Diplomski studij

Informacijska i komunikacijska tehnologija:

Telekomunikacije i informatika

Računarstvo:

Programsko inženjerstvo i informacijski sustavi

Računarska znanost

Ak.g. 2009./2010.

Raspodijeljeni sustavi

10.

Vrednovanje nefunkcionih karakteristika raspodijeljenih sustava

Dr. Dalibor Vrsalović

dalibor.vrsalovic@fer.hr

08.12.2009.

Sadržaj predavanja

- Uvod
- Dio I: Prirodne granice rasta performansi i kapaciteta
- Dio II: Najčešći načini raspoređivanja u praksi
- Dio III: Vrednovanje performansi sustava teorijom repova
- Dio IV: Performanse paralelnih algoritama
- Rekapitulacija

Literatura

- **Sadržaj ovog predavanja nastao je na temelju:**
  - N.J. Gunther: **"The practical performance analyst"**, *Mcgraw Hill i Authors Choice Press*, 1998 i 2000. ISBN 0-595-12674-X (poglavlje 2 i 3)
  - D.A. Menasce, V.A.F.Almeida: **"Capacity planning for web services"**, *Prentice Hall*, 2002 ISBN 0-13-065903-7 (poglavlje 1 i 5)
  - D.F. Vrsalovic, et. al: **"Performance prediction and calibration for a class of multiprocessors“**, *IEEE Transactions on Computers*, Volume: 37 Issue: 11 , Nov. 1988, pp. 1353 -1365

Za one koji žele dalje istraživati

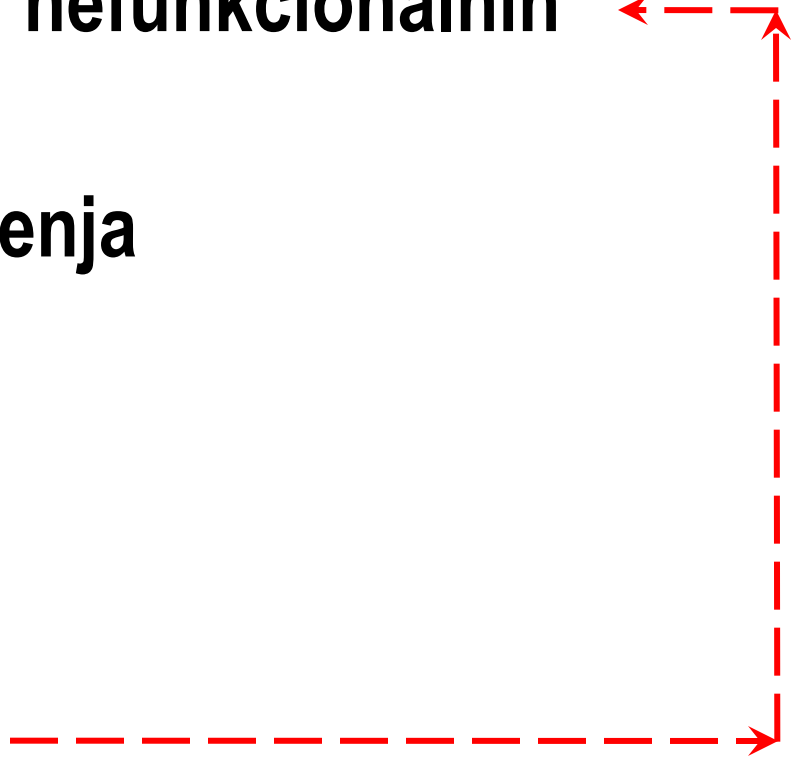
- A.O. Allen: **"Probability, Statistic, and Queueing Theory with Computer Science Applications"**, Academic Press 1978.
- S. Joines, R. Willenborg, K. Hygh: **"Performance analysis for Java Web Sites"**, Addison Wesley, 2003
- S. Sounders: **"High Performance Web Sites"**, O'Reilly, 2007.
- T. Schlosssnagle: **"Scalable Internet Architectures"**, Sams Publishing, 2007.
- D.A. Menasce, V.A.F. Almeida, L.W. Dowdy: **"Performance by Design"**, Prentice Hall, 2004
- N.J. Gunther: **"Analyzing computer system performance with Perl::PDQ"**, Springer, 2005. ISBN 3-540-20865-8.

Uvod

Što je vrednovanje nefunkcionalnih karakteristika sustava i zašto je to područje važno u praksi ?

Ciklus života računarskih sustava

- Definicija zahtjeva
- Analiza funkcionalnih i nefunkcionalnih karakteristika
- Razvoj odabranog rješenja
- Ispitivanje
- Pogon
- Mjerenja
- Modifikacija zahtjeva



Osnovni pojmovi nefunkcionalnih karakteristika sustava

- **Nefunkcionalne karakteristike sustava se skupno zovu kvaliteta usluge (QoS):**
- **Definicija garantiranih kvaliteta usluge naziva se ugovor o razini usluge (SLA)**
  - SLA je dio ugovora između pružaoca i primaoca usluga
  - Sve češće je i dio opisa posla internog ITC odjela
- **Postoje tri važne kategorije kvaliteta usluge:**
  - Performanse (Performance)
  - Pouzdanost/dostupnost (Reliability/availability)
  - Ukupni trošak vlasništva (TCO)

Primjer primjene Web aplikacije u trgovini

- **Organizacija prodaje putem Interneta. Aplikacija za prodaju ima sljedeće značajke:**
  - Neuspješni posjeti zbog loše kvalitete usluge
    - **60 %** kupaca napušta Web stranicu aplikacije ako je odziv aplikacije **između 4 i 6 sekundi**
    - **95 %** kupaca napušta Web stranicu aplikacije ako je odziv aplikacije **veći od 6 sekundi**
  - Uspješni posjeti s ostvarenom prodajom
    - **5 %** kupaca od svih koji su posjetili Web stranicu aplikacije kupi produkte za **prosječnu cijenu 1200 kn**

Analiza značajki Web aplikacije

- **Projektiranje i održavanje Web aplikacije ostvaruje se u skladu s očekivanim brojem i porastom korisnika aplikacije**
- **Ako se broj posjeta Web aplikaciji poveća za 30%, 60% ili 90%:**
  - Da li će odziv aplikacije biti zadovoljavajući ?
  - U kojim uvjetima će odziv aplikacije preći u nezadovoljavajuće područje ?
  - Koliki gubitak prihoda uzrokuje gubitak kupaca zbog slabog odziva aplikacije ?
  - Koje investicije su potrebne da se uz povećanje prometa zadrži sav posao ?
  - Kada će se, uz trenutačni trend, potreba za kapacitetom udvostručiti ?

Rezultati analize značajki Web aplikacije

	Povećanje broja korisnika			
	Danas	**+30 %**	**+60 %**	**+90 %**
Maks. posjeta/sat	900.00	1,170.00	1,440.00	1,710.00
Vrijeme odziva (s)	2.96	3.80	5.31	8.83
Izgubljeni kupci (%)	0.00	0.00	60.00	95.00
Mogući broj prodaja / sat (kn)	45.00	58.50	72.00	85.50
Mogući prihod / sat (kn)	54,000.00	70,200.00	86,400.00	102,600.00
Stvarni prihod / sat (kn)	54,000.00	70,200.00	34,560.00	5,130.00
Izgubljeni prihod / sat (kn)	0.00	0.00	51,840.00	97,470.00

Rezultati analize značajki Web aplikacije (nastavak)

- **Poduzeće će izgubiti više od 95% mogućeg prometa na Internetu ako se broj potencijalnih kupaca udvostruči**
- **Na ovom predavanju saznati će te kako pristupiti vrednovanju performansi sustava i planiranju rasta kapaciteta**
- **Na temelju prikazanih rezultata može se zaključiti da:**
  - Linearna ekstrapolacija najčešće daje netočne rezultate
  - Pogrešno projektirana aplikacija može imati katastrofalne posljedice na poslovanje

- **Intuicija i iskustvo**
  - Raspodijeljeni sustavi pokazuju izrazito nelinearno ponasanje pa su procjene vrlo teske
- **Modeliranje**
  - Podrazumijeva razvoj matematičkog modela koji opisuje ovisnost performansi o pojedinim parametrima sustava
- **Simulacija**
  - Najtočnija metoda ali često preskupa za upotrebu

Koraci kod razvoja modela

- Razumijevanja funkcioniranja sustava
- Modeliranje tereta
- Mjerenje sustava u pogonu radi utvrđjivanja parametara tereta
- Razvoj modela performansi
- Verifikacija i validacija modela
- Analiza mogućih scenaria promjena u budućnosti
- Prognoza promjena tereta u budućnosti
- Prognoza performansi sustava nakon pustanja u pogon te u budućnosti

Pojam modela tereta sustava

- **Prirodni modeli tereta**
  - Aplikacije
- **Umjetni modeli tereta**
  - Benchmarks (SPEC, TPCc, …)
- **Neizvodivi modeli tereta**
  - Ritam dolazaka, servisni zahtjevi, klase komponenti i njihova razina konkurentnog izvodjenja, u/i zahtjevi

Utvrdivanje značajki realnog tereta

- Sednja vrijednost parametara ne mora biti reprezentatvna ako se pojedinačne vrijednosti nalaze u grupama koje se značajno razlikuju po vrijednostima
- U takvom slučaju potrebno je provesti grupiranje i utvrditi značajke za svaku grupu
- Postoje različite metode grupiranja i programi koji obavljaju grupiranje ali najčešće se koristi grupiranje na temelju Euklidske udaljenosti

$$d = \sqrt{\sum_{n=1}^{K} (x_{in} - x_{jn})^2}$$ K= broj parametara

Primjer

- Teret Web poslužioca sastoji se od slijedećih grupa zahtjeva:

Dokument	Veličina (KB)	Broj pristupa
1	12	281
2	150	28
3	5	293
4	25	123
5	7	259
6	4	241
7	35	75

- Grupiraj teret u tri odvojene grupe

Primjer

- Budući da su vrijednosti različite treba prvo obaviti promjenu mjerila. U ovom slučaju koristimo $\log_{10}$

Dokument	Veličina (KB)	Broj pristupa
1	1.08	2.45
2	2.18	1.45
3	0.70	2.47
4	1.40	2.09
5	0.85	2.41
6	0.60	2.38
7	1.54	1.88

Primjer

- Podrazumjevajući da je težište grupe od 1 točke ta točka izračunamo Euklidske udaljenosti izmedu tih težišta:

Grupa	G1	G2	G3	G4	G5	G6	G7
G1	0	1.49	0.38	0.48	0.24	0.48	0.74
G2		0	1.79	1.01	1.01	1.64	1.83
G3			0	0.79	0.16	0.13	1.03
G4				0	0.64	0.85	0.26
G5					0	0.25	0.88
G6						0	1.07
G7							0

- Budući da je udaljenost izmedu G3 i G6 najmanja izračunamo težište nove grupe G36:
  - (0.7+0.6)/2 = 0.65 i (2.47+2.38)/2 = 2.43

Primjer

- Sada izračunamo Euklidske udaljenosti između novih težišta:

Grupa	G1	G2	G36	G4	G5	G7
G1	0	1.49	0.43	0.48	0.24	0.74
G2		0	1.81	1.01	1.64	0.76
G36			0	0.82	0.19	1.05
G4				0	0.64	0.26
G5					0	0.88
G7						0

- Budući da je udaljenost između G36 i G5 sada najmanja izračunamo težište nove grupe G365:
  - (0.65+0.85)/2 = 0.75 i (2.43+2.41)/2 = 2.42

Primjer

- Postupak iterativno ponavljamo dok ne dobijemo željene tri grupe:

Grupa	G1365	G2	G47
G1365	0	1.60	0.72
G2		0	0.89
G47			0

- Nakon što smo dobili broj željenih grupa vratimo parametre u izvorno mjerilo

Primjer

- Nakon vraćanja parametara u izvorno mjerilo dobivamo novi model tereta

Grupa	Tip dokumenta	Velicina (KB)	Broj zahtjeva
G1356	Mali	8.19	271.51
G47	Srednji	20.58	96.05
G2	Veliki	150.0	28

- Ovakovo grupiranje daje puno realističnije postavke za modeliranje sustava

Najčešće pogreške kod modeliranja

- Prekompleksna analiza
- Nema specifičnog cilja
- Prejudiciranje (model treba dokazati da je nas sustav bolji od njihovog)
- Nedovoljno razumijevanje sustava
- Neadekvatne mjere (napr. TPS za Web i DB usporedbu)
- Nereprezentativni teret
- Neuključivanje važnih parametara
- Promatranje u krivom intervalu vrijednosti parametara
- Krivo baratanje ekstremima
- Nedovoljno promatranje evolucije sustava i tereta
- Kriva intepretacija rezultata

Rezultat zavisi o interpretaciji

	Teret 1 (ms)	Teret 2 (ms)	Prosječno (ms)	
Sustav 1	10	20	15	Srednja vrijednost
Sustav 2	20	10	15	
Sustav 1	1.00	1.00	1.00	Sustav 1 referentan
Sustav 2	2.00	0.50	1.25	
Sustav 1	0.50	2.00	1.25	Sustav 2 referentan
Sustav 2	1.00	1.00	1.00	

Osnove utvrđivanja pouzdanosti, dostupnosti i ukupne cijene vlasništva

Pouzdanost i dostupnost sustava

- Srednje vrijeme između grešaka (MTBF)
- Srednje vrijeme popravka (MTTR)
- Dostupnost
  - Postotaka ukupnog vremena koje je sustav u pogonu ili vjerojatnost da će sustav funkcionirati u vremenu t = ti
  - Računa se kao MTBF/(MTBF + MTTR)
  - Dostupnost od 0.9999 znaci da ce sustav biti izvan funkcije (1-0.9999)x30x24x60 = 4.32 min/mjesec
  - Dostupnost serijske kombinacije dva podsustava jednaka je produktu dostupnosti pojedinih podsustava
  - Dostupnost paralelne kombinacije dva podsustava jednaka je
    - Dp = D1(1-D2) + D2(1-D1) + D1D2 (predpostvlja da je jedan dostupan podsustav dovoljan za pravilno funkcioniranje sustava)

Primjer računanja dostupnosti

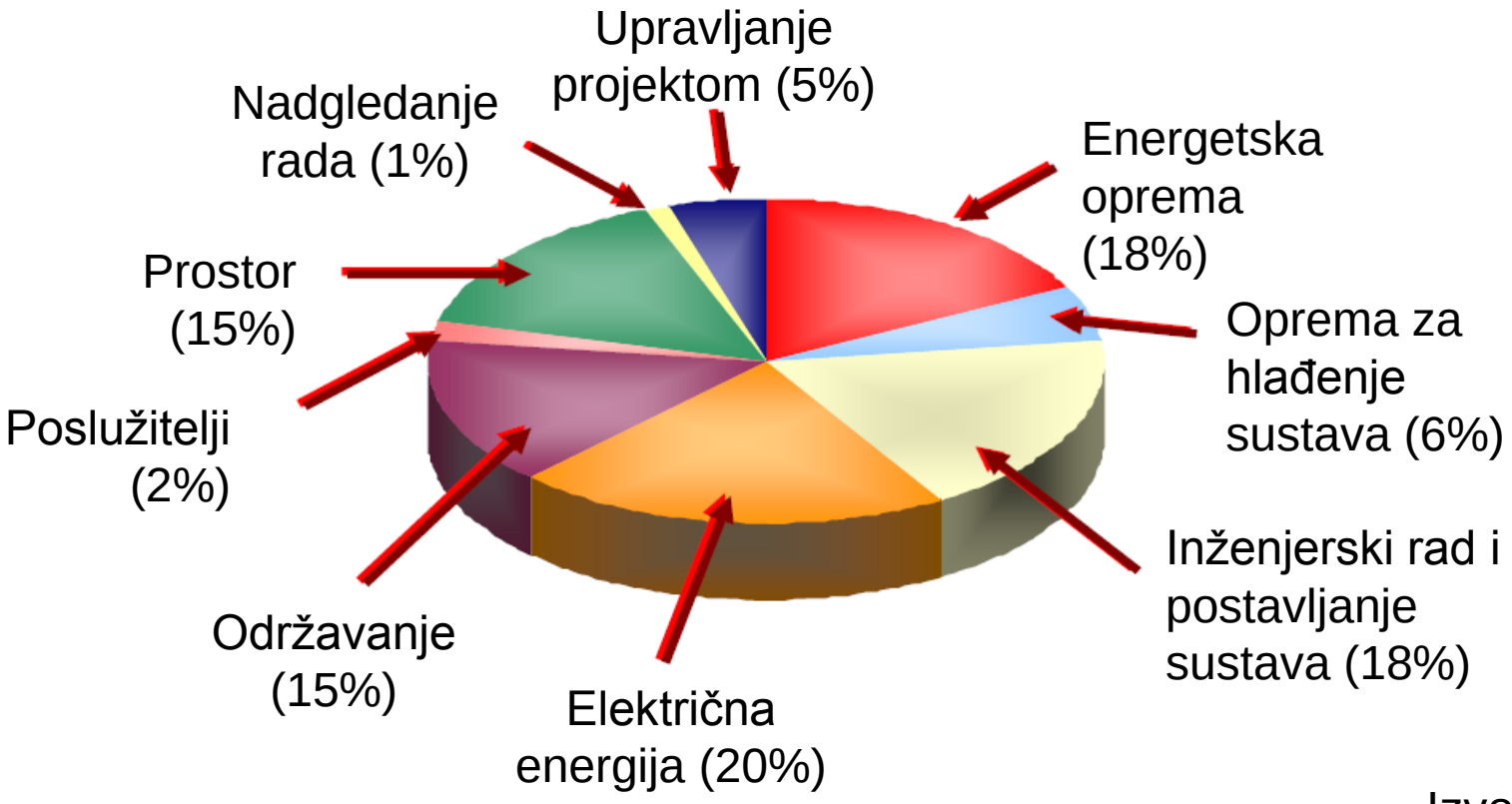
- Izračunaj dostupnost sustava koji uključuje dva paralelna Web poslužioca sa dostupnosću od 0.99 te jedne mrežne sklopke za raspodjelu tereta sa MTBF = 1 godine i MTTR od 2 sata.
- Dostupnost sklopke = (365*24)/(365*24 +2)= 0.9998
- Dostupnost 2 paralelna Web posluzioca = 0.0099 + 0.0099 +0.9801 = 0.9999
- Dostupnost sustava = 0.9998 * 0.9999 = 0.9997

Totalni trošak vlasništva

- Trošak amortizacije sustava (CAPEX)
  - zavisi o propisanom vremenu trajanja (obično 3 godine) za sustave računala. Napr. za 3 godine računa se 33.33% nabavne cijene godišnje
- Trošak pogona sustava (OPEX)
  - Zaposlenici (ITC odjel)
    - Razvoj, pogon
  - Prostor
  - Režije
    - Energija, komunikacijske usluge, fizička sigurnost

Ukupna cijena vlasništva

- **Prosječni trošak uporabe sustava poslužitelja**
  - Kroz tri godine korištenja (CAPEX/OPEX = 50/50)
  - Ankete na Internetu pokazuju slijedeći prosjek (promjenljivo s obzirom na geografiju i vrijeme) :



Izvor: Intel

Mogući načini ostvarenja sustava

- **Iznajmljivanje infrastrukture poslužitelja**
  - Udomljivanje sustava
    - Udomitelj infrastrukture
      pruža i upravlja fizičkom infrastrukturom, kao što je zgrada, napajanje te pristup Internetu
    - Zakupnik infrastrukture
      postavlja i upravlja sredstvima koja se poslužuju
  - Upravljani sustav
    - Udomitelj podrzava operacijski sustav s listom poznatih aplikacija
  - Dedicirani sustavi imaju samo jednog korisnika
  - Zajednički sustavi imaju vise korisnika koji rade u raspodjeljenom vremenu
    - Niza cijena ali korisnici ne mogu utjecati na performanse
- **Izgradnja vlastitog poslužitelja**

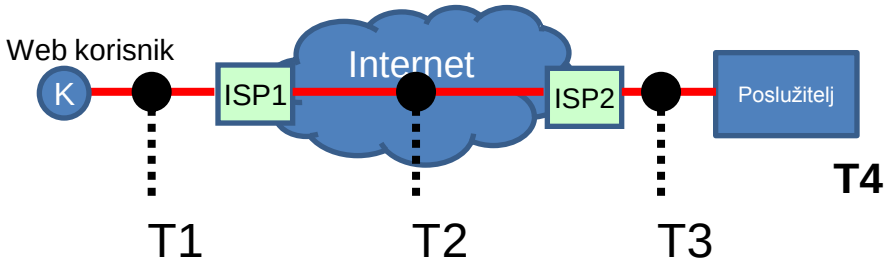
Osnovni pojmovi vezani uz performanse sustava

Pojam performansi sustava

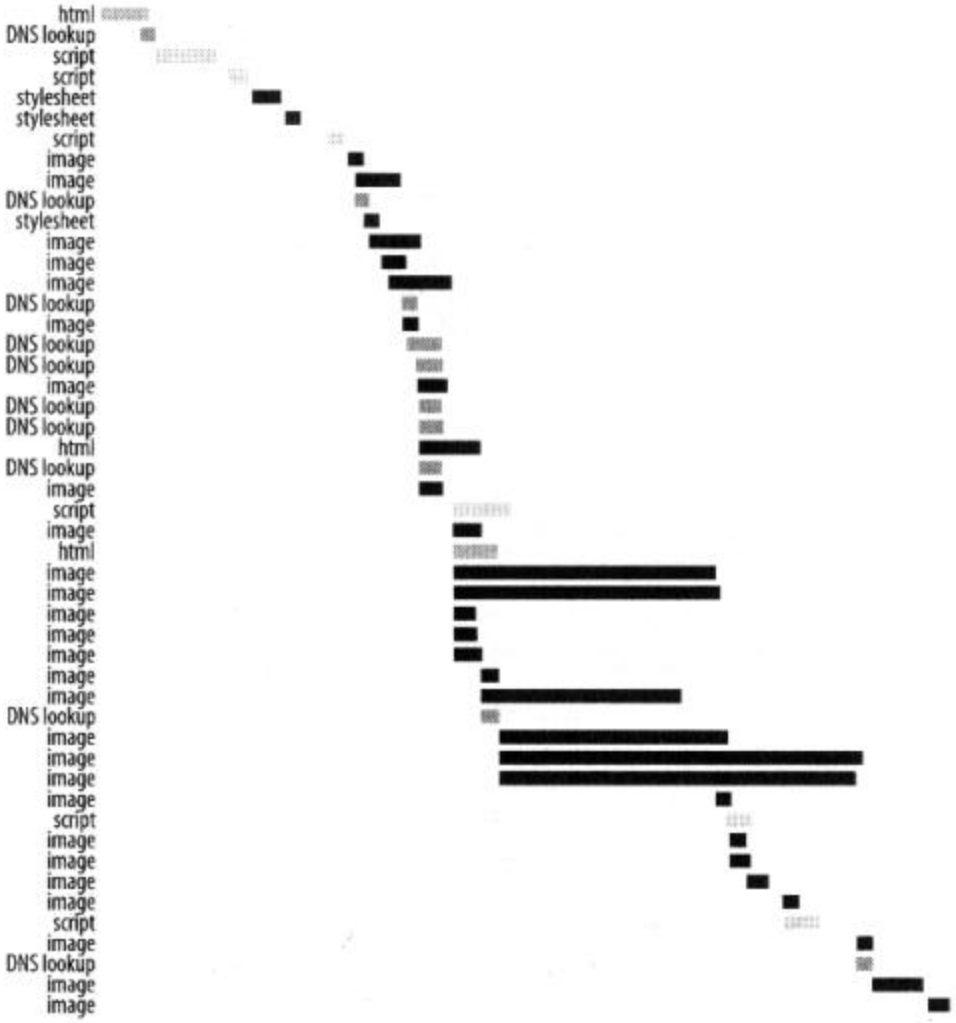
- **Vrijeme odziva sustava (Response time)**
  - Vrijeme potrebno da sustav odgovori na zahtjev za posluživanje (napr. vrijeme izmedu pritiska na miš i pojave Web stranice na zaslonu)
- **Propusnost sustava (Throughput)**
  - Broj posluženih zahtjeva u jedinici vremena (razlicito od 1/vrijeme odziva jer se zahtjevi mogu posluživati paralelno)
  - Propusnost je funkcija tereta i kapaciteta sustava
- **Kapacitet sustava je jednak maksimalnoj mogućoj propusnosti sustava (Capacity)**
  - Kod maksimalne propusnosti najopterećeniji poslužitelj u sustavu je zauzet 100% vremena

Vrijeme odziva u raspodjeljenom sustavu

- Vrijeme odziva
  - Vrijeme u mreži(T1 + T2 + T3)
    - Kašnjenje
    - Vrijeme prijenosa
  - Vrijeme u posluziocu (T4)
    - Vrijeme posluživanja
      - CPU
      - Disk
      - LAN
    - Vrijeme čekanja na resurse kao što su:
      - CPU
      - Disk
      - LAN



Primjer dobavljanja Web stranice: My Space

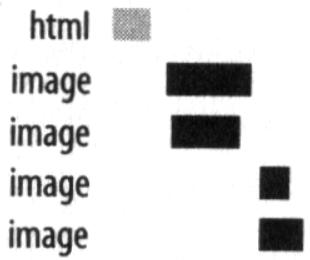


Vremenski dijagram dohvaćanja objekata stranice

Veličina: 205K	Vrijeme: 7.8 s
Broj zahtjeva: 39	YSlow: D

Prikaz stranice pregledniku

Bolji primjer: Google

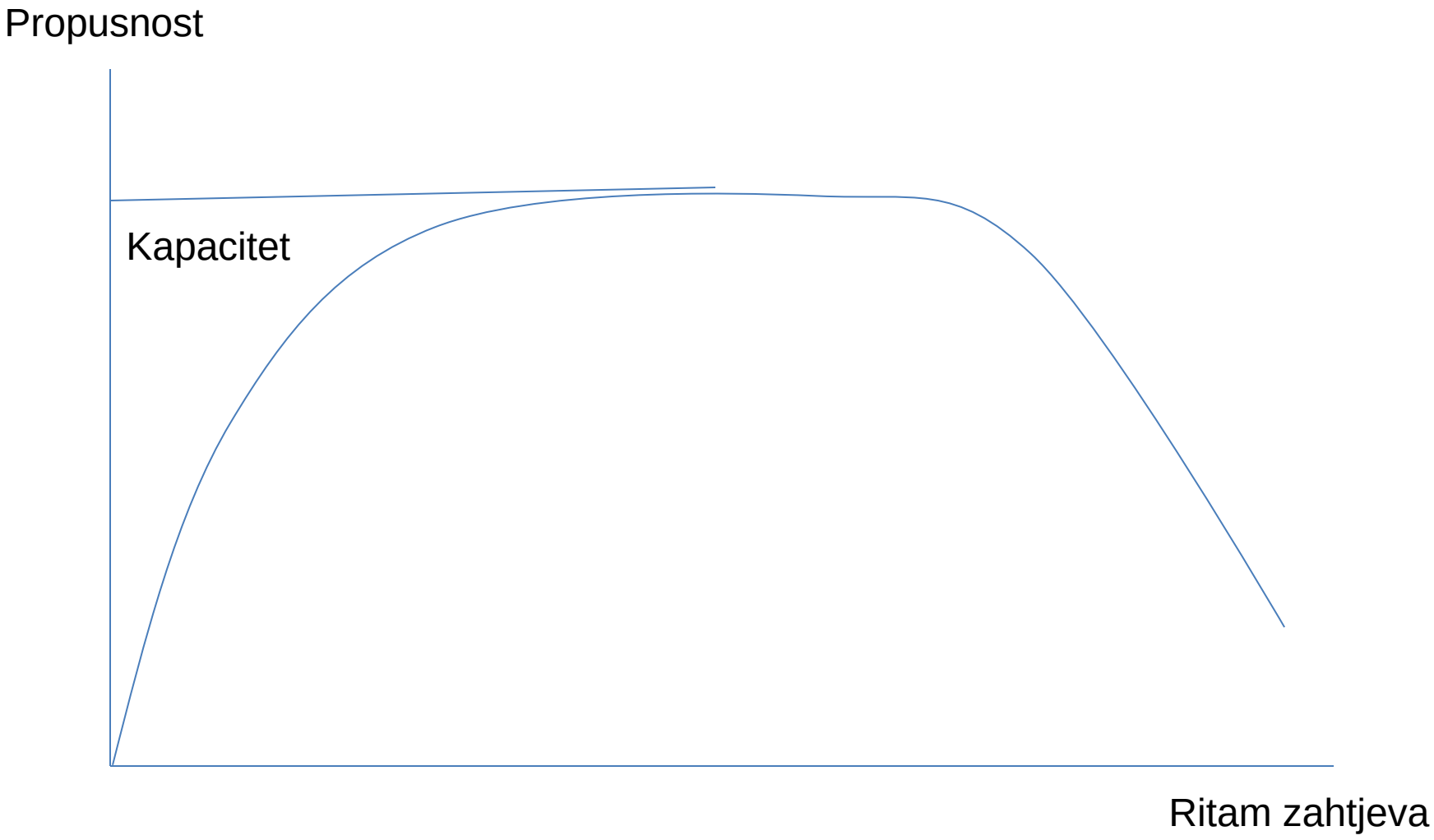


Vremenski dijagram dohvaćanja objekata stranice

Veličina: 18K	Vrijeme: 1.7 s
Broj zahtjeva: 3	YSlow: A

Prikaz stranice pregledniku

Propusnost kao funkcija tereta



Tipične jedinice za propusnost

- **Broj ispunjenih zahtjeva za posluživanje u jedinici vremena**
- **Jedinica zavisi o razini sustava na kojoj se promatra**
- **Primjeri:**
  - Broj transakcija u sekundi
  - Broj pretinaca u sekundi
  - Broj Web stranica u sekundi
  - Broj poruka u sekundi
  - Broj instrukcija u sekundi

Primjer 1.1

- I/O operacija diska u sustavu za “on-line” transakcije traje prosječno 10ms
- Iskorištenje diska je 100% (t.j. stalno zauzet)
- Kolika je maksimalna propusnost t.j. kapacitet diska?
- Propusnost = min [kapacitet, teret]
- Maksimalna propusnost tj. kapacitet diska je 1/0.01 = 100 operacija u sekundi

Kapacitet poslužitelja određuje se kod 100% zaposlenosti

Prirodne granice rasta ubrzanja i kapaciteta

Ostvarivanje razmjernog rasta aplikacije (skaliranje)

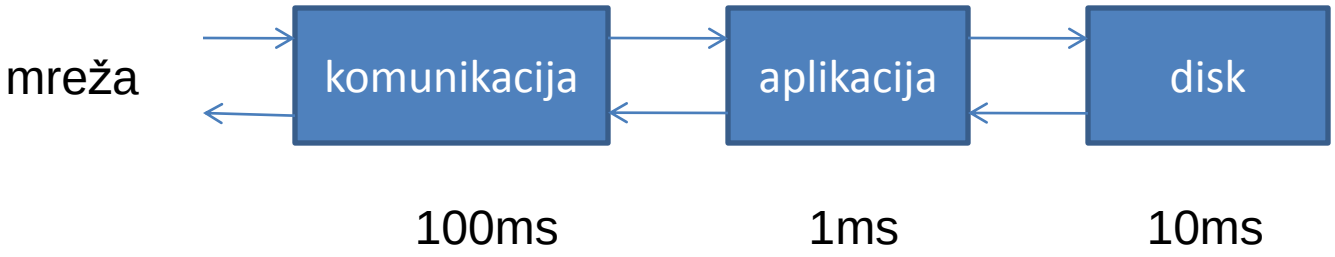
- **Osnovni modeli ostvarivanja razmjernog rasta kapaciteta sustava**
  - **Vertikalno skaliranje** podrazumijeva prijelaz na server sa većim kapacitetom
  - **Horizontalno skaliranje** podrazumijeva dodavanje poslužitelja (obično istog kapaciteta).
  - **Skaliranje prema gore** (većem kapacitetu) je jednako važno kao i **skaliranje prema dolje** (manjem kapacitetu) zbog potrebe da se troškovi prilagode prihodima!

