**Strežna mreža:**

* *poljubna vezava poljubnega št. Strežnih enot*

* - intenzivnost strežbe [št. Zahtev/sec]*

* - povprečni strežni čas*

* - intenzivnost prihajanja zahtev [št. Zahtev/sec]*

* - uporabnostni faktor [  ]*

*Ne sme biti  >...pomeni da sis. ni zmožen sprocesirat toliko zahtev*

**Strežni sistem:**

KENDALOVA NOTACIJA

A/B/m/K/M/Q

A – porazdelitev medprihodnih časov

B – porazdelitev strežnih časov:

m – število strežnikov

K – kapaciteta sistema K= št. strežnikov + vsota čakalnih vrst, neskončna

M – velikost populacije zahtev: končna, neskončna

Q – čakalna disciplina: fifo, lifo, rnd, priority, time sharing

velja za A in B:

D – determinitična porazdelitev

M - eksponentna porazdelitev

E - erlangova porazdelitev

G - splošna porazdelitev

**Numerične značilnosti strežnih sis.**

*N – povprčno št, zahtev v sis.*

*T – povprečen čas bivanja zahteve v sis.*

*– čas bivanja k-te zahteve v sis. (čas v vrstah + čas za strežbo)*

* - verjetnost k – zahtev v sis. v času t*

LITTLEOV TEOREM:

**

* - povprečno št. zahtev v vrsti*

* - povprečno število zahtev v strežnikih*

Zakon o ohranitvi pretoka:

*A – število prispelih zahtev*

*B – število postreženih zahtev*

*zakon 🡪 A=B*

*če je A>B pride do explozije sis. ( > 1)*

*A<B to pa ni možno ker sistem ne generera zahtev*

**POISSONOVI PROCESI:**

* *velika vstopajoča populacija ekvivalentnih zahtev*
* *lahko si jih predstavljamo, kot proces štetja nakjučno se porajajočih točk na časovnem int. [0,t]*

* sistem je na začetku prazen (pred pogoj)*

* verjetnost da je v času t v sis. vstopilo k zahtev*

Lastnosti POISSONOVIH procesov:

*Superpozicija: ob predpostavki, da združimo k neodvisnih Poissonovih proc., bo tudi nov proc. Poissonov proces*

*brez pomnenja:vsaka novoporojena zahteva se je porodila brez vednosti, kdaj se je rodila njena predhodnica*

**Stohastični procesi:**

* *je proces, kjer je dinamika oz. del dinamike pogojena verjetnostno*

Stohatični proc. definiramo na podlagi treh zakonitosti:

1. *prostor stanj (če je končna mn. stanj, pomeni da so stanja diskretna(stohatične verige), če ni končna mn. stanj. pa pomeni da so stanja na intervalih na zvezni osi(proces z zveznimi stanji))*
2. *indexni parameter (uporaba zveznega časa(proces z zveznim časom), uporaba diskretnega časa(stohastično zaporedje))*
3. *statistična odvisnost (vsaj del dinamike mora biti verjetnostno pogojen)*

**Markovski proces:**

* *je stohatični proces, brez pomnenja*

*Poznamo zvezno časovne in diskretne časovne M.P.*

**Markovske verige (diskretni časovni markovski procesi):**

*Imamo opravka s stohastičnostjo, diskretnim časom in brez pomnenja ob menjavi stanj.*

*Prehajalne verjetnosti so lahko odvisne od časa(se pravi se spreminjajo) ali neodvisne od časa(stacionarne prehajalne verj. ...verjetnosti so skos enake). Če so vse prehajalne verjetnosti stacionarne, govorimo o homogeni markovski verigi.*

**

* - vektor verjetnosti za k-to stanje (k=0 vektor verjetnosti za začetno stanje)*

*M – matrika verjetnosti prehajanja stanj (matrika je n\*n n=št. stanj)*

Lastnosti diskretnih časovnih M.V.:

1. *Stacionarnost: obstajajo limitne vrednosti verjetnosti stanj, ki niso odvisne od začetne porazdelitve verjetnosti.*
2. *Reducibilnost: je nereducibilna, če je vsako stanje dosegljivo iz vseh ostalih stanj v končnem št. korakov. Je reducibilna, če vsebuje več kot eno izolirano podmnožico stanj.*
3. *Peroidičnost: sistem M.V. je periodičen s periodo t, če se po t\*n korakih vrača v isto stanje M.V.*
4. *Povrnljivost (rekurenčnost): povrnljivo stanje j je tiso, pri katerem je verjetnost nahajanja v stanju j pri zelo velikem številu korakov (lim🡪neskončno) večja od 0. *

**Rojstno smrtni proces:**

* *posebna oblika zvezno časovnih M.V.*
* *prehodi so možni le v sosednja stanja*
* *predpostavka: rojstvo in smrt zahteve ne nastopita istočasno*

**Petrijeve mreže:**

*So logična struktura. Glavni cilj je postavljanje modelov in proženje simulacij. Petrijeve mreže izpostavijo pomankljivosti kot so: deadlock (čakanje drug na drugega), ozka grla, neželjena stanja sistema.*

*Osnovni gradniki :*

*- pogoji*

* *akcije*
* *povezave akcij in pogojev*
* *žetoni v pogojih -> št. žetonov ∈{N,0}*
* *št. žetonov odraža kolikokrat je nek pogoj izpolnjen*

graf Petrijeve mreže: - pogoj

- akcija

- povezava

- žeton

Def: PM je četvorček C=(P,T,I,O)

*P – končna množica pogojev (places)*

*T – končna množica akcij (tranzitions)*

*I – matrika vhodne funkcije*

*O – matrika izhodne funkcije*

*Za I in O velja da je št. stolpcev = št. pogojev in št. vrstic = št. akcij*

*P,T,I,O - so časovno nespremenljivi*

*Je usmerjen graf, povezave vodijo iz pogoja v akcijo, in obratno. Žetoni se zadržujejo večji čas samo v pogojah. Prehod skozi akcijo je hipen.*

Omogočenost akcije tj :

*Akcija tj je omogočena če velja: *

*o(t) – postavitev žetonov v času t*

*e[j] – enotski vektor (na j-tem mestu je 1 [0,1,0])*

Posledica proženja akcije tj :

**

*o(t+1) – postavitev žetonov po proženju akcije tj.*

*e[j] – enotski vektor (na j-tem mestu je 1 [0,1,0])*

Dinamika P.M.

*- dinamika je odvisna od označitve*

*- posamezna akcija se izvede, če je omogočena*

*- akcija je omogočena, če se po vseh povezavah, ki vanjo vodijo iz pogojev lahko pripeljejo žetoni*

*- po definiciji so časi trajanja akcij HIPNI-NIČNI*

*- paralelnega proženja akcij ni; v enem diskr.čas.koraku se lahko sproži samo ena!*

*- če je v času t omogočenih več akcij, se nedeterministično izbere le ena in se izvede*

Varnost P.M.

*Pogoj je varen, če se v času simulacije št. žetonov v njem ne povzpne nad 1. P.M. je varna, če so v njej varni vsi pogoji.*

Omejenost P.M.

*Pogoj pi v PM z začetno označitvijo O je k-omejen, če za vse dosegljive označitve velja o'(pi)≤ k oz. Pogoj vPM z začetno označitvijo je k-omejen, če se št. žetonov med simulacijo v njem ne povzpne nad k, ga pa doseže.*

*P.M. je k-omejena, če je k enak pogoju, ki ima največji k.*

Konzervativnost P.M.

*V P.M. skušamo zadrževati enako št. žetonov. P.M. je striktno konzervativna, če za vsako označitev, velja, da ima enako št. žetonov kot v začetni označitvi.*

**>razlika med strežno enoto in strežno mrežo**

*Strezna enota: - ponuja en tip strežbe*

*- vrši jo lahko več strežnikov*

*- čakalna vrsta je le ena*

*Strežna mreža: - poljubna kombinacija/vezava strežnih enot*

**>opisat strežno enoto lambda, ni, ena vrsta,...več strežnikov vzporednih**

**>M/M/1**

***∞∞***

∞ μ

λ

FIFO vrsta

*1 strežnik => 1 vrsta*

*poissonov proces prihajanja zahtev*

*exponentna porazdelitev strežnih časov ()*

*velja: statičen v času λk= λ, μk= μ*

*strežnik lahko obdeluje le 1 zahtevo*

*strežnik je pripravljen na takojšni sprejem zahteve, če je prazen*

**> omahljiv M/M/1**

*-omahljiv M/M/1 je tisti pri katerem se intenzivnost prihajanja zahtev spreminja od št.zahtev v sistemu*

**>M/M/1/S**

(S-1) μ

λ λ'

λ\*Pb(verjetnost zasedenosti sistema Pb=Ps)

λ'≤ λ

*S -> 1 streznik*

*S-1 mest v vrsti*

*- ko je v sistemu S zahtev začne sistem zahteve odklanjati(»izguba!«)*

**>M/M/m**

*- imamo paralelno vezanih* ***m****, funkcionalno ekvivalentnih strežnikov*

*- vsaka zahteva obdela natanko en strežnik*

*- SHEMA ROJEVANJA IN SMRTI*

**> Round Robin model**

*- zgod. Model*

*- procesiranje zahtev v časovnih rezinah(dodeljevanje časovnih rezin – TIME SHARING)*

*λ*

*- značilnost strežbe : zahteva dobi le t-procesnih rezin*

*-* ***n*** *je povprečno št.zahtev 🡪 (n-1)t*

*čakalni čas zahteve*

**KENDALOVA NOTACIJA**

**notacija A/B/m/M/K/S**

*A . . . porazdelitev intenzivnosti prihajanja D – diskretna*

*M – exponentna (default)*

*G – verjetnostna*

*B . . . porazdelitev intenzivnosti strežbe D – diskretna*

*M – exponentna (default)*

*G – verjetnostna*

*m . . . št. strežnikov*

*M . . . velikost populacije zahtev(default = ∞)*

*K . . . kapaciteta strežnega sistema: max. št. zahtev, ki se lahko zadržuje v sistemu*

*K=m+dolžina vrste*

*S . . . strategija jemanja iz vrste(dolžina vrste -> default = ∞ )*

*FIFO,LIFO(sklad),RND,Prioriteno,TIME SHARING(dodeljevanje časovnih rezin)*

**>implicitno, eksplicitno, kaj je kaj**

**Npr. Zasedanje resursov:**

**Implicitno ( *Poštena do vseh )***

D1

D3

D2

R1 R1 R1 *STRATEGIJA ODELJEVANJA*

**Eksplicitno ( *Favorizira prvoprispele )***

G

D1

D2

D3

F

R1

**>littleov teorem**

*N=λ\*T*

|  |  |
| --- | --- |
|  | ***Generate-generator zahtev***  A GENERATE activity generates the arrival of entities into the  model. Arrivals may be random, deterministic or conditional. An  example of a GENERATE activity is the arrival of patients in a  clinic. A GENERATE activity may have values for arrival time,  quantity, frequency and occurrences.  ***Dispose***  A DISPOSE activity disposes of the entities when they are finished  with processing. A DISPOSE activity can be used for collecting  customized statistics for throughput or throughput time.  ***Delay***  A DELAY activity defines value added or non value added activity  times. It is one of the most commonly used activities in SIMPROCESS.  A DELAY activity with resource constraints provides queue  statistics that can be used for analyzing wait times. |

***Assemble***

*An ASSEMBLE activity assembles multiple entities coming from*

*multiple sources to create a single entity. For example, the development*

*of a business proposal may contain three documents that are*

*merged using an assembly activity.*

***Branch – Vejišče poti(narezen)***

*Izbira poti se vrši glede na: verjetnost ali prioritete ali tip entitete ali vrednost atributa entitete*

*A BRANCH activity allows for defining alternative routings for flow objects. Branching may be based on a probability or a condition. For example, the outcome of an inspection process may bemodeled using probabilistic branching.*

***Merge – Velišče poti(skupaj)***

*A MERGE activity provides a mechanism for merging a number of*

*connectors into a single connector.*

|  |  |
| --- | --- |
| ***Batch – Lepljenje entitet(začasno)***  *Lepljenje odvisno od: -št.entitet*  *-vredn. atributov entitet*  *A BATCH activity combines a given quantity of entities into a single batch. An example of a batching activity is the accumulation of mail for delivery.*  ***Unbatch – razcepljanje entitet(začasno)***  *An UNBATCH activity splits a previously batched entity into individual entities. For example, unloading of a truck that results in*  *multiple loads may be modeled with an unbatch activity.* | B  U  vidno nevidno vidno |

***Split – rezceplja zahteve (trajno)***

*A SPLIT activity takes an incoming entity and creates clones of that entity as well as providing an output of the original entity. For example, clones of a purchase order may be created with a SPLIT*

*activity and sent to accounts payable and shipping.*

***Join – Lepi zahteve (trajno)***

*A JOIN activity takes the clones and original entity that were split up, and matches them to produce the original one. For example, a JOIN activity may be used for matching the paperwork with the*

*shipment.*

***Transform – Preslika entitetni tip v drugega***

*A TRANSFORM activity converts an incoming entity into another entity. For example, a prospective buyer is transformed into a customer when an order is placed. This activity can be modeled using*

*the transformation construct.*

***Copy***

*A COPY activity makes multiple copies of the original entity. For example, if a document is being edited in a groupware software that results in multiple copies of a file, this activity may be modeled*

*using a COPY activity.*

***Gate - vrata***

*A GATE activity holds entities in a queue, until a signal is received. For example, a GATE activity would be used to model orders held in inventory until a signal is received from the distributor to fulfill*

*the demand.*

***Assign – priredi (+,-,/,\*)***

*An ASSIGN activity provides a mechanism for defining or changing attributes values.*

***Synchronize – sinhronizira vhode več entitet in jih prenaša na izhodno stran***

*A SYNCHRONIZE activity takes inputs that arrive at different times and outputs them in a synchronized fashion. For example, passengers and their baggage must be synchronized at a terminal.*

***Replenish Resource***

*This activity allows for replenishment of consumable resources.*

***Get Resource***

*This activity provides a mechanism for capturing resources that may be used for a number of downstream activities.*

***Free Resource***

*This activity provides a mechanism for releasing resources that were captured by a GET RESOURCE activity.*