Diplomski studij

Informacijska i komunikacijska tehnologija:

Telekomunikacije i informatika

Računarstvo:

Programsko inženjerstvo i informacijski sustavi

Računarska znanost

Ak.g. 2008./2009.

Raspodijeljeni sustavi

10.

Vrednovanje performansi raspodijeljenih aplikacija

Dr. Dalibor F. Vrsalović

dalibor.f.vrsalovic@fer.hr

15.1.2009

Sadržaj predavanja

- Uvod
- Dio I: Raspodijeljeni sustavi u praksi
- Dio II: Vrednovanje performansi aplikacija teorijom repova
- Dio III: Primjer analize performansi web aplikacije
- Rekapitulacija

Uvod

Zašto je vrednovanje performansi sustava važno u praksi ?

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 sva 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

- **Na ovom predavanju saznati će te kako vrednovati performanse aplikacije**
  - Linearna ekstrapolacija najčešće daje netočne rezultate
  - Pogrešno projektirana aplikacija može imati katastrofalne posljedice na poslovanje

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 zele dalje istrazivati

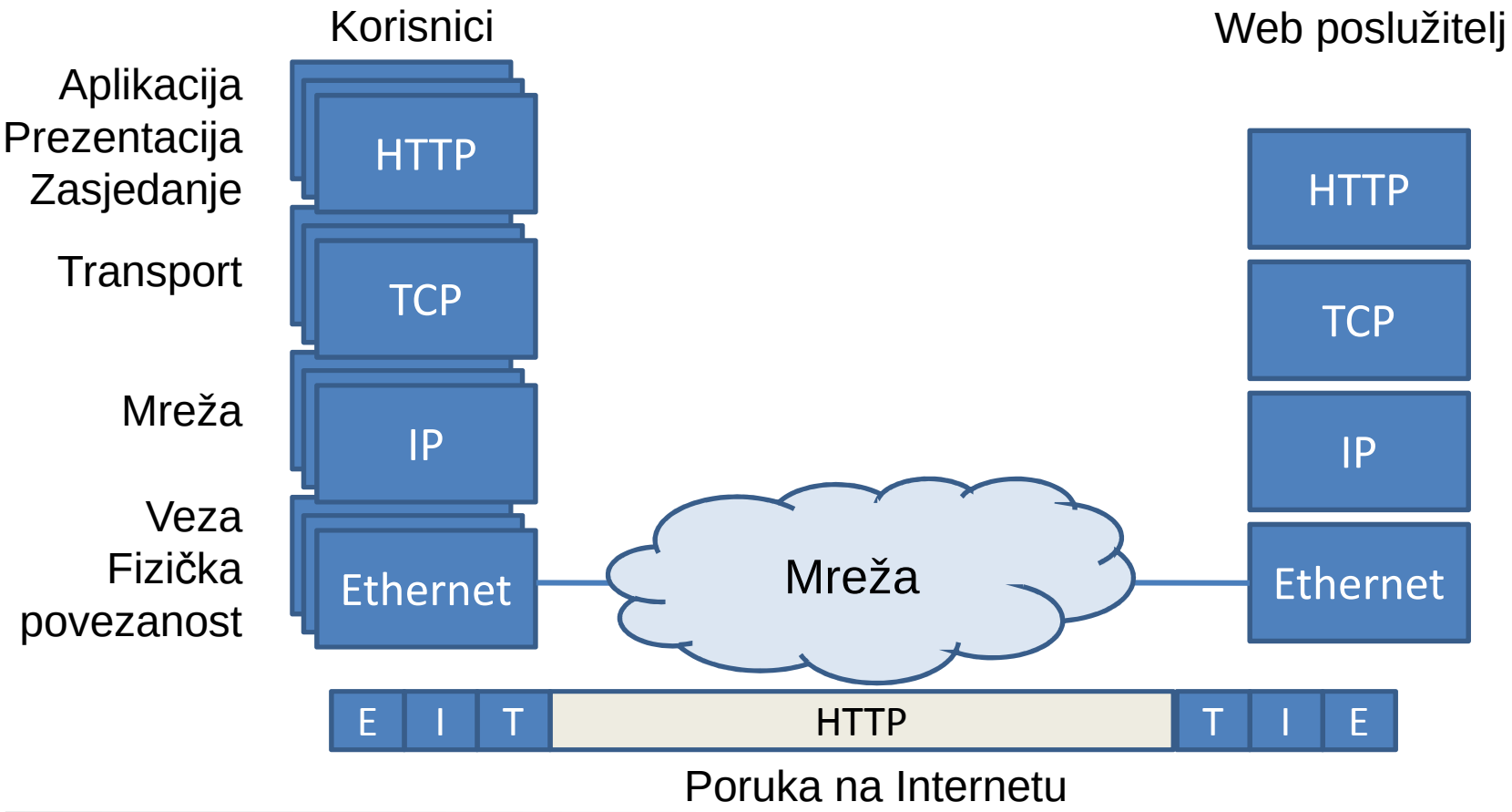
- A.O. Allen: **"Probability, Statistic, and Queueing Theory with Computer Science Applications"**, Academic Press 1978.
- S. Joines, R. Willenborg, K. Hygh: **"Performance analysis for Java Web Sites"**, Addison Wesley, 2003
- S. Sounders: **"High Performance Web Sites"**, O'Reilly, 2007.
- T. Schlosssnagle: **"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.

Raspodijeljeni sustavi u praksi

Modeliranje sustava zahtjeva razumijevanje sustava

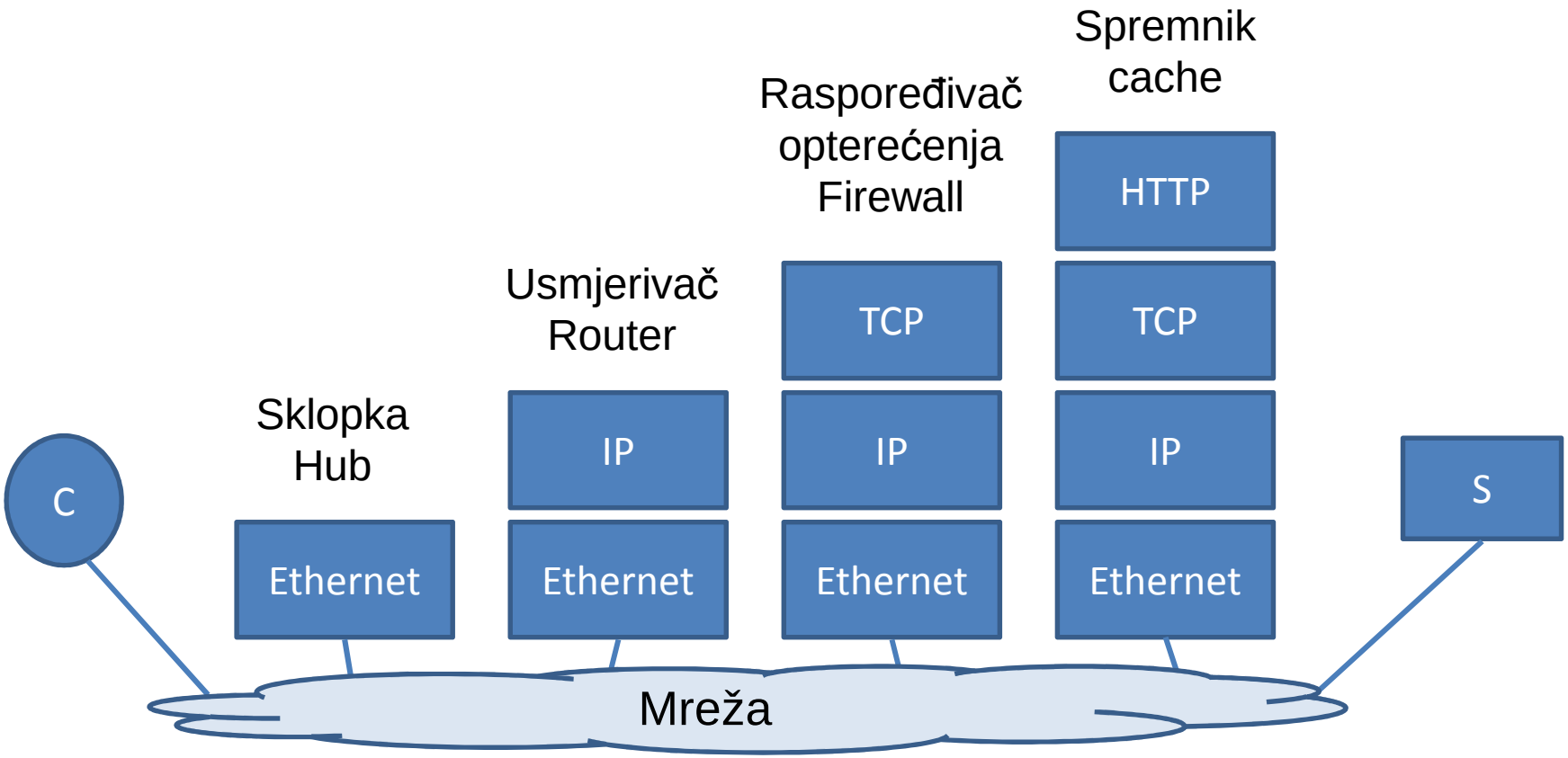
Komunikacijski protokoli u globalnoj mreži Internet

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

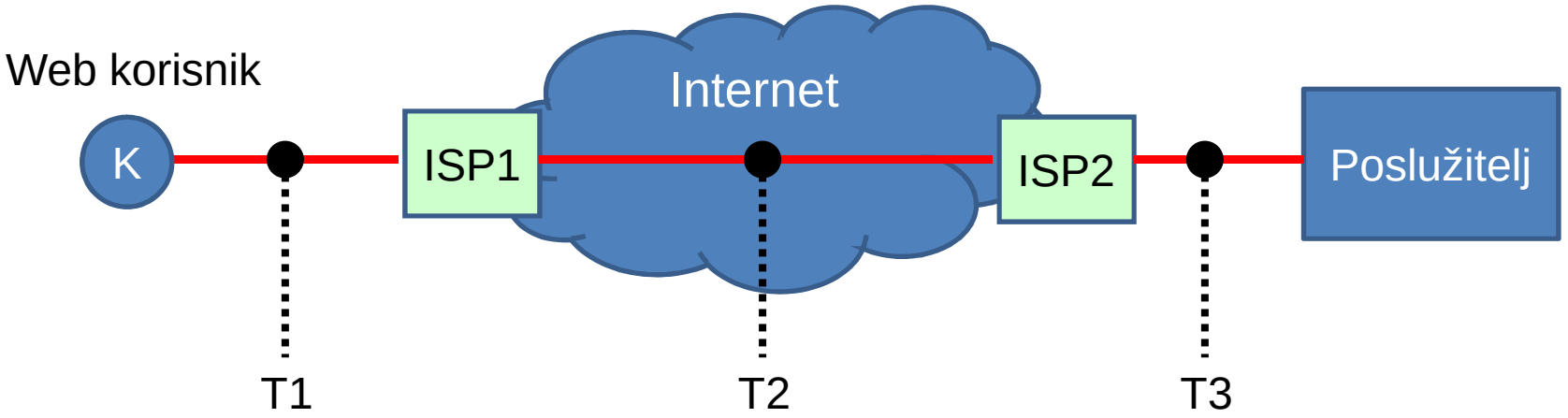


Komunikacijski protokoli u globalnoj mreži Internet

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

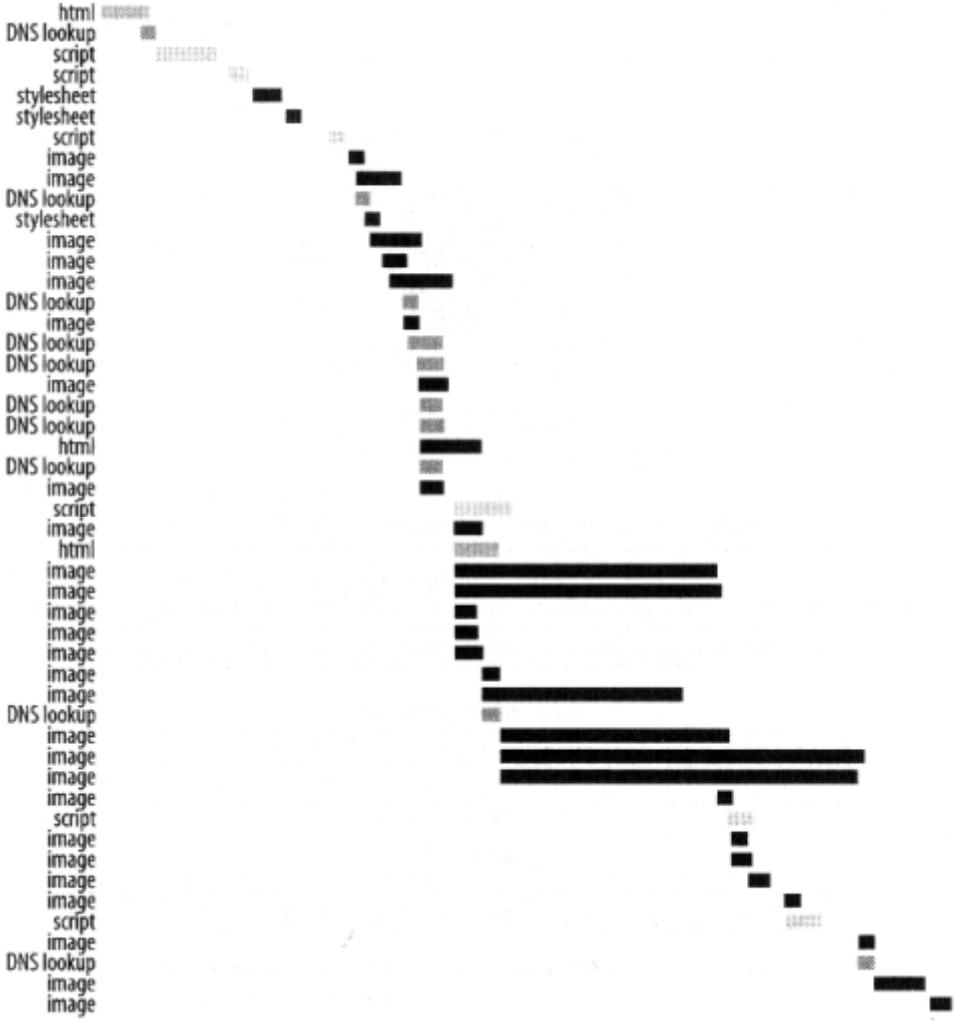


Poslužitelj-korisnik sustavi



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

Primjer dobavljanja Web stranice: My Space

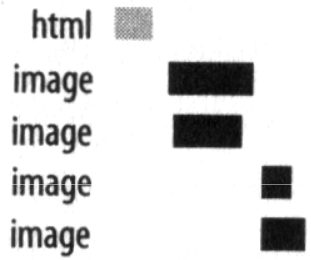


Vremenski dijagram dohvaćanja objekata stranice

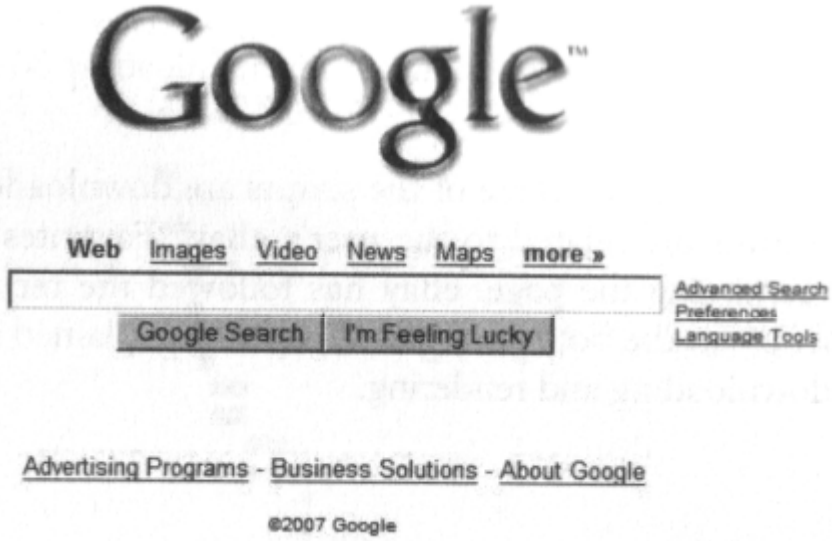
Veličina: 205K	Vrijeme: 7.8 s
Broj zahtjeva: 39	YSlow: D

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

Ostvarivanje razmjernog rasta aplikacije

- **Osnovni modeli ostvarivanja razmjernog rasta kapaciteta aplikacija**
  - **Vertikalno skaliranje** podrazumijeva prijelaz na server sa većim kapacitetom
  - **Horizontalno skaliranje** podrazumijeva dodavanje paralelnih servera (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!

Horizontalno 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

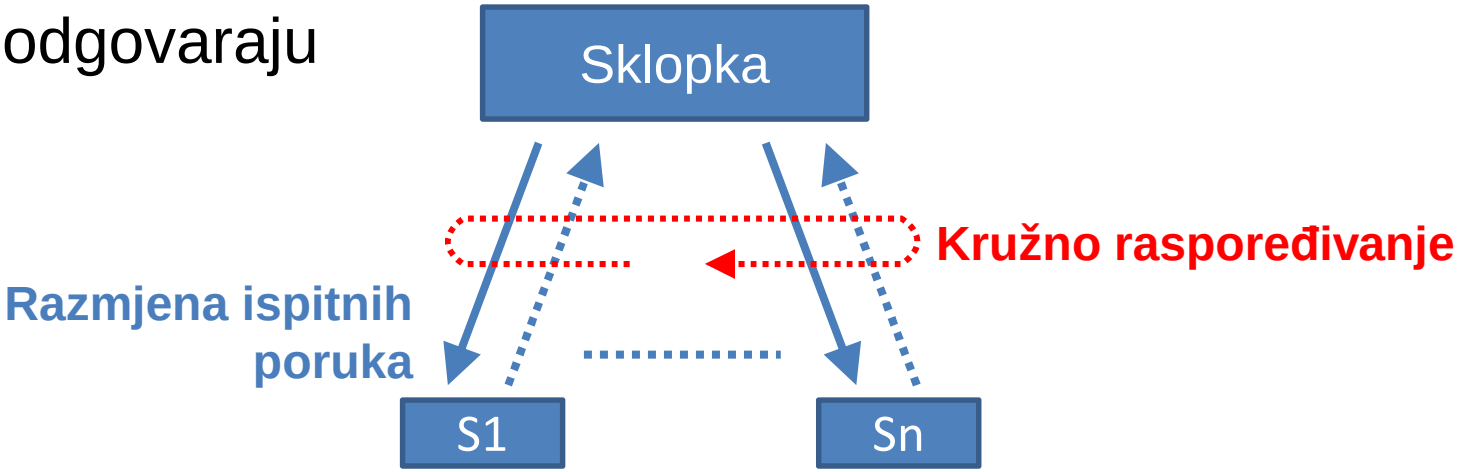
- **U slučaju web aplikacija, u praksi se koriste oba termina jer se visoka dostupnost osigurava kroz neosjetljivost na greške**

- **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

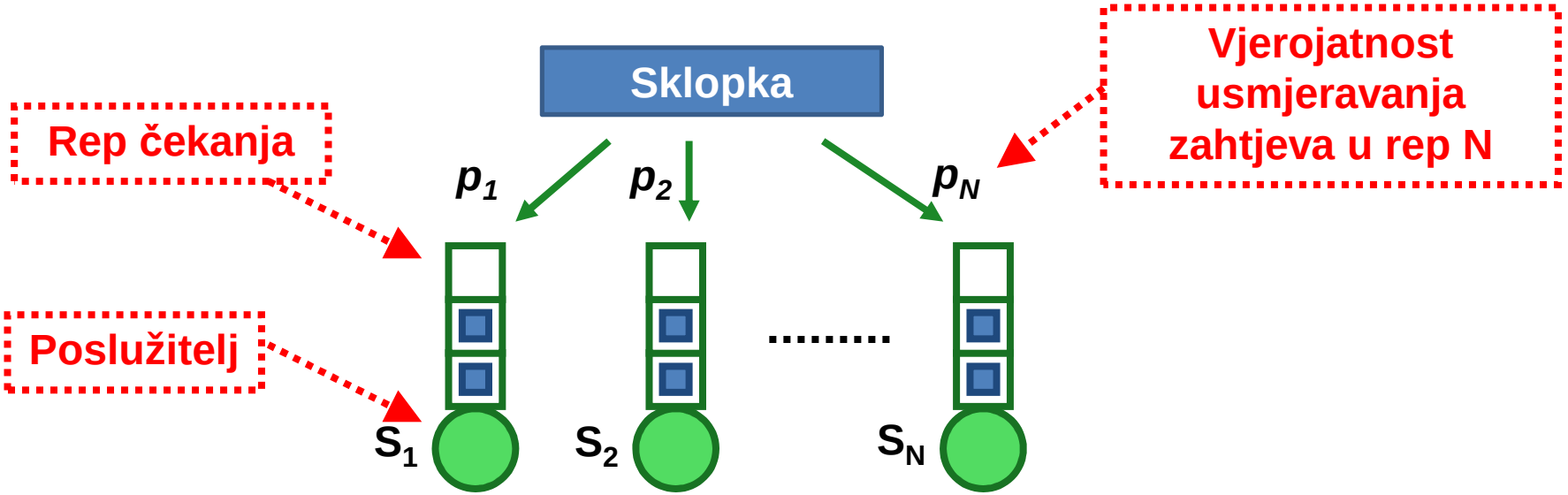
Raspoređivanje opterećenja u sustavu

- **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



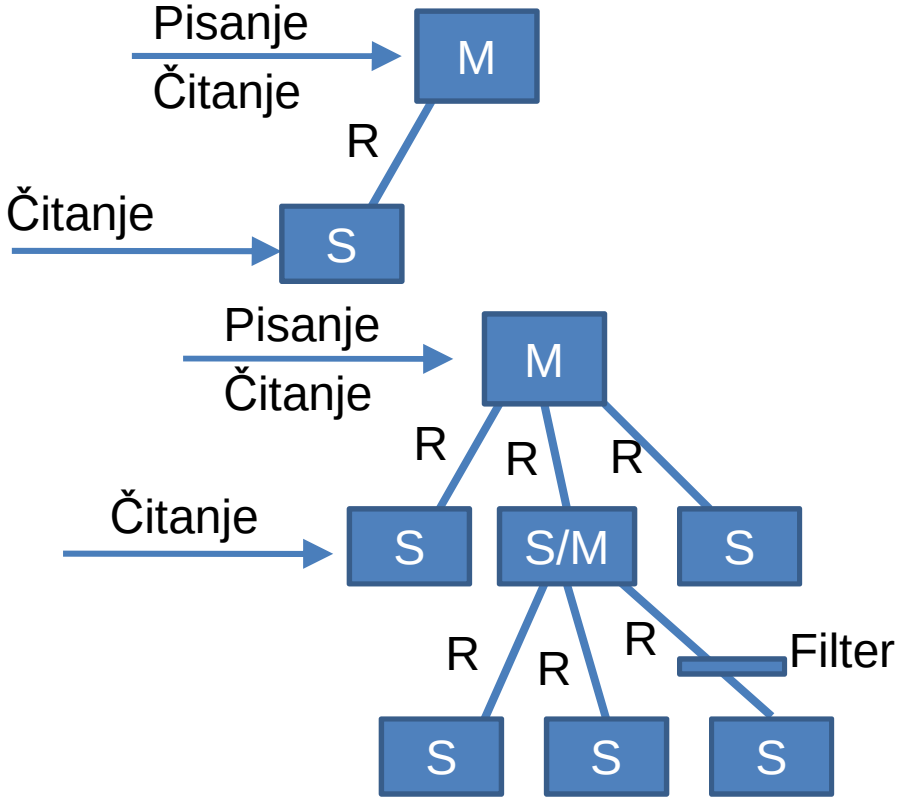
- **Način posluživanja određuje vjerojatnosti raspoređivanja zahtjeva na poslužitelje (p_1…p_n)**

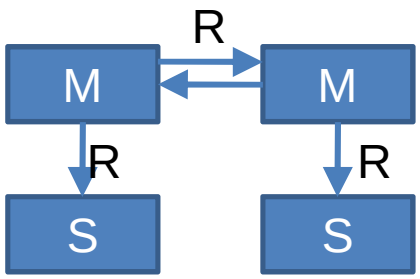
Raspoređivanje opterećenja na aplikacijskoj razini

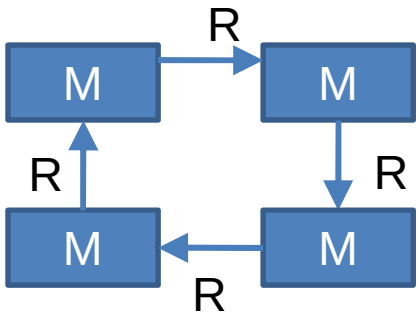
- **HTTP**
  - Obično na razini URL tako da se različite stranice dohvaćaju iz različitih spremnika
- **DNS**
  - Pruža mogućnost geografske podjele
- **SMTP**
  - Elektronička pošta je zasnovana na ASCII protokolu te se može lako prosljeđivati prema ciljnoj adresi

