Optične tehnologije in nanotehnologije

4. l. UNI-RS

Poročilo 2. seminarske

na temo DNK procesiranje

Mazalin Matija
Laharnar Vasja
Rangus Simon

Ljubljana, 14. 1. 2010

Kazalo

1 Uvod

Ena od možnih alternativnih platform procesiranja podatkov postajajo tudi biološki sistemi. Temelj takega procesiranja je zapis dinamike v obliki DNK zaporedij in prisotnost specifičnih proteinov kot prožilcev te dinamike. Glavni del DNK zaporedja je zapis pogojev za proizvajanje proteinov in navodila za njihovo izdelavo. Na tak način si lahko DNK zaporedje predstavljamo kot računalniški program, posamezne proteine pa po eni strani kot vhodne podatke, s pomočjo katerih se samo procesiranje sproži, po drugi strani pa kot izhodne rezultate procesiranja. Z realizacijo pomnjenja lahko v biološke sisteme vpeljemo tudi sekvenčno procesiranje. Procesiranje v bioloških sistemih lahko poteka v eni ali več gostiteljevih celicah. V celici se nahajajo naslednji pomembni gradniki, ki vplivajo na programiranje dinamike v njej:

- **zapis DNK**
  Kodiran je s štirimi nukleotidnimi bazami (A, G, C in T) in je poljubne dolžine. Poleg kontrolnih informacij hrani osnovna navodila za generiranje proteinov.

- **kodon**
  Je osnovni sestavni del zaporedja DNK. Posamezni kodon lahko definira tvorbo ciljnih aminokislin ali pa določa t.i. stop kodon.

- **promotor**
  Je del zapisa DNK, na katerega se mora vezati encim RNK polimeraze. S tem je izpolnjen potreben pogoj za proženje proizvajanja proteina v celici po programu zapisanem na ostalem delu DNK zaporedja.

- **operator**
  Je del promotorja, na katerega se vežejo vnaprej določeni transkripcijski faktorji.

- **transkripcijski faktor**
  Je specifičen protein, ki s svojo vezavo na operator promotorja pospešuje oziroma zavira vezavo RNK polimeraze na promotorski del DNK zapisa in s tem neposredno odloča o intenzivnosti tvorbe izhodnega proteina. Transkripcijske faktorje delimo na aktivatorje in represorje.

Dinamiko v celici oz. tvorbo novih proteinov delimo na fazi transkripcije in translacije. Tvorba proteinov se sproži ob vezavi RNK polimeraze na promotor DNK zapisa ob prisotnosti oziroma odsotnosti specifičnih transkripcijskih faktorjev. Po vezavi na promotor RNK polimeraza potuje vzdolž dvojne vijačnice DNK in shranjen genetski zapis prepisuje v mRNK (sporočilno RNK). Proces prepisovanja DNK zapisa v mRNK zapis imenujemo transkripcija. Proces transkripcije se ustavi ob stop kodonu. Po procesu transkripcije se na novo nastalo molekulo mRNK vežejo ribosomi, s čimer se začne prevajanje mRNK zapisa v zaporedje aminokislin, ki tvorijo ciljni protein. Ta proces imenujemo translacija.

Intenzivnost generiranja določenega proteina spremljamo tako, da vzporedno z njim proizvajamo še proteine zaznavne z danes dosegljivimi tehnologijami (npr. zeleni fluorescirajoči protein ali t.i. GFP).

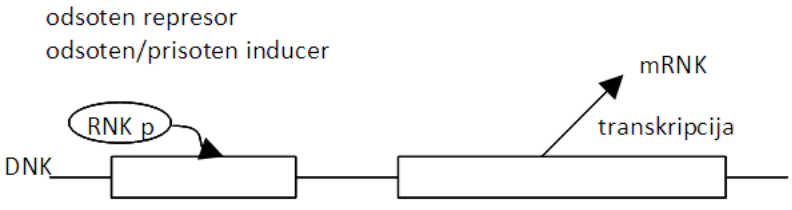
Transkripcijske faktorje glede na njihov učinek delimo na aktivatorje, ki vezavo RNK polimeraze pospešujejo (aktivirajo, inducirajo) in represorje, ki vezavo zavirajo (represirajo, inhibirajo). Aktivatorji z vezavo na operator večajo frekvenco transkripcij mRNK, oziroma sam proces transkripcije sploh omogočijo, represorji pa proces transkripcije zavirajo. Vplive transkripcijskih faktorjev lahko grafično ponazorimo z notacijo predstavljeno na spodnji sliki:


inhibicija transkripcije (vpliv represorja)