Promjene u arhitekturi sustava koje omogućavaju redukciju vremena odziva i povećanje kapaciteta

- **Web poslužitelj sa 3 modula:**
  - Mrežni modul sa odzivom od 100ms
  - Modul aplikacije sa odzivom od 1ms
  - Modul diska sa odzivom od 10ms
  - Obrada se odvija u slijedu pa je ukupni odaziv 111ms
- **Poboljšanje preformansi može se dobiti:**
  - Serijskim preklapanjem
  - Paralelnim preklapanjem
  - Paralelnim izvođenjem
  - Privremenim spremnikom

Serijsko preklapanje (pipelining)

- Povečava propusnost ali ne smanjuje vrijeme odziva

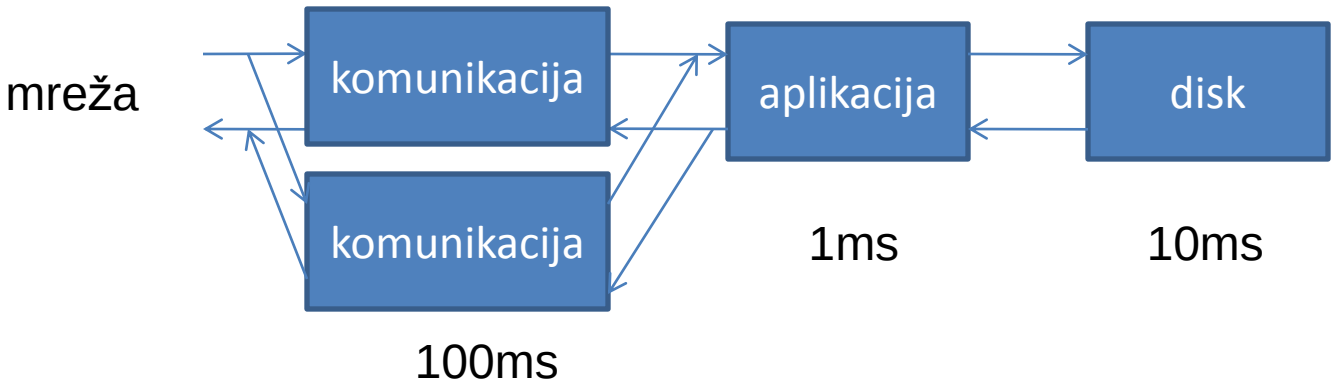


Todziva = 111ms
Propusnost = 1/0.1s = 10/s

Napomena: Determinističko razmatranje bez uticaja razdiobe vremena posluživanja čekanja u repovima i ograničenja zbog interakcija paralelnih aktivnosti

Paralelno preklapanje

- Povečava propusnost ali ne smanjuje vrijeme odziva

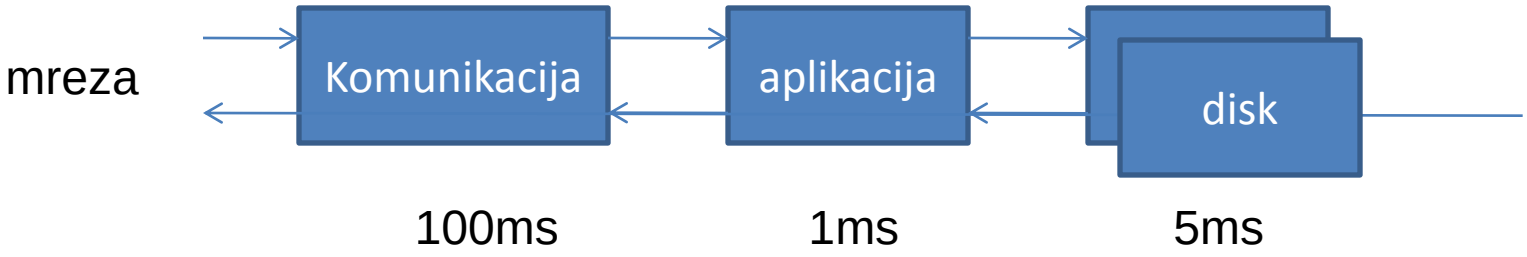


Todziva = 111ms
Propusnost = 2/0.1s = 20/s

Paralelno izvodjenje

- Smanjuje vrijeme odziva

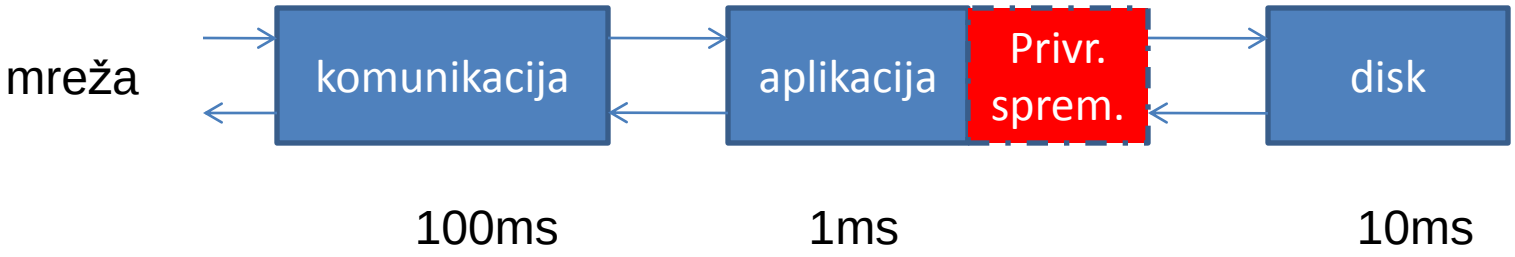
Zapisi istog pretinca raspodijeljeni su preko dva fizička diska pa se dohvaćaju paralelno



Todziva = 106ms
Propusnost = 1/0.1s = 10/s

Privremeno spremanje (cache)

- Smanjuje vrijeme odziva

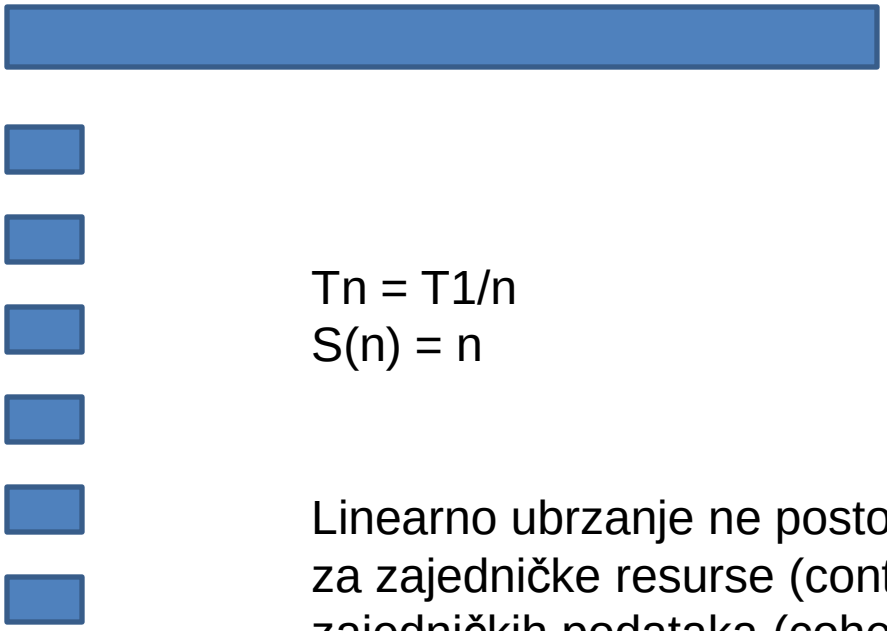


Todziva = 101+ 10(1-p) ms gdje je p vjerojatnost da se pretinac ne nalazi u privremenom spremniku

Propusnost = 1/0.1s = 10/s

Idealni paralelizam

- Ubrzanje (speedup) je omjer vremena odziva jednog i n paralenih podsustava uy jednaki teret
  - S(n) = T1/Tn



Tn = T1/n
S(n) = n

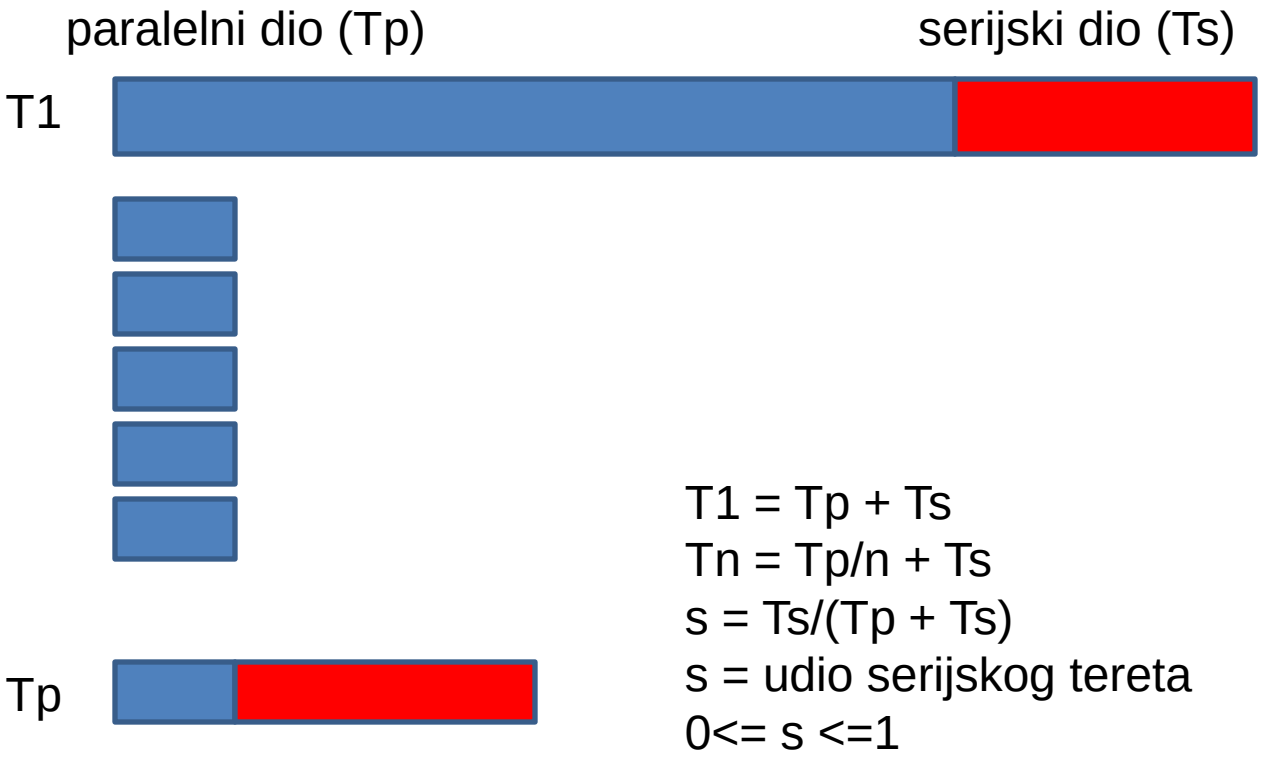
Linearno ubrzanje ne postoji u praksi zbog natjecanja za zajedničke resurse (contention) i usklađjivanja zajedničkih podataka (coherence)

Utjecaj OS I zajedničkih resursa

- Tipični uzroci sukoba zahtjeva (contention) ili potrebe za usklađjivanjem podataka (coherence) su:
  - Zajedničke funkcije i variable u OS
  - Izmjena zajedničkih podataka koji se mijenjaju u privremenim spremnicima (CACHE)
  - Promet podataka u/iz glavne memorije
  - Sinhronizacione primitive
  - Čekanje na ulaz/izlaz

Serijski dio tereta i ubrzanje

- Teret se sastoji od dijela koji se moze paralelizirati i dijela koji je sekvencijalan zbog pristupa zajedničkim resursima



$T1 = Tp + Ts$

$Tn = Tp/n + Ts$

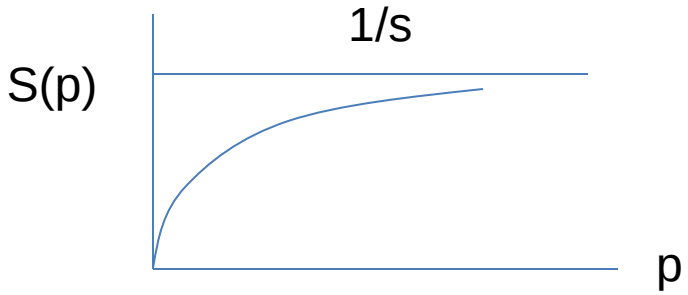
$s = Ts/(Tp + Ts)$

s = udio serijskog tereta

$0 <= s <= 1$

Amdahl-ov zakon

- Tp = sT1 + (1 – s)T1/p (s – serijski udio tereta)
- Ubrzanje S(p) = T1/Tp
- S(p) = T1/(T1(s + (1-s)/p))
- Amdahl-ov zakon:
  - S(p) = p/(1 + s(p – 1))
- Amdahl-ova asimptota
  - S(p) = 1/(1/p + s(1-1/p))
  - P-> beskonačno S(p) -> 1/s u praksi [s << 1]

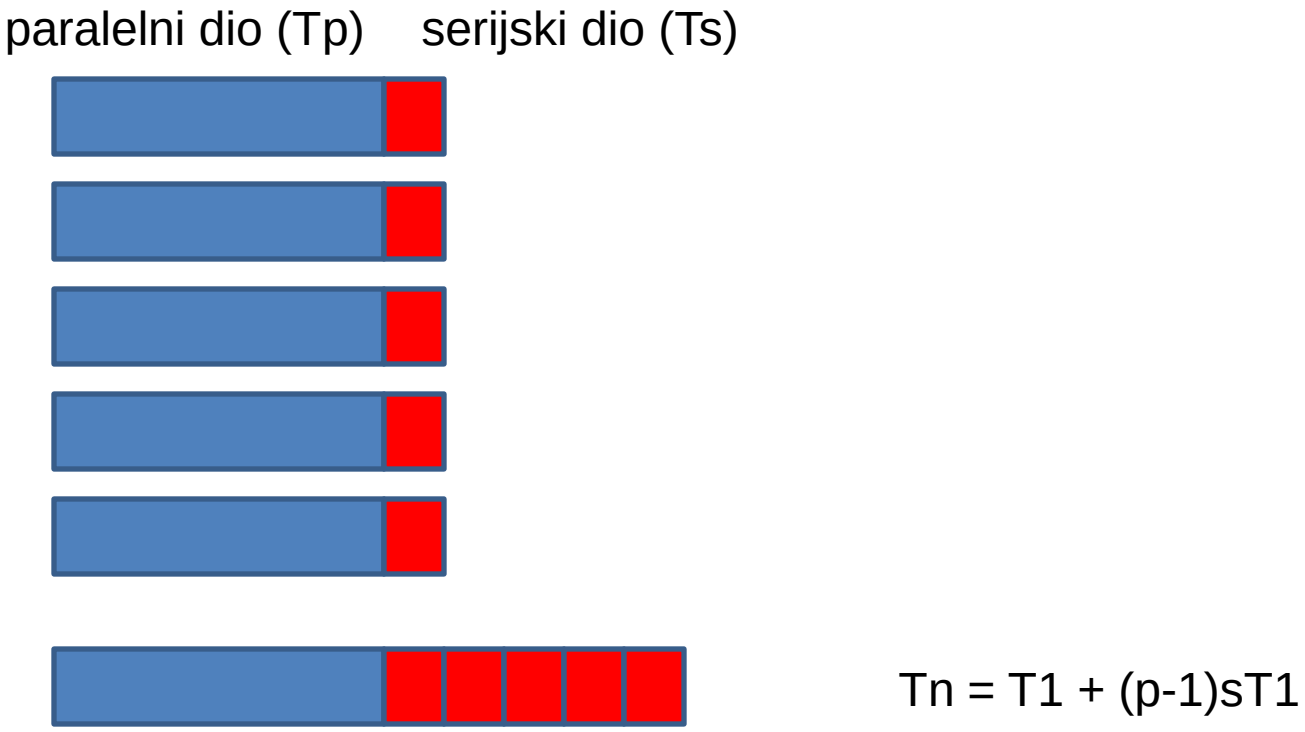


Efikasnost

- Mjeri koliko svaki paralani procesor doprinosi poboljšanju vremena odziva
- $E(p) = S(p)/p = 1/(1 + s(p – 1))$
- $d/dp\ 1/E(p) = s$
- Udio serijskog tereta s može se dobiti na temelju mjerenja ubrzanja S(p)
  - $S(p) = p/(1 + s(p – 1))$
  - $1+s(p-1) = p/S(p)$
  - $s(p-1) = p/S(p) -1$
  - $s = ((p/S(p)) -1)/(p-1)$

Skaliranje Kapaciteta

- Skaliranje mjeri rast kapaciteta sa rastom broja paralelnih tereta za razliku od ubrzanja koje mjeri odziv na jedan teret koji se izvodi paralelno



$$Tn = T1 + (p-1)sT1$$

Rast kapaciteta

- Relativni kapacitet $C(p) = Xp/X1$ (X == protok)
- $X1 = C1/T1$
- $C2 = 2C1$
- $T2 = T1 + sT1$
- $X2 = C2/T2 = 2C1/(T1 + sT1) = 2(C1/T1)/(1+s)$
- $X2 = 2\ X1/(1 + s)$
- $Xp = pC1/(T1 + (p-1)sT1) = pX1/(1 + s(p-1))$
- $C(p) = Xp/X1 = p/(1 + s(p-1))$
- Ista funkcija kao Amdah-ov zakon

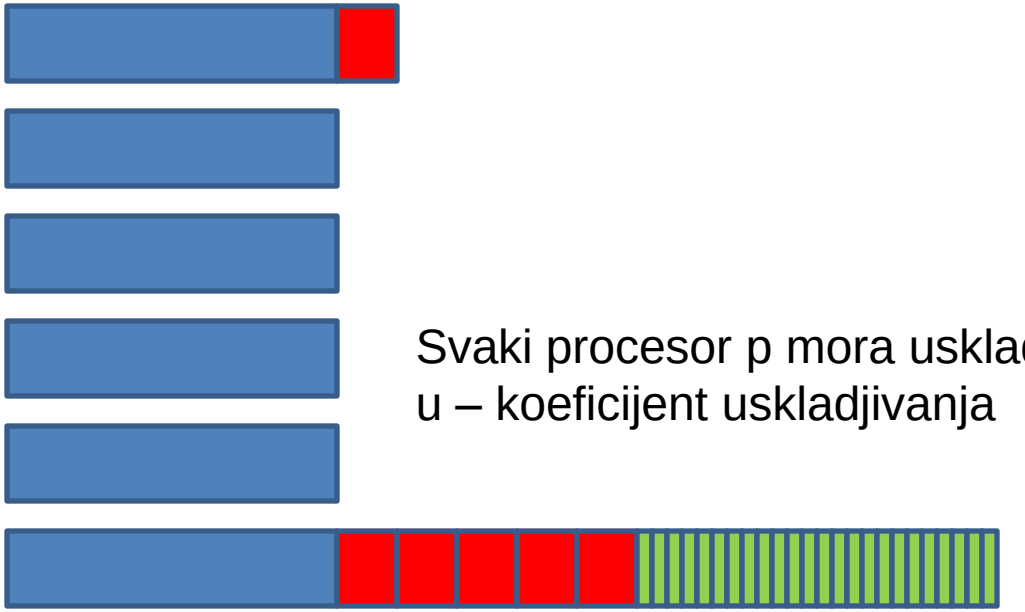
Skalirano ubrzanje

- Ako umjesto dijeljenja paralelnog dijela tereta dodamo svakom paralelnom podsustavu isti paralelni teret (1-s)T1
- T1’ = sT1 + (1-s)pT1
- Css(p) = (s +(1 –s)p)T1/(s + (1-s)p/p)T1
- Css (p) = s + (1-s)p
- Teoretski mogući bolji rezultati od Amdahl-ovog zakona jer kapacitet ovisi linearno o p
- Fiksni teret zamijenjen je sa teretom kojem se paralelni dio moze povećati p puta
- Tesko ostvarljivo u praksi jer nema potpuno neovisnih operacija (bar utjecaj OS) pa se i serijski dio povećava

Utjecaj sukoba i usklađivanja na skaliranje

- Serijski dio tereta ovisi ne samo o natjecanju za zjednicke resurse nego i o potrebi usklađivanja p paralenih podsustava (izmjena promjenjenih zajedničkih vrijednosti)

T1 = paralelni dio (Tp) + serijski dio (Ts)

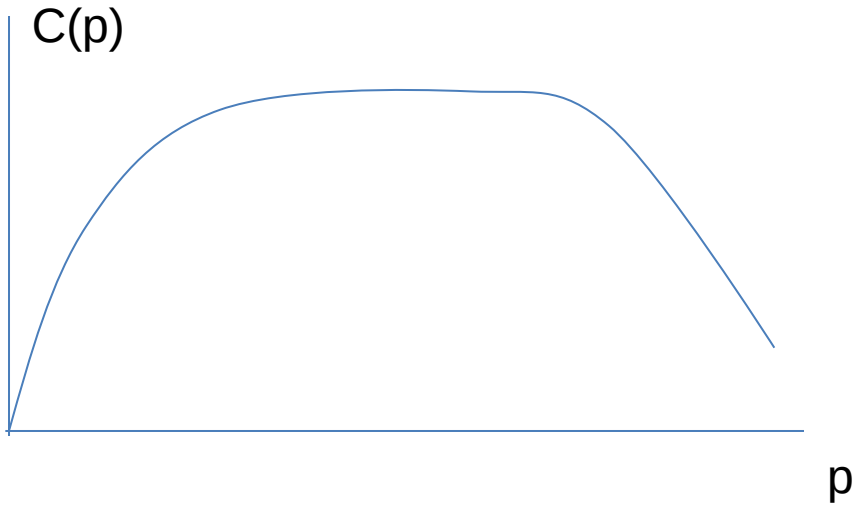


Svaki procesor p mora uskladivati (p-1) drugih procesora
u – koeficijent uskladjivanja

$$Tn = T1 + (p-1)sT1 + p(p-1)uT1$$

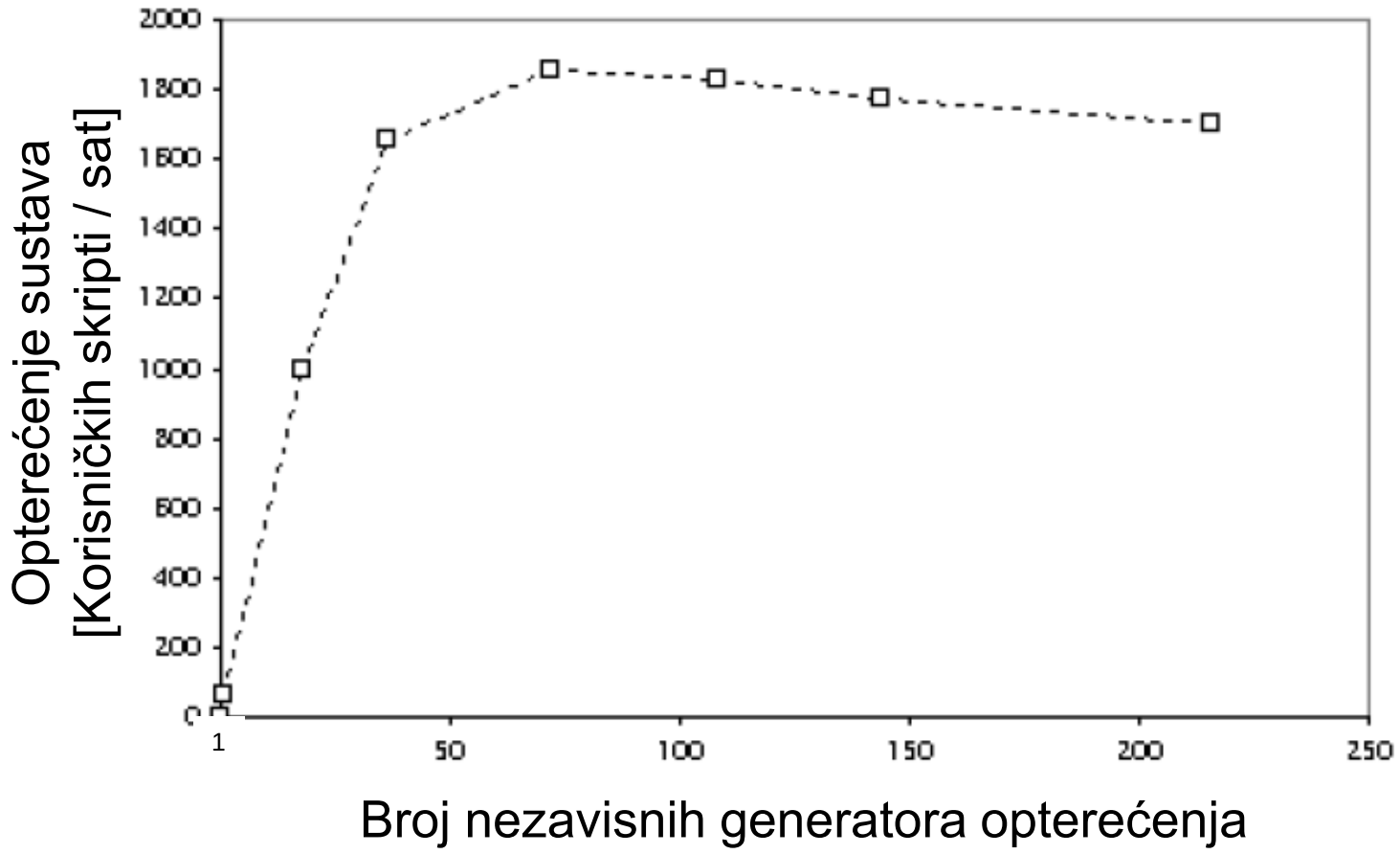
Univerzalni model skaliranja

- C(p) = p/(1 + s(p-1) + up(p-1))
- C(p) ima maksimum za broj procesora p*
- F(p) = 1 + s(p-1) + up(p-1);
- C(p) = p/F
- dC/dp = 0
- F = up(p-1) +up**2 + sp
- (1-s) = up**2
- p* = SQRT((1 – s)/u)

