UDŽBENIK ELEKTROTEHNIČKOG FAKULTETA U BEOGRADU

Milan Bjelica

TELEKOMUNIKACIONE I RAČUNARSKE MREŽE

zbirka rešenih zadataka

drugo izdanje

Mensa Srbije
Novi Sad, 2021.

dr Milan Bjelica,
Elektrotehnički fakultet Univerziteta u Beogradu
e–mail: milan@etf.rs

TELEKOMUNIKACIONE I RAČUNARSKE MREŽE
zbirka rešenih zadataka
drugo izdanje

Recenzentkinje:
dr Aleksandra Smiljanić
dr Nataša Maksić

Nastavno-naučno veće Elektrotehničkog fakulteta odobrilo je objavljivanje ovoga pomoćnog udžbenika odlukom broj 1354/2 od 22.10.2021. godine.

Izdavač:
Mensa Srbije
Bulevar oslobođenja 22
Novi Sad

ISBN: 978-86-80994-12-3

Tekst ove knjige složen je u programskom paketu $\LaTeX\,2_{\varepsilon}$.

CIP - Каталогизација у публикацији
Библиотека Матице српске, Нови Сад

621.39:004.722(075.8)(076)
004.7(075.8)(076)

БЈЕЛИЦА, Милан, 1977-
Telekomunikacione i računarske mreže [Elektronski izvor] : zbirka rešenih zadataka / Milan Bjelica. - 2. izd. - Novi Sad : Mensa Srbije, 2021

Način pristupa (URL): https://www.mensa.rs/publikacije/zbirka_bjelica.pdf. - Opis zasnovan na stanju na dan 26.11.2021. - Nasl. s naslovnog ekrana. - Bibliografija.

ISBN 978-86-80994-12-3

a) Телекомуникационе мреже - Задаци б) Рачунарске мреже - Задаци

COBISS.SR-ID 52111881

Sadržaj

1. Uvod

Zadatak 1.1 Definišite pojmove: telekomunikacije, telekomunikaciona mreža, telekomunikacioni servis.

Telekomunikacije su svako emitovanje, prenos ili prijem poruka (govora, zvuka, teksta, slike ili podataka) u vidu signala, korišćenjem žičnih, radio, optičkih ili bilo kojih drugih elektromagnetnih sistema.

Telekomunikaciona mreža je skup sistema prenosa i, tamo gde je to primenjeno, uređaja za komutaciju i rutiranje i drugih resursa, uključujući pasivne mrežne elemente, koji omogućavaju prenos signala pomoću žičnih, radio, optičkih ili drugih elektromagnetnih sredstava, bez obzira na vrstu podataka i informacija koje se prenose.

Telekomunikacioni servis je usluga koja se po pravilu pruža uz naknadu, a sastoji se u celini ili pretežno od prenosa signala u telekomunikacionim mrežama.

Zadatak 1.2 Izračunajte kapacitet telefonskog kanala, u kome je odnos signal-šum 30 dB.

Kapacitet kanala odredićemo po Shannonovom obrascu:

$$C = B \log_2 \left(1 + \frac{S}{N}\right).$$

Pri tome je sa B označena širina propusnog opsega, a sa S/N odnos signal-šum.

Granice propusnog opsega telefonskog kanala su 300 Hz i 3400 Hz. Širina propusnog opsega stoga je $B = 3100$ Hz.

Odnos signal-šum u kanalu je

$$\frac{S}{N} = 10^3 = 1000,$$

pa je kapacitet ovoga kanala

$$C = 30898{,}4 \text{ Sh/s}.$$

Zadatak 1.3 Rezolucija digitalnog video-signala iznosi 480 × 360 piksela. Analiza je progresivna, s 25 slika u sekundi. Svakom pikselu odgovara jedan od 32 nivoa osvetljaja. Je li moguć pouzdani prenos ovoga signala kanalom čija širina propusnog opsega iznosi 4 MHz i u kome je odnos signal-šum 35 dB?

Za predstavljanje 32 nivoa osvetljaja, potrebno je $\log_2 32 = 5$ bita. Binarni protok posmatranog video signala stoga je

$$V_b = 480 \cdot 360 \cdot 25 \text{ s}^{-1} \cdot 5 \text{ b} = 21{,}6 \, \frac{\text{Mb}}{\text{s}}.$$

Kapacitet kanala odredićemo po Shannonovom obrascu:

$$C = B \log_2 \left(1 + \frac{S}{N} \right).$$

Kako je

$$\frac{S}{N} = 10^{3{,}5} = 3162{,}28,$$

biće

$$C = 46{,}5 \cdot 10^6 \, \frac{\text{Sh}}{\text{s}}.$$

Pošto je kapacitet kanala veći od binarnog protoka signala, moguć je pouzdani prenos.

Zadatak 1.4 Odredite efektivni protok korisnih podataka na linku fizičkog protoka $R = 9600$ b/s, ako se koristi

a) asinhroni prenos s poljem korisnih podataka dužine 7 bita i ispitivanjem parnosti ili

b) sinhroni prenos, s okvirima koji se sastoje od 48 kontrolnih i 128 informacionih bita.

a) Na svakih sedam korisnih bita, prenesu se još tri dodatna (START, zaštitni bit i STOP).

Slika 1.4: *Sekvenca bita pri asinhronom prenosu.*

Efektivni protok korisnih podataka u ovom slučaju je

$$R' = \frac{7}{10} R = 6720 \, \frac{\text{b}}{\text{s}}.$$

b) Od ukupno 176 bita u okviru, korisno je njih 128. Efektivni protok korisnih podataka je

$$R' = \frac{128}{176}\,R = 6982\,\frac{\mathrm{b}}{\mathrm{s}}.$$

Zadatak 1.5 Dva uređaja razmenjuju 7-bitske podatke po asinhronoj sprezi. Ako je verovatnoća greške na liniji p, kolika je verovatnoća sledećih događaja:

a) došlo je do greške na START ili STOP bitu, usled čega su odbačeni svi primljeni biti,

b) START i STOP biti su ispravni, ali paritet korisnog sadržaja nije saglasan primljenom zaštitnom bitu,

c) START i STOP biti su ispravni, ali je došlo do greške koja se ne može otkriti ispitivanjem pariteta?

a) Ovaj događaj možemo shvatiti kao uniju triju uzajamno disjunktih događaja: (1) pogrešan je START bit, STOP bit je ispravan, (2) pogrešan je STOP bit, a START bit je ispravan i (3) pogrešni su i START i STOP biti. Njegova verovatnoća stoga je

$$P_a = 2p(1-p) + p^2 = 2p - p^2.$$

b) Biti START i STOP su ispravni s verovatnoćom $(1-p)^2$. Da bi došlo do greške koja se otkriva ispitivanjem pariteta, potrebno je da među osam preostalih bita (sedam korisnih i jednog zaštitnog) njih neparan broj budu pogrešni. Verovatnoća ovog združenog događaja je

$$P_b = (1-p)^2 \sum_{i=0}^{3} \binom{8}{2i+1} p^{2i+1}(1-p)^{7-2i}.$$

c) Jednim zaštitnim bitom pariteta ne mogu se otkriti greške na parnom broju bita. Verovatnoća ovog događaja je

$$P_c = (1-p)^2 \sum_{i=1}^{4} \binom{8}{2i} p^{2i}(1-p)^{8-2i}.$$

Zadatak 1.6 Dva telekomunikaciona uređaja povezana su asinhronom spregom. Sekvence bita koje se razmenjuju sastoje se od jednog start bita, 8 informacionih bita, jednog bita provere na parnost i dvaju stop bita. Odredite dopuštenu neusaglašenost frekvencija generatora takta na predaji i na prijemu, ukoliko se uzorci signala na prijemu uzimaju na sredini svakog signalizacionog intervala. Pretpostavite da je generator takta u prijemniku u fazi sa start bitom na njegovom početku.

Sekvence podataka se sastoje od 12 bita. Ako s T_T označimo trajanje signalizacionog intervala u predajniku, tada će trajanje svake sekvence podataka iznositi $12T_T$.

Označimo s T_R period generatora takta u prijemniku. Poslednji, dvanaesti bit sekvence tada se uzorkuje u trenutku $11{,}5\,T_R$. Da ne bi došlo do preskoka, potrebno je da se ovaj trenutak nađe unutar signalizacionog intervala dodeljenog dvanaestom bitu sekvence:

$$11\,T_T < 11{,}5\,T_R < 12\,T_T.$$

Odavde dobijamo

$$\frac{11}{f_T} < \frac{11{,}5}{f_R} < \frac{12}{f_T},$$

odnosno

$$11 f_R < 11{,}5 f_T < 12 f_R,$$

što konačno daje

$$0{,}958 f_T < f_R < 1{,}045 f_T,$$

gde su f_T i f_R frekvencije takta u predajniku i prijemniku, respektivno.

Zadatak 1.7 Dva telekomunikaciona uređaja povezana su sinhronom spregom. Nestabilnosti njihovih generatora takta su jednake i iznose 1 min/god. Uzorci signala na prijemu moraju se uzeti unutar intervala ±20% od sredine svakog signalizacionog intervala. Odredite količinu podataka koju će uređaji razmeniti pre pojave preskoka.

U najgorem slučaju, periodi generatora takta su neusaglašeni u suprotnim smerovima. Rezultantna nestabilnost tada iznosi

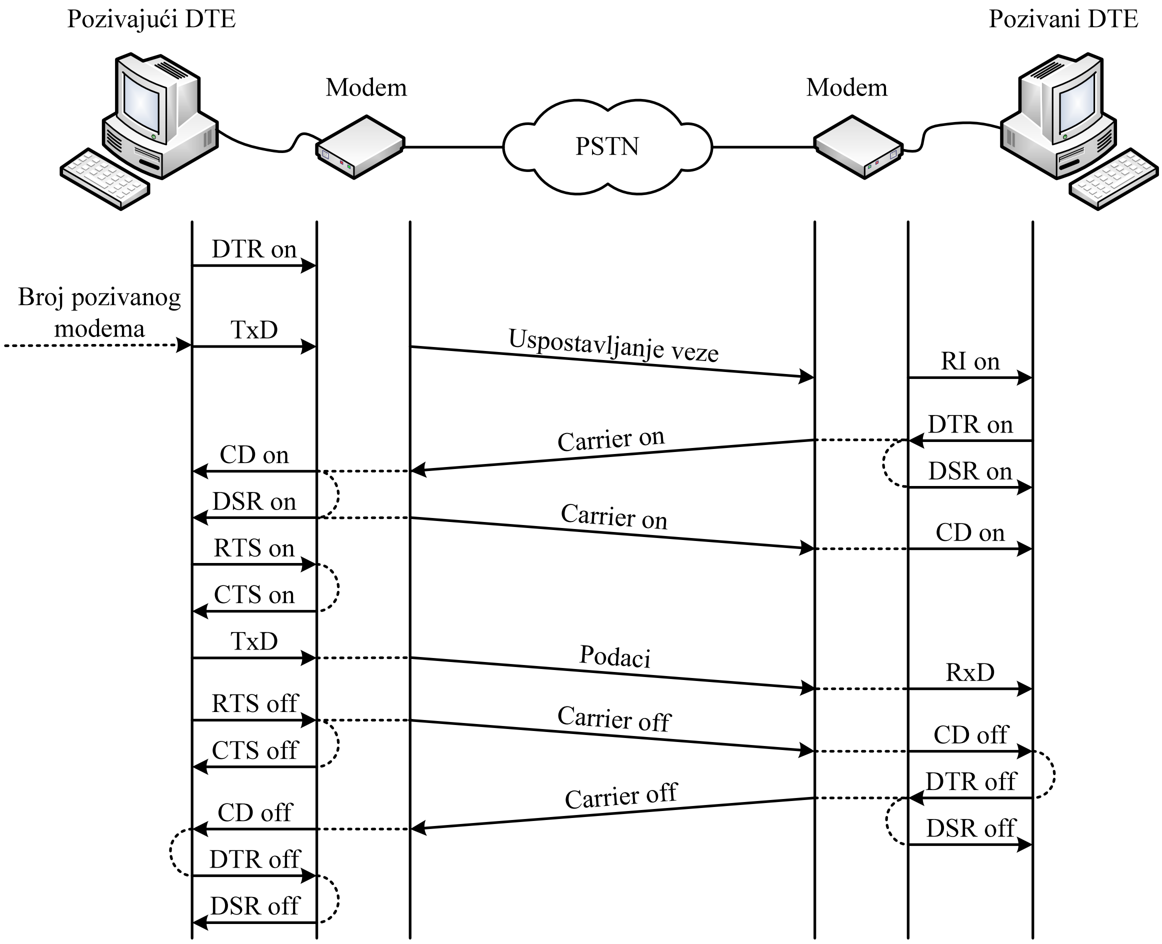
$$\left|\frac{\Delta T}{T}\right| = 2 \text{ min/god} = 0{,}00038\%.$$

Nakon N intervala takta (tj. N prenesenih bita), akumulirana neusaglašenost iznosi $N \cdot \Delta T$. Do preskoka dolazi onda kada ona dostigne vrednost $\delta \cdot T$, gde je $\delta = \pm 0{,}2$. To znači da će se do pojave preskoka preneti

$$N = |\delta| \left|\frac{T}{\Delta T}\right| = 52560$$

bita.

Zadatak 1.8 Dva računara komuniciraju preko javne telefonske mreže, uz korišćenje modema. Modemi su na računare povezani serijskom spregom RS 232. Nacrtajte vremenski dijagram razmene signala tokom ove sesije.



Slika 1.8: *Signali u sprezi RS 232.*

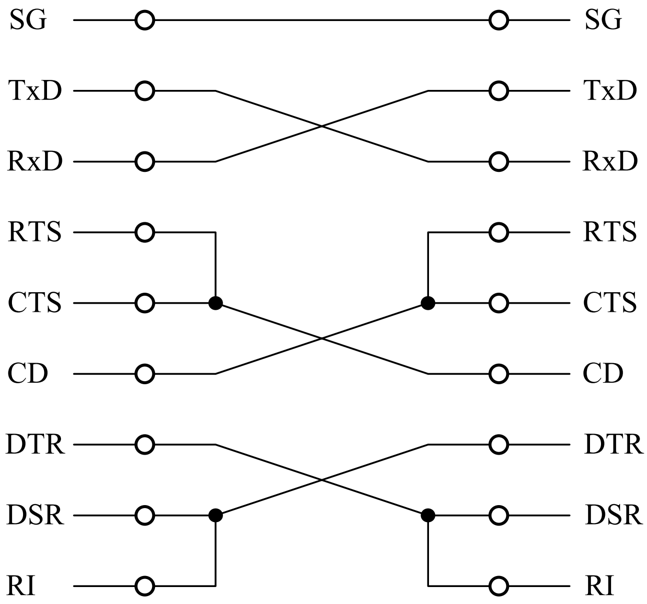
Traženi dijagram dat je na slici 1.8. Uobičajeno je da se za računar koji koristi spregu RS 232 upotrebljava oznaka DTE (*Data Terminal Equipment*), a za modem DCE (*Data Circuit-terminating Equipment*).

Signali koji se razmenjuju između DTE i DCE su:
TxD (*Transmitted Data*) — Podaci koje računar (DTE) šalje modemu (DCE),
RxD (*Received Data*) — Podaci koje računar (DTE) prima od modema (DCE),
DTR (*Data Terminal Ready*) — DTE je spreman za rad,
DSR (*Data Set Ready*) — DCE je spreman za rad,
RI (*Ring Indicator*) — DCE detektuje dolazni poziv,
RTS (*Request to Send*) — DTE obaveštava DCE da želi poslati podatke,
CTS (*Clear to Send*) — DCE označava spremnost za prijem podataka,
CD (*Carrier Detect*) — DCE prima signal propisanog nivoa.

Zadatak 1.9 Nacrtajte električnu shemu kabla kojim se omogućava direktno povezivanje dvaju računara serijskom spregom RS 232, bez korišćenja telefonskih modema.

U slučaju kada su računari (DTE) fizički blizu, moguće je izbeći korišćenje modema.

Kablom je tada potrebno povezati signalne mase računara, ukrstiti linije za slanje i prijem podataka, a kontrolne linije povezati tako da se emulira rad modema. Ovakav kabl se naziva *null modem* kablom i njegova shema data je na slici.



Slika 1.9: *Null modem kabl.*

Zadatak 1.10 Odredite poslatu AMI sekvencu, ako je primljena sekvenca $+-0+-0-+$. Pretpostavite da je samo jedan bit pogrešan.

Posmatranjem primljene sekvence, zaključujemo da se u prenosu desila greška, jer je narušen princip alternativne promene znaka (AMI). Tri mogća scenarija po kojima se to moglo desiti su:

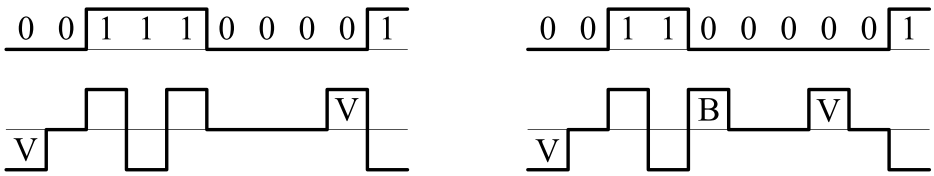
- pogrešan je peti bit, koji treba da bude „0”, pa je poslata sekvenca $+-0+00-+$,
- pogrešan je šesti bit, koji treba da bude „$+$”, pa je poslata sekvenca $+-0+-+-+$, ili
- pogrešan je sedmi bit, koji treba da bude „0”, pa je poslata sekvenca $+-0+-00+$.

Zadatak 1.11 Linijski kod HDB 3 je AMI kod, koji se formira prema sledećim pravilima:

1. Ako se pojavi sekvenca četiriju nula, onda se četvrta nula zamenjuje bitom narušavanja (V). Polaritet V-bita isti je kao polaritet prethodnog bita „1”, čime se narušava princip alternativne promene znaka (AMI).
2. Ako se između novog i prethodnog V-bita nalazi paran broj jedinica, prva nula sekvence četiriju nula se zamenjuje B-bitom čiji je polaritet u saglasnosti s principom AMI.

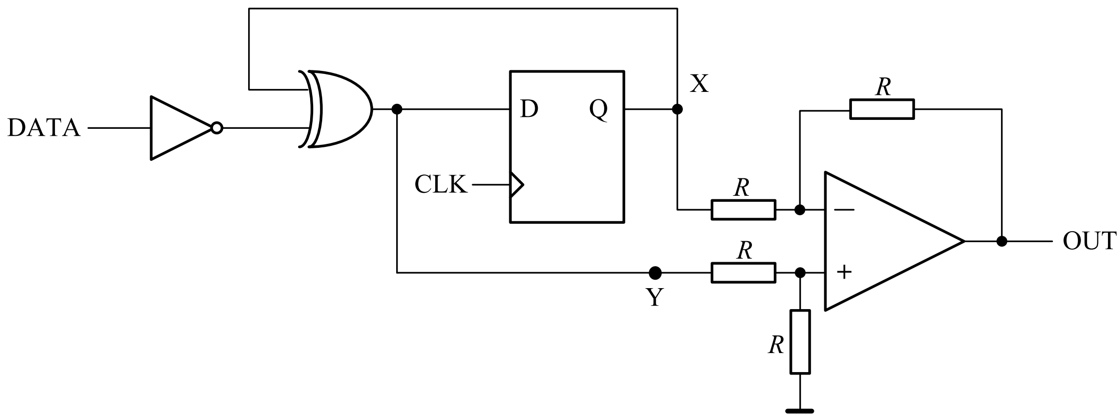
Primenjujući ova pravila, predstavite sekvence 0011100001 i 0011000001 HDB 3 kodom. Pretpostavite da je prva nula u sekvencama bit narušavanja negativnog polariteta.

U slučaju prve sekvence, primenjuje se samo pravilo 1, dok se za drugu sekvencu primenjuju oba pravila. Rezultati su prikazani na narednoj slici.



Slika 1.11: *Predstavljanje sekvenci kodom HDB 3.*

Zadatak 1.12 Na slici je prikazana shema linijskog kodera. Na njegov ulaz dolazi signal podataka (DATA), predstavljen unipolarnim NRZ kodom. CLK je signal takta. U početnom trenutku, u D flip–flop je upisana jedinica.



Slika 1.12: *Linijski koder.*

Analizirajte rad uređaja. Kojim je kodom predstavljen signal na njegovome izlazu?

Deo kola desno od tačaka X i Y predstavlja idealan oduzimač, pa je

$$\mathrm{OUT} = \mathrm{Y} - \mathrm{X}.$$

Signal na ulazu D flip–flopa je

$$\mathrm{D} = \mathrm{X} \oplus \overline{\mathrm{DATA}},$$

dok su signali X i Y dati izrazima

$$\mathrm{X} = \mathrm{Q},$$
$$\mathrm{Y} = \mathrm{D}.$$

Logički nivoi signala.

DATA	D	Q_t	Q_{t+1}	OUT
1	1	1	1	0
1	1	1	1	0
0	0	1	0	–1
0	1	0	1	1
1	1	1	1	0
0	0	1	0	–1
1	0	0	0	0

Rad uređaja najlakše ćemo analizirati ako na njegov ulaz dovedemo neku sekvencu, npr. 1100101 i posmatramo šta se dobija na izlazu. Tada možemo popuniti sledeću tabelu.

Vidimo da linijski koder sve logičke jedinice s ulaza predstavlja nultim naponom na izlazu, dok se logičke nule s ulaza naizmenično predstavljalju kao -1 i $+1$. To znači da se radi o tzv. pseudoternarnom kodu, koji se koristi u ISDN.

Zadatak 1.13 Video-signal čiji spektar zauzima opseg frekvencija od 556 kHz do 6,056 MHz prenosi se koaksijalnim kablom čiji su parametri: $a = 1{,}3$ mm, $b = 4{,}7$ mm, $\epsilon_r = 1{,}05$, $\rho_p = 0{,}01785$ $\Omega\text{mm}^2/\text{m}$, $L = 1200$ m. Ako je nivo signala na ulazu kabla -3 dBm, koliki je nivo signala na njegovome izlazu?

Karakteristična impedansa ovoga kabla je

$$Z_c \approx \frac{60\ \Omega}{\sqrt{\epsilon_r}} \ln\frac{b}{a} \approx 75\ \Omega,$$

što je uobičajena vrednost za televizijsku tehniku.

Kada su u koaksijalnom kablu izraženi gubici usled skin-efekta, njegovo podužno slabljenje dato je izrazom

$$\alpha = \sqrt{\mu_0 \pi f \rho_p}\ \frac{\frac{1}{a}+\frac{1}{b}}{4\pi Z_c}.$$

Vidimo da podužno slabljenje raste s korenom frekvencije. Pošto spektar prenošenog signala zauzima širok opseg frekvencija, u proračunu ćemo posmatrati najgori slučaj, kada je $f = f_{max} = 6{,}056$ MHz, jer je tada podužno slabljenje kabla najveće i iznosi $6{,}8 \cdot 10^{-4}$ Np/m (nepera po metru).

Ukupno slabljenje kabla je

$$a = \alpha L \approx 0{,}82 \text{ Np}.$$

Danas je uobičajeno da se slabljenja, dobici i nivoi izražavaju u decibelima, dok neperi imaju istorijski značaj. Između ovih dveju jedinica važi odnos

$$1 \text{ Np} = 8{,}686 \text{ dB}.$$

Slabljenje kabla na najvišoj prenošenoj frekvenciji stoga je

$$a = 7{,}12 \text{ dB}.$$

Nivo signala na izlazu kabla je

$$p_{out} = p_{in} - a = -10{,}12 \text{ dBm}.$$

Zadatak 1.14 Na slici je data jednopolna shema kablovskog distributivnog sistema (KDS) u stambenoj zgradi, dok su u tabelama navedene karakteristike upotrebljenih komponenti. U kom se opsegu može nalaziti nivo signala na ulazu?

Slabljenja utičnica i kablova [dB]:

	85 MHz	860 MHz
utičnica	0,3	0,7
RG 6 (/100 m)	5	23
RG 11 (/100 m)	4	20

Karakteristike razdelnika snage (kataloški podaci Televés):

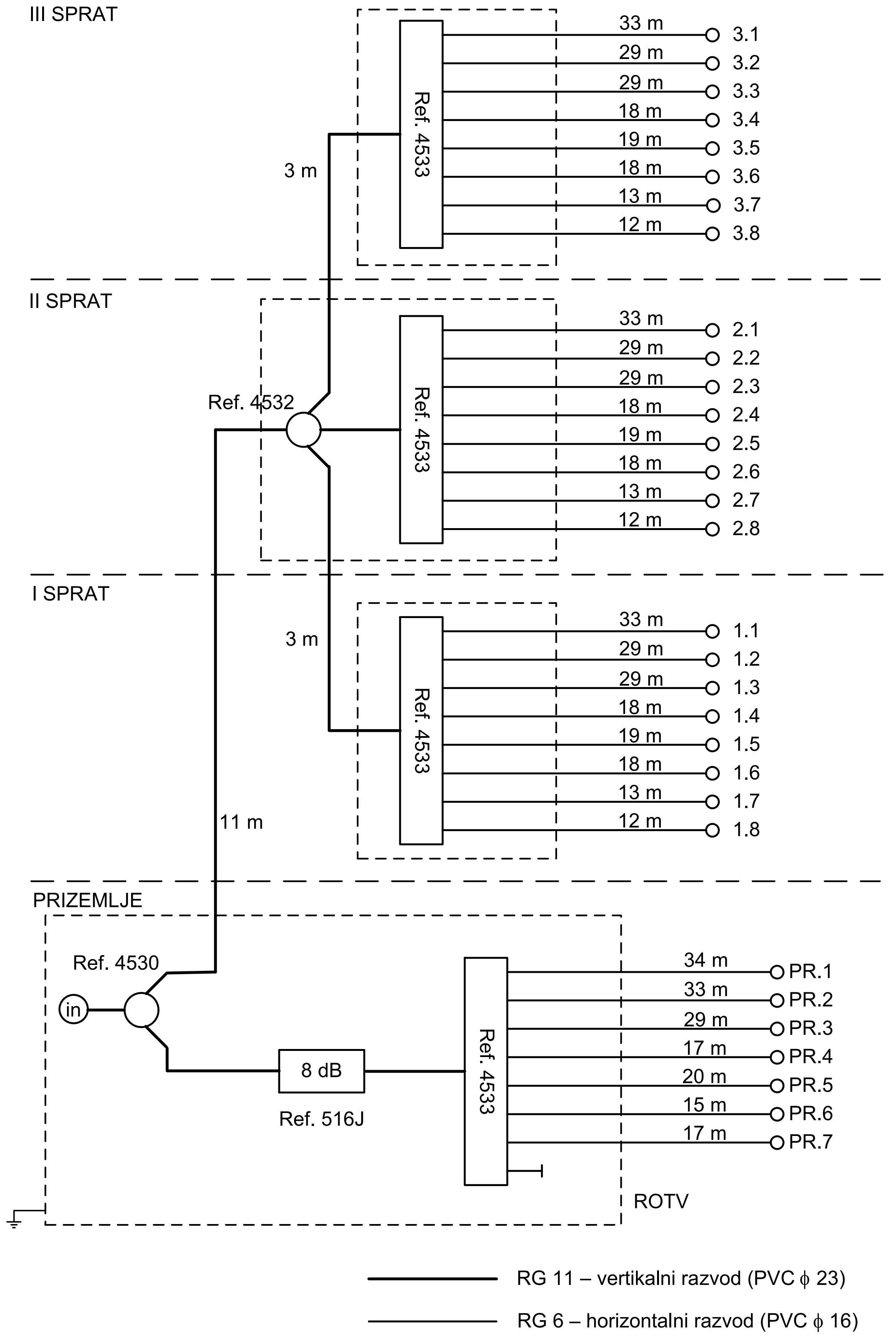
Reference			4530	4532	4531	4534	4533
Nº of outputs		nº	2	3	4	6	8
Frequency range			SCATV				
Insertion losses In →Out		dB	4	6	8	10	11
Rejection between outputs		dB	> 30			> 28	
Screening	30-300 MHz	dB	> 85				
	300-470 MHz	dB	> 80				
	470-1000 MHz	dB	> 75				
Dimensions(Wid x Hei x Dep)		mm	55x50x27	78x50x27		122x60x29	
Weight		gr	120		175	225	

Prema standardu EN 60728, propisani nivo signala u KDS (na utičnicama) je od 54 dBμV do 74 dBμV. Da bismo odredili pripadajući opseg signala na ulazu instalacije, najpre moramo odrediti njeno slabljenje. Primetimo da slabljenja utičnica i kablova zavise od frekvencije, a razdelnika snage (parametar *insertion loss*) ne.

Najveće slabljenje signala tražimo na najvišoj prenošenoj frekvenciji (UHF, 860 MHz), na utičnici do koje signal prolazi kroz najdužu trasu kabla i/ili kroz sklopove koji unose najveće slabljenje. Kandidati su PR.1 i 1.1/3.1.

Za utičnicu PR.1, signal prolazi redom kroz sklopove 4530, 516J i 4533, zatim kroz 34 m kabla RG 6 i dolazi do utičnice. Slabljenje ove trase iznosi 31,52 dB.

Za utičnice 1.1 i 3.1, signal prolazi kroz sklop 4530, zatim kroz 11 m kabla RG 11, pa kroz sklop 4532, još 3 m kabla RG 11, sklop 4533, 33 m kabla RG 6 i, konačno, dolazi



Slika 1.14: *Jednopolna shema instalacije KDS.*

do utičnice. Ukupno slabljenje ove trase je 32,09 dB. Zaključujemo da je ovo najveće slabljenje u instalaciji.

Najmanje slabljenje tražimo na donjoj granici prenošenog frekvencijskog opsega (VHF, 85 MHz), za šta su kandidati utičnice PR.6 i 2.8.

U slučaju PR.6, na trasi signala su sklopovi 4530, 516J i 4533, 15 m kabla RG 6 i, na kraju, utičnica. Ukupno slabljenje je 24,05 dB.

Do utičnice 2.8, signal prolazi kroz sklop 4530, 11 m kabla RG 11, sklopove 4532 i 4533, 12 m kabla RG 6 i dolazi do utičnice. Slabljenje ove trase je 22,34 dB i to je najmanje slabljenje u instalaciji.

Minimalni dopušteni nivo signala na ulazu sada je

$$l_{in,min} = l_{min} + a_{max} = 54 \text{ dB}\mu\text{V} + 32{,}09 \text{ dB} = 86{,}09 \text{ dB}\mu\text{V},$$

dok je maksimalni dopušteni nivo na ulazu

$$l_{in,max} = l_{max} + a_{min} = 74 \text{ dB}\mu\text{V} + 22{,}34 \text{ dB} = 96{,}34 \text{ dB}\mu\text{V}.$$

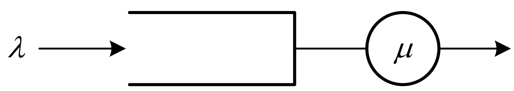
2. Servisni sistemi

Zadatak 2.1 Izvedite izraze za verovatnoću stanja servisnog sistema M/M/1, prosečan broj korisnika u njemu, prosečne brojeve korisnika u čekaonici i radionici, kao i prosečna vremena zadržavanja u sistemu i čekaonici.

Za sistem M/M/1 (ili M/M/1/∞/∞/FCFS, kako je njegova puna oznaka po Kendallovoj notaciji) važe sledeće pretpostavke:

- proces dolazaka korisnika je Poissonov, s prosečnim protokom λ,
- vreme obrade u radionici ima eksponencijalnu raspodelu, s prosečnim trajanjem $\overline{\tau} = 1/\mu$, gde je μ protok obrade,
- radionica ima jednog servisera,
- čekaonica je beskonačnog kapaciteta,
- populacija korisnika je beskonačna i
- disciplina opsluživanja korisnika je FCFS (*First–Come–First–Served*).

Simbolička oznaka ovog servisnog sistema prikazana je na slici.

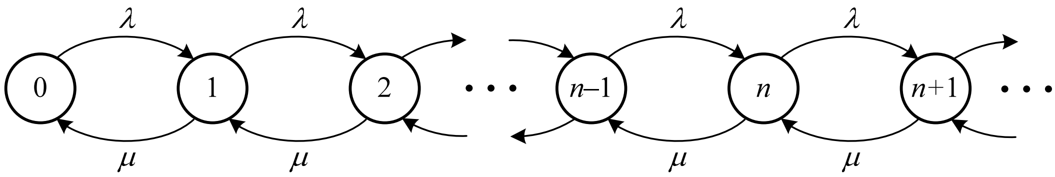


Slika 2.1: *Servisni sistem M/M/1.*

Stanje sistema odgovara broju korisnika, n, koji se u njemu nalaze. Za $n = 0$, u sistemu se ne nalazi nijedan korisnik. Za $n \geqslant 1$, u sistemu se nalazi ukupno n korisnika i to $n-1$ u čekaonici i 1 u radionici. Dijagram stanja sistema dat je na sledećoj slici.

Označimo s p_n verovatnoću da se sistem nalazi u stanju n, tj. da u njemu ima n korisnika. Na osnovu principa konzervacije protoka korisnika, u stacionarnom stanju važi

$$p_n\lambda = p_{n+1}\mu, \ n = 0, 1, 2, \ldots .$$



Dijagram stanja sistema M/M/1.

Uvedemo li veličinu $\rho = \lambda/\mu$, koja se naziva iskorišćenošću servera i predstavlja stacionarnu verovatnoću njegovog zauzeća, možemo pisati

$$p_{n+1} = \rho p_n = \rho^2 p_{n-1} = \ldots = \rho^{n+1} p_0.$$

S druge strane, zbog normiranosti verovatnoće biće

$$\sum_{n=0}^{\infty} p_n = 1.$$

Odavde dobijamo izraz za verovatnoću da se u sistemu nalazi n korisnika:

$$p_n = \rho^n (1-\rho),\ \ n = 0, 1, 2, \ldots$$

Da bi sistem bio stabilan, neophodno je da iskorišćenost servera bude manja od jedan.

Do istog rezultata mogli smo doći i na drugi način. Dijagram stanja ovog servisnog sistema odgovara markovskom lancu, za koji je matrica prelaska

$$\mathbf{Q} = \begin{bmatrix} 1-\lambda & \mu & 0 & 0 & \cdots \\ \lambda & 1-\lambda-\mu & \mu & 0 & \cdots \\ 0 & \lambda & 1-\lambda-\mu & \mu & \cdots \\ 0 & 0 & \lambda & 1-\lambda-\mu & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}.$$

Definišemo li vektor verovatnoća stanja

$$\mathbf{P} = [p_0\ \ p_1\ \ \ldots]^T,$$

u stacionarnom stanju će važiti

$$\mathbf{QP} = \mathbf{P},$$

odakle se, uz pozivanje na normiranost, mogu odrediti verovatnoće stanja.

U literaturi se matrica prelaska definiše i drugačije. Matričnu jednačinu stacionarnog stanja možemo napisati i kao

$$(\mathbf{Q} - \mathbf{I})\,\mathbf{P} = \mathbf{0}.$$

Transponovanjem obeju strana, dobijamo

$$\mathbf{P}^T (\mathbf{Q} - \mathbf{I})^T = \mathbf{0},$$

odnosno

$$\mathbf{P}^T \mathbf{W} = \mathbf{0},$$

gde je

$$\mathbf{W} = \begin{bmatrix} -\lambda & \lambda & 0 & 0 & \cdots \\ \mu & -\lambda-\mu & \lambda & 0 & \cdots \\ 0 & \mu & -\lambda-\mu & \lambda & \cdots \\ 0 & 0 & \mu & -\lambda-\mu & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}.$$

I ovaj sistem jednačina po verovatnoćama stanja je homogen. Netrivijalno rešenje je jedino moguće ukoliko je barem jedna jednačina linearna kombinacija ostalih, pa je stoga sistem potrebno dopuniti i uslovom normiranosti verovatnoće.

Primetimo da broj korisnika u sistemu ima geometrijsku raspodelu. Prosečan broj korisnika jednak je njenom očekivanju i iznosi

$$N = \sum_{n=0}^{\infty} np_n = \frac{\rho}{1-\rho} = \frac{\lambda}{\mu-\lambda}.$$

Prosečan broj korisnika u čekaonici je

$$N_Q = \sum_{n=1}^{\infty} (n-1)p_n = \frac{\rho^2}{1-\rho},$$

jer se, po pretpostavci, u čekaonici nalazi $n-1$ korisnik. Prosečan broj korisnika u radionici je

$$N_S = \sum_{n=1}^{\infty} 1 \cdot p_n = 1 \cdot (1-p_0) = \rho,$$

jer je server zauzet uvek kada se u sistemu nalazi barem jedan korisnik.

Po Littleovoj teoremi, prosečno zadržavanje u sistemu je

$$T = \frac{N}{\lambda} = \frac{1}{\mu-\lambda}.$$

Prosečno trajanje čekanja u čekaonici je

$$T_Q = \frac{N_Q}{\lambda} = \frac{\rho}{\mu-\lambda},$$

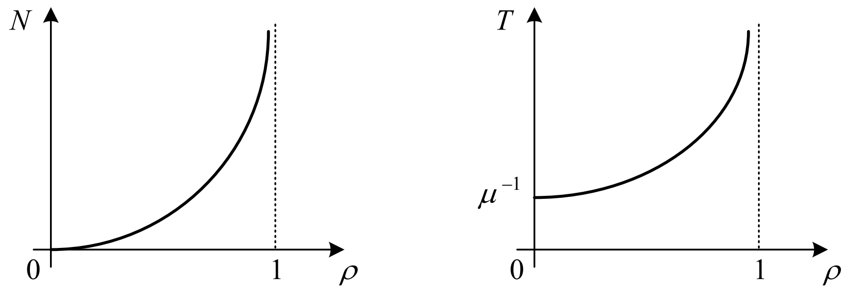
a u radionici

$$T_S = \frac{N_S}{\lambda} = \frac{1}{\mu},$$

što je u skladu sa polaznom pretpostavkom. Primetimo da zaista važi $T = T_Q + T_S$.

Zavisnosti broja korisnika u sistemu i njihovog zadržavanja od iskorišćenosti servera prikazane su na narednoj slici.

Zadatak 2.2 Merenjem je utvrđeno da javnu telefonsku govornicu svakog dana u intervalu od 11 h do 15 h u proseku koristi 18 osoba, pri čemu je srednje trajanje korišćenja 5 minuta. Uz pretpostavku da su dolasci korisnika nezavisni, s Poissonovom raspodelom, a da trajanje korišćenja govornice ima eksponencijalnu raspodelu, odredite:



Performanse sistema M/M/1.

a) verovatnoću da je govornica slobodna,

b) prosečan broj osoba koje čekaju da se oslobodi govornica i

c) prosečno trajanje čekanja da se oslobodi govornica.

Na osnovu postavke, zaključujemo da se posmatrana telefonska govornica može modelirati sistemom M/M/1, s parametrima

$$\lambda = \frac{18\ \mathrm{kor}}{4\ \mathrm{h}} = 4{,}5\ \frac{\mathrm{kor}}{\mathrm{h}} = 0{,}075\ \frac{\mathrm{kor}}{\mathrm{min}},$$

$$\mu = \frac{1}{5\ \mathrm{min/kor}} = 0{,}2\ \frac{\mathrm{kor}}{\mathrm{min}}.$$

Iskorišćenost servera je

$$\rho = \frac{\lambda}{\mu} = 0{,}375.$$

Na osnovu rešenja prethodnog zadatka, jednostavno dobijamo:

a) verovatnoća da je govornica slobodna je

$$p_0 = 1 - \rho = 0{,}625,$$

b) prosečan broj osoba koje čekaju da se oslobodi govornica je

$$N_Q = \frac{\rho^2}{1-\rho} = 0{,}225,$$

c) prosečno trajanje čekanja da se oslobodi govornica je

$$T_Q = \frac{\rho}{\mu - \lambda} = 3\ \mathrm{min}.$$

Zadatak 2.3 U bafer mrežnog uređaja svake minute u proseku dolazi 30 poruka, čija je dužina eksponencijalno raspodeljena, sa srednjom vrednošću 1000 B. Protok na odlaznom linku je 9600 b/s. Odredite potreban kapacitet bafera, tako da verovatnoća odbacivanja poruke ne bude veća od 1%.

Bafer i odlazni link mogu se modelirati sistemom M/M/1/m. Poruke dolaze u sistem s protokom

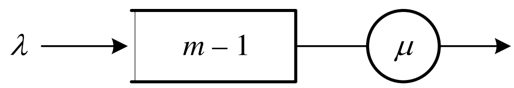
$$\lambda = \frac{30\ \mathrm{por}}{60\ \mathrm{s}} = 0{,}5\ \frac{\mathrm{por}}{\mathrm{s}},$$

dok je srednji protok obrade

$$\mu = \frac{9600\ \mathrm{b/s}}{1000 \cdot 8\ \mathrm{b/por}} = 1{,}2\ \frac{\mathrm{por}}{\mathrm{s}}.$$

Iskorišćenost servera stoga je

$$\rho = \frac{\lambda}{\mu} = 0{,}417.$$



Slika 2.3: *Servisni sistem M/M/1/m.*

Verovatnoća stanja u sistemu M/M/1/m data je izrazom

$$p_n = \begin{cases} \dfrac{(1-\rho)\rho^n}{1-\rho^{m+1}}\,, & \text{za } 0 \leqslant n \leqslant m \\ 0\,, & \text{inače} \end{cases}.$$

Novi korisnici se odbacuju onda kada je čekaonica puna, tj. kada se u sistemu nalazi m korisnika, od toga $m-1$ u čekaonici i 1 u radionici. Verovatnoća ovoga događaja naziva se verovatnoćom blokade i data je izrazom

$$p_B = p_m = \frac{(1-\rho)\rho^m}{1-\rho^{m+1}},$$

jer važi osobina PASTA (*Poisson Arrivals See Time Averages*). Da se ne bi premašila zadata verovatnoća blokade, kapacitet sistema treba da bude

$$m = \frac{1}{\log\rho}\log\frac{p_B}{1-(1-p_B)\rho}.$$

Uslov iz postavke našeg zadatka je $p_B \leqslant 0{,}01$. Uvrštavanjem brojčanih vrednosti, dobijamo da je u graničnom slučaju $m = 4{,}656$. Usvajamo prvu veću celobrojnu vrednost, tj. $m = 5$. Ovo je kapacitet *celog sistema*, dok je traženi kapacitet bafera 4 poruke, ili 4000 B.

Zadatak 2.4 Umreženi štampač ima memoriju kapaciteta 1,5 MB i može da štampa 10 stranica u minuti. Korisnici svakog sata na štampanje šalju u proseku 40 dokumenata od po 3 stranice, veličine 150 kB. Korišćenjem modela sistema M/M/1/m, odredite prosečno vreme koje protekne od slanja dokumenta na štampu, dok se on ne bude odštampao.

Parametri ovog servisnog sistema su:

- protok dolazaka korisnika (dokumenata) $\lambda = \frac{2}{3}\ \mathrm{min}^{-1}$,
- prosečno trajanje obrade korisnika $\mu^{-1} = 0{,}3$ min,
- kapacitet sistema $m = 10{,}24$ korisnika; usvajamo prvu manju celobrojnu vrednost, tj. $m = 10$,
- iskorišćenost servera $\rho = 0{,}2$.

Prosečno zadržavanje korisnika u sistemu je

$$T = \frac{N}{\gamma} = \frac{N}{\lambda(1 - p_B)},$$

gde je N prosečan broj korisnika u sistemu, $\gamma = \lambda(1 - p_B)$ protok korisnika na izlazu i p_B verovatnoća blokade.

Prosečan broj korisnika u sistemu je

$$N = \sum_{n=0}^{m} n p_n = \sum_{n=0}^{m} n \frac{(1-\rho)\rho^n}{1-\rho^{m+1}} = \\ = \rho \frac{1 - \rho^m(1 + m(1-\rho))}{(1-\rho)(1-\rho^{m+1})}$$

i u našem slučaju iznosi 0,25.

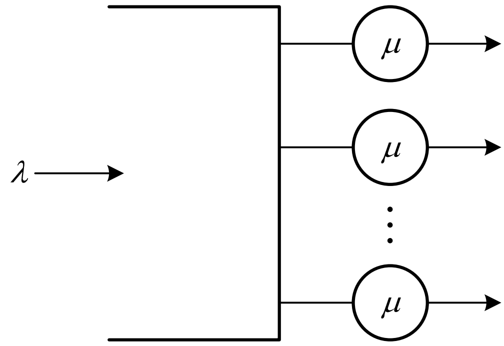
Verovatnoća blokade je

$$p_B = p_m = \frac{(1-\rho)\rho^m}{1-\rho^{m+1}} = 8{,}192 \cdot 10^{-8}.$$

Konačno dobijamo da dokument u proseku čeka 22,5 s da bi se odštampao. Zanimljivo je primetiti da dokument 4,5 s čeka u redu, dok samo štampanje traje 18 s.

Zadatak 2.5 U internet-kafeu se nalazi 8 računara. Svakog sata, u kafe u proseku uđe 9 korisnika, čija vremena međudolazaka imaju eksponencijalnu raspodelu. Vreme zadržavanja korisnika za računarom takođe ima eksponencijalnu raspodelu, sa srednjom vrednošću 50 min. Pod pretpostavkom da je broj mesta u kafeu veoma velik, odredite verovatnoću da u kafeu nema slobodnih računara.

U ovome se slučaju radi o servisnom sistemu M/M/m, čiji je simbolički prikaz dat na slici.



Slika 2.5: *Servisni sistem M/M/m.*

Verovatnoća stanja ovoga sistema data je izrazom

$$p_n = \begin{cases} \dfrac{(m\rho)^n}{n!} p_0, & 1 \leqslant n < m \\ \dfrac{m^m \rho^n}{m!} p_0, & n \geqslant m \end{cases}.$$

Pri tome je s p_0 označena verovatnoća da u sistemu nema nijednog korisnika:

$$p_0 = \frac{1}{\displaystyle\sum_{n=0}^{m-1} \frac{(m\rho)^n}{n!} + \sum_{n=m}^{\infty} \frac{m^m \rho^n}{m!}},$$

dok je $\rho = \lambda/(m\mu)$ iskorišćenost servera. Primetimo da se u sistemu M/M/m veličina $A = \lambda/\mu$ naziva *intenzitetom saobraćaja* i izražava u erlanzima (E).

Verovatnoća zauzeća svih m servera naziva se i verovatnoćom čekanja. Ona je jednaka verovatnoći da se u sistemu nalazi barem m korisnika:

$$P_Q = \sum_{n=m}^{\infty} p_n = \frac{(m\rho)^m}{m!(1-\rho)} p_0.$$

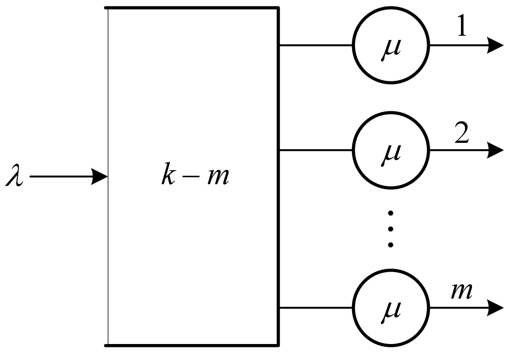
Ova formula je poznata i kao *Erlangova C formula.*

U našem zadatku je $m = 8$, $\lambda = 0{,}15\ \text{min}^{-1}$ i $\mu = 0{,}02\ \text{min}^{-1}$, dok je iskorišćenost računara

$$\rho = \frac{\lambda}{m\mu} = 0{,}9375.$$

Uvrštavanjem izračunatih brojčanih vrednosti, dobijamo da je $p_0 = 2{,}032 \cdot 10^{-4}$, pa je verovatnoća čekanja $P_Q = 0{,}80725$.

Zadatak 2.6 Ponovite prethodni zadatak ukoliko u kafeu ima mesta za 20 osoba.



Slika 2.6: *Servisni sistem M/M/m/k.*

Kada je kapacitet kafea ograničen, primenićemo model servisnog sistema M/M/m/k, koji je simbolički prikazan na slici 2.6.

Verovatnoća da se u ovom sistemu nalazi n korisnika je

$$p_n = \begin{cases} \dfrac{(m\rho)^n}{n!} p_0, & 1 \leqslant n < m \\ \dfrac{m^m \rho^n}{m!} p_0, & m \leqslant n \leqslant k \\ 0, & n > k \end{cases}.$$

S p_0 opet je označena verovatnoća da u sistemu nema nijednog korisnika,

$$p_0 = \frac{1}{\displaystyle\sum_{n=0}^{m-1} \frac{(m\rho)^n}{n!} + \sum_{n=m}^{k} \frac{m^m \rho^n}{m!}},$$

dok je $\rho = \lambda/(m\mu)$ iskorišćenost servera.

Zbog osobine PASTA, verovatnoća čekanja ponovo je jednaka verovatnoći da se u sistemu nalazi barem m korisnika:

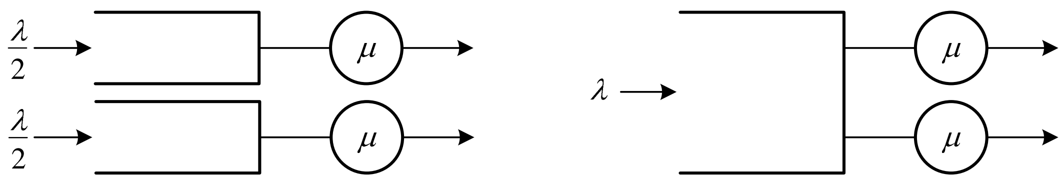
$$P_Q = \sum_{n=m}^{k} p_n = \frac{m^m}{m!} \frac{\rho^m - \rho^{k+1}}{1-\rho} p_0.$$

Zamenom brojčanih vrednosti, dobijamo da je $p_0 = 3{,}121 \cdot 10^{-4}$, pa je verovatnoća čekanja $P_Q = 0{,}70399$. Zbog čega je ova vrednost manja nego u prethodnom zadatku?

Zadatak 2.7 „Usko grlo” u telekomunikacionom centru predstavlja mrežni uređaj koji se sastoji od bafera velikog kapaciteta i procesorske jedinice. Dolasci paketa u ovaj uređaj mogu se opisati Poissonovim slučajnim procesom, protoka 4000 s^{-1}. Srednje trajanje obrade paketa u procesoru uređaja je 0,2 ms. Da bi se povećala propusnost, moguće je paralelno ovom uređaju vezati još jedan takav, koji bi s njim radio u režimu deljenja saobraćajnog opterećenja, ili je moguće postojeći uređaj proširiti još jednom procesorskom jedinicom. Analizirajte ove dve strategije sa stanovišta prosečnog zadržavanja paketa u uređaju.

Kapacitet bafera je *veliki*, pa ćemo posmatrani uređaj modelirati servisnim sistemom M/M/1.

Paralelnu vezu uređaja možemo ekvivalentirati paralelnom vezom sistema M/M/1 jednakih karakteristika, u koje korisnici dolaze s prosečnim protokom $\lambda/2$. Ako se originalnom uređaju dodaje još jedna procesorska jedinica, ekvivalentni servisni sistem biće M/M/2, što je ilustrovano na slici.



Slika 2.7: *Strategije za povećanje propusne moći uređaja.*

U oba slučaja, iskorišćenost servera je

$$\rho = \frac{\lambda}{2\mu} = 0{,}4.$$

Zadržavanje paketa u slučaju paralelne veze je

$$T = \frac{1}{\mu}\frac{1}{1-\rho} = 0{,}33 \text{ ms}.$$

Ako se primenjuje druga strategija, verovatnoća da u sistemu nema korisnika biće

$$p_0 = \frac{1}{1 + 2\rho + 2\sum_{n=2}^{\infty}\rho^n} = \frac{1-\rho}{1+\rho}.$$

Verovatnoća da se u sistemu nalazi $n \geqslant 1$ korisnika biće

$$p_n = 2\rho^n p_0.$$

Prosečan broj korisnika u sistemu je

$$N = \sum_{n=1}^{\infty} np_n = 2p_0\sum_{n=1}^{\infty} n\rho^n = \frac{2\rho}{1-\rho^2}.$$

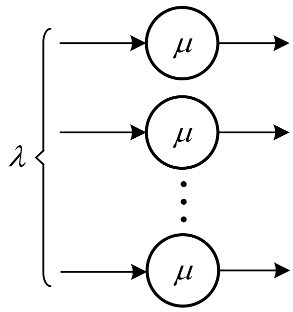
Pozivanjem na Littleovu teoremu, dobijamo izraz za prosečno zadržavanje u sistemu M/M/2:

$$T = \frac{N}{\lambda} = \frac{1}{\mu}\frac{1}{1-\rho^2},$$

što iznosi 0,24 ms. Zaključujemo da je zadržavanje paketa manje u sistemu sa zajedničkim baferom.

Zadatak 2.8 Telefonska centrala u poslovnoj zgradi ima 20 linija ka javnoj telefonskoj mreži. Ako se tokom sata vršnog opterećenja javlja 80 spoljnjih poziva, prosečnog trajanja 12 minuta, odredite verovatnoću da su sve linije zauzete. Pretpostavite da je generisanje poziva Poissonov slučajni proces, a da njihovo trajanje ima eksponencijalnu raspodelu.

Telefonske linije predstavljaju servisni sistem M/M/m/m. To je sistem *bez čekaonice*, s m servera, kao što je prikazano na slici.



Slika 2.8: *Servisni sistem M/M/m/m.*

Verovatnoća stanja ovoga sistema data je izrazom

$$p_n = \frac{\dfrac{A^n}{n!}}{\displaystyle\sum_{i=0}^{m} \frac{A^i}{i!}}.$$

Veličina $A = \lambda/\mu$ naziva se intenzitetom saobraćaja, dok je iskorišćenost servera $\rho = A/m$.

Verovatnoća blokade data je *Erlangovom B formulom*:

$$p_B = p_m = \frac{\dfrac{A^m}{m!}}{\displaystyle\sum_{i=0}^{m} \frac{A^i}{i!}}.$$

Za centralu iz postavke zadatka je $m = 20$ i $A = 16$ E. Verovatnoća zauzeća svih linija je $p_B = 0{,}0644$.

Zadatak 2.9 Odredite potreban broj spojnih vodova, m, između dveju telefonskih centrala, ukoliko je srednja vrednost ponuđenog saobraćaja $A = 16{,}68$ E, a zahtevani nivo servisa $p_B = 0{,}001$.

Posmatrani spojni vodovi predstavljaju servisni sistem $M/M/m/m$. Verovatnoća njegove blokade naziva se i nivoom servisa (GoS – *grade of service*).

Probom ili očitavanjem iz tablice Erlangove B formule, koja je data na strani 244, jednostavno dobijamo $n = 30$.

Zadatak 2.10 Poruke fiksne dužine 1000 B, koje se generišu po Poissonovoj raspodeli, prenose se linijom na kojoj je protok 9600 b/s. Iskorišćenost linije iznosi 70%. Odredite prosečno trajanje čekanja.

Dužina poruke je fiksna, pa je fiksno i vreme potrebno za njihov prenos i iznosi

$$\tau = \mu^{-1} = \frac{1000 \cdot 8 \text{ b}}{9600 \text{ b/s}} = 0{,}83 \text{ s}.$$

Zbog toga se ovakva linija može modelirati sistemom M/D/1. Iskorišćenost njegovog servera je $\rho = 0{,}7$.

Prosečno trajanje čekanja u sistemu M/D/1 odredićemo iz Pollaczek-Khinchinove (P--K) formule:

$$T_Q = \frac{\lambda\overline{\tau^2}}{2(1-\rho)}.$$

U sistemu M/D/1, trajanje obrade je konstantno, pa je očekivanje njegovog kvadrata (tj. drugi moment)

$$\overline{\tau^2} = \frac{1}{\mu^2}.$$

Sada je trajanje čekanja

$$T_Q = \frac{\rho}{2\mu(1-\rho)}$$

i u posmatranom slučaju iznosi 0,97 s.

Zadatak 2.11 Ponovite prethodni zadatak za slučaj kada je dužina poruke slučajna promenljiva, uniformno raspodeljena na intervalu [100, 10000] b.

Trajanje obrade poruke čija je dužina L je $\mu^{-1} = L/R$, gde je R protok na liniji. Pošto je L slučajna promenljiva s uniformnom raspodelom, to isto će biti i trajanje obrade, i to na intervalu [0,01042, 1,042] s; stoga se radi o sistemu M/G/1, pa ćemo srednje trajanje čekanja odrediti iz Pollaczek-Khinchinove formule:

$$T_Q = \frac{\lambda\overline{\tau^2}}{2(1-\rho)}.$$

Srednje trajanje obrade jednako je očekivanju odgovarajuće uniformne raspodele:

$$\overline{\tau} = \frac{1}{\mu} = \frac{0{,}01042 \text{ s} + 1{,}042 \text{ s}}{2} = 0{,}526 \text{ s}.$$

Pošto je $\rho = 0{,}7$, biće $\lambda = \mu \cdot \rho = 1{,}331\ \mathrm{s}^{-1}$.

Očekivanje kvadrata trajanja obrade je

$$\overline{\tau^2} = \frac{(0{,}01042\ \mathrm{s})^2 + 0{,}01042\ \mathrm{s} \cdot 1{,}042\ \mathrm{s} + (1{,}042\ \mathrm{s})^2}{3} = 0{,}366\ \mathrm{s}^2.$$

Srednje trajanje čekanja stoga iznosi 0,812 s.

Zadatak 2.12 Paketi koji se generišu u skladu s Poissonovim slučajnim procesom, prenose se linijom na kojoj je protok 12000 b/s i čija je iskorišćenost 81%. Odredite prosečno zadržavanje paketa u predajnom baferu, ako je matematičko očekivanje njihove dužine 1000 B, a varijansa 10000 B^2.

Primetimo da se i ovde radi o servisnom sistemu M/G/1. Ako dužina paketa, L ima matematičko očekivanje μ i varijansu σ^2, trajanje obrade imaće matematičko očekivanje μ/R i varijansu σ^2/R^2, gde je R protok na liniji. Odavde je matematičko očekivanje trajanje obrade

$$\overline{\tau} = \frac{\mu}{R} = \frac{2}{3}\ \mathrm{s},$$

dok je drugi moment trajanja obrade

$$\overline{\tau^2} = \frac{\sigma^2}{R^2} + \overline{\tau}^2 = 449\ \mathrm{ms}.$$

Postavkom zadatka dato je $\rho = 0{,}81$, pa je protok dolazaka paketa

$$\lambda = \frac{\rho}{\overline{\tau}} = 1{,}215\ \mathrm{s}^{-1}.$$

Imamo sve potrebne podatke, pa uvrštavanjem u Pollaczek-Khinchinovu formulu dobijamo $T_Q = 1{,}436$ s.

Zadatak 2.13 Izvedite izraz za funkciju gustine verovatnoće trajanja čekanja u servisnom sistemu M/M/m. Korišćenjem ovoga rezultata, izvedite izraz za funkciju gustine verovatnoće trajanja zadržavanja korisnika u servisnom sistemu M/M/1.

Da bismo odredili funkciju gustine verovatnoće trajanja čekanja korisnika, odredićemo prvo funkciju raspodele vremena čekanja. Treba razlikovati slučajeve kada korisnik zatiče slobodnog servisera i stoga ne čeka u čekaonici i kada su serviseri zauzeti, pa korisnik mora čekati da bi došao na red:

$$F_{T_Q}(t) = \mathrm{P}\left(t_Q \leqslant t\right) = \mathrm{P}\left(t_Q = 0\right) + \mathrm{P}\left(t_Q \in (0,\ t]\right).$$

Korisnik neće čekati ako pri dolasku bude zatekao $n \leqslant m-1$ drugih korisnika u sistemu. Zbog osobine PASTA, verovatnoća zaticanja n korisnika u sistemu, q_n, jednaka je verovatnoći stanja n, pa je

$$\mathrm{P}\left(t_Q = 0\right) = \sum_{n=0}^{m-1} q_n = \sum_{n=0}^{m-1} p_n.$$

U suprotnom slučaju, ako u sistemu bude zatekao $n \geqslant m$ korisnika, od kojih je m u radionici i $n-m$ u čekaonici, moraće sačekati da $n-m+1$ korisnik završi obradu, pre nego što bude ušao u radionicu. Svaki od njih nosi količinu posla koja je eksponencijalno raspodeljena slučajna promenljiva. Ukupno trajanje čekanja jednako je zbiru $n-m+1$ nezavisnisne eksponencijalno raspodeljene slučajne promenljive, pa je

$$\mathrm{P}\left(t_Q \in (0,\, t]\right) = \sum_{n=m}^{\infty} q_n \int_0^t f_{\Xi^{(k)}}(x)\, \mathrm{d}x,$$

gde je $f_{\Xi^{(k)}}(x)$ funkcija gustine verovatnoće zbira $k = n-m+1$ nezavisne eksponencijalne slučajne promenljive. Zbog osobine PASTA i ovde je $q_n = p_n$, pa je

$$F_{T_Q}(t) = \sum_{n=0}^{m-1} p_n + \sum_{n=m}^{\infty} p_n \int_0^t f_{\Xi^{(k)}}(x)\, \mathrm{d}x.$$

Jednostavnim transformacijama dobijamo

$$\begin{aligned} F_{T_Q}(t) &= \left(\sum_{n=0}^{\infty} p_n - \sum_{n=m}^{\infty} p_n\right) + \sum_{n=m}^{\infty} p_n \int_0^t f_{\Xi^{(k)}}(x)\, \mathrm{d}x = \\ &= 1 - \sum_{n=m}^{\infty} p_n \left(1 - \int_0^t f_{\Xi^{(k)}}(x)\, \mathrm{d}x\right) = \\ &= 1 - \sum_{n=m}^{\infty} p_n \int_t^{\infty} f_{\Xi^{(k)}}(x)\, \mathrm{d}x. \end{aligned}$$

Odredimo sada funkciju fustine verovatnoće zbira eksponencijalnih slučajnih promenljivih. Nju možemo pronaći u statističkim priručnicima, a možemo ju i jednostavno izračunati preko karakterističnih funkcija. Podsetimo se da se sabiranjem slučajnih promenljivih njihove funkcije gustine verovatnoće konvoluiraju, a karakteristične funkcije množe. Ako je

$$\xi^{(k)} = \sum_{i=1}^{k} \tau_i,$$

pri čemu je $\tau_i \sim \mathrm{Exp}(\mu)$, karakteristična funkcija odgovarajuće raspodele će biti

$$\Phi_{\Xi^{(k)}}(s) = \mathcal{L}\left\{f_{\Xi^{(k)}}(x)\right\} = \prod_{i=1}^{k} \mathcal{L}\left\{f_T(\tau)\right\} = \prod_{i=1}^{k} \frac{\mu}{s+\mu} = \left(\frac{\mu}{s+\mu}\right)^k.$$

Funkciju gustine verovatnoće dobijamo nalaženjem inverzne Laplaceove transformacije:

$$f_{\Xi^{(k)}}(x) = \mathcal{L}^{-1}\left\{\Phi_{\Xi^{(k)}}(s)\right\} = \mu\,\frac{(\mu x)^{k-1}}{(k-1)!}\,e^{-\mu x}, \quad x \geqslant 0.$$

Ova raspodela se u literaturi naziva *k-Erlangovom raspodelom*. Sada dobijamo da je

$$F_{T_Q}(t) = 1 - \sum_{n=m}^{\infty} p_n \int_t^{\infty} \mu\,\frac{(\mu x)^{n-m}}{(n-m)!}\,e^{-\mu x}\,\mathrm{d}x.$$

Podsetimo li se izraza za verovatnoću stanja sistema M/M/m (zadatak 2.5), imaćemo

$$F_{T_Q}(t) = 1 - \sum_{n=m}^{\infty} \frac{m^m \rho^n}{m!}\,p_0 \int_t^{\infty} \mu\,\frac{(\mu x)^{n-m}}{(n-m)!}\,e^{-\mu x}\,\mathrm{d}x.$$

Integral i suma mogu zameniti mesta, pa je

$$F_{T_Q}(t) = 1 - \frac{m^m}{m!}\,p_0\mu \int_t^{\infty} \sum_{n=m}^{\infty} \rho^n\,\frac{(\mu x)^{n-m}}{(n-m)!}\,e^{-\mu x}\,\mathrm{d}x.$$

Primetimo da je

$$\sum_{n=m}^{\infty} \rho^n\,\frac{(\mu x)^{n-m}}{(n-m)!} = \rho^m e^{\mu\rho x},$$

pa će biti

$$F_{T_Q}(t) = 1 - \frac{m^m}{m!}\,p_0\mu\rho^m \int_t^{\infty} e^{\mu x(\rho-1)}\,\mathrm{d}x.$$

Pošto je $\rho < 1$, konačno dobijamo

$$F_{T_Q}(t) = 1 - \frac{m^m}{m!}\,\frac{\rho^m}{1-\rho}\,p_0 e^{-\mu(1-\rho)t}, \quad t \geqslant 0.$$

Funkciju gustine verovatnoće dobijamo nalaženjem izvoda:

$$f_{T_Q}(t) = \frac{\mathrm{d}F_{T_Q}(t)}{\mathrm{d}t} = A\delta(t) + \frac{m^m}{m!}\,\frac{\rho^m}{1-\rho}\,p_0\mu(1-\rho)e^{-\mu(1-\rho)t}h(t).$$

Vrednost konstante A odredićemo iz uslova normiranosti verovatnoće,

$$\int_{-\infty}^{+\infty} f_{T_Q}(t)\,\mathrm{d}t = A + \frac{m^m}{m!}\,\frac{\rho^m}{1-\rho}\,p_0 = 1,$$

odakle je

$$A = 1 - \frac{m^m}{m!}\,\frac{\rho^m}{1-\rho}\,p_0.$$

Za sistem M/M/1 je $p_0 = 1-\rho$, $m = 1$ i $A = 1-\rho$, pa je

$$f_{T_Q}(t) = (1-\rho)\delta(t) + \rho\mu(1-\rho)e^{-\mu(1-\rho)t}h(t).$$

Trajanje obrade u radionici ima eksponencijalnu raspodelu,

$$f_{T_S}(t) = \mu e^{-\mu t}h(t).$$

Ukupno zadržavanje korisnika u sistemu jednako je zbiru trajanja čekanja u čekaonici i trajanja obrade u radionici,

$$t_{uk} = t_Q + t_S.$$

Funkciju gustine verovatnoće zbira slučajnih promenljivih možemo izračunati kao konvoluciju pojedinačnih funkcija gustine verovatnoće, ili preko proizvoda karakterističnih funkcija. Ovde ćemo primeniti potonji pristup.

Karakteristična funkcija trajanja čekanja je

$$\Phi_{T_Q}(s) = \mathcal{L}\left\{f_{T_Q}(t)\right\} = 1-\rho+\mu\rho(1-\rho)\frac{1}{s+\mu(1-\rho)},$$

dok je karakteristična funkcija trajanja obrade

$$\Phi_{T_S}(s) = \mathcal{L}\left\{f_{T_S}(t)\right\} = \frac{\mu}{s+\mu}.$$

Karakteristična funkcija ukupnog zadržavanja korisnika u sistemu jednaka je njihovom proizvodu:

$$\Phi_T(s) = \Phi_{T_Q}(s)\Phi_{T_S}(s) = \frac{(1-\rho)\mu}{s+\mu} + \frac{\mu^2\rho(1-\rho)}{(s+\mu)(s+\mu(1-\rho))}.$$

Radi nalaženja inverzne Laplaceove transformacije, drugi sabirak ćemo razviti u sledeći oblik:

$$\frac{\mu^2\rho(1-\rho)}{(s+\mu)(s+\mu(1-\rho))} = \frac{A}{s+\mu} + \frac{B}{s+\mu(1-\rho)} = \frac{(A+B)s + A\mu(1-\rho) + B\mu}{(s+\mu)(s+\mu(1-\rho))}.$$

Izjednačavanjem koeficijenata, dobijamo

$$A + B = 0,$$
$$(A+B)\mu - A\mu\rho = \mu^2\rho(1-\rho),$$

odakle je $A = (\rho-1)\mu$ i $B = (1-\rho)\mu$. Uvrštavanjem u izraz za karakterističnu funkciju, nakon sređivanja dobijamo

$$\Phi_T(s) = \frac{(1-\rho)\mu}{s+\mu(1-\rho)}.$$

Funkcija gustine verovatnoće je

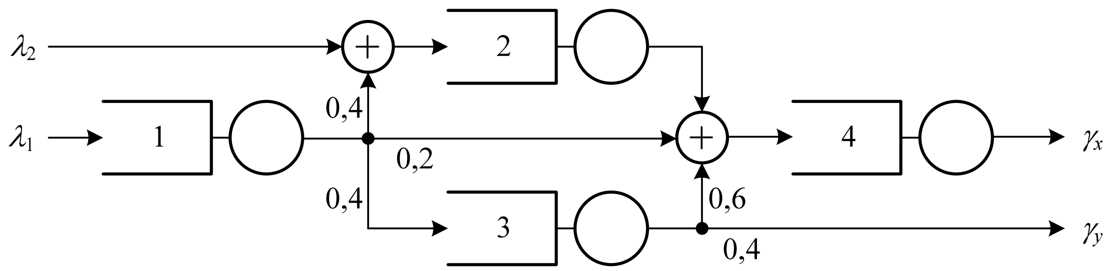
$$f_T(t) = \mathcal{L}^{-1}\left\{\frac{(1-\rho)\mu}{s+\mu(1-\rho)}\right\} = (1-\rho)\mu e^{-\mu(1-\rho)t}h(t).$$

Ovaj rezultat možemo proveriti nalaženjem matematičkog očekivanja:

$$\mathrm{E}T = \int_0^\infty t f_T(t)\,\mathrm{d}t = \frac{1}{\mu - (1-\rho)} = \frac{1}{\mu - \lambda},$$

što smo prethodno izveli u zadatku 2.1.

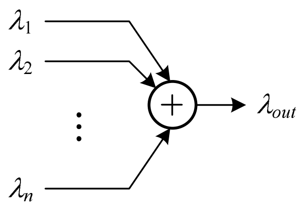
Zadatak 2.14 U mreži sa slike, poznati su protoci Poissonovih dolazaka korisnika $\lambda_1 = 2\ \mathrm{s}^{-1}$ i $\lambda_1 = 0{,}8\ \mathrm{s}^{-1}$. U tačkama u kojima dolazi do razdvajanja tokova, navedene su vrednosti koeficijenata (verovatnoća) rutiranja. Odredite protoke korisnika na izlazima mreže u stacionarnom stanju.



Slika 2.14: *Mreža servisnih sistema.*

Zadatak ćemo rešiti pozivanjem na neke osobine servisnih sistema.

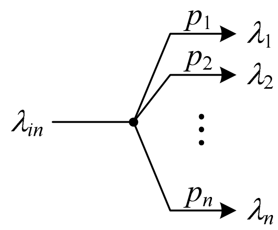
Združivanje Poissonovih procesa: Združivanjem n nezavisnih Poissonovih procesa čiji su protoci $\lambda_1, \ldots, \lambda_n$, dobija se Poissonov proces protoka $\lambda_{out} = \sum_{i=1}^{n} \lambda_i$.



Združivanje Poissonovih procesa.

Razdvajanje Poissonovog procesa: Razdvajanjem Poissonovog procesa protoka λ_{in} na n procesa, tako da su verovatnoće rutiranja $p_1, \ldots, p_n$, pri čemu je $\sum_{i=1}^{n} p_i = 1$, dobija se n Poissonovih procesa, čiji su protoci redom $p_1\lambda_{in}, \ldots, p_n\lambda_{in}$.

Burkeova teorema: U stacionarnom stanju, protok izlazaka korisnika iz servisnog sistema M/M/m/∞ takođe je Poissonov.



Razdvajanje Poissonovog procesa.

Princip konzervacije protoka: Ako se u sistemu ne generišu i ne nestaju korisnici, ukupan protok dolazaka korisnika u sistem jednak je ukupnom protoku izlazaka korisnika iz sistema.

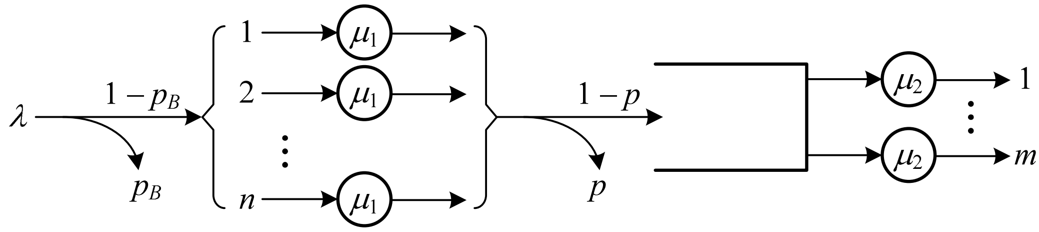
Iz Burkeove teoreme sledi da su u stacionarnom stanju svi tokovi u posmatranoj mreži Poissonovi. Protok dolazaka korisnika u prvi sistem je λ_1, pa je, zbog konzervacije protoka, toliki i protok na njegovom izlazu. Zbog osobina razdvajanja i združivanja Poissonovih tokova, ulazni protok u drugi sistem će biti $0{,}4\lambda_1 + \lambda_2$, a u treći $0{,}4\lambda_1$; toliki će redom biti i protoci izlazaka korisnika iz ovih sistema. Ulazni protok u četvrti sistem je $(0{,}4\lambda_1 + \lambda_2) + 0{,}2\lambda_1 + 0{,}6 \cdot 0{,}4\lambda_1$. Konačno dobijamo da je

$$\gamma_x = 0{,}84\lambda_1 + \lambda_2 = 2{,}48 \text{ s}^{-1}$$

i

$$\gamma_y = 0{,}16\lambda_1 = 0{,}32 \text{ s}^{-1}.$$

Zadatak 2.15 Na slici je data pojednostavljena blok-shema pozivnog centra banke. Tokom sata vršnog opterećenja, centar primi 70 poziva po n telefonskih linija. Zahteva se da se manje od $p_B = 1\%$ dolaznih poziva odbaci zbog zauzeća linija. Na početku svakog poziva, korisnici odslušaju poruku s govornog automata; $p = 10\%$ korisnika ovako će dobiti tražene informacije i u proseku će posle 30 s okončati poziv. Preostali korisnici se upućuju na prvog slobodnog od m raspoloživih operatera, pri čemu trajanje čekanja na operatera u 90% slučajeva treba biti kraće od 10 s. Prosečno trajanje razgovora s operaterom iznosi 35 s. Odredite potreban broj telefonskih linija, n i operatera, m, tako da se ispune ovi zahtevi.



Slika 2.15: *Ekvivalentna blok-shema pozivnog centra.*

Protok dolazaka poziva je

$$\lambda = \frac{70}{3600 \text{ s}} = 0{,}01944 \text{ s}^{-1}.$$

Protok poziva koji se prosleđuju operaterima je

$$\lambda_o = \lambda(1 - p_B)(1 - p) = 0{,}017325\ \mathrm{s}^{-1}.$$

Prema rezultatu zadatka 2.13, funkcija raspodele trajanja čekanja u servisnom sistemu M/M/m je

$$F_{T_Q}(t) = 1 - \frac{m^m}{m!}\frac{\rho^m}{1-\rho}p_0 e^{-\mu(1-\rho)t}, \quad t \geqslant 0,$$

pri čemu je

$$\rho = \frac{\lambda_o}{m\mu_2}$$

i

$$p_0 = \frac{1}{\displaystyle\sum_{n=0}^{m-1}\frac{(m\rho)^n}{n!} + \sum_{n=m}^{\infty}\frac{m^m\rho^n}{m!}},$$

što smo izveli u zadatku 2.5. Probom (pomoću računara) dobijamo da je za dva operatera $F_{T_Q}(10\ \mathrm{s}) = 0{,}8844$, dok je za tri operatera $F_{T_Q}(10\ \mathrm{s}) = 0{,}9798$. Zaključujemo da su potrebna $m = 3$ operatera.

Da bismo odredili potreban broj ulaznih telefonskih linija, odredimo najpre prosečno trajanje čekanja korisnika koji posle odslušane poruke s govornog automata žele razgovarati s operaterom. Još jednom ćemo se podsetiti rezultata zadatka 2.13 i dobiti

$$T_Q = \int_0^\infty t f_{T_Q}(t)\,\mathrm{d}t = \frac{m^m}{m!}\frac{\rho^m}{(1-\rho)^2}\frac{p_0}{\mu_2}.$$

Za $m = 3$, dobijamo

$$T_Q = \frac{27}{6}\frac{\rho^3}{(1-\rho)^2}\frac{p_0}{\mu_2},$$

pri čemu je sada $\rho = 0{,}202125$ i $p_0 = 0{,}544427$. Izračunavanjem dobijamo da je $T_Q = 1{,}112271$ s. To dalje znači da je sa stanovišta ulaznih linija prosečno trajanje poziva

$$\bar{\tau}_1 = p \cdot 30\ \mathrm{s} + (1-p)(T_Q + 35\ \mathrm{s}) = 35{,}501044\ \mathrm{s}.$$

Ponuđeni saobraćaj na ulaznim linijama je

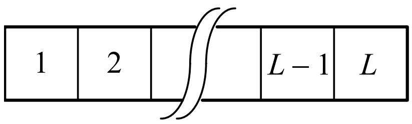
$$A = \lambda\bar{\tau}_1 = 0{,}690230\ \mathrm{E}.$$

Probom iz Erlangove B formule, dobijamo da veovatnoća blokade za tri ulazne linije iznosi 0,028, a za četiri 0,005. Potreban broj ulaznih telefonskih linija stoga je $n = 4$.

3. Mreže s komutacijom paketa

Zadatak 3.1 Okviri dužine L prenose se linijom na kojoj je verovatnoća greške p. Odredite verovatnoću pogrešnog prijema okvira.

Svaki okvir se sastoji od L bita. Greške na bitima okvira su nezavisni događaji, koji se realizuju s pojedinačnom verovatnoćom p.



Slika 3.1: *Struktura okvira.*

Definišimo događaje:
X — okvir je primljen ispravno i
Y — okvir je primljen pogrešno.

Očigledno, X i Y su komplementarni događaji. Događaj X se realizuje ako i samo ako se svi biti okvira prime ispravno. Verovatnoća ispravnog prijema svakog pojedinačnog bita je $1 - p$, pa je

$$\mathrm{P(X)} = (1-p)^L.$$

Verovatnoća događaja Y sada je

$$\mathrm{P(Y)} = 1 - \mathrm{P(X)} = 1 - (1-p)^L.$$

Za verovatnoću pogrešnog prijema okvira koristi se i oznaka P_F.

Zadatak 3.2 Okviri F_1 i F_2, čije su dužine L_1 i L_2, respektivno, pri čemu je $L_1 > L_2$, simultano dolaze u čvor mreže, koji je pre toga bio slobodan. Čvor treba da prosledi ove okvire na liniju na kojoj je protok R; za to može primeniti tri strategije:

a) prvo šalje okvir F_1,

b) prvo šalje okvir F_2 ili

c) baca fer novčić da bi utvrdio koji će okvir prvo slati.

Analizirajte ove strategije sa stanovišta prosečnog zadržavanja okvira u čvoru.

Zadržavanje okvira odgovara vremenu koje on provede u čvoru, dok ga u potpunosti ne napusti.

a) Zadržavanje prvog okvira jednako je vremenu potrebnom da se njegovih L_1 bita prosledi na liniju i iznosi

$$D_1 = \frac{L_1}{R}.$$

Drugi okvir čeka da se pošalje prvi, pa da se potom i njegovih L_2 bita prosledi na liniju. Stoga je

$$D_2 = D_1 + \frac{L_2}{R}.$$

Ukupno zadržavanje ovih dvaju okvira je

$$D = D_1 + D_2,$$

a prosečno

$$\overline{D}_a = \frac{D_1 + D_2}{2} = \frac{L_1 + 0{,}5\,L_2}{R}.$$

b) Primenjujući sličan rezon, zaključujemo da je sada

$$\overline{D}_b = \frac{0{,}5\,L_1 + L_2}{R}.$$

c) Ovaj scenario možemo shvatiti kao kombinaciju scenarija a) i b), koji se realizuju s podjednakim verovatnoćama. Stoga je

$$\overline{D}_c = \frac{\overline{D}_a + \overline{D}_b}{2} = 0{,}75\,\frac{L_1 + L_2}{R}.$$

Pošto je prvi okvir duži, jednostavnom proverom utvrđujemo da važi

$$\overline{D}_b < \overline{D}_c < \overline{D}_a,$$

pa, da bi se minimiziralo prosečno zadržavanje okvira, prvo treba slati kraći okvir.

Zadatak 3.3 Okviri F_1 i F_2, čije su dužine L_1 i L_2, respektivno, pri čemu je $L_1 > L_2$, simultano dolaze u čvor mreže, koji je pre toga bio slobodan. Čvor treba da prosledi ove okvire na liniju na kojoj je protok R. Verovatnoća da će se posmatrani okvir prvi proslediti na liniju obrnuto je srazmerna njegovoj dužini. Odredite prosečno zadržavanje okvira u čvoru.

Okvir F_1 šalje se prvi s verovatnoćom $P_1 = \alpha/L_1$, a okvir F_2 s verovatnoćom $P_2 = \alpha/L_2$, gde je α konstanta čiju vrednost treba odrediti. Kako je $P_1 + P_2 = 1$, dobijamo da je

$$\alpha = \frac{L_1 L_2}{L_1 + L_2},$$

pa je

$$P_1 = \frac{L_2}{L_1 + L_2}$$

i

$$P_2 = \frac{L_1}{L_1 + L_2}.$$

Rezonujući kao u prethodnom zadatku, dobijamo da je prosečno zadržavanje okvira u čvoru

$$\begin{aligned}\overline{D} &= \frac{P_1}{R}\left(L_1 + \frac{L_2}{2}\right) + \frac{P_2}{R}\left(\frac{L_1}{2} + L_2\right) = \\ &= \frac{(L_1 + L_2)^2 + 2L_1L_2}{2R\,(L_1 + L_2)}.\end{aligned}$$

Zadatak 3.4 Ispitajte kako dužina paketa utiče na efikasnost prenosa poruke, definisanu kao odnos broja informacionih bita i ukupnog broja bita koji se šalju na liniju. Dužina poruke (u bitima) je X, I je broj informacionih bita u paketu, a H broj bita u zaglavlju.

a) Izvedite izraz za efikasnost prenosa ako su paketi fiksne dužine.

b) Ako je dužina paketa promenljiva, tada se u zaglavlju pored H bita nalazi i H_v bita koji sadrže informaciju o dužini paketa. Izvedite izraz za efikasnost u ovom slučaju.

Slika 3.4: *Struktura paketa.*

a) Za prenos X bita poruke potrebno je $N = \lceil X/I \rceil$ paketa. Svaki paket se sastoji od $H + I$ bita, pa je efikasnost prenosa

$$E = \frac{X}{\left\lceil \frac{X}{I} \right\rceil (H + I)}.$$

Maksimalna efikasnost se postiže onda kada je količnik X/I prirodan broj i data je izrazom

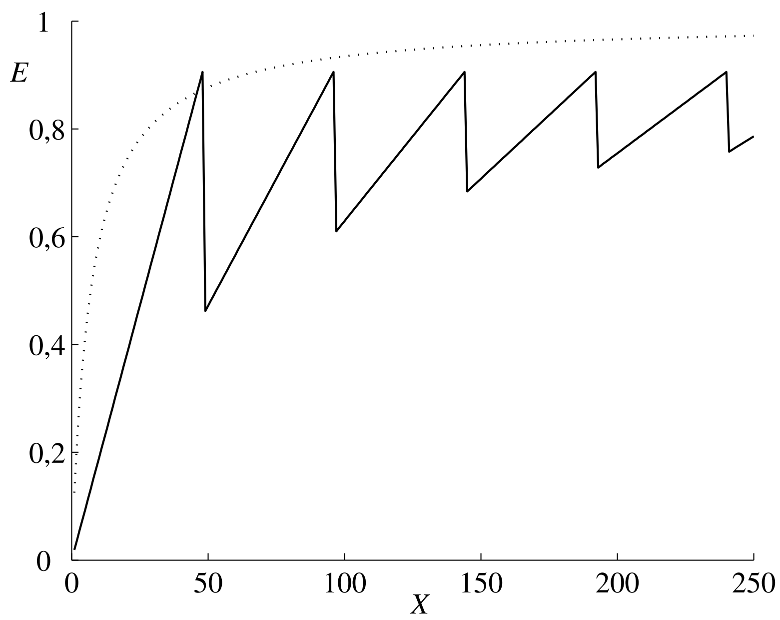
$$E_{max} = \frac{X}{\left(\frac{X}{I}\right)(H + I)} = \frac{I}{H + I}.$$

b) Najveća efikasnost će se ostvariti ako se svih X bita poruke prenese jednim paketom. Tada je $X = I$, pa je efikasnost

$$E = \frac{X}{X + H + H_v}.$$

Primetimo da je ovaj slučaj nepogodan za praktičnu primenu, jer bi smeštanje celokupne poruke u samo jedan paket značilo da je takav paket velike dužine, pa je i podložniji greškama u prenosu.

Zavisnost efikasnosti prenosa od dužine poruke, X, prikazana je na narednoj slici. Puna linija odgovara paketima fiksne dužine (za slučaj $H = 5$ B i $I = 48$ B – ATM ćelije), a isprekidana paketima promenljive dužine (uz $H_v = 2$ B). Maksimalna efikasnost koja se ostvaruje u tom slučaju je 91%.



Efikasnost prenosa za pakete fiksne i promenljive dužine.

Zadatak 3.5 Poruka dužine $L = 3200$ b prenosi se paketima fiksne dužine $F = 1024$ b. Zaglavlje paketa sastoji se od $H = 16$ bita. Između izvorišnog i odredišnog čvora, nalaze se i tri tranzitna. Propagaciono kašnjenje između svaka dva čvora je $t_p = 1$ ms. Protoci na svim linkovima su $V = 9600$ b/s. Odredite kašnjenje pri prenosu ove poruke, ako je primenjen prenos na principu

a) datagrama ili

b) virtuelnih kola.

Vreme uspostave virtuelnog kola je $t_s = 0{,}2$ s. Pretpostavite da se ne koriste poruke za potvrđivanje, kao i da se kašnjenje usled obrade paketa u čvorovima može zanemariti.

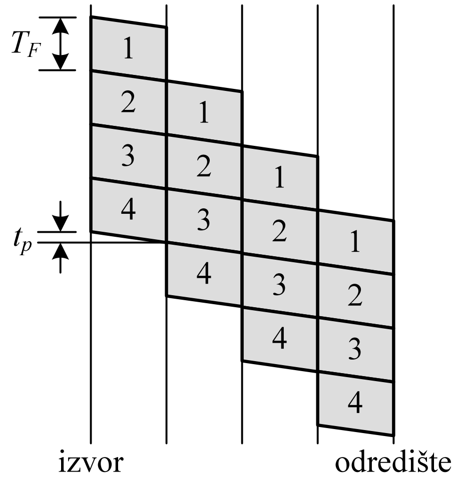
a) Korisni deo paketa čini $L - H$ bita. Za prenos poruke dužine L, trebaće

$$N = \left\lceil \frac{L}{F-H} \right\rceil = 4$$

paketa, čije je trajanje po

$$T_F = \frac{F}{V} = 107 \text{ ms.}$$

Ukupno kašnjenje jednako je zbiru vremena potrebnih za prenos poruke duž četiriju linkova, prema narednoj slici.



Slika 3.5: *Prenos poruke datagramima.*

Primetimo da tranzitni čvorovi ne čekaju da prime kompletnu poruku da bi počeli s prosleđivanjem paketa, već pakete prosleđuju čim ih prime.

Vreme potrebno za prenos poruke na prvom linku (između izvorišnog i prvog tranzitnog čvora) je

$$T_1 = 4T_F + t_p = 428 \text{ ms.}$$

Na ostalim linkovima putanje, vremena potrebna za prenos će biti jednaka i iznosiće

$$T_2 = T_3 = T_4 = T_F + t_p = 108 \text{ ms.}$$

Ukupno kašnjenje stoga će iznositi

$$T_a = T_1 + T_2 + T_3 + T_4 = 752 \text{ ms.}$$

b) Ako se uspostavlja virtuelno kolo, kašnjenje iz prethodne tačke će se uvećati za vreme uspostave kola. Ukupno kašnjenje će biti

$$T_b = t_s + T_a = 952 \text{ ms.}$$

Po pitanju kašnjenja s kraja na kraj, prednost je na strani prenosa na principu datagrama.

Zadatak 3.6 Poruka velike dužine L prenosi se od izvorišnog do odredišnog čvora preko dvaju paketskih komutatora i triju linkova. Poruka se u izvorišnom čvoru segmentira u pakete koji sadrže $S \ll L$ korisnih bita i H bita zaglavlja. Odredite optimalnu veličinu segmenta poruke, S, tako da za njen prenos do odredišta bude potrebno minimalno vreme. Protoci na linkovima su jednaki i iznose R, a trajanja obrade u komutatorima, čekanja u baferima i propagaciona kašnjenja na linkovima mogu se zanemariti.

Broj segmenata je

$$N = \left\lceil \frac{L}{S} \right\rceil \approx \frac{L}{S},$$

jer je $L \gg S$. Rezonujući kao u prethodnom zadatku, dobijamo da je za prenos poruke potrebno vreme

$$T = \frac{H+S}{R}\left(\frac{L}{S}+2\right).$$

Optimalnu vrednost S odredićemo izjednačavanjem izvoda s nulom:

$$\frac{\mathrm{d}T}{\mathrm{d}S} = \frac{1}{R}\left(\frac{L}{S}+2\right) - \frac{H+S}{R}\frac{L}{S^2} = 0,$$

odakle dobijamo

$$S = \sqrt{\frac{HL}{2}}.$$

Zadatak 3.7 Poruka dužine L prenosi se po principu komutacije paketa od izvorišnog do odredišnog čvora preko k tranzitnih. Protoci na linkovima su jednaki i iznose R, dok propagaciona kašnjenja iznose t. Trajanja obrade u komutatorima i čekanja u baferima mogu se zanemariti. Odredite optimalan format paketa – koji se sastoje od zaglavlja fiksne dužine H i polja korisnog sadržaja dužine S – tako da kašnjenje pri prenosu poruke bude minimalno. Koliko ono iznosi?

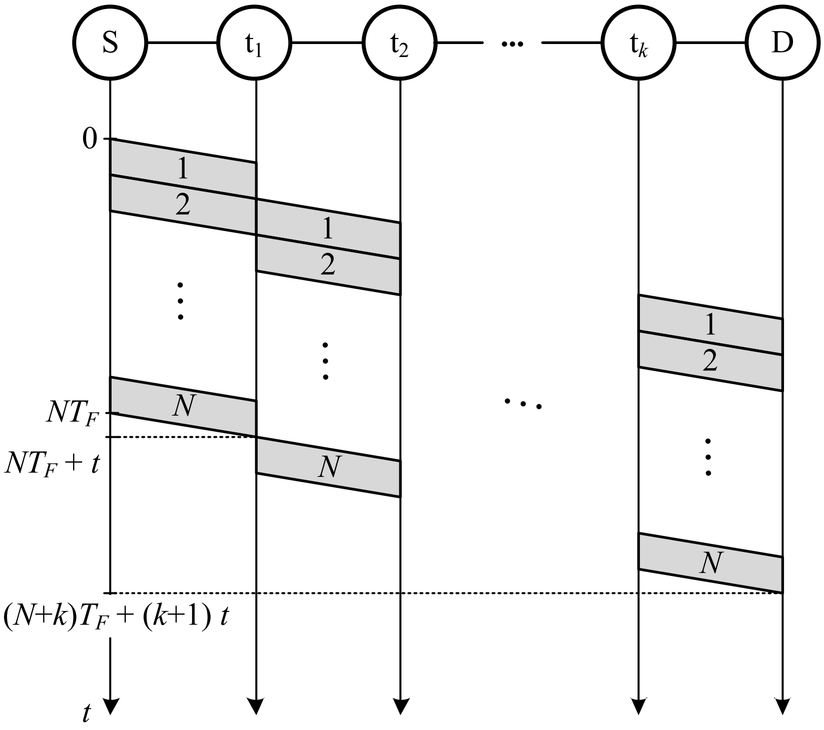
Po principu komutacije paketa, poruka se deli na N paketa, dužine $H+S$; pošto je H fiksno, treba odrediti S tako da se poruka prenese za najkraće vreme. Prema zaključku zadatka 3.5, pri prenosu treba koristiti datagrame.

Potreban broj paketa za predstavljanje poruke dat je izrazom

$$N = \left\lceil \frac{L}{S} \right\rceil \approx \frac{L}{S},$$

dok je trajanje jednog paketa

$$T_F = \frac{H+S}{R}.$$



Slika 3.7: *Vremenski dijagram prenosa poruke.*

Prema dijagramu sa slike, trajanje prenosa je

$$T = \left(\frac{L}{S} + k\right)\frac{H+S}{R} + (k+1)t.$$

Optimalnu veličinu polja korisnog sadržaja u paketu odredićemo izjednačavanjem izvoda s nulom:

$$\frac{\partial T_F}{\partial S} = \left(\frac{L}{S} + k\right)\frac{1}{R} + \left(-\frac{L}{S^2}\right)\frac{H+S}{R}\bigg|_{S=S_{opt}} = 0,$$

odakle dobijamo

$$S_{opt} = \sqrt{\frac{HL}{k}}.$$

Primetimo da je ovo generalizacija rezultata zadatka 3.6.

Minimalno trajanje prenosa sada je

$$T_{min} = \left(\sqrt{\frac{kL}{H}} + k\right)\frac{H + \sqrt{\frac{HL}{k}}}{R} + (k+1)t.$$

Zadatak 3.8 Odredite vreme potrebno za prenos $L = 1$ MB podataka između dveju stanica udaljenih $d = 1$ km u mreži s topologijom

a) magistrale, ili

b) prstena, obima $2d$, s $R = 10$ obnavljača, od kojih svaki unosi kašnjenje jednako trajanju jednog bitskog intervala.

U oba slučaja, protok na linku je $V = 1$ Mb/s, a brzina propagacije signala $v = 200000$ km/s. Dužina okvira je $F = 256$ b, s $H = 80$ b zaglavlja. Kada je primenjena topologija magistrale, pre slanja narednog okvira potrebno je da se tekući potvrdi okvirom dužine $A = 88$ b. Za topologiju prstena, prijem okvira se potvrđuje tako što odredišna stanica pušta da se okvir vrati izvorišnoj, pri čemu mu dodaje jedan bit potvrde.

Korisni deo okvira čini $F - H$ bita. Broj okvira potrebnih za prenos poruke dužine L je

$$N = \left\lceil \frac{L}{F-H} \right\rceil = 47663.$$

a) Prenos na magistrali sastoji se od niza ciklusa prenosa okvira s korisnim podacima i potvrda. Trajanje jednog ciklusa je

$$T_{bus} = \frac{F}{V} + \frac{A}{V} + 2\frac{d}{v} = 3{,}54 \cdot 10^{-4} \text{ s.}$$

Ukupno kašnjenje biće

$$T_a = NT_{bus} = 16{,}87 \text{ s.}$$

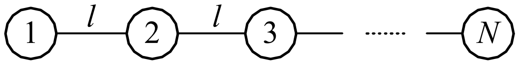
b) Obnavljači unose kašnjenje R/V. Trajanje ciklusa stoga je

$$T_{ring} = \frac{F}{V} + \frac{F+1}{V} + \frac{R}{V} + 2\frac{d}{v} = 5{,}33 \cdot 10^{-4} \text{ s,}$$

pa je ukupno kašnjenje

$$T_b = NT_{ring} = 25{,}4 \text{ s.}$$

Zadatak 3.9 U mreži s topologijom lanca (*daisy chain*), razmenjuju se okviri dužine $L = 70$ B. Mrežu čini $N = 10$ ekvidistantnih stanica, spojenih vodovima dužine $l = 500$ m, na kojima je brzina propagacije signala $v = \frac{2}{3}c$. Protok na linku je $R = 64$ kb/s, a trajanje obrade okvira $\tau = 15$ µs. Koliko iznosi prosečno vreme potrebno za prenos okvira u ovoj mreži?



Slika 3.9: *Mreža s topologijom lanca.*

U razmatranoj mreži, postoji $N-1$ mogućnost da komuniciraju susedne stanice, koje su na rastojanju l, za šta je potrebno vreme

$$t_1 = \frac{L}{R} + \frac{l}{v} = 8{,}7525 \text{ ms}.$$

Dalje, postoje $N-2$ mogućnosti da komuniciraju stanice na rastojanju $2l$, kada prenos okvira traje

$$t_2 = 2\frac{L}{R} + 2\frac{l}{v} + \tau = 2t_1 + \tau.$$

Nastavljanjem postupka, vidimo da će biti dve mogućnosti za komunikaciju stanica na rastojanju $(N-2)l$, čemu odgovara vreme

$$t_{N-2} = (N-2)t_1 + (N-3)\tau,$$

kao i jedna mogućnost za komunikaciju krajnjih stanica, koje su na rastojanju $(N-1)l$ i za šta je potrebno vreme

$$t_{N-1} = (N-1)t_1 + (N-2)\tau.$$

U opštem slučaju, prenos okvira do stanice koja je udaljena $i \cdot l$, $i = 1, \ldots, N-1$, trajaće

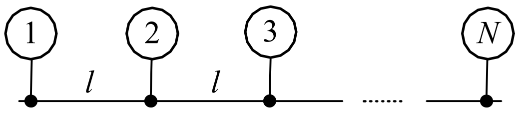
$$t_i = it_1 + (i-1)\tau,$$

dok će postojati $N-i$ mogućnosti za realizaciju ovoga scenarija.

Ukupan broj mogućnosti za komuniciranje je $N_{tot} = 1 + 2 + \cdots + N - 1 = \frac{N(N-1)}{2}$, pa je prosečno trajanje prenosa okvira

$$\begin{aligned}\bar{t} &= \frac{1}{N_{tot}} \sum_{i=1}^{N-1} (N-i)\,(it_1 + (i-1)\tau) = \\ &= \frac{N}{3}\,((N+1)t_1 + (N-2)\tau) = 32{,}133 \text{ ms}.\end{aligned}$$

Zadatak 3.10 U mreži s topologijom magistrale, koju čini osam stanica, razmenjuju se okviri dužine 120 B. Stanice su ekvidistantne, pri čemu je dužina segmenta kabla 200 m, dok je protok na linku 64 kb/s. Koliko iznosi prosečno vreme potrebno za prenos okvira u ovoj mreži?



Slika 3.10: *Mreža s topologijom magistrale.*

Uočimo da stanice u mreži s topologijom magistrale komuniciraju neposredno, tj. bez tranzitnih čvorova. Ukoliko se između stanica koje komuniciraju nalazi i segmenata kabla, $i \in \{1, 2, \ldots, N-1\}$, vreme potrebno za prenos biće

$$t_i = T_F + i t_p,$$

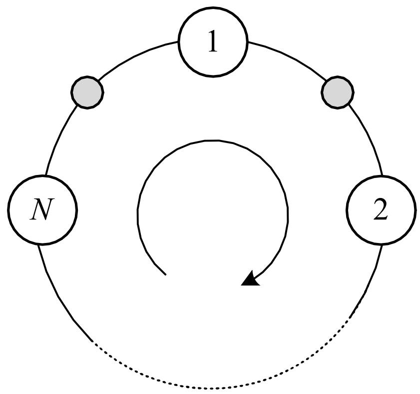
gde je $T_F = 15$ ms trajanje okvira i $t_p = 1$ µs propagaciono kašnjenje na jednom segmentu kabla (brzina propagacije na kablovskim vodovima je $\frac{2}{3}c$). Ovaj scenario realizuje se na $N - i$ načina, pri čemu je ukupan broj mogućnosti za komunikaciju

$$N_{tot} = \sum_{i=1}^{N-1} (N-i) = \frac{N(N-1)}{2} = 28.$$

Prosečno trajanje prenosa sada je

$$\begin{aligned} \bar{t} &= \frac{1}{N_{tot}} \sum_{i=1}^{N-1} (N-i) t_i = \\ &= T_F + \frac{N+1}{3} t_p = 15{,}003 \text{ ms.} \end{aligned}$$

Zadatak 3.11 U unidirekcionoj mreži s topologijom prstena, razmenjuju se okviri dužine $L = 70$ B. Mrežu čini $N = 8$ ekvidistantnih stanica, pri čemu se između svakih dveju stanica nalazi obnavljač. Obim prstena je $d = 2$ km, brzina propagacije signala na linku $v = \frac{2}{3}c$, a protok $R = 64$ kb/s. Trajanje obrade okvira je $\tau = 15$ µs, dok svaki obnavljač unosi kašnjenje od dvaju bitskih intervala. Koliko je prosečno vreme potrebno za prenos okvira u ovoj mreži?



Slika 3.11: *Unidirekciona prstenasta mreža.*

Vreme potrebno za prenos okvira između susednih stanica, na rastojanju d/N, dato je izrazom

$$t_1 = \frac{L}{R} + \frac{d/N}{v} + \frac{2}{R} = 8{,}7825 \text{ ms.}$$

U opštem slučaju, ukoliko su stanice na rastojanju $i \cdot d/N$, $i = 1, 2, \ldots, N-1$, prenos će trajati

$$t_i = it_1 + (i-1)\tau,$$

a ovaj će se scenario realizovati na N načina. Ukupan broj mogućnosti za komunikaciju je

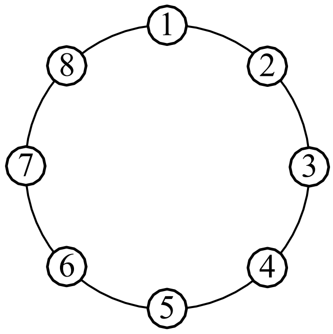
$$N_{tot} = \sum_{i=1}^{N-1} N = N(N-1) = 56.$$

Prosečno trajanje prenosa prema tome je dato izrazom

$$\bar{t} = \frac{1}{N_{tot}} \sum_{i=1}^{N-1} N\left(it_1 + (i-1)\tau\right) = \frac{N}{2}\, t_1 + \frac{N-2}{2}\, \tau$$

i iznosi 35,175 ms.

Zadatak 3.12 U bidirekcionoj mreži s topologijom prstena, razmenjuju se okviri dužine $L = 70$ B. Mrežu čini $N = 8$ ekvidistantnih stanica, pri čemu je obim prstena $d = 2$ km i protok na kablovskom linku $R = 64$ kb/s. Trajanje obrade okvira je $\tau = 15$ µs. Koliko je prosečno vreme potrebno za prenos okvira u ovoj mreži?



Slika 3.12: *Bidirekciona prstenasta mreža.*

U ovoj mreži postoje dve mogućnosti da komuniciraju susedne stanice, koje su na rastojanju d/N, za šta je potrebno vreme

$$t_1 = \frac{L}{R} + \frac{d/N}{v} = 8{,}75125 \text{ ms},$$

gde je $v = 2c/3$ brzina propagacije na kablu. Dalje, postoje još po dve mogućnosti za komunikaciju stanica na rastojanju $2d/N$ i $3d/N$, za šta su redom potrebna vremena

$$t_2 = 2t_1 + \tau$$

i

$$t_3 = 3t_1 + 2\tau,$$

kao i jedna mogućnost za komunikaciju na rastojanju $4d/N$, za šta treba

$$t_4 = 4t_1 + 3\tau.$$

Prosečno trajanje prenosa stoga je

$$\begin{aligned}\bar{t} &= \frac{1}{2+2+2+1}\left(2t_1 + 2t_2 + 2t_3 + t_4\right) = \\ &= \frac{1}{7}\left(16t_1 + 9\tau\right) = 20{,}022\ \mathrm{ms}.\end{aligned}$$

Zadatak 3.13 Izvedite izraze za prosečno kašnjenje duž zadate trase (rute) i u segmentu mreže s komutacijom paketa.

Posmatrajmo najpre telekomunikacioni link kapaciteta C [b/s], na kome je propagaciono kašnjenje τ. Pretpostavićemo da se linkom prenose paketi čiji dolasci odgovaraju Poissonovom slučajnom procesu, dok su im dužine eksponencijalno raspodeljene. Pretpostavićemo, takođe, da su trenuci dolazaka paketa nezavisni od trajanja njihove obrade, što je Kleinrockova pretpostavka nezavisnosti.

Paketi koje treba preneti linkom prvo se smeštaju u odlazni bafer, u kome čekaju na utiskivanje u link. Bafer je beskonačnog kapaciteta, pa se paketi ne odbacuju, tj. nema gubitaka saobraćaja. Prosečan protok dolazaka/odlazaka paketa stoga je R/L, gdje je R prosečan binarni protok na linku, a L prosečna dužina paketa. Prosečno trajanje obrade paketa je L/C. Na linku se prosečno nalazi $\tau/(L/R)$ utisnutih paketa koji propagiraju ka prijemnoj strani.

Na osnovu navedenog, odlazni bafer linka se, u odnosu na pakete, ponaša kao servisni sistem M/M/1, u koji korisnici dolaze s prosečnim protokom $\lambda = R/L$. Protok obrade je $\mu = C/L$.

Prosečan broj paketa koji se u stacionarnom stanju nalaze na linku je

$$N_l = \frac{R/L}{C/L - R/L} + \frac{\tau}{L/R}.$$

Po Littleovoj teoremi, prosečno kašnjenje paketa biće

$$T_l = \frac{1}{R/L} N_l = \frac{L}{C-R} + \tau.$$

Uočimo sada trasu ili rutu, koju čine sukcesivno povezani linkovi. Uz zanemarivanje trajanja obrade paketa u čvorovima, kašnjenje duž trase biće jednako zbiru kašnjenja na pojedinim linkovima koji čine tu trasu. Prosečno kašnjenje *paketa* stoga je

$$T_{tr} = \sum_{i \in Tr} \left(\frac{L}{C_i - R_i} + \tau_i \right),$$

dok će kašnjenje *po bitu* biti L puta manje:

$$T_{\substack{tr\\b}} = \sum_{i \in Tr} \left(\frac{1}{C_i - R_i} + \frac{\tau_i}{L} \right).$$

Prosečno kašnjenje paketa u mreži – ili jednom njenom segmentu – možemo odrediti usrednjavanjem po svim mogućim rutama, ili preko Littleove teoreme; potonji način daleko je elegantniji.

Prosečan broj paketa koji se u svakom trenutku nalaze u mreži jednak je zbiru prosečnih brojeva paketa koji se nalaze na svakom od njenih linkova, tj.

$$N = \sum_i N_{l,i} = \sum_i \left(\frac{R_i}{C_i - R_i} + \frac{\tau}{L/R_i} \right).$$

Ako je prosečan izlazni bitski protok iz uočenog segmenta mreže γ, odgovarajući protok paketa biće γ/L; prosečno kašnjenje paketa stoga je

$$T = \frac{1}{\gamma/L} N = \frac{L}{\gamma} \sum_i R_i \left(\frac{1}{C_i - R_i} + \frac{\tau}{L} \right).$$

Prosečno kašnjenje po bitu i ovdje će biti L puta manje i iznositi

$$T_b = \frac{1}{\gamma} \sum_i R_i \left(\frac{1}{C_i - R_i} + \frac{\tau}{L} \right).$$

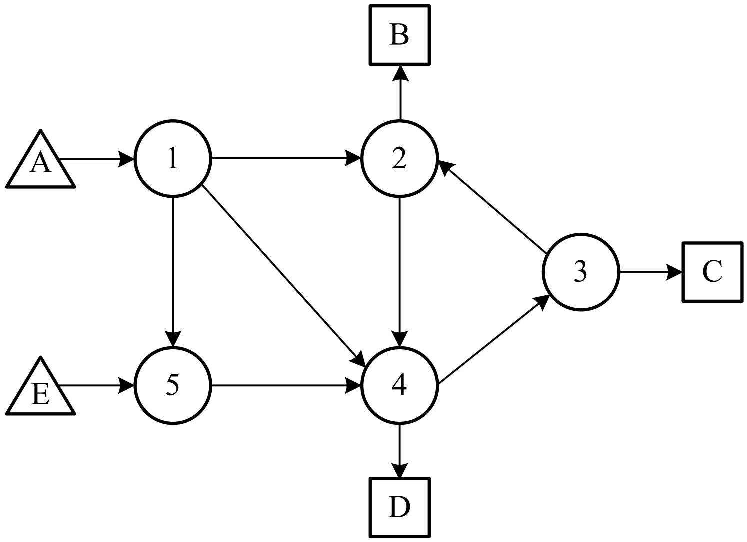
Zadatak 3.14 Na slici je dat graf telekomunikacione mreže. Kružićima su označeni njeni čvorovi, trouglićima izvori, a kvadratićima odredišta saobraćaja. Parametri mreže dati su u tabeli. Protoci i kapaciteti izraženi su u Mb/s.

link	protok	kapacitet	izvor	protok	odredište	protok
1–2	50	100	A	70	B	50
1–4	?	20	E	110	C	?
1–5	12	40			D	50
2–4	?	90				
3–2	35	50				
4–3	115	150				
5–4	?	150				

Modelirajući linkove servisnim sistemima M/M/1 i zanemarujući propagaciona kašnjenja, odredite:

a) prosečno kašnjenje u mreži i

b) prosečno kašnjenje na trasi 1–4–3–2.



Slika 3.14: *Graf telekomunikacione mreže.*

a) Linkove mreže ćemo modelirati servisnim sistemima M/M/1, a prosečno kašnjenje u mreži odredićemo primenom rezultata prethodnog zadatka. Ukupan protok koji izlazi iz mreže jednak je zbiru protoka ka odredištima B, C i D. S druge strane, zbog konzervacije protoka, on je jednak i ukupnom protoku koji u mrežu ulazi iz izvora A i E,

$$\gamma = V_B + V_C + V_D = V_A + V_E,$$

odakle dobijamo

$$\gamma = 180 \text{ Mb/s},$$

a takođe i

$$V_C = 80 \text{ Mb/s}.$$

Prosečno kašnjenje (po bitu) dato je izrazom

$$T = \frac{1}{\gamma} \sum_i \frac{V_i}{C_i - V_i}.$$

Da bismo ga odredili, potrebno je da prvo odredimo nepoznate protoke na linkovima 1–4, 2–4 i 5–4.

Iz zakona konzervacije protoka za čvor 1 je

$$V_{12} + V_{14} + V_{15} - V_A = 0,$$

odakle dobijamo $V_{14} = 8$ Mb/s.

Za čvor 2 važi

$$-V_{12} + V_{24} - V_{32} + V_B = 0,$$

pa je $V_{24} = 35$ Mb/s.

Konačno, za čvor 4 je

$$-V_{14} - V_{24} + V_{43} - V_{54} + V_D = 0,$$

pa je $V_{54} = 122$ Mb/s.

Uvrštavanjem izračunatih vrednosti, dobijamo da prosečno kašnjenje u posmatranoj mreži iznosi $7{,}06 \cdot 10^{-8}$ s/b.

b) Kašnjenje na trasi *tr* odredićemo kao zbir kašnjenja na pojedinim linkovima te trase:

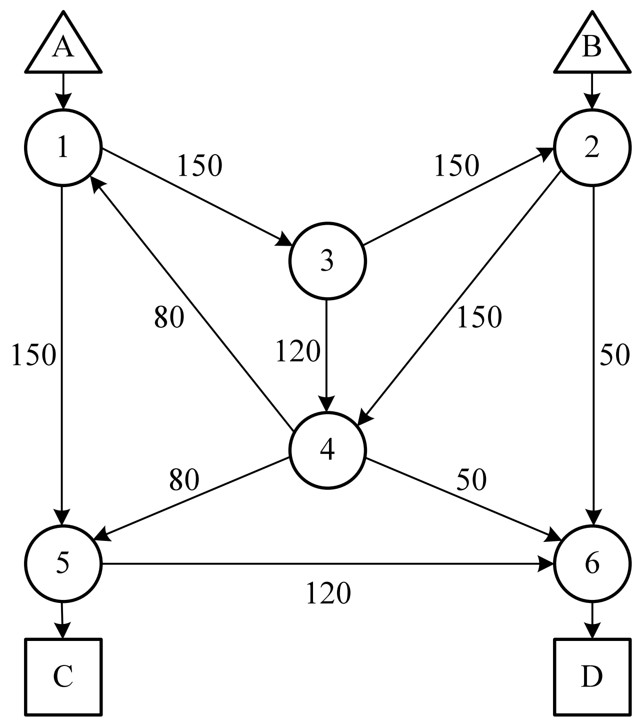
$$T_{tr} = \sum_{i \in tr} \frac{1}{C_i - V_i}.$$

Trasi 1–4–3–2 pripadaju linkovi 1–4, 4–3 i 3–2, pa će biti

$$T_{tr} = \frac{1}{C_{14} - V_{14}} + \frac{1}{C_{43} - V_{43}} + \frac{1}{C_{32} - V_{32}},$$

što iznosi $1{,}79 \cdot 10^{-7}$ s/b.

Zadatak 3.15 Na slici je dat graf telekomunikacione mreže s komutacijom paketa. Kružićima su označeni njeni čvorovi, trouglićima izvori, a kvadratićima odredišta saobraćaja. Uz svaki link, naveden je i njegov kapacitet u Mb/s.



Slika 3.15: *Graf telekomunikacione mreže.*

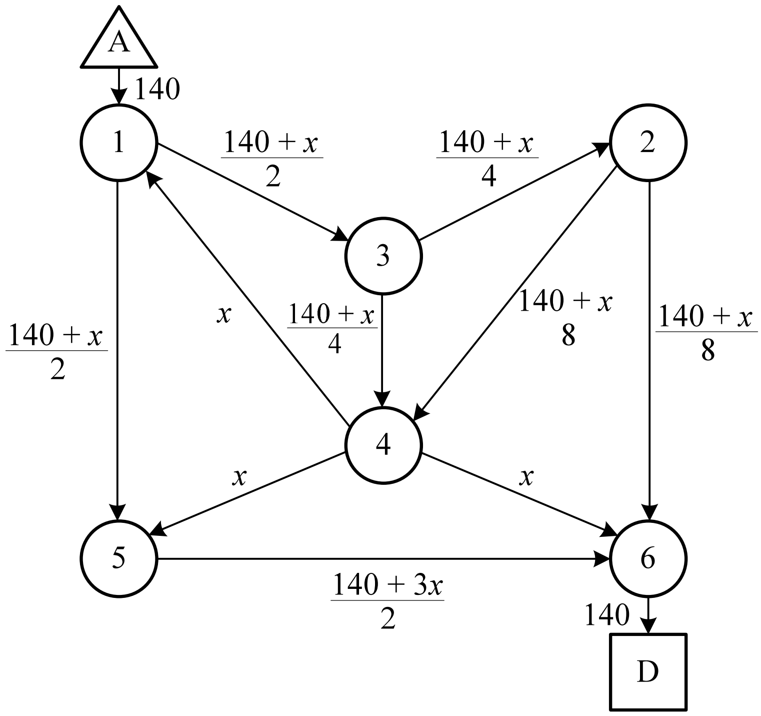
Izvor A emituje podatke ka odredištu D, s protokom 140 Mb/s. Izvor B emituje podatke ka odredištu C, s protokom 60 Mb/s. Rutiranje je takvo da se u svakom čvoru dolazni saobraćaj ravnomerno raspodeljuje po svim odlaznim linkovima koji vode do traženog odredišta. Primenom teorije sistema M/M/1, uz zanemarivanje propagacionih kašnjenja, odredite:

a) protoke na svim linkovima i

b) prosečno kašnjenje u mreži.

a) Protoke na linkovima odredićemo primenom svojstva superpozicije.

Posmatrajmo prvo mrežu kada su aktivni samo izvor A i odredište D. Primenjujući princip ravnomernog raspodeljivanja saobraćaja i zakon konzervacije protoka, dolazimo do protoka koji su označeni na slici *i*).



i) *Slučaj kada je aktivan izvor A.*

Za čvor 4 je

$$\frac{140+x}{4}+\frac{140+x}{8}=3x,$$

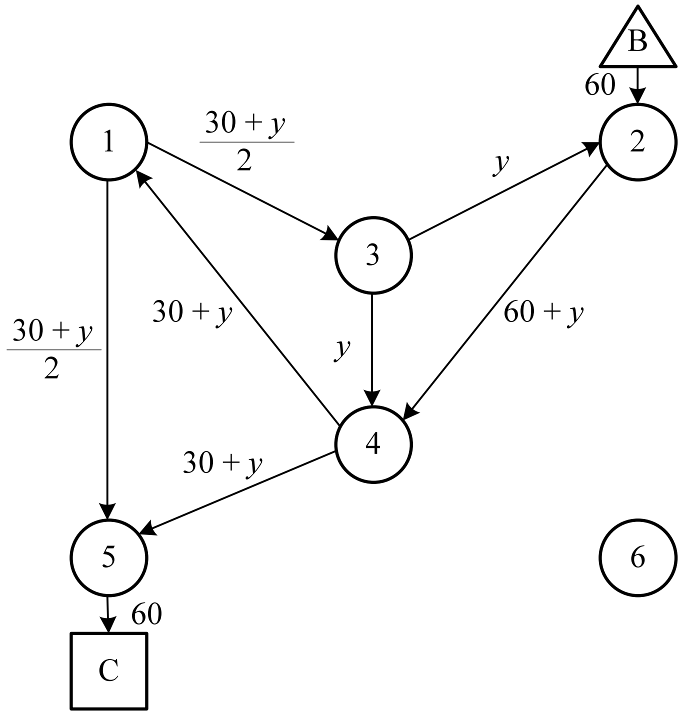
pa je $x = 20$ i možemo izračunati sve protoke.

Kada su aktivni samo izvor B i odredište C, imamo situaciju kao na slici *ii*), na narednoj strani.

Za čvor 5 važi

$$30+y+\frac{30+y}{2}=60,$$

pa je $y = 10$.



ii) Slučaj kada je aktivan izvor B.

Ukupne protoke na linkovima odredićemo sabiranjem ovih rešenja. Rezultati su prikazani u tabeli. Radi preglednosti, uz svaki link naveden je i njegov kapacitet.

Konačni rezultati:

link	1–3	1–5	2–4	2–6	3–2	3–4	4–1	4–5	4–6	5–6
protok [Mb/s]	100	100	90	20	50	50	60	60	20	100
kapacitet [Mb/s]	150	150	150	50	150	120	80	80	50	120

b) Prosečno kašnjenje u mreži dato je izrazom

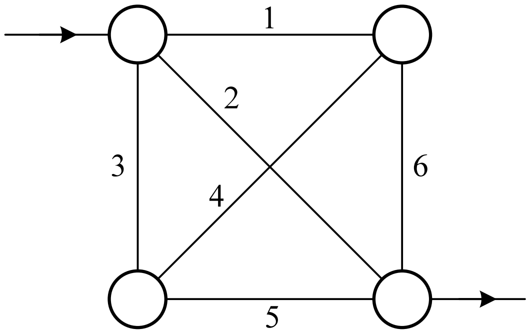
$$T = \frac{1}{\gamma} \sum_i \frac{V_i}{C_i - V_i},$$

gde je

$$\gamma = V_A + V_B = V_C + V_D = 200 \text{ Mb/s}.$$

Uvrštavanjem izračunatih vrednosti protoka, dobijamo da kašnjenje u posmatranoj mreži iznosi $9{,}52 \cdot 10^{-8}$ s/b.

Zadatak 3.16 Transportnom mrežom čiji je graf prikazan na slici, a kapaciteti linkova dati u tabeli, treba preneti signal protoka 1 Mb/s. Odredite optimalnu putanju, tako da kašnjenje pri prenosu bude minimalno.



Slika 3.16: *Graf transportne mreže.*

i	1	2	3	4	5	6
C_i [Mb/s]	2	1,5	3	4	1,5	3

Graf mreže je jednostavan, pa problem možemo rešiti direktnim pretraživanjem kombinatornog prostora.

U posmatranoj mreži postoji pet putanja od ulaznog do izlaznog čvora i to 1–6, 2, 3–5, 1–4–5 i 3–4–6. Kašnjenje na svakoj od njih odredićemo po obrascu

$$T_{tr} = \sum_{i \in tr} \frac{1}{C_i - V},$$

gde je $V = 1$ Mb/s. Izračunavanjem dobijamo rezultate koji su navedeni u tabeli.

Kašnjenja na putanjama:

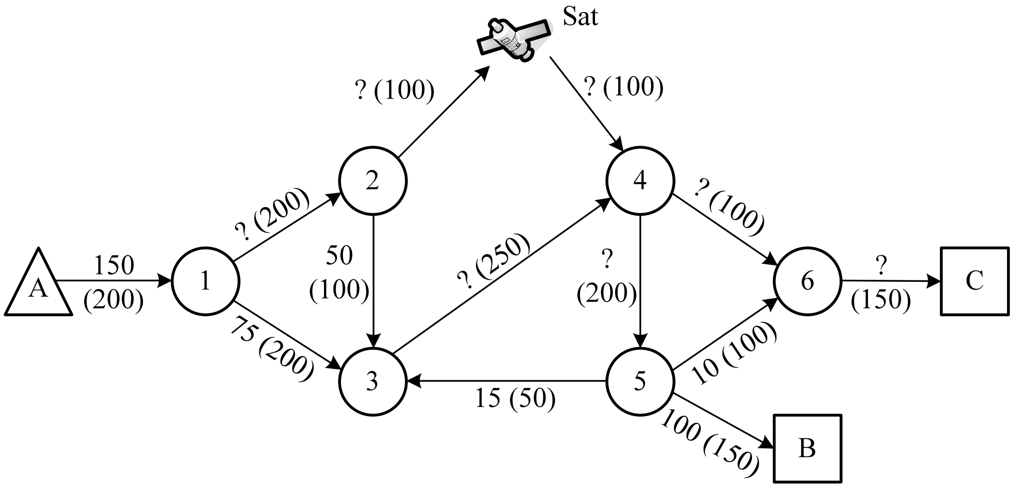
Putanja	1–6	2	3–5	1–4–5	3–4–6
T [µs/b]	1,5	2	2,5	3,33	1,33

Najkraće kašnjenje je na putanji 3–4–6 iako se ona sastoji od triju linkova. U praksi, pri određivanju optimalne putanje u obzir se uzimaju i vremena obrade paketa u tranzitnim čvorovima.

Zadatak 3.17 Na slici je dat graf telekomunikacione mreže. Kružićima su označeni njeni čvorovi, trouglićem izvor, a kvadratićima odredišta saobraćaja. Uz svaki link, označeni su protok i kapacitet (u zagradama), u Mb/s. Modelirajući linkove servisnim sistemima M/M/1, odredite prosečno kašnjenje pri prenosu okvira dužine $L = 1000$ b trasama A–1–2–Sat–4–6–C ili A–1–2–Sat–4–5–B. Rastojanje 2–Sat iznosi 35000 km, a Sat–4 40000 km. Propagaciona kašnjenja na zemaljskim linkovima mogu se zanemariti.

Počnimo rešavanje zadatka tako što ćemo odrediti nepoznate protoke na linkovima. Protok ka odredištu C nalazimo iz

$$V_A - V_B - V_C = 0,$$



Slika 3.17: *Graf telekomunikacione mreže.*

odakle je $V_C = 50$ Mb/s.

Za čvor 1 je

$$V_A - V_{12} - V_{13} = 0,$$

pa je $V_{12} = 75$ Mb/s.

Za čvor 2 je

$$V_{12} - V_{23} - V_{2Sat} = 0,$$

pa je $V_{2Sat} = 25$ MB/s.

Za satelit je

$$V_{2Sat} - V_{Sat4} = 0.$$

Stoga je $V_{Sat4} = 25$ Mb/s.

Jednačina konzervacije protoka za čvor 3 glasi

$$V_{13} + V_{23} - V_{34} + V_{53} = 0$$

i iz nje dobijamo $V_{34} = 140$ Mb/s.

Za čvor 5 je

$$V_{45} - V_{53} - V_{56} - V_B = 0,$$

odakle je $V_{45} = 125$ Mb/s.

Konačno, za čvor 6 važi

$$V_{46} + V_{56} - V_C = 0,$$

što znači da je $V_{46} = 40$ Mb/s.

Kašnjenje na trasi odredićemo kao zbir kašnjenja usled zadržavanja okvira u servisnim sistemima M/M/1 i propagacionog kašnjenja na linkovima koji pripadaju toj trasi:

$$T_{tr} = \sum_{i \in tr} \left(\frac{L}{C_i - V_i} + T_{prop,\, i} \right),$$

gde je L dužina okvira (u bitima). Po postavci zadatka, propagaciona kašnjenja na zemaljskim linkovima se zanemaruju, pa treba uračunati samo kašnjenja na satelitskim linkovima 2–Sat i Sat–3. Ako s $d_{2,\, Sat}$ i $d_{Sat,\, 3}$ budemo označili dužine satelitskih linkova, važiće

$$T_{tr} = \sum_{i \in tr} \frac{L}{C_i - V_i} + \frac{d_{2,\, Sat}}{c} + \frac{d_{Sat,\, 3}}{c},$$

gde je c brzina prostiranja svetlosti u vakuumu.

Na osnovu ovih razmatranja, dobijamo da kašnjenje pri prenosu okvira trasom A–1–
–2–Sat–4–5–B iznosi 0,250088 s, a trasom A–1–2–Sat–4–6–C 0,250081 s. U oba slučaja, dominantan je uticaj propagacionog kašnjenja na satelitskim linkovima.

Zadatak 3.18 Dva čvora u računarskoj mreži povezana su linkom čiji je kapacitet C i na kome je protok R.

a) Koliko puta treba povećati kapacitet linka, tako da se, uz nepromenjen protok, kašnjenje usled čekanja u njegovom baferu smanji k puta?

b) Za ovako određen kapacitet, odredite novi protok, tako da kašnjenje bude jednako kašnjenju pre povećanja kapaciteta.

Link ćemo modelirati servisnim sistemom M/M/1. Kašnjenje usled čekanja u baferu stoga će biti

$$T_Q = \frac{R}{C(C-R)}.$$

a) Rešavanjem po C, dobijamo kvadratnu jednačinu

$$C^2 - RC - \frac{R}{T_Q} = 0.$$

Njena rešenja su

$$C_{1,\,2} = \frac{R \pm \sqrt{R^2 + 4\dfrac{R}{T_Q}}}{2}.$$

Usvajamo rešenje sa znakom „+”, pa je

$$C = \frac{R + \sqrt{R^2 + 4R/T_Q}}{2}.$$

Da bi se ostvarilo kašnjenje $T'_Q = T_Q/k$, novi kapacitet treba da bude

$$C' = \frac{R + \sqrt{R^2 + 4R/T'_Q}}{2}.$$

Odavde je

$$\frac{C'}{C} = \frac{R + \sqrt{R^2 + 4R/T'_Q}}{R + \sqrt{R^2 + 4R/T_Q}}.$$

b) Sada treba da važi

$$T''_Q = \frac{R''}{C'(C' - R'')} = T_Q,$$

pa se dobija

$$R'' = \frac{C'^2 T_Q}{1 + C' T_Q}.$$

Zadatak 3.19 Posmatra se telekomunikaciona mreža s M spojnih puteva (linkova). Kapaciteti linkova su C_i, protoci R_i, a koeficijenti „cena” d_i. Odredite vrednosti kapaciteta linkova, tako da se:

a) za zadatu ukupnu cenu, D, ostvari minimalno kašnjenje u mreži, ili

b) za zadato ukupno kašnjenje, T, ostvari minimalna cena.

Ukupna cena linkova je

$$D = \sum_{i=1}^{M} d_i C_i.$$

Uz zanemarivanje propagacionog kašnjenja, ukupno kašnjenje u mreži odredićemo iz Littleove teoreme:

$$T = \frac{1}{\gamma} \sum_{i=1}^{M} \frac{R_i}{C_i - R_i},$$

gde je γ protok na izlazu/ulazu mreže.

a) Formirajmo tzv. funkciju cilja

$$G(C_1, \ldots, C_M) = \frac{1}{\gamma} \sum_{i=1}^{M} \frac{R_i}{C_i - R_i} + \beta \left(\sum_{i=1}^{M} d_i C_i - D \right),$$

gde je β *Lagrangeov multiplikator*. Odredimo njen izvod po C_i:

$$\frac{\partial G}{\partial C_i} = -\frac{1}{\gamma} \frac{R_i}{(C_i - R_i)^2} + \beta d_i.$$

Izjednačavanjem s nulom, dobijamo

$$C_i = R_i \pm \frac{1}{\sqrt{\beta\gamma}}\sqrt{\frac{R_i}{d_i}}.$$

Da bi pretpostavljeni servisni sistem M/M/1 bio stabilan, iskorišćenost njegovog servera mora biti manja od jedan, pa stoga usvajamo rešenje sa znakom „+", za koje je $C_i > R_i$. Kako je $D = \sum_{i=1}^{M} d_i C_i$, biće

$$\frac{1}{\sqrt{\beta\gamma}} = \frac{D - \sum_{j=1}^{M} d_j R_j}{\sum_{j=1}^{M} \sqrt{d_j R_j}}.$$

Uvrštavanjem u izraz za C_i, konačno dobijamo

$$C_i = R_i + \sqrt{\frac{R_i}{d_i}}\;\frac{D - \sum_{j=1}^{M} d_j R_j}{\sum_{j=1}^{M} \sqrt{d_j R_j}}.$$

b) Ciljna funkcija sada ima oblik

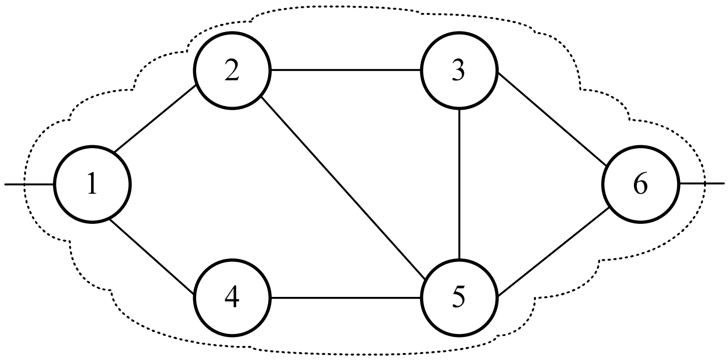
$$G(C_1, \ldots, C_M) = \sum_{i=1}^{M} d_i C_i + \beta\left(\frac{1}{\gamma}\sum_{i=1}^{M}\frac{R_i}{C_i - R_i} - T\right).$$

Primenom sličnog postupka kao u prethodnoj tački, dobijamo

$$C_i = R_i + \frac{1}{\gamma T}\sqrt{\frac{R_i}{d_i}}\sum_{j=1}^{M}\sqrt{d_j R_j}.$$

Zadatak 3.20 U mreži sa slike, normalizovani kapaciteti svih linkova iznose 1, a normalizovani protok od čvora 1 ka čvoru 6 je 1. Primenjen je sledeći algoritam rutiranja: ukoliko između dolaznog i odredišnog čvora postoji direktan link, sav dolazni saobraćaj se prosleđuje po njemu; u suprotnom, dolazni saobraćaj se ravnomerno prosleđuje po svim odlaznim linkovima.

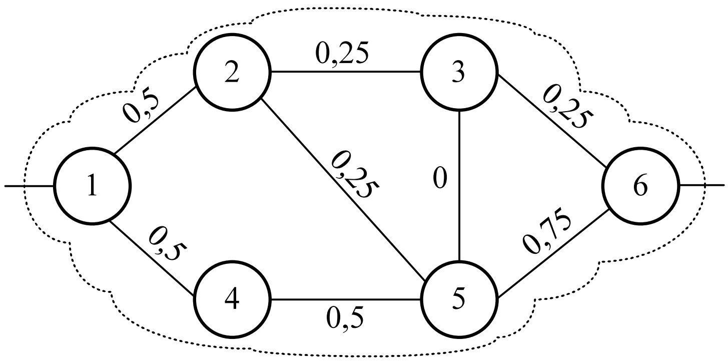
a) Odredite prosečno kašnjenje pri prenosu paketa normalizovane dužine $L = 10^{-3}$ u ovoj mreži.



Slika 3.20: *Graf telekomunikacione mreže.*

b) Odredite optimalne kapacitete linkova, tako da se u mreži ostvari minimalno kašnjenje, a da se ne promeni ukupan kapacitet svih linkova. Koliko iznosi to minimalno kašnjenje za pakete iz tačke a)?

Počnimo rešavanje zadatka tako što ćemo odrediti normalizovane protoke po svim linkovima u mreži, R_i, $i = 1, \ldots, 8$. Primenom usvojenog algoritma rutiranja, dobijamo rezultate koji su prikazani na donjoj slici.



Normalizovani protoci na linkovima.

a) Kašnjenje pri prenosu paketa dužine L biće

$$T = \frac{L}{R_{in}} \sum_i \frac{R_i}{C_i - R_i},$$

gde je $R_{in} = 1$ i $C_i = 1$. Kašnjenje u ovom slučaju iznosi $7 \cdot 10^{-3}$.

b) Optimalne vrednosti kapaciteta linkova odredićemo pozivanjem na rezultat tačke c) prethodnog zadatka:

$$C_i = R_i + \sqrt{R_i}\, \frac{C - \sum\limits_j R_j}{\sum\limits_j \sqrt{R_j}},$$

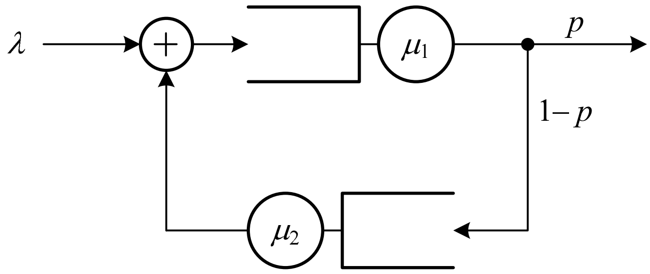
pri čemu je $C = 8$, $\sum_j R_j = 3$ i $\sum_j \sqrt{R_j} = 4{,}487$. Rezultati proračuna prikazani su u tabeli.

Optimalni kapaciteti linkova.

link	1–2	1–4	2–3	2–5	3–5	3–6	4–5	5–6
protok	0,5	0,5	0,25	0,25	0	0,25	0,5	0,75
kapacitet	1,288	1,288	0,807	0,807	0	0,807	1,288	1,715

Vidimo da se linkovima s većim protokom sada dodeljuje veći kapacitet. Novo kašnjenje iznosi $4{,}03 \cdot 10^{-3}$ i za 42,4% je manje od prethodnog.

Zadatak 3.21 U mrežu sa slike, poruke dolaze s prosečnim protokom λ. Primenom modela servisnih sistema M/M/1, odredite prosečno zadržavanje poruka u mreži.



Slika 3.21: *Mreža servisnih sistema.*

Posmatrana mreža predstavlja primer otvorene Jacksonove mreže servisnih sistema. U opštem slučaju ovakve mreže koja se sastoji od K servisnih sistema M/M/m, jednačine konzervacije protoka u stacionarnom stanju glase

$$\lambda_j = \lambda_{s,j} + \sum_{i=1}^{K} q_{ij}\lambda_i, \quad j = 1, 2, \ldots, K,$$

gde je λ_j ukupan protok dolazaka u servisni sistem j, $\lambda_{s,j}$ protok spoljašnjih dolazaka ka ovome sistemu, dok su sa q_{ij} označeni koeficijenti rutiranja, tj. verovatnoće događaja da je korisnik po završetku obrade u i-tom sistemu upućen u j-ti. Jednačine konzervacije protoka možemo pregledno napisati u matričnoj formi,

$$\mathbf{\Lambda} = \mathbf{\Lambda}_s + \mathbf{Q}^T\mathbf{\Lambda},$$

gde je

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_K \end{bmatrix},$$

$$\mathbf{\Lambda}_s = \begin{bmatrix} \lambda_{s,1} \\ \lambda_{s,2} \\ \vdots \\ \lambda_{s,K} \end{bmatrix}$$

i

$$\mathbf{Q} = \begin{bmatrix} q_{11} & q_{12} & \cdots & q_{1K} \\ q_{21} & q_{22} & \cdots & q_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ q_{K1} & q_{K2} & \cdots & q_{KK} \end{bmatrix}.$$

Odavde se nepoznati protoci mogu odrediti kao

$$\mathbf{\Lambda} = (\mathbf{I} - \mathbf{Q}^T)^{-1}\mathbf{\Lambda}_s.$$

Da bi se u ovakvoj mreži moglo uspostaviti stacionarno stanje, iskorišćenost svakog servera mora biti manja od jedan,

$$\rho_j = \frac{\lambda_j}{m_j \mu_j} < 1, \quad j = 1, 2, \ldots, K,$$

gde je s m_j označen broj servera u j-tom sistemu.

Jackson je pokazao da je pod navedenim pretpostavkama združena verovatnoća stanja mreže jednaka proizvodu verovatnoća stanja pojedinačnih servisnih sistema,

$$\mathrm{P}\left(n_1, n_2, \ldots, n_K\right) = \prod_{i=1}^{K} p_i(n_i), \quad i = 1, 2, \ldots, K.$$

Ovaj rezultat znači da se pojedinačni servisni sistemi mogu posmatrati nezavisno od ostalih.

Vratimo se rešavanju našega zadatka. Neka je protok dolazaka paketa u prvi sistem λ_1, a u drugi λ_2; po Burkeovoj teoremi, u stacionarnom stanju toliki će redom biti i protoci izlazaka paketa iz ovih sistema. Pošto u mreži nema ni generisanja, ni odbacivanja paketa, izlazni protok će biti λ, pa je zbog svojstva razdvajanja Poissonovog procesa $\lambda = p\lambda_1$ i $\lambda_2 = (1-p)\lambda_1$. Odavde je

$$\lambda_1 = \frac{\lambda}{p},$$

$$\lambda_2 = \frac{\lambda(1-p)}{p_1}.$$

Iskorišćenosti servera stoga su

$$\rho_1 = \frac{\lambda}{\mu_1 p}$$

i

$$\rho_2 = \frac{\lambda(1-p)}{\mu_2 p}.$$

Prosečan broj paketa u prvom sistemu je

$$N_1 = \frac{\rho_1}{1-\rho_1},$$

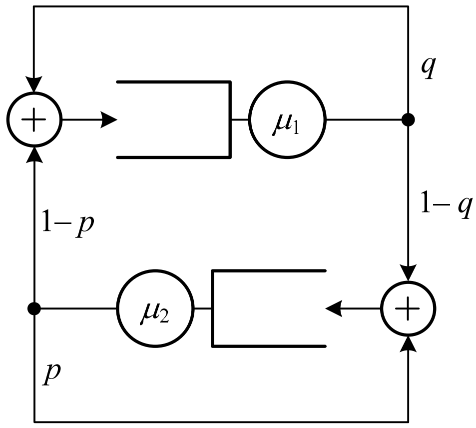
a u drugom

$$N_2 = \frac{\rho_2}{1-\rho_2}.$$

Prema Littleovoj teoremi, prosečno zadržavanje paketa u mreži je

$$T = \frac{N_1 + N_2}{\lambda} = \frac{\rho_1}{\lambda(1-\rho_1)} + \frac{\rho_2}{\lambda(1-\rho_2)}.$$

Zadatak 3.22 U zatvorenoj Jacksonovoj mreži sa slike kruži M paketa. Modelirajući redove čekanja servisnim sistemima M/M/1, odredite verovatnoće njenih stanja.



Slika 3.22: *Mreža servisnih sistema.*

Zatvorene Jacksonove mreže su mreže bez izvora i ponora saobraćaja. Zadržavajući oznake iz prethodnog zadatka, u opštem slučaju zatvorene mreže s K servisnih sistema biće

$$\sum_{j=1}^{K} q_{ij} = 1, \quad i = 1, 2, \ldots, K,$$

jer korisnici u ovu mrežu niti dolaze spolja, niti iz nje izlaze.

Sistem jednačina konzervacije protoka sada je homogen,

$$(\mathbf{I} - \mathbf{Q}^T)\mathbf{\Lambda} = \mathbf{0}$$

i da bismo ga rešili, postupićemo na sledeći način. Izabraćemo jedan servisni sistem za referentni; bez umanjenja opštosti, neka je to sistem broj jedan. Za njega ćemo pretpostaviti protok dolazaka $\lambda_1^* = \alpha$, pri čemu je α proizvoljna vrednost. Radi daljeg rešavanja, pogodno je usvojiti $\alpha = \mu_1$, pa ćemo to i učiniti. Potom se reši sistem jednačina konzervacije protoka da bi se dobili relativni protoci $\lambda_2^*, \ldots, \lambda_K^*$, u funkciji α. Neka je $\hat{\rho}_j = \lambda_j^*/\mu_j$ relativna iskorišćenost j-tog servera, $j = 1, \ldots, K$. Verovatnoća stanja sistema tada je

$$\mathrm{P}\left(n_1, n_2, \ldots, n_K\right) = \frac{1}{G(M,K)} \prod_{i=1}^{K} \hat{\rho}_i^{n_i}, \quad i = 1, 2, \ldots, K,$$

pri čemu se vrednost člana $G(M,K)$ određuje iz uslova normiranosti verovatnoće:

$$\sum_{\substack{n_1+\cdots+n_K=M\\ 0\leqslant n_i\leqslant M}}\!\!\cdots\sum \mathrm{P}\,(n_1,n_2,\ldots,n_K)=1,$$

odakle sledi

$$G(M,K)=\sum_{\substack{n_1+\cdots+n_K=M\\ 0\leqslant n_i\leqslant M}}\!\!\cdots\sum\prod_{i=1}^{K}\hat{\rho}_i^{n_i}.$$

Buzen je predložio iterativni metod za izračunavanje $G(M,K)$, po kome je

$$G(m,k)=G(m,k-1)+\hat{\rho}_k G(m-1,k),$$

uz granične uslove

$$G(m,1)=\hat{\rho}_1^m,\quad m=0,1,\ldots,M,$$
$$G(0,k)=1,\quad k=1,2,\ldots,K.$$

Koeficijenti $G(m,k)$ na ovaj se način najlakše računaju pomoću sledeće tabele:

$k \backslash m$	0	1	2	$\cdots$	M
1	1	$\hat{\rho}_1$	$\hat{\rho}_1^2$	$\cdots$	$\hat{\rho}_1^M$
2	1				
$\vdots$	$\vdots$	$G(m,k)=G(m,k-1)+\hat{\rho}_k G(m-1,k)$			
K	1				

Ako za mrežu iz zadatka usvojimo $\lambda_1^*=\mu_1$, biće

$$\lambda_2^*=\lambda_1^*\frac{1-q}{1-p}=\mu_1\frac{1-q}{1-p}.$$

Relativne iskorišćenosti servera sada su

$$\hat{\rho}_1=1,$$
$$\hat{\rho}_2=\frac{1-q}{1-p}\,\frac{\mu_1}{\mu_2}.$$

Verovatnoća stanja je

$$\mathrm{P}\,(n_1,n_2)=\mathrm{P}(M-n,n)=\frac{\hat{\rho}_2^n}{G(M,2)},$$

gde je

$$G(M,2)=\sum_{n=0}^{M}\hat{\rho}_2^n=\frac{1-\hat{\rho}_2^{M+1}}{1-\hat{\rho}_2}.$$

4. Kontrola pristupa sredini za prenos

Zadatak 4.1 $N = 120$ korisnika dele zajedničku sredinu za prenos. Verovatnoća da uočeni korisnik ima okvir za slanje je $p = 0{,}1$. Odredite verovatnoću da u datom trenutku okvire želi slati barem $n = 21$ korisnik.

Verovatnoća da u datom trenutku tačno k korisnika ima okvir za slanje, $k = 0, 1, \ldots, N$, data je binomnom raspodelom:

$$\mathrm{P}(X = k) = \binom{N}{k} p^k \, (1-p)^{N-k}.$$

Verovatnoću da *barem* n korisnika ima spreman okvir odredićemo kao

$$\mathrm{P}(X \geqslant n) = 1 - \mathrm{P}(X < n) = 1 - \sum_{k=0}^{n-1} \binom{N}{k} p^k \, (1-p)^{N-k}.$$

U primeru iz zadatka je

$$\mathrm{P}(X \geqslant 21) = 1 - \sum_{k=0}^{20} \binom{120}{k} 0{,}1^k \, 0{,}9^{120-k}.$$

Da bismo izračunali ovu verovatnoću, pozvaćemo se na centralnu graničnu teoremu. Neka su Y_i nezavisne Bernoullijeve slučajne promenljive, koje su jednake 1 ako i samo ako korisnik i ima okvir za slanje, dok su u suprotnom jednake 0. Tada je $\mathrm{P}(Y = 1) = p = 0{,}1$, $\mu = \mathrm{E}(Y_i) = p = 0{,}1$ i $\sigma^2 = \mathrm{Var}(Y_i) = p(1-p) = 0{,}09$. Verovatnoća da barem n korisnika istovremeno ima pripremljen okvir je

$$\mathrm{P}(X \geqslant n) = \mathrm{P}\left(\sum_{i=1}^{N} Y_i \geqslant n\right) = 1 - \mathrm{P}\left(\sum_{i=1}^{N} Y_i < n\right),$$

odnosno

$$\mathrm{P}(X \geqslant 21) = 1 - \mathrm{P}\left(\sum_{i=1}^{120} Y_i < 21\right) = 1 - \mathrm{P}\left(\sum_{i=1}^{120} Y_i \leqslant 20\right).$$

Po centralnoj graničnoj teoremi, slučajna promenljiva

$$\frac{1}{\sqrt{N}}\,\frac{\sum\limits_{i=1}^{N} Y_i - N\mu}{\sigma}$$

konvergira u raspodeli ka standardnoj normalnoj promenljivoj $Z \sim \mathcal{N}(0,1)$. Stoga je

$$\mathrm{P}\left(\sum_{i=1}^{120} Y_i \leqslant 20\right) = \mathrm{P}\left(\frac{1}{\sqrt{120}}\,\frac{\sum\limits_{i=1}^{120} Y_i - 120\cdot 0{,}1}{\sqrt{0{,}09}} \leqslant \frac{20 - 120\cdot 0{,}1}{\sqrt{120\cdot 0{,}09}}\right) =$$

$$= \mathrm{P}\left(Z \leqslant 2{,}43\right) \approx 0{,}992.$$

Odavde je, konačno, $\mathrm{P}(X \geqslant 21) \approx 0{,}008$.

Zadatak 4.2 Za pristup kanalu u telekomunikacionoj mreži koristi se protokol ALOHA. Pod pretpostavkom da je broj korisnika veliki, izvedite izraz za iskorišćenost kanala.

Označimo s T trajanje jednog okvira. Neka je N broj korisnika u mreži i p verovatnoća da korisnik šalje okvir u posmatranom intervalu trajanja T.

Verovatnoća da će n korisnika pokušati da pristupe kanalu tokom intervala trajanja T data je binomnom raspodelom,

$$\mathrm{P}(X = n) = \binom{N}{n} p^n (1-p)^{N-n},\ n \geqslant 0.$$

Za veliko N, binomna se raspodela može aproksimirati Poissonovom:

$$\mathrm{P}(X = n) = \frac{G^n}{n!}\,e^{-G},\ n \geqslant 0.$$

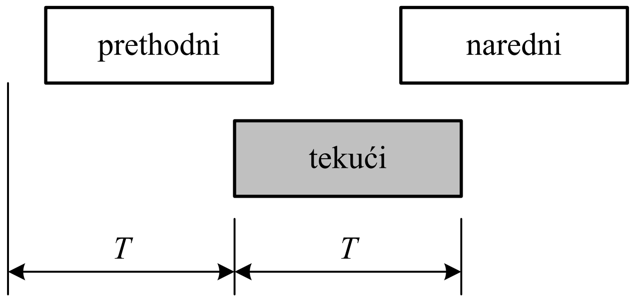
Pri tome je G ponuđeni saobraćaj u mreži, tj. očekivani broj okvira koji se generišu i pokušaju poslati tokom intervala T,

$$G = Np.$$

Do sudara dolazi onda kada emitovanje jednog okvira počne pre završetka emitovanja prethodno poslatog. Sa slike se vidi da je trajanje kritičnog vremenskog intervala za sudar $2T$. Da ne bi došlo do sudara, ovaj interval mora biti prazan – svi prethodni okviri moraju se završiti pre njega i naredni početi posle njega.

Broj pokušaja slanja tokom intervala trajanja $2T$ ponovo ima Poissonovu raspodelu, čije matematičko očekivanje sada iznosi $2G$. Verovatnoća da će interval trajanja $2T$ biti prazan je

$$[\mathrm{P}(X = 0)]^2 = e^{-2G}$$



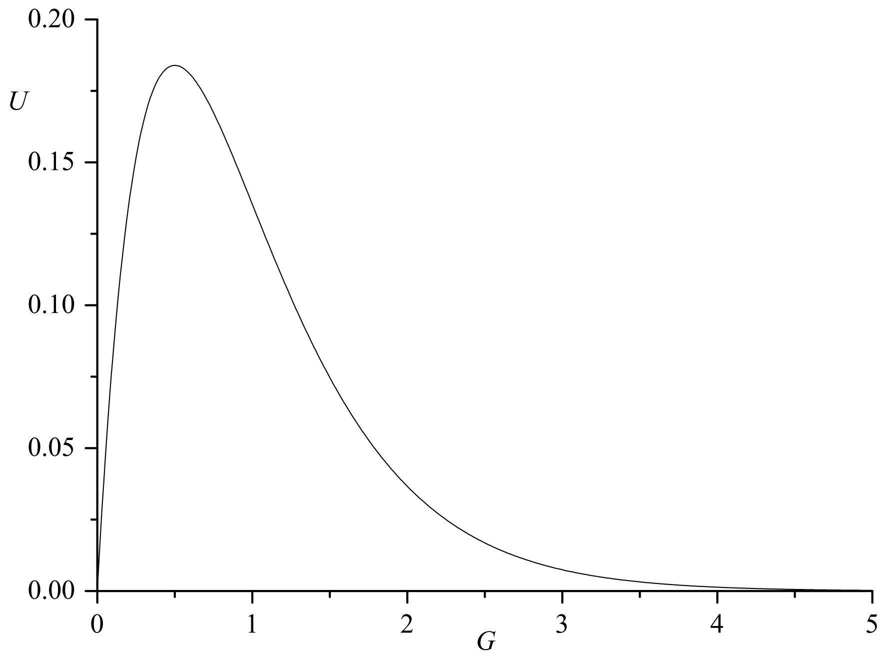
Slika 4.2: *Nadmetanje u ALOHI.*

i to je verovatnoća uspešnog slanja okvira iz prvog pokušaja. Za nju ćemo koristiti i oznaku P_1.

Iskorišćenost ili propusnost kanala, U, predstavlja prosečan broj okvira koji se uspešno pošalju iz prvog pokušaja,

$$U = G\mathrm{P}_1 = Ge^{-2G}.$$

Grafik zavisnosti iskorišćenosti kanala od ponuđenog saobraćaja dat je na narednoj slici.



Iskorišćenost kanala u ALOHI.

Maksimalna iskorišćenost se postiže za ponuđeni saobraćaj koji nalazimo iz uslova

$$\left.\frac{\mathrm{d}U}{\mathrm{d}G}\right|_{G=G_{opt}} = 0,$$

što daje

$$G_{opt} = \frac{1}{2}$$

i

$$U_{max} = \frac{1}{2e} = 18{,}394\%.$$

Zadatak 4.3 Za višestruki pristup telekomunikacionom kanalu koristi se protokol ALOHA. Prosečni broj pokušaja slanja okvira je 4. Odredite:

a) ponuđeni saobraćaj,

b) verovatnoću uspešnog slanja okvira u manje od četiri pokušaja i

c) propusnost kanala.

a) Na osnovu analize iz prethodnog zadatka, zaključujemo da je verovatnoća uspeha u k–tom pokušaju slanja data geometrijskom raspodelom,

$$\mathrm{P}_k = (1 - \mathrm{P}_1)^{k-1}\,\mathrm{P}_1 = \left(1 - e^{-2G}\right)^{k-1} e^{-2G}.$$

Prosečni broj pokušaja slanja je

$$E = \sum_{k=1}^{\infty} k\,\mathrm{P}_k = \sum_{k=1}^{\infty} k\left(1 - e^{-2G}\right)^{k-1} e^{-2G}.$$

Uvedemo li smenu $x = 1 - e^{-2G} \in [\,0,\ 1)$, imaćemo

$$E = e^{-2G} \sum_{k=1}^{\infty} \frac{\mathrm{d}x^k}{\mathrm{d}x}.$$

Ovaj red apsolutno konvergira, pa nakon što operatori diferenciranja i sumiranja zamene mesta dobijamo

$$\begin{aligned} E &= e^{-2G} \frac{\mathrm{d}}{\mathrm{d}x} \sum_{k=1}^{\infty} x^k = \\ &= e^{-2G} \frac{\mathrm{d}}{\mathrm{d}x} \left(\frac{1}{1-x} - 1\right) = \\ &= \frac{e^{-2G}}{(1-x)^2} = \\ &= e^{2G}. \end{aligned}$$

Odavde je ponuđeni saobraćaj

$$G = \frac{\ln E}{2} = 0{,}693.$$

b) Verovatnoća uspeha u manje od četiri pokušaja je

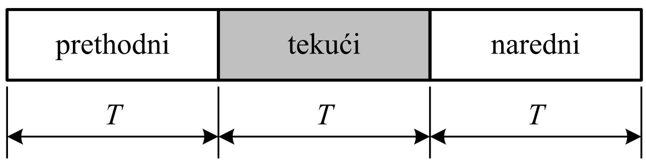
$$\begin{aligned}\mathrm{P}(k<4) &= \mathrm{P}_1+\mathrm{P}_2+\mathrm{P}_3 = \\ &= e^{-2G}\left(1+1-e^{-2G}+\left(1-e^{-2G}\right)^2\right) = \\ &= 0{,}578.\end{aligned}$$

c) Propusnost kanala je

$$U = Ge^{-2G} = 0{,}173.$$

Zadatak 4.4 Odredite verovatnoću uspešnog slanja, prosečan broj pokušaja slanja i propusnost kanala za *slotted* ALOHA protokol.

Kod protokola *slotted* ALOHA, okviri se šalju u diskretnim intervalima vremena, tzv. *slotovima*; kritični interval za sudare stoga je dvostruko kraći u odnosu na izvornu ALOHU.



Slika 4.4: *Nadmetanje u slotted ALOHI.*

Verovatnoća uspešnog slanja sada je

$$\mathrm{P}_1 = e^{-G},$$

pa je propusnost

$$U = G\mathrm{P}_1 = Ge^{-G}.$$

Grafik zavisnosti propusnosti od ponuđenog saobraćaja dat je na narednoj strani.

Maksimalna propusnost se postiže za ponuđeni saobraćaj koji nalazimo iz uslova

$$\left.\frac{\mathrm{d}U}{\mathrm{d}G}\right|_{G=G_{opt}} = 0,$$

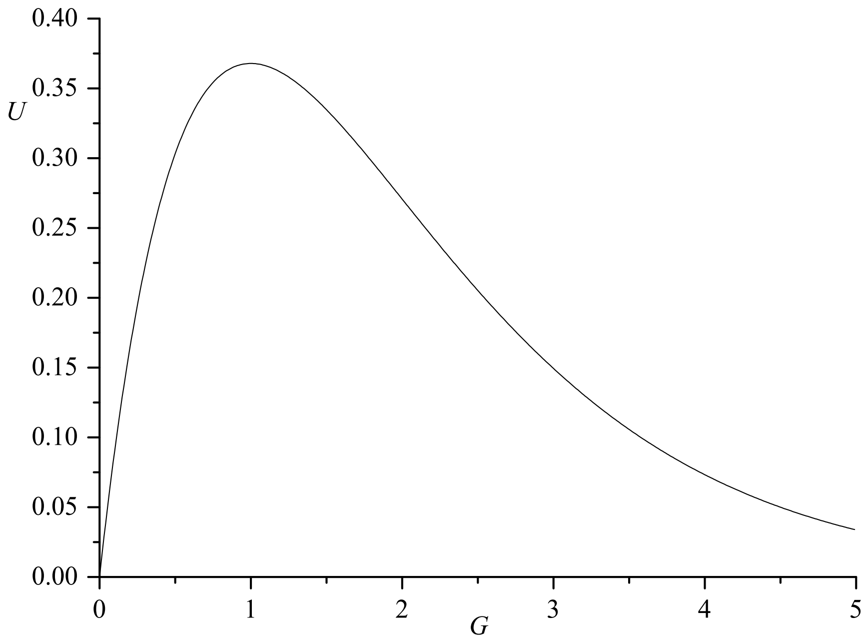
što daje

$$G_{opt} = 1$$

i

$$U_{max} = \frac{1}{e} = 36{,}788\%,$$

što je dvostruko više od izvorne ALOHE.



Propusnost kanala u slotted ALOHI.

Verovatnoća sudara je $1 - \mathrm{P}_1 = 1 - e^{-G}$. Verovatnoća uspeha u k-tom pokušaju slanja, $k = 1,\ 2,\ \ldots$ ima geometrijsku raspodelu,

$$\mathrm{P}_k = (1 - \mathrm{P}_1)^{k-1}\,\mathrm{P}_1 = \left(1 - e^{-G}\right)^{k-1} e^{-G}.$$

Prosečan broj pokušaja slanja okvira je

$$E = \sum_{k=1}^{\infty} k\,\mathrm{P}_k = \sum_{k=1}^{\infty} k\left(1 - e^{-G}\right)^{k-1} e^{-G}.$$

Uvešćemo smenu $x = 1 - e^{-G} \in [\,0,\ 1)$, pa će biti

$$E = e^{-G} \sum_{k=1}^{\infty} \frac{\mathrm{d}x^k}{\mathrm{d}x}.$$

Ovaj red apsolutno konvergira, pa nakon što operatori diferenciranja i sumiranja zamene mesta dobijamo

$$\begin{aligned} E &= e^{-G} \frac{\mathrm{d}}{\mathrm{d}x} \sum_{k=1}^{\infty} x^k = \\ &= e^{-G} \frac{\mathrm{d}}{\mathrm{d}x} \frac{x}{1-x} = \\ &= \frac{e^{-G}}{(1-x)^2} = \\ &= e^{G}. \end{aligned}$$

Zadatak 4.5 U mreži s velikim brojem korisnika, koristi se protokol *slotted* ALOHA. Korisnici prosečno generišu 50 okvira u sekundi, uključujući originale i retransmisije. Trajanje okvira je 40 ms. Odredite:

a) verovatnoću uspešnog slanja u prvom pokušaju,

b) verovatnoću uspešnog slanja nakon dva sudara i

c) prosečan broj pokušaja slanja.

Ponuđeni saobraćaj jednak je proizvodu brzine generisanja okvira i trajanja okvira,

$$G = 50\ \mathrm{s}^{-1} \cdot 0{,}04\ \mathrm{s} = 2.$$

Korišćenjem obrazaca iz prethodnog zadatka, sada jednostavno dobijamo:

a) $\mathrm{P}_1 = e^{-G} = 0{,}135$,

b) $\mathrm{P}_3 = \left(1 - e^{-G}\right)^2 e^{-G} = 0{,}1$,

c) $E = e^G = 7{,}4$.

Zadatak 4.6 Merenjem na *slotted* ALOHA kanalu s velikim brojem korisnika utvrđeno je da je 10% slotova prazno.

a) Koliki je ponuđeni saobraćaj, G?

b) Kolika je propusnost kanala, U?

a) Verovatnoća praznog slota je

$$\mathrm{P}(X = 0) = \frac{G^0}{0!}\, e^{-G} = e^{-G}$$

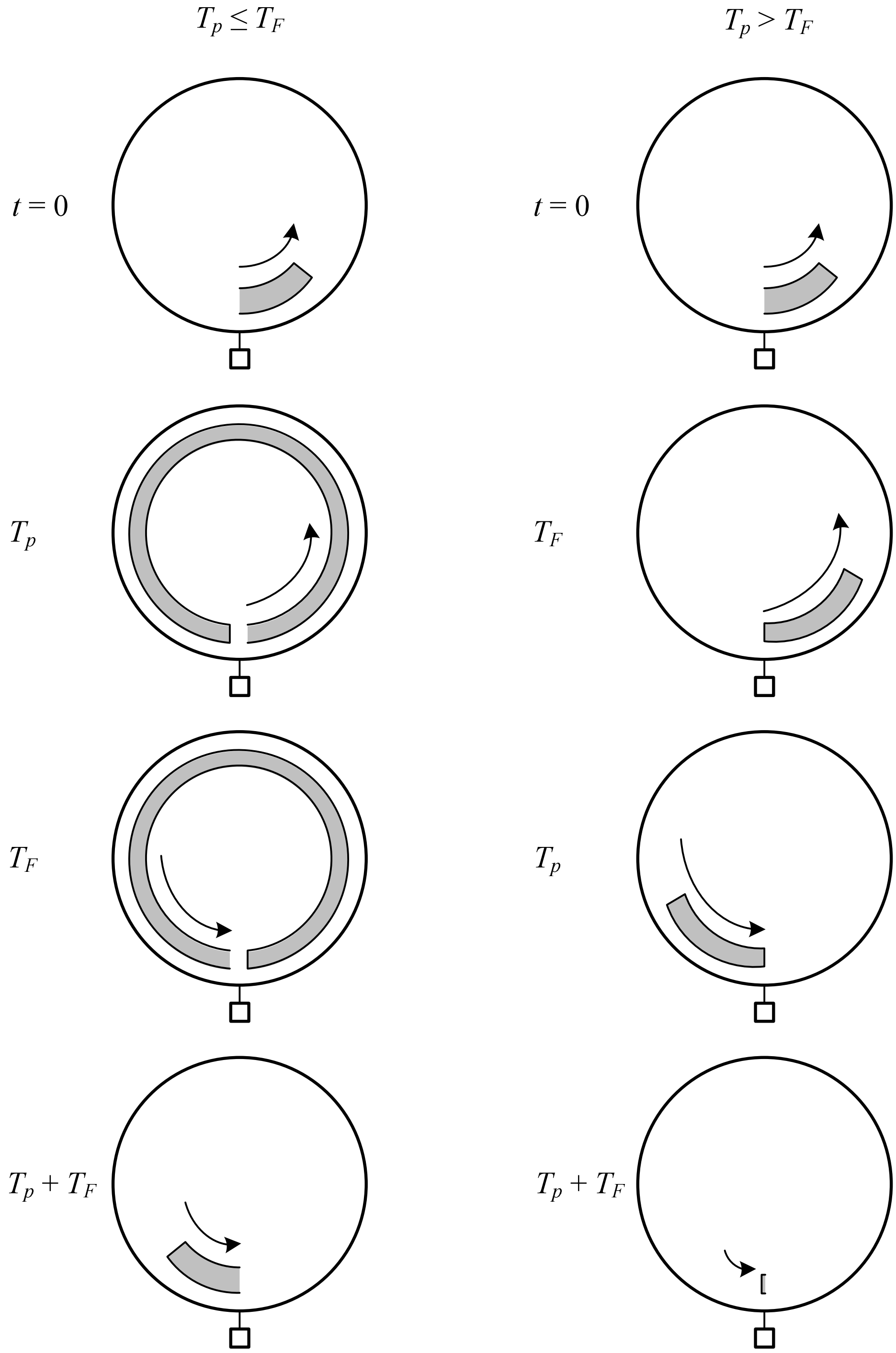
i, po postavci zadatka, iznosi 0,1. Odavde dobijamo

$$G = -\ln \mathrm{P}(X = 0) = 2{,}3.$$

b) Propusnost kanala je

$$U = Ge^{-G} = 0{,}23.$$

Zadatak 4.7 N ekvidistantnih stanica povezano je u mrežu s topologijom prstena. Za kontrolu pristupa sredini za prenos, koristi se procedura sa žetonom. Ukoliko je prosečno trajanje okvira T_F i propagaciono kašnjenje u prstenu T_p, izvedite izraz za iskorišćenost kanala.



Slika 4.7: *Slanje okvira u mreži s topologijom prstena.*

Razlikovaćemo dva slučaja, kada je $T_p \leqslant T_F$ i kada je $T_p > T_F$, koji su ilustrovani na slici. Pretpostavićemo da, osim jedne stanice koja želi poslati okvir, ostale nisu aktivne.

Ako je trajanje okvira veće ili jednako propagacionom kašnjenju, tada će se slanje u posmatranoj mreži odvijati po sledećem scenariju:

- uočena stanica u trenutku $t = 0$ uzima žeton i započinje slanje okvira,
- okvir dolazi do odredišta, nastavlja kruženje prstenom i u trenutku $t = T_p$ vraća

se stanici koja ga je poslala,

- stanica završava slanje okvira u trenutku $t = T_F$,
- stanica oslobađa žeton,
- žeton stiže do naredne stanice u trenutku $t = T_F + \frac{T_p}{N}$.

Iskorišćenost kanala u ovom slučaju je

$$U = \frac{T_F}{T_F + \dfrac{T_p}{N}}.$$

Ako je, pak, trajanje okvira kraće od propagacionog kašnjenja, scenario slanja biće drugačiji:

- uočena stanica u trenutku $t = 0$ uzima žeton i počinje slanje okvira,
- stanica završava slanje okvira u trenutku $t = T_F$,
- okvir dolazi do odredišta, nastavlja kruženje prstenom i u trenutku $t = T_p$ vraća se stanici koja ga je poslala,
- stanica oslobađa žeton,
- žeton stiže do naredne stanice u trenutku $t = T_p + \frac{T_p}{N}$.

Iskorišćenost kanala sada će biti

$$U = \frac{T_F}{T_p\left(1 + \dfrac{1}{N}\right)}.$$

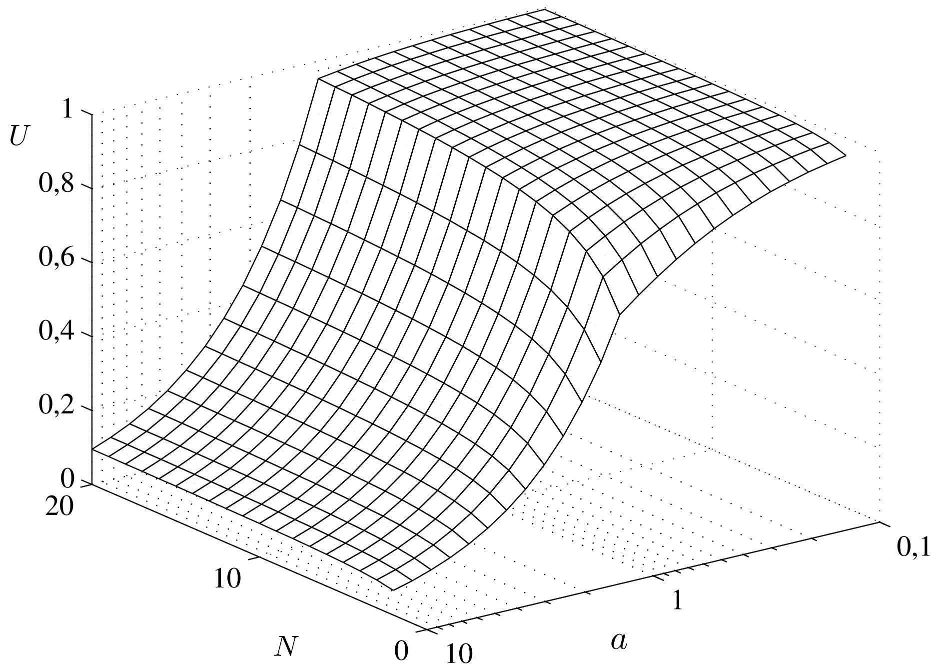
Definišimo normalizovano kašnjenje u mreži,

$$a = \frac{T_p}{T_F}.$$

Izraz za iskorišćenost kanala kod procedure prsten sa žetonom (*token ring*) tada možemo napisati u obliku

$$U = \begin{cases} \dfrac{1}{1 + \dfrac{a}{N}}, & a \leqslant 1 \\ \dfrac{1}{a\left(1 + \dfrac{1}{N}\right)}, & a > 1 \end{cases}.$$

Ova zavisnost je ilustrovana na narednoj slici.



Iskorišćenost kanala u prstenastoj mreži sa žetonom.

Sa slike se vidi da iskorišćenost kanala u ovakvim mrežama *raste* s povećavanjem broja stanica. Zbog čega se to dešava?

Za veliki broj stanica u mreži, asimptotska iskorišćenost kanala biće

$$\lim_{N\to\infty} U = \begin{cases} 1, & a \leqslant 1 \\ \dfrac{1}{a}, & a > 1 \end{cases}.$$

Zadatak 4.8 N stanica dele zajednički kanal kapaciteta R po principu „anketiranja" (*polling*). U toku jednog ciklusa, stanica može emitovati najviše Q bita podataka. Trajanje zaštitnog intervala između aktivnosti dveju stanica je t_{poll}. Kolika je maksimalna propusnost ovoga kanala?

Trajanje jednog ciklusa slanja je

$$T = N\left(\frac{Q}{R} + t_{poll}\right).$$

Tokom ovoga intervala, kanalom se prenese najviše NQ bita podataka. Maksimalna propusnost stoga je

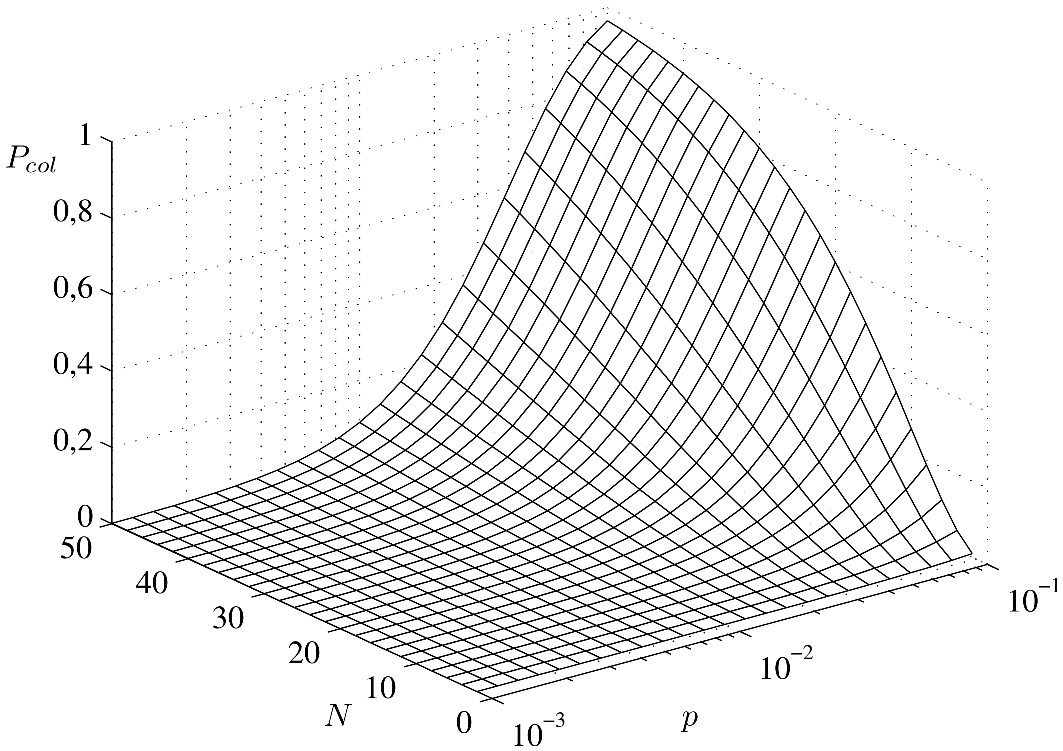
$$U = \frac{NQ}{N\left(\dfrac{Q}{R} + t_{poll}\right)} = \frac{R}{1 + \dfrac{Rt_{poll}}{Q}}.$$

Zadatak 4.9 U lokalnoj mreži, N stanica komunicira uz korišćenje CSMA/CD. Verovatnoća da stanica emituje u datom trenutku je p. Kolika je verovatnoća sudara?

Do sudara će doći svaki put kada barem dve stanice budu istovremeno pokušale poslati podatke. Verovatnoća ovoga događaja je

$$\begin{aligned} P_{col} &= P(n \geqslant 2) = \\ &= 1 - [P(n=0) + P(n=1)] = \\ &= 1 - (1-p)^N - Np(1-p)^{N-1}. \end{aligned}$$

Ova zavisnost ilustrovana je na narednoj slici.



Slika 4.9: *Verovatnoća sudara u mreži u kojoj se koristi CSMA/CD.*

Zadatak 4.10 Izvedite izraz za maksimalnu iskorišćenost kanala u mreži u kojoj se koristi CSMA/CD, ako su poznati broj stanica u mreži, N, trajanje okvira, T_F i propagaciono kašnjenje, T_p.

Neka je trajanje intervala posmatranja jednako maksimalnom vremenu potrebnom da bi se detektovao sudar, a to je dvostruko trajanje propagacionog kašnjenja. Neka je, dalje, verovatnoća da uočena stanica unutar tog intervala ima okvir za slanje p.

Verovatnoća da unutar intervala posmatranja *samo jedna* stanica ima okvir za slanje data je binomnom raspodelom:

$$\mathrm{P}(X=1)=\binom{N}{1}p^1(1-p)^{N-1}=Np(1-p)^{N-1}.$$

Neka sada uspešnom slanju prethodi i sudara. Prosečan broj sudara je

$$n=\sum_{i=0}^{\infty} i\,(1-\mathrm{P}(X=1))^i\,\mathrm{P}(X=1).$$

Sumiranjem dobijamo

$$n=\frac{1-\mathrm{P}(X=1)}{\mathrm{P}(X=1)}=\frac{1-Np(1-p)^{N-1}}{Np(1-p)^{N-1}}.$$

Iskorišćenost kanala data je izrazom

$$U=\frac{T_F}{T_F+2T_p n}.$$

Ako definišimo normalizovano kašnjenje,

$$a=\frac{T_p}{T_F},$$

izraz za iskorišćenost kanala možemo napisati kao

$$U=\frac{1}{1+2an}.$$

Izjednačavanjem izvoda po p s nulom, dobijamo da je iskorišćenost maksimalna za $p=1/N$. Nije teško videti da je u ovome slučaju maksimalna verovatnoća da samo jedna stanica ima okvir za slanje i da je data izrazom

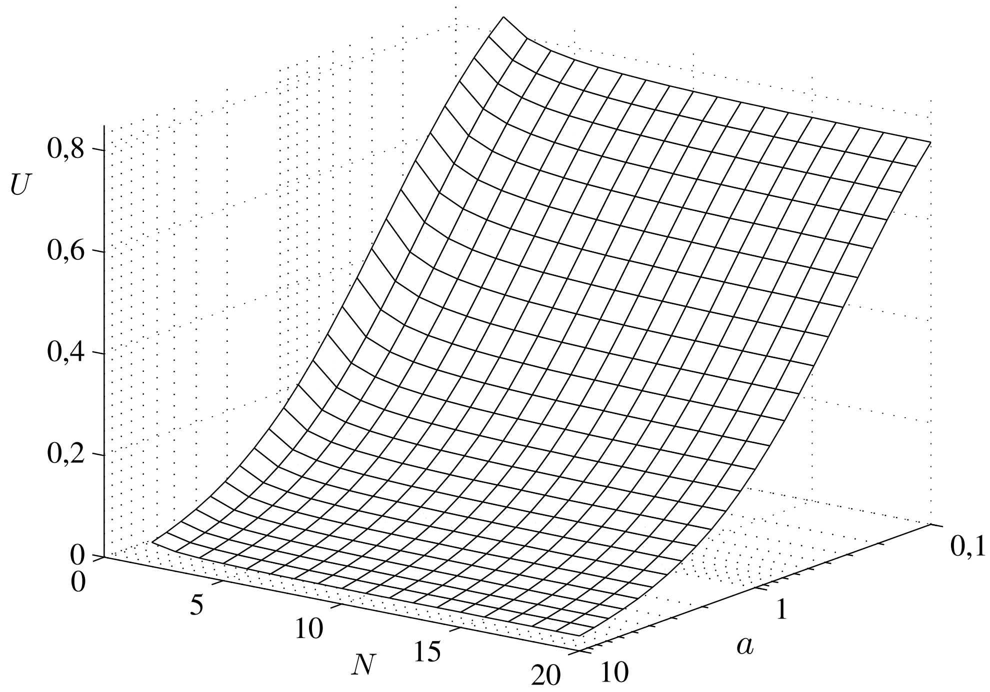
$$\mathrm{P}(X=1)=\left(1-\frac{1}{N}\right)^{N-1}.$$

Očekivani broj sudara sada je

$$n=\frac{1-\left(1-\frac{1}{N}\right)^{N-1}}{\left(1-\frac{1}{N}\right)^{N-1}}.$$

Grafik zavisnosti maksimalne iskorišćenosti kanala od broja stanica i normalizovanog kašnjenja prikazan je na slici 4.10. Čitaocu se prepušta da uporedi ovaj rezultat s rezultatom zadatka 4.7.

Zadatak 4.11 Koliko iznosi očekivani broj sudara u CSMA/CD mreži s velikim brojem korisnika?



Slika 4.10: *Maksimalna iskorišćenost kanala u mreži u kojoj se koristi CSMA/CD.*

U prethodnom zadatku smo pokazali da je očekivani broj sudara u mreži s N korisnika koji komuniciraju uz korišćenje CSMA/CD

$$n = \frac{1 - \left(1 - \dfrac{1}{N}\right)^{N-1}}{\left(1 - \dfrac{1}{N}\right)^{N-1}}.$$

Kada $N \to \infty$, odavde dobijamo

$$n_\infty = \lim_{N\to\infty} n = \frac{1 - e^{-1}}{e^{-1}},$$

jer je $\lim\limits_{N\to\infty} \left(1 - \dfrac{1}{N}\right)^{N-1} = e^{-1}$.

Izračunavanjem dobijamo da očekivani broj sudara iznosi iznosi 1,718. Je li to dobar ili loš rezultat?

Zadatak 4.12 Izvedite izraz za prosečno vreme koje će stanica koja koristi p-perzistentni CSMA protokol čekati do otpočinjanja slanja. Pretpostavite da posmatrana stanica ima spreman okvir za slanje i da osim nje, nijedna druga stanica neće slati okvire.

Po p-perzistentnom CSMA protokolu, ako je sredina za prenos slobodna, stanica će otpočeti slanje s verovatnoćom p, ili će s verovatnoćom $1 - p$ čekati fiksno vreme T, koje je jednako maksimalnom propagacionom kašnjenju. U potonjem slučaju, stanica će nakon isteka vremena T ponovo otpočeti slanje s verovatnoćom p, ili će čekati još T itd.

Verovatnoća događaja da je stanica emitovala u i-tom pokušaju, $i = 1, 2, \ldots$ data je geometrijskom raspodelom:

$$P_i = (1-p)^{i-1}p.$$

Prosečan broj pokušaja jednak je njenom očekivanju

$$\mathrm{E}I = \sum_{i=1}^{\infty} i(1-p)^{i-1}p = \frac{1}{p}.$$

Ako stanica u proseku završi slanje posle $\mathrm{E}I$ pokušaja, to znači da je čekala $\mathrm{E}I - 1$ interval trajanja T. Prosečno čekanje stoga je

$$\begin{aligned} Q &= (\mathrm{E}I - 1)\,T = \\ &= \left(\frac{1}{p} - 1\right) T. \end{aligned}$$

Zadatak 4.13 U telekomunikacionoj mreži je primenjena sledeća modifikacija p-perzistentnog CSMA protokola: Ukoliko je kanal slobodan, verovatnoća emitovanja u prvom pokušaju je p, dok u i-tom, $i > 1$, poraste za $\frac{1-p_{i-1}}{2}$.

a) Izvedite izraz za verovatnoću događaja da je stanica emitovala nakon $k \geqslant 1$ povlačenja.

b) Koliko će se stanica u proseku puta povući, ukoliko je $p = 0{,}3$?

a) Po postavci zadatka, verovatnoća da stanica emituje u i-tom pokušaju ($i > 1$) je

$$p_i = p_{i-1} + \frac{1 - p_{i-1}}{2}.$$

Da bismo stekli uvid u ovu rekurzivnu relaciju, ispišimo izraze za prvih nekoliko slučajeva:

$$\begin{aligned} p_1 &= p, \\ p_2 &= p + \frac{1-p}{2}, \\ p_3 &= 1 - \frac{1-p}{4}, \\ p_4 &= 1 - \frac{1-p}{8}, \\ p_5 &= 1 - \frac{1-p}{16} \quad \text{itd.} \end{aligned}$$

Pošto i p_2 možemo napisati kao $1 - \frac{1-p}{2}$, zaljučujemo da u opštem slučaju važi

$$p_i = 1 - \frac{1-p}{2^{i-1}}, \quad i > 1.$$

Razmotrimo sada verovatnoću združenog događaja, da je uspeh zabeležen nakon serije od k povlačenja, $k \geqslant 1$:

$$\mathrm{P}_1 = (1-p)\left(1 - \frac{1-p}{2}\right),$$

$$\mathrm{P}_2 = (1-p)\,\frac{1-p}{2}\left(1 - \frac{1-p}{4}\right),$$

$$\mathrm{P}_3 = (1-p)\,\frac{1-p}{2}\,\frac{1-p}{4}\left(1 - \frac{1-p}{8}\right) \quad \text{itd.}$$

U opštem slučaju, važiće

$$\begin{aligned}
\mathrm{P}_k &= \prod_{i=1}^{k} \frac{1-p}{2^{i-1}}\left(1 - \frac{1-p}{2^k}\right) = \\
&= (1-p)^k \left(1 - \frac{1-p}{2^k}\right) \prod_{i=1}^{k} 2^{1-i} = \\
&= (1-p)^k \left(1 - \frac{1-p}{2^k}\right) 2^{k(1-k)/2}.
\end{aligned}$$

b) Prosečan broj povlačenja jednak je matematičkom očekivanju gornje raspodele:

$$\mathrm{E}\,K = \sum_{k=1}^{\infty} k P_k.$$

Uvrštavanjem $p = 0{,}3$, ustanovljavamo da članovi ove sume indeksa većeg od četiri postaju zanemarivo mali, pa je očekivani broj povlačenja 0,99.

5. Kontrola logičkog linka

Zadatak 5.1 Odredite kombinaciju bita koji se šalju na liniju veze, za slovo `P`, ako se koriste asinhroni prenos i 7-bitni kod JUS I.B1.002 (YUSCII).

Podsetimo se zadatka 1.4. Pri asinhronom prenosu, na liniju se šalju jedan START bit (0), korisni biti, počevši od bita najmanje težine (tzv. *little endian* format), zaštitni bit (prema standardu ISO 1177, koristi se parni paritet) i jedan STOP bit (1).

Slika 5.1: *Sekvenca bita pri asinhronom prenosu.*

Prema kodnoj tabeli koda JUS I.B1.002 (sada SRPS I.B1.002:1982), slovo `P` predstavlja se kodnom rečju 1010000. Zaštitni bit jednak je 0, jer se u korisnom sadržaju nalazi paran broj jedinica. Na liniju se stoga šalje sledeća kombinacija bita: **0**0000101$\underline{0}$**1**.

Zadatak 5.2 Odredite kombinaciju bita za reč `MREŽA`, u 7-bitnom kodu JUS I.B1.002, ako se koriste longitudinalna i vertikalna provera parnosti za sinhroni prenos.

Prema kodnoj tabeli koda JUS I.B1.002, slovima reči `MREŽA` odgovaraju sledeće kodne reči:

	b7	b6	b5	b4	b3	b2	b1
`M`	1	0	0	1	1	0	1
`R`	1	0	1	0	0	1	0
`E`	1	0	0	0	1	0	1
`Ž`	1	0	0	0	0	0	0
`A`	1	0	0	0	0	0	1

Pri sinhronom prenosu se za zaštitu podataka koristi neparni paritet, što je propisano međunarodnim standardom ISO 1177. Tako se dobija sledeća tabela provere na parnost.

M	R	E	Ž	A	
1	0	1	0	1	0
0	1	0	0	0	0
1	0	1	0	0	1
1	0	0	0	0	0
0	1	0	0	0	0
0	0	0	0	0	1
1	1	1	1	1	0
1	0	0	0	1	1

Biti se iz tabele očitavaju kolonu po kolonu – odozgo nadole, sleva udesno, pa se tako dobija sekvenca:

101100110100101010100010000000101000001100100101.

Zadatak 5.3 Za kontrolu ispravnosti podataka u telekomunikacionoj mreži, koristi se procedura provere ciklične redundantnosti. Generišući polinom je CRC-8 ($x^8 + x^2 + x + 1$). Odredite rezultat izračunavanja FCS u predajniku za ulaznu sekvencu 10 1101 0101.

Zadatak ćemo rešiti na dva načina i to direktnim računanjem ostatka pri deljenju produžene ulazne sekvence generišućim polinomom i propuštanjem produžene sekvence kroz logički automat. Sve računske operacije pri tome obavljamo *po modulu 2*.

Ulaznu sekvencu možemo predstaviti polinomom

$$D(x) = x^9 + x^7 + x^6 + x^4 + x^2 + 1,$$

dok je generišući polinom

$$P(x) = x^8 + x^2 + x + 1.$$

Pomnožimo polinom $D(x)$ faktorom x^8:

$$D(x) \cdot x^8 = x^{17} + x^{15} + x^{14} + x^{12} + x^{10} + x^8.$$

Podelimo li polinom $D(x) \cdot x^8$ generišućim polinomom, imaćemo

$$\frac{D(x) \cdot x^8}{P(x)} = Q(x) + \frac{R(x)}{P(x)},$$

gde je $R(x)$ ostatak pri deljenju, koji predstavlja rezultat provere ciklične redundantnosti.

U našem primeru ćemo imati:

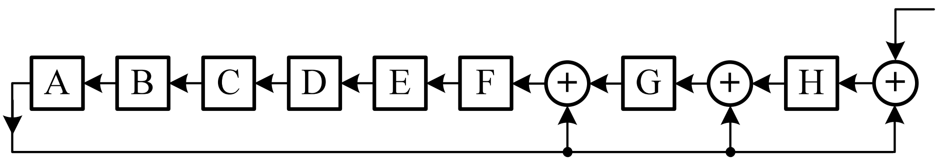
$$(x^{17}+x^{15}+x^{14}+x^{12}+x^{10}+x^{8}) : (x^8+x^2+x+1) = x^9+x^7+x^6+x^4+x^3+1.$$

$$\begin{array}{r}
x^{17}+x^{11}+x^{10}+x^{9} \\
\hline
x^{15}+x^{14}+x^{12}+x^{11}+x^{9}+x^{8} \\
x^{15}+x^{9}+x^{8}+x^{7} \\
\hline
x^{14}+x^{12}+x^{11}+x^{7} \\
x^{14}+x^{8}+x^{7}+x^{6} \\
\hline
x^{12}+x^{11}+x^{8}+x^{6} \\
\underline{x^{12}+x^{6}+x^{5}+x^{4}} \\
x^{11}+x^{8}+x^{5}+x^{4} \\
\underline{x^{11}+x^{5}+x^{4}+x^{3}} \\
x^{8}+x^{3} \\
x^{8}+x^{2}+x+1 \\
\hline
x^{3}+x^{2}+x+1
\end{array}$$

Dakle, ostatak pri deljenju je x^3+x^2+x+1 i njemu odgovara sekvenca 1111, odnosno, ako je produžimo do 8 bita, 0000 1111.

Na drugi način, do rezultata možemo doći deljenjem binarnih sekvenci. Količnik nas ne interesuje, pa samo evidentiramo ostatak:

$$\begin{array}{r}
10110101010 0000000 \\
\underline{100000111} \\
110110110 \\
\underline{100000111} \\
101100010 \\
\underline{100000111} \\
110010100 \\
\underline{100000111} \\
100100110 \\
\underline{100000111} \\
100001000 \\
\underline{100000111} \\
1111
\end{array}$$



Slika 5.3: *Automat za CRC.*

Problem možemo rešiti i na treći način, propuštanjem ulazne sekvence koja je pomerena ulevo za osam mesta kroz automat prikazan na slici 5.3. Automat se sastoji iz elemenata za kašnjenje, koji su označeni slovima A–H, sabirača po modulu 2 i veza između njih. Povratna sprega zatvara se na način koji odgovara stepenima x koji postoje u generatorskom polinomu. Elementu koji je označen slovom A odgovara stepen x^8, elementu B x^7 i tako redom do elementa H, kome odgovara stepen x^1. Sabirač po modulu 2 koji se nalazi ispred ulaza elementa H odgovara stepenu x^0.

Pretpostavićemo da je u početnom trenutku stanje svih elemenata za kašnjenje 0. Na ulaz automata dovodimo sekvencu 10 1101 0101 0000 0000. Stanja pojedinih elemenata za kašnjenje tokom prolaska sekvence kroz automat prikazana su u sledećoj tabeli.

Stanja automata za CRC.

takt	A	B	C	D	E	F	G	H	ulaz
0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	1	0
2	0	0	0	0	0	0	1	0	1
3	0	0	0	0	0	1	0	1	1
4	0	0	0	0	1	0	1	1	0
5	0	0	0	1	0	1	1	0	1
6	0	0	1	0	1	1	0	1	0
7	0	1	0	1	1	0	1	0	1
8	1	0	1	1	0	1	0	1	0
9	0	1	1	0	1	1	0	1	1
10	1	1	0	1	1	0	1	1	0
11	1	0	1	1	0	0	0	1	0
12	0	1	1	0	0	1	0	1	0
13	1	1	0	0	1	0	1	0	0
14	1	0	0	1	0	0	1	1	0
15	0	0	1	0	0	0	0	1	0
16	0	1	0	0	0	0	1	0	0
17	1	0	0	0	0	1	0	0	0
18	0	0	0	0	1	1	1	1	—

Nakon propuštanja sekvence kroz automat, stanje elemenata za kašnjenje odgovaraće ostatku pri deljenju propuštene sekvence generatorskim polinomom. Vidimo da sadržaj registara automata na kraju provere, 0000 1111, zaista odgovara ostatku koji smo direktno izračunali.

Zadatak 5.4 Za kontrolu ispravnosti podataka u telekomunikacionoj mreži koristi se procedura provere ciklične redundantnosti. Generišući polinom je CRC-12 ($x^{12}+x^{11}+x^3+x^2+x+1$). Ako na ulaz prijemnika dolazi sekvenca 1101 0101 1011 0000 0111, odredite rezultat provere.

Na samom početku, naglasimo da se u prijemniku na sekvencu *ne nadovezuju nule.* Primljenoj sekvenci iz postavke zadatka stoga odgovara polinom

$$S(x) = x^{19} + x^{18} + x^{16} + x^{14} + x^{12} + x^{11} + x^9 + x^8 + x^2 + x + 1.$$

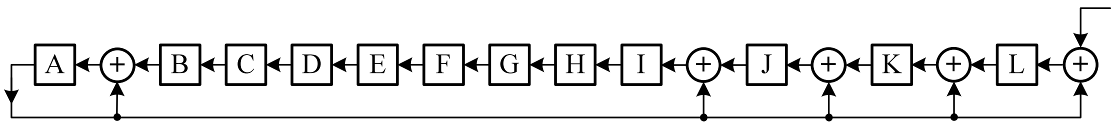
Pri deljenju polinoma, imaćemo:

$$
\begin{array}{r}
(x^{19}+x^{18}+x^{16}+x^{14}+x^{12}+x^{11}+x^{9}+x^{8}+x^{2}+x+1) : \\
: (x^{12}+x^{11}+x^{3}+x^{2}+x+1) = x^{7}+x^{4}+x^{3}+1.
\end{array}
$$

$$
\begin{array}{r}
x^{19}+x^{18}+x^{10}+x^{9}+x^{8}+x^{7} \\
\hline
x^{16}+x^{14}+x^{12}+x^{11}+x^{10}+x^{7}+x^{2}+x+1 \\
x^{16}+x^{15}+x^{7}+x^{6}+x^{5}+x^{4} \\
\hline
x^{15}+x^{14}+x^{12}+x^{11}+x^{10}+x^{6}+x^{5}+x^{4}+x^{2}+x+1 \\
x^{15}+x^{14}+x^{6}+x^{5}+x^{4}+x^{3} \\
\hline
x^{12}+x^{11}+x^{10}+x^{3}+x^{2}+x+1 \\
x^{12}+x^{11}+x^{3}+x^{2}+x+1 \\
\hline
x^{10}
\end{array}
$$

Pošto je ostatak *različit od nule*, zaključujemo da je nastupila greška u prenosu.

Automat kojim se realizuje deljenje ulazne sekvence polinomom CRC-12 prikazan je na slici.



Slika 5.4: *Automat za deljenje polinomom CRC-12.*

Čitaocu se prepušta da proveri prethodno dobijeni rezultat popunjavanjem tabele stanja logičkog automata i direktnim deljenjem binarnih sekvenci. Koji je postupak rešavanja najbrži?

Zadatak 5.5 Informaciona sekvenca 1101 0110 1101 0001 1100 prenosi se kanalom u kome je verovatnoća greške $p = 10^{-4}$. Pri prenosu se koristi procedura provere ciklične redundantnosti, pri čemu je generišući polinom CRC-CCITT $(x^{16}+x^{12}+x^{5}+1)$. Kolika je verovatnoća događaja da će pri prenosu doći do greške na trećem bitu informacione sekvence, koja se neće moći detektovati na prijemu?

Pažljivim izračunavanjem, dobijamo da je vrednost FCS za originalnu informacionu sekvencu

$$\text{FCS} = 1110\ 1100\ 1001\ 1010.$$

Ova vrednost se, zajedno s informacionom sekvencom, šalje prijemniku.

Neka se pri prenosu pogreši na trećem bitu informacione sekvence. Ta greška se neće moći detektovati na prijemu onda kada bude došlo i do greške na pojedinim bitima polja FCS, tako da *pogrešna* vrednost polja FCS bude saglasna informacionoj sekvenci s greškom na trećem bitu.

Za sekvencu 11**1**1 0110 1101 0001 1100, vrednost polja FCS je

FCS' = 1**00**0 **001**0 1**11**1 1010.

Vidimo da se sekvence FCS i FCS' razlikuju u 7 bita, što znači da je, uz jedan bit informacione sekvence, pogrešno ukupno 8 bita, dok je preostalih 28 bita ispravno. Verovatnoća ovoga događaja je

$$\mathrm{P} = p^8(1-p)^{28} \approx 9{,}972 \cdot 10^{-33}$$

i praktično je zanemariva.

Zadatak 5.6 Na telekomunikacionom kanalu se koristi ARQ procedura *stani i čekaj*. Poznati su dužina linka, l, brzina prostiranja, v, dužina okvira, L, binarni protok, V_b i verovatnoća greške po bitu, P_e.

a) Izvedite izraz za iskorišćenost kanala kada nema grešaka u prenosu.

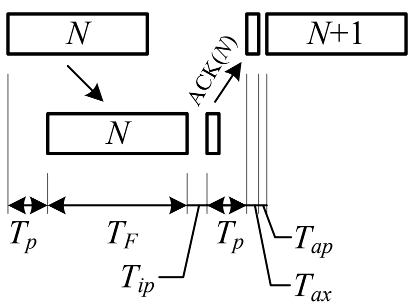
b) Izvedite izraze za iskorišćenost kanala i prosečno vreme potrebno za prenos kada se javljaju greške.

c) Nacrtajte i objasnite dijagrame stanja za predajnik i prijemnik.

a) Iskorišćenost kanala ćemo definisati kao odnos vremena potrebnog za slanje okvira, T_F i prosečnog ukupnog vremena koje protekne između dvaju slanja, T_t:

$$U = \frac{T_F}{T_t}.$$

Kada nema grešaka u prenosu, slanje okvira će se odvijati kao što je prikazano na slici.



Slika 5.6: *Slanje okvira po proceduri „stani i čekaj" kada nema grešaka u prenosu.*

Sa slike se vidi da je ukupno vreme potrebno za slanje okvira (*round trip time*)

$$T_t = T_p + T_F + T_{ip} + T_p + T_{ax} + T_{ap},$$

gde je T_p propagaciono kašnjenje, T_F trajanje okvira, T_{ip} vreme potrebno za obradu okvira na prijemu, T_{ax} trajanje pozitivne potvrde i T_{ap} vreme potrebno za obradu potvrde na predaji. Pošto nema grešaka u prenosu, izdaju se (i primaju) samo pozitivne potvrde, ACK. Pretpostavili smo da su propagaciona kašnjenja ista za oba smera prenosa.

Pošto je potvrda mnogo kraća od okvira, zanemarićemo vreme T_{ax}. Takođe, zanemarićemo i vremena potrebna za obradu okvira i potvrde. Stoga će biti

$$T_t \approx T_F + 2T_p.$$

Iz prethodne analize, zaključujemo da je iskorišćenost kanala u ovome slučaju

$$U = \frac{T_F}{T_F + 2T_p}.$$

Kako je $T_F = L/V_b$ i $T_p = l/v$, biće

$$U = \frac{\dfrac{L}{V_b}}{\dfrac{L}{V_b} + 2\dfrac{l}{v}}.$$

b) Prema rešenju zadatka 3.1, verovatnoća pogrešnog prijema okvira je

$$P_F = 1 - (1 - P_e)^L.$$

Svaki put kada se okvir bude pogrešno primio, izdaće se negativna potvrda, NAK i pokrenuće se se procedura retransmisije. Verovatnoća uspeha nakon k retransmisija data je geometrijskom raspodelom,

$$\mathrm{P}_k = P_F^k \left(1 - P_F\right).$$

Vreme koje se pri tome utroši je

$$T_k = (k+1)T_t.$$

Prosečno vreme potrebno za prenos okvira stoga je

$$\overline{T} = \sum_{k=0}^{\infty} T_k \mathrm{P}_k = \sum_{k=0}^{\infty} (k+1)T_t P_F^k \left(1 - P_F\right).$$

Pozivanjem na osobine geometrijske raspodele, ili direktnim izračunavanjem sume, dobijamo rezultat

$$\overline{T} = \frac{T_t}{1 - P_F}.$$

Iskorišćenost kanala sada je

$$U = \frac{T_F}{\overline{T}} = \frac{T_F}{T_F + 2T_p}\left(1 - P_F\right),$$

odnosno

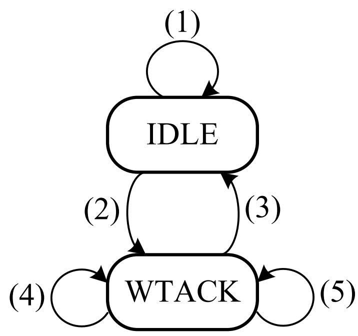
$$U = \frac{\dfrac{L}{V_b}}{\dfrac{L}{V_b} + 2\dfrac{l}{v}} (1 - P_e)^L .$$

c) Predajnik se može nalaziti u dvama stanjima, *IDLE* i *WTACK*.

Stanje IDLE je neaktivno. U njemu su moguća dva događaja.

(1) Ako predajnik u stanju IDLE bude primio pozitivnu (ACK) ili negativnu potvrdu (NAK), inkrementiraće brojač grešaka i ostaće u ovome stanju.

(2) Ako predajnik bude primio podatke od korisnika, formatiraće ih u okvir koji će poslati, startovaće tajmer, inkrementiraće brojač okvira i preći će u stanje WTACK.



Dijagram stanja predajnika.

U stanju WTACK, predajnik čeka potvrdu prijema.

(3) Ako bude dobio pozitivnu potvrdu, zaustaviće tajmer, resetovaće brojač retransmisija i preći će u stanje IDLE.

(4) Ako bude dobio negativnu potvrdu, ponovo će poslati taj okvir, startovaće tajmer i inkrementiraće brojač retransmisija. Predajnik ostaje u stanju WTACK.

(5) Ako tajmer bude izbrojao do nule pre nego što se bude dobila potvrda (pozitivna ili negativna), okvir će se ponovo poslati, startovaće se tajmer i inkrementiraće se brojač retransmisija. Predajnik ostaje u stanju WTACK.

Prijemnik se može nalaziti u jednom stanju, WTIFM.

Dijagram stanja prijemnika.

(1) Ako prijemnik bude primio neispravan okvir, izdaće negativnu potvrdu (NAK) i inkrementiraće brojač grešaka.

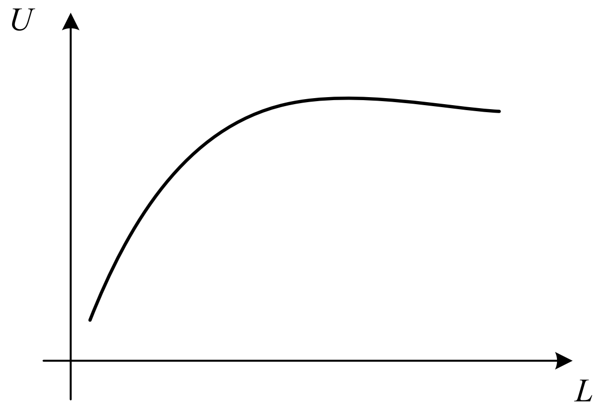
(2) Ako prijemnik bude primio ispravan okvir, izdaće pozitivnu potvrdu (ACK), prosleđiće informacioni sadržaj okvira korisniku i inkrementiraće brojač okvira.

Zadatak 5.7 Okviri dužine L prenose se uz korišćenje ARQ procedure „stani i čekaj". Protok signala je $V_b = 20$ kb/s, verovatnoća pogrešnog prijema bita $P_e = 10^{-8}$, dužina linka $l = 5$ km i brzina prostiranja $v = 2 \cdot 10^8$ m/s. Za koju će dužinu okvira iskorišćenost kanala biti maksimalna?

Polazimo od izraza za iskorišćenost kanala, koji smo izveli u prethodnom zadatku:

$$U = \frac{\dfrac{L}{V_b}}{\dfrac{L}{V_b} + 2\dfrac{l}{v}} (1 - P_e)^L .$$

Iz njega vidimo da iskorišćenost kanala, između ostalog, zavisi i od dužine okvira, što je ilustrovano na slici.



Slika 5.7: *Zavisnost iskorišćenosti kanala od dužine okvira.*

Pošto je $P_e \ll 1$ i pošto očekujemo da će dužina okvira iznositi barem nekoliko stotina bita, prethodni izraz možemo uprostiti, tako da dobijamo

$$U \approx \frac{L\,(1 - LP_e)}{L + 2\dfrac{lV_b}{v}}.$$

Optimalnu dužinu paketa odredićemo iz uslova

$$\left.\frac{\mathrm{d}U}{\mathrm{d}L}\right|_{L=L_{opt}} = 0,$$

odakle se dobija kvadratna jednačina

$$vP_eL_{opt}^2 + 4lV_bP_eL_{opt} - 2lV_b = 0.$$

Njena rešenja data su izrazom

$$L_{opt} = -2V_b\frac{l}{v} \pm \sqrt{4V_b^2\frac{l^2}{v^2} + 2\frac{V_bl}{P_ev}}.$$

Usvajamo rešenje sa znakom „+”, jer negativna dužina paketa nema fizičkog smisla. Uvrštavanjem zadatih brojčanih vrednosti, konačno dobijamo $L_{opt} \approx 9999$ b.

Zadatak 5.8 Okviri dužine $L = 500$ b prenose se uz korišćenje procedure „stani i čekaj”. Protok signala na linku je $V_b = 9600$ b/s, dužina linka $l = 7$ km i brzina prostiranja $v = 2 \cdot 10^5$ km/s. Verovatnoća ostanka predajnika u stanju WTACK je 0,03. Odredite prosečno vreme potrebno za uspešno slanje okvira i verovatnoću greške po bitu. Pretpostaviti da se vreme potrebno za obradu okvira i potvrde može zanemariti.

Pošto se zanemaruju vremena obrade okvira i potvrde, prosečno vreme koje je potrebno da bi se okvir uspešno poslao biće

$$\overline{T} = \frac{T_t}{1 - P_F},$$

pri čemu je

$$T_t = \frac{L}{V_b} + 2\frac{l}{v}$$

i

$$P_F = 1 - (1 - P_e)^L.$$

Stoga će biti

$$\overline{T} = \frac{\frac{L}{V_b} + 2\frac{l}{v}}{(1 - P_e)^L}.$$

Podsetimo se dijagrama stanja predajnika, koji smo razmatrali na stranici 82. Predajnik *ostaje* u stanju WTACK u dvama slučajevima:

i) kada dobije negativnu potvrdu ili

ii) kada ne dobije nikakvu potvrdu pre isteka vremena čekanja.

U oba slučaja, jasno je da je pri prenosu nastupila greška. Verovatnoća ostanka predajnika u stanju WTACK stoga je jednaka verovatnoći pogrešnog prijema okvira,

$$\mathrm{P(WTACK|WTACK)} = P_F = 1 - (1 - P_e)^L.$$

Odavde je

$$P_e = 1 - \sqrt[L]{1 - \mathrm{P(WTACK|WTACK)}}$$

i

$$\overline{T} = \frac{\dfrac{L}{V_b} + 2\dfrac{l}{v}}{1 - \mathrm{P(WTACK|WTACK)}}.$$

Za zadate brojčane vrednosti, dobijamo $P_e = 6 \cdot 10^{-5}$ i $\overline{T} = 53{,}77$ ms.

Zadatak 5.9 Okviri trajanja 25 µs prenose se linkom na kome je normalizovano kašnjenje $a = 0{,}8$ uz korišćenje procedure „stani i čekaj”. Koliko iznose iskorišćenost linka i očekivano vreme potrebno za prenos ukoliko je verovatnoća da će se okvir jedanput retransmitovati 4/25?

Verovatnoća da će se okvir *jedanput* retransmitovati je $P_F(1 - P_F)$, odakle je verovatnoća pogrešnog prijema okvira $P_F = 1/5$ ili $P_F = 4/5$.

Iskorišćenost linka je

$$U = \frac{1 - P_F}{1 + 2a},$$

a očekivano vreme potrebno za prenos

$$\overline{T} = \frac{T_F}{U}.$$

U prvom slučaju ($P_F = 1/5$), biće $U = 0{,}31$ i $\overline{T} = 81{,}25$ µs, a u drugom ($P_F = 4/5$) $U = 0{,}077$ i $\overline{T} = 325$ µs.

Zadatak 5.10 Na telekomunikacionom kanalu se koristi sledeća procedura za ispravljanje greške. Predajnik emituje W okvira, zaustavi se i čeka potvrdu. Ako dobije pozitivnu potvrdu za prvi okvir, nastavlja sa slanjem, a u suprotnom, ako dobije negativnu potvrdu ili ako ne dobije nijednu potvrdu do isteka vremena čekanja, ponovo šalje ovih W okvira. Ako je dužina okvira L i ako prosečan broj emitovanih okvira do uspeha iznosi μ, odredite verovatnoću pogrešnog prijema bita, P_e.

Iz postavke zadatka, zaključujemo da se radi o varijanti ARQ procedure *vrati se za N*.

Ako je uspešnom slanju okvira prethodilo k neuspeha, ukupan broj prenesenih okvira iznosiće $kW + 1$. Verovatnoća ovoga događaja je

$$\mathrm{P}_k = P_F^k \left(1 - P_F\right),$$

gde je P_F verovatnoća pogrešnog prijema okvira.

Prosečan broj okvira koji se emituju do uspeha je

$$\mu = \sum_{k=0}^{\infty} (kW+1)\mathrm{P}_k = \frac{1+(W-1)P_F}{1-P_F}.$$

Za $W = 1$, procedura se svodi na „stani i čekaj", što se vidi i iz gornjeg rezultata.

Verovatnoća pogrešnog prijema okvira sada je

$$P_F = \frac{\mu - 1}{W + \mu - 1}.$$

Dalje je

$$1 - P_F = (1 - P_e)^L \approx 1 - LP_e,$$

pa je

$$P_F \approx LP_e.$$

Stoga je

$$P_e = \frac{\mu - 1}{L(W + \mu - 1)}.$$

Zadatak 5.11 Izvedite izraze za iskorišćenost linka kod ARQ procedura *vrati se za N* i *selektivno ponavljanje*.

ARQ procedure „vrati se za N" i „selektivno ponavljanje" spadaju u grupu procedura s klizećim prozorom.

Označimo otvor prozora s W, propagaciono kašnjenje s T_p i trajanje okvra s T_F. Uz zanemarivanje trajanja potvrde i vremena potrebnih da bi se obradili okvir i potvrda, vreme potrebno za prenos u oba smera je

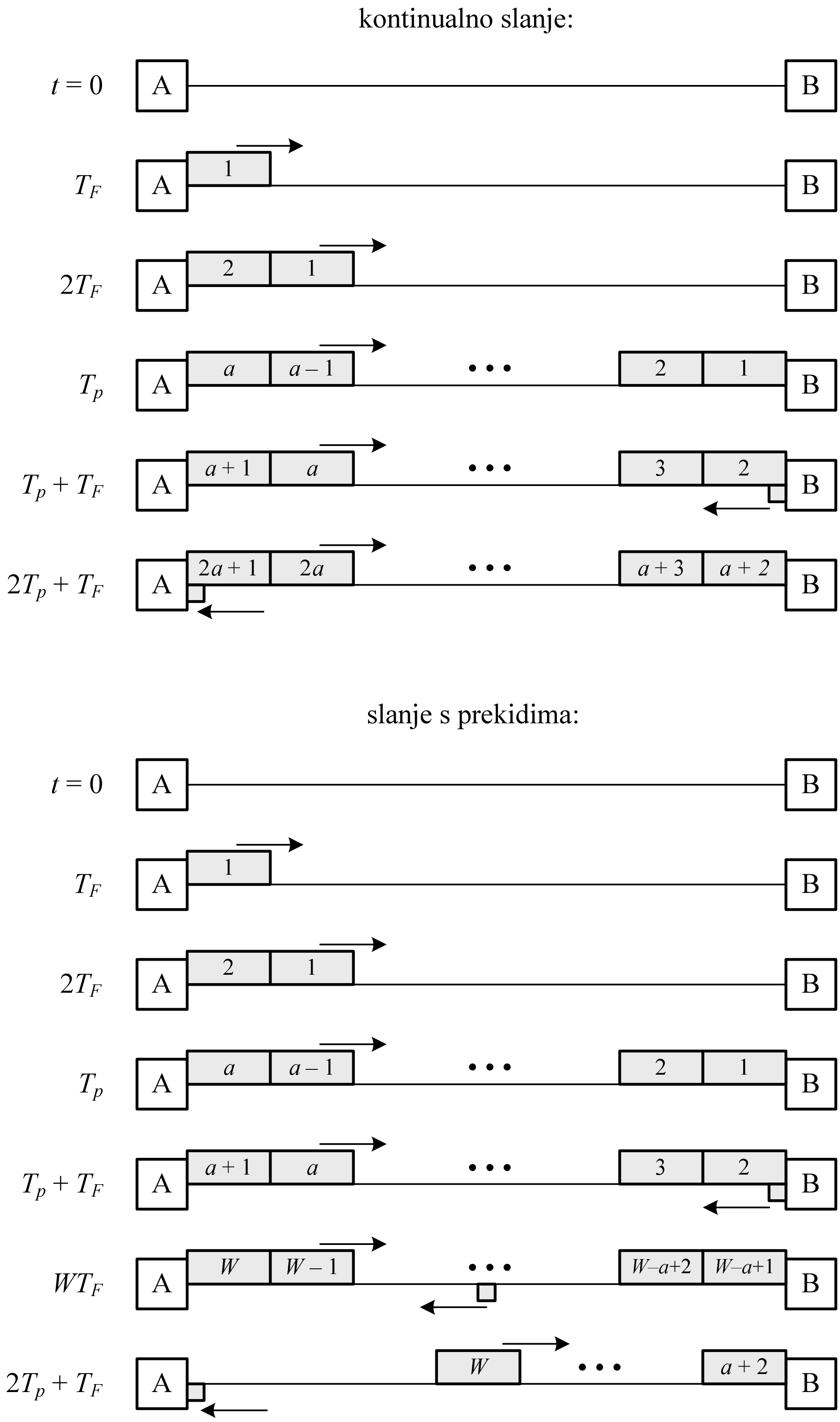
$$T_t \approx T_F + 2T_p.$$

Ako trajanje bloka od W okvira nije kraće od vremena T_t, radiće se o tzv. kontinualnom slanju, dok se u suprotnom slučaju radi o slanju s prekidima. Ova dva slučaja ilustrovana su na slici 5.11. Radi preglednosti, pretpostavljen je celobrojni odnos $a = T_p/T_F$, koji se naziva normalizovanim kašnjenjem.

Razmotrimo prvo slučaj kada nema grešaka u prenosu. Ako se radi o kontinualnom slanju, iskorišćenost linka je, očigledno, jednaka jedinici, jer se okviri šalju bez pauze. Ako se radi o slanju s prekidima, biće

$$U = \frac{WT_F}{T_t} = \frac{WT_F}{T_F + 2T_p}.$$

Neka je sada verovatnoća pogrešnog prijema okvira P_F. Razmotrimo prvo performanse procedure „selektivno ponavljanje".



Slika 5.11: *Scenariji slanja okvira uz primenu procedure s klizećim prozorom.*

Verovatnoća uspeha u i-tom pokušaju (nakon $i-1$ neuspeha) data je geometrijskom raspodelom,

$$\mathrm{P}_i = P_F^{i-1}\,(1-P_F)\,.$$

Prosečan broj pokušaja slanja jednak je njenom očekivanju

$$\overline{N} = \sum_{i=1}^{\infty} i\,\mathrm{P}_i = \frac{1}{1-P_F}.$$

Ako se radi o kontinualnom slanju, iskorišćenost linka će biti

$$U = \frac{1}{\overline{N}} = 1 - P_F,$$

a ako postoje prekidi, biće

$$U = \frac{WT_F}{\overline{T}} = \frac{WT_F}{T_F + 2T_p}(1-P_F),$$

gde je

$$\overline{T} = \frac{T_t}{1-P_F}.$$

prosečno vreme potrebno za prenos okvira.

Kod procedure „vrati se za N”, svaka greška uzrokuje retransmisiju K sukcesivnih okvira. Prosečan broj okvira koji se emituju do uspeha stoga je

$$\mu = \sum_{i=0}^{\infty}(iK+1)P_F^i\,(1-P_F) = \frac{1+(K-1)P_F}{1-P_F}.$$

Prema slici 5.11, nije teško zaključiti da je kod kontinualnog slanja

$$K \approx 1 + 2\frac{T_p}{T_F},$$

dok je kod slanja s prekidima

$$K = W.$$

Iskorišćenost linka kod ARQ procedure „vrati se za N” stoga je

$$U = \frac{T_F}{T_F + 2T_pP_F}(1-P_F),$$

ako je slanje kontinualno, odnosno

$$U = \frac{WT_F}{(T_F+2T_p)(1-P_F+WP_F)}(1-P_F),$$

ako postoje prekidi u slanju.

Zadatak 5.12 Poruka dužine L prenosi se okvirima dužine F. Okviri se sastoje od zaglavlja, dužine H i informacionog dela. Verovatnoća greške po bitu je P_e. Primenjena je ARQ procedura *selektivno ponavljanje*.

a) Izvedite izraz za optimalnu dužinu okvira, tako da ukupan broj prenesenih bita bude minimalan.

b) Izračunajte ovu optimalnu dužinu za $L = 12$ kb, $H = 135$ b, $P_e = 10^{-4}$.

a) Kada ne bi bilo greške, broj okvira potrebnih za slanje poruke iznosio bi

$$k = \left\lceil \frac{L}{F-H} \right\rceil .$$

Za potrebe daljeg rešavanja zadatka, izvršićemo aproksimaciju

$$k \approx \frac{L}{F-H}.$$

Primetimo da je učinjena aproksimacija to tačnija što je poruka duža, a okvir kraći.

Posmatrajmo prenos jednog okvira. Verovatnoća njegovog ispravnog prijema je

$$\mathrm{P}_{CF} = (1-P_e)^F,$$

a pogrešnog

$$\mathrm{P}_F = 1-(1-P_e)^F.$$

Ako se okvir ispravno prenese u i-tom pokušaju (nakon $i-1$ neuspeha), prosečan broj prenesenih bita biće

$$\overline{N} = \sum_{i=1}^{\infty} iF \left[1-(1-P_e)^F\right]^{i-1}(1-P_e)^F.$$

Iz osobina geometrijske raspodele, dobijamo

$$\overline{N} = \frac{F}{(1-P_e)^F}.$$

Prosečan broj prenesenih bita za celu poruku biće

$$\overline{N_{uk}} = k\overline{N},$$

što daje

$$\overline{N_{uk}} = \frac{L}{F-H}\frac{F}{(1-P_e)^F}.$$

Optimalnu dužinu okvira odredićemo tako što ćemo naći izvod prethodnog izraza po F i izjednačiti ga s nulom. Tako ćemo dobiti kvadratnu jednačinu

$$F_{opt}^2 \ln(1-P_e) - F_{opt} H \ln(1-P_e) + H = 0.$$

Njena rešenja su

$$F_{opt\,1,2} = \frac{H \pm \sqrt{H^2 - 4\dfrac{H}{\ln(1-P_e)}}}{2}.$$

Dužina okvira mora biti pozitivna, pa usvajamo rešenje sa znakom „+". Stoga je, konačno

$$F_{opt} = \frac{H + \sqrt{H^2 - 4\dfrac{H}{\ln(1-P_e)}}}{2}.$$

b) Uvrštavanjem zadatih brojčanih vrednosti u prethodnu formulu, dobijamo $F_{opt} = 1231{,}325$ b. Dužina okvira mora biti prirodan broj, pa se postavlja pitanje koju od vrednosti za F_{opt} usvojiti – 1231 b ili 1232 b. Primetimo da je ovaj problem direktna posledica toga što smo na početku izvođenja u tački a), *diskretnu* funkciju aproksimirali *kontinualnom* ($\lceil x \rceil \approx x$). *Tačan* izraz za očekivani broj prenesenih bita je

$$\overline{N_{uk}} = \left\lfloor \frac{L}{F-H} \right\rfloor \frac{F}{(1-P_e)^F} + \frac{L - \left\lfloor \frac{L}{F-H} \right\rfloor (F-H) + H}{(1-P_e)^{L - \left\lfloor \frac{L}{F-H} \right\rfloor (F-H) + H}}.$$

Ako bismo usvojili $F = 1231$ b, za slanje posmatrane poruke trebalo bi $k = \lceil 10{,}949 \rceil = 11$ okvira, od kojih bi 10 bilo potpuno popunjeno sa po $F - H = 1096$ b korisnog sadržaja, dok bi se u poslednjem, jedanaestom, prenelo preostalih 1040 bita poruke, pa bi njegova dužina iznosila 1175 b. Prosečan broj ukupno prenesenih bita tada bi iznosio

$$\overline{N_{uk}} = 10\frac{1231\text{ b}}{(1-P_e)^{1231}} + \frac{1175\text{ b}}{(1-P_e)^{1175}} = 15244{,}17\text{ b}.$$

Ako bismo usvojili veću vrednost, $F = 1232$ b, za slanje posmatrane poruke opet bi trebalo $k = \lceil 10{,}939 \rceil = 11$ okvira. Prvih 10 bilo bi potpuno popunjeno, sa po $F - H = 1097$ b korisnog sadržaja, dok bi se u poslednjem, jedanaestom, prenelo preostalih 1030 bita poruke. Dužina ovoga okvira stoga bi iznosila 1165 b. Prosečan broj ukupno prenesenih bita bio bi

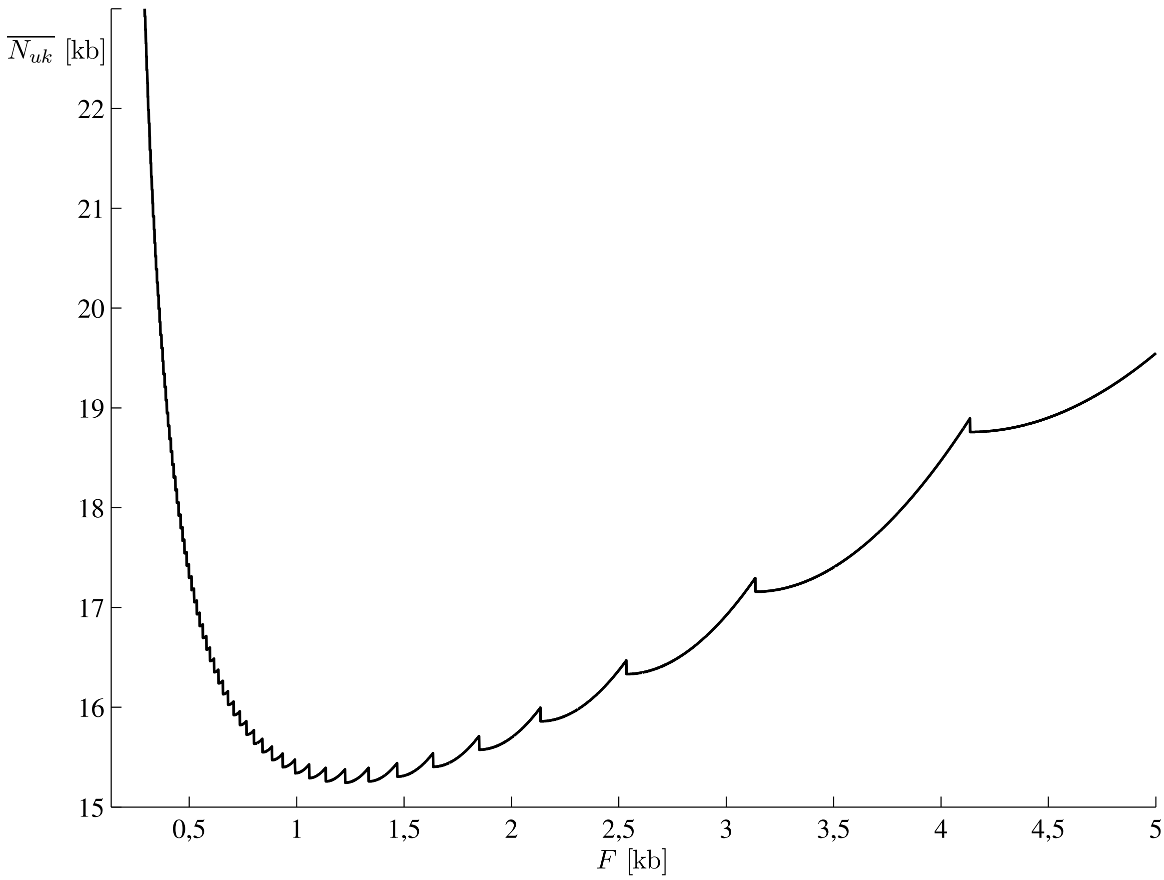
$$\overline{N_{uk}} = 10\frac{1232\text{ b}}{(1-P_e)^{1232}} + \frac{1165\text{ b}}{(1-P_e)^{1165}} = 15244{,}32\text{ b}.$$

Kao što vidimo, bolji izbor je $F = 1231$ b, iako je razlika u performansama neznatna.

Zavisnost broja prenesenih bita od dužine okvira, za numeričke podatke iz postavke zadatka, prikazana je na slici 5.12.

Zadatak 5.13 Verovatnoća pogrešnog prijema okvira na telekomunikacionom linku je $P_F = 10^{-6}$. Odredite iskorišćenosti kanala za sledeće ARQ procedure:

- *stani i čekaj*,
- *vrati se za N*, za $W = 7$ i $W = 127$ i



Slika 5.12: *Zavisnost očekivanog broja prenesenih bita od dužine okvira.*

- *selektivno ponavljanje*, za $W = 7$ i $W = 127$.

Razmotrite slučajeve $T_p/T_F \in \{0{,}1,\ 1,\ 10,\ 100\}$.

Podsetimo se izraza za iskorišćenost kanala u različitim ARQ procedurama.

Za proceduru *stani i čekaj* (SW) je

$$U = \frac{1 - P_F}{1 + 2\dfrac{T_p}{T_F}}.$$

Kod procedura s klizećim prozorom, iskorišćenost će zavisiti i od otvora prozora.

Za proceduru *vrati se za N* (GBN), važiće

$$U = \begin{cases} \dfrac{1 - P_F}{1 + 2\dfrac{T_p}{T_F} P_F}, & W \geqslant 1 + 2\dfrac{T_p}{T_F} \\[2ex] W \dfrac{1 - P_F}{\left(1 + 2\dfrac{T_p}{T_F}\right)(1 - P_F + W P_F)}, & \text{u suprotnom} \end{cases}.$$

Iskorišćenost kanala kod procedure *selektivno ponavljanje* (SR) data je izrazom

$$U = \begin{cases} 1 - P_F, & W \geqslant 1 + 2\dfrac{T_p}{T_F} \\ W\dfrac{1 - P_F}{1 + 2\dfrac{T_p}{T_F}}, & \text{u suprotnom} \end{cases}.$$

Rezultati izračunavanja prikazani su u tabeli.

Iskorišćenosti kanala za različite ARQ procedure.

T_p/T_F	SW	GBN(7)	GBN(127)	SR(7)	SR(127)
0,1	0,83	1	1	1	1
1	0,33	1	1	1	1
10	0,05	0,33	1	0,33	1
100	0,005	0,035	0,63	0,035	0,63

Iz tabele vidimo da performanse procedure „stani i čekaj" brzo opadaju. Procedure s klizećim prozorom su u posmatranom primeru imale znatno bolje performanse, koje su uz to bile i približno jednake. Kod ovih metoda, najbolje performanse se ostvaruju za slučaj kontinualnog slanja ($W \geqslant 1 + 2T_p/T_F$).

Zadatak 5.14 U telekomunikacionoj mreži se koristi ARQ procedura „stani i čekaj". Dužina okvira je slučajna promenljiva, uniformno raspodeljena na intervalu [500 b, 1500 b]. Izdaju se samo pozitivne potvrde, čija je dužina 20 b. Protoci na direktnom i povratnom kanalu su jednaki i iznose 9600 b/s. Verovatnoća greške po bitu iznosi 10^{-4}, na oba kanala. Retransmisioni tajmer se podešava na vreme čekanja $T_{to} = 20$ ms. Trajanje ciklusa slanja je slučajna promenljiva, čija je funkcija raspodele data izrazom

$$F_T(t) = \begin{cases} 1 - \exp[-200(t\,[\mathrm{s}] - 0{,}01)], & t > 10 \text{ ms} \\ 0, & \text{u suprotnom} \end{cases}.$$

a) Odredite verovatnoću uspešnog slanja okvira.

b) Odredite prosečno vreme potrebno da bi se okvir uspešno poslao.

a) Okvir će se uspešno poslati ako i samo ako se budu ispunili sledeći uslovi:

- okvir ispravno stiže do odredišta,
- pozitivna potvrda (ACK) ispravno stiže do izvora i
- izvor prima pozitivnu potvrdu pre isteka vremena čekanja, T_{to}.

Ovi događaji su uzajamno nezavisni, pa je ukupna verovatnoća uspešnog slanja okvira jednaka proizvodu njihovih verovatnoća.

Označimo s L dužinu okvira. Prema postavci zadatka je $L \sim Unif\,[500,\ 1500]$ b. Verovatnoća uspešnog prijema okvira dužine L je

$$\mathrm{P}_{CF}(L) = (1-p)^L,$$

gde je p verovatnoća greške po bitu. Prosečnu verovatnoću uspešnog prijema okvira nalazimo usrednjavanjem po svim dužinama, L:

$$\begin{aligned}\mathrm{P}_{CF} &= \int_{-\infty}^{+\infty} f_L(l)\mathrm{P}_{CF}(l)\,\mathrm{d}l = \\ &= \int_{500}^{1500} \frac{1}{1500-500}\,(1-p)^l\,\mathrm{d}l.\end{aligned}$$

Izračunavanjem integrala, dobijamo $\mathrm{P}_{CF} = 0{,}905$.

Pozitivne potvrde su fiksne dužine $\mu = 20$ b, pa je verovatnoća njihovog uspešnog prijema

$$\mathrm{P}_{ACK} = (1-p)^{\mu} = 0{,}998.$$

Verovatnoća da je vreme potrebno za prenos u oba smera manje od vremena čekanja je

$$\mathrm{P}(T < T_{to}) = F_T(T_{to}) = 0{,}865.$$

Verovatnoća uspešnog slanja okvira konačno je

$$\mathrm{P}_{OK} = \mathrm{P}_{CF}\mathrm{P}_{ACK}\mathrm{P}(T < T_{to}) = 0{,}781.$$

b) Neka je uspešnom prenosu prethodilo k neuspeha, $k = 0, 1, 2, \ldots$ Vreme koje je potrebno za prenos tada je

$$T_k = kT_{NOK} + T_{OK},$$

gde je T_{NOK} vreme potrebno za neuspešan prenos i T_{OK} vreme potrebno za uspešan prenos. Ovaj događaj se realizuje s verovatnoćom $\mathrm{P}_{OK}\,(1-\mathrm{P}_{OK})^k$.

Prema proceduri „stani i čekaj”, izvor emituje okvir, aktivira tajmer i do isteka vremena T_{to} čeka pozitivnu potvrdu. Smatra se da je slanje neuspešno ako do isteka vremena čekanja izvor ne bude dobio ovu potvrdu. Stoga je

$$T_{NOK} = T_F + T_{to},$$

gde je T_F prosečno trajanje okvira,

$$T_F = \int_{-\infty}^{+\infty} f_L(l)\frac{l}{V_b}\,\mathrm{d}l = 104 \text{ ms}.$$

Stoga je $T_{NOK} = 124$ ms.

U slučaju uspeha je

$$T_{OK} = T_F + T_d,$$

gde je T_d prosečno kašnjenje pri prenosu u oba smera:

$$\begin{aligned}
T_d &= \int\limits_{-\infty}^{+\infty} t f_T(t)\,\mathrm{d}t = \\
&= \int\limits_{-\infty}^{+\infty} t\,\frac{\mathrm{d}F_T(t)}{\mathrm{d}t}\,\mathrm{d}t = \\
&= 200 \int\limits_{0{,}01}^{\infty} t \exp[-200(t-0{,}01)]\,\mathrm{d}t.
\end{aligned}$$

Primenom parcijalne integracije, dobijamo rezultat $T_d = 15$ ms, pa je $T_{OK} = 119$ ms.

Prosečno vreme potrebno za uspešno slanje okvira odredićemo usrednjavanjem po broju neuspeha, k:

$$\begin{aligned}
\overline{T} &= \sum_{k=0}^{\infty} \mathrm{P}_{OK}(1-\mathrm{P}_{OK})^k\,T_k = \\
&= \sum_{k=0}^{\infty} \mathrm{P}_{OK}(1-\mathrm{P}_{OK})^k(T_{OK} + kT_{NOK}) = \\
&= \mathrm{P}_{OK}T_{OK}\sum_{k=0}^{\infty}(1-\mathrm{P}_{OK})^k + \mathrm{P}_{OK}T_{NOK}\sum_{k=0}^{\infty}k(1-\mathrm{P}_{OK})^k = \\
&= \mathrm{P}_{OK}T_{OK}\sum_{k=0}^{\infty}(1-\mathrm{P}_{OK})^k + \\
&\qquad + \mathrm{P}_{OK}T_{NOK}\sum_{k=0}^{\infty}(k+1)(1-\mathrm{P}_{OK})^k - \mathrm{P}_{OK}T_{NOK}\sum_{k=0}^{\infty}(1-\mathrm{P}_{OK})^k = \\
&= \frac{\mathrm{P}_{OK}T_{OK}}{1-(1-\mathrm{P}_{OK})} + \frac{\mathrm{P}_{OK}T_{NOK}}{[1-(1-\mathrm{P}_{OK})]^2} - \frac{\mathrm{P}_{OK}T_{NOK}}{1-(1-\mathrm{P}_{OK})} = \\
&= T_{OK} + T_{NOK}\,\frac{1-\mathrm{P}_{OK}}{\mathrm{P}_{OK}}.
\end{aligned}$$

Uvrštavanjem izračunatih vrednosti, konačno dobijamo

$$\overline{T} = 153{,}78 \text{ ms}.$$

Zadatak 5.15 Dve stanice komuniciraju uz korišćenje ARQ procedure „vrati se za N" s maksimalnim otvorom prozora. Normalizovano kašnjenje je 2,5. Za numeraciju okvira na raspolaganju su tri bita. Prosečno se šalju tri okvira do uspeha. S kojom verovatnoćom će uspeh nastupiti u ne više od tri pokušaja slanja?

Otvor prozora je

$$W = W_{max} = 2^n - 1 = 7.$$

Ispitajmo koji je režim rada uspostavljen. Pošto je $W \geq 1 + 2a$, radi se o kontinualnom slanju. Broj okvira koji se reemituju pri svakoj grešci stoga će biti

$$K \approx 1 + 2a = 6.$$

Prosečan broj pokušaja slanja dat je izrazom

$$\mu = \frac{1 + (K-1)P_F}{1 - P_F},$$

odakle dobijamo da je verovatnoća pogrešnog prijema okvira $P_F = \frac{1}{4}$.

Verovatnoća uspeha u ne više od tri pokušaja slanja je

$$\begin{aligned} \mathrm{P}(i \leq 3) &= \mathrm{P}(i = 1) + \mathrm{P}(i = 2) + \mathrm{P}(i = 3) = \\ &= 1 - P_F \;+\; P_F(1 - P_F) \;+\; P_F^2(1 - P_F) = \\ &= 1 - P_F^3 \end{aligned}$$

i iznosi 0,984.

Zadatak 5.16 Dve stanice komuniciraju razmenjujući okvire dužine 1000 b, koji imaju 56 b zaglavlja. Kapacitet linka je 1 Mb/s, a propagaciono kašnjenje iznosi 8 ms. Kolika je iskorišćenost linka ukoliko se koristi ARQ procedura „vrati se za N", s maksimalnim otvorom prozora, pri čemu su za numeraciju okvira na raspolaganju 4 bita, a verovatnoća da uspeh nastupi u ne više od 4 pokušaja je 0,8704?

Prema zadatim podacima, trajanje okvira, tj. vreme potrebno da bi se okvir utisnuo u link je $T_F = 1$ ms, pa je normalizovano kašnjenje $a = 8$.

Kada je na raspolaganju n bita za numeraciju okvira, maksimalni otvor prozora je

$$W_{max} = 2^n - 1$$

što u našem slučaju iznosi 15.

Pošto je $1 + 2a = 17 > W$, slanje je s prekidima. Iskorišćenost linka stoga je data izrazom

$$U = \frac{W(1 - P_F)}{(1 + 2a)(1 - P_F + WP_F)}$$

i da bismo ju proračunali, moramo prvo odrediti verovatnoću neuspeha, P_F. Postavkom zadatka je data verovatnoća da uspešno slanje nastupa u ne više od četiri pokušaja; ona je opisana izrazom

$$\begin{aligned}\mathrm{P}(i \leq 4) &= \mathrm{P}(i=1) + \mathrm{P}(i=2) + \mathrm{P}(i=3) = \mathrm{P}(i=4) = \\ &= 1 - P_F \ + \ P_F(1-P_F) \ + \ P_F^2(1-P_F) + \ P_F^3(1-P_F) = \\ &= 1 - P_F^4.\end{aligned}$$

Nakon odbacivanja mogućnosti koje nemaju fizičkog smisla, odavde dobijamo da je $P_F = 0{,}6$. „Okrugao” rezultat u školskim zadacima uvek je signal dobrog rešenja.

Imamo sve podatake koji su nam potrebni, pa njihovim uvrštavanjem u izraz za iskorišćenost linka dobijamo konačni rezultat $U = 0{,}0375$.

Zadatak 5.17 Okviri dužine $L = 500$ b prenose se kanalom na kome je protok $R = 1$ Mb/s, normalizovano kašnjenje $a = 1{,}3$ i verovatnoća pogrešnog prijema bita $p = 10^{-5}$. Primenjena je ARQ procedura „vrati se za N”. Pretpostavljajući da je generisanje okvira Poissonov slučajni proces, protoka $\lambda = 750\ \mathrm{s}^{-1}$ i da u povratnom smeru ne nastupaju greške (tj. da se pozitivne potvrde uvek primaju ispravno), odredite prosečno zadržavanje okvira u predajnom baferu.

Predajni bafer ćemo modelirati servisnim sistemom M/G/1, pa ćemo prosečno zadržavanje okvira u njemu odrediti primenom Pollaczek-Khinchinove formule.

Verovatnoća pogrešnog prijema okvira je

$$P_F = 1 - (1-p)^L = 4{,}9875 \cdot 10^{-3}.$$

Retransmisioni tajmer se podešava na vrednost

$$T_t = \frac{L}{R}\,(1 + 2a).$$

Verovatnoća da će se posmatrani okvir uspešno preneti nakon k retransmisija je

$$\mathrm{P}_k = P_F^k(1 - P_F),$$

za šta je potrebno vreme

$$T_k = kT_t + \frac{L}{R} = \frac{L}{R}\left(k(1+2a) + 1\right).$$

Prosečno vreme potrebno za uspešan prenos okvira je

$$\mathrm{E}\,T = \sum_{k=0}^{\infty} T_k \mathrm{P}_k = \frac{L}{R}\,\frac{1 + 2aP_F}{1 - P_F} = 5{,}11 \cdot 10^{-4}\ \mathrm{s}.$$

Drugi moment vremena potrebnog za uspešan prenos okvira je

$$
\begin{aligned}
\mathrm{E}\,T^2 &= \sum_{k=0}^{\infty} T_k^2\,\mathrm{P}_k = \\
&= \left(\frac{L}{V}\right)^2 \frac{(1-P_F)^2 + (1+2a)P_F(2(1-P_F) + (1+2a)(1+P_F))}{(1-P_F)^2} = \\
&= 2{,}85 \cdot 10^{-7}\ \mathrm{s}^2.
\end{aligned}
$$

Prosečno vreme koje okvir provede u predajnom baferu konačno je

$$T_Q = \frac{\lambda\,\mathrm{E}\,T^2}{2\,(1-\lambda\,\mathrm{E}\,T)} = 1{,}74 \cdot 10^{-4}\ \mathrm{s}.$$

Zadatak 5.18 Dva čvora telekomunikacione mreže s komutacijom paketa nalaze se na rastojanju $D = 300$ km. Čvorovi su povezani optičkim kablom, na kome je protok $R = 10$ Mb/s. Čvorovi razmenjuju pakete fiksne dužine $Q = 250$ B, koji imaju zaglavlje od $X = 16$ B i za kontrolu toka koriste klizeći prozor, tako da je iskorišćenost kanala na granici maksimalne. Koliko je bita u zaglavlju potrebno odvojiti za označavanje broja paketa u sekvenci? Pretpostavite da se trajanja potvrda prijema i vremena obrade paketa i potvrda mogu zanemariti.

Kod procedure kontrole toka na principu klizećeg prozora, maksimalna iskorišćenost kanala se ostvaruje u kontinualnom režimu slanja; on nastupa za

$$W = 1 + 2a.$$

Iz zadatih brojčanih vrednosti je propagaciono kašnjenje 1,5 ms i trajanje paketa 0,2 ms, pa je normalizovano kašnjenje 7,5, odakle je otvor prozora $W = 16$.

Pri kontroli toka se ne razmatraju greške u prenosu, niti se postavlja pitanje višeznačnosti kumulativnih potvrda; stoga je potreban broj bita za numerisanje W paketa

$$n = \lfloor \log_2 W \rfloor = 4.$$

6. L2 mrežne tehnologije

Zadatak 6.1 Odredite potreban kapacitet bafera za prihvatanje HDLC okvira u uređaju koji koristi protokol X.25, tako da ne dolazi do zastoja u prenosu usled njegove popunjenosti. Veličina otvora prozora je $W = 7$, a maksimalna veličina informacionog dela okvira je 256 B.

Otvor prozora određuje koliko poslatih okvira može biti nepotvrđeno. Ako je veličina otvora prozora W, tada u baferu treba biti mesta za barem W okvira. Ako uz to želimo i da imamo spreman okvir za slanje u trenutku kada bude stigla potvrda ispravnog prijema ranije poslatih okvira, onda u baferu treba biti mesta i za još jedan okvir. Kapacitet bafera stoga treba da iznosi

$$M = (W + 1)L_F,$$

gde je L_F dužina okvira.

Struktura HDLC okvira bez proširenih polja prikazana je na slici.

FLAG	ADR	C	I	FCS	FLAG

Slika 6.1: *Struktura HDLC okvira.*

Dužina okvira jednaka je zbiru veličina informacionog polja i veličina ostalih delova okvira: polja za razgraničenje, adresnog, upravljačkog (kontrolnog) i zaštitnog polja. Po postavci zadatka, maksimalna veličina informacionog dela okvira je 256 B. Veličina svakog polja za razgraničenje (FLAG) je 8 b, koliko iznose i veličine adresnog (ADR) i upravljačkog polja (C). Zaštitno polje (FCS) dvaput je veće i čini ga 16 b. Prema tome, dužina okvira je $L_F = 262$ B $= 2096$ b.

Potreban kapacitet bafera sada je $M = 2096$ B $= 16768$ b.

Zadatak 6.2 U mreži s protokolom X.25, uređaji A (DTE) i B (DCE), komuniciraju uz korišćenje protokola LAPB.

a) Odredite sadržaj okvira koji na liniju veze šalje uređaj A kada izdaje komandu SABM (P = 1).

b) Odredite sadržaj okvira koji uređaj B šalje kao odgovor.

a) Prema preporuci X.25, komanda SABM (*Set Asynchronous Balance Mode*) šalje se tzv. nenumerisanim ili U-okvirom, čija je struktura prikazana na slici.

FLAG	ADR	C	FCS	FLAG

Slika 6.2: *Struktura U-okvira u LAPB.*

Okvir počinje i završava sekvencama za razgraničenje (FLAG). To su sekvence 0111 1110. ADR je adresno polje, C kontrolno polje i FCS polje za kontrolu ciklične redundantnosti. Ukoliko se unutar okvira nađe sekvenca bita koja odgovara sekvenci za razgraničenje, tada se u sadržaj okvira utiskuju dodatne nule.

Pošto DTE uređaj izdaje komandu DCE uređaju, primeniće se tzv. B-adresa, 1000 0000.

Kontrolno polje naredbe SABM je C = 1111 P100. Po postavci zadatka, vrednost bita P (*poll*) je 1, pa je C = 1111 1100.

Vrednost polja FCS određuje se po sledećem obrascu:

$$\mathrm{FCS} = \mathrm{Res}\left\{\frac{x^{16}M(x) + x^m L(x)}{G(x)}\right\} + L(x),$$

gde je M sekvenca dužine m koja odgovara poljima ADR i C, $L(x)$ polinom petnaestog stepena:

$$L(x) = x^{15} + x^{14} + \ldots + x + 1,$$

$G(x)$ generišući polinom CRC-CCITT

$$G(x) = x^{16} + x^{12} + x^5 + 1$$

i $\mathrm{Res}\{\cdot\}$ ostatak pri deljenju. Sve računske operacije se izvode po modulu 2.

U našem slučaju je $M = 1000\,0000\,1111\,1100$ i $m = 16$, pa je

$$M(x) = x^{15} + x^7 + x^6 + x^5 + x^4 + x^3 + x^2.$$

Stoga je

$$x^{16}M(x) + x^m L(x) = x^{30} + x^{29} + \ldots + x^{24} + x^{17} + x^{16},$$

čemu odgovara binarna sekvenca

111 1111 0000 0011 0000 0000 0000 0000.

Ostatak pri deljenju je

$$\mathrm{Res}\left\{\frac{x^{16}M(x)+x^{m}L(x)}{G(x)}\right\}=x^{13}+x^{11}+x^{2},$$

pa je

$$\mathrm{FCS}=x^{15}+x^{14}+x^{12}+x^{10}+\ldots+x^{3}+x+1,$$

ili

$$\mathrm{FCS}=1101\,0111\,1111\,1011.$$

Okvir bi izgledao ovako:

$$\underbrace{0111\,1110}_{\text{FLAG}}\;\underbrace{1000\,000\mathbf{0}}_{\text{ADR}}\;\underbrace{\mathbf{1111\,110}0}_{\text{C}}\;\underbrace{1101\,0\mathbf{111\,1111\,1}011}_{\text{FCS}}\;\underbrace{0111\,1110}_{\text{FLAG}}.$$

Vidimo da se unutar okvira poljavljuje sekvenca za razgraničenje, kao i sekvenca od pet jedinica, što bi na prijemu dovelo do pogrešnog interpretiranja sadržaja. Stoga se u okvir pre slanja utiskuju nule:

$$\underbrace{0111\,1110}_{\text{FLAG}}\;\underbrace{1000\,0000}_{\text{ADR}}\;\underbrace{1111\,1\mathbf{0}100}_{\text{C}}\;\underbrace{1101\,0111\,11\mathbf{0}11\,1011}_{\text{FCS}}\;\underbrace{0111\,1110}_{\text{FLAG}}.$$

b) Uređaj B odgovara nenumerisanom potvrdom, UA (*Unnumbered Acknowledgement*). Ona se takođe prenosi U-okvirom. Ponovo se koristi B-adresa, a kontrolno polje je C = 1100 F110, gde je F = 1. Ponavljajući proceduru iz prethodne tačke, dobijamo da se na liniju veze šalje okvir

$$\underbrace{0111\,1110}_{\text{FLAG}}\;\underbrace{1000\,0000}_{\text{ADR}}\;\underbrace{1100\,1110}_{\text{C}}\;\underbrace{1100\,0001\,1110\,1010}_{\text{FCS}}\;\underbrace{0111\,1110}_{\text{FLAG}}.$$

Čitaocu se preporučuje da proveri prethodni rezultat.

Zadatak 6.3 Odredite maksimalan protok paketa koje može da emituje čvor u Ethernet mreži.

Struktura paketa u Etherenet mreži prikazana je na slici.

PA	SFD	DA	SA	T/L	PL	FCS

Slika 6.3: *Struktura Ethernet paketa prema standardu IEEE 802.3.*

Paket počinje preambulom PA, koja se sastoji od sedam identičnih bajtova 10101010. Preambula prijemniku „najavljuje" dolazak paketa i omogućava mu da sinhronizuje svoj generator takta na takt dolazne povorke bita.

Označavač početka okvira (*Start-of-Frame Delimiter*, SFD) je sekvenca dužine jednog bajta, 10101011.

Adresa odredišta, DA i adresa izvorišta, SA, dužine su po šest bajtova i ukazuju na to kojoj je stanici namenjen paket i koja ga stanica šalje.

Polje T/L (*Type/Length*) označava ili protokol mrežnog sloja, čiji se podaci prenose, ili dužinu polja korisnih podataka. Ovo polje je dužine dva bajta.

Polje korisnog sadržaja (*Payload*, PL) je promenljive dužine, od 46 do 1500 bajtova. Ukoliko je broj korisničkih bajtova koje treba preneti jednim paketom manji od 46, oni će se dopuniti nulama do ovog minimalnog iznosa, što se naziva *padding*.

Polje FCS čine četiri bajta dobijena proverom ciklične redundantnosti, polinomom CRC-32 $(x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1)$.

Minimalan razmak između dvaju Ethernet paketa iznosi 12 bajtova.

Maksimalan protok paketa će se ostvariti onda kada je aktivan jedan čvor (tada neće biti sudara), koji emituje najkraće pakete, ukupne dužine 72 B, praćene razmakom od 12 B. Za protok od 10 Mb/s, odavde se dobija protok paketa 14880,95 s^{-1}, dok u slučaju *fast* Etherneta (100 Mb/s) on iznosi 148809,5 s^{-1}.

Zadatak 6.4 Odredite maksimalni protok koji u Ethernet mreži nudi servis sloja linka podataka.

Maksimalni protok podataka će se ostvariti za pakete s najvećim dozvoljenim poljem korisnog sadržaja, koje emituje jedan čvor, jer tada nema sudara.

Najduži paket se sastoji od 1500 B polja korisnog sadržaja, pa je njegova ukupna dužina 1526 B, uz 12 B razmaka među susednim paketima. Ponavljanjem procedure iz prethodnog zadatka, dobijamo da protok najdužih paketa za 10 Mb/s Ethernet iznosi 812,74 s^{-1}. U svakom ovakvom paketu biće $1500 \cdot 8$ korisnih bita, pa će maksimalni protok korisnih podataka iznositi 9,75 Mb/s.

Zadatak 6.5 Jedan čvor u *fast* Ethernet mreži emituje pakete čije je polje korisnog sadržaja dužine 100 B s protokom $R_1 = 100\ \mathrm{s}^{-1}$, a drugi pakete za koje je polje korisnog sadržaja dužine 1000 B s protokom $R_2 = 20\ \mathrm{s}^{-1}$. Kolika je iskorišćenost ove mreže?

Pretpostavićemo da broj sudara nije značajan, što je opravdano sve dok je iskorišćenost mreže manja od 10%.

Dužina paketa koje emituje prvi čvor iznosi $L_1 = 126$ B, a onih koje emituje drugi čvor $L_1 = 1026$ B. Tokom vremena T, oba čvora će emitovati ukupno N bita, gde je

$$N = T\,(L_1R_1 + L_2R_2) \cdot 8 \text{ b}.$$

Nominalan broj bita koji bi se za isto vreme mogao preneti s protokom 100 Mb/s je $N_{nom} = T \cdot 100$ Mb/s. Iskorišćenost ove mreže stoga je

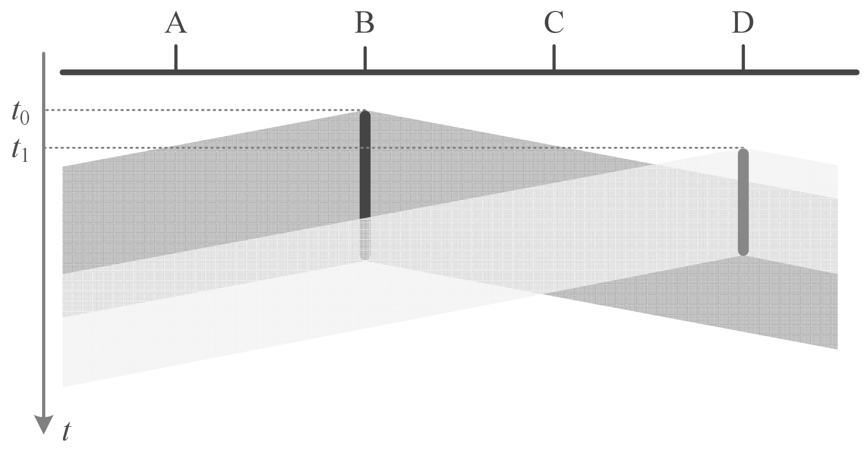
$$\eta = \frac{N}{N_{nom}} \cdot 100\% = 0{,}28\%.$$

Vidimo da je ovaj rezultat doista manji od 10%, pa je polazna pretpostavka o zanemarivom broju sudara bila opravdana.

Zadatak 6.6 Odredite maksimalni domet (*cable span*) Ethernet mreže realizovane prema standardu 10BASE2.

Prema standardu 10BASE2 za tzv. *thin Ethernet*, primenjena je topologija magistrale, uz proceduru CSMA/CD. Sredina za prenos je koaksijalni kabl karakteristične impedanse 50 Ω, dok je protok na linku 10 Mb/s.

Maksimalni domet mreže odredićemo iz uslova pouzdane detekcije sudara. Posmatrajmo ovakvu mrežu, koja je ilustrovana na slici. Neka je u trenutku t_0 stanica B otpočinje emitovanje paketa.



Slika 6.6: *Sudar u mreži 10BASE2.*

Druga stanica (D) osluškuje magistralu u trenutku t_1, kada do nje još nije došao signal stanice B. Stanica D zaključuje da je magistrala slobodna i počinje emitovati svoj paket, koji će se uskoro sudariti s paketom stanice B.

Paket koji je stanica D počela emitovati dolazi do stanice B u trenutku $t_1 + t_{p,BD}$, gde je $t_{p,BD}$ propagaciono kašnjenje između stanica B i D. Da bi stanica B zaključila da se njen paket sudario, mora ga emitovati i u trenutku $t_1 + t_{p,BD}$. Odavde zaključujemo da je po pitanju detekcije sudara najnepovoljnija situacija u kojoj se sudaraju najudaljenije stanice na magistrali, koje emituju najkraće pakete, pri čemu druga stanica počinje emitovati neposredno pre pristizanja paketa prve. Ukoliko se sudar može detektovati tada, moći će se detektovati i u svakoj drugoj situaciji.

Ako je maksimalni domet mreže – tj. maksimalna dužina magistrale – L, propagaciono kašnjenje s kraja na kraj biće L/v, gde je $v \approx 200\,000$ km/s brzina propagacije signala. Dužina najkraćeg paketa iznosi 72 B, pa je njegovo trajanje $T_{min} = 57{,}6$ µs. Iz uslova

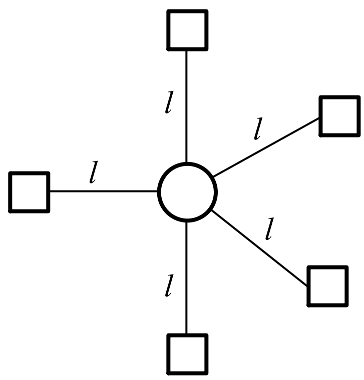
$$T_{min} \geqslant 2\frac{L_{max}}{v}$$

dobijamo $L_{max} = 5{,}76$ km.

Zadatak 6.7 Ispitajte može li se uvek detektovati sudar u nekomutiranoj mreži realizovanoj prema standardu 100BASE-TX, u kojoj je svaki kabl dužine po 85 m.

Parametri standarda 100BASE-TX su:

- koristi se topologija zvezde (*star*),
- sredina za prenos su UTP/FTP/SSTP kablovi, na kojima je brzina propagacije signala $v \approx \frac{2}{3}c$,
- protok na linku je $V = 100$ Mb/s.



Slika 6.7: *Topologija mreže 100BASE-TX.*

Dodatno, pošto je mreža nekomutirana, kao centralni čvor – mrežni stožer – koristi se *hub*.

U ovakvoj mreži, uočimo dve stanice, A i B i razmotrimo sledeći scenario dešavanja:

- u trenutku $t = 0$, stanica A počinje emitovati paket,
- u trenutku $t = t_p = l/v$, ovaj paket stiže u centralni čvor,
- u trenutku $t = 2t_p - 0$, stanica B osluškuje sredinu za prenos, zaključuje da je slobodna i počinje emitovati svoj paket,

- u trenutku $t = 2t_p$, paket stanice A stiže do ostalih stanica u mreži,
- u trenutku $t = 2t_p + 0$, stanica B postaje svesna sudara i prekida emitovanje,
- u trenutku $t = 4t_p$, ono što je stanica B emitovala stiže i do ostalih stanica u mreži.

Iz ove analize vidimo da će stanica A biti svesna sudara ako i samo ako u trenutku $t = 4t_p$ i dalje bude emitovala svoj (sudareni) paket. Prema tome, po pitanju detekcije sudara, kritični su najkraći paketi koji se razmenjuju u mreži. Njihova dužina u Ethernetu je 72 B, pa je njihovo trajanje u mreži 100BASE-TX 5,76 µs. S druge strane, propagaciono kašnjenje na jednom linku je $t_p = 4{,}25 \cdot 10^{-7}$ s. Kritični vremenski interval $4t_p$ iznosi 1,7 µs i kraći je od trajanja najkraćeg paketa, pa se sudar u ovoj mreži uvek može detektovati. Primetimo da ovaj rezultat ne zavisi od broja stanica u mreži.

Zadatak 6.8 Kada stanice u Ethernet mreži detektuju sudar, povlače se na neko vreme. Standardom IEEE 802, definisan je binarni eksponencijalni algoritam čekanja na sledeći način. Pre n-tog pokušaja emitovanja paketa, treba čekati r slotova, pri čemu je r prirodan broj, uniformno raspodeljen na intervalu $\left[0, 2^K\right)$, gde je $K = \min(n, 10)$. Trajanje slota približno je jednako dvostrukom vremenu potrebnom za prenos u oba smera. Neka dve stanice uvek imaju pakete za slanje. Koliko iznosi prosečan broj pokušaja slanja jedne stanice nakon sudara?

Neuspeh u j-tom pokušaju, $j = 1, 2, \ldots$ nastupa onda kada posmatrana stanica izabere da čeka isti broj slotova kao i stanica s kojom se prethodno sudarila. Pošto je broj slotova koje treba čekati uniformno raspodeljen na intervalu $\left[0, 2^K\right)$, verovatnoća ovoga događaja data je izrazom

$$P_j^* = \frac{1}{2^K},$$

gde je $K = \min(j, 10)$. Primetimo da ona zavisi od broja prethodnih pokušaja.

Združena verovatnoća uspeha iz i-tog pokušaja, nakon $i-1$ neuspešnih je

$$P_i = (1 - P_i^*) \prod_{j=1}^{i-1} P_j^*.$$

Prosečan broj pokušaja biće jednak matematičkom očekivanju ove raspodele (tj. slučajne promenljive I):

$$\mathrm{E}I = \sum_{i=1}^{\infty} i P_i.$$

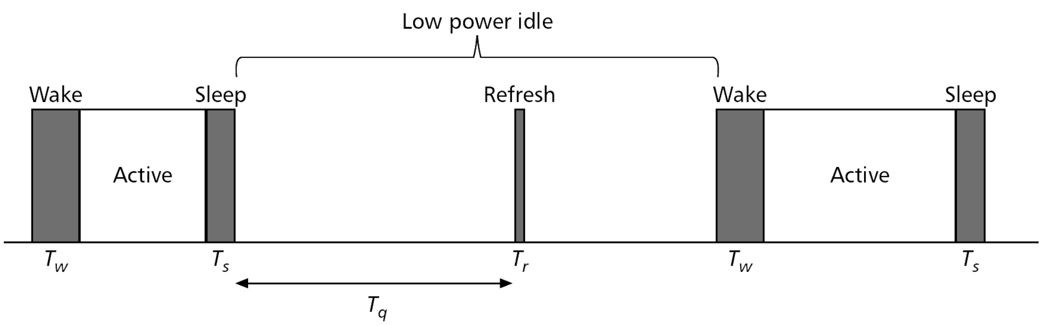
Rezultat nije moguće napisati u zatvorenoj formi, pa ćemo do njega doći numeričkim putem. Tabelirajmo red iP_i, pri čemu dobijamo rezultate koji su prikazani na narednoj strani.

Uz izračunavanje prosečnog broja pokušaja slanja paketa.

i	P_i	iP_i
1	$\frac{1}{2}$	0,5
2	$\frac{3}{4}\frac{1}{2}$	0,75
3	$\frac{7}{8}\frac{1}{2}\frac{1}{4}$	0,33
4	$\frac{15}{16}\frac{1}{2}\frac{1}{4}\frac{1}{8}$	0,059
5	$\frac{31}{32}\frac{1}{2}\frac{1}{4}\frac{1}{8}\frac{1}{16}$	0,0047

Članovi reda brzo opadaju, pa je za izračunavanje sume dovoljno uzeti samo nekoliko prvih sabiraka. Dobijamo da prosečan broj pokušaja slanja iznosi 1,64.

Zadatak 6.9 Na slici je, kao podsetnik, dat primer događanja na energetski efikasnom Ethernet linku, s označenim vremenskim intervalima. Minimalne vrednosti parametara T_s i T_w navedene su u tabeli. Odredite iskorišćenost ovoga linka za slučajeve slanja najkraćih i najdužih paketa.



Slika 6.9: *Stanja EEE linka (izvor: IEEE).*

standard	T_s [µs]	T_w [µs]
100BASE-TX	200	30
1000BASE-T	182	16,5
10GBASE-T	2,88	4,48

Iskorišćenost EEE linka jednaka je odnosu vremena potrebnog da bi se paket utisnuo u link i ukupnog vremena koje treba da bi se predajnik aktivirao, emitovao paket i potom ponovo vratio u neaktivno stanje,

$$U = \frac{T_F}{T_w + T_F + T_s}.$$

U svim razmatranim standardima, dužine paketa su u opsegu od 72 B do 1526 B. Rezultati proračuna pregledno su prikazani u tabeli na narednoj strani.

Iskorišćenost EEE linka.

standard	najkraći paket	najduži paket
100BASE-TX	2,4%	34,7%
1000BASE-T	0,3%	5,8%
10GBASE-T	0,78%	12,2%

Vidimo da je prednost na strani dužih paketa. Čime to objašnjavate?

Zadatak 6.10 Efikasnost registracije ONU uređaja u mrežama realizovanim prema standardu IEEE 802.3av definiše se kao odnos vremena efektivno potrebnog za prenos nesudarenih poruka `REGISTER_REQ` i trajanja prozora pronalaženja. Odredite efikasnost registracije u ovakvoj mreži koja se sastoji od 64 ekvidistantno raspoređena ONU uređaja, u kojoj je dužina „feeder" segmenta kabla 2 km, dužina „dropa" 0,5 km, maksimalno slučajno čekanje pre slanja poruke `REGISTER_REQ` 150 µs i verovatnoća uspešne registracije 0,15. Ukupno vreme potrebno za slanje poruke `REGISTER_REQ` sa svim zaštitnim vremenskim intervalima, prema standardu nije duže od 2,2912 µs.

Broj nesudarenih ONU je NP_{OK}, gde je N broj ONU u mreži i P_{OK} verovatnoća njihove uspešne registracije. Ako je trajanje poruke `REGISTER_REQ`, vreme koje je efektivno potrebno za prenos ovih poruka koje se nisu sudarile biće $NP_{OK}T_{\texttt{REGISTER\_REQ}}$.

Prozor pronalaženja mora omogućiti registraciju i najudaljenije ONU, koja bira najduže čekanje. Njegovo trajanje stoga je $t_{rtt} + T_w + T_{\texttt{REGISTER\_REQ}}$. U ovome izrazu, t_{rtt} je dvostruko propagaciono kašnjenje (s kraja na kraj),

$$t_{rtt} = 2\,\frac{l_{feeder} + l_{drop}}{v},$$

gde su l_{feeder} i l_{drop} redom dužine „feeder" i „drop" segmenata kabla i $v = \frac{2}{3}c$ brzina propagacije signala; T_w je maksimalno slučajno čekanje koje ONU bira pre slanja poruke `REGISTER_REQ`.

Imajući ovo u vidu, izraz za efikasnost registracije možemo napisati kao

$$U = \frac{NP_{OK}}{t_{rtt} + T_w + T_{\texttt{REGISTER\_REQ}}} T_{\texttt{REGISTER\_REQ}}.$$

Uvrštavanjem zadatih brojčanih vrednosti, dobijamo da je $U = 12,4\%$.

7. Mrežni sloj

Zadatak 7.1 Za adresne klase A, B i C u IPv4, odredite broj bita namenjenih označavanju mreže, broj bita namenjenih označavanju hosta, broj dozvoljenih mreža, broj dozvoljenih hostova u mreži i celobrojni opseg vrednosti prvog okteta.

Adrese klase A počinju bitom 0. Za označavanje mreže namenjeno je 7 bita, dok se preostala 24 koriste za označavanje hosta. Mrežne adrese kod kojih je prvi oktet 0 (0000 0000) ili 127 (0111 1111) su rezervisane, pa u klasi A preostaje 126 dozvoljenih adresa mreže. S 24 bita, moguće je označiti $2^{24} = 16\,777\,216$ hostova. Celobrojni opseg vrednosti prvog okteta adresa klase A je od 1 do 126.

Adrese klase B počinju bitima 10. Za označavanje mreže namenjeno je 14 bita, a za označavanje hosta 16. U ovoj klasi ima $2^{14} = 16\,384$ adresa mreže i $2^{16} = 65\,536$ hostova. Opseg vrednosti prvog okteta je od 128 (1000 0000) do 191 (1011 1111).

U klasi C, adrese počinju bitima 110. Na raspoloaganju je 21 bit za označavanje mreže i 8 bita za označavanje hosta; stoga je broj mogućih mreža $2^{21} = 2\,097\,152$, a hostova $2^{8} = 256$. Prvi oktet uzima vrednost iz celobrojnog opsega od 192 (1100 0000) do 223 (1101 1111).

Ovi rezultati su pregledno prikazani u tabeli.

Parametri adresnih klasa u IPv4.

Klasa	A	B	C
Broj bita za mrežu	7	14	21
Broj bita za hostove	24	16	8
Broj mreža	126	16 384	2 097 152
Broj hostova	16 777 216	65 536	256
Prvi oktet	1–126	128–191	192–223

Napomenimo da postoje i tzv. višedifuzne (*multicast*) adrese, klase D, koje počinju bitima 1110. Adrese klase E, koje počinju bitima 1111, rezervisane su za buduću upotrebu.

Čitaocu se prepušta da odredi koliki procenat IPv4 adresnog prostora zauzima svaka od navedenih klasa.

Zadatak 7.2 IP adresa uređaja je 172.45.116.136, dok je maska njegove podmreže 255.255.224.0. Koja je adresa podmreže kojoj pripada ovaj uređaj?

Da bismo odredili adresu podmreže, predstavićemo IP adresu uređaja i masku u binarnom obliku i zatim nad njima izvršiti operaciju bit po bit konjunkcije (logičko I).

Binarni prikaz IP adrese je

$$10101100.00101101.01110100.10001000,$$

dok je maska

$$11111111.11111111.11100000.00000000.$$

Izvršavanjem operacije I, dobijamo binarni zapis mrežne adrese

$$10101100.00101101.01100000.00000000.$$

Njemu odgovara decimalni oblik 172.45.96.0.

Zadatak 7.3 Odredite broj podmreža i broj hostova za masku 255.255.255.192, u mreži čija je adresa 195.186.100.0.

U binarnoj notaciji, adresa mreže je

$$11000011.10111010.01100100.00000000,$$

dok je maska

$$11111111.11111111.11111111.11000000.$$

Prepoznajemo da se radi o adresi klase C, u kojoj je nominalno moguće 256 hostova (zadatak 7.1). Zbog korišćenja maske, dva bita se koriste za označavanje podmreže (moguće su četiri), dok za označavanje hosta unutar svake od njih preostaje 6 bita. Broj hostova u svakoj podmreži stoga je $2^6 = 64$.

Zadatak 7.4 Koliko je hostova moguće adresirati sledećim opsezima IP adresa:

a) 191.186.120.0/24,

b) 191.186.120.0/23,

c) 191.186.120.0/25?

U adresama iz postavke zadatka, primenjena je CIDR (*Classless Interdomain Routing*) notacija. Broj iza kose crte označava broj uzastopnih bita 1 u maski.

a) Maska se sastoji od 24 bita 1, pa je oblika 255.255.255.0. Za označavanje hosta, na raspolaganju je čitav poslednji oktet adrese, pa je moguće 256 adresa.

b) Maska je 255.255.254.0. S devet bita, može se adresirati 512 hostova.

c) U ovom slučaju, maska je 255.255.255.128. Sa sedam raspoloživih bita, može se označiti 128 hostova.

Zadatak 7.5 Na ruter su povezane tri podmreže, koje koriste adrese iz opsega 23.1.17.0/24. Podmreža 1 treba da podrži do 64 hosta, podmreža 2 106, a podmreža 3 15. Navedite primere adresa u CIDR notaciji koje zadovoljavaju ove uslove.

Primeri adresnih opsega koji zadovoljavaju postavljene uslove su: za podmrežu 1 – 223.1.17.0/26; za podmrežu 2 – 223.1.17.128/25 i za podmrežu 3 – 223.1.17.192/28.

Zadatak 7.6 TCP segment veličine 4480 B prenosi se Ethernet mrežom. Koliko je IPv4 paketa potrebno za prenos ovoga segmenta i kolike su vrednosti polja `Total Length`, `More Flag` i `Fragment Offset` u njima?

Maksimalna veličina polja korisnog sadržaja u Ethernet paketu iznosi 1500 B, pa je posmatrani segment neophodno izdeliti na fragmente. U svaki Ethernet paket može se smestiti 20 B zaglavlja IPv4 paketa i do 1480 B korisnog sadržaja. Pošto je potrebno da veličina polja korisnih podataka u svim IP fragmentima sem poslednjeg bude umnožak 64 bita (8 B), izabraćemo upravo vrednost 1480 B. Da bismo preneli 4480 B podataka, u tom slučaju će nam trebati 3 ovakva Ethernet paketa, u koje će se smestiti ukupno 4440 B podataka i još jedan, za preostalih 40 B podataka (i još 20 B IPv4 zaglavlja).

Sadržaj relevantnih polja IP fragmenata prikazan je u tabeli.

Vrednosti polja u zaglavljima prenošenih fragmenata.

Fragment	1	2	3	4
`Total Length`	1500	1500	1500	60
`More Flag`	1	1	1	0
`Fragment Offset`	0	185	370	555

Zadatak 7.7 Odredite vrednost polja kontrolnog zbira za IP zaglavlje heksadecimalnog oblika 45 00 05 D4 00 05 20 17 14 06 00 00 7D F5 A8 01 7D F5 A8 0C.

Polje kontrolnog zbira (IP *checksum*) zauzima 11. i 12. bajt zaglavlja IPv4 paketa. Prema dokumentu RFC 1071, vrednost ovog polja se računa tako što se zaglavlje posmatra kao sekvenca 16-bitskih reči, koje se potom saberu u sistemu s komplementom jedinice. Pri izračunavanju vrednosti kontrolnog zbira, smatra se da je sadržaj ovoga polja jednak nuli (zbir se računa za sva ostala polja). Konačna vrednost polja dobija se kao komplement jedinice izračunatog zbira.

Pri proveri na prijemu, zbir u sistemu s komplementom jedinice računa se za sva polja zaglavlja. Provera je uspešna ako rezultat budu bile sve jedinice.

U sistemu s komplementom 1, brojeve jednostavno sabiramo na sledeći način. Izračunamo prvo sumu u sistemu s komplementom 2:

$$4500 + 5\mathrm{D}4 + 5 + 2017 + 1406 + 7\mathrm{DF}5 + \mathrm{A}801 + 7\mathrm{DF}5 + \mathrm{A}80\mathrm{C} = 2\,\mathrm{CAED}.$$

Da bismo rezultat izrazili u sistemu s komplementom 1, sabraćemo prenos (2) sa 16--bitskim rezultatom (CAED). Tako ćemo dobiti CAEF. Ostaje još da odredimo komplement 1 ovog međurezultata, čime dobijamo konačan rezultat polja IP *checksum* 3510.

Zaglavlje posmatranog paketa imalo bi oblik

45 00 05 D4 00 05 20 17 14 06 35 10 7D F5 A8 01 7D F5 A8 0C.

Čitaocu se prepušta da potvrdi ovaj rezultat izračunavanjem kontrolnog zbira na prijemu.

Zadatak 7.8 Pošto se u ruterima menjaju vrednosti nekih polja u zaglavlju IP paketa, potrebno je izračunati i nove vrednosti polja IP *checksum*. Pokažite na koji se način može izračunati njegova nova vrednost, ukoliko su poznate stara vrednost, C i promena parametara zaglavlja, Z.

Neka se zaglavlje sastoji od M 16-bitskih blokova, $W_1, W_2, \ldots, W_M$. Prema objašnjenju iz prethodnog zadatka, vrednost polja IP *checksum* tada je

$$C = \,\sim (W_1 +_1 W_2 +_1 \cdots +_1 W_M)\,,$$

gde je $+_1$ sabiranje, a $\sim$ komplementiranje u sistemu s komplementom 1.

Neka se sada vrednost jednog od blokova W_i, $i \in 1, 2, \ldots M$ promeni za Z. Nova vrednost polja C tada će biti

$$\begin{aligned} C' &= \,\sim (W_1 +_1 W_2 +_1 \cdots +_1 W_M +_1 Z) = \\ &= C +_1 \sim Z. \end{aligned}$$

Na ovaj način, ruteri mogu jednostavnije i brže izračunati novu vrednost polja kontrolne sume.

Zadatak 7.9 Sledeće IPv6 adrese zapišite u sažetom obliku:

a) 1080:0000:0000:0000:0008:0800:200C:417A,

b) FF01:0:0:0:0:0:0:101,

c) 0:0:0:0:0:0:0:1,

d) 0:0:0:0:0:0:0:0,

e) 0:0:0:0:0:FFFF:129.144.52.38 i

f) 12AB:0000:0000:CD30:0000:0000:0000:0000.

IPv6 adrese su dužine 128 bita. Uobičajeno je da se zapisuju tako što se grupe od po 16 uzastopnih bita izraze u heksadecimalnom obliku i razdvoje dvotačkama.

Vodeće nule u pojedinačnim 16-bitskim poljima IPv6 adrese mogu se izostaviti. Dodatno sažimanje zapisa moguće je ukoliko IPv6 adresa sadrži dugačak niz nula. Tada se koristi simbol „::", koji zamenjuje jednu ili više grupa od po 16 bita 0. Ovaj simbol se u adresi može upotrebiti samo jednom.

a) Ovo je primer „unicast" adrese, čiji je sažeti zapis 1080::8:800:200C:417A.

b) Sada se radi o „multicast" adresi, koja se može zapisati kao FF01::101.

c) Ovo je „loopback" adresa, koja se skraćeno zapisuje kao ::1.

d) Adresa „sve nule" se naziva nespecificiranom i skraćeno zapisuje kao ::. Ona se nikada ne sme dodeliti uređaju. Koriste je npr. hostovi u fazi inicijalizacije, pre nego što dobiju svoju adresu.

e) Skraćeni zapis je ::FFFF:129.144.52.38. Ovo je primer IPv6 adrese koja sadrži IPv4 adresu.

f) Ovde su moguća dva sažeta zapisa: 12AB::CD30:0:0:0:0 i 12AB:0:0:CD30::. Primetimo da zapisi 12AB:0:0:CD3, 12AB::CD30 ili 12AB::CD3 nisu ispravni.

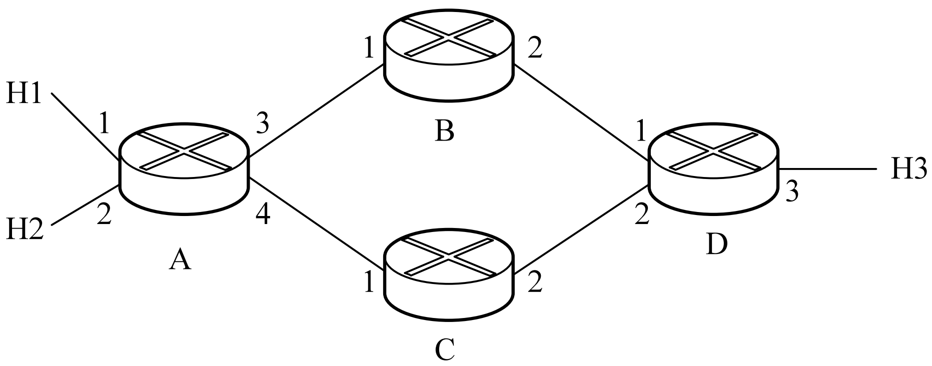
Zadatak 7.10 Kojoj podmreži pripada računar čije su IPv6 adresa i maska oblika 12AB:0:0:CD30:123:4567:89AB:CDEF/60?

CIDR notacija se može primeniti i na IPv6 adrese, pri čemu i ovde broj iza kose crte označava broj uzastopnih bita 1 u maski. Adresa podmreže kojoj pripada ovaj računar je 12AB:0:0:CD30::.

Zadatak 7.11 Data je mreža sa slike.

a) Konstruišite tabelu prosleđivanja u ruteru A, tako da se sav saobraćaj ka hostu H3 šalje preko interfejsa 3.

b) Može li se ova tabela konstruisati tako da se saobraćaj od H1 ka H3 šalje preko interfejsa 3, a saobraćaj od H2 ka H3 preko interfejsa 4?



Slika 7.11: *Primer računarske mreže.*

a) U praksi, tabele prosleđivanja (rutiranja) najčešće ne sadrže informaciju o kompletnoj putanji ka traženom odredištu, već samo o narednom čvoru na njoj.

Da bi se saobraćaj ka odredištu H3 prosleđivao na odlazni interfejs 3, tabela prosleđivanja u ruteru A treba da izgleda na sledeći način:

Odredište	Odlazni interfejs
H3	3

b) To nije moguće, jer se rutiranje zasniva isključivo na adresi odredišta.

Zadatak 7.12 U mreži u kojoj su „cene" svih linkova 1, poznate su tabele prosleđivanja za čvorove A i F. Nacrtajte graf ove mreže.

A:

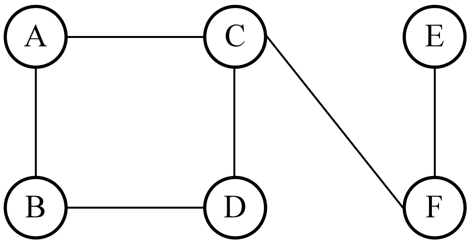
Odredište	Cena	Naredni čvor
B	1	B
C	1	C
D	2	B
E	3	C
F	2	C

F:

Odredište	Cena	Naredni čvor
A	2	C
B	3	C
C	1	C
D	2	C
E	1	E

Posmatranjem datih tabela, zaključujemo da je čvor A direktno povezan sa čvorovima B i C, dok je čvor F direktno povezan sa čvorovima C i E. Čvor D povezan je sa čvorovima B i C.

Jedan graf mreže koji zadovoljava ove uslove prikazan je na slici 7.12. Podatak o „ceni” putanje za sada nam nije od interesa.



Slika 7.12: *Graf telekomunikacione mreže.*

Zadatak 7.13 Ruter ima četiri interfejsa na koje prosleđuje pakete prema tabeli.

Opseg odredišnih adresa	Interfejs
od 1110 0000 0000 0000 0000 0000 0000 0000 do 1110 0000 0000 0000 1111 1111 1111 1111	0
od 1110 0000 0000 0001 0000 0000 0000 0000 do 1110 0000 0000 0001 1111 1111 1111 1111	1
od 1110 0000 0000 0010 0000 0000 0000 0000 do 1110 0001 1111 1111 1111 1111 1111 1111	2
inače	3

a) Napišite tabelu prosleđivanja po principu poklapanja s najdužim prefiksom.

b) Objasnite kuda se prosleđuju paketi s odredišnim adresama

$$1111\,1000\,1001\,0001\,0101\,0001\,0101\,0101,$$

$$1110\,0000\,0000\,0000\,1100\,0011\,0011\,1100$$

i

$$1110\,0001\,1000\,0000\,0001\,0001\,0111\,0111.$$

a) Tražena tabela izgleda ovako:

Prefiks	Interfejs
1110 0000 0000 0000	0
1110 0000 0000 0001	1
1110 000	2
inače	3

Vidimo da je ovo značajno kompaktniji i pregledniji zapis.

b) Da bismo odredili na koji se interfejs rutira paket, potražićemo s kojim se od datih prefiksa adresa njegovog odredišta poklapa u najvećoj meri. Prva adresa se ne poklapa s navedenim prefiksima, pa se ovaj paket šalje na interfejs 3. Drugoj adresi odgovara prvi prefiks, pa se paket šalje na interfejs 0. Trećoj adresi odgovara treći prefiks, pa se ovaj paket šalje na interfejs 2.

Zadatak 7.14 Data je tabela prosleđivanja u mreži u kojoj se koriste osmobitne adrese. Navedite opsege adresa koji odgovaraju svakom od interfejsa i odredite koliko adresa ima u svakome od njih.

Prefiks	Interfejs
00	0
01	1
100	2
inače	3

Interfejsu 0 odgovaraju adrese od 0000 0000 do 0011 1111, kojih ima $2^6 = 64$.

Interfejsu 1 odgovaraju adrese od 0100 0000 do 0111 1111, kojih takođe ima 64.

Interfejsu 2 odgovaraju adrese od 1000 0000 do 1001 1111, kojih ima $2^5 = 32$.

Konačno, adrese od 1010 0000 do 1111 1111 odgovaraju interfejsu 3 i ima ih $2^6+2^5 = 96$.

Zadatak 7.15 Data je tabela prosleđivanja rutera.

Odredišna podmreža	Maska	Prosledi na
128.96.170.0	255.255.254.0	Interfejs 0
128.96.168.0	255.255.254.0	Interfejs 1
128.96.166.0	255.255.254.0	R2
128.96.164.0	255.255.252.0	R3
(podrazumevana)		R4

Ako ruter primenjuje poklapanje s najdužim prefiksom, odredite kuda se prosleđuju paketi čije su odredišne adrese:

a) 128.96.171.92,

b) 128.96.167.151,

c) 128.96.163.151,

d) 128.96.169.192 i

e) 128.96.165.121.

Na svaku od odredišnih adresa primenićemo maske iz tabele prosleđivanja i ispitati poklapa li se rezultat s nekom od adresa podmreža iz prve kolone. Ako nema poklapanja, paket će se proslediti ka podrazumevanom narednom čvoru, a to je ruter R4.

a) Maska 255.255.254.0 daje rezultat 128.96.170.0, pa se paket šalje na interfejs 0.

b) Maska 255.255.254.0 daje 128.96.166.0, što znači da bi naredni čvor bio R2. Valjan rezultat se dobija i za masku 255.255.252.0 (128.96.164.0), pa bi se paket mogao proslediti i ka R3. Pošto duži prefiks ima prednost, paket će se proslediti ka R2.

c) Nijedna od maski ne daje valjan rezultat, pa se paket prosleđuje ka R4.

d) Primenom maske 255.255.254.0, dobijamo rezultat 128.96.168.0, pa se paket šalje na interfejs 1.

e) Maska 255.255.252.0 daje rezultat 128.96.164.0, pa se paket prosleđuje ka R3.

Zadatak 7.16 Data je tabela prosleđivanja rutera, korišćenjem CIDR notacije.

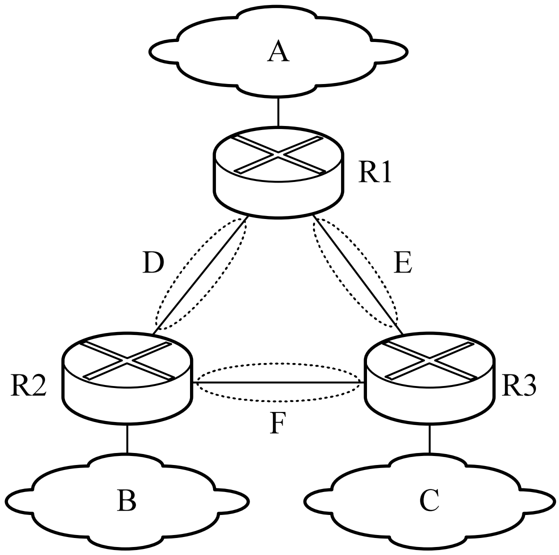
Mreža/Maska	Prosledi ka
C4.5E.2.2/23	A
C4.5E.4.0/22	B
C4.5E.C0.0/19	C
C4.5E.40.0/18	D
C4.4C.0.0/14	E
C0.0.0.0/2	F
80.0.0.0/1	G

Ako ruter primenjuje poklapanje s najdužim prefiksom, odredite kuda se prosleđuju paketi čije su odredišne adrese C4.4B.31.2E, C4.5E.05.09, C4.4D.31.2E, C4.5E.03.87 i C4.5E.D1.02.

Primenićemo postupak iz zadatka 7.13, s tim što su sada adrese date u heksadecimalnom obliku i što je maska označena u CIDR. Posmatrani paketi tako će se redom slati na izlaze F, B, E, A i C.

Zadatak 7.17 Na slici 7.17, prikazana je mreža koja se sastoji od šest podmreža, označenih slovima A-F.

a) Dodelite ovim mrežama adrese iz opsega 214.97.254.0/23, tako da se u podmreži A omogući adresiranje 250 interfejsa, u podmrežama B i C po 120, a u D, E i F po 2.



Slika 7.17: *Struktura mreže.*

b) Za ovako dodeljene adrese, konstruišite tabele prosleđivanja u ruterima R1-3.

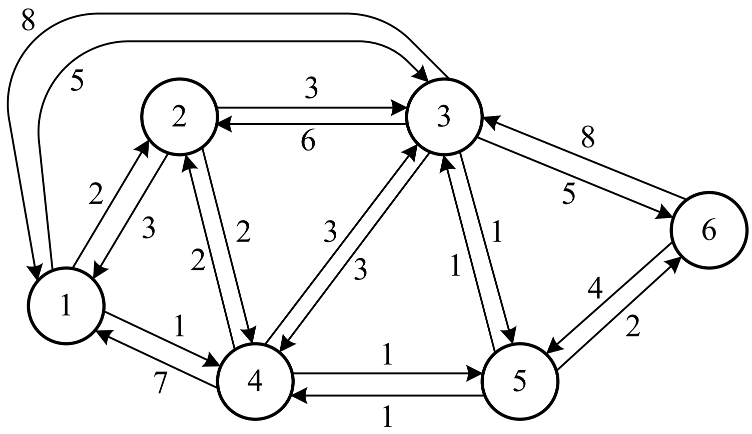
a) Jedan raspored adresa bio bi sledeći: podmreži A se dodeljuje opseg 214.97.255.0/24 u kome ima 256 adresa; podmreži B se dodeljuju odrese od 214.97.254.0/25 do 214.97.254.0/29, kojih ima $128 - 8 = 120$; adrese u podmreži C biće iz opsega 214.97.254.128/25 i ima ih 128. Konačno, podmreži D se dodeljuju dve adrese iz opsega 214.97.254.0/31, podmreži E takođe dve adrese, iz opsega 214.97.254.2/31, dok podmreža F na raspolaganju ima četiri adrese iz opsega 214.97.254.4/30.

b) Tabele prosleđivanja date su u nastavku.

Ruter	Prefiks	Prosledi ka
R1	1101 0110 0110 0001 1111 1111	A
	1101 0110 0110 0001 1111 1110 0000 000	D
	1101 0110 0110 0001 1111 1110 0000 001	E
R2	1101 0110 0110 0001 1111 1110 0	B
	1101 0110 0110 0001 1111 1110 0000 000	D
	1101 0110 0110 0001 1111 1110 0000 01	F
R3	1101 0110 0110 0001 1111 1110 1	C
	1101 0110 0110 0001 1111 1110 0000 001	E
	1101 0110 0110 0001 1111 1110 0000 01	F

Zadatak 7.18 Na slici je dat graf telekomunikacione mreže. Uz svaki link, označena je i njegova „težina” („cena”). Odredite tabelu prosleđivanja za čvor 2, primenom algoritama Dijkstra i Bellman–Ford.

Algoritmi Dijkstra i Bellman–Ford namenjeni su nalaženju najkraće putanje u grafovima čije grane imaju pridružene „težine” ili „cene”. Dijkstrin algoritam je primenjen u



Slika 7.18: *Graf telekomunikacione mreže.*

link-state protokolima rutiranja OSPF i IS-IS, dok se algoritam Bellman-Ford koristi u *distance-vector* protokolu RIP.

Uvedimo oznake:

N — skup čvorova u mreži,
s — izvorišni čvor,
k — korak izvršavanja algoritma,
h — najveći dopušteni broj linkova u putanji za tekući korak izvršavanja algoritma,
M — tekući skup čvorova,
$w(i, j)$ — „cena" linka od čvora i do čvora j, koja zadovoljava uslove:
$w(i, i) = 0$;
$w(i, j) = \infty$ ako čvorovi i i j nisu direktno povezani;
$w(i, j) \geqslant 0$ ako su čvorovi i i j direktno povezani,
$L(n)$ — tekuća najmanja cena putanje od izvorišnog čvora s do odredišnog n,
$L_h(n)$ — najmanja cena putanje od izvorišnog čvora s do odredišnog n koja ima najviše h linkova.

Dijkstrin algoritam tada je opisan koracima:

1. *Inicijalizacija*: $\mathbf{M} = \{s\}$;
   $L(n) = w(s, n)$, za $n \neq s$;

2. *Nalaženje narednog čvora*: Nađi $x \notin \mathbf{M}$, za koji je $L(x) = \min\limits_{j \notin \mathbf{M}} L(j)$;
   Dodaj tekućoj putanji link koji je incidentan na x i na postojeći čvor iz **M** koji doprnosi traženoj putanji;
   Dodaj x u **M**;

3. *Ažuriranje cena*: $(\forall n \notin \mathbf{M})\ L(n) = \min\left(L(n), L(x) + w(x, n)\right)$;

4. *Ponavljanje*: Ponavljaj korake 2 i 3 sve dok ne bude bio ispunjen uslov $\mathbf{M} = \mathbf{N}$.

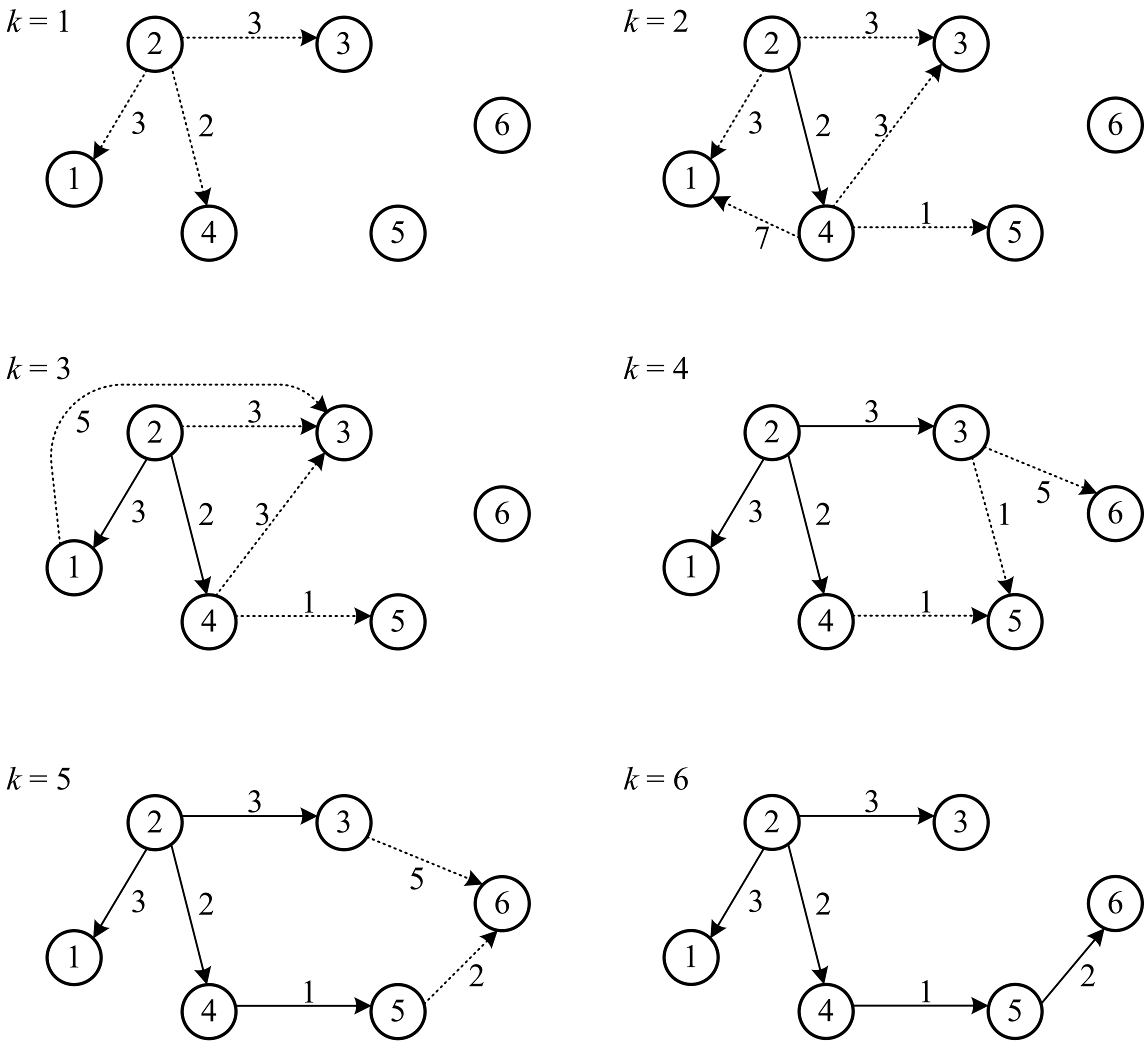
Algoritam Bellman–Ford opisan je koracima:

1. *Inicijalizacija*: $(\forall n \neq s)\ L_0(n) = \infty$;
   $(\forall h)\ L_h(s) = 0$;

2. *Ažuriranje*: $(\forall h \geqslant 0)$, $(\forall n \neq s)$, $L_{h+1}(n) = \min_{j} \left(L_h(j) + w(j,n)\right)$;
   Poveži n s prethodnim čvorom j za koji se dostiže minimum cene i eliminiši veze n s drugim prethodnim čvorovima koje su eventualno ranije formirane.

3. *Završetak*: Putanja od s do n završava se linkom od j ka n.

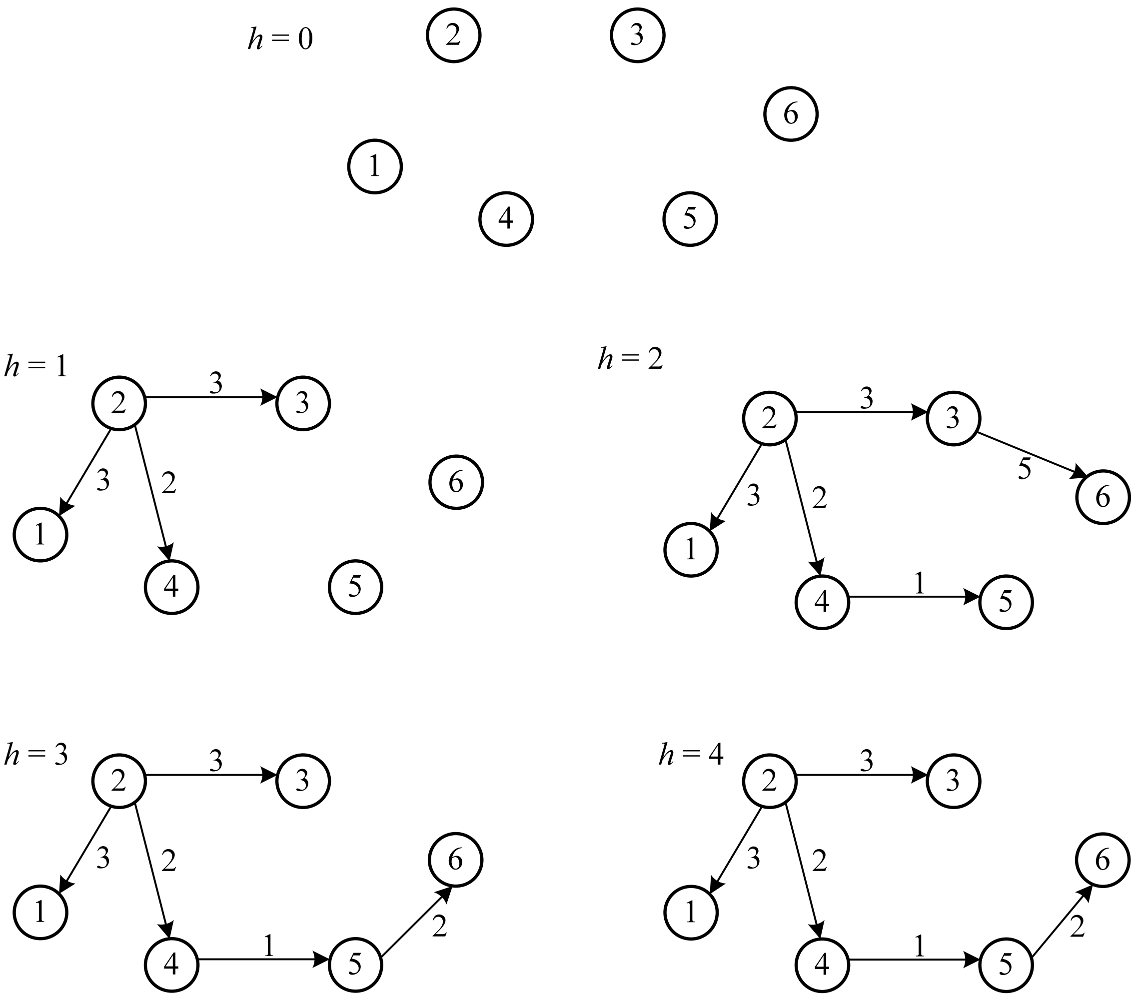
a) Rezultati za Dijkstrin algoritam dati su grafički i u tabeli.



k	**M**	čvor 1		čvor 3		čvor 4		čvor 5		čvor 6	
		L	put	L	put	L	put	L	put	L	put
1	2	3	2–1	3	2–3	2	2–4	∞	—	∞	—
2	2, 4	3	2–1	3	2–3	2	2–4	3	2–4–5	∞	—
3	2, 4, 1	3	2–1	3	2–3	2	2–4	3	2–4–5	∞	—
4	2, 4, 1, 3	3	2–1	3	2–3	2	2–4	3	2–4–5	8	2–3–6
5	2, 4, 1, 3, 5	3	2–1	3	2–3	2	2–4	3	2–4–5	5	2–4–5–6
6	2, 4, 1, 3, 5, 6	3	2–1	3	2–3	2	2–4	3	2–4–5	5	2–4–5–6

Rezultati za Dijkstrin algoritam.

b) Rezultati algoritma Bellman–Ford takođe su dati grafički i tabelarno. Primetimo da oba algoritma daju isti rezultat.

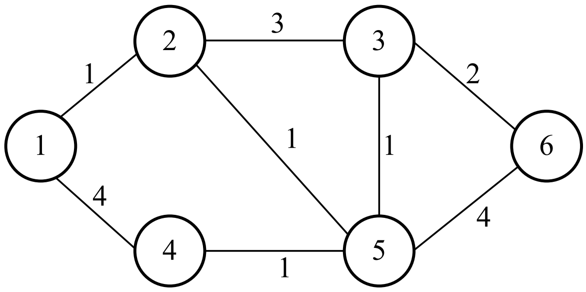


h	čvor 1		čvor 3		čvor 4		čvor 5		čvor 6	
	L_h	putanja	L_h	putanja	L_h	putanja	L_h	putanja	L_h	putanja
0	∞	—	∞	—	∞	—	∞	—	∞	—
1	3	2–1	3	2–3	2	2–4	∞	—	∞	—
2	3	2–1	3	2–3	2	2–4	3	2–4–5	8	2–3–6
3	3	2–1	3	2–3	2	2–4	3	2–4–5	5	2–4–5–6
4	3	2–1	3	2–3	2	2–4	3	2–4–5	5	2–4–5–6

Rezultati za algoritam Bellman–Ford.

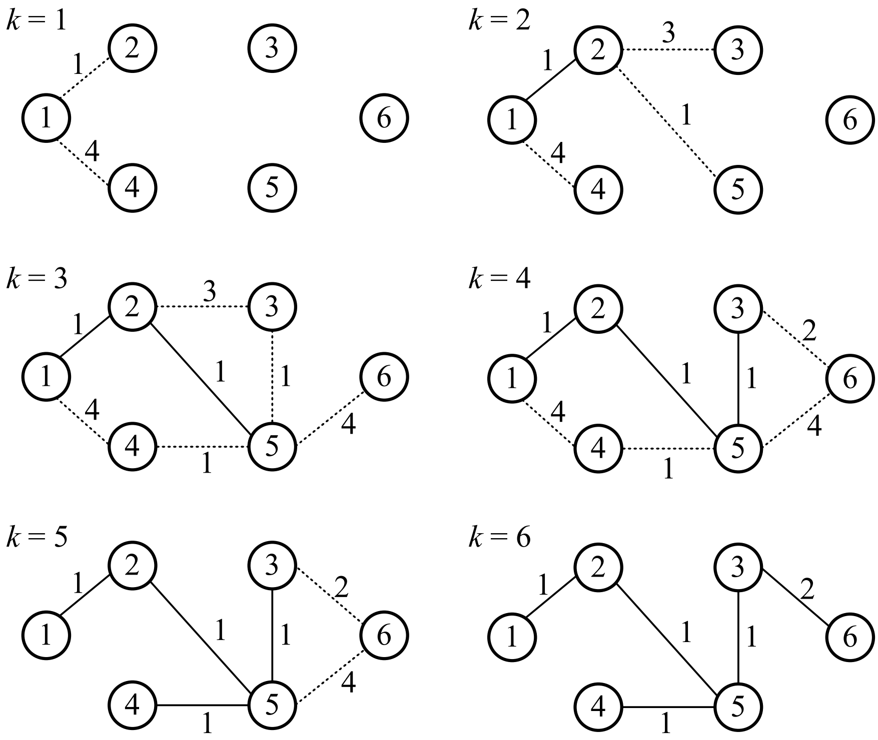
Čitaocu se preporučuje da tabele prosleđivanja odredi i za ostale čvorove mreže. Hoće li rezultati biti simetrični za svaki par čvorova?

Zadatak 7.19 Na slici je dat graf telekomunikacione mreže. Linkovi su dvosmerni, a uz svaki link, označena je i njegova „težina” („cena”). Odredite tabelu prosleđivanja za čvor 1.



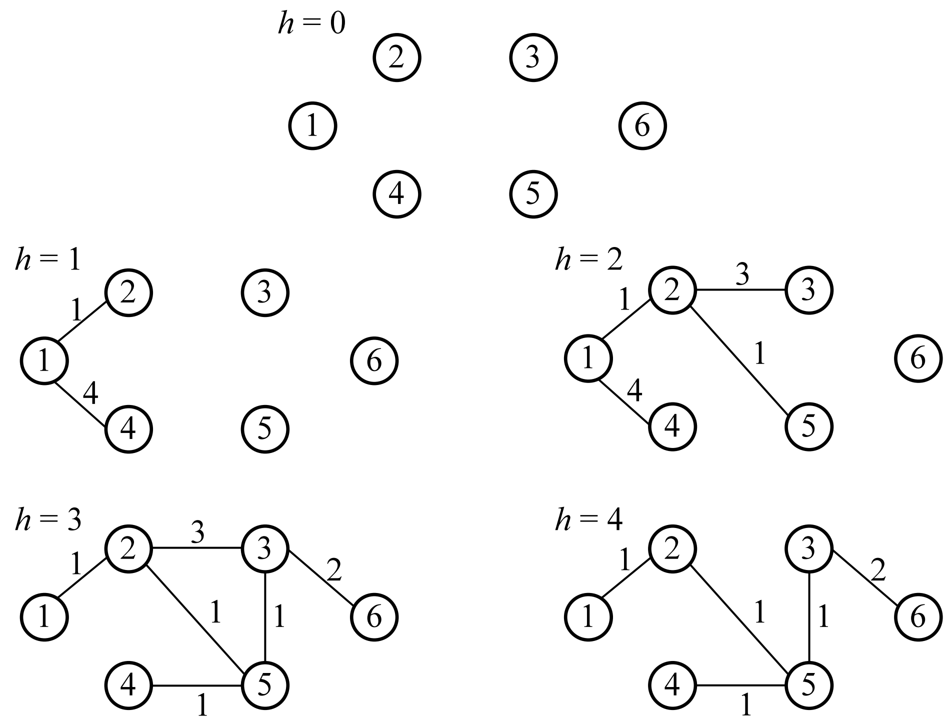
Slika 7.19: *Graf telekomunikacione mreže.*

I ovaj zadatak ćemo rešiti na dva načina, primenom algoritama Dijkstra i Bellman –Ford. Rezultati izvršavanja algoritama dati su grafički i tabelarno. Korišćene su iste oznake kao i u prethodnom zadatku.



k	**M**	čvor 2		čvor 3		čvor 4		čvor 5		čvor 6	
		L	put	L	put	L	put	L	put	L	put
1	1	1	1-2	∞	—	4	1-4	∞	—	∞	—
2	$1, 2$	1	1-2	4	1-2-3	4	1-4	2	1-2-5	∞	—
3	$1, 2, 5$	1	1-2	3	1-2-5-3	3	1-2-5-4	2	1-2-5	6	1-2-5-6
4	$1, 2, 5, 3$	1	1-2	3	1-2-5-3	3	1-2-5-4	2	1–2-5	5	1-2-5-3-6
5	$1, 2, 5, 3, 4$	1	1-2	3	1-2-5-3	3	1-2-5-4	2	1-2-5	5	1-2-5-3-6
6	$1, 2, 5, 3, 4, 6$	1	1-2	3	1-2-5-3	3	1-2-5-4	1	1-2-5	5	1-2-5-3-6

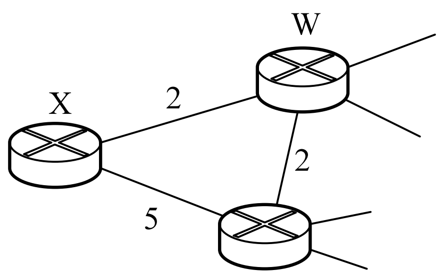
Rezultati za Dijkstrin algoritam.



h	čvor 2		čvor 3		čvor 4		čvor 5		čvor 6	
	L_h	putanja	L_h	putanja	L_h	putanja	L_h	putanja	L_h	putanja
0	∞	—	∞	—	∞	—	∞	—	∞	—
1	1	1–2	∞	—	4	1–4	∞	—	∞	—
2	1	1–2	4	1–2–3	4	1–4	1	1–2–5	∞	—
3	1	1–2	3	1–2–5–3	3	1–2–5–4	2	1–2–5	6	1–2–3–6
4	1	1–2	3	1–2–5–3	3	1–2–5–4	2	1–2–5	5	1–2–5–3–6

Rezultati za algoritam Bellman–Ford.

Zadatak 7.20 Na slici je prikazan segment računarske mreže, s označenim „cenama" linkova. Minimalna „cena" putanje od rutera W do odredišta U (koje nije prikazano na slici) je 5, a od rutera Y do U 6. Primenjeno je rutiranje na bazi vektora rastojanja.



Slika 7.20: *Segment mreže.*

a) Odredite vektor rastojanja u ruteru X za odredišta W, Y i U.

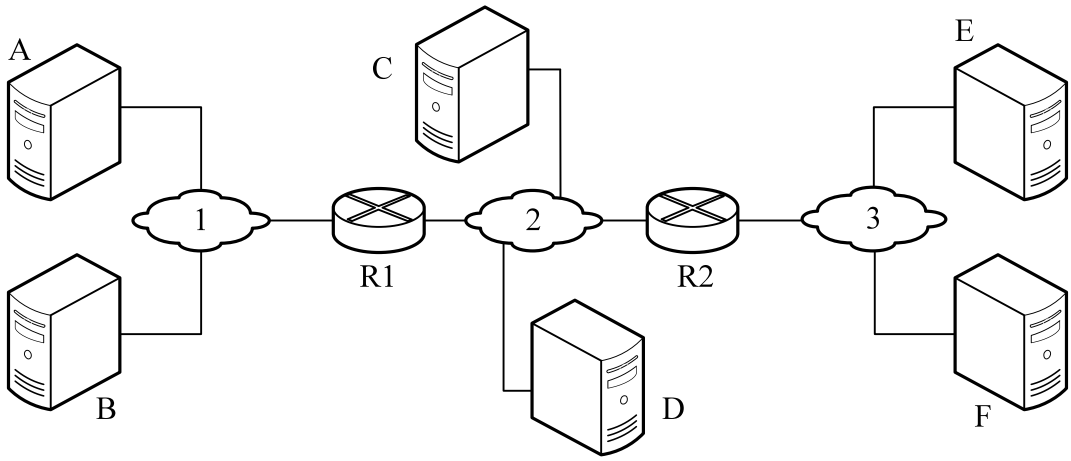
b) Odredite potrebnu promenu „cene” $w(X, W)$ ili $w(X, Y)$ za koju će X obavestiti svoje susede o novoj putanji ka U.

a) Prema zadatim brojčanim podacima, rastojanja od X do traženih odredišta su $D_X(W) = 2$, $D_X(Y) = 4$ i $D_X(U) = 7$.

b) Sve dok je $w(X, Y) \geqslant 1$, putanja od X ka U će voditi preko W, pa X neće obaveštavati susede o promeni „cene” ovoga linka. Ako je $w(X, Y) < 1$, nova putanja najniže „cene” prolazi kroz Y, o čemu X obaveštava susede.

Ako je $0 < w(X, Y) \leqslant 6$, putanja ka U i dalje prolazi kroz W, a X obaveštava susede o promeni njene „cene”. Ako je $w(X, Y) > 6$, nova putanja do U ide preko Y; njena „cena” je 11 i X o tome obaveštava susede.

Zadatak 7.21 Tri podmreže čije IP adrese redom pripadaju opsezima 192.168.1.0/24, 192.168.2.0/24 i 192.168.3.0/24 povezane su dvama ruterima, kao što je to prikazano na slici.



Slika 7.21: *Povezivanje podmreža.*

a) Dodelite IP adrese svim interfejsima i MAC adrese svim mrežnim adapterima.

b) Navedite IP adrese izvorišta i odredišta za IP datagrame i MAC adrese izvorišta i odredišta za Ethernet pakete u koje se oni enkapsuliraju pri slanju poruke od hosta A do hosta F.

c) Pod pretpostavkom da su ARP tabele u ruterima ažurne, opišite postupak slanja IP datagrama od hosta E ka hostu B.

d) Šta bi se desilo da je ARP tabela u hostu E prazna?

a) Hostu A možemo dodeliti IP adresu 192.168.1.1 i MAC adresu 00-00-00-00-00-00. Hostu B dodeljujemo adrese 192.168.1.3 i 22-22-22-22-22-22.

Interfejsu rutera R1 na koji je povezana podmreža 1 dodelićemo IP adresu 192.168.1.2, a njegovom adapteru MAC adresu 11-11-11-11-11-11. Interfejsu koji ide ka drugoj podmreži dodeljujemo IP adresu 192.168.2.2, a njegovom adapteru MAC adresu 33-33-33-33-33-33.

Adrese hosta C su 192.168.2.1 i 44-44-44-44-44-44, a hosta D 192.168.2.4 i 66-66-66-66-66-66.

Na strani druge podmreže, ruter R2 koristi adrese 192.168.2.3 i 55-55-55-55-55-55, a na strani treće podmreže 192.168.3.2 i 88-88-88-88-88-88.

U trećoj podmreži, host E koristi adrese 192.168.3.1 i 77-77-77-77-77-77, a host F 192.168.3.3 i 99-99-99-99-99-99.

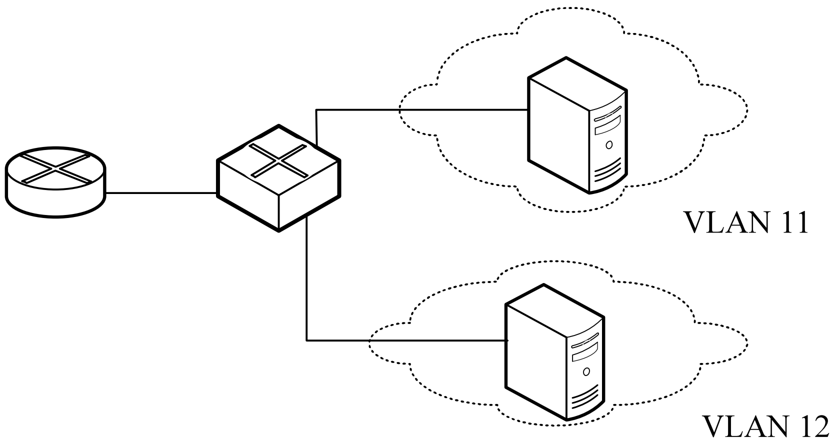
b) Svi IP datagrami koji se šalju od A do F imaće adresu izvorišta 192.168.1.1 i adresu odredišta 192.168.3.3; MAC adrese Ethernet paketa će se razlikovati u zavisnosti od toga na kojoj se deonici šalje taj paket.

Paket koji se prenosi od A do R1 imaće MAC adresu izvorišta 00-00-00-00-00-00 i odredišta 11-11-11-11-11-11. Na trasi R1-R2, paket će imati adresu izvorišta 33-33-33-33-33-33 i odredišta 55-55-55-55-55-55. Konačno, paketi koji se šalju od R2 do F imaju adresu izvorišta 88-88-88-88-88-88 i odredišta 99-99-99-99-99-99-99.

c) Tabela prosleđivanja u hostu E pokazuje da IP datagram treba rutirati na interfejs 192.168.3.2. Mrežni adapter u čvoru E pravi Ethernet paket s odredišnom MAC adresom 88-88-88-88-88-88, koju je očitao iz svoje ARP tabele. Ruter R2 prima ovaj paket i iz njega izdvaja IP datagram. Tabela prosleđivanja u R2 pokazuje da datagram treba rutirati ka interfejsu 192.168.2.2. R2 potom pravi Ethernet paket s izvorišnom adresom 55-55-55-55-55-55 i odredišnom 33-33-33-33-33-33 i šalje ga na svoj interfejs čija je IP adresa 192.168.2.3. Paket dolazi u R1, koji iz njega izdvaja IP datagram. Iz svoje tabele prosleđivanja zaključuje da ga treba rutirati preko interfejsa čija je IP adresa 192.168.1.2. R1 pravi Ethernet paket s adresom izvorišta 11-11-11-11-11-11 i odredišta 22-22-22-22-22-22. Paket dolazi u host B, koji iz njega izdvaja IP datagram.

d) Ako je ARP tabela u hostu E prazna, on će putem širokodifuznog Ethernet paketa poslati ARP upit da bi saznao MAC adresu koja odgovara interfejsu 192.168.3.2. Ruter R2 prima ovaj paket i hostu E šalje ARP odgovor u vidu Ethernet paketa s odredišnom adresom 77-77-77-77-77-77.

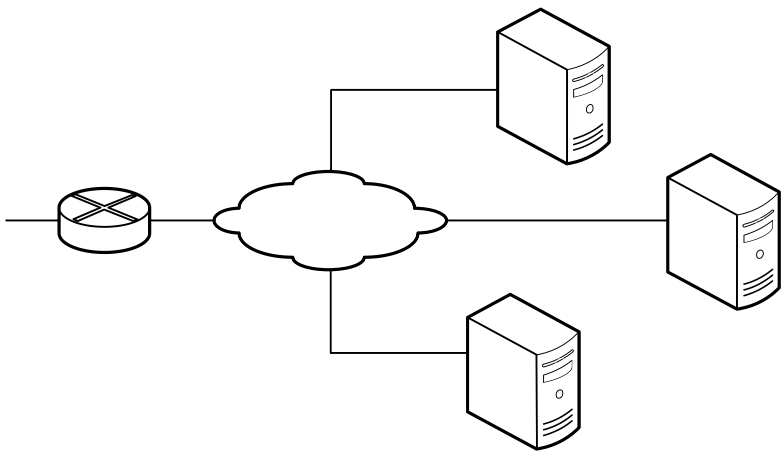
Zadatak 7.22 Dve virtuelne lokalne mreže priključene su na zajednički komutator. Adresni opseg prve VLAN je 111.111.1.0/24, a druge 111.111.2.0/24. Na port 1 komutatora priključen je ruter, čiji je mrežni adapter konfigurisan tako da podržava dva podinterfejsa s IP adresama 111.111.1.0 i 111.111.2.0. Prvi podinterfejs odgovara prvoj VLAN, čija je identifikacija VLAN 11, a drugi drugoj VLAN, čija je identifikacija VLAN 12. Opišite proceduru slanja IP datagrama iz jedne VLAN u drugu.



Slika 7.22: *Povezivanje virtuelnih lokalnih mreža.*

Neka host 111.111.1.1 šalje datagram hostu 111.111.2.1. Izvorišni host enkapsulira ovaj datagram u Ethernet paket čija je odredišna adresa jednaka MAC adresi adaptera rutera koji je povezan na port 1 komutatora. Kada primi paket, ruter iz njega izdvaja IP datagram i prosleđuje ga svom mrežnom sloju, koji zaključuje da paket treba proslediti podmreži 111.111.2.0/24 preko interfejsa 111.111.2.0. Ruter potom enkapsulira ovaj datagram u novi Ethernet paket, kome pridružuje oznaku VLAN ID 12 i šalje ga ka portu 1 komutatora. Komutator prosleđuje paket hostu 111.111.2.1, koji iz njega uklanja identifikaciju virtuelne LAN i izdvaja IP datagram.

Zadatak 7.23 U mreži sa slike, primenjeno je transliranje mrežne adrese (NAT). ISP je ruteru dodelio adresu 24.34.112.235, dok je adresa LAN 192.168.1.0/24.



Slika 7.23: *Transliranje mrežne adrese.*

a) Dodelite adrese interfejsima u LAN.

b) Ako svaki host ima dve aktivne veze ka portu 80 na hostu 128.119.40.83, navedite odgovarajuće unose u NAT tabeli.

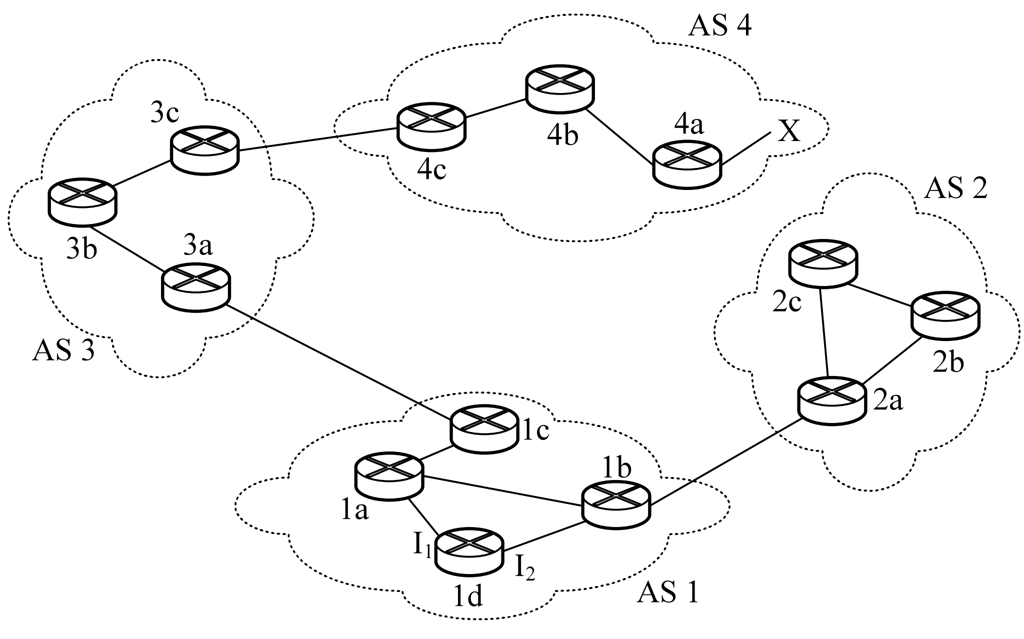
a) Trima prikazanim hostovima možemo redom dodeliti adrese 192.168.1.1, 192.168.1.2 i 192.168.1.3. Adresa interfejsa rutera koji je spojen na LAN je npr. 192.168.1.4.

b) Za rešavanje zadatka, pretpostavimo da su izvorišni portovi na prvom hostu 3345 i 3346; na drugom 3445 i 3446, a na trećem 3545 i 3546. U komunikaciji s WAN, NAT ruter preslikava privatne IP adrese hostova u svoju adresu (24.34.112.235), a takođe i proizvoljno preslikava brojeve njihovih portova, kao što je to prikazano u tabeli.

Tabela transliranja mrežne adrese.

WAN	LAN
24.34.112.235, 4000	192.168.1.1, 3345
24.34.112.235, 4001	192.168.1.1, 3346
24.34.112.235, 4002	192.168.1.2, 3445
24.34.112.235, 4003	192.168.1.2, 3446
24.34.112.235, 4004	192.168.1.3, 3545
24.34.112.235, 4005	192.168.1.3, 3546

Zadatak 7.24 U mreži sa slike, autonomni sistemi AS 1 i AS 4 koriste RIP za intra-AS rutiranje, a sistemi AS 2 i AS 3 OSPF. Za inter-AS rutiranje se koriste eBGP i BGP.



Slika 7.24: *Mreža autonomnih sistema.*

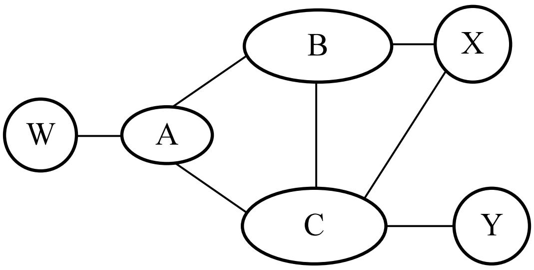
a) Preko kog protokola rutiranja ruteri 1c, 1d, 3a i 3c saznaju za prefiks X?

b) Kada ruter 1d bude saznao za X, u svoju tabelu rutiranja će staviti unos (X, I). Čemu će biti jednako I? Hoće li se ovaj unos promeniti kada se bude uspostavio direktni link između rutera 2c i 4a?

a) Ruteri 1c, 3a i 3c uspostavljaju eBGP sesije, dok ruter 1d uspostavlja iBGP sesiju.

b) Upisaće se I_1, jer ovim interfejsom započinje najkraća putanja od 1d ka 1c, koji je povezan s AS 3. Ukoliko bi se uspostavio direktni link između AS 2 i AS 4, ovaj unos bi bio I_2. Naime, obe rute imaju jednaku AS-PATH dužinu, ali ruta koja započinje interfejsom I_2 ima najbliži NEXT-HOP ruter.

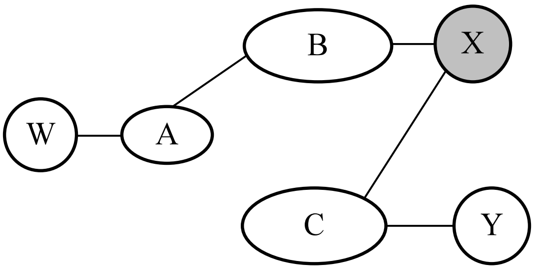
Zadatak 7.25 Na levoj slici je prikazano šest autonomnih sistema koji koriste BGP, dok je na desnoj prikazana topologija koju „vidi" sistem Y.



Slika 7.25: *Mreža autonomnih sistema (levo) i topologija koju „vidi" Y (desno).*

Koje topologije „vide" sistemi X i W?

Iz topologije koju „vidi" Y zaključujemo da se link između B i C ne koristi. Topologije sa stanovišta sistema X i W date su na narednoj slici.



Topologije koje „vide" X (levo) i W (desno).

Zadatak 7.26 Kolika je veličina adresnog prostora u IPv4 koji je namenjen višedifuziji? Kolika je verovatnoća da dve višedifuzne grupe izaberu istu adresu, a kolika da od 1000 grupa svaka izabere različitu?

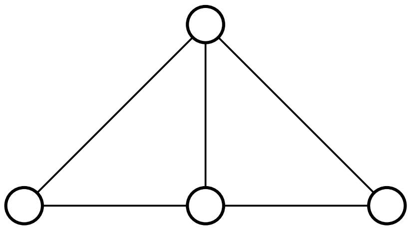
Višedifuzne adrese su klase D i one počinju bitima 1110. Na raspolaganju ostaje 28 b, pa ovih adresa ima $N = 2^{28}$.

Verovatnoća da se u dva izvlačenja izabere ista adresa je je $1/N = 2^{-28} = 3{,}73 \cdot 10^{-9}$.

Verovatnoća da se izabere 1000 različitih adresa je

$$\begin{aligned}
\mathrm{P}_{1000} &= \frac{N(N-1)(N-2)\cdots(N-999)}{N^{1000}} = \\
&= \left(1-\frac{1}{N}\right)\left(1-\frac{2}{N}\right)\cdots\left(1-\frac{999}{N}\right) = \\
&\approx 1-\frac{1+2+\cdots+999}{N} = 1-\frac{999\cdot 1000}{2N} = \\
&= 0{,}998.
\end{aligned}$$

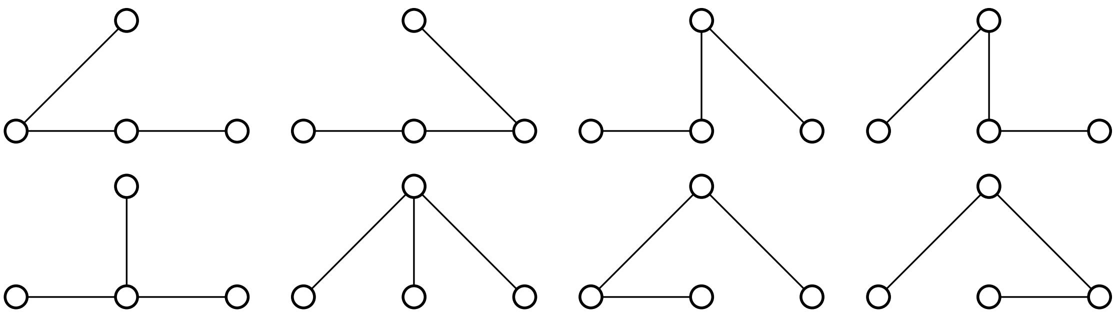
Zadatak 7.27 Odredite sva razapinjuća stabla grafa sa slike.



Slika 7.27: *Graf telekomunikacione mreže.*

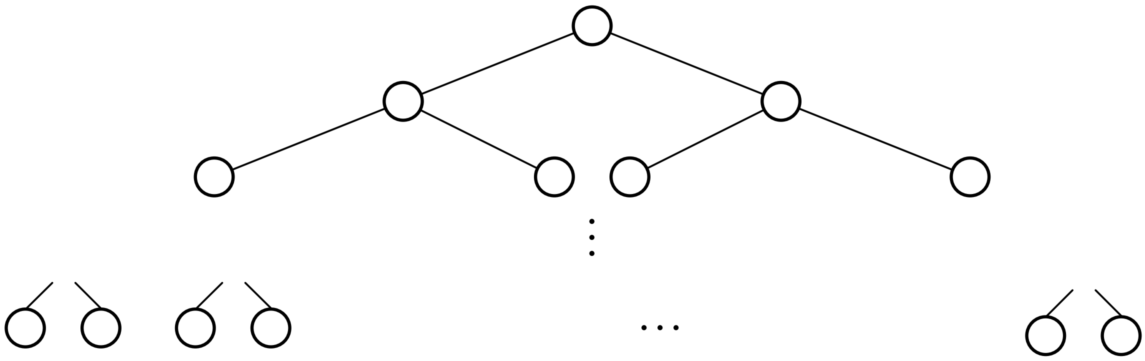
Razapinjuće stablo grafa (*spanning tree*) je njegov podgraf koji se sastoji od svih čvorova tog grafa i podskupa grana koje obezbeđuju potpunu povezanost (postoji putanja između bilo kojih dvaju čvora), bez postojanja petlji.

Povezani grafovi mogu imati više od jednog razapinjućeg stabla. U primeru iz zadatka postoji osam razapinjućih stabala, koja su prikazana na narednoj slici.



Razapinjuća stabla grafa sa slike 7.27.

Zadatak 7.28 Odredite prosečno rastojanje čvorova u mreži s topologijom binarnog stabla. Pretpostavite da se svi parovi čvorova javljaju s jednakim verovatnoćama, kao i da čvorovi uzajamno komuniciraju isključivo preko korena stabla.



Slika 7.28: *Binarno stablo.*

Izgled binarnog stabla prikazan je na slici 7.28.

Na i-tom nivou stabla se nalazi 2^{i-1} čvorova. Ako stablo ima N nivoa, ukupan broj čvorova je

$$\sum_{i=1}^{N} 2^{i-1} = 2^N - 1.$$

Pošto čvorovi komuniciraju isključivo preko korena stabla, prosečno rastojanje dvaju čvorova dvaput je veće od prosečnog rastojanja čvora i korena. Kako je udaljenost čvora na i-tom nivou od korena $i-1$, prosečno rastojanje dvaju čvorova je

$$\begin{aligned}\bar{L} &= 2\sum_{i=1}^{N} \frac{2^{i-1}(i-1)}{2^N-1} = \frac{2}{2^N-1}\sum_{i=1}^{N} 2^{i-1}(i-1) = \\ &= \frac{2}{2^N-1}\left(2+2^N(N-2)\right).\end{aligned}$$

Za veliko N, dobijamo $\bar{L} \approx 2N-4$.

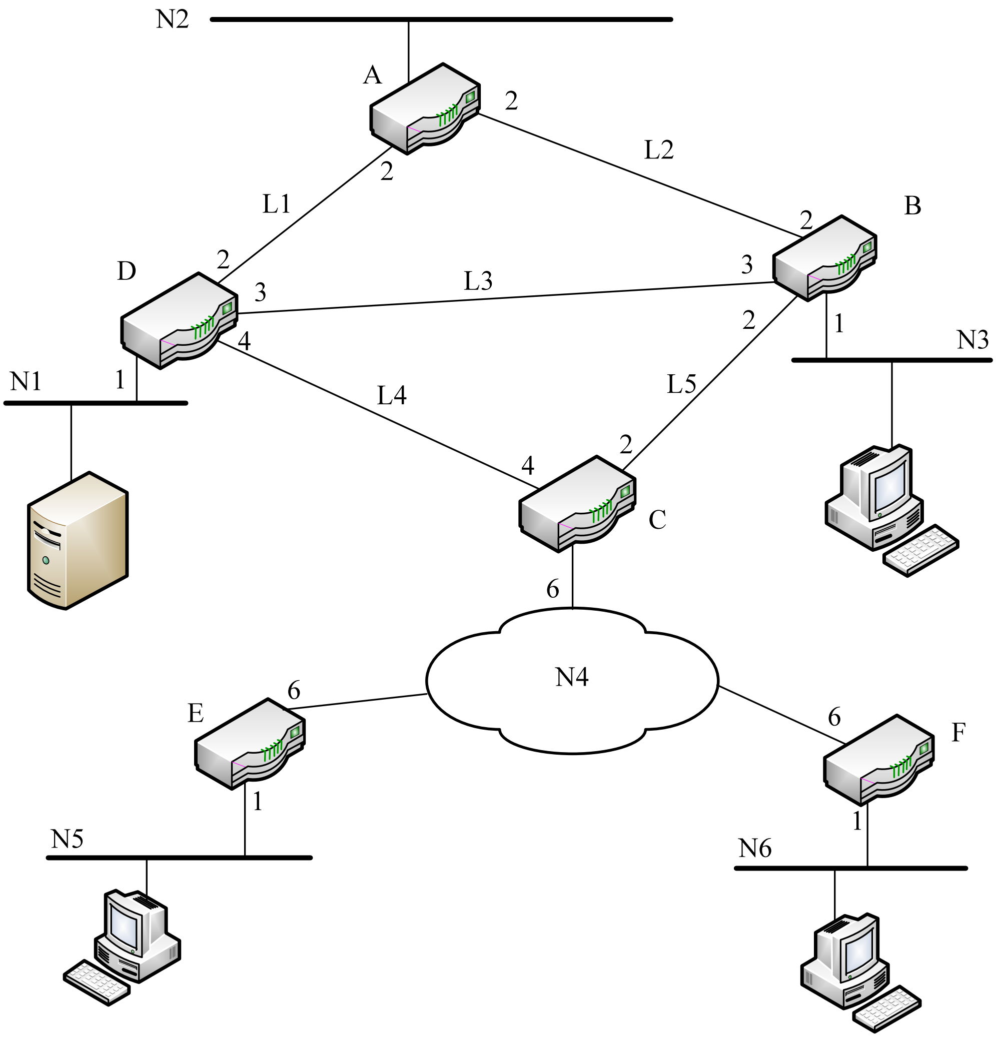
Zadatak 7.29 Na slici je prikazana struktura računarske mreže, koja se sastoji od LAN N1, N2, N3, N5 i N6, koje su povezane ruterima A, B, C, D, E i F. Ruteri su međusobno povezani linkovima velikog kapaciteta, L1-L5, ili preko WAN (N4). Uz svaki link, označena je „cena" za posmatrani smer prenosa. Višedifuzni server, koji je povezan na LAN N1, emituje pakete namenjene stanicama koje su povezane na LAN N3, N5 i N6. Odredite putanje i broj emitovanih paketa, ukoliko se koriste sledeće strategije:

a) plavljenje (*flooding*),

b) difuzija (*broadcast*), po putanji najniže cene,

c) višestruki *unicast* ili

d) prava višedifuzija (*multicast*).



Slika 7.29: *Struktura mreže.*

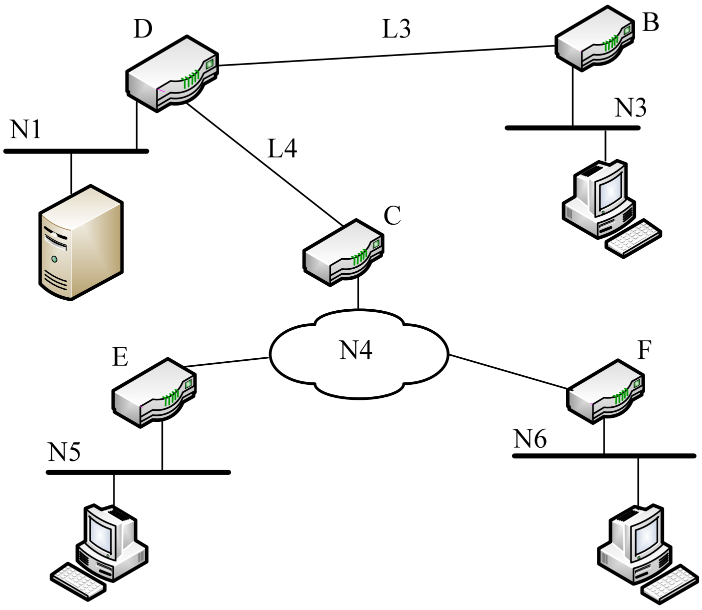
a) U strategiji plavljenja, izvor emituje po jednu kopiju paketa svakom od rutera na koje je povezan. Svaki ruter će emitovati kopije primljenog paketa ka svim linkovima, osim ka onom s kog je dobio taj paket. Svakom paketu se dodeljuje jedinstvena oznaka, tako da se izbegava ponovljeno kopiranje istog paketa u ruterima.

U našem zadatku, server će, preko LAN N1, emitovati paket do rutera D. On će potom poslati kopije ka ruterima A , B i C. U drugom koraku, A će poslati svoju kopiju ka N2 i B; B ka A, C i N3, dok će C emitovati kopije ka B i N4. WAN N4 će proslediti kopije do rutera E i F, a preko njih i do odredišnih LAN N5 i N6. Emitovaće se ukupno 15 kopija.

b) Ova strategija je pogodna za slučaj kada server ne zna lokacije članova višedifuzne grupe. Server će, preko LAN N1 i rutera D, poslati kopije svakog paketa svim mrežama u konfiguraciji, preko putanja najniže cene. Kopija namenjena mreži N2 ići će putanjom N1—D—L1—A; kopija za N3 preko N1—D—L3—B; kopija za N5 preko N1—D—L4—C—N4—E i kopija za N6 preko N1—D—L4—C—N4—F. Ukupno će se emitovati 16 kopija.

c) Ako server zna lokacije članova višedifuzne grupe, slaće kopije paketa samo mrežama u kojima se oni nalaze. U primeru iz našeg zadatka, to znači da šalje ukupno 13 kopija ka LAN N3, N5 i N6, po istim putanjama kao u tački b).

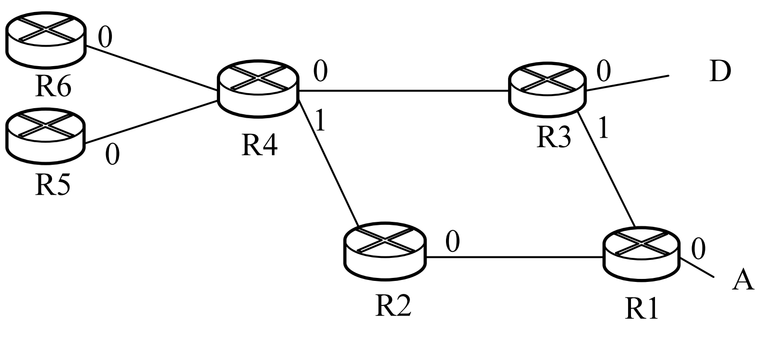
d) Kod prave višedifuzije, određuju se putanje najniže cene ka mrežama koje sadrže članove višedifuzne grupe, tako da se formira stablo od izvora saobraćaja do svih članova grupe. Stablo za slučaj posmatrane konfiguracije prikazano je na slici.



Višedifuzno stablo.

Izvor emituje jedan paket, koji se kopira samo u tačkama grananja stabla; stoga se u ovom slučaju emituje 9 kopija paketa.

Zadatak 7.30 U MPLS mreži sa slike potrebno je implementirati inženjering saobraćaja, tako da se saobraćaj iz R6 namenjen odredištu A rutira po putanji R6–R4–R3–R1, a saobraćaj iz R5 ka A po putanji R5–R4–R2–R1. Konstruišite MPLS tabele u ruterima R1-6 kojima se ovo omogućuje.



Slika 7.30: *MPLS mreža.*

U zadatku 7.11 videli smo da se klasično rutiranje zasniva isključivo na adresi odredišta, pa u njemu nije moguć inženjering saobraćaja. Uvođenjem komutacije labela, tokove saobraćaja koji se stiču u jednom čvoru i idu ka istom odredištu možemo rutirati po različitim putanjama. Za proizvoljno usvojene labele, na ovaj način dobijamo sledeću tabelu.

Ruter	In label	Out label	Odredište	Izlazni interfejs
R1	6		A	0
R2	4	6	A	0
R3	3	6	A	1
	5	7	D	0
R4	1	3	A	0
	2	4	A	1
		5	D	0
R5		2	A	0
R6		1	A	0

8. Transportni sloj

Zadatak 8.1 Poruka sa sloja transporta sastoji se od 160 b zaglavlja i 1500 b korisnog sadržaja. Ova poruka se dostavlja mrežnom sloju, koji joj dodaje još 160 b zaglavlja. Poruka se potom prenosi kroz mrežu koja koristi zaglavlja veličine 24 b, dok je maksimalna veličina okvira 800 b. Koliko se ukupno bita dostavlja protokolu sloja linka podataka na odredištu?

Ukupna veličina korisnog sadržaja, zaglavlja sa sloja transporta i zaglavlja s mrežnog sloja je 1820 b. Ovi podaci se isporučuju u vidu sekvence okvira, od kojih se svaki sastoji od 24 bita zaglavlja i do 776 bita koji mogu biti zaglavlja viših slojeva i/ili korisni podaci. Da bi se prenelo svih 1820 b, potrebna su tri takva okvira. Protokolu sloja mreže stoga će se ukupno isporučiti

$$1820 \text{ b} + 3 \cdot 24 \text{ b} = 1892 \text{ b}.$$

Zadatak 8.2 Odredite maksimalni protok po konekciji u mreži u kojoj se koriste paketi čija je maksimalna veličina 128 B, maksimalni vek 30 s i polje broja sekvence dužine 8 b.

Tokom intervala jednakog maksimalnom veku paketa, korisnik može emitovati do $2^8 = 256$ paketa, s ukupno 32768 B podataka. Ovome odgovara protok od 8738,13 b/s.

Zadatak 8.3 Poput IP, UDP i TCP takođe koriste proveru kontrolne sume da bi utvrdili ispravnost primljenih podataka. Je li ipak moguće da se u prenosu desi greška koja se ne može detektovati na ovaj način?

Neka su u dve reči biti na istim pozicijama komplementarni (0 i 1). Ako se u prenosu bude pogrešilo na oba ova bita, kontrolna suma će ostati nepromenjena, pa prijemnik neće detektovati ovakvu grešku.

Zadatak 8.4 Posmatraju se dva IP datagrama koji nose UDP segmente. IP adresa izvorišta prvog datagrama je A1, odredišta B, izvorišni UDP port je P1, a odredišni T. Za drugi datagram, IP adresa izvorišta je A2, odredišta B, izvorišni UDP port je P2, a odredišni T. Ako je A1≠A2 i P1≠P2, hoće li se UDP paketi primiti preko istog socketa?

UDP socketi se identifikuju uređenim parom (*IP adresa odredišta*, *odredišni port*). Pošto su IP adrese odredišta iste i pošto su isti odredišni portovi, posmatrani paketi će se primiti preko istog socketa.

Zadatak 8.5 Koliko je IP datagrama ne dužih od 1200 B potrebno da bi se datoteka veličine 4 miliona bajtova prenela posredstvom TCP?

Svaki TCP segment i IP datagram imaju po 20 B zaglavlja, pa se u jedan zadati IP datagram može smestiti do 1160 B sadržaja datoteke. Za prenos će stoga trebati

$$N = \left\lceil \frac{4 \cdot 10^6 \text{ B}}{1160 \text{ B}} \right\rceil = 3449$$

IP datagrama. Svi sem poslednjeg biće veličine 1200 B, dok će on imati 320 B + 40 B = 360 B.

Zadatak 8.6 Velika datoteka se prenosi posredstvom TCP. Maksimalna veličina segmenta iznosi 512 B.

a) Kolika je maksimalna veličina datoteke, tako da se ne iscrpe brojevi sekvence TCP segmenata?

b) Odredite vreme potrebno za prenos datoteke maksimalne veličine po linku protoka 165 Mb/s. Pretpostavite da transportni i niži slojevi svakom segmentu dodaju „overhead" od ukupno 64 B. Zanemarite kontrolu toka i zagušenja.

a) Polje broja sekvence u TCP je veličine 4 B, pa je njime moguće označiti $2^{4 \cdot 8} = 4\,294\,967\,296$ vrednosti. Pri prenosu, vrednost ovoga polja se uvećava za broj prenesenih bajtova. Maksimalna veličina datoteke stoga je 2^{32} B ≈ 4,19 GB i ne zavisi od maksimalne veličine TCP segmenta.

b) Potreban broj segmenata je

$$N = \left\lceil \frac{2^{32} \text{ B}}{512 \text{ B}} \right\rceil = 8\,388\,608.$$

Ukupni „overhead" je $N \cdot 64$ B = 536 870 912 B. Uz njega, ukupno će se preneti $4{,}832 \cdot 10^9$ B, pa će prenos trajati 249 s.

Zadatak 8.7 Hostovi A i B komuniciraju korišćenjem TCP. Svaki put kad primi segment od hosta A, host B mu odgovori potvrdom. Nakon što je host B dobio 126 B podataka, host A mu šalje dva segmenta koji sadrže 80 B i 40 B podataka. Broj sekvence prvog segmenta je 127, broj izvorišnog porta 302, a broj odredišnog porta 80.

a) Odredite brojeve sekvence, izvorišnog i odredišnog porta u drugom segmentu.

b) Ako B prvo prima prvi segment, odredite brojeve sekvence, izvorišnog i odredišnog porta u potvrdi njegovog prijema.

c) Ako B prvo prima drugi segment, odredite brojeve sekvence, izvorišnog i odredišnog porta u potvrdi njegovog prijema.

a) Do trenutka slanja drugog segmenta, host A je poslao 126 B + 80 B = 206 B podataka. Broj sekvence drugog segmenta stoga je 207. Brojevi izvorišnog i odredišnog porta isti su kao i u prvom segmentu i redom iznose 302 i 80.

b) Host B je uspešno primio 206 B podataka i očekuje prijem 207. bajta, pa je vrednost polja broja sekvence u potvrdi 207. Broj izvorišnog porta je 80, a broj odredišnog 302.

c) B je uspešno primio 126 B podataka i očekuje prijem 127. bajta i narednih. Vrednost polja broja sekvence u potvrdi je 127, broj izvorišnog porta je 80, a broj odredišnog 302.

Zadatak 8.8 Uporedite procedure „vrati se za *N*", „selektivno ponavljanje" i kontrolu zagušenja u TCP. Pretpostavite da je trajanje retransmisionog tajmera dovoljno da se prenese pet uzastopnih segmenata podataka s pripadajućim pozitivnim potvrdama. Posmatrajte scenario po kome host A ima pet segmenata za slanje hostu B, pri čemu se drugi segment izgubi. Posle retransmisije, host B uspešno prima svih pet segmenata.

U proceduri „vrati se za *N*", prijemnik odbacuje segmente koji su primljeni posle izgubljenog i očekuje da mu ih predajnik ponovo pošalje. Host A stoga šalje redom segmente 1, 2, 3, 4, 5, 2, 3, 4 i 5, a host B mu odgovara potvrdama 1, 1, 1, 1, 2, 3, 4 i 5.

U proceduri „selektivno ponavljanje", primljeni segmenti se memorišu, a ponovo se šalje samo izgubljeni. Host A tako šalje segmente 1, 2, 3, 4, 5 i 2, a host B mu odgovara potvrdama 1, 3, 4, 5 i 2.

U slučaju TCP, host A šalje segmente 1, 2, 3, 4, 5, 2. Host B memoriše primljene segmente i odgovara očekivanim brojem sekvence narednog segmenta: 2, 2, 2, 2, 2, 6.

Zadatak 8.9 Skicirajte promenu veličine otvora prozora zagušenja u TCP Tahoe i ilustrujte mehanizme „sporog starta" i izbegavanja zagušenja.

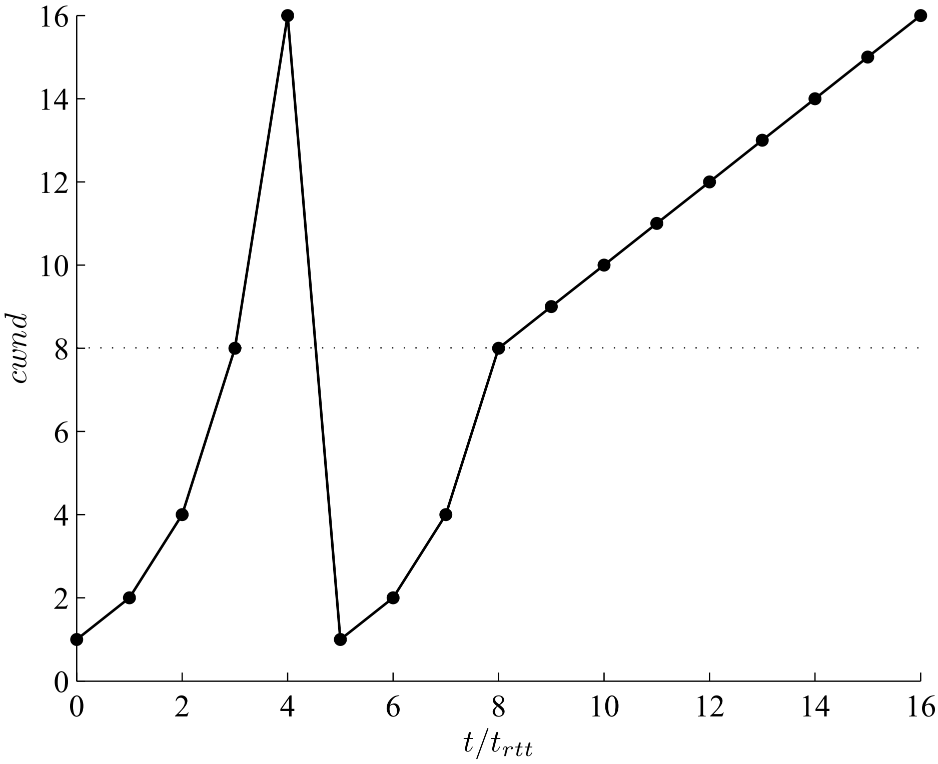
Neka je t_{rtt} vreme potrebno za prenos u oba smera (*round trip time*), tj. vreme potrebno da bi se poslao segment i dobila potvrda njegovog uspešnog prijema. Neka je na početku vrednost praga za „spori start" $ssthresh = 20$. Pretpostavimo scenario po kome se prvih pet segmenata šalje po principu „sporog starta", dok u trenutku $t = 5t_{rtt}$ izostaje prijem pozitivne potvrde, što znači da je u mreži nastupilo zagušenje.

Početna veličina otvora prozora zagušenja je $cwnd = 1$, što znači da TCP može poslati samo jedan segment i da potom treba sačekati pozitivnu potvrdu njegovog prijema. Kada ta potvrda bude stigla (u trenutku $t = t_{rtt}$), vrednost $cwnd$ se postavlja na 2 i TCP može poslati dva segmenta. U trenutku $t = 2t_{rtt}$, biće $cwnd = 4$ i šalju se 4 segmenta, u $t = 3t_{rtt}$, $cwnd = 8$ (šalje se 8 segmenata) i u trenutku $t = 4t_{rtt}$, biće $cwnd = 16$ (šalje se 16 segmenata).

Pozitivna potvrda koja je trebalo da stigne u trenutku $t = 5t_{rtt}$ izostaje, pa entitet TCP-a zaključuje da je nastupilo zagušenje. Pokreće se mehanizam izbegavanja zagušenja. Vrednost praga za „spori start" postavlja se na polovinu vrednosti $cwnd$ ($ssthresh = cwnd/2 = 8$). Vrednost $cwnd$ postavlja se na 1 i otpočinje slanje po principu „sporog starta", sve dok $cwnd$ ne dostigne vrednost praga, $ssthresh$. U našem slučaju, to će se desiti u trenutku $t = 8t_{rtt}$.

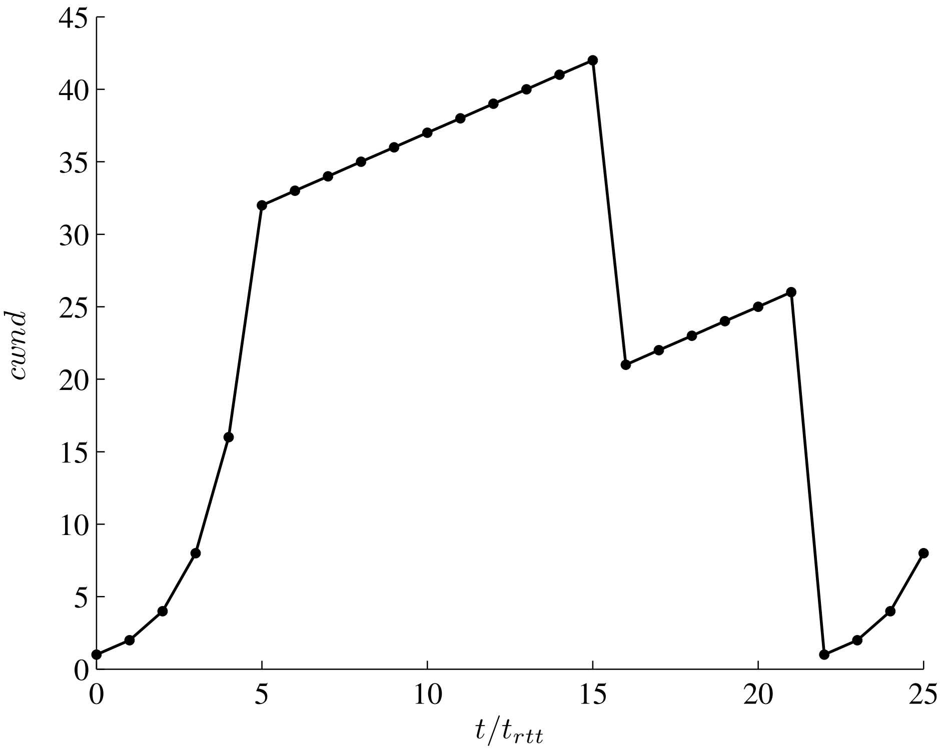
Posle dostizanja praga, vrednost $cwnd$ se uvećava za 1 u svakom intervalu t_{rtt}.

Ovo razmatranje ilustrovano je na slici 8.9.



Slika 8.9: *„Spori start" i izbegavanje zagušenja u TCP Tahoe.*

Zadatak 8.10 Na slici je ilustrovana promena veličine otvora prozora zagušenja u TCP Reno.



Slika 8.10: *TCP Reno.*

a) Objasnite šta se desilo u 15. i 21. trenutku.

b) Kolike su vrednosti *ssthresh* u 1, 18. i 24. trenutku?

c) Kada će se preneti 70. segment?

d) Ako se posle 25. trenutka bude detektovao gubitak segmenta, kolike će biti nove vrednosti *cwnd* i *ssthresh*?

a) Iz promene *cwnd* zaključujemo da je gubitak segmenta u $t = 15t_{rtt}$ detektovan preko trostruke duplirane pozitivne potvrde, a u $t = 21t_{rtt}$ preko isteka retransmisionog tajmera.

b) U prvom trenutku je $ssthresh = 32$, jer za tu vrednost *cwnd* prestaje spori start. U $t = 18t_{rtt}$ je $ssthresh = cwnd/2 = 21$, a u $t = 24t_{rtt}$ $ssthresh = cwnd/2 = 13$.

c) Tokom šest rundi „sporog starta", do $t = 5t_{rtt}$, prenesu se ukupno 63 segmenta; potom sledi izbegavanje zagušenja s linearnim porastom *cwnd*. Sedamdeseti segment stoga će se preneti u trinaestoj rundi, tj. za $t = 12t_{rtt}$.

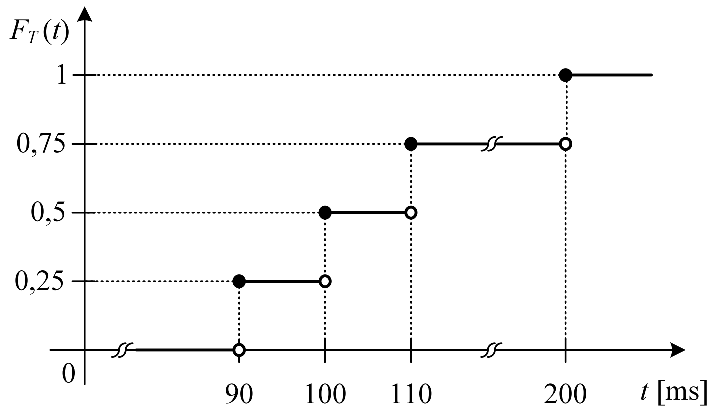
d) Vrednost polja *ssthresh* se postavlja na polovinu zatečene vrednosti *cwnd*; pošto je ona u $t = 25t_{rtt}$ iznosila 8, biće $ssthresh = 4$. Nova vrednost parametra *cwnd* je $cwnd = ssthresh + 3 = 7$.

Zadatak 8.11 Merenjem kašnjenja u računarskoj mreži, dobijene su sledeće vrednosti: 100 ms, 110 ms, 90 ms, 200 ms. Koliko, na osnovu ovoga uzorka, iznose percentil reda 50 i medijana?

Prema dokumentu RFC 2330, percentil reda y je najmanja vrednost x za koju važi $F_X(x) \geqslant y$, gde je $F_X(x)$ empirijska funkcija raspodele izmerenih vrednosti. Ako je x manje od najmanje izmerene vrednosti, tada je $F_X(x) = 0\,\%$, a ako je x veće ili jednako maksimalnoj izmerenoj vrednosti, tada je $F_X(x) = 100\,\%$.

Ukoliko je broj merenja neparan, medijana je jednaka pedesetom percentilu. Ukoliko je broj merenja paran, medijana se određuje tako što se merenja poredaju po neopadajućem redosledu i izračuna aritmetička sredina dvaju centralnih merenja.

Empirijska funkcija raspodele izmerenog uzorka kašnjenja prikazana je na slici.



Slika 8.11: *Empirijska funkcija raspodele kašnjenja.*

Posmatranjem ovoga grafika, zaključujemo da je vrednost pedesetog percentila 100 ms, a vrednost medijane 105 ms.

Čitaocu se preporučuje da odredi i vrednosti percentila reda 0, 25 i 100.

Zadatak 8.12 Posmatranjem prenosa 20 TCP segmenata, dobijene su vrednosti vremena potrebnog za prenos u oba smera koje su navedene u tabeli. Predloženo je više načina za procenu vrednosti vremena potrebnog za prenos u oba smera, na osnovu poznatih vrednosti za prethodne segmente. Uporedite performanse jednostavnog usrednjavanja,

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
t_{rtt}	1	2	3	4	5	6	7	8	9	10	10	10	10	10	10	10	10	10	10	10

$$ARTT(k+1) = \frac{1}{k+1} \sum_{i=1}^{k+1} RTT(i),$$

s eksponencijalnim usrednjavanjem, prema RFC 793,

$$SRTT(k+1) = \alpha SRTT(k) + (1-\alpha) RTT(k+1),$$

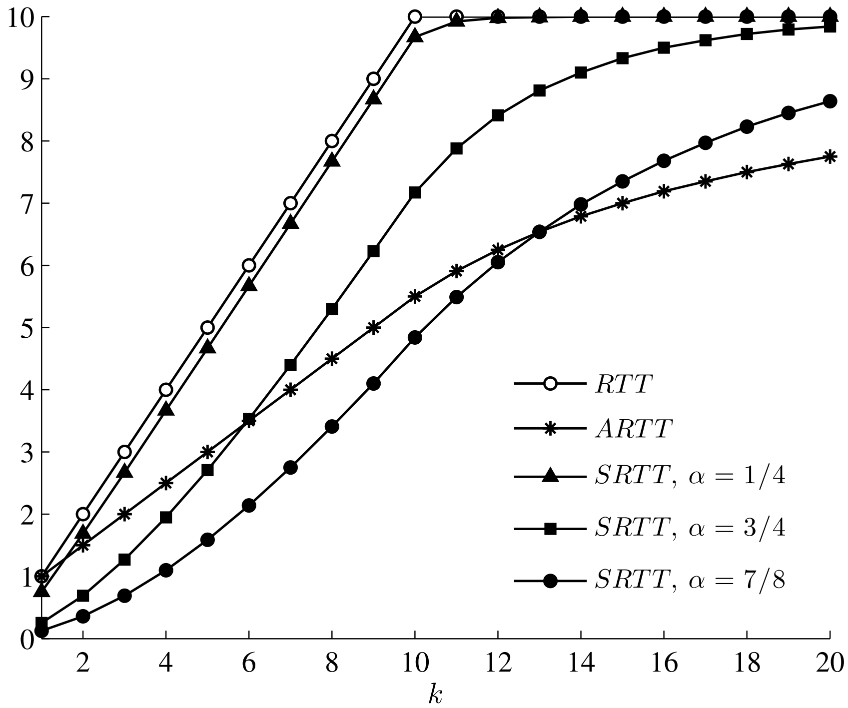
pri čemu je $RTT(i)$ vrednost vremena potrebnog za prenos u oba smera koja je izmerena za i-ti segment i $SRTT(0) = 0$. Posmatrajte opseg vrednosti $\alpha \in \{1/4,\ 3/4,\ 7/8\}$.

Rezultati izračunavanja prikazani su tabelarno i grafički, na slici 8.12. Radi lakšeg poređenja, date su i izmerene vrednosti vremena potrebnog za prenos u oba smera.

Procena vremena potrebnog za prenos u oba smera.

k	$RTT(k)$	$ARTT(k)$	$SRTT(k)$		
			$\alpha = 1/4$	$\alpha = 3/4$	$\alpha = 7/8$
1	1	1	0,75	0,25	0,125
2	2	1,5	1,69	0,69	0,36
3	3	2	2,67	1,27	0,69
4	4	2,5	3,67	1,95	1,10
5	5	3	4,67	2,71	1,59
6	6	3,5	5,67	3,53	2,14
7	7	4	6,67	4,40	2,75
8	8	4,5	7,67	5,30	3,41
9	9	5	8,67	6,23	4,10
10	10	5,5	9,67	7,17	4,84
11	10	5,91	9,92	7,88	5,49
12	10	6,25	9,98	8,41	6,05
13	10	6,54	9,99	8,81	6,54
14	10	6,79	10	9,10	6,98
15	10	7	10	9,33	7,35
16	10	7,19	10	9,50	7,68
17	10	7,35	10	9,62	7,97
18	10	7,5	10	9,72	8,23
19	10	7,63	10	9,79	8,45
20	10	7,75	10	9,84	8,64

Poređenjem procenjenih vrednosti sa stvarnim, zaključujemo da eksponencijalno usrednjavanje može pratiti promene posmatrane veličine brže nego jednostavno usrednjavanje, naročito za male vrednosti parametra α. Preko ovog parametra se zadaju težinski faktori s kojima starije vrednosti posmatrane veličine utiču na trenutnu procenu.



Slika 8.12: *Procena vremena potrebnog za prenos u oba smera.*

Kada je poznata vrednost $SRTT$, vrednost retransmisionog tajmera, RTO, prema dokumentu RFC 793 određuje se po obrascu

$$RTO(k+1) = \min\{UBOUND,\ \max[LBOUND,\ \beta SRTT(k+1)]\},$$

gde su $UBOUND$ i $LBOUND$ redom gornja i donja granica vrednosti tajmera, dok je β konstanta. Primetimo da se u dokumentu RFC 793 ne zadaju konkretne vrednosti parametara α i β, već se kao „primer" navode opsezi vrednosti od 0,8 do 0,9 za α i od 1,3 do 2 za β.

Čitaocu se preporučuje da za različite vrednosti β odredi i skicira vrednosti retransmisionog tajmera, RTO.

Zadatak 8.13 Jacobson je predložio sledeći obrazac za podešavanje vrednosti retransmisionog tajmera:

$$RTO(k+1) = SRTT(k+1) + f \cdot SDEV(k+1),$$

pri čemu je

$$SRTT(k+1) = (1-g) \cdot SRTT(k) + g \cdot RTT(k+1),$$
$$SDEV(k+1) = (1-h) \cdot SDEV(k) + h \cdot |SERR(k+1)|,$$
$$SERR(k+1) = RTT(k+1) - SRTT(k).$$

Za vrednosti parametara $g = 1/8$, $h = 1/4$ i $f = 4$ i izmerene vrednosti vremena potrebnog za prenos u oba smera kao u prethodnom zadatku, odredite vrednosti retransmisionog tajmera, RTO.

Jacobsonov algoritam je prihvaćen kao standard za TCP. Ideja na kojoj se zasniva je podešavanje vrednosti retransmisionog tajmera RTO na osnovu procenjene standardne devijacije vremena potrebnog za prenos u oba smera, RTT. Što je ova vrednost veća, biće brža i promena vrednosti RTO.

Sređivanjem navedenih izraza, RTO možemo izraziti samo preko RTT, na sledeći način:

$$SRTT(k+1) = g\sum_{i=1}^{k+1}(1-g)^{k+1-i}RTT(i),$$

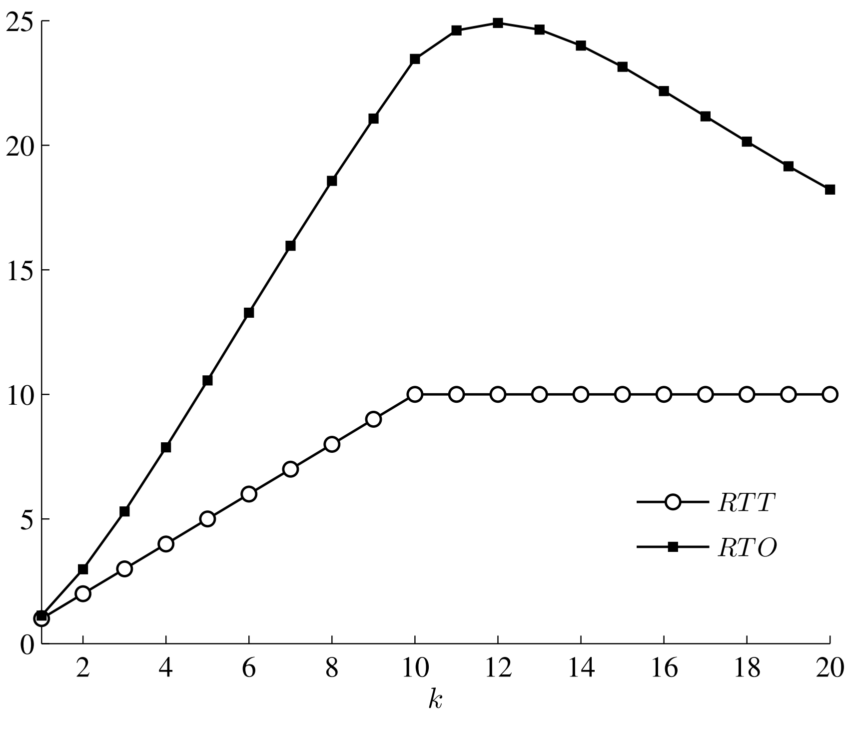
$$SDEV(k+1) = h\sum_{i=1}^{k+1}(1-h)^{k+1-i}\left|SERR(i)\right|,$$

$$SERR(k+1) = RTT(k+1) - g\sum_{i=1}^{k}(1-g)^{k-i}RTT(i),$$

pa je

$$\begin{aligned} RTO(k+1) =& g\sum_{i=1}^{k+1}(1+g)^{k+1-i}RTT(i)+ \\ &+ fh\sum_{i=1}^{k+1}(1-h)^{k+1-i}\left|RTT(i) - g\sum_{j=1}^{i-1}(1-g)^{i-1-j}RTT(j)\right|. \end{aligned}$$

Rezultati izračunavanja dati su grafički i tabelarno.



Slika 8.13: *Jacobsonov algoritam za podešavanje retransmisionog tajmera.*

Vrednosti retransmisionog tajmera

k	$RTT(k)$	$RTO(k)$
1	1	1,12
2	2	2,98
3	3	5,30
4	4	7,87
5	5	10,56
6	6	13,28
7	7	15,96
8	8	18,57
9	9	21,07
10	10	23,46
11	10	24,61
12	10	24,91
13	10	24,64
14	10	24,00
15	10	23,15
16	10	22,17
17	10	21,16
18	10	20,14
19	10	19,16
20	10	18,22

Zadatak 8.14 Izvedite izraz za verovatnoću gubitka segmenta u TCP. Koja veza postoji između ove vrednosti i protoka TCP konekcije?

Posmatraćemo dugotrajnu TCP konekciju. U analizi ćemo zanemariti fazu sporog starta, jer zbog eksponencijalnog porasta veličine otvora prozora zagušenja ona kratko traje.

Neka se gubitak dešava kada otvor prozora zagušenja iznosi W. Broj segmenata koji su preneseni u tome ciklusu je

$$N = \sum_{i=0}^{W/2} \left(\frac{W}{2} + i \right) = \frac{3}{8} W^2 + \frac{3}{4} W.$$

Verovatnoća gubitka segmenta stoga je

$$L = \frac{1}{N} = \frac{1}{\frac{3}{8} W^2 + \frac{3}{4} W}.$$

Ako se gubici segmenata retko dešavaju, biće $W^2 \gg W$, pa je

$$L \approx \frac{8}{3W^2},$$

odakle je

$$W \approx \sqrt{\frac{8}{3L}}.$$

U trenutku gubitka segmenta, protok je iznosio najviše $W \cdot MSS/t_{rtt}$, gde je MSS maksimalna veličina segmenta. Protok potom pada na polovinu ove vrednosti i u svakoj rundi (intervalu trajanja t_{rtt}) uvećava se za MSS/t_{rtt} sve dok ponovo ne bude dostigao maksimalnu vrednost. Prosečni protok stoga je

$$B = \frac{3}{4}\frac{W}{t_{rtt}} MSS,$$

odnosno

$$B = \frac{3}{4}\sqrt{\frac{8}{3L}}\frac{MSS}{t_{rtt}} = \frac{1{,}22MSS}{\sqrt{L}\,t_{rtt}}.$$

Zadatak 8.15 Kolika je verovatnoća gubitka TCP segmenta u konekciji čiji protok iznosi 10 Gb/s?

Polazimo od izraza za protok TCP konekcije koji smo izveli u prethodnom zadatku:

$$B = \frac{1{,}22MSS}{\sqrt{L}\,t_{rtt}}.$$

Usvojićemo standardne vrednosti $MSS = 1500$ B i $t_{RTT} = 100$ ms. Za $B = 10$ Gb/s, tada ćemo dobiti $L = 2{,}14 \cdot 10^{-10}$.

Zadatak 8.16 Izvedite izraz za vreme potrebno da otvor prozora zagušenja u TCP poraste od vrednosti $W/2$ na W, gde je W njegova maksimalna veličina.

Prosečan protok TCP konekcije je

$$B = \frac{1{,}22MSS}{\sqrt{L}\,t_{rtt}},$$

odakle je

$$L = \left(\frac{1{,}22MSS}{B\,t_{rtt}}\right)^2.$$

Označimo sa T vreme potrebno da otvor prozora zagušenja poraste s $W/2$ na W; očigledno, ovo vreme odgovara intervalu između dvaju uzastopnih gubitaka segmenta. Tokom njega, izvor će s protokom B emitovati $1/L$ segmenata dužine po MSS, pa je

$$T = \frac{MSS}{L\,B}.$$

Odavde je, konačno,

$$T = \frac{B}{MSS}\left(\frac{t_{rtt}}{1{,}22}\right)^2.$$

Zadatak 8.17 Izvedite izraze za verovatnoću gubitka segmenta i protok konekcije modifikovanog TCP, u kome se veličina otvora prozora zagušenja posle faze sporog starta povećava s multiplikativnim faktorom $1 + a$, $0 < a < 1$ svaki put kada predajnik primi pozitivnu potvrdu.

Označimo maksimalnu veličinu otvora prozora zagušenja s W. Broj segmenata koji se pošalju dok otvor poraste s $W/2$ na W je

$$N = \sum_{i=0}^{k} \frac{W}{2} (1+a)^i,$$

gde je $k = \log_{1+a} 2$. Sumiranjem dobijamo

$$N = \frac{2a+1}{2a} W.$$

Vreme koje protekne između dvaju gubitaka je

$$T = kt_{rtt} = (\log_{1+a} 2) t_{rtt}.$$

Verovatnoća gubitka segmenta je

$$L = \frac{2a}{(2a+1)W}.$$

Prosečni protok ovakve konekcije je

$$B = MSS \frac{N}{(k+1)t_{rtt}} = \frac{MSS}{(1 + \log_{1+a} 2) L t_{rtt}}.$$

9. Aplikacioni sloj

Zadatak 9.1 Objasnite šta se dešava kada *web browser* zahteva neki URL.

Po pravilu, korisnik u *browser* unosi URL, a *browser* onda pomoću DNS saznaje IP adresu koja odgovara tome URL-u. Iz unetog URL-a, *browser* izdvaja naziv hosta i predaje ga klijentskoj strani DNS aplikacije, koja se izvršava na korisnikovom računaru. DNS klijent potom šalje upit u vezi s ovim hostom DNS serveru; kao odgovor, dobiće IP adresu hosta. Ova adresa se potom prosleđuje *browseru*, koji inicira TCP konekciju ka HTTP serverskom procesu lociranom na portu 80 te IP adrese.

Zadatak 9.2 Nađite naziv i IP adresu jednog DNS servera u *whois* bazama podataka koje su dostupne na internetu.

Upitom u bazu Registra nacionalnog internet domena Srbije (`www.rnids.rs/lat/whois-upit`), saznajemo da jedan od DNS servera pripada Elektrotehničkom fakultetu u Beogradu (domen `www.etf.rs`). Njegov URL je `ns2.etf.rs`, a IP adresa 147.91.8.62.

Zadatak 9.3 Pomoću ARIN *whois* baze odredite opseg IP adresa koji koristi Elektrotehnički fakultet u Beogradu.

U prethodnom zadatku smo videli da je IP adresa jednog od DNS servera u mreži ETF Beograd 147.91.8.62. Upitom

```
http://whois.arin.net/rest/nets;q=147.91.8.62?showDetails=true
```

dobijamo da se koristi opseg IP adresa 147.91.0.0 – 147.91.255.255, ili 147.91.0.0/16.

Zadatak 9.4 Korišćenjem Linux alata `dig` s jednog od *root* servera `[a-m].root-servers.net`, nađite sekvencu DNS upita za adresu `www.etf.rs`.

Izvršavanjem naredbe

```
dig +norecurse @a.root-servers.net any www.etf.rs
```

dobijamo odgovor

```
;; AUTHORITY SECTION:
etf.rs. 10800 IN NS ns1.nic.rs.
etf.rs. 10800 IN NS ns2.etf.rs.
```

Naredni upit ćemo poslati npr. prvom nađenom DNS serveru:

```
dig +norecurse @ns1.nic.rs any www.etf.rs
```

.

Kao odgovor ćemo dobiti

```
www.etf.rs. 3600 IN CNAME vhost4.etf.rs..
```

Konačno, slanjem trećeg upita

```
dig +norecurse @ns1.nic.rs any vhost4.etf.rs
```

dobijamo odgovor

```
vhost4.etf.rs. 3600 IN A 147.91.14.197.
```

Zadatak 9.5 Dat je sadržaj HTTP GET poruke, dobijen programom Wireshark. Sekvence `\r` i `\n` odgovaraju redom simbolima *carriage return* i *line feed*.

```
GET /predmeti/ot4mis/index.htm HTTP/1.1\r\n
Host: telekomunikacije.etf.rs\r\n
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:14.0)
Gecko/20100101 Firefox/14.0.1\r\n
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
Accept-Language: en-us,en;q=0.5\r\n
Accept-Encoding: gzip, deflate\r\n
Connection: keep-alive\r\n
\r\n
```

a) Koji je URL traženog dokumenta?

b) Koja verzija HTTP se izvršava u brauzeru?

c) Koji je tip brauzera koji inicira ovu poruku?

d) Koja je IP adresa hosta na kome je brauzer?

e) Zahteva li se neperzistentna ili perzistentna konekcija?

a) Traženi URL je `telekomunikacije.etf.rs/predmeti/ot4mis/index.htm`.

b) U brauzeru se izvršava HTTP verzija 1.1.

c) Poruku inicira brauzer Mozilla/5.0.

d) IP adresa hosta na kome je brauzer ne može se saznati iz sadržaja HTTP GET poruke.

e) Zahteva se perzistentna konekcija, jer je vrednost polja `Connection:` *keep-alive*.

Zadatak 9.6 Na HTTP GET poruku iz prethodnog zadatka dobijen je odgovor čiji je početak dat u nastavku:

```
HTTP/1.1 200 OK\r\n
Date: Sun, 25 Nov 2012 16:52:48 GMT\r\n
Server: Apache/2.0.59 (Gentoo) mod_ssl/2.0.59 OpenSSL/0.9.7d\r\n
Last-Modified: Tue, 05 Jun 2012 08:33:03 GMT\r\n
ETag: "6d4006-575b-7fba4dc0"\r\n
Accept-Ranges: bytes\r\n
Content-Length: 22363\r\n
Connection: close\r\n
Content-Type: text/html\r\n
\r\n
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">\n
<!-- saved from url=(0050)http://telekomunikacije.etf.rs/zaposleni.htm -->\n
<HTML><HEAD>\n
<meta http-equiv="Content-Language" content="en-us">\n
<TITLE>Katedra za telekomunikacije</TITLE>\n
<META content="text/html; charset=windows-1250" http-equiv=Content-Type>\n
<SCRIPT language=JavaScript> \n
```

a) Je li server pronašao traženi dokument?

b) Kada je generisan odgovor?

c) Kada je dokument poslednji put izmenjen?

d) Koliko bajtova ima u dokumentu?

e) Je li server prihvatio perzistentnu konekciju?

a) Statusni kod 200 i fraza `OK` ukazuju na to da je server uspešno pronašao traženi dokument.

b) Odgovor je generisan u nedelju, 25. novembra 2012. godine, u 16:52:48 GMT.

c) Dokument je poslednji put izmenjen u utorak, 5. juna 2012. godine u 8:33:03 GMT.

d) Dokument se sastoji od 22 363 bajta.

e) Server nije prihvatio perzistentnu konekciju, jer je sadržaj polja `Connection:` *close*.

Zadatak 9.7 U *web browseru* se zahteva URL čija IP adresa nije lokalno keširana, pa je potrebno posetiti n DNS servera. Za prvih k servera, vreme potrebno za prenos u oba smera (RTT) iznosi po D_1, a za preostale po D_2. Neka se, dalje, na traženoj internet-stranici nalazi m malih objekata, tako da RTT između lokalnog hosta i servera za svaki objekat iznosi RTT_o. Vreme potrebno za prenos objekta može se zanemariti. Koliko vremena protekne od kada korisnik klikne na link, pa dok primi sve objekte, ako se koristi:

a) neperzistentni HTTP bez paralelnih TCP konekcija,

b) neperzistentni HTTP s p paralelnih TCP konekcija, ili

c) perzistentni HTTP?

a) Ukupno vreme potrebno da *browser* sazna IP adresu stranice je $kD_1+(n-k)D_2$. Potom je potrebno uspostaviti m TCP konekcija, te zahtevati i primiti svaki od objekata. Ukupno kašnjenje u ovome slučaju stoga iznosi

$$\tau_a = kD_1 + (n-k)D_2 + mRTT_o.$$

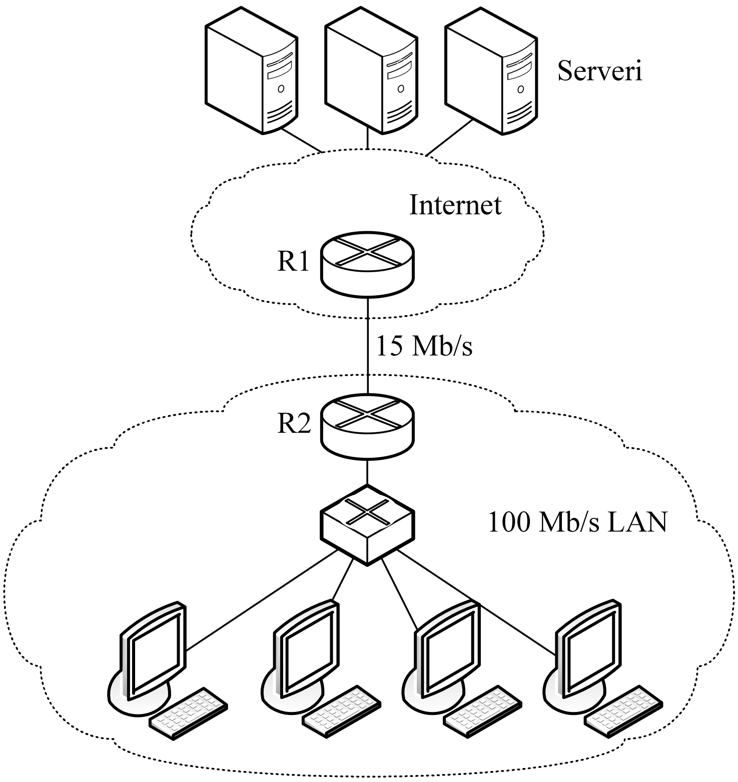
b) Sada se može uspostaviti do p paralelnih TCP konekcija. Ukupno kašnjenje u ovome slučaju stoga iznosi

$$\tau_a = kD_1 + (n-k)D_2 + \left\lceil \frac{m}{p} \right\rceil RTT_o.$$

c) TCP konekcija sada se uspostavlja jedanput i koristi za sve objekte. Kašnjenje stoga iznosi

$$\tau_c = kD_1 + (n-k)D_2 + RTT_o.$$

Zadatak 9.8 Mreža u poslovnoj zgradi povezana je na internet prema slici 9.8. Kapacitet pristupnog linka je 15 Mb/s. Svake sekunde, *browseri* u proseku generišu 15 zahteva prema serverima. Prosečna veličina odgovora je 960 000 b. Prosečno kašnjenje od kada ruter R1 prosledi HTTP zahtev serverima do kada dobije odgovor je 2 s. Uz pretpostavku da se upiti generišu saglasno Poissonovom slučajnom procesu, a da je veličina odgovora eksponencijalno raspodeljena slučajna promenljiva, izvedite izraz za prosečno vreme odziva za slučajeve kada se proksi-server



Slika 9.8: *Povezivanje lokalne mreže na internet.*

a) ne koristi i

b) koristi, za *miss rate* 0,4.

Posmatrajmo pristupni link. U smeru naviše (*uplink*), njime se prenose kratki upiti, pa je ovde kašnjenje zanemarivo. Ovi upiti, koji odgovaraju Poissonovom slučajnom procesu, potom se obrađuju u serverima. Odgovori će u smeru naniže (*downlink*) pristizati takođe kao Poissonov slučajni proces, a njihove dužine biće eksponencijalno raspodeljene; stoga je opravdano da na *downlinku* primenimo model servisnog sistema M/M/1 s parametrima $\lambda = 15\ \mathrm{s}^{-1}$, $\mu^{-1} = \frac{960\,000\ \mathrm{b}}{15\ \mathrm{Mb/s}} = 64\ \mathrm{ms}$.

a) Prosečno kašnjenje na linku je

$$T = \frac{1}{\mu - \lambda} = 1{,}6\ \mathrm{s}.$$

Uz uračunavanje kašnjenja od rutera R1 ka serverima, ukupno vreme odziva je

$$T_{tot} = T + T_{serv} = 3{,}6\ \mathrm{s}.$$

b) Sada se 60% zahteva opslužuje unutar lokalne mreže, sa zanemarivim vremenom odziva. Preostalih 40% zahteva se upućuje preko pristupnog linka i za njih vreme odziva iznosi

$$T_{internet} = \frac{1}{\mu - 0{,}4\lambda} + T_{serv} = 2{,}1\ \mathrm{s}.$$

Prosečno vreme odziva stoga je

$$T'_{tot} = 0{,}6 \cdot 0 + 0{,}4\,T_{internet} = 0{,}84 \text{ s}.$$

Zadatak 9.9 S kog je hosta došla poruka elektronske pošte čije je zaglavlje dato?

```
Return-Path: <europeanbusinesspress@mailer.subscription.co.uk>
Delivered-To: milan@etf.rs
Received: from localhost (avs1.etf.rs [147.91.14.172])
     by mx2.etf.bg.ac.rs (Postfix) with ESMTP id 553D96012F
     for <milan@etf.rs>; Thu, 21 Jan 2021 11:07:09 +0100 (CET)
Received: from mx1.etf.bg.ac.rs ([147.91.14.169])
     by localhost (avs1.etf.rs [147.91.14.172]) (amavisd-new, port 10024)
     with ESMTP id SKoFG+Ruf+dB for <milan@etf.rs>;
     Thu, 21 Jan 2021 11:07:09 +0100 (CET)
Received: from mailer4.neteffekt.co.uk
  (mailer4.neteffekt.co.uk [78.33.8.116])
     by mx1.etf.bg.ac.rs (Postfix) with ESMTP id DD404120132
     for <milan@etf.rs>; Thu, 21 Jan 2021 11:07:09 +0100 (CET)
Received: by mailer4.neteffekt.co.uk id hlidlu17smgc for <milan@etf.rs>;
Thu, 21 Jan 2021 11:07:09 +0000
(envelope-from <europeanbusinesspress@mailer.subscription.co.uk>)
From: "Texas Instruments" <europeanbusinesspress@mailer.subscription.co.uk>
To: "milan@etf.rs" <milan@etf.rs>
Date: Thu, 21 Jan 2021 11:07:09 +0000
Reply-to: Texas Instruments <europeanbusinesspress@airegistrations.co.uk>
Subject: TI Analog Design Newsletter
```

Host šalje poruku elektronske pošte agentu za prenos (MTA). Zaglavlje primljene poruke sadrži sekvencu agenata koji su uključeni u njeno prosleđivanje do primaoca. Prateći ovu sekvencu u zaglavlju poruke iz zadatka, zaključujemo da je traženi host `mailer4.neteffekt.co.uk`.

Zadatak 9.10 Datoteka veličine F distribuira se od servera ka N korisnika. Kapacitet odlaznog linka od servera je u_s, dok su kapaciteti odlaznog i dolaznog linka i-tog korisnika, $i = 1, \ldots, N$ redom u_i i d_i. Izvedite izraz za minimalno vreme potrebno da se datoteka prosledi korisnicima, ako se koristi

a) arhitektura klijent-server ili

b) arhitektura P2P (*peer-to-peer*).

a) U arhitekturi klijent-server, server šalje primerak kompletne datoteke do svakog od N korisnika. Ukupna količina podataka koju server mora preneti svojim odlaznim linkom je NF, pa vreme potrebno za prosleđivanje datoteke korisnicima nije kraće od NF/u_s.

S druge strane, vreme potrebno da i-ti korisnik preuzme datoteku je F/d_i; stoga pre isteka vremena $F/\min_i\{d_i\}$ svi korisnici neće imati na raspolaganju kompletnu datoteku.

Iz ovih razmatranja, zaključujemo da je donja granica vremena potrebnog da bi se datoteka prosledila korisnicima u slučaju arhitekture klijent-server

$$D_{cs} = \max\left\{\frac{NF}{u_s}, \frac{F}{\min\limits_i\{d_i\}}\right\}.$$

b) U arhitekturi P2P, server šalje samo jedan primerak datoteke, za šta mu je potrebno vreme F/u_s. Najsporiji korisnik ponovo neće preuzeti svoj primerak pre vremena $F/\min_i\{d_i\}$. Za razliku od slučaja a), korisnici sada učestvuju u distribuiranju datoteke, tako da je ukupan odlazni kapacitet sistema $u_s + u_1 + \cdots + u_N$. Sistem mora isporučiti količinu podataka NF s ovim protokom, za šta je potrebno vreme $NF/(u_s + u_1 + \cdots + u_N)$. Donja granica vremena potrebnog da bi se datoteka proslediła korisnicima u slučaju arhitekture *peer-to-peer* stoga je

$$D_{P2P} = \max\left\{\frac{F}{u_s}, \frac{F}{\min\limits_i\{d_i\}}, \frac{NF}{u_s + \sum\limits_{i=1}^{N} u_i}\right\}.$$

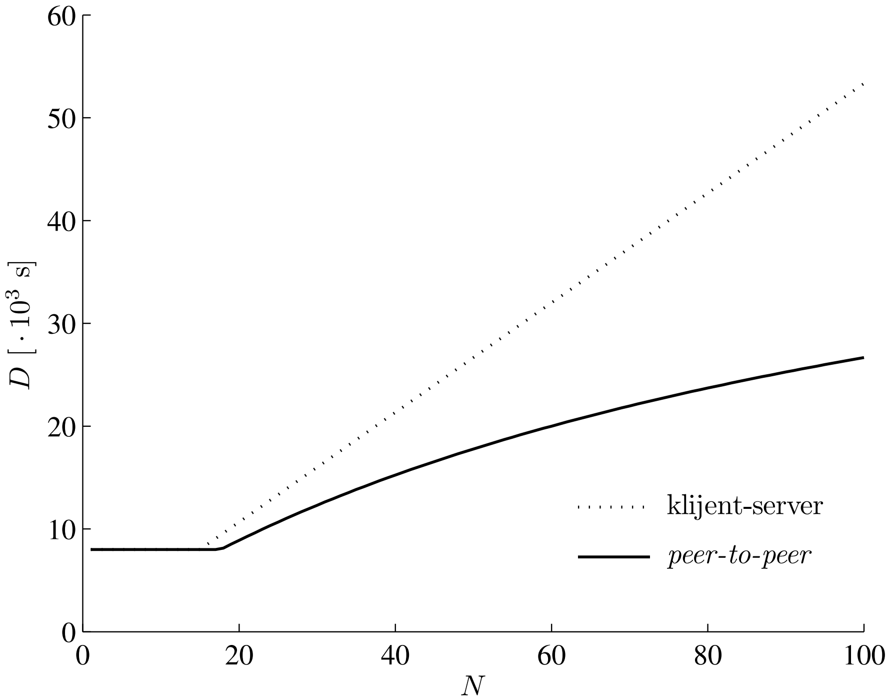
Zavisnosti D_{cs} i D_{P2P} od broja korisnika, za $F = 16$ Gb, $u_s = 30$ Mb/s, $d_i = 2$ Mb/s i $u_i = 300$ kb/s, prikazane su na slici 9.10. Vidimo da je za veliki broj korisnika prednost na strani arhitekture P2P.

Zadatak 9.11 Sesija internet-telefonije ostvaruje se posredstvom protokola Skype. Protok digitalizovanog signala govora je 8000 B/s. Svakih 20 ms formira se blok, koji se prenosi posredstvom RTP, UDP i IP.

a) Koliko se bita zaglavlja dodaje bloku pre slanja?

b) Koliki je protok podataka na odlaznom linku ako se IP datagram emituje svakih 40 ms?

a) Zaglavlje IPv4 datagrama je veličine 20 B, UDP paketa 8 B, a RTP paketa 12 B. Ukupan broj bita zaglavlja stoga je $(20 + 12 + 8) \cdot 8 = 320$.



Slika 9.10: *Poređenje arhitektura za distribuciju datoteka.*

b) Svaki blok se sastoji od $8 \cdot 8000 \cdot 0{,}02 = 1280$ bita podataka. Uz 320 b zaglavlja, ukupna veličina IP datagrama je 1600 b. IP datagrami se emituju na 40 ms, pa je protok na odlaznom linku 40 kb/s.

Zadatak 9.12 Konferencijski poziv sa $N > 2$ učesnika realizuje se posredstvom protokola Skype. Svaki učesnik generiše konstantan protok r. Koliki je ukupan generisani protok ako se koristi

a) audio poziv ili

b) video poziv?

a) Kada bi svaki učesnik slao kopiju svog audio „strima" drugim učesnicima, trebalo bi preneti $N(N-1)$ strimova. Da bi se smanjio ovaj broj, učesnici Skype konferencijskog audio poziva šalju svoje strimove inicijatoru poziva, koji ih kombinuje sa svojim strimom i potom emituje svima. Ukupan broj strimova koji se šalju je $2(N-1)$. Inicijator poziva generiše protok $(N-1)r$, svaki od ostalih učesnika r, tako da je ukupan generisani protok u mreži $2(N-1)r$.

b) Svaki učesnik sada šalje svoj video strim Skype serveru u Estoniji, koji mu onda prosleđuje strimove preostalih $N-1$ učesnika. Ukupan generisani protok je $N(N-1)r$.

Zadatak 9.13 Korišćenjem programskog jezika Python3 i njegovog modula `socket`, realizujte UDP klijent-server aplikaciju u kojoj:

- klijent čita tekstualni unos s tastature i prosleđuje ga serveru,
- server prima ovaj unos i sva mala slova u njemu prevodi u velika,
- server šalje modifikovane podatke klijentu,
- klijent prima modifikovane podatke i prikazuje ih na monitoru.

Pri programiranju klijenta, IP adresu ili URL servera zadajemo kao string (na primer `'78.30.142.18'`), dok za port proizvoljno biramo 12000.

Klijentski „socket" pravimo naredbom `socket`, čiji su argumenti redom adresna familija (za IPv4 biramo `AF_INET`) i vrsta socketa (za UDP je `SOCK_DGRAM`). Funkcijom `input` ćemo učitati podatak s tastature, prevešćemo ga u bajtove metodom `encode` i zatim metodom `sendto` poslati odredišnom hostu posredstvom otvorenog UDP socketa.

Odgovor ćemo primiti metodom `recvfrom`; kao njen argument, navešćemo potrebnu veličinu prijemnog bafera, npr. 2048 B. Da bismo odgovor, dekodiran iz povorke bajtova u string, prikazali na monitoru, poslužićemo se funkcijom `print`. Na kraju ćemo metodom `close` zatvoriti socket.

Listing klijentskog programa dat je u nastavku.

```
from socket import *
serverName = '78.30.142.18'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
message = input('Unesite recenicu malim slovima: ')
clientSocket.sendto(message.encode(), (serverName, serverPort))
modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
print('Odgovor:', modifiedMessage.decode())
clientSocket.close()
```

Na strani servera, definisaćemo port (12000) i napraviti UDP socket. Potom ćemo metodom `bind` ovome socketu dodeliti port 12000.

Podatke ćemo primati u beskonačnoj `while` petlji. Metodom `upper`, mala slova ćemo prevesti u velika. Na kraju ćemo izmenjeni podatak vratiti pošiljaocu (klijentu).

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))
print("Server je spreman za prijem poruke.")
while True:
       message, clientAddress = serverSocket.recvfrom(2048)
       modifiedMessage = message.upper()
       serverSocket.sendto(modifiedMessage, clientAddress)
```

Pošto je UDP nekonektivni protokol, svejedno je hoćemo li pri testiranju prvo pokrenuti serversku ili klijentsku aplikaciju.

Zadatak 9.14 Ponovite prethodni zadatak za slučaj TCP.

I ovde ćemo prvo definisati IP adresu servera i port. Potom ćemo napraviti TCP socket (drugi argument funkcije `socket` sada će biti `SOCK_STREAM`). Pošto je TCP konektivni protokol, metodom `connect` iniciraćemo TCP konekciju između klijenta i servera. S tastature ćemo pročitati podatak i poslati ga preko socketa. Primljeni odgovor ćemo prikazati na monitoru i na kraju ćemo zatvoriti socket.

```
from socket import *
serverName = '78.30.142.18'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
message = input('Unesite recenicu malim slovima: ')
clientSocket.send(message.encode())
modifiedMessage = clientSocket.recv(2048)
print('Odgovor:', modifiedMessage.decode())
clientSocket.close()
```

Na strani servera, napravićemo TCP socket i vezaćemo ga za port 12000. Metodom `listen` čekaćemo da klijentski proces pošalje podatke na ovaj socket; njen argument ukazuje na maksimalni broj konekcija koje čekaju u redu.

Kada budu prispeli podaci od klijenta, ulazi se u `while` petlju. Metodom `accept` na strani servera se pravi novi socket (`connectionSocket`), koji se dodeljuje tekućem klijentu. Dovršava se procedura „rukovanja” i formira se TCP konekcija između socketa na strani klijenta (`clientSocket`) i na strani servera (`connectionSocket`), po kojoj se onda razmene podaci. Na kraju se zatvara `connectionSocket`, ali `serverSocket` ostaje otvoren, tako da naredni klijent može tražiti uspostavljanje konekcije i poslati svoje podatke.

Listing programa dat je u nastavku.

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind(('', serverPort))
serverSocket.listen(1)
print('Server je spreman za prijem poruke.')
while True:
       connectionSocket, addr = serverSocket.accept()
       message = connectionSocket.recv(2048)
```

```
        modifiedMessage = message.upper()
        connectionSocket.send(modifiedMessage)
        connectionSocket.close()
```

TCP je konektivni protokol, pa je stoga neophodno da pri testiranju prvo pokrenemo serversku aplikaciju.

Zadatak 9.15 Korišćenjem programskog jezika Python3, realizujte klijentsko-serversku aplikaciju kojom se implementira ping test posredstvom UDP. Klijent serveru šalje 10 *ping* poruka, koje se sastoje od slovne niske `Ping` posle koje slede redni broj poruke i trenutak njenog generisanja. Za svaki primljeni odgovor (*pong*) treba odrediti kašnjenje u prenosu (RTT); na ekranu treba prikazati tekst odgovora i RTT. Pošto je UDP nepouzdan protokol, ping i pong poruke mogu se izgubiti; verovatnoća ovoga događaja je 30%. Ako klijent ne dobije odgovor u roku od jedne sekunde, pretpostaviće da se paket izgubio u prenosu, pa tada na ekranu treba prikazati poruku „Zahtev je istekao".

Ping (*Packet Internetwork Groper*) testom se ispituje je li udaljeni host aktivan, kao i koliko je kašnjenje pri komunikaciji s njim. Test je opisan dokumentom RFC 2151 i zasniva se na korišćenju ICMP (*Internet Control Message Protocol*) Echo poruka. U ovome zadatku, ping test ćemo realizovati korišćenjem UDP, modifikovanjem klijentsko--serverske aplikacije iz zadatka 9.13, tako da se:

- otvori UDP socket čiji *timeout* parametar iznosi 1 s,
- formira ping poruka zadatog formata i pošalje serveru posredstvom UDP socketa,
- realizuje zadata verovatnoća gubitka poruka tako što će server ignorisati 30% primljenih ping zahteva,
- iz primljenog odgovora računa kašnjenje u prenosu,
- na ekranu prikazuju relevantni rezultati testa,
- novi ping zahtev generiše na svaku sekundu.

Listinzi programa kojima se implementira ovaj test dati su u nastavku. Napominjemo da je pre njihovog izvršavanja, iz koda potrebno ukloniti dijakritičke znake, koji su ovde zadržani radi bolje čitljivosti.

```
# UDPPingerClient.py
# Programu se predaju dva argumenta:
# 1) IP adresa hosta na kome se izvršava UDPPingServer.py
# 2) broj UDP porta

import sys, time
from socket import *
```

```
# Čitamo IP adresu i port
argv = sys.argv
host = argv[1]
port = int(argv[2])

timeout = 1

# Pravimo UDP socket
clientsocket = socket(AF_INET, SOCK_DGRAM)
clientsocket.settimeout(timeout)

# Redni broj ping poruke
p = 0

# Šaljemo 10 ping zahteva
while p < 10:
     p += 1
     # Formiramo ping poruku
     data = "Ping " + str(p) + " " + time.asctime()

     try:
        # Trenutak slanja ping poruke
        RTTb = time.time()
        # Šaljemo UDP paket s ping porukom
        clientsocket.sendto(data.encode(),(host, port))
        # Primamo odgovor
        message, address = clientsocket.recvfrom(1024)
        # Trenutak prijema odgovora
        RTTa = time.time()
        # Prikazujemo odgovor
        print("Odgovor od " + address[0] + ": " + message)
        # Računamo RTT
        print("RTT: " + str(RTTa - RTTb))
     except:
        # Server ne odgovara, pretpostavljamo da se paket izgubio
        print("Zahtev je istekao.")
        continue

# Zatvaramo socket
clientsocket.close()
```

```
# UDPPingerServer.py

from socket import *
import random

# Pravimo UDP socket
```

```
port = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', port))

while True:
     # Primamo ping poruku od klijenta
     message, address = serverSocket.recvfrom(1024)
     # Realizujemo gubitak paketa s verovatnoćom 30%
     rand = random.randint(0, 10)
     if rand < 4:
          continue

     # Ako nije bilo gubitka, klijentu se vraća odgovor
     serverSocket.sendto(message, address)
```

Prilikom testiranja, klijentskom programu kao argumente iz komandne linije treba predati IP adresu serverskog hosta i broj porta serverske aplikacije (u našem primeru 12000). Ako se obe aplikacije izvršavaju na istom hostu, tada se kao prvi argument navodi tzv. *localhost* IP adresa `127.0.0.1`.

Zadatak 9.16 U programskom jeziku Python3, realizujte ping aplikaciju koja koristi ICMP. Zahtevanom hostu, ping poruka se šalje na svaku sekundu. Ako se odgovor ne bude dobio u roku od jedne sekunde, pretpostaviće da se paket izgubio u prenosu, ili da server na hostu nije aktivan.

ICMP (*Internet Control Message Protocol*) opisan je dokumentom RFC 792. Za potrebe implementiranja ping testa, koristićemo njegove poruke `ECHO` i `ECHO REPLY`. Napisaćemo samo klijentki program, jer su serverske funkcionalnosti ugrađene u sve moderne operativne sisteme.

Struktura `ECHO` i `ECHO REPLY` poruka prikazana je na slici.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |     Code      |          Checksum             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Identifier          |        Sequence Number        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Data ...
+-+-+-+-+-
```

Slika 9.16: *Zaglavlje ICMP paketa.*

Naša aplikacija treba da pravi ECHO poruke. Vrednost polja Type u njihovom zaglavlju je 8, a Code 0. Kontrolna suma (Checksum) se izračunava kao komplement jedinice sume (u sistemu komplementa 1) 16-bitskih reči iz ICMP paketa (zaglavlja i korisnog sadržaja), pri čemu se pretpostavlja da su inicijalno u polje Checksum upisane sve nule. Ako se ICMP paket sastoji od neparnog broja okteta, za potrebe izračunavanja kontrolne sume mu se nadovezuje oktet nula. U polje Identifier upisaćemo identifikaciju ICMP procesa, a kao vrednost polja Sequence Number koristićemo 1.

Listing programa kojim se „pinguje" host www.google.com dat je u nastavku. Pre njegovog izvršavanja, iz koda je potrebno ukloniti dijakritičke znake.

```
from socket import *
import os
import sys
import struct
import time
import select
import binascii

ICMP_ECHO_REQUEST = 8

def checksum(string):
     # Računanje kontrolne sume zaglavlja ICMP paketa
     csum = 0
     countTo = (len(string) // 2) * 2

     count = 0
     while count < countTo:
          thisVal = ord(string[count+1]) * 256 + ord(string[count])
          csum = (csum + thisVal) & 0xffffffff
          count = count + 2

     if countTo < len(string):
          csum = csum + ord(string[len(string) - 1])
          csum = csum & 0xffffffff

     csum = (csum >> 16) + (csum & 0xffff)
     csum = csum + (csum >> 16)
     answer = ~csum & 0xffff
     answer = answer >> 8 | (answer << 8 & 0xff00)
     return answer


def sendOnePing(mySocket, destAddr, ID):
     # Procedura za slanje jednog ping zahteva
     # Zaglavlje ICMP paketa čine:
     # tip (8 b), kod (8 b), kontrolna suma (16 b),
     # identifikacija (16 b) i redni broj sekvence (16 b)
```

```
    myChecksum = 0
    # Pravimo inicijalno zaglavlje s kontrolnom sumom jednakom nuli
    header = struct.pack("bbHHh", ICMP_ECHO_REQUEST, 0, myChecksum, ID, 1)
    # "bbHHh" je format zapisa:
    # b - 1 B integer (signed char),
    # H - 2 B integer (unsigned short),
    # h - 2 B integer (short)
    # U polju korisnog podatka prenosimo informaciju o trenutku slanja
    data = struct.pack("d", time.time())
    # d - 1 B float (double)
    # Računamo kontrolnu sumu podataka i inicijalnog zaglavlja
    myChecksum = checksum(str(header + data))
    # Kontrolnu sumu u zaglavlje upisujemo u formatu "big endian"
    # (prvo viši, pa niži bajt)
    myChecksum = htons(myChecksum)

    header = struct.pack("bbHHh", ICMP_ECHO_REQUEST, 0, myChecksum, ID, 1)
    packet = header + data

    mySocket.sendto(packet, (destAddr, 1))


def receiveOnePong(mySocket, ID, timeout, destAddr):
    # Procedura za prijem jednog pong odgovora
    timeLeft = timeout

    while True:
        startedSelect = time.time()
        whatReady = select.select([mySocket], [], [], timeLeft)
        howLongInSelect = (time.time() - startedSelect)
        if whatReady[0] == []: # Timeout
            return "Zahtev je istekao."

        timeReceived = time.time()
        recPacket, addr = mySocket.recvfrom(1024)

        # Izdvajamo zaglavlje ICMP paketa iz IP datagrama
        icmpHeader = recPacket[20:28]
        rawTTL = struct.unpack("s", bytes([recPacket[8]]))[0]
        # Konvertujemo binarne podatke u ASCII
        TTL = int(binascii.hexlify(rawTTL), 16)

        # Čitamo podatke iz zaglavlja
        icmpType, code, checksum, packetID, sequence = \
        struct.unpack("bbHHh", icmpHeader)

        if packetID == ID:
            byte = struct.calcsize("d")
            timeSent = struct.unpack("d", recPacket[28:28 + byte])[0]
            return "Odgovor od %s: bytes = %d; time = %f5 ms; TTL = %d" \
```

```
            % (destAddr, len(recPacket), (timeReceived - timeSent)*1000,
            TTL)

        timeLeft = timeLeft - howLongInSelect
        if timeLeft <= 0:
            return "Zahtev je istekao."


def doOneTest(destAddr, timeout):
    # Procedura za izvršavanje jednog testa
    icmp = getprotobyname("icmp")

    mySocket = socket(AF_INET, SOCK_RAW, icmp)

    myID = os.getpid() & 0xFFFF  # ID trenutnog procesa
    sendOnePing(mySocket, destAddr, myID)
    delay = receiveOnePong(mySocket, myID, timeout, destAddr)

    mySocket.close()
    return delay

def ping(host, timeout=1):
    # timeout = 1 znači da ako se odgovor od servera ne bude dobio
    # u roku od 1 s, klijent će pretpostaviti da se zagubila
    # ping ili pong poruka
    dest = gethostbyname(host)
    print("Pingujem " + dest + " pomoću Pythona:")
    print("")
    # Serveru se šalju ping zahtevi na svaku sekundu
    while True:
        delay = doOneTest(dest, timeout)
        print(delay)
        # Ponavljamo posle 1 s
        time.sleep(1)

    return delay

ping("www.google.com")
```

Zadatak 9.17 Korišćenjem programskog jezika Python3, realizujte SMTP klijentsku aplikaciju.

Listing jednostavnog programa kojim se poruka elektronske pošte šalje s adrese `milan@etf.rs` na `rc@etf.rs`, prema osnovnoj specifikaciji SMTP iz dokumenta RFC 5321, dat je u nastavku.

```
from socket import *
```

```
# Pravimo TCP socket
# (broj porta zavisi od izabranog mail servera)
# i čekamo potvrdu uspostave veze
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect(('smtp.etf.bg.ac.rs', 25))
recv = clientSocket.recv(1024).decode()
print(recv)
if recv[:3] != '220':
     print('220: Server nije odgovorio.')

# Serveru šaljemo HELO komandu
clientSocket.send(b'HELO etf.bg.ac.rs\r\n')
recv1 = clientSocket.recv(1024).decode()
print(recv1)
if recv1[:3] != '250':
     print('250: Server nije odgovorio.')

# Šaljemo komandu MAIL FROM
clientSocket.send(b'MAIL FROM: <milan@etf.rs>\r\n')
recv2 = clientSocket.recv(1024).decode()
print(recv2)
if recv2[:3] != '250':
     print('250: Server nije odgovorio.')

# Šaljemo komandu RCPT TO
clientSocket.send(b'RCPT TO: <rc@etf.rs>\r\n')
recv3 = clientSocket.recv(1024).decode()
print(recv3)
if recv3[:3] != '250':
     print('250: Server nije odgovorio.')

# Šaljemo komandu DATA
clientSocket.send(b'DATA\r\n')
recv4 = clientSocket.recv(1024).decode()
print(recv4)
if recv4[:3] != '354':
     print('354: Server nije odgovorio.')

# Šaljemo poruku
clientSocket.send(b'SUBJECT: Pozdrav\r\n')
clientSocket.send(b'\r\nOva poruka je poslata preko Python SMTP klijenta.')
clientSocket.send(b'\r\n.\r\n')
recv5 = clientSocket.recv(1024).decode()
print(recv5)
if recv5[:3] != '250':
     print('250: Server nije odgovorio.')

# Šaljemo komandu QUIT
clientSocket.send(b'QUIT\r\n')
```

```
recv6 = clientSocket.recv(1024).decode()
print(recv6)
if recv6[:3] != '221':
     print('221: Server nije odgovorio.')
```

Neki mail serveri, kao što je npr. Google mail (adresa: `smtp.gmail.com`, port: 587) zahtevaju od klijenta da pre komande `MAIL FROM` izvrši TLS (*Transport Layer Security*) ili SSL (*Secure Sockets Layer*) autentifikaciju. Zainteresovanim čitaocima se prepušta da prošire gornji kod TLS/SSL komandama i da potom isprobaju klijentsku aplikaciju na Google mail serveru.

Zadatak 9.18 Korišćenjem programskog jezika Python3, realizujte web server koji:

- na zahtev klijenta (brauzera) formira TCP socket,
- preko tog socketa prima HTTP zahtev,
- iz zahteva čita naziv tražene datoteke,
- „dohvata" ovu datoteku sa svog hosta,
- na nju dodaje HTTP zaglavlje i tako pravi odgovor (ukoliko zahtevana datoteka nije pronađena, šalje se poruka „`404 Not Found`"),
- šalje odgovor brauzeru posredstvom TCP socketa.

Pretpostavite da server u svakom trenutku opslužuje najviše jedan zahtev.

Listing traženog programa dat je u nastavku, a značenje pojedinih komandi objašnjeno je komentarima. I ovde je pre izvršavanja programa potrebno iz komentara ukloniti dijakritičke znake, koji su zadržani radi čitljivosti koda.

```
from socket import *
import sys

# Pravimo TCP socket
serverSocket = socket(AF_INET, SOCK_STREAM)

# Definišemo broj porta, npr. 6789
serverPort = 6789

# Socketu dodeljujemo IP adresu servera i broj TCP porta
serverSocket.bind(("", serverPort))

# Dozvoljavamo najviše jednu TCP konekciju u datom trenutku
serverSocket.listen(1)
```

```
# Server čeka dolazni zahtev
while True:
    print('Server je spreman...')

    # Uspostavljamo konekciju prema klijentu
    connectionSocket, addr = serverSocket.accept()

    try:
        # Primamo zahtev od klijenta
        message =  connectionSocket.recv(1024).decode()
        # Iz zahteva izdvajamo naziv datoteke;
        # to je drugi deo HTTP zaglavlja
        filename = message.split()[1]
        # Pošto je prvi simbol u izdvojenom nazivu '\',
        # putanju do datoteke čitamo počevši od drugog simbola
        f = open(filename[1:])
        # Sadržaj tražene datoteke smeštamo u privremeni bafer
        outputdata = f.read()
        # Na uspostavljeni socket šaljemo zaglavlje HTTP odgovora
        connectionSocket.send(b"HTTP/1.1 200 OK\r\n\r\n")
        # Šaljemo i sadržaj tražene datoteke
        for i in range(0, len(outputdata)):
            connectionSocket.send(outputdata[i].encode())

        connectionSocket.send(b"\r\n")

        # Zatvaramo klijentski socket
        connectionSocket.close()

    except IOError:
        # Šaljemo HTTP odgovor da datoteka nije pronađena
        connectionSocket.send(b"HTTP/1.1 404 Not Found\r\n\r\n")
        connectionSocket.send(b'''<html><head></head><body><h1>\\
                         404 Not Found</h1></body></html>\r\n''')
        # Zatvaramo socket u slučaju greške 404
        connectionSocket.close()

serverSocket.close()
sys.exit()
```

Da bi se demonstrirao rad servera, potrebno je da se HTML datoteka (npr. `mreze.html`) i serverski program nalaze u istom direktorijumu. Pokrene se serverski program i očita IP adresa njegovog hosta (npr. `78.30.142.18`). Na drugom hostu zatim se otvori veb-brauzer i u njega unese URL datoteke u sledećem formatu:

`http://78.30.142.18:6789/mreze.html`

Ako je sve urađeno ispravno, u brauzeru bi trebalo da se prikaže sadržaj datoteke.

Zainteresovanim čitateljkama i čitaocima se prepušta da modifikuju ovaj program tako da se omogući paralelno opsluživanje više HTTP zahteva (tzv. *multithreading*).

Zadatak 9.19 Korišćenjem programskog jezika Python3, realizujte proksi veb server koji ima funkcionalnost „keširanja" internet-stranica. Dovoljno je da server razume samo GET zahteve, ali treba da opslužuje sve vrste objekata (HTML stranice, slike itd).

Za svaki HTTP GET zahtev, proksi server proverava je li traženi sadržaj „keširan". Ako jeste, prosledić́e ga klijentu (brauzeru), a ako nije, zatražiće ga od udaljenog hosta, proslediti klijentu i napraviti keširanu verziju.

Listing programa kojim se implementira ovakav proksi veb server dat je u nastavku.

```
# ProxyServer.py
from socket import *
import sys

if len(sys.argv) <= 1:
     print('Poziv programa: "python ProxyServer.py server_ip"\n\r \
     [server_ip je IP adresa proksi servera]')
     sys.exit(2)

# Pravimo TCP socket, port = 8888
tcpSerSock = socket(AF_INET, SOCK_STREAM)
tcpSerSock.bind((sys.argv[1], 8888))
tcpSerSock.listen(100)

while 1:
     # Primamo podatke od klijenta
     print('Proksi server je spreman...')
     tcpCliSock, addr = tcpSerSock.accept()
     print('Uspostavljena veza s:', addr)
     message = tcpCliSock.recv(1024)
     print(message.decode())
     # Iz primljene poruke izdvajamo naziv datoteke
     print(message.decode().split()[1])
     filename = message.decode().split()[1].partition("/")[2]
     print(filename)
     fileExist = "false"
     filetouse = "/" + filename
     print(filetouse)
     try:
          # Proveravamo je li datoteka keširana
          f = open(filetouse[1:], "r")
          outputdata = f.readlines()
```

```
        fileExist = "true"
        # Ako postoji keširana verzija, pravimo HTTP odgovor
        tcpCliSock.send(b"HTTP/1.0 200 OK\r\n")
        tcpCliSock.send(b"Content-Type:text/html\r\n")
        for i in range(0, len(outputdata)):
            tcpCliSock.send(outputdata[i].encode())
            print('Čita se iz keša')
    # Ako datoteka nije keširana
    except IOError:
        if fileExist == "false":
            # Na proksi serveru se otvara socket
            c = socket(AF_INET, SOCK_STREAM)
            hostn = filename.replace("www.","",1)
            print(hostn)
            try:
                # Socket povezujemo na port 80
                c.connect((hostn, 80))
                # Pravimo privremenu datoteku,
                # a preko porta 80 preuzimamo traženi sadržaj
                fileobj = c.makefile('r', 0)
                fileobj.write("GET "+"http://" + filename + \
                " HTTP/1.0\n\n")
                # Odgovor upisujemo u bafer
                buff = fileobj.readlines()
                # Pravimo novu datoteku u kešu, koja odgovara
                # preuzetom objektu;
                # sadržaj bafera prosleđujemo klijentu i
                # upisujemo u keš datoteku
                tmpFile = open("./" + filename,"wb")
                for line in buff:
                    tmpFile.write(line);
                    tcpCliSock.send(line.encode());
            except:
                print('Nedozvoljen zahtev')
        else:
            # HTTP odgovor u slučaju kada datoteka nije pronađena
            tcpCliSock.send(b"HTTP/1.0 404 sendErrorErrorError\r\n")
            tcpCliSock.send(b"Content-Type:text/html\r\n")
            tcpCliSock.send(b"\r\n")
    # Zatvaramo sockete
    tcpCliSock.close()
tcpSerSock.close()
```

Proksi server se aktivira tako što se iz programskog koda uklone dijakritički znaci i potom pokrene program `ProxyServer.py`. Ako se proksi server i veb-brauzer izvršavaju na istom računaru, serveru se kao argument predaje `localhost`, dok je u suprotnom potrebno navesti IP adresu hosta na kome se izvršava serverska aplikacija.

Željenoj veb-stranici, npr. `www.google.com` iz brauzera pristupamo tako što kucamo `http://localhost:8888/www.google.com`

Ukoliko se serverska aplikacija i brauzer izvršavaju na različitim hostovima, tada umesto `localhost` treba navesti IP adresu serverskog hosta.

Ako se u brauzeru podese IP adresa i port proksi servera, željenoj stranici se pristupa na uobičajen način, navođenjem njene internet adrese – `http://www.google.com`.

Zainteresovanim čitaocima se prepušta da unaprede ovaj program tako da se omogući opsluživanje i ostalih HTTP poruka, kao i naprednije keširanje objekata, prema dokumentu RFC 2068.

Zadatak 9.20 Realizovati aplikaciju za „striming” videa, predstavljenog formatom MJPEG, posredstvom RTSP (*Real-Time Streaming Protocol*) i RTP.

MJPEG (*Motion JPEG*) je tzv. *intraframe* i *proprietary* format kompresije videa, po kome se svaka slika u video-sekvenci (*frame*) komprimuje postupkom JPEG. U video-sekvenci kojom smo raspolagali, svakom frejmu, trajanja 50 ms, dodaje se zaglavlje dužine 5 B, koje označava njegovu veličinu u bajtima.

Za rešavanje zadatka, koristićemo programski jezik Python, jer omogućava jednostavan i direktan pristup socketima, kao i jednostavno manipulisanje binarnim podacima, što rezultuje preglednim i konciznim programskim kodovima. U literaturi se navodi da mrežne aplikacije koje su programirane u Pythonu imaju i do *trideset puta* manje linija koda od onih koje su pisane u Javi.

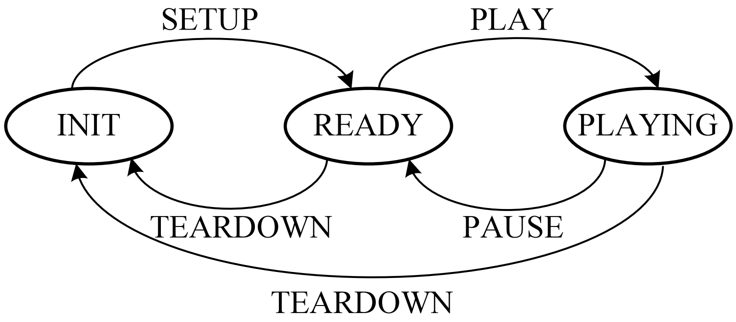
Aplikaciju ćemo realizovati na principu arhitekture klijent-server. Klijent i server će razmenjivati kontrolne poruke posredstvom RTSP i TCP, dok će server klijentu slati korisne podatke posredstvom RTP i UDP.

RTSP je protokol aplikacionog sloja koji služi za kontrolu isporučivanja podataka u realnom vremenu. Poruke koje generiše RTSP su tekstualne, sa skupom simbola prema standardu ISO 10646, koji su kodirani kao UTF-8; ovakav format poruka je izabran stoga što omogućava preglednije praćenje njihove razmene i jednostavniju implementaciju mehanizama protokola u skriptnim jezicima, u koje spada i Python. Poruke protokola RTSP mogu se prenositi bilo kojim transportnim protokolom koji podržava prenos na nivou celih bajtova.

Na strani servera, RTSP nema stanja. RTSP klijent, s druge strane, ima tri stanja, čiji su nazivi INIT, READY i PLAYING. Dijagram stanja klijenta prikazan je na slici.

Uobičajena RTSP interakcija između klijenta i servera odvija se na sledeći način:

1. klijent šalje poruku `SETUP`, kojom se uspostavlja sesija i postavljaju parametri transporta,



Slika 9.20: *Stanja RTSP klijenta.*

2. klijent šalje poruku `PLAY`, kojom se inicira reprodukovanje sadržaja u plejeru,
3. ukoliko želi da privremeno zaustavi reprodukciju, klijent može poslati poruku `PAUSE`,
4. klijent šalje poruku `TEARDOWN`, kojom se okončava sesija.

Server odgovara na svaku poruku koju primi od klijenta. Odgovor 200 znači da je zahtev uspešno izvršen, dok odgovori 404 i 500 odgovaraju greškama `file_not_found` i `connection error`, respektivno. Prelazak u novo stanje se realizuje svaki put kada klijent dobije pozitivan odgovor (200) na svoj zahtev.

Klijentska aplikacija će imati grafički korisnički interfejs (GUI) preko koga će se izdavati komande SETUP, PLAY, PAUSE i TEARDOWN i preko koga će se reprodukovati video-snimak. U operativnom smislu, klijentska aplikacija pri pokretanju otvara TCP socket ka serveru, preko koga potom šalje sve RTSP zahteve:

- Po izdavanju komande SETUP, serveru se šalje zahtev `SETUP`. Treba koristiti tzv. transportno polje RTSP paketa i u njemu specificirati port RTP socketa koji će služiti za prenos podataka. Klijent će čekati odgovor servera, iz čijeg će polja sesije očitati identifikaciju RTSP sesije. Na kraju, klijent će napraviti UDP socket za prijem podataka posredstvom RTP i njegov *timeout* parametar će podesiti na npr. 0,5 s;
- Po izdavanju komande PLAY, serveru se šalje zahtev `PLAY`. Treba koristiti polje sesije i u njemu identifikaciju sesije koju je server vratio u odgovoru na zahtev `SETUP`. Klijent će potom sačekati na serverov odgovor;
- Po izdavanju komande PAUSE, serveru se šalje zahtev `PAUSE`. Koristi se polje sesije i identifikacija sesije koju je server vratio u odgovoru na zahtev `SETUP`. Klijent će potom sačekati na serverov odgovor;
- Po izdavanju komande TEARDOWN, serveru se šalje zahtev `TEARDOWN`. Koristi se polje sesije i identifikacija sesije koju je server vratio u odgovoru na zahtev `SETUP`. Klijent će potom sačekati na serverov odgovor.

U svaki zahtev treba staviti polje `CSeq`, čija se vrednost inkrementira, počevši od 1, za svaki poslati zahtev.

Primer interakcije klijenta (C) i servera (S) dat je u nastavku.

C: SETUP movie.Mjpeg RTSP/1.0
C: CSeq: 1
C: Transport: RTP/UDP; client_port= 25000

S: RTSP/1.0 200 OK
S: CSeq: 1
S: Session: 123456

C: PLAY movie.Mjpeg RTSP/1.0
C: CSeq: 2
C: Session: 123456

S: RTSP/1.0 200 OK
S: CSeq: 2
S: Session: 123456

C: PAUSE movie.Mjpeg RTSP/1.0
C: CSeq: 3
C: Session: 123456

S: RTSP/1.0 200 OK
S: CSeq: 3
S: Session: 123456

C: PLAY movie.Mjpeg RTSP/1.0
C: CSeq: 4
C: Session: 123456

S: RTSP/1.0 200 OK
S: CSeq: 4
S: Session: 123456

C: TEARDOWN movie.Mjpeg RTSP/1.0
C: CSeq: 5
C: Session: 123456

S: RTSP/1.0 200 OK
S: CSeq: 5
S: Session: 123456

Po prijemu zahteva `PLAY`, server otpočinje paketizaciju videa u RTP pakete. Server čita jedan frejm iz MJPEG datoteke, odbacuje 5 B zaglavlja u koje je upisana veličina toga frejma i enkapsulira frejm u RTP paket, koji potom šalje posredstvom UDP; ova procedura se ponavlja na svakih 50 ms.

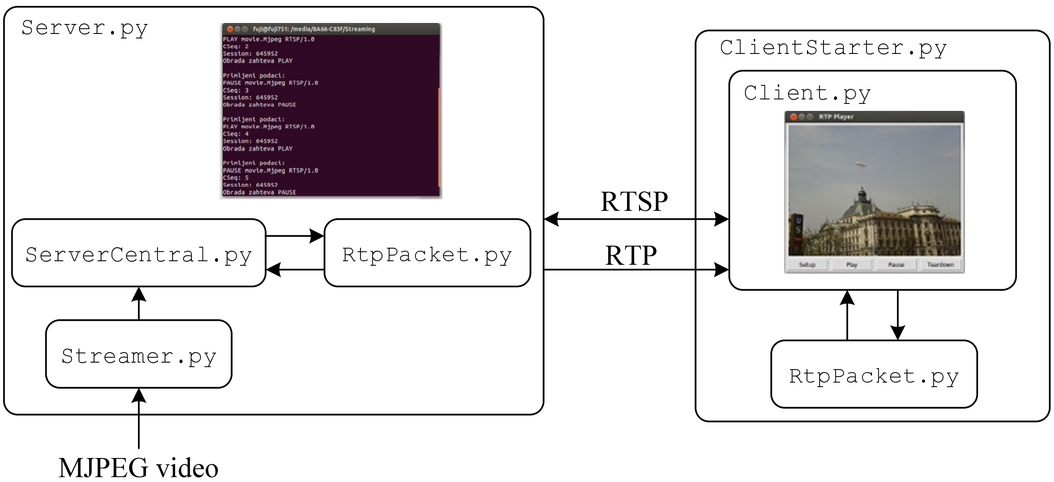
Struktura zaglavlja RTP paketa prikazana je na slici.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|X|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|            contributing source (CSRC) identifiers             |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Zaglavlje RTP paketa.

U polje verzije protokola (`V`), upisivaćemo vrednost 2. Vrednosti polja `P` (*padding*), `X` (*extension*), `CC` (*contributing sources count*) i `M` (*marker*) biće nule. U polje `PT` (*payload type*) treba upisivati 26, što odgovara formatu MJPEG. Broj sekvence odgovaraće rednom broju tekućeg MJPEG frejma, a vremensku oznaku ćemo očitati iz Pythonovog modula `time`. U polje identifikacije izvora (`SSRC`) možemo upisati bilo koju celobrojnu vrednost. Pošto je u našoj aplikaciji samo jedan izvor sadržaja (`CC` $= 0$), polje `CSRC` ne postoji, pa je ukupna dužina zaglavlja RTP paketa 12 B. U polje korisnog sadržaja treba upisati tekući frejm (JPEG sliku).

Arhitektura aplikacije prikazana je na slici.



Arhitektura aplikacije.

Aplikacija se sastoji od sledećih modula:

- `ClientStarter.py`, koji pokreće modul `Client.py` i otvara grafički korisnički interfejs;

- `Client.py`, koji šalje naredbe serveru, a od njega prima odgovore i frejmove za prikazivanje;
- `Server.py`, koji je zadužen za RTSP interakciju s klijentom;
- `ServerCentral.py`, koji obrađuje RTSP interakciju i priprema enkapsuliranje video-frejmova u RTP pakete;
- `RtpPacket.py`, koji na strani servera formira RTP pakete, a na strani klijenta ih raspakuje;
- `Streamer.py`, koji čita podatke iz MJPEG datoteke.

Njihovi listinzi dati su u nastavku.

ClientStarter.py

```
import sys
from Tkinter import Tk
from Client import Client

if __name__ == "__main__":
     try:
         serverAddr = sys.argv[1]
         serverPort = sys.argv[2]
         rtpPort = sys.argv[3]
         fileName = sys.argv[4]
     except:
         print("[Sintaksa naredbe: ClientStarter.py Server_name \
         Server_port RTP_port Video_file]\n")

     root = Tk()
   # Pravljenje novog klijentskog objekta
     app = Client(root, serverAddr, serverPort, rtpPort, fileName)
     app.master.title("RTP Player")
     root.mainloop()
```

Client.py

```
from Tkinter import *
import tkMessageBox
from PIL import Image, ImageTk
import socket, threading, sys, traceback, os

from RtpPacket import RtpPacket

CACHE_FILE_NAME = "cache-"
CACHE_FILE_EXT = ".jpg"
```

```
class Client:
     INIT = 0
     READY = 1
     PLAYING = 2
     state = INIT

     SETUP = 0
     PLAY = 1
     PAUSE = 2
     TEARDOWN = 3

     # Inicijalizacija
     def __init__(self, master, serveraddr, serverport, rtpport, filename):
          self.master = master
          self.master.protocol("WM_DELETE_WINDOW", self.handler)
          self.createWidgets()
          self.serverAddr = serveraddr
          self.serverPort = int(serverport)
          self.rtpPort = int(rtpport)
          self.fileName = filename
          self.rtspSeq = 0
          self.sessionId = 0
          self.requestSent = -1
          self.teardownAcked = 0
          self.connectToServer()
          self.frameNbr = 0

     def createWidgets(self):
          """Pravljenje GUI-ja"""
          # Taster Setup
          self.setup = Button(self.master, width=10, padx=3, pady=3)
          self.setup["text"] = "Setup"
          self.setup["command"] = self.setupMovie
          self.setup.grid(row=1, column=0, padx=2, pady=2)

          # Taster Play
          self.start = Button(self.master, width=10, padx=3, pady=3)
          self.start["text"] = "Play"
          self.start["command"] = self.playMovie
          self.start.grid(row=1, column=1, padx=2, pady=2)

          # Taster Pause
          self.pause = Button(self.master, width=10, padx=3, pady=3)
          self.pause["text"] = "Pause"
          self.pause["command"] = self.pauseMovie
          self.pause.grid(row=1, column=2, padx=2, pady=2)

          # Taster Teardown
          self.teardown = Button(self.master, width=10, padx=3, pady=3)
          self.teardown["text"] = "Teardown"
```

```
        self.teardown["command"] =  self.exitClient
        self.teardown.grid(row=1, column=3, padx=2, pady=2)

        # Labela za prikaz filma
        self.label = Label(self.master, height=19)
        self.label.grid(row=0, column=0, columnspan=4, sticky=W+E+N+S, \
        padx=5, pady=5)

    def setupMovie(self):
        """Opsluzivanje tastera Setup"""
        if self.state == self.INIT:
            self.sendRtspRequest(self.SETUP)

    def exitClient(self):
        """Opsluzivanje tastera Teardown"""
        self.sendRtspRequest(self.TEARDOWN)
        # Zatvaranje prozora GUI-ja
        self.master.destroy()
        # Brisanje kesa
        os.remove(CACHE_FILE_NAME + str(self.sessionId) + CACHE_FILE_EXT)

    def pauseMovie(self):
        """Opsluzivanje tastera Pause"""
        if self.state == self.PLAYING:
            self.sendRtspRequest(self.PAUSE)

    def playMovie(self):
        """Opsluzivanje tastera Play"""
        if self.state == self.READY:
            # Pravljenje nove niti za osluskivanje RTP paketa
            threading.Thread(target=self.listenRtp).start()
            self.playEvent = threading.Event()
            self.playEvent.clear()
            self.sendRtspRequest(self.PLAY)

def listenRtp(self):
        """Osluskivanje RTP paketa"""
        while True:
            try:
                data = self.rtpSocket.recv(20480)
                if data:
                    rtpPacket = RtpPacket()
                    rtpPacket.decode(data)

                    currFrameNbr = rtpPacket.seqNum()
                    print("Tekuci broj sekvence: " + str(currFrameNbr))

                    if currFrameNbr > self.frameNbr:
                        # Zakasneli paket se odbacuje
                        self.frameNbr = currFrameNbr
```

```
                        self.updateMovie(self.writeFrame(\
                        rtpPacket.getPayload()))
            except:
                # Osluskivanje se prekida posle pritiska tastera
                # PAUSE ili TEARDOWN
                if self.playEvent.isSet():
                    break

                # RTP socket se zatvara po prijemu ACK za TEARDOWN
                if self.teardownAcked == 1:
                    self.rtpSocket.shutdown(socket.SHUT_RDWR)
                    self.rtpSocket.close()
                    break

    def writeFrame(self, data):
        """Zapisivanje primljenog frejma u privremenu datoteku"""
        cachename = CACHE_FILE_NAME + str(self.sessionId) + CACHE_FILE_EXT
        file = open(cachename, "wb")
        file.write(data)
        file.close()

        return cachename

    def updateMovie(self, imageFile):
        """Prikaz privremene datoteke kao video-frejma u GUI-ju"""
        photo = ImageTk.PhotoImage(Image.open(imageFile))
        self.label.configure(image = photo, height=288)
        self.label.image = photo

    def connectToServer(self):
        """Povezivanje sa serverom i otpocinjanje RTSP/TCP sesije"""
        self.rtspSocket = socket.socket(socket.AF_INET, \
        socket.SOCK_STREAM)
        try:
            self.rtspSocket.connect((self.serverAddr, self.serverPort))
        except:
            tkMessageBox.showwarning('Neuspelo povezivanje! ', \
            'Veza sa \'%s\ ' nije uspostavljena.' %self.serverAddr)

    def sendRtspRequest(self, requestCode):
        """Slanje RTSP zahteva serveru"""

        # Setup
        if requestCode == self.SETUP and self.state == self.INIT:
            threading.Thread(target=self.recvRtspReply).start()
            # Azuriranje broja sekvence RTSP
            self.rtspSeq += 1

            # Formiranje RTSP zahteva
            request = 'SETUP ' + self.fileName + ' RTSP/1.0\nCSeq: ' + \
```

```
            str(self.rtspSeq) + '\nTransport: RTP/UDP; client_port= ' + \
            str(self.rtpPort)

            # Podsetnik na poslati zahtev
            self.requestSent = self.SETUP

        # Play
        elif requestCode == self.PLAY and self.state == self.READY:
            self.rtspSeq += 1
            request = 'PLAY ' + self.fileName + ' RTSP/1.0\nCSeq: ' + \
            str(self.rtspSeq) + '\nSession: ' + str(self.sessionId)
            self.requestSent = self.PLAY

        # Pause
        elif requestCode == self.PAUSE and self.state == self.PLAYING:
            self.rtspSeq += 1
            request = 'PAUSE ' + self.fileName + ' RTSP/1.0\nCSeq: ' + \
            str(self.rtspSeq) + '\nSession: ' + str(self.sessionId)
            self.requestSent = self.PAUSE

        # Teardown
        elif requestCode == self.TEARDOWN and not self.state == self.INIT:
            self.rtspSeq += 1
            request = 'TEARDOWN ' + self.fileName + ' RTSP/1.0\nCSeq: ' \
            + str(self.rtspSeq) + '\nSession: ' + str(self.sessionId)
            self.requestSent = self.TEARDOWN
        else:
            return

        # Slanje RTSP zahteva metodom rtspSocket
        self.rtspSocket.send(request.encode())

        print('\nPoslati podatak:\n' + request)

    def recvRtspReply(self):
        """Prijem RTSP odgovora od servera"""
        while True:
            reply = self.rtspSocket.recv(1024)

            if reply:
                self.parseRtspReply(reply.decode("utf-8"))

            # RTSP socket se zatvara posle zahteva Teardown
            if self.requestSent == self.TEARDOWN:
                self.rtspSocket.shutdown(socket.SHUT_RDWR)
                self.rtspSocket.close()
                break

    def parseRtspReply(self, data):
        """Parsiranje RTSP odgovora servera"""
```

```
        lines = str(data).split('\n')
        seqNum = int(lines[1].split(' ')[1])

        # Obradjuje se samo ako je broj sekvence serverovog odgovora
        # jednak broju sekvence zahteva
        if seqNum == self.rtspSeq:
            session = int(lines[2].split(' ')[1])
            # Nova identifikacija RTSP sesije
            if self.sessionId == 0:
                self.sessionId = session

            # Obradjuje se samo ako su identifikacije sesije iste
            if self.sessionId == session:
                if int(lines[0].split(' ')[1]) == 200:
                    if self.requestSent == self.SETUP:
                        # Azuriranje stanja RTSP
                        self.state = self.READY
                        # Otvaranje RTP porta
                        self.openRtpPort()
                    elif self.requestSent == self.PLAY:
                        self.state = self.PLAYING
                    elif self.requestSent == self.PAUSE:
                        self.state = self.READY
                        # Napusta se play nit,
                        # a pravi nova po obnavljanju reprodukcije
                        self.playEvent.set()
                    elif self.requestSent == self.TEARDOWN:
                        self.state = self.INIT
                        # Setuje se teardownAcked
                        # da bi se zatvorio socket
                        self.teardownAcked = 1

    def openRtpPort(self):
        """Otvaranje RTP socketa"""
        # Pravi se novi socket za prijem RTP paketa
        self.rtpSocket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        self.rtpSocket.settimeout(0.5)

        try:
            # Socket se veze za specificirani RTP port
            self.rtpSocket.bind(("", self.rtpPort))
        except:
            tkMessageBox.showwarning('Neuspelo povezivanje! ', \
            'Nije moguce otvoriti port %d' %self.rtpPort)

    def handler(self):
        """Pitalica za eksplicitno zatvaranja prozora GUI-ja"""
        self.pauseMovie()
        if tkMessageBox.askokcancel("Zatvoriti?", \
        "Jeste li sigurni da zelite zatvoriti aplikaciju?"):
```

```
            self.exitClient()
        else:
            # Ako korisnik izabere "cancel", reprodukcija se nastavlja
            self.playMovie()
```

Server.py

```
import sys, socket

from ServerCentral import ServerCentral

class Server:

    def main(self):
        try:
            SERVER_PORT = int(sys.argv[1])
        except:
            print("[Sintaksa: Server.py Server_port]\n")
        rtspSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        rtspSocket.bind(('', SERVER_PORT))
        rtspSocket.listen(5)

        # Podaci o klijentu (addresa, port) primaju se
        # posredstvom RTSP/TCP sesije
        while True:
            clientInfo = {}
            clientInfo['rtspSocket'] = rtspSocket.accept()
            ServerCentral(clientInfo).run()

if __name__ == "__main__":
    (Server()).main()
```

ServerCentral.py

```
from random import randint
import sys, traceback, threading, socket

from Streamer import Streamer
from RtpPacket import RtpPacket

class ServerCentral:
    SETUP = 'SETUP'
    PLAY = 'PLAY'
    PAUSE = 'PAUSE'
    TEARDOWN = 'TEARDOWN'
```

```
INIT = 0
READY = 1
PLAYING = 2
state = INIT

OK_200 = 0
FILE_NOT_FOUND_404 = 1
CON_ERR_500 = 2

clientInfo = {}

def __init__(self, clientInfo):
     self.clientInfo = clientInfo

def run(self):
     threading.Thread(target=self.recvRtspRequest).start()

def recvRtspRequest(self):
     """Prijem RTSP zahteva od klijenta"""
     connSocket = self.clientInfo['rtspSocket'][0]
     while True:
          data = connSocket.recv(256)
          if data:
               print("Primljeni podaci:\n" + data.decode("utf-8"))
               self.processRtspRequest(data.decode("utf-8"))

def processRtspRequest(self, data):
     """Obrada primljenog RTSP zahteva"""
     # Tip zahteva
     request = data.split('\n')
     line1 = request[0].split(' ')
     requestType = line1[0]

     # Naziv video-datoteke
     filename = line1[1]

     # Broj RTSP sekvence
     seq = request[1].split(' ')

     # Obrada zateva SETUP
     if requestType == self.SETUP:
          if self.state == self.INIT:
               # Azuriranje stanja
               print("Obrada zahteva SETUP\n")

               try:
                    self.clientInfo['Streamer'] = \
                    Streamer(filename)
                    self.state = self.READY
```

```
            except IOError:
                self.replyRtsp(self.FILE_NOT_FOUND_404, seq[1])

            # Pravljenje slucajnog identifikacionog broja
            # RTSP sesije
            self.clientInfo['session'] = randint(100000, 999999)

            # Slanje RTSP odgovora
            self.replyRtsp(self.OK_200, seq[1])

            # Citanje RTP/UDP porta iz poslednje linije
            self.clientInfo['rtpPort'] = request[2].split(' ')[3]

    # Obrada zateva PLAY
    elif requestType == self.PLAY:
        if self.state == self.READY:
            print("Obrada zahteva PLAY\n")
            self.state = self.PLAYING

            # Pravljenje novog socketa za RTP/UDP
            self.clientInfo["rtpSocket"] = \
            socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

            self.replyRtsp(self.OK_200, seq[1])

            # Pravljenje nove niti
            # i slanje RTP paketa
            self.clientInfo['event'] = threading.Event()
            self.clientInfo['worker']= \
            threading.Thread(target=self.sendRtp)
            self.clientInfo['worker'].start()

    # Obrada zateva PAUSE
    elif requestType == self.PAUSE:
        if self.state == self.PLAYING:
            print("Obrada zateva PAUSE\n")
            self.state = self.READY

            self.clientInfo['event'].set()

            self.replyRtsp(self.OK_200, seq[1])

    # Obrada zateva TEARDOWN
    elif requestType == self.TEARDOWN:
        print("Obrada zateva TEARDOWN\n")

        self.clientInfo['event'].set()

        self.replyRtsp(self.OK_200, seq[1])
```

```
            # Zatvaranje RTP socketa
            self.clientInfo['rtpSocket'].close()

    def sendRtp(self):
        """Slanje RTP paketa preko UDP"""
        while True:
            self.clientInfo['event'].wait(0.05)

            # Slanje se obustavlja za PAUSE i TEARDOWN
            if self.clientInfo['event'].isSet():
                break

            data = self.clientInfo['Streamer'].nextFrame()
            if data:
                frameNumber = self.clientInfo['Streamer'].frameNbr()
                try:
                    address = self.clientInfo['rtspSocket'][1][0]
                    port = int(self.clientInfo['rtpPort'])
                    self.clientInfo['rtpSocket'].sendto(self.makeRtp(\
                    data, frameNumber),(address,port))
                except:
                    print("Connection Error - greska povezivanja")

    def makeRtp(self, payload, frameNbr):
        """Pakovanje videa u RTP pakete"""
        version = 2
        padding = 0
        extension = 0
        cc = 0
        marker = 0
        pt = 26
        seqnum = frameNbr
        ssrc = 0

        rtpPacket = RtpPacket()
        rtpPacket.encode(version, padding, extension, cc, seqnum, \
        marker, pt, ssrc, payload)
        return rtpPacket.getPacket()

    def replyRtsp(self, code, seq):
        """Slanje RTSP odgovora klijentu"""
        if code == self.OK_200:
            reply = 'RTSP/1.0 200 OK\nCSeq: ' + seq + '\nSession: ' + \
            str(self.clientInfo['session'])
            connSocket = self.clientInfo['rtspSocket'][0]
            connSocket.send(reply.encode())
        # Poruke o greskama
        elif code == self.FILE_NOT_FOUND_404:
            print("404 NOT FOUND")
        elif code == self.CON_ERR_500:
```

```
            print("500 CONNECTION ERROR")
```

RtpPacket.py

```
import sys
from time import time
HEADER_SIZE = 12

class RtpPacket:
    header = bytearray(HEADER_SIZE)

    def __init__(self):
        pass

    def encode(self, version, padding, extension, cc, seqnum, marker, \
    pt, ssrc, payload):
        """Formiranje RTP paketa sa zadatim zaglavljem \
        i korisnim sadrzajem"""
        timestamp = int(time())
        header = bytearray(HEADER_SIZE)

        # Popunjavanje polja u zaglavlju
        header[0] = (version << 6) | (padding << 5) | \
        (extension << 4) | cc
        header[1] = (marker << 7) | pt
        header[2] = (seqnum >> 8) & 255 # gornji biti
        header[3] = seqnum & 255
        header[4] = timestamp >> 24 & 255
        header[5] = timestamp >> 16 & 255
        header[6] = timestamp >> 8 & 255
        header[7] = timestamp & 255
        header[8] = ssrc >> 24 & 255
        header[9] = ssrc >> 16 & 255
        header[10] = ssrc >> 8 & 255
        header[11] = ssrc & 255

        self.header = header

        # Korisni sadrzaj se preuzima iz argumenta
        self.payload = payload

    def decode(self, byteStream):
        """Dekodiranje RTP paketa"""
        self.header = bytearray(byteStream[:HEADER_SIZE])
        self.payload = byteStream[HEADER_SIZE:]

    def version(self):
        """Izdvajanje verzije RTP"""
```

```
        return int(self.header[0] >> 6)

    def seqNum(self):
        """Izdvajanje broja sekvence (frejma)"""
        seqNum = self.header[2] << 8 | self.header[3]
        return int(seqNum)

    def timestamp(self):
        """Izdvajanje vremenske oznake"""
        timestamp = self.header[4] << 24 | self.header[5] << 16 | \
        self.header[6] << 8 | self.header[7]
        return int(timestamp)

    def payloadType(self):
        """Izdvajanje vrste korisnog sadrzaja"""
        pt = self.header[1] & 127
        return int(pt)

    def getPayload(self):
        """Izdvajanje korisnog sadrzaja"""
        return self.payload

    def getPacket(self):
        """Vracanje RTP paketa"""
        return self.header + self.payload
```

Streamer.py

```
class Streamer:
    def __init__(self, filename):
        self.filename = filename
        try:
            self.file = open(filename, 'rb')
        except:
            raise IOError
        self.frameNum = 0

    def nextFrame(self):
        """Uzimanje narednog frejma"""
        # Duzina frejma se cita iz prvih 5 bajtova
        data = self.file.read(5)
        if data:
            framelength = int(data)

            # Citanje tekuceg frejma
            data = self.file.read(framelength)
            self.frameNum += 1
        return data
```