Raspoređivanje podataka

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

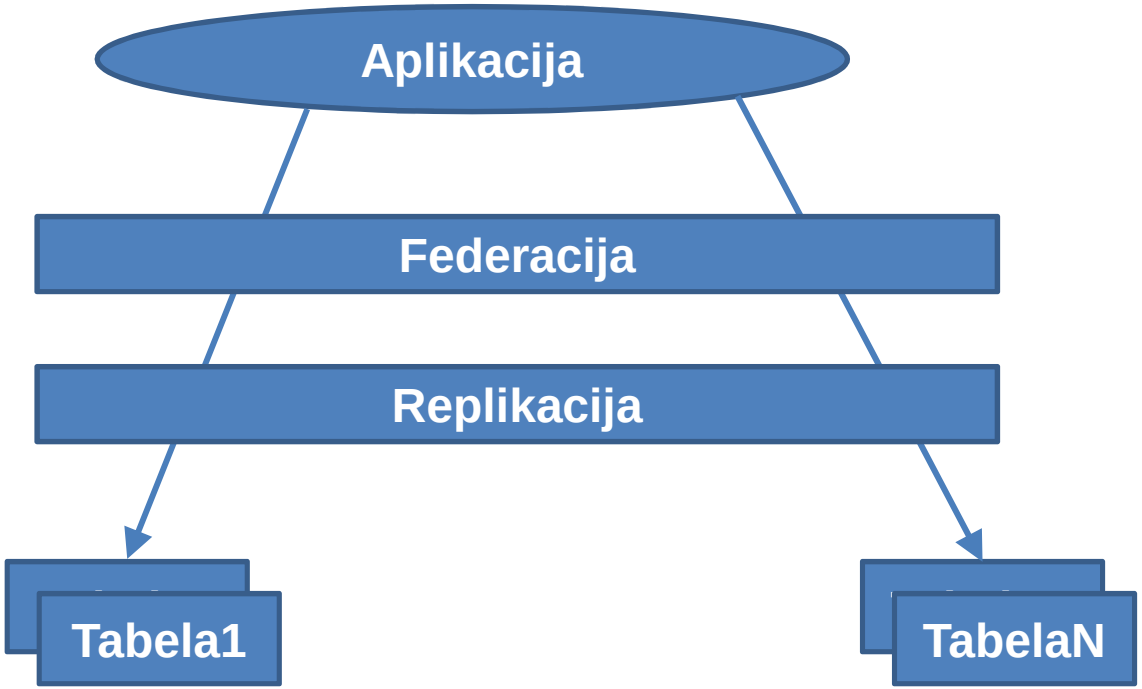






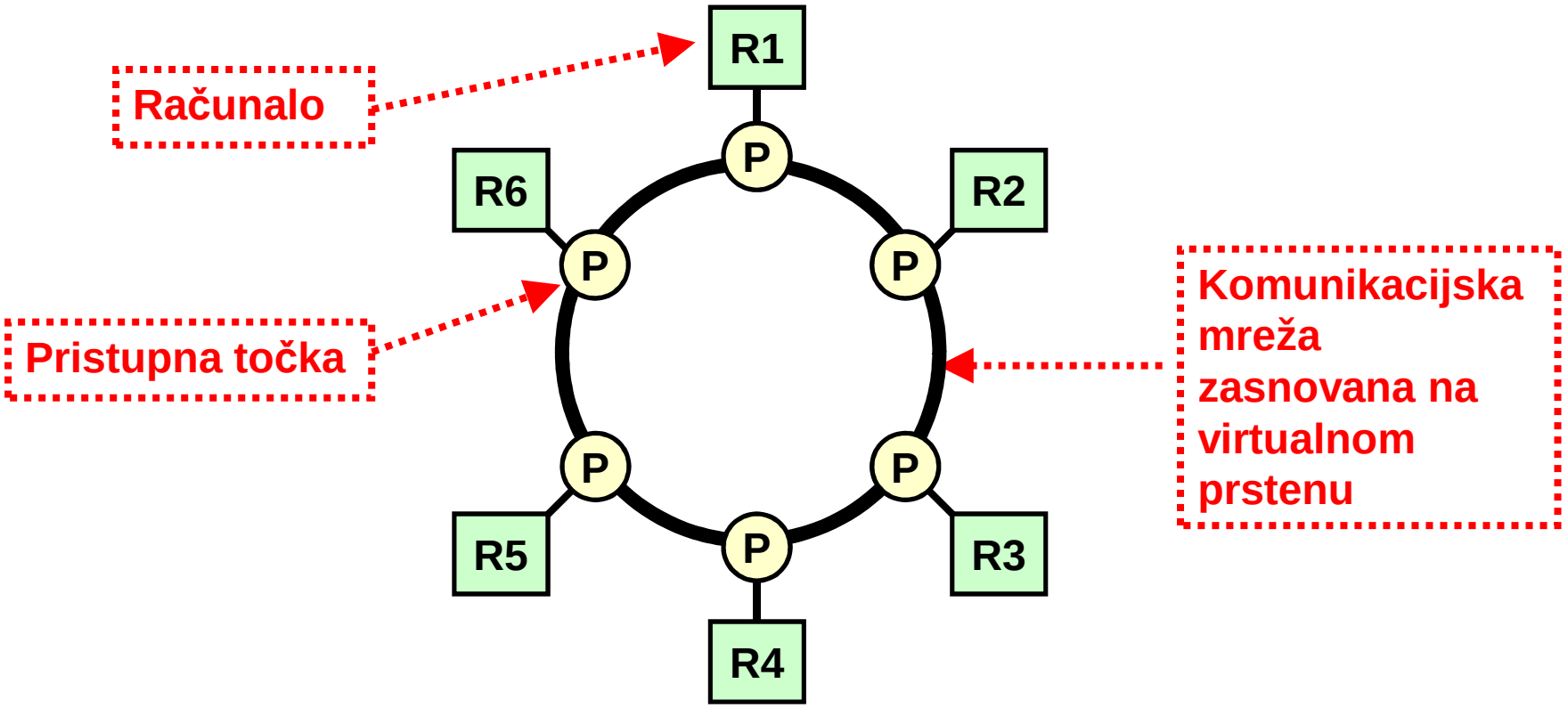
Raspoređivanje podataka

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

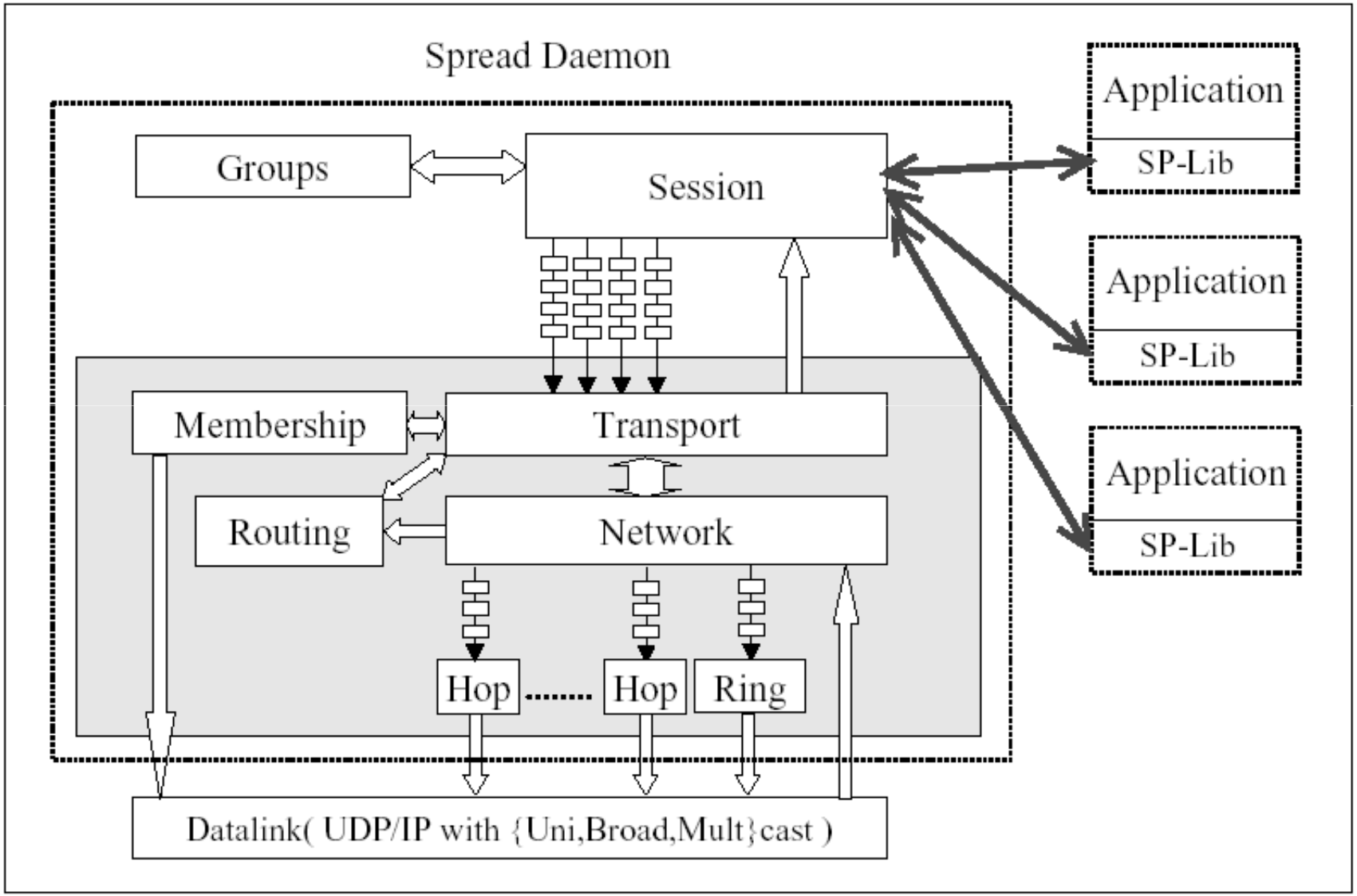


Sinkronizacija i protokoli za komunikaciju u grupi

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



Arhitektura čvora

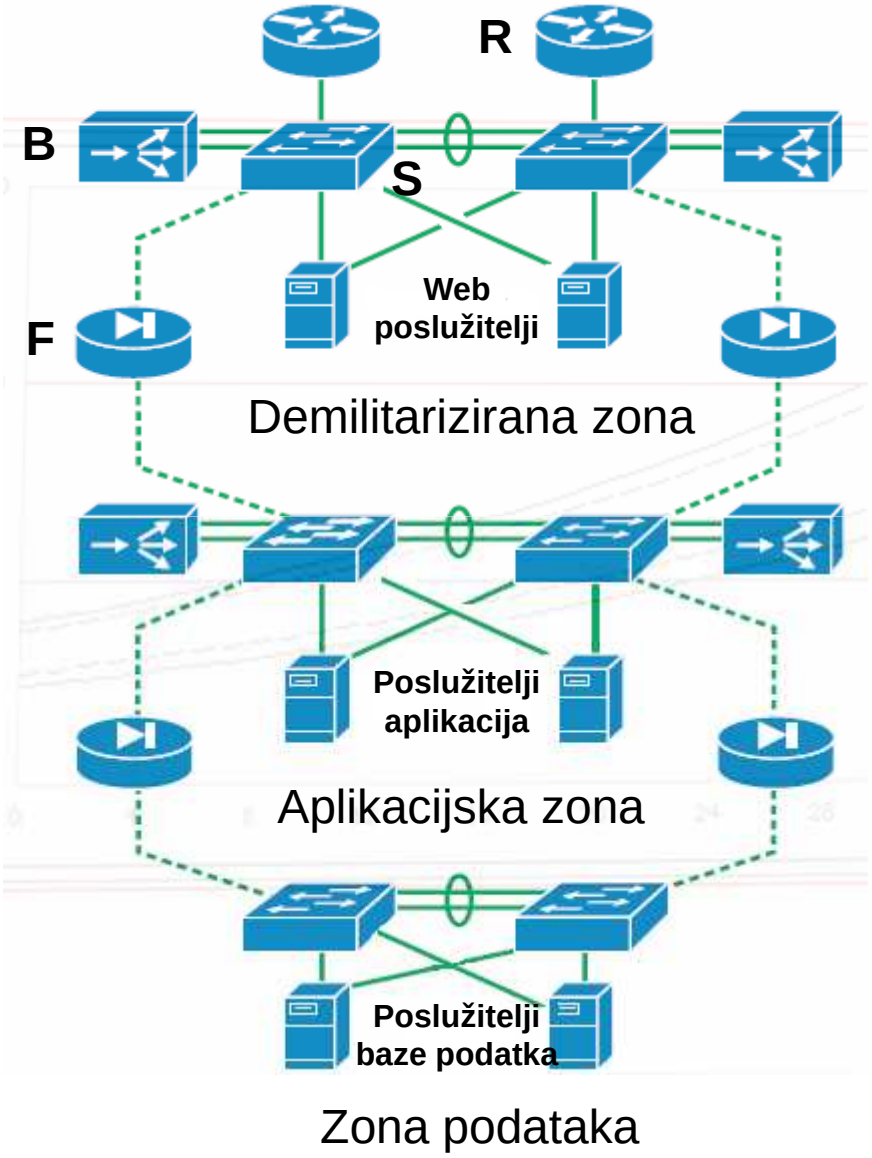


Garancije slijeda poruka u protokolima

- **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

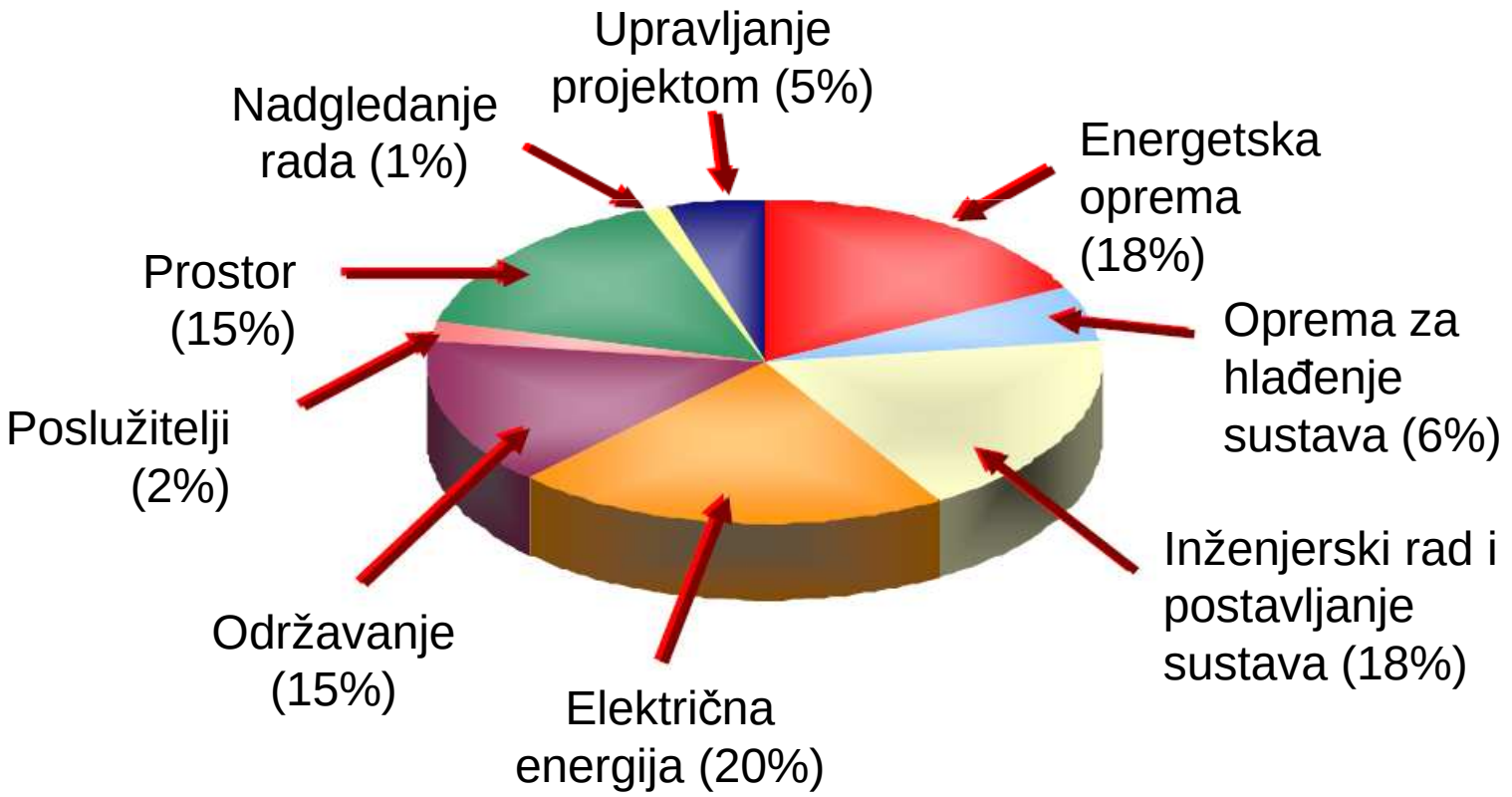
Tipična arhitektura web poslužitelja

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



Ukupna cijena vlasništva

- **Prosječni trošak uporabe sustava poslužitelja u nizu (rack)**
  - $120K kroz tri godine korištenja (CAPEX/OPEX = 50/50)

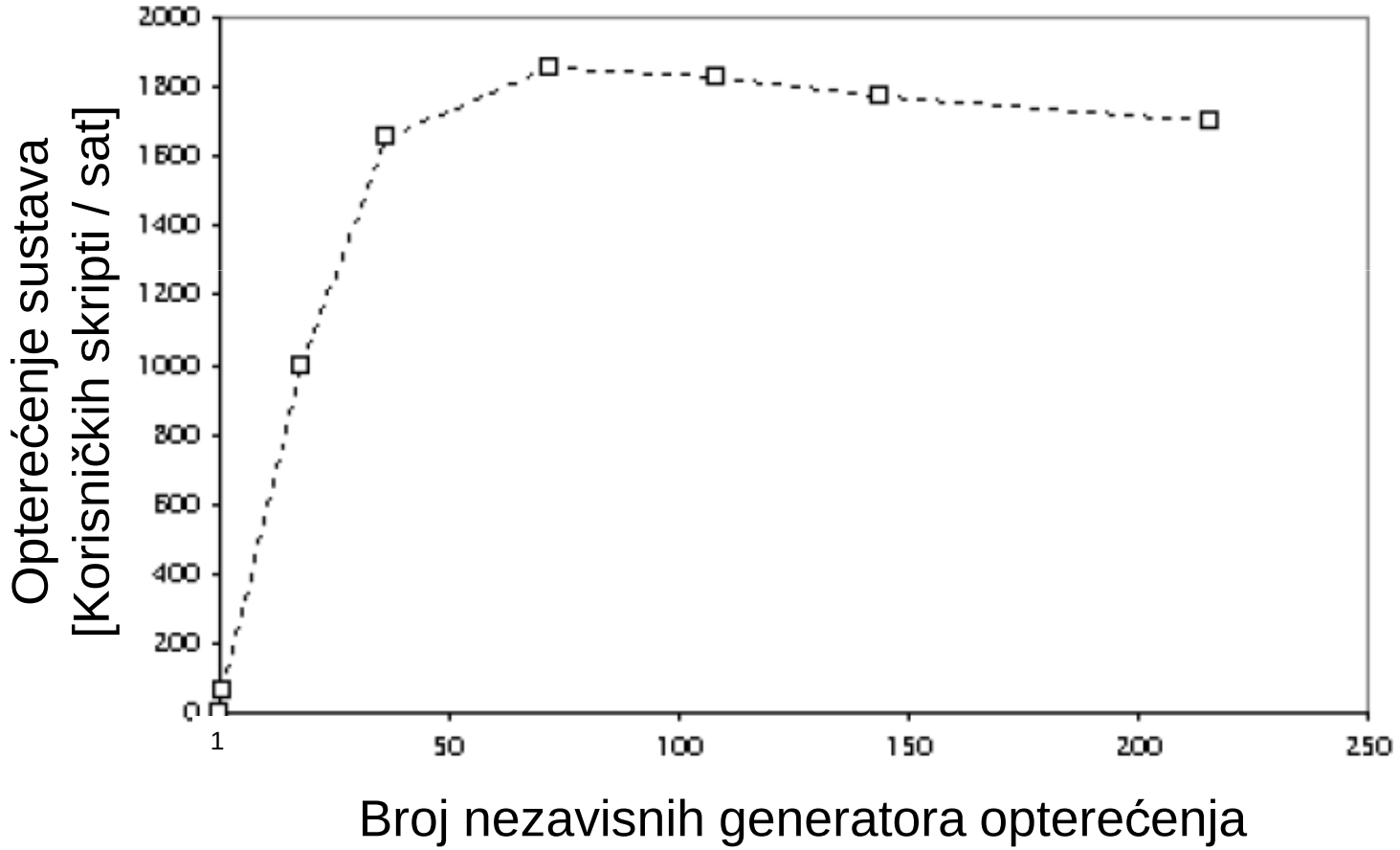


Mogući načini ostvarenja sustava

- **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
    - Operacijski sustav s listom poznatih aplikacija
  - Dedicirani sustavi
  - Zajednički sustavi
- **Izgradnja vlastitog poslužitelja**

Mjerenje propusnosti sustava

- **Mjerenja se ostvaruju primjenom generatora ispitnog opterećenja** (*Load generatora*)



Modeliranje kapaciteta

$$S(\alpha,\beta,N)_{future} = \frac{N}{1+\alpha\left[(N-1)+\beta N(N-1)\right]} \quad [1]$$

Parametri α i β povezani su sa degradacijom performansi zbog čekanja na zajednička sredstva i kašnjenja zbog održavanja koherencije

Slijedeća formula omogućava modeliranje u Excel-u

$$= \frac{Nr}{1+A1\left((Nr-1)+B1*Nr*(Nr-1)\right)}$$

A1 = α i B1 = β
Nr = vrijednost N u redu r

[1] *Gunther, N. J., "Hit-and-Run Tactics Enable Guerrilla Capacity Planning", IEEE IT Professional, Jul-Aug, 2002, pp. 40-46*

Određivanje parametara

- **Osnovni koraci:**

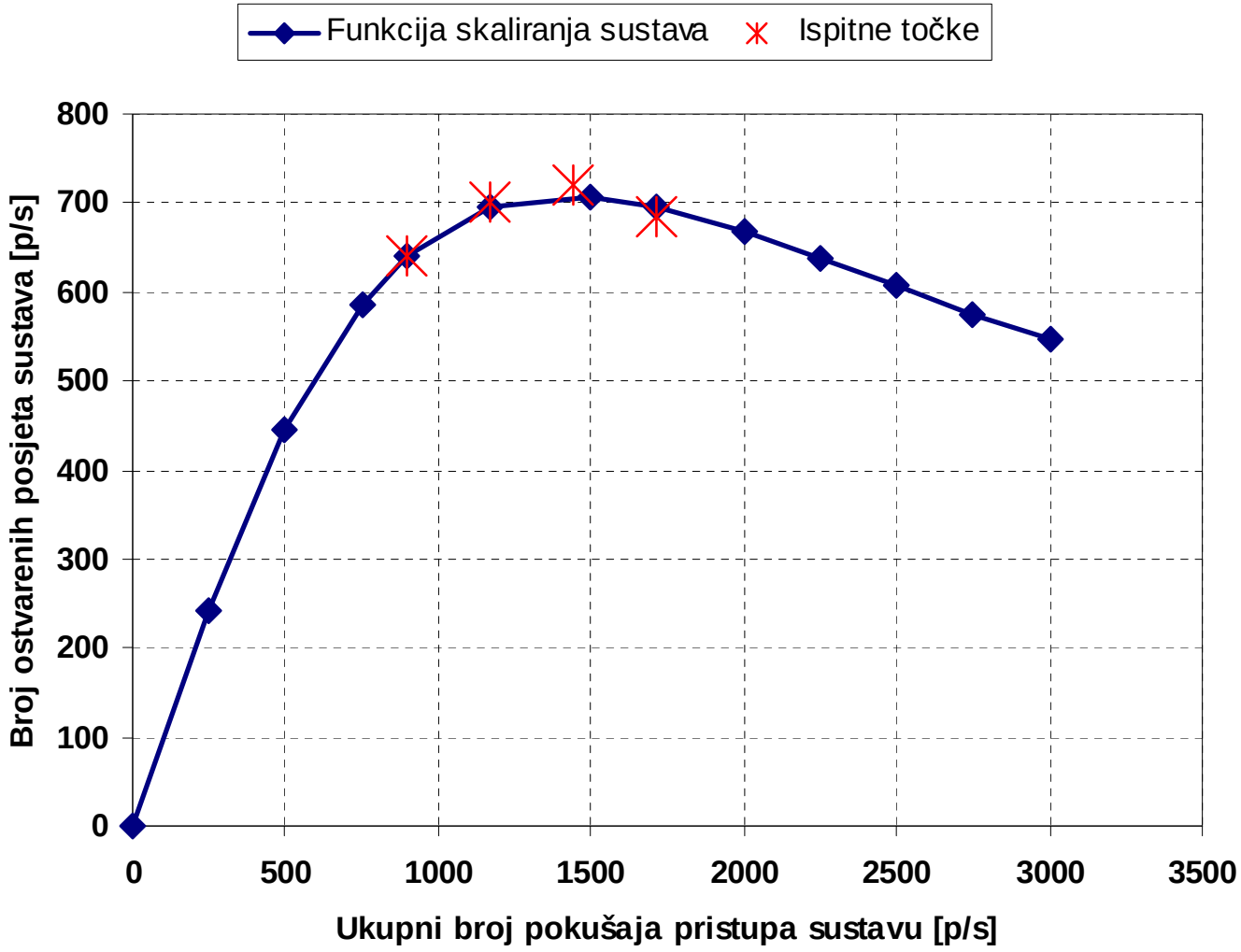
1) Izmjeri performanse kao funkciju od *N* koristeći alate kao *WebLoad* ili *LoadRunner*

2) Obično je dovoljan mali uzorak (najmanje 4 točke)

3) Odredi α i β primjenom regresije koristeći alat EXCEL

4) Koristi dobivene vrijednosti za izračunavanje skaliranja prema Excel modelu iz prethodnog teksta

Uporaba regresije za određivanje funkcije skaliranja



Skaliranje.xls

Prognoza potrebnog kapaciteta

- **Ako se iskorištenje servera U mjeri u pravilnim intervalima, dugoročna potreba za kapacitetom se može ustanoviti podrazumijevajući eksponencijalni trend model**

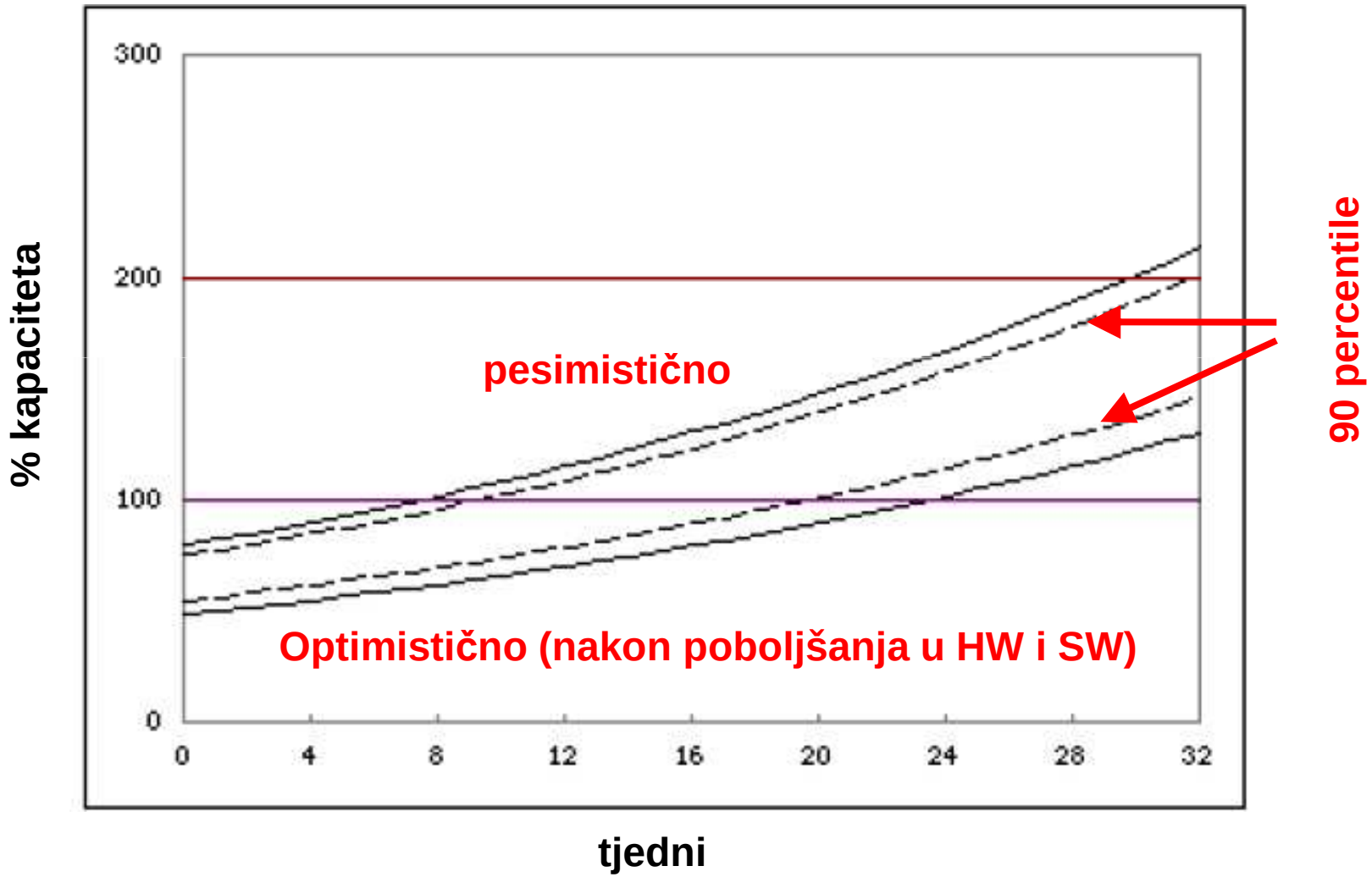
$$U_{future} = U_{now} * e^{L*W}$$

L = trend rasta, određen primjenom Excel opcije "*Add Trendline*“
W = broj tjedana kroz koje je trend aproksimiran

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

$$T_{double} = \frac{\ln(2)}{L}$$

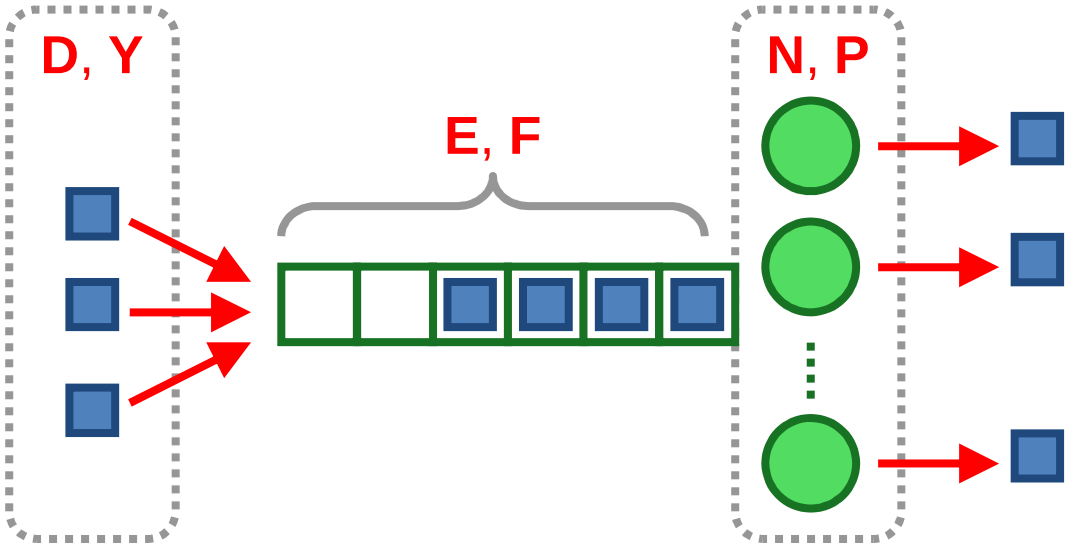
Prognoza kapaciteta kroz 32 tjedna



Vrednovanje performansi sustava teorijom repova

Osnovni pojmovi teorije repova

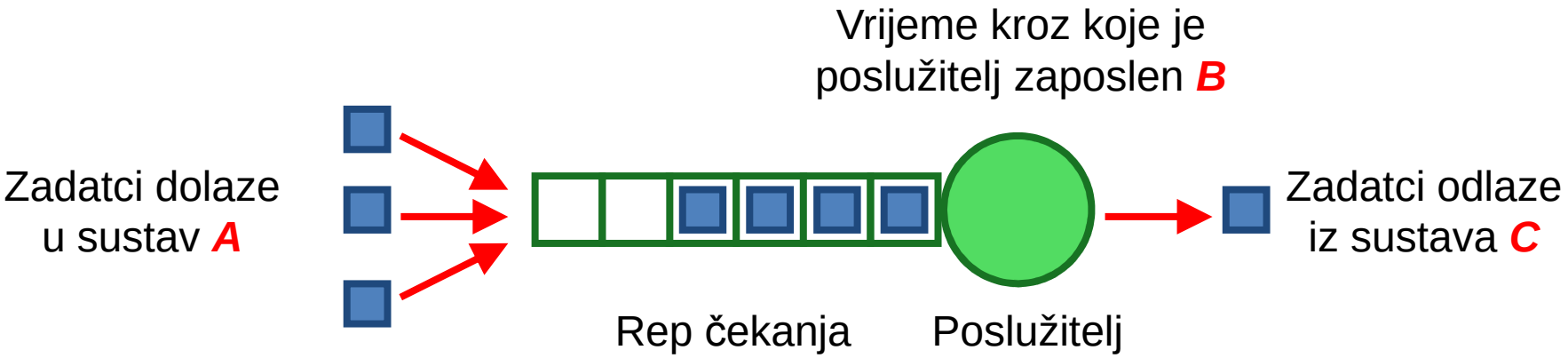
- **Zapis značajki modela D/P/N/Y/E/F (*Kendall* notacija)**



- ***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, …)

Osnovni pojmovi teorije repova

- **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!

$$Q = X * R$$

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)

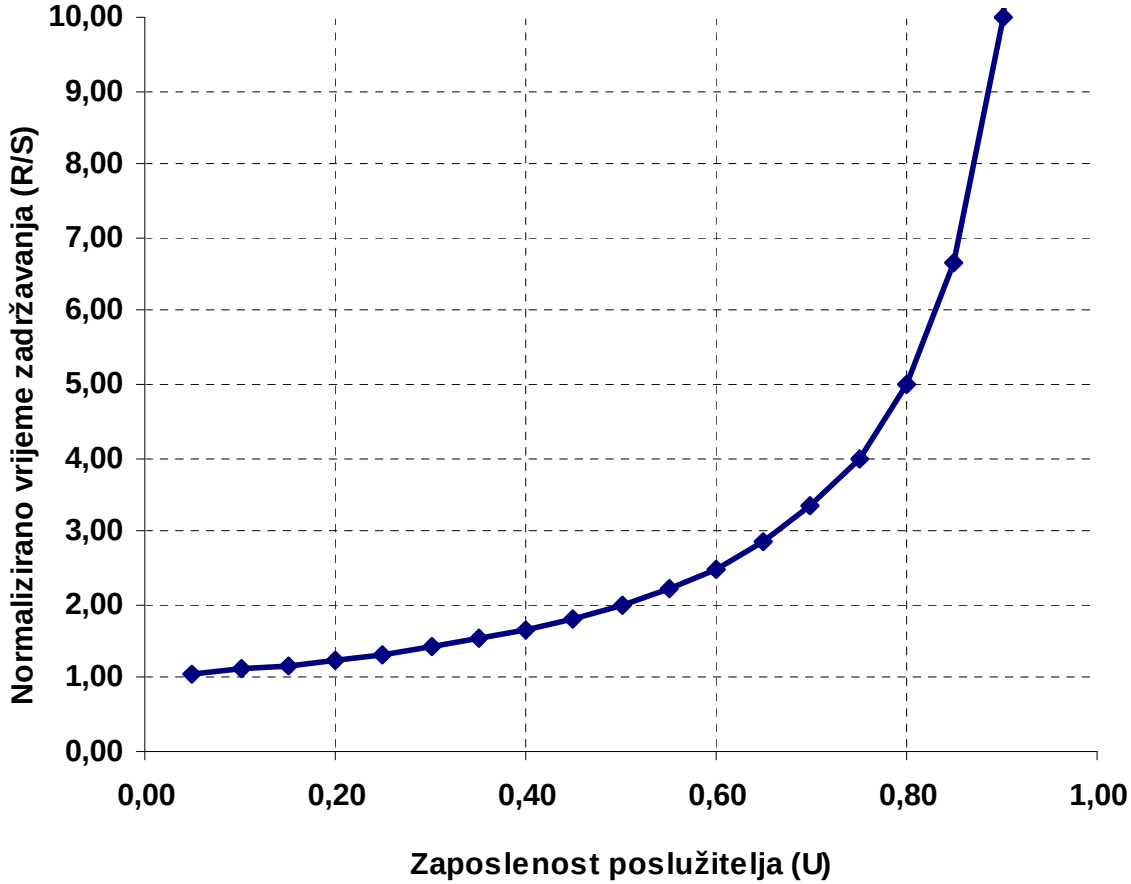
    R = S/(1 - U) = (0.002 s/p)/(1 - 0.25) = 0.0026666 s
  - Srednji broj paketa u repu (Q)

    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

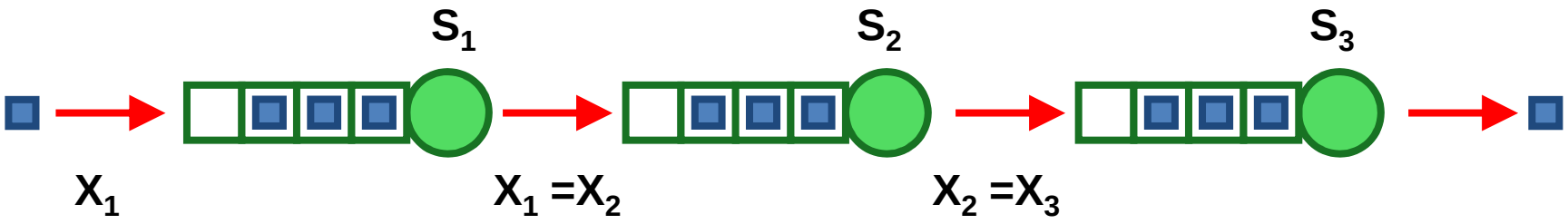
pr4.c

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 $R_N = S_N /(1 – X*S_N)$

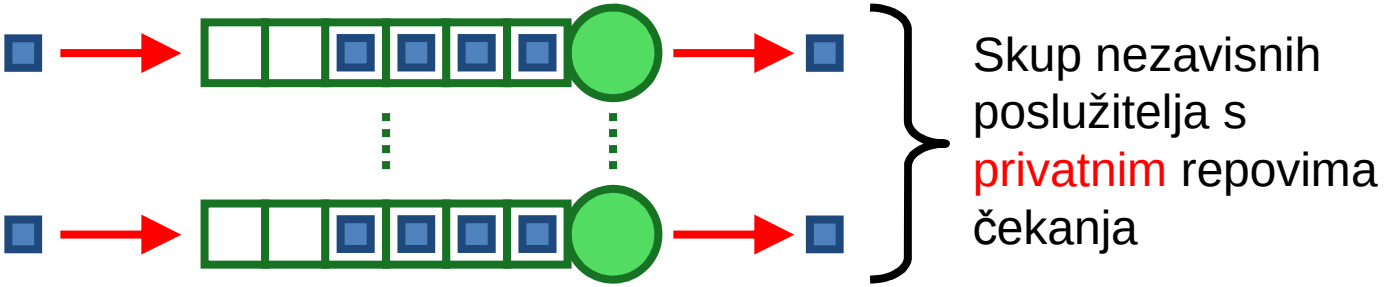
    $R_1 = 1.11$s, $R_2 = 2.5$s, $R_3 = 4.29$s
  - Prosječni broj zahtjeva u repu Q

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

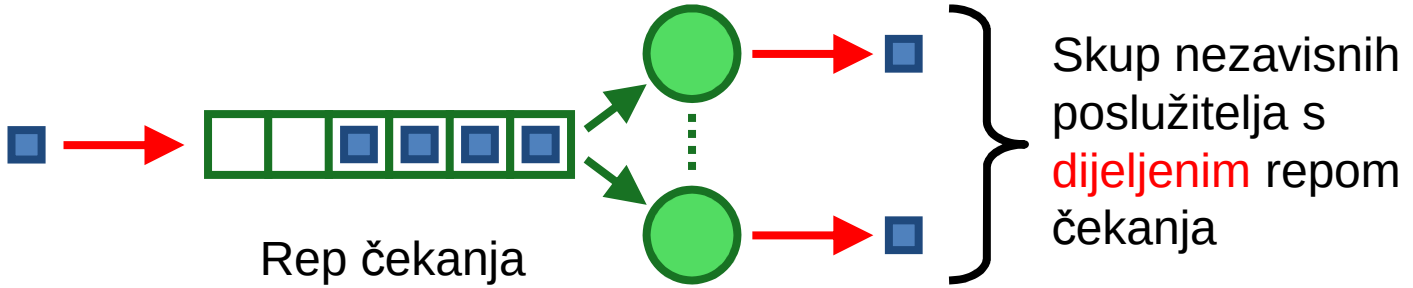
pr5.c

Višestruki paralelni repovi i poslužitelji

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



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



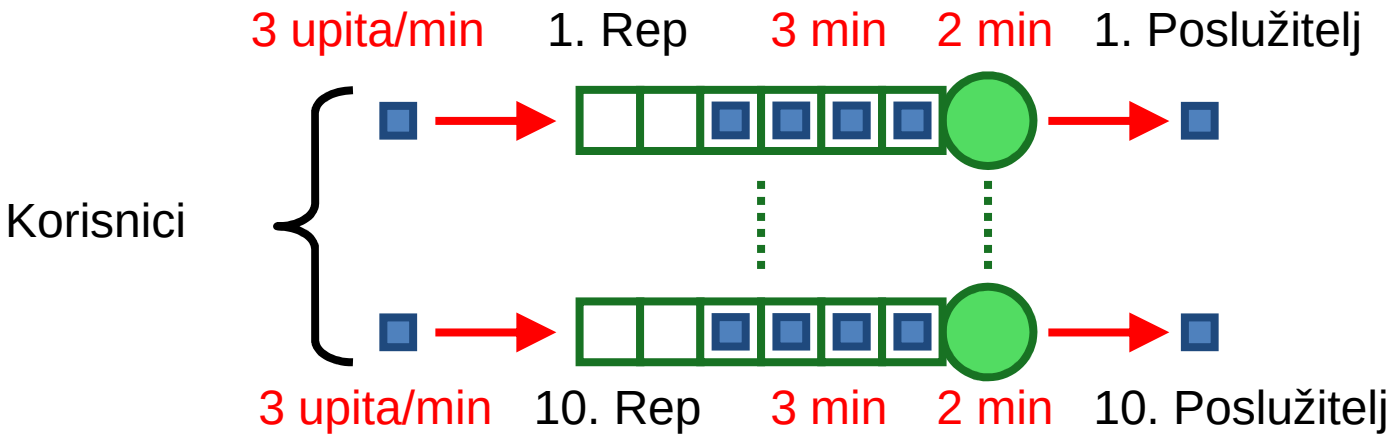
Sustav više paralelnih repova čekanja

- **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
  - Broj pristiglih zahtjeva u jednom repu L = 3 z/min

  - Prosječna zaposlenost sustava (U)

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

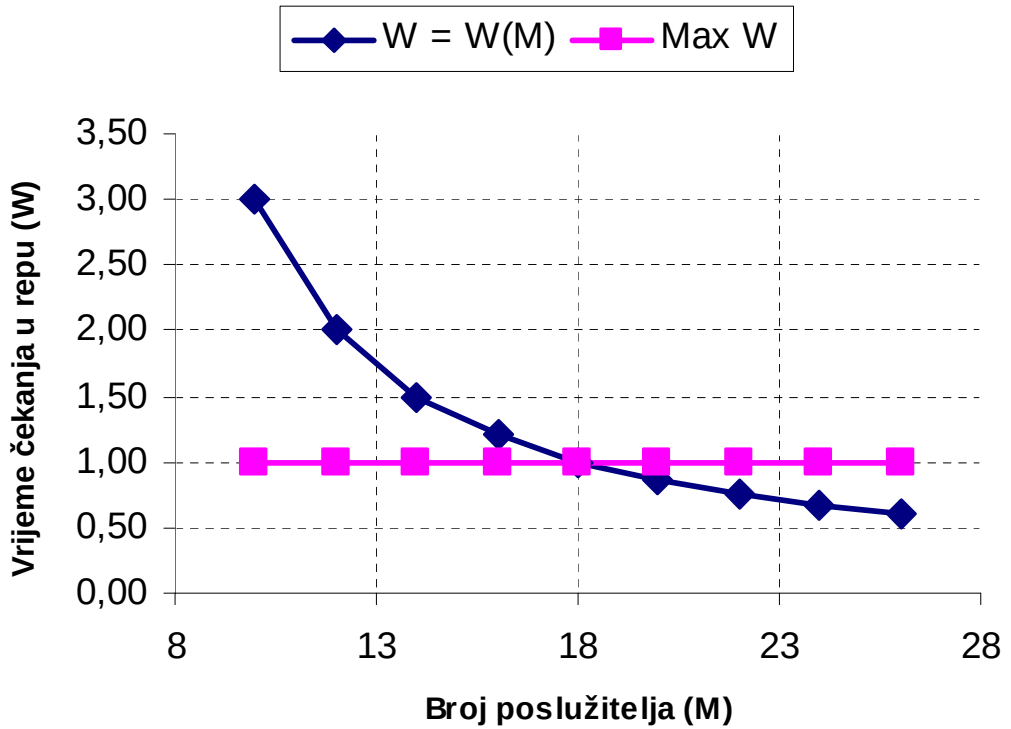
    ro = U/N = 6/10 = 0.6
  - Srednje vrijeme zadržavanja korisnika u sustavu (R)

    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**
  - Za zadani sustav ne postoji analitičko rješenje. Rješenje se određuje 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?