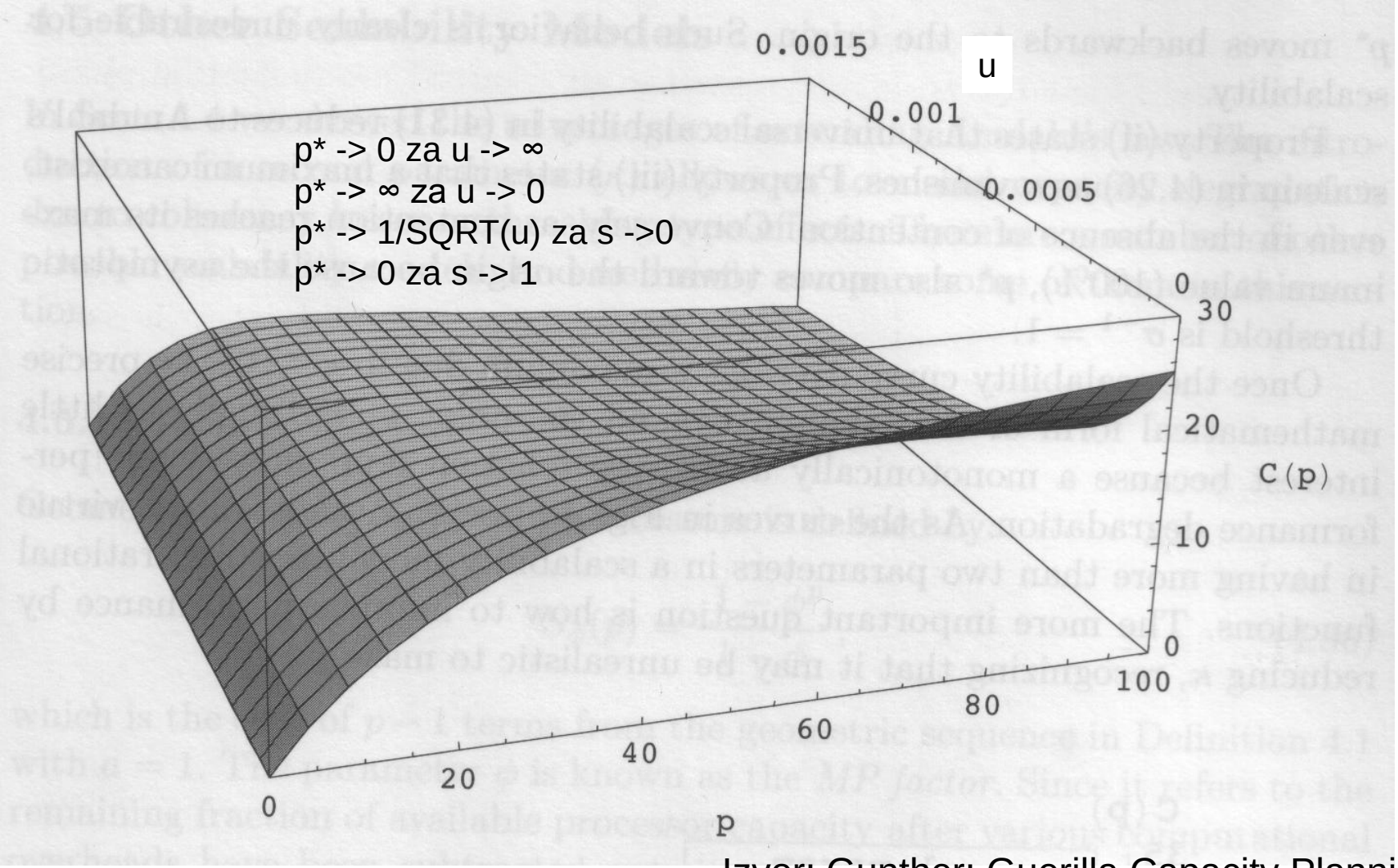


- s = utjecaj natjecanja za izvođenje serijskog dijela tereta
- u = utjecaj usklađivanja kopija zajedničkih vrijednosti

Rezultati mjerenja kapaciteta Web poslužitelja



Osjetljivost C(p) i p* u odnosu na u



Izvor: Gunther: Guerilla Capacity Planning

Geometrijski model

- Geometrijsko ubrzanje je definirano:
  - $S(p) = (1 – F^p)/(1 – F)$
- F se još zove u literaturi MP faktor jer ukazuje na dio kapaciteta procesora nakon sto su izuzete svi kapaciteti koji se koriste za dodatne sistemske poslove
- 0<= F =< 1
- Kadkad se koristi u industriji za opisivanje multiprocesora sa relativno malim brojem procesora uz vrlo konzervativne rezultate

Kvadratni model

- $S(p) = p - jp(p - 1)$
- J je parametar koji opisuje dodatne sistemske radnje
- $0 <= j < 1$
- S(p) ima maksimum za $p = p^*$
  - $p^* = (1+j)/2j$

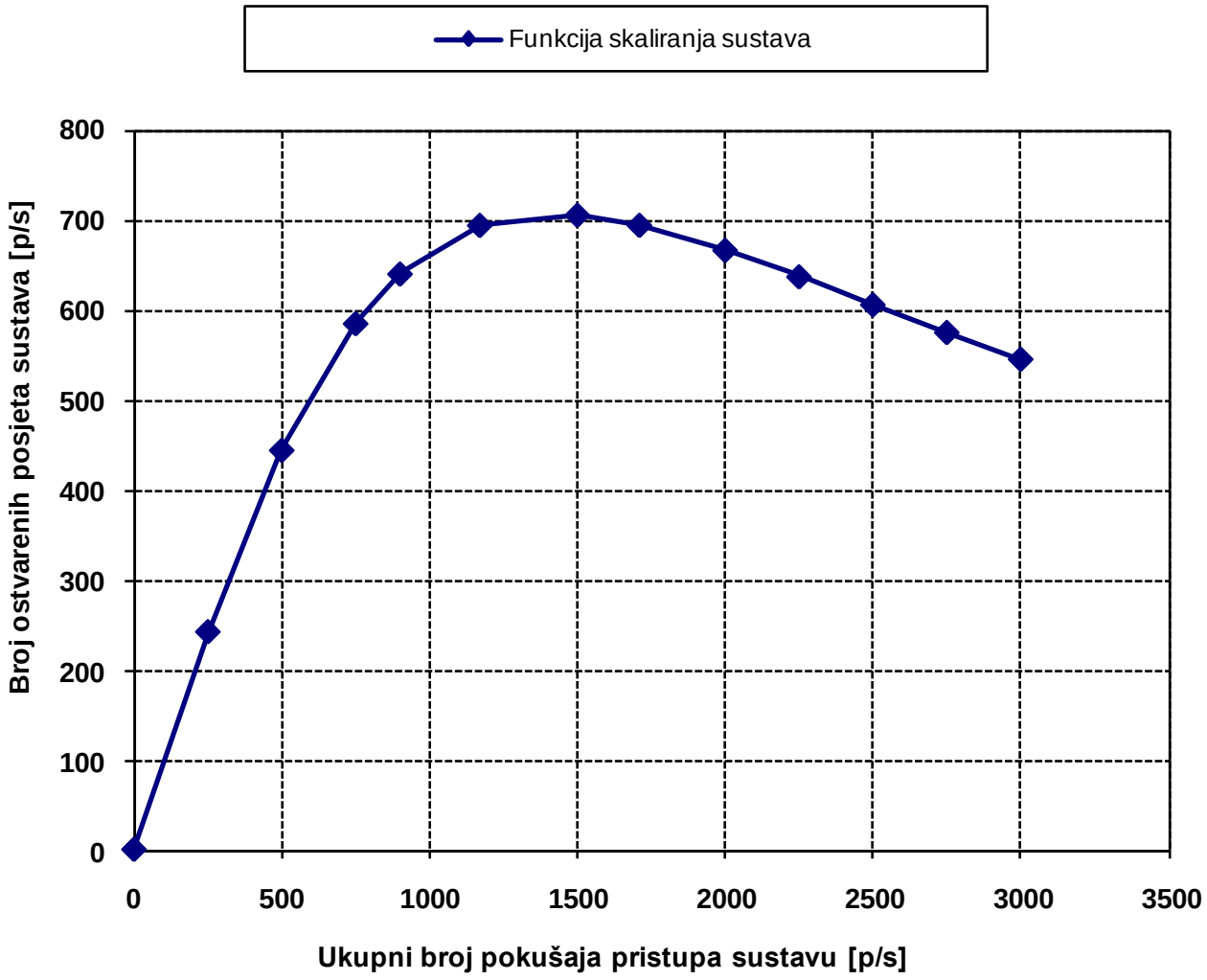
Eksponencijalni model

- $S(p) = p(1-a)^{(p-1)}$ $0 <= a < 1$
- $S(p) \rightarrow p\, e^{-a(p-1)}$ za $p \rightarrow \infty$
- Upotrebljava se za:
  - ALOHA paket radio protokol
  - Rezoluciju konflikata na zajedničkoj sabirnici
- U praksi dobar za mali broj čvorova
- Pokazuje preveliki pad ubrzanja za veliki broj čvorova

Skaliranje klastera

- Podrazumijeva klaster više čvorova kod kojih svaki čvor ima vise procesora/računala
- $C(p,n) = nC(pn)/(1 + sg(n-1)C(pn) + ug*n(n-1)C(pn)^2)$
  - Gdje su pojedini parametri definirani kao:
    - n – broj čvorova
    - p – ukupni broj procesora
    - sg – natjecanje između čvorova
    - ug – usklađivanje između čvorova
    - pn – broj procesora po čvoru = p/n (homogeni sustav)
- C(pn) – kapacitet svakog čvora od p/n procesora/računala
- Ukupni kapacitet jednak je kapacitetu čvora pomnoženim sa brojem čvorova umanjenom prema univerzalnom zakonu skaliranja zbog natjecanja i usklađivanja između čvorova

Uporaba regresije za određivanje funkcije skaliranja



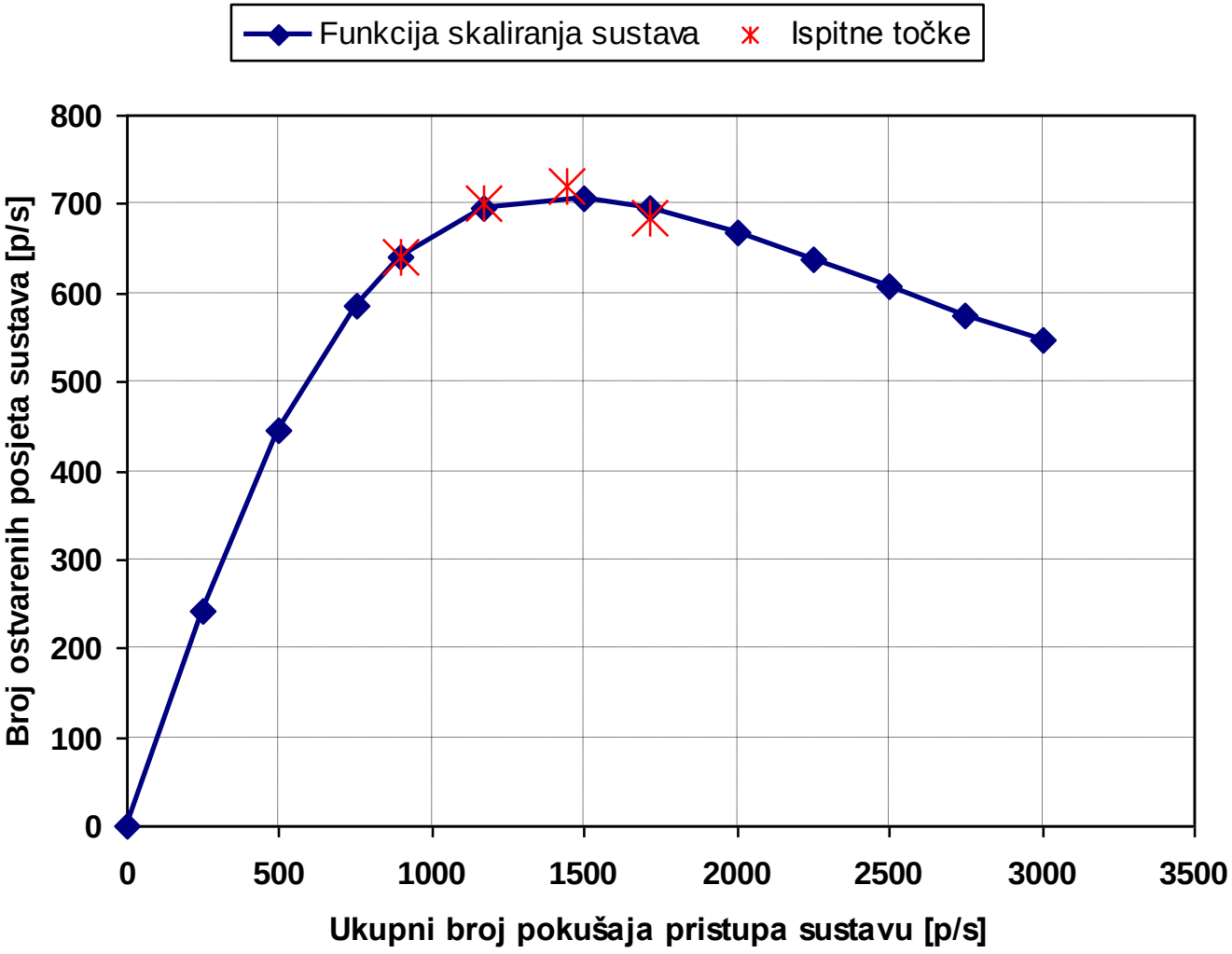
Određivanje parametara iz mjerenja

- $C(p) = p/(1 + s(p-1) + up(p-1))$
- $p/C(p) = 1 + s(p-1) + up(p-1)$
- $y = ax^2 + bx + c$
- Transformacije:
  - $Y = (p/C(p)) - 1$
  - $X = p-1$
- $Y = uX^2 + (s + u)X$
- $u = a;\ s = b-a$
- $p^* = SQRT((1+a-b)/a)$
- $C(p^*) = p^*/(1 + (b-a)(p^*-1)+ap^*(p^*-1))$

Određivanje parametara

- **Osnovni koraci:**
  1) Izmjeri performanse kao funkciju od *N* koristeći alate kao *WebLoad* ili *LoadRunner*
  2) Obično je dovoljan mali uzorak (najmanje 4 točke)
  3) Odredi a i b primjenom regresije koristeći alat EXCEL ili neku drugu metodu
  4) Koristi dobivene vrijednosti za izračunavanje skaliranja prema modelu iz prethodnog teksta

Uporaba regresije za određivanje funkcije skaliranja



Skaliranje.xls

Prognoza potrebnog kapaciteta

- **Ako se kapacitet posluzitelja C mjeri u pravilnim intervalima, dugoročna potreba za kapacitetom se može ustanoviti podrazumijevajući eksponencijalni trend model**

  - $C_{budući} = C_{sada} * e^{(LT)}$

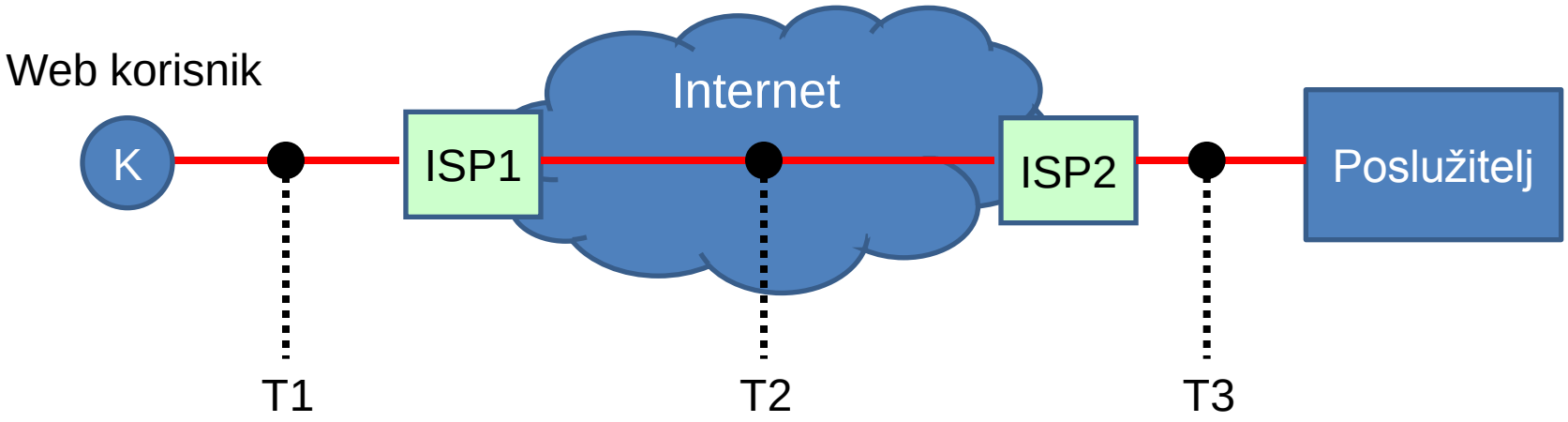
    L = trend rasta, određen primjenom Excel opcije "*Add Trendline*"
    T = vrijeme kroz koje je trend aproksimiran

- **Vrijeme do udvostručenja potrebnog kapaciteta može se izračunati primjenom:**

  - $T_{dvostruko} = \ln(2)/L$

Najčešći načini raspodjeljivanja u praksi

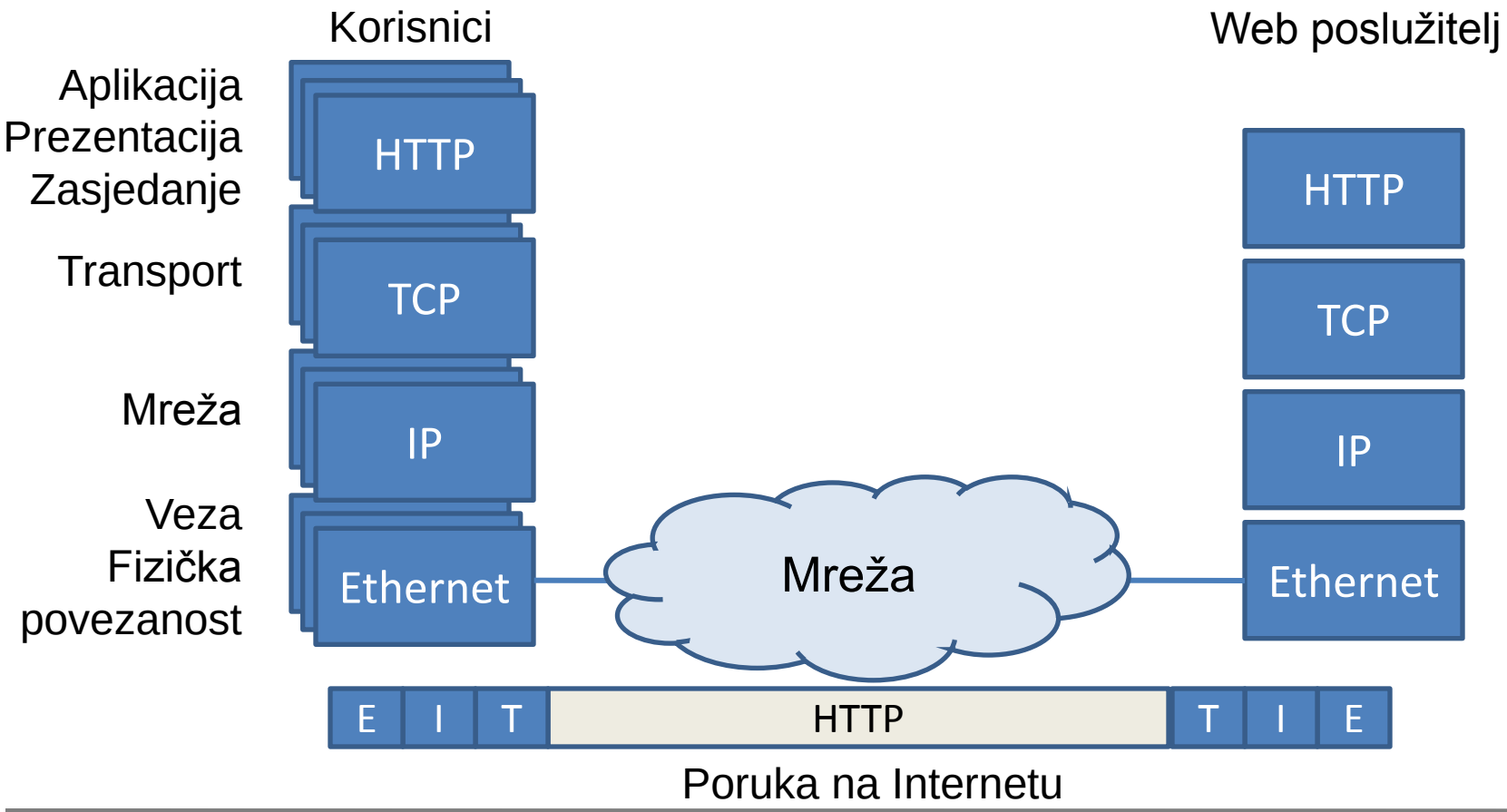
Poslužitelj-korisnik sustavi



- **Pružatelj pristupa mreži Internet (ISP, Internet Service Provider)**
- **Vremena kašnjenja**
  - Korisnik – ISP1 (T1)
  - ISP1 – ISP2 (T2)
  - ISP2 – Poslužitelj (T3)

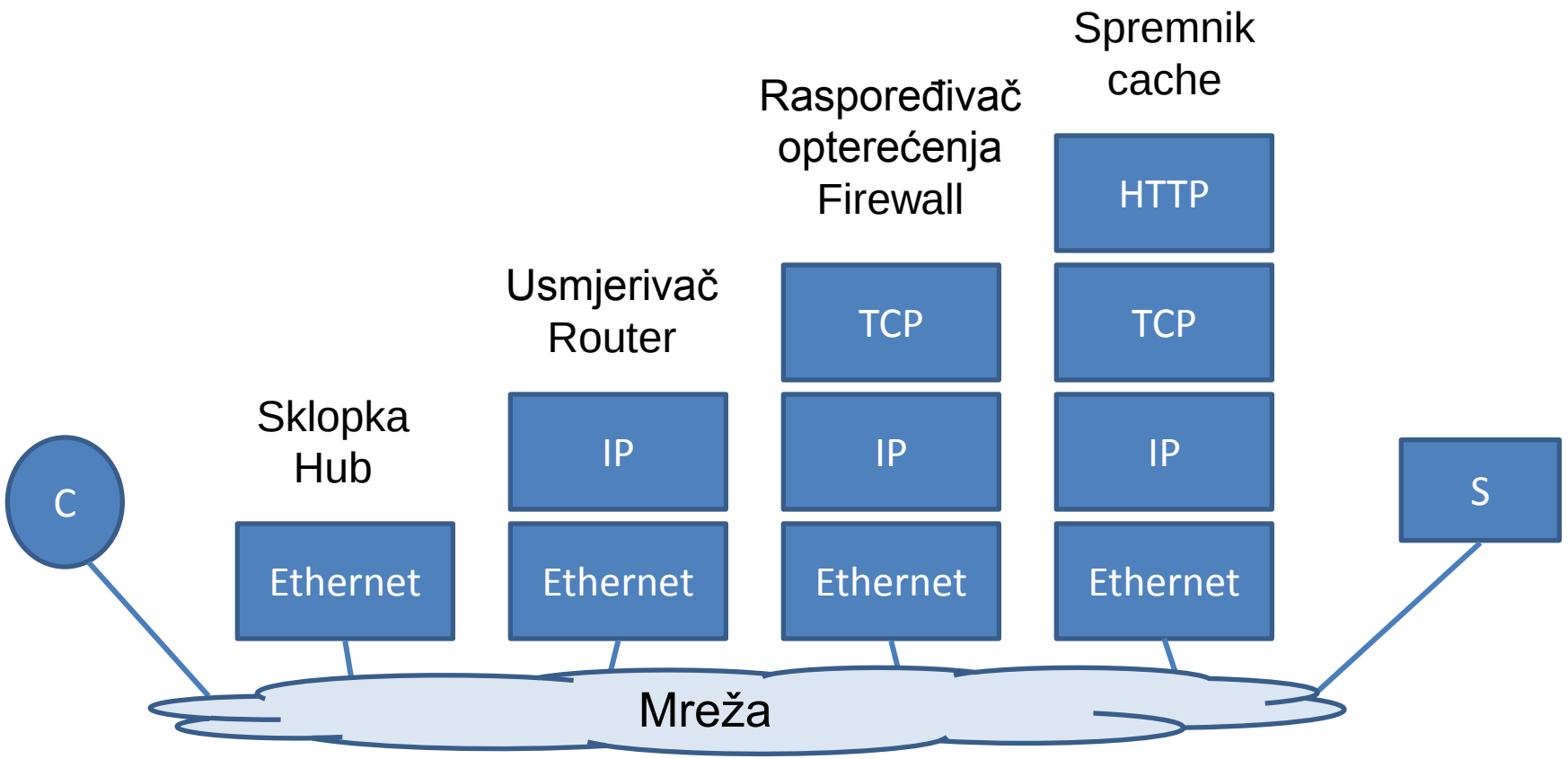
Komunikacijski protokoli u globalnoj mreži Internet

- **Sedam slojeva OSI skupa protokola u praksi se preslikava se na četiri razine protokola u globalnoj mreži Internet**



Komunikacijski protokoli u globalnoj mreži Internet

- **Komunikacijski posrednički sustavi koriste se na različitim razinama protokola**



Skaliranje donosi nove izazove…

- **Raspoređivanje opterećenja u sustavu**
  - Osigurava jednakomjerno opterećenje paralelnih podsustava
- **Raspoređivanje podataka**
  - Osiguravanje koherencije (Replikacija)
  - Osiguravanje podjele (Federacija)
- **Protokoli za sinkronizaciju rada grupe**
  - Osiguravanje vremenskog slijeda
  - Garantirana dostava

…ali i nove mogućnosti

- **Visoka dostupnost sustava** (*high availability*)
  - Uobičajena implementacija kroz neosjetljivost na greške (*fault tolerance*)
    - Neosjetljivost na greške omogućava ostvarivanje dostupnosti Web aplikacija uz prisutnost grešaka u podsustavima
  - Visoki stupanj eliminacije grešaka je preskup (engl. žargon – "platinum tank")
  - Budući da povećanje kapaciteta obično zahtjeva ostvarenje replikacije, replikacija se može iskoristiti za toleranciju pogrešaka

Visoka dostupnost ili neosjetljivost na pogreške

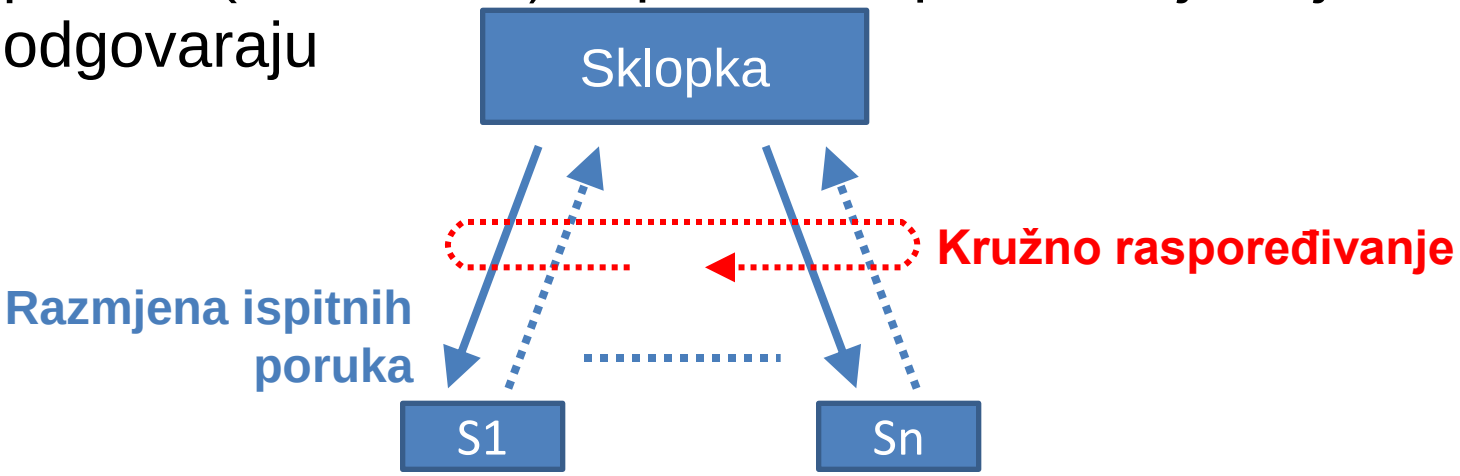
- **U slučaju web aplikacija, u praksi se koriste oba termina jer se visoka dostupnost osigurava kroz neosjetljivost na greške**

- **U praksi se koriste dva modela organizacije sustava**
  - Aktivan → Pripravan (Active → Standby)
  - Aktivan → Aktivan (Active → Active)

- **Sustavi u pripravnom stanju mogu se koristiti kada nije potrebno zapamtiti stanje sustava**
  - Web serveri ne čuvaju stanje dok baze podataka moraju sačuvati sve transakcije

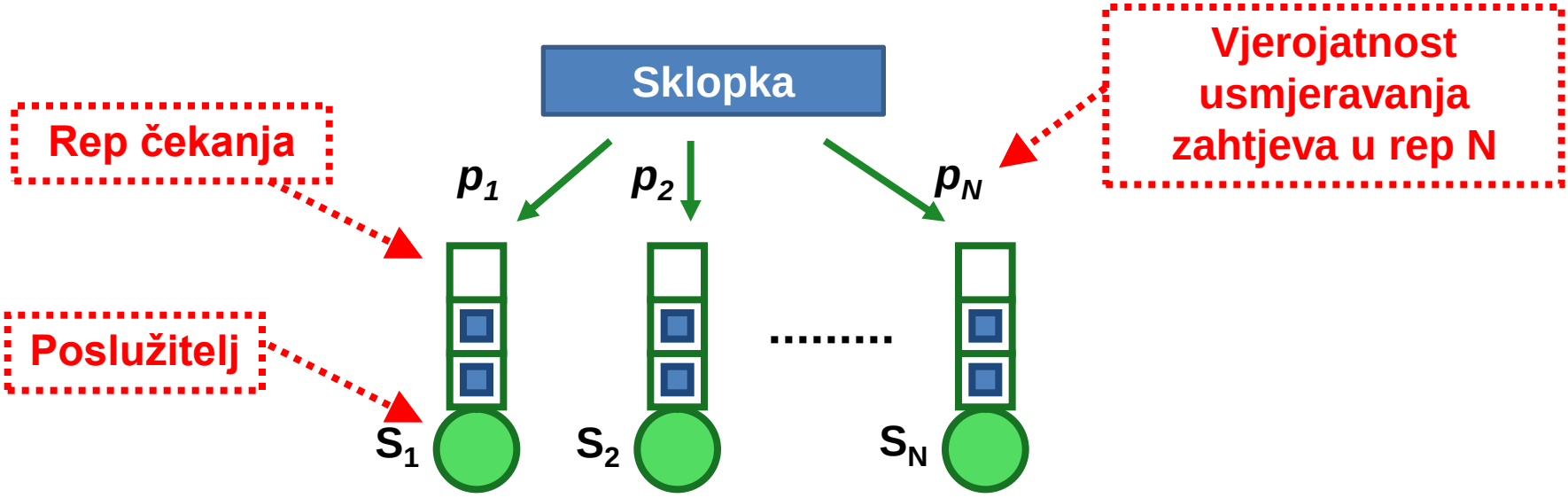
Raspoređivanje opterećenja u sustavu

- **Sklopke za raspoređivanje opterećenja**
  - Najčešće se ostvaruju na razini transportnog protokola (TCP/IP)
  - Najčešće discipline raspoređivanja je kružno posluživanje (*Round Robin*) i najmanjem opterećenju (*Least Loaded First*)
  - Sklopka provjerava stanje poslužitelja slanjem ispitnih poruka (*heart beat*) te preskače poslužitelje koji ne odgovaraju



Model sustava sa sklopkom za raspoređivanje

- **Elementi modela**
  - Sklopka koja prosljeđuje zahtjeve na skup računala
  - Repovi za spremanje dolaznih zahtjeva
  - Poslužitelji koji obrađuju zahtjeve u repu



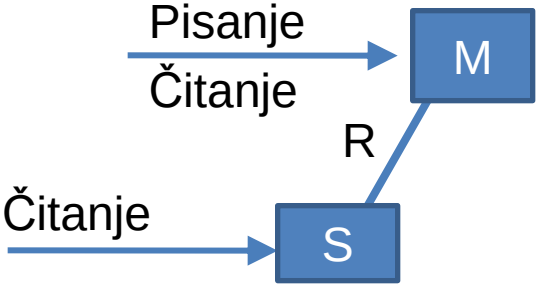
- **Način posluživanja određuje vjerojatnosti raspoređivanja zahtjeva na poslužitelje ($p_1 \ldots p_n$)**

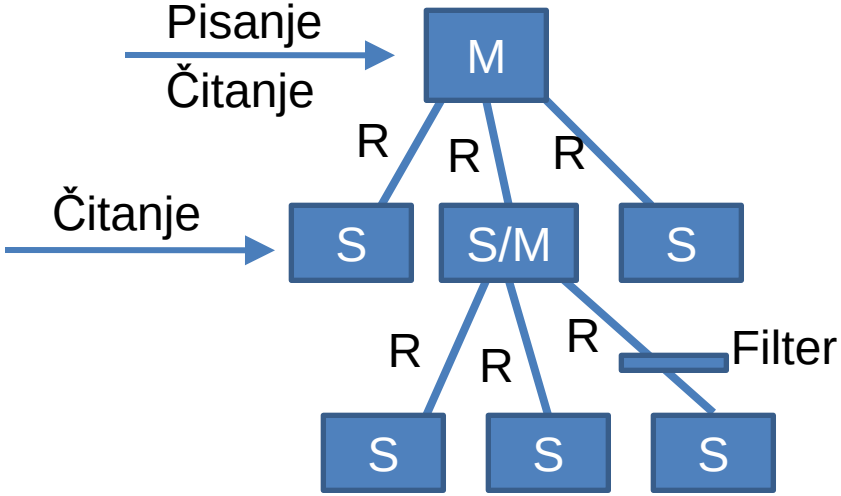
Raspoređivanje opterećenja na aplikacijskoj razini

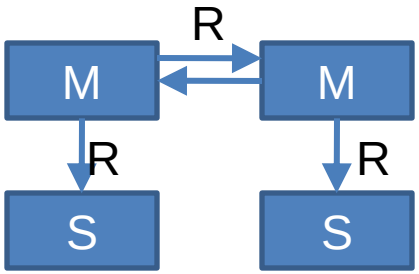
- **HTTP**
  - Obično na razini URL tako da se različite stranice dohvaćaju iz različitih spremnika
- **DNS**
  - Pruža mogućnost geografske podjele
- **SMTP**
  - Elektronička pošta je zasnovana na ASCII protokolu te se može lako prosljeđivati prema ciljnoj adresi

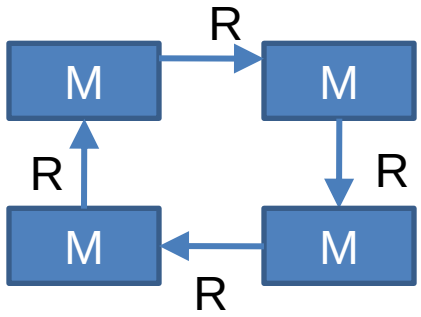
Raspoređivanje podataka

- **Replikacija**
  - Master – slave
  - Master – slaves
  - Stablo
  - Stablo sa filterima
  - Master – master
  - Master – master - slaves
  - Master ring



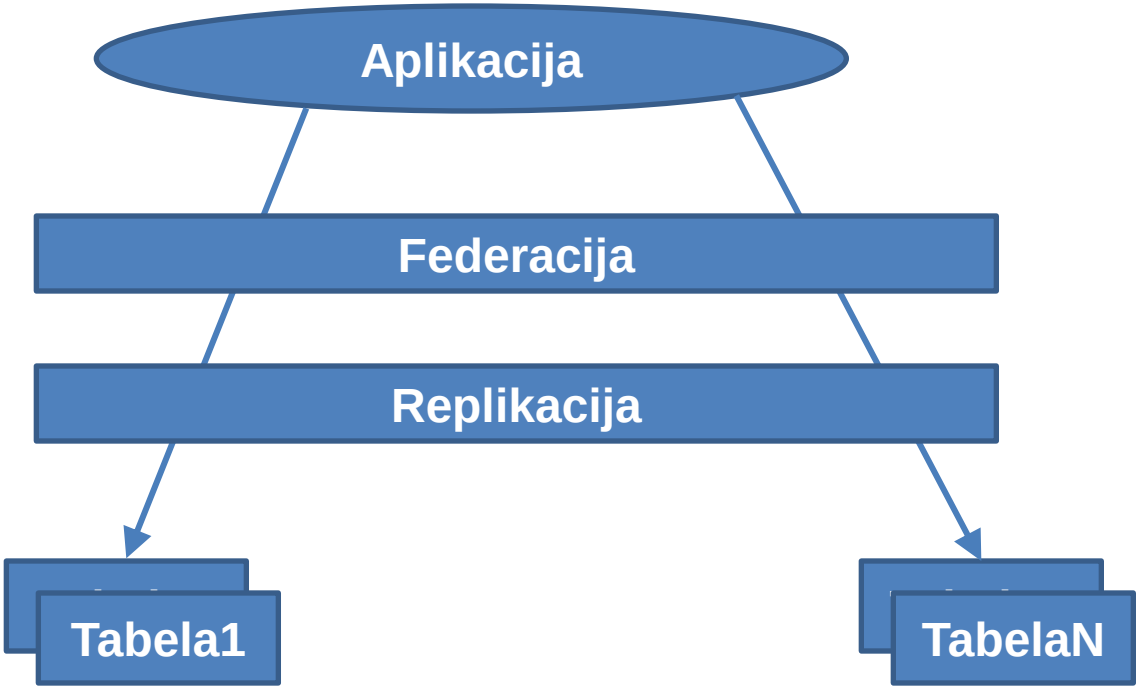






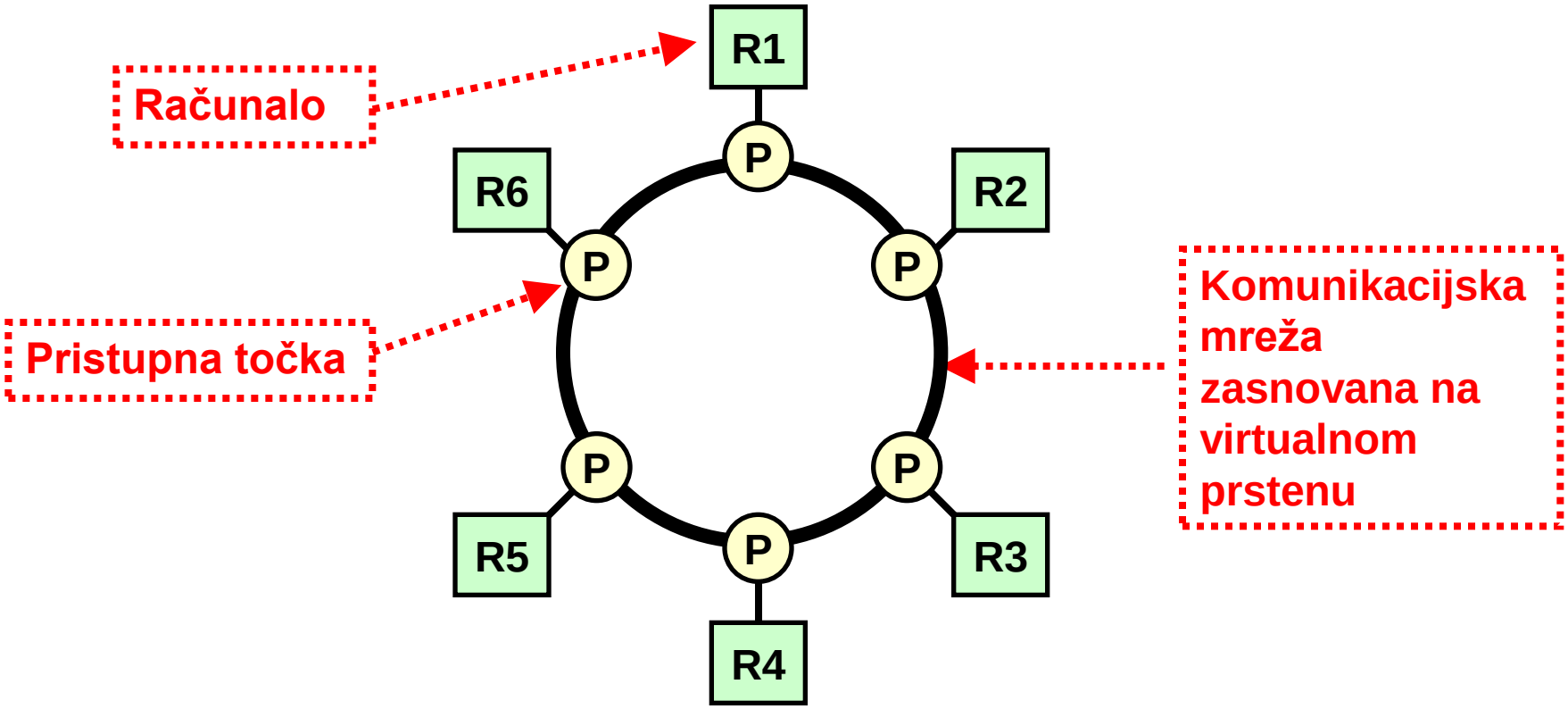
Raspoređivanje podataka

- **Particioniranje**
  - Clustering
  - Federacija
  - Federacija replikacija



Sinkronizacija i protokoli za komunikaciju u grupi

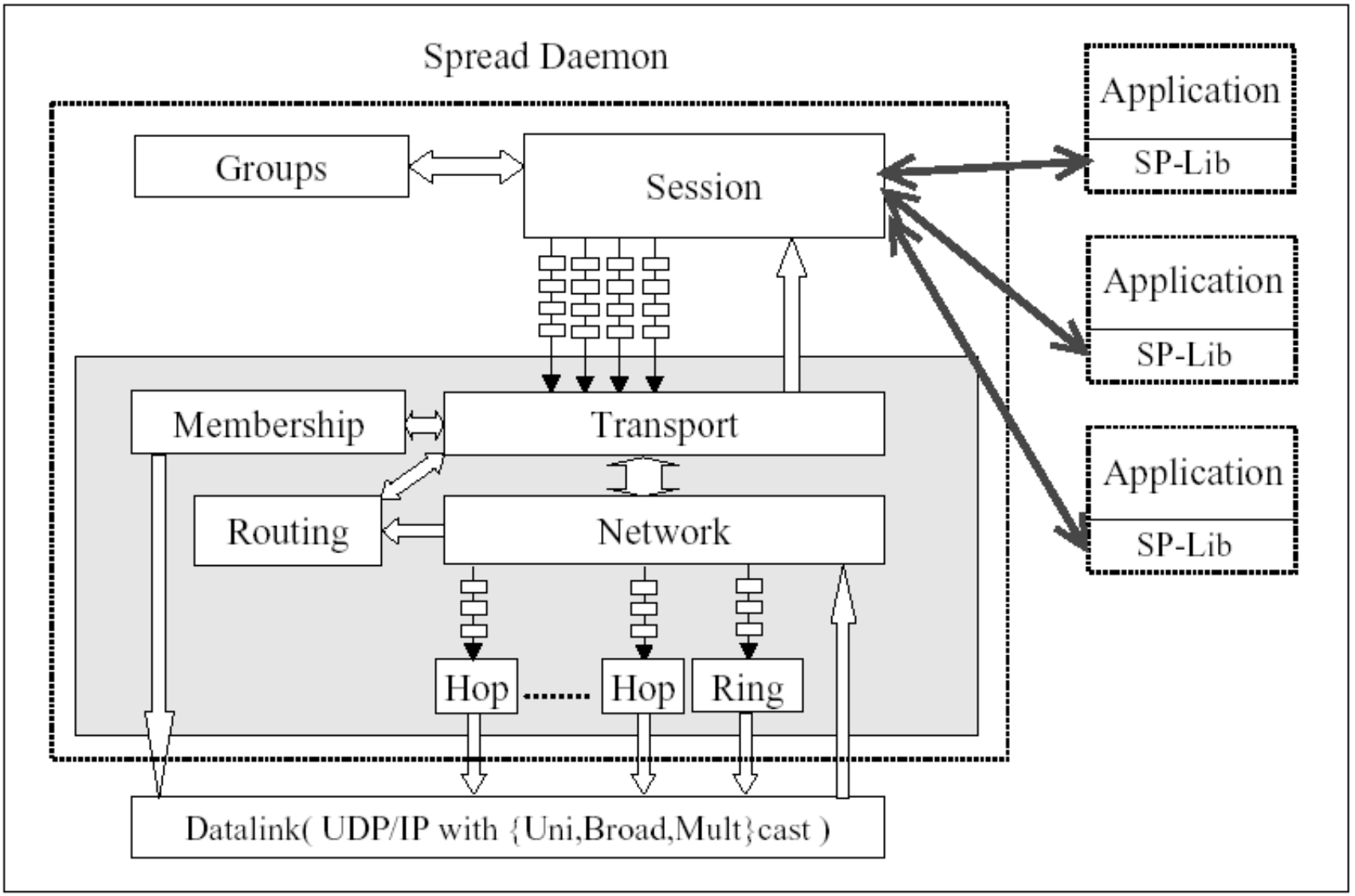
- **Primjer:** Spread - Wide Area Multicast and Group Communication Toolkit
  - Center for Networking and Distributed Systems, John Hopkins University



Garancije slijeda poruka u protokolima

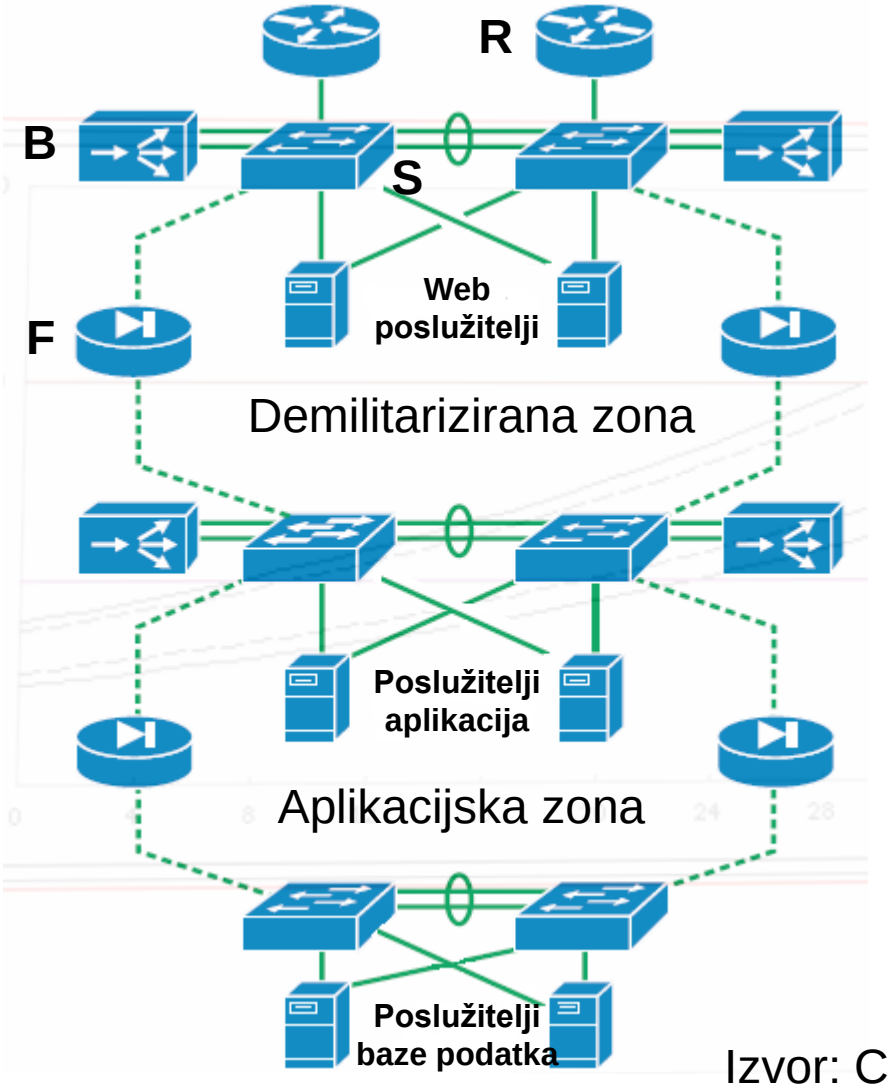
- **Redoslijed prispijeća** (*FIFO*)
  - Proces X šalje poruke A i B u slijedu
  - Svi procesi dobivaju poruku A prije poruke B
- **Međuzavisnost** (*Casual*)
  - Proces X šalje poruku B nakon prijema poruke A
  - Svi procesi dobivaju poruku B nakon poruke A
- **Potpuna garancija** (*Total*)
  - Proces X prima poruke A i B u slijedu
  - Svi procesi primaju poruke A i B u istom slijedu

Arhitektura čvora



Tipična arhitektura web portala

- **Usmjernik (R)**
  Router
- **Sklopka (S)**
  Request Switch
- **Raspoređivač (B)**
  Load balancer
- **Zaštitnik pristupa (F)**
  Firewall

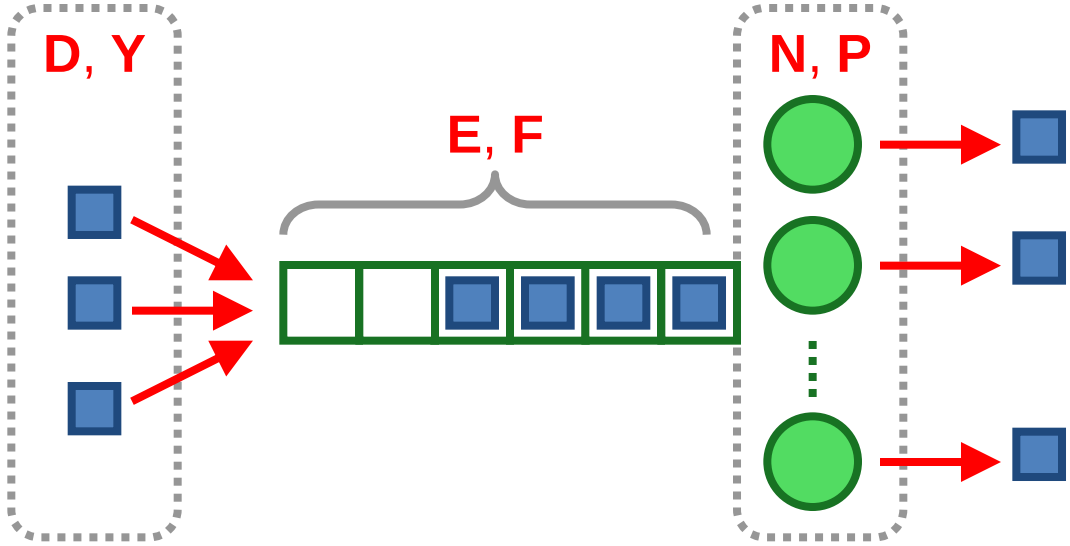


Izvor: Cisco

Vrednovanje performansi sustava modelom repova

Osnovni pojmovi teorije repova

- **Zapis značajki modela D/P/N/Y/E/F (*Kendall* notacija)**



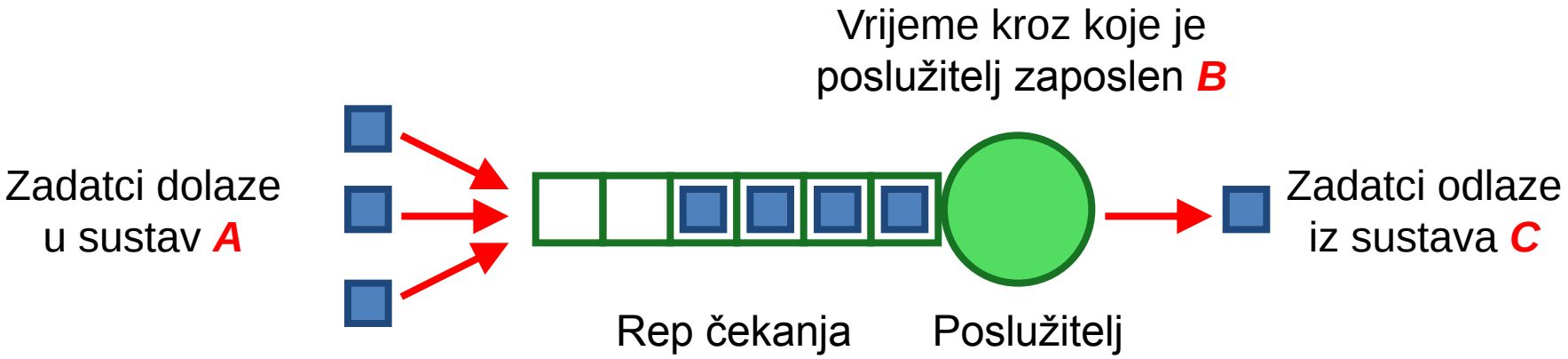
- ***D*** – Razdioba dolazaka zahtjeva (G=Proizvoljna, M=Exp, …)
- ***P*** – Razdioba posluživanja zahtjeva (G=Proizvoljna, M =Exp, …)
- ***N*** – Broj poslužitelja
- ***Y***– Maksimalni broj zahtjeva (m zahtjeva ili ∞ zahtjeva)
- ***E*** – Maksimalni kapacitet repa (m ćelija ili ∞ ćelija)
- ***F*** – Disciplina posluživanja (FIFO, LIFO, …)

Utvrdivanje parametara modela

- **Model crne kutije ima dva ulazna parametra:**
  - Teret sustava karakteriziran prosječnim ulaznim ritmom i raspodjelom zahtjeva
  - Prosječno vrijeme posluživanja za jedan zahtjev (bez čekanja) te pripadna raspodjela
- **Na temelju dva gornja parametra mogu se izračunati:**
  - Prosjecni broj zahtjeva u sustavu
  - Prosječno vrijeme odziva sustava
  - Prosječna iskoristivost sustava

Model repa

- **Model jednoprocesorskog sustava M/M/1**
  - Poslužitelj i rep čekanja



- **Osnovne značajke modela**
  - ***T*** – Ukupno vrijeme promatranja rada sustava
  - ***A*** – Broj dolazaka zadataka u vremenu *T*
  - ***B*** – Vrijeme kroz koje je poslužitelj zaposlen u vremenu *T*
  - ***C*** – Broj odlazaka u vremenu *T*

Osnovni pojmovi teorije repova

- **Učestalost dolazaka zadataka** (L [zad/s])
  - L = A / T
- **Propusnost sustava** (X [zad / s])
  - X = C / T
- **Srednje vrijeme posluživanja** (S [s / zad])
  - S = B / C
- **Srednja zaposlenost poslužitelja** (U [])
  - U = B / T
  - U = (B / T) * (C / C) = (B / C) * (C / T) = S * X

Primjer 1: Posluživanje zahtjeva na disku

- **Disk za trajno spremanje podataka ispunjava 50 zahtjeva u sekundi. Srednje vrijeme obrade zahtjeva operacija pisanja i čitanja je 10 ms.**
  - Kolika je prosječna zaposlenost diska?