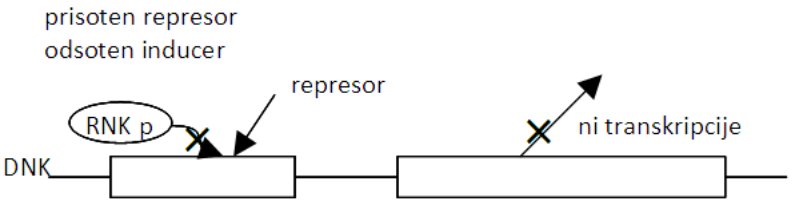

aktivacija transkripcije (vpliv aktivatorja)

Spodnja slika prikazuje tri možne primere ob prisotnosti in odsotnosti represorjev in aktivatorjev oz. inducerjev.

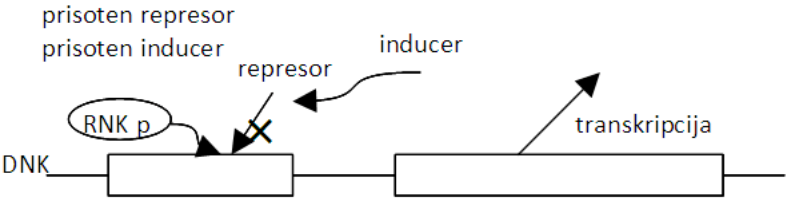
1.



2.



3.



Z vidika klasičnih logičnih digitalnih struktur bi lahko predhodno navedene gradnike razdelili na vhodne, procesne in izhodne entitete po naslednjem principu:

- vhodni segment: prisotnost oziroma odsotnost proteina s funkcijo transkripcijskega faktorja;
- decizijski segment: zapis DNK (program), ki vsebuje navodila za tvorbo ciljnega izhodnega proteina;
- izhodni segment: novogenerirani protein, ki ima lahko tudi funkcijo transkripcijskega faktorja, ni pa to nujno.

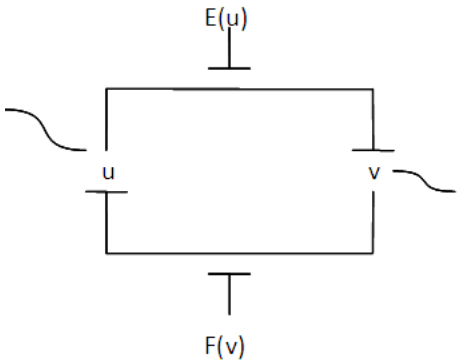
2 Model realizacije funkcionalnosti RS celice v Matlabu

V biološkem sistemu želimo realizirati funkcionalnost RS pomnilne celice, ki deluje na naslednji način:

u in v ... notranje stanje celice (izhod)
I_1, I_2 ... vhoda
I_1 ... R
I_2 ... S
u ... q
v ... $\overline{q}$

I_1	I_2	D^1_q	$D^1_{\overline{q}}$
0	0	q	$\overline{q}$
0	1	1	0
1	0	0	1
1	1	x	x

Model je sestavljen iz dveh vhodnih proteinov u in v in dveh aktivatorjev E(u) in F(v), kot je prikazano na sliki:



$u(t+\tau)$ = u(t) + sinteza - degradacija
$v(t+\tau)$ = v(t) + sinteza - degradacija
u ... 1 => u ... 0
v ... 0 => v ... 1

Izhajamo iz Matlab modela, ki je sestavljen na naslednji način.

```
   toggle_molecular.m
   % MODELIRANJE PREKLOPNEGA STIKALA % PODATKI SO PODANI IN DOBLJENI
V ENOTAH število molekul - gledamo molekularna števila
   n = 3; % n = koeficient kooperativnosti
   % koliko enot vhodnega proteina se mora naenkrat vezati na operator
```

```
   alpha1 = 0.2*500; % alpha1 = efektivna hitrost transkripcije
   alpha2 = 0.2*500; % alpha2 = efektivna hitrost transkripcije
   beta1 = 4*500; % beta1 = hitrost degradacije proteina
   beta2 = 4*500; % beta2 = hitrost degradacije proteina
   epsilon = 1; % epsilon = 1
   d1 = 1; % d1
   d2 = 1; % d2
   K1 = 1*500; % K1
   K2 = 1*500; % K2
   gamma = 1; %
   gamma s = 1.7; % s
   tau = 0.1; % tau = časovni korak
   u0 = 2125; % u0 = začetna koncentracija u
   v0 = 125; % v0 = začetna koncentracija v
   clear u;
   clear v;
   N = 1200; % število korakov
   [u,v] = toggle_model(u0, v0, n, alpha1, alpha2, beta1, beta2, epsilon,
d1, d2, K1, K2, gamma, s, tau, N);
   clear x;
   for i=1:N x(i) = (i-1)*tau; end;
   plot(x,u); % risanje grafa hold on; plot(x,v,'r');
   hold off;

   toggle_model.m
   % MODEL PREKLOPNEGA STIKALA
   function [u,v] = toggle_model(u0, v0, n, alpha1, alpha2, beta1,
beta2, epsilon, d1, d2, K1, K2, gamma, s, tau, N)
   u(1) = u0;
   v(1) = v0;
   for (t=2:N)
   sint_u = poissrnd( average_sint(epsilon, alpha1, beta1, K1, n, v(t-1),
tau) );
   degrade_u = poissrnd( u(t-1)*(d1 + (gamma*s)/(1+s))*tau );
   sint_v = poissrnd( average_sint(epsilon, alpha2, beta2, K2, n, u(t-1),
tau) );
   degrade_v = poissrnd( d2*v(t-1)*tau );
   u(t) = max(0,u(t-1) + sint_u - degrade_u);
   v(t) = max(0,v(t-1) + sint_v - degrade_v);
   end;

   average_sint.m
   % RAČUNANJE SREDNJE VREDNOSTI SINTETIZIRANEGA PROTEINA
```

```
   function sint = average_sint(epsilon, alpha, beta, K, n, protein,
tau) sint = epsilon * (alpha + (beta * K^n)/(K^n + protein^n)) * tau;
```

3 Idejne rešitve in izboljšave

Sprememba sintezne funkcije iz

```
   function sint = average_sint(epsilon, alpha, beta, K, n, protein,
tau) sint = epsilon * (alpha + (beta * K^n)/(K^n + protein^n)) * tau;
```

v

```
   function sint = average_sint(epsilon, alpha, beta, K, n, protein,
tau,E) sint = epsilon * (alpha + (beta/(1+(protein^n/K^n/(1+(E/K)^n)))))
* tau;
```

Vpeljemo torej Hillovo enačbo, ki upošteva še učinke represorjev in aktivatorjev pri sintezi proteinov.

3.1 Mrežni motivi (Network motif)

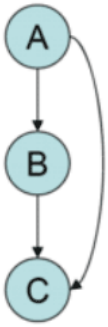
»Network motif« je pojem vzorcev ki se dogodijo znotraj velikih omrežij večkrat kot bi lahko pričakovali. Večina motifov je obravnavana v biologiji. V genskih »dnk« omrežjih je bilo na to temo že ogromno poskusov ter prelitega je bilo že ogromno črnila. Načeloma se ta omrežja odzivajo na različne biološke signale(aktivator, represor, inducer,...). Omrežja so definirana tako, da so vozlišča geni, povezave pa transkripcijski faktor gena. Glavna hipoteza pravi da se najboljši model izbere s pomočjo evolucije. Tako smo priča sintezi in degradaciji proteinov v nekem sprejemljivem časovnem okvirju. Iz tega pa sledi da so »network motifi« nekakšni osnovni gradniki v genskih regulatornih vezjih, prav tako se pa odvijajo tudi v naravi.

V tem poglavju se bom osredotočal na avtoregulativne in »Feed forward loop motife« ter implementacija in uporaba le-teh v naši seminarski.

Autoregulativni Motif

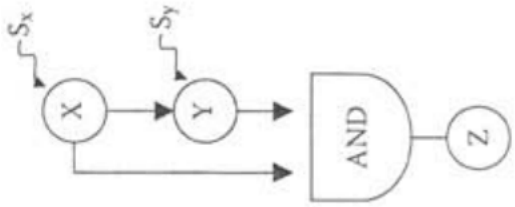
Avtoregulativni model je, kot je prikazano na sliki, je model kjer transkripcijski faktor podeduje takoj svoj prejsnji rezultat in število proteinov. Imajo počasen reakcijski čas.

Feed-forward loops (FFL)

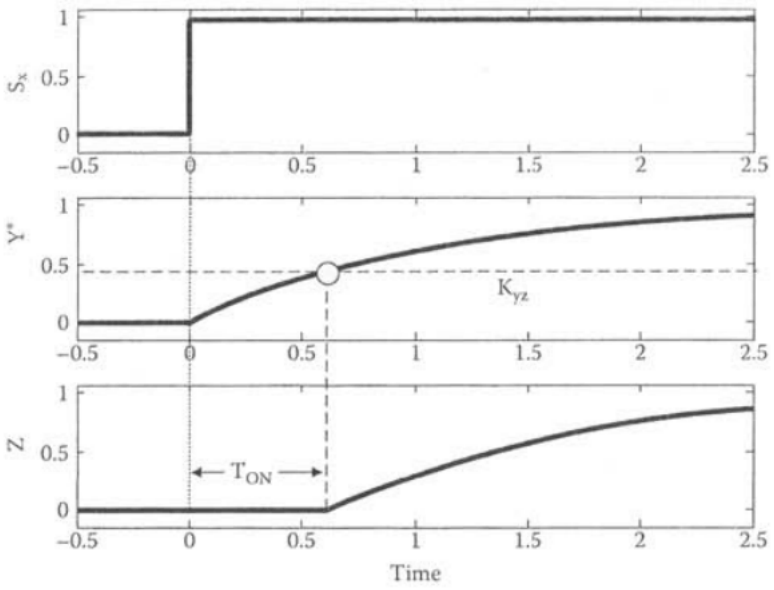


To je najpogosteje najden in uporabljan motif v genskih regulativnih vezjih. Motif je sestavljen iz treh genov in treh regulativnih interakcij. Ciljni gen Z regulirata 2 ostala gena, X in Y. Poleg tega je Y tudi krmiljen s strani X. Zaradi vseh možnih permutacij genov(spomnimo: vsak je lahko pozitiven ali negativen) obstaja 8 različnih tipov FFL motifov. 2 izmed njih, enega bomo tukaj opisali bolj obširno, drugega pa se le dotaknili sta C1-FFL (koherentni tip) in I1-FFL(nekoherentni tip), oba sta tudi najbolj navzočav transkripcijskih regulativnih vezjih. Z njim najpogosteje realiziramo AND ter OR vrata(konjunkcijo in disjunkcijo), možna je pa tudi realizacija drugih funkcij.

C1-FFL



C1-FFL je bil pri večkrat prikazan pri AND vratih in izkazalo se je, da imajo te funkcije precejšen »delay«. Predstavljajmo si, da vsaka celica, torej X,Y in Z vsebuje velikansko število proteinov. Vhodni signal, stikalo, aktivator proteina X je S_X. Brez tega je X v neaktiven. V času t=0 močan S_X (aktivator) sproži reakcijo. Transkripcija hitro spremeni gen v mRNK(X*) ki je pripravljen z novo informacijo za naprej, X* se tako veže na promotor gena Y in tako začne z sintezo proteina Y, drugo fazo transkripcije. Poleg tega X* sodeluje tudi pri generiranju proteina Z. Sinteza Z potrebuje za svoj pričetek tako X* kot tudi Y*, saj imamo IN funkcijo. Sinteza Z-ja potrebuje, kot že rečeno vezavo X* in Y*, kar pomeni da se mora koncentracija Y povečati do ustreznega nivoja ki je določen kot prag (treshold K) za gen Z. Poleg tega potrebujemo tudi Y* ki se sproži le ko se sproži močan signal S_y, tako kot smo na začetku čakali na S_x. Vse to vpliva na zakasnitev Z-ja. Nekoliko poenostavljeno se stopnja proizvodnje kjer je βy stopnja ko X* preseže prag aktivacije. Y=βy*ϑ(X*>Kxy)



Slika nam ponazarja zakasnitev sinteze Z proteina po tem ko se pojavi aktivator Sx. Torej v času t=0 dobimo signal Sx.Ta takoj ustvari X* in prične se sinteza Y*. Šele ko Y* doseže prag Kyz se začne sinteza proteina z. Ta čas (Ton) je iz vidika načrtovalca vezja »vržen proč«. Za Sy je privzeto da je ves čas prisoten. Zanimivo je to da obratna varianta, torej ko naenkrat prekinemo signal Sx začne tudi koncentracij Z padati tako, saj nam pada istočasno tudi koncentracija Y.

Pohitritev odziva

Najlažji način pohitritve načina je predvsem povečati hitrost degradacije ter sinteze, poleg tega pa tudi pomanjšanje praga Ky. S tem ko premaknemo prag postane funkcija tudi bolj »ranljiva« za zunanje vplive, kot so šum in ostale zadeve. To smo tudi implementirali v našo seminarsko kjer smo imeli sicer malček drugačen tip regulativnega vezja.

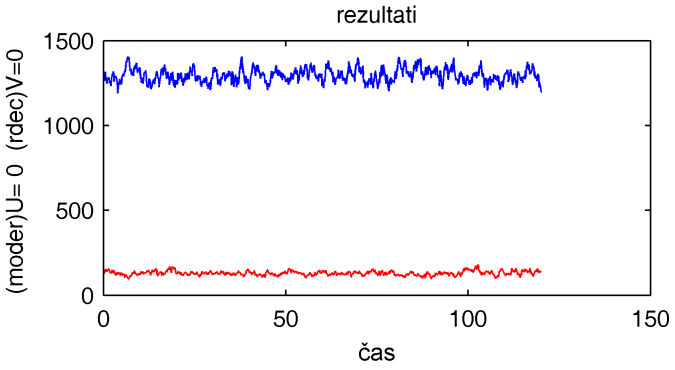
Pri naši seminarski smo poskusili meriti »hitrost« preklopa med funkcijami ter prišli do zanimivih spoznanj. Pri običajni degradaciji proteinov smo zabeležili povprečno (pri 500 poskusih) 3,61 minute za preklop, pri pospešeni pa le 3,12 minute. Pri tem velja omeniti da začetne koncentracije niso igrale nikakršne vloge, idealni prag koncentracije je pa zelo težko najti. Za koncetracije proteinov med 0 in 2000 bi ga bilo po naših opazovanjih najbolje vmestiti nekje okrog 700.

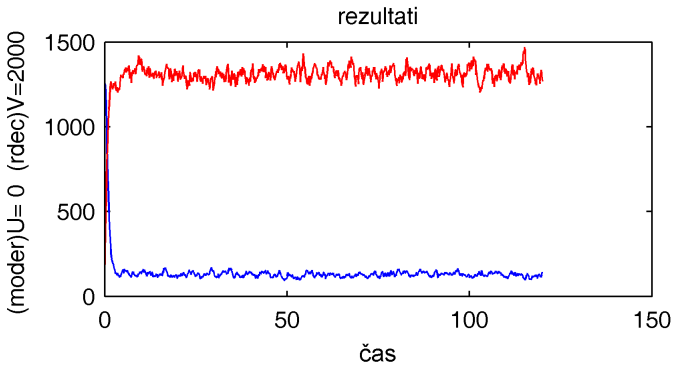
4 Simulacija in rezultati

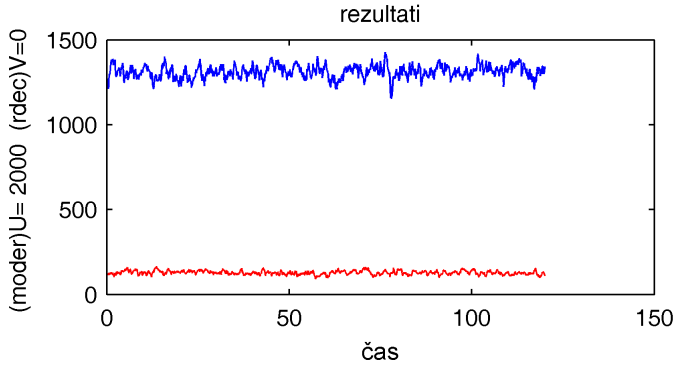
```
% RAČUNANJE SREDNJE VREDNOSTI SINTETIZIRANEGA PROTEINA
function sint = average_sint(epsilon, alpha, beta, K, n, protein, tau,E)
sint=epsilon*(alpha+(beta/(1+(protein^n/K^n/(1+(E/K)^n)))))*tau;

% MODEL PREKLOPNEGA STIKALA
function [u,v] = toggle_model(u0, v0, n, alpha1, alpha2, beta1, beta2,
epsilon, d1, d2, K1, K2, gamma, s, tau, deterministic, N,E,F,stevilo)
a=0;
u=zeros(1,N);
v=zeros(1,N);
u(1,1) = u0;
v(1,1) = v0;
stevec=1;
index=1;
variacija=0;
for t=2:N
if stevilo==length(E)
if mod(stevec,200)==0
a=rand()*2000;
end;
sint_u = poissrnd( average_sint(epsilon, alpha1, beta1, K1, n, v(t-1),
tau,a));
degrade_u = poissrnd( d1*u(t-1)*tau );
sint_v = poissrnd( average_sint(epsilon, alpha2, beta2, K2, n, u(t-1),
tau,4*K1-a));
degrade_v = poissrnd( d2*v(t-1)*tau );
stevec=stevec+1;
elseif stevilo==length(E)-1
if mod(index,200)==0
variacija=abs(variacija-2000);
end sint_u = poissrnd( average_sint(epsilon, alpha1, beta1, K1, n,
v(t-1), tau,variacija));
degrade_u = poissrnd( d1*u(t-1)*tau );
sint_v = poissrnd( average_sint(epsilon, alpha2, beta2, K2, n, u(t-1),
tau,abs(2000-variacija)));
degrade_v = poissrnd( v(t-1)*(d2 + (gamma*s)/(1+s))*tau );
index=index+1;
else
sint_u = poissrnd( average_sint(epsilon, alpha1, beta1, K1, n, v(t-1),
tau,E(stevilo)));
degrade_u = poissrnd( u(t-1)*(d1 + (gamma*s)/(1+s))*tau );
sint_v = poissrnd( average_sint(epsilon, alpha2, beta2, K2, n, u(t-1),
tau,F(stevilo)));
degrade_v = poissrnd( v(t-1)*(d2 + (gamma*s)/(1+s))*tau );
```

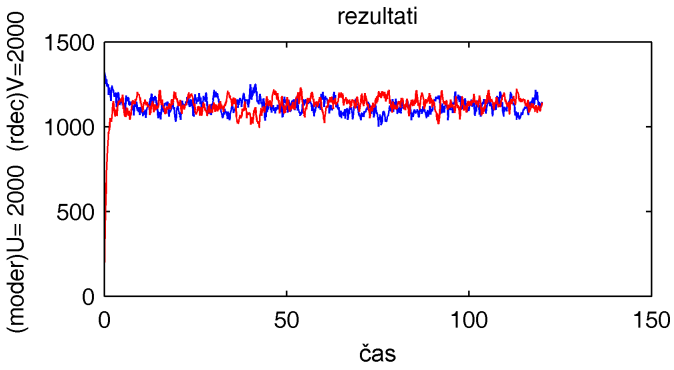
```
end;
u(t) = max(0,u(t-1) + sint_u - degrade_u);
v(t) = max(0,v(t-1) + sint_v - degrade_v);
end;

% MODELIRANJE PREKLOPNEGA STIKALA. % PODATKI SO PODANI IN DOBLJENI
V ENOTAH število molekul - gledamo torej molekularna števila
n = 3;
alpha1 = 0.2*500;
alpha2 = 0.2*500;
beta1 = 4*500;
beta2 = 4*500;
epsilon = 1;
d1 = 1;
d2 = 1;
K1 = 1*500;
K2 = 1*500;
gamma = 1;
s = 1.5;
E=[0 0 2000 2000 500 1000 1500 0 0];
F=[0 2000 0 2000 1500 1000 500 0 0];
tau = 0.1; % časovni korak
u0 = 125; % začetne koncentracije, moder
v0 = 125; %rdeč
Konc=length(E);
N = 1200; % število korakov
u=zeros(Konc,N);
v=zeros(Konc,N);
x=linspace(0,N*tau,N);
x2=linspace(0,100,3);
deterministic = 0; % deterministic = 1 ...  deterministična simulacija;
sicer stohastična
nizki=zeros(2,3);
visoki=zeros(2,3);
clf;
temp=5;
gor=1;
for g=1:Konc
[u(g,:),v(g,:)]  = toggle_model( u0, v0, n, alpha1, alpha2, beta1,
beta2, epsilon, d1, d2, K1, K2, gamma, s, tau, deterministic, N,E,F,g);

end
for a=1:3
nizki(1,a)=switch_speed(u(8,:),temp,abs(1-gor));
visoki(2,a)=switch_speed(v(8,:),temp,gor);
if a==1 temp=temp+394;
```

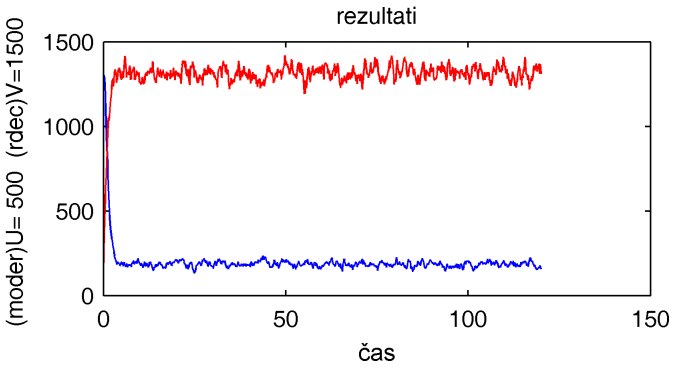
```
else temp=temp+400;
end;
end
temp=201;
for a=1:3
nizki(2,a)=switch_speed(v(8,:),temp,abs(1-gor));
visoki(1,a)=switch_speed(u(8,:),temp,gor);
temp=temp+400;
end
visoki
nizki
for h=1:Konc
subplot(4,3,h);
plot(x,u(h,:));
title('rezultati');
hold on;
plot(x,v(h,:),'r');
s=['(moder)U= ' int2str(E(h)) ' (rdec)V=' int2str(F(h)) ' '];
ylabel(s);
xlabel('čas');
end
subplot(4,3,Konc+1);
plot(x2,visoki(1,:));
title('rezultati preklopa navadne degradacije');
hold on;
plot(x2,nizki(1,:),'r');
ylabel('št korakov za preklop');
xlabel('');
subplot(4,3,Konc+2);
plot(x2,visoki(2,:));
title('rezultati preklopa pospešene degradacije');
hold on;
plot(x2,nizki(2,:),'r');
ylabel('čas preklopa');
xlabel('');
hold off;
```

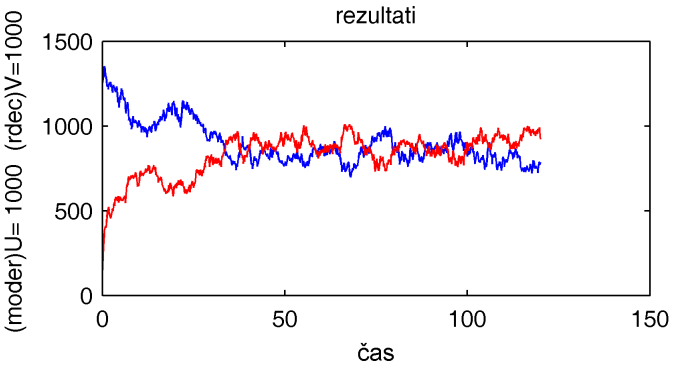

rezultati
(moder)U= 0 (rdec)V=0
1500
1000
500
0
0
50
100
150
čas

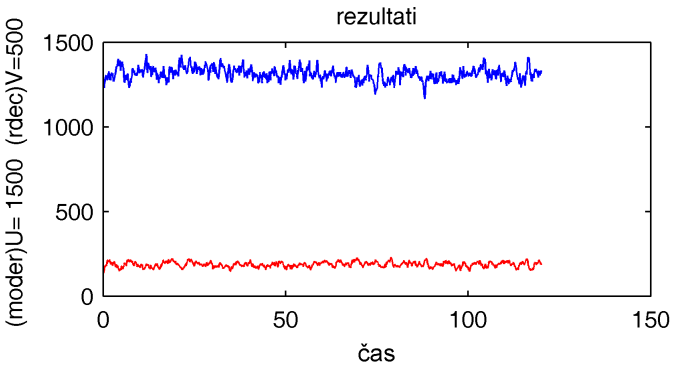

rezultati
(moder)U= 0 (rdec)V=2000
1500
1000
500
0
0
50
100
150
čas

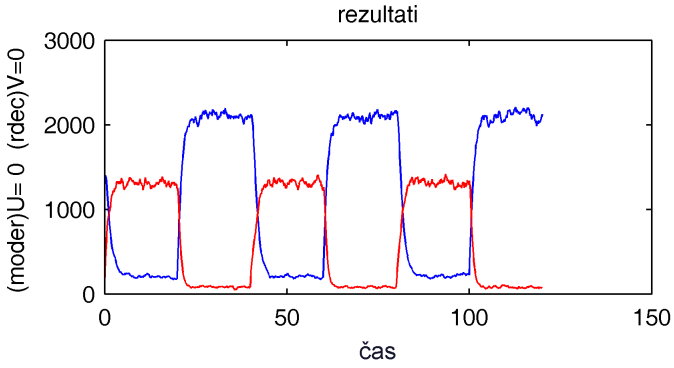

rezultati
(moder)U= 2000 (rdec)V=0
1500
1000
500
0
0
50
100
150
čas

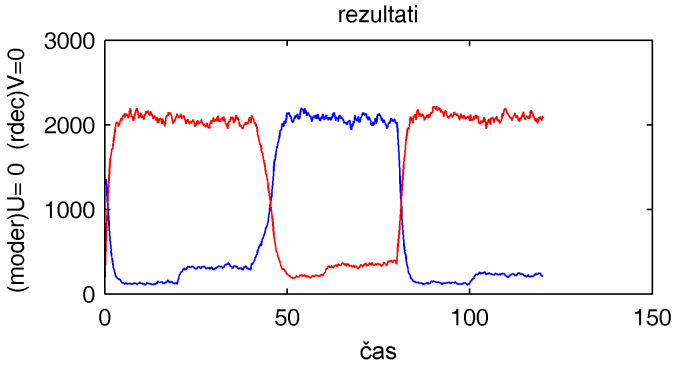

rezultati
(moder)U= 2000 (rdec)V=2000
1500
1000
500
0
0
50
100
150
čas

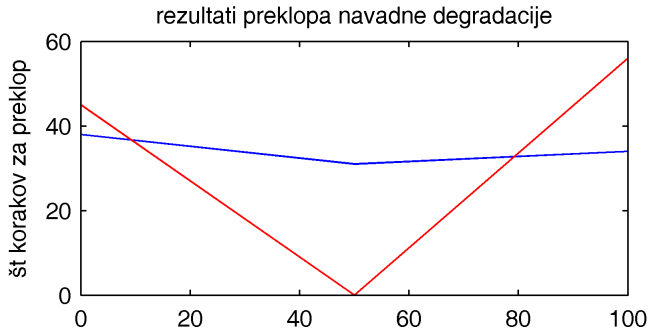

rezultati
(moder)U= 500 (rdec)V=1500
1500
1000
500
0
0
50
100
150
čas

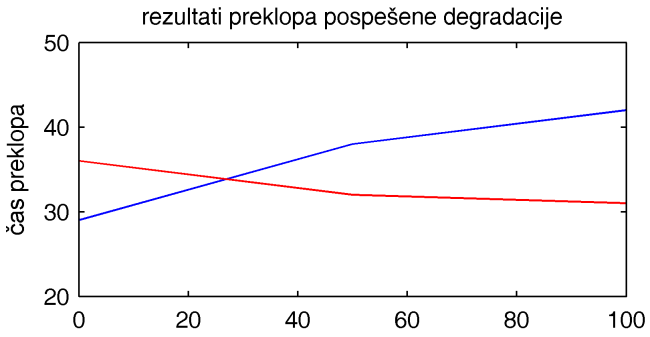

rezultati
(moder)U= 1000 (rdec)V=1000
1500
1000
500
0
0
50
100
150
čas


rezultati
(moder)U= 1500 (rdec)V=500
1500
1000
500
0
0
50
100
150
čas


rezultati
(moder)U= 0 (rdec)V=0
3000
2000
1000
0
0
50
100
150
čas


rezultati
(moder)U= 0 (rdec)V=0
3000
2000
1000
0
0
50
100
150
čas


rezultati preklopa navadne degradacije
št korakov za preklop
0
20
40
60
0
20
40
60
80
100


rezultati preklopa pospešene degradacije
čas preklopa
20
30
40
50
0
20
40
60
80
100

5 Zaključek

Iz grafov je razvidno, da Hillova enačba kar dobro simulira časovni potek proteinov in sam medsebojni vpliv. Z dodajanjem indukcije lahko sprožimo preklop iz logične enice v logično ničlo in obratno. Zanimiv je graf, ki prikazuje inducirana tako u kot v. V tem primeru nosita oba proteina logično enico in v tem primeru imamo nedefinirano oz. prepovedano stanje, kar se kaže tudi v nestabilnen poteku. Opazili smo tudi, da je dobro, če so začetne koncentracije u in v med samo kar se da različne. Na primer, da je u 125 in v 1250, saj je tako sam potek bolj stabilen.

6 Literatura

- Miha Moškon, Monika Ciglič, Roman Jerala, Nikolaj Zimic, Miha Mraz, »Model realizacije funkcionalnosti RS pomnilne celice v biološkem sistemu«, Elektrotehniški vestnik XX(Y): 1-6, YEAR
- http://www.weizmann.ac.il/mcb/UriAlon/Papers/Network_motifs_nature_genetics_review.pdf
- Uri Alon, An Introduction to Systems Biology: Design Principles of Biological Circuits, Chapman & Hall; 1 edition (July 7, 2006)