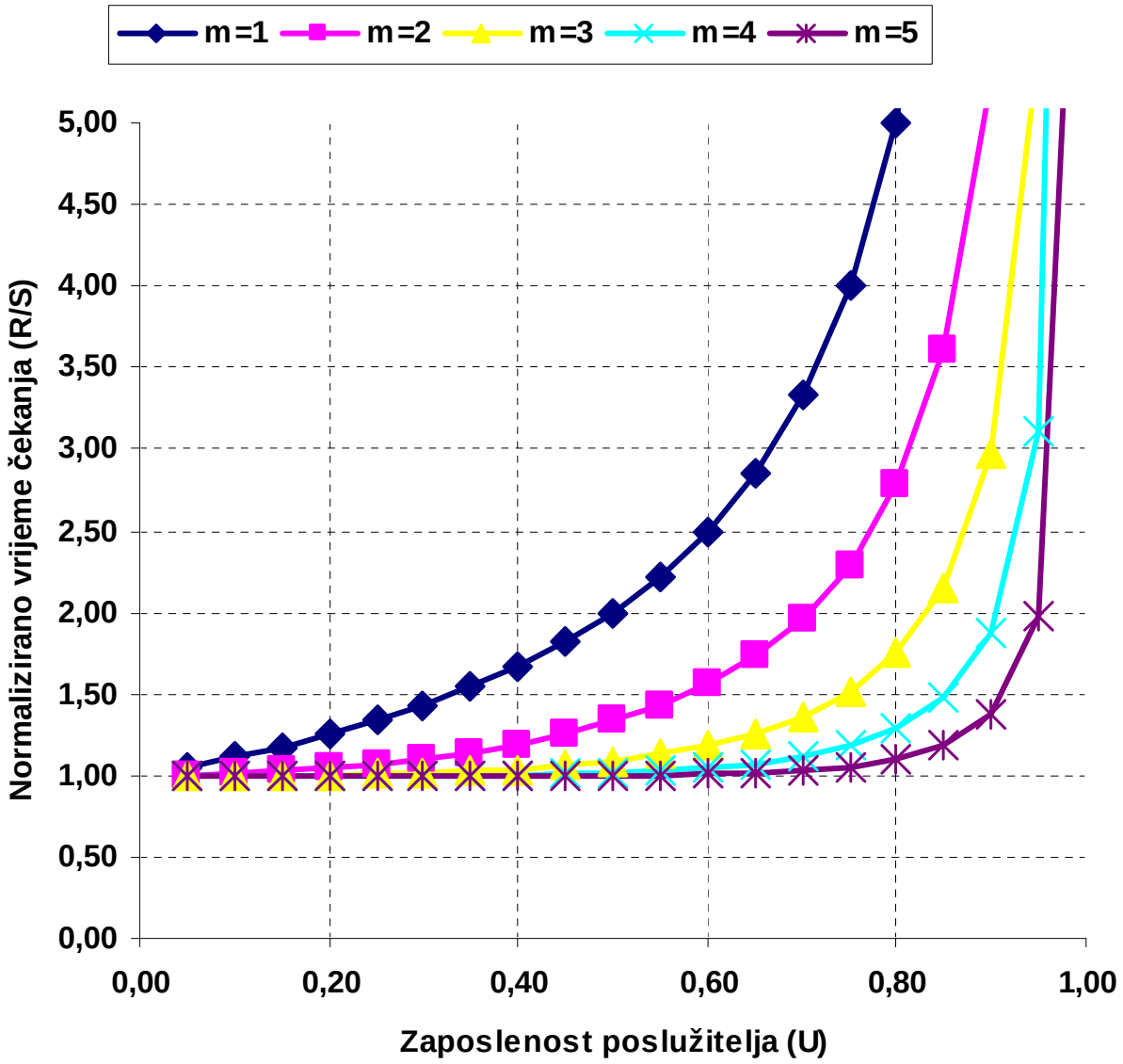
Sustav s jednim repom i dva poslužitelja

- **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

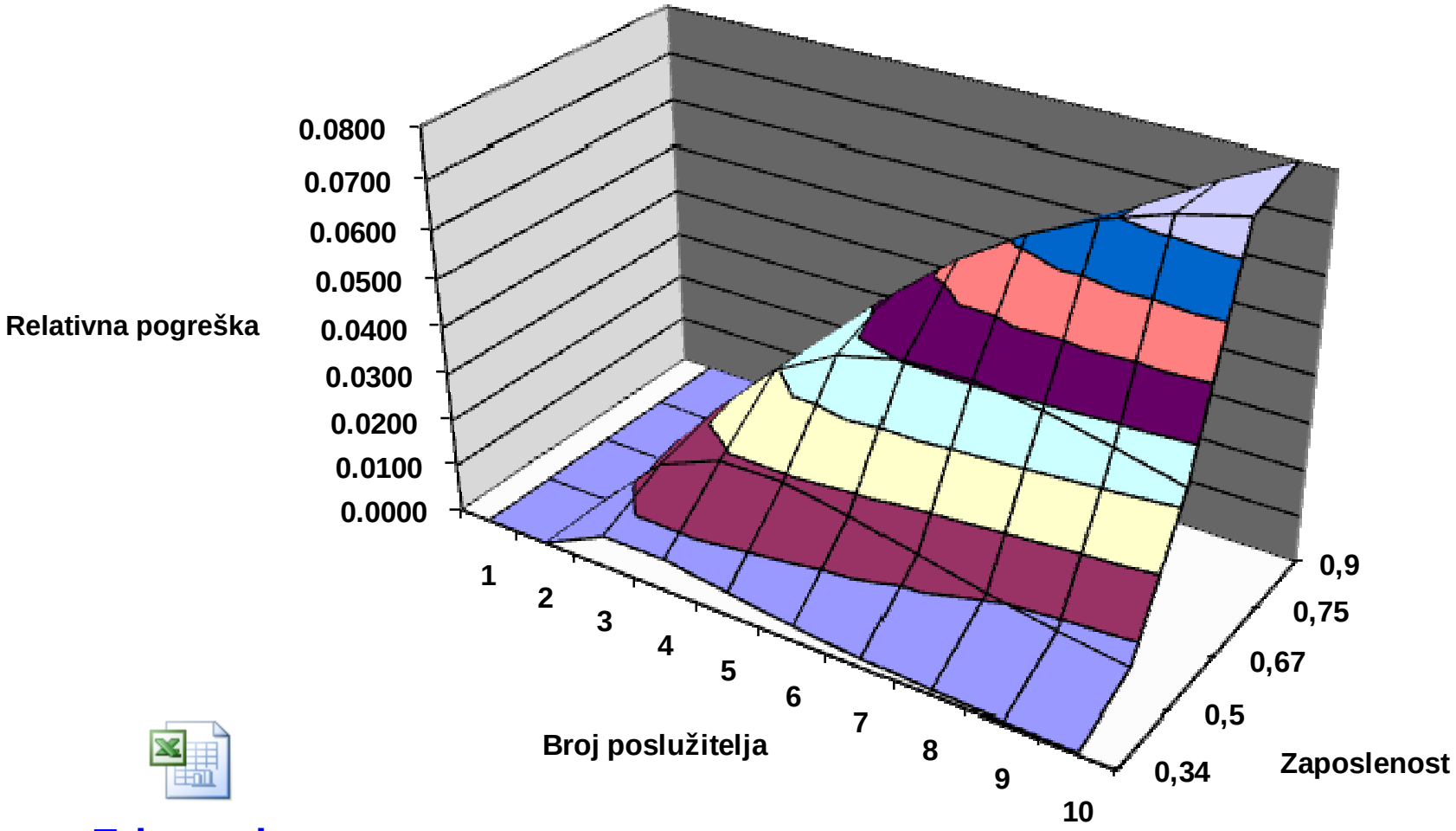
$$R = S * \left[1 + \frac{C(N, ro)}{N * (1 - ro)} \right]$$

  - 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

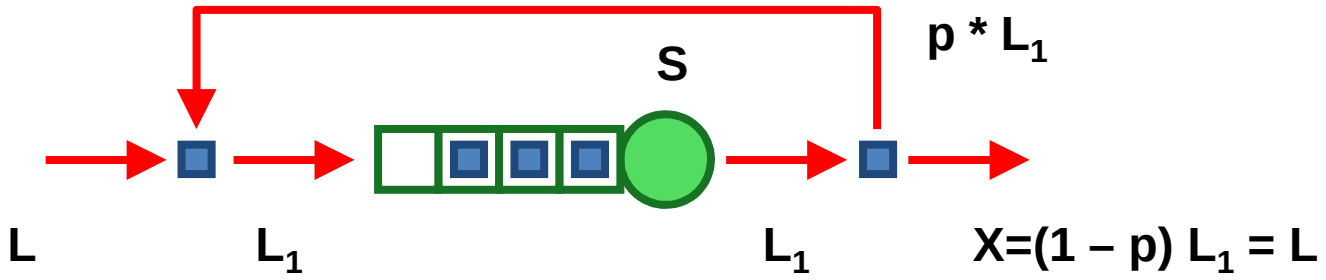
Pogreška aproksimacije



Erlang.xls

Poslužitelji s povratnom vezom

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

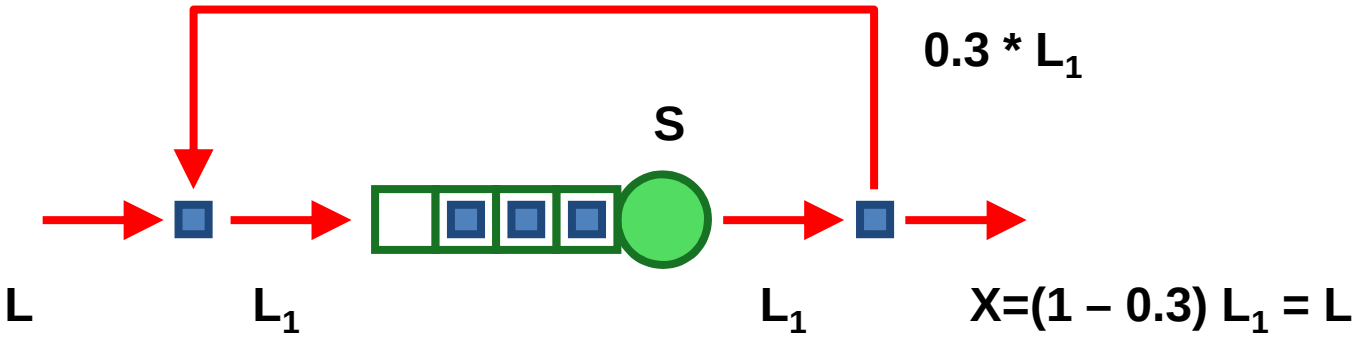


- Dva Poissonova procesa L1 i L2 rezultiraju u sa novim Poissonovim procesom L = L1 + L2
- $L_1 = L + L_1 p = L / (1 - p)$
- $U = L_1 * S = L*S/(1 - p)$
- $R_1 = S/(1 – U)$ (zadržavanje za jedan prolaz)
- Vrijeme zadržavanja u sustavu s povratnom vezom

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

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

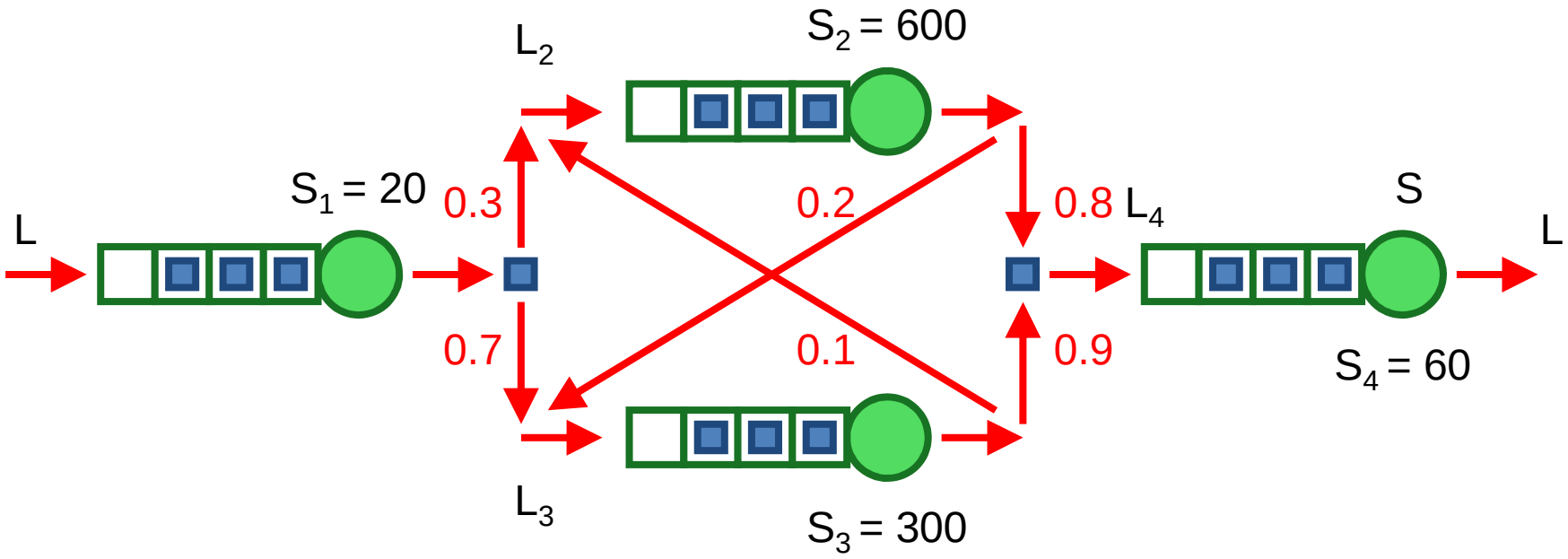
    $W = S*U / (1 – U) = 0.866$ s/p
  - Srednje vrijeme zadržavanja paketa u kanalu (R1)

    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

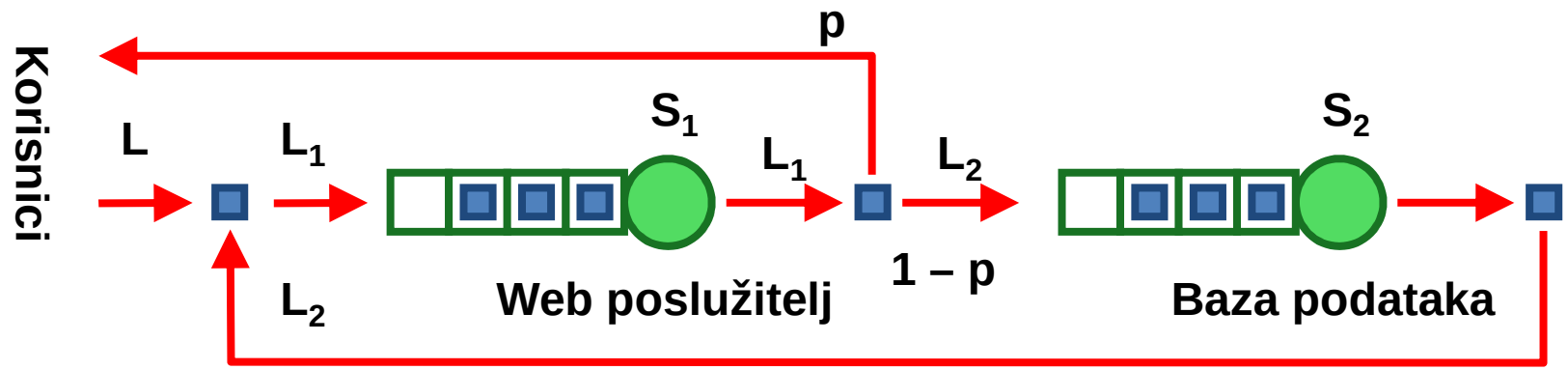
- **Mreža repova poruka**
  - Mrežna struktura proizvoljne složenosti s povratnim granama



Što znamo na temelju prethodnih razmatranja

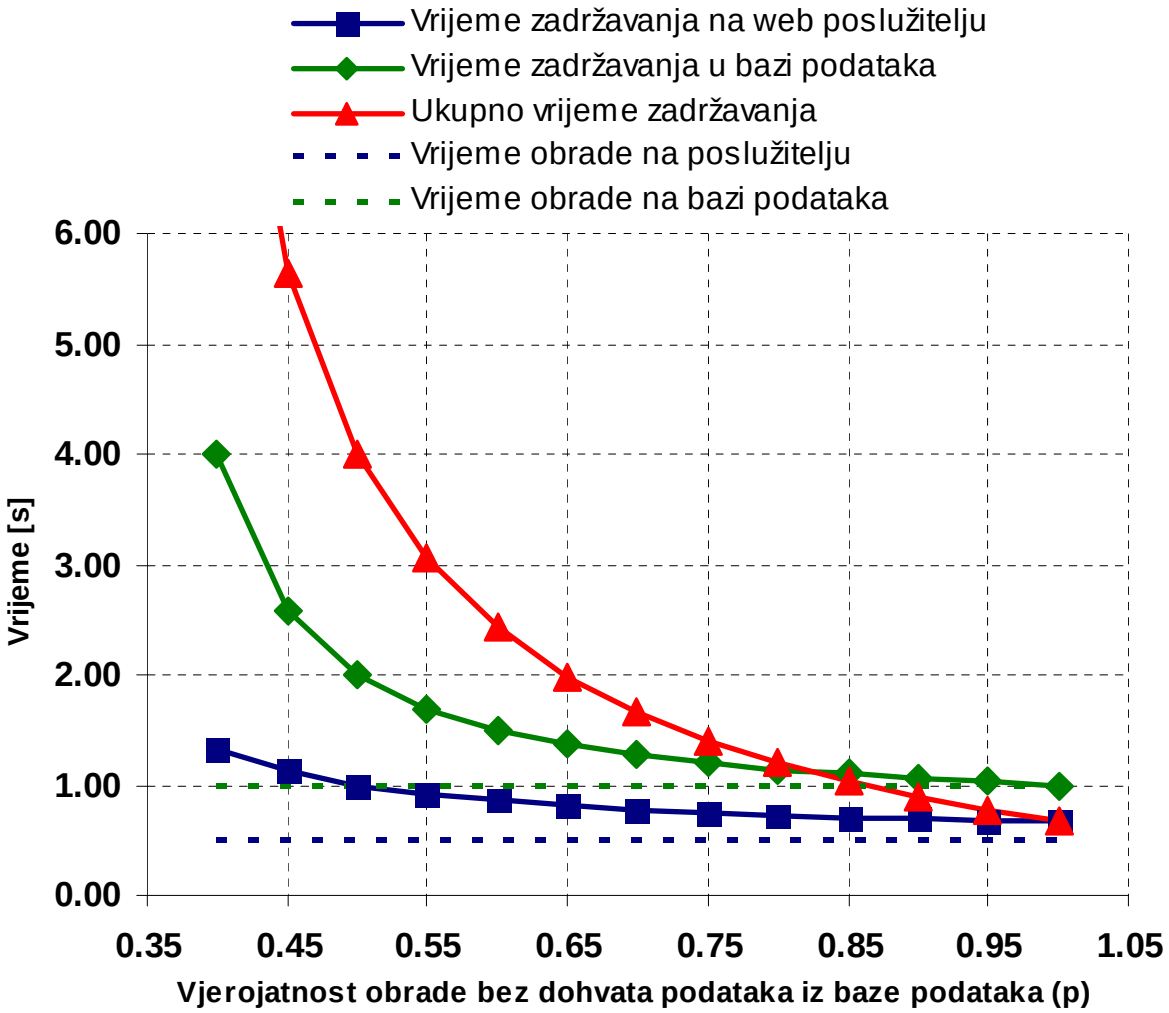
- **U stabilnom stanju sustava**
  - $U_N = X_N * S_N$ ($X_N = L_N$ za stabilni slučaj)
  - $U_1 = L * S_1 = 20 * L$
  - $U_2 = 600 * L_2 = 600\ (0.3 * L + 0.1 * L_3)$
  - $U_3 = 300 * L_3 = 300\ (0.7 * L + 0.2 * L_2)$
  - $U_4 = 60 * L$
- **Nakon rješenja za L_2 i L_3 i izračunavanja U_1 do U_4, izračunavamo 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



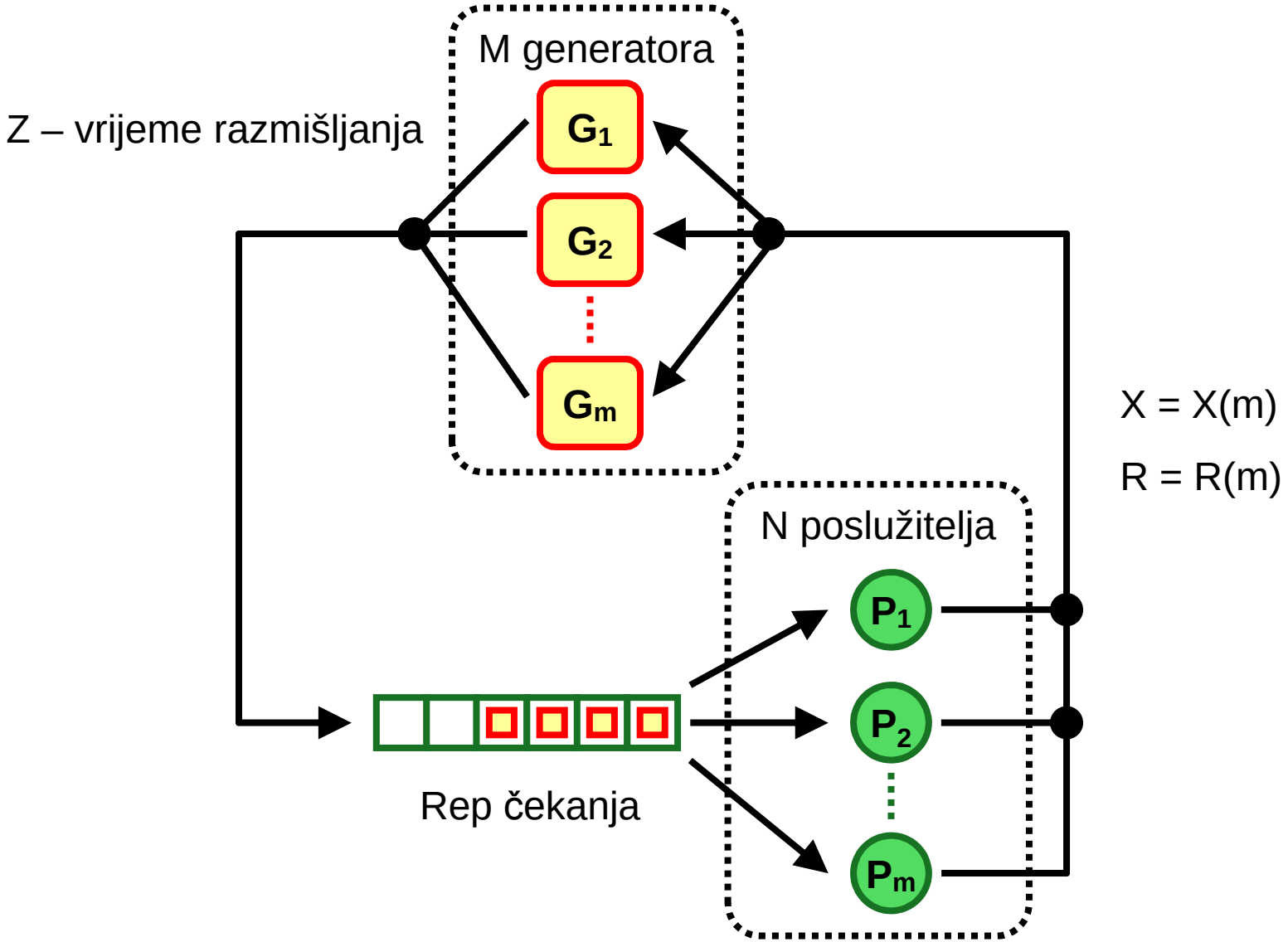
- **Učestalost dolazaka zahtjeva (L1, L2)**
  - L1 = L + L2 = L + (1-p)L1 = L/p
  - L2 = (1 – p)L1 = ((1 – p)/p)*L
- **Vrijeme zadržavanja zahtjeva u sustavu (R)**
  - X1 = L1; X2 = L2
  - U1 = X1*S1 = L*S1/p; U2 = ((1 – p)/p)*L*S2
  - R1 = S1/(1 – U1) ; R2 = S2/(1 – U2)
  - R = R1 * (1 + (1 – p)/p)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

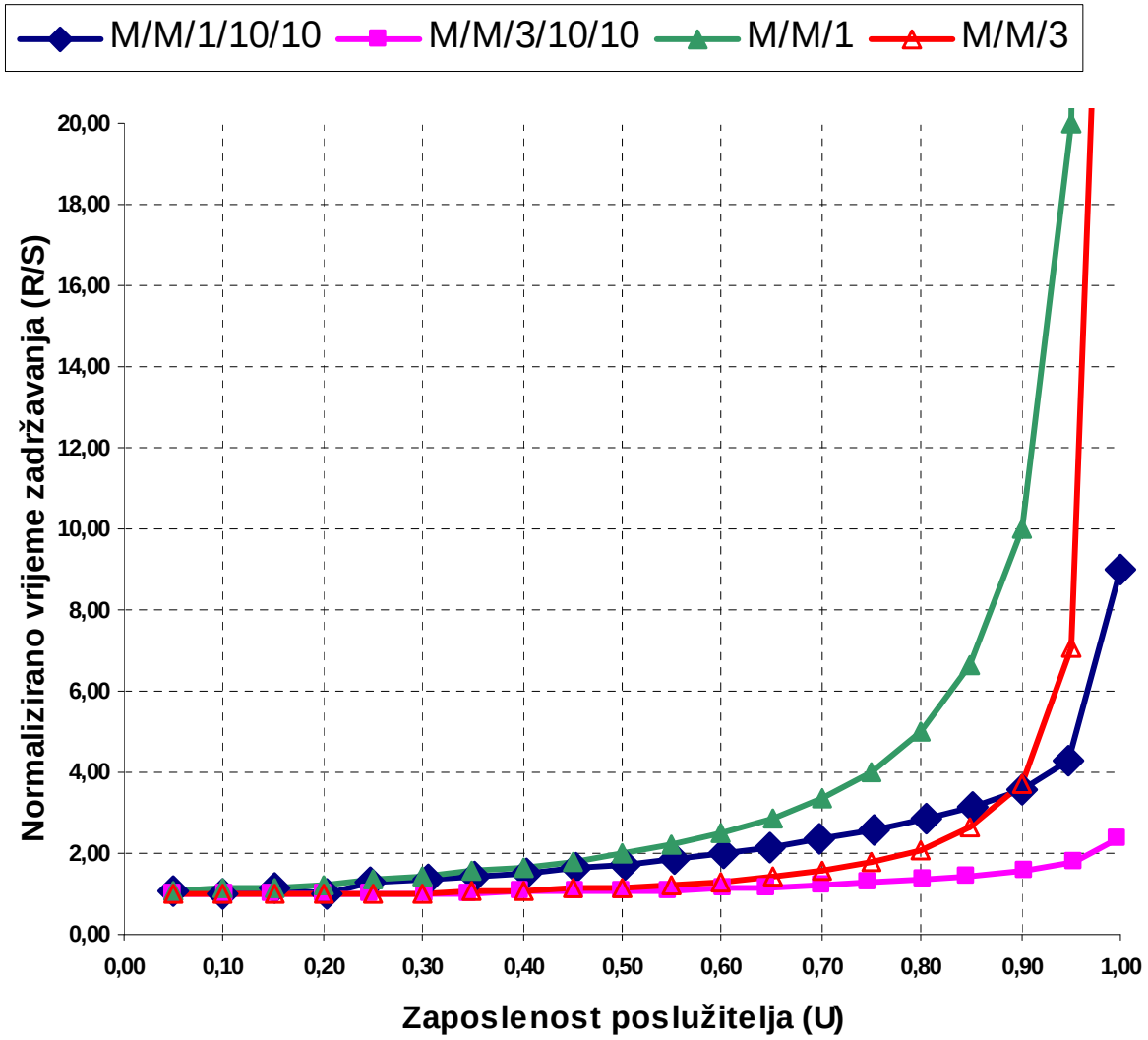
    Odgovor korištenjem *repair.c*:
    - Praksa je pokazala da se u multi-procesorskim sistemima postoji dodatni teret zbog sinkronizacije procesora. Uobičajeni faktor je 3 - 5% tj. u našem slučaju uzmimo da se S rate se povećava na ~ 0.66s
    - Program daje slijedeće rezultate:
      - U = 0.47; R = 0.8465s

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

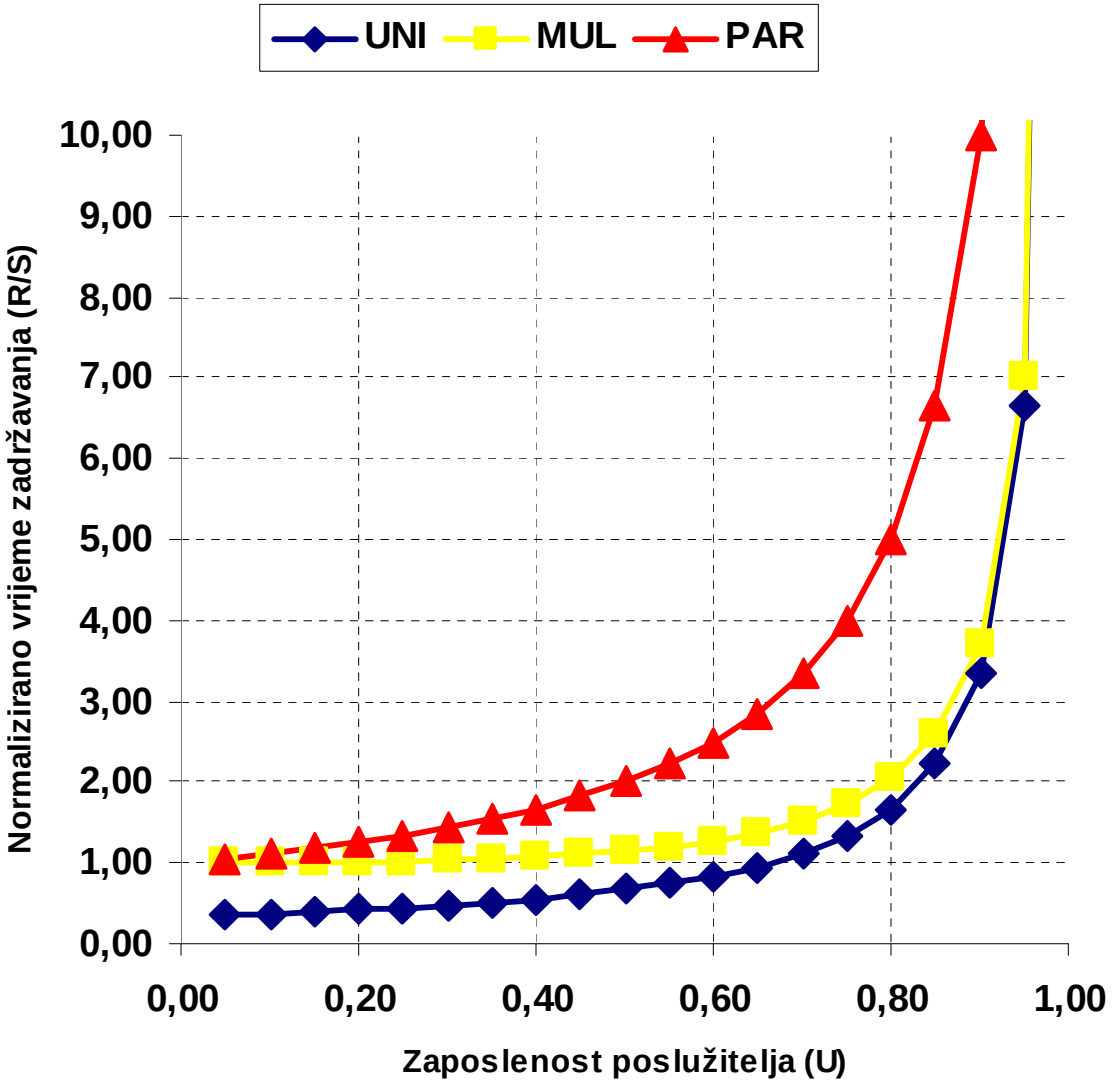
- **U slučaju kada ro $\rightarrow$ 1 i Z $\rightarrow$ 0**
  - R/S $\rightarrow$ m/N
  - R/S ne raste u beskonačno jer je broj zahtjeva u repu ograničen (t.j. zatvoreni sustav)

Usporedba otvorenih i zatvorenih sustava



Usporedba.xls

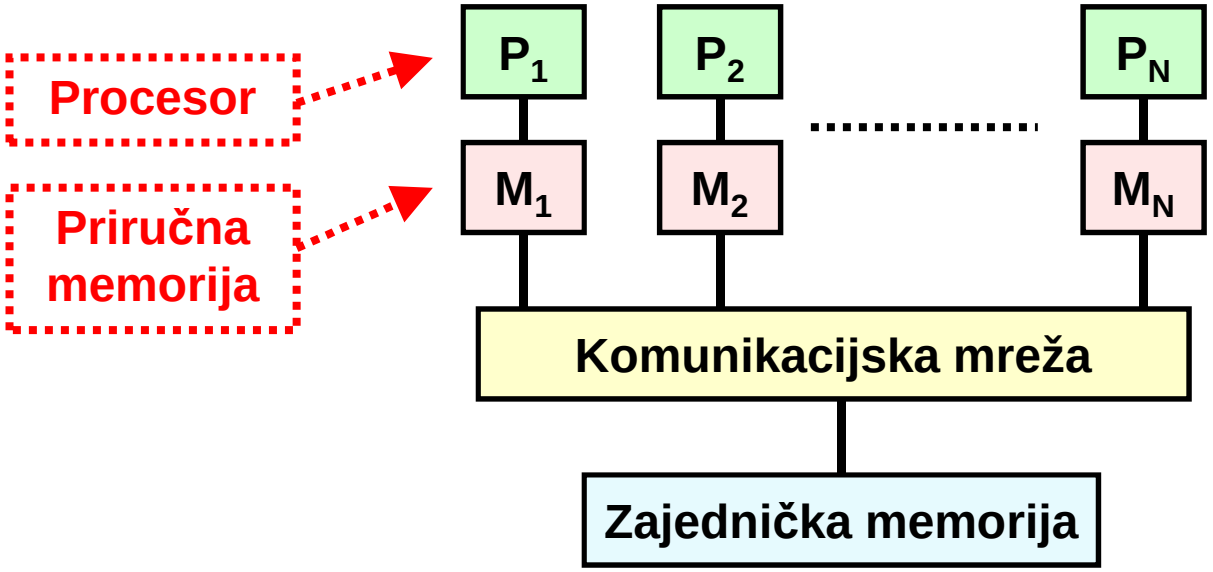
Koji je model bolji: Multi-računalo ili Multi-procesor ?



Uniprocesor ima
3x procesnu moć

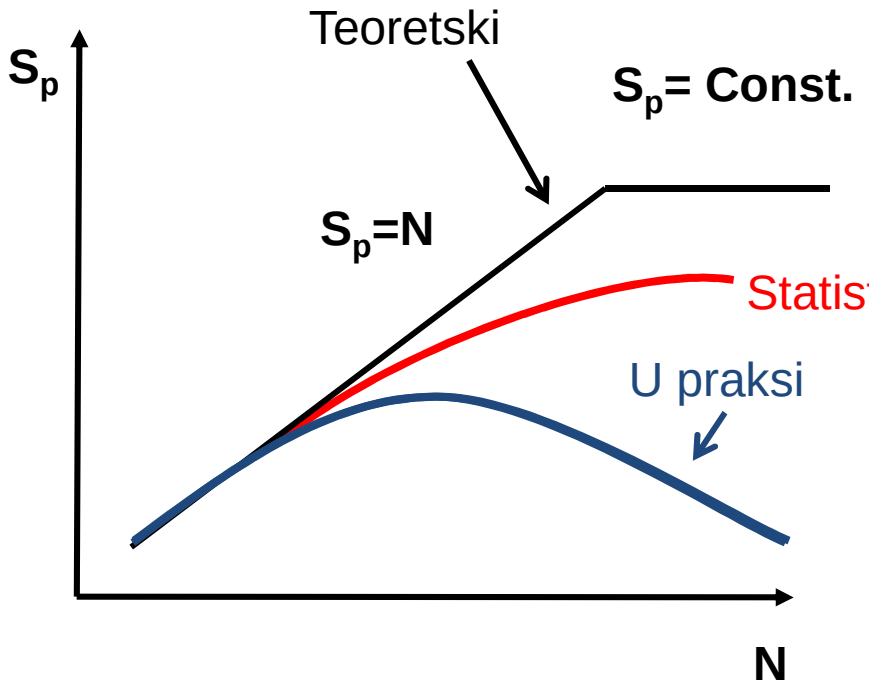
Sustavi.xls

Performanse paralelnih sustava



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

Ubrzanje (Speedup) je nelinerna funkcija od N

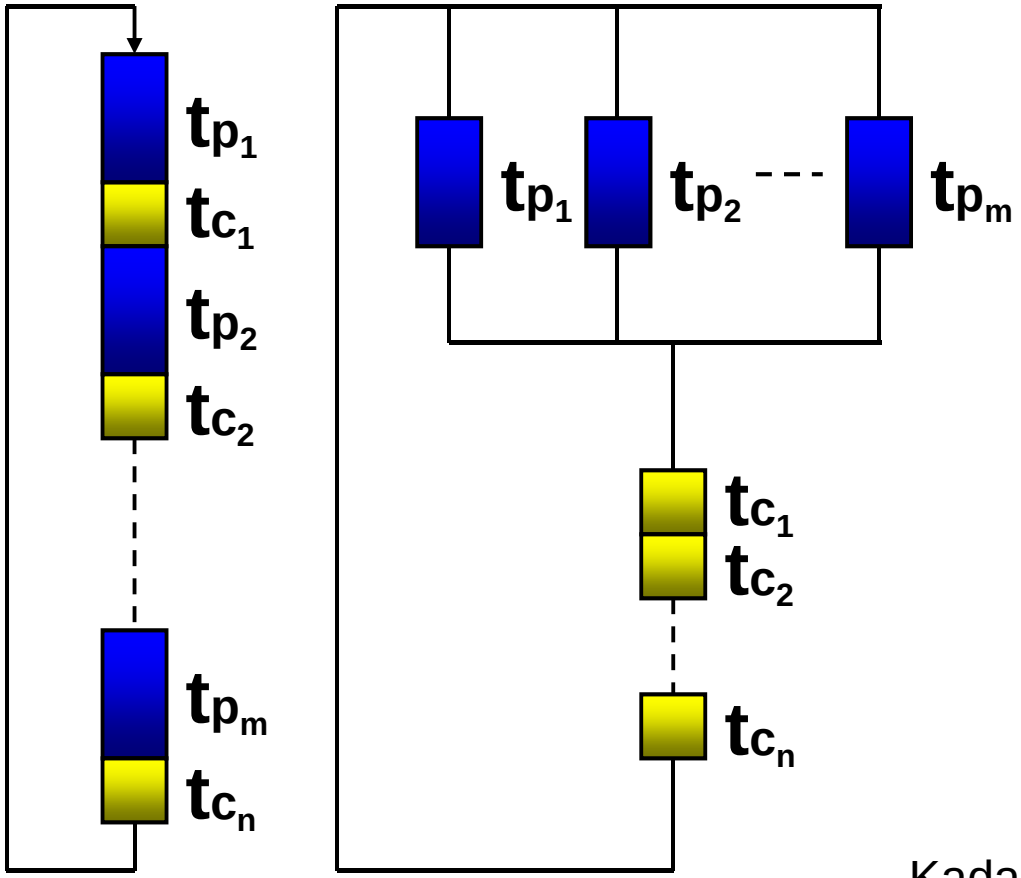


Sabirnica sustava (kao i izlazna jedinica) ne omogućava punu brzinu kod 100% iskorištenja sustava, već sustav prelazi u potpuno zasicenje

Bolje razumijevanje parametara

- **Uzrok degradacija performansi u paralelnim sustavima**
  - Ostvarivanje sekvencijalnog pristupa zajedničkim sredstvima u aplikaciji i sklopovima
  - Kašnjenja zbog održavanja koherencije i sinkronizacije
- **Modeliranje paralelnih aplikacija**
  - Deterministički modeli
  - Statistički modeli

Deterministički model



$$Tp = \sum_i tp_i$$

$$Tc = \sum_j tc_j$$

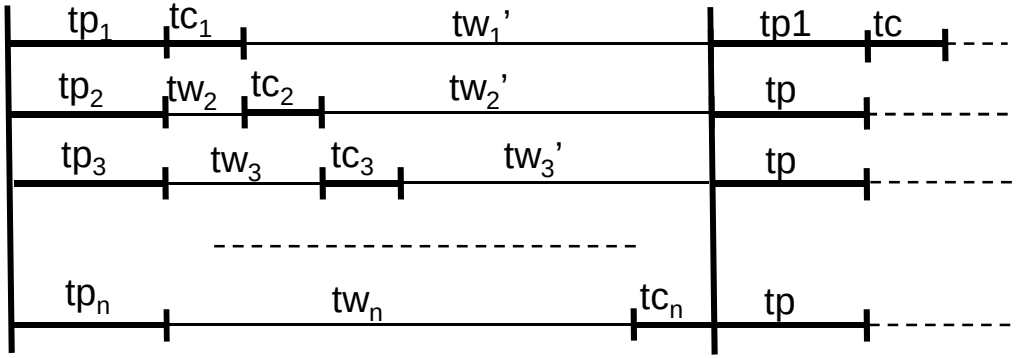
$$x = Tp / Tc$$

Amdahl's Law

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

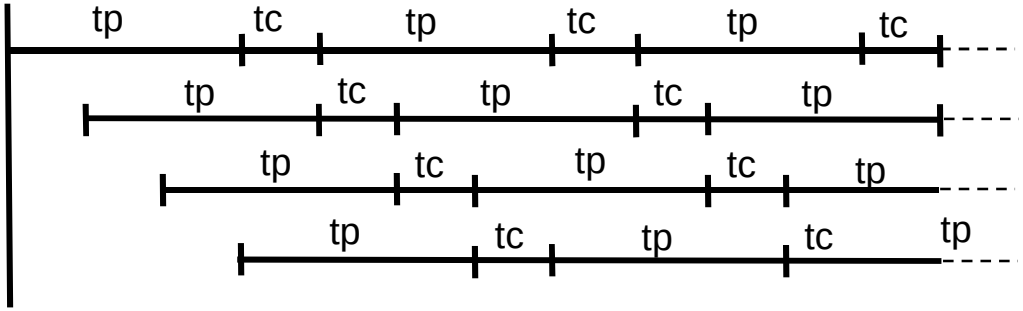
Kada je tp = 0, tj. paralelno izvođenje u 0 vremena kada N tezi beskonačno

Dva granična slučaja



Sinkroni (Najgori slučaj)

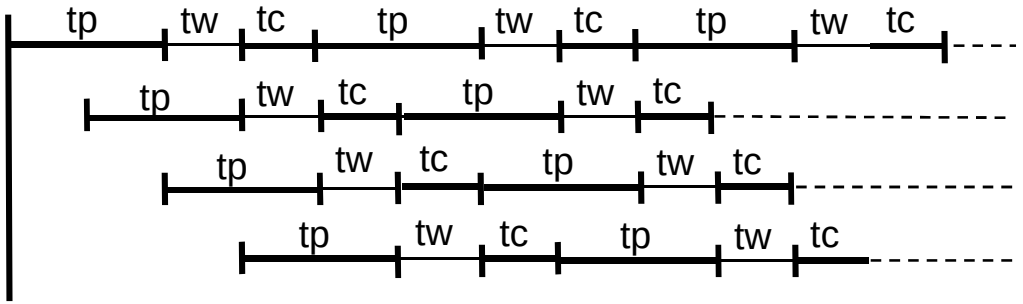
$tw = (N - 1)*tc$



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**

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

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

- **Asinkroni slučaj**

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

- **Sinkroni slučaj**

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

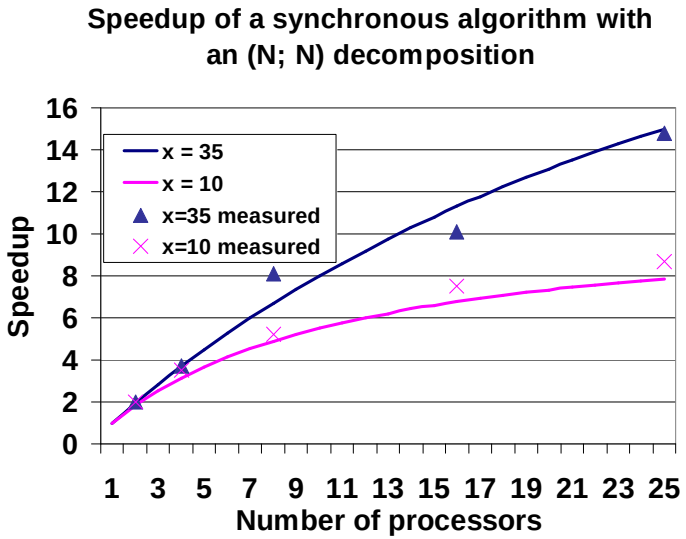
Rezultati za prikazane funkcije

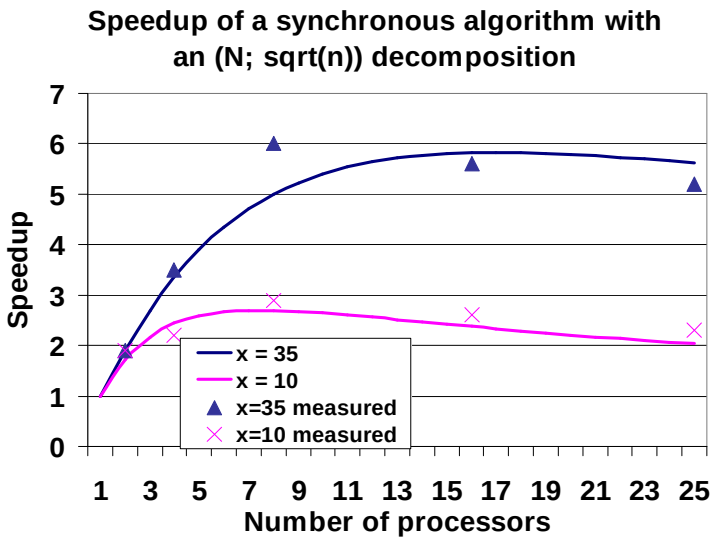
Dekompozicija		Sinkroni slučaj			Asinkroni slučaj		
tp	tc	SP	SP_{max}	N_{max}	SP	SP_{max}	N_{max}
N	N	$\frac{(1+x)N}{N+x}$	$1+x$	∞	$\min[N, 1+x]$	$1+x$	$1+x$
N	$\sqrt{N}$	$\frac{(1+x)N}{N^{3/2}+x}$	$\frac{2^{2/3}(1+x)}{3x^{1/3}}$	$2x^{2/3}$	$\min\left[\frac{(1+x)N}{\sqrt{N}+x}, \frac{(1+x)}{\sqrt{N}}\right]$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

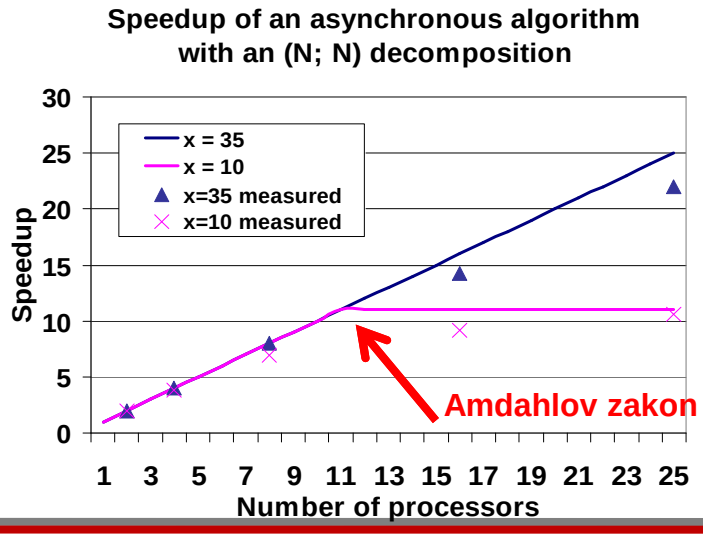
*) **Samo numeričko rješenje**

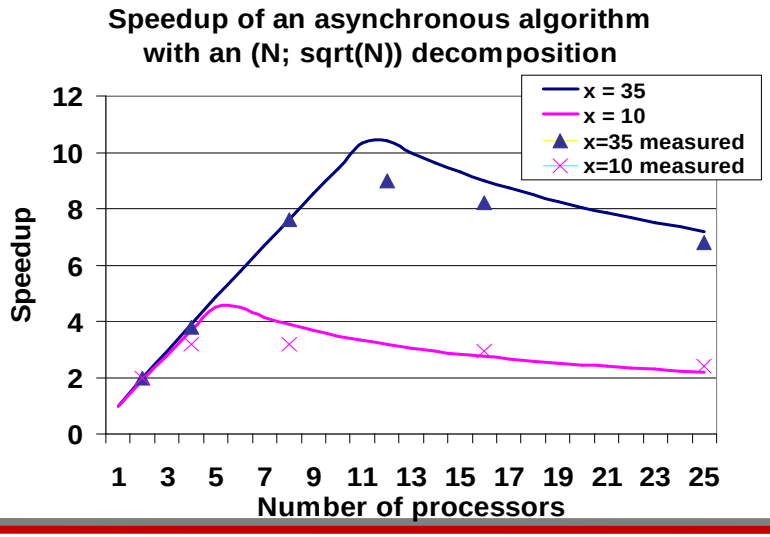
Preuzeto iz: **Performance prediction and calibration for a class of multiprocessors**
Vrsalovic, D.F.; Siewiorek, D.P.; Segall, Z.Z.; Gehringer, E.F.;
IEEE Transactions on Computers, Volume: 37 Issue: 11 , Nov. 1988. Page(s): 1353 -1365