- **Rješenje**
  - Propusnost sustava X = 50 z/s
  - Srednje vrijeme obrade zahtjeva S = 10 ms/z

  - Prosječna zaposlenost diska U

    U = X * S = 50 z/s * 0.01 s/z = 0.5 (50 %)

pr1.c

Little-ov zakon

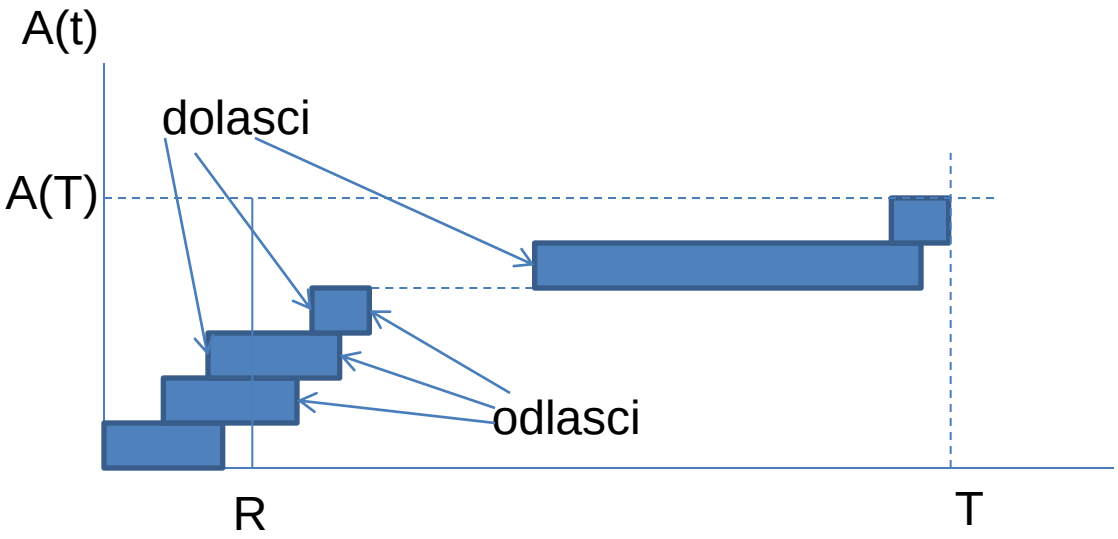
- **Broj zahtjeva u repu jednak je umnošku učestalosti dolazaka zahtjeva u rep i prosječnog vremena zadržavanja zahtjeva u sustavu**
  - L [z/s] – Učestalost dolazaka zahtjeva u rep zahtjeva
  - R [s] – Prosječno vrijeme zadržavanja zahtjeva u sustavu
  - Q [z] – Broj zahtjeva u repu

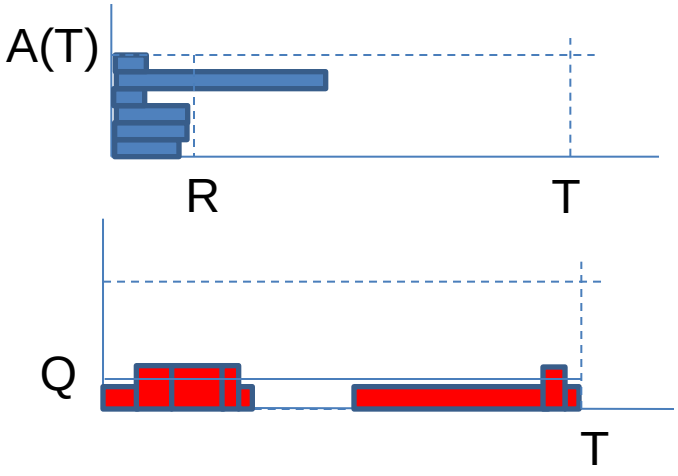
$$Q = L * R$$

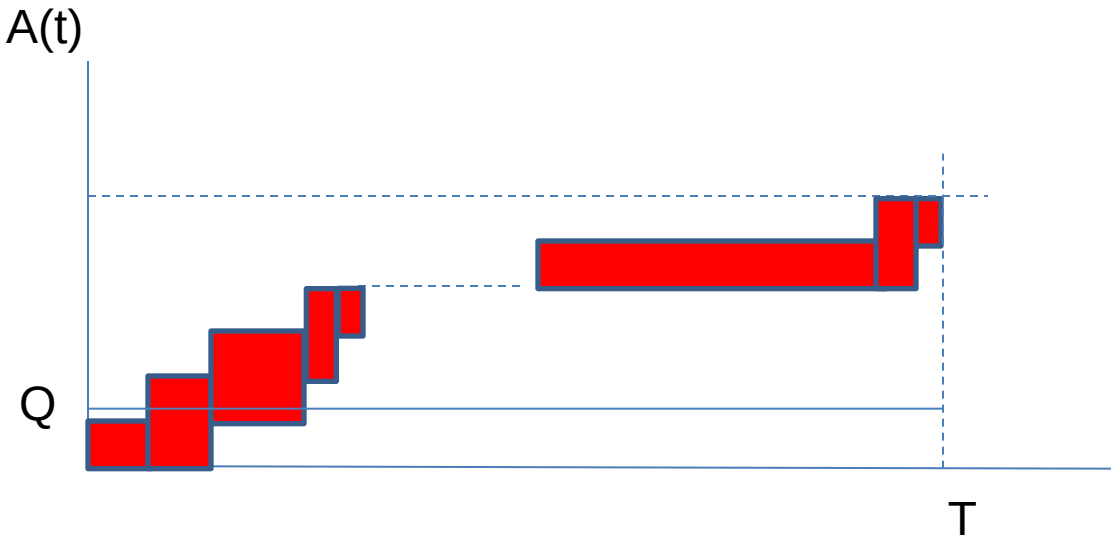
- **Ako je sustav stabilan**
  - Broj prispjelih zahtjeva u vremenu jednak je broju zahtjeva koji napuštaju sustav (L = X)
  - Napomena: sve veličine su srednje vrijednosti!

$$Q = X * R$$

Dokaz Little-ovog zakona







Plava površina = A(T) * R
Crvena površina = Q * T

A(T) * R = Q * T
A(T) * R * T/T = Q * T
R * A(T)/T = Q

$$R * L = Q$$

Primjer 2: Čekanje na posluživanje zahtjeva s diska

- **Disk iz prethodnog slučaja ima prosječno 1 zahtjev u repu**
  - Koliko je prosječno vrijeme čekanja na obradu zahtjeva ?
- **Rješenje**
  - Ulazni ritam zahtjeva L = 50 z/s
  - Broj zahtjeva u repu Q = 1 z

  - Vrijeme zadržavanja zahtjeva u sustavu R

    R = Q/L = (1 z) / (50 z/s) = 20 ms
  - Vrijeme zadržavanja uključuje vrijeme čekanja u repu (W) i vrijeme obrade zahtjeva (S): R = W + S
  - Vrijeme čekanja na obradu W

    W = R – S = 20 ms – 10 ms = 10 ms

pr2.c

Ukupno vrijeme čekanja u sustavu

- **Ukupno vrijeme zadržavanja zahtjeva u sustavu (R)**
  - Vrijeme obrade svih Q zahtjeva u repu ispred novog zahtjeva uvećano za vrijeme obrade novog zahtjeva
  - R = S + W = S + S*Q
- **U stabilnom stanju, primjenom Littleovog Q=X*R**
  - R = S + S*X*R, R(1 – X*S) = S, R = S/(1 – X*S)
- **Primjenom supstitucije X*S = U**
  - R = S/(1 – U)
- **Množenje obje strane sa X ->** [X*R = (X*S)/(1 – U)]
  - Q = U/(1 – U)
- **Množenje obje strane sa S ->** [S*Q = (S*U)/(1 – U)]
  - W = (S*U)/(1 – U)

Primjer 3: Komunikacijski kanal

- **Mjerenjem na pristupnoj točki mreže dobivamo srednji protok od 125 paketa u sekundi i srednje vrijeme posluživanja 0.002 sekunde.**
  - Što je sve moguće zaključiti o promatranom kanalu ?
- **Rješenje**
  - Srednji protok X = 125 p/s
  - Srednje vrijeme posluživanja S = 0.002 s/p
  - Prosječna zaposlenost komunikacijskog sustava U

    U = X * S = (125 p/s) * (0.002 s/p) = 0.25 (25 %)
  - Srednje vrijeme zadržavanja paketa u sustavu (R)

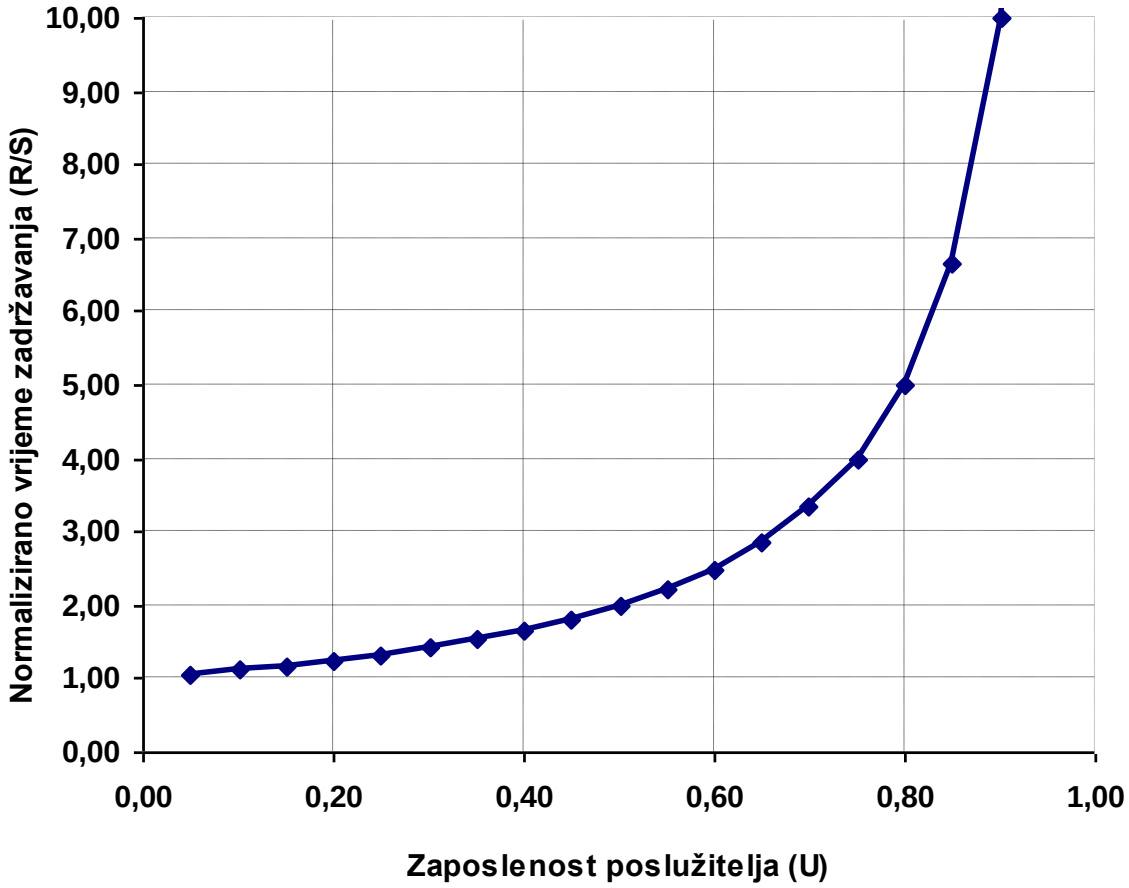
    R = S/(1 - U) = (0.002 s/p)/(1 - 0.25) = 0.0026666 s
  - Srednji broj paketa u repu (Q)

    Q = X*R = (125 p/s)*(0.0026 s) = 0.333 p

pr3.c

Odziv sustava s repovima je izrazito nelinearan

- **Graf normilizirano cekanje (R/S) kao funkcija optrerecenja poslužitelja U**



R/S = 1/(1-U)

Odziv.xls

Primjer 4: Vrijeme čekanja i broj zahtjeva

- **Sustav ima prosječno vrijeme posluživanja 1 sekunda i učestalost dolazaka zahtjeva je 0.5 zadatka u sekundi.**
  - Kolika je srednja vrijednost ukupnog vremena čekanja (R) i srednja vrijednost broja zahtjeva u repu (Q)?
- **Rješenje**
  - Prosječno vrijeme posluživanja S = 1 s/z
  - Učestalost pristiglih zahtjeva L = 0.5 z/s

  - Prosječna zaposlenost sustava U

    U = S * L = (1 s/z) * (0.5 z/s) = 0.5 (50 %)
  - Srednje vrijeme zadržavanja paketa u sustavu (R)

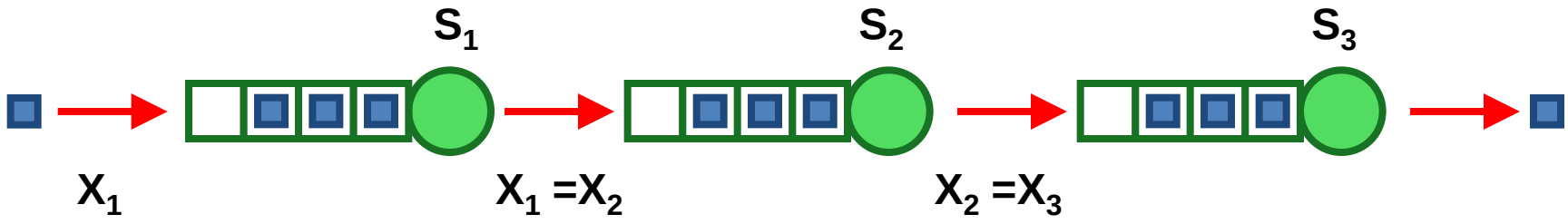
    R = S / (1 - U) = (1) /(1 - 0.5) = 2 s
  - Srednja vrijednost broja zahtjeva u sustavu (Q)

    Q = U / (1 - U) = 0.5/(1 - 0.5) = 1 z

pr4.c

Serijski repovi

- **Little-ov zakon**
  - Q = L * (R1 + R2 + R3)
- **U stabilnom stanju sustava** (L = X i X1 =X2 = X3 = X)
  - Q = X * R = X * (R1 + R2 + R3)
- **Uz Uz $R_N = S_N / (1 – X*S_N)$**
  - Q = X((S_1/(1 – X*S_1)) + (S_2/(1 – X*S_2)) + (S_3/(1 – X*S_3))
  - R = R1 + R2 + R3



Primjer 5: Posluživanje u seriji

- **Sustav sadrži 3 serijske procesne jedinice s prosječnim vremenima posluživanja 1 s, 2 s i 3 s.**
  - Koliko će biti vrijeme zadržavanja u sustavu uz ulazni ritam zahtjeva od 0.1 z/s ?
  - Koliki će biti prosječni broj zahtjeva u sustavu ?
- **Rješenje**
  - Prosječna vremena posluživanja $S_1 = 1$ s/z, $S_2 = 2$ s/z, $S_3 = 3$ s/z
  - Propusnost sustava $X = 0.1$ z/s

  - Vremena zadržavanja $R_N = S_N /(1 – X*S_N)$

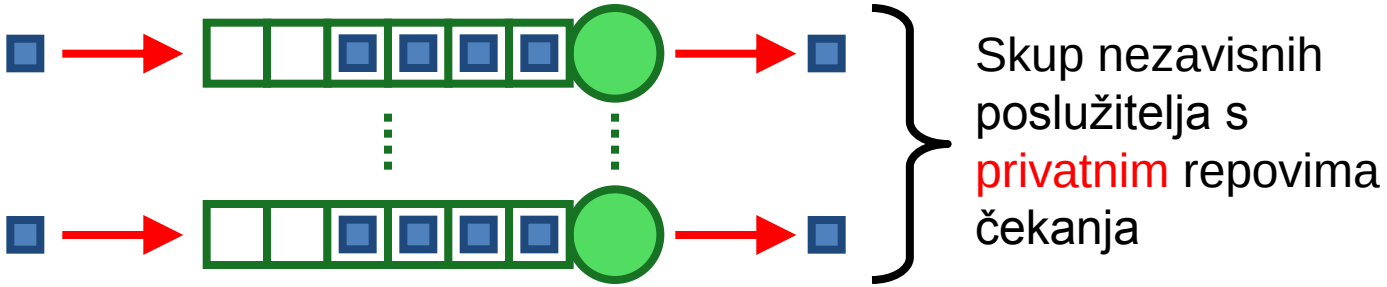
    $R_1 = 1.11$s, $R_2 = 2.5$s, $R_3 = 4.29$s
  - Prosječni broj zahtjeva u repu Q

    $Q = X * (R_1 + R_2 + R_3) = 0.1$ z/s $* (1.11 + 2.5 + 4.29) = 0.79$ z

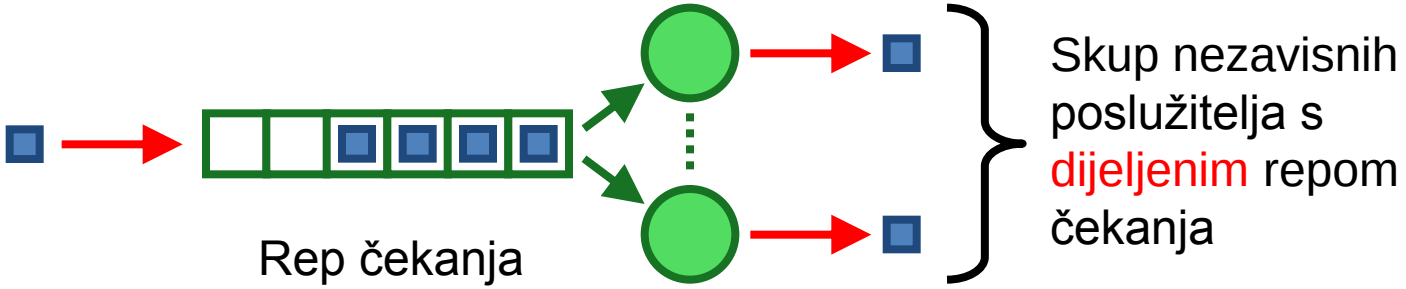
pr5.c

Višestruki paralelni repovi i poslužitelji

- **Multi-računalo**
  - Model koji se primjenjuje u supermarketima



- **Multi-procesor**
  - Model koji se primjenjuje u bankama



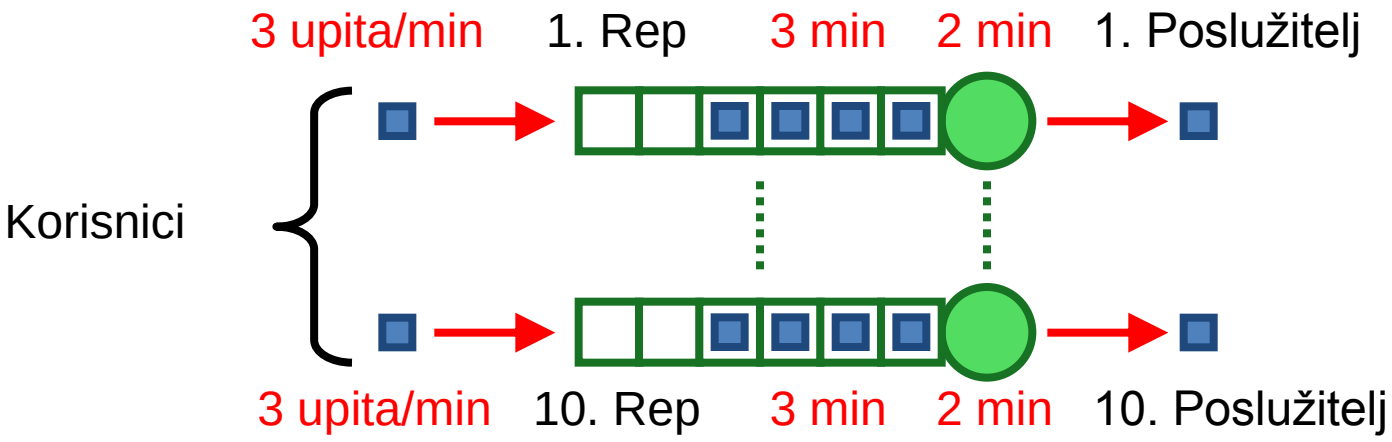
Sustav više paralelnih repova čekanja

- **Ukupno vrijeme čekanja (R) za dva repa**
  - R = S + (S * (0.5 * Q))
- **Primjenom Little-ovog zakona Q=X*R**
  - R = S/(1 – (0.5*X*S))
- **Primjenom supstitucije X*S = U**
  - R = S/(1 – 0.5 * U)
- **Ukupna zaposlenost U sustava podijeljena s brojem repova N je faktor iskorištenja ro koji predstavlja vjerojatnost da je poslužitelj zaposlen**
  - ro = U/N, R = S / (1 - ro)
  - Za beskonačno mnogo repova

    $N \to \infty$; $ro \to 0$; $R \to S$ (Nema čekanja na posluživanje)

Primjer 6: Aplikacija korisničke podrške

- **Web aplikacija uključuje podršku korisnicima putem *chat* usluge. Kupci sami odabiru jedan od 10 repova čekanja. Mjerenja pokazuju da zahtjevi prosječno dolaze 3 upita u minuti te da svaki kupac prosječno čeka 3 minute u repu i prosječno provodi 2 minute u konverzaciji.**
  - Koliko bi dodatnih tehničara trebalo zaposliti da se prosječno vrijeme čekanja svede na 1 minutu ?



pr6.c

Primjer 6: Aplikacija korisničke podrške

- **Rješenje**
  - Prosječno vrijeme posluživanja S = 2 min/z
  - Ritam pristiglih zahtjeva L = 3 z/min

  - Prosječna zaposlenost sustava (U)

    U = S L = (2 min/z) (3 z/min) = 6
  - Faktor iskorištenja (ro)

    ro = U/N = 6/10 = 0.6
  - Srednje vrijeme zadržavanja korisnika u sustavu (R)

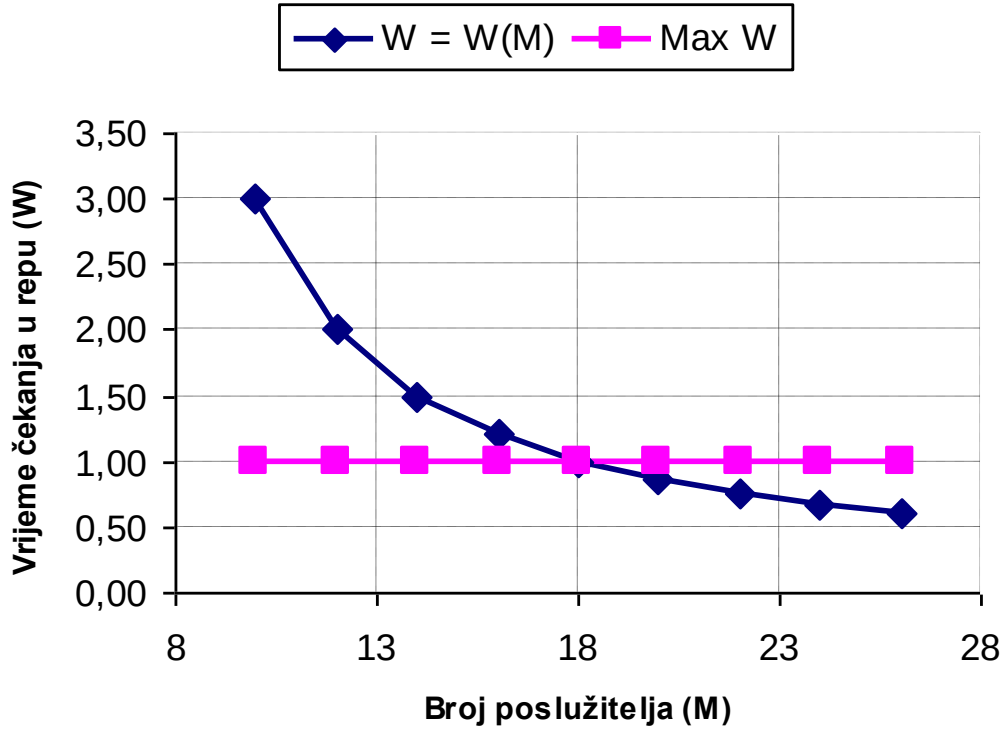
    R = S / (1 - ro) = 2 / (1 – 0.6) = 5 min
  - Srednje vrijeme čekanja u repu (W)

    W = R - S = 5 min – 2 min = 3 min

Primjer 6: Aplikacija korisničke podrške

Rješenje za broj tehničara

- Rješenje se moze odrediti primjenom numeričkih metoda ili primjenom metode pokušaja i promašaja.



- Kao rješenje dobije se da je potrebno 18 tehničara

pr5.xls

Primjer 6: Aplikacija korisničke podrške

- **Rješenje**
  - Broj poslužitelja (tehničara) N = 18
  - Prosječno vrijeme posluživanja S = 2 min/z
  - Propusnost sustava X = 3 z/min

  - Prosječna zaposlenost sustava U

    U = X * S = (3 z/min) * (2 min/z) = 6
  - Faktor iskorištenja ro

    ro = U/N = 6/18 = 0.33
  - Srednje vrijeme zadržavanja korisnika u sustavu (R)

    R = S / (1 - ro) = 2 / (1 – 0.33) = 2.985 min
  - Srednje vrijeme čekanja u repu (W)

    W = 2.985 – 2 = 0.985 min

Primjer 6: Vrijeme čekanja i broj zahtjeva

- **Zadatci za vježbu**
  - **Zadatak 1:** Kakvi će biti odzivi sa 10 i 18 tehničara ako publiciranje Web stranice sa odgovorima na najčešća pitanja smanji broj upita na 2 u minuti?
  - **Zadatak 2:** Kakve će rezultate dati smanjenje razgovora na 1.5 minutu?

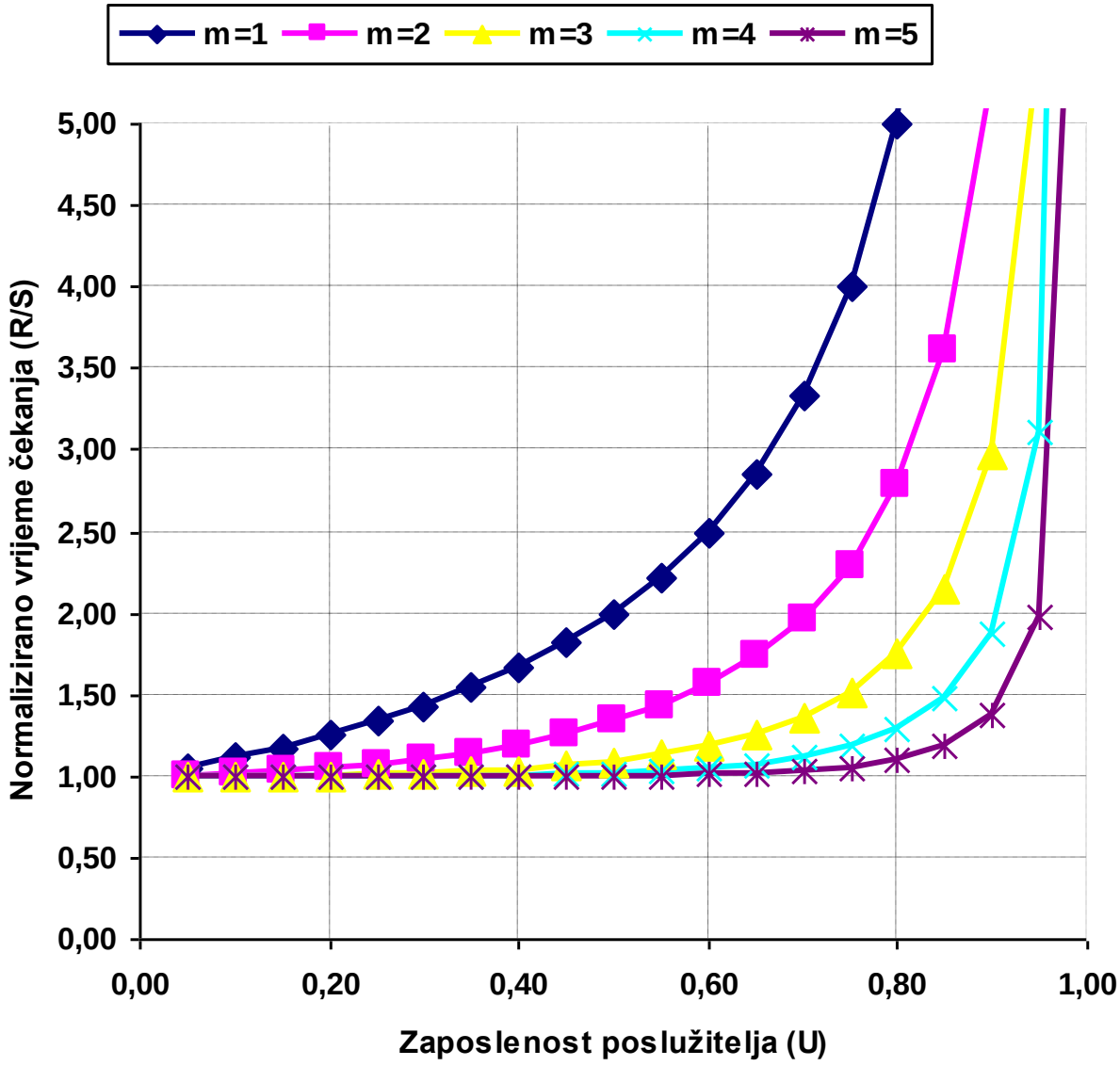
Sustav s jednim repom i dva poslužitelja

- **Efektivno vrijeme posluživanja ovisi o dva čimbenika**
  - Dodatni poslužitelj smanjuje vrijeme posluživanja za faktor 0.5
  - Vrijeme posluživanja se množi sa vjerojatnošću da je poslužitelj zaposlen ro = U/2
- **Zbog navedenog vrijedi**
  - S(ro) = (0.5*S)*ro
  - R = S + Q * S(ro) = S + 0.5*S*ro*Q
- **Primjenom supstitucije Q = X*R**
  - R = S + (0.5*S*ro*X*R)
- **Primjenom supstitucije 0.5*S*X = 0.5*U = ro**
  - $R = S + R * ro^2$
  - $R = S/(1 - ro^2)$ -> množenjem sa X i supstitucijom S*X = 2*ro
  - $Q = 2*ro/(1 - ro^2)$

Poopćenje na sustav s N paralelnih poslužitelja

- **Za sustav s N paralelnih poslužitelja vrijedi**
  - $ro = U/N$
  - $R = S + Q*(S/N)*(ro^{(N-1)})$ (aproksimacija)
- **Primjenom supstitucija $Q=X*R$ i $S=U/X$**
  - $R = S + (X*R) * (U/(X*N)) * (ro^{(N-1)})$
  - $R = S + R * (U/N) * (ro^{(N-1)})$ uz $ro = U/N$
  - $R = S + R * ro^N$
  - $R * (1 - ro^N) = S$
  - $R = S/(1 - ro^N)$ množenjem sa X
  - $X * R = X * S /(1 - ro^N)$ uz $Q = X*R$ i $U = XS$
  - $Q = U /(1 - ro^N)$ uz $U = N*ro$
  - $Q = (N * ro)/(1 - ro^N)$

Usporedba sustava M/M/m



Erlang.xls

Erlangova formula i egzaktno rješenje sustava

- **Erlangova formula**
  - Analitičko rješenje za vrijeme zadržavanja R u sustavu s N paralelnih poslužitelja

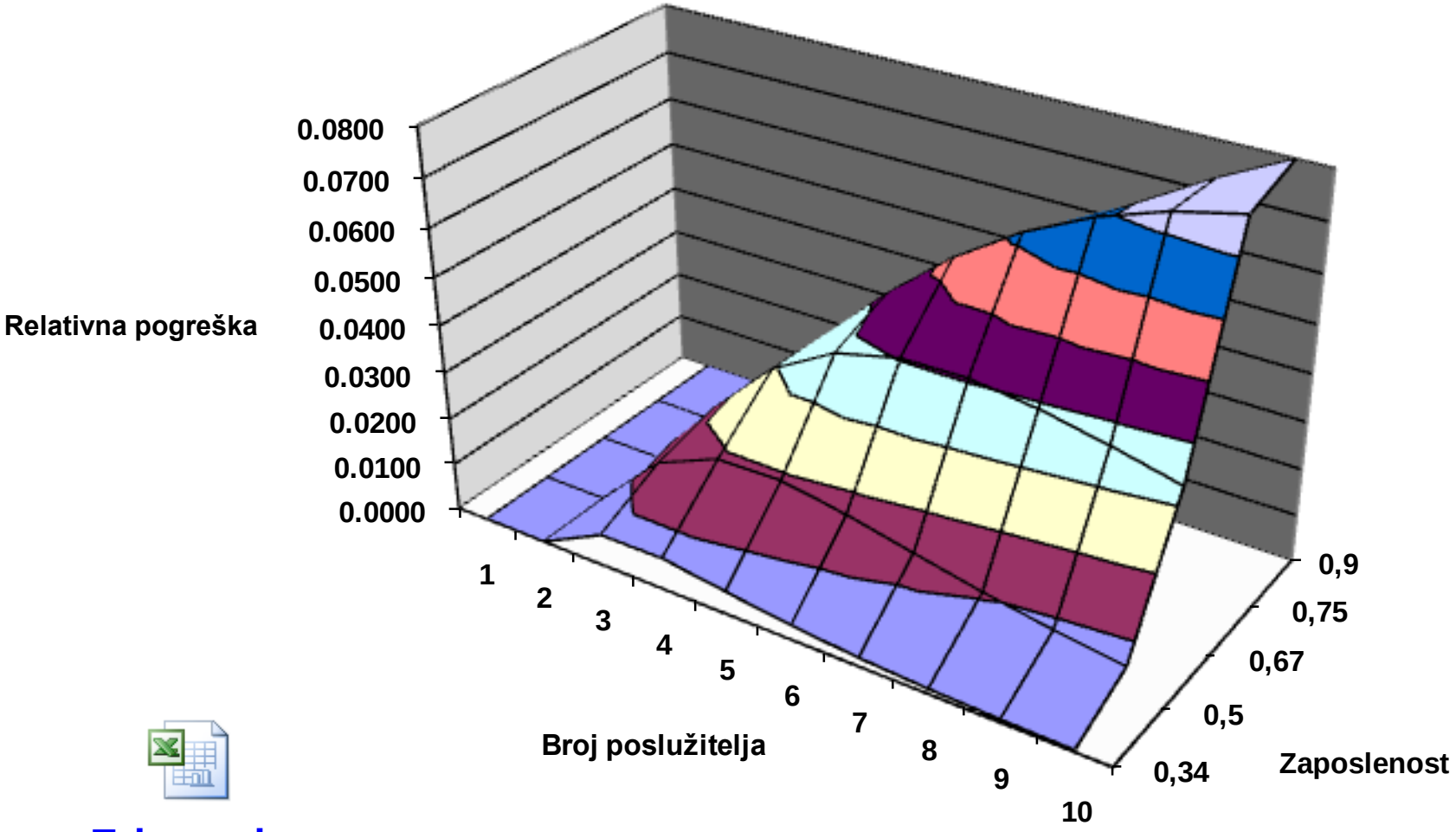
$$R = S * \left[1 + \frac{C(N, ro)}{N * (1 - ro)} \right]$$

  - Koeficijent C(*N*, *ro*)

$$C(N, ro) = \frac{\frac{(N * ro)^N}{N!}}{(1 - ro) * \sum_{k=0}^{N-1} \frac{(N * ro)^k}{k!} + \frac{(N * ro)^N}{N!}}$$

Aproksimacija i egzaktno rješenje sustava

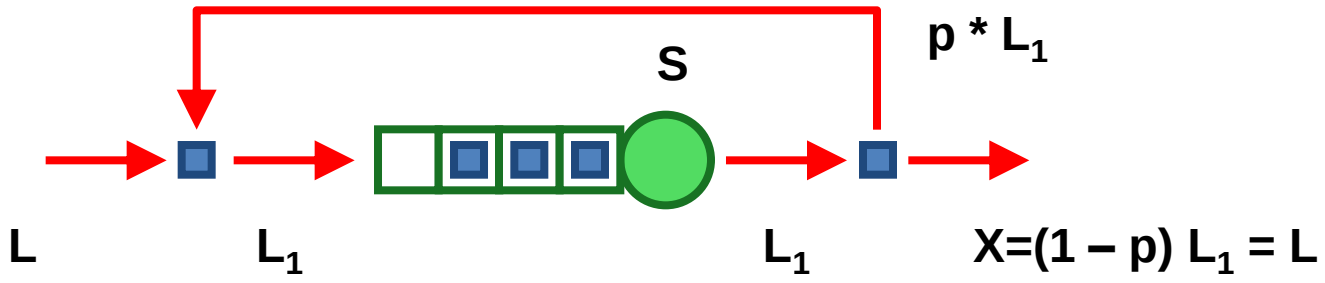
Pogreška aproksimacije



Erlang.xls

Poslužitelji s povratnom vezom

- **Dio dolaznih zahtjeva nakon posluživanja ponovno se vraća u rep za čekanje**

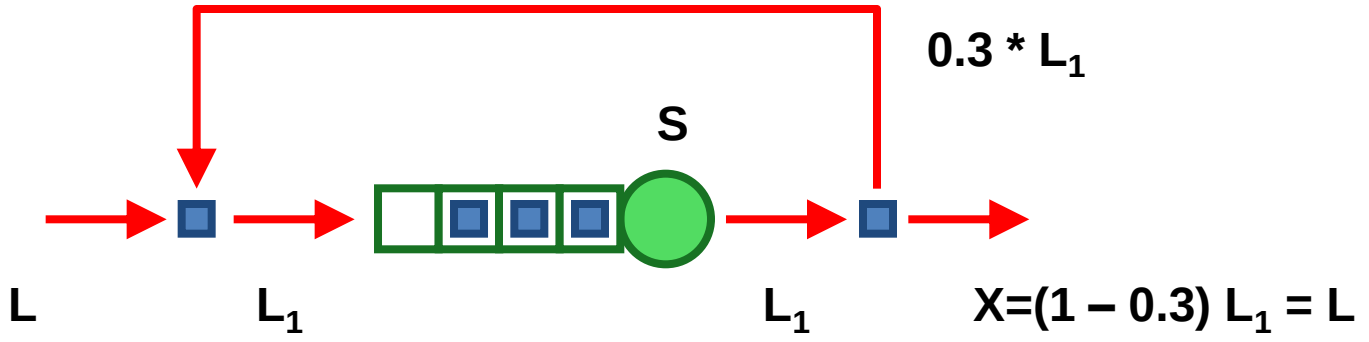


- Dva Poissonova procesa L1 i L2 rezultiraju u sa novim Poissonovim procesom L = L1 + L2
- $L_1 = L + L_1 p = L / (1 - p)$
- $U = L_1 * S = L*S/(1 - p)$
- $R_1 = S/(1 – U)$ (zadržavanje za jedan prolaz)
- Vrijeme zadržavanja u sustavu s povratnom vezom

  $R = R_1 * (1 + (p/(1 – p))) = R_1/(1 – p)$

Primjer 7: Komunikacijski kanal s pogreškom

- **Paketi dolaze u komunikacijski kanal s učestalošću 0.5 paketa u sekundi i zahtijevaju 0.75 sekundi za obradu. Za 30 % paketa dogodi se pogreška pri prijenosu i takvi paketi se umeću u rep za ponovno slanje.**
  - Koliko vremena paket prosječno provede u kanalu ?



pr7.c

Primjer 7: Komunikacijski kanal s pogreškom

- **Rješenje**
  - Broj pristiglih paketa u sekundi L = 0.5 p/s
  - Prosječno vrijeme obrade paketa S = 0.75 s/p
  - Vjerojatnost pogreške paketa pri prijenosu p = 0.3

  - $L_1 = L / (1 - p) = 0.5 / 0.7 = 0.714$ p/s
  - Prosječna zaposlenost kanala U

    $U = L_1 * S = 0.714$ p/s * 0.75 s/p = 0.536 (53.6 %)
  - Srednje vrijeme čekanja u repu W

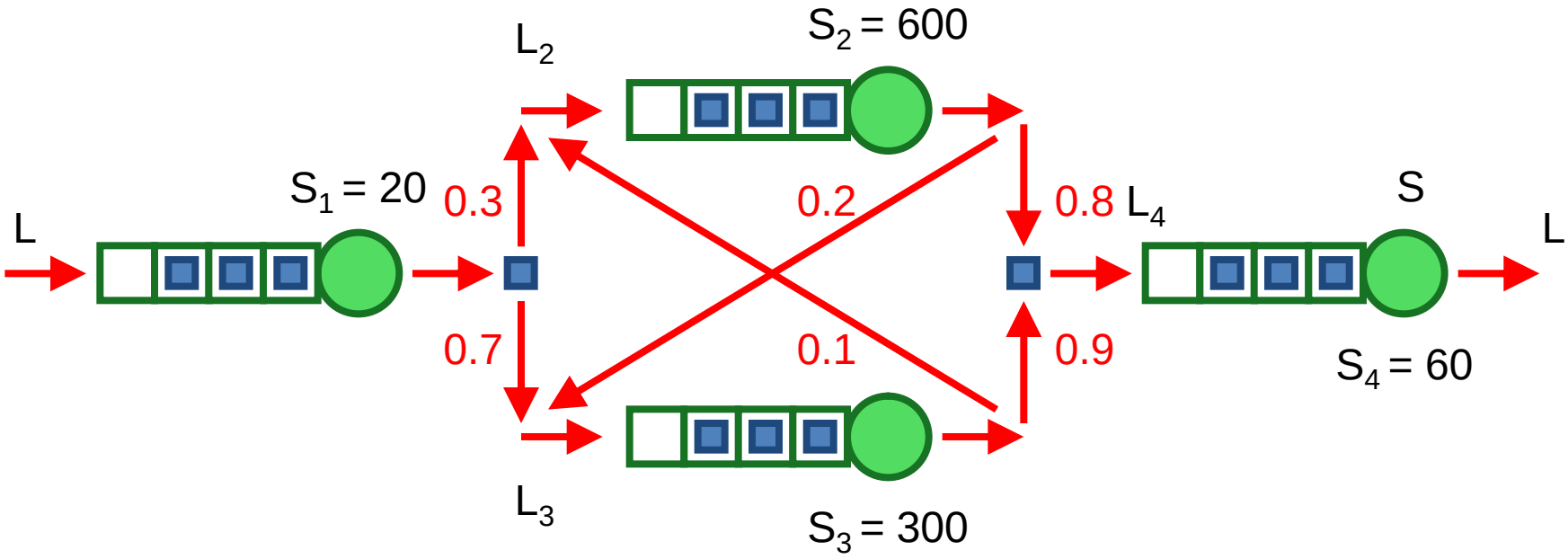
    $W = S*U / (1 – U) = 0.866$ s/p
  - Srednje vrijeme zadržavanja paketa u kanalu (R1)

    R1 = W + S = 0.866 s/p + 0.75 s/p = 1.616 s/p

Prosječno vrijeme u kanalu: R = R1 /(1-p) = 2.31

Višestruke povratne veze

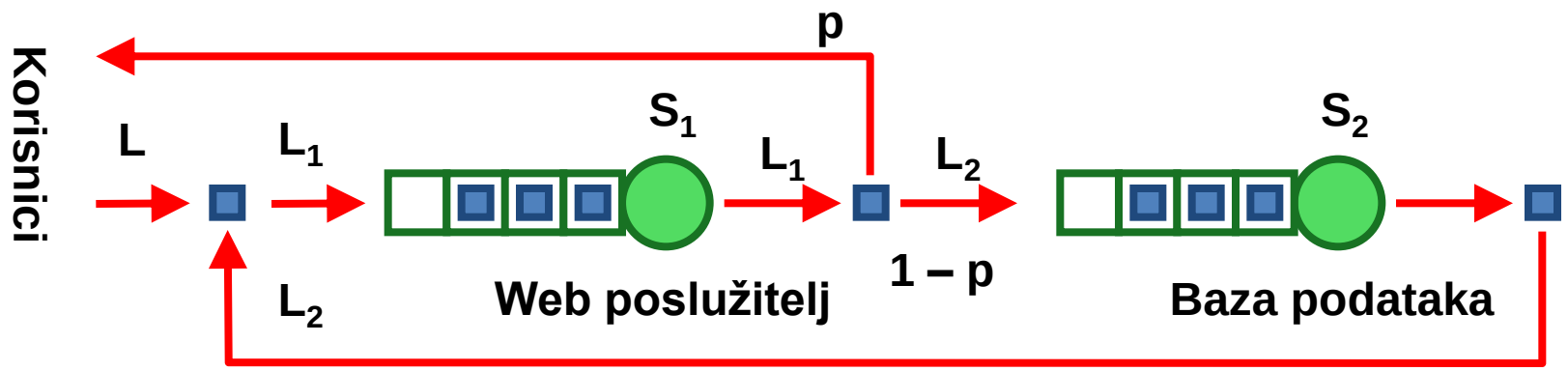
- **Mreža repova poruka**
  - Mrežna struktura proizvoljne složenosti s povratnim granama



Što znamo na temelju prethodnih razmatranja

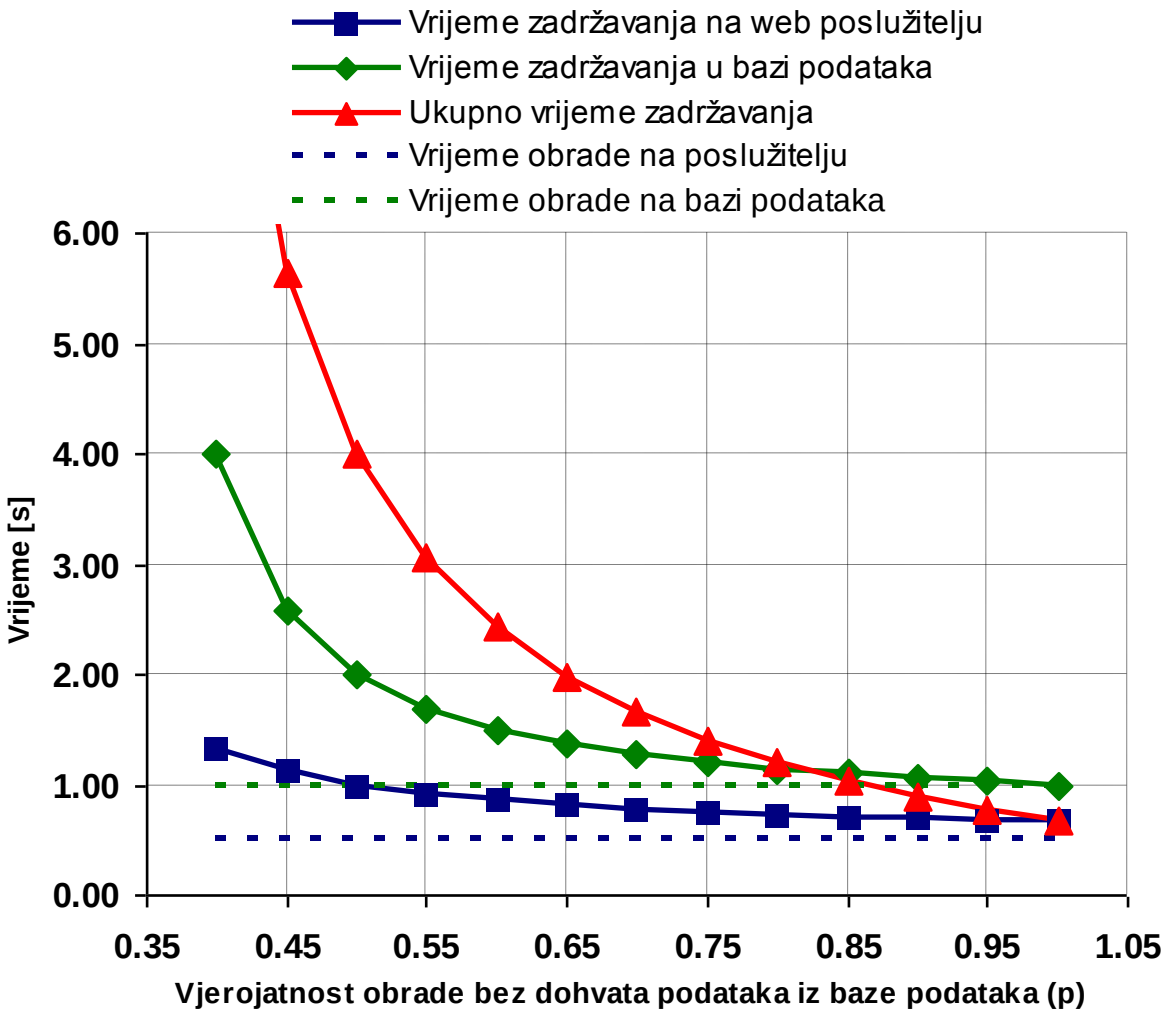
- **U stabilnom stanju sustava**
  - $U_N = X_N * S_N$ ($X_N = L_N$ za stabilni slučaj)
  - $U_1 = L * S_1 = 20 * L$
  - $U_2 = 600 * L_2 = 600\ (0.3 * L + 0.1 * L_3)$
  - $U_3 = 300 * L_3 = 300\ (0.7 * L + 0.2 * L_2)$
  - $U_4 = 60 * L$
- **Nakon rješenja za L_2 i L_3 i izračunavanja U_1 do U_4, izračunavamo Q_1 do Q_4 iz:**
  - $Q_N = U_N/(1 - U_N)$
- **Vrijeme zadržavanja u sustavu R**
  - $R = Q/L = (Q1 + Q2 + Q3 + Q4)/L$

Jednostavan model Web servisa



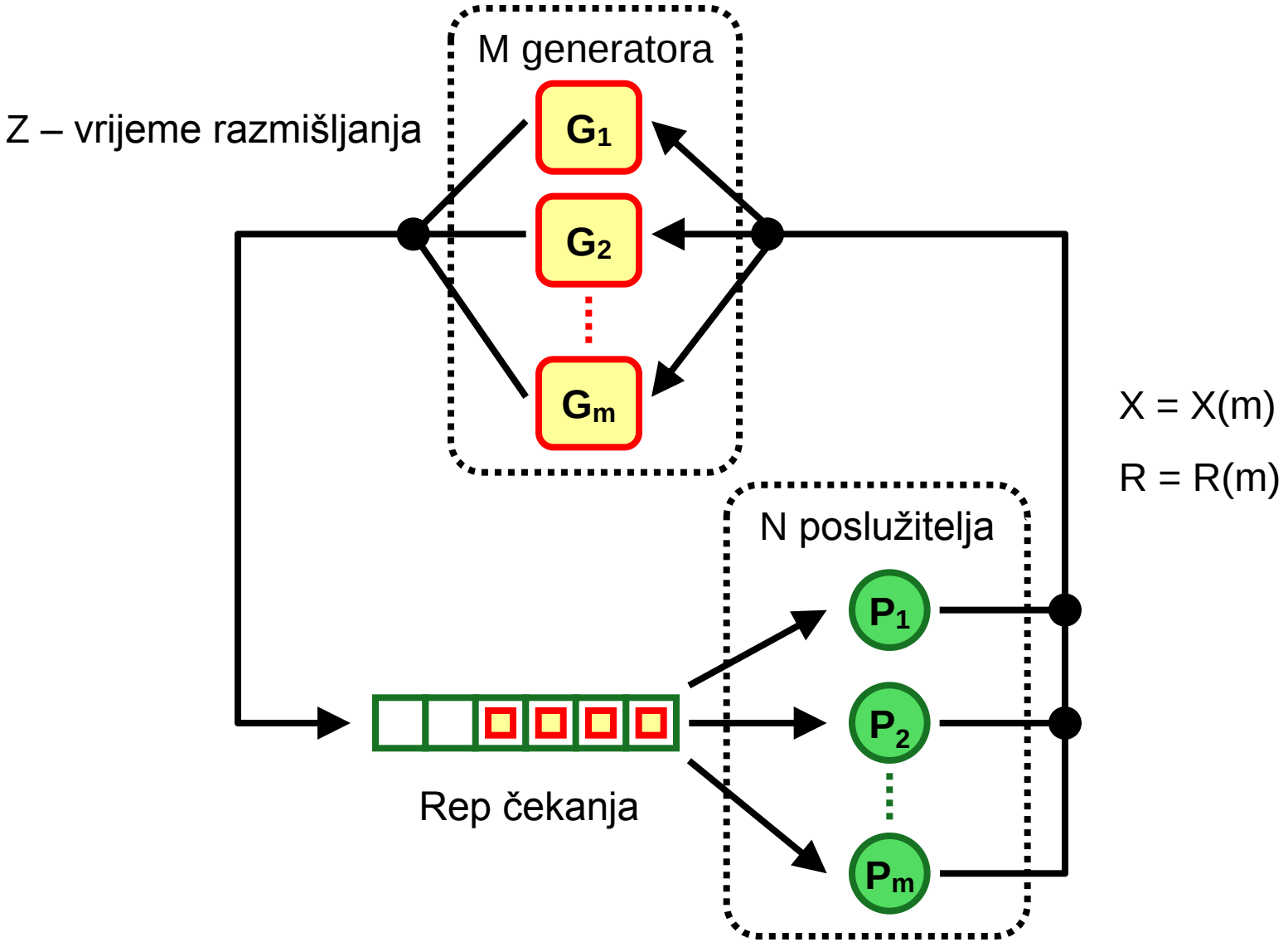
- **Učestalost dolazaka zahtjeva (L1, L2)**
  - L1 = L + L2 = L + (1-p)L1 = L/p
  - L2 = (1 – p)L1 = ((1 – p)/p)*L
- **Vrijeme zadržavanja zahtjeva u sustavu (R)**
  - X1 = L1; X2 = L2
  - U1 = X1*S1 = L*S1/p; U2 = ((1 – p)/p)*L*S2
  - R1 = S1/(1 – U1) ; R2 = S2/(1 – U2)
  - R = R1 * (1 + (1 – p)/p) + R2 * (1 – p)/p

Utjecaj parametara na ponašanje Web poslužitelja



Posluzitelj.xls

Zatvoreni centar za posluživanje



Osnovne jednadžbe za sustav s jednim poslužiteljem

- **Propusnost sustava**
  - Jednaka je ritmu razmišljanja pomnoženom s brojem slobodnih mislioca (ukupan broj m minus broj u repu)
  - X(m) = (m - Q)/Z
- **Množenjem s Z te primjenom supstitucije Q = X*R**
  - Z*X(m) = m – X(m)*R
  - X(m) = m / (R + Z)
- **Prosječno vrijeme zadržavanja zahtjeva R**
  - R = m / X(m) – Z

Primjer 8: Poslužitelj aplikacija

- **Poslužitelj aplikacija omogućava skupini inženjera razvoj programa u dijeljenom vremenu. Mjerenjem su utvrđene sljedeće značajke sustava:**
  - Srednji broj aktivnih razvojnih inženjera **m = 230**
  - Srednje vrijeme između kompilacija je **Z = 300 s**
  - Srednje iskorištenje poslužitelja je **U = 0.48**
  - Srednje vrijeme kompilacije **S = 0.63 s**

- **Upravitelj sustava želi odrediti:**
  - Propusnost sustava (**X**)?
  - Koliko je srednje vrijeme kompilacije (**R**)?

pr8.c

Primjer 8: Poslužitelj aplikacija

- **Rješenje**
  - Broj generatora zahtjeva **m=230**
  - Srednje vrijeme između kompilacija je **Z = 300 s**
  - Srednje iskorištenje poslužitelja je **U = 0.48**
  - Srednje vrijeme kompilacije **S = 0.63 s/kom**

  - **Propusnost sustava (X)**

    X = U/S = 0.48 / 0.63 s = **0.7619 kom/s**
  - **Srednje vrijeme zadržavanja u sustavu (R)**

    R = m / X(m) - Z

    R = (230 kom / 0.7636 kom/s) – 300 s = **1.21s**

Poopćenje na sustav posluživanja s N poslužitelja

- **Poznato u operacijskim istraživanjima kao "machine repair center" problem**
- **Kendall-ova notacija**
  - M/M/N/m/m
- **Egzaktno numeričko rješenje dano je izvornim kodom u jeziku C**
  - repair.c

repair.c

Primjer 8: Zatvoreni sustav s paralelnim poslužiteljima

- **Proširivanje sustava iz prethodnog primjera**
  - Što će se dogoditi sa sustavom ako poduzeće zaposli novih 200 programera?