```
    def frameNbr(self):
        """Broj frejma"""
        return self.frameNum
```

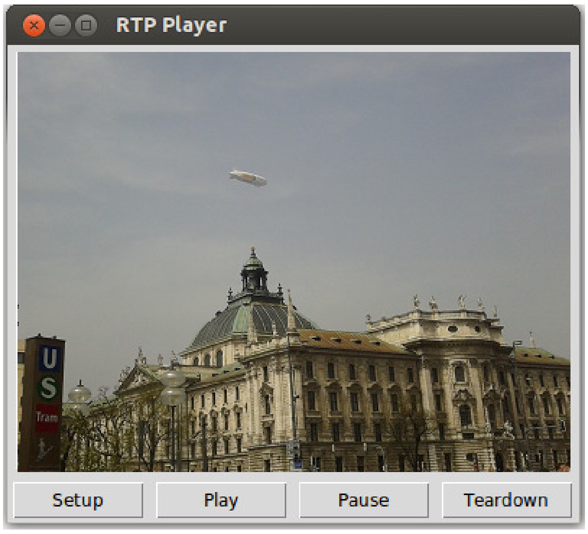
Aplikacija se testira tako što se na serverskom hostu pokrene server naredbom

```
python3 Server.py server_port
```

gde je `server_port` broj RTSP porta (njegova standardna vrednost je 554, ali se preporučuje da u ovome primeru izaberete vrednost veću od 1024). Potom je potrebno pokrenuti klijentsku aplikaciju naredbom

```
python3 ClientStarter.py server_host server_port RTP_port video_file
```

gde je `server_host` adresa hosta na kome se izvršava server, `server_port` broj RTSP porta, `RTP_port` broj porta preko koga će se primati RTP paketi i `video_file` naziv MJPEG datoteke.



Grafički korisnički interfejs klijentske aplikacije.

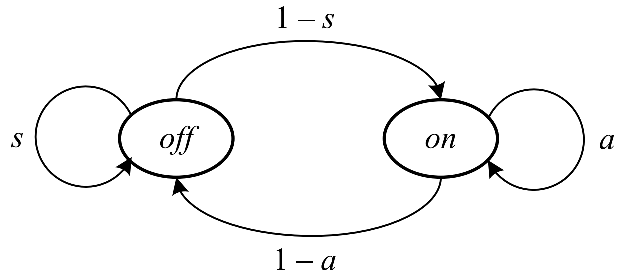
Ukoliko je povezivanje klijenta i servera bilo uspešno, otvoriće se grafički korisnički interfejs u kome će se reprodukovati zahtevana video-datoteka.

Zainteresovanim čitateljkama i čitaocima se prepušta da modifikuju aplikaciju tako da „plejer” ima tri uobičajena tastera (PLAY, PAUSE i STOP).

10. Modeliranje mrežnog saobraćaja

Zadatak 10.1 Vršni protok izvora digitalizovanog govora iznosi 64 kb/s. Izvor se modelira *on-off* modelom, pri čemu je prosečno trajanje perioda aktivnosti $T_a = 0{,}45$ s, a perioda mirovanja $T_s = 1{,}5$ s. Odredite parametre izvora i prosečni protok na njegovom izlazu.

On-off izvor saobraćaja ima dva stanja, aktivno (*on*), u kome emituje pakete i tiho, (*off*) u kome miruje. Ovaj model dobro opisuje signal govora i CBR saobraćaj u mrežama s ATM. Dijagram stanja ovakvog izvora prikazan je na slici.



Slika 10.1: *Dijagram stanja on-off izvora.*

Označimo s a verovatnoću zadržavanja izvora u aktivnom stanju i sa s verovatnoću zadržavanja u stanju mirovanja. Neka je λ protok koji izvor emituje kada se nalazi u aktivnom stanju.

Verovatnoća da će aktivno stanje trajati n jedinica vremena data je geometrijskom raspodelom,

$$A(n) = a^n(1-a), \quad n \geqslant 1.$$

Prosečno trajanje aktivnog stanja je

$$T_a = \frac{a}{1-a}.$$

Odavde dobijamo verovatnoću ostanka izvora u aktivnom stanju:

$$a = \frac{T_a}{1+T_a} = 0{,}3103.$$

Na sličan način, verovatnoća da će stanje mirovanja trajati n jedinica vremena je

$$S(n) = s^n(1-s), \quad n \geqslant 1,$$

a njegovo prosečno trajanje je

$$T_s = \frac{s}{1-s}.$$

Odavde dobijamo verovatnoću ostanka izvora u stanju mirovanja:

$$s = \frac{T_s}{1+T_s} = 0{,}6.$$

Prosečni protok koji generiše ovaj izvor je

$$\lambda_a = \lambda \frac{T_a}{T_a + T_s}$$

i u našem slučaju iznosi 14,77 kb/s.

Zadatak 10.2 Prosečni protok na ATM kanalu iznosi 500 kb/s. Odredite parametar Poissonovog modela i verovatnoću da će se tokom intervala od 1 ms generisati 10 ćelija.

Verovatnoća da će Poissonov izvor tokom vremena t generisati $k \geqslant 0$ jedinica podataka data je izrazom

$$p\,(k) = \frac{(\lambda_a t)^k}{k!}\, e^{-\lambda_a t},$$

gde je λ_a prosečni protok jedinica podataka.

Svaka ATM ćelija dužine je 53 B, pa je prosečni protok ćelija na kanalu iz postavke zadatka

$$\lambda_a = \frac{500 \cdot 10^3 \text{ b/s}}{53 \cdot 8 \text{ b}} = 1{,}1792 \cdot 10^3 \text{ s}^{-1}.$$

Verovatnoća da će se tokom intervala od 1 ms generisati 10 ćelija je

$$p\,(10) = \frac{(\lambda_a t)^{10}}{10!}\, e^{-\lambda_a t} = 4{,}407 \cdot 10^{-7}.$$

Zadatak 10.3 Poissonov izvor prosečno emituje $\lambda_a = 10^3$ paketa u sekundi, dok je vršni protok $\sigma = 3\lambda_a$. Odredite parametre eksponencijalne raspodele koja opisuje vreme međudolazaka ovih paketa.

Vreme međudolazaka paketa koje generiše sporadični (engl. *bursty*) Poissonov izvor opisano je pomerenom eksponencijalnom raspodelom,

$$f_T(t) = \begin{cases} 0, & t < a \\ b\exp(-b(t-a)), & t \geqslant a \end{cases}.$$

S $a \geqslant 0$ označen je *parametar položaja* (izražava se u s), a s $b > 0$ *parametar oblika* (izražava se u s^{-1}).

Ovakav model dobro opisuje telnet i FTP saobraćaj.

Vrednosti parametara a i b odredićemo na sledeći način. Prosečni vremenski razmak susednih paketa je

$$\begin{aligned} T_a &= \int\limits_a^\infty tb\exp(-b(t-a))\,\mathrm{d}t = \\ &= a + \frac{1}{b}. \end{aligned}$$

Minimalno vreme između susednih paketa se dobija za $b \gg 1$ i iznosi $(T_a)_{min} = a$.

Maksimalni broj paketa koji se mogu generisati tokom vremena t je

$$N_m = \sigma t,$$

gde je σ vršni protok (u paketima u sekundi). Minimalno vreme između generisanja susednih paketa odavde je

$$a = \frac{t}{N_m} = \frac{1}{\sigma}.$$

Parametar a u primeru iz postavke zadatka iznosi $3{,}33 \cdot 10^{-4}$ s.

Prosečni broj paketa koje će izvor generisati tokom vremena t je

$$N_a = \lambda_a t,$$

pa je prosečno vremensko rastojanje između paketa

$$T_a = \frac{t}{N_a} = \frac{1}{\lambda_a}.$$

S druge strane, pokazali smo da iz osobina pomerene eksponencijalne raspodele važi

$$T_a = a + \frac{1}{b},$$

te je vrednost parametra oblika

$$b = \frac{\sigma\lambda_a}{\sigma - \lambda_a}.$$

U našem zadatku je $b = 1500\ \mathrm{s}^{-1}$, pa je funkcija gustine verovatnoće vremena međudolazaka paketa

$$f_T(t) = \begin{cases} 0, & t < 3{,}33 \cdot 10^{-4}\ \mathrm{s} \\ 1500\exp(-1500(t - 3{,}33 \cdot 10^{-4})), & t \geqslant 3{,}33 \cdot 10^{-4}\ \mathrm{s} \end{cases}.$$

Zadatak 10.4 Period posmatranja Poissonovog izvora je $T = 1$ ms. Parametri izvora su $\lambda_a = 50$ s^{-1} i $\sigma = 150$ s^{-1}. Odredite parametre raspodele i verovatnoću generisanja paketa tokom perioda posmatranja.

Prema rezultatu prethodnog zadatka, parametri raspodele su $a = 6{,}7$ ms i $b = 75$ s^{-1}.

Trajanje perioda posmatranja kraće je od minimalnog vremena između dvaju paketa, pa očekujemo da će se tokom intervala posmatranja generisati najviše jedan paket.

Posmatrajmo šta bi se desilo kada bismo izvor posmatrali tokom n intervala trajanja po T. Prosečni broj paketa koje bi izvor tada generisao bio bi

$$N_a = \lambda_a nT.$$

S druge strane, iz očekivanja binomne raspodele je

$$N_a = xn,$$

gde je x verovatnoća generisanja paketa tokom jednog intervala T. Stoga je

$$x = \lambda_a T,$$

što iznosi 0,05.

Zadatak 10.5 Period posmatranja Poissonovog izvora je $T = 1$ ms. Parametri izvora su $\lambda_a = 1000$ s^{-1} i $\sigma = 5000$ s^{-1}. Odredite parametre raspodele i verovatnoću generisanja paketa.

Parametri raspodele su

$$a = \frac{1}{\sigma} = 0{,}2 \text{ ms},$$

$$b = \frac{\lambda_a \sigma}{\sigma - \lambda_a} = 1250 \text{ s}^{-1}.$$

Trajanje perioda posmatranja sada je duže od minimalnog vremena između dvaju paketa, pa očekujemo da će izvor generisati više od jednog paketa.

Maksimalni broj paketa koje izvor generiše je

$$N_m = \lceil \sigma T \rceil = 5,$$

a prosečni

$$N_a = \lambda_a T = 1.$$

Neka je x verovatnoća da će se tokom perioda posmatranja generisati paket. Iz osobina binomne raspodele je

$$N_a = xN_m,$$

pa je verovatnoća generisanja paketa tokom perioda posmatranja

$$x = \frac{N_a}{N_m} = 0{,}2.$$

Primetimo da važi $x \leqslant \lambda_a/\sigma$.

Na kraju, verovatnoća da će se tokom *jednog* perioda posmatranja generisati k paketa je

$$p\,(k) = \binom{N_m}{k} x^k (1-x)^{N_m-k}.$$

Zadatak 10.6 Prosečni protok paketa fiksne dužine $L = 53$ B na telekomunikacionom kanalu je $\lambda_a = 2{,}3 \cdot 10^4$ s^{-1}, dok je vršni binarni protok jednak kapacitetu kanala i iznosi $\sigma = 155{,}52$ Mb/s. Odredite parametre ekvivalentne binomne raspodele i verovatnoću generisanja 10 paketa tokom intervala trajanja $t = 1$ ms.

Prosečni broj paketa koji se generišu tokom intervala posmatranja je

$$N_a = \lambda_a t = 23.$$

Najkraće trajanje paketa određeno je kapacitetom linije

$$\Delta = \frac{L \cdot 8\ \mathrm{b}}{\sigma} = 2{,}7263 \cdot 10^{-6}\ \mathrm{s}.$$

Maksimalni broj paketa koji se mogu generisati tokom intervala t stoga je

$$N_m = \left\lceil \frac{t}{\Delta} \right\rceil = 367.$$

Ako je x verovatnoća generisanja paketa tokom intervala posmatranja, tada će važiti

$$N_a = x N_m.$$

Odavde je $x = 0{,}0627$.

Verovatnoća da će se k paketa generisati tokom n intervala posmatranja data je binomnom raspodelom,

$$p\,(k) = \binom{n}{k} x^k (1-x)^{n-k},$$

pa je verovatnoća da će se od potencijalnih N_m paketa generisati njih 10

$$p\,(10) = \binom{N_m}{10} x^{10} (1-x)^{N_m-10} = 9{,}3192 \cdot 10^{-4}.$$

Zadatak 10.7 ATM izvor emituje ćelije po binomnoj raspodeli. Parametri izvora su $\lambda_a = 400\ \mathrm{s}^{-1}$ i $\sigma = 10^3\ \mathrm{s}^{-1}$. Odredite parametre pomerene geometrijske raspodele koja opisuje vreme međudolazaka ovih ćelija. Trajanje perioda posmatranja je $T = 0{,}1$ ms.

Broj ćelija (ili paketa) koje tokom intervala trajanja t generiše Bernoullijev izvor dat je binomnom raspodelom, dok je vreme njihovih međudolazaka dato pomerenom geometrijskom raspodelom:

$$\mathrm{P}(N = n) = \begin{cases} 0, & n < \alpha \\ x(1-x)^{n-\alpha}, & n \geqslant \alpha \end{cases},$$

gde je n broj perioda posmatranja između paketa, $\alpha \geqslant 0$ je *parametar položaja* (u periodima posmatranja) i x je verovatnoća generisanja paketa tokom jednog perioda posmatranja.

Prosečni broj perioda posmatranja između susednih paketa je

$$\begin{aligned} n_a &= \sum_{n=\alpha}^{\infty} nx(1-x)^{n-\alpha} = \\ &= \alpha + \frac{1-x}{x}. \end{aligned}$$

Vrednost α stoga možemo shvatiti i kao minimalni razmak susednih paketa.

Neka je trajanje intervala tokom koga posmatramo izvor t. Njemu će odgovarati

$$n = \frac{t}{T}$$

perioda posmatranja. Maksimalni broj paketa koji se tom prilikom mogu generisati je

$$N_m = \sigma t.$$

Minimalni broj perioda posmatranja između susednih paketa je

$$\alpha = \left\lfloor \frac{n}{N_m} \right\rfloor = \left\lfloor \frac{1}{\sigma T} \right\rfloor.$$

Odavde dobijamo $\alpha = 10$.

Prosečni broj generisanih paketa je

$$N_a = xn = \lambda_a t,$$

odakle se dobija $x = 0{,}04$.

Raspodela verovatnoće vremena međudolazaka ćelija posmatranog ATM izvora stoga je

$$\mathrm{P}(N = n) = \begin{cases} 0, & n < 10 \\ 0{,}04 \cdot 0{,}96^{n-10}, & n \geqslant 10 \end{cases}.$$

Zadatak 10.8 Sporadični izvor emituje pakete prosečne dužine 400 b. Prosečni protok iznosi 5 Mb/s, a vršni 20 Mb/s. Odredite parametre Pareto raspodele koja opisuje ovaj izvor.

Funkcija gustine verovatnoće trenutnog protoka sporadičnog Pareto izvora data je izrazom

$$f_\Lambda(\lambda) = \begin{cases} 0, & \lambda < \lambda_{min} \\ b\,\dfrac{a^b}{\lambda^{b+1}}, & \lambda \geqslant \lambda_{min} \end{cases},$$

gde je λ_{min} minimalni protok, a parametar položaja i b parametar oblika.

Funkcija gustine verovatnoće vremena međudolazaka paketa je

$$f_T(t) = \frac{ba^b}{t^{b+1}}, \quad a \leqslant t < \infty.$$

Vidimo da je a ujedno i minimalni vremenski razmak između susednih paketa.

Tokom vremena t, ovakav izvor će generisati najviše

$$N_m = \sigma t$$

paketa, gde je σ vršni protok. Minimalni vremenski razmak između dvaju paketa stoga će biti

$$a = \frac{t}{N_m} = \frac{1}{\sigma}.$$

Vodeći računa o jedinicama u kojima se izražavaju pojedine veličine, dobijamo da je u našem zadatku $a = 20 \cdot 10^{-6}$ s.

Prosečni broj generisanih paketa je

$$N_a = \lambda_a t,$$

gde je λ_a prosečni protok izvora. Prosečno vreme između dvaju paketa stoga je

$$T_a = \frac{t}{N_a} = \frac{1}{\lambda_a}.$$

S druge strane, iz osobina Pareto raspodele je

$$T_a = \int_a^\infty t\,\frac{ba^b}{t^{b+1}}\,\mathrm{d}t =$$

$$= \frac{ab}{b-1}.$$

Odavde je

$$b = \frac{\sigma}{\sigma - \lambda_a},$$

što iznosi 1,333.

Hurstov parametar Pareto izvora je

$$H = \frac{3-b}{2}$$

i za posmatrani izvor iznosi 0,833.

Zadatak 10.9 Period posmatranja sporadičnog izvora je $T = 0{,}1$ ms. Parametri izvora su $\lambda_a = 100\ \mathrm{s}^{-1}$ i $\sigma = 1500\ \mathrm{s}^{-1}$. Odredite parametre Pareto raspodele i verovatnoću generisanja paketa.

Prema rezultatu prethodnog zadatka, parametri Pareto raspodele su $a = 6{,}667 \cdot 10^{-4}$ s i $b = 1{,}071$.

Trajanje perioda posmatranja kraće je od minimalnog vremena između dvaju paketa, pa ćemo verovatnoću generisanja paketa odrediti slično kao u zadatku 10.4. Posmatrajmo izvor u dužem vremenskom intervalu trajanja t. Broj perioda posmatranja tokom vremena t je

$$n = \frac{t}{T}.$$

Prosečni broj paketa koje izvor generiše tokom vremena t je

$$N_a = \lambda_a t = \lambda_a n T.$$

S druge strane, iz osobina binomne raspodele je

$$N_a = xn,$$

gde je x verovatnoća generisanja paketa tokom jednog perioda posmatranja. Stoga je

$$x = \lambda_a T$$

i iznosi 0,1.

Zadatak 10.10 Period posmatranja sporadičnog izvora je $T = 2$ ms. Parametri izvora su $\lambda_a = 2000\ \mathrm{s}^{-1}$ i $\sigma = 5000\ \mathrm{s}^{-1}$. Odredite parametre Pareto raspodele i verovatnoću generisanja paketa.

Parametri Pareto raspodele su

$$a = \frac{1}{\sigma} = 0{,}2\ \mathrm{ms},$$

$$b = \frac{\sigma}{\sigma - \lambda_a} = 1{,}667.$$

U ovom slučaju je trajanje perioda posmatranja duže od minimalnog vremena između dvaju paketa, pa očekujemo da će izvor generisati više od jednog paketa.

Maksimalni broj paketa koje izvor generiše je

$$N_m = \lceil \sigma T \rceil = 10,$$

a prosečni

$$N_a = \lambda_a T = 4.$$

Neka je x verovatnoća da će se tokom perioda posmatranja generisati paket. Iz osobina binomne raspodele je

$$N_a = x N_m,$$

pa je verovatnoća generisanja paketa tokom perioda posmatranja

$$x = \frac{N_a}{N_m} = 0{,}4.$$

Primetimo da važi $x \leqslant \lambda_a / \sigma$.

Na kraju, verovatnoća da će se tokom *jednog* perioda posmatranja generisati k paketa je

$$p\,(k) = \binom{N_m}{k} x^k (1 - x)^{N_m - k}.$$

11. Opsluživanje mrežnog saobraćaja

Zadatak 11.1 Sesija internet-telefonije ostvaruje se kroz mrežu u kojoj se propagaciono kašnjenje kreće u opsegu od 50 ms do 200 ms. Sesija počinje u trenutku $t = 0$, paketi digitalizovanog govora su dužine $L = 160$ b i šalju se na svakih 20 ms. Oba učesnika pri reprodukciji uvode jednako kašnjenje $q = 150$ ms.

a) Odredite trenutke isporučivanja paketa čija su vremena pristizanja do odredišta navedena u tabeli.

b) Odredite potrebnu veličinu prijemnog bafera u koji se smeštaju paketi čiju isporuku treba odložiti.

Redni broj paketa	Trenutak dolaska [ms]
1	95
2	145
3	170
4	135
6	160
8	220
5	275
7	280
9	285
10	305

a) Prvi paket se šalje u trenutku $t = 20$ ms, pa na red za isporučivanje (reprodukciju) dolazi u $t = 20 \text{ ms} + q = 170$ ms. Svi naredni paketi će se isporučiti s razmakom od 20 ms, osim ako zbog prevelikog kašnjenja ne budu propustili svoj red. Trenuci isporučivanja prikazani su u tabeli na narednoj strani.

b) Minimalno propagaciono kašnjenje iznosi 50 ms, pa je maksimalno dozvoljeno kašnjenje paketa pri baferisanju 100 ms.

Izvor šalje pakete na svakih 20 ms, pa tokom dozvoljenog perioda od 100 ms do odredišta može doći najviše pet paketa. Da bi se omogućio prijem novopridošlog paketa

Redni broj paketa	Trenutak dolaska [ms]	Trenutak isporučivanja [ms]
1	95	170
2	145	190
3	170	210
4	135	230
6	160	270
8	220	310
5	275	250 — odbacuje se
7	280	290
9	285	330
10	305	350

dok se prvi u redu isporučuje, a još četiri čekaju da dođu na red, kapacitet bafera treba da iznosi 6 paketa, ili 960 b.

Zadatak 11.2 VoIP sesija između dvaju čvorova ostvaruje se posredstvom linka na kome je propagaciono kašnjenje 8 ms i čiji je kapacitet 4 Mb/s. Protok digitalizovanog govora je 128 kb/s, a dužina paketa 64 B. Čim izvorišni čvor formira paket, prosleđuje ga odredišnom. Po prijemu kompletnog paketa, odredišni čvor ga konvertuje u analogni signal. Odredite vreme koje protekne od formiranja bita u izvorišnom čvoru do njegovog dekodiranja u odredišnom.

Traženo vreme odgovara ukupnom kašnjenju pri prenosu, koje je jednako zbiru vremena potrebnog za prenos paketa, vremena potrebnog da bi se paket „utisnuo” u link i propagacionog kašnjenja.

Vreme potrebno za formiranje paketa iznosi $64 \cdot 8$ b/128 kb/s = 4 ms. Da bi se paket „utisnuo” u link, potrebno je još $64 \cdot 8$ b/4 Mb/s = 128 µs. Uz propagaciono kašnjenje od 8 ms, ukupno kašnjenje u prenosu je 12,128 ms.

Zadatak 11.3 Kapacitet prijemnog bafera u aplikaciji video striminga je B. Bafer se prazni s konstantnim protokom r, a sve dok u njemu ima mesta, server mu šalje podatke s konstantnim protokom x. Da bi otpočela reprodukcija sadržaja, potrebno je primiti količinu podataka Q.

a) Odredite trajanje perioda reprodukcije i „zamrzavanja” slike, ako je $x < r$.

b) U kom trenutku će se napuniti bafer ako je $x > r$?

a) Trajanje perioda zamrzavanja slike jednako je vremenu potrebnom da bi se prazan bafer napunio količinom podataka Q,

$$T_{freeze} = \frac{Q}{x}.$$

Tokom perioda reprodukcije, bafer se prazni s protokom r i puni s protokom x. Trajanje perioda reprodukcije stoga je

$$T_{playout} = \frac{Q}{r - x}.$$

b) Bafer se prvo puni s protokom x, do količine podataka Q; potom počinje reprodukcija i bafer se puni s protokom $x-r$. Bafer će se napuniti do količine podataka B u trenutku

$$t = \frac{Q}{x} + \frac{B - Q}{x - r}.$$

Zadatak 11.4 Baferu iz prethodnog zadatka server šalje podatke s promenljivim protokom

$$x(t) = \frac{H}{T}\,(t - kT), \quad kT \leqslant t < (k+1)T,\ k = 0, 1, \ldots,$$

pri čemu je $Q \leqslant HT/2$.

a) Koliki je prosečni protok podataka koje šalje server?

b) Kada će početi reprodukcija sadržaja?

c) Odredite najmanje Q tako da nema zamrzavanja slike, ako je $H > 2r$.

a) Prosečni protok podataka je

$$X = \frac{1}{T}\int_0^T \frac{H}{T}\,t\,\mathrm{d}t = \frac{H}{2}.$$

b) Bafer će primiti količinu podataka Q u trenutku t_1 za koji važi

$$Q = \int_0^{t_1} \frac{H}{T}\,t\,\mathrm{d}t = \frac{H}{2T}\,t_1^2.$$

Odavde je

$$t_1 = \sqrt{2\frac{QT}{H}}.$$

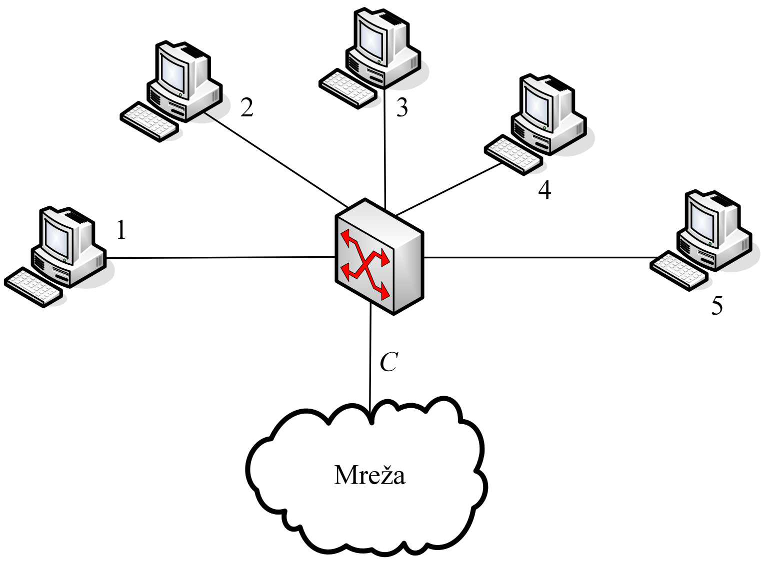
c) Količina podataka u baferu za $t_1 \leqslant t < T$ data je izrazom

$$q(t) = \frac{H}{2T}\,t^2 - r(t - t_1).$$

Ova funkcija dostiže minimalnu vrednost za $t = rT/H$ i ona iznosi $rt_1 - \frac{r^2T}{2H}$. Da bi se dobila nenegativna vrednost, potrebno je da bude $Q \geqslant r^2T/(8H)$, pa je

$$Q_{min} = \frac{r^2T}{8H}.$$

Zadatak 11.5 Pet stanica povezano je preko pristupnog uređaja na telekomunikacionu mrežu. Normalizovani kapacitet odlaznog linka od ovog uređaja ka ostatku mreže je $C = 1$. Stanica i emituje podatke s prosečnim protokom r_i, $i = 1, \ldots, 5$. Kada ukupan dolazni protok premašuje kapacitet odlaznog linka, pristupni uređaj odbacuje pakete iz toka i s verovatnoćom p_i. Ako protoci r_i redom iznose 0,1, 0,2, 0,3, 0,4 i 0,5, izvedite vezu između r_i, p_i i C.



Slika 11.5: *Povezivanje stanica preko pristupnog mrežnog uređaja.*

Ako se paketi iz i-tog toka odbacuju s verovatnoćom p_i, ekvivalentni protok koji se iz ovog toka prosleđuje na odlazni link biće

$$r_i^{'} = (1 - p_i)r_i.$$

Da se ne bi premašio kapacitet odlaznog linka, mora važiti

$$\sum_i^5 r_i^{'} \leqslant C,$$

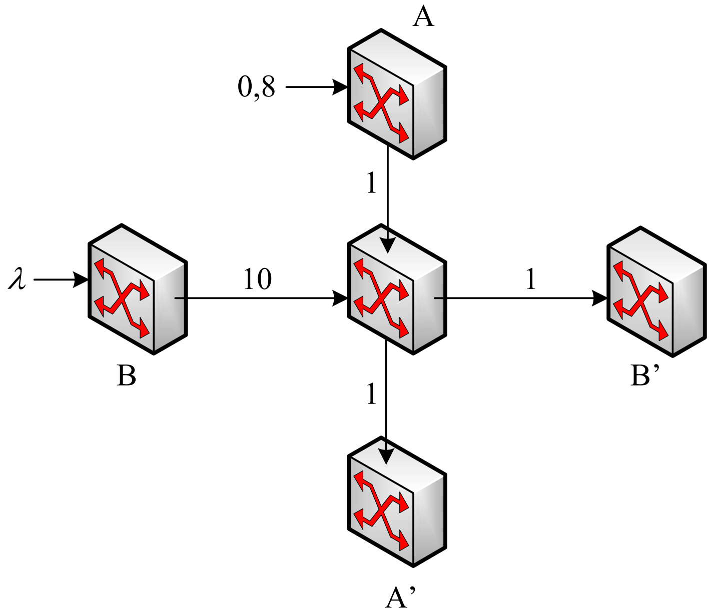
odnosno, u graničnom slučaju

$$\sum_i^5 (1 - p_i)r_i = C.$$

Neke od mogućih strategija bile bi:

- $p_i = 1/3$ (raspoloživi kapacitet se ravnomerno deli svim korisnicima),
- $p_1 = 0{,}091$, $p_2 = 2p_1$, $p_3 = 3p_1$, $p_4 = 4p_1$, $p_5 = 5p_1$ (ravnomerno se „nagrađuju” manje zahtevni korisnici),
- $p_1 = 0$, $p_2 = 0$, $p_3 = 0$, $p_4 = 0$, $p_5 = 1$ (kapacitet se redom dodeljuje korisnicima, počevši od najmanje zahtevnih, sve dok se resursi ne iscrpe).

Zadatak 11.6 Na slici je prikazana telekomunikaciona mreža. Svake sekunde, čvor A u proseku emituje 0,8 paketa ka čvoru A', a čvor B λ paketa ka čvoru B'. Uz svaki link, naznačen je njegov kapacitet u paketima u sekundi. Centralni čvor ima zajednički bafer za dolazni saobraćaj i, u slučaju zagušenja, odbacuje sve prekomerne pakete.



Slika 11.6: *Struktura posmatrane mreže.*

Odredite zavisnost ukupnog protoka na odlaznim linkovima iz centralnog čvora od λ.

Kada je $0 \leqslant \lambda \leqslant 1$, nema zagušenja i celokupni saobraćaj se prenosi do odgovarajućih odredišta: 0,8 paketa u sekundi ka A' i λ paketa u sekundi ka B'.

Za $1 < \lambda \leqslant 1{,}2$, nastupa zagušenje zbog premašenja kapaciteta linka ka čvoru B'. Protok ka čvoru A' je $0{,}8\ \mathrm{s}^{-1}$, a ka čvoru B' je jednak kapacitetu linka, $1\ \mathrm{s}^{-1}$.

Za $1{,}2 < \lambda \leqslant 10$, premašen je kapacitet obrade u centralnom čvoru. Njegova ukupna propusna moć od 2 paketa u sekundi tada se raspodeljuje na saobraćaj ka čvorovima A' i B'. Ka čvoru A' paketi se prosleđuju s protokom

$$V_{A'} = 2\ \mathrm{s}^{-1} \frac{0{,}8\ \mathrm{s}^{-1}}{0{,}8\ \mathrm{s}^{-1} + \lambda},$$

a ka čvoru B' s protokom

$$V_{B'} = 2\ \mathrm{s}^{-1} \frac{\lambda}{0{,}8\ \mathrm{s}^{-1} + \lambda},$$

jer ukupni odlazni saobraćaj iz centralnog čvora ne može da premaši vrednost od 2 paketa u sekundi.

Za još veću vrednost λ, premašuje se i kapacitet linka koji čvor B povezuje s centralnim čvorom. Protok na tom linku dostiže svoj maksimum od 10 paketa u sekundi. Ukupan ulazni saobraćaj u centralni čvor stoga iznosi 10,8 paketa u sekundi. Protok ka čvoru A' iznosi

$$V_{A'} = 2\ \mathrm{s}^{-1} \frac{0{,}8\ \mathrm{s}^{-1}}{0{,}8\ \mathrm{s}^{-1} + 10\ \mathrm{s}^{-1}} = 0{,}148\ \mathrm{s}^{-1},$$

a ka čvoru B'

$$V_{B'} = 2\ \mathrm{s}^{-1} \frac{10\ \mathrm{s}^{-1}}{0{,}8\ \mathrm{s}^{-1} + 10\ \mathrm{s}^{-1}} = 1{,}852\ \mathrm{s}^{-1}.$$

Čitaocu se preporučuje da nacrta grafike ovih zavisnosti.

Zadatak 11.7 Parametri uobličavača saobraćaja na principu „levka" (*leaky bucket*) su: verovatnoća da je izvor konforman $p = 0{,}3$, srednji protok na liniji kada je izvor konforman $\lambda_a = 5$ Mb/s, srednji protok *bursta* (kada izvor nije konforman) $\sigma = 30$ Mb/s, vršni protok $\lambda_1 = 200$ Mb/s, maksimalni izlazni protok $\lambda_b = 10$ Mb/s, veličina paketa $L = 400$ b, kapacitet bafera $B = 5$ paketa. Korišćenjem teorije servisnog sistema M/M/1/$B+1$, odredite:

a) propusnost uobličavača,

b) protok odbačenih paketa i

c) verovatnoću odbacivanja paketa.

Prosečan protok na liniji je

$$\overline{\lambda} = p\lambda_a + (1-p)\sigma = 22{,}5\ \mathrm{Mb/s}.$$

Minimalno trajanje paketa je $T = L/\lambda_1$. Verovatnoća dolaska paketa tokom intervala trajanja T je

$$a = \frac{\overline{\lambda}}{\lambda_1} = 0{,}1125.$$

Minimalna verovatnoća odlaska paketa iz uobličavača je

$$c = \frac{\lambda_b}{\lambda_1} = 0{,}05.$$

Podsetimo se izraza za verovatnoću stanja servisnog sistema M/M/1/$B+1$:

$$p_i = \frac{(1-\rho)\rho^i}{1-\rho^{B+2}}, \quad 0 \leqslant i \leqslant B+1,$$

pri čemu je u ovom slučaju

$$\rho = \frac{a(1-c)}{(1-a)c} = 2{,}408.$$

a) Propusnost uobličavača na principu „levka" data je izrazom

$$Th = \frac{\lambda_1}{L}\, c\,(1-(1-a)p_0)$$

i iznosi $2{,}493 \cdot 10^4$ s^{-1}.

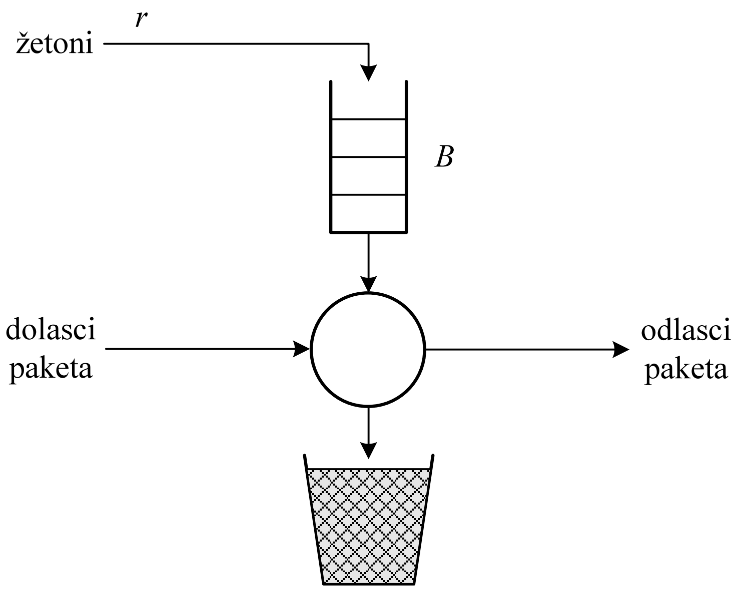
b) Protok odbačenih paketa je

$$\lambda_L = \frac{\lambda_1}{L} a(1-c)p_{B+1} = 3{,}131 \cdot 10^4 \text{ s}^{-1}.$$

c) Verovatnoća gubitka paketa je

$$p_L = \frac{\lambda_1}{\overline{\lambda}} a(1-c)p_{B+1} = 0{,}56.$$

Zadatak 11.8 Na slici je ilustrovan uobličavač saobraćaja na principu „kantice sa žetonima" (*token bucket*), čiji su parametri: kapacitet „kantice" $B = 250$ KB, protok dolaska žetona $r = 2$ Mb/s, maksimalni izlazni protok $M = 25$ Mb/s. Odredite maksimalno trajanje toka koji iz uobličavača može izlaziti s protokom M.



Slika 11.8: *Uobličavač saobraćaja na principu „kantice sa žetonima".*

Tok koji iz uobličavača izlazi s maksimalnim protokom M naziva se *burst*. Označimo maksimalno trajanje *bursta* sa S. Ovo maksimalno trajanje odgovara slučaju kada je „kantica" bila puna.

Da bi iz uobličavača izašla neka količina podataka sadržana u paketima, potrebno je da se utroši ista količina podataka sadržana u žetonima. Tokom vremena S, iz uobličavača će izaći količina podataka MS. „Kantica" kapaciteta B će se isprazniti, ali će u nju doći i novi žetoni, u ukupnom iznosu rS. Na osnovu ove diskusije, zaključujemo da mora važiti

$$B + rS = MS,$$

odakle dobijamo

$$S = \frac{B}{M - r}.$$

Za uobličavač iz zadatka, maksimalno trajanje *bursta* iznosi 89 ms.

Zadatak 11.9 Parametri uobličavača saobraćaja na principu „kantice sa žetonima" su: verovatnoća da je izvor konforman $p = 0{,}6$, srednji protok na liniji kada je izvor konforman $\lambda_a = 10$ Mb/s, srednji protok *bursta* (kada izvor nije konforman) $\sigma = 50$ Mb/s, vršni protok $\lambda_1 = 100$ Mb/s, protok dolazaka žetona $\lambda_t = 15$ Mb/s, prosečna veličina paketa $L = 400$ b, kapacitet bafera paketa $B_p = 5$, kapacitet bafera žetona $B_t = 2$. Uz pretpostavku da u svakom trenutku u uobličavač ulazi ili iz njega izlazi izlazi najviše jedan paket ili žeton, odredite verovatnoću da se u uobličavaču nalaze dva paketa.

Prosečan protok na ulaznoj liniji je

$$\overline{\lambda} = p\lambda_a + (1 - p)\sigma = 26 \text{ Mb/s}.$$

Minimalno trajanje paketa je $T = L/\lambda_1$. Verovatnoća dolaska žetona tokom intervala trajanja T je

$$a = \frac{\lambda_t}{\lambda_1} = 0{,}15,$$

dok je verovatnoća odlaska žetona iz uobličavača

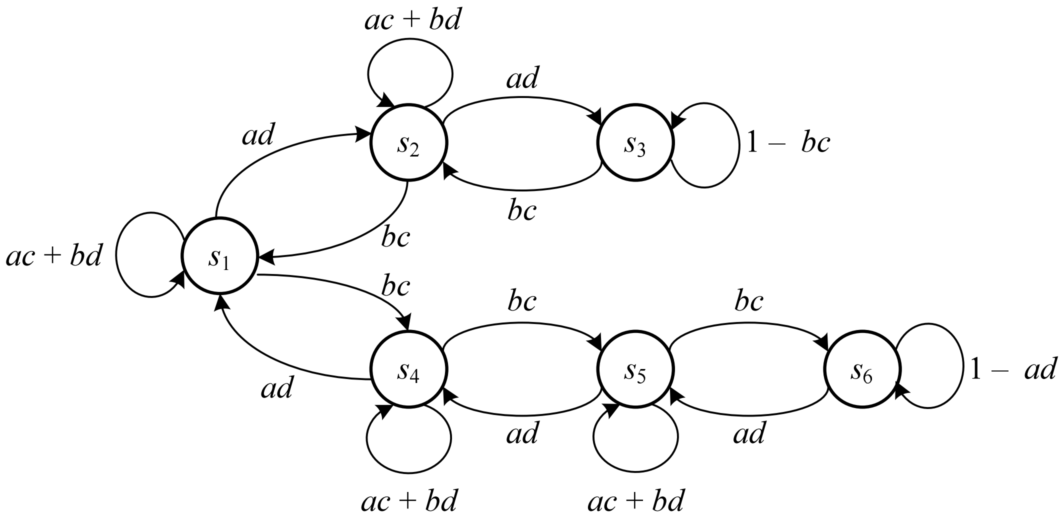
$$c = \frac{\overline{\lambda}}{\lambda_1} = 0{,}26.$$

Markovski lanac koji opisuje moguća stanja ovoga uobličavača prikazan je na slici, pri čemu je $b = 1 - a$ i $d = 1 - c$.

Oznake pojedinih stanja objašnjene su u tabeli. Slučaju kada se u uobličavaču nalaze dva paketa odgovara stanje s_5.

Oznake stanja lanca.

Stanje	Popunjenost bafera žetona	Popunjenost bafera paketa
s_1	0	0
s_2	1	0
s_3	2	0
s_4	0	1
s_5	0	2
s_6	0	3



Slika 11.9: *Dijagram stanja markovskog lanca.*

Matrica prelaska lanca je

$$\mathbf{Q} = \begin{bmatrix} ac+bd & bc & 0 & ad & 0 & 0 \\ ad & ac+bd & bc & 0 & 0 & 0 \\ 0 & ad & 1-bc & 0 & 0 & 0 \\ bc & 0 & 0 & ac+bd & ad & 0 \\ 0 & 0 & 0 & bc & ac+bd & ad \\ 0 & 0 & 0 & 0 & bc & 1-ad \end{bmatrix}.$$

Označimo verovatnoće stanja lanca s_i, $i = 1, 2, \ldots, 6$ redom s P_i i uredimo ih u vektor $\mathbf{P} = [P_1\ P_2\ \cdots\ P_6]$. U stacionarnom stanju će važiti

$$\mathbf{QP} = \mathbf{P}.$$

Rešavanjem ovoga sistema, uz uslov $\sum_{i=1}^{6} P_i = 1$, dobijamo vektor verovatnoća

$$\mathbf{P} = [0{,}0641 \quad 0{,}0322 \quad 0{,}0162 \quad 0{,}1276 \quad 0{,}2541 \quad 0{,}5059]^T.$$

Odavde je verovatnoća stanja s_5 0,2541.

Zadatak 11.10 U telekomunikacionoj mreži s ATM, koristi se uobličavač na bazi virtuelnog raspoređivača ćelija (VS, *virtual scheduler*). Prosečan protok na ulaznoj liniji je $\lambda = 150$ Mb/s. Verovatnoća da trenutak dolaska ćelije, t, zadovoljava nejednakost $t \geqslant TAT$, gde je TAT teorijsko vreme dolaska je $a = 0{,}2$. Verovatnoća da važi $TAT - \Delta \leqslant t \leqslant TAT$ je $b = 0{,}5$, dok je verovatnoća da važi $t < TAT - \Delta$ $c = 0{,}3$; pri tome je Δ kratki period vremena. Kapacitet bafera je $B = 5$ ćelija. Korišćenjem teorije servisnog sistema M/M/1/$B+1$, odredite:

a) propusnost uobličavača,

b) verovatnoću odbacivanja ćelije i

c) protok odbačenih ćelija.

Teorijsko vreme dolaska ćelije je

$$TAT = \frac{L}{\lambda},$$

gde je $L = 424$ b dužina ćelije. TAT predstavlja procenu vremena između početaka zaglavlja dvaju sukcesivno pristiglih ćelija. Primetimo takođe da važi $a + b + c = 1$.

Izraz za verovatnoću stanja servisnog sistema M/M/1/$B + 1$ je

$$p_i = \frac{(1-\rho)\rho^i}{1-\rho^{B+2}}, \quad 0 \leqslant i \leqslant B+1,$$

pri čemu je sada

$$\rho = \frac{c}{a} = 1{,}5.$$

a) Propusnost VS uobličavača je

$$Th = (1 - c\,p_{B+1})\lambda = 134{,}1 \cdot 10^6 \text{ s}^{-1}.$$

b) Verovatnoća odbacivanja ćelije je

$$p_L = c\,p_{B+1} = 0{,}11.$$

c) Protok odbačenih ćelija je

$$\lambda_L = p_L \lambda = 15{,}93 \cdot 10^6 \text{ s}^{-1}.$$

Zadatak 11.11 Multiplekser s ravnomernim opsluživanjem (*round robin*) opslužuje n redova čekanja. Trajanje ciklusa opsluživanja je $T_c = 1{,}2$ ms. Izlazni link je SDH STM-1. Koliko najkraćih Ethernet paketa ovaj multiplekser može opslužiti u svakom ciklusu?

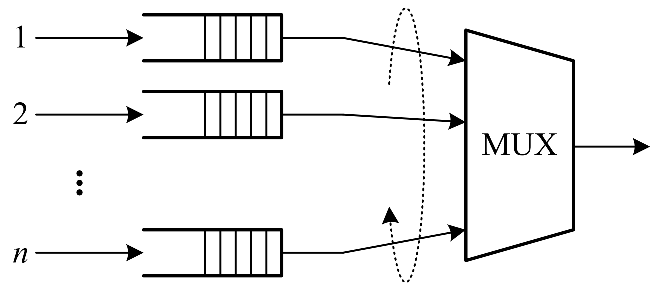
Linkom čiji je kapacitet C svake sekunde može se preneti

$$c = \frac{C}{L}$$

paketa dužine L.

U svakom ciklusu trajanja T_c, ovakav multiplekser može opslužiti

$$n = c\,T_c$$



Slika 11.11: *Ravnomerno opsluživanje redova čekanja.*

paketa.

Podsetimo se da dužina najkraćeg paketa u Ethernetu iznosi $L = 72$ B (zadatak 6.3). Protok na linku STM-1 je 155,52 Mb/s, pri čemu je kapacitet njegovog korisnog segmenta $C = 149{,}76$ Mb/s. Uvrštavanjem ovih brojčanih vrednosti, dobijamo da multiplekser iz zadatka u svakom ciklusu može opslužiti 312 paketa.

Zadatak 11.12 Četiri saobraćajna toka opslužuju se po principu WRR (*weighted round robin*). Težinski koeficijenti tokova, w_i i dužine paketa u njima, L_i, dati su u tabeli.

i	w_i	L_i [b]
1	3	400
2	2	600
3	1	500
4	1	200

Ukoliko kapacitet izlaznog linka iznosi $C = 20$ Mb/s, odredite:

a) trajanje ciklusa opsluživanja i

b) deo kapaciteta izlaznog linka koji se dodeljuje svakom od tokova.

a) Kod discipline WRR, u svakom ciklusu opsluživanja iz reda i se na izlazni link šalje w_i paketa. Trajanje jednog ciklusa stoga je

$$T_c = \frac{1}{C}\sum_{i=1}^{n} w_i L_i,$$

gde je n broj tokova saobraćaja koji se opslužuju. Trajanje ciklusa opsluživanja za raspoređivač iz našeg zadatka iznosi $1{,}55 \cdot 10^{-4}$ s.

b) Udeo toka i u ukupnom kapacitetu izlaznog linka je

$$f_i = \frac{w_i L_i}{\sum\limits_{j=1}^{n} w_j L_j}.$$

Odavde dobijamo: $f_1 = f_2 = 0{,}387$, $f_3 = 0{,}161$ i $f_4 = 0{,}065$. Tokovima 1 i 2 stoga na raspolaganju stoji kapacitet 7,74 Mb/s, toku 3 3,22 Mb/s i toku 4 1,3 Mb/s.

Zadatak 11.13 Pet tokova saobraćaja, čiji su protoci (u Mb/s) $\lambda_1 = 10$, $\lambda_2 = 20$, $\lambda_3 = 60$, $\lambda_4 = 80$ i $\lambda_5 = 80$, dele izlazni link kapaciteta $C = 155$ Mb/s po principu „max-min”. Odredite kapacitete koji se dodeljuju svakome od njih.

„Max-min” algoritam je iterativna tehnika za deljenje resursa. Primenjuje se u FQ (*fair queuing*) raspoređivaču paketa. Algoritam se izvršava na sledeći način.

Neka je kapacitet izlaznog linka C i neka ga deli n tokova saobraćaja. Neka dalje tok i zahteva kapacitet λ_i. Pretpostavićemo da su tokovi sortirani po neopadajućem redosledu zahtevanih kapaciteta, tako da važi

$$\lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n.$$

Neka je preostali raspoloživi kapacitet izlaznog linka nakon što je obrađeno k tokova C_k. Tada se toku $(k+1)$ dodeli kapacitet

$$\lambda'_{k+1} = \min\left(\lambda_{k+1},\ \frac{C_k}{n-k}\right).$$

Opisana procedura se izvršava sve dok se svim tokovima ne bude dodelio udeo u kapacitetu izlaznog linka.

U našem zadatku, ukupni zahtevi posmatranih tokova iznose 250 Mb/s, što je veće od kapaciteta izlaznog linka. Početni deo kapaciteta iznosi 155 Mb/s $\cdot \frac{1}{5} = 31$ Mb/s. Toku broj 1 se dodeljuje kapacitet

$$\lambda'_1 = \min(10\ \text{Mb/s},\ 31\ \text{Mb/s}) = 10\ \text{Mb/s}.$$

Preostali kapacitet izlaznog linka od 155 Mb/s $-$ 10 Mb/s $=$ 145 Mb/s potom se raspodeljuje na 4 preostala toka: 145 Mb/s $\cdot \frac{1}{4} = 36{,}25$ Mb/s. Toku broj 2 se dodeljuje

$$\lambda'_2 = \min(20\ \text{Mb/s},\ 36{,}25\ \text{Mb/s}) = 20\ \text{Mb/s}.$$

Za tokove 3, 4 i 5, sada na raspolaganju ukupno preostaje 125 Mb/s, ili po 41,7 Mb/s. Toku broj 3 će se dodeliti

$$\lambda'_3 = \min(60\ \text{Mb/s},\ 41{,}7\ \text{Mb/s}) = 41{,}7\ \text{Mb/s}.$$

Za tokove 4 i 5 preostaje po 41,7 Mb/s. Stoga će biti

$$\lambda'_4 = \min(80\ \text{Mb/s},\ 41{,}7\ \text{Mb/s}) = 41{,}7\ \text{Mb/s}$$

i

$$\lambda'_5 = 41{,}7\ \text{Mb/s}.$$

Zadatak 11.14 Tri toka saobraćaja se opslužuju po principu deljenja procesora (PS, *processor sharing*). Paketi dolaze u raspoređivač prema sledećoj tabeli. Nula označava da tok u datom trenutku nije aktivan, te da ne zahteva kapacitet izlaznog linka, dok jedinica označava da je tok aktivan i da mu treba dodeliti ravnopravan deo raspoloživog kapaciteta.

vreme	0	1	2	3	4
tok 1	0	1	0	0	1
tok 2	1	1	1	0	1
tok 3	0	0	1	1	1

Odredite deo kapaciteta izlaznog linka koji se dodeljuje aktivnim tokovima u svakom od posmatranih trenutaka.

U disciplini deljenja procesora, ulazni tokovi saobraćaja se tretiraju kao idealni fluidi. Ako je u trenutku t aktivno $m(t)$ tokova, svakom od njih se dodeljuje kapacitet

$$\lambda(t) = \frac{C}{m(t)},$$

gde je C kapacitet izlaznog linka.

U našem zadatku, to znači da se u trenutku 0 svih 100% kapaciteta izlaznog linka dodeljuje toku 1. U trenutku 1, tokovi 1 i 2 dobijaju po 50% kapaciteta, u trenutku 2, tokovi 2 i 3 po 50%, u trenutku 3 tok 3 svih 100% i, na kraju, u trenutku 4 sva tri toka dobijaju po 33,3% raspoloživog kapaciteta izlaznog linka.

Zadatak 11.15 Raspoređivač paketa na principu generalisanog deljenja procesora (*generalized processor sharing*, GPS) opslužuje četiri sesije, čiji su težinski koeficijenti $w_1 = 4$, $w_2 = 2$, $w_3 = 3$ i $w_4 = 1$. Kapacitet izlaznog linka je $C = 10$ Mb/s. Redosled dolazaka paketa prikazan je u tabeli, pri čemu broj u odgovarajućem redu označava dužinu paketa koji stiže u posmatranom trenutku, u kb.

vreme [ms]	0	1	2
sesija 1	3	—	2
sesija 2	—	1	3
sesija 3	2	—	—
sesija 4	—	7	—

Odredite kapacitete koji se dodeljuju sesijama i vremena završetka prenosa pristiglih paketa.

GPS raspoređivač radi na sledeći način. Neka je w_i težinski koeficijent i-te sesije ili toka saobraćaja. U trenutku t, ovoj sesiji se na raspolaganje stavlja deo kapaciteta izlaznog linka $c_i(t)$, koji je dat izrazom

$$c_i(t) = C \frac{w_i}{\sum\limits_{j \in B(t)} w_j},$$

gde je $B(t)$ skup sesija koje treba opslužiti u trenutku t. Broj bita iz posmatrane sesije koji se prenesu u intervalu vremena $(t_1,\, t_2)$ je

$$s_i = \int\limits_{t_1}^{t_2} c_i(t)\,\mathrm{d}t.$$

Vreme T koje je potrebno za prenos paketa dužine L_i iz sesije i određuje se kao rešenje jednačine

$$L_i = \int\limits_{0}^{T} c_i(t)\,\mathrm{d}t.$$

U našem zadatku, u trenutku $t = 0$ ms, treba opslužiti sesije 1 i 3. Njima se dodeljuju kapaciteti

$$c_1(0) = C \frac{w_1}{w_1 + w_3} = 5{,}714 \text{ Mb/s}$$

i

$$c_3(0) = C \frac{w_3}{w_1 + w_3} = 4{,}286 \text{ Mb/s}.$$

Slanje paketa iz prve sesije završiće se u

$$T_1 = \frac{L_1}{c_1(0)} = 5{,}25 \cdot 10^{-4} \text{ s},$$

a paketa iz treće sesije u

$$T_3 = \frac{L_3}{c_3(0)} = 4{,}67 \cdot 10^{-4} \text{ s}.$$

U trenutku $t = 1$ ms, treba opslužiti sesije 2 i 4. Njima se dodeljuju kapaciteti $c_2(1) = 6{,}667$ Mb/s i $c_4(1) = 3{,}333$ Mb/s. Završetak slanja paketa iz sesije 2 biće u trenutku

$$T_2 = 1 \text{ ms} + \frac{L_2}{c_2(1)} = 1{,}15 \text{ ms},$$

dok bi se slanje paketa iz četvrte sesije završilo u trenutku

$$T_4 = 1 \text{ ms} + \frac{L_4}{c_4(1)} = 3{,}1 \text{ ms}.$$

Zaključujemo da se do trenutka $t = 2$ ms neće preneti celokupan paket četvrte sesije, već deo od

$$s_4 = c_4(1)(2 \text{ ms} - 1 \text{ ms}) = 3{,}333 \text{ kb},$$

te stoga za slanje preostaje još 3,667 kb.

U trenutku $t = 2$ ms, sesije dobijaju kapacitete $c_1(2) = 5{,}714$ Mb/s, $c_2(2) = 2{,}857$ Mb/s i $c_4(2) = 1{,}429$ Mb/s. Pošto pretpostavljamo da u raspoređivač više ne stižu novi paketi, vremena završetka prenosa biće

$$T_1 = 2\text{ ms} + \frac{L_1}{c_1(2)} = 2{,}35\text{ ms},$$

$$T_2 = 2\text{ ms} + \frac{L_2}{c_2(2)} = 3{,}05\text{ ms},$$

$$T_4 = 2\text{ ms} + \frac{L_4 - s_4}{c_4(2)} = 4{,}57\text{ ms}.$$

Zadatak 11.16 Raspoređivač paketa na principu nepreemptivnog PGPS/WFQ (*packet-by-packet GPS/weighted fair queuing*) opslužuje četiri saobraćajna toka, jednakih težinskih koeficijenata. Kapacitet izlaznog linka je $C = 1$ Mb/s. Redosled dolazaka paketa prikazan je u tabeli, pri čemu broj u odgovarajućem redu označava dužinu paketa koji stiže u posmatranom trenutku, u kb. Odredite virtuelno vreme u sistemu i brojeve završetka prenosa pojedinih paketa.

vreme [ms]	0	1	2	3
tok 1	—	—	2	—
tok 2	3	2	1	—
tok 3	—	1	—	2
tok 4	1	—	5	—

Neka je $B(t)$ skup tokova (sesija) koje treba opslužiti u trenutku t. Virtuelno vreme, $V(t)$, definiše se na sledeći način:

$$V(0) = 0,$$

$$\frac{\mathrm{d}V(t)}{\mathrm{d}t} = \frac{1}{\sum\limits_{i \in B(t)} w_i},$$

gde je $w_i \geqslant 1$ težinski koeficijent i-tog toka. Ako u datom trenutku ne treba opslužiti nijedan tok, tada je $V(t) = 0$.

Ako je paket k došao u red za tok i, tada se njegov broj završetka slanja računa po obrascu

$$F_i(k) = \max\left(F_i(k-1),\ V(t)\right) + \frac{L_k}{w_i},$$

gde je $F_i(k-1)$ broj završetka prethodnog paketa u redu i L_k dužina novopridošlog paketa. Praznom redu odgovara nulti broj završetka.

Pošto se izabranom redu dodeljuje celokupan raspoloživi kapacitet izlaznog linka, vreme potrebno za prenos paketa biće

$$T = \frac{L_k}{C}.$$

Pogledajmo šta se dešava u raspoređivaču iz našeg zadatka. U trenutku $t = 0$, biće $V(0) = 0$, a pošto su aktivna dva toka (broj 2 i broj 4), biće $\mathrm{d}V/\mathrm{d}t = 1/2$. Brojevi završetka za odgovarajuće pakete su

$$F_2(1) = \max(0,\ 0) + 3 = 3$$

i

$$F_4(1) = \max(0,\ 0) + 1 = 1.$$

Paket iz četvrtog toka ima manji broj završetka, pa se on prvi šalje. Njegov prenos će trajati 1 ms i završiće se u $T = 1$ ms.

U trenutku $t = 2$ ms, pored paketa iz toka broj 2, koji čeka u redu, dolazi i novi paket iz ovog toka, kao i paket iz toka broj 3. Virtuelno vreme sada je $V(1) = 1/2$, a pošto su i dalje aktivna dva toka ostaje $\mathrm{d}V/\mathrm{d}t = 1/2$. Brojevi završetka za nove pakete su

$$F_2(2) = \max(3,\ 0{,}5) + 2 = 5$$

i

$$F_3(1) = \max(0,\ 0{,}5) + 1 = 1{,}5.$$

Sada se šalje paket iz trećeg reda, za šta je potrebna 1 ms. Prenos će se završiti u trenutku $T = 2$ ms.

U trenutku $t = 2$ ms, dolaze novi paketi iz tokova broj 1, 2 i 5, pri čemu se u redu iz drugog toka već nalaze dva paketa. Novo virtuelno vreme je $V(2) = 1$, dok je $\mathrm{d}V/\mathrm{d}t = 1/3$. Brojevi završetka za nove pakete su

$$F_1(1) = \max(0,\ 1) + 2 = 3,$$

$$F_2(3) = \max(5,\ 1) + 1 = 6$$

i

$$F_4(2) = \max(0,\ 1) + 5 = 6.$$

Za slanje se bira paket iz prvog toka. Za njegovo slanje bi trebale 2 ms, što znači da se ono ne bi završilo do trenutka $t = 3$ ms, kada dolazi novi paket. Pošto se, po postavci zadatka, radi o *nepreemptivnom* raspoređivaču, slanje ovog paketa se *ne prekida.*

U trenutku $t = 3$ ms, nastavlja se slanje paketa iz prvog toka. U redovima za drugi i četvrti tok čekaju paketi, dok iz trećeg toka stiže novi paket. Treba opslužiti ukupno četiri reda, pa je $\mathrm{d}V/\mathrm{d}t = 1/4$ i $V(3) = 1{,}33$. Broj završetka za novi paket je

$$F_3(2) = \max(0,\ 1{,}33) + 2 = 3{,}33.$$

U trenutku $t = 4$ ms, dovršava se slanje paketa iz prvog toka. Paketi više ne dolaze, pa se opslužuju oni koji čekaju u redovima, saglasno njihovim brojevima završetaka.

U ovom trenutku se šalje paket $p_2(1)$, čije se slanje završava u $t = 7$ ms. Tada na red dolazi $p_3(2)$. U $t = 9$ ms, šalje se $p_2(2)$, u $t = 11$ ms $p_2(3)$ i konačno u $t = 12$ ms $p_4(2)$; njegovo slanje će se završiti u trenutku $t = 17$ ms.

Zadatak 11.17 Raspoređivač paketa na principu FFQ (*frame-based fair queuing*) opslužuje četiri saobraćajna toka, jednakih težinskih koeficijenata. Kapacitet izlaznog linka je $C = 1$ Mb/s, a veličina otvora prozora $F = 10^4$ b. Redosled dolazaka paketa prikazan je u tabeli, pri čemu broj u odgovarajućem redu označava dužinu paketa koji stiže u posmatranom trenutku, u kb.

vreme [ms]	0	1	2	3
tok 1	3	—	2	—
tok 2	—	1	—	4
tok 3	2	—	7	—
tok 4	—	—	—	5

Odredite potencijal sistema i vremenske oznake pojedinih paketa.

Neka je izvor započeo slanje paketa u trenutku t_s. Potencijal sistema u trenutku $t \geqslant t_s$ dat je izrazom

$$P(t) = P(t_s) + \frac{C(t - t_s)}{F},$$

gde je $P(t_s)$ potencijal sistema u trenutku t_s, C kapacitet izlaznog linka i F otvor prozora, u bitima. Ako nema sesija koje treba opslužiti, potencijal sistema jednak je nuli.

Otvor prozora bira se tako da bude veći od najveće dužine paketa, $F > L_{max}$.

Ako paket k dolazi u sesiju i, pridružuje mu se vremenska oznaka

$$S_i(k) = \max\left(S_i(k-1),\ P(t)\right) + \frac{L_k}{\lambda_i},$$

gde je $S_i(k-1)$ oznaka prethodnog paketa u redu, L_k dužina novopridošlog paketa i λ_i kapacitet koji je rezervisan za sesiju i. Praznom redu odgovara nulta vremenska oznaka.

Svaki put kada se izabere paket za slanje, ažurira se vrednost brojača prenesenih bita,

$$B(k) = B(k-1) + L_k.$$

Ako je $B(k) \leqslant F$, nastavlja se tekući prozor i paket se šalje na izlazni link; u suprotnom, započinje se novi prozor i posmatrani paket se šalje u njemu.

Pošto se izabranom redu dodeljuje celokupan raspoloživi kapacitet izlaznog linka, vreme potrebno za prenos prvog paketa (HoL, *head of line*) biće

$$T = \frac{L_k}{C}.$$

U našem zadatku, u trenutku $t = 0$ biće $B(0) = 0$ i $P(0) = 0$. U raspoređivač dolaze paketi iz prvog i trećeg toka. Njihove vremenske oznake biće

$$S_1(1) = \max(0,\ 0) + 3 = 3,$$

$$S_3(1) = \max(0,\ 0) + 2 = 2.$$

Prvi se šalje paket $p_3(1)$, za šta su potrebne 2 ms.

U trenutku $t = 1$ ms, biće $B(1) = 1$ i $P(1) = 0{,}1$. Vremenska oznaka novopridošlog paketa $p_2(1)$ je

$$S_2(1) = \max(0,\ 0{,}1) + 1 = 1{,}1.$$

Nastavlja se slanje paketa $p_3(1)$.

U trenutku $t = 2$ ms, biće $B(2) = 2$ i $P(2) = 0{,}2$. Vremenske oznake novih paketa su

$$S_1(2) = \max(3,\ 0{,}2) + 2 = 5$$

i

$$S_3(2) = \max(2,\ 0{,}2) + 7 = 9.$$

Šalje se paket $p_2(1)$.

U trenutku $t = 3$ ms, biće $B(3) = 3$ i $P(3) = 0{,}3$. Vremenske oznake novih paketa su

$$S_2(2) = \max(1{,}1,\ 0{,}3) + 4 = 5{,}1$$

i

$$S_4(1) = \max(0,\ 0{,}3) + 5 = 5{,}3.$$

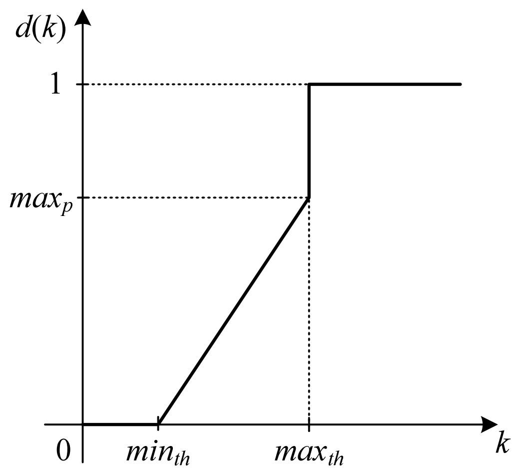
Šalje se paket $p_1(1)$.

Pod pretpostavkom da u raspoređivač više ne dolaze paketi, redosled slanja će biti: $p_1(2)$, $p_2(2)$, $p_4(1)$ i $p_3(2)$. Čitaocu se preporučuje da odredi vremena završetka njihovih slanja.

Zadatak 11.18 Posmatra se bafer u kome je primenjena detekcija zagušenja unapred (*Random Early Detection*, RED). Kapacitet bafera je $K = 60$ kB, donji prag je $min_{th} = 20$ kB, gornji prag $max_{th} = 50$ kB, dok je maksimalna verovatnoća odbacivanja konformnih paketa $max_p = 0{,}1$. Procenjena prosečna popunjenost bafera u datom trenutku je $L = 30$ kB, dok je broj paketa koji su pristigli posle poslednjeg odbačenog $n = 10$. Kolika je verovatnoća odbacivanja paketa?

Ukoliko se u baferu RED opsluživača nalazi k paketa, $0 \leqslant k \leqslant K$, novopridošli paket će se odbaciti s verovatnoćom

$$d(k) = \begin{cases} 0, & k < min_{th} \\ max_p \, \dfrac{k - min_{th}}{max_{th} - min_{th}}, & min_{th} \leqslant k \leqslant max_{th} \\ 1, & k > max_{th} \end{cases}.$$



Slika 11.18: *Verovatnoća odbacivanja paketa.*

Ova zavisnost je ilustrovana na slici.

Kada nije poznat tačan broj paketa u baferu, već samo procena njegove popunjenosti, L i broj novopridošlih paketa posle poslednjeg odbačenog, n, verovatnoću odbacivanja ćemo odrediti kao

$$\tilde{d} = \frac{d(L)}{1 - nd(L)}.$$

U našem zadatku je $d(L) = 0{,}033$, pa je $\tilde{d} = 0{,}05$.

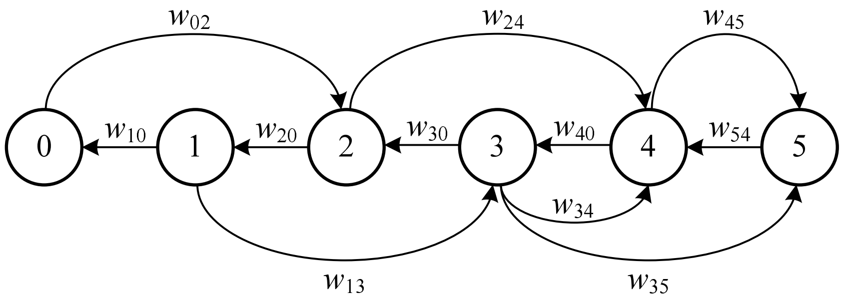
Zadatak 11.19 Na ulaz RED opsluživača saobraćaja dolazi tok za koji je veličina „bursta" $B = 2$ paketa, a ponuđeni saobraćaj $\rho = 2$ E. Protok opsluživanja je $\mu = 1\ \mathrm{s}^{-1}$, kapacitet bafera $K = 4$ paketa, donji prag $min_{th} = 2$, gornji prag $max_{th} = 4$, maksimalna verovatnoća odbacivanja konformnih paketa $max_p = 1$. Primenom Bonald-May-Bolotovog rezultata, odredite verovatnoću blokade ovog uobličavača.

Rešenje koje su predložili Bonald, May i Bolot zasniva se na analizi markovskih lanaca. Pretpostavka od koje su pošli je da paketi dolaze u grupama od po B, pri čemu su vremena međudolazaka eksponencijalno raspodeljena.

Dijagram stanja opsluživača prikazan je na slici 11.19. Ukoliko se u opsluživaču nalazi $i = k + 1$ paket, u baferu će ih biti k, $0 \leqslant k \leqslant K$, pri čemu će verovatnoča odbacivanja biti $d(k)$.

Kada su poznate verovatnoće stanja, P_i, verovatnoća blokade se računa kao

$$\mathrm{P}_{RED} = \sum_{i=1}^{K+1} P_i \, d(i-1).$$



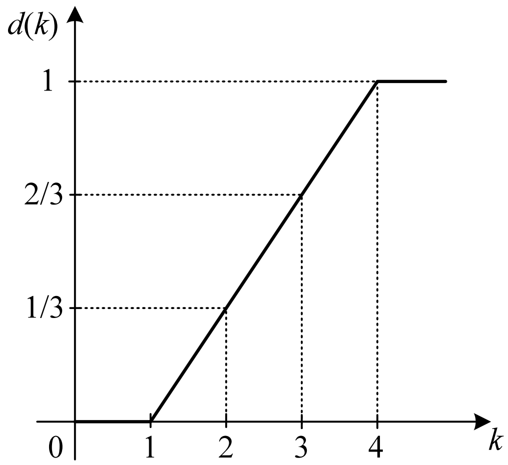
Slika 11.19: *Dijagram stanja opsluživača.*

Ponuđeni saobraćaj u našem zadatku je

$$\rho = B\frac{\lambda}{\mu},$$

pa je protok dolazaka „burstova" $\lambda = 0{,}3\ \mathrm{s}^{-1}$.

Grafik zavisnosti verovatnoće odbacivanja paketa od broja paketa u baferu, k, prikazan je na narednoj slici.



Verovatnoća odbacivanja paketa.

Odredimo protoke prelaska između pojedinih stanja. U svakom stanju, u uobličavač može doći „burst" od $B = 2$ paketa. U stanjima 0, 1 i 2 nema odbacivanja novopridošlih paketa, pa je

$$w_{02} = w_{13} = w_{24} = \lambda.$$

U stanju 3, u baferu se nalaze dva paketa, pa je verovatnoća odbacivanja novog paketa $d(2) = 1/3$. U stanje 4 se prelazi ako se odbaci tačno jedan paket iz „bursta", dok se u stanje 5 prelazi ako se ne odbaci nijedan:

$$w_{34} = \lambda\binom{2}{1}d(2)\,(1 - d(2)) = 0{,}44\lambda,$$

$$w_{35} = \lambda\,(1 - d(2))^2 = 0{,}44\lambda.$$

U stanju 4, novi paketi se odbacuju s verovatnoćom $d(3) = 2/3$. Prelazak iz stanja 4 u stanje 5 moguć je samo ako se ne odbace svi novi paketi iz „bursta”, pa je

$$w_{45} = \lambda\left(1 - (d(3))^2\right) = 0{,}56\lambda.$$

Protok povratka iz višeg u prvo niže stanje je μ, pa je

$$w_{54} = w_{43} = w_{32} = w_{21} = w_{10} = \mu.$$

Ostali prelasci nisu mogući, pa su njihovi protoci formalno jednaki nuli.

Matrica prelaska markovskog lanca je

$$\mathbf{W} = \begin{bmatrix} -0{,}3 & 0 & 0{,}3 & 0 & 0 & 0 \\ 1 & -1{,}3 & 0 & 0{,}3 & 0 & 0 \\ 0 & 1 & -1{,}3 & 0 & 0{,}3 & 0 \\ 0 & 0 & 1 & -1{,}264 & 0{,}132 & 0{,}132 \\ 0 & 0 & 0 & 1 & -1{,}168 & 0{,}168 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}.$$

Rešavanjem sistema

$$[P_0\ P_1\ P_2\ P_3\ P_4\ P_5]\,\mathbf{W} = \mathbf{0},$$

uz uslov

$$\sum_{i=0}^{5} P_i = 1,$$

dobijamo

$$[P_0\ P_1\ P_2\ P_3\ P_4\ P_5] = [0{,}471\ \ 0{,}141\ \ 0{,}184\ \ 0{,}097\ \ 0{,}081\ \ 0{,}026]\,.$$

Verovatnoća blokade ovog uobličavača stoga je 0,112.

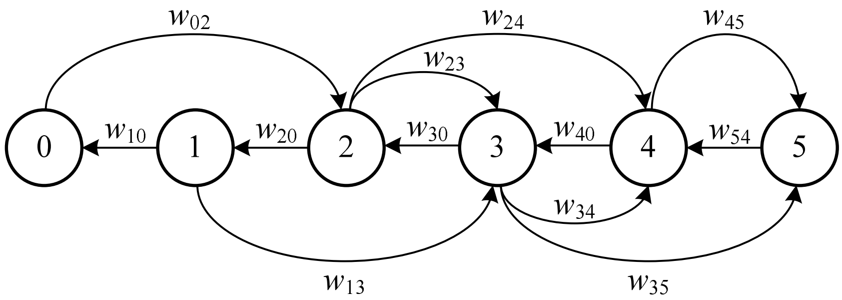
Zadatak 11.20 Veličina „bursta” koji dolazi u WRED (*Weighted* RED) opsluživač saobraćaja je $B = 2$ paketa, dok je ponuđeni saobraćaj $\rho = 2$ E. Parametri opsluživača su: protok opsluživanja $\mu = 1\ \mathrm{s}^{-1}$, kapacitet bafera $K = 4$ paketa, donji prag klase 1 $min_{th1} = 2$, donji prag klase 2 $min_{th2} = 1$, gornji prag za obe klase $max_{th} = 4$, maksimalna verovatnoća odbacivanja konformnih paketa obeju klasa $max_p = 1$. Odredite verovatnoću blokade ovog uobličavača.

Dijagram stanja opsluživača prikazan je na slici 11.20, dok su verovatnoće odbacivanja paketa u razmatranim klasama date u tabeli.

Ponuđeni saobraćaj je

$$\rho = B\frac{\lambda}{\mu},$$

pa je protok dolazaka „burstova” $\lambda = 0{,}3\ \mathrm{s}^{-1}$.



Slika 11.20: *Dijagram stanja opslužívača.*

Verovatnoće odbacivanja paketa.

Broj paketa u sistemu, i	$d_1(i)$	$d_2(i)$
0	0	0
1	0	0
2	0	1/4
3	1/3	1/2
4	2/3	3/4
5	1	1

Protoke prelaska odredićemo na sledeći način. U stanjima 0 i 1 nema odbacivanja paketa, pa je

$$w_{02} = w_{13} = \lambda = 0{,}3.$$

U stanju 2, odbacivanju su podložni samo paketi klase 2, pa je

$$w_{23} = \frac{1}{2}\lambda\binom{2}{1}d_2(2)\,(1 - d_2(2)) = 0{,}0563,$$

$$w_{24} = \frac{1}{2}\lambda + \frac{1}{2}\lambda\,(1 - d_2(2))^2 = 0{,}2344.$$

U stanju 3, obe klase su podložne odbacivanju. Stoga će biti

$$w_{34} = \frac{1}{2}\lambda\binom{2}{1}d_1(3)\,(1 - d_1(3)) + \frac{1}{2}\lambda\binom{2}{1}d_2(3)\,(1 - d_2(3)) = 0{,}1417,$$

$$w_{35} = \frac{1}{2}\lambda\,(1 - d_1(3))^2 + \frac{1}{2}\lambda\,(1 - d_2(3))^2 = 0{,}1042.$$

Prelazak iz stanja 4 u stanje 5 moguć je samo ako se ne budu odbacili svi paketi, pa je

$$w_{45} = \frac{1}{2}\lambda\left(1 - d_1^2(4)\right) + \frac{1}{2}\lambda\left(1 - d_2^2(4)\right) = 0{,}1490.$$

Protoci svih prelazaka u prvo niže stanje su μ:

$$w_{54} = w_{43} = w_{32} = w_{21} = w_{10} = \mu = 1.$$

Ostali prelasci nisu mogući.

Matrica prelaska sada glasi

$$\mathbf{W} = \begin{bmatrix} -0{,}3 & 0 & 0{,}3 & 0 & 0 & 0 \\ 1 & -1{,}3 & 0 & 0{,}3 & 0 & 0 \\ 0 & 1 & -1{,}29 & 0{,}0563 & 0{,}2344 & 0 \\ 0 & 0 & 1 & -1{,}2459 & 0{,}1417 & 0{,}1042 \\ 0 & 0 & 0 & 1 & -1{,}1490 & 0{,}1490 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}.$$

Odavde dobijamo vektor verovatnoća stanja

$$[P_0\ P_1\ P_2\ P_3\ P_4\ P_5] = [0{,}4817\ \ 0{,}1445\ \ 0{,}1879\ \ 0{,}0978\ \ 0{,}0680\ \ 0{,}0202]\,.$$

Verovatnoća blokade za pakete klase 1 je

$$\mathrm{P}_1 = \sum_{i=0}^{5} P_i\, d_1(i) = 0{,}0981,$$

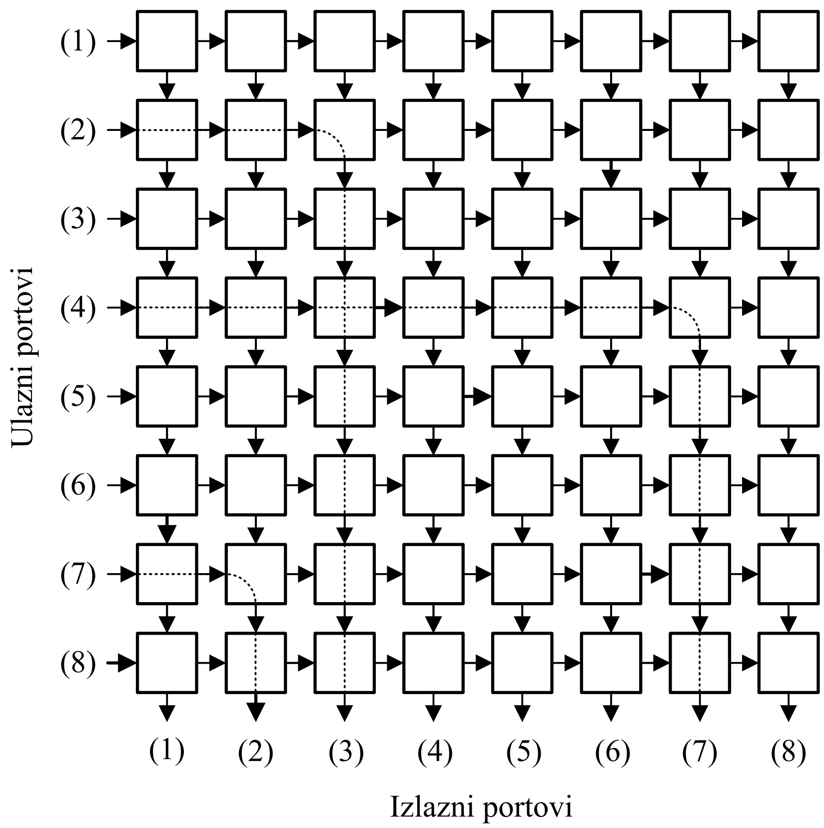
dok je za pakete klase 2

$$\mathrm{P}_2 = \sum_{i=0}^{5} P_i\, d_2(i) = 0{,}1671.$$

12. Mreže za međupovezivanje

Zadatak 12.1 Nacrtajte strukturnu shemu „crossbar” komutatora dimenzija 8×8. Pokažite kako se uspostavljaju veze (2, 3), (4, 7) i (7, 2).

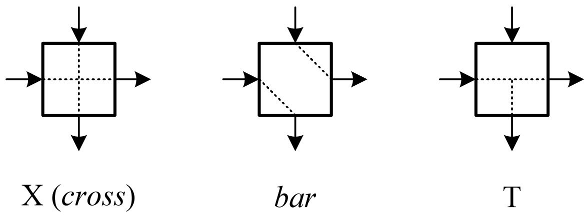
Crossbar komutator dimenzija $N \times N$ je matrične strukture, s N^2 ukrsnih tačaka. Shema crossbar komutatora 8×8 prikazana je na slici.



Slika 12.1: *Crossbar komutator.*

Moguća stanja crossbar ukrsne tačke ilustrovana su na slici na narednoj strani.

Veza između ulaznog porta i i izlaznog j komutatora ostvaruje se tako što se ukrsne tačke na položajima $(i, 1)$–$(i, j-1)$ i $(i+1, j)$–(N, j) postavljaju u stanje „X” (ranije



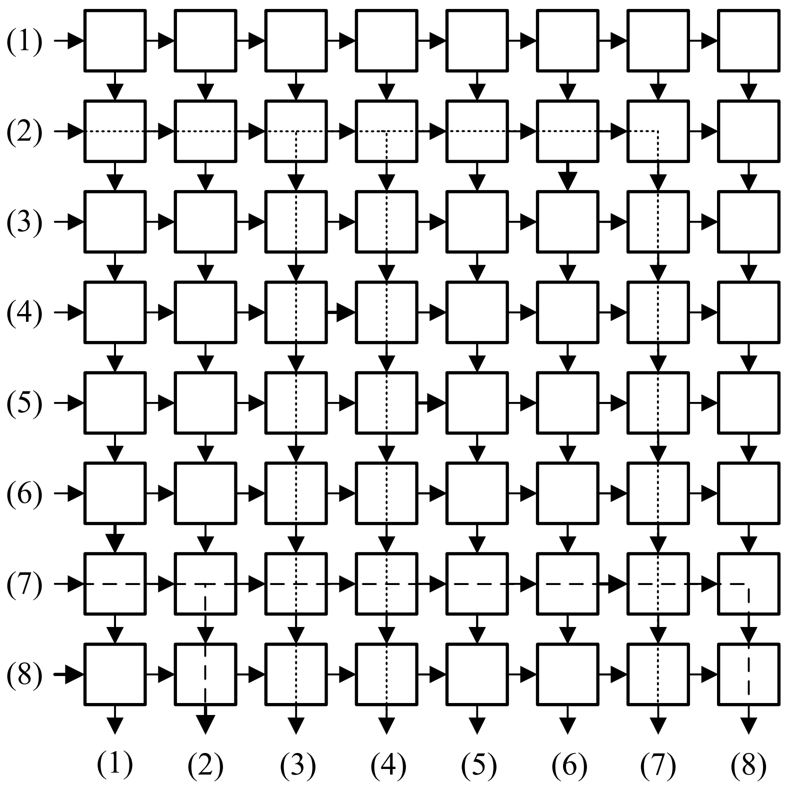
Stanja ukrsne tačke.

nazivano *cross*), dok se ukrsna tačka (i, j) postavlja u stanje „T” (umesto koga se ranije koristilo stanje *bar*).

Na slici 12.1, veze između zadatih parova portova označene su isprekidanim linijama.

Zadatak 12.2 Objasnite na koji se način u crossbar komutatoru dimenzija 8×8 uspostavljaju veze (2, 3, 4, 7) i (7, 2, 8).

Uvođenjem stanja „T”, omogućeno je kopiranje u crossbar mreži, tj. istovremeno prosleđivanje paketa s jednog ulaznog porta na više izlaznih.



Slika 12.2: *Kopiranje u crossbar komutatoru.*

Ako je paket potrebno kopirati s ulaznog porta i do izlaznih portova $j_1,\ j_2, \ldots, j_k$, tada se ukrsne tačke na položajima $(i, j_1), (i, j_2), \ldots, (i, j_k)$ postavljaju u stanje „T”, dok se ostale ukrsne tačke na putanjama između pojedinih parova ulazno-izlaznih portova postavljaju u stanje „X”. Na slici 12.2, to je ilustrovano za portove iz postavke zadatka.

Zadatak 12.3 Verovatnoća dolaska paketa na ulazni port crossbar komutatora dimenzija $N \times N$ tokom intervala jednakog prosečnom trajanju paketa je a. Izvedite izraze za verovatnoću prihvatanja paketa i prosečno kašnjenje u komutatoru. Posebno razmotrite sledeće asimptotske slučajeve:

- $a \ll 1$ (lako opterećenje),
- $N \to \infty$ (velika mreža) i
- $N \to \infty,\ a = 1$ (velika mreža, puno opterećenje).

Svi izlazni portovi komutatora su ravnopravni. Ako je verovatnoća dolaska paketa na neki od ulaznih portova a, tada je verovatnoća združenog događaja da na ulazni port dolazi paket koji je namenjen uočenom izlaznom portu

$$a' = \frac{a}{N}.$$

Verovatnoća da tokom intervala posmatranja na i ulaznih portova dolaze paketi namenjeni uočenom izlaznom portu data je binomnom raspodelom,

$$\mathrm{P}(i) = \binom{N}{i} \left(\frac{a}{N}\right)^i \left(1 - \frac{a}{N}\right)^{N-i}.$$

Očekivanje ove slučajne promenljive predstavlja ulazni saobraćaj koji je u intervalu posmatranja namenjen uočenom izlaznom portu:

$$N_a^{(in)} = \sum_{i=0}^{N} i\,\mathrm{P}(i) = a.$$

Propusnost izlaznog porta jednaka je saobraćaju kroz njega i data je izrazom

$$Th = N_a^{(out)} = \sum_{i=1}^{N} \mathrm{P}(i) = 1 - \mathrm{P}(0).$$

Nakon uvrštavanja izraza za $\mathrm{P}(0)$, dobijamo

$$Th = N_a^{(out)} = 1 - \left(1 - \frac{a}{N}\right)^N.$$

U slučaju slabog opterećenja je $Th \approx a$, pa se skoro svi paketi prosleđuju do željenog izlaza. Za velike mreže se dobija

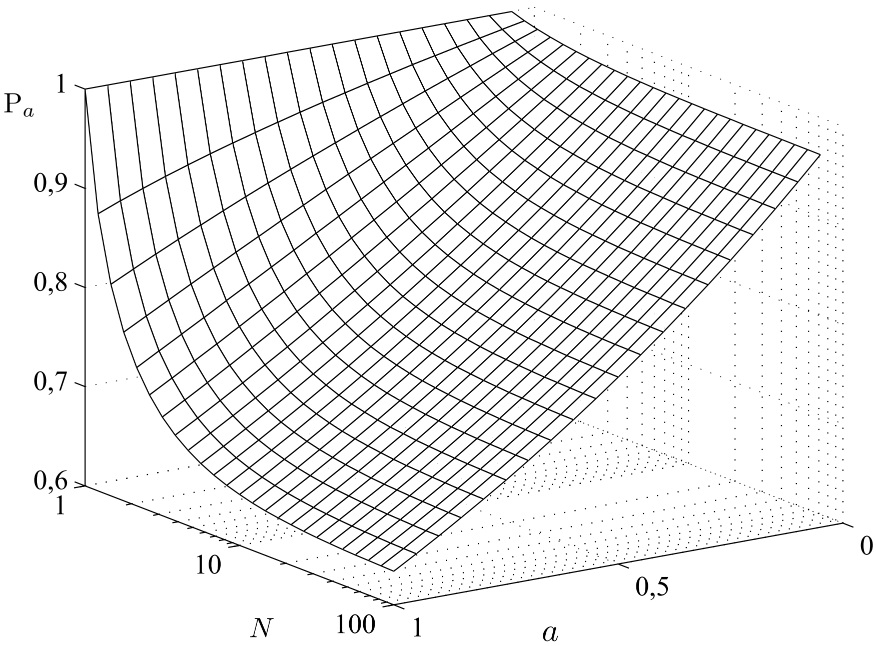
$$Th = \lim_{N\to\infty} \left(1 - \left(1 - \frac{a}{N}\right)^N\right) = 1 - e^{-a}.$$

Ako je ovakva mreža slabo opterećena, ponovo će biti $Th \approx a$. U slučaju punog opterećenja je $Th_{max} = 63{,}21\%$.

Verovatnoća prihvatanja paketa predstavlja odnos saobraćaja na izlazu i ulazu:

$$\mathrm{P}_a = \frac{N_a^{(out)}}{N_a^{(in)}} = \frac{1}{a}\left(1 - \left(1 - \frac{a}{N}\right)^N\right).$$

Ova zavisnost ilustrovana je na slici.



Verovatnoća prihvatanja paketa.

Ako je opterećenje slabo, biće $\mathrm{P}_a \approx 1$, što ukazuje na veliku efikasnost komutatora. Ako je mreža velika, biće

$$\lim_{N\to\infty} \mathrm{P}_a = \frac{1 - e^{-a}}{a}.$$

Minimalna verovatnoća prihvatanja javlja se kada je opterećenje maksimalno:

$$(\mathrm{P}_a)_{min} = 1 - e^{-1} = 63{,}21\%.$$

Vidimo da crossbar komutator zadržava dobre performanse i pod najnepovoljnijim uslovima.

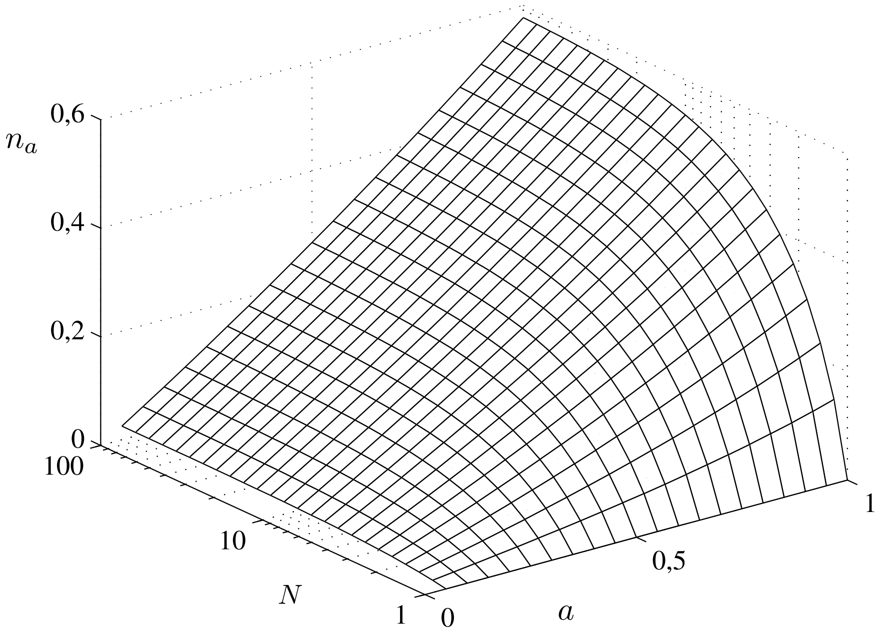
Kašnjenje kroz komutator jednako je prosečnom broju pokušaja da se dosegne željeni izlazni port. Verovatnoća da će paket uspeti da dođe do izlaza nakon k pokušaja data je geometrijskom raspodelom,

$$p_k = \mathrm{P}_a(1 - \mathrm{P}_a)^k.$$

Prosečno kašnjenje je

$$n_a = \sum_{k=0}^{\infty} k\, p_k = \frac{1 - \mathrm{P}_a}{\mathrm{P}_a}.$$

Ova zavisnost prikazana je na donjem grafiku. U uslovima slabog opterećenja, kašnjenje je približno jednako nuli, što znači da paketi do željenog izlaza dolaze iz prvog pokušaja. Kod velikih komutatora koji rade pod punim opterećenjem, prosečno kašnjenje iznosi 0,582 trajanja intervala posmatranja.



Kašnjenje u crossbar komutatoru.

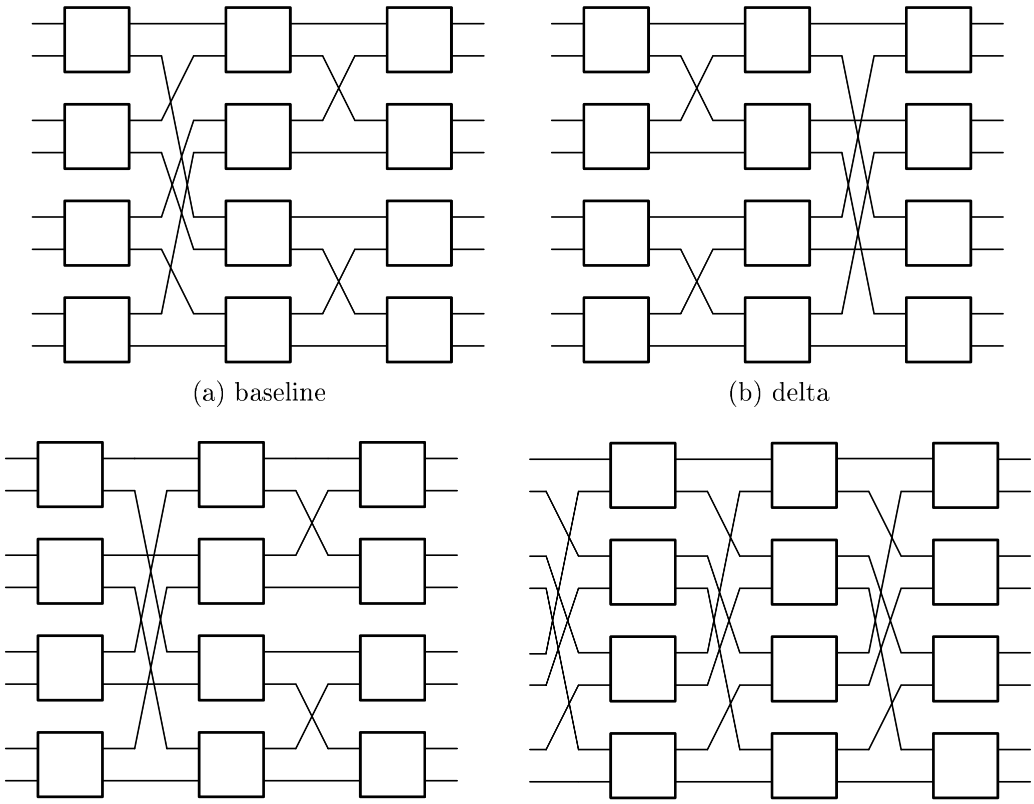
Zadatak 12.4 Nacrtajte sheme „banyan" komutacionih mreža s 8 ulaznih i 8 izlaznih portova, koje su izgrađene od komutacionih elemenata dimenzija 2×2.

Za razliku od crossbar komutatora, koji se baziraju na matričnoj topologiji, banyan mreže se baziraju na topologiji stabla. Postoje različiti načini za formiranje binarnih stabala, pa postoje i različite topologije banyan mreža. Na slici 12.4, prikazane su četiri banyan mreže s topologijama *baseline*, *delta*, *modifikovani manipulator podacima* i *omega*.

Osnovne odlike banyan mreže dimenzija $N \times N$, koja je izgrađena od komutacionih elemenata dimenzija $n \times n$ su:

1. svaki komutacioni element je crossbar komutator dimenzija $n \times n$,
2. banyan mreža se sastoji od $\log_n N$ kaskada i N/n redova,
3. ukupan broj ukrsnih tačaka u banyan mreži je $nN \log_n N$.

Primetimo da se banyan mreže odlikuju izuzetnom proširivošću. Njihova modularna struktura dozvoljava konstrukciju većih mreža od manjih, bez potrebe modifikacije algoritama koji se koriste u radu mreže. Zbog svoje regularnosti i oblika međupovezivanja, banyan mreže su naročito pogodne za realizaciju u tehnici integrisanih kola s visokim stepenom integracije (VLSI).



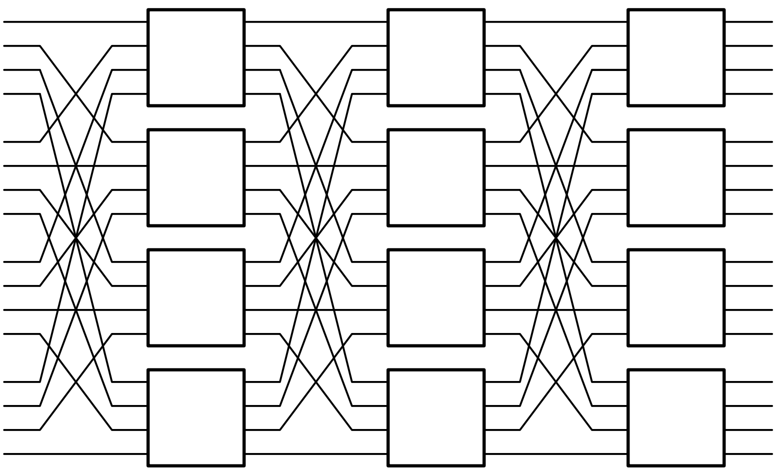
(a) baseline (b) delta

(c) modifikovani manipulator podacima (d) omega

Slika 12.4: *Arhitekture banyan mreža dimenzija* 8×8.

Zadatak 12.5 Korišćenjem crossbar komutatora 4×4, realizujte banyan mrežu dimenzija 16×16.

Tražena shema prikazana je na slici. Primenjeno je omega povezivanje.



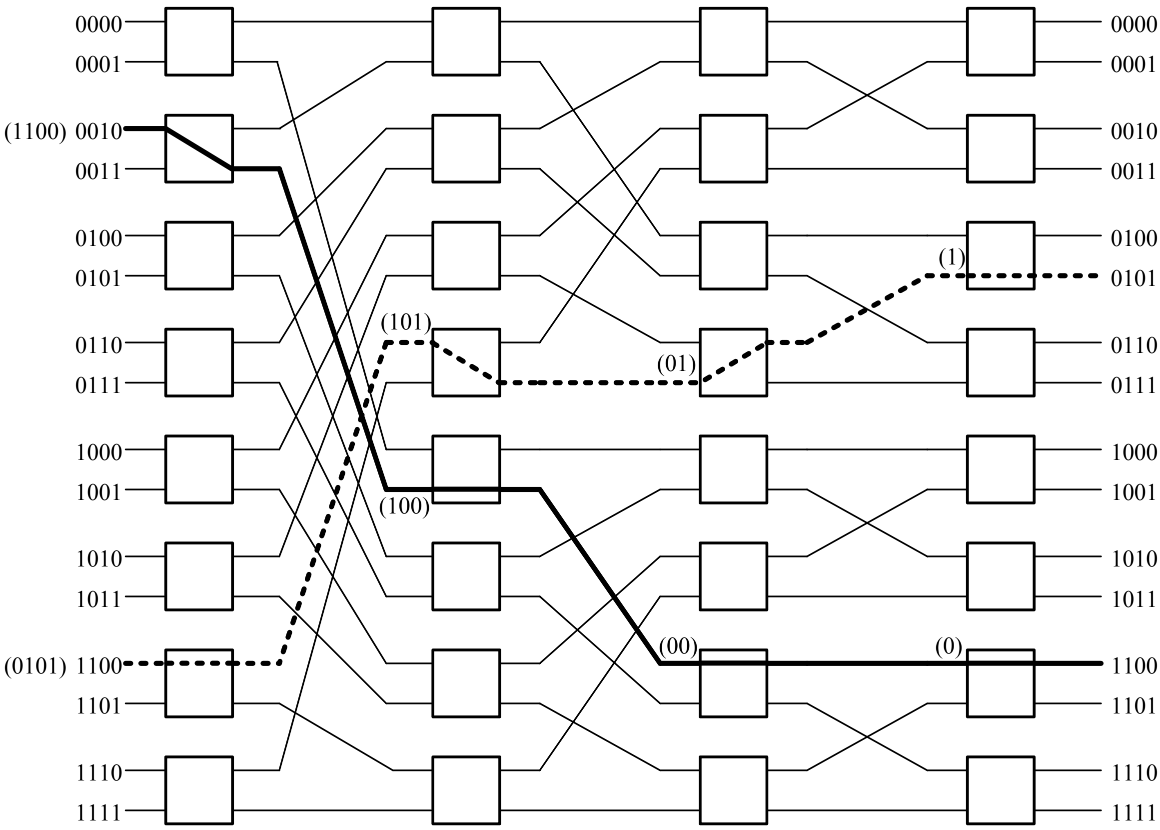
Slika 12.5: *Banyan mreža dimenzija* 16×16 *realizovana korišćenjem blokova* 4×4.

Zadatak 12.6 Na ulazne portove banyan baseline mreže dimenzija 16×16, čije su adrese 0010 i 1100, istovremeno dolaze dva paketa, koja treba proslediti na izlazne portove 1100 i 0101, respektivno. Odredite putanje ovih paketa i objasnite princip samoprosleđivanja paketa u banyan komutatoru.

Paketi se prosleđuje kroz banyan komutacionu mrežu na osnovu adrese željenog izlaznog porta. Ova adresa pridružuje se zaglavlju paketa kao deo etikete za prosleđivanje. Bez obzira na to na kom se ulazu komutatora nalazi posmatrani paket, uvek će postojati *samo jedna* putanja do željenog izlaza.

Uzastopni biti adrese izlaznog porta interpretiraju se od stepena do stepena komutatora, tako da se na i-tom stepenu posmatra i-ti bit adrese željenog izlaznog porta. Ako je taj bit vrednosti 0, paket će se proslediti na gornji izlaz tekućeg komutacionog elementa 2×2, a ako je vrednost ovoga bita 1, paket će se proslediti na donji izlaz komutacionog elementa.

Primer samoprosleđivanja paketa za zadate ulazne i izlazne adrese dat je na slici. Čitaocu se preporučuje da zadatak reši i za ostale arhitekture banyan komutacionih mreža.



Slika 12.6: *Samoprosleđivanje paketa u banyan baseline mreži dimenzija* 16×16.

Zadatak 12.7 Verovatnoća dolaska paketa na ulazni port banyan komutatora dimenzija $N \times N$, koji je izgrađen od blokova dimenzija 2×2 je a. Izvedite izraze za verovatnoću prihvatanja paketa i prosečno kašnjenje u komutatoru.

Gradivni elementi banyan mreže su crossbar komutatori dimenzija 2×2, pa ćemo iskoristiti rezultate koji važe za njih.

Broj kaskada posmatranog banyan komutatora je

$$k = \log_2 N.$$

Na elementima prve kaskade, ulazni saobraćaj je

$$a_1^{(in)} = a,$$

a izlazni

$$a_1^{(out)} = 1 - \left(1 - \frac{a}{2}\right)^2.$$

Za elemente i-te kaskade, $i = 2, \ldots, k$, važiće

$$a_i^{(in)} = a_{i-1}^{(out)},$$

$$a_i^{(out)} = 1 - \left(1 - \frac{a_i^{(in)}}{2}\right)^2.$$

Ulazni saobraćaj u mrežu je

$$N_a^{(in)} = a,$$

a izlazni

$$N_a^{(out)} = a_k^{(out)}.$$

Izlazni saobraćaj i ovde predstavlja propusnost mreže. Primetimo da, zbog nelinearne veze ulaznog i izlaznog saobraćaja pojedinačnih komutacionih elemenata, izraz za propusnost banyan komutatora nije moguće dati u zatvorenoj formi, već se za konkretan slučaj određuje iterativno.

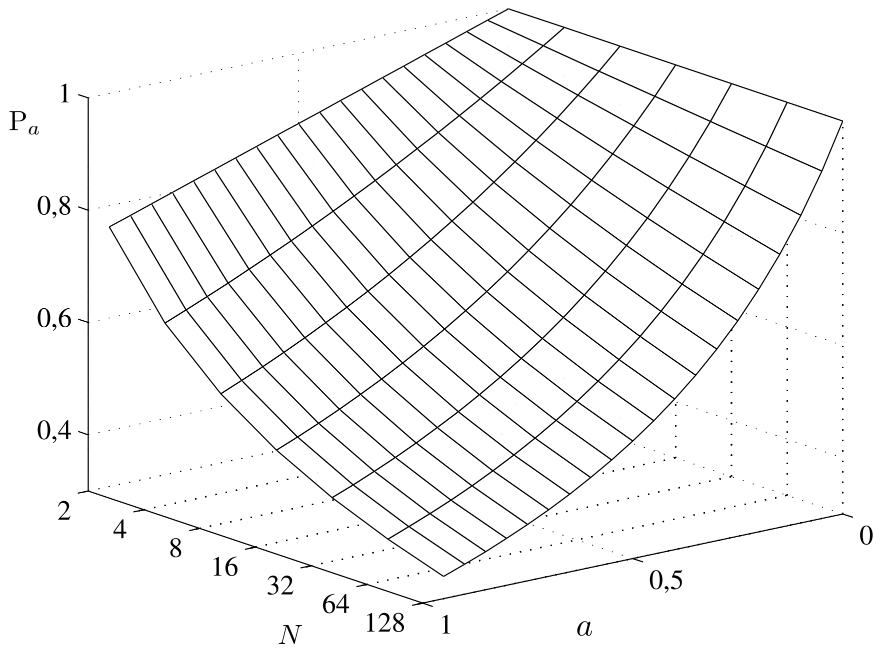
Verovatnoća prihvatanja paketa je

$$\mathrm{P}_a = \frac{N_a^{(out)}}{N_a^{(in)}}.$$

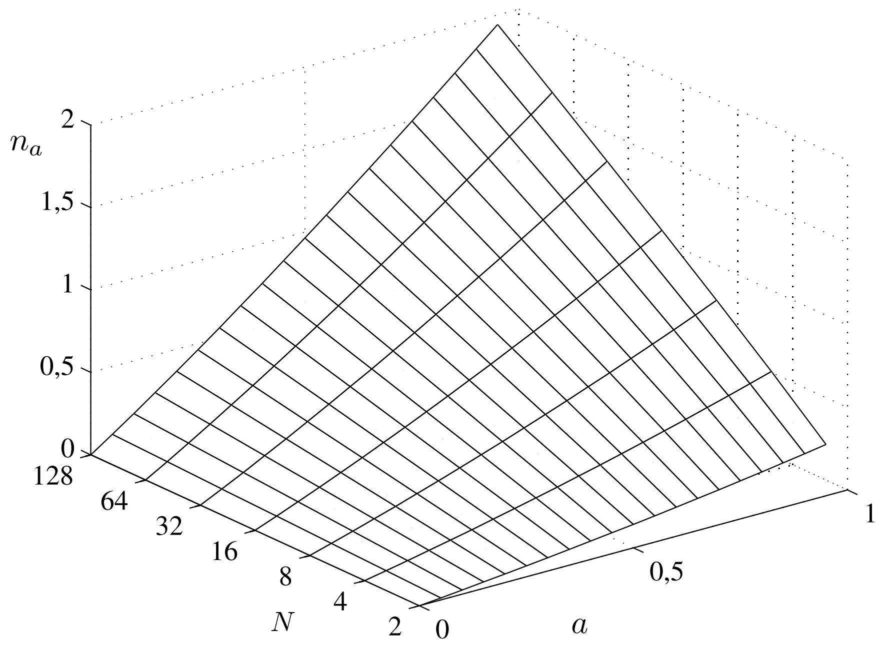
Kašnjenje kroz komutacionu mrežu jednako je prosečnom broju pokušaja da paket dosegne željeni izlazni port:

$$n_a = \sum_{i=0}^{\infty} i\,\mathrm{P}_a(1-\mathrm{P}_a)^i = \frac{1-\mathrm{P}_a}{\mathrm{P}_a}.$$

Grafici performansi banyan komutatora dati su na narednoj strani. Na gornjoj slici prikazana je zavisnost verovatnoće prihvatanja paketa, a na donjoj kašnjenja paketa u banyan komutatoru od broja ulaznih portova i saobraćaja na ulazu.



Verovatnoća prihvatanja paketa.



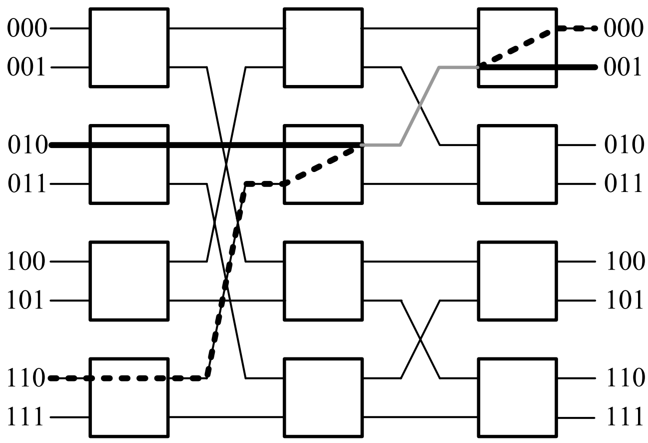
Kašnjenje u banyan mreži.

Šta možete zaključiti poređenjem s graficima koje smo dobili analizom performansi crossbar komutatora?

Zadatak 12.8 Ilustrujte pojavu interne blokade u banyan komutatoru.

Banyan mreže su interno blokirajuće. Do konflikta dolazi onda kada se paketi nadmeću za isti resurs, npr. za link, bafer itd. Ovakvi paketi se u opštem slučaju odbacuju.

Na slici je prikazan slučaj kada pakete s ulaznih portova 010 i 110 treba proslediti na izlaze 001 i 000, respektivno. Do konflikta dolazi na drugoj kaskadi komutatora, gde oba paketa traže isti link ka trećoj kaskadi.



Slika 12.8: *Konflikt u banyan komutatoru.*

U narednom zadatku ćemo videti na koji se način može izbeći pojava interne blokade u banyan komutatoru.

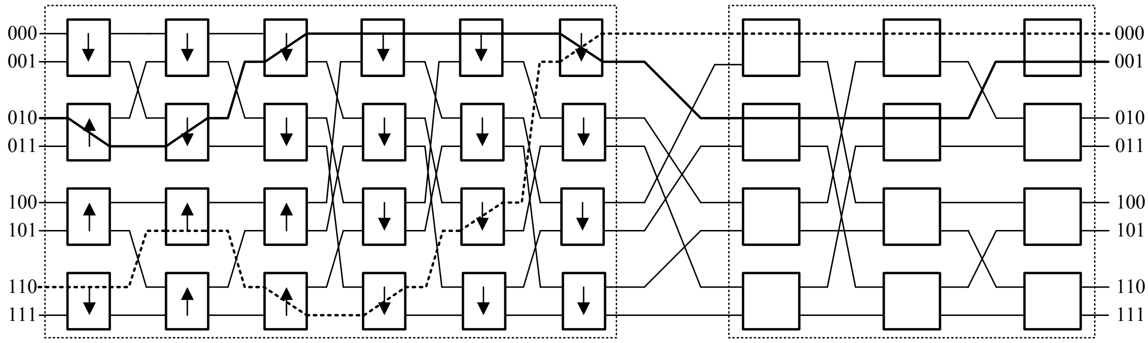
Zadatak 12.9 Nacrtajte shemu *Batcher-banyan* mreže i pokažite kako se u njoj otklanja pojava interne blokade.

Batcher mreža za sortiranje predstavlja način za prevazilaženje internog blokiranja u banyan mreži. Kombinacija Batcher-sortera, mreže za mešanje i banyan mreže predstavlja interno neblokirajuću mrežu.

Gradivni blokovi Batcher mreže su sorter naviše i sorter naniže, dimenzija 2×2. Prosleđivanje paketa s ulaza elementarnog sortera na njegov izlaz zavisi od prioriteta paketa. Ako paket dolazi samo na jedan ulaz elementarnog sortera, smatra se da je nižeg prioriteta i prosleđuje se suprotno smeru strelice na tom sorteru. Ako na oba ulaza sortera dolaze paketi, paket višeg prioriteta biće onaj koja ima veću adresu željenog izlaznog porta u banyan mreži. Ovakav paket se prosleđuje na izlaz sortera ka kome pokazuje strelica. Drugi paket je nižeg prioriteta i prosleđuje se suprotno pokazivanju strelice.

Na slici 12.9, prikazana je Batcher-banyan mreža sa osam portova, na kojoj je ilustrovan slučaj iz prethodnog zadatka. Paketi su se sada sortirali tako da više ne dolaze u konflikt.

Primetimo da je interna blokada u Batcher-banyan mreži otklonjena po cenu povećanja kašnjenja paketa.

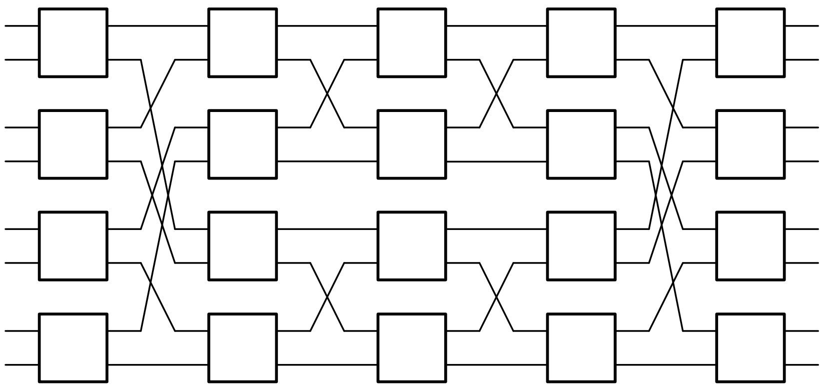


Slika 12.9: *Batcher-banyan mreža.*

Zadatak 12.10 Nacrtajte shemu Benešove mreže i objasnite kako se u njoj otklanja pojava interne blokade.

Benešova mreža dimenzija $N \times N$ sastoji se od dve banyan mreže dimenzija $N \times N$ koje su postavljene kao predmet i lik u ogledalu. Ove mreže imaju jednu zajedničku kaskadu, tako da je ukupan broj kaskada Benešove mreže $2\log_2 N - 1$.

Primer Benešove mreže dimenzija 8×8 dat je na slici.



Slika 12.10: *Benešova mreža.*

Zbog uvođenja dodatnih kaskada, u Benešovoj mreži za svaki par ulazno-izlaznih portova postoji više od jedne putanje. Alternativne putanje smanjuju verovatnoću nastupanja konflikta.

Čitaocu se prepušta da u Benešovoj mreži sa slike pronađe sve putanje za uočeni par ulazno-izlaznih portova.

13. Sigurnost u računarskim mrežama

Zadatak 13.1 Šifrirajte poruku M primenom algoritma RSA, ako je:

a) $p = 3$, $q = 11$, $d = 7$ i $M = 5$,

b) $p = 5$, $q = 11$, $e = 3$ i $M = 9$,

c) $p = 17$, $q = 31$, $e = 7$ i $M = 2$.

Algoritam RSA za šifriranje i dešifriranje poruke koristi javni i privatni ključ. Algoritam se izvršava na sledeći način:

1. Izaberu se dva prosta broja, p i q,
2. Izračuna se $n = pq$ i $\phi(n) = (p-1)(q-1)$,
3. Izabere se broj e, $1 < e < \phi(n)$, tako da e i $\phi(n)$ budu uzajamno prosti,
4. Odredi se broj $d < \phi(n)$, za koji važi $de \equiv 1 \pmod{\phi(n)}$,
5. Formiraju se javni ključ $PU = \{e, n\}$ i privatni ključ $PR = \{d, n\}$,
6. Poruka M se šifrira po obrascu $C \equiv M^e \pmod{n}$. Šifrirana poruka, C, naziva se *siferttekstom*.
7. Dešifriranje je opisano obrascem $M \equiv C^d \pmod{n}$.

a) Na osnovu datih podataka je $n = 33$ i $\phi(n) = 20$. Iz uslova $de \equiv 1 \pmod{\phi(n)}$ imamo

$$7e \equiv 1 \pmod{20},$$

pa jednostavno dobijamo $e = 3$. Privatni ključ je $\{7, 33\}$, a javni $\{3, 33\}$.

Sifertekst je

$$C \equiv M^e \pmod{n},$$

odnosno

$$C \equiv 5^3 \pmod{33}.$$

Kako je $5^3 = 125 \equiv 26 \pmod{33}$, zaključujemo da je traženi sifertekst $C = 26$.

b) Sada je $n = 55$ i $\phi(n) = 40$. Poznato je $e = 3$, pa određujemo d tako da važi

$$3d \equiv 1 \pmod{40}.$$

Odavde je $d = 27$. Privatni ključ je $\{27,\ 55\}$, a javni $\{3,\ 55\}$.

Sifertekst je

$$C \equiv 9^3 \pmod{55}.$$

Pošto je $9^3 = 729$, biće $C = 14$.

c) U ovom slučaju je $n = 527$ i $\phi(n) = 480$. Parametar d iznosi 343. Pošto je $M^e < n$, biće $C = M^e$, pa je sifertekst 128.

Zadatak 13.2 Primenom algoritma RSA dešifrirajte poruku C, ako je

a) $C = 10$, $e = 5$, $n = 35$,

b) $C = 128$, $n = 527$, $d = 343$.

a) U ovom slučaju je poznat javni ključ, a privatni nije. Da bismo mogli dešifrirati poruku, potrebno je da odredimo i vrednost privatnog ključa.

Pošto je vrednost n mala, možemo probati da je faktorizujemo preko prostih brojeva. U ovome primeru, jednostavno dobijamo $n = 5 \cdot 7$. Sada je $\phi(n) = 24$, pa ostaje još da odredimo vrednost d, iz uslova

$$5d \equiv 1 \pmod{24}.$$

Odavde dobijamo $d = 5$, pa je privatni ključ $\{5,\ 35\}$.

Originalnu poruku dobijamo iz

$$M \equiv 10^5 \pmod{35},$$

što daje $M = 5$.

U praktičnim primenama se brojevi p i q biraju tako da budu veliki, pa „razbijanje” privatnog ključa nije trivijalno.

b) Sada je poznat privatni ključ. Originalnu poruku ćemo odrediti tako da važi

$$M \equiv 128^{343} \pmod{527}.$$

Broj 128^{343} je prevelik da bismo ga izračunali. Da bismo rešili ovaj problem, pozvaćemo se na sledeću osobinu kongruencija: ako je $x \equiv y \pmod{m}$ i $u \equiv v \pmod{m}$, tada je i $xu \equiv yv \pmod{m}$.

Napišimo broj 128^{343} u obliku

$$128^{343} = \left(128^4\right)^{85} \cdot 128^3.$$

Pošto je

$$128^4 \equiv 101 \pmod{527},$$

biće

$$\left(128^4\right)^{85} \equiv 101^{85} \pmod{527}.$$

Pažljivim ponavljanjem ovoga postupka, dobijamo

$$\left(128^4\right)^{85} \equiv 373 \pmod{527}.$$

Pošto je i

$$128^3 \equiv 219 \pmod{527},$$

konačno će biti

$$128^{343} \equiv 2 \pmod{527},$$

pa je $M = 2$.

Primetimo da je ovo inverzan problem onom iz tačke c) zadatka 13.1. Dok je šifriranje originalne poruke bilo elementarno, dešifriranje siferteksta bilo je znatno teže.

Zadatak 13.3 Dva učesnika u komunikaciji koriste Diffie-Hellman algoritam kriptozaštite s javnim ključem.

a) Pokažite da oba učesnika imaju isti simetrični ključ.

b) Za $p = 11$, $g = 2$ i privatne ključeve učesnika $S_a = 5$ i $S_b = 12$, odredite njihove javne ključeve i zajednički simetrični ključ.

c) Pokažite kako zlonamerni posrednik može izvršiti napad na ovaj algoritam.

Algoritam Diffie-Hellmann opisan je narednom tabelom.

Algoritam Diffie-Hellmann.

	Učesnik A	Učesnik B
Tajni ključ	S_a	S_b
Javni ključ	$T_a \equiv g^{S_a} \pmod{p}$	$T_b \equiv g^{S_b} \pmod{p}$
Zajednički ključ	$S \equiv T_b^{S_a} \pmod{p}$	$S' \equiv T_a^{S_b} \pmod{p}$

a) Iz definicije parametara u algoritmu i na osnovu osobina kongruencija je

$$
\begin{aligned}
S &\equiv T_b^{S_a} \pmod{p} \\
&\equiv \left(g^{S_b}\right)^{S_a} \pmod{p} \\
&\equiv \left(g^{S_a}\right)^{S_b} \pmod{p} \\
&\equiv T_a^{S_b} \pmod{p} \\
&\equiv S' \pmod{p},
\end{aligned}
$$

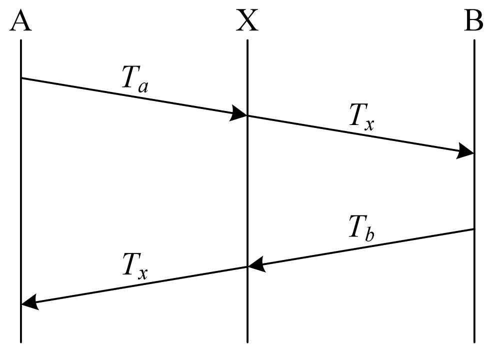
pa su ključevi S i S' jednaki.

b) Postupak izračunavanja dat je u tabeli.

Vrednosti parametara.

	Učesnik A	Učesnik B
Tajni ključ	$S_a = 5$	$S_b = 12$
Javni ključ	$T_a \equiv 2^5 \pmod{11}$; $T_a = 10$	$T_b \equiv 2^{12} \pmod{11}$; $T_b = 4$
Zajednički ključ	$S \equiv 4^5 \pmod{11}$; $S = 1$	$S' \equiv 10^{12} \pmod{11}$; $S' = 1$

c) Algoritam je podložan napadu zlonamernog posrednika (*man-in-the-middle*). Prema scenariju sa slike 13.3, posrednik X prima javni ključ prvog učesnika, T_a i zadržava ga, a drugom učesniku šalje svoj javni ključ, T_x. Ne znajući za to, drugi učesnik odgovara svojim javnim ključem (T_b), koji posrednik takođe zadržava, a umesto njega prvom učesniku prosleđuje svoj ključ (T_x). Sada su se prvi učesnik i posrednik dogovorili oko zajedničkog ključa (S_{ax}), a posrednik i drugi učesnik oko drugog zajedničkog ključa (S_{xb}), pa posrednik može presretati komunikaciju učesnika A i B.



Slika 13.3: *Napad posrednika na algoritam Diffie-Hellman.*

Zadatak 13.4 Rezultat izvršavanja *hash* funkcije je dužine 512 b. Kolike su njena otpornost na sudar i otpornost na razbijanje?

Hash funkcija čiji je rezultat dužine L ima otpornost na sudar $L/2$ i otpornost na razbijanje (nalaženje inverzije) L. Za funkciju iz zadatka, otpornost na sudar iznosi 256 b, dok je otpornost na razbijanje 512 b.

Zadatak 13.5 Kolika je sigurnosna jačina HMAC algoritma, ako se koristi:

a) algoritam SHA-256, uz ključ jačine 128 b,

b) algoritam SHA-256, uz ključ jačine 256 b,

c) algoritam SHA-1, uz ključ jačine 256 b?

Sigurnosna jačina HMAC algoritma data je izrazom $\min(K, 2L)$, gde je K jačina kriptografskog ključa i L dužina tzv. sažetka poruke. Parametri nekoliko SHA algoritama dati su u tabeli.

Algoritam	Dužina sažetka [b]
SHA-1	160
SHA-256	256
SHA-384	384
SHA-512	512
SHA-512/224	224
SHA-512/256	256

Na osnovu ovih podataka, jednostavno dobijamo da sigurnosna jačina postupka a) iznosi 128 b, dok je za postupke b) i c) 256 b.

Zadatak 13.6 Kolika je potrebna jačina ključa za:

a) HMAC algoritam jačine 256 b, koji koristi SHA-512/256,

b) HMAC algoritam jačine 256 b, koji koristi SHA-512/224,

c) HMAC algoritam jačine 512 b, koji koristi SHA-512/256,

d) KDF algoritam jačine 128 b, koji koristi SHA-512/256,

e) KDF algoritam jačine 256 b, koji koristi SHA-256?

a) Prema tabeli iz prethodnog zadatka, dužina sažetka primenjenog SHA algoritma iznosi 256 b, pa je 256 b $= \min(K,\ 2 \cdot 256\ \text{b})$. Odavde je $K = 256$ b.

b) Sada je $L = 224$ b, pa je ponovo $K = 256$ b.

c) $K \geqslant 512$ b.

d) Sigurnosna jačina KDF algoritma je $\min(K, L)$, pa je sada $K = 128$ b.

e) $K \geqslant 256$ b.

Zadatak 13.7 Kolika je dužina inicijalizacionog vektora, ako se u CBC režimu rada koristi algoritam:

a) AES-128,

b) AES-256,

c) 3DES, uz ključ dužine 112 b,

d) 3DES, uz ključ dužine 168 b?

Inicijalizacioni vektor (IV) u CBC režimu rada iste je dužine kao i blok izvornog teksta. Za algoritam AES, ova vrednost iznosi 128 b, a za algoritam 3DES 64 b.

Zadatak 13.8 Kolika je veličina dopune pri šifriranju datoteke veličine 4015 b u CBC režimu, ako se koristi algoritam

a) AES-128,

b) AES-256,

c) 3DES, uz ključ dužine 112 b,

d) 3DES, uz ključ dužine 168 b?

U CBC režimu šifriranja, datoteka se deli na blokove koji su u algoritmu AES veličine 128 b, a u algoritmu 3DES 64 b. Poslednji blok se po potrebi dopunjava nulama do ove vrednosti.

Kada se primenjuje algoritam AES, potreban broj blokova je $\left\lceil \frac{4015\ \text{b}}{128\ \text{b}} \right\rceil = 32$. Veličina poslednjeg bloka iznosila bi $4015\ \text{b} - 31 \cdot 128\ \text{b} = 47\ \text{b}$, pa se on dopunjava s 81 nulom.

Kada se primenjuje algoritam 3DES, potreban broj blokova je $\left\lceil \frac{4015\ \text{b}}{64\ \text{b}} \right\rceil = 63$. Veličina poslednjeg bloka ponovo bi iznosila 47 b, pa se on sada dopunjava sa 17 nula.

Zadatak 13.9 Server prima zahtev unutar UDP paketa i na njega odgovara takođe posredstvom UDP. Ako klijent s IP adresom X pogrešno oglasi adresu Y, gde će server poslati odgovor?

U posmatranom scenariju, server ne može znati je li IP adresa koju mu je dostavio klijent tačna ili ne. Jedini podatak o IP adresi klijenta kojim server raspolaže je Y, pa će stoga na tu (pogrešnu) adresu i poslati svoj odgovor.

Zadatak 13.10 Server prima TCP SYN poruku s IP adreseom X, da bi nakon odgovaranja SYNACK porukom dobio ACK s IP adresom X i ispravnim brojem potvrde. Pod pretpostavkom da server na slučajan način bira početni broj sekvence, kao i da u komunikaciji nema posrednika (*man-in-the-middle*), može li server biti siguran u to da je IP adresa klijenta X?

Server je siguran u to da je IP adresa klijenta X. Kada bi napadač lažno oglašavao adresu X, SYNACK bi se svejedno poslao ka X i TCP agent u njemu ne bi odgovorio ACK segmentom. Ako bi napadač ipak uspeo da pošalje pravovremeni ACK, ne bi mogao znati tačan broj sekvence.

Zadatak 13.11 Kako se može sprečiti napad na protokol rutiranja u kome napadač šalje lažne *link-state* poruke?

Link-state poruka sadrži liste neposrednih suseda rutera i metrike („cene") direktnih putanja do njih. Napad slanjem lažnih *link-state* poruka može se onemogućiti proverom integriteta poruke – po prijemu poruke od drugog rutera, posmatrani ruter treba da verifikuje pošiljaoca i da utvrdi je li iko menjao poruku u tranzitu.

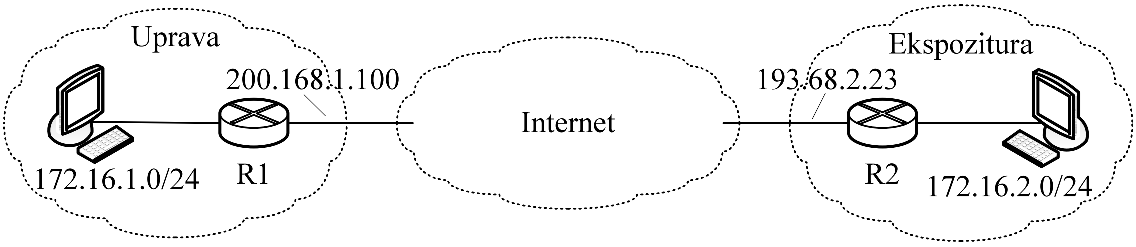
Zadatak 13.12 U BitTorrent protokolu za distribuiranje datoteka po principu P2P, izvor „razbija" datoteku u blokove, koje onda korisnici međusobno redistribuiraju. Bez mehanizama zaštite, napadač bi se mogao predstaviti kao dobronamerni korisnik i slati lažne blokove drugim korisnicima. Predložite mehanizam pomoću koga korisnici mogu verifikovati integritet bloka pre nego što ga proslede drugima.

Jednostavan mehanizam zaštite može se realizovati prema sledećem scenariju. Datoteka koja se distribuira deli se u blokove jednake veličine. Za svaki blok se izračuna *hash*, npr. metodom MD5 ili SHA-1; rezultati izračunavanja smeštaju se u datoteku `.torrent`, koja se onda posredstvom verodostojnog izvora distribuira novopridruženim korisnicima. Po preuzimanju bloka, korisnik će izračunati njegov *hash* i uporediti ga s vrednošću iz `.torrent` datoteke. Ako se ove dve vrednosti podudaraju, blok je valjan, dok je u suprotnom lažan i treba ga odbaciti.

Zadatak 13.13 Uprava i ekspozitura banke povezane su virtuelnom privatnom mrežom na bazi IPsec, prema slici.

a) Koje će se IP adrese izvorišta i odredišta i broj protokola označiti u IPsec datagramu kada host iz uprave bude slao podatke hostu u ekspozituri?

b) Hoće li se datagram koji host iz uprave bude poslao javnom web serveru zaštititi pomoću IPsec?



Slika 13.13: Virtuelna privatna mreža.

a) Između rutera R1 i R2 formira se VPN tunel na bazi tzv. sigurnosne asocijacije. Ruter R1 će u zaglavlje IPsec datagrama upisati IP adresu svog interfejsa ka tunelu (200.168.1.100) kao izvorišnu, a adresu interfejsa rutera R2 ka tunelu (196.68.2.23) kao odredišnu. Kao broj protokola, upisaće se 50 (ESP).

b) Datagram ka javnom web serveru neće se štititi, osim ako to nije naznačeno u bazi pravila sigurnosti (SPD).

Zadatak 13.14 Na slici je dat izgled programskog prozora alata Wireshark. Posmatranjem detalja koji se odnose na paket broj 349, odgovorite na sledeća pitanja.

a) Ko šalje uočeni paket, klijent ili server? Koje su IP adrese i brojevi portova klijenta i servera?

b) Ako nema ni gubitaka okvira ni retransmisija, koji će biti broj sekvence narednog TCP segmenta koji šalje klijent?

c) Koliko TLS zapisa sadrži paket?

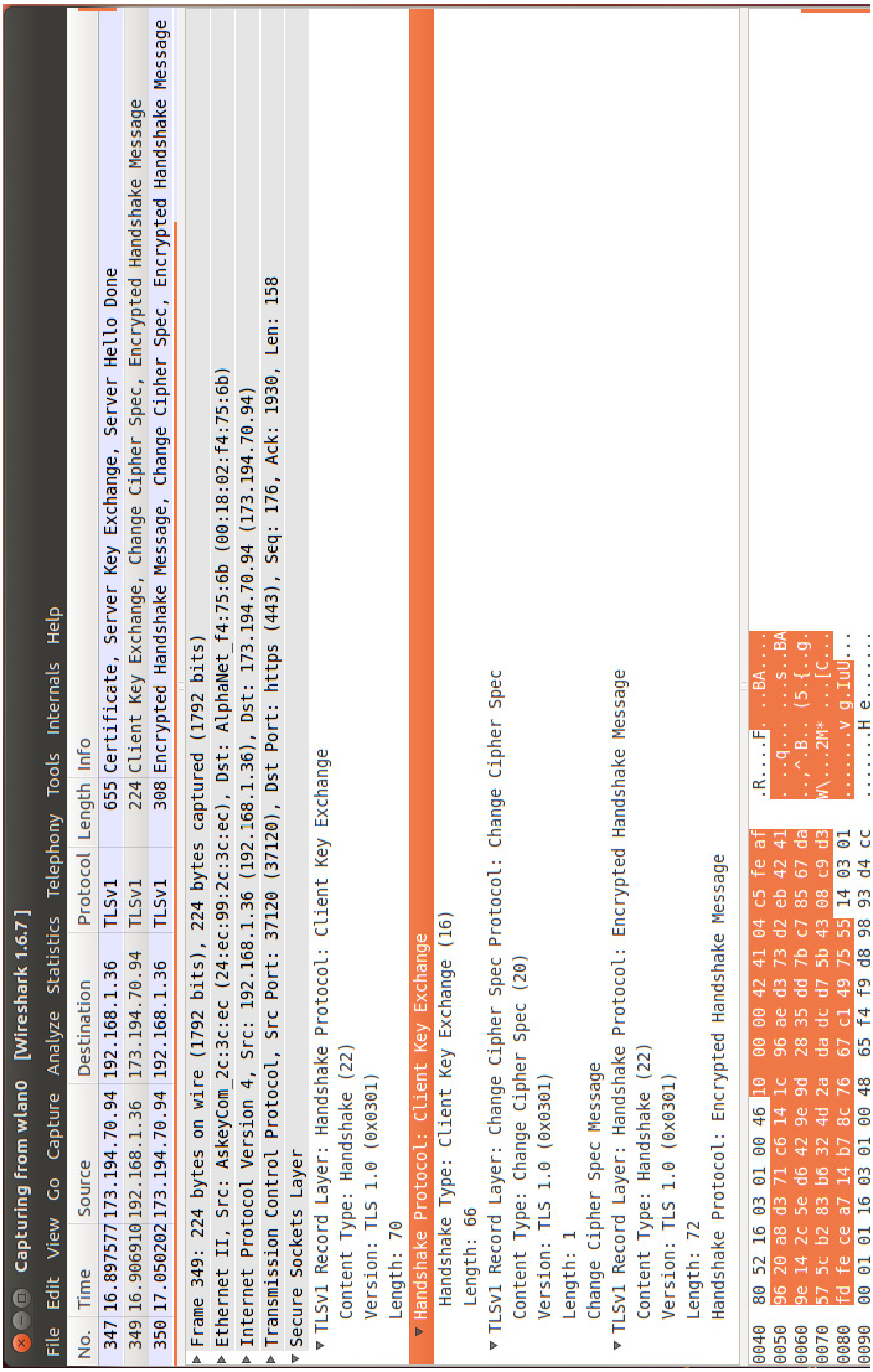
d) Koji je prvi, a koji poslednji bajt kriptografskog ključa koji se šalje ovim paketom?

a) Posmatrani okvir šalje klijent, s IP adrese 192.168.1.36 i porta 37120. IP adresa servera je 173.194.701.94, dok je broj porta 443 (https).

b) U enkapsuliranom TCP segmentu, vrednost polja broja sekvence (`Seq`) je 176, a on nosi 158 B podataka (`Len`). Ako ne bude ni gubitaka ni retransmisija, naredni broj sekvence će biti $176 + 158 = 334$.

c) Okvir sadrži tri TLS zapisa, koji se odnose na klijentov kriptografski ključ, početak primene mehanizma kriptozaštite, kao i šifriranu poruku „rukovanja".

d) Dužina označenog sadržaja u prvom TLS zapisu, u kome se nalazi klijentov ključ, iznosi 66 B. Brojanjem unazad, od kraja označenog sadržaja, dobijamo da je prvi bajt ključa 41, a poslednji 55.



Slika 13.14: Sadržaj Ethernet paketa dobijen analizom u programu Wireshark.

Zadatak 13.15 Posmatra se sledeći pseudo-WEP protokol. Zajednički tajni ključ je veličine 4 b, dok je inicijalizacioni vektor (IV) veličine 2 b. Pri generisanju kriptografskog ključa, IV se nadovezuje na kraj zajedničkog ključa. Kriptografski ključevi za četiri moguće vrednosti IV su:

```
101000: 00101011010101010010110101001000...
101001: 10100110110010101101001001011010...
101010: 00011010001111000101001010011111...
101011: 11111010000000001010101000101111...
```

Poruke su dužine 8 b. Vektor provere integriteta (ICV) je veličine 4 b, a računa se primenom ekskluzivne disjunkcije (XOR) prvih četiriju bita s poslednjim četirima bitima podataka. Pseudo-WEP paket se sastoji od triju polja: inicijalizacionog vektora, šifrirane poruke i šifrirane provere integriteta.

a) Za IV = `11`, koji pseudo-WEP paket odgovara poruci `10100000`?

b) Pokažite kako se iz primljenog paketa dešifruju poruka i ICV.

c) Pretpostavimo da napadač u presretnutom paketu komplementira prvi ICV bit. Šta još treba da uradi, da bi izmenjeni paket prošao proveru integriteta na prijemu?

a) Kada se zadati IV nadoveže na zajednički tajni ključ, dobija se `101011`, pa kriptografski ključ odgovara četvrtoj kombinaciji (`11111010000000001010101000101111...`)

ICV dobijamo primenom XOR operacije na blokove bita poruke: $\texttt{1010} \oplus \texttt{0000} = \texttt{1010}$.

Šifriranu poruku dobijamo izvršavanjem XOR operacije nad originalnom porukom i prvih osam bita kriptografskog ključa: $\texttt{10100000} \oplus \texttt{11111010} = \texttt{01011010}$.

Šifrirani ICV dobijamo tako što izvršimo operaciju XOR nad originalnim ICV i narednim četirima bitima kriptografskog ključa: $\texttt{1010} \oplus \texttt{0000} = \texttt{1010}$.

Sadržaj paketa je stoga `11010110101010`.

b) Prijemnik iz primljenog paketa izdvaja IV (`11`) i na osnovu njega generiše kriptografski ključ `11111010000000001010101000101111...`

Originalna poruka se dobija primenom operacije XOR nad šifriranom porukom i prvih 8 b ključa: $\texttt{01011010} \oplus \texttt{11111010} = \texttt{10100000}$.

Originalni ICV se dobija tako što se operacija XOR izvrši nad šifriranim ICV i narednim četirima bitima ključa: $\texttt{1010} \oplus \texttt{0000} = \texttt{1010}$.

Konačno, da bi izvršio proveru integriteta, prijemnik računa ICV iz dešifrirane poruke i upoređuje dobijene vrednosti: $\texttt{1010} \oplus \texttt{0000} = \texttt{1010}$. Vrednosti se poklapaju, pa je integritet poruke potvrđen.

c) Pošto se ICV računa primenom ekskluzivne disjunkcije nad blokovima od po 4 b poruke, napadač pored invertovanja prvog treba invertovati i peti bit poruke.

Zadatak 13.16 Mreža neke kompanije, koja koristi opseg adresa 222.22.0.0/16, štiti se *firewallom*. Predložite tabelu filtriranja koja dopušta sledeće:

- dozovljava se da unutrašnji korisnici uspostavljaju Telnet sesije sa spoljašnjim,
- dozvoljava se da spoljašnji korisnici pristupaju kompanijskom internet sajtu, čija je adresa 222.22.0.12,

dok se sav ostali dolazni i odlazni saobraćaj blokira. Kako bi se moglo unaprediti predloženo rešenje?

Tabela filtriranja koja zadovoljava postavku zadatka ima sledeći oblik.

Tabela filtriranja.

Akcija	IP adresa		Protokol	Port		Fleg
	Izvor	Odredište		Izvor	Odredište	
Dozvoli	u	van	TCP	> 1023	23	—
Dozvoli	van	u	TCP	23	> 1023	ACK
Dozvoli	van	222.22.0.12	TCP	> 1023	80	—
Dozvoli	222.22.0.12	van	TCP	80	> 1023	—
Zabrani	sve ostale	sve ostale	svi	svi	svi	svi

u – unutar opsega 222.22.0.0/16
van – van opsega 222.22.0.0/16

Prvi red dozvoljava odlazni Telnet saobraćaj (TCP port 23), dok drugi red dozvoljava dolazni Telnet saobraćaj, pri čemu je potrebno da ACK fleg u zaglavlju TCP segmenta ima vrednost 1. Treći i četvrti red dozvoljavaju HTTP saobraćaj (TCP port 80) s kompanijskim internet sajtom. Sav ostali saobraćaj se blokira.

Predloženo rešenje moglo bi se unaprediti kroz praćenje aktivnih TCP konekcija. Uobičajeno je da se podaci o ovim konekcijama čuvaju u tabeli i onda koriste za proveru dolaznog saobraćaja. U primeru iz zadatka, *firewall* bi npr. dozvolio ulazak u kompanijsku mrežu IP datagramima čiji je enkapsulirani protokol TCP, broj izvorišnog porta 23 i vrednost ACK flega 1 samo ukoliko u tabeli postoji zapis kojim se potvrđuje da neki od hostova iz štićene mreže održava konekciju s hostom koji šalje taj datagram.

Zadatak 13.17 Dat je početak liste za kontrolu pristupa na *firewall* uređaju:

```
interface Ethernet 0/1
ip address 131.204.1.1 255.255.255.0
ip access-group 101 in
```

Dopunite ovu listu tako da se:

a) dozvoli da poruka elektronske pošte koja dolazi van mreže 131.204.0.0/16 dođe na server 131.204.128.3,

b) dozvoli korisnicima iz mreže 131.204.0.0/16 da koriste IMAP za čitanje elektronske pošte na serveru 131.204.128.3,

c) dozvoli hostovima u 131.204.0.0/16 da koriste mail agent za slanje pošte na server 131.204.128.3,

d) zabrani hostovima u 131.204.0.0/16, sem 131.204.128.3, da isporučuju poštu vanjskim serverima.

a) Vanjskim hostovima treba dozvoliti da uspostavljaju TCP konekcije preko porta 25 (SMTP) servera 131.204.128.3:

```
access-list 101 permit tcp any ge 1024 131.204.128.3 eq 25
```

b) Sada treba dozvoliti TCP konekcije preko IMAP porta 143:

```
access-list 101 permit tcp host 131.204.0.0/16 ge 1024 131.204.128.3 eq 143
```

c) Ovde ponovo treba dozvoliti SMTP:

```
access-list 101 permit tcp host 131.204.0.0/16 ge 1024 131.204.128.3 eq 25
```

d) Pravilo filtriranja sada je:

```
access-list 101 permit tcp host 131.204.128.3 any eq 25
access-list 101 deny tcp host 131.204.0.0/16 any eq 25
```

Druga linija se može izostaviti, jer je redundantna.

Prilozi

Sume nekih redova

$$\sum_{i=0}^{n-1}(a+ib) = \frac{n}{2}\,(2a+(n-1)b)$$

$$\sum_{i=1}^{n} i^2 = \frac{n}{6}\,(n+1)(2n+1)$$

$$\sum_{i=0}^{n-1} ar^i = a\,\frac{1-r^n}{1-r}, \quad r \neq 1$$

$$\sum_{i=0}^{\infty} ar^i = \frac{a}{1-r}, \quad -1 < r < 1$$

$$\sum_{i=0}^{n-1}(a+ib)r^i = a\,\frac{1-r^n}{1-r} + rb\,\frac{1-nr^{n-1}+(n-1)r^n}{(1-r)^2}, \quad r \neq 1$$

$$\sum_{i=0}^{\infty}(a+ib)r^i = \frac{a}{1-r} + \frac{rb}{(1-r)^2}, \quad -1 < r < 1$$

$$\sum_{i=0}^{\infty} i^2 r^i = \frac{r(1+r)}{(1-r)^3}, \quad -1 < r < 1$$

$$\sum_{i=0}^{\infty}(i+1)\frac{a^i}{i!} = e^a(a+1)$$

Erlangova B formula

Verovatnoća blokade servisnog sistema M/M/m/m data je *Erlangovom B formulom*:

$$p_B = p_m = \frac{\dfrac{A^m}{m!}}{\displaystyle\sum_{i=0}^{m} \dfrac{A^i}{i!}},$$

gde je $A = \lambda/\mu$ intenzitet saobraćaja. Verovatnoća blokade se naziva i nivoom servisa (engl. *grade of service*).

U narednoj tabeli, date su vrednosti intenziteta saobraćaja (u erlanzima), za neke vrednosti broja servera, m i nivoa servisa, p_B.

m	Intenzitet saobraćaja A [E]			
	$p_B = 0{,}001$	$p_B = 0{,}002$	$p_B = 0{,}005$	$p_B = 0{,}01$
10	3,09	3,43	3,96	4,46
15	6,08	6,58	7,38	8,11
20	9,41	10,07	11,09	12,03
25	12,97	13,76	15,00	16,12
30	16,68	17,61	19,03	20,34
35	20,52	21,56	23,17	24,64
40	24,44	25,60	27,38	29,01
45	28,45	29,71	31,66	33,43
50	32,51	33,88	35,98	37,90
55	36,63	38,09	40,35	42,41
60	40,79	42,35	44,76	46,95
65	45,00	46,65	49,20	51,52
70	49,24	50,98	53,66	56,11
75	53,51	55,34	58,15	60,73
80	57,81	59,72	62,67	65,36
85	62,14	64,13	67,20	70,02
90	66,48	68,56	71,76	74,68
95	70,85	73,00	76,32	79,37
100	75,24	77,47	80,91	84,06
110	84,07	86,45	90,12	93,49
120	92,96	95,48	99,38	102,96
130	101,91	104,57	108,68	112,47
140	110,90	113,70	118,02	122,01
150	119,94	122,86	127,40	131,58

Literatura

A. Ahmad: *Data Communication Principles for Fixed and Wireless Networks*, Kluwer Academic Publishers, 2002.

T. Bonald, M. May, J. Bolot: "Analytical evaluation of RED performance", *Proc. IEEE Infocom 2000*, vol. 3, pp. 1415 – 1424.

A. Brumnić: *Računarske komunikacije — zbirka primjera*, Naučna knjiga, 1990.

K. Christensen *et al.*: "IEEE 802.3az: The Road to Energy Efficient Ethernet", *IEEE Communications Magazine*, vol. 48, No. 11, November 2010, pp. 50–56.

J. N. Daigle: *Queuing Theory With Application to Packet Telecommunication*, Springer Science + Business Media, 2005.

B. A. Forouzan: *Data Communications and Networking*, McGraw–Hill, 2007.

F. Gebali: *Analysis of Computer and Communications Networks*, Springer Science + Business Media, 2008.

ITU-D Study Group 2, Question 16/2: Handbook "Teletraffic Engineering", 2008.

V. B. Iversen: *Teletraffic Engineering and Network Planning*, Technical University of Denmark, 2009.

J. F. Kurose, K. W. Ross: *Computer Networking: A Top-Down Approach*, 8$^{\text{th}}$ Edition, Pearson Education Limited, 2021.

A. Leon–Garcia, I. Widjaja: *Communication Networks: Fundamental Concepts and Key Architectures*, McGraw–Hill, 2001.

I. Marsic: *Computer Networks: Performance and Quality of Service*, Rutgers University, 2008.

D. Minoli: *Voice Over IPv6*, Newnes, 2006.

L. L. Peterson & B. S. Davie: *Computer Networks: A Systems Approach*, Morgan Kaufmann Publishers, 2003.

Z. Petrović: *Širokopojasne digitalne mreže integrisanih servisa: ATM komutacija*, Akademska misao, 2002.

K. V. Prasad: *Principles of Digital Communication Systems and Computer Networks*, Charles River Media, 2003.

N. Sathaye: *Python Multimedia Beginner's Guide*, Packt Publishing, 2010.

W. Stallings: *Data and Computer Communications*, 8^{th} Edition, Pearson Education, 2009.

W. Stallings: *Local & Metropolitan Area Networks*, 6^{th} Edition, Prentice Hall, 2000.

F. A. Tobagi: "Fast packet switch architectures for broadband integrated services digital networks", *Proc. IEEE*, vol. 78, No. 1, January 1990, pp. 133 – 167.

C. H. Wu, J. D. Irwin: *Introduction to Computer Networks and Cybersecurity*, CRC Press, 2013.

Preporuke i standardi organizacija IEEE, IETF, ISO, ISS, ITU–T.

Katalozi Televés S.A.