Funkcije raspodjele opterećenja u praksi

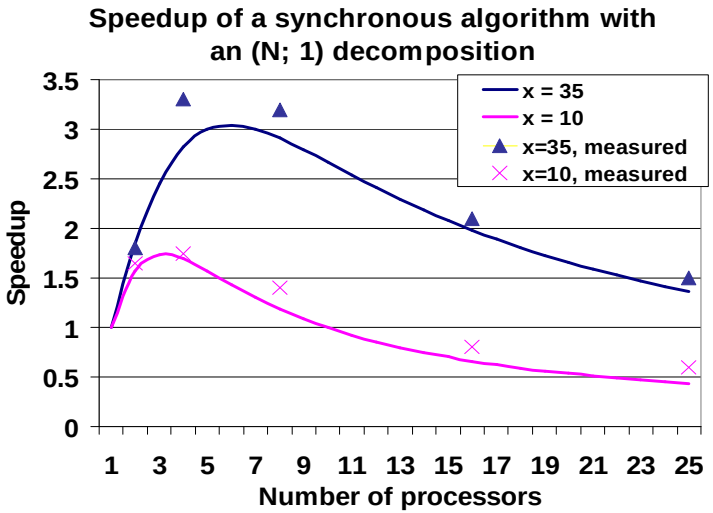


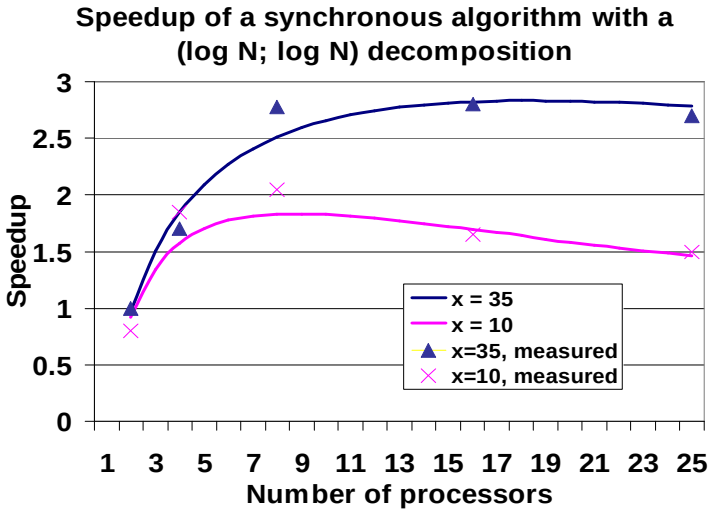


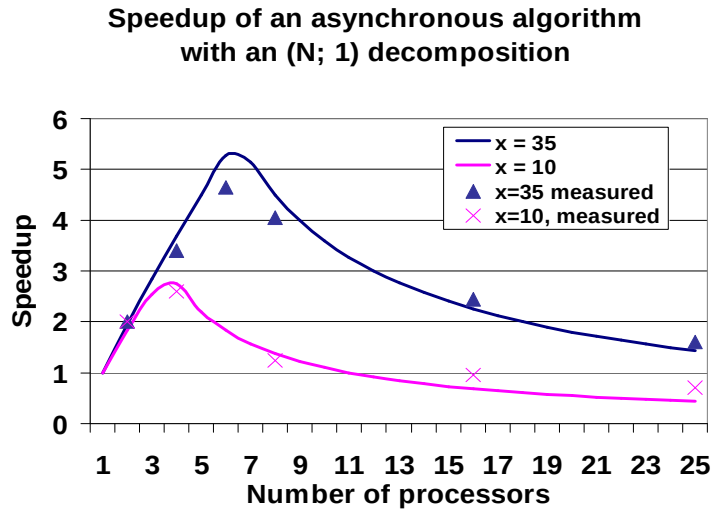


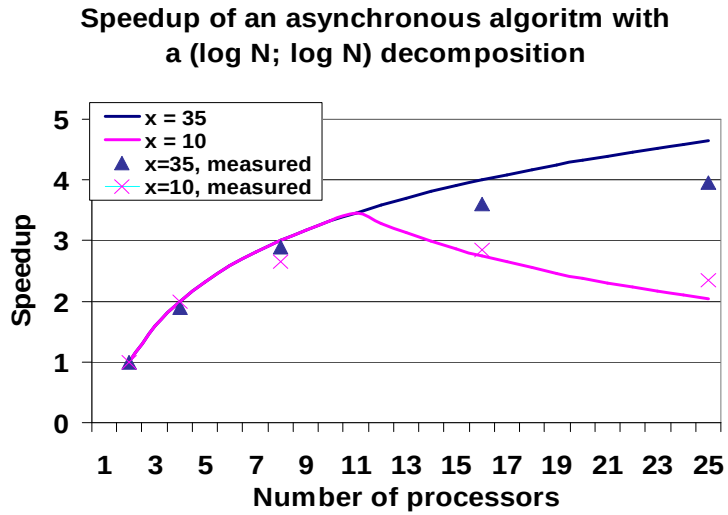


Funkcije raspodjele opterećenja u praksi



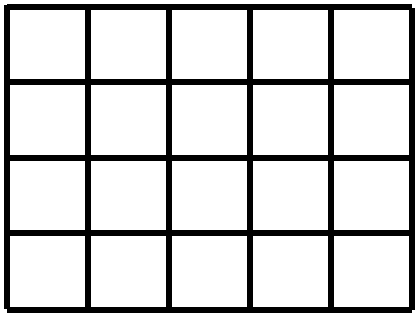


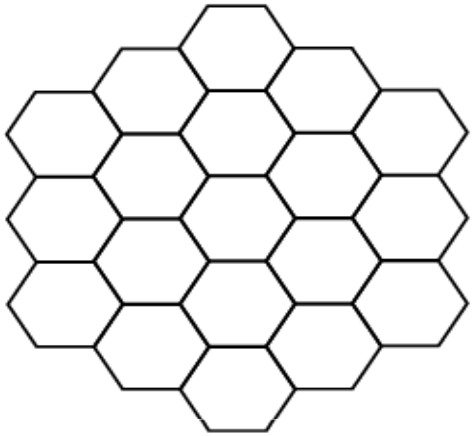




Razumijevanje modela pomaže programiranju

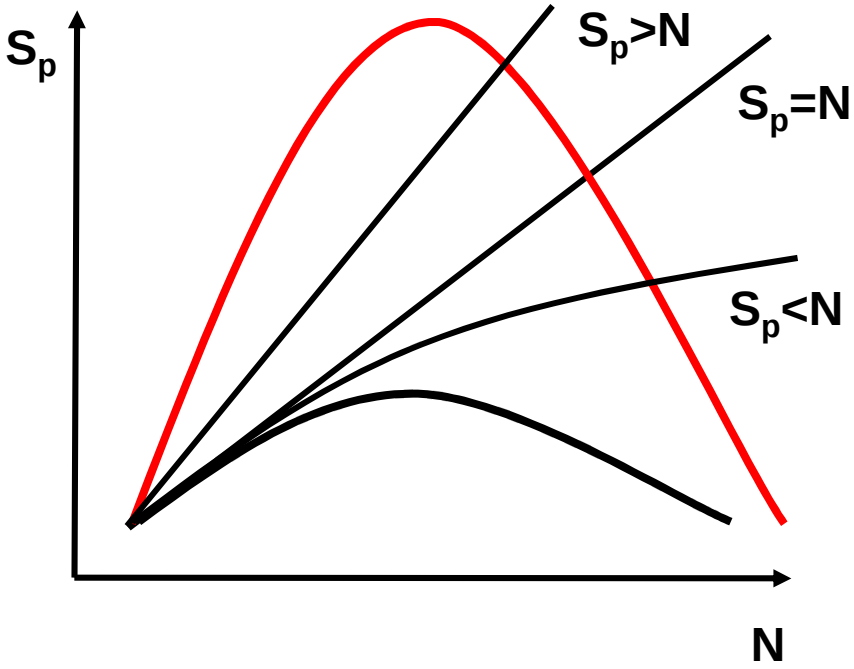
- **Primjer: Algoritmi najbližih susjeda**





- **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

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

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

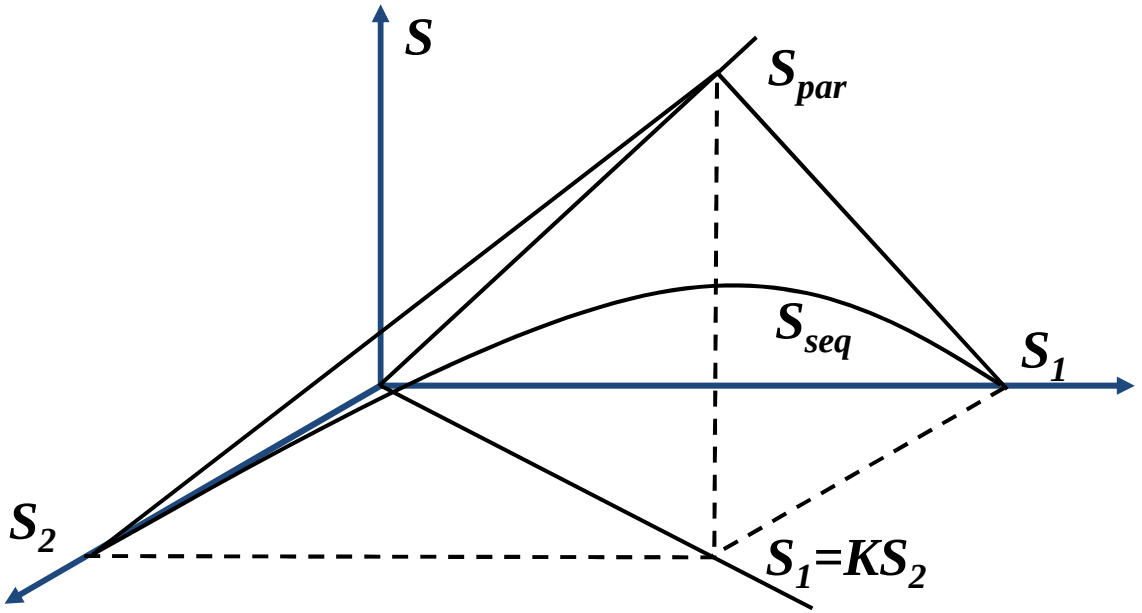
$$Sp = f(t_i)$$

Kombiniranje dviju paralelnih aplikacija

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

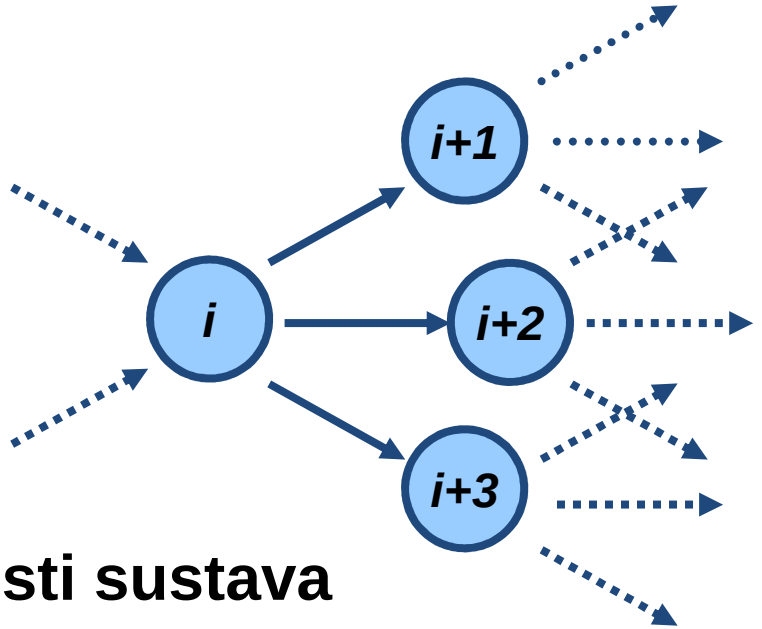
$$T_i = T_p + T_c, \quad t_j = t_p + t_c + t_w$$

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



Stohastički modeli zatvorenih sustava

- **Dijagram stanja sustava**
  - Opisuje izmjenu stanja sustava tijekom rada



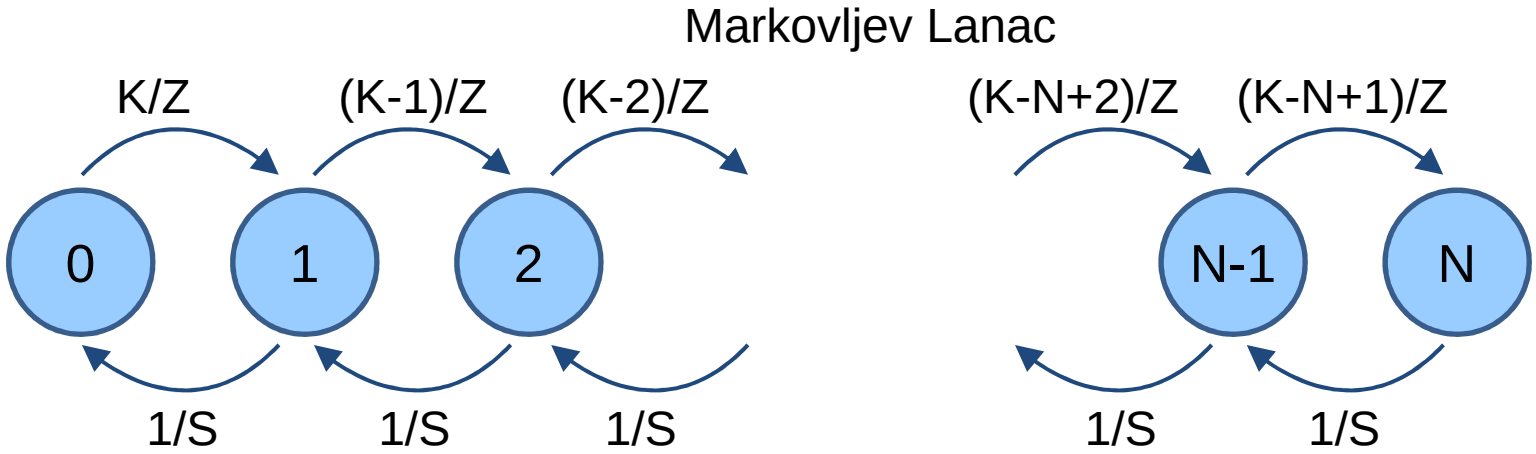
- **Matrica prijelaznih vjerojatnosti sustava**
  - Element matrice u retku *i* te stupcu j određuje vjerojatnosti da će sustav iz stanja *i* preći u stanje *j*

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

Stohastički modeli zatvorenih sustava

- **Faktor iskorištenja sustava (ro)**
  - Sustav se koristi ako se nalazi u bilo kojem stanju osim početnom
  - Vjerojatnost da se sustav nalazi u stanju koje nije početno: ro = 1 - p_0
- **Vrijeme čekanja u sustavu (W)**
  - ro = U = L*S; L = ro/S
  - L = K/(Z + W + S) = ro/S
  - W = (K*S/ro) – Z – S
- **Vrijeme zadržavanja u sustavu (R)**
  - R = S + W = (K*S/ro) – Z; Q = L*R

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**

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

$$p_0 = \frac{1}{\sum_{k=0}^{K} \frac{\frac{k!}{(k-N)!}}{(S/Z)^n}}$$

Stohastički model (M/M/1/K/K)

- **Faktor iskorištenja sustava (ro)**
  - ro = 1 – p0 = 1 – B(K, Z/S)

    (B(K,Z/S) je Erlangova formula)

- **Vrijeme čekanja u sustavu (W)**
  - ro = U = LS; L = ro/S
  - L = K/(Z + W + S) = ro/S
  - W = (KS/ro) – Z – S

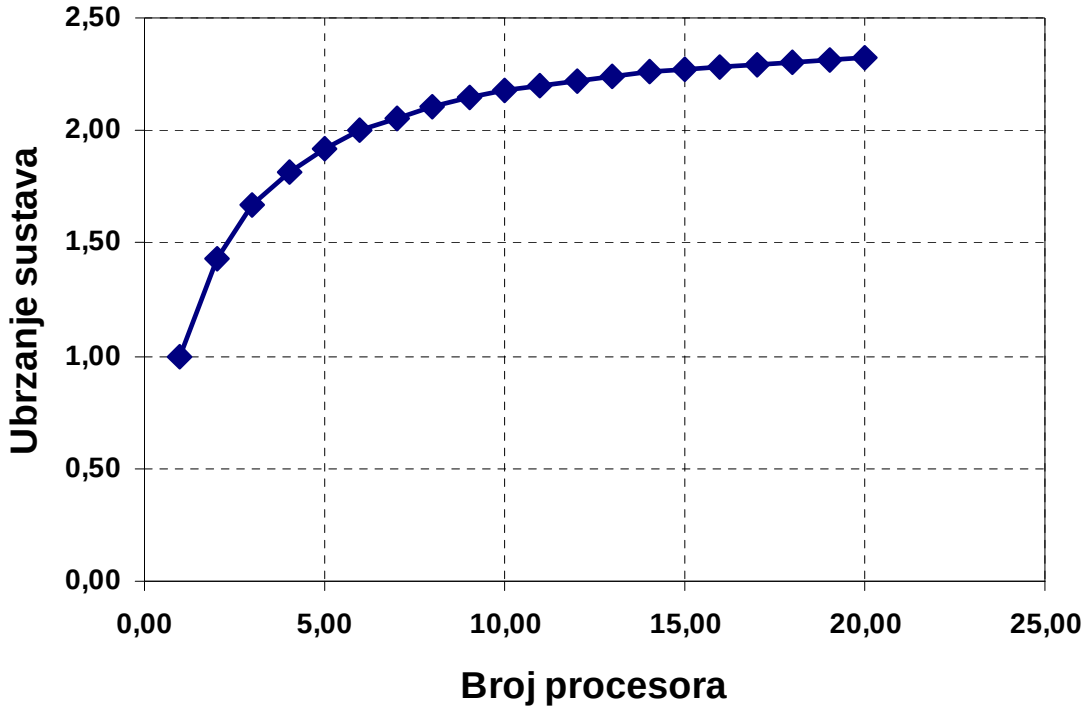
- **Vrijeme zadržavanja u sustavu (R)**
  - R = S + W = (KS/ro) - Z;
  - Q = L R

Efikasnost i ubrzanje

- **Efikasnost procesora (E)**
  - E = (Z + S)/(Z + S + W)
  - E = (Z + S)/(Z + R) = (Z + S)/(Z + K*S/ro – Z)
  - E = (Z + S)*ro/K*S
- **Ubrzanje (S_P)**
  - Ubrzanje = K*E

Podrazumijeva idealnu dekompoziciju tereta na K procesora

Ubrzanje.xls

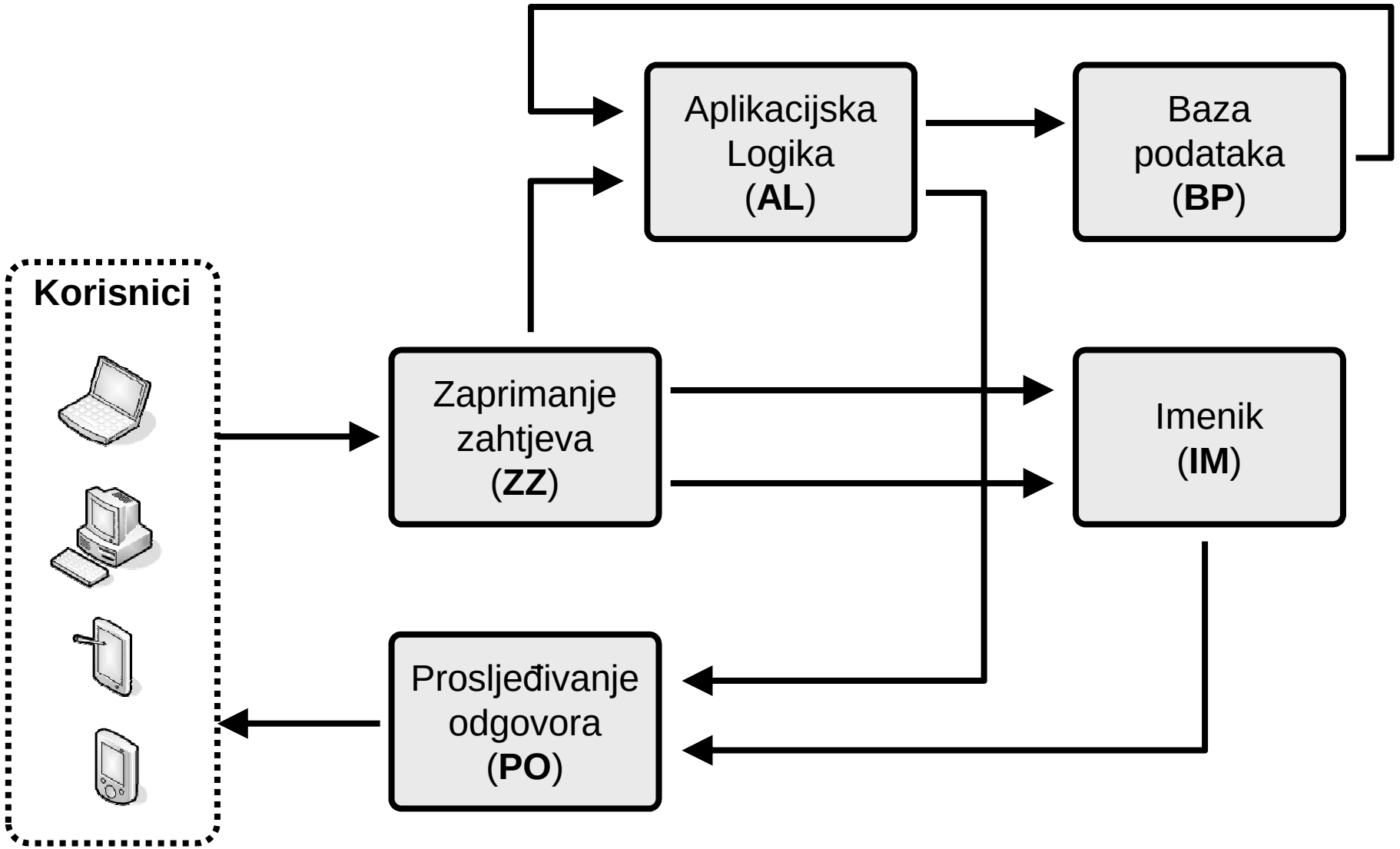


Primjer analize performansi web aplikacije

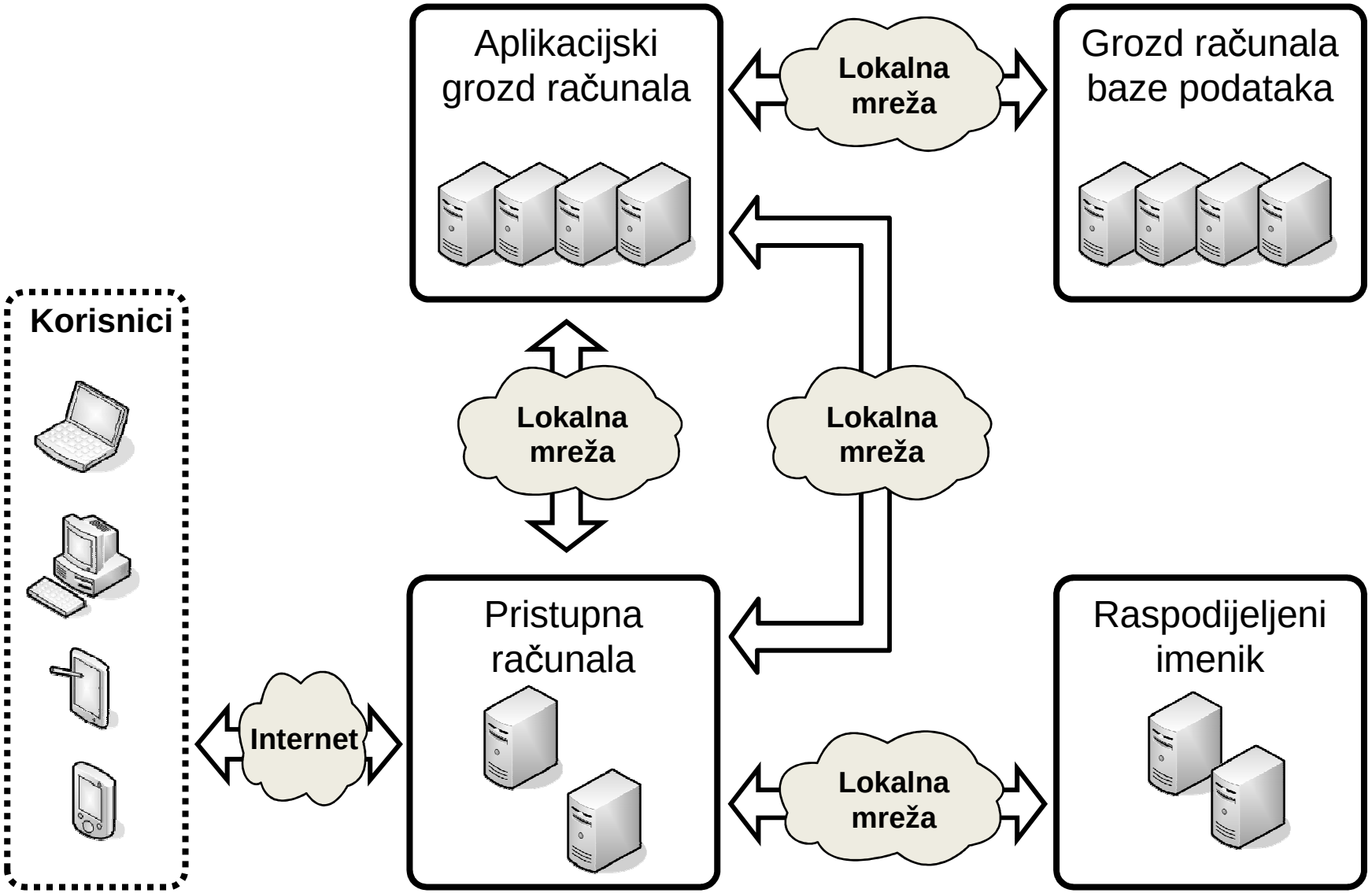
Primjer analize raspodijeljene aplikacije

- Logička arhitektura raspodijeljene aplikacije
- Fizička arhitektura raspodijeljene aplikacije
- Model raspodijeljene aplikacije
- Vrednovanje značajki performansi aplikacije