    Odgovor: U = 0.88; R = 5.009 s

  - **Koliko će se situacija popraviti ako se poslužitelju doda drugi procesor sa istim značajkama?**

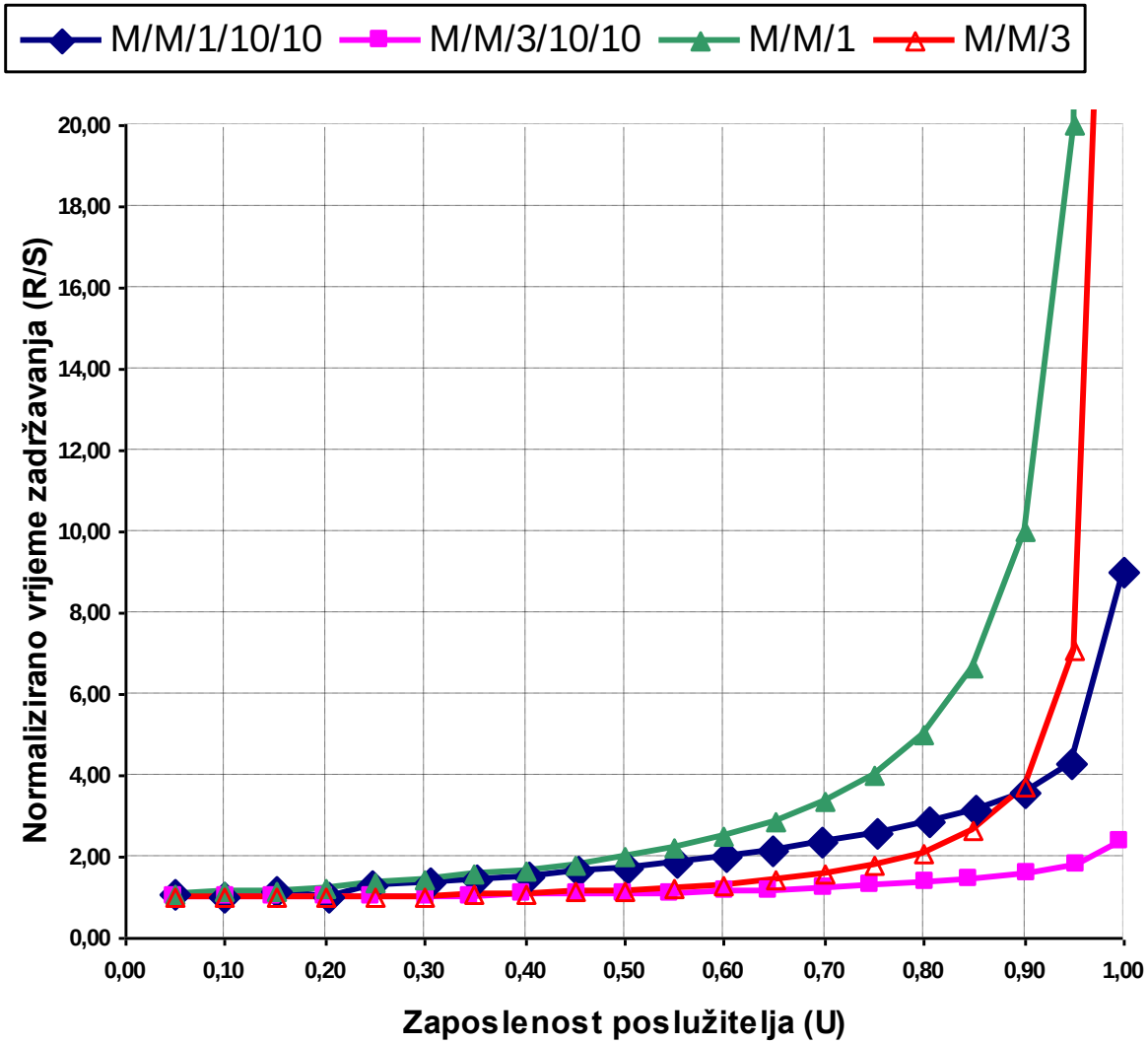
    Odgovor korištenjem *repair.c*:
    - Praksa je pokazala da se u multi-procesorskim sistemima postoji dodatni teret zbog sinkronizacije procesora. Uobičajeni faktor je 3 - 5% tj. u našem slučaju uzmimo da se S rate se povećava na ~ 0.66s
    - Program daje slijedeće rezultate:
      - U = 0.47; R = 0.8465s

Zatvoreni sustav s paralelnim poslužiteljima

- **U zatvorenom sustavu rep ne može narasti preko ukupno m – 1 zahtjeva (plus jedan u izvršavanju)**
  - R = m/X - Z
  - R/S = m/XS – Z/S
  - R/S = m/U – Z/S
  - U = ro*N
  - R/S = m/(ro*N) – Z/S

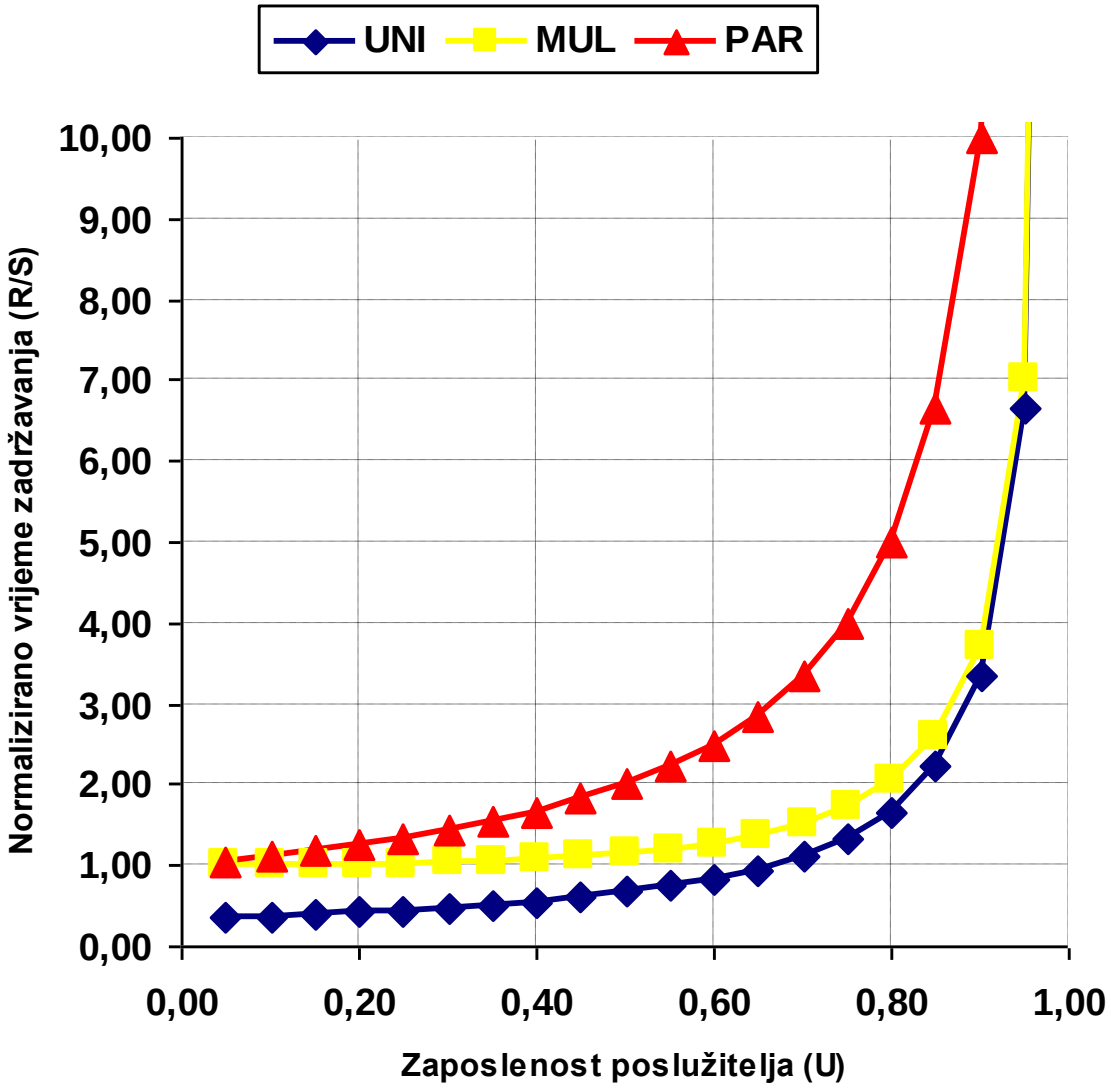
- **U slučaju kada ro $\rightarrow$ 1 i Z $\rightarrow$ 0**
  - R/S $\rightarrow$ m/N
  - R/S ne raste u beskonačno jer je broj zahtjeva u repu ograničen (t.j. zatvoreni sustav)

Usporedba otvorenih i zatvorenih sustava



Usporedba.xls

Koji je model bolji: Multi-računalo ili Multi-procesor ?

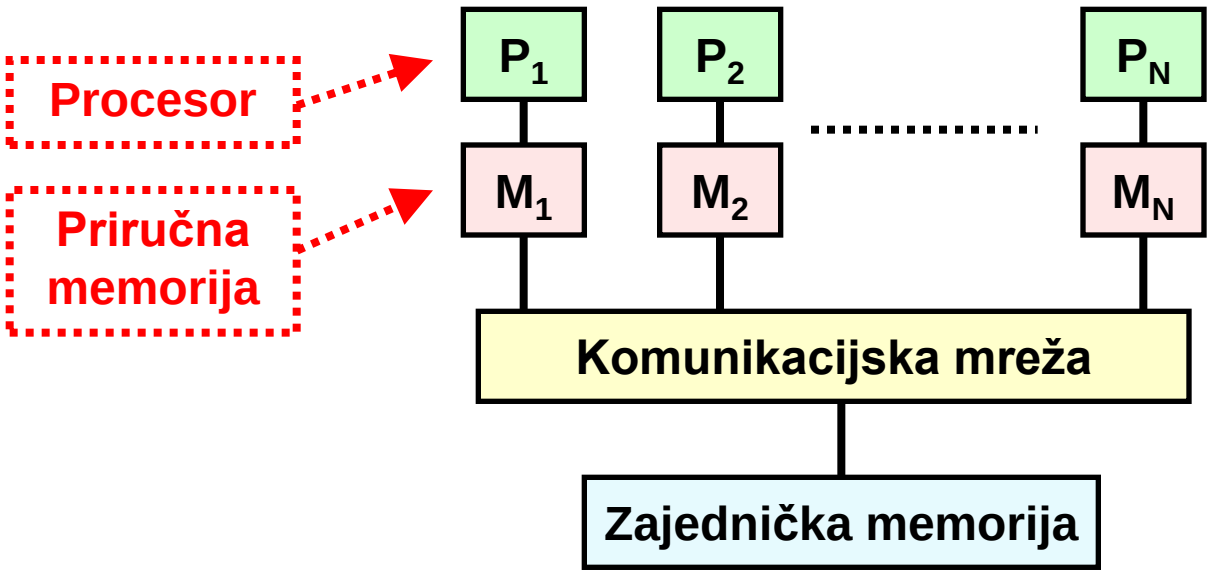


Uniprocesor ima
3x procesnu moć

Sustavi.xls

Performanse paralelnih aplikacija

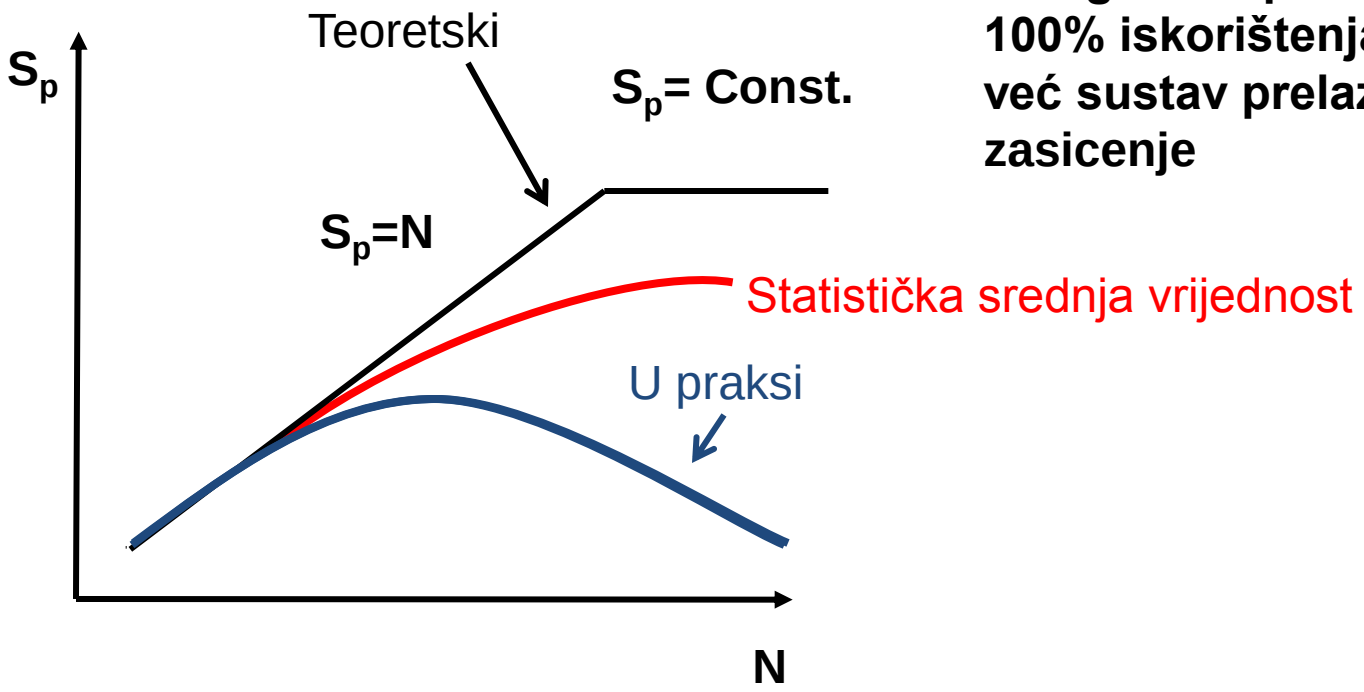
Karakteristicna paralelna arhitektura



* Moze biti fizicka ili virtualna

- **Komunikacijska mreža**
  - sabirnica (bus), zbir (crossbar), prsten (ring), stablo (tree), matrica (mesh), hiper-kocka (hiper-cube)

Ubrzanje (Speedup) je nelinerna funkcija od N



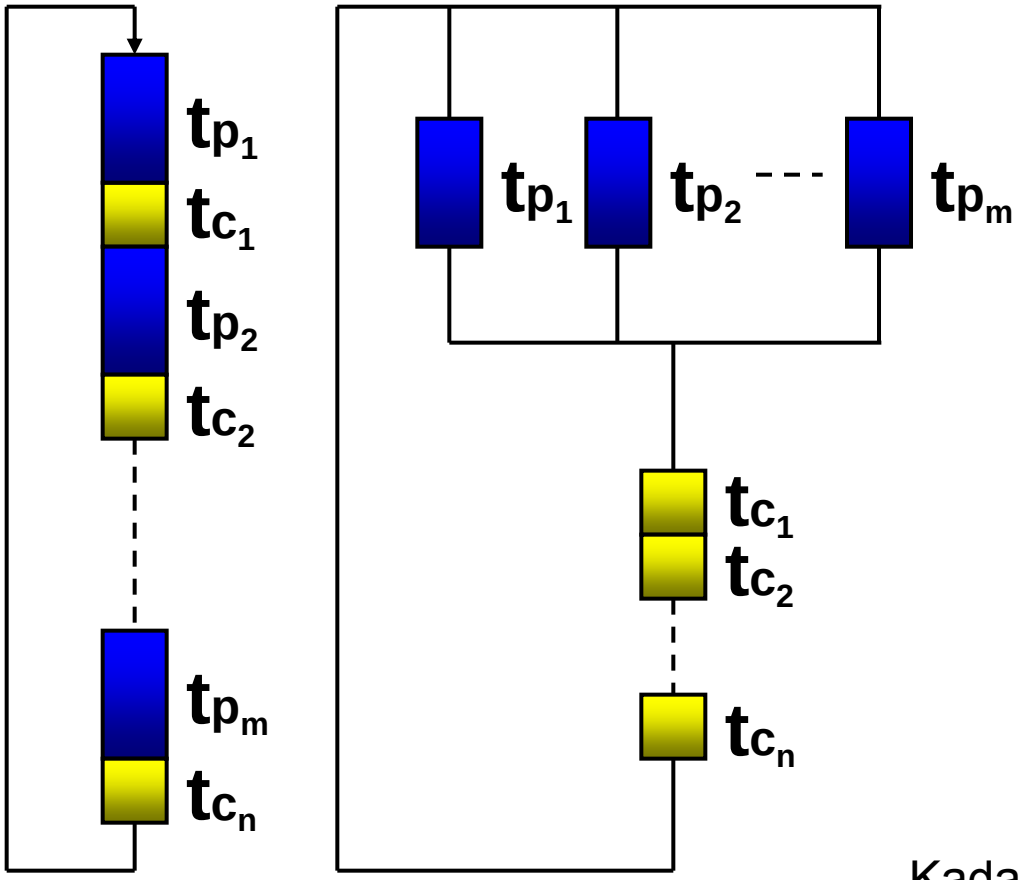
Sabirnica sustava (kao i izlazna jedinica) ne omogućava punu brzinu kod 100% iskorištenja sustava, već sustav prelazi u potpuno zasicenje

Bolje razumijevanje parametara

- **Uzrok degradacija performansi u paralelnim sustavima**
  - Ostvarivanje sekvencijalnog pristupa zajedničkim sredstvima u aplikaciji i sklopovima
  - Kašnjenja zbog održavanja koherencije i sinkronizacije
  - Dodatni posao zbog neproporcionalne podjele na n paralenih aktivnosti

- **Modeliranje paralelnih aplikacija**
  - Deterministički modeli
  - Statistički modeli

Deterministički model



$$Tp = \sum_i tp_i$$

$$Tc = \sum_j tc_j$$

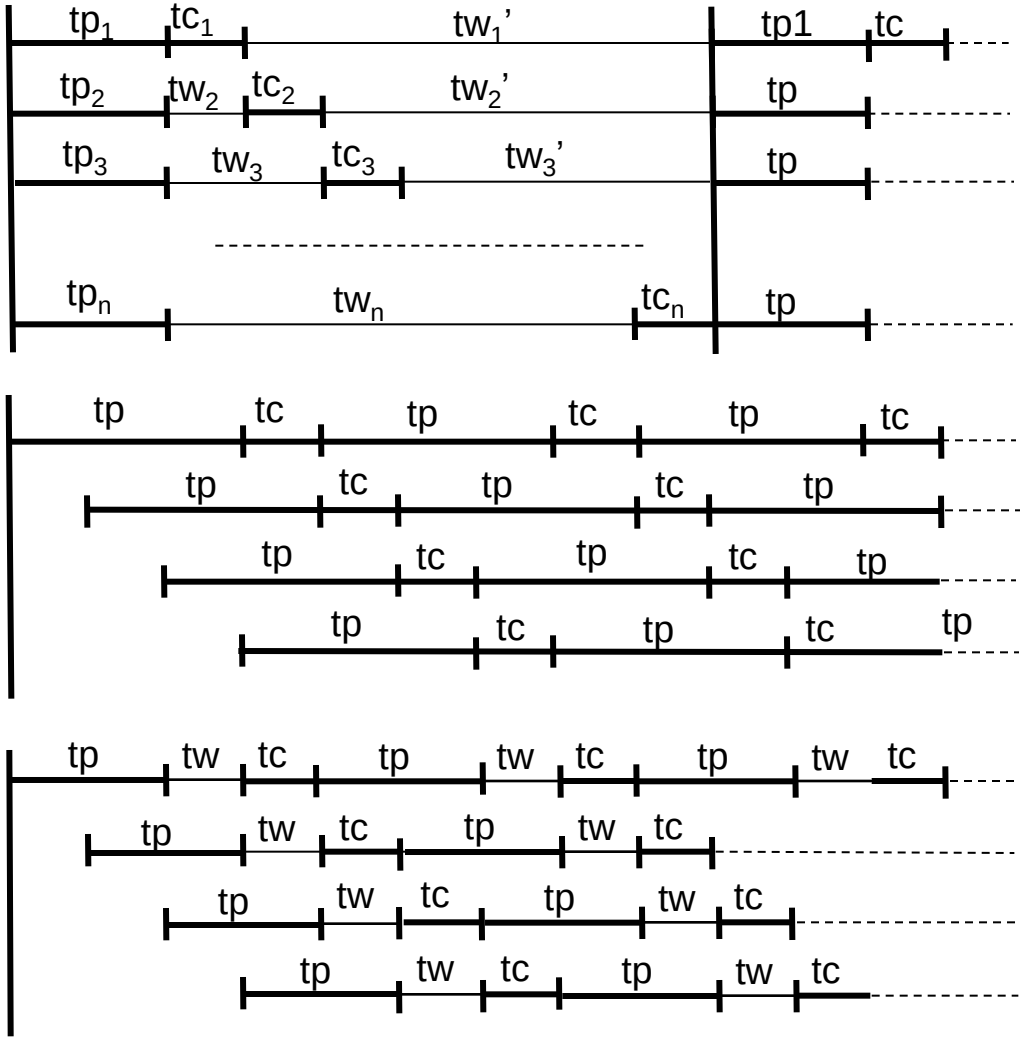
$$x = Tp / Tc$$

Amdahl's Law

$$S_{max} = \frac{Tp + Tc}{Tc} = x + 1$$

Kada je tp = 0, tj. paralelno izvođenje u 0 vremena kada N tezi beskonačno

Dva granična slučaja



Sinkroni (Najgori slučaj)

$tw = (N - 1)*tc$

Asinkroni (Najbolji slučaj)

$tw = 0$ ako $tp \geq (N - 1)*tc$

$tw = (N - 1)*tc - tp$ ako $tp < (N - 1)*tc$

Osnovne determinističke formule

- **Osnovne veličine**

$$E = \frac{tp + tc}{tp + tc + tw}, \quad S = E\frac{Tp + Tc}{tp + tc}$$

$$fp(N) = \frac{Tp}{tp_N}, \quad fc(N) = \frac{Tc}{tc_N}$$

Utjecaj neproporcionalne raspodjele

- **Asinkroni slučaj**

$$S = \min\left[\frac{fpfc(x+1)}{fp + xfc}, \frac{fc(x+1)}{N}\right]$$

X = Tp/Tc

Ako fp = fc =N imamo Amdahl-ov zakon

- **Sinkroni slučaj**

$$S = \frac{fpfc(x+1)}{Nfp + xfc}$$

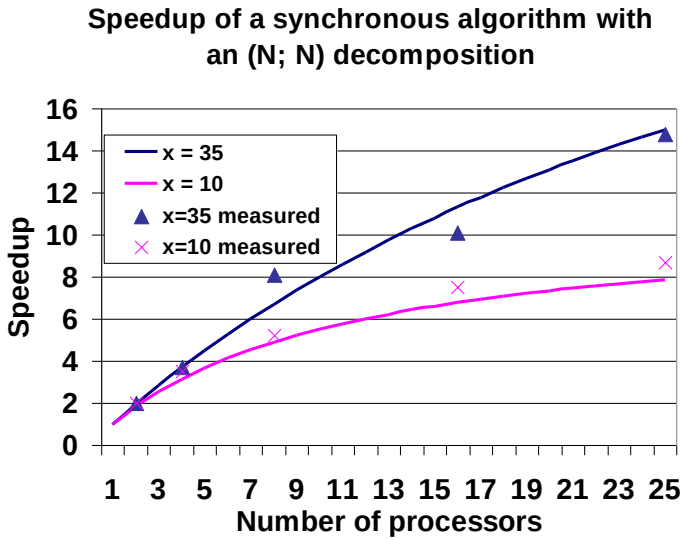
Rezultati za prikazane funkcije

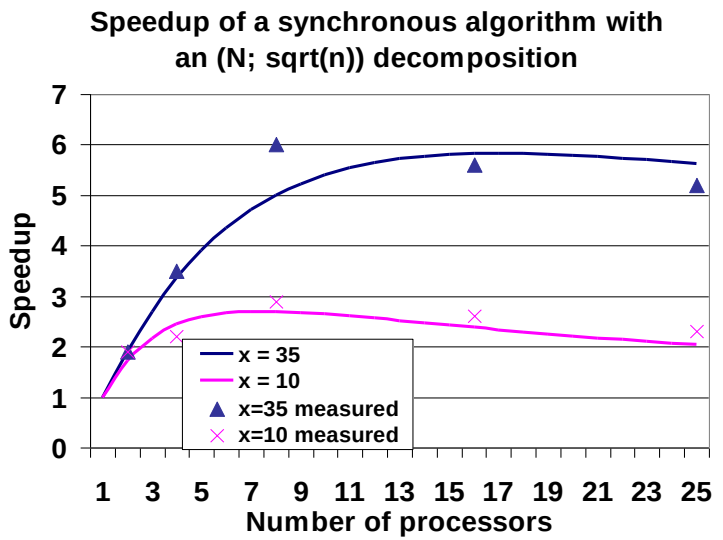
Dekompozicija		Sinkroni slučaj			Asinkroni slučaj		
tp	tc	SP	SP_{max}	N_{max}	SP	SP_{max}	N_{max}
N	N	$\frac{(1+x)N}{N+x}$	$1+x$	∞	$\min[N, 1+x]$	$1+x$	$1+x$
N	$\sqrt{N}$	$\frac{(1+x)N}{N^{3/2}+x}$	$\frac{2^{2/3}(1+x)}{3x^{1/3}}$	$2x^{2/3}$	$\min\left[\frac{(1+x)N}{\sqrt{N}+x}, \frac{(1+x)}{\sqrt{N}}\right]$	$\frac{(1+x)x^{1/3}}{1+x^{2/3}}$	$x^{2/3}$
N	1	$\frac{(1+x)N}{N^2+x}$	$\frac{1+x}{2\sqrt{x}}$	$\sqrt{x}$	$\min\left[\frac{N(1+x)}{N+x}, \frac{1+x}{N}\right]$	$\frac{2(1+x)}{1+\sqrt{1+4x}}$	$\frac{1+\sqrt{1+4x}}{2}$
$logN$	$logN$	$(1+x)\frac{\log N}{N+x}$	- *)	$N(\log N - 1) = x$ *)	$\min\left[\log N, \frac{(1+x)\log N}{N}\right]$	$\log(1+X)$	$1+x$

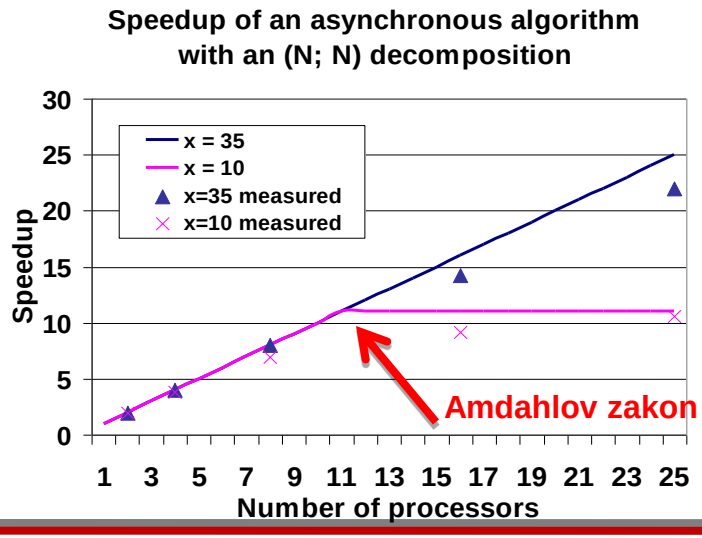
*) **Samo numeričko rješenje**

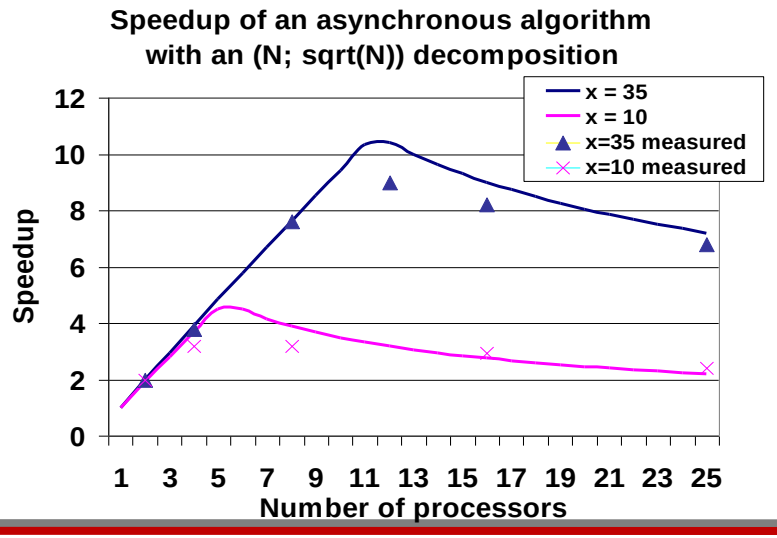
Preuzeto iz: **Performance prediction and calibration for a class of multiprocessors**
Vrsalovic, D.F.; Siewiorek, D.P.; Segall, Z.Z.; Gehringer, E.F.;
IEEE Transactions on Computers, Volume: 37 Issue: 11 , Nov. 1988. Page(s): 1353 -1365

