Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

MODELIRANJE PREKLOPNEGA STIKALA Z ORODJEM SGN SIM

2. SEMINARSKA NALOGA PRI PREDMETU OPTIČNE NANOTEHNOLOGIJE

Matej Pangerc,63060290,
Grega Škaper,63050112,
Leon Golob,63060052,
Gašper Cvenkel, 63050019

Ljubljana, januar 2011

1. UVOD

Genetsko regulatorska omrežja danes predstavljajo eno izmed najbolj plodnih panog raziskovanja moderne bioinformatike. Iz računalniškega vidika so genetsko regulatorska omrežja zanimiva, saj z njimi lahko enostavno realiziramo model v biološkem sistemu, ki ima osnovno logično delovanje električnega digitalnega vezja. Za izdelavo takega sistