât durata unei conexiuni poate fi folosită o schemă cap la cap. De exemplu, în timpul stabilirii apelului se pot negocia rata de vârf şi rata în regim staționar. Apoi poate fi folosită o găleată găurită, de sursă sau de rețea pentru a asigura ca intrarea să respecte parametrii negociați.

Astfel se poate spune că algoritmii de formare sau nivelare a traficului sunt în buclă deschisă, în sensul că parametrii nu pot fi modificați dinamic dacă se detectează congestia după negociere. La schemele în buclă închisă însă sursele sunt informate dinamic despre starea congestiei rețelei şi li se cere să-şi modifice rata de intrare în rețea. Reacția poate fi cap la cap, la nivelul transport, sau nod la nod, la nivelul legătură de date. Reacția nod – la – nod este eficientă în cazul suprasarcinilor de durată mică. Pentru vârfuri de trafic foarte scurte, cea mai bună soluție este asigurarea unei memorii suficiente.

Deoarece soluțiile pentru congestia pe termen scurt şi cea pe termen lung sunt diferite şi în rețele apar în general toate tipurile de congestie se folosesc de obicei scheme hibride. UNI permite controlul admisiei conexiunii, formarea traficului şi reacția binară. Algoritmii pentru controlul admisiei conexiunii nu sunt specificați. Algoritmii de control ai admisiei au fost prezentați în capitolul 2, iar facilitățile oferite reacției în rețele ATM sunt prezentate în cele ce urmează.

BUPT

3.4 Facilități pentru reacție în rețele ATM

Interfața utilizator-rețea UNI din rețelele ATM oferă două facilități pentru controlul cu reacție, şi anume: [J2, LLD, DK, RB]

1) Control general al fluxului GFC (Generalized Flow Control).
2) Indicație explicită despre congestie, transmisă înainte EFCI (Explicit Forward Congestion Indication)

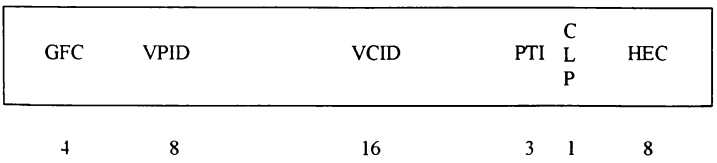


Figura 3.1 Antetul celulelor ATM

GFC – Generalized Flow Control – Control general al fluxului,
VPI – Virtual Path Identifier – Indentificatorul căii virtuale,
VCI – Virtual Channel Identification – Identificarea canalului virtual,
PTI – Payload Type – Tipul încărcăturii utile,
CLP – Cell Loss Priority – Prioritate la eliminarea celulei,
HEC – Header Error Check – Suma de control a antetului,

După cum se vede din figura 3.1 primii 4 biți ai antetului au fost rezervați pentru GFC, controlul fluxului. La interfața UNI nu se face acest control şi GFC este folosit ca o extensie a câmpului VPI. S-a intenționat ca GFC să fie folosit de rețea pentru controlul fluxului sursei, urmând ca algoritmul GFC să fie proiectat ulterior, lucru la care s-a renunțat însă.

3.5 Criterii de selecție

În cadrul grupului de lucru pentru managementul congestiei ale Forum-ului ATM au fost propuşi o serie de algoritmi de control a congestiei. Analiza şi decizia asupra lor s-a făcut conform unor criterii după cum urmează: scalabilitate, optimalitate, imparțialitate, robustețe şi posibilitatea de implementare.

3.5.1 Scalabilitatea

Rețelele se clasifică în general după numărul de noduri, numărul de utilizatori sau viteză. Trebuie ca orice schemă de control să fie independentă de

BUPT

viteză, distanță, număr de comutatoare sau număr de circuite virtuale, deci să poată fi aplicată aceeaşi schemă atât pentru LAN-uri cât şi pentru WAN-uri.

3.5.2 Optimalitatea

Într-un mediu partajat, rata alocată unei surse depinde nu numai de capacitatea sa ci şi de cererile altor surse. Cea mai utilizată metodă pentru o alocare imparțială a lățimii de bandă este "alocarea max-min". Considerăm o configurație cu n surse şi fiecare sursă i are alocată o lățime de bandă x_i rezultă vectorul de alocare $\{x_1, x_2 \ldots, x_n\}$, fezabil dacă gradul de încărcare al fiecărei linii este mai mic sau egal cu 100%. Numărul de vectori fezabili posibil este infinit. Pentru fiecare vector de alocare există o sursă cu alocarea cea mai mică, sau cea mai "nefericită" sursă. Având setul de vectori fezabili trebuie găsit vectorul care maximizează alocarea celei mai nefericite surse. De fapt, rezultă de asemenea un număr infinit de vectori.

Se procedează în consecință astfel: se extrage această sursă nefericită şi problema se reduce la $n-1$ surse care operează într-o rețea cu capacități reduse ale liniilor. Se caută din nou cea mai nefericită sursă dintre cele $n-1$ surse, i se dă acesteia alocarea maximă, se extrage apoi această sursă, etc., până se ajunge la o singură sursă. Procesul se repetă până ce tuturor surselor li s-a acordat alocarea maximă pe care o pot obține.

Alocarea min-max este eficientă şi imparțială [J1]. Este eficientă în sensul că fiecare linie este utilizată la încărcarea maximă posibilă şi imparțială în sensul că alocă părți egale fiecărei surse din lățimea de bandă a fiecărei linii.

Criteriul de alocare imparțială max-min e doar unul din criteriile de optimalitate care pot fi folosite, şi el nu are în vedere garantarea MCR. Un alt criteriu propus este imparțialitatea ponderată pentru determinarea alocării optimale a resurselor.

3.5.3 Indicele de imparțialitate

Alocarea optimă se poate determina dacă s-a adoptat unul din criteriile de optimalitate. Dacă o schemă conduce la o alocare diferită de cea optimală, se poate determina gradul de neimparțialitate după cum urmează. Presupunem că o schemă alocă capacitățile $\{\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n\}$ în loc de cele optime $\{\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n\}$. Se normalizează alocările pentru fiecare sursă $x_i = \tilde{x}_i / \hat{x}_i$ şi se calculează indicele de imparțialitate F (Fairness): $F = \frac{(\Sigma_i x_i)^2}{n(\Sigma_i x_i^2)}$

Cum alocările x_i variază de obicei în timp, F poate fi reprezentat ca o funcție de timp. Pentru calculul lui F poate fi folosit şi debitul pentru un interval de timp dat.

BUPT

3.5.4 Robustețea

Schema aleasă trebuie să fie insensibilă la abateri mici. De exemplu mici nepotriviri ale parametrilor sau pierderilor unor mesaje de control nu trebuie să paralizeze rețeaua. Este posibilă şi izolarea surselor cu comportare necorespunzătoare şi protejarea celorlalți utilizatori. De asemenea algoritmul trebuie să se adapteze rapid fără oscilații la schimbările topologice ale rețelei.

3.5.5 Implementabilitatea

Schemele nu trebuie să impună o anumită configurație a modului de comutare. Acesta argument joacă un rol hotărâtor în selecția finală (deoarece multe scheme s-a constatat că nu lucrează cu planificarea FIFO). În plus cu cât este mai simplu, cu atât este mai uşor de implementat, eventual chiar prin hardware deci este mai rapid.

3.5.6 Configurații de simulare

Pentru compararea diferitelor propuneri s-au propus o serie de configurații de rețele, multe dintre acestea având nodurile conectate în serie. Cea mai populară dintre configurații este cea de tip "loc de parcare" pentru studiul imparțialității. Numele şi structura este asemănătoare cu cea a marilor parcări de lângă teatre care constă din câteva loturi de parcare conectate printr-o singură cale de ieşire (vezi fig.3.2). Congestia apare la sfârşitul spectacolului când toate

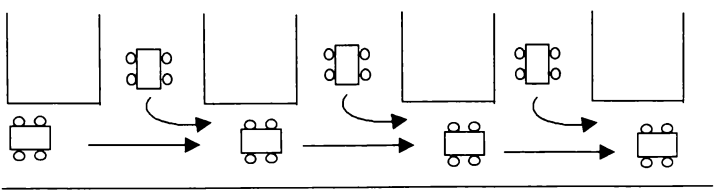


Fig 3.2 a Ieşirea maşinilor dintr-o parcare

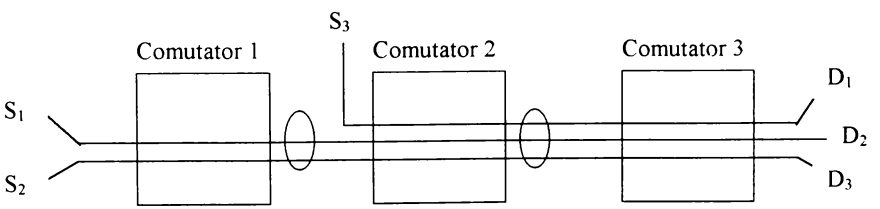


Figura 3.2 b Configurație echivalentă pentru rețeaua de comunicații
***S* – surse *D* – destinații *F* – fluxurile generate de surse**

Figura 3.2 Configurația de tip loturi de parcare

BUPT

maşinile încearcă să intre pe aceeaşi cale de ieşire. Pentru reţelele de calculatoare loturile de parcare se asociază cu comutatoarele şi dacă există *n* comutatoare există *n* CV-uri, cu primul CV pornind de la primul comutator până la sfârşit. În figura 3.2 este prezentată o structură cu 3 comutatoare.

3.6 Structura traficului

Dintre structurile de trafic folosite în diferite simulări cele mai des folosite au fost următoarele [AK, AST, Str, JSD]:

1) Surse persistente: aceste surse, "infinite" sau "lacome" au întotdeauna celule de transmis şi astfel reţeaua e mereu congestionată,
2) Surse decalate: sursele pornesc la momente de timp diferite. Acestea permit studierea schemelor în regim de pornire şi de oprire,
3) Surse în rafale: aceste surse oscilează între starea activă şi cea inactivă. Pe timpul stării active generează o rafală de celule. Acesta e un model mai realist decât al surselor infinite. Cu aceste surse, dacă încărcarea totală a liniei e sub 100%, atunci nu se pune problema imparţialităţii şi a traficului eficace, parametrul care se urmăreşte fiind timpul de răspuns la rafală, timpul de la recepţia primei celule şi transmisia ultimei celule. Dacă rafala soseşte la o rată fixă, ea este numită în buclă deschisă. Un scenariu mai realist este cel în care următoarea rafală apare la un timp după primirea răspunsului la rafala precedentă. În acest caz apariţia rafalei e afectată de congestia reţelei şi modelul de trafic se numeşte în buclă închisă.

3.7 Scheme de tratare a congestiei

În cele ce urmează vor fi descrise câteva propuneri de scheme de tratare a congestiei care au fost prezentate Forumului ATM, [JK, JSD, MS, J1, F1, KR, GK, KO]. Cele mai importante dintre ele sunt cele bazate pe credite şi cele bazate pe rată, care vor fi descrise mai în detaliu.

3.7.1 Rezervarea rapidă

Această propunere, făcută de France Telecom, pretinde ca sursa să transmită o celulă de rezervare de resurse RM (Resource Management) care să conţină cererea despre lăţimea de bandă solicitată, înainte de a transmite celelalte celule. Dacă un ruter nu poate accepta cererea el distruge pur şi simplu celula RM; sursa va intra în depăşire de timp şi va retransmite cererea. Dacă ruterul poate accepta cererea el pasează mai departe celula RM următorului ruter. În final, destinaţia returnează celula sursei care va putea atunci să-şi transmită rafala.

BUPT

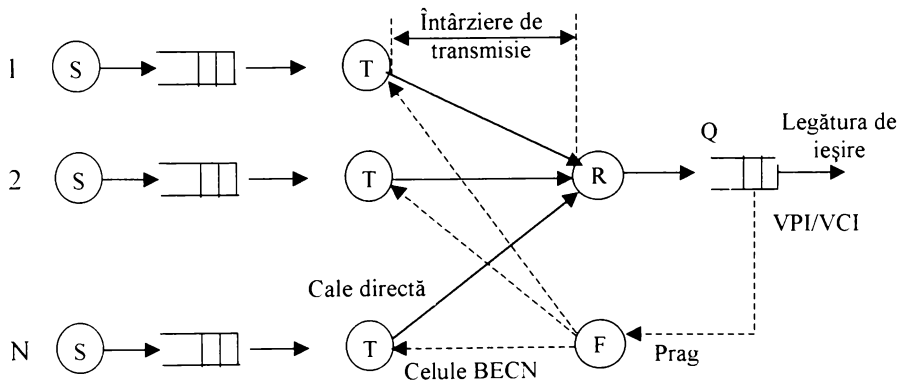
Rafala trebuie deci să accepte cel puțin timpul de parcurs tur-retur RTT (Round Trip Time) chiar şi atunci când rețeaua este neutilizată, ca în majoritatea cazurilor. Pentru a evita această întârziere a fost propus şi modul de transmisie imediată IT (Immediate Transmission), în care rafala este transmisă imediat după celula RM. Dacă un ruter nu poate satisface cererea, el distruge şi celula şi rafala şi transmite o indicație sursei. În plus celula RM poate conține dimensiunea rafalei, iar ruterul va accepta cererea dacă are suficientă memorie disponibilă. Rezervarea rapidă nu a fost acceptată de Forumul ATM pentru că duce la întârzieri excesive în timpul operării normale, respective la pierderi excesive în timpul congestiei.

3.7.2. Controlul ratei bazat pe întârziere

Această propunere făcută de Fujitsu şi similară cu cea a lui Jain presupune că sursa urmărește întârzierea RTT prin transmiterea periodică a unor celule RM care conțin o etichetă de timp. Celulele sunt returnate de către destinație. Sursa folosește eticheta de timp pentru a măsura RTT şi va deduce astfel nivelul congestiei. Această abordare are avantajul că nu se cere vreo reacție din partea rețelei şi poate fi folosită şi dacă porțiuni de traseu conțin rețele non-ATM. În plus nu necesită nici o standardizare deoarece fiecare pereche poate coopera fără să implice rețeaua. Nici această propunere nu a fost acceptată de Forumul ATM.

3.7.3. Notificarea explicită, înspre înapoi, a congestiei BECN

Această metodă BECN (Backward Explicit Congestion Notification constă din urmărirea de către fiecare ruter a imaginii propriilor cozi de pachete, şi transmiterea unor celule RM sursei dacă este congestionat. La primirea unei celule RM sursele îşi vor înjumătăți ratele de transmisie. Apoi dacă nu se mai primeşte nici o celulă în timpul perioadei de revenire, rata fiecărui CV e dublată



VPI – identificator cale virtuală; VCI – identificator circuit virtual
Fig. 3.3 Controlul BECN, de informare a sursei despre congestie

BUPT

la fiecare perioadă, până se ajunge la rata de vârf. Pentru asigurarea imparțialității, perioada de revenire a fost făcută proporțională cu rata CV-ului, astfel că scăderea ratei de transmisie duce la scurtarea perioadei de revenire, (vezi figura 3.3 şi 3.4)

Metoda nu a fost acceptată deoarece s-a considerat că nu este imparțială: sursele care primesc semnalul de congestie nu sunt totdeauna cele care cauzează congestia.

3.7.4 Distrugerea timpurie a pachetelor, EPD

Această metodă EPD (Early Packet Discard), prezentată de SUN Microsystems, se bazează pe observația că un pachet este compus din mai multe celule. Este preferabil să fie distruse toate celulele unui pachet decât să fie distruse la întâmplare celule din pachete diferite. În AAL5 când primul bit al tipului de încărcătură utilă este 0 atunci al treilea bit indică sfârşitul mesajului EOM (End of Message). Când cozile ruterului încep să se umple, ruterul caută fanionul EOM şi distruge toate celulele între acesta şi următorul EOM, de pe respectivul CV. Se impune şi aici observația că celulele care ajung la un tampon de memorie plin pot să nu aparțină CV-ului care produce congestia. Deoarece metoda permite lucrul individual al fiecărui ruter, fără comunicări cu sursa şi fără standardizări ea a fost implementată de mulți producători de ruter.

3.7.5 Fereastra pe legătură şi controlul binar cap la cap al ratei

Această metodă constă din combinarea calităților schemelor bazate pe rată şi a celor bazate pe credite. Pe fiecare legătură se foloseşte un control al fluxului prin fereastră şi în plus se face un control cap la cap al ratei printr-o notificare înspre înainte a congestiei. Controlul cu ferestre este realizat pe legătură şi nu pe CV ca la schemele bazate pe credite. Algoritmul se scalează bine şi garantează absența pierderilor. Dar el nu a fost acceptat nici de partizanii schemelor bazate pe rată, nici de partizanii schemelor bazate pe credite, deoarece conținea prea mult din propunerile părții adverse.

3.7.6 Alocare imparțială cu reacție despre memorie şi rată

Această propunere, de la XEROX şi CISCO constă în trimiterea periodică de către sursă a unor celule RM pentru determinarea gradului de utilizare a tampoanelor de memorie şi a lățimii de bandă în porțiunile strangulate. Ruterele calculează alocarea corectă pentru fiecare CV. Valoarea minimă rezultată după alocarea de la respectivul ruter şi cele precedente, este plasată în celula RM. Ruterele urmăresc şi lungimea cozilor fiecărui CV. În aceeaşi celulă RM se

BUPT

plasează şi valoarea maximă a cozii din respectivul ruter şi cele precedente. Fiecare ruter implementează FQ (Fair Queuing) ceea ce înseamnă că se menţin cozi separate pentru fiecare CV şi că se calculează timpul la care celula ar trebui să-şi termine transmisia dacă cozile sunt servite în manieră FQ Bit (adică are loc o explorare ciclică a cozilor şi s-ar transmite un bit odată din fiecare coadă). Celulele sunt planificate pentru a transmite în această ordine calculată.

Alocarea imparţială a CV-ului este determinată ca inversul intervalului de timp dintre sosirea şi transmisia celulei. Intervalul reflectă numărul celorlalte CV active. Dacă numărul şi deci intervalul este aleator, se recomandă folosirea mediei câtorva intervale observate. Deoarece schema necesită câte o coadă pe fiecare CV ea este relativ complexă.

3.7.7 Abordarea bazată pe credite

Aceasta a fost una din principalele abordări, implementată de Digital, Mitsubishi ş.a. Abordarea constă dintr-un control al fluxului prin ferestre per CV şi per legătură. Fiecare legătură constă dintr-un nod transmiţător, care poate fi un ruter sau o sursă, şi un nod receptor, care poate fi un ruter sau o destinaţie. Fiecare nod menţine câte o coadă separată pentru fiecare CV. Receptorul verifică lungimea cozilor asociate fiecărui CV şi determină numărul de celule pe care transmiţătorul le poate emite pe acel CV. Acest număr se numeşte credit. Transmiţătorul nu are voie să emită mai multe celule decât îi sunt alocate prin credit (vezi figura 3.5).

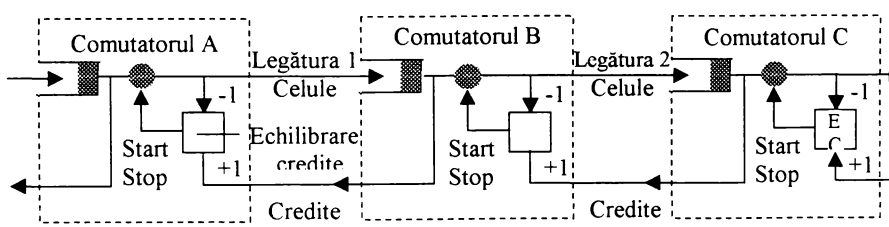


Fig. 3.5 Schemă de control a fluxului cu credite legătură pe legătură

Dacă există un singur circuit virtual activ, creditul trebuie să fie suficient de mare ca să permită utilizarea integrală a liniei în orice moment, adică:

$$Credit \geq Rata\ celulelor\ pe\ legătură \times RTT\ a\ legăturii$$

Rata celulelor pe legătură poate fi calculată împărţind lăţimea de bandă a liniei (Mbps) la dimensiunea celulei (biţi).

Schema descrisă mai departe este numită schema circuitului virtual cu controlul fluxului FCVC (Flow Controlled Virtual Circuit). La ea apar două probleme. Prima problemă apare în cazul pierderii creditelor, caz în care sursa

BUPT

nu îl mai poate afla. Apoi, fiecare CV trebuie să rezerve pe toată durata RTT toate tampoanele de memorie, chiar dacă linia este partajata între mai multe CV-uri. Această problemă s-a rezolvat prin introducerea algoritmului cu re-sincronizare bazat pe credite şi o versiune adaptivă.

Algoritmul cu resincronizare bazat pe credite constă din menţinerea atât la transmiţător cât şi la receptor a câte unui numărător de celulele transmise şi recepţionate pe fiecare CV şi comunicarea reciprocă periodică a acestor numărătoare. Diferenţa dintre celulele transmise şi cele recepţionate reprezintă celulele pierdute pe linie. Receptorul va reacorda atâtea credite suplimentare pentru acel CV cât este diferenţa numărătoarelor.

Algoritmul adaptiv FCVC constă din alocarea pentru fiecare CV a unei părţi din necesarul de tampoane de memorie pe un RTT. Fracţia depinde de rata la care CV-ul foloseşte creditele. Pentru un CV foarte activ fracţia este mai mare iar pentru un CV lent este mai mică. CV-urile inactive primesc o fracţie mică, fixa de credite. Dacă un CV nu-şi utilizează creditele, şi observă că rata de utilizare pe o perioadă este scăzută şi va primi o alocare redusă de tampoane de memorie, şi deci de credite, în următorul ciclu. Algoritmul FCVC adaptiv reduce considerabil necesarul de tampoane de memorie, dar şi introduce un timp de intrare în regim. Dacă CV devine activ, va trece un anume timp înainte de a-şi putea folosi întreaga capacitate a legăturii, chiar dacă nu există alţi utilizatori activi.

3.7.8 Abordarea bazată pe rată

Această soluţie a fost iniţial propusă de Mike Hlucihyj şi apoi a fost modificată prin contribuţia altor 22 de companii [J1]. Propunerea originală pleca de la versiunea DEC Bit a schemei bazată pe rata, care consta dintr-un control cap-la-cap cu reacţie pe un singur bit din partea reţelei. Ruterele verifică lungimea cozilor şi în caz de congestie poziţionează EFCI din antetul celulelor. Destinaţia verifică această indicaţie la intervale periodice şi transmite înapoi sursei o celulă RM. Sursele folosesc un algoritm de creştere aditivă şi de scădere multiplicativă pentru a-şi ajusta rata.

Acest algoritm foloseşte "polaritatea negativă a reacţiei" în sensul că celulele RM sunt transmise doar pentru scăderea ratei, nu şi pentru creşterea sa. Pe de altă parte, polaritatea pozitivă poate fi folosită doar pentru creşterea ratei, nu şi pentru scăderea sa. Dacă celulele RM se folosesc atât pentru creşterea cât şi pentru scăderea ratei algoritmul se numeşte bipolar.

În cazul polarităţii negative se poate întâmpla ca datorită congestiei, celulele RM să fie pierdute pe calea de retur. În acest caz, pe calea directă, sursa va continua să-şi crească rata de transmisie şi ar putea să-şi depăşească capacitatea. Acest dezavantaj a fost eliminat în versiunea următoare prin folosirea polarităţii pozitive. Sursa poziţionează bitul EFCI din toate celulele, cu excepţia celulei *n*. Destinaţia va transmite o celulă RM sursei de creştere a ratei, dacă a recepţionat o celulă oarecare cu EFCI nepoziţionat. Sursele îşi descresc rata până ce primesc o reacţie pozitivă. Deoarece îşi scad rata

BUPT

proporţional cu rata actuală, schema a fost numită "algoritm de control proporţional a ratei" PRCA (Proportional Rate Control Algorithm).

Algoritmul PRCA are probleme legate de impartialitatea schemei. Dacă toate ruterele au acelaşi nivel de congestie, CV-urile care traversează mai multe noduri au o probabilitate mai mare de a avea EFCI = 1 decât cele care traversează un număr redus de noduri. Dacă p este probabilitatea ca EFCI să fie pus pe 1 de un nod, atunci probabilitatea ca el să fie pus pe 1 după n noduri va fi $1-(1-p)^n$ sau np. Deci pe căile lungi, CV-urile au puţine ocazii să-şi crească rata, aceasta fiind scăzută mult mai des decât pentru CV-urile de pe căile scurte. Această problemă a fost numită "problema reducerii". O soluţie la problema reducerii este reacţia selectivă sau marcarea inteligentă, unde ruterul congestionat ia în considerare rata curentă a CV-ului împreună cu nivelul său de congestie, pentru a decide dacă să pună pe 1 bitul EFCI din celulă. Ruterul calculează partea corectă şi în caz de congestie pune pe 1 biţi EFCI din celulele aparţinând doar acelor CV-uri ale căror rate depăşesc această parte corectă. Acele CV-uri care au ratele sub această valoare nu sunt afectate.

3.7.8.1 Schema MIT

Reacţia binară este prea lentă pentru controlul bazat pe rată din reţelele de mare viteză; indicarea explicită a ratei este mai rapidă şi oferă o flexibilitate mai mare proiectanţilor de rutere.

Reacţia binară pe un singur bit poate spune sursei doar să-şi crească sau să-şi scadă rata. Ea a fost concepută în 1986 pentru reţele fără stabilirea conexiunii în care nodurile intermediare nu ştiu nimic despre fluxuri şi cererile lor. Reţelele ATM sunt orientate pe conexiune, în sensul că înainte ca două sisteme să comunice, ele trebuie să informeze ruterele intermediare despre pretenţiile lor asupra serviciului şi parametrii de trafic. Ruterele ştiu deci exact cine le foloseşte resursele şi căile fluxurilor sunt mai degrabă statice. Această informaţie nu este folosită de schema de reacţie binară.

Cel mai important lucru este faptul că reacţia binară a fost proiectată pentru controlul cu ferestre şi prea lentă pentru controlul bazat pe rată. La controlul cu ferestre diferenţa dintre fereastra curentă şi fereastra optimă va duce la o uşoară creştere în lungime a cozilor. În controlul bazat pe rată, o diferenţă mică între rata curentă şi cea optimă duce la o creştere continuă a lungimii cozilor. Reacţia trebuie să fie foarte rapidă, nu se va mai permite să treacă un interval de câteva RTT-uri, ca cel necesar reacţiei binare pentru a atinge optimul în operare.

Reacţia implicită bazată pe rată mai are şi alte avantaje. Controlul este direct. Ruterele de intrare pot monitoriza celulele RM returnate şi să folosească rata direct în algoritmul lor de control. Apoi, datorită convergenţei rapide, sistemul ajunge rapid la punctul de operare optim, rata iniţială având o influenţă redusă. Schema este robustă faţă de eronarea şi pierderea celulelor RM. Următoarea celulă RM corectă va conduce sistemul la punctul de operare corect.

BUPT

În schema MIT (Massachusets Institute of Technology) [CRL], fiecare sursă trimite o celulă RM la fiecare *n* celule. Această celulă conține rata curentă a celulelor de pe respectivul circuit virtual CCR (Current Cell Rate) şi rata dorită. Ruterele monitorizează ratele tuturor circuitelor virtuale şi calculează partajul corect de bandă alocată. Dacă vreun circuit virtual lucrează la o rată mai mică decât cea corectă alocată i se creşte rata la cea dorită. Dacă rata dorită a unui CV este mai mare decât rata alocată imparțial atunci câmpul cu rata dorită se "reduce" prin înscrierea valorii ratei alocată imparțial; iar în celulele RM se pune pe 1 bitul de "reducere". Destinația returnează celula RM sursei, care-şi va ajusta rata sa la valoare indicată în celula RM. Dacă bitul de reducere este 0 sursa poate cere o rată dorită mai mare în următoarea celulă RM. Dacă
bitul este pus pe 1, sursa trebuie să folosească rata curentă ca rata dorită în următoarea celulă RM. Dacă bitul este pus pe 1, sursa trebuie să folosească rata curentă ca rată dorită în următoarea celulă RM.

Rata alocată imparțial e calculată folosind un procedeu iterativ. Inițial, alocarea imparțială a ratei se face împărțind lățimea de bandă a liniei la numărul de CV-uri active. Circuitele virtuale a căror rată este mai mică decât cea alocată imparțial se numesc "subîncărcate". Dacă numărul de CV-uri subîncărcate creşte la fiecare iterație, rata alocată imparțial FSR (Fair Share Rate) este recalculată astfel:

$$FSR = \frac{Latimea\ de\ banda\ a\ liniei - \sum Latimile\ de\ banda\ ale\ CV\ subincarcate}{Numar\ de\ CV - Numar\ de\ CV\ subincarcate} \qquad (3.3)$$

Iterația se repetă până rămân neschimbate numărul de CV-uri subîncărcate şi FSR. În [CRL] se arată că sunt suficiente două iterații pentru convergența procedurii. De asemenea se arată că schema MIT atinge optimul max-min în $4k$ RTT-uri unde k este numărul de strangulări. Această propunere a fost bine primită, cu excepția calculului FSR care necesită n operații, unde n este numărul de circuite virtuale.

3.7.8.2 Algoritm îmbunătățit de control proporțional a ratei

Dorință de a crea o schemă care să implice un număr $0(1)$ operații a condus la schema EPRCA (Enhanced Proportional Rate-Control Algorithm).

Sursele transmit celulele cu CTCI = 0, iar după n celule transmit o celulă RM. Celula RM conține explicit rata dorită, ER (Explicit Rate), rata curentă CER (Current Explicit Rate) şi bitul CI (Congestion Indication) de indicare a congestiei. Sursa inițializează ER la rata de vârf PCR şi pune CI pe 0.

Apoi rutele calculează FSR şi dacă este necesar reduc câmpul ER în celula RM returnată. Se calculează apoi rata medie permisă a celulelor MACR (Mean Allowed Cell Rate), folosind o medie ponderată exponențial, iar FSR se asignează la o fracție din această medie.

BUPT

$$MACR = (1 - \alpha)\, MACR + \alpha \times CER$$
$$FSR = SWDPF \times MACR \tag{3.4}$$

Aici, α este factorul de mediere exponențială, iar SWDPF este un coeficient (Switch Down Presure Factor) apropiat de 1 dar mai mic decât 1. Valorile optime sunt pentru α =1/16 iar SWDPF = 7/8. Destinația urmăreşte bitul CTCI din celulele de date. Dacă ultima celulă de date văzută avea EFCI =1, atunci bitul CI se pune pe 1 în celula RM. Acest bit poate fi pus pe 1 în celulele RM returnate şi dacă, în rutere, cozile depăşesc o valoare de prag. Sursele îşi scad continuu rata, după fiecare celulă.

$$ACR = ACR \times RDF, \tag{3.5}$$

unde ACR este rata disponibilă a celulelor iar RDF este un factor de reducere. Când sursa primeşte celula RM returnată ea îşi creşte rata cu un volum ATR dacă i se permite:

Dacă CI = 0 atunci ACR = Min (ACR + AIR, ER, PCR),

iar

Dacă CI = 1 atunci ACR este neschimbat

Algoritmul EPRCA permite lucrul ruterelor atât cu reacția binară cât şi cu reacție explicită. Problema principală a EPRCA este algoritmul de detecție a congestiei ruterului, bazat pe un prag al lungimilor cozilor. Dacă lungimea cozilor depăşeşte un prag, ruterul este considerat congestionat. Acest lucru poate duce la nonimparțialitate. Astfel sursele care încep să transmită mai târziu vor fi mai defavorizate decât cele care pornesc mai devreme. Problema a fost rezolvată ulterior folosind ca indicator al sarcinii ritmul de creştere al cozilor.

3.7.8.3 Editarea congestiei prin algoritmul OSU

La Universitatea de Stat din Ohio au fost dezvoltate o serie de scheme de evitare a congestiei, explicite, bazate pe rata. Prima schemă numită OSU-TUB (Ohio State University-Target Utilization Band) prevede că ruterele să-şi măsoare rata de intrare pe un interval fix, numit interval de mediere, şi compararea sa cu rata dorită, "țintă", pentru a calcula factorul de încărcare *z*:

$$z = \frac{rata\ de\ intrare}{rata\ dorita} \tag{3.6}$$

BUPT

Rata dorită se alege sub lățimea de bandă a liniei (85% / 35% din acesta). Toate CV-urile trebuie să-şi împartă rata de intrare cu z: dacă $z = 2$, CV-urile îşi vor înjumătăţii rata de intrare, iar dacă $z = 0,5$ ele îşi vor dubla rata de intrare. FSR se va calcula ca:

$$FSR = \frac{rata\ dorita}{numar\ de\ surse\ active} \tag{3.7}$$

Toate sursele a căror rată este mai mare decât FSR îşi vor divide rata cu z / (1+D) (unde D este abaterea mică a lui z faţă de 1) iar cele care au rata sub FSR îşi vor divide rata cu z / (1 – D). Acest algoritm OSU-TUB este imparţial. El duce la evitarea congestiei, obţinându-se un debit eficace mare şi întârzierii reduse. Luând rata dorită uşor mai mică decât capacitatea liniei algoritmul asigură o dimensiune foarte redusă a cozilor, deci întârzieri reduse. În plus ruterul lucrează cu foarte puţini parametrii, faţă de EPRCA, care sunt uşor de calculat.

Timpul de atingere a stării staţionare este foarte redus de 10 / 20 ori mai scurt ca cel de EPRCA.

3.7.8.4 Evitarea congestiei folosind controlul proporţional

Această schemă de evitare a congestiei CAPC (Congestion Avoidance using Proportional Control) foloseşte utilizarea dorită uşor subunitară ca la OSU-TUB, ceea ce va menţine redusă lungimea cozii. Ruterele măsoară rata de intrare şi z, factorul de încărcare, cu care actualizează apoi FSR.

Dacă încărcare este mică, $z < 1$, FSR este crescută.

$$FSR = FSR \times Min\ (ERU,\ 1 + (1 - z) \times RUP) \tag{3.8}$$

Unde ERU = creşterea maximă permisă (aici 1,5) şi
RUP = panta, parametru între 0,025 şi 0,1.

Dacă este supraîncărcare, $z > 1$, FSR este scăzută

$$FSR = FSR \times Max\ (EFR,\ 1 - (1 - z) \times RDN) \tag{3.9}$$

Unde ERF = scăderea minimă cerută (aici 0,5) şi
RDN = panta, parametru între 0,02 şi 0,8.

FSR este rata maximă care poate fi garantată fiecărei CV de către ruter. Pe lângă z, schema mai foloseşte un prag pentru lungimea cozii. Dacă acest prag este depăşit bitul CI din toate celulele RM este pus pe 1. Aceasta împiedică sursele să-şi crească rata şi permite scăderea lungimii cozii. Caracteristica acestui algoritm este oscilaţia liberă în jurul stării staţionare. Frecvenţa

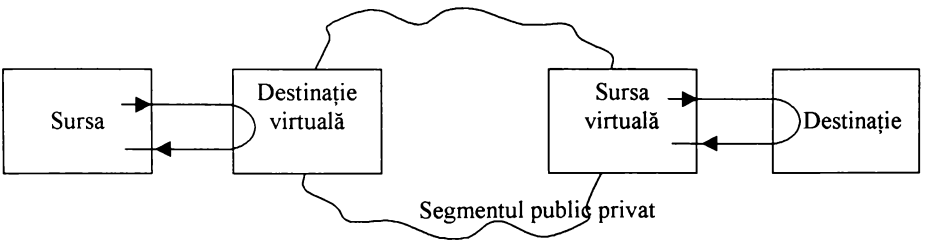
BUPT

oscilațiilor este proporțională cu $1 - z$; astfel în starea staționară z fiind 1, frecvența fiind 0 şi perioada infinită.

În prezent nu s-a hotărât asupra standardizării vreunuia dintre algoritmi astfel că fiecare producător e liber să folosească propriul algoritm.

3.8 Surse şi destinații virtuale

Unul dintre dezavantajele controlului cap la cap bazat pe rată este că RTT poate fi foarte mare. Problema se poate rezolva segmentând rețeaua în părți mai mici, iar ruterele de graniță or trebui să acționeze ca "surse virtuale" respectiv "destinații virtuale". Segmentarea rețelei permite reducerea buclelor de reacție. Astfel sistemele intermediare pot să folosească orice schemă de control a congestiei. Acest lucru permite rețelelor de telecomunicații să folosească interfețe standard doar la ruterele de intrare / ieşire din rețea. Cel mai important lucru este că sursele / destinațiile virtuale furnizează o interfață mai robustă cu



Sursă şi destinație virtuale

Controlul ratei nod – la – nod

Figura 3.6

rețele publice, în sensul că resursele din interiorul rețelei nu sunt influențate de disciplina utilizatorilor. Comportarea necorespunzătoare a utilizatorilor va fi izolată de prima buclă. Prin utilizator se înțelege inclusiv rețelele private cu rutere care pot sau nu să fie disciplinate.

Trebuie totuşi semnalat că soluția cu destinații şi surse virtuale este relativ costisitoare, în sensul că necesită cozi de aşteptare per CV. Nu există limită pentru numărul de segmente care pot fi create, în cazul extrem fiecare ruter poate acționa ca o sursă / destinație virtuală şi se ajunge la controlul nod la nod.

BUPT

3.9 Comparaţia între abordarea bazată pe credite şi cea bazată pe rată

În cele ce urmează vor fi prezentate pe scurt principalele caracteristici ale celor două abordări, care servesc ca argumente pentru adoptarea uneia şi celeilalte soluţii. Forumul ATM, după lungi discuţii, a optat pentru soluţia bazată pe rată. Partizanii fiecărei soluţii au avut în minte scopuri diferite, şi n-au fost dispuşi la nici un compromis.

1. Scalabilitatea: în cazul soluţiei cu credite, trebuie menţionată câte o coadă pentru fiecare CV, chiar dacă acesta este inactiv. Dacă numărul de CV-uri ajunge la câteva milioane, complexitatea ruterului creşte în asemenea măsură încât soluţia cu credite nu mai poate fi folosită. Acesta este singurul argument serios împotriva soluţiei cu credite şi principalul motiv pentru care aceasta n-a fost adoptată. Abordarea pe baza ratei nu necesită câte o coadă pe fiecare CV, putând să lucreze cu sau fără aceste cozi per CV, soluţia rămânând la alegerea proiectantului.
2. Pierdere zero de celule: abordarea cu credite garantează transmisia fără pierderi de celule, independent de structura traficului, numărul de conexiuni, dimensiunea tampoanelor de memorie, numărul de noduri, gama lăţimilor de bandă şi a timpilor de propagare. Chiar în prezenţa unei supraîncărcări severe lungimea cozilor nu poate depăşi valoarea permisă de credite. Soluţia bazată pe rată nu poate garanta o transmisie fără pierderi de celule. În suprasarcină lungimea cozilor poate creşte într-atât încât tampoanele să fie depăşite şi celulele să fie pierdute. Partizanii soluţiei bazată pe rată consideră acceptabilă pierderea de celule a cărei probabilitate e redusă în cazul tampoanelor mari. În plus, ei argumentează că oricum apar pierderi datorate erorilor şi oricum utilizatorii trebuie să ia măsuri împotriva pierderilor, indiferent care este cauza lor.
3. Intrarea în regim permanent are loc foarte repede la soluţia cu credite, unde CV-urile ajung să lucreze imediat la viteza maximă, deoarece capacitatea liberă poate fi folosită imediat. Soluţia adaptivă cu credite şi soluţia bazată pe rată au nevoie de câteva RTT-uri pentru a intra în regim.
4. Izolarea şi utilizatorii nedisciplinaţi: marele avantaj al cozilor asociate fiecărui CV este izolarea utilizatorilor indisciplinaţi, care nu pot întrerupe sau perturba activitatea celor disciplinaţi. Acest lucru este adevărat pentru schemele statice cu credite şi mai puţin adevărat pentru cele adaptive, unde sursele nedisciplinate pot să obţină o parte mai mare de memorie crescând rata de transmisie. Dar izolarea este obţinută nu din cauza creditelor ci din cauza asocierii câte unei cozi la fiecare CV, soluţie care se poate aplica, dacă trebuie, şi la abordarea bazată pe rată (deşi aceasta a fost principalul argument împotriva soluţiei cu credite).
5. Necesarul de tampoane de memorie este mai redus la soluţia cu credite decât la cea bazată pe rată cu reacţie binară, dar acest avantaj dispare când se foloseşte soluţia bazată pe rată şi reacţie explicită. În soluţia statică cu

BUPT

credite necesarul de tampoane per CV este proporţional cu întârzierea pe linie, iar la cea bazată pe rată necesarul total de tampoane depinde de întârzierea cap la cap. Schema adaptivă cu credite permite un volum mai redus de memorie decât cea statică, dar cu preţul creşterii timpului de intrare în regim.

6. Estimarea întârzierii: iniţializarea parametrilor pentru controlul congestiei în schema cu credite pretinde date despre RTT a liniei, sau cunoaşterea cel puţin a lungimii şi vitezei acesteia; acest lucru nu este necesar, deşi este de mare ajutor, la soluţia bazată pe rată.
7. Flexibilitatea proiectării: în schema bazată pe rată ruterelor li se permite o mare flexibilitate în decizia de alocare a resurselor. Rutere diferit pot folosi mecanisme diferite, fără să fie afectată colaborarea dintre ele în aceeaşi reţea. De exemplu unele rutere pot opta pentru minimizarea lungimii cozilor dar, altele pentru maximizarea profitului lor. Pe de altă parte, soluţia cu credite pretinde fiecărui ruter folosirea cozilor asociate fiecărui CV cu servire carusel.
8. Complexitatea ruterelor şi a sistemelor terminale: soluţia cu credite face ca ruterele să fie complexe, sarcina sistemelor terminale fiind mai simplă. Partizanii săi susţin că interfaţa între calculatorul gazdă şi reţeaua NIC (Network Interface Card) este mult mai simplă deoarece nu trebuie planificate fiecare şi toate celulele. Acolo unde există credite disponibile, celulele pot fi transmise la rata de vârf. Partizanii soluţiei bazate pe rată socotesc că toate NIC-urile lor trebuie să aibă planificatoare pentru traficul CBR şi VBR şi utilizarea aceluiaşi mecanism pentru traficul ABR nu complică prea mult placa.

Mai există şi alte argumente în favoarea uneia sau celeilalte dintre soluţii. Argumentul principal în defavoarea soluţiei cu credite, a fost însă existenţa cozilor asociate fiecărui CV, ceea ce a determinat şi respingerea soluţiei unei propuneri integrate, care să permită producătorilor să aleagă între cele două soluţii. Astfel abordarea bazată pe rată a fost votată ca unică soluţie.

BUPT

Capitolul 4

PLANIFICAREA TRAFICULUI

Rețelele de mare viteză sau de bandă largă, cu integrarea serviciilor, trebuie să implementeze algoritmi de planificare a traficului în comutatoare sau rutere, pentru a asigura garantarea unei mari game de calități a serviciilor. În rețelele ATM nodurile de comutare sunt denumite comutatoare, termenul de ruter fiind folosit în general pentru Internet [ZK]. Rolul algoritmilor de planificare este de a alege, pentru fiecare linie de ieşire a comutatorului, pachetele care vor fi transmise în următorul ciclu. Alegerea se face dintre pachetele aparținând fluxurilor care partajează linia de ieşire. De asemenea, noțiunile de canal, circuit, conexiune, conversație respectiv flux sunt similare.

Controlul sau evitarea congestiei în rețele au constituit subiectul unor largi dezbateri din ultimii ani, fiind prezentate o serie de soluții (vezi capitolul 3). Majoritatea acestora pretind, sau pot fi îmbunătățite prin, o proiectare corectă a unei componente a rețelei şi anume algoritmul de planificare. Deşi ponderea sa este redusă, el constituie o componentă critică a oricărei scheme de control a congestiei.

Mecanismele de control pot fi clasificate în două categorii, în funcție şi de tipul de trafic, [Ne] şi anume: i) mecanisme în buclă deschisă asociate traficului "garantat" sau "capitalist", respectiv, ii) mecanismele în buclă închisă, care se ocupă de traficul de efort maxim sau "socialist" [fig 4.1]. În cazul mecanismelor

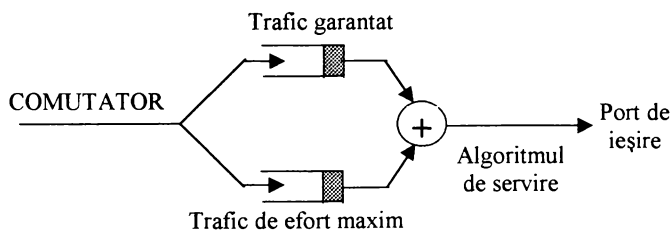


Fig. 4.1 Tampon de ieşire cu două clase de trafic

în buclă deschisă, fiecare aplicație specifică parametrii traficului (rata de vârf, rata medie, rafala, etc.) în momentul stabilirii apelului, precum şi pretențiile asupra calității serviciului (întârziere, jitter, rata de pierderi, etc.). Funcție de aceşti parametri, rețeaua alocă necesarul de resurse (lățime de bandă a liniei, memorie în comutare) aplicației respective. Acest mod de lucru implică de fapt existența rețelelor orientate pe conexiune. În aceste rețele algoritmul de planificare se ocupă cu alocare dinamică a resurselor rețelei, la fiecare nod din rețea, astfel încât pachetele să aibă garantată o anumită întârziere, să nu apară

BUPT

pierderi de pachete şi asigurarea acestor servicii să fie independentă de comportarea celorlalte aplicaţii [SV1, PS, KalV, Kes]. În acelaşi timp, planificarea nu trebuie să afecteze utilizarea capacităţii liniei.

La controlul în buclă închisă participă traficul de efort maxim. Nodurile terminale verifică starea reţelei şi modifică traficul aplicaţiilor astfel încât ele să primească o parte egală, corectă din resursele reţelei, fără să introducă congestia. Pentru detectarea stării reţelei şi informarea nodurilor terminale asupra acestui lucru au fost propuse o serie de metode [KKK, Kes, LiH, MS, RF, JSD, JK]. La apariţia congestiei sursele vor fi informate într-un interval RTT şi trebuie să-şi reducă rata de transmisie pentru evitarea congestiei. Unele dintre aceste metode depind de metode sofisticate de planificare, şi îşi pot îmbunătăţi performanţele funcţie de acestea. Câteva dintre acestea sunt: pachetele perechi, controlul traficului nod la nod, dirijarea multicăi orientată pe congestie şi chiar metode bazate pe rată. Toate pretind de fapt un algoritm de planificare a traficului care să asigure pentru fiecare aplicaţie împărţirea corectă a lăţimii de bandă a liniei. Chiar şi schemele de control a fluxului bazate pe rată pentru serviciul ABR din reţelele ATM, îşi îmbunătăţesc vizibil comportarea (comportarea în regim tranzitoriu, imparţialitatea şi întârzierea) în prezenţa algoritmilor de planificare. În plus, protocoalele de nivel superior, ca TCP/IP, obţin un debit eficace mai mare, iar mecanismele de urmărire (policing) pot fi considerabil simplificate [BM, DS, KLO, MS, PT, DK, J1, RL, KalV].

Cum nici o reţea din lume nu transportă trafic de un singur tip, rezultă că atât metodele în buclă închisă cât şi cele în buclă deschisă vor trebui integrate în reţea, pentru a putea satisface cererile diferitelor aplicaţii. Controlul în buclă deschisă va fi utilizat pentru traficul de timp real, iar controlul în buclă închisă pentru traficul de date şi nu de timp real. Dar cum linia nu poate fi partajată de doi algoritmi, rezultă că acelaşi algoritm de planificare trebuie să satisfacă cerinţele ambelor tipuri de trafic. În plus, algoritmul trebuie să se supună constrângerilor impuse de mecanismul tampoanelor de memorie a comutatorului şi arhitecturii hardware a acestuia.

O altă problemă care apare la proiectarea algoritmilor de planificare a traficului, TSA (Traffic Scheduling Algorithm), este cauzată de creşterea vitezei liniilor, ceea ce duce la necesitatea implementării hardware a algoritmului. În reţelele ATM, cu pachetele (celulele) de 53 octeţi, planificatorul trebuie să fie capabil şă-şi încheie operaţiile în intervalul de transmisie a unei celule. Dacă linia are viteza de 622 Mbps acest timp este mai mic de 680 ns, deci complexitatea implementării este un factor dominant în selecţia TSA.

Comutatoarele se pot clasifica, după modul de memorare a pachetelor [SV2, CT, LC, PB], în: i) comutatoare cu memorare la intrare, IBS (Input Buffered Switch), ii) comutatoare cu memorare la ieşire, OBS (Output-Buffered Switch). La prima categorie IBS, cum pachetele sunt memorate la intrarea comutatorului, planificarea traficului poate fi simplificată prin împărţirea problemei pe două niveluri: i) planificarea transmisiei între porturile de intrare ale comutatorului, care transmit la un port comun de ieşire şi ii) planificarea pachetelor pentru portul de intrare ales. La comutatoarele OBS, pachetele sunt

BUPT

disponibile pentru transmisie imediat ce sosesc în comutator şi pot fi tratate ca fiind memorate la porturile de ieşire.

Trebuie identificate soluţiile de planificare apărute până în prezent şi caracterizate în funcţie de necesităţile impuse de reţelele cu servicii integrate. Trebuie relevate cele mai importante caracteristici ale TSA şi definită metrica performanţelor. Trebuie separată analiza TSA pentru IBS şi OBS.

4.1. Planificarea traficului în IBS

Există o mulţime de modele de comutatoare propuse [SV2, SSL]. Cea mai răspândită arhitectură este cea a comutatoarelor fără blocare, de obicei implementată în reţele cu comutatoare crossbar. Acesta fiind modelul luat în considerare în cele ce urmează, presupunem că acest comutator este capabil să transmită orice set de pachete oferite la intrarea sa, fără blocare internă, când porturile destinaţie ale pachetelor sunt distincte.

În IBS (figura 4.2) pachetele sunt memorate în porturile de intrare ale comutatorului şi trebuie stabilit un set de căi pentru transmiterea lor de la intrarea la ieşirea acestora. Planificarea se poate face în două etape: i) stabilirea

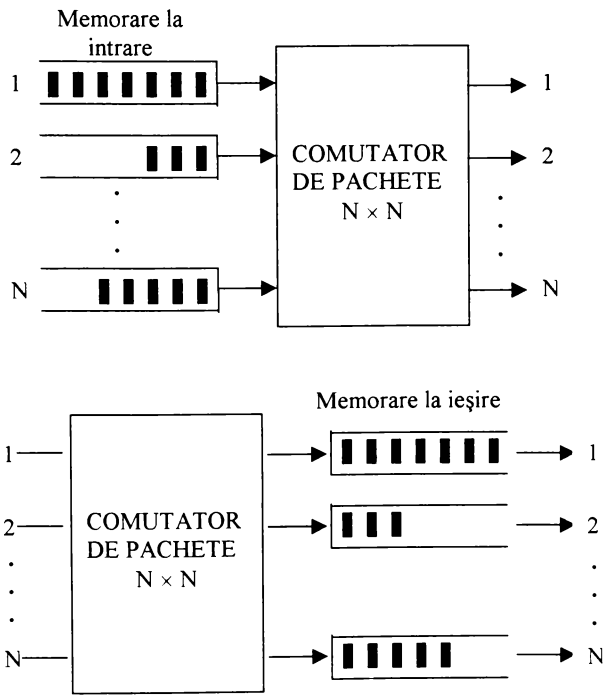


Fig. 4.2 Comutatoare cu memorare la intrare şi la ieşire

BUPT

căilor de transmitere a pachetelor de la intrarea la ieşirea comutatorului, conform cu destinaţia acestora; ii) selectarea pachetului de transmis, dintre pachetele care aşteaptă la intrare şi sunt destinate portului de ieşire ales (figura 4.3). Prima etapă este problema principală a comutării între intrare şi ieşire. Chiar în cazul

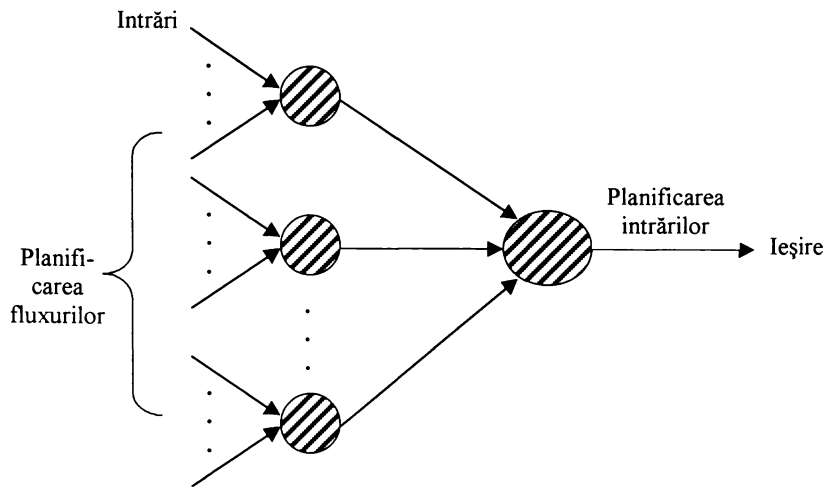


Fig. 4.3 Panificarea în două etape la comutatoarele IBS

comutatorului fără blocare, cererile conexiunilor trebuie să concureze cu alte cereri. Dacă două pachete sosesc la două porturi de intrare diferite, dar sunt direcţionate la acelaşi port de ieşire, doar unul va putea fi transmis imediat, iar celălalt va trebui păstrat în tampon. Acesta este conflictul de ieşire. Metoda folosită pentru memorarea pachetelor are o mare influenţă asupra costului şi performanţei implementării. Ideea memorării pachetelor la intrarea comutatorului este simplă şi uşor de implementat prin hardware. Dar apare o mare problemă, a blocării începutului cozii HOL, (Head-of-Line Blocking), când un pachet dintr-o coadă de aşteptare nu poate fi transmis chiar dacă destinaţia solicitată este liberă, din cauza conflictului de ieşire la capul cozii. Conflictul HOL reduce eficienţa IBS crossbar la aproximat 58% din capacitatea sa, în condiţiile unui trafic uniform distribuit.

Performanţa IBS crossbar poate fi îmbunătăţită permiţând comutatorului să aleagă unul din cele câteva pachete din coadă pentru a fi transmis într-un ciclu. Dacă se foloseşte o fereastră de memorare W, oricare din primele W pachete din coadă poate fi transmis. Îmbunătăţirea este considerabilă faţă de cozile strict FIFO, chiar cu ferestre mici. Dacă planificatorul are acces la mai mult de un pachet din fiecare tampon de intrare, eficienţa poate fi maximizată prin asocierea pachetelor din cozile de intrare cu porturile de ieşire, astfel încât în fiecare ciclu să se transmită cel mai mare număr posibil de pachete.

Problema comutării unui număr maxim de pachete în IBS corssbar, cu acces aleator la cozi, sau prin ferestre, poate fi redusă la o problemă de asociere într-un graf din două părţi [SV]. Graful bipartit este construit reprezentând

BUPT

fiecare port de intrare / ieşire printr-un vârf (punct) în primul / al doilea grup. Fiecare pachet accesibil este reprezentat printr-o legătură (muchie) de la vârful reprezentând intrarea la care este memorat pachetul, la vârful reprezentând portul de ieşire spre care este direcţionat pachetul.

Asocierea bipartită încearcă să maximizeze numărul de legături din primul grup, conectată la al doilea grup, prin alegerea unui subset de legături astfel încât nici o legătură să nu aibă vreun vârf comun; acest lucru este echivalent cu faptul că cel mult un pachet să fie comutat dinspre fiecare port de intrare spre fiecare port de ieşire, într-un ciclu.

Asocierea este maximală, dacă adăugarea unui nou pachet la asocierea existentă ar altera asignările curente. În asocierea maximală, pachetele sunt fie planificate fie blocate. Dacă în toate asocierile numărul de pachete planificate este maxim, atunci s-a atins asocierea maximă. Asocierea maximă este întotdeauna şi maximală, dar reciproca nu este întotdeauna adevărată. Asocierea maximală nu conţine niciodată mai puţin de jumătate din numărul de legături ale asocierii maxime, pentru respectivul graf bipartit.

Cele mai sus afirmate se pot folosi şi în cazul în care intrările solicită porţiuni inegale din banda legăturii de ieşire şi, chiar în cazul asocierii maxime se poate ajunge la inechitate şi instabilitate [BC, BT, HLG, LC, LM, LeSe, MS]. În plus, nici un TSA pentru IBS nu conţine vreun mijloc pentru a asigura garantarea benzii.

4.2 Planificarea traficului în OBS

În comutatoarele OBS, cu memorarea pachetelor la ieşire, pachetele care sosesc la intrare sunt disponibile imediat pentru transmisie pe legătura de ieşire. Rolul planificatorului este să selecteze pentru fiecare legătură de ieşire, dintre pachetele disponibile, aparţinând fluxurilor care partajează legătura, pachetul de transmis în ciclul următor. Astfel, planificatorul poate selecta imediat oricare dintre aceste pachete pentru transmisie (figura 4.4). În plus se presupune că pachetele nu sunt secţionate, adică nu se începe transmisia unui pachet înainte de recepţionarea ultimului bit din pachet.

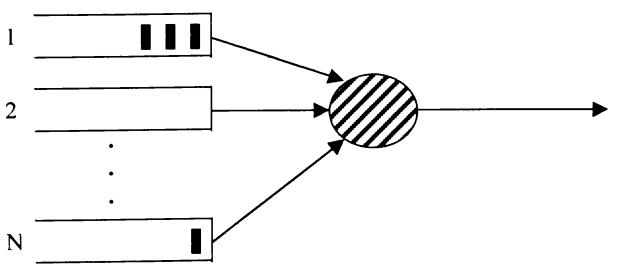


Fig. 4.4 Planificarea la comutatoare TSA

BUPT

Implementarea TSA poate fi făcută prin hardware sau software. În rețelele ATM, unde pachetele sunt celule de dimensiune redusă şi fixă, se practică o implementare prin hardware. În rețelele cu comutare de pachete cu lungimi mai mari de pachete, ca de obicei in Internet, algoritmul poate fi implementat prin software.

Pentru alocarea benzii şi planificarea transmisiei la comutatoarele OBS sunt cunoscute o serie de discipline [ZK]. Planificatoarele pot fi, în general, cu sau fără conservarea lucrului. Un planificator cu conservarea lucrului nu este inactiv niciodată când există vreun pachet memorat în sistem, dar planificatorul fără conservarea lucrului poate rămâne inactiv în acest caz. Un server poate amâna transmisia unui pachet când aşteaptă sosirea iminentă a unui pachet de prioritate mai mare, chiar dacă starea sa curentă e inactivă. Pentru rețelele ATM unde timpul de transmisie a unui pachet este foarte mic, acest mod de lucru nu se justifică. Algoritmul fără conservarea lucrului se mai poate folosi pentru controlul jitter-ului întârzierii prin întârzierea pachetelor sosite prea de vreme. Serverele cu conservarea lucrului au întotdeauna întârzieri medii de vârf mai reduse decât cele fără conservarea lucrului. Dintre planificatoarele cu conservarea lucrului fac parte: GPS (Genelalized Processor Sharring) [Kes, PG1, PG2, DKS, BT, XL, SV, VS, BZ, GV, GoV, ZK, DKS], VCL (Virtual Clock), FQ (Fair Queuing), WFQ (Weighted Fair Queuing), DEDD (Delay Earliest Due Date), WRR (Weighted Round Robin) şi DRR (Deficit Round Robin). Ca planificatoare fără conservarea lucrului pot fi menționate: HRR (Hierarchical Round Robin), SGQ (Stop-and-Go Queuing), şi JEDD (Jitter Earliest-Due-Date).

Altă clasificare a planificatoarelor poate fi făcută după structura internă şi anume: TSA cu priorități sortate, respectiv TSA bazate pe cadre. La TSA cu priorități sortate există o variabilă globală, timpul virtual, asociată fiecărei legături de ieşire a comutatorului. Variabila este actualizată de fiecare dată când un pachet soseşte sau este servit. Fiecărui pachet i se asociază o etichetă de timp, calculată ca funcție de această variabilă. Pachetele sunt sortate funcție de eticheta lor de timp şi transmise în această ordine. Complexitatea implementării algoritmilor cu priorități sortate depinde de doi factori: i) complexitatea actualizărilor listelor de prioritate şi selectarea pachetului de prioritate maximă, care este de cel puțin $O(\log V)$, V fiind numărul de conexiuni care partajează linia de ieşire, şi ii) complexitatea calculării etichetelor de timp ale fiecărui pachet, care depinde de fiecare algoritm. De exemplu, menținerea timpului virtual în WFQ implică procesarea a maxim V evenimente pe durata de transmisie a unui singur pachet, în timp ce calculul etichetelor de timp la algoritmul ceasului virtual VCL, poate fi făcut într-un timp $O(1)$.

La planificatoarele bazate pe cadre, timpul e divizat în cadre de lungime fixă sau variabilă. Rezervarea sesiunii se face în sensul volumului maxim de trafic care i se permite sesiunii să-l transmită în timpul unei perioade. Planificatoarele HRR şi SGQ folosesc cadre de lungime constantă. În consecință serverul poate rămâne inactiv, dacă sesiunea transmite trafic mai redus decât rezervarea făcută pentru acel cadru. În contrast WRR şi DRR permit o variație,

BUPT

până la o valoare maximă, a dimensiunii cadrului. Astfel că, dacă volumul de trafic este inferior celui rezervat, poate fi pornit un cadru nou mai repede. Ambele planificatoare sunt deci cu conservarea lucrului.

Garantarea serviciilor implică protejarea clienților de către rețea, față de comportarea incorectă a unor surse şi față de fluctuațiile rețelei. Sursele cu comportare incorectă pot transmite pachete la o rată mai mare decât cea alocată, iar fluctuațiile rețelei pot genera o rată de sosire instantanee mai mare pe un canal la un comutator oarecare, chiar dacă la intrarea în rețea canalul satisface constrângerile alocării de bandă. Distorsionarea traficului de către fluctuațiile rețelei sugerează ca protecția să fie implementată în interiorul rețelei folosind o disciplină de serviciu bazată pe rata [ZK]. Asemenea discipline asigură protecția prin alocarea unei rate de serviciu garantată pentru fiecare canal, indiferent de comportarea celorlalte canale.

Disciplinele de serviciu bazate pe rată pot fi clasificate în două categorii, în funcție de politica adoptată: i) disciplinele de serviciu cu alocarea ratei care servesc pachetele care au o rată mai mare, atâta timp cât nu afectează performanța celorlalte canale. Acestea sunt discipline cu conservarea lucrului, dintre care pot fi amintite Delay-EDD, Virtual Clock şi Fair Queuing, şi ii) disciplinele de serviciu cu rată controlată, care nu servesc sub nici un motiv pachete cu rate mai mari. Ca exemple sunt Stop-and-Go, Jitter-EDD, HRR. Toate aceste discipline sunt fără conservarea lucrului, ceea ce nu este o coincidență: doar disciplinele fără conservarea lucrului pot oferi o margine superioară a ratei serviciului pe canal.

4.2.1 Planificatoare reprezentative

În literatură au fost propuse un număr mare de TSA. În acest paragraf vor fi prezentate cele mai importante planificatoare şi proprietățile lor.

4.2.1.1 Primul sosit primul servit, FCFS

Cea mai simplă disciplină de servire având o coadă cu priorități sortate este disciplina FCFS (First-Come First-Served) sau FIFO (First-In First-Out). Ordinea sosirii pachetelor determină complet ordinea servirii. Eticheta de timp asignată fiecărui canal este chiar timpul de sosire a pachetului, iar pachetele sunt servite conform timpului lor de sosire. Marele dezavantaj al FCFS este că nu oferă nici un fel de izolare între canale. Dacă unul din canalele virtuale prezintă rafale mari, el va determina creşterea întârzierii pentru celelalte surse, afectându-le astfel calitatea serviciului. FCFS nu poate oferi nici o garanție despre întârzierea limită sau abaterea acesteia, independent de starea rețelei. O sursă care trimite pachete cu viteză mare, poate monopoliza o parte arbitrară din lățimea de bandă a liniei de ieşire.

BUPT

4.2.1.2 Servirea imparțială, FQ

Ideea de bază la FQ este simplă [Kes]: dacă există *N* canale care partajează o linie de ieşire, atunci fiecare va primi *1/N* din lățimea de bandă a acestuia, cu rezerva că dacă vreun canal utilizează mai puțină bandă decât cea alocată, diferența este egal distribuită celorlalte. Acest lucru ar putea fi obținut printr-o explorare carusel bit cu bit a canalelor BR (Bit-by-bit Round-Robin), ceea ce e total nepractic, deci FQ, doar încearcă să emuleze BR. Fiecărui pachet i se asociază un număr de sfârşit, care corespunde numărului trecerii în care ar fi servit, dacă serverul ar rula BR. Apoi pachetele sunt servite în ordinea numărului de sfârşit, ceea ce face ca FQ să emuleze BR [ZK]. Dacă se acordă ponderi canalelor, li se pot aloca părți diferite de bandă; ponderea corespunde numărului de biți primiți de canal într-o trecere dacă s-ar rula serviciul BR. În acest caz se obține WFQ (vezi paragraful 4.2.1.4.).

La viteze mari implementarea lui FQ este dificilă deoarece pretinde *0(log(n))* operații pe pachet, unde *n* este numărul de fluxuri conectate (backlogged) care concurează în comutator. Un flux se numeşte conectat pe o perioadă *T* a unei execuții, dacă pe acea perioadă coada nu este niciodată vidă.

4.2.1.3 Servirea imparțială stohastică, SFQ

Variantă stohastică a servirii imparțiale a fost propusă de McKenney [SV] pentru a elimina deficitele schemei propusă de Nagle, care a propus ca alocarea într-un comutator alocarea benzii liniei de ieşire fluxurilor de intrare să se facă în funcție de flux, şi nu după topologie (vezi figura 3.2). În alocarea după topologie, banda primită de fluxuri scade exponențial cu numărul de noduri traversate. Dacă *W* e lățimea liniei de ieşire a comutatorului *C3*, fluxul *F4* va primi *W/2*, *F3* va primi *W/4* iar *F1* şi *F2* câte *W/8*. Apoi Nagle a propus ca fiecare flux posibil să aibă o coadă separată pentru pachetele sale, explorate RR. Dezavantajul schemei este că nu se ține cont de lungimea pachetelor, ceea ce poate duce la alocări de bandă mai avantajoase fluxurilor cu pachete de dimensiune mare.

În servirea imparțială stohastică SFQ (Stochastic Fair Queuing) se foloseşte rafinarea şi înlănțuirea pentru asocierea identificatorilor de pachete cu cozile corespunzătoare lor. Deşi ar trebui să existe teoretic câte o coadă pentru fiecare flux posibil, McKenney sugerează că sunt suficiente mai puține, deoarece toate fluxurile care sunt rafinate în aceeaşi găleată sunt tratate la fel. Numărul de cozi trebuie să fie un multiplu redus al numărului fluxurilor active. Scade şi complexitatea calculului rafinării, față de *0(1)* actualmente. Dezavantajul este tratarea neimparțială a fluxurilor care intră în coliziune. Garantarea imparțialității este probabilistică. Dacă indexul de rafinare este suficient de mare față de numărul de fluxuri active prin comutator, probabilitatea non-imparțialității e mică. Servirea cozilor se face RR, fără a ține cont de lungimea pachetelor. Dacă nu există tampoane libere pentru memorarea

BUPT

pachetului, ultimul pachet al celei mai lungi cozi e distrus. Implementarea "furtului" de tampoane se face într-un timp *0(1)*, folosind tehnica găleții de sortare. Contribuțiile majore ale autorului sunt: algoritmul furtului de tampoane şi ideea folosirii rafinării şi ignorarea coliziunilor.

4.2.1.4 Ceasul virtual, VCL

Disciplina VCL încearcă să emuleze TDM, la fel cum FQ emulează BR [XL]. Fiecărui pachet i se asignează un timp virtual de transmisie, timpul la care pachetul ar fi transmis dacă ar rula TDM; într-un exemplu simplificat, dacă unui client i se alocă o rată a serviciului de 5 pachete / secundă, pachetelor de intrare de la acel client li se pune eticheta de timp de 0,2 secunde fiecăruia. Prin transmiterea pachetelor în ordinea timpului virtual, VCL emulează TDM.

Timpul virtual de transmisie se determină prin măsurarea ratei de sosire a pachetelor precedente şi cu rata medie de sosire specificată de utilizator. Eticheta TS_i^k este asociată cu pachetul k din conexiunea i. Dacă pentru această conexiune i, timpul real de sosire este AT, a pachetului de dimensiune L_i^k, şi ρ_i este rata rezervată conexiunii i, atunci eticheta de timp asociată acestui pachet va fi:

$$TS_i^k \leftarrow Max\left(AT, TS_i^{k-1}\right) + \frac{L_i^k}{\rho_i}$$

Dacă pachetul a sosit mai târziu decât era aşteptat el va fi lăsat după un timp maxim L_i^k/ρ_i. Dacă el soseşte mai devreme, în cel mai rău caz el va fi lăsat la momentul $TS_i^{k-1} + L_i^k/\rho_i$. Cazul cel mai defavorabil perceput de o aplicație nu va fi influențat de comportarea celorlalte conexiuni.

4.2.1.5 Procesorul generalizat, GPS

Partajarea cu procesorul generalizat GPS (Generalized Processor Sharing), este o disciplină de planificare ideală (fig. 4.5). Multiplexarea GPS este definită ținând cont de modelul de tip fluid, în care pachetele sunt considerate infinit divizibile. Partajarea benzii rezervată de sesiune i este reprezentată de numărul real ϕ_i. Fie $B(\tau, t)$ setul de conexiuni existente în

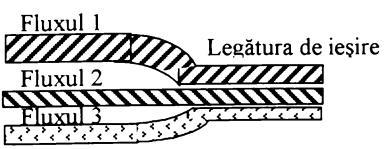


Fig. 4.5 GPS presupune divizibilitatea infinită a pachetelor. La fiecare moment pot fi servite conexiuni multiple proporțional cu rezervarea

BUPT

intervalul $(\tau, t]$. Dacă r este rata serverului, serviciul $W_i(\tau, t)$oferit conexiunii i care aparţine de $B(\tau, t)$ este proporţional cu ϕ_i. Aceasta înseamnă că:

$$W_i(\tau, t) \geq \frac{\phi_i}{\sum_{j \in B(\tau,t)} \phi_j} r(t - \tau). \tag{4.2}$$

Serviciul minim pe care această conexiune îl poate primi în orice interval de timp este:

$$\frac{\phi_i}{\sum_{j=1} \phi_j} r(t - \tau), \tag{4.3}$$

unde V este numărul maxim de conexiuni care pot exista în server în acelaşi timp. Astfel, GPS serveşte fiecare conexiune la o rată minimă cel puţin egală cu rata sa rezervată; în plus, excesul de bandă disponibilă de la sesiunile care nu-şi folosesc rezervările este distribuit celorlalte conexiuni, proporţional cu rezervările lor. Efectul este o izolare perfectă, o impărţialitate ideală şi întârzieri cap la cap scăzute ale sesiunilor.

Versiunea pachet cu pachet a algoritmului, cunoscută ca PGPS sau alocarea impărţială ponderată WFQ (Weighted Fair Queuing) a fost definită în termenii ceasului care este crescut cu o rată egală cu

$$\frac{1}{\sum_{i \in B(\tau,t)} \phi_i}. \tag{4.4}$$

Sistemul GPS este stimulat cu cel pachet cu pachet pentru a identifica setul de conexiuni conectate la fiecare moment. Timpul virtual $v(t)$ este o funcţie liniară de timpul real t, şi panta sa se schimbă în funcţie de numărul de sesiuni ocupate şi ratele lor de serviciu. La sosirea unui nou pachet, timpul virtual trebuie calculat primul. Atunci se calculează eticheta de timp asociată pachetului k a canalului virtual i astfel:

$$TS_i^k \leftarrow Max\left(TS_i^{k-1}, v(t)\right) + \frac{L_i^k}{\phi_i}, \tag{4.5}$$

unde L_i^k este dimensiunea pachetului k. Recent a fost propus o variantă îmbunătăţită a lui WFQ şi anume cazul cel mai defavorabil al servirii impărţiale ponderate W^{2}FQ (Worst – case WFQ) pentru a creşte şi mai mult impărţialitatea algoritmului.

În ambele abordări, maxim V evenimente pot fi comutate în simulatorul GPS pe durata transmisiei unui pachet. Astfel, procesul de luare a unei decizii este $O(V)$, ceea ce face ca implementarea algoritmului să fie costisitoare şi în cele mai multe cazuri prohibitivă.

BUPT

4.2.1.6 Servirea imparțială cu autosincronizarea, SCFQ

Pentru a reduce complexitatea lui WFQ, a fost impusă o implementare aproximativă, denumită SCFQ (Self Clocked Fair Queueing). În această implementare, eticheta de timp la sosirea unui pachet este calculată în funcție de pachetul curent aflat în serviciu. Astfel, dacă TS_{crt} indică timpul pachetului în serviciu, şi dacă noul pachet este pachetul k al sesiunii i, eticheta de timp a noului pachet se calculează astfel:

$$TS_i^k \leftarrow Max\left(TS_{crt}, TS_i^{k-1}\right) + \frac{L_i^k}{\phi_i}. \tag{4.6}$$

Complexitatea algoritmului se reduce mult în acest fel. Prețul plătit constă în reducerea gradului de izolare dintre sesiuni, ceea ce face ca limita întârzierii cap la cap să crească liniar cu numărul de sesiuni care partajează linia de ieşire. Astfel cazul cel mai defavorabil al întârzierii unui sesiuni nu mai poate fi controlat doar prin controlul rezervării sale, ca în PGPS. Întârzierile cap la cap mai mari afectează şi gradul de rafală al sesiunii în rețea crescând necesarul de memorie.

4.2.1.7 Planificarea cu oprire şi pornire, SG

Planificarea cu oprire şi pornire SG (Stop and Go) menține traficul "neted" la traversarea prin rețea [ZK] fiind propusă de Golestani I. Timpul este divizat în cadre de durată T, atât pentru liniile de intrare cât şi pentru liniile de ieşire. Într-un cadru sunt eligibile pentru transmisie doar pachetele sosite în cadrul precedent. Astfel se introduce o întârziere constantă, θ, $0 \le \theta \le T$. Toate pachetele care sosesc într-un cadru şi merg spre o anumită legătură de ieşire sunt întârziate cu θ şi sunt puse în cadrul de plecare corespunzător. Conform disciplinei SG, transmisia unui pachet poate fi amânată până la începutul următorului cadru şi deci algoritmul este fără conservarea lucrului. Dacă numărul maxim de pachete care pot ajunge la o conexiune, în timpul unui cadru nu depăşeşte numărul de diviziuni rezervate din cadru, algoritmul poate asigura o limită pentru întârziere şi pentru jitterul întârzierii. Prima problemă care apare este că din cauză că sistemul este fără conservarea lucrului, nu există câştigul multiplexării statistice. A doua problemă este cea a fineței de alocare a benzii. Deoarece atât întârzierea cât şi jitterul întârzierii sunt proporționale cu dimensiunea cadrului sunt preferate cadrele mici. Totuşi pentru a permite o alocare cât mai fină a benzii, dimensiunea cadrului trebuie să fie mare. SG permite deci lucrul cu dimensiuni multiple de cadre. În interiorul unui cadru, ordinea de servire este arbitrară.

4.2.1.8 Planificarea carusel, RR

La planificatoarele carusel, RR (Round Robin) axa de timp este împărțită în cadre, de dimensiune maximă F. În plus, deoarece fluxurile pot avea cerințe

BUPT

diferite asupra benzii de frecvență, numărul de pachete care pot fi servite într-un cadru este limitat. Fiecărei conexiuni îi este asociat un contor de credite, care este decrementat de fiecare dată când este servit un pachet al acelei conexiuni. La începutul fiecărui cadru numărătorul este inițializat la valoarea traficului maxim pe care îl poate transmite conexiunea în cadrul respectiv; dacă numărătorul de credite este zero, conexiunea nu este eligibilă pentru serviciu. Ca şi în SGQ întârzierea maximă este proporţională cu dimensiunea maximă a cadrului. Necesitatea unui alocări cât mai rafinate a benzii de frecvență face să se adopte cadre de dimensiuni mari şi deci creşte limita întârzierii cap la cap. Planificarea RR poate fi părtinitoare dacă fluxurile folosesc pachete de lungimi diferite.

4.2.1.9 Planificarea carusel cu recuperarea deficitului, DRR

Problema principală a servirii carusel este nonimparțialitatea cauzată de lungimea diferită a pachetelor [SV]. Planificarea carusel cu deficit DRR (Deficit Round Robin) înlătură acest dezavantaj, menținând dezideratul timpului constant. Asigurarea fluxurilor se face ca la SFQ, iar servirea cozilor se face ca la RR, cu un "cuantum" al serviciului asignat fiecărei cozi. Diferența față de RR este că în cazul în care o coadă nu a putut transmite un pachet într-o rundă precedentă din cauza lungimii prea mari, restul cuantumului precedent se adaugă la cuantumul pentru runda următoare. Astfel deficitul va fi recuperat, (vezi figura 4.6). La început, toate variabilele CD (contor deficit) sunt inițializate la

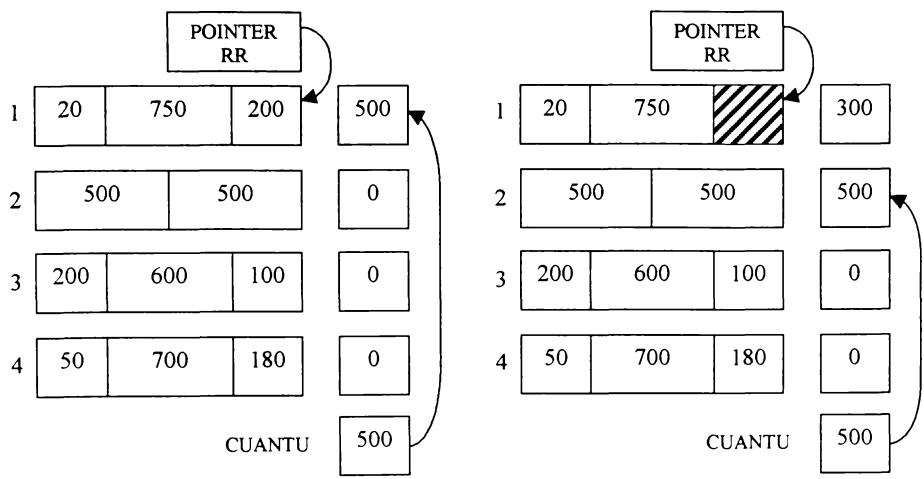


Fig. 4.6 Planificarea carusel cu recuperarea deficitului DRR

valoarea zero. Indicatorul RR indică spre capul listei active. Când prima coadă este servită valoarea 500 a cuantumului este adăugată la valoarea CD, (fig. 4.5a). După transmiterea pachetului cu lungimea 200, primei cozi îi rămân în CD, 300

BUPT

de octeți din cuantum, pe care nu îi poate folosi în runda curentă pentru că următorul pachet este 750 octeți. Astfel, în runda următoare va putea trimite un pachet de 800 de octeți, format din 300 octeți deficitul din runda precedentă şi 500 octeți cuantumul rundei în curs (fig.4.5b).

4.2.1.10 Planificarea carusel ierarhizată, HRR

Serverul HRR (Hierarchical Round Robin) [BZ, LMS] are câteva niveluri de serviciu, iar fiecare nivel realizează un serviciu carusel pentru un număr fix de diviziuni (vezi figura 4.7). Unui canal îi sunt alocate un anume număr de diviziuni de serviciu şi serverul baleiază diviziunile fiecărui nivel ciclic, timpul de baleiere per nivel fiind timpul de cadru al acelui nivel. Ideea de bază a

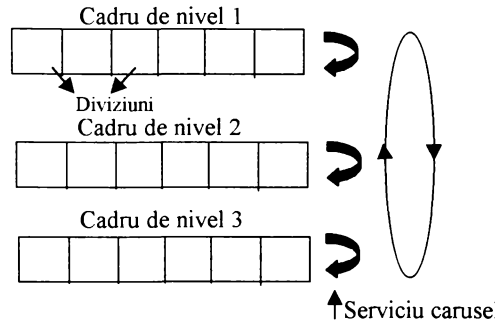


Figura 4.7 Cadre HRR

HRR este alocarea unui părți constante de bandă fiecărui nivel. Nivelurile superioare primesc o bandă de frecvențe mai mare decât cele inferioare, astfel că timpul de cadru alocat nivelurilor superioare este mai mic decât timpul de cadru al nivelurilor inferioare. Deoarece serverul realizează întotdeauna un ciclu complet de parcurgere al diviziunilor într-un cadru el poate furniza o limită a întârzierii canalelor alocate acelui nivel.

4.2.1.11 Planificarea cu întârzierea datei planificată anterior, DEDD

În planificarea clasică EDD (Earliest Due Date) fiecărui pachet îi este asignat un timp de expirare iar pachetele sunt transmise în ordinea crescătoare a acestuia. DEDD (Delay EDD) este o extensie a acesteia, în care serverele negociază câte un contract de serviciu cu fiecare sursă. Contractul stabileşte că dacă sursa respectă o anumită rată maximă şi medie, atunci serverul va furniza o limită a întârzierii. Problema cheie este asignarea timpilor de expirare a pachetelor. Serverul asignează timpul de expirare la timpul la care el ar trebui trimis ca să fie recepționat conform contractului. Acesta este tocmai timpul estimat de sosire însumat cu întârzierea limită din server. De exemplu, dacă un client promite că va transmite pachete la fiecare 0,2 secunde şi limita întârzierii

BUPT

în server este 1 secundă, atunci pachetul k de la client va primi un timp de expirare 0,2 k +1. Dacă banda de frecvențe se rezervă conform ratei maxime de transmitere, DEDD poate asigura o întârziere limită sigură fiecărui canal.

4.2.1.12 Planificarea DEDD cu controlul jitterului, JEDD

Disciplina JEDD (Jitter EDD) extinde DEDD pentru a furniza o limită a jitterului, abaterii, întârzierii, atât pentru întârzierea minimă cât şi pentru cea maximă. După ce pachetul a fost servit la fiecare server, lui i se înscrie diferența dintre timpul de înscriere şi timpul de terminare actual. La intrarea în următorul comutator, un regulator blochează pachetele pe durata acestei diferențe înainte ca el să devină eligibil pentru planificare. Acest lucru furnizează garanția asupra întârzierii minime şi maxime. În figura 4.8 este prezentat modul de progresare al pachetului prin două comutatoare adiacente.

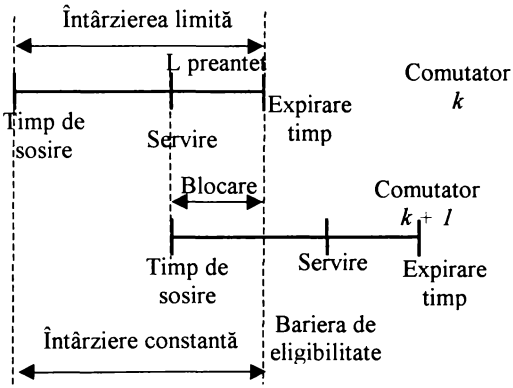


Figura 4.8 Servirea unui pachet la JEDD

4.3 Parametrii planificatoarelor

Există o serie de metode pentru proiectarea planificatoarelor în rețele cu integrarea serviciilor [Ast, BC, CC]. Majoritatea dintre ele pleacă de la ideea că traficul de timp real este prioritar față de traficul de date, aplicând conceptul de priorități statice. Aplicațiile de prioritate mică vor putea folosi resursele rețelei numai dacă nu sunt active aplicații de prioritate ridicată care să solicite acele surse.

În sistemul MARS [SV1] există trei clase de bază pentru trafic şi anume: i) traficul cu întârziere garantată este cel mai prioritar, ii) traficul care pretinde doar o margine pentru pierderea de celule constituie a doua clasă şi iii) traficul de efort maxim, cel mai puțin prioritar, constituie a treia clasă. Sistemul asigură garanții pentru întreaga clasă de trafic nu pentru conexiunile individuale; astfel orice conexiune dintr-o anumită clasă de trafic poate să afecteze performanța

BUPT

celorlalte conexiuni ale clasei. Prin simulare se identifică condiţiile în care sistemul poate garanta o anumită performanţă. Controlul admisiei se face în urma rezultatelor simulărilor, rezultate care depind de caracterizarea traficului. Dar pe măsură ce apar noi aplicaţii, caracterizarea devine imposibil de făcut.

Clark, foloseşte o metodă similară de caracterizare a claselor de trafic, deşi metoda de asigurare a garanţiilor diferă. Sunt tot trei clase de trafic şi anume: i) traficul cu întârziere garantată are prioritate maximă. Se foloseşte aici o planificare imparţială ponderată WFQ (Weighted Fair Queuing) pentru izolare conexiunilor clasei, ii) aplicaţiile adaptive formează clasa următoare care pot folosi doar lăţimea de bandă rămasă de la prima clasă. Controlul admisiei pentru această clasă este extrem de dificil, din cauza priorităţilor statice dintre cele două clase; iii) traficul de efort maxim foloseşte doar lăţimea de bandă rămasă de la primele două.

Soluţia Tenet [KKMMR] introduce trei tipuri de garanţii de QOS şi anume: i) garanţii deterministe pentru clasa cea mai prioritară prin admisia pe baza ratei de vârf şi folosirea disciplinei de planificare DEDD. Implementarea este costisitoare deoarece pretinde remodelarea traficului fiecărei conexiuni în fiecare comutator; ii) garanţii statistice pentru clasa a doua de prioritate. Controlul admisiei se face pe baza ratei de vârf şi ratei medii a conexiunilor, iii) nici o garanţie pentru traficul de efort maxim.

Cea mai generală abordare este cea a lui Sriram [SV1] care propune o arhitectura unificată de planificare, independentă de priorităţile statice. Pentru toate clasele de trafic este folosit un planificator comun, de tip carusel ponderat WRR (Weighted Round Robin). Cele două clase sunt: i) clasa aplicaţiilor cu întârziere garantată încărcate într-o coadă separată a planificatorului RR, care primesc o parte garantată din lăţimea de bandă a liniei de ieşire şi, ii) clasa aplicaţiilor de joasă prioritate, încărcate într-o altă coadă a planificatorului RR. Deşi nu există un mecanism de izolare pentru clasele de trafic de prioritate mai mică, schema nu determină blocarea conexiunilor de prioritate mică, ca în metoda priorităţilor statice. Claselor de prioritate mică li se garantează primirea unei părţi minime din lăţimea de bandă, chiar dacă există în aşteptare pachete cu prioritate ridicată. Totuşi, nu sunt afectate marginile întârzierii şi productivităţii.

Necesitatea alocării unei benzi minime traficului de efort maxim a fost sesizată şi de Clark, precum şi de specificaţiile pentru traficul ABR ale Forumului ATM [SV1, SV]. Unele din metodele precedente presupun lucrul în mod nonconservativ pentru clasa de prioritate maximă, astfel încât să poată fi servită şi clasa de prioritate mică. Deci, algoritmul de planificare trebuie să funcţioneze ca un mecanism central de distribuire a benzii între conexiunile de prioritate maximă cu garantarea întârzierii şi conexiunile de prioritate mai mică.

Clasificările mecanismelor de planificare ţineau cont fie de arhitectura internă a planificatorului fie de modul de lucru conservativ sau nu, ignorând cerinţele reale ale aplicaţiilor. Conform cu aceste cerinţe, câteva caracteristici ale planificatoarelor sunt:

1) Scalabilitatea: algoritmul trebuie să poată lucra bine în comutatoare cu un număr mare de conexiuni şi cu linii cu o gamă extinsă de viteze.

BUPT

2) Întârzierea redusă cap la cap: algoritmul trebuie să poată garanta întârzierea cap la cap pentru sesiunile individuale. În modul de lucru conservativ, întârzieri scăzute cap la cap implică un necesar redus de tampoane de memorie pentru a garanta că nu se pierd pachete. Astfel, selectarea unui anume planificator poate afecta direct costul implementării.
3) Izolarea fluxurilor: algoritmul trebuie să izoleze sesiunea unei aplicații de efectul nedorit al altor sesiuni, care pot avea o comportare necorespunzătoare, pentru a fi capabil să mențină garanțiile asupra QOS. Izolarea este necesară dacă se folosesc mecanisme de urmărire pentru formarea fluxurilor în punctele de intrare în rețea, deoarece fluxurile se pot transforma în rafale în rețea [SSD, RB, LT, LSY]
4) Utilizarea: algoritmul trebuie să folosească eficient lățimea de bandă a liniei, deci trebuie să obțină câştigul maxim prin multiplexarea statistică. El trebuie deci să fie capabil să manevreze eficient sursele în rafală.
5) Imparțialitatea: lățimea de bandă disponibilă a liniei trebuie împărțită echitabil între conexiuni. Algoritmi cu aceleaşi garanții despre întârzierea maximă pot diferi semnificativ în ceea ce priveşte imparțialitatea. Un algoritm neechitabil poate oferi în intervale de timp, rate diferite de serviciu unor conexiuni cu aceeaşi rată rezervată.
6) Simplitatea implementării: algoritmul trebuie să fie uşor de implementat, deoarece în rețelele ATM timpul disponibil pentru luarea şi îndeplinirea unei decizii a planificatorului este extrem de scurt. Aceasta implică o implementare prin hardware. Implementarea prin software ar fi potrivită la viteze scăzute sau dimensiuni mari de pachete, dar decizia de planificare trebuie luată la rate apropiate de rata pachetelor.

Ținând cont de aceste caracteristici pot fi definite trei măsuri de apreciere a algoritmilor de planificare: i) garanțiile asupra întârzierii cap la cap, ii) imparțialitatea şi iii) simplitatea de implementare.

4.3.1 Garanții asupra întârzierii cap la cap

Algoritmul trebuie să garanteze întârzierea cap la cap fără o subutilizare severă a resurselor rețelei. Pentru un algoritm ideal de planificare comportarea întârzierii trebuie să aibă următoarele atribute:

- insensibilitatea la structura traficului altor sesiuni: aceasta este o măsură a gradului de izolare asigurat de planificator sesiunilor individuale;
- întârzierea limită trebuie să fie independentă de numărul de sesiuni care partajează linia de ieşire pentru ca planificatorul să poată lucra şi în comutatoare destinate unui mare număr de fluxuri;

BUPT

- capacitatea de a controla întârzierea limită a sesiunii, controlând doar lățimea de bandă rezervată. Această proprietate a algoritmului poate duce la creşterea flexibilității în realizarea compromisului întârziere-bandă.

După cum se vede sunt trei atribute corelate. Pentru o sesiune, întârzierea în cel mai defavorabil caz, poate diferi mult de la un algoritm de planificare la altul.

4.3.2 Imparțialitate

Algoritmii de planificare capabili să asigure aceleaşi garanții despre întârziere pot avea mari diferențe în comportarea imparțială. Anumite planificatoare pot penaliza sesiunile pentru serviciul primit în exces datorită rezervărilor anterioare, până când alte sesiuni vor primi resursele corespunzătoare serviciului normalizat. Pot deci apare inechități pe intervale mici. Nu s-a ajuns încă la o metodă general acceptată de definire a imparțialității planificatoarelor.

În general, dorim ca sistemul i) să servească întotdeauna conexiunile proporțional cu rezervările, ii) să distribuie banda neutilizată proporțional sesiunilor active, iii) sesiunile să nu fie penalizate pentru banda primită în exces pe timpul inactivității altor sesiuni. După Golestani [SV1] parametrul imparțialitate a algoritmului de planificare apare ca diferență maximă dintre serviciul normalizat primit de două conexiuni pe un interval în care amândouă sunt active. De exemplu algoritmul GPS serveşte fiecare sesiune la rata minimă egală cu cea rezervată, iar excesul de bandă datorat sesiunilor inactive este distribuit celor active proporțional cu rezervările lor. Aceasta duce la o izolare perfectă, imparțialitate ideală şi întârziere scăzută cap la cap.

Algoritmul Virtual Clock [ZK, RF, KKK, GoV, KMR] nu limitează diferența dintre serviciile primite de cele două sesiuni pe un interval mai mic decât perioada de conectare. Acesta este rezultatul planificatorului care încearcă să emuleze TDM static, pe baza medierii ratei serviciilor sesiunilor individuale. La Virtual Clock intervalul de mediere poate fi oricât de lung. Un exemplu tipic de neechitate a acestui planificator este (vezi figura 4.9) în care două conexiuni partajează aceeaşi linie de ieşire şi au alocate părți egale de bandă. Presupunem

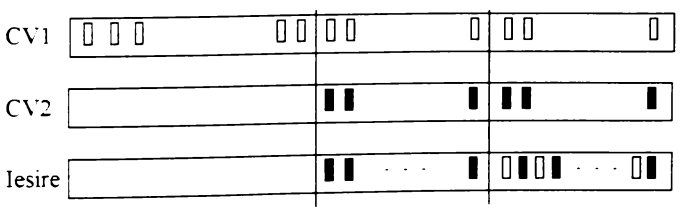


Figura 4.9 Exemplu de comportare neechitabilă a algoritmului Virtual Clock

că fiecare pachet are lungimea unitară şi rata serverului este tot unitară. În intervalul 1-1000 doar CV1 este activ şi transmite 1000 de pachete. La

BUPT

momentul 1000 devine şi CV2 activ şi transmit ambele CV-uri. Planificatorul rulează algoritmul de ceas virtual iar serverul este cu conservarea lucrului, astfel că va servi toate cele 1000 de pachete ale CV1 până la t = 1000, următorul pachet de la CV1 va primi eticheta de timp reflectând media serviciului primită de CV1 până la 1000. Astfel CV1 va fi blocată până la t = 1500, când etichetele de timp ale CV1 şi CV2 devin egale. Dacă nu este limitat gradul maxim de rafală, intervalul în care este interzis serviciul pentru conexiunile active poate creşte la infinit.

Planificatorul GPS e la cealaltă extremă, unde nu se menţine nici o evidenţă a utilizării anterioare a benzii. În exemplul precedent GPS va servi egal ambele CV-uri şi după t = 1000, indiferent că CV1 a primit un serviciu în exces mai înainte. De remarcat este că, conform noii definiţii, este inevitabilă o oarecare inechitate pe termen scurt, la orice planificator lucrând la nivel de pachete, atunci când fiecare pachet este servit. În practică putem pretinde doar ca diferenţa dintre serviciile normalizate primite de două sesiuni să fie constantă şi limitată.

4.3.3 Complexitatea implementării

Algoritmul de planificare trebuie să poată fi implementat hardware, în reţele de mare viteză şi este de dorit să aibă complexitatea de timp independentă de numărul de conexiuni. Dacă numărul de conexiuni care pot partaja linia de ieşire este V, implementarea planificatorului cu priorităţi sortate are trei paşi principali la procesarea celulelor:

1. Calculul etichetei de timp: planificatorul PGPS are cea mai mare complexitate, deoarece în paralel trebuie emulat planificatorul GPS pentru actualizarea timpului virtual. În cel mai rău caz rezultatul simulării poate duce la un antet $O(V)$ al procesului pe transmisia pachetului. Pe de altă parte, atât la VCL cât şi la Self-Clocked FQ, calculul etichetelor de timp necesită doar un număr constant de operaţii, rezultând complexitatea $O(1)$ în cel mai rău caz.
2. Inserarea în lista cu priorităţi sortate: prima celulă a fiecărei cozi a sesiunii trebuie memorată într-o listă de priorităţi sortate. Când o celulă ajunge la o coadă vidă, inserarea sa în lista de priorităţi pretinde $O(\log V)$ paşi;
3. Selecţia celulelor cu eticheta minimă de timp pentru transmisie: deoarece celulele sunt memorate într-o structură de priorităţi sortate regăsirea celulei cu prioritate maximă se face într-un timp $O(\log V)$.

Ultima operaţie este identică pentru orice arhitectură cu priorităţi sortate. A fost propusă o implementare paralelă a acestei operaţii cu o complexitate $O(1)$ de timp, utilizând un set de $O(V)$ de elemente de procesare.

Algoritmii bazaţi pe cadre ca WRR şi DRR pot fi implementaţi într-un timp $O(1)$, fără nici un calcul de etichetă de timp. Dar aceşti algoritmi duc la o întârziere care creşte liniar cu numărul de sesiuni care partajează linia de ieşire.

BUPT

Practic, algoritmii de planificare trebuie să realizeze un compromis între complexitatea implementării şi celelalte proprietăți dorite ca întârziere scăzută şi inechitate pe termen scurt limitată.

BUPT

Capitolul 5

ALGORITMI DE IMPLEMENTARE A REZERVĂRILOR DE BANDĂ ÎN COMUTATOARE CU MEMORARE LA INTRARE

Problema planificării pachetelor în comutatoarele cu memorare la intrare poate fi simplificată divizând-o în două: i) planificarea transmisiei între porturile de intrare *PI* spre acelaşi port de ieşire *PE*, ii) respectiv planificarea pachetului pentru *PI* ales [AOST]. În cele ce urmează va fi tratată doar prima problemă a implementării rezervărilor de bandă între porturile de intrare şi de ieşire, pentru o arhitectură fără blocare, ţinând cont de limitările comutatoarelor cu memorare la intrare. Problema comutării poate fi redusă la cea a asocierii într-un graf bipartit (vezi § 4.1). Dimensiunea fixă a celulelor ATM permite utilizarea algoritmului de asociere pentru planificare unde, controlerul alege un subset de pachete din cozile de intrare, în fiecare ciclu, şi le planifică simultan. Algoritmii optimali existenţi pentru asocierea bipartită au o complexitate de $O(N^2)$, N fiind numărul de porturi ale comutatorului. Practic ei nu pot fi folosiţi în reţele ATM, unde timpul disponibil este cel de transmisie a unei celule. În plus, ei sunt seriali, deci nu pot beneficia de operarea paralelă pentru a le creşte viteza.

5.1 Algoritm paralel de asociere probabilistică iterativă, PIM

Algoritmul paralel de asociere iterativă probabilistică PIM (Probabilistic Iterative Matching), propus de Anderson, Owicki şi Saxe [AOST] oferă o asociere maximală pentru planificator fiind simplu şi uşor de implementat în hardware.

Scopul algoritmului de planificare propus de Anderson este de a găsi rapid pachetele de intrare – ieşire fără conflicte, luând în considerare doar acele perechi care au celule memorate pentru transmisie. Procesul de asociere determină care intrări şi spre ce ieşire vor transmite celule într-o diviziune dată de timp. Se are în vedere, pentru evitarea efectului HAL, un acces aleator la tamponul de intrare, astfel că o intrare poate transmite spre oricare din ieşirile pentru care are celule memorate. Condiţia pusă este însă ca fiecare intrare să poată fi asociată cu cel mult o ieşire, respectiv o ieşire să poată fi asociată cu cel mult o intrare. Algoritmul paralel de asociere iterativă propus în [AOST] foloseşte calculul paralel, accesul aleator şi calculul iterativ pentru a realiza

BUPT

asocierea cât mai eficient. Inițial, toate intrările şi ieşirile sunt nesociative. Se parcurg următorii trei paşi:

i) Fiecare intrare are neasociată transmite câte o cerere spre fiecare ieşire pentru care are celule memorate.
ii) Dacă una din ieşirile neasociate primeşte câteva cereri, ea va alege în mod aleator una dintre ele, spre care îşi transmite acordul.
iii) Dacă o intrare primeşte câteva acorduri alege aleator unul dintre ele şi trimite spre ieşirea respectivă un semnal de acceptare.

Fiecare dintre aceşti paşi se desfăşoară independent şi în paralel pentru fiecare pereche de porturi de intrare – ieşire. Nu există un planificator central. Astfel, la sfârşitul unei iterații a protocolului, vom avea o asociere legală a intrărilor şi a ieşirilor. Deoarece mai multe intrări şi nu una singură poate trimite cereri spre aceeaşi ieşire, în etapa de acord se alege dintre ele, astfel încât fiecare ieşire să fie asociată cu cel mult o intrare. Mai multe ieşiri pot să-şi trimită acordul spre aceeaşi intrare, dacă intrarea a transmis mai multe cereri. În faza de acceptare, această intrare alege unul dintre acordurile primite, astfel încât fiecare să fie asociată cu cel mult o ieşire. Deşi după o iterație asocierea este legală mai pot rămâne intrări neasociate cu celule memorate pentru ieşiri neasociate. O ieşire al cărui acord nu a fost acceptat, rămâne aptă de a fi asociată cu una din intrările care n-au primit un accept. Pentru rezolvarea acestei situații se repetă protocolul de cerere, acord şi acceptare, fără perechile asociate în iterația anterioară. În abordarea propusă efectul HOL nu mai apare, deoarece se consideră la fiecare iterație toate conexiunile posibile.

În fig. 5.1 este prezentată o iterație a algoritmului paralel de asociere iterativă. Sunt emise cinci cereri, din care trei primesc acordul şi două sunt acceptate. Mai departe, la sfârşitul primei iterații, rămâne o cerere, de la intrarea

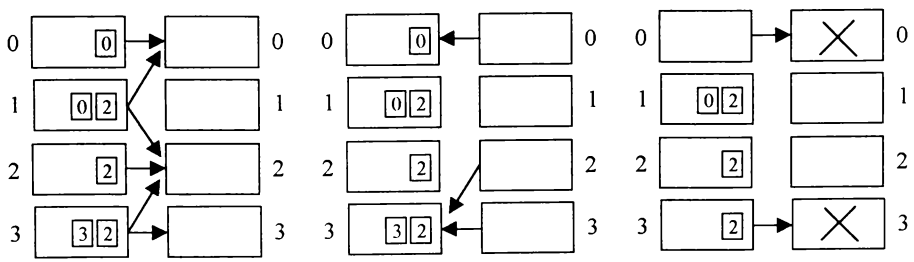


Fig. 5.1 Algoritmul PIM de asociere probabilistic iterativ (0 iterații)

3 la ieşirea 3, neasociată cu ieşirea. Această cerere va fi din nou emisă şi i se va transmite acordul şi acceptarea în a doua iterație, după care nu mai rămâne alte asocieri de făcut. După un număr de iterații, rezultatul asocierii paralele iterative este folosit la inițializarea comutatorului pentru următoarea diviziune de timp. Celulele sunt atunci efectiv transmise prin comutator şi se relansează în lucru algoritmul paralel de asociere iterativă, pentru următoarea diviziune de timp.

BUPT

Algoritmul ia în considerare, pentru asociere, fluxurile care au deja celule memorate precum şi fluxurile ale căror celule au sosit între timp.

Asocierea paralelă iterativă poate fi generalizată pentru operarea în comutatoare pe mai multe niveluri k, ca de exemplu un comutator Batcher – Banyan. Având k niveluri, pentru o singură pereche de intrare – ieşire există k căi distincte prin comutator [Tane]. La un astfel de comutator pot fi livrate aceleaşi ieşiri, într-o singură diviziune de timp, până la k celule (ceea ce evident implică existenţa unor tampoane de memorare la ieşire deoarece o singură celulă poate fi transmisă într-o diviziune). Algoritmul se modifică în acest caz permiţând fiecărei ieşiri să transmită până la k acorduri în planul (2). Modificarea este asemănătoare dacă algoritmul se aplică unor comutatoare care permit transmiterea, într-o diviziune, a mai multor celule nu doar una singură. Problema principală este numărul de iteraţii necesar algoritmului paralel de asociere probabilistică iterativă pentru a-şi încheia lucrul, adică de a ajunge în acel punct în care nu mai există intrări, cu celule memorate, neasociate cu ieşirile corespunzătoare. În cel mai rău caz, pentru un comutator $N \times N$ sunt necesare N iteraţii şi anume în cazul în care toate acordurile se referă la aceeaşi intrare. Dacă situaţia s-ar repeta, asocierea iterativă paralelă nu ar fi mai rapidă decât un algoritm serial de asociere. Pe de altă parte, în cel mai bun caz, acordul fiecărei ieşiri este destinat la câte o intrare distinctă, caz în care algoritmul are nevoie doar de o iteraţie pentru a-şi încheia lucrul.

Pentru a evita cazul cel mai defavorabil se ataşează [AOST] fiecărei ieşiri câte un generator independent de numere aleatoare. Ieşirile aleg care intrări să-i trimită acordul, conform cu aceste generatoare, ceea ce face puţin probabilă apariţia cazului cel mai defavorabil. Folosind aleatorizarea acordurilor algoritmul are nevoie în medie de $0(\log N)$ iteraţii, deoarece în fiecare iteraţie se rezolvă în medie ¾ din numărul de cereri nerezolvate rămase. Astfel dacă un comutator 16 × 16, va rula patru iteraţii, numărul de cereri rezolvate este cuprins între (99,9 ÷ 100) %, în funcţie de încărcarea cu trafic.

Tabelul 5.1 Procentul de asocieri totale găsite în k iteraţii pentru sarcină uniformă

Pr. {intrarea i să aibă o celulă pentru ieşirea j}	Număr de iteraţii (K)			
	1	2	3	4
0,10	87 %	99,8 %	100 %	--
0,25	75 %	97,6 %	99,97 %	100 %
0,50	69 %	93 %	99,6 %	99,997 %
0,75	66 %	90 %	98,6 %	99,97 %
1,00	64 %	88 %	97 %	99,9 %

În tabelul 5.1 [AOST] sunt prezentate rezultatele simulărilor folosind algoritmul PIM, efectuate pentru a determina numărul practic de iteraţii necesar, cu diferite structuri de cereri. Practic se preferă ca algoritmul să lucreze timp de patru iteraţii, decât până la epuizarea sigură a numărului de cereri rămase nerezolvate.

BUPT

Pentru evaluarea performanţelor algoritmului paralel de asociere iterativă a fost făcută comparaţia cu servirea FIFO, respectiv cu servirea cozilor de memorare de la ieşirea comutatoarelor, OB, servire considerată perfectă POQ (Perfect Output Queuing). Tehnica FIFO este uşor de implementat, dar are performanţe scăzute; tehnica POQ nu se pretează la implementare chiar pentru comutatoare de dimensiuni moderate la viteze de Gbps, dar indică performanţa optimă a unui comutator cu resurse hardware nelimitate. La sarcină mică (fig. 5.2) cei trei algoritmi au aproximativ aceeaşi performanţă, deci, cu excepţia costului implementării nu contează care dintre ei este folosit. La încărcarea medie, tehnica FIFO conduce deja la întârzieri importante, din cauza efectului

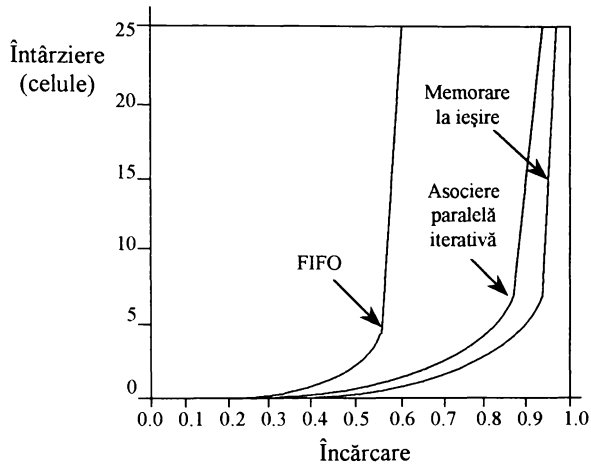


Fig. 5.2 Performanţele unor algoritmi de planificare, la încărcarea uniformă

HOL; ceilalţi doi algoritmi au întârzieri destul de apropiate, ceva mai mare pentru asocierea paralelă iterativă. Diferenţa apare deoarece în cazul memorării la ieşire, o celulă memorată este întârziată doar de celelalte celule ale aceleaşi ieşiri. În asocierea paralelă iterativă o celulă memorată trebuie să concureze pentru linie atât cu celulele memorate la aceeaşi intrare cât şi cu celulele destinate aceleaşi ieşiri. În sarcină mare performanţa celor doi algoritmi este foarte apropiată. Chiar şi la sarcină mare, întârzierea de memorare a algoritmului PIM este rezonabilă, sub 13 μs la o utilizare de 95% a liniilor.

5.2 Numărul de iteraţii al algoritmului PIM

Pentru un comutator $N \times N$, algoritmul PIM ajunge la asocierea maximală în $O(\log N)$ iteraţii în medie; această margine este independentă de structura cererilor. Ideea de bază a demonstraţiei constă în observaţia că dacă o ieşire neasociată primeşte o cerere, o iteraţie a algoritmului va consta fie din: i) asocierea ieşirii cu una din intrările care i-au emis cereri, ii) fie din asocierea

BUPT

celor mai multe intrări care au emis cereri spre acea ieşire sau spre alte ieşiri. Rezultatul este că fiecare iteraţie reduce, în medie, numărul de cereri de ¾ ori minim. O cerere este nerezolvată dacă atât portul său de intrare cât şi cel de ieşire rămân neasociate. Considerăm separat cererile spre fiecare ieşire. Presupunem că o ieşire Q primeşte cereri de la n intrări într-o iteraţie anumită. O parte din cele n intrări vor emite cereri şi primi acorduri de la alte ieşiri decât Q, iar restul nu vor primi acorduri de la ieşirile respective. Fie k numărul de intrări care sunt cereri spre Q şi nu primesc alte accepturi. Q alege aleator una din cele n intrări pentru ai trimite un acord. Deoarece Q alege echiprobabil dintre cererile intrărilor şi alegerea lui Q este independentă de alegerile făcute de celelalte ieşiri probabilitatea ca Q să accepte una din ieşirile intrărilor care nu au concurat pentru o altă ieşire este k/n. În acest caz, acordul lui Q va fi acceptat şi ca rezultat toate cele n cereri spre Q vor fi rezolvate: una va fi acceptată, iar restul nu vor fi niciodată acceptate.

Pe de altă parte, cu probabilitatea $1 - (k/n)$, Q va trimite un acord spre o intrare care mai primeşte acorduri şi de la alte intrări. Dacă intrarea alege intrarea lui Q pentru acceptare, toate cererile spre Q au fost rezolvate. Dar chiar dacă acordul lui Q nu este acceptat, toate cele $n - k$ intrări care au primit un acord vor fi asociate (la alte ieşiri) în timpul acestei iteraţii; nici una din cele $n - k$ cereri spre Q nu rămân pentru iteraţia următoare.

Astfel, cu probabilitatea k / n toate cererile spre Q sunt rezolvate, iar cu probabilitatea $1 - (k / n)$ cel mult k cereri rămân nerezolvate. Ca rezultat, numărul mediu de cereri nerezolvate spre Q este cel mult $(1 - (k / n)) \times k$ care nu este mai mare ca $n / 4$ pentru orice k. Cum se începe cu cel mult N^2 cereri, aceasta duce la consecinţa că după a i-a iteraţie, numărul de cereri nerezolvate este cel mult $N^2/4^i$.

Rămâne de arătat că algoritmul atinge asocierea maximală, în medie în $O(\log N)$ paşi. Fie C pasul în care şi ultima cerere este rezolvată. Atunci, valoarea estimată a lui C este:

$$E[C] = \sum_{i=1}^{\infty} i \, Pr(C = i) \tag{5.1}$$

sau

$$E[C] = \sum_{i=0}^{\infty} i \, Pr(C > i) \tag{5.2}$$

Dar probabilitatea ca $C \quad i$ este tocmai probabilitatea ca cel mult o asociere să rămână nerezolvată la sfârşitul celei de-a i-a iteraţii,

$$Pr\{C > i\} = \sum_{j=1}^{\infty} Pr\{\text{să rămână } \mathbf{j} \text{ cereri după } \mathbf{i} \text{ iteraţii}\} \tag{5.3}$$

Înlocuind suma cu o sumă evident mai mare, rezultă:

$$Pr\{C > i\} = \sum_{j=1}^{\infty} j \, Pr\{\text{să rămână } \mathbf{j} \text{ cereri după } \mathbf{i} \text{ iteraţii}\} \le \frac{N^2}{4^i}, \tag{5.4}$$

BUPT

ultima inegalitate provenind din marginea dedusă anterior pentru numărul mediu de asocieri nerezolvate după a *i*-a iterație.

Deoarece probabilitatea nu poate fi niciodată supraunitară, $Pr\{C > i\} \le min(1, N^2/4^i)$. Substituind în (5.2), rezultă

$$E\{C\} \le \sum_{i=0}^{\infty} min\left(1, \frac{N^2}{4^i}\right), \tag{5.5}$$

Cum $N^2 = 4^i$ când $log_2 N = i$, suma nu are mai mult de $log_2 N$ termeni cu valoarea 1, restul sumei fiind o serie de puteri mărginită prin 4/3. astfel

$$E[C] = log_2 N + \frac{4}{3}. \tag{5.6}$$

Algoritmul poate furniza doar o limită dar este foarte eficient în menținerea liniei de ieşire în stare de utilizare. Nu are însă nici un mecanism de alocare a benzii unei linii de ieşire între liniile de intrare cu trafic destinat ei. El poate doar garanta că fluxurile celor *N* intrări care concurează pentru o linie de ieşire vor primi câte *1/N* din banda acelei linii. Nu permite o alocare flexibilă a benzii şi nu asigură imparțialitatea între *PI*-urile care transmit spre acelaşi *PE*, când solicitările lor de bandă nu sunt identice.

Pentru acest algoritm au fost propuse două alternative pentru a putea asigura garanții asupra benzii. Prima variantă se referă la traficul CBR, şi are în vedere o planificare explicită a transmisiei pachetelor între perechile de intrare-ieşire, ținând cont de cererile de bandă de frecvență ale fiecărui flux. Problem care apare este însă că traficul de timp real nu este mereu la fel cu cel CBR, ci prezintă deseori rafale. În plus, complexitatea calculului perechilor de intrare-ieşire nu permit schimbări prea dese în cererile de bandă ale fluxurilor. A doua variantă numită asociere statică permite o alocare mai flexibilă a benzii perechilor de intrare-ieşire în competiție. Aici procesul de asociere este declanşat de *PE*-uri. Fiecare *PE* generează un semnal de acceptare pentru selectarea aleatoare a unui *PI*, cu probabilitatea dată de banda rezervată de *PI*. Fiecare *PI* care primeşte semnale de acceptare, selectează unul dintre accepturile ieşirilor, după ce a ponderat fiecare accept cu o distribuție de probabilitate. Cum procesul de asociere este inițiat de *PE*-uri, este posibilă selecția unu *PI* care nu are pachete de transmis, ceea ce limitează eficiența algoritmului la 72% din capacitatea liniei de ieşire. În plus, algoritmul pretinde calculul câtorva distribuții de probabilitate la fiecare pas, ceea ce face dificilă implementarea hardware.

BUPT

5.3 Asocierea probabilistică cu filtrarea cererilor, APFC

În cele ce urmează se propune un nou algoritm, pe care l-am denumit "asociere probabilistică cu filtrarea cererilor" APFC, care permite o alocare flexibilă a lățimii de bandă între intrările comutatorului care partajează aceeaşi linie de ieşire. APFC alocă intrărilor de bandă legătura de ieşire, conform cu rezervările făcute în faza de control a admisiei şi poate garanta că traficul fiecărei intrări îşi primeşte partea din banda legăturii de ieşire. Algoritmul asigură şi o izolare între fluxurile sosite la două *PI* distincte şi dirijate spre aceeaşi legătură de ieşire; astfel comportarea incorectă a unor surse nu va afecta celelalte fluxuri. Lățimea de bandă care poate fi alocată fluxurilor de timp-real e limitată doar de algoritmul în sine şi poate ajunge la 85-95%. Hardware-ul necesar implementării este modest, mult mai redus decât cel necesar planificării precalculate şi asocierii statistice. Garanțiile asigurate de algoritm sunt probabilistice, ca şi în cazul asocierii iterative probabilistice.

Considerăm un comutator fără blocare cu memorarea pachetelor de intrare, cu N porturi. Fiecare legătură de intrare sau ieşire poate fi partajată de mai multe fluxuri. Un pachet care soseşte la un *PI* poate aparține unuia din fluxurile care partajează respectiva legătură de intrare; pachetul este dirijat de comutator spre *PE* corespunzător respectivului flux, unde este multiplexat cu alte fluxuri partajând aceeaşi legătură de ieşire. În general, aceeaşi pereche de porturi sursă destinație poate fi partajată de mai multe fluxuri. Fluxurile care ajung la acelaşi *PI* şi sunt destinate la acelaşi *PE* vor fi denumite conexiune. Planificatorul nu poate face distincție, datorită schemei noastre de implementare a rezervărilor de bandă, între fluxurile individuale aparținând aceleaşi conexiuni.

Alocarea lățimii de bandă la nivelul fluxului se poate face combinând algoritmul APFC propus cu un algoritm de planificare pentru alocarea benzii agregate a legăturii, între fluxurile care partajează legătura. Astfel planificarea poate fi împărțită în două niveluri:

1 – planificarea la nivel de conexiune. Fiecare *PE* a comutatorului selectează *PI* de servit, în funcție de rezervările de bandă de nivel de conexiune.

2 – planificarea la nivel de flux. Fiecare *PI* selectează un pachet de la unul din fluxurile de intrare partajând aceeaşi conexiune intrare-ieşire, *I-E* în funcție de partea ce-i revine din banda totală de frecvențe.

APFC rezolvă prima problemă prin alocarea explicită a unei părți din banda fiecărei legături de ieşire, setului de fluxuri destinate ei de la fiecare legătură de intrare. Dacă cererea totală de bandă a fluxurilor partajând aceeaşi conexiune de *I-E* nu depăşeşte rezervarea, atunci garanțiile lor referitoare la bandă pot fi satisfăcute independent de cererile de bandă ale altor conexiuni de *I-E*. Rolul APFC este cu alte cuvinte, să aloce resursele comutatorului într-o manieră corectă, iar nivelul doi al planificării trebuie folosit pentru alocarea benzii legăturii de ieşire între fluxurile concurente.

BUPT

Pentru implementarea rezervărilor, împărţim axa de timp în cadre, fiecare având un număr fix de diviziuni. La reţelele ATM o diviziune corespunde timpului de transmisie a unei celule. Rezervarea de bandă pentru fiecare conexiune de *I-E* se face prin alocarea fiecăreia, a unui număr minim de diviziuni din cadru pe care le vom numi creditele conexiunii. Fiecare conexiune de *I-E* solicită un număr de credite corespunzător cu partea de bandă dorită a legăturii de ieşire. Cererea este acceptată dacă totalul creditelor solicitate de toate conexiunile de *I-E* care partajează linia de ieşire este mai mic decât lungimea cadrului. În practică, va fi luată în considerare acea eficienţă a algoritmului de asociere probabilistică, care conduce la o utilizare a liniei de 85-95% tipic. Dacă c_{ij} este creditul alocat conexiunii de la intrarea i la ieşirea j, şi f este lungimea cadrului, atunci obiectivul algoritmului APFC este să garanteze că în medie, cel puţin c_{ij} pachete pot fi transmise de la portul i la portul j în fiecare cadru.

În continuare vom prezenta operaţiile de bază ale algoritmului APFC. Presupunem că fiecare intrare a comutatorului menţine câte o coadă separată cu pachetele destinate fiecărei legături de ieşire a comutatorului. În cadrul cozii considerăm că pachetele sunt ordonate de un algoritm de planificare le nivel de flux, folosit pentru alocarea benzii fluxurilor individuale care partajează aceeaşi conexiune de *I-E*. Fiecare *PE* trebuie să menţină un contor de pachete servite în cadrul curent de la fiecare intrare. O iteraţie APFC constă din patru etape:

1. **Cererea**: fiecare *PI*, care încă nu a fost asociat unui *PE*, trimite o cerere tuturor *PE*-urilor corespunzătoare destinaţiilor pachetelor din cozile sale.
2. **Filtrarea cererilor**: la recepţia cererilor de la *PI*-uri, fiecare *PE* le trece printr-un filtru care conţine câte un bit asociat fiecărei cereri. Acele *PI*-uri care au de transmis un număr de pachete mai mare sau egal cu numărul de credite au bitul pus pe 1, iar celelalte pus pe 0. În procesul de asociere vor fi luate în considerare doar cererile nefiltrate (bitul pe 0) primite de *PE*, restul sunt ignorate.
3. **Acordul**: *PE* selectează la întâmplare, cu o probabilitate uniformă, una din cererile rămase după filtrare şi transmite un semnal de acord respectivului *PI*.
4. **Acceptarea**: fiecare *PI* neasociat care primeşte unul sau mai multe semnale de acord, selectează unul cu aceeaşi probabilitate şi informează *PE* corespunzător. Porturile de intrare şi de ieşire sunt acum asociate şi pot fi excluse din următoarele iteraţii.

Ciclul este repetat fie de un număr de ori, fie până nu mai există cereri nefiltrate. Deosebirea principală dintre APFC şi PIM este adăugarea etapei de filtrare a cererilor. Deşi etapa de filtrare a cererilor apare separat, ea poate fi realizată împreună cu etapa de acord, în implementarea hardware, astfel ca timpul fiecărei iteraţii să nu crească semnificativ. În figura 5.3 sunt reprezentaţi cei patru paşi ai algoritmului. În etapa de filtrare se presupune că, pentru cadrul curent *PI 1* şi-a depăşit creditele spre *PE 0* şi *PE 2*. De aceea, ambele cereri ale *PI 1* sunt eliminate prin filtrare, ca şi cererea de la *PI 3* spre *PE 3*. În etapa de

BUPT

acord *PE 2* selectează aleator *PI 2*, şi cum acesta nu a primit alte acorduri el acceptă acordul de la *PE 2*.

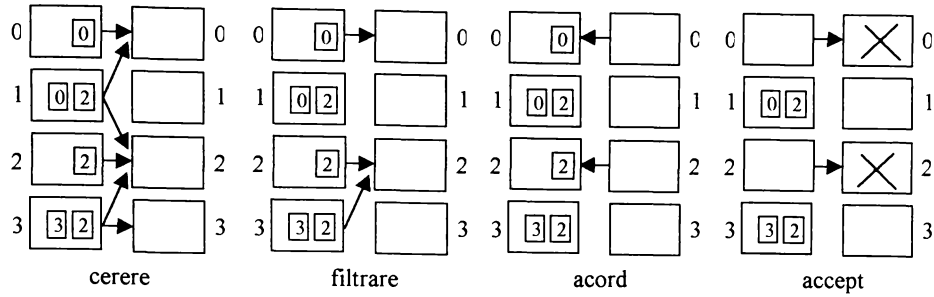


Fig. 5.3 O singură iteraţie a primei faze a algoritmului APFC

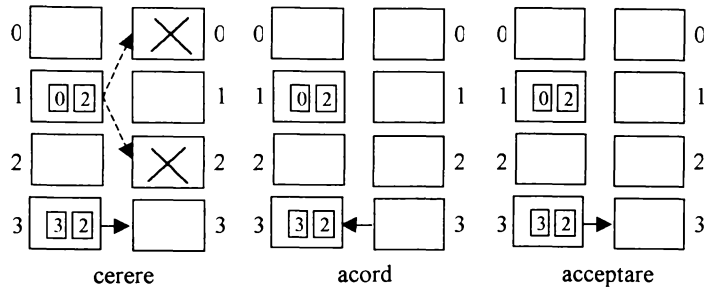


Fig. 5.4 O singură iteraţie a fazei a doua a algoritmului APFC

După ce prima fază a algoritmului este încheiată, este posibil ca anumite perechi *I-E* să rămână neasociate, chiar dacă au pachete disponibile pentru transmisie, deoarece cererile corespunzătoare au fost blocate în etapa de filtrare. Pentru maximizarea numărului de pachete planificate, algoritmul poate fi repetat după punerea pe 0 a tuturor biţilor din filtru, corespunzători *PI* rămase neasociate. Pentru intrările rămase neasociate se aplică algoritmul PIM, pentru alocarea benzii rămase după onorarea rezervărilor de bandă. Revenind la exemplul precedent, în figura 5.4, se vede că acum poate fi satisfăcută cererea de la *PI 3* spre *PE 3*. Cererile spre *PE 0* şi *PE 2* sunt ignorate deoarece aceste ieşiri sunt deja asociate.

Rolul etapei de filtrare este de a bloca cererile iniţiale de intrările care folosesc mai multă bandă decât cea convenită la rezervarea legăturii de ieşire, permiţând astfel celorlalte intrări să-şi obţină partea de bandă cuvenită. Dacă traficul de la unele *PI* spre un *PE* este mai mic decât rezervarea, restul de bandă este alocat egal tuturor conexiunilor spre acel *PE*.

Dacă se alocă c_{ij} credite conexiunii $PI_i - PE_j$, şi cadrul are lungimea f, conexiunii i se alocă o lăţime de bandă:

$$b_{ij} \geq \frac{c_{ij}}{f}, \tag{5.7}$$

BUPT

pentru fiecare i şi j. dacă, în plus, presupunem că o parte a benzii este alocată explicit tuturor conexiunilor de I-E, atunci:

$$\sum_{i=1}^{N} c_{ij} = f . \tag{5.8}$$

Din (5.7) şi (5.8) putem trage concluzia că în suprasarcină

$$b_{ij} = \frac{c_{ij}}{f} . \tag{5.9}$$

Dacă nu se alocă explicit o parte din banda legăturii de ieşire, fiecare intrare îşi va primi banda rezervată şi o parte egală din banda rămasă. Lucrul provine din faptul că atunci când toate creditele sunt satisfăcute, algoritmul APFC se reduce la o asociere iterativă simplă, care împarte egal banda legăturii de ieşire tuturor intrărilor care cer acces la respectivul port de ieşire în acest sens, algoritmul îndeplineşte criteriul de imparţialitate definit în [DKS]. Când o conexiune de I-E îşi creşte viteza traficului, performanţa celorlalte conexiuni va fi la fel diminuată, dar ele vor continua să primească cel puţin rata rezervată lor.

La algoritmul PIM numărul mediu de iteraţii pentru a atinge asocierea maximă este $O(\log_2 N)$, deoarece numărul cererilor nerezolvate, scade în medie cu cel puţin ¾ în fiecare iteraţie. Plecând de la maximum N^2 cereri nerezolvate, numărul mediu de iteraţii până la convergenţa algoritmului este totdeauna mai mic decât $\log_2 N + 4/3$.

Operarea APFC poate fi văzută ca aplicarea de două ori consecutiv a algoritmului PIM. În prima fază algoritmul PIM se aplică cererilor de credite în exces, iar în faza a doua cererilor rămase nerezolvate şi a celor filtrate în prima fază. În prima fază a algoritmului se porneşte cu maximum $N^2/2$ cereri şi convergenţa se va obţine în medie, în mai puţin de $\log_2 N + 7/6$ iteraţii. În faza a doua, cele $N^2/2$ cereri filtrate în prima fază, vor fi activate şi algoritmul va converge din nou, în medie, în mai puţin de $\log_2 N + 7/6$ iteraţii. Deci numărul mediu de iteraţii va fi mai mic de $2\log_2 N + 7/3$.

Îmbunătăţirea numărului mediu de iteraţii se poate face comasând cele două faze ale algoritmului într-una singură; lucrul se poate obţine permiţând fiecărui PE să-şi şteargă toţi biţii din filtru când toate cererile de intrare au fost filtrate şi PE a rămas neasociat. Această modificare nu va afecta alocarea benzii. Filtrele sunt ignorate doar dacă nici una din intrările cu credite în exces n-are pachete de transmis unui anume PE, sau dacă toate aceste intrări au fost deja asociate la alte PE. Cu această modificare, limita superioară a numărului mediu de operaţii necesare pentru obţinerea asocierii maxime se reduce la $\log_2 N + 4/3$, aceeaşi ca la PIM. Simulările pentru APFC ţin cont de această modificare.

Teorema 2.1 Numărul mediu de iteraţii pentru a obţine asocierea maximă cu algoritmul APFC nu depăşeşte $(\log_2 N + 4/3)$.

BUPT

Demonstrația este identică cu cea a lui Anderson [AOST] cu excepția faptului că nu toate cererile emise de *PI* în fiecare ciclu participă la procesul de asociere. Inițial, maximum $N^2/4$ cereri sunt generate de *PI*-uri. O cerere este rezolvată când fie portul de intrare fie cel de ieşire care au emis-o a fost asociat. Algoritmul a ajuns la convergență când toate cererile au fost rezolvate.

Considerăm orice port de ieşire q şi presupunen că acesta primeşte n cereri de la porturile de intrare, într-o iterație oarecare a APFC. Unele din aceste cereri pot fi mascate, dar cu modificarea menționată, se garantează că unul din cele n *PI* care au generat cereri, va primi un acord de la q. Fie acest port p, ales de q pentru a primi un acord. Dacă p nu primeşte acorduri care sunt în competiție, atunci p şi q sunt asociate, şi toate cele n cereri direcționate la q sunt rezolvate. Pe de altă parte, dacă p primeşte una sau mai multe cereri în competiție, atunci q poate rămâne neasociat la sfârşitul acestui ciclu. Presupunem că k din n porturi de intrare care au emis cereri spre q primesc acorduri de la unul sau mai multe alte porturi de ieşire. Deoarece fiecare selectează un *PI* cu o distribuție uniformă, probabilitatea ca p să aparțină setului de k porturi este k/n. aceste k porturi vor fi asociate şi apoi eliminate din procesul de asociere la sfârşitul iterației curente. Astfel, cel mult $(n - k)$ din cele n cereri rămân nerezolvate la sfârşitul iterației curente. Cum probabilitatea lui p de a primi un acord în competiție este k/n numărul estimat de cereri nerezolvate direcționate la q la sfârşitul iterației curente este $(n - k)\,k/n$. Deoarece valoarea maximă a lui $(n - k)\,k$ n este $n/4$, rezultă că fiecare iterație reduce numărul estimat al creşterilor rezolvate cu cel puțin ¾ . Cum, inițial sunt cel mult N^2 cereri, rezultă că numărul estimat de cereri rămase nerezolvate la sfârşitul iterației i este $N^2/4^i$. Mai rămâne de demonstrat că algoritmul ajunge la asocierea maximă în medie în $O(\log_2 N)$ paşi.

Fie C pasul în care ultima cerere este rezolvată. Atunci, valoarea estimată a lui C este:

$$E[C] = \sum_{i=1}^{\infty} i \, Pr\{C = i\} \tag{5.10}$$

adică:

$$E[C] = \sum_{i=1}^{\infty} Pr\{C > i\} \tag{5.11}$$

Probabilitatea ca $C > i$ este tocmai probabilitatea ca cel puțin o cerere să rămână neasociată la sfârşitul iterației i:

$$Pr\{C > i\} = \sum_{j=1}^{n} j \, Pr\{j \text{ cereri sa ramana dupa iteratia } i\} \tag{5.12}$$

şi mărginind suma rezultă:

$$Pr\{C > i\} \le \frac{N^2}{4^i} \tag{5.13}$$

inegalitate care rezultă din marginea numărului mediu de asocieri nerezolvate după a i-a iterație.

BUPT

Cum probabilitatea nu poate fi niciodată mai mare ca 1, rezultă că $Pr\{C > i\} \leq min(1, N^2/4^i)$, şi substituind, rezultă pentru $E[C]$:

$$E[C] \leq \sum_{i=0}^{\infty} i \, min\left(1, \frac{N^2}{4^i}\right) \tag{5.14}$$

Cum $N^2 = 4^i$, deci $i = log_2 N$, suma nu are mai mult de $log_2 N$ termeni de valoarea 1 iar restul sumei este o serie de puteri mărginită de $4/3$. Astfel:

$$E[C] \leq log_2 N + 4/3 \tag{5.15}$$

Astfel valoarea estimată a numărului de iteraţii pentru a ajunge la convergenţă nu depăşeşte $log_2 N + 4/3$.

Algoritmul APFC converge la fel de repede ca şi algoritmul PIM, fără a fi afectată eficienţa comutatorului. În practică numărul de iteraţii poate fi limitat la $log_2 N$ pentru un comutator cu N intrări.

Mecanismul de alocare a benzii la APFC poate ghida operarea asocierii iterative probabiliste, îmbunătăţind astfel imparţialitatea în alocarea benzii, chiar dacă nu sunt necesare garanţii despre bandă. De exemplu, mecanismul poate fi folosit pentru corectarea oricărei incorectitudini ale generatorului de numere aleatoare. Presupunem ca exemplu că *PI*-A nu a fost asociat cu *PE*-Bde mult timp, deşi au fost emise cereri în fiecare ciclu; acest lucru se poate întâmpla fie dacă B nu a selectat cererea lui A ca să-i trimită semnalul de acord, fie dacă A nu a selectat semnalul de acceptare a lui B. Deoarece probabil că B a servit cereri de la alte porturi în această perioadă, cererile acestora vor fi filtrate curând ca rezultatul satisfacerii rezervărilor lor de bandă, crescând probabilitatea ca B să selecteze cererea lui A pentru acord. În acelaşi timp, cum A a selectat acordurile altor porturi decât B, creditele sale pentru acele porturi se vor epuiza rapid, reducându-se numărul de acorduri recepţionate de A şi crescând probabilitatea ca acordul lui B să fie acceptat. Aceasta face alocarea benzii relativ insensibilă la operarea generatorului de numere aleatoare.

Se impune observaţia că prin introducerea etapei de filtrare a cererilor, numărul de cereri şi emise spre o ieşire oarecare va fi n' n, numărul de cereri luat în considerare la demonstrarea algoritmului PIM. Acest lucru este echivalent cu scăderea volumului de trafic de la intrarea comutatorului, deci (vezi tabelul 5.1) cu scăderea numărului de iteraţii dintr-un pas. În acest fel scade întârzierea celulelor prin comutator.

BUPT

Capitolul 6

CLASIFICAREA GENERALĂ A PLANIFICATOARELOR UNIFICATE

Majoritatea analizei algoritmilor de planificare a traficului se referă la un algoritm particular şi nu la analiza caracteristicilor traficului sesiunilor individuale, în reţele în care în nodurile căii sunt folosiţi algoritmi diferiţi de planificare. Cum reţelele actuale şi viitoare nu sunt omogene, un model general de analiză a algoritmilor de planificare este util pentru proiectarea şi analiza acestor reţele. Găsirea acelor proprietăţi ale planificatoarelor care să permită utilizarea resurselor reţelelor cât mai eficient ar fi de un real folos. De asemenea prezintă un interes deosebit comportarea şi limita întârzierii pentru topologii arbitrare de reţea.

6. 1 Servere cu latenţa ratei, *LR*

Presupunem că într-un comutator de pachete, V conexiuni partajează aceeaşi linie de ieşire, ρ_i fiind rata alocată fiecărei conexiuni. În plus presupunem că pachetele nu sunt transmise decât după recepţia ultimului bit. Cu $A_i(\tau,t)$ notăm sosirile sesiunii, conexiunii, sau fluxului i în intervalul (τ,t) şi $W_i(\tau,t)$ volum serviciilor primite de sesiune în acest interval. În sistemele bazate pe modelul cu fluide, $A_i(\tau,t)$ şi $W_i(\tau,t)$ sunt funcţii continue. În modelul pachet cu pachet $A_i(\tau,t)$ creşte doar când ultimul bit al pachetului este recepţionat de server, respectiv $W_i(\tau,t)$ creşte când ultimul bit al pachetului părăseşte serverul. Astfel modelul cu fluide este un caz particular al modelului pachet cu pachet, când pachetele sunt infinit de mici.

Perioada ocupată a sistemului, POSIS, este intervalul maxim de timp în care serverul nu e niciodată neutilizat. În această perioadă serverul transmite mereu pachete.

Perioada de conectare a sesiunii i, PCS, este orice perioadă în care pachetele acelei sesiuni sunt continuu memorate în sistem.

Fie $Q_i(t)$ volumul de trafic al sesiunii i, memorat în server la momentul t, astfel că:

$$Q_i(t) = A_i(0,t) - W_i(0,t). \tag{6.1}$$

BUPT

O sesiune este conectată la timpul t, dacă $Q_i(t) > 0$. Perioada de ocupare a unei sesiuni i, POS, este intervalul maxim $(\tau_1, \tau_2]$ astfel încât pentru orice $t \in (\tau_1, \tau_2]$ pachetele sesiunii i sosesc cu o rată mai mare sau egală cu ρ_i, sau,

$$A_i(\tau, t) \geq \rho_i(t - \tau_1). \tag{6.2}$$

Perioada de ocupare a unei sesiuni este intervalul maxim de timp în care, dacă sesiunea ar fi servită exact la rata garantată, ar rămâne continuu conectată (fig. 6.1). Pe durata de ocupare a sistemului pot apare mai multe perioade de ocupare a sesiunii i.

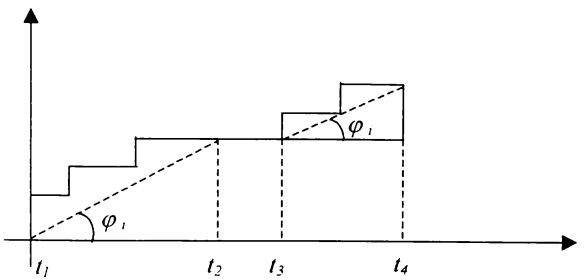


Fig. 6.1 Intervalele $(t_1, t_2]$ şi $(t_3, t_4]$ reprezintă două perioade de ocupare diferite

Perioada de ocupare a sesiunii este definită doar în funcție de sosire şi de rata alocată. Există o deosebire între PCS şi POS. POS este definit în funcție de un sistem ideal în care o sesiune conectată i este servită la o rată constantă ρ_i, în timp ce PCS se referă la un sistem real la care rata instantanee a serviciului variază în funcție de numărul de conexiuni active şi ratele lor de servire. Astfel POS poate conține intervale în care traficul sesiunii i conectate este 0; acest lucru se întâmplă când sesiunea primeşte o rată instantanee de serviciu mai mare decât ρ_i din timpul perioadei ocupate.

Pentru o perioadă dată de conectare a sesiunii i, perioada corespunzătoare de ocupare poate fi mai mare. În plus, dacă $(s_1, f_1]$ este perioada de ocupare pentru sesiunea i, pot apare mai multe perioade de conectare în acest interval în sistemul real. Începutul perioadei de ocupare este întotdeauna cauzat de sosirea pachetelor în sistem.

Trebuie remarcat că atunci când se aplică acelaşi trafic la două planificatoare diferite cu rezervări identice, perioadele corespunzătoare de conectare pot fi diferite, ceea ce face dificilă utilizarea PCS pentru analiza unei clase mari de planificatoare. Pe de altă parte POS depinde doar de structura traficului de intrare şi rata sa alocată, şi poate fi deci folosită ca un invariant în analiza diferitelor planificatoare. Acesta este motivul principal pentru care definirea serverelor LR o vom face pe baza serviciului primit.

BUPT

Acum putem defini clasa generală a serverelor cu rata latentă *LR*. Un server din această clasă este caracterizat de doi parametri: latența θ_i şi rata alocată ρ_i. Presupunem că perioada de ocupare j a sesiunii i începe la momentul τ, şi notăm cu $W_{i,j}^S(\tau,t)$ serviciul total furnizat pachetelor sesiunii care ajung după momentul τ şi înainte de t, de către serverul S. Este de remarcat faptul că serviciul total oferit sesiunii i în intervalul (τ,t) poate fi mai mare decât $W_{i,j}^S(\tau,t)$, deoarece vor fi servite şi pachetele deja memorate în sistem care provin dintr-o perioadă ocupată anterioră.

Un server S aparține clasei *LR* dacă şi numai dacă pentru orice timp t, ulterior lui τ, începutul perioadei ocupate j şi până când pachetele care ajung în această perioadă sunt servite,

$$W_{i,j}^S(\tau,t) \geq max\left(0, \rho_i\left(t-\tau-\theta_i^S\right)\right). \tag{6.3}$$

θ_i^S este numărul minim nenegativ care satisface inegalitatea precedentă.

Definiția serverelor *LR* implică doi parametri, latența şi rata. Partea stângă a ecuației (6.3) defineşte o anvelopă pentru a mărgini serviciul minim oferit sesiunii i pe timpul perioadei ocupate (fig. 6.2). Este uşor de observat că latența θ_i^S reprezintă întârzierea văzută de primul pachet din perioada ocupată a sesiunii i, în cazul cel mai defavorabil. Restricția impusă este că serviciul furnizat sesiunii de la începutul perioadei sale ocupate este mărginit inferior. Astfel, unui algoritm de planificare al acelei clase, i se cere doar ca rata medie a serviciului oferit de planificator sesiunii ocupate, pe orice interval (τ,θ_i^S) să fie cel puțin egală cu rata sa rezervată. Acest lucru este mai puțin restrictiv decât multiplexarea GPS unde se cere ca banda instantanee oferită sesiunii să fie mărginită. Aceasta este limita inferioară a ratei serviciului la multiplexarea GPS pe un interval $(\tau,t]$ în care sesiunea este conectată, în timp ce restricția pentru serverele *LR* prevede doar intervalul începând cu perioada ocupată. Astfel, multiplexarea GPS nu este decât o parte a clasei serverelor *LR*.

Latența depinde de algoritmul de planificare folosit precum şi de rata alocată şi parametrii de trafic ai sesiunii analizate. Pentru un anumit algoritm de

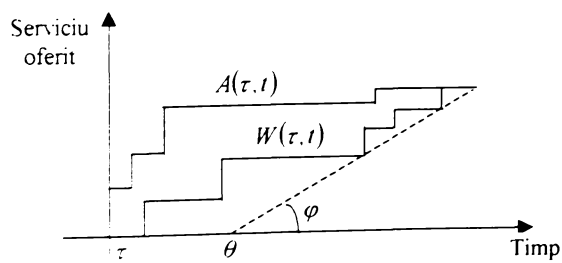


Fig. 6.2 Exemplu de comportare a unui planificator *LR*

BUPT

planificare, câțiva parametri, ca rata sa de transmisie pe linia de ieşire, numărul de sesiuni care partajează linia şi ratele lor alocate, pot influența latența. În cele ce urmează vom arăta că întârzierea maximă cap-la-cap pe care o pot avea pachetele într-o rețea de planificatoare poate fi calculată doar din latența planificatoarelor individuale ale căii sesiunii şi din parametrii de trafic ai sesiunii care generează pachetele. Cum întârzierea maximă într-un planificator creşte direct proporțional cu latența sa, modelul relevă importanța folosirii planificatoarelor cu latență mică pentru a obține întârzieri reduse cap la cap. De asemenea pot fi obținute direct, din latența planificatoarelor, limita superioară a dimensiunilor cozilor şi gradul de rafală al sesiunilor individuale, în orice punct al rețelei.

La definirea serverelor *LR* nu s-a specificat dacă serverele se bazează pe modelul cu fluid sau pachet-cu-pachet. Singura cerință impusă a fost ca pachetul să nu poată fi transmis până la recepția ultimului său bit. Astfel plecările pachetelor pot fi considerate ca impulsuri. Această presupunere este necesară pentru a limita sosirile în următorul comutator din lanțul de planificatoare. În sistemul cu fluide, este necesar să considerăm că toate planificatoarele operează cu fluide şi dimensiunea maximă a pachetului poate fi infinit de mică.

Pentru a determina limitele întârzierii în planificatoarele *LR* considerăm întâi comportarea unei sesiuni într-un singur mod şi apoi analiza va fi extinsă unei rețele de servere *RL*. În ambele cazuri presupunem că traficul de intrare analizat este nivelat de o găleată găurită şi că rata alocată este cel puțin egală cu rata medie de sosire. Astfel că, pentru sesiunea i analizată, rata sa la intrarea rețelei pe durata $(\tau, t]$ satisface inegalitatea:

$$A_i(\tau, t) \le \sigma_i + \rho_i (t - \tau), \tag{6.4}$$

unde τ_i şi ρ_i indică gradul de rafală, respectiv rata medie. Despre traficul de intrare al altor sesiuni nu vom face nici o presupunere.

6.1.1. Analiza unui singur server *LR*

Presupunem că există V sesiuni care partajează aceeaşi linie de ieşire a uni server *LR*. Mai întâi vom arăta că pachetele dintr-o perioadă ocupată din server îşi încheie serviciul cu cel mult θ_i^S după terminarea perioadei ocupate.

Lema 6.1. Dacă $(t_1, t_2]$ este perioada ocupată a sesiunii i, toate pachetele care sosesc de la sesiunea i în timpul perioadei ocupate vor fi servite până la momentul $t_2 + \theta_i^S$.

Pentru demonstrație presupunem că ultimul pachet al perioadei j ocupate îşi încheie serviciul la momentul t. Atunci la t,

BUPT

$$A_i(t_1,t_2)=W_{i,j}^S(t_1,t) \tag{6.5}$$

Dar din definiția perioadei ocupate știm că:

$$A_i(t_1,t_2)=\rho_i(t_2-t_1) \tag{6.6}$$

Din definiția serverelor *LR* și ec. (6.5) și (6.6)

$$\rho_i\left(t-t_1-\theta_i^S\right)-\rho_i(t_2-t_1)\le 0, \tag{6.7}$$

sau,

$$t\le t_2+\theta_i^S. \tag{6.8}$$

Următoarea teoremă mărginește întârzierea în cozi în server ca și necesarul de tampoane de memorie, pentru sesiunea *i*.

Teorema 6.1. Dacă *S* este un server *LR* atunci sunt adevărate următoarele limite:

1. Dacă $Q_i^S(t)$ este conectarea sesiunii *i* la timpul *t*, atunci

$$Q_i^S\le\sigma_i+\rho_i\theta_i^S. \tag{6.9}$$

2. Dacă D_i^S este întârzierea oricărui pachet al sesiunii *i* în serverul *S* atunci,

$$Q_i^S\le\frac{\sigma_i}{\rho_i}+\theta_i^S. \tag{6.10}$$

3. Traficul de ieșire se conformează modelului găleții găurite cu parametrii $\left(\tau_i+\rho_i\theta_i^S,\rho_i\right)$.

Demonstrația o vom face pentru fiecare afirmație separat.

1. Pentru deducerea limitei superioare (6.9) a conectării sesiunii *i* presupunem că perioada *j* de ocupare este intervalul $(s_j,f_j]$. Se notează cu $W_{i,j}^S(\tau,t)$ serviciul oferit pachetelor sesiunii *i* în perioada ocupată *j* în intervalul $(\tau,t]$, cu $\tau\ge s_j$ și $t\le f_j+\theta_i^S$, iar cu $W_i^S(\tau,t)$ serviciul total oferit sesiunii *i* în același interval de timp. Se observă că $W_i^S(\tau,t)\ge W_{i,j}^S(\tau,t)$ deoarece în intervalul $(\tau,t]$ serverul poate transmite și pachete din perioada ocupată precedentă.

Pe durata POS, se notează cu τ_j timpul la care ultimul pachet al POS *j* își încheie serviciul. La acest moment de timp toate pachetele conectate aparțin POS care începe după momentul f_j. În plus se observă că $\tau_j\le f_j+\theta_i^S$. Demonstrarea relației (6.9) se face prin inducție pe intervalul $(\tau_{j-1},\tau_j]$.

BUPT

În primul pas presupunem că perioada de ocupare a sesiunii i începe la momentul $S_1 = \tau_0$. Cum aceasta este prima perioadă ocupată nu există alte pachete conectate ale sesiunii i la momentul τ_0. Din definiția serverelor LR știm că pentru orice t astfel ca $\tau_0 \le t \le \tau_1$,

$$W_{i,1}^s(\tau_0, t) = W_{i,1}^s(s_1, t) \ge max\left(0, \rho_i\left(t - s_1 - \theta_i^s\right)\right).$$

Din definiția găleții găurite se știe că,

$$A_i(s_1, t) \le \sigma_i + \rho_i(t - s_1).$$

Comutarea la timpul t este definită ca volumul total de trafic care a sosit de la începutul POS minus pachetele care au fost servite. Astfel,

$$Q_i(t) = A_i(s_1, t) - W_{i,1}^s(s_1 - t) \tag{6.11}$$

Pentru t vom considera două cazuri:

Cazul 1: Dacă $t < s_1 + \theta_i^s$,

atunci $W_{i,1}^s(s_1, t) \ge 0$

$$\begin{aligned} Q_i^s(t) &\le A_i(s_1, t) \\ \text{deci} \quad &\le \tau_i + \rho_i(t - s_1) \\ &\le \tau_i + \rho_i Q_i^s \end{aligned} \tag{6.12}$$

Prima inegalitate apare din definiția găleții găurite iar a doua din condiția $t < s_1 + \theta_i^s$.

Cazul 2: Dacă $t \ge s_1 + \theta_i^s$, atunci din definiția serverelor LR,

$$W_{i,1}^s(s, t) \ge \rho_i\left(t - s_1 + \theta_i^s\right).$$

Din ecuațiile (6.4) și (6.11) rezultă:

$$\begin{aligned} Q_i^s(t) &\le \sigma_i + \rho_i(t - s_1) - \rho_i\left(t - s_1 + \theta_i^s\right) \\ &\le \sigma_i + \rho_i \theta_i^s \end{aligned} \tag{6.13}$$

În etapa de inducție presupunem că teorema este valabilă până la sfârșitul celei de n-a perioade ocupate. Aceasta înseamnă că pentru orice $t \le \tau_n$, $Q_i^s(t) \le \sigma_i + \rho_i \theta_i^s$. Trebuie să demonstrăm că ea este valabilă și în intervalul $(\tau_n, \tau_{n+1}]$. După momentul τ_n, doar pachetele care provin de la perioada ocupată $n \cdot 1$ vor fi servite, și până la momentul τ_{n+1} toate pachetele care aparțin perioadei

BUPT

ocupate $n+1$ îşi vor termina serviciul. În plus nici un pachet din a $n+1$-a perioadă ocupată nu a fost servit înainte de momentul τ_n. Astfel, pentru orice moment t, cu $\tau_n \le t \le \tau_{n+1}$, putem scrie:

$$W_{i,n+1}^{S}(\tau_n, t) = W_{i,n+1}^{S}(s_{n+1}, t).$$

Din definiția serverelor *LR*,

$$W_{i,n+1}^{S}(\tau_n, t) \ge max\left(0, \rho_i\left(t - s_{n+1} - \theta_i^S\right)\right).$$

Volumul de pachete care sunt conectate în server este egal cu acele pachete care au sosit după sfârşitul de-a n-a perioade ocupate minus pachetele servite după τ_n; dar ştim că, deoarece s_{n+1} este începutul POS a $(n+1)$-*a*, primul pachet al perioadei ocupate a sosit la momentul s_{n+1}. Astfel că:

$$\begin{aligned} Q_i^S(t) &\le A_i(s_{n+1}, t) - W_{i,n+1}^{S}(\tau_n, t) \\ &\le \sigma_i + \rho_i(t - s_{n+1}) - max\left(0, \rho_i\left(t_i - s_{n+1} - \theta_i^S\right)\right) \\ &\le \sigma_i + \rho_i \theta_i^S \end{aligned}$$

2. Pentru deducerea limitei superioare a întârzierii, presupunem că D_i^S întârzierea maximă a fost pentru pachetul care a sosit la momentul t^* din perioada ocupată j. Aceasta înseamnă că pachetul a fost servit la momentul $t^* + D_i^S$. Deci volumul serviciului oferit sesiunii până la momentul $t^* + D_i^S$ este egal cu volumul de trafic sosit de la sesiune până la momentul t. Dar cum s_j este începutul perioadei ocupate j,

$$W_{i,j}^{S}\left(s_j, t^* + D_i^S\right) = A_i\left(s_j, t^*\right). \tag{6.14}$$

Dar cum serverul nu poate asigura un serviciu la momentul θ_i^S, $D_i^S \ge \theta_i^S$. Din definiția serverelor *RL* rezultă:

$$W_{i,j}^{S}\left(s_j, t^* + D_i^S\right) \ge \rho_i\left(t^* + D_i^S - s_j - \theta_i^S\right). \tag{6.15}$$

Din ecuația (6.14) şi (6.15) şi din condiția (6.4) a găleții găurite, avem,

$$\rho_i\left(t^* + D_i^S - s_j - \theta_i^S\right) \le \sigma_i + \rho_i\left(t^* - s_j\right), \tag{6.16}$$

sau, echivalent

$$D_i^S \le \frac{\sigma_i}{\rho_i} + \theta_i^S. \tag{6.17}$$

3. Pentru a arăta că traficul de la ieşire planificatorului se conformează modelului găleții găurite este suficient să arătăm că, în intervalul $(\tau, t]$ din perioada ocupată a sesiunii i,

BUPT

$$W_i(\tau,t) \le \sigma_i + \rho_i \theta_i^S + \rho_i (t-\tau).$$

Notăm cu σ_i^{out} gradul de rafală al sesiunii i la ieşirea comutatorului, şi încercăm să găsim valoarea sa maximă. Presupunem că serverul e cu conservarea lucrului şi deci poate servi orice volum de trafic al unei sesiuni fără întrerupere, dacă respectiva sesiune este singura activă în sistem. Cu $c_i(\tau)$ notăm numărul de jetoane rămase în găleată la momentul τ, iar $\theta_i^S(\tau)$ este numărul de pachete conectate în server la acelaşi moment. Dacă presupunem că liniile de intrare sunt de capacitate infinită, la momentul $\tau+$ este posibil ca un număr de $c_i(\tau)$ pachete să fie adăugate cozii serverului şi nici un pachet nu este servit; dar am arătat că maximul de conectare al sesiunii i este mărginit prin $\sigma_i + \rho_i \theta_i^S$. Astfel că:

$$c_i(\tau) + Q_i^S(\tau) \le \sigma_i + \rho_i \theta_i^S . \tag{6.18}$$

Sosirile $A_i(\tau,t)$ nu pot depăşi numărul de jetoane din găleată la momentul τ plus numărul de jetoane primite în intervalul $(\tau,t]$, minus numărul de jetoane la momentul t. Aceasta înseamnă că,

$$A_i(\tau,t) \le c_i(\tau) + \rho_i(t-\tau) - c_i(t). \tag{6.19}$$

Serviciul oferit sesiunii i, în intervalul $(\tau,t]$ este:

$$\begin{aligned} W_i^S(\tau,t) &= Q_i(\tau) + A_i(\tau,t) - Q_i(t) \\ &\le Q_i(\tau) + A_i(\tau,t) \end{aligned} . \tag{6.20}$$

Din ecuaţia (6.19) şi (6.20), rezultă:

$$\begin{aligned} W_i^S(\tau,t) &\le Q_i(\tau) + c_i(\tau) - \rho_i(t-\tau) - c_i(t) \\ &\le (\sigma_i + \rho_i \theta_i^S) + \rho_i(t-\tau) \end{aligned} . \tag{6.21}$$

Astfel că:

$$\sigma_i^{out} \le \sigma_i + \rho_i \theta_i^S . \tag{6.22}$$

Dacă sesiunea i este singura activă în sistem, ea va fi servită explicit. Presupunem că maximul de conectare este atins la momentul t. După t, pachetele sosesc de la sesiune cu rata ρ_i. Astfel avem:

$$\sigma_i^{out} \ge \sigma_i + \rho_i \theta_i^S . \tag{6.23}$$

Din ecuaţia (6.22) şi (6.23) se poate trage concluzia că $\sigma_i^{out} = \sigma_i + \rho_i \theta_i^S$.

BUPT

6.1.2 Analiza unei rețele de servere *LR*

Pentru a calcula marginile pentru întârziere şi conectare în cazul unei rețele, cea mai simplă cale este însumarea întârzierilor maxime ale tuturor nodurilor [ABB, CO, FF, GF]. Această metodă nu ține cont de corelația între sosirile la două servere succesive şi de aceea conduc la o margine prea generală. O limită mai redusă poate fi dedusă, urmând abordarea lui Parekh şi Gallager [PG/1, PG/2] care încearcă să surprindă comportarea sesiunii în mai multe noduri simultan.

Singura condiție pe care o vom impune rețelei este că toate serverele aparținând clasei *LR* şi deci tot traficul sesiunii i e format la sursă cu o găleată găurită (σ_i, ρ_i). De asemenea presupunem că banda de frecvență rezervată sesiunii în fiecare nod este cel puțin ρ_i. Pentru început enunțăm şi demonstrăm următoarea lemă.

Lema 6.2. Într-un lanț de servere *LR*, procesul traficului după nodul k, este un proces de tip găleată găurită cu parametrii $\sigma_i + \rho_i \sum_{j=1}^{k} \theta_i^{(s_j)}$ şi ρ_i, unde $\theta_i^{(s_j)}$ este latența planificatorului j din calea sesiunii.

La demonstrarea teoremei 6.1 s-a arătat că traficul de ieşire a unui server *RL* se conformează modelului găleții găurite cu parametrii $(\sigma_i + \rho_i \theta_i^s, \rho_i)$. Aceasta înseamnă că traficul de intrare al următorului nod se conformează aceluiaşi model, astfel că după nodul k-1 traficul de ieşire al sesiunii i, şi corespunzător traficul de intrare în nodul k, se vor conforma modelului găleții găurite. Aceasta însemnă că:

$$A_k(\tau, t) \le \left(\sigma_i + \rho_i \sum_{j=1}^{k-1} \theta_i^{(s_j)} \right) + \rho_i (t - \tau). \tag{6.24}$$

Acest rezultat ne permite să mărginim gradul de rafală a traficului sesiunii i la fiecare punct din rețea. Trebuie observat că creşterea gradului de rafală pe care sesiunea o poate vedea în rețea este proporțională cu suma latențelor serverelor traversate de ea. Deci, o sesiune care nu are nici un caracter de rafală la intrarea în rețea poate, din cauza latenței serverul să câştige un mare grad de rafală.

Următoarea lemă va servi la determinarea marginii conectării sesiunii i, în fiecare nod al rețelei.

Lemma 3.3. Conectarea maximă $\theta_i^{(s_k)}(t)$ în nodul k al sesiunii este mărginită astfel:

$$Q_i^{(s_k)}(t) \le \sigma_i + \rho_i \sum_{j=1}^{k} \theta_i^{(s_j)}.$$

Pentru demonstrație notăm cu σ_i^k gradul de rafală a sesiunii i la ieşirea nodului k. Din teorema 3.1 şi lemma 3.2 avem,

BUPT

$$Q_i^{(s_k)} \le \sigma_i^{k-1} + \rho_i \theta_i^{(s_k)}$$
$$\le \sigma_i + \rho_i \sum_{j=1}^{k} \theta_i^{(s_j)}. \qquad (6.25)$$

Ca urmare a creşterii gradului de rafală, conectarea maximă pentru fiecare nod al reţelei este proporţională cu latenţa serverelor. În lema 6.3 am mărginit maximul conectării pentru sesiunea i în orice nod al reţelei. Această margine poate fi privită ca numărul maxim de tampoane de memorie necesar în nodul corespunzător al reţelei pentru a garanta că nu există pierderi pentru sesiunea i. Totuşi, aceasta nu înseamnă că putem limita numărul maxim de pachete în reţea pentru sesiunea i, prin însumarea fiecărei margini a conectării fiecărui comutator din cale. O asemenea margine nu ia în considerare corelaţia dintre sosirile proceselor la nodurile individuale şi poate fi deci prea generală.

Pentru a deduce o limită mai apropiată pentru întârzierea unei sesiuni într-o reţea de servere LR, vom arăta mai întâi că întârzierea cap la cap într-o reţea de două servere LR înseriate, este aceeaşi ca pentru o reţea cu un singur server a cărui latenţă este suma latenţelor celor două servere pe care le substituie. Rezultatul permite extrapolarea la un număr arbitrar de servere RL aflate pe calea sesiunii de modelat.

Analiza a două servere LR în serie comportă o dificultate, în sensul că dacă primul server are latenţa nenulă, perioada de ocupare a sesiunii în al doilea server poate să nu coincidă cu POS corespunzătoare din primul server. Aceasta înseamnă că un pachet care porneşte la începutul POS din primul server, poate să nu pornească la începutul POS din al doilea server. Astfel, cum rata actuală a serviciului percepută de sesiunea i în primul server poate fi mai mare decât ρ_i, o singură POS din primul server poate conduce la POS multiple în al doilea (vezi figura 6.3). Vom lua acest efect în considerare în analiza reţelei de servere LR.

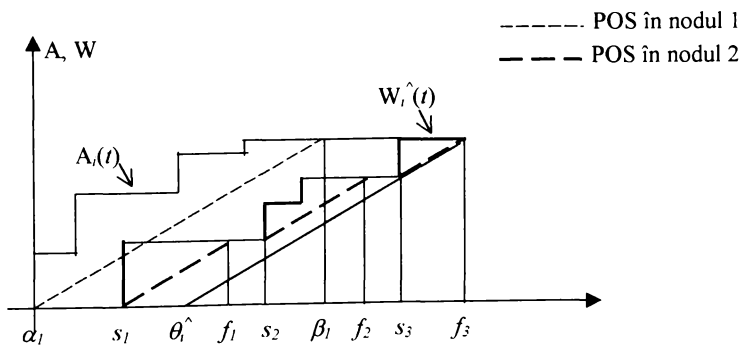


Fig. 6.3 Ilustrarea a POS în două servere înseriate. În acest caz POR pentru sesiunea i, i $(\alpha_1, \beta_1]$ din primul server se împarte în mai multe POS în al doilea server, $(s_1, f_1]$, $(s_2, f_2]$, $(s_3, f_3]$. Linia oblică ce porneşte din $\theta_i^{\wedge}$ mărgineşte toate aceste POS.

BUPT

SOR, sesiunii i într-un interval de timp, semnifică volumul de trafic al sesiunii i, transmis de ultimul server din cale, în intervalul considerat. Cu $W_{i,j}(\tau,t)$ notăm cantitatea serviciilor oferite de rețea sesiunii i, în intervalul $(\tau,t]$ în perioada j a POR, iar cu $W_{i,j}^{1}(\tau_1,t_1)$ cantitatea de servicii în intervalul (τ_1,t_1) din perioada de ocupare locală, respectiv $W_{i,j}^{2}(\tau_2,t_2)$ pentru serverul al doilea. Mai întâi vom demonstra următoarea lemă pentru mărginirea serviciului furnizat de rețea unei sesiuni într-un interval din POR.

Lemma 6.4. În cazul unei rețele de două servere S_1 şi S_2 în serie, cu latențele $\theta_i^{(S_1)}$ respectiv $\theta_i^{(S_2)}$, dacă ρ_i este rata alocată sesiunii i în rețea, SOR pachetelor celei de-a j-a POR a sesiunii, pe intervalul $(\tau,t]$ din perioada de ocupare, este mărginit astfel,

$$W_{i,j}(\tau,t) \geq max\left(0, \rho_i\left(t-\tau-\left(\theta_i^{(S_1)}+\theta_i^{(S_2)}\right)\right)\right).$$

Demonstrarea lemmei se face prin inducție față de POR. Notăm cu $(\alpha_j,\beta_j]$ intervalul celei de-a j perioade de ocupare a rețelei a sesiunii j, iar cu $W_{i,j}^{1}(\alpha_j,t)$ serviciul oferit de primul server S_1 pachetelor celei de-a j-a perioade de ocupare a rețelei până la momentul t. Trebuie remarcat că a j-a POR este aceeaşi cu aceeaşi j-a POS din primul server. Totuşi perioada ocupată percepută de serverul S_2 poate fi diferită de aceste perioade. Notăm cu $(s_k,f_k]$ a k-a POS a sesiunii i în S_2. Atunci $W_{i,k}^{2}(s_k,t)$ este serviciul oferit de S_2 în perioada k-a de ocupare, până la momentul t.

Notăm cu τ_k momentul de timp la care pachetele de la POS a sesiunii i din S_2 încep serviciul în S_2. La fel, t_k este momentul la care ultimul pachet din a k-*a* sesiune ocupată din S_2 îşi termină serviciul în S_2. Astfel este uşor de observat că:

$$\tau_k \geq s_k, \qquad t_k \leq f_k + \theta_i^{(S_2)} \qquad \text{şi} \qquad t_k \leq \tau_{k-1}.$$

În pasul principal considerăm prima POR, astfel încât $\alpha_1 \leq t \leq \beta_1$. Când prima POS pentru sesiunea i începe, nu există pachete din sesiunea precedentă. După cum s-a văzut mai înainte, prima POR a sesiunii i poate degenera în mai multe POS în al doilea server, şi anume $(s_1,f_1],(s_2,f_2],\ldots,(s_m,f_m]$ perioade succesive în care în al doilea server, pachetele provenind din prima POR vor fi servite.

Fie t^* ultimul moment de timp la care un pachet de la prima POR a fost în serviciu în S_2. Trebuie să arătăm că, pentru orice t, $\alpha_1 \leq t \leq t^*$,

$$W_{i,1}(\alpha_1,t) \geq max\left(0, \rho_i\left(t-\alpha_1-\left(\theta_i^{(S_1)}+\theta_i^{(S_2)}\right)\right)\right).$$

Pentru t, trebuie să considerăm trei cazuri.

BUPT

Cazul 1: $t \le \tau_1$. În acest caz nu a fost prestat nici un serviciu în S_2, astfel că $W_{i,1}(\alpha_1, t) = 0$, deci:

$$s_1 \le \alpha_1 + \theta_i^{(s_1)},$$

şi

$$\tau_1 \le s_1 + \theta_i^{(s_1)}.$$

Astfel că,

$$t - \alpha_1 \le \theta_i^{(s_1)} + \theta_i^{(s_2)}. \tag{6.26}$$

Deci, putem scrie:

$$W_{i,1}(\alpha_1, t) \ge max\left(0, \rho_i\left(t - \alpha_1 - \left(\theta_i^{(s_1)} + \theta_i^{(s_2)}\right)\right)\right) \tag{6.27}$$

Cazul 2: $\tau_k \ge t \le t_k$, $1 \le k \le m$ (în perioada de ocupare k din S_2).

$$W_{i,1}(\alpha_1, t) = W_{i,1}(\alpha_1, \tau_k) + W_{i,1}(\tau_k, t). \tag{6.28}$$

Dar,

$$W_{i,1}(\alpha_1, \tau_k) = W_{i,1}^1(\alpha_1, s_k);$$

şi

$$W_{i,1}(\tau_k, t) = W_{i,k}^2(\tau_k, t) = W_{i,k}^2(s_k, t).$$

Substituind în ecuația (6.28), rezultă:

$$\begin{aligned} W_{i,1}(\alpha_1, t) &= W_{i,1}^2(\alpha_1, s_k) + W_{i,k}^2(s_k, t) \\ &\ge max\left(0, \rho_i\left(s_k - \alpha_1 - \theta_i^{(s_1)}\right)\right) + max\left(0, \rho_i\left(t - s_k - \theta_i^{(s_2)}\right)\right) \\ &\ge max\left(0, \rho_i\left(t - \alpha_1 - \left(\theta_i^{(s_1)} + \theta_i^{(s_2)}\right)\right)\right). \end{aligned}$$

Cazul 3: $t_k \le t \le \tau_{k+1}$ (între perioadele ocupate din S_2); în acest caz putem scrie:

$$W_{i,1}(\alpha_1, t) = W_{i,1}(\alpha_1, \tau_{k+1}) - W_{i,1}(t, \tau_{k+1}). \tag{6.29}$$

Cum nici un pachet nu ajunge la S_2 între perioadele de ocupare, al doilea termen din partea dreaptă a relației este nul. Astfel, $W_{i,1}(\alpha_1, \tau_{k-1}) = W_{i,1}^1(\alpha_1, s_{k-1})$, deci ecuația (6.29) devine:

$$\begin{aligned} W_{i,1}(\alpha_1, t) &= W_{i,1}^1(\alpha_1, s_{k+1}) \\ &\ge max\left(0, \rho_i\left(s_{k+1} - \alpha_1 - \theta_i^{(s_1)}\right)\right). \end{aligned} \tag{6.30}$$

Dar, $t \le \tau_{k-1} \le s_{k+1} + \theta_i^{(s_2)}$

sau $s_{k-1} \ge t - \theta_i^{(s_2)}$.

Astfel că putem rescrie ecuația (6.30) sub forma:

$$W_{i,1}(\alpha_1, t) = max\left(0, \rho_i\left(t - \alpha_1 - \left(\theta_i^{(s_1)} + \theta_i^{(s_2)}\right)\right)\right)$$

BUPT

Dacă cu f_m sunt indicate momentele de timp la care un pachet din prima POR a fost servit de S_2, atunci putem contoriza tot traficul sosit în rețea în intervalul $(\alpha_1, \beta_1]$. În orice caz, este posibil ca transmisia unui pachet sosit în intervalul $(\alpha_1, \beta_1]$ să apară în a $(m+1)$ - a perioadă ocupată din S_2. Dacă este aşa, fie t^* ultimul moment de timp la care pachetul de la prima POR a fost servit de S_2. Apoi trebuie să avem în vedere, în demonstrație intervalele de timp $(t_m, \tau_{m+1}]$ şi $(\tau_{m+1}, t^*]$. Acestea pot fi folosite ca în cazul 3 respectiv 2. Acestea încheie pasul principal al demonstrației.

În etapa de inducție, presupunem că pentru toate POR, *1*, ..., *n*, lema este adevărată şi astfel:

$$W_{i,n}(\alpha_n, t) = max\left(0, \rho_i\left(t - \alpha_n - \left(\theta_i^{(S_1)} + \theta_i^{(S_2)}\right)\right)\right).$$

Vom arăta acum că, pentru a $(m+1)$- a perioadă de ocupare:

$$W_{i,n+1}(\alpha_{n+1}, t) = max\left(0, \rho_i\left(t - \alpha_{n+1} - \left(\theta_i^{(S_1)} + \theta_i^{(S_2)}\right)\right)\right).$$

În cazul cel mai simplu, primul pachet al sesiunii *i* din $(m+1)$- a POR determină începutul unei perioade ocupate din al doilea server, caz care poate fi tratat ca şi în pasul de bază. Cel mai dificil caz apare dacă, atunci când a $(m+1)$- a POR începe, unele pachete ale sesiunii *i* din perioada ocupată precedentă sunt încă conectate în al doilea server. Astfel că , începutul POR a sesiunii *i* nu va coincide întotdeauna cu începutul perioadei ocupate a lui S_2, ca în cazul pasului principal. De altfel acest lucru nu afectează analiza noastră.

Presupunem că a *m*-a perioadă ocupată a lui S_2 se desfăşoară când primul pachet a POS $(m+1)$ soseşte la S_2. De asemenea presupunem că a m - a perioadă ocupată a lui S_2 începe la momentul s_m, şi că primul pachet al celei a $(m+1)$ POR a fost servit de S_i la momentul τ. Astfel,

$$\begin{aligned} W_{i,n+1}(\alpha_{n+1}, t) &= W_{i,n+1}(\tau, t) \\ &= W_{i,m}^2(s_m, t) - W_{i,m}^2(s_m, \tau). \end{aligned} \tag{6.31}$$

A *m*-*a* perioadă ocupată care se desfăşoară în S_2 poate conține pachete de la mai multe POR, deoarece mai multe perioade ocupate a lui S_1 se pot contopi într-o singură perioadă ocupată a lui S_2.

Presupunem că *m*-a perioadă ocupată a lui S_2 conține pachete de la POR $n - L, n - L + 1, \ldots, n + 1$. Numărul total de pachete care sunt servite de S_2 până la τ este egal cu numărul total de pachete servite de POR $n - L, n - L + 1, \ldots, n$, minus acele pachete care au fost servite de a $(n - L)$ –a perioadă înainte de momentul s_m. Astfel,

$$\sum_{j=n-L}^{n} A_i(\alpha_j, \beta_j) = W_{i,n-L}^1(\alpha_{n-L}, s_m) + W_{i,m}^2(s_m, \tau), \tag{6.32}$$

BUPT

sau echivalent,

$$W_{i,m}^{2}(s_m,\tau)=\sum_{j=n-L}^{n}A_i(\alpha_j,\beta_j)-W_{i,n-L}^{1}(\alpha_{n-L},s_m)$$

$$\le \sum_{j=n-L}^{n}\rho_i(\alpha_j,\beta_j)-max\left(0,\rho_1\left(s_m-\alpha_{n-L}-\theta_i^{(S_1)}\right)\right) \quad (6.33)$$

$$\le \rho_i(\beta_n,\alpha_{n-L})-max\left(0,\rho_1\left(s_m-\alpha_{n-L}-\theta_i^{(S_1)}\right)\right)$$

$$\le \rho_i\left(\beta_n-s_m+\theta_i^{(S_1)}\right) \quad (6.34)$$

Din ecuațiile (6.31) şi (6.32) şi definiția serverelor *LR*, rezultă,

$$\begin{aligned}W_{i,n+1}(\alpha_{n+1},t)&=W_{i,m}^{2}(s_m,t)-W_{i,m}^{2}(s_m,\tau)\\&\ge max\left(0,\rho_i\left(t-s_m-\theta_i^{(S_2)}\right)\right)-\rho_1\left(\beta_n-s_m+\theta_i^{(S_1)}\right)\\&\ge max\left(0,\rho_i\left(t-\beta_n-\left(\theta_i^{(S_1)}+\theta_i^{(S_2)}\right)\right)\right)\\&\ge max\left(0,\rho_i\left(t-\alpha_{n+1}-\left(\theta_i^{(S_1)}+\theta_i^{(S_2)}\right)\right)\right).\end{aligned}$$

Ultima inegalitate este adevărată dacă $\beta_n \le \alpha_{n+1}$.

Acum, dacă a $(n+1)$ - a POR este împărțită în mai multe perioade în S_2, putem folosi aceeaşi abordare din pasul de bază pentru următoarele perioade ocupate din S_2, pentru a termina demonstrația.

Lemma 6.4 arată că serviciul oferit unei sesiuni într-o rețea de două servere *LR* conectate în serie, în cel mai defavorabil caz nu este mai mic decât serviciul oferit sesiunii de un singur server *LR* a cărei latență este egală cu suma latențelor serverelor pe care le înlocuieşte. Deoarece nu s-a făcut nici o presupunere despre serviciul maxim oferit unei sesiuni de către server, putem contopi un număr arbitrar de servere conectate în serie, pentru a estima serviciul oferit după nodul *k*. În lema precedentă, am presupus că rata alocată sesiunii este aceeaşi în toate serverele rețelei. Mai departe vom demonstra că, dacă nu ținem cont de această ipoteză, cel mai defavorabil caz al serviciului oferit va fi determinat simplu din rata cea mai mică dintre ratele alocate serverelor căii sesiunii.

Lemma 6.5 se referă la faptul că un server *LR* care poate garanta rata g_i cu latența θ_i, poate de asemenea garanta o rată $g_i' \le g_i$ cu latența θ_i.

Pentru demonstrație presupunem că un server *S* de tip *LR* poate garanta rata g_i după o latență θ_i a sesiunii *i*. Astfel că, dacă τ este începutul perioadei ocupate *j*, a sesiunii *i* din *S*, serviciul oferit sesiunii *i* în intervalul $[\tau,t)$ pachetelor perioadei *j* ocupată, este dat de:

$$W_{i,j}(\tau,t)\ge max\left(0,g_i\left(t-\tau-\theta_i\right)\right).$$

BUPT

Considerăm acum perioada ocupată a sesiunii i, în care rata serviciului $g_i' \le g_i$. Dacă τ' este începutul perioadei j de ocupare cu rata g_i', trebuie să arătăm că serviciul oferit pachetelor perioadei k a sesiunii i, cu rata g_i' de serviciu, în intervalul $(\tau', t']$ satisface inegalitatea:

$$W_{i,j}'(\tau', t') \ge max(0, g_i'(t' - \tau' - \theta_i)).$$

Trebuie remarcat că perioada ocupată cu rata g_i' poate consta din mai multe perioade cu rata g_i. Astfel, în general, a k-a perioadă ocupată cu rata g_i' constă din m perioade consecutive ocupate cu rata g_i. Aceste m perioade le notăm cu $k+1, k+2, \ldots, k+m$, şi momentele lor de început şi de sfârşit astfel $[s_1 + f_1], [s_2 + f_2], \ldots, [s_m + f_m]$, astfel s_1 indică şi începutul perioadei k ocupate cu rata g_i'. În plus este uşor de arătat că perioada ocupată cu rata g_i, nu se poate suprapune cu mai mult de o perioadă ocupată cu rata g_i'.

Notăm de asemenea cu τ_p momentul la care pachetele de la sesiunea ocupată $k + p$, cu rata g_i îşi încep serviciul în sistem. Atunci putem scrie, pentru orice moment de timp t, $s_1 \le t \le \tau_2$,

$$W_{i,k}'(s_1, t) = W_{i,k+1}(s_1, t) \tag{6.35}$$

$$\ge max(0, g_i(t - s_1 - \theta_i)) \tag{6.36}$$

$$\ge max(0, g_i'(t - s_1 - \theta_i)). \tag{6.37}$$

La fel, pentru orice moment de timp $t \ge \tau_2$, pachetele care provin de la perioada ocupată k cu rata g_i' sunt încă servite, sau $t_p \le t \le t_{p+1}$, când $1 < p \le m+1$ putem scrie,

$$W_{i,k}'(s_1, t) = \sum_{n=1}^{p-1} W_{i,k+n}(\tau_n, \tau_{n+1}) + W_{i,k+p}(\tau_p, t) \tag{6.38}$$

$$\ge \sum_{n=1}^{p-1} A_i(s_n, f_n) + W_{i,k+p}(\tau_p, t) \tag{6.39}$$

$$\ge A_i(s_1, s_p) + W_{i,k+p}(\tau_p, t) \tag{6.40}$$

$$\ge g_i'(s_p - s_1) + max(0, g_i(t - s_p - \theta_i)) \tag{6.41}$$

$$\ge g_i'(s_p - s_1) + max(0, g_i'(t - s_p - \theta_i)) \tag{6.42}$$

$$\ge max(0, g_i'(t - \tau - \theta_i)) \tag{6.43}$$

Folosind lema (6.5) putem enunța următorul corolar.

Corolar 6.1. Dacă τ este începutul perioadei j de ocupare a rețelei a sesiunii i într-o rețea de servere LR şi dacă ρ_i este lățimea minimă de bandă

BUPT

alocată sesiunii i în rețea, serviciul oferit pachetelor POR j după nodul k este dat de:

$$W_{i,j}^{s_k}(\tau,t) \geq max\left(0, \rho_i\left(t-\tau-\sum_{j=1}^{k}\theta_i^{(s_j)}\right)\right),$$

unde $\theta_i^{(s_j)}$ este latența serverelui j din rețea, pentru sesiunea i.

Folosind acest corolar, putem mărgini întârzierea cap-la-cap a sesiunii i, dacă traficul de intrare este format de o găleată şi rata medie de sosire este mai mică decât ρ_i.

Teorema 6.2. Întârzierea maximă D_i a sesiunii i într-o rețea de servere LR, constând din K servere succesive, este mărginită astfel,

$$D_i \leq \frac{\tau_i}{\rho_i} + \sum_{j=1}^{K}\theta_i^{(s_j)} \quad , \tag{6.44}$$

unde $\theta_i^{(s_j)}$ este latența serverelui j din rețea pentru sesiunea i.

Demonstrația pleacă de la corolarul precedent, conform căruia putem trata întreaga rețea ca un singur server LR echivalent, cu latența egală cu suma latențelor serverelor. Folosind teorema 6.1, putem trage direct concluzia că întârzierea maximă este:

$$D_i \leq \frac{\tau_i}{\rho_i} + \sum_{j=1}^{k}\theta_i^{(s_j)}$$

Această întârziere maximă este independentă de topologia rețelei. Ea este funcție de doi parametri şi anume gradul de rafală a traficului sesiunii la sursă şi latențele serverelor individuale din calea sesiunii. Cum singura presupunere făcută este că serverele aparțin clasei LR, rezultatul este mult mai general decât cel obținut în [PG/1, PG/2] despre limita întârzierii.

În următorul paragraf vom arăta că planificatoarele cu conservarea lucrului sunt planificatoare LR, astfel că limita întârzierii găsită aici se poate aplica la aproape orice arhitectură de rețea. Limita întârzierii din ecuația (6.44) arată că sunt două căi de a minimiza întârzierea şi necesarul de tampoane de memorie într-o rețea de servere LR: i) prin alocarea suplimentară de bandă sesiunii, reducându-se astfel termenul τ_i/ρ_i, sau ii) folosirea de servere LR de latență redusă. Cum latența se însumează prin multiple noduri, a doua abordare este de preferat în rețele mari. Prima abordare reduce utilizarea rețelei, permițând doar unui număr mai redus de sesiuni simultane să fie în lucru, decât în cazul serverelor de latență minimă. Minimizând latența se minimizează şi necesarul de tampoane ale sesiunii în serverele rețelei. Întârzierea cap-la-cap şi creşterea gradului de rafală a sesiunii într-o rețea de servere LR este proporțională cu latența θ_i^s a serverelor. Ambii parametrii pot fi minimizați proiectând serverele cu latența minimă. Latența unui server depinde, în general, de parametrii săi interni şi de alocarea benzii pentru sesiunea considerată. În plus, latența poate de asemenea varia cu numărul de sesiuni active şi de alocarea

BUPT

lor. Această dependență de latența altor sesiuni indică o izolare slabă a planificatorului. Astfel că în anumite planificatoare, latența poate depinde mult de parametrii interni şi mai puțin de alocarea benzii pentru sesiunea observată. Asemenea planificatoare nu ne permit să controlăm latența sesiunii prin banda alocată ei. Pe de altă parte, latența unui server PGPS depinde mult de banda alocată sesiunii considerate astfel că este de dorit să existe o flexibilitate.

6.1.3 Limita întârzierii pentru surse formate cu o găleată dublă

Cum definiția unui server LR nu este făcută în vreo ipoteză asupra traficului său de intrare, este uşor de dedus limita întârzierii pentru alte distribuții de trafic decât modelul (σ, ρ). De exemplu, dacă rata maximă a sursei este cunoscută, poate fi obținută o limită superioară modificată a întârzierii unui server LR. Notăm cu g_i rata de serviciu alocată conexiunii i şi cu ρ_i rata medie, respectiv cu P_i rata maximă a sursei, pentru conexiunea i. Sosirile la intrarea serverului în intervalul $(\tau, t]$ satisface inegalitatea,

$$A_i(\tau, t) \le min\left(\sigma_i + \rho_i(t-\tau), P_i(t-\tau)\right) \tag{6.45}$$

Putem demonstra lema următoare.

Lemma 6.6. Întârzierea maximă D_i a sesiunii i într-un server LR, când se cunoaşte rata maximă a sursei, este mărginită astfel:

$$D_i^s \le \left(\frac{P_i - g_i}{g_i}\right)\left(\frac{\sigma_i}{P_i - \rho_i}\right) + \theta_i^s \tag{6.46}$$

Pentru demonstrație presupunem că întârzierea maximă D_i^s a fost determinată pentru pachetele care sosesc la momentul t^* din perioada ocupată j. Aceasta înseamnă că pachetele au fost servite la momentul $t^* + D_i^s$. Astfel volumul serviciilor oferite sesiunii până la momentul $t^* + D_i^s$ este egal cu traficul sosit de la sesiune până la momentul t. Cum s_j este începutul perioadei j ocupate,

$$W_{i,j}^s\left(s_j, t^* + D_i^s\right) = A_i\left(s_j, t^*\right). \tag{6.47}$$

Cu ecuația (6.45), aceasta devine,

$$W_{i,j}^s\left(s_j, t^* + D_i^s\right) = min\left(\sigma_i + \rho_i\left(t^* - s_j\right), P_i\left(t^* - s_j\right)\right). \tag{6.48}$$

Cum serverul nu produce un serviciu la momentul θ_i^s, $D_i^s \ge \theta_i^s$. Din definiția serverelor RL,

BUPT

$$W_{i,j}^{s}(s_j, t^* + D_i^s) \geq g_i(t^* + D_i^s - s_j - \theta_i^s). \tag{6.49}$$

Din ecuația (6.48) şi (6.49), avem:

$$g_i(t^* + D_i^s - s_j - \theta_i^s) \leq min(\sigma_i + \rho_i(t^* - s_j), P_i(t^* - s_j)). \tag{6.50}$$

Cazul 1: Când $P_i(t^* - s_j) \leq \sigma_i + \rho_i(t^* - s_j)$, avem:

$$D_i^s \leq \left(\frac{P_i - g_i}{g_i}\right)(t^* - s_j) + \theta_i^s \tag{6.51}$$

şi

$$P_i(t^* - s_j) \leq \sigma_i + \rho_i(t^* - s_j)$$

sau,

$$t^* - s_j \leq \frac{\sigma_i}{P_i - \rho_i}.$$

Substituind $(t^* - s_j)$ în ecuația (6.51), avem:

$$D_i^s \leq \left(\frac{P_i - g_i}{g_i}\right)\left(\frac{\sigma_i}{P_i - \rho_i}\right) + \theta_i^s. \tag{6.52}$$

Cazul 2: Când $P_i(t^* - s_j) > \sigma_i + \rho_i(t^* - s_j)$, avem:

$$(t^* - s_j) > \frac{\sigma_i}{P_i - \rho_i}. \tag{6.53}$$

Din ecuația (6.50),

$$(P_i - g_i)(t^* - s_j) \leq \sigma_i - \rho_i D_i^s + g_i \theta_i^s \tag{6.54}$$

şi substituind $(t^* - s_j)$ din ecuația (6.53), rezultă:

$$D_i^s \leq \left(\frac{P_i - g_i}{g_i}\right)\left(\frac{\sigma_i}{P_i - \rho_i}\right) + \theta_i^s. \tag{6.55}$$

O rețea de servere *LR* poate fi modelată ca un singur server *LR* cu latența egală cu suma latențelor. Astfel poate fi dedus următorul rezultat important:

Corolarul 6.2. Întârzierea maximă D_i a sesiunii i, într-o rețea de servere *LR* formată din K servere în serie, la care se cunoaşte rata maximă a sursei, este mărginită astfel,

BUPT

Unde $\theta_i^{(s,)}$ este latența nodului j.

6.2 Planificatoare din clasa *LR*

În acest paragraf vom arăta că unele din cele mai cunoscute planificatoare cu conservarea lucrului aparțin clasei *LR* şi se calculează latența lor. Definiția serverelor *LR* este făcută în funcție de POS. Practic este totuşi mai uşor să se analizeze algoritmii de planificare în funcție de PCS. Lema următoare permite estimarea latenței serverelor *LR* conform cu comportarea în perioadele de conectare a sesiunii.

Lemma 6.7, Fie $(s_j, t_j]$ intervalul de timp în care sesiunea i este continuu conectată în serverul S. Dacă serviciul oferit pachetelor care sosesc în intervalul $(s_j, t_j]$ poate fi mărginit la orice moment t, cu $s_j < t \le t_j$ astfel:

$$W_i(s_j, t) \ge max\left(0, \rho_i(t - s_j - \theta_i)\right),$$

atunci S este un server *LR* cu latența mai mică sau egală cu θ_i.

Această lemă ne va permite să estimăm latența unui server *LR*. Cu toate acestea nu este necesar să determinăm o margine compactă pentru θ_i. O cale simplă de a determina dacă marginea este compactă este de a da cel puțin un exemplu, în care serviciul oferit este de fapt egal cu marginea, metodă pe care o vom folosi în acest paragraf pentru a determina latența unor servere *LR*.

Demonstrarea lemei se face prin inducție. Notăm cu $(\alpha_k, \beta_k]$ perioada k de ocupare a sesiunii i în server.

În pasul de bază, la momentul α_1 care este începutul primei POS, sistemul devine conectat pentru prima oară. Presupunem că prima POS constă din câteva perioade de conectare $(s_j, t_j]$. Din definiția POS, la începutul perioadei j de conectare trebuie să existe următoarea inegalitate,

$$A(\alpha_1, s_j) \ge \rho_i (s_j - \alpha_1)$$

De altfel sistemul este vid la momentul s_j. Astfel,

$$A(\alpha_1, s_j) = W_{i,1}(\alpha_1, s_j).$$

Considerăm orice moment t în intervalul $(s_j, t_j]$, şi putem scrie:

BUPT

$$\begin{aligned} W_i(\alpha_1, t) &= W_{i,1}(\alpha_1, t) \\ &= W_{i,1}(\alpha_1, t) + W_{i,1}(s_j, t) \\ &\geq \rho_i (s_j - \alpha_1) + max(0, \rho_i (t - s_j - \theta_i)) \\ &\geq max(0, \rho_i (t - \alpha_1 - \theta_i)). \end{aligned} \tag{6.58}$$

Considerăm acum orice moment t din perioada ocupată, dar între două POS, adică $t_j \leq t \leq s_{j+1}$. Cum sesiunea i nu primeşte serviciu în intervalul $(t, s_{j+1}]$, putem scrie:

$$W_i(\alpha_1, t) = W_{i,1}(\alpha_1, t) = W_{i,1}(\alpha_1, s_{j+1}). \tag{6.59}$$

Dar,

$$\begin{aligned} W_{i,1}(\alpha_1, s_{j+1}) &= A_i(\alpha_1, s_{j+1}) \\ &\geq \rho_i (s_{j+1} - \alpha_1) \\ &\geq \rho_i (t - \alpha_1) \end{aligned} \tag{6.60}$$

Astfel, din (6.59) şi (6.60) rezultă,

$$W_i(\alpha_1, t) \geq \rho_i (t - \alpha_1). \tag{6.61}$$

Din (6.58) şi (6.61) putem trage concluzia că la orice moment t din prima perioadă ocupată $(\alpha_1, \beta_1]$,

$$W_i(\alpha_1, t) \geq max(0, \rho_i (t - \alpha_1 - \theta_i)). \tag{6.62}$$

În etapa de inducţie presupunem că lema este adevărată pentru toate perioadele ocupate *1, 2, ... n* şi demonstrăm că este adevărat şi pentru perioada ocupată $n+1$.

Dacă atunci când începe perioada ocupată $n+1$, sistemul nu este conectat, se poate repeta aceeaşi demonstraţie ca şi în pasul de bază. Totuşi este posibil, ca atunci când începe perioada ocupată $n+1$, sistemul să fie încă conectat cu pachete din perioada ocupată n, precedentă. Presupunem acum că perioada conectată care se derulează când începe perioada $n+1$ ocupată, este perioada conectată m. Această POS începe la momentul s_m; în cazul general putem presupune că pachetele din L perioade anterioare au fost servite în intervalul de conectare $(s_m, t_m]$. Fie τ momentul la care ultimul pachet din perioada ocupată n a fost servit. Atunci,

$$\begin{aligned} W_i(s_m, \tau) &= \sum_{j=n-L}^{n} A_i(\alpha_j, \beta_j) - W_{i,n-L}(\alpha_{n-L}, s_m) \\ &\leq \rho_i (\beta_n - \alpha_{n-L}) - \rho_i (s_m - \alpha_{n-L}) \\ &\leq \rho_i (\beta_n - s_m). \end{aligned} \tag{6.63}$$

Trebuie să arătăm că $W_{i,n-1}(\alpha_{n-1}, t) \geq max(0, \rho_i (t - \alpha_{n-1} - \theta_i))$. Procedăm astfel,

BUPT

$$
\begin{aligned}
W_{i,n-1}(\alpha_{s-1}, t) &= W_i(\tau, t) \\
&= W_i(s_n, t) - W_i(s_n, \tau) \\
&\geq \max(\rho_i(t - s_n - \theta_i) - \rho_i(\beta_s - s_n)) \\
&\geq \max(\rho_i(t - \beta_s - \theta_i)) \\
&\geq \max(\rho_i(t - \alpha_{s-1} - \theta_i)).
\end{aligned}
$$

Ultima inegalitate provine din faptul că $\beta_s < \alpha_{s-1}$ şi că serviciul oferit sesiunii nu poate fi negativ. Dacă după momentul t_1 pachetele perioadei ocupate $n-1$ formează perioade conectate multiple, demonstraţia din pasul de bază poate fi repetată pentru completarea demonstraţiei lemmei.

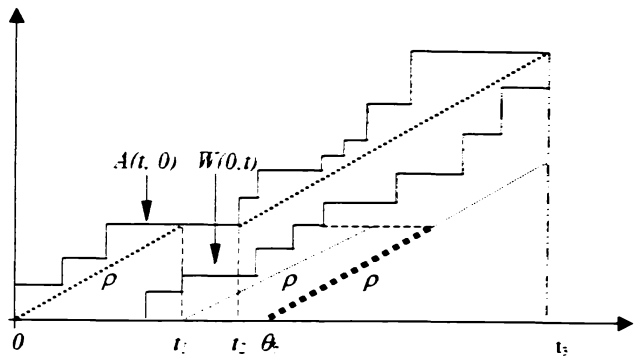


Figura 6.4. Diferenţa dintre mărginirea serviciului funcţie de perioada corectată respectiv perioada ocupată

Contribuţia principală a teoriei serverelor *LR* este noţiunea de perioadă ocupată. Marginea serviciului oferit de un server este bazată pe noţiunea de perioadă ocupată. Aceasta este o abordare mult mai generală care mărgineşte serviciul oferit de server, bazată pe conceptul de perioadă ocupată, în loc de perioadă de conectare. Se obţin astfel margini mai compacte pentru întârzierea cap-la-cap şi o clasă mai extinsă de planificatoare care pot oferi aceste întârzieri.

În lema 6.7 am demonstrat că, dacă folosim perioade de conectare pentru a mărgini serviciul oferit de serverul S, atunci S este un server *LR* şi latenţa sa nu poate fi mai mare decât cea determinată pentru perioada de conectare. Dar opusul nu este adevărat. Dacă considerăm exemplul din fig. 6.4 şi presupunem că avem un server *LR* cu rata ρ şi latenţa θ_i atunci intervalele $(0, t_1]$ şi $(t_2, t_3]$ sunt două perioade ocupate. Totuşi serverul rămâne conectat pe tot intervalul $(t_1, t_3]$. Dacă pentru determinare marginii serviciului a fost folosită perioada de conectare, atunci ar putea rezulta o latenţă $\theta_i' > \theta_i$. Extinzând exemplul precedent pe mai multe POS este mai uşor de verificat că θ_i' nu poate fi mărginită; acest lucru arată că dacă se foloseşte PCS în loc de POS în definirea serverelor *LR*, întârzierea cap-la-cap a serverului nu poate fi mărginită. Folosind lema precedentă, putem analiza câteva servere cu conservarea lucrului şi demonstra

BUPT

că ele aparţin clasei *LR*. În următoarea demonstraţie L_i indică lungimea maximă a pachetelor sesiunii *i*, iar L_{max} lungimea maximă a pachetelor din toate sesiunile.

6.2.1 Servirea imparţială ponderată WFQ

Începem prin a demonstra mai întâi că planificatorul GPS este un server *LR*.

Lemma 6.8. Un planificator GPS aparţine clasei *LR* şi latenţa sa este zero.

Pentru demonstraţie, apelăm la definiţia multiplexorului GPS, pe intervalul de timp în care conexiunea *i* este continuu conectată,

$$W_i^{GPS}(\tau,t) \geq \rho_i(t-\tau).$$

Din lemma (6.7) este uşor de conchis că planificatorul GPS este un planificator *LR* cu latenţa nulă.

Algoritmul de planificare cu servire imparţială ponderată WFQ sau GPS pachet-cu-pachet PGPS, este echivalentul multiplexării GPS lucrând pachet cu pachet. În [PG/1] s-a demonstrat că, dacă t^F şi t^P sunt momentele de timp la care pachetele termină în cele două discipline, atunci

$$t^P \leq t^F + \frac{L_{max}}{r},$$

unde *r* este rata de transmisie a serverului. Pentru analiza unei reţele de server *LR* este necesar ca serviciul să fie mărginit la orice moment de timp după începerea POS. În plus, trebuie doar să considerăm că pachetul părăseşte serverul pachet-cu-pachet, doar după ce ultimul său bit a plecat din server. Această condiţie este necesară pentru a furniza o imagine exactă a gradului de rafală din reţea. Astfel, chiar înainte de t^P nu a plecat încă tot pachetul din serverul pachet-cu-pachet. Serviciul oferit conexiunii *i* în serverul pachet-cu-pachet va fi egal cu serviciul oferit aceleaşi conexiuni de multiplexorul GPS până la momentul t^P minus ultimul pachet. Este de remarcat că în multiplexorul GPS, deoarece are latenţa zero, începutul POS găseşte mereu sistemul vid. Deci,

$$\begin{aligned} W_{i,j}^{P}(\tau,t) &\geq W_{i,j}^{F}\left(\tau, t-\frac{L_{max}}{r}\right) - L_i \\ &\geq max\left(0,\ \rho_i\left(t-\tau-\frac{L_{max}}{r}\right) - L_i\right) \\ &\geq max\left(0,\ \rho_i\left(t-\tau-\frac{L_{max}}{r}-\frac{L_i}{\rho_i}\right)\right). \end{aligned}$$

Astfel, putem enunţa următorul corolar:

BUPT

Corolarul 6.3. Planificatorul WFQ este un server *LR* cu latența $\frac{L_{max}}{r}+\frac{L_i}{\rho_i}$, latență care se poate arăta că este compactă.

6.2.2. Ceasul virtual, VCL

Disciplina VCL se bazează pe calculul etichetei de timp pentru fiecare pachet care soseşte în sistem [XL]. Eticheta de timp este calculată funcție de etichetele de timp ale pachetelor precedente ale aceleaşi conexiuni şi timpul real care a trecut după începutul POS.

Teorema 6.3. Disciplina de servire VCL este un server *LR* cu latența $L_{max}/r + L_i/\rho_i$. Demonstrarea acestei teoreme se face cu teoria potențialelor.

6.2.3. Servirea imparțială cu autosincronizare, SCFQ

SCFQ a fost propusă pentru simplificarea algoritmului de sortare a priorităților de la serverul PGPS. Când o conexiune devine conectată, timpul virtual este estimat de timpul virtual de sfârşit a pachetului aflat în serviciu, fie acesta TScrt. Atunci timpul virtual de sfârşit al pachetului *j* a conexiunii *i*, pachet cu lungimea L_i^j, se estimează cu:

$$F_i^j = max\left(F_i^{j-1}, TS\,crt\right) + L_i^j/\rho_i .$$

Pachetele sunt servite în ordinea crescătoare a timpilor de sfârşit virtuali. Când sistemul devine vid, toate variabilele sunt puse pe zero.

Lemma 6.9. Dacă pachetul *k* al sesiunii *i* îşi începe PCS în serverul SCFQ, timpul virtual de sfârşit va fi estimat din timpul virtual al sistemului,

$$F_i^k = TS\,crt + L_i^j/\rho_k .$$

Demonstrația se face prin reducere la absurd. Dacă presupunem că $F_i^{k-1} > TS\,crt$, atunci pachetul F_i^{k-1} nu a fost încă servit şi deci pachetul *j* va găsi sistemul vid.

Teorema 6.4. La orice moment *t* din PCS a sesiunii *i* într-un server SCFQ,

$$W_i(\tau, t) \geq \rho_i \left(t - \tau - (I' - 1)L_{max}/r - L_i/\rho_i\right).$$

Pentru demonstrație fie începutul PCS a sesiunii *i* în serverul SCFQ. Presupunem că PCS a început prin sosirea pachetului *k* a sesiunii *i*. Din lemma 3.9, timpul virtual de sfârşit al acestui pachet va fi:

BUPT

$$F_i^k = TS\,crt + L_i^j / \rho_i \,. \tag{6.64}$$

Considerăm a *n*-a POS *i*, ceea ce înseamnă că este pachetul $(k+n-1)$ al sesiunii *i*, şi marcăm acest pachet. Timpul său virtual de sfârşit va fi

$$F_i^{k+n-1} = TS\,crt + n\,L_i / \rho_i \,. \tag{6.65}$$

Calculăm acum cât trafic al altor sesiuni va fi servit înaintea pachetului marcat, în cazul cel mai defavorabil. Fiecare din celelalte sesiuni poate avea o dimensiune L_{max} de pachete cu timpul de sfârşit *TScrt*. În plus, fiecare conexiune $j \neq i$ poate transmite $(n\,L_i/\rho_i)\rho_j$ traficul maxim înainte de transmiterea pachetului marcat iar eticheta sa de timp va fi în $TS\,crt + n\,L_i/\rho_i$. Astfel momentul la care pachetul marcat va fi transmis, este:

$$\begin{aligned} D_i^{k+n-1} &\le \tau + (V-1)\frac{L_{max}}{r} + \frac{1}{r}\left(\sum_{j=k}^{k+n-1} L_i + \sum_{j=1,\,j\neq i}^{V} n\frac{L_i}{\rho_i}\rho_j\right) \\ &\le \tau + (V-1)\frac{L_{max}}{r} + \frac{1}{r}\left(n\,L_i + n\frac{r-\rho_j}{\rho_i}L_i\right) \\ &\le \tau + (V-1)\frac{L_{max}}{r} + n\,\frac{L_i}{\rho_i}. \end{aligned} \tag{6.66}$$

Considerăm momentul $t \ge \tau$ în timpul PCS şi presupunem că pachetul marcat a fost ultimul servit în intervalul $(\tau, t]$ din sesiunea *i*. Atunci serviciul total primit de sesiunea *i* în $(\tau, t]$ este:

$$W_i(\tau, t) = n\,L_i / \rho_i \,. \tag{6.67}$$

Dar *t* trebuie să fie înaintea momentului la care pachetul $k-n$ al sesiunii va părăsi serverul. Deci:

$$t \le \tau + (V-1)\frac{L_{max}}{r} + (n+1)\,\frac{L_i}{\rho_i}, \tag{6.68}$$

sau:

$$n\,L_i \ge \rho_i\left(t - \tau + (V-1)\frac{L_{max}}{r} + \frac{L_i}{\rho_i}\right). \tag{6.69}$$

Substituind nL_i din (6.69) în (6.67),

$$W_i(\tau, t) \ge \rho_i\left(t - \tau + (V-1)\frac{L_{max}}{r} + \frac{L_i}{\rho_i}\right). \tag{6.70}$$

Cum $W_i(\tau, t)$ nu poate fi negativ, putem scrie:

BUPT

$$W_i(\tau,t) \geq max\left(0,\ \rho_i\left(t - \tau - (V-1)\frac{L_{max}}{r} - \frac{L_i}{\rho_i}\right)\right). \qquad (6.71)$$

Folosind lemma 6.7 putem enunța următorul corolar:

Corolarul 6.4 Planificatorul SCFQ aparține clasei serverelor *LR* şi latența sa este mai mică cel mult egală cu $(V-1)\frac{L_{max}}{r} - \frac{L_i}{\rho_i}$.

Pentru demonstrație este suficient să folosim un exemplu. Fie 1 rata de transmisie a serverului, iar sesiunea i are alocată rata ρ_i şi celelalte V-1 conexiunii au alocate ratele $(1-\rho_i)/(V-1)$. Mai presupunem că toate sesiunile cu excepția lui i, devin conectate la momentul 0 cu pachete cu lungimea L_{max}. Atunci toate vor avea acelaşi timp virtual de sfârşit *TS crt*. Pachetul cu lungimea L_i la conexiunea i, ajunge chiar după ce primul pachet al uneia din celelalte conexiuni începe să fie servit, astfel $F_i^1 = TS\,crt + L_i/\rho_i$. Acest pachet şi toate celelalte pachete au timpul virtual de sfârşit *TS crt*. Acum presupunem că fiecare conexiune $j \neq i$ transmite pachete cu lungimea totală $S = L_i\,\rho_j/\rho_i$. Timpul virtual de sfârşit al ultimului pachet din grup va fi egal cu $TS\,crt + L_i/\rho_i$, pentru fiecare conexiune j. În cazul cel mai defavorabil, pachetul conexiunii i va putea fi servit doar după ce toate aceste pachete îşi încheie serviciul. Astfel, primul pachet al conexiunii i îşi va termina serviciul doar la momentul:

$$\begin{aligned} D_i &= (V-1)\frac{L_{max}}{r} + \frac{1}{r}\sum_{j=1, j\neq i}^{V} L_i\,\rho_j/\rho_i + \frac{L_i}{r} \\ &= \frac{(V-1)L_{max} + (L_i/\rho_i)(r-\rho_i) + L_i}{r} \\ &= (V-1)\frac{L_{max}}{r} + \frac{L_i}{\rho_i}. \end{aligned}$$

6.2.4 Planificarea carusel cu recuperarea deficitului DRR

Începem cu ***Lemma 6.10***. Dacă t_o este începutul PCS într-un server DRR, atunci la orice moment din PCS,

$$W_i(t_0,t) \geq max\left(0,\ \rho_i\left(t - t_0 - \frac{3F - 2\phi_i}{r}\right)\right).$$

Pentru demonstrație presupunem o conexiune conectată la t_o, iar t_k indică timpul de sfârşit al rundei în algoritmul DRR. În [SV] se arată că dacă o conexiune i este continuu conectată în intervalul $(\tau, t_k]$, atunci la sfârşitul rundei k,

$$W_i(t_0,t_k) \geq k\,\phi_i - D_i^k. \qquad (6.72)$$

BUPT

Pentru fiecare interval $(t_{k-1}, t_k]$ putem scrie:

$$t_k - t_{k-1} \le \frac{1}{r}\left(F + \sum_{j=1}^{V} D_j^{k-1} - \sum_{j=1}^{V} D_j^k\right). \tag{6.73}$$

Însumând după k,

$$\begin{aligned} t_k - t_0 &\le k\frac{F}{r} + \frac{1}{r}\sum_{j=1}^{V} D_j^0 - \frac{1}{r}\sum_{j=1}^{V} D_j^k \\ &\le k\frac{F}{r} + \frac{F-\phi_i}{r}, \end{aligned}$$

sau

$$k \ge \frac{t_k - t_0}{F} r + \frac{\phi_i}{F} - 1 \tag{6.74}$$

Înlocuind k în ecuația (6.72), obținem:

$$\begin{aligned} W_i(t_0, t_k) &\ge \phi_i\left(\frac{t_r - t_0}{F} r + \frac{\phi_i}{F} - 1\right) - D_i^k \\ &\ge \rho_i(t_k - t_0) - \phi_i\left(1 - \frac{\phi_i}{F}\right) - D_i^k. \end{aligned} \tag{6.75}$$

În cel mai defavorabil caz conexiunea i va fi ultima servită în runda k-a. Distingem două cazuri. În primul caz, conexiunea transmite mai mult decât ϕ_i în runda k, şi atunci numărul maxim de pachete transmise este mărginit prin $2\phi_i - D_i^k$. Astfel, pentru orice moment de la începutul rundei k până când pachetul începe să fie servit,

$$\begin{aligned} W_i(t_0, t) &\ge max\left(0, \rho_i(t_k - t_0) - \phi_i\left(1 - \frac{\phi_i}{F}\right) - D_i^k - 2\phi_i + D_i^k\right) \\ &\ge max\left(0, \rho_i(t_k - t_0) - 3\phi_i + \phi_i\frac{\phi_i}{F}\right) \\ &\ge max\left(0, \rho_i\left(t - t_0 - \frac{3F - 2\phi_i}{r}\right)\right). \end{aligned}$$

Ultima inegalitate provine din faptul că cel puțin ϕ_i octeți au fost serviți în runda k, de la conexiunea i. Astfel, $t \le t_k - \left(\frac{\phi_i}{r} + D_i^k\right)$. Când aceste pachete încep să fie servite, ele sunt servite cu rata $r \ge \rho_i$ şi de aceea relația de mai sus este valabilă pentru orice t din runda k.

În cazul 2 numărul de pachete servite în runda k, pentru conexiunea i a fost $\le \phi_i$. Astfel, din nou, pentru orice t înainte ca pachetele să fie servite în runda k,

BUPT

$$W_i(t_0,t) \geq max\left(0, \rho_i\ (t_k - t_0) - \phi_i\left(1 - \frac{\phi_i}{F}\right) - D_i^k - \phi_i + D_i^k\right)$$

$$\geq max\left(0, \rho_i\ (t_k - t_0) - 2\phi_i + \phi_i \frac{\phi_i}{F}\right)$$

$$\geq max\left(0, \rho_i \left(t - t_0 - 2\frac{F}{r} + \frac{\phi_i}{r}\right)\right).$$

Cum $\phi_i \leq F$,

$$W_i(t_0,t) \geq max\left(0, \rho_i \left(t - t_0 - \frac{3F - 2\phi_i}{r}\right)\right).$$

Pentru cazul în care conexiunea i ar transmite mai puțin de ϕ_i octeți în ultima rundă, contorul de deficit se pune pe zero doar după sfârşitul rundei. Astfel contorul va fi egal cu ϕ_i minus numărul de biți transmişi în timpul rundei, chiar dacă coada a fost vidă în această rundă.

Folosind lemma 3.7 putem enunța următorul corolar.

Corolarul 3.5. DRR este un planificator LR cu latența $(3F - 2\phi_i)r$.

Pentru demonstrație folosim un exemplu care să confirme relația. Presupunem că la momentul τ, conexiunea i devine conectată când toate celelalte sesiuni au un contor de deficit $D_j = \phi_j - \Delta\phi_i$. Conexiunea i va fi ultima servită în această rundă şi deci trebuie să aştepte un timp minim de $\sum_{j \neq i}(\phi_j + D_j)$. Presupunem acum că primul pachet al conexiunii i are dimensiunea $\Delta\phi_i$ iar ultimul are dimensiunea ϕ_i. Apoi doar primul pachet este servit, după care pachetele ϕ_j ale oricăreia din celelalte conexiuni vor fi servite înaintea servirii celui de al doilea pachet. Timpul la care acest pachet îşi termină serviciul va fi:

$$\sum_{j \neq i}(\phi_j + D_j) + \sum_{j \neq i}\phi_j + \phi_i + \Delta\phi_j$$

sau:

$$3\sum_{j=1}^{V}\phi_j + 2\phi_i - \sum_{j \neq i}\Delta\phi_j + \Delta\phi_i.$$

Dacă $\Delta\phi_j << F$ se vede că această margine este foarte apropiată de $3F - 2\phi_i$.

6.2.5 Planificarea carusel ponderată WRR

WRR poate fi considerată ca un caz particular al DRR. Dacă ϕ_i este întotdeauna un multiplu întreg de pachete, sau celule, nu există condiții suplimentare impuse contorului de deficit. Extensia demonstrației precedente pentru WRR este directă.

BUPT

Lemma 6.11. WRR este un server *LR* cu latența $(F-\phi_i+L_c)/r$, unde L_c este dimensiunea unei celule ATM.

Demonstrația: în cazul cel mai defavorabil toate celulele sesiunii *i* vor fi transmise la sfârşitul rundei. Notăm cu $t_1, \ldots t_k$ timpii de sfârşit ai rundei *k* după începutul PCS la timpul t_o. Fie *t* timpul dintre runda *k* şi după începutul PCS *i*, la care celula *j* a sesiunii *i* îşi începe transmisia. Astfel,

$$W_i(t_0,t)=W_i(t_0,t_{k-1})+W_i(t_{k-1},t) \tag{6.76}$$

$$=max\ (0,(k-1)\phi_i)+(j-1)L_c \tag{6.77}$$

Dar cum sesiunea *i* este conectată continuu, $t_{k-1}-t_0 \le (k-1)\frac{F}{r}$. Rezultă:

$$W_i(t_0,t)\ge max\left(0,(t_{k-1}-t_0)-\frac{\phi_i}{F}r\right)+(j-1)L_c \tag{6.78}$$

$$\ge max\ (0,\rho_i(t_{k-1}-t_0))+(j-1)L_c \tag{6.79}$$

Dar ştim la momentul *t* că,

$$t \le t_{k-1}+\frac{1}{r}\left(\sum_{n=1,n\neq i}^{V}\phi_n+jL_c\right)$$

$$\le t_{k-1}+\frac{1}{r}(F-\phi_i+jL_c)$$

sau,

$$t_{k-1}\ge t-\frac{F-\phi_i+jL_c}{r}. \tag{6.80}$$

Substituind în ecuația (6.79) obținem :

$$W_i(t_0,t)\ge max\left(0,\rho_i\left(t-t_0-\frac{F-\phi_i+jL_c}{r}\right)\right)+(j-1)L_c. \tag{6.81}$$

Valoarea minimă a părții drepte apare când $j \quad 1$, deci se poate scrie

$$W_i(t_0,t)\ge max\left(0,\rho_i\left(\frac{t-t_0+L_c}{r}\right)\right). \tag{6.82}$$

Aceasta este latența celulelor care sosesc la începutul rundei şi sunt servite la sfârşitul ei.

În tabelul 6.1 sunt prezentate rezultatele pe scurt.

BUPT

Tabelul 6.1. Servere *RL* şi latenţele lor

Server	*Latenţa*
GPS	0
PGPS	$L_i/\rho_i + L_{max}/r$
SCFQ	$L_i/\rho_i + (V - 1)L_{max}/r$
VCL	$L_i/\rho_i + L_{max}/r$
DRR	$(3F - 2\phi_i)/r$
WRR	$(F - \phi_i + L_c)/r$

L_i este dimensiunea maximă a pachetelor sesiunii i, iar $L_{i\ max}$ dimensiunea maximă a pachetelor pentru toate sesiunile. La WRR şi DRR, F este dimensiunea cadrului şi ϕ_i volumul de trafic din cadru alocat sesiunii i. La WRR, L_c este dimensiunea fixă a pachetului (celulei).

Este uşor de văzut că la PGPS şi VCL sunt cele mai mici latenţe, şi în plus acestea nu depind de numărul de conexiuni care partajează aceeaşi linie de ieşire. Dar (vezi § 6.4) VCL nu este un algoritm imparţial.

La SCFQ, latenţa depinde liniar de numărul maxim de conexiuni partajând linia de ieşire. La DRR, latenţa depinde de dimensiunea F a cadrului care, conform cu definiţia algoritmului, depinde de granularitatea alocării benzii şi dimensiunea maximă a pachetului sesiunii. Astfel că, $\sum_{i=1}^{V} L_i \le F$, unde L_i este dimensiunea de pachet în sesiunea i. Astfel, latenţa la DRR este şi funcţie de numărul maxim de conexiuni ce partajează linia de ieşire. WRR poate fi tratat ca un caz special al DRR iar latenţa sa depinde şi ea de numărul maxim de conexiuni ce partajează linia de ieşire.

6.3 O margine îmbunătăţită a întârzierii

Latenţa serverelor *RL* calculată anterior se baza pe ipoteza că un pachet este considerat servil doar când ultimul său bit a părăsit serverul. Astfel latenţa θ_i^s a fost calculată astfel încât serviciul desfăşurat într-o POS la momentul t, $W_{i,j}(\tau,t)$ este totdeauna mai mare sau egal cu $\rho_i(t - \tau - \theta_i^s)$. Deoarece diferenţa maximă dintre $W_{i,j}(\tau,t)$ şi $\rho_i(t - \tau)$ apare chiar înainte ca pachetul sesiunii i să plece din sistem, latenţa θ_i^s este calculată la acest moment. Acest lucru este necesar pentru a fi capabili să limităm sosirile în serverul următor din lanţul de servere; cum serverul nostru nu taie pachetul, un pachet poate fi servit doar după recepţia ultimului său bit. Presupunerea că pachetele pleacă ca un impuls din server ne permite să modelăm sosirile pachetelor în serverul următor tot ca impulsuri.

BUPT

Când calculăm întârzierea cap-la-cap a sesiunii, totuşi, suntem interesaţi să găsim timpul la care ultimul bit al pachetului pleacă din server. Astfel, pentru ultimul server din şir putem determina latenţa θ_i^s funcţie doar de momentele de timp imediat după ce pachetul a fost servit de sesiune. Acesta duce la o valoare mai scăzută a latenţei şi în consecinţă la o limită mai compactă a întârzierii cap-la-cap într-o reţea de servere decât cea dată de ecuaţia (6.41).

Pentru a aplica această idee analiza reţelei se face în două etape. Dacă sesiunea are k salturi, mărginim serviciul oferit sesiunii în primul server k-1, considerând momente arbitrare din POS. în ultimul nod, totuşi, calculăm latenţa doar funcţie de momentul imediat următor momentului în care pachetul şi-a încheiat serviciul.

Cel mai bine este ilustrată ideea considerând un server PGPS. Presupunem că POS începe la momentul τ, şi că pachetul părăseşte serverul PGSP la momentul t_k. Deci, la un server GPS corespunzător, acest pachet pleacă la $t_k - L_{max}/r$ sau mai târziu. Astfel dacă considerăm doar astfel de puncte t_k, putem scrie:

$$W_{i,j}^P(\tau, t_k) \geq W_{i,j}^P\left(\tau, t_k - \frac{L_{max}}{r}\right)$$

$$\geq \rho_i\left(t_k - \tau - \frac{L_{max}}{r}\right)$$

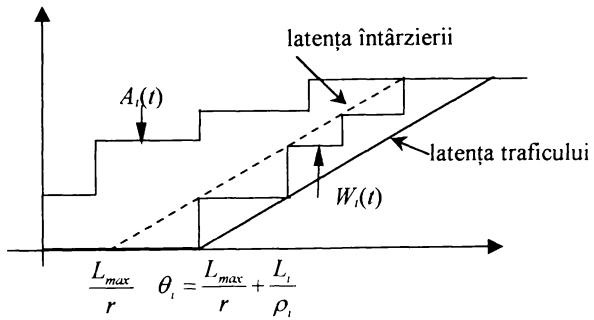


Fig. 6.5 Cele două anvelope folosite pentru a mărgini serviciul primit de sesiunea *i* în POS. Fiecare pas al funcţiei de sosire indică un pachet nou. Anvelopa de jos este marginea inferioară valabilă pentru serviciul în orice moment din POS, iar anvelopa de sus este valabilă doar pentru momentul când pachetul părăseşte sistemul.

Aceasta conduce la o latenţă L_{max}/r faţă de $L_i/\rho_i + L_{max}/r$ calculată cu relaţia (6.64). În figura 6.5 sunt prezentate cele două anvelope rezultate prin limitarea serviciului în cele două moduri. Anvelopa de jos este valabilă pentru punctele arbitrare din POS, iar cea de sus doar pentru punctele când pachetul părăseşte sistemul. Pentru calculul marginii întârzierii cap-la-cap într-o reţea de servere, putem folosi anvelopa de mai sus din ultimul server. În toate planificatoarele cu conservarea lucrului studiate, cele două anvelope sunt

BUPT

separate prin L_i/ρ_i, L_i fiind lungimea maximă de pachet a sesiunii i. Astfel că, pentru aceste servere LR, putem obține o margine mai bună a întârzierii cap-la-cap în rețea, scăzând L_i/ρ_i din ecuația (6.44). Astfel,

$$D_i \le \frac{\tau_i}{\rho_i} + \sum_{j=1}^{k} \theta_i^{(s_j)} - \frac{L_i}{\rho_i} \tag{6.83}$$

Dacă substituim latența obținută pentru PGPS din ecuația (6.64) în această expresie, atunci $\theta_i^{(s_j)} = L_i/\rho_i + L_{max}/r$ şi obținem:

$$D_i \le \frac{\tau_i}{\rho_i} + (k-1)\frac{L_i}{\rho_i} + k\frac{L_{max}}{r}\ , \tag{6.84}$$

care corespunde cu marginea obținută în [PG/2] pentru o rețea de servere PGPS. Cum latențele pentru PGPS şi VCL sunt identice, marginea (6.84) este valabilă şi pentru VCL; acest lucru este în concordanță şi cu rezultatele obținute de Lam şi Xie [LX].

Am verificat că îmbunătățirea cu L_i/ρ_i este valabilă pentru toate serverele LR analizate, dar rămâne o problemă deschisă dacă este adevărată pentru toate serverele LR.

Cu analiza serverelor RL se poate însă furniza o margine mai compactă a întârzierii decât cea din [PG/1, PG/2]. Presupunem că avem o rețea de K servere PGPS la care conexiunii i îi este alocată rata g_i^m în nodul m. Atunci folosind lema 6.5 este uşor de verificat că marginea întârzierii va fi:

$$D_i \le \frac{\tau_i}{min_{m=1}^{K} g_i^m} + \sum_{m=1}^{K-1} \frac{L_i}{g_i^m} + K\frac{L_{max}}{r}\ . \tag{6.85}$$

Deşi timpul necesar pentru a drena rafala maximă este determinat de rata minimă alocată, latențele din comutatoarele individuale sunt determinate de ratele actuale alocate.

6.4. Imparțialitatea serverelor *LR*

În paragraful 6.1 am arătat că în cel mai defavorabil caz comportarea sesiunilor individuale într-o rețea de servere LR poate fi analizată cunoscând doar latențele lor. Dar latența nu oferă informații despre imparțialitate. De exemplu VCL şi PGPS sunt două servere diferite cu aceeaşi latență, dar total diferite în ceea ce priveşte imparțialitatea. De aceea vom analiza imparțialitatea câtorva servere LR bine cunoscute şi le vom compara.

Parametrul imparțialitate este bazat pe definiția lui Golestani pentru analiza SCFQ [GoV]. Presupunem că $W_i^s(\tau, t)$ este serviciul oferit conexiunii în

BUPT

intervalul $(\tau,t]$ de către serverul S. Dacă ρ_i este banda alocată conexiunii i, fracția $W_i^S(\tau,t)/\rho_i$ atunci este serviciul normalizat oferit conexiunii i în intervalul $(\tau,t]$. Un planificator este perfect imparțial dacă diferența dintre serviciul normalizat oferit oricăror două conexiuni continuu conectate în sistem în intervalul $(\tau,t]$ este zero. Aceasta înseamnă că:

$$\left|\frac{W_i^S(\tau,t)}{\rho_i}-\frac{W_j^S(\tau,t)}{\rho_j}\right|=0.$$

Multiplexarea GPS s-a demonstrat că are această proprietate. Totuşi această condiție nu poate fi întâlnită la orice algoritm pachet-cu-pachet deoarece pachetele trebuie să fie exclusiv servite. La serverele pachet-cu-pachet deci, putem pretinde doar ca diferența dintre serviciul normalizat primit de conexiuni să fie mărginită printr-o constantă.

Diferența dintre serviciul normalizat oferit oricăror două conexiuni a fost propusă ca măsură a imparțialității algoritmului [SV1, SV2]. Mai precis, un algoritm este considerat aproape imparțial, pentru oricare două conexiuni i, j continuu conectate într-un interval de timp $(t_1,t_2]$, dacă:

$$\left|\frac{W_i^S(t_1,t_2)}{\rho_i}-\frac{W_j^S(t_1,t_2)}{\rho_j}\right|=F^S.$$

unde F^S este o constantă pe care o denumim imparțialitatea serverului S. Apar însă dificultăți în folosirea precedentei definiții la compararea imparțialității diferitelor planificatoare. Pentru aceeaşi structură a sosirii sesiunilor, PCS poate varia funcție de planificator. Compararea imparțialității diferitelor planificatoare poate conduce la rezultate eronate, dacă structura intrărilor nu este aleasă astfel încât să ducă la aceleaşi PCS în toate planificatoarele. Astfel, modificăm puțin definiția anterioară a imparțialității. Considerăm momentul τ la care conexiunile i şi j comparate au o alimentare infinită cu pachete. Acest lucru le forțează să fie continuu conectate în servere, indiferent de algoritmul de planificare folosit. Ca măsură a imparțialității folosim diferența dintre serviciul normalizat oferit de cele două conexiuni pentru orice interval de timp.

De exemplu tipic de nonimparțialitate apare la algoritmul VCL (vezi figura 6.6) presupunem că cele două conexiuni partajează o linie de ieşire şi au alocate părți egale din lățimea de bandă a liniei. Presupunem că fiecare pachet are lungimea unitară şi rata serverului este tot unitară. Considerăm un interval de timp 0 – 1000 în care doar conexiunea 1 este activă şi trimite 1000 de pachete. Conexiunea 2 devine activă la momentul 1000 şi după momentul 1000 ambele conexiuni transmit pachete. Presupunem că planificatorul este de tip VCL. Cum serverul este cu conservarea lucrului, el va servi toate cele 1000 de pachete de la conexiunea 1 până la t *1000*.Totuşi la t *1000*, următorul pachet al conexiunii 1

BUPT

va primi eticheta de timp 2000, reflectând serviciul mediu primit de conexiune până la 1000. Astfel conexiunea 1 va fi blocată până la 1500, când etichetele de timp ale celor două conexiuni devin egale. Dacă gradul maxim de rafală al celor două conexiuni nu este mărginit, putem vedea că intervalul de timp în care unei conexiuni conectate i se refuză serviciul creşte la infinit. Un server PGPS prin contrast, furnizează un serviciu egal celor două conexiuni după $t=1000$, indiferent de excesul de bandă primită anterior de conexiunea 1.

Vom evalua acum imparţialitatea câtorva servere cunoscute. Cu L_i notăm din nou dimensiunea maximă a pachetelor sesiunii i, respectiv cu L_{max} dimensiunea maximă a pachetelor faţă de toate sesiunile. Pentru VCL, am văzut deja că nu există o valoare finită pentru F^S care să satisfacă definiţia noastră a imparţialităţii.

6.4.1 Imparţialitatea planificatorului PGPS

Imparţialitatea serverului PGPS este dată în funcţie de măsura precedentă a imparţialităţii, în lema următoare:

Lemma 6.12. Pentru un planificator PGPS,

$$F^S = max\left(max\left(C_j + \frac{L_{max}}{\rho_i} + \frac{L_j}{\rho_j}, C_i + \frac{L_{max}}{\rho_j} + \frac{L_i}{\rho_i} \right) \right),$$

unde C_i este serviciul normalizat maxim pe care o sesiune îl poate primi într-un server PGPS în exces faţă de acela din serverul GPS, ci este dat de relaţia:

$$C_i = min\left((V-1)\frac{L_{max}}{\rho_i}, \underset{1 \le n \le V}{max}\left(\frac{L_n}{\rho_j} \right) \right).$$

Se poate arăta că marginea este compactă.

6.4.2 Imparţialitatea planificatorului SCFQ

Pentru SCFQ a fost stabilită următoarea margine [SV]

$$F^S = \frac{L_i}{\rho_i} + \frac{L_j}{\rho_j}$$

Vom demonstra că această limită este compactă printr-un exemplu în care se atinge limita. Presupunem că la momentul t_1, pachetul k de la conexiunea i a fost tocmai servit de serverul SCFQ. Următorul pachet va avea timpul virtual de terminare egal cu,

BUPT

$$F_i^{k+1} = F_i^k + \frac{L_i}{\rho_i}.$$

Pachetele de la conexiunea j pot fi servite după pachetul k al conexiunii i dacă timpul lor virtual de sfârşit este mai mic sau egal cu F_i^{k+1}. Presupunem acum că conexiunea j are un pachet x de lungime L_j cu acelaşi timp de sfârşit F_i^k. De asemenea presupunem că j are un set de pachete în coadă în urma lui x cu dimensiunea totală $L_i\rho_j/\rho_i$. Ultimul din aceste pachete va avea un timp virtual de sfârşit.

$$F_j^m = F_i^k + \frac{L_i}{\rho_i}.$$

Astfel, un volum total de trafic egal cu $L_j + L_i\frac{\rho_j}{\rho_i}$ va putea fi servit de la conexiunea j înainte ca pachetul $k+1$ al conexiunii i să fie servit. Fie t_2 momentul la care pachetele lui j îşi termină serviciul. Atunci,

$$\frac{W_j(t_1,t_2)}{\rho_j} - \frac{W_i(t_1,t_2)}{\rho_i} = \frac{1}{\rho_j}\left(L_j + \frac{L_i}{\rho_i}\rho_j\right)$$

$$= \frac{L_i}{\rho_i} + \frac{L_j}{\rho_j}.$$

6.4.3 Imparţialitatea planificatorului RR

Planificatorul carusel cu recuperarea deficitului DRR, propus în [SV] este un algoritm *0(1)* pentru asigurarea garanţiilor asupra lăţimii de bandă în comutatoare cu memorarea pachetelor la ieşire. DRR este o generalizare a WRR [KKK] propus pentru reţele ATM. WRR presupune că pachetele de la toate conexiunile au aceeaşi dimensiune şi conexiunile sunt servite în ordine RR. Timpul este divizat în cadre de lungime maximă F şi o conexiune nu are voie să transmită mai mult de ϕ_i pachete într-un cadru. Astfel banda alocată conexiunii este $\rho_i \geq \phi_i \frac{r}{F}$. DRR estimează serviciul oferit conexiunii într-o rundă în termeni de octeţi şi nu de pachete, permiţând astfel lucrul cu dimensiuni variabile de pachete. Conexiunile sunt servite în ordine RR. Odată o conexiune selectată, pot fi transmişi până la ϕ_i octeţi de la această conexiune. Dacă transmisia unui pachet întreg forţează conexiunea să transmită mai mult de ϕ_i pachete, ultimul pachet nu este transmis şi contorul de deficite D_i^k este actualizat indicând

BUPT

serviciul conexiunii omis în runda k. Acest deficit este apoi adăugat lui ϕ_i în runda următoare de servire a conexiunii i.

O altă variantă de RR este planificarea carusel cu surplus SRR (Surplus Round Robin) [FJ1, FJ/2, FJ/3]. Diferența față de DRR este că dacă un pachet forțează conexiunea să transmită mai mult decât ϕ_i octeți într-un cadru, pachetul este transmis, dar se actualizează un contor de surplus S_i^k care indică cantitatea de servicii oferite în exces conexiunii i în runda k. Acest serviciu în exces va diminua serviciul din runda următoare.

La DRR s-a arătat că, diferența dintre serviciile oferite oricăror două conexiuni care au aceeaşi rezervare de bandă este mărginită de $3\phi_i$, unde ϕ_i este numărul de octeți alocați acestor conexiuni în fiecare cadru. Vom extinde acum rezultatul pentru două conexiuni cu alocări arbitrare de bandă.

Lemma 6.13. Pentru un planificator DRR,

$$F^S = \frac{3F}{r}.$$

Pentru demonstrație presupunem un interval de timp $(t_0, t_n]$ unde t_0 şi t_n sunt începutul, respectiv sfârşitul rundei n. Pentru fiecare $k = 1, 2, \ldots, n$ când conexiunea este continuu conectată.

$$W_i(t_{k-1}, t_k) = \phi_i + D_i^{k-1} - D_i^k, \tag{6.86}$$

unde D_i^k este valoarea contorului de deficit a conexiunii i la sfârşitul rundei k. Însumând după k,

$$W_i(t_0, t_k) = k\,\phi_i + D_i^0 - D_i^k. \tag{6.87}$$

Dacă $D_i^k \le \phi_i$ pentru orice k, putem scrie,

$$W_i(t_0, t_k) \le k\,\phi_i + \phi_i. \tag{6.88}$$

Similar, dacă conexiunea j este continuu conectată pe acelaşi interval

$$\begin{aligned} W_j(t_0, t_k) &= k\,\phi_j + D_j^0 - D_j^k \\ &\ge k\,\phi_j - \phi_j \end{aligned} \tag{6.89}$$

Din ecuația (6.88) şi (6.89) se deduce simplu că:

$$\left(\frac{W_i(t_0, t_k)}{\rho_i} - \frac{W_j(t_0, t_k)}{\rho_i}\right) \le \frac{\phi_i}{\rho_i} + \frac{\phi_j}{\rho_j} \le \frac{2F}{r}.$$

BUPT

Această margine se aplică intervalelor de timp care cuprind un număr întreg de cadre. Pentru un interval arbitrar, diferența dintre serviciile oferite celor două conexiuni poate crește cu ϕ_i sau ϕ_j octeți. Normalizând această diferență suplimentară, obținem o margine generală a imparțialității,

$$F^S = \frac{3F}{r}. \tag{6.90}$$

În cazul WRR, unde deficitul este mereu zero, extinderea demonstrației precedente conduce la următorul corolar.

Corolarul 6.6. Pentru un planificator WRR

$$F^S = \frac{F}{r}.$$

În concluzia acestui capitol putem spune că, cu excepția VCL, algoritmii de planificare studiați pot fi considerați imparțiali conform cu definiția enunțată aici a imparțialității. Este interesant de remarcat că SCFQ este cea mai imparțială disciplină de planificare pachet-cu-pachet, în anumite cazuri mai bună chiar decât PGPS. Pe de altă parte, latența serverului SCFQ poate fi mult mai mare decât latența serverului PGPS, deoarece SCFQ poate întârzia servirea conexiunii dacă aceasta se conectează după un interval de pauză, în timp ce PGPS îşi penalizează conexiunile care şi-au primit deja banda alocată, pentru a servi noile conexiuni.

6.5 Concluzii

În tabelul 6.2. au fost trecute caracteristicile unor algoritmi de planificare din clasa *LR*, în funcție de trei parametri: latență, imparțialitate şi complexitate.

Se vede că planificatorul PGPS are cea mai bună performanță, atât pentru latență cât şi imparțialitate, dar are cea mai mare complexitate de implementare. Deşi VCL are latența identică cu PGPS nu este un algoritm imparțial. Ceilalți algoritmi au nonimparțialitatea mărginită, dar au latențe mai mari decât PGPS. Din analiza serverelor *LR* rezultă clar că, creşterea latenței duce la întârzieri limită cap-la-cap crescute, un necesar crescut de tampoane de memorie în comutatoare şi un grad de rafală crescut în rețea. Chiar în cazul surselor cu trafic CBR, sesiunile pot acumula rafale mari după mai multe treceri prin rețea, dacă serverele au latențe mari. Astfel, este foarte important să se folosească servere cu latența mică, în rețelele de bandă largă. Atât la serverele SCFQ cât şi RR, latența şi imparțialitatea sunt puternic afectate de numărul de conexiuni care partajează linia de ieşire. Acest lucru face dificil controlul întârzierii cap-la-cap al sesiunii, în rețelele în care un număr mare de fluxuri pot partaja linia.

BUPT

Tabelul 6.2. Caracteristicile planificatoarelor din clasa *LR*

Server	Latență	Imparțialitatea	Complexitatea
GPS	0	0	-
PGPS	$\frac{L_i}{\rho_i} + \frac{L_{max}}{r}$	$max\left(C_j + \frac{L_{max}}{\rho_i} + \frac{L_j}{\rho_j}, C_i + \frac{L_{max}}{\rho_i} + \frac{L_i}{\rho_i}\right)$, unde $C_i = min\left((V-1)\frac{L_{max}}{\rho_i}, \max_{1 \le n \le V}\left(\frac{L_n}{\rho_n}\right)\right)$	$O(V)$
SCFQ	$\frac{L_i}{\rho_i} + \frac{L_{max}}{r}(V-1)$	$\frac{L_i}{\rho_i} + \frac{L_j}{\rho_j}$	$O(\log V)$
VCL	$\frac{L_i}{\rho_i} + \frac{L_{max}}{r}$	∞	$O(\log V)$
DRR	$\frac{3F - 2\phi_i}{r}$	$\frac{3F}{r}$	$O(1)$
WRR	$\frac{F - \phi_i + L_c}{r}$	$\frac{F}{r}$	$O(1)$
FQ cu cadre	$\frac{L_i}{\rho_i} + \frac{L_{max}}{r}$	$\frac{2F}{r} + max\left(\frac{L_i}{\rho_i}, \frac{L_j}{\rho_j}\right)$	$O(\log V)$

L_i este dimensiunea maximă a pachetelor sesiunii *i*, respectiv L_{max} pentru toate sesiunile. C_i este serviciul normalizat maxim pe care o sesiune o poate primi în exces de la un server PGPS. În WRR și DRR, F este dimensiunea cadrului și ϕ_i volumul de trafic în cadre alocate sesiunii *i*. L_c este dimensiunea fixă a pachetelor (celule) la WRR.

Comparația planificatoarelor, realizată în acest capitol, după cei trei parametri, lasă deschisă problema proiectării unui algoritm de planificare cu aceeași parametri ca PGPS.

Abordarea adoptată de a modela întreaga rețea de servere *LR* ca un singur server *LR* nu impune nici o condiție traficului de intrare. Deși marginile pentru întârzierea cap la cap și necesarul de tampoane de memorie, au fost deduse în ipoteza unui trafic de intrare format de o găleată găurită, același model poate fi folosit cu alte modele de trafic de intrare.

Analiza imparțialității din § 6.4 nu are în vedere intervalele de timp în care conexiunea este inactivă, care trebuie cuprinse într-o analiză mai completă. Într-un algoritm imparțial, serviciul pierdut de două conexiuni când sunt inactive trebuie să fie și el proporțional cu rezervările lor. În plus, dacă o conexiune este inactivă și altele sunt conectate, prima trebuie să piardă un serviciu normalizat proporțional cu serviciul pe care-l primesc celelalte. Lucrările viitoare trebuie să includă o metodă universală de estimare a imparțialității algoritmilor de planificare. Această măsură trebuie să depindă doar de caracteristicile serverului și nu de parametrii traficului sesiunilor individuale. De altfel ar trebui să fie suficientă demonstrația că acel server furnizează un serviciu conexiunii comparabil cu cel al multiplexorului GPS.

BUPT

Capitolul 7

SERVERE PROPORȚIONALE CU RATA

Aşa cum s-a arătat în capitolul precedent, aproximarea pachet cu pachet a GPS (WFQ) are latența cea mai redusă dintre serverele pachet cu pachet, cele mai reduse margini ale întârzierii cap la cap şi necesarul de tampoane de memorie este minim. Dezavantajul WFQ este însă complexitatea cea mai mare a implementării VCL are aceeaşi latență ca WFQ dar nu este imparțial. Planificatoarele SCFQ şi RR furnizează o margine a nonimparțialității, dar latența lor depinde de numărul de conexiuni care partajează linia de ieşire. În rețelele de bandă largă, marginea întârzierii cap la cap devine prohibitiv de mare.

Rămâne o problemă crearea unui algoritm de planificare care să combine întârzierea şi gradul de rafală a lui WFQ, calculul simplu al etichetelor de timp şi mărginirea nonimparțialității. În acest capitol vom dezvolta un cadru analitic pentru dezvoltarea unui asemenea algoritm şi vom analiza proprietățile sale.

7.1 Funcțiile de potențial

Planificatorul GPS asigură o imparțialitate ideală, oferind acelaşi serviciu normalizat tuturor conexiunilor conectate, la orice moment de timp. Dacă se reprezintă serviciul total primit de fiecare sesiune ca o funcție, denumită timp virtual, VT (Virtual Time), [ZK, XL], ea va creşte cu aceeaşi rată pentru orice sesiune conectată şi anume rata serviciului normalizat furnizat ei de planificator la acel moment. Similar, putem defini o funcție globală de timp virtual, GVT care creşte cu rata serviciului total produs de planificator la fiecare moment din perioada ocupată a severului. La GPS, VT al tuturor conexiunilor conectate este identic la orice moment şi egal cu GVT. Acest lucru se obține inițializând VT al conexiunii cu GVT când aceasta devine conectată şi apoi crescând VT cu rata serviciului normalizat instantaneu primit de conexiune pe timpul conectării sale. Acest lucru permite unei conexiuni inactive să primească imediat serviciu când devine conectată, rezultând o latență nulă. Această funcție pe care o vom denumi potențial, P, reprezintă starea fiecărei conexiuni din planificator. Potențialul unei conexiuni este o funcție nedescrescătoare în timpul POSIS. Când o conexiune i se conectează, P_i creşte funcție de serviciul normalizat primit: dacă cu $P_i(t)$ se notează potențialul conexiunii i la momentul t, atunci în intervalul $(\tau, t]$ al PCS al sesiunii i

BUPT

$$P_i(t) - P_i(\tau) = \frac{W_i(\tau, t)}{\rho_i}.$$

Potențialele tuturor conexiunilor se pot inițializa pe zero la începutul POSIS, deoarece toate informațiile de stare se şterg când sistemul devine inactiv.

Din definiția potențialului este clar că un algoritm imparțial trebuie să aştepte să crească potențialele tuturor conexiunilor conectate la aceeaşi rată, rata de creşterea potențialului sistemului. Obiectul de bază este deci egalizarea potențialului fiecărei conexiuni, ceea ce încearcă de fapt să facă planificatoarele cu sortarea priorităților, ca GPS, WFQ, SCFQ şi VCL. Definiția potențialului ne spune însă cum se actualizează P când conexiunea este inactivă, doar că P este nedescrescător. Algoritmii de planificare actualizează diferit P al conexiunilor inactive. Ideal este ca, în timpul cât conexiunea i, C_i, este neconectată, P_i să crească cu serviciul normalizat pe care C_i l-ar primi dacă ar fi conectată. În acest caz când C_i devine din nou ocupată P_i va fi identic cu al altor conexiuni conectate în sistem, ceea ce-i permite să fie imediat servită.

O cale de actualizare a potențialului conexiunii când ea devine conectată este de a defini $P(t)$ potențialul sistemului care urmăreşte cum progresează lucrul total al planificatorului. $P(t)$ este o funcție nedescrescătoare de timp. Când C_i inactivă, devine conectată la momentul t, $P_i(t)$ poate fi inițializată la $P(t)$ pentru a ține cont de serviciul pierdut de ea. Planificatoarele folosesc diferite funcții pentru a menține $P(t)$ ceea ce duce la diferite comportări ale întârzierii şi ale imparțialității. În general, potențialul sistemului la un moment t poate fi definit ca o funcție nedescrescătoare, de potențialele conexiunii individuale, înainte de t, şi de t.

$$P(t) = \boldsymbol{F}(P_1(-t), P_2(-t), \ldots P_V(-t), t) \qquad (7.1)$$

Serverul GPS, de exemplu, inițializează potențialul conexiunilor nou create la cel al conexiunii curente conectate în sistem. Aceasta înseamnă că $P(t)$ $P_i(t)$, pentru orice $i \in B(t)$, unde $B(t)$ este setul de conexiuni conectate la momentul t. Planificatorul VCL inițializează potențialul conexiunii la timpul real când aceasta devine conectată astfel că $P_2(t) - P_1(t) = t_2 - t_1$.

7.2 Servere proporționale cu rata, SPR

Pentru a defini o nouă clasă generală de servere proporționale cu rata, SPR, definim mai întâi aceste planificatoare în modelul cu fluid şi apoi extindem definiția la modelul pachet cu pachet. Aceste planificatoare sunt caracterizate prin disciplina de servire, care reglează rata instantanee a serviciului pentru sesiunile individuale conectate, astfel încât să le egalizeze potențialele. Definiția cere ca potențialul $P(t)$ al sistemului să fie menținut la, sau dedesubtul potențialului oricărei conexiuni, în orice moment în care serverul este ocupat.

BUPT

Acest lucru asigură ca noile conexiuni să nu aibă un potențial inițial mai mare decât celelalte conexiuni şi deci să primească serviciu imediat. Astfel SPR este un server de latență nulă. Dar definiția nu indică modul de sintetizare a lui $P(t)$, care este diferit de exemplu pentru GPS şi VCL care sunt şi ele servere RPS. Dar SCFQ nu este SPR deoarece nu satisface condiția asupra potențialului sistemului.

Pentru a defini formal clasa SPR, notăm cu $B(t)$ setul de conexiuni conectate la momentul t. astfel SPR are următoarele proprietăți:

1. rata ρ_i este alocată conexiunii i şi $\sum_{i=1}^{V} \rho_i \le r$, unde r este rata totală a serviciului serverului.
2. Potențialul unei conexiuni $P_i(t)$, descrie starea conexiunii la momentul t, şi satisface proprietățile:
   a) $P_i(t)$ rămâne constant când conexiunea i e neconectată;
   b) Când conexiunea devine conectată la momentul t,

$$P_i(\tau) = max\left(P_i(-\tau), P(\tau -)\right) \tag{7.2}$$

   c) Pentru orice $t > \tau$, conexiunea rămâne conectată şi $P_i(t)$ este crescută cu serviciul normalizat oferit conexiunii în intervalul $(\tau, t]$, deci

$$P_i(t) = P_i(\tau) + \frac{W_i(\tau, t)}{\rho_i} \tag{7.3}$$

3. Potențialul $P(t)$ al sistemului descrie starea acestuia la momentul t, iar $P(t)$ satisface condițiile:
   a) Pentru orice interval $(t_1, t_2]$ din POSIS

$$P(t_2) - P(t_1) \ge t_2 - t_1 \tag{7.4 a}$$

   b) $P(t)$ este întotdeauna mai mic sau egal decât potențialele conexiunilor conectate la momentul t,

$$P(t) \le \min_{j \in B(t)} \left(P_j(t)\right) \tag{7.4 b}$$

4. La orice moment t, conexiunile sunt servite conform potențialelor lor instantanee după regulile:
   a) Dintre toate conexiunile conectate sunt servite cele de potențial minim la acel moment t,
   b) Fiecare conexiune din set este servită cu o rată instantanee proporțională cu rezervarea sa, deci $P_i(t)$ creşte cu aceeaşi rată.

Definiția specifică condițiile funcției $P(t)$ care să permită proiectarea unui server de latență nulă, dar nu şi modul de implementare, care trebuie să fie eficient.

BUPT

Perioada activă a unei sesiuni i, PAS_i, este intervalul maxim de timp pe durata POSIS, în care potențialul sesiunii nu este mai mic decât cel al sistemului, celelalte perioade fiind considerate inactive. Conceptul de PAS este util pentru analiza comportării PRS. O conexiune nu este conectată în perioada inactivă şi deci nu poate primi serviciu. PAS este aceeaşi cu POS, deoarece $P_i(t) < P(t)$ doar când conexiunea este nefolosită; trecerea de la starea inactivă la cea activă are loc doar la sosirea unui pachet. O conexiune activă poate să nu fie servită în PAS dacă alte conexiuni au un potențial mai scăzut. Planificatorul serveşte în orice moment conexiunile de potențial minim. Conexiunea va fi întotdeauna servită la începutul PAS dacă $P_i(t) = P(t)$.

Dacă o conexiune cu potențial mai mic devine activă, se suspendă servirea celorlalte conexiuni până ce acesta ajunge la aceeaşi potențial cu al celorlalte. Dacă o conexiune îşi termină serviciul, rata instantanee a celorlalte conexiuni va creşte, deoarece planificatorul conservă lucrul; dacă o conexiune a primit mai multă bandă în PAS decât cea alocată în perioada activă anterioară i se suspendă lucrul.

Deoarece serverele *LR* sunt definite funcție de POS, trebuie stabilită corespondența POS – PAS, şi vom vedea că începutul POS este acelaşi cu al PAS pentru SPR (lema 7.1). Demonstrația sa se va face prin reducere la absurd. Presupunem că începuturile diferă $\tau_{POS} \neq \tau_{PAS}$. Apar două cazuri: i) τ aparține unei perioade inactive. Dar, τ fiind începutul POS, poate sosi un pachet, ceea ce înseamnă că $P_i(t)$ poate deveni egal cu $P(t)$ ceea ce înseamnă că $\tau_{POS} = \tau_{PAS}$. ii) PAS începe la $\tau_0 < \tau$ şi deci pentru $\forall t \in (\tau_0, \tau]$ trebuie să avem:

$$P_i(t) \geq P(t). \tag{7.5}$$

Pe intervalul $(\tau_0, \tau]$, P_i creşte cu serviciul normalizat oferit sesiunii i, astfel că pentru $\forall t \in (\tau_0, \tau]$,

$$P_i(t) - P(\tau_0) = \frac{W_i(\tau_0, t)}{\rho_i}. \tag{7.6}$$

Dar τ_0 este începutul PAS şi deci:

$$P_i(\tau_0) = P(\tau_0). \tag{7.7}$$

Din ecuația (7.5) şi (7.7)

$$\begin{aligned} P_i(t) &= P(\tau_0) \geq P(t) - P(\tau_0) \\ &\geq (t - t_0). \end{aligned} \tag{7.8}$$

Astfel din ecuația (7.6) şi (7.8),

$$W_i(\tau_0, t) \geq \rho_i (t - \tau). \tag{7.9}$$

Cum înainte de τ_0 sistemul nu a fost conectat, rezultă că:

BUPT

$$A_i(\tau_0, t) \geq W_i(\tau_0, t) \geq \rho_i (t - \tau_0). \qquad (7.10)$$

Deci τ_0 aparține aceleași POS în orice moment $\forall t \in (\tau_0, \tau]$ și înseamnă că τ nu poate fi începutul POS. La activarea unei conexiuni $P_i(t)$ este minim față de oricare $P_j(t)$ al conexiunilor, deci poate fi servită imediat.

Servirea sa poate fi suspendată, dacă o altă conexiune j devine activă în POS_i, până când $P_i(t) = P_j(t)$. Vom deduce marginea inferioară a volumului serviciului primit de conexiunea i în PAS în lema 7.2: dacă τ este începutul PAS_i în SPR, atunci la orice $t > \tau$ și $t \in PAS_i$, serviciul oferit conexiunii i, $W_i(\tau, t) \geq \rho_i (t - \tau)$. Pentru demonstrație considerăm un moment t din perioada activă a conexiunii. Atunci din definiția PAS $P_i(t) \geq P(t)$ și $P_i(\tau) \geq P(\tau)$ iar din ecuația (7.8) și (7.9) rezultă: $W_i(\tau, t) \geq \rho_i (t - \tau)$.

Dar o POS poate consta din PAS multiple. Pentru a demonstra că SPR este un server *LR* de latență nulă trebuie să demonstrăm că pentru orice t după începutul POS la momentul τ, $W_{i,j}(\tau, t) \geq \rho_i (t - \tau)$. Din lemele precedente, ajungem la:

Teorema 7.1: SPR este un server *LR* de latență nulă.

În timpul perioadelor inactive conexiunea nu primește serviciu, nefiind conectată. Conexiunea nu poate primi mai puțin serviciu decât rata alocată doar pe perioada inactivă PIS (perioada inactivă a sesiunii). Cum în PIS nici un pachet nu este în așteptarea serviciului, POS trebuie să se termine cu ea.

Evoluția $P_i(t)$ poate fi trasată împărțind POS în mai multe PAS respectiv PIS. Demonstrația o facem prin reducere la absurd. Presupunem că POS, $(\tau_0, \tau]$ și $t^* \in (\tau, t]$, t^* fiind primul moment în care,

$$W_i(\tau, t^*) < \rho_i (t^* - \tau).$$

Dacă $t^* \in PAS$ și t_a este începutul PAS, știm din lema 7.1 că $t_a \geq \tau$, și cum $t^* > t_a$ rezultă că:

$$W_i(\tau, t_a) \geq \rho_i (t_a - \tau).$$

Din lema 4.2 știm că pentru t^* aparținând aceleași PAS

$$W_i(t_a, t^*) \geq \rho_i (t^* - t_a).$$

Din ultimele două ecuații, deducem că:

$$W_i(\tau, t^*) \geq \rho_i (t^* - \tau). \qquad (7.11)$$

Dacă $t^* \in PIS$, considerăm momentul $t^* - \Delta t$. Știm că:

BUPT

$$W_i(\tau, t^* - \Delta t) \geq \rho_i (t^* - \Delta t - \tau).$$

În PIS nu sunt pachete conectate deci,

$$W_i(\tau, t^* - \Delta t) = A_i(\tau, t^* - \Delta t),$$

şi nici un pachet nu a fost servit în intervalul $(t^* - \Delta t, t^*]$, deci:

$$W_i(\tau, t^* - \Delta t) = W_i(\tau, t^*).$$

În acest interval nu a apărut nici o sosire de pachet, fiindcă astfel s-ar fi trecut în PAS, deci

$$A_i(\tau, t^* - \Delta t) = A_i(\tau, t^*).$$

Din ecuaţia (7.11) şi următoarele trei ecuaţii rezultă:

$$A_i(\tau, t^*) < \rho_i(t^* - \tau), \tag{7.12}$$

ceea ce înseamnă că t^* nu poate aparţine aceleaşi POS ca $t^* - \Delta t$.

Astfel definiţia SPR nu oferă calea de proiectare algoritmilor de planificare cu latenţă nulă. Cum şi GPS şi VCL pot fi considerate SPR, conform teoremei 7.1, ele au aceeaşi comportare a întârzierii în cazul cel mai defavorabil.

7.2.1 Servere proporţionale cu rata, pachet cu pachet

La SPR definite anterior conform modelului cu fluid, pe presupunerea că pachetele de la conexiuni diferite pot fi servite în acelaşi timp, cu rate diferite. În sistemele reale însă, la un moment dat doar o conexiune poate fi servită şi pachetele nu pot fi divizate. Un server proporţional cu rata pachet cu pachet SPRP pot fi definite similar, ca SPR care transmite pachetele în ordinea crescătoare a timpilor lor de sfârşit. Dacă TSI este potenţialul conexiunii i când un pachet îşi sfârşeşte serviciul în serverul cu fluid, acest TS_i va fi folosit ca etichetă de timp, pachetele fiind planificate şi transmise în ordinea crescătoare a acestor etichete.