Funkcije raspodjele opterećenja u praksi

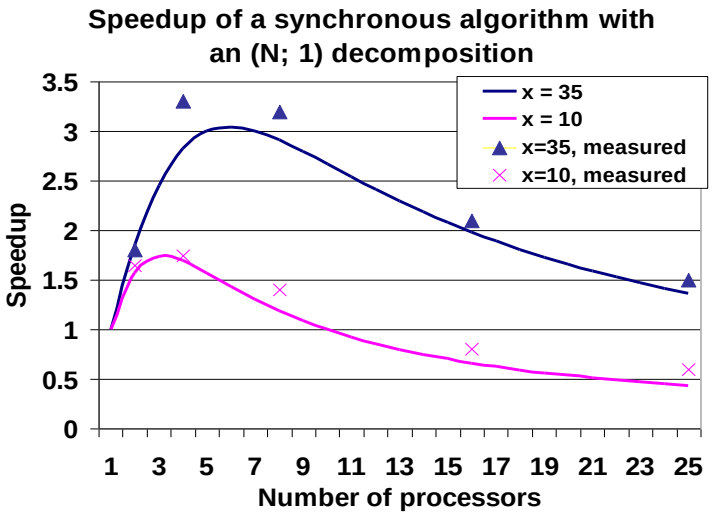


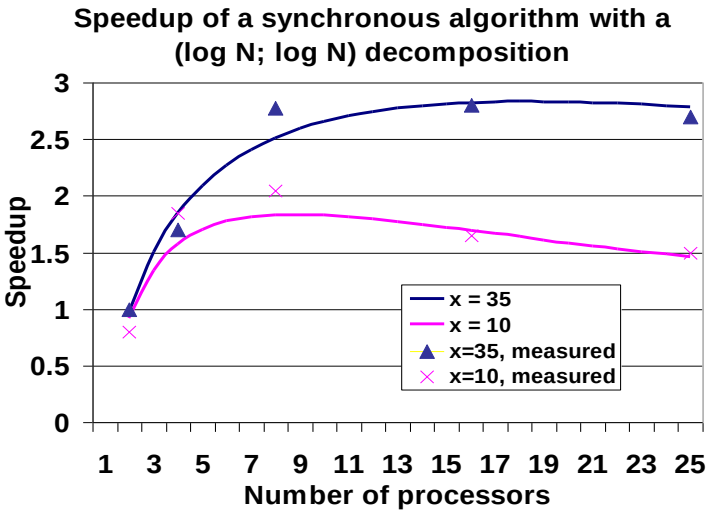


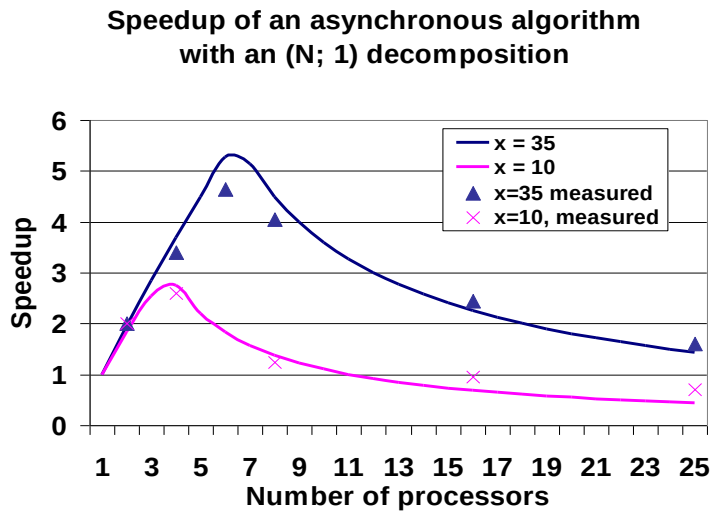


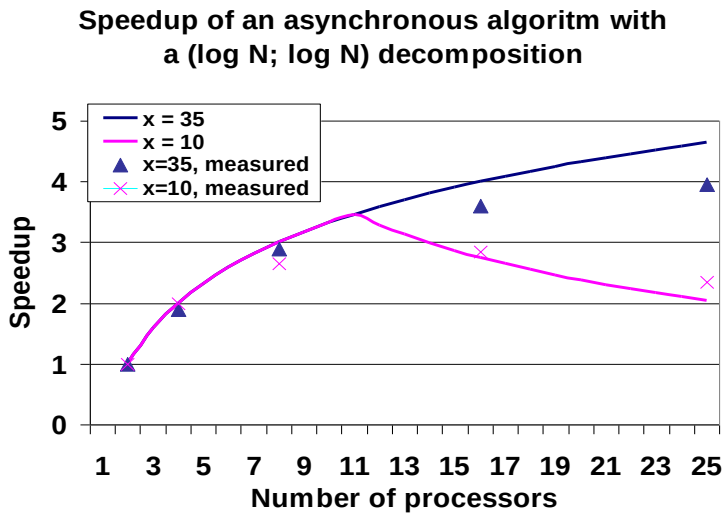


Funkcije raspodjele opterećenja u praksi



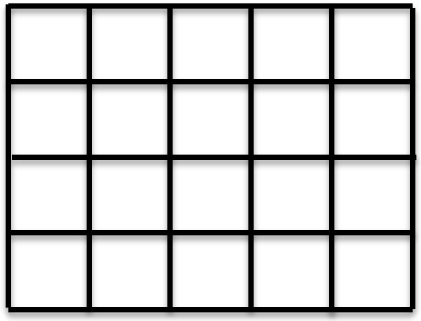


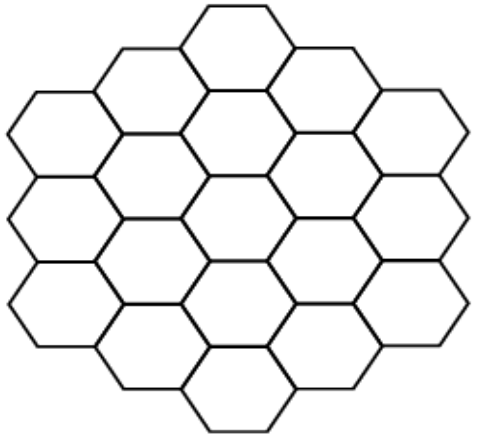




Razumijevanje modela pomaže programiranju

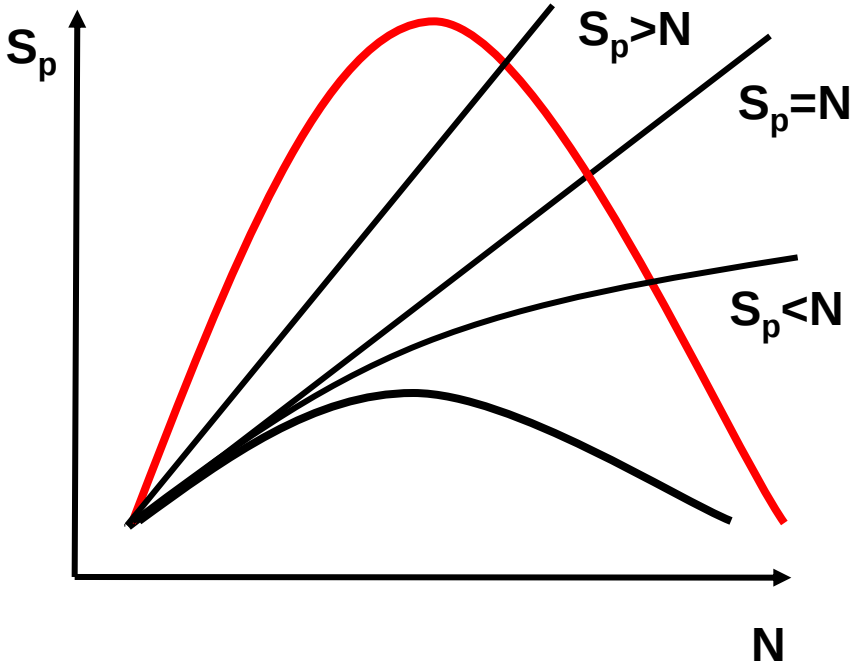
- **Primjer: Algoritmi najbližih susjeda**





- **Trend analiza ili težinska suma najbližih susjeda**
  - C(i,j) = K1 x C(i-1,j) + K2 x C(i+1,j) + K3 x C(i, j-1) + K4 x C(i,j+1) – 4K5 x C(i,j)
- **Za ovu klasu algoritama ubrzanje je proporcionalno omjeru površine (tp) i opsega (tc) osnovne ćelije podataka**
- **Šesterokut ima bolji omjer površine i opsega od kvadrata**

Da li je superlinearno ubrzanje moguće?



Utjecaj primjene međuspremnika (cache)

$$tc = f(\text{veličina spremnika}) \times fc$$

Primjena genetskih algoritama

$$P(T_i \le T) = p$$

$$t_{i[N=2]} \le T = p + p + p^2 > 2p$$

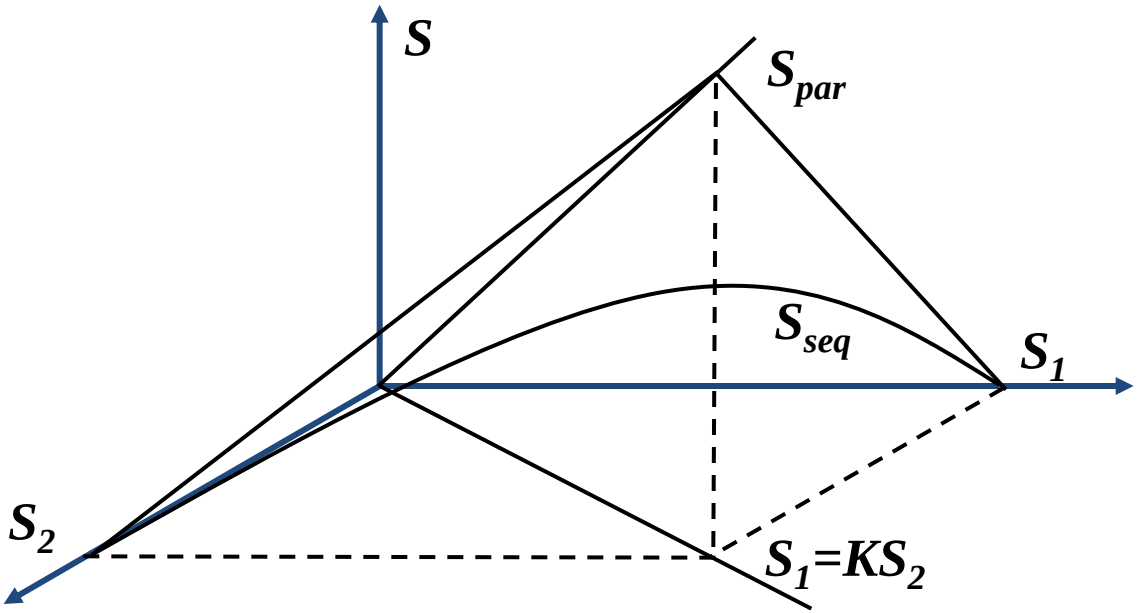
$$Sp = f(t_i)$$

Kombiniranje dviju paralelnih aplikacija

$$S_{seq} = \frac{T_1 + T_2}{t_1 + t_2} = \frac{S_1 S_2 (1+K)}{S_1 + K S_2}, \quad K = \frac{T_1}{T_2}, \quad S_1 = \frac{T_1}{t_1}, \quad S_2 = \frac{T_2}{t_2}$$

$$T_i = T_p + T_c, \quad t_j = t_p + t_c + t_w$$

$$S_{par} = \min\left[\frac{T_1}{t_1}, \frac{T_2}{t_2}\right] = \min\left[S_1, S_2\right]$$



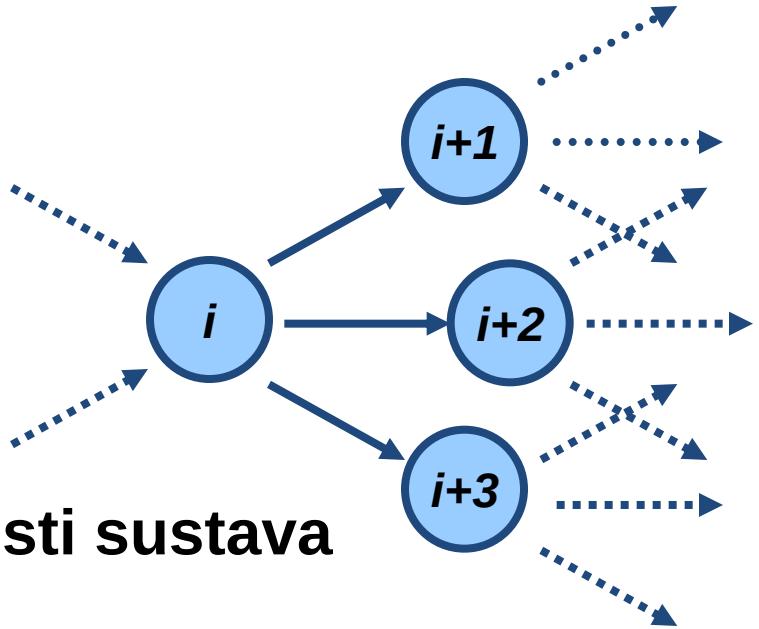
Probabilistički modeli paralelnih aplikacija

Stohastički modeli zatvorenih sustava

- **Zajednički resursi se prikazuju kao servis**
- **Kašnjenje se modelira kao čekanje na servis**
- **Faktor iskorištenja sustava (ro)**
  - Servis se koristi ako se nalazi u bilo kojem stanju osim početnom
  - Vjerojatnost da se servis nalazi u stanju koje nije početno: ro = 1 - p_0
- **Ritam dolazaka zahtjeva za posluživanje je K/Z**
  - Z – vrijeme razmišljanja
  - K – broj mislioca
- **Ritam posluživanja je 1/S**
  - S – srednje vrijeme posluživanja

Diagram stanja i matrica prijelaznih vjerojatnosti

- **Dijagram stanja sustava**
  - Opisuje izmjenu stanja sustava tijekom rada
  - Pn je vjerojatnost da sustav bude u stanju n
  - Suma Pn = 1

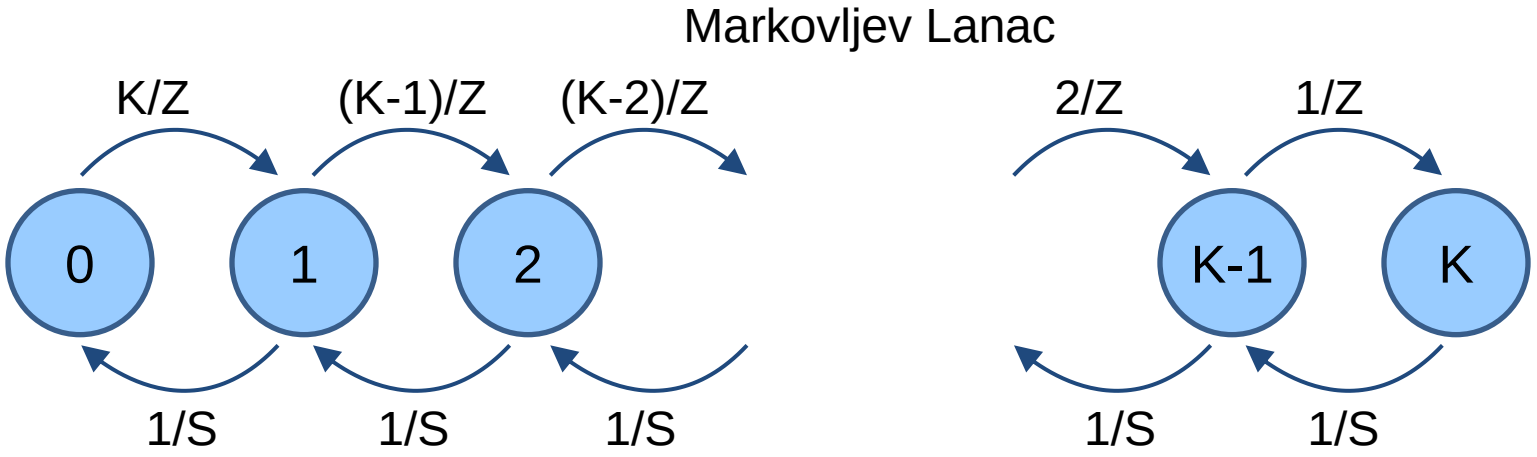


- **Matrica prijelaznih vjerojatnosti sustava**
  - Element matrice u retku *i* te stupcu j određuje vjerojatnosti da će sustav iz stanja *i* preći u stanje *j*

$$P_n = \sum_{k=0}^{K} P_k * p_{k,n}$$

$$\begin{bmatrix} p_{0,0} & p_{0,1} & \cdots & & \\ p_{1,0} & \cdots & & & \\ \cdots & \cdots & p_{i,j} & \cdots & \cdots \\ & & & \cdots & p_{N-2,N-1} \\ & & \cdots & p_{N-1,N-2} & p_{N-1,N-1} \end{bmatrix}$$

Stohastički model paralelnog sustava (M/M/1/K/K)



- **Vjerojatnost da se Markovljev lanac nalazi u *n-tom* (p_n) i početnom (p_0) stanju**

$$p_n = \frac{K!}{(K-n)!}\left(\frac{S}{Z}\right)^n p_0$$

$$\sum_n p_n = 1 \qquad p_0 = \left[\sum_{k=0}^{K} \frac{K!}{(K-k)!}\left(\frac{S}{Z}\right)^k\right]^{-1}$$ (Erlang-ova formula)

Stohastički model (M/M/1/K/K)

- **Faktor iskorištenja sustava (ro)**
  - ro = 1 – p0 = 1 – B(K, Z/S) (Erlang-ova formula)
- **Vrijeme čekanja u sustavu (W)**
  - ro = U = LS
  - L = ro/S
  - L = K/(Z + W + S)
  - K/(Z + W + S) = ro/S
  - W = (KS/ro) – Z – S
- **Vrijeme zadržavanja u sustavu (R)**
  - R = S + W = (KS/ro) - Z;
- **Broj zahtjeva u repu (Q)**
  - Q = L R

Efikasnost i ubrzanje

- Efikasnost procesora je definirana kao omjer korisnog vremena i ukupnog vremena:
  - E(K) = (Z + S)/(Z + S + W)
- Ubrzanje je definirano kao efikasnost procesora pomnožena sa brojem procesora:
  - S(K) = K * E(K)

Rekapitulacija

Rekapitulacija

- Koji su elemnti cuklusa života računarskog sustava?
  - Definicja zahtjeva, analiza rješenja, sinteza, ispitivanje, pogon, mjerenja i modifikacija zahtjeva
- Koji su najvažniji nefunkcionalni zahtjevi?
  - Performanse, dostupnost i ukupna cijena vlasništva
- Kako se oni skupno zovu?
  - Kvaliteta usluga (QOS)
- Sto je to ugovor o razini usluge (SLA)?
  - Ugovor između korisnika i pružaoca usluge koji definira razinu usluge
- Zašto je vrednovanje performansi važno ?
  - Zbog potrebe za planiranjem kapaciteta koji je potreban za uspješno poslovanje
- Koje metode analize se koriste u praksi?
  - Iskustvo, modeliranje i simulacija
- Kakve vrste modela tereta postoje u praksi?
  - Prirodne aplikacije, umjetne aplikacije (benchmarks) i neizvodivi modeli opisani ritmom zahtjeva, prosječnim vremenom obrade i sl.

Rekapitulacija

- Koji su koraci pri razvoju modela sustava?
  - Razumijevanja funkcioniranja, modeliranje tereta, mjerenje sustava u pogonu radi utvrđjivanja parametara tereta, razvoj modela, verifikacija i validacija, analiza mogućih scenaria promjena, prognoza promjena tereta u budućnosti, prognoza performansi sustava nakon pustanja u pogon te u budućnosti
- Koje su najčešće greške kod modeliranja?
  - Prekompleksna analiza, nema specifičnog cilja, prejudiciranje (model treba dokazati da je nas sustav bolji od njihovog), nedovoljno razumijevanje sustava, neadekvatne mjere (napr. TPS za Web i DB usporedbu), nereprezentativni teret, neuključivanje važnih parametara, promatranje u krivom intervalu vrijednosti parametara, krivo baratanje ekstremima, nedovoljno promatranje evolucije sustava i tereta, kriva intepretacija rezultata
- Što definiraju pojmovi MTBF i MTTR?
  - Srednje vrijeme između pogrešsaka i srednje vrijeme popravka
- Kako je definirana dostupnost sustava?
  - D=MTBF/(MTBF + MTTR)
  - Postotak vremena u kojemu je sustav dostupan korisnicima

Rekapitulacija

- Kako se računa dostupnost paralelnih i serijskih sustava?
  - Dp = (1-D1)*D2 + (1-D2)*D1 + D1D2
  - Ds = D1*D2
- Koje su dvije osnovne grupe troškova za gradnju i pogon Web sustava i kako su obično raspodijeljeni?
  - Kapitalni i operacioni troškovi grubo raspodijeljeni 50/50 kroz tri godine
- Koje su mogućnosti za udomljivanje Web servisa?
  - Udomitelj infrastrukture ili vlastito udomljavanje
- Objasni razliku između vremena odziva i kapaciteta
  - Vrijeme odziva je vrijeme odziva na jedan zahtjev, dok kapacitet daje maksimalni broj zahtjeva koji se mogu obraditi u jedinici vremena
- Koje se metode za poboljšanje performansi koriste u arhitekturi raspodijeljenih sustava?
  - Serijsko preklapanje, paralelno preklapanje, paralelno izvođenje i privremeni spremnici
- Sto je ubrzanje
  - Ubrzanje je omjer vremena izvođenja na jednom i više paralelnih podsustava

Rekapitulacija

- Koji su osnovni faktori koji utječu na skaliranje?
  - Sukobi za zajednicke resurse i usklađivanje zajedničkih varijabli
- Kako je definiran univerzalni zakon skaliranja?
  - C(p) = p/(1 + s(p-1) + up(p-1))
  - s opisuje utjecaj sukoba dok u opisuje utjecaj usklađivanja
- Koji se još zakoni skaliranja viđaju u literaturi?
  - Geometrijski, kvadratni i eksponencijalni
- Kako se određuje skaliranje klastera?
  - Prvo se primjeni univerzalni zakon skaliranja na svaki čvor, pa se zatim isti zakon primjeni za tako skalirane čvorove na cijeli klaster
- Kako se mjeri i modelira kapacitet servisnog sustava?
  - Mjerenjem pomoću generatora tereta, te aproksimacijom pomoću univerzalnog zakona skaliranja
- Kako se određuje vrijeme potrebno da se dostigne potreba za dvostrukim kapacitetom?
  - Podrazumijevajući eksponencijalni trend

Rekapitulacija

- **Koliko razina protokola uključuju Web aplikacije ?**
  - Četiri: Ethernet, IP, TCP/IP i HTTP
- **Koji su osnovni dijelovi odziva Web aplikacije ?**
  - Klijent-ISP, ISP-ISP, ISP-servis
- **Kako se ostvaruje razmjeran rast aplikacije ?**
  - Vertikalno ili horizontalno
- **Koje probleme donosi horizontalan rast ?**
  - Sinkronizacija, razdioba tereta i razdioba podataka
- **Koju mogućnost donosi horizontalan rast ?**
  - Manja osjetljivost na greške
- **Koje konfiguracije podržavaju neosjetljivost na greške ?**
  - Aktivan-pripravan i Aktivan-aktivan
- **Koje su tipične zone u sustavu Web servisa ?**
  - Demilitarizirana, aplikacijska i zona podataka

Rekapitulacija

- **Kako se ostvaruje razdioba tereta ?**
  - Sklopkama koje rade na IP ili aplikacijskoj razini
- **Što je replikacija ?**
  - Replikacija osigurava razdiobu istih podatka na više sustava
- **Kakve vrste replikacije postoje ?**
  - Master-slave, master-master-slave, stablo, stablo s filterima, master-master i master ring
- **Što je federacija ?**
  - Federacija dijeli podatke u različite grupe koje raspoređuje na različite sustave
- **Zašto se koriste grupni protokoli ?**
  - Za pouzdanu dostavu svim članovima grupe i garantirani slijed poruka
- **Koje su tipične garancije za slijed poruka?**
  - FIFO, posljedična i totalna

Rekapitulacija

- **Koje su osnovne veličine u modelu repa čekanja ?**
  - Vrijeme obzervacije (T), broj dolazaka (A), broj odlazaka (C) i vrijeme zaposlenosti poslužitelja (B)
- **Koje su izvedene veličine ?**
  - Ulazni ritam (L=A/T), izlazni ritam (X=C/T), srednje vrijeme posluživanja (S=B/C) i zaposlenost poslužitelja (U=B/T)
- **Kako se definira stacionarno stanje ?**
  - X = L
- **Kako glasi Little-ov zakon ?**
  - Broj zahtjeva u repu proporcionalan je ritmu dolaska zahtjeva i vremenu provedenom u sustavu (Q = L*R)
- **Kako je definirano vrijeme čekanja u repu ?**
  - W = Q*S
- **Kako je definirano ukupno vrijeme provedeno u sustavu?**
  - R = S + W = Q/L

Rekapitulacija

- **Kako se izračunava vrijeme odziva za serijske repove?**
  - R = (Q1 + Q2 +…..+QN)/L
- **Kako je definirana iskoristivost procesora ro?**
  - ro = U/N
- **Kako se izračunava vrijeme odziva za paralelne repove?**
  - R = S/(1 – ro)
- **Sto utječe na R kod paralelnih poslužitelja?**
  - $R = S/(1 – ro^N) ; ro = U/N$ (aproksimacija)
  - R = S(1 + (C(N,ro) / (N*(1 – ro))))
    (egzaktno rješenje uz Erlangovu formulu)
- **Kolika je najveća greška aproksimacije?**
  - Manja od 8% u praktičnim situacijama
- **Kako se računa R za sustav sa vjerojatnošću povratne veze p?**
  - L1 = L/(1 – p); U = L1 * S); R1 = S*(1 + U/(1-U)); R = R1/(1 – p)

Rekapitulacija

- **Kako je definiran zatvoreni sustav ?**
  - Broj zahtjeva je ograničen
- **Kako se izračunava R za zatvoreni sustav sa m izvora zahtjeva ?**
  - X(m) = (m – Q)/Z ; Q = X(m) * R; R = m / X(m) – Z; X(m) = U/S)
- **Što je efikasnost procesora a što ubrzanje u paralelnim sustavima ?**
  - E = (tp + tc)/(tp + tc + tw); S = ((Tp + Tc)/tp + tc)) * E
- **Kako je definirana funkcija dekompozicije ?**
  - Funkcija od N koja definira kako se paralelna aplikacija dijeli na N procesora
- **Koji je nagori i najbolji slučaj rada paralelne aplikacije ?**
  - Sinkrono i asinkrono izvršavanje
- **Koji stohastički proces modelira ponašanje paralelnog sustava podrazumjevajući M/M/1/K/K model?**
  - Markovljev lanac