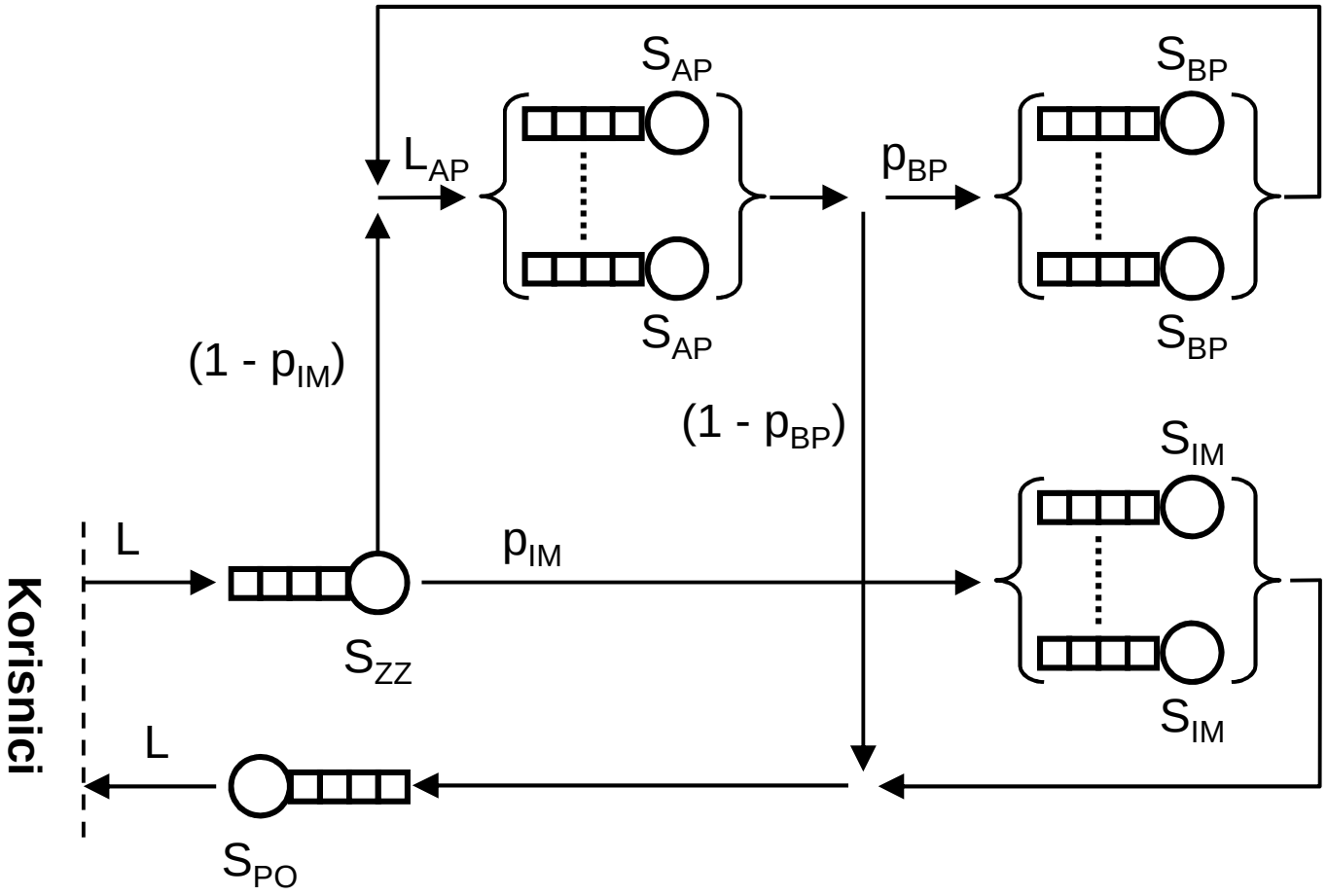
Logička arhitektura aplikacije



Fizička arhitektura aplikacije



Model aplikacije



Primjer analize raspodijeljene aplikacije

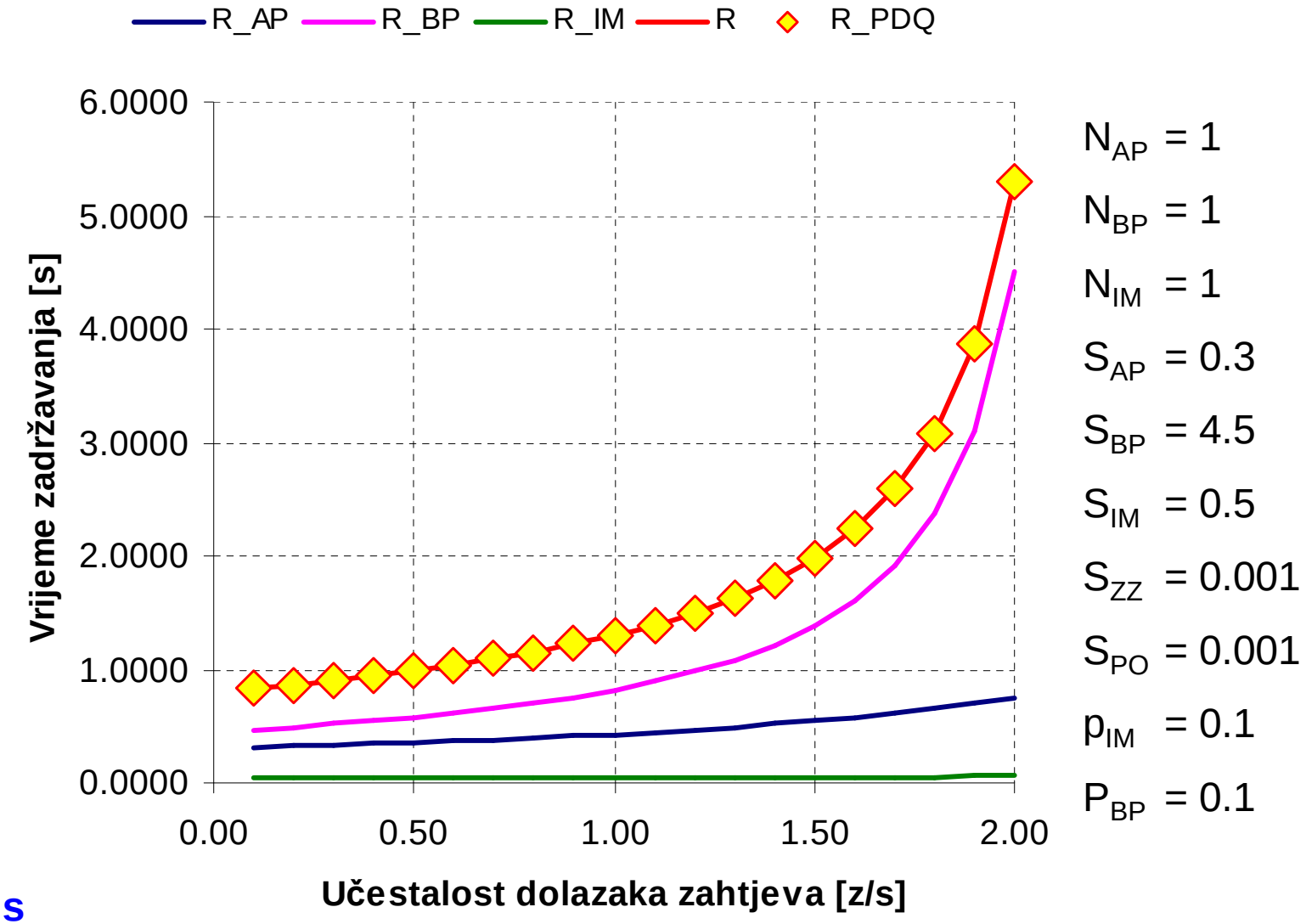
- Svaki od podsustava aplikacije na jednom računalu
- Učestalost dolazaka zahtjeva na podsustave
  - $L_{IM} = p_{IM}L$, $v_{IM} = p_{IM}$
  - $L_{AP} = p_{BP}L_{AP} + (1 - p_{IM})L \Rightarrow$
    $L_{AP} = [(1 - p_{IM})/(1 - p_{BP})]L$
    $v_{AP} = [(1 - p_{IM})/(1 - p_{BP})]$
  - $L_{BP} = p_{BP}\, L_{AP} \Rightarrow$
    $L_{BP} = p_{BP}\, [(1 - p_{IM})/(1 - p_{BP})]\, L$
    $v_{BP} = p_{BP}\, [(1 - p_{IM})/(1 - p_{BP})]$
  - $L_{ZZ} = L$, $v_{ZZ} = 1$
  - $L_{PO} = L$, $v_{PO} = 1$

Primjer analize raspodijeljene aplikacije

- **Skalirana vremena posluživanja**
  - $D_{IM} = v_{IM}\, S_{IM} = p_{IM}\, S_{IM}$
  - $D_{AP} = v_{AP}\, S_{AP} = [(1 - p_{IM})/(1 - p_{BP})]\, S_{AP}$
  - $D_{BP} = v_{BP}\, S_{BP} = p_{BP}\, [(1 - p_{IM})/(1 - p_{BP})]\, S_{BP}$
  - $D_{ZZ} = v_{ZZ}\, S_{ZZ} = 1\, S_{BP}$
  - $D_{PO} = v_{PO}\, S_{PO} = 1\, S_{PO}$

- **Vrijeme zadržavanja zahtjeva**
  - $R = D_{IM}/(1 - L\, D_{IM}) + D_{AP}/(1 - L\, D_{AP}) + D_{BP}/(1 - L\, D_{BP}) + D_{ZZ}/(1 - L\, D_{ZZ}) + D_{PO}/(1 - L\, D_{PO})$

Vrijeme zadržavanja zahtjeva



$N_{AP} = 1$

$N_{BP} = 1$

$N_{IM} = 1$

$S_{AP} = 0.3$

$S_{BP} = 4.5$

$S_{IM} = 0.5$

$S_{ZZ} = 0.001$

$S_{PO} = 0.001$

$p_{IM} = 0.1$

$P_{BP} = 0.1$

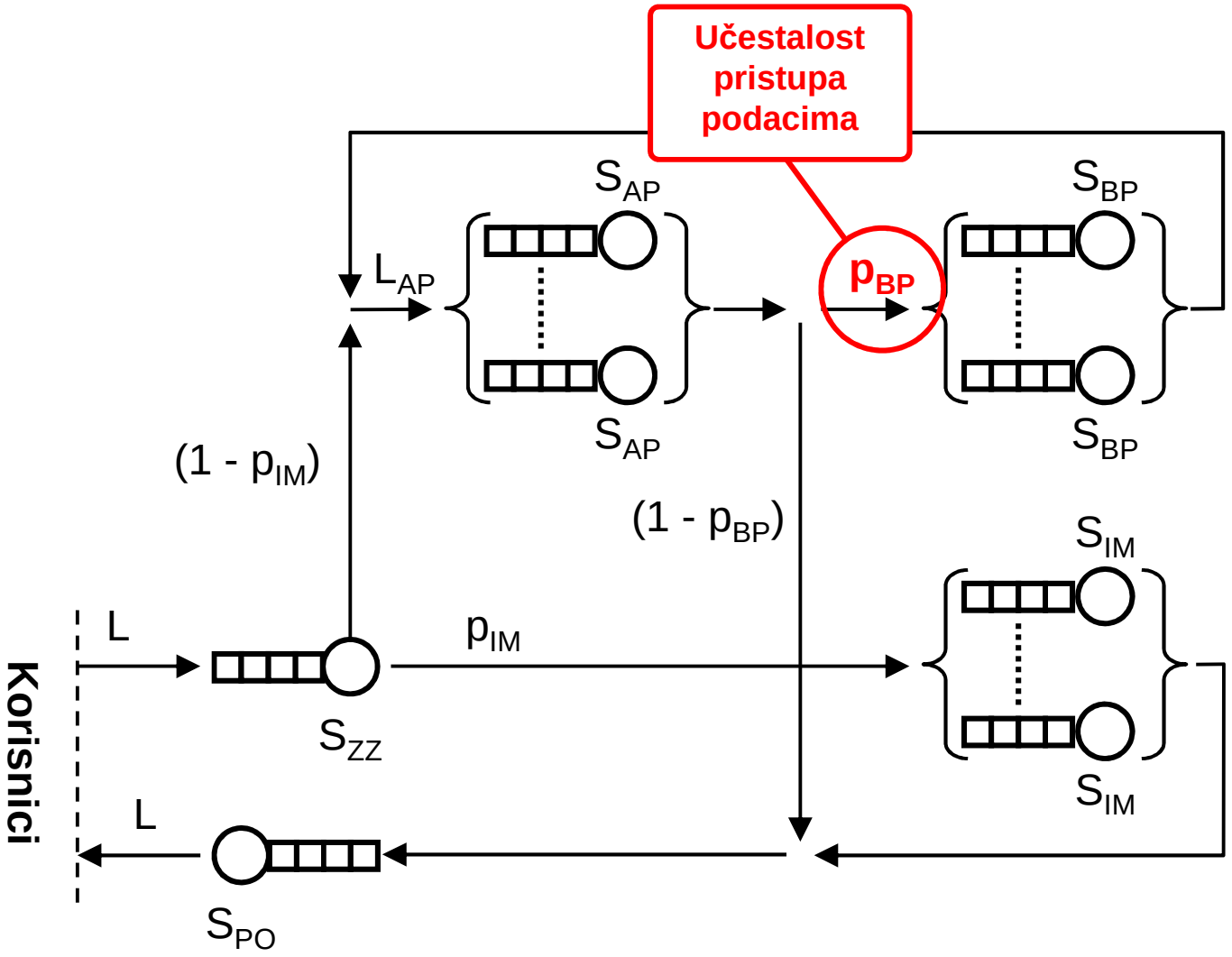
ap-Zad.c

Ap_Zad.xls

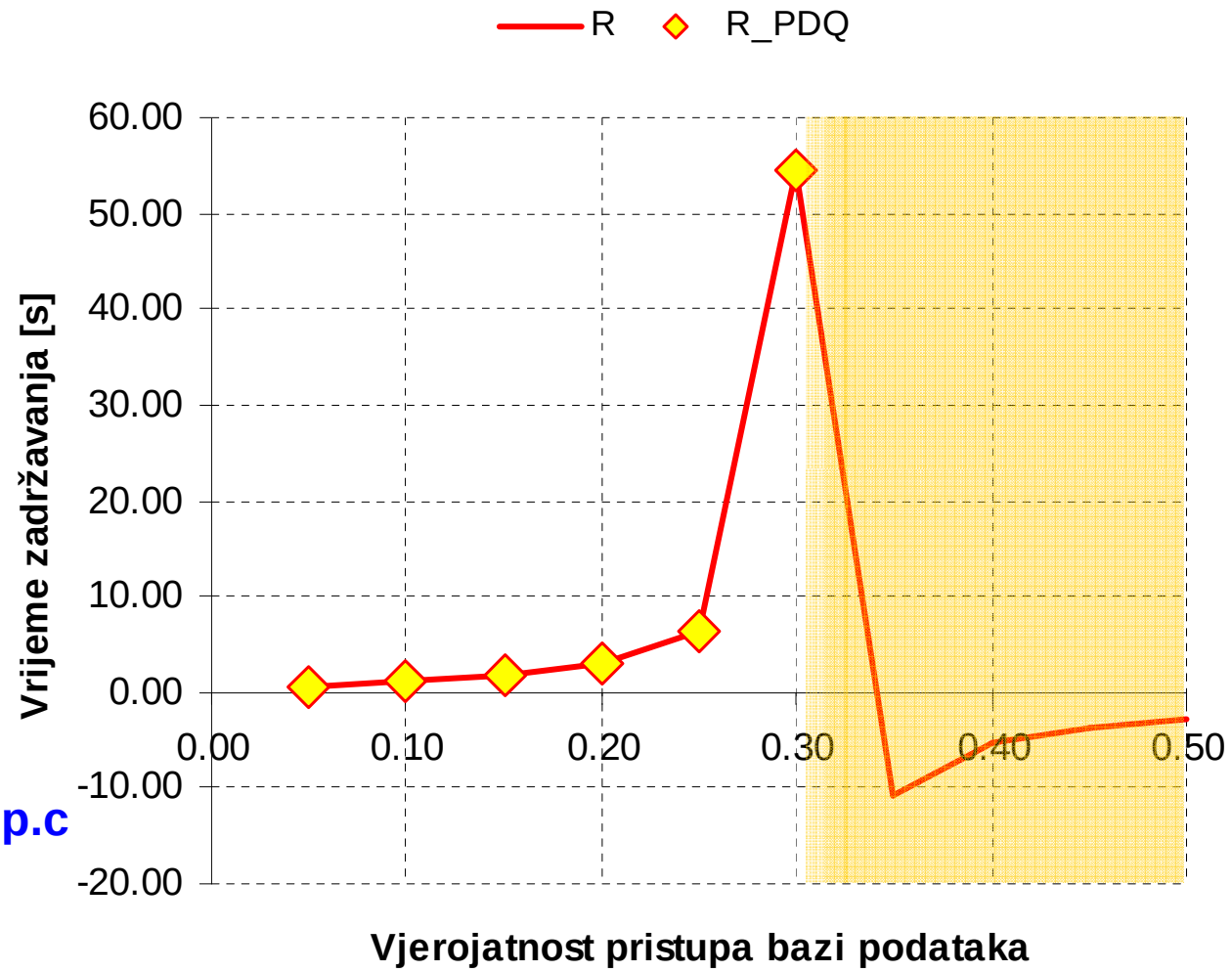
Vrednovanje značajki performansi

- Učestalosti pristupa podacima
- Veličina grozda baze podataka
- Promjena organizacije podataka
- Promjena stupnja sigurnosti

Učestalost pristupa podacima



Vrijeme zadržavanja zahtjeva

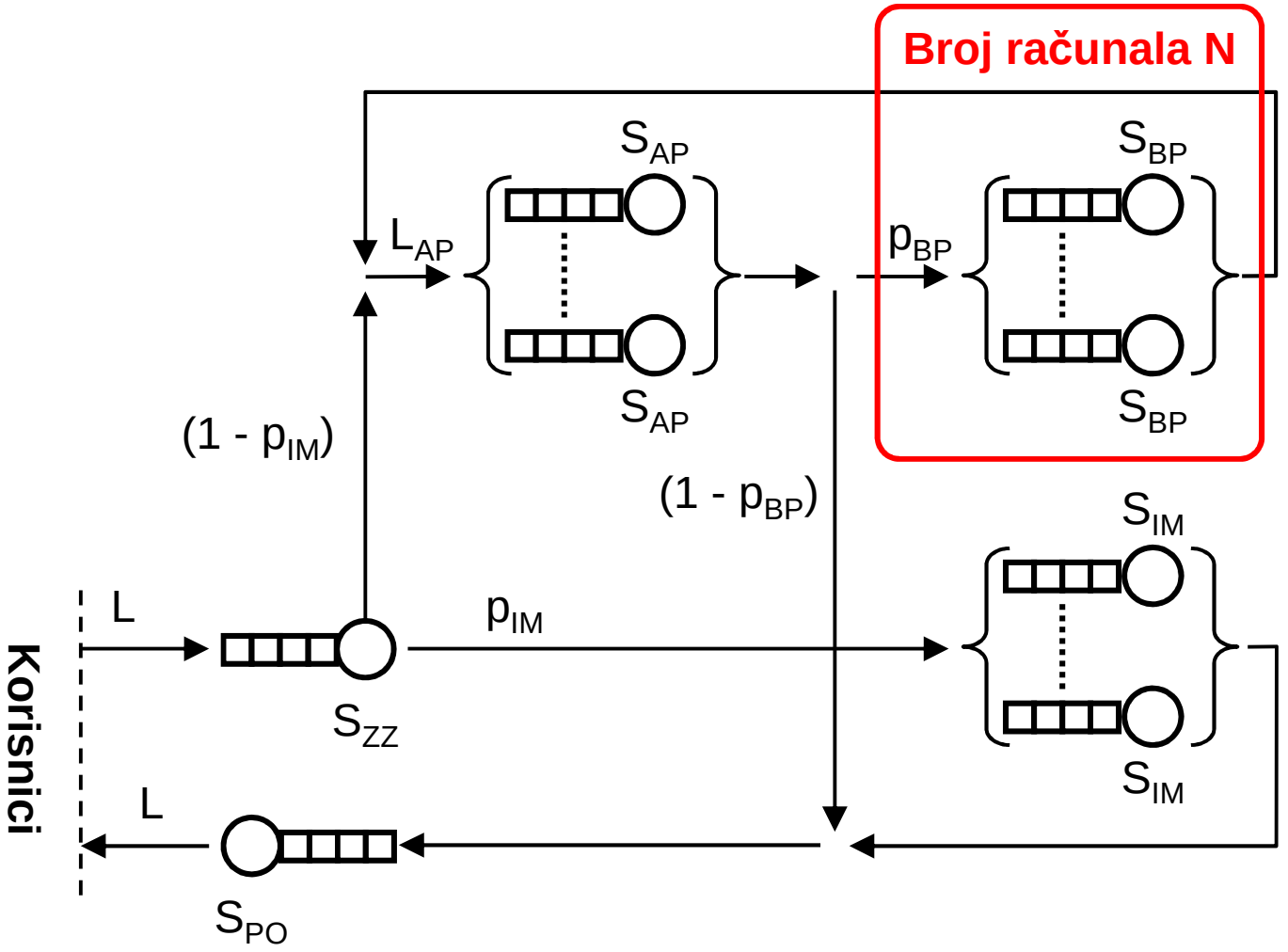


N_{AP} = 1
N_{BP} = 1
N_{IM} = 1
S_{AP} = 0.3
S_{BP} = 4.5
S_{IM} = 0.5
S_{ZZ} = 0.001
S_{PO} = 0.001
p_{IM} = 0.0
L = 0.5

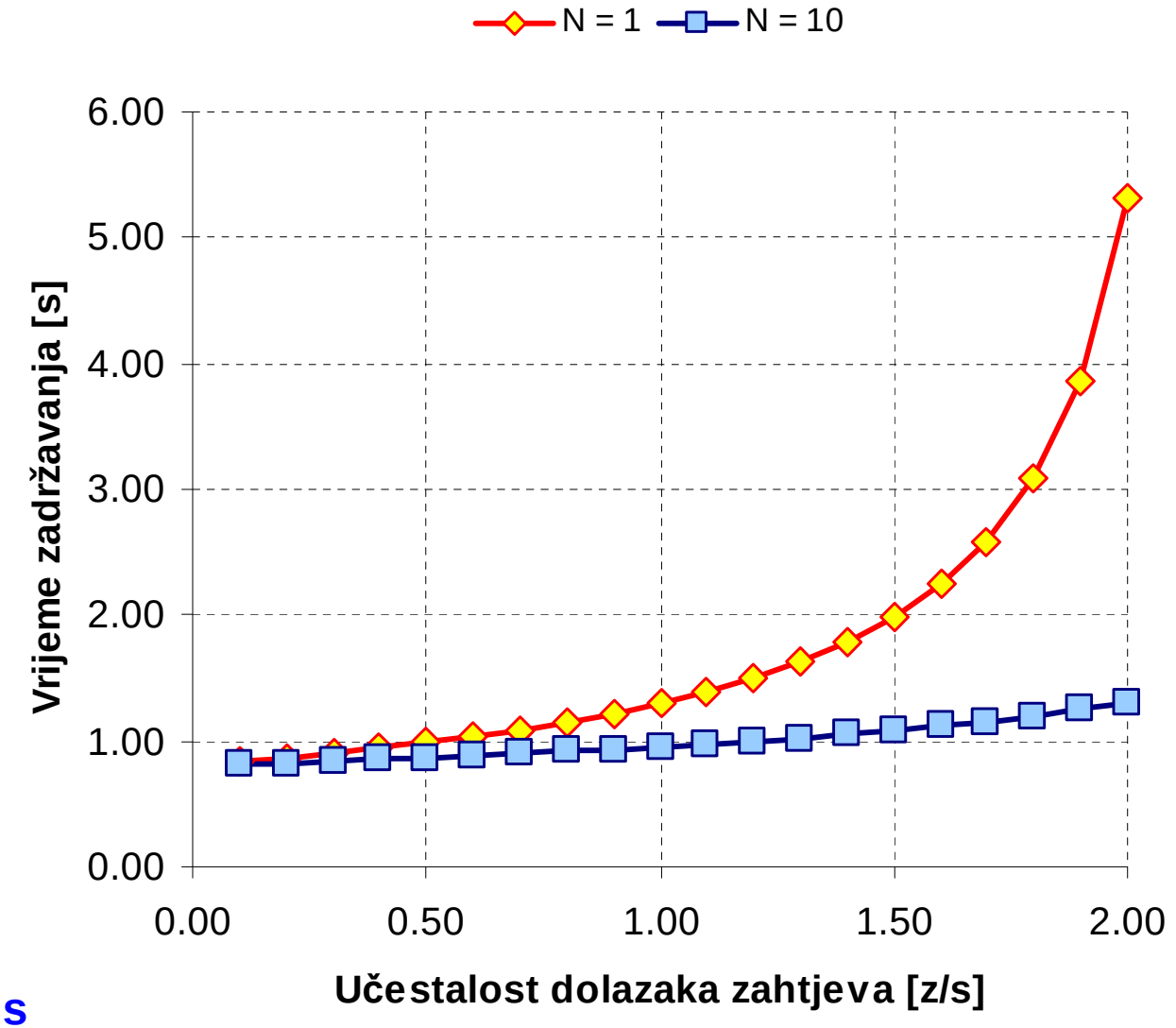
ap-Pristup.c

Ap_Prist.xls

Veličina grozda baze podataka



Vrijeme zadržavanja zahtjeva



S_{AP} = 0.3

S_{BP} = 4.5

S_{IM} = 0.5

S_{ZZ} = 0.001

S_{PO} = 0.001

p_{IM} = 0.1

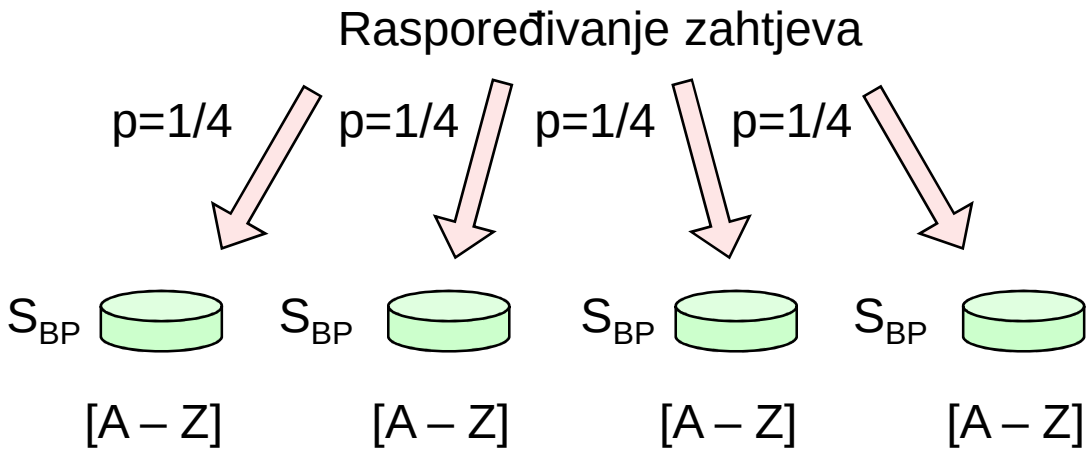
P_{BP} = 0.1

ap-Zad.c

Ap_Rac.xls

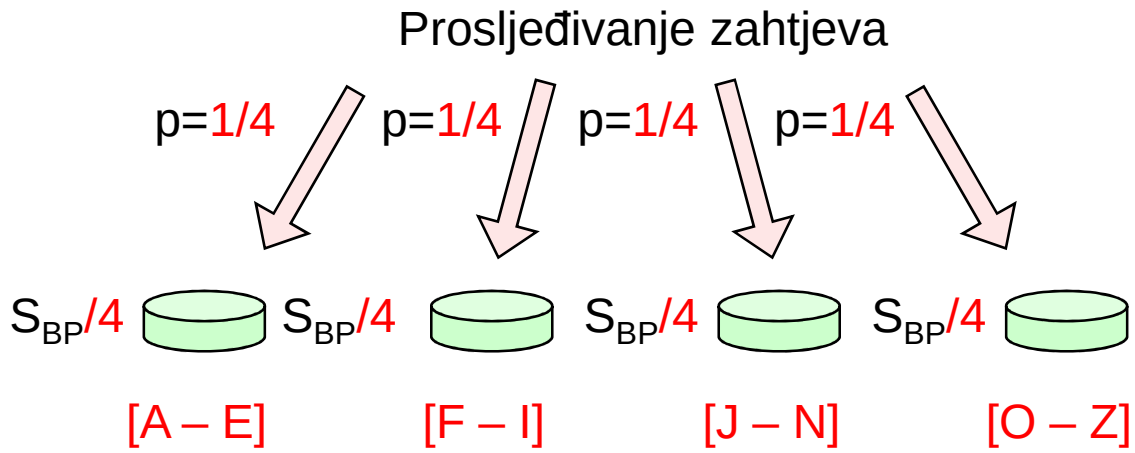
Promjena organizacije podataka

- **Replikacija podataka**
  - Skup računala od kojih svako u spremniku sadrži kopiju cijele baze podataka
  - Zahtjevi se raspoređuju na računala s ciljem raspoređivanja opterećenja



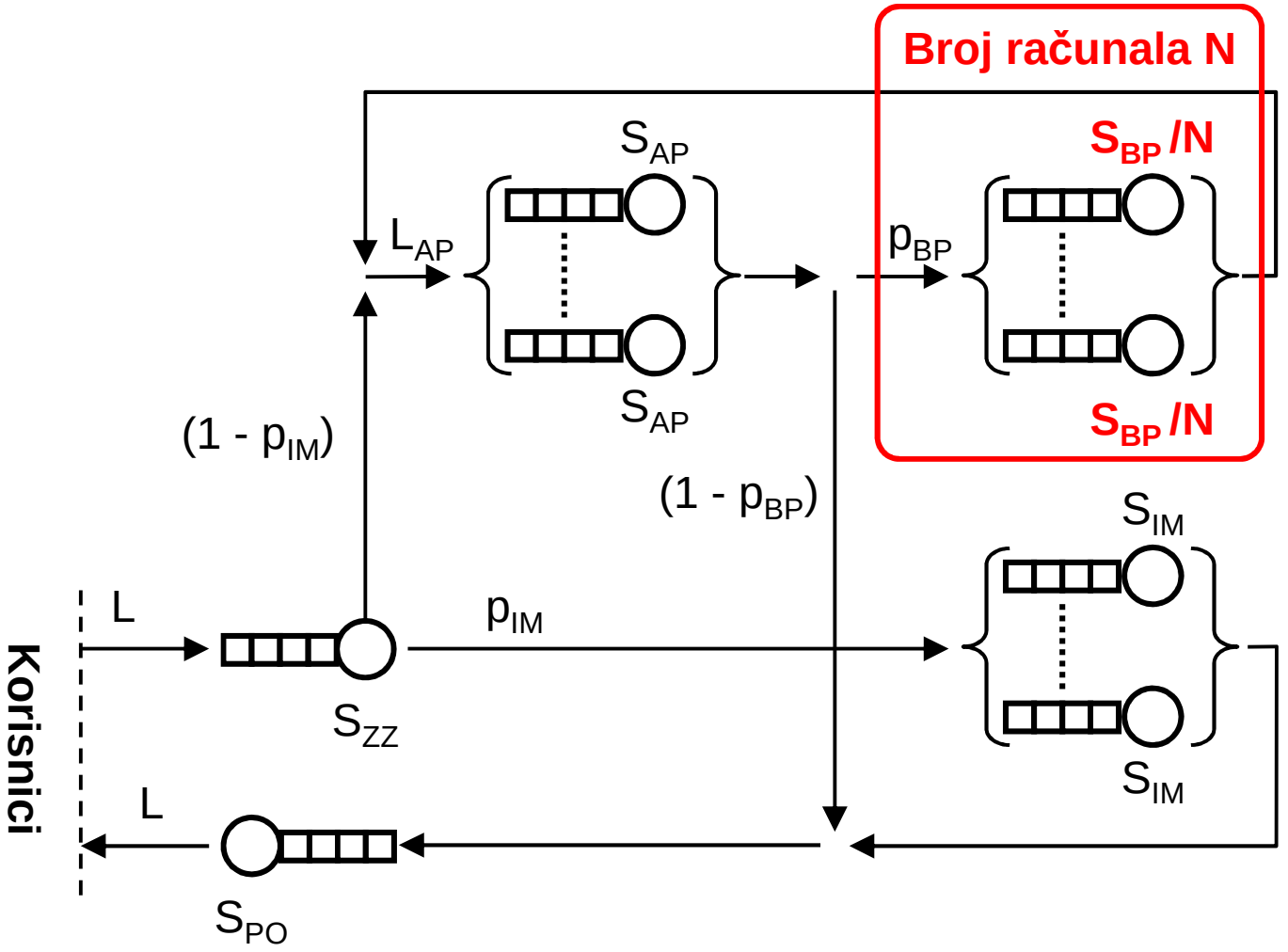
Promjena organizacije podataka

- **Segmentacija podataka**
  - Skup računala od kojih svako u spremniku sadrži dio cijele baze podataka
  - Zahtjevi se prosljeđuju prema računalu s traženim zapisima

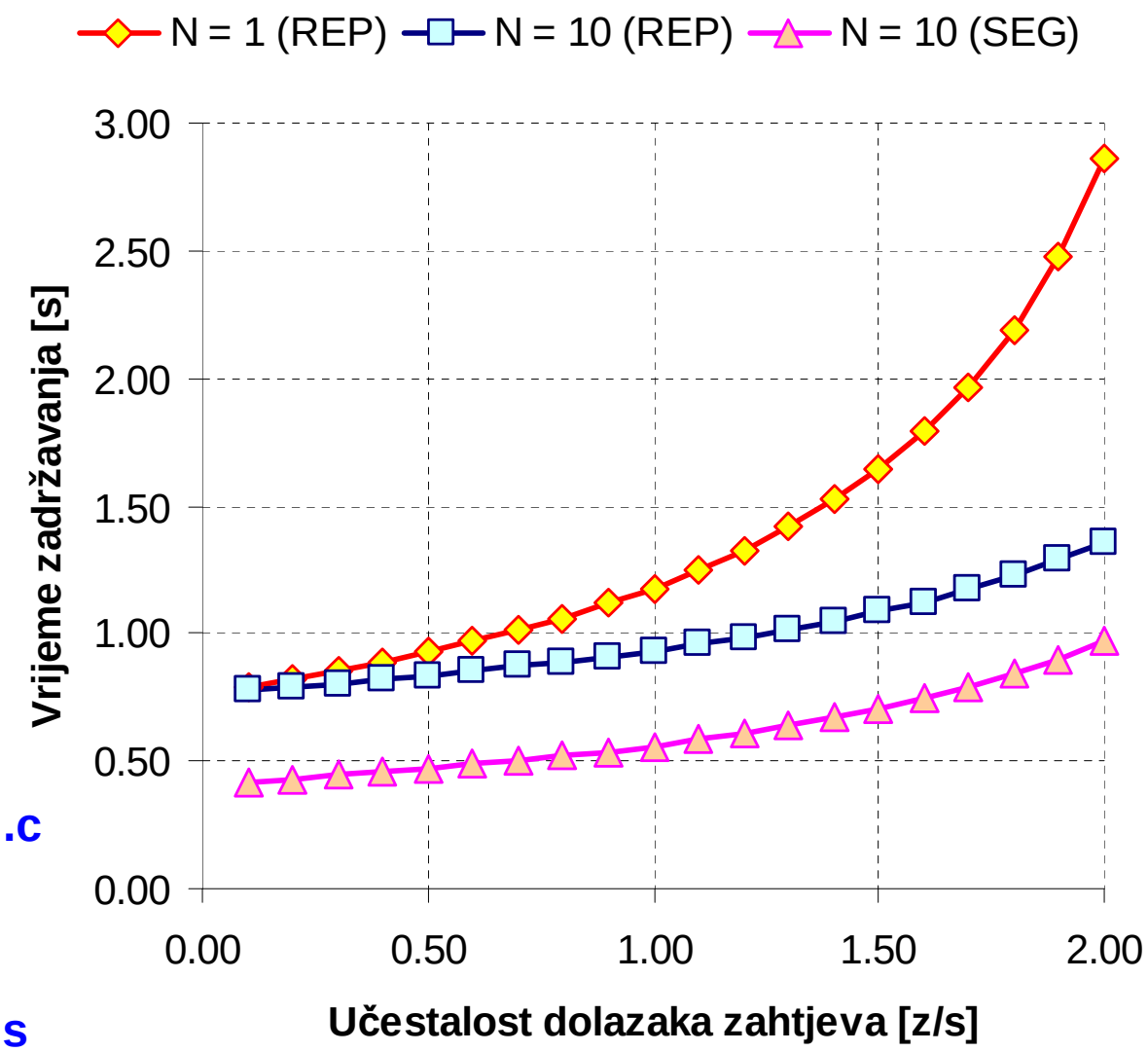


  - Pretpostavke
    - Uniformna raspodjela zahtjeva na zapise
    - Linearna složenost obrade zahtjeva o količini zapisa

Promjena organizacije podataka



Vrijeme zadržavanja zahtjeva



N_{AP} = 1
N_{BP} = 1
N_{IM} = 1
S_{AP} = 0.3
S_{BP} = 2.5
S_{IM} = 0.5
S_{ZZ} = 0.001
S_{PO} = 0.001
p_{IM} = 0.1
p_{BP} = 0.15

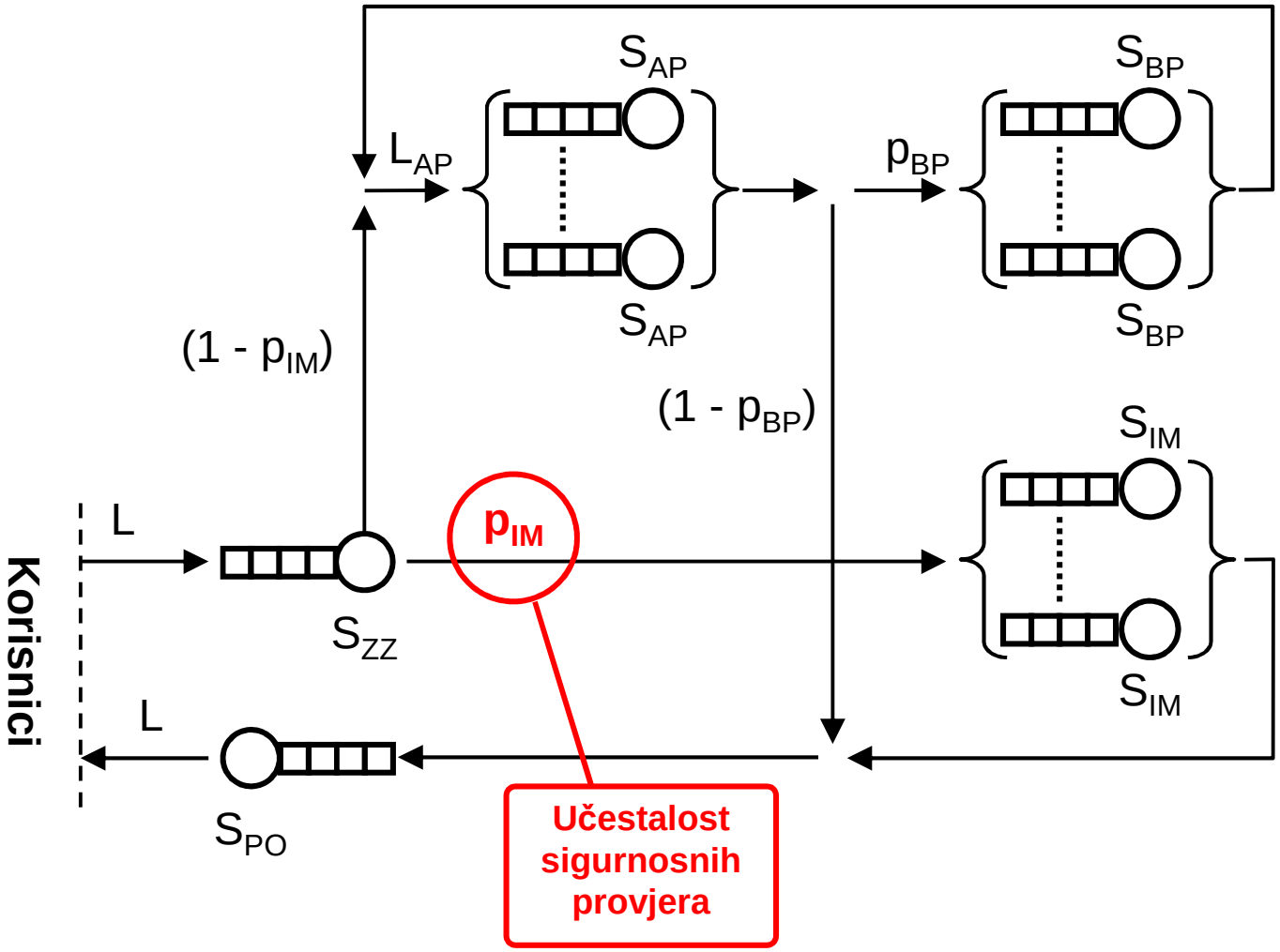
ap-PodOrg.c

Ap_Pod.xls

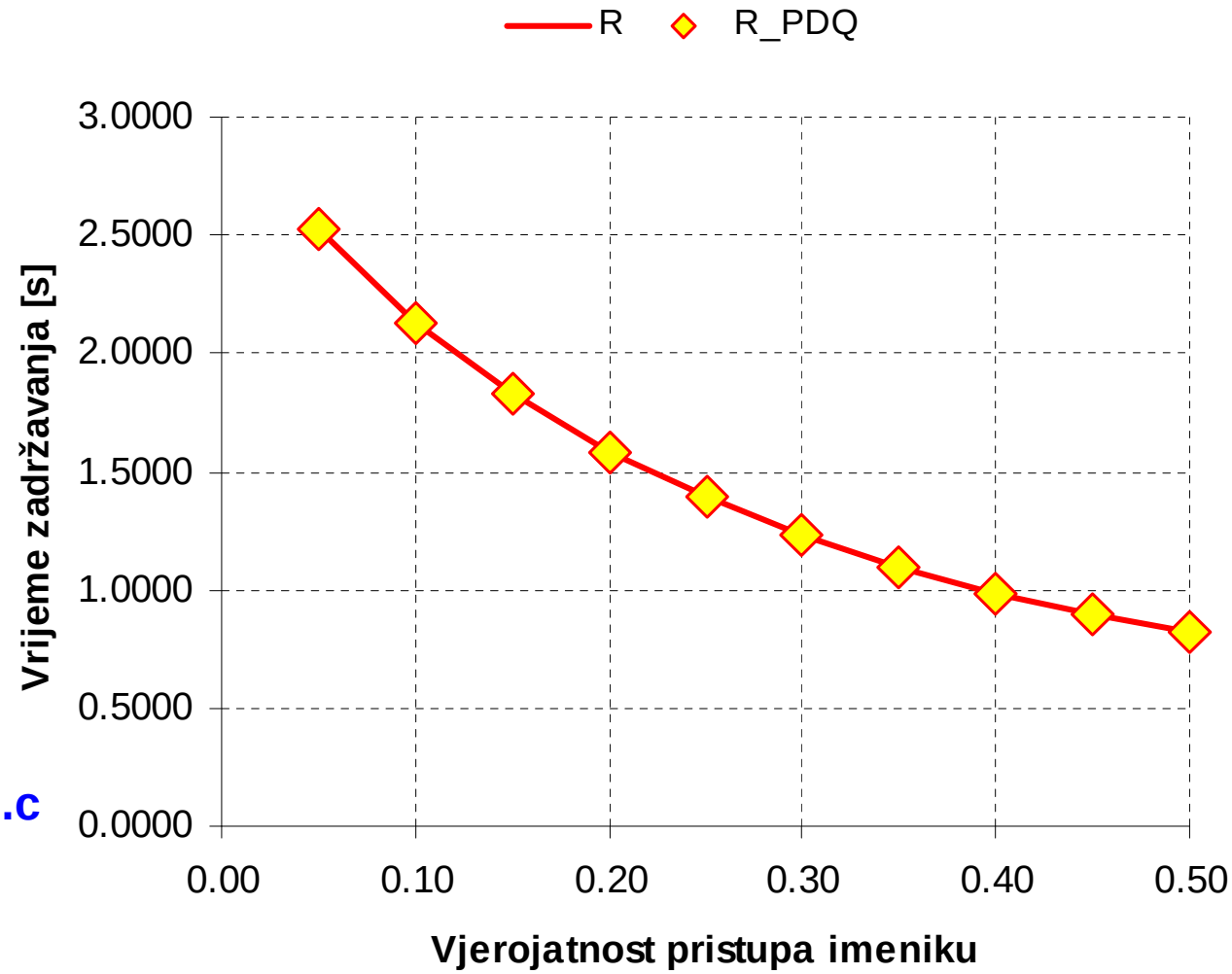
Utjecaj stupnja sigurnosti

- **Imenik aplikacije sadrži informacije o korisnicima**
  - Korisnički identiteti
  - Korisnička prava pristupa
- **Sigurnosna značka**
  - Određuje sigurnosne postavke korisnika aplikacije
  - Značka se dohvaća iz imenika
- **Životni vijek sigurnosne značke**
  - Ograničeni broj pristupa
  - Zadano vrijeme korištenja
  - Ostali sigurnosni modeli

Promjena stupnja sigurnosti



Vrijeme zadržavanja zahtjeva



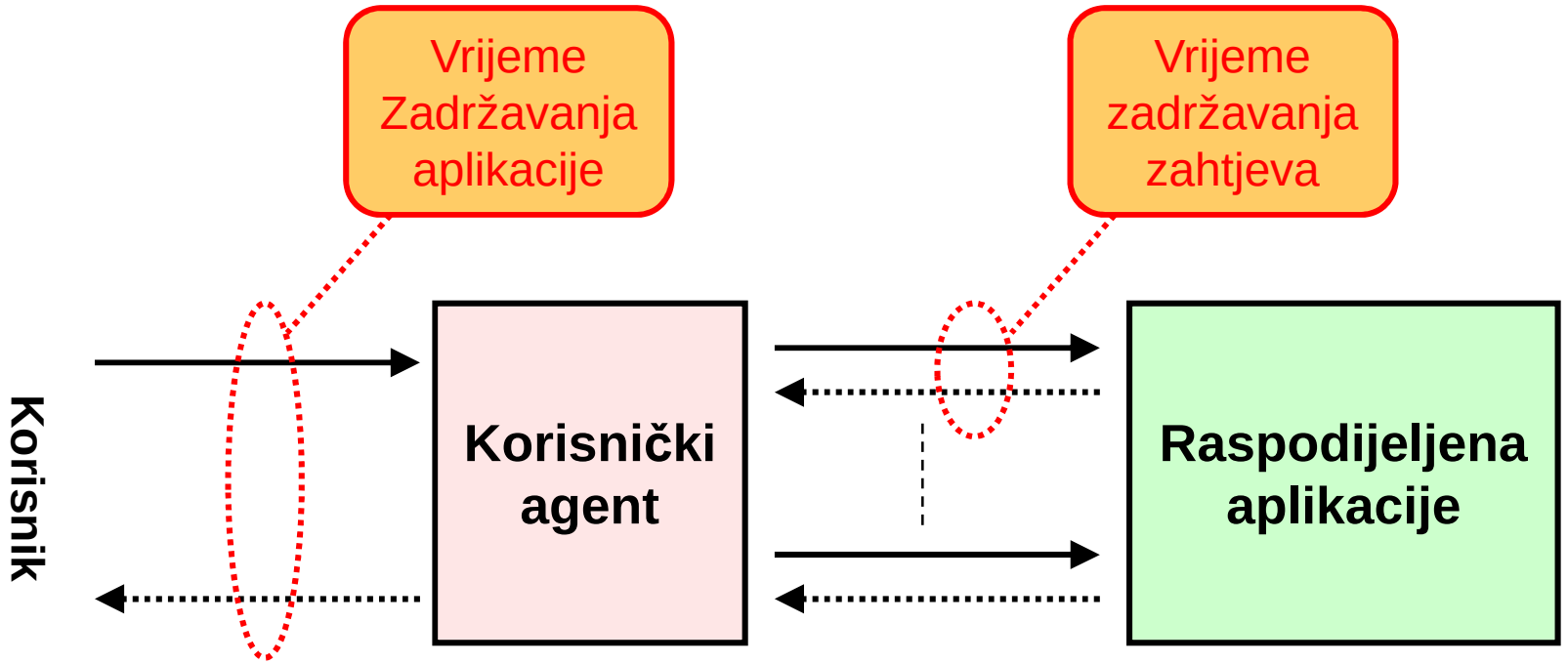
N_{AP} = 1
N_{BP} = 1
N_{IM} = 1
S_{AP} = 0.3
S_{BP} = 4.5
S_{IM} = 0.5
S_{ZZ} = 0.001
S_{PO} = 0.001
p_{BP} = 0.2
L = 0.5

ap-Sig.c

Ap_Sig.xls

Utjecaj stupnja sigurnosti

- **Zašto vrijeme zadržavanja opada?**
  - Modelirano je vrijeme zahtjeva zadržavanja ali ne i ukupno vrijeme zadržavanja aplikacije koje doživljava korisnik



Domaća zadaća

- **Zadatak 1:** Web aplikacija uključuje podršku korisnicima putem chat usluge. Kupci sami odabiru jedan od 10 repova čekanja u kojima upite poslužuje po jedan tehničar. 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 razgovoru.

  (nadogradnja primjera 6)

  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?

  2) Kakve će rezultate dati smanjenje razgovora na 1.5 minutu?

Domaća zadaća

- **Zadatak 2:** Oblikovati proizvoljnu raspodijeljenu aplikaciju i ostvariti analizu performansi ostvarene aplikacije

  1) Definirati logičku i fizičku arhitekturu aplikacije

  2) Izgraditi model aplikacije primjenom teorije repova

  - Odrediti analitičko rješenje funkcije zadržavanja zahtjeva u aplikaciji R=f(L)

  3) Izgraditi model aplikacije za alat PDQ

  - Primjenom izgrađenog modela odrediti vrijednosti funkcije zadržavanja zahtjeva R=f(L) u nekoliko točaka

  4) Usporediti i obrazložiti dobivene rezultate

Rekapitulacija

Rekapitulacija

- **Zašto je vrednovanje performansi važno ?**
  - Zbog potrebe za planiranjem kapaciteta koji je potreban za uspješno poslovanje
- **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

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 tipične zone u sustavu Web servisa ?**
  - Demilitarizirana, aplikacijska i zona podataka
- **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
- **Kako se mjeri i modelira kapacitet servisnog sustava?**
  - Mjerenjem pomoću generatora tereta, te aproksimacijom pomoću polinoma
- **Kako se određuje vrijeme potrebno da se dostigne potreba za dvostrukim kapacitetom?**
  - Podrazumijevajući eksponencijalni trend

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 ?**
  - Markovljev lanac