Folosim din nou notaţia L_i pentru lungimea maximă a pachetului sesiunii i, respectiv L_{max} lungimea maximă a pachetelor tuturor sesiunilor. Pentru a analiza performanţa SPRP comparativ cu SPR cu fluid presupunem că ambelor li se oferă aceeaşi structură de date. Cu $W_i^F(\tau, t)$ notăm serviciul oferit sesiunii i în intervalul $(\tau, t]$ în serverul cu fluid, respectiv $W_i^P(\tau, t)$ în cel pachet cu pachet. Presupunem că pachetul k părăseşte cele două servere la momentul t_k^F respectiv t_k^P. Conform cu [PG/1, PG/2] se poate demonstra lema 4.3 care spune că pentru toate pachetele SPRP,

BUPT

$$t_k^P \le t_k^F + \frac{L_{max}}{r}. \tag{7.13}$$

Dacă ţinem cont de serviciul parţial primit de pachete în transmisie, întârzierea maximă în serviciu pentru o sesiune i la SPRP apare când pachetul îşi începe serviciul. Dacă cu $\hat{W}_i(t)$ este serviciul oferit conexiunii i la momentul t, inclusiv serviciul parţial, atunci la momentul când pachetul k îşi începe serviciul:

$$\begin{aligned} \hat{W}_i^F(0,t_k) &\le \hat{W}_i^F\left(0,t_k - \frac{L_{max}}{r}\right) + L_{max} \\ &\le \hat{W}_i^P(0,t_k) + L_{max}. \end{aligned} \tag{7.14}$$

Putem enunţa acum corolarul (7.1): la orice moment t

$$\hat{W}_i^F(0,t) - \hat{W}_i^P(0,t) \le L_{max}. \tag{7.15}$$

Pachetele în SPRP sunt servite în ordinea crescătoare a TSI. Dacă pachetele de la mai multe conexiuni au acelaşi TS, pentru transmisie va fi selectat unul, determinând ca sesiunea să primească temporar un serviciu suplimentar faţă de SPR. Pentru a determina diferenţa dintre serviciile primite şi marginea ei, folosim următoarea lemă:

Lemma 4.4 Dacă $(0,t]$ este POS în SPR şi sesiunea i e conectată la momentul t astfel încât i primeşte un serviciu suplimentar în SPRP în acest interval, atunci există o altă sesiune j, cu $P_j(t) \le P_i(t)$ care primeşte un serviciu suplimentar în SPRP în acelaşi interval.

Cum ambele servere sunt cu conservarea lucrului, dacă sesiunea i notată cu S_i primeşte un serviciu suplimentar în SPRP, atunci trebuie să existe şi o altă sesiune conectată S_j care primeşte mai puţin serviciu în $(0,t]$. Deci trebuie să demonstrăm că $P_j(t) \le P_i(t)$. În primul caz, dacă S_i are potenţialul maxim în SPR, atunci pentru oricare j, $P_j(t) \le P_i(t)$. În al doilea caz, există şi alte sesiuni S cu potenţial mai mare ca P_i, la momentul t. Atunci, din definiţia SPR, aceste sesiuni S nu primesc serviciu la momentul t. Dacă τ a fost ultimul moment la care una din sesiunile S a fost în serviciu în SPR, atunci în $(\tau,t]$ nici o conexiune din S nu este servită. Deci,

$$W_k^F(\tau,t) \ge W_k^P(\tau,t), \quad \forall k \in S. \tag{7.16}$$

În plus, S_i nu a fost conectată în SPR, fiindcă astfel, o conexiune din S nu ar fi putut fi servită chiar înainte de τ. Astfel:

$$W_i^F(0,\tau) \ge W_i^P(0,\tau). \tag{7.17}$$

BUPT

Dar după cum ştim, la momentul t, S_i a primit mai mult serviciu în SPRP, astfel că:

$$\hat{W}_i^F(0,t) < \hat{W}_i^P(0,t). \tag{7.18}$$

Scăzând ultimele două relaţii obţinem:

$$\hat{W}_i^F(\tau,t) < \hat{W}_i^P(\tau,t). \tag{7.19}$$

Adică în $(\tau,t]$, S_i a primit mai mult serviciu în SPRP decât în SPR, concluzie care se poate deduce printr-un procedeu similar pentru toate conexiunile S. Dar cum ambele servere sunt cu conservarea lucrului, trebuie să existe cel puţin o conexiune S_j care în $(\tau,t]$ să primească un serviciu în SPR suplimentar faţă de SPRP, deci:

$$\hat{W}_j^F(\tau,t) > \hat{W}_j^P(\tau,t), \tag{7.20}$$

Conexiune care nu face parte din S, deci $P_j(t) \le P_i(t)$, Dar cum S_j se conectează în SPR la τ sau imediat după, deci:

$$\hat{W}_j^F(0,\tau) > \hat{W}_j^P(0,\tau). \tag{7.21}$$

Adunând ultimele două ecuaţii obţinem,

$$\hat{W}_j^F(0,t) > \hat{W}_j^P(0,t). \tag{7.22}$$

Din lemma precedentă şi metoda prezentată în [SV, SV1] pentru un server WFQ vom determina o margine superioară a diferenţei de serviciu primit de o sesiune în SPRP faţă de SPR, care este (lemma 7.5)

$$\hat{W}_i^P(0,t) - \hat{W}_i^F(0,t) \le min\left((V-1)L_{max}, \rho_i \max_{1 \le n \le V}\left(\frac{L_n}{\rho_n}\right)\right) \tag{7.23}$$

Lemma stabileşte două margini superioare distincte pentru serviciul în exces primit de o sesiune într-un server SPRP. Considerăm o sesiune S_i conectată în ambele servere. Atunci S_i^P poate fi întârziată cu cel mult L_{max}. faţă de S_i^F. Astfel în cazul extrem, oricare S_j^P, mai puţin S_i^P poate fi întârziată cu L_{max}. Cum serverul este cu conservarea lucrului, S_i^P poate fi în faţă cu cel mult $(V-1)L_{max}$, V fiind numărul de sesiuni care partajează linia de ieşire.

BUPT

Marginea $(V-1)$ L_{max} poate fi uneori prea vagă, caz în care a doua limită e mai compactă. Presupunem că $L_i/\rho_i = \max_{1 \le n \le V} L_n/\rho_n$ şi că sosesc la server două pachete simultan de la S_i şi S_j, cu acelaşi potenţial asignat de sfârşit. Dacă îşi încep simultan serviciul, la t, vor fi sfârşi simultan la $t + L_i/\rho_i$. Dacă SPRP transmite S_j primul pachet, serviciul primit de S_j în SPRP va fi în avans cu $(L_i - \rho_i)/\rho_j$, ceea ce conduce la o a doua margine superioară.

7.2.2 Analiza întârzierii

Vom demonstra că SPRP este un server LR şi îi vom estima latenţa. Dacă un pachet părăseşte cele două servere la t^P respectiv t^F atunci $t_k^P \le t_k^F + L_{max}/r$. Pentru analiza unei reţele LR trebuie ca serviciul să fie mărginit pentru oricare t după începutul POS. Pachetul se consideră că a părăsit serverul când ultimul său bit a fost transmis, deci chiar înainte de $t^P + L_{max}/r$ acesta nu a plecat încă din SPRP. Dacă L_i e lungimea maximă a pachetului conexiunii C_i, serviciul oferit lui C_i în SPRP va fi egal cu cel oferit în SPR până la momentul t^P minus ultimul pachet. Astfel serviciul primit de C_i în a j-a POS, în SPRP este:

$$\begin{aligned} W_{i,j}^P(\tau, t) &\ge W_{i,j}^F(\tau, t - L_{max}/r) - L_i \\ &\ge max(0, \rho_i (t - \tau - L_{max}/r) - L_i) \\ &\ge max(0, \rho_i - \tau - L_{max}/r - L_i/\rho_i) \end{aligned} \qquad (7.24)$$

Astfel putem enunţa corolarul 7.2: un SPRP este un server LR cu latenţa egală cu $L_{max}/r + L_i/\rho_i$.

După cum se vede latenţa este aceeaşi cu WFQ; deci orice SPRP are aceeaşi margine superioară a întârzierii cap – la – cap şi a necesarului de tampoane de memorie ca WFQ cu trafic format de o găleată găurită.

7.3 Imparţialitatea serverelor proporţionale cu rata

La definirea SPR singura condiţie impusă a fost ca $P(t)$ să aibă latenţa nulă, fără a vedea cum influenţează comportarea planificatorului alegerea lui $P(t)$. Această alegere are o mare influenţă asupra imparţialităţii serviciului sesiunilor. Am arătat că o sesiune conectată în SPR primeşte un serviciu minim egal cu rezervarea sa; dar există mari diferenţe între planificatoare, cu privire la serviciul furnizat pe termen scurt. Cum la SPR conexiunile conectate sunt servite la aceeaşi rată normalizată în starea staţionară, nonimparţialitatea în servire poate apare doar când o sesiune inactivă i se conectează, astfel că dacă $P_i(t) < P(t)$, C_i va fi servită exclusiv, până când P_i ajunge egal cu al celorlalte conexiuni conectate. Notăm cu ΔP diferenţa maximă dintre potenţialul sistemului şi a conexiunilor aflate în serviciu în SPR.

BUPT

Teorema 4.2. Dacă $P(t)$ în SPR nu diferă cu mai mult de $\Delta P < \infty$, finit, față de $P_i(t)$ a conexiunilor servite, atunci diferența între serviciul normalizat oferit oricăror conexiuni $i, j \in B(t_1, t_2)$ în $(t_1, t_2]$ e de asemenea mărginită prin ΔP.

$$\left| \frac{\hat{W}_i(t_1, t_2)}{\rho_i} - \frac{\hat{W}_j(t_1, t_2)}{\rho_j} \right| \le \Delta P \tag{7.25}$$

Pentru demonstrație presupunem că $P_j(t_1) > P_i(t_1)$ şi dacă i este conectată:

$$P_j(t_1) \ge P_i(t_1). \tag{7.26}$$

Mai ştim că:

$$P_j(t_1) \le P(t_1) + \Delta P. \tag{7.27}$$

Cum ambele sesiuni sunt conectate în $(t_1, t_2]$, P_i şi P_j cresc în acest interval doar cu serviciul normalizat oferit celor două conexiuni. Deci

$$P_i(t_2) - P_i(t_1) = \frac{W_i(t_1, t_2)}{\rho_i}, \tag{7.28}$$

$$P_j(t_2) - P_j(t_1) = \frac{W_j(t_1, t_2)}{\rho_j}. \tag{7.29}$$

La momentul t_2, $P_i(t_2) \le P_j(t_2)$ şi notăm

$$x = P_j(t_2) - P_j(t_1) \ge 0. \tag{7.30}$$

Din cele trei ecuații precedente rezultă:

$$\frac{W_i(t_1, t_2)}{\rho_i} \le P_j(t_2) - P(t_1) - x. \tag{7.31}$$

Şi din (7.26) şi (7.28) rezultă:

$$\frac{W_j(t_1, t_2)}{\rho_j} \ge P_j(t_2) - P_j(t_1) - \Delta P. \tag{7.32}$$

Din cele două ecuații precedente rezultă:

BUPT

$$\frac{W_i(t_1,t_2)}{\rho_i} - \frac{W_j(t_1,t_2)}{\rho_j} \le \Delta P - x \le \Delta P . \tag{7.33}$$

Iar dacă $P_j(t_2) \le P_i(t_2)$, deducem la fel că:

$$\frac{W_j(t_1,t_2)}{\rho_j} - \frac{W_i(t_1,t_2)}{\rho_i} \le \Delta P . \tag{7.34}$$

adică,

$$\left| \frac{W_i(t_1,t_2)}{\rho_i} - \frac{W_j(t_1,t_2)}{\rho_j} \right| \le \Delta P . \tag{7.35}$$

Astfel dacă ΔP este finit, diferența serviciilor normalizate oferite celor două sesiuni este mărginită. Această teoremă se aplică SPR, cu model fluid, dar un sistem real lucrează cu pachete. Pentru a o extinde la SPRP definim:

$$C_i = min\left((V-1)\frac{L_{max}}{\rho_i}, \max_{1 \le n \le V}\left(\frac{L_n}{\rho_n} \right) \right), \tag{7.36}$$

serviciul normalizat maxim pe care o conexiune îl poate primi în SPRP în exces față de SPR.

Teorema 7.3. Dacă la SPRP există două conexiuni i şi j care la momentul τ cer un serviciu în exces, atunci pentru orice interval $(t_1,t_2]$, diferența serviciului lor este mărginită,

$$\left| \frac{\hat{W}_j(t_1,t_2)}{\rho_j} - \frac{\hat{W}_i(t_1,t_2)}{\rho_i} \right| \le max\left(\Delta P + C_j + \frac{L_{\max}}{\rho_i} + \frac{L_j}{\rho_j}, \Delta P + C_i + \frac{L_{\max}}{\rho_i} + \frac{L_i}{\rho_i} \right). \tag{7.37}$$

Pentru a deduce o limită a imparțialității algoritmului, definim potențialul SPRP. Notăm cu $a_i(t)$ potențialul $C_i(t)$ la SPRP, definită similar cu a serverului cu model cu fluid. Deci C_i nu primeşte, când este inactivă, acelaşi serviciu. Diferența este că atunci când C_i se conectează potențialul său creşte doar când transmite un pachet. Cu $P_i(t)$ se notează din nou potențialul în modelul cu fluid.

După momentul τ, ambele conexiuni sunt alimentate permanent cu pachete. Potențialul conexiunii SPR este crescut de rata serviciului normalizat oferit, deci pentru serviciul C_i în $(t_1,t_2]$ putem scrie:

$$\frac{W_i(t_1,t_2)}{\rho_i} \le max\left(a_i(t_2) - P_i(t_1)\right) + \frac{L_{max}}{\rho_i} . \tag{7.38}$$

Dacă $P_i(t_2) > P_i(t_1)$ atunci serviciul normalizat oferit lui C_i în $(t_1,t_2]$ este egal cu creşterea de potențial după t_1 plus serviciul primit de C_i suplimentar în SPR față

BUPT

de SPRP; până la t_1. Dacă $P_i(t_2) < P_i(t_1)$, atunci pachetele servite după t_1 în SPR, nu au fost încă servite în SPRP; astfel singurul serviciu oferit C_i a fost deja oferit înainte de t_1 în SPR, serviciu mărginit de L_{max}. pentru C_j putem la fel scrie:

$$\frac{\hat{W}_i(t_1,t_2)}{\rho_i} \geq max\left(a_j(t_2) - P_i(t_1)\right) - C_j \tag{7.39}$$

C_j poate să fi primit mai multe servicii în SPR înainte t_1. Dar diferența potențialelor este mărginită,

$$\left|P_i(t_1) - P_j(t_1)\right| \leq \Delta P \tag{7.40}$$

Dacă presupunem că C_i a primit un serviciu normalizat suplimentar, şi din definiția SPRP, rezultă că:

$$a_i(t_2) \leq a_j(t_2) + L_j/\rho_j \,. \tag{7.41}$$

Din ultimele patru ecuații rezultă:

$$\frac{\hat{W}_i(t_1,t_2)}{\rho_i} - \frac{\hat{W}_j(t_1,t_2)}{\rho_j} \leq \Delta P + C_j + \frac{L_{max}}{\rho_i} + \frac{L_{max}}{\rho_j} \,. \tag{7.42}$$

Similar, dacă C_j a primit un serviciu normalizat suplimentar, ecuația precedentă se poate rescrie inversând i cu j şi rezultă din aceste ultime ecuații, ecuația (7.36). Dar cum WFQ este un SPRP cu $\Delta P = 0$, imparțialitatea sa poate fi dedusă din ecuația (7.36).

7.4 Algoritm de planificare imparțială pentru rețele cu pachete

Vom prezenta un algoritm de planificare cu *0(1)* complexitate de calcul a etichetelor de timp şi aceleaşi margini ale întârzierii cap la cap şi necesar de tampoane de memorie ca WFQ. Definiția SPR nu specifică metoda de menținere a potențialului sistemului. Definiția $P(t)$ permite o mare flexibilitate în aproximarea stării globale a sistemului, folosită în cele ce urmează pentru proiectarea unui algoritm de planificare denumit planificare imparțială bazată pe cadre, PIC. Potențialul sistemului, menținut la fel ca în modelul cu fluid SPR, este recalibrat periodic pentru a corecta discrepanțele.

7.4.1 Metoda de menținere a potențialului sistemului

Problema de bază la SPR este menținerea $P(t)$. Imparțialitatea algoritmului depinde de alegerea funcției $P(t)$, deoarece nonimparțialitatea pe termen scurt depinde de diferența dintre $P(t)$ şi $P_i(t)$ a conexiunilor i conectate.

BUPT

Un algoritm de planificare este un server proporțional cu rata imparțial SPRI, dacă poate fi găsită o constantă finită $\Delta P \geq 0$ astfel încât $P(t) \geq P_i(t) - \Delta P$, pentru $\forall i \in B(t)$, unde $B(t)$ este setul de conexiuni conectate la momentul t. Această condiție poate fi realizată prin folosirea unui mecanism de recalibrare periodică pentru a limita diferența maximă dintre potențialul sistemului și cel al conexiunilor conectate. Valoarea $S^P(t) \leq P_i(t)$, $\forall i \in B(t)$ o vom numi potențial de bază. Poate fi găsită o constantă ΔP finită $\Delta P \geq 0$ astfel încât:

$$S^P(t) \geq P_i(t) - \Delta P, \forall i \in B(t). \tag{7.43}$$

Dacă τ_0 este începutul POS curentă și $\tau_0 < \tau_1 < ... < \tau_k$, se face o recalibrare prin actualizarea $P(t)$ la valoarea $S^P(t)$ dacă $P(t) < S^P(t)$, adică,

$$P(\tau_j) = max\left(S(\tau_j -), S^P(\tau_j)\right), \tag{7.44}$$

cu $(\tau_j -)$ notând momentul anterior actualizării. Între actualizări, $P(t)$ crește liniar:

$$P(t) = P(\tau_j) + S^P(t - \tau_j), \quad \tau_j \leq t \leq \tau_{j+1}, . \tag{7.45}$$

Intervalul între două recalibrări succesive este mărginit:

$$\tau_{j+1} - \tau_j \leq \Delta T, \text{ cu } \Delta T \text{ finit.} \tag{7.46}$$

Imparțialitatea SPR depinde de alegerea $S^P(t)$ și frecvența recalibrării, deci de alegerea momentelor $\tau_0 < \tau_1 < ... < \tau_k$, momente pe care le vom alege egale cu momentele de sfârșit ale serviciului pachetelor. Astfel frecvența recalibrării este mărginită superior de rata de plecare a pachetelor. Dacă $P(t)$ este menținut după regula anterioară atunci, pentru orice interval $(t_1, t_2]$din POS, $P(t_2) - P(t_1) = t_2 - t_1$ și $P(t) \leq P_i(t), \forall i \in B(t)$.

7.4.2 Planificarea imparțială bazată pe cadre

Alegerea $S^P(t)$ și a intervalelor de recalibrare poate conduce la diferiți algoritmi. Alegem perioada maximă de recalibrare egală cu F, dimensiunea cadrului și algoritmul îl vom denumi algoritm de planificare imparțială bazat pe cadre, PIC. Plecăm de la modelul cu fluid și apoi extindem raționamentul pentru cadre. Parametrul F reprezintă numărul de biți transmiși în perioada T a cadrului, adică

$$T \quad F r. \tag{7.47}$$

Notăm cu ϕ_i volumul maxim de trafic pe care C_i îl poate primi într-un interval T, adică:

BUPT

$$\phi_i = r_i \cdot F = \rho_i \cdot T . \tag{7.48}$$

Dacă o conexiune rămâne conectată, potențialul său creşte cu serviciul normalizat oferit ei: deci dacă de la o conexiune C_i sunt serviți ϕ_i biți, potențialul său va creşte cu $T = \phi_i / \rho_i$. Mai impunem o condiție lui ϕ_i şi anume ca cel mai mare pachet L_i al C_i să poată fi transmis în t, deci

$$L_i \le \phi_i . \tag{7.49}$$

Recalibrarea $P(t)$ se face pe cadru, fiecare operație de recalibrare marcând un nou cadru. Dacă conexiunile sunt conectate continuu, actualizările pe cadru în serverul cu fluid se fac la intervalul T de cadru. Actualizările pot apărea mai devreme dacă sosirile de la unele sesiuni sunt mai mici ca rezervările lor, determinând creşterea mai rapidă a potențialului conexiunilor conectate. Astfel în serverul cu fluid actualizarea k poate avea loc când potențialul conexiunilor conectate atinge valoarea kT, lucru care se petrece simultan pentru toate conexiunile. Acest lucru nu este valabil însă în serverul cu pachete. Pentru a evita simularea sistemului cu fluid în vederea determinării momentelor de actualizare, vom defini momentele de recalibrare astfel: dacă τ_{K-1} este momentul ultimei actualizări, atunci următoarea reactualizare va avea loc dacă potențialul tuturor conexiunilor conectate aparțin cadrului următor, adică:

$$P_i(t) \ge kT, \qquad \forall i \in B(t) \tag{7.50}$$

şi

$$P_i(t) \ge (k+1)T, \qquad i = 1, 2, \ldots V \tag{7.51}$$

Condițiile precedente pot fi satisfăcute acum într-o fereastră de timp. Dacă recalibrarea se face la momentul τ_K atunci:

$$P(\tau_K) \leftarrow max\,(P(\tau_K), kT). \tag{7.52}$$

Momentele τ_K de actualizare sunt identice la serverul cu fluid şi cu pachete şi trebuie determinate doar din informațiile disponibile în serverul cu pachete, pentru a cădea în fereastra definită prin (7.50) şi (7.51). Această definiție permite ca recalibrarea să aibă loc doar la sfârşitul sau începutul serviciului unui pachet. Durata cadrului, adică a intervalului dintre două actualizări succesive, nu va depăşi $2T$.

Potențialul de bază $S^P(t)$ pentru PIC este definit ca o funcție treaptă, cu valoarea 0 când serverul este inactiv şi creşte cu T la fiecare calibrare (vezi figura 7.1). La momentele τ_1, τ_2, τ_3 şi τ_4 apar recalibrări. Aceste momente pot cădea oriunde în fereastra în care $P_i(t) \in [kT, (k+1)T]$.

Momentele actualizării pot fi determinate în serverul cu pachete doar din etichetele de timp ale pachetelor din cozi, etichete care indică de fapt potențialul

BUPT

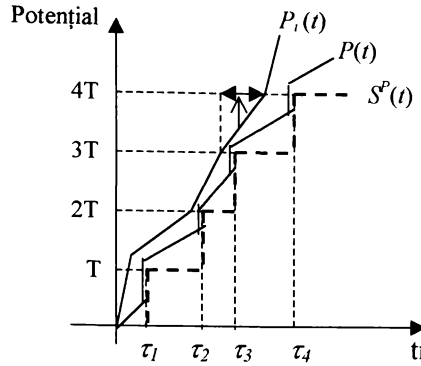


Fig. 7.1 Comportarea $S^P(t)$ şi $P(t)$ într-un server PIC

corespunzător al conexiunii când pachetul îşi va fi încheiat serviciul.

Potențialul de pornire al pachetului j al conexiunii i, s_i^j este potențialul conexiunii când pachetul j îşi începe serviciul în serverul cu fluid corespunzător, iar $S_i(t)$ este potențialul de pornire al conexiunii la momentul t. Cu $\hat{B}(t)$ notăm setul de conexiuni conectate la momentul t în serverul cu pachete. Astfel, dacă la momentul t pentru fiecare conexiune conectată în sistemul pachet cu pachet, potențialul de pornire al primului pachet aparține următorului cadru, atunci, dacă τ_k a fost ultimul moment de actualizare $S_i(t) \geq (k+1)T,\ \forall i \in \hat{B}(t)$. Astfel, potențialul fiecărei conexiuni în serverul cu fluid la momentul t este şi el mai mare sau egal cu $(k+1)T$. Deci putem determina un moment de actualizare doar din informațiile serverului pachet cu pachet. Planificatorul urmăreşte conexiunile conectate şi constată că sunt pachete cu potențial de pornire în următorul cadru. Când potențialul de pornire al pachetelor din fruntea cozilor

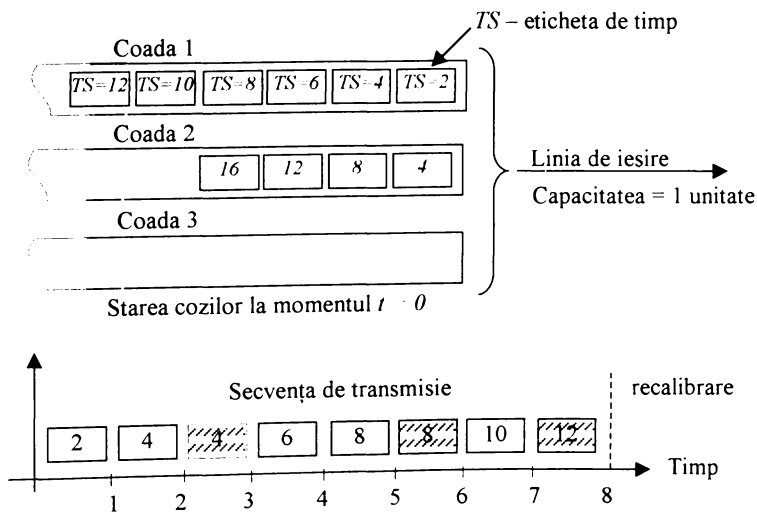


Fig. 7.2 Exemplu de operare al serverului PIC

BUPT

tuturor conexiunilor conectate depăşeşte limita ştim că acelaşi lucru s-a întâmplat cu potenţialul conexiunilor din sistemul cu fluid. Deci, momentul depăşirii limitei ultimei conexiuni este un moment valid de actualizare a cadrului şi a potenţialului sistemului. Acest lucru rezultă mai clar dintr-un exemplu (figura 7.2).

Dimensiunea F a cadrului este de 10 celule şi rata serverului este 1. Conexiunea 1 are rezervată 50% din banda de ieşire iar celelalte două conexiuni câte 25%. Înainte de $t=0$ considerăm că sistemul a fost inactiv. La $t=0$, C_1 şi C_2 transmit multe pachete planificatorului iar C_3 este inactivă. La $t = 0$, $P(t) = P_i(t) = 0$. În serverul cu fluid C_1 şi C_2 vor fi servite proporţional cu rezervările lor. Astfel rata serviciului pentru C_1 şi C_2 va fi 0,5 / (0,5+0,25) = 2/3 şi 0,5/0,75 = 1/3. Conexiunea 3 rămâne inactivă, iar potenţialele conexiunilor 1 şi 2 vor atinge valoarea $T = F/1 = 10$ în acelaşi timp, la momentul τ. Ambele conexiuni fiind conectate potenţialele lor cresc cu serviciul normalizat oferit lor, deci

$$\frac{W_1(0,\tau)}{0,5} = \frac{W_2(0,\tau)}{0,25} = 10,$$

şi din ecuaţie rezultă τ

$$(2/3)\,(\tau - 0) / 0,6 = 10$$

adică $\tau = 7,57$. Acesta este începutul ferestrei în care poate apărea prima actualizare de cadru.

Considerând acum serverul pachet cu pachet, pachetele sunt etichetate cu potenţialul conexiunii la momentul în care ele îşi termină serviciul în serverul cu fluid. Toate pachetele au dimensiunea 1. În figura 7.2, la momentul 8 toate pachetele cu potenţialul de pornire în primul cadru au fost transmise. La acest moment potenţialul conexiunilor conectate are o valoare mai mare sau egală cu 10 în serverul cu fluid. Astfel, la acest moment potenţialul sistemului poate fi actualizat la 10, fără violarea proprietăţilor funcţiei potenţial a sistemului.

Putem acum descrie versiunea pachet cu pachet a algoritmului PIC. Presupunem că rata serviciului serverului este 1, deci timpul de transmisie a F biţi este tot F. Din lăţimea de bandă a liniei de ieşire, conexiunii i i se alocă fracţia r_i deci $\phi_i = F \cdot r_i$ biţi care pot fi transmişi de la C_i într-un cadru. În versiunea cu fluid dimensiunea maximă a pachetului trebuie să fie mai mică decât ϕ_i, astfel că într-un cadru poate fi transmis un singur pachet.

La sosirea unui pachet se execută algoritmul din figura 7.3. pentru a calcula eticheta de timp asociată pachetului. Variabila P ţine evidenţa potenţialului sistemului, fiind reprezentată în virgulă mobilă, cu partea întreagă reprezentând numărul de cadru curent, iar partea fracţionară timpul care a trecut de la ultima actualizare. La sosirea unui pachet se estimează potenţialul sistemului. Cum variabila P este actualizată doar la sfârşitul transmisiei fiecărui pachet, potenţialul curent al sistemului este obţinut prin adăugarea la P a timpului real scurs de la începutul transmisiei pachetului aflat curent în serviciu. Potenţialul de pornire a pachetelor nou sosite este apoi calculat ca maximul

BUPT

dintre potențialul de sfârşit la pachetului precedent ale aceleaşi sesiuni şi potențialul sistemului. Pachetul este apoi etichetat cu potențialul de sfârşit, calculat din valorile cunoscute ale lungimii sale şi ratei rezervate. Dacă potențialele de început şi de sfârşit ale pachetului aparțin unor cadre diferite, pachetul va trece în cadrul următorilor. Astfel pachetul e marcat pentru a indica faptul că e primul pachet al sesiunii din cadrul următor. În plus, contorul este incrementat pentru a urmări numărul de conexiuni ce trec în cadrul următor. Se menține câte un contor pentru fiecare cadru, pentru a ține evidența numărului de sesiuni ale căror pachete ajung în următorul cadru. Apoi pachetul marcat este planificat pentru transmisie şi contorul corespunzător este edcrementat, astfel că atunci când devine zero potențialul tuturor conexiunilor conectate a ajuns în cadrul următor şi poate fi făcută actualizarea de cadru.

Calculul valorii curente a potențialului sistemului.

Fie t timpul curent şi t_S timpul la care pachetul aflat în transmisie îşi începe transmisia.

1. *temp* ← $P + (t - t_S) / F$

Se calculează potențialul de pornire al noului pachet

2. *SP (i, k)* ← *max (TS (i, k-1), timp)*

Se calculează eticheta de timp a pachetului

3. *TS(i, k)* ← *SP(i, k) + lungimea (i, k)*/ρ_i

Se verifică dacă pachetul depăşeşte cadrul

4. *n1*← *int (SP(i, k); n2*← *int (TS(i, k)*
5. *if (n1<n2) then (dacă potențialul de sfârşit este în cadrul următor)*
6. *B[n1]* ← *B[n1] + 1 (incrementare contor); marcare pachet*
7. *End if*

Fig. 7.3 Operațiile executate la sosirea pachetului

Contoarele B sunt folosite pentru numărarea conexiunilor cu potențial de pornire din fiecare cadru. Deşi teoretic trebuie servite un număr infinit de cadre, practic numărul de cadre distincte cu potențiale în cadru, este limitat de tamponul alocat conexiunii. Astfel dacă b_i este dimensiunea tamponului alocat conexiunii i, dimensiunea zonei B se poate limita la $M = \max_{i \in I} (b_i / \phi_i)$. Dacă M este rotunjit la cea mai apropiată putere a lui 2, atunci zona poate fi adresată cu $log_2 M$ cei mai puțin semnificativi biți ai numărului cadrului curent. Numărul de contoare mai poate fi redus la trei dacă paşii 4 – 8 ai algoritmului sunt executați doar când pachetul ajunge în capul cozii sesiunii corespunzătoare.

Când un pachet îşi termină transmisia se execută algoritmul din figura 7.4 pentru a actualiza starea sistemului. Se creşte mai întâi potențialul sistemului cu timpul de transmisie al pachetului care tocmai a fost transmis, j.

BUPT

1. $P \leftarrow P$ + *lungimea (j)/F*

Se determină eticheta de timp a următorului pachet de transmis

2. $TS_{min} \leftarrow min_{i \in B}$ *(TS (i))*

Se determină numărul corespunzător de cadru

3. $F_{min} \leftarrow$ *int* (TS_{min})

Dacă este necesar se actualizează cadrul

4. *if (pachetul j a fost marcat) then*
5. *B[cadru curent]* $\leftarrow$ *B[cadru curent] - 1*
6. *end if*
7. *if (B[cadru curent] = 0 şi* F_{min} *> cadru curent) then*
8. *cadru curent* $\leftarrow$ *cadru curent + 1*
9. $P \leftarrow$ *max (cadru curent, P)*
10. *end if*

Memorează în t_S timpul de transmisie al pachetului următor)

11. $t_S \leftarrow$ *timpul curent*
12. *selectează şi transmite pachetul din capul cozii*

Fig. 7.4 Operaţiile executate la transmiterea pachetului

timpul de transmisie al pachetului tocmai servit şi apoi se selectează pentru transmisie pachetul cu eticheta de timp minimă. Variabila cadru curent ţine evidenţa indexului cadrului curent tratat. Dacă a fost marcat cadrul transmis se decrementează contorul de cadru curent corespunzător. Dacă aceasta devine zero, sesiunea servită este ultima care a ajuns la limita cadrului curent. Dar mai trebuie verificată o condiţie, înainte de actualizarea cadrului, deoarece este posibil ca un pachet să sosească având potenţialul său de sfârşit în cadrul curent de transmisie a ultimului pachet marcat. Acest lucru poate rezulta dacă potenţialul sistemului este presupus la o valoare mai mare decât conexiunilor conectate, în contradicţie cu definiţia SPR. Se evită această situaţie asigurând că nici o etichetă de timp a pachetelor din coadă să nu cadă în cadrul curent chiar înainte de actualizarea cadrului. Dacă sunt satisfăcute ambele condiţii din pasul 7, se actualizează cadrul prin incrementare şi se actualizează potenţialul sistemul la potenţialul de bază corespunzător. Algoritmul se supune lemelor enunţate, este deci corect şi imparţial.

BUPT

CONCLUZII

În cele ce urmează, parcurgând lucrarea, vom căuta să subliniem care sunt contribuţiile sale originale.

În capitolul 1 se analizează pentru început definiţia standard a congestionării unei reţele, congestie manifestată prin creşterea valorilor unor parametrii ca întârzierea în cozi şi pierderea de pachete, respectiv diminuarea traficului util adică a eficienţei. S-au arătat că primii doi parametrii menţionaţi sunt indicii de performanţă a căror modificare poate semnala şi alte fenomene. **S-a propus astfel o nouă definiţie a congestiei reţelei, din perspectiva utilizatorului, care constată că o reţea este congestionată atunci când descreşte utilitatea percepută de el de la reţea, descreştere datorată creşterii încărcării reţelei.** Se mai propun şi **termenii de reţea strict congestionată** respectiv **strict necongestionată** pentru cazurile în care toţi utilizatorii percep congestia, respectiv nici un utilizator nu o percepe.

Se face apoi o analiză şi detaliere a perioadelor de apariţie şi manifestare a congestiei. **Sunt analizate diferite propuneri de scheme pentru evitarea şi controlul congestiei, reliefându-se calităţile respectiv părţile slabe ale acestora**. Astfel sunt prezentate părerile, multă vreme în opoziţie, asupra unor tipuri de control a congestiei şi anume: i) rezervarea în avans a resurselor faţă de alocarea acestora în momentul sosirii traficului în ruter, ii) controlul prin ferestre faţă de controlul bazat pe rată şi iii) control la sursă faţă de controlul în rutere. Se analizează şi argumentează situaţiile în care este utilă sau nu folosirea contrapresiunii. **Concluzia importantă care este expusă este că metoda de control a congestiei depinde de durata suprasarcinii.** Se susţine ideea că folosirea unei singure scheme de control şi evitare a congestiei nu este suficientă, şi că este necesară combinarea mai multor scheme pentru un control complet şi eficient al reţelei.

În capitolul 2 se prezintă cele mai noi metode cunoscute în literatură pentru controlul admisiei traficului în reţele. **Contribuţia acestui capitol constă în relevarea deficienţelor şi limitărilor fiecărui algoritm prezentat.**

Se arată că, deoarece una din principalele cauze ale apariţiei congestiei este traficul în rafală trebuie folosite mecanisme de nivelare sau formarea traficului. Sunt prezentate două din cele mai cunoscute metode de filtrare cunoscute sub numele de găleată găurită respectiv găleata cu jetoane, care pot fi folosite separat, sau împreună sub forma unei găleţi duble.

Se prezintă, se analizează şi se compară diferite scheme de control a admisiei traficului în reţele, aplicabile în principal la sursă dar şi în rutere, scheme necesare pentru a asigura îndeplinirea angajamentelor reţelei, adică asigurarea calităţii negociate a serviciului. Aceste scheme pot să fie bazate pe parametrii traficului de intrare sau pot să fie bazate pe măsurări.

La clasa algoritmilor bazaţi pe măsurări se relevă pentru fiecare criteriul adoptat pentru estimarea capacităţii disponibile: suma măsurată, regiunea de acceptanţă, respectiv capacitatea echivalentă. În cazul algoritmilor utilizând

BUPT

capacitatea echivalentă se relevă importanţa modelului adoptat pentru estimarea ratei sosirilor într-o clasă de trafic şi limitările fiecărui model. Astfel modelul distribuţiei normale poate fi folosit doar pentru anumite clase de trafic de timp real, cu sute de conexiuni conectate şi rate de vârf asemănătoare. Dacă însă ratele de vârf diferă mult şi numărul de conexiuni este redus sau mediu, estimarea capacităţii echivalente cu distribuţia normală poate duce la subestimări severe, caz în care este indicată folosirea marginea Hoeffding pentru estimare.

Folosirea conceptului de capacitate echivalentă respectiv lăţimea de bandă efectivă este atractivă datorită simplităţii calculului rezultat, calculul capacităţii necesare de serviciu fiind făcut prin aproximare cu o margine superioară, respectiv prin însumarea lăţimilor de bandă a fluxurilor individuale. Se lucrează însă în ipoteza tamponului infinit de memorie şi nu este luat în considerare efectul multiplexării statistice a surselor care partajează tamponul. Astfel când încărcarea scade apar supraestimări exagerate ale capacităţii de serviciu necesare. Când numărul de surse este mare, aproximarea gaussiană, propusă pentru a beneficia de avantajul multiplexării statistice, conduce de asemenea la supraestimări ale lăţimii de bandă agregată necesară, ceea ce duce la subutilizarea resurselor scumpe ale reţelei, lăţimea de bandă şi memoria. Dacă în plus, se folosesc filtre de tip găleată, traficul are un caracter de rafală mai scăzut decât traficul Poisson, şi atunci aproximarea cu lăţimea de bandă echivalentă nu mai este conservativă, conducând la o subestimare a capacităţii de serviciu necesară.

Se expune în consecinţă un alt criteriu pentru admisia unui apel, criteriu bazat pe rata de pierdere a celulelor admisă de apel. Decizia de admisie sau respingere a apelului se face pe baza estimării ratei reale de pierderi, estimare făcută în funcţie de capacitatea legăturii sau a serviciului pretins. Dar cum rata de pierdere celulelor şi capacitatea serviciului sunt două mărimi dependente, este dificil de determinat capacitatea serviciului, chiar având specificate caracteristicile traficului şi dimensiunea tampoanelor de memorie. Este în consecinţă prezentat un algoritm de control a admisiei folosind rata de pierdere a celulelor, care pentru estimarea benzii agregate foloseşte logica fuzzy. Fiind un sistem adaptiv estimează eficient rata de pierdere a celulelor în sisteme de dimensiuni mari, folosind un număr redus de informaţii despre sistem, fie cunoscute apriori fie obţinute prin măsurări.

Se relevă faptul că **aproximările mai noi**, bazate pe estimările deviaţilor mari ale lungimii cozilor de aşteptare din comutatoare, conduc la o estimare mai precisă a marginii superioare a ratei de pierdere a celulelor, decât cel bazat pe conceptul de bandă efectivă, dar **necesită cunoştinţe complete despre procesul de sosire în comutator.**

În capitolul 3 se prezintă facilităţi oferite reacţiei pentru controlul congestiei în reţelele ATM, precum şi criteriile de selecţie a algoritmilor propuşi pentru realizarea acestui control. **Contribuţia acestui capitol constă în analiza critică a celor mai importanţi algoritmi propuşi în ultimii ani.** Astfel:

BUPT

- Rezervarea rapidă (France Telecom), conduce la întârzieri excesive în timpul operării normale, respectiv la pierderi excesive în timpul congestiei.
- Controlul ratei bazat pe întârziere (Fujitsu) prezintă avantajul că nu se cere vreo reacţie din partea reţelei şi poate fi folosit în orice tip de reţele. Dezavantajul este însă timpul relativ mare necesar pentru reglarea volumului de trafic.
- Notificarea explicită înapoi a congestiei şi distrugerea timpurie a pachetelor nu sunt imparţiale, deoarece nu toate sursele care primesc reacţia, respectiv conversaţiile ale căror pachete sunt distruse, sunt responsabile de apariţia congestiei.
- Controlul cu câte o fereastră pe legătură şi control binar cap la cap al ratei reuneşte atât calităţile schemelor bazate pe rată cât şi ale celor bazate pe credite, respingerea sa fiind determinată de cauze subiective.
- Abordarea cu credite (DEC, Mitsubishi), are ca dezavantaje posibilitatea pierderii creditelor şi menţinerea câte unui cozi pentru fiecare circuit virtual, chiar inactiv, ceea ce face dificilă folosirea sa în cazul existenţei a sute de mii de conversaţii.
- Algoritmul MIT are ca dezavantaj timpul mare de calcul a ratei alocată imparţial.
- Algoritmul EPRCA este nonimparţial, deoarece indicatorul de congestie este lungimea cozilor depăşind un anume prag; astfel ruterele care-şi încep cu întârziere transmisia vor fi dezavantajate.

În anexă sunt prezentate rezultatele simulărilor realizate cu simulatorul NIST, pentru unii dintre aceşti algoritmi. Cu acest simulator pentru reţelele ATM, deşi s-a modificat încărcarea cu trafic şi structura acestuia, comportarea algoritmilor de planificare EPD, (Early Packet Discard) şi FQ este sensibil egală.

În capitolul 4 **se identifică problemele legate de planificarea traficului şi se propune tratarea separată a planificatoarelor pentru comutatoarele cu memorarea traficului la intrare, respectiv pentru comutatoarele cu memorare la ieşire.**

Se prezintă principalii algoritmi de planificare apăruţi în ultimii ani şi criteriile de clasificare existente, funcţie de arhitectura internă sau modul de lucru, care nu ţin însă cont de cerinţele reale ale aplicaţiilor. **Contribuţia acestui capitol constă din identificarea unor caracteristici ale planificatoarelor: scalabilitatea, întârzierea cap la cap, izolarea fluxurilor, folosirea eficientă a lăţimii de bandă, imparţialitatea şi complexitatea implementării.** Ţinând cont de aceste caracteristici **se propun în acest capitol trei măsuri de apreciere a algoritmilor de planificare: garanţiile asupra întârzierii cap la cap, imparţialitatea şi simplitatea implementării.**

În anexă sunt prezentate rezultatele simulărilor realizate cu simulatorul REAL, pentru unii din algoritmii de planificare. Astfel se constată că la încărcare mică şi trafic uniform, comportarea celor doi algoritmi de planificare FCFS şi FQ este practic aceeaşi dar pe măsură ce

BUPT

creşte atât încărcarea cu trafic cât şi ponderea traficului în rafală, comportarea algoritmului FQ este mult mai bună.

Contribuţia capitolului 5 **constă din propunerea şi analiza unui algoritm nou şi eficient de planificare, denumit algoritm de asociere probabilistică cu filtrarea cererilor, pentru alocarea benzii în comutatoare cu memorarea pachetelor la intrare. Algoritmul de asociere probabilistică cu filtrarea cererilor alocă lăţimea de bandă a ieşirilor, intrărilor, conform cu rezervările acestora. Algoritmul poate oferi garanţii probabilistice asupra faptului că traficul fiecărei intrări îşi va obţine partea de bandă a legăturii de ieşire. Algoritmul este capabil să ofere garanţii probabilistice asupra lăţimii de bandă şi întârzierii chiar şi în prezenţa traficului cu comportare necorespunzătoare.** În tabelul 5.4 **e reprezentată alocarea lăţimii de bandă a legăturii de ieşire, pentru patru intrări cu rezervări diferite atât pentru algoritmul PIM, analiză care nu apare în [AOST], cât şi pentru algoritmul APFC. Atâta timp cât sarcina de intrare totală e mai mică decât 1, fiecare intrare primeşte o parte egală din lăţimea de bandă, indiferent de algoritmul de planificare folosit, PIM sau APFC. Când sarcina la fiecare intrare depăşeşte 25 %, prin algoritmul PIM i se alocă fiecărei din cele patru intrări active câte o parte egală, adică 25% din lăţimea de bandă a legăturii de ieşire. Prin algoritmul APFC, intrările care primeau mai mult decât rezervarea, încep să primească mai puţin, astfel ca să se poată asigura rezervările pentru celelalte intrări. Principalul avantaj al algoritmului APFC este asigurarea garantării lăţimii de bandă conform rezervărilor, chiar în prezenţa surselor cu comportare necorespunzatoare.**

În capitolul 6 **s-au considerat comutatoare cu memorare la ieşire, pentru care s-a dezvoltat un model general, denumit al serverelor cu latenţa ratei *LR*. Acest model şi analiza care îi urmează constituie contribuţia capitolului 6. Astfel, modelul serverelor *LR* a fost folosit pentru a caracteriza performanţa**, în cazul cel mai defavorabil, a traficului **unei sesiuni într-o reţea oarecare de comutatoare care folosesc algoritmi de planificare diferiţi.** Folosind modelul propus **s-a dedus o margine deterministă pentru întârzierea cap la cap, imparţialitate, necesarul de tampoane de memorie şi gradul de rafală din interiorul reţelei.**

Dintre serverele *LR* s-a văzut că servirea imparţială ponderată este algoritmul ideal de planificare, din punct de vedere al întârzierii şi imparţialităţii. Totuşi calculul etichetei de timp la planificatorul cu servire imparţială ponderată servind *N* sesiuni are o complexitate de timp de *0(N)* per timpul de transmisie al unui pachet, făcând dificilă implementarea sa. Eforturile făcute în trecut pentru simplificarea implementării servirii imparţiale ponderate au condus la algoritmul de servire imparţială cu autosincronizare, ale cărei proprietăţi de izolare între conversaţii sunt însă degradate, lucru care afectează şi marginea întârzierii.

Astfel, în capitolul 7, **am dezvoltat o nouă metodologie de proiectare a algoritmilor de planificare, care să îmbine marginea întârzierii de la servirea imparţială ponderată, cu o nonimparţialitate limitată şi o**

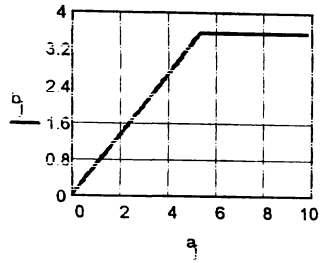
BUPT

implementare simplă. A rezultat **o nouă clasă de algoritmi de planificare, denumite servere proporţionale cu rata, care au parametrii identici cu cei ai planificării imparţiale ponderate, şi anume aceeaşi margine a întârzierii cap la cap, necesarul de tampoane de memorie şi gradul intern de rafală.**

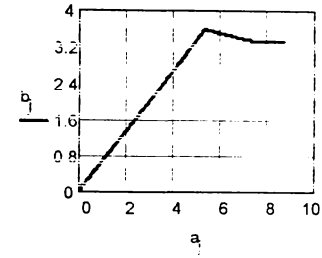
Folosind clasa serverelor proporţionale cu rata **am proiectat un algoritm de planificare imparţială bazat pe cadre, cu aceeaşi margine a întârzierii ca şi planificarea imparţială ponderată, şi o imparţialitate comparabilă cu acesta. Noul algoritm necesită un timp doar de *0(1)* pentru calculul etichetei de timp.** În funcţie de implementare, diferiţi algoritmi din clasa serverelor proporţionale cu rata, pot avea proprietăţi de imparţialitate foarte diferite. Astfel, **metodologia nou propusă poate fi folosită de către proiectanţii de reţele, pentru implementarea eficientă a algoritmilor de servire imparţială, compensând imparţialitatea lor cu complexitatea implementării.**

BUPT

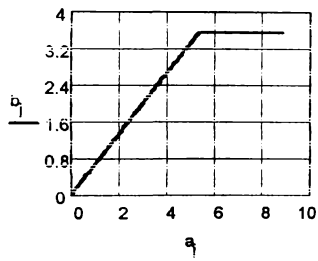
End to end delay 1-14 with 3 Generic sources and 5 Poisson sources-FCFS



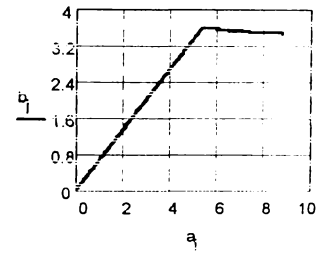
End to end delay 1-14 with 3 Generic sources and 5 Poisson sources-FQ



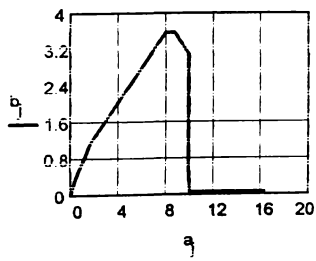
End to end delay 2-16 with 3 Generic sources and 5 Poisson sources-FCFS



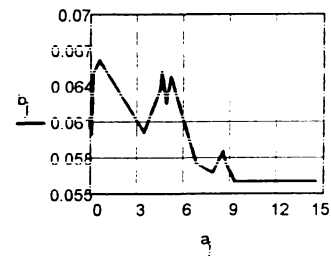
End to end delay 2-16 with 3 Generic sources and 5 Poisson sources-FQ



End to end delay 4-16 with 3 Generic sources and 5 Poisson sources-FCFS

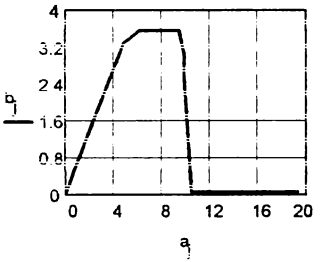


End to end delay 4-16 with 3 Generic sources and 5 Poisson sources-FQ

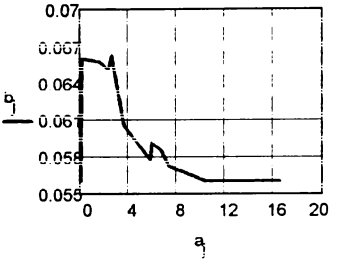


BUPT

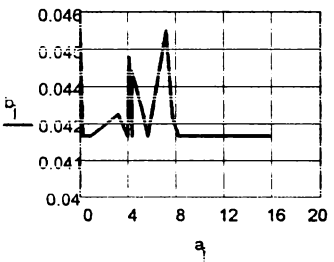
End to end delay 5-15 with 3 Generic sources and 5 Poisson sources-FCFS



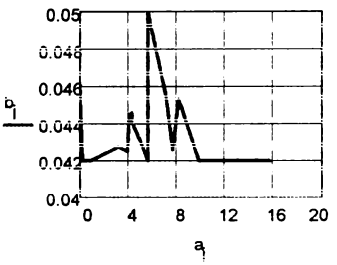
End to end delay 5-15 with 3 Generic sources and 5 Poisson sources-FQ



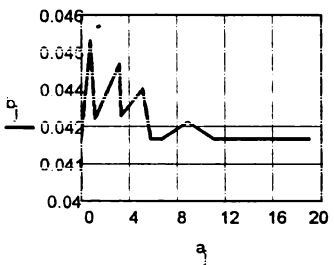
End to end delay 6-15 with 3 Generic sources and 5 Poisson sources-FCFS



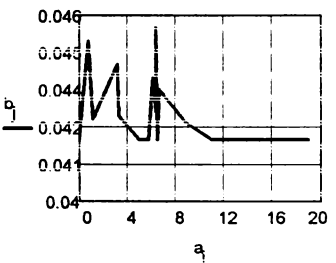
End to end delay 6-15 with 3 Generic sources and 5 Poisson sources-FQ



End to end delay 7-15 with 3 Generic sources and 5 Poisson sources-FCFS

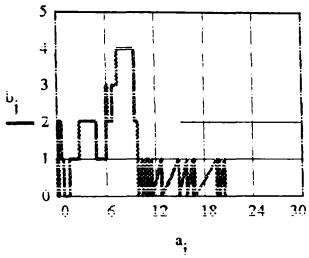


End to end delay 7-15 with 3 Generic sources and 5 Poisson sources-FQ

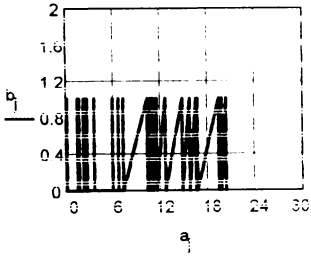


BUPT

Queue length of source 5 in router 10 with 3 Generic sources and 5 Poisson sources - FCFS

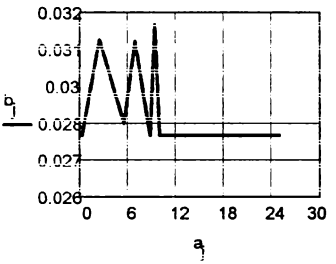


Queue length of source 5 in router 10 with 3 Generic sources and 5 Poisson sources - FQ

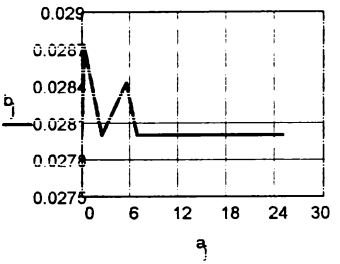


BUPT

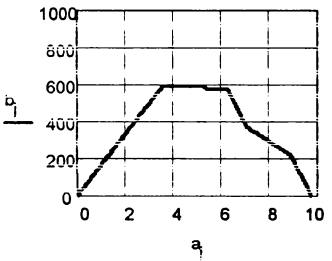
End to end delay 8-14 with 3 Generic sources and 5 Poisson sources-FCFS



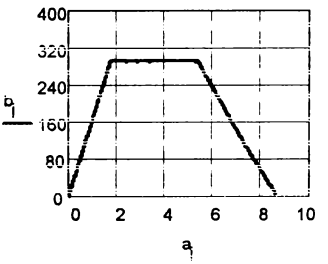
End to end delay 8-14 with 3 Generic sources and 5 Poisson sources-FQ



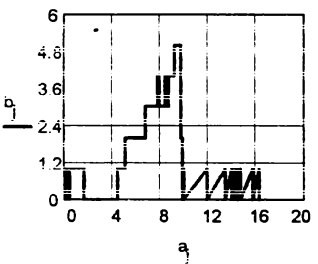
Queue length of source 1 in router 10 with 3 Generic sources and 5 Poisson sources - FCFS



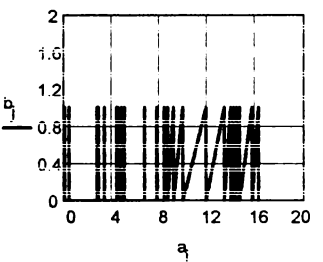
Queue length of source 1 in router 10 with 3 Generic sources and 5 Poisson sources - FQ



Queue length of source 4 in router 10 with 3 Generic sources and 5 Poisson sources - FCFS

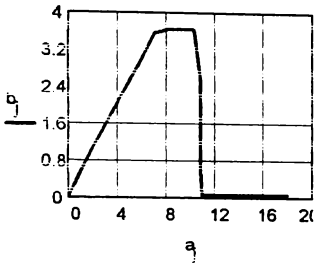


Queue length of source 4 in router 10 with 3 Generic sources and 5 Poisson sources - FQ

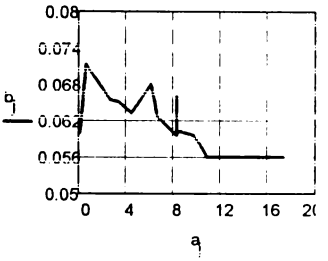


BUPT

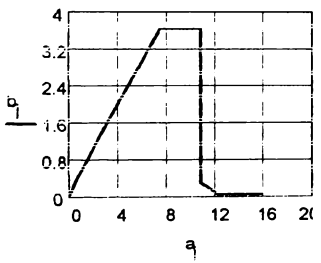
End to end delay 4-16 with 3 Generic sources and 4 Poisson sources-FCFS



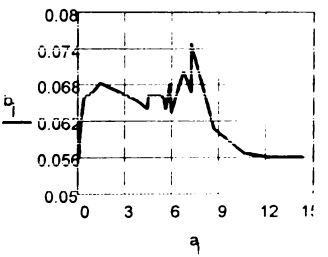
End to end delay 4-16 with 3 Generic sources and 4 Poisson sources-FQ



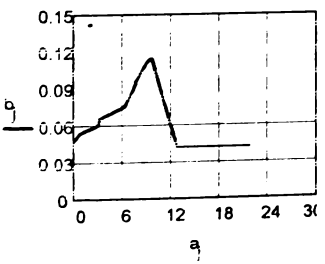
End to end delay 5-15 with 3 Generic sources and 4 Poisson sources-FCFS



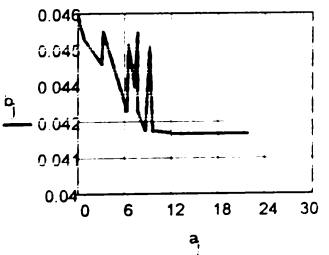
End to end delay 5-15 with 3 Generic sources and 4 Poisson sources-FQ



End to end delay 6-15 with 3 Generic sources and 4 Poisson sources-FCFS

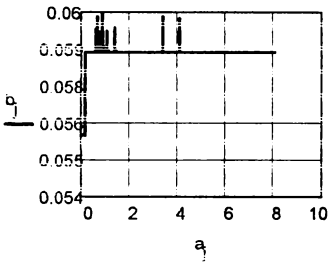


End to end delay 6-15 with 3 Generic sources and 4 Poisson sources-FQ

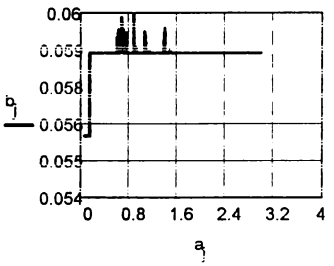


BUPT

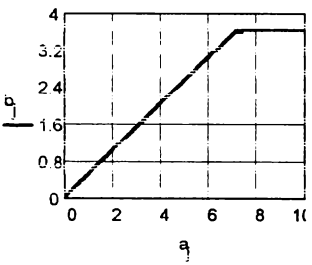
End to end delay 1-14 with 3 Generic sources and 4 Poisson sources-FCFS



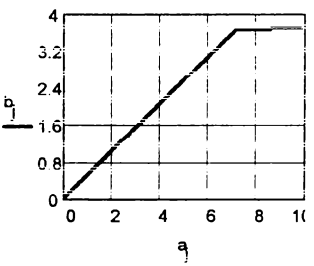
End to end delay 1-14 with 3 Generic sources and 4 Poisson sources-FQ



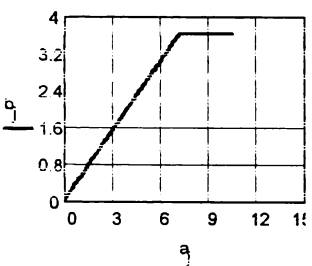
End to end delay 2-16 with 3 Generic sources and 4 Poisson sources-FCFS



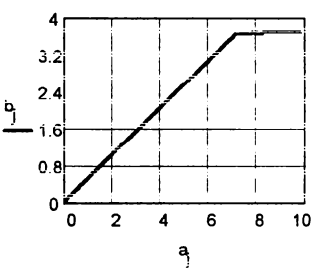
End to end delay 2-16 with 3 Generic sources and 4 Poisson sources-FQ



End to end delay 3-16 with 3 Generic sources and 4 Poisson sources-FCFS

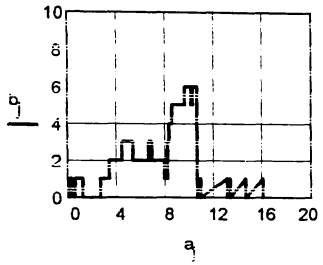


End to end delay 3-16 with 3 Generic sources and 4 Poisson sources-FQ

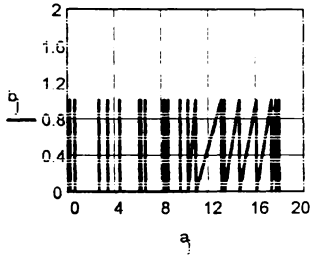


BUPT

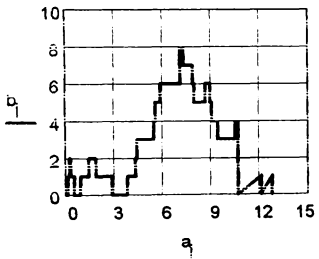
Queue length of source 4 in router 10 with 3 Generic sources and 4 Poisson sources - FCFS



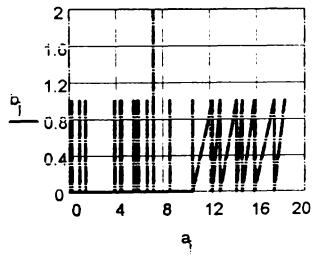
Queue length of source 4 in router 10 with 3 Generic sources and 4 Poisson sources - FQ



Queue length of source 5 in router 10 with 3 Generic sources and 4 Poisson sources - FCFS

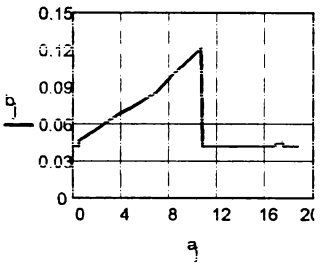


Queue length of source 5 in router 10 with 3 Generic sources and 4 Poisson sources - FQ

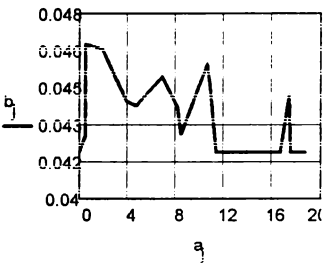


BUPT

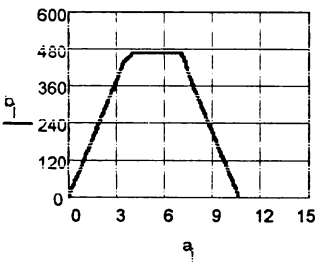
End to end delay 7-15 with 3 Generic sources and 4 Poisson sources-FCFS



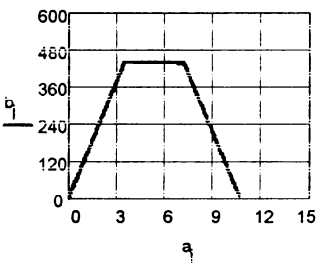
End to end delay 7-15 with 3 Generic sources and 4 Poisson sources-FCFS



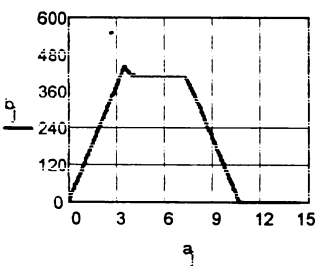
Queue length of source 2 in router 10 with 3 Generic sources and 4 Poisson sources - FCFS



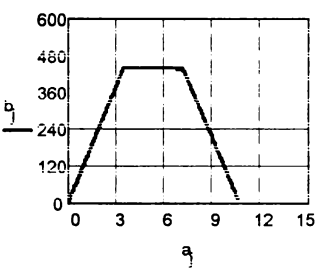
Queue length of source 2 in router 10 with 3 Generic sources and 4 Poisson sources - FQ



Queue length of source 3 in router 10 with 3 Generic sources and 4 Poisson sources - FCFS

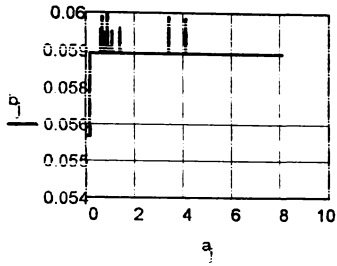


Queue length of source 3 in router 10 with 3 Generic sources and 4 Poisson sources - FQ

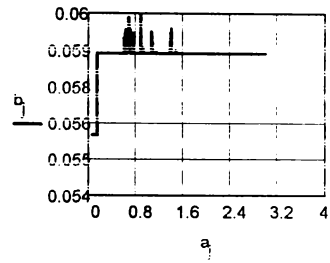


BUPT

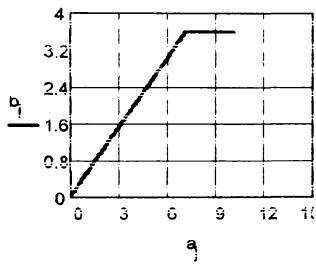
End to end delay 1-14 with 3 Generic sources and 3 Poisson sources-FCFS



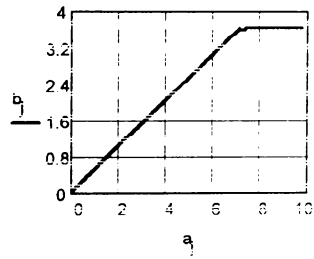
End to end delay 1-14 with 3 Generic sources and 3 Poisson sources-FQ



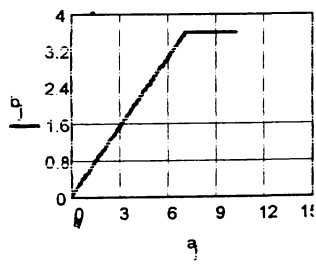
End to end delay 2-16 with 3 Generic sources and 3 Poisson sources-FCFS



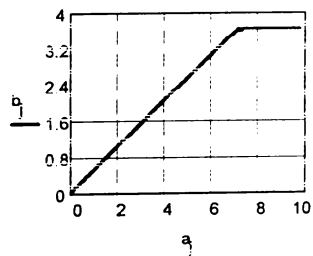
End to end delay 2-16 with 3 Generic sources and 3 Poisson sources-FQ



End to end delay 3-16 with 3 Generic sources and 3 Poisson sources-FCFS

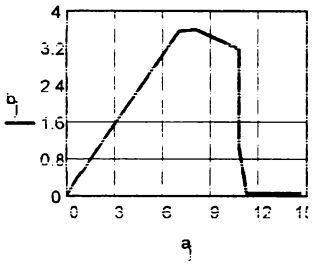


End to end delay 3-16 with 3 Generic sources and 3 Poisson sources-FQ

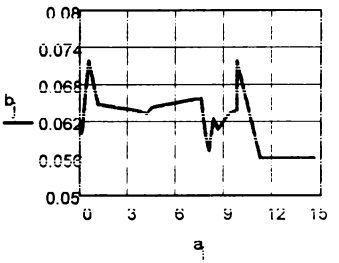


BUPT

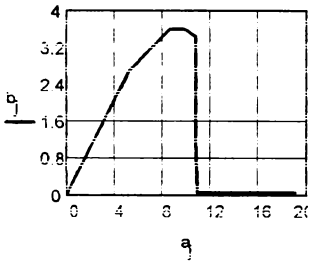
End to end delay 4-16 with 3 Generic sources and 3 Poisson sources-FCFS



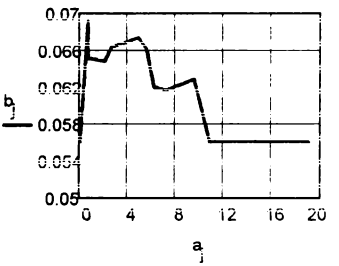
End to end delay 4-16 with 3 Generic sources and 3 Poisson sources-FQ



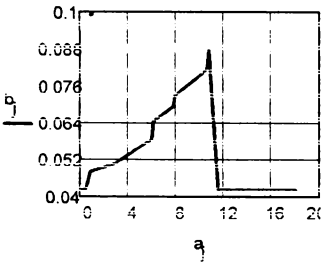
End to end delay 5-15 with 3 Generic sources and 3 Poisson sources-FCFS



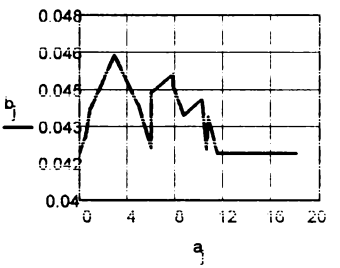
End to end delay 5-15 with 3 Generic sources and 3 Poisson sources-FQ



End to end delay 6-15 with 3 Generic sources and 3 Poisson sources-FCFS



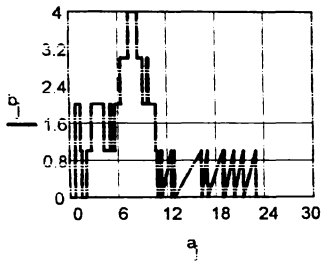
End to end delay 6-15 with 3 Generic sources and 3 Poisson sources-FQ



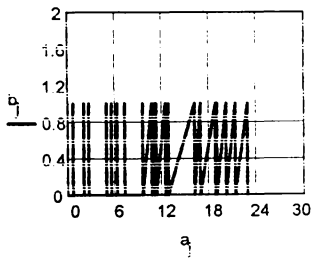
BUPT

BUPT

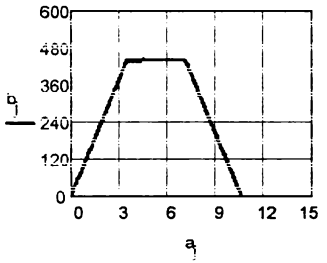
Queue length of source 5 in router 10 with 3 Generic sources and 3 Poisson sources - FCFS



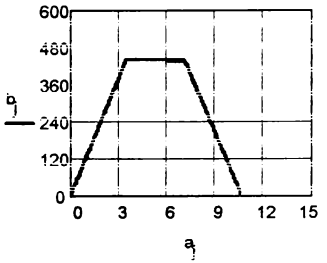
Queue length of source 5 in router 10 with 3 Generic sources and 3 Poisson sources - FQ



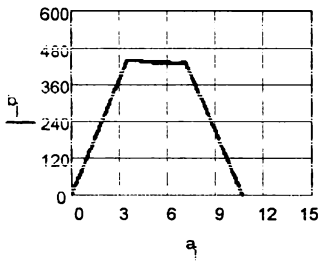
Queue length of source 2 in router 10 with 3 Generic sources and 3 Poisson sources - FCFS



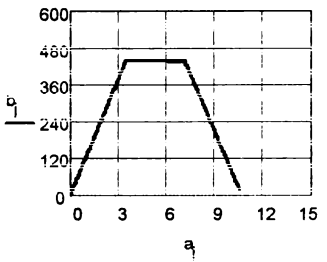
Queue length of source 2 in router 10 with 3 Generic sources and 3 Poisson sources - FQ



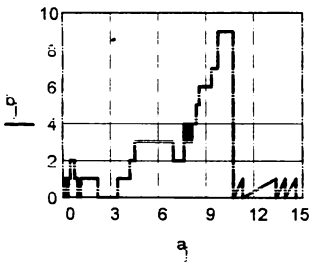
Queue length of source 3 in router 10 with 3 Generic sources and 3 Poisson sources - FCFS



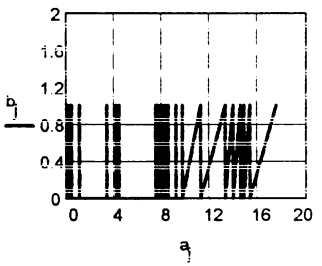
Queue length of source 3 in router 10 with 3 Generic sources and 3 Poisson sources - FQ



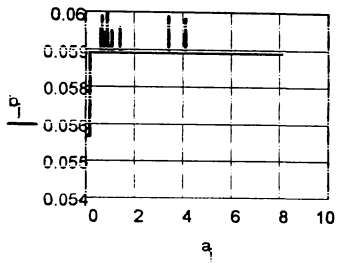
Queue length of source 4 in router 10 with 3 Generic sources and 3 Poisson sources - FCFS



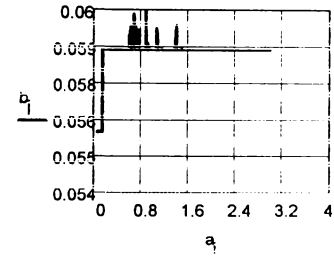
Queue length of source 4 in router 10 with 3 Generic sources and 3 Poisson sources - FQ



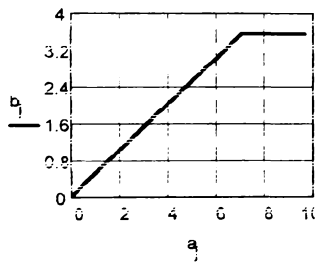
End to end delay 1-14 with 3 Generic sources and 2 Poisson sources-FCFS



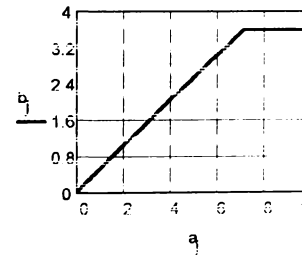
End to end delay 1-14 with 3 Generic sources and 2 Poisson sources-FQ



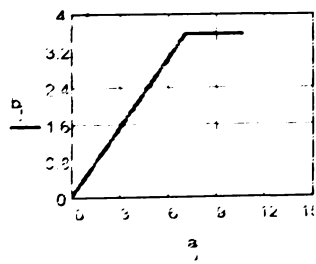
End to end delay 2-16 with 3 Generic sources and 2 Poisson sources-FCFS



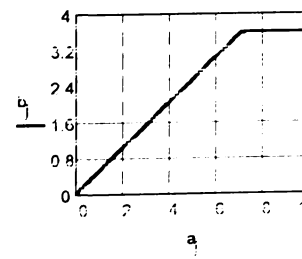
End to end delay 2-16 with 3 Generic sources and 2 Poisson sources-FQ



End to end delay 3-16 with 3 Generic sources and 2 Poisson sources-FCFS

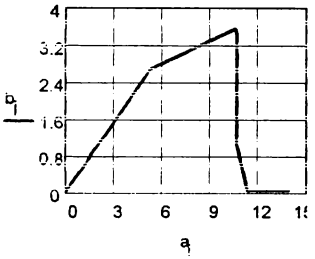


End to end delay 3-16 with 3 Generic sources and 2 Poisson sources-FQ

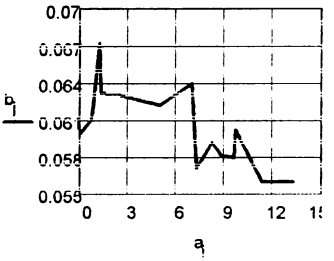


BUPT

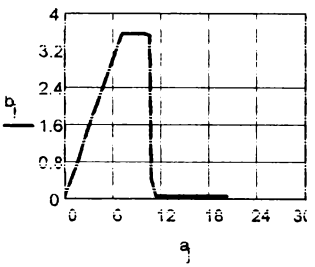
End to end delay 4-16 with 3 Generic sources and 2 Poisson sources-FCFS



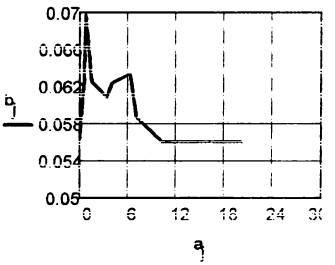
End to end delay 4-16 with 3 Generic sources and 2 Poisson sources-FQ



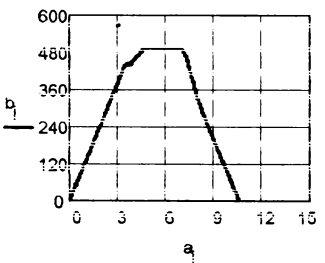
End to end delay 5-15 with 3 Generic sources and 2 Poisson sources-FCFS



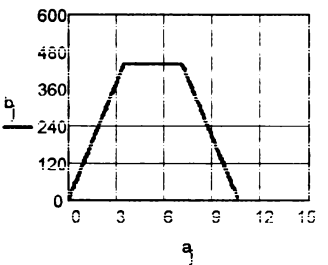
End to end delay 5-15 with 3 Generic sources and 2 Poisson sources-FQ



Queue length of source 2 in router 10 with 3 Generic sources and 2 Poisson sources - FCFS

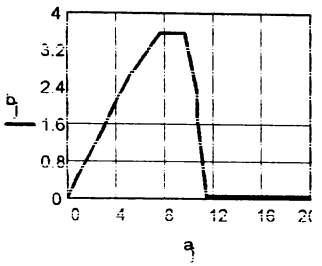


Queue length of source 2 in router 10 with 3 Generic sources and 2 Poisson sources - FQ

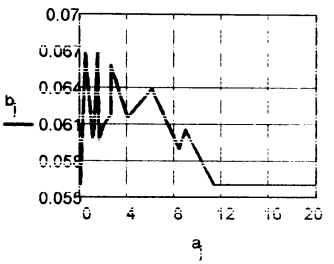


BUPT

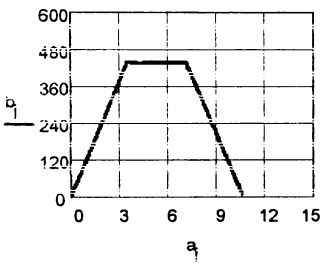
End to end delay 4-16 with 3 Generic sources and 1 Poisson source-FCFS



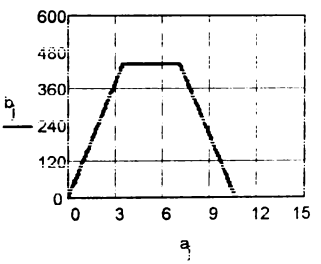
End to end delay 4-16 with 3 Generic sources and 1 Poisson source-FQ



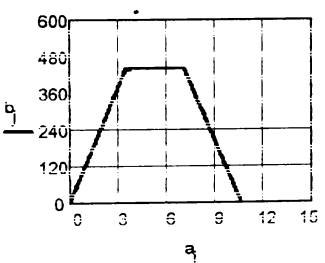
Queue length of source 2 in router 10 with 3 Generic sources and 1 Poisson source - FCFS



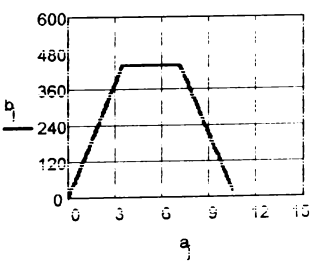
Queue length of source 2 in router 10 with 3 Generic sources and 1 Poisson source - FQ



Queue length of source 3 in router 10 with 3 Generic sources and 1 Poisson source - FCFS

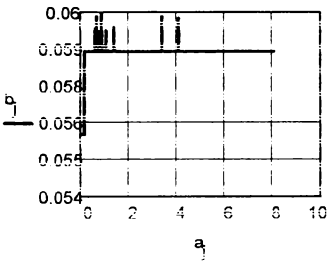


Queue length of source 3 in router 10 with 3 Generic sources and 1 Poisson source FQ

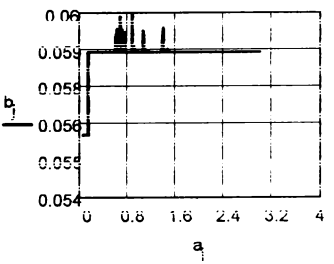


BUPT

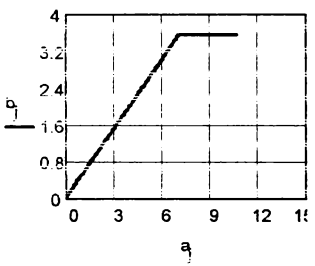
End to end delay 1-14 with 3 Generic sources and 1 Poisson source-FCFS



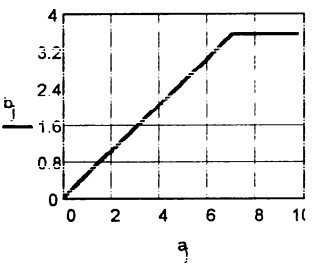
End to end delay 1-14 with 3 Generic sources and 1 Poisson source-FQ



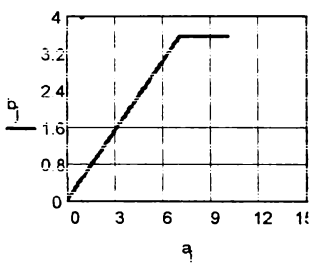
End to end delay 2-16 with 3 Generic sources and 1 Poisson source-FCFS



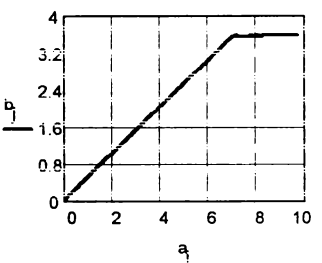
End to end delay 2-16 with 3 Generic sources and 1 Poisson source-FQ



End to end delay 3-16 with 3 Generic sources and 1 Poisson source-FCFS

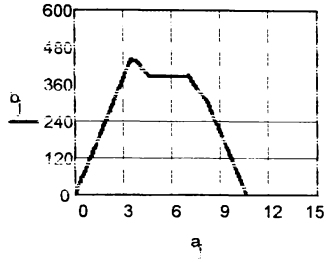


End to end delay 3-16 with 3 Generic sources and 1 Poisson source-FQ

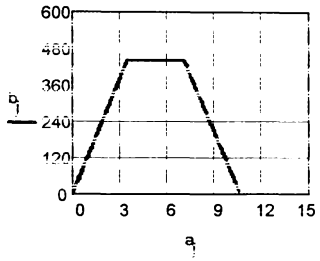


BUPT

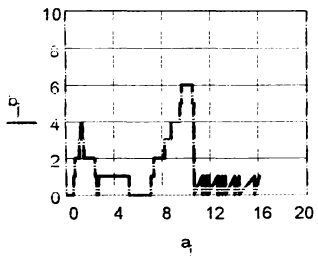
Queue length of source 3 in router 10 with 3 Generic sources and 2 Poisson sources - FCFS



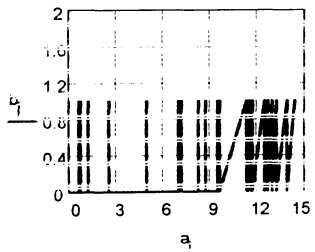
Queue length of source 3 in router 10 with 3 Generic sources and 2 Poisson sources - FQ



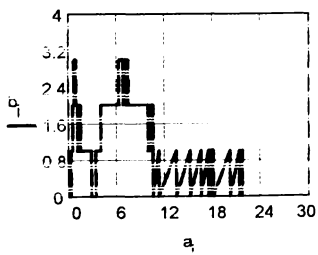
Queue length of source 4 in router 10 with 3 Generic sources and 2 Poisson sources - FCFS



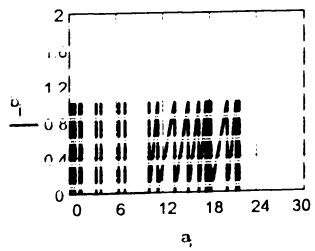
Queue length of source 4 in router 10 with 3 Generic sources and 2 Poisson sources - FQ



Queue length of source 5 in router 10 with 3 Generic sources and 2 Poisson sources - FCFS

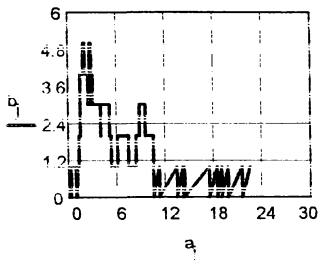


Queue length of source 5 in router 10 with 3 Generic sources and 2 Poisson sources - FQ

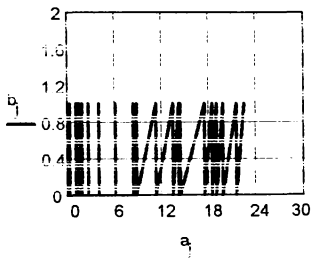


BUPT

Queue length of source 4 in router 10 with 3 Generic sources and 1 Poisson source - FCFS



Queue length of source 4 in router 10 with 3 Generic sources and 1 Poisson source – FQ



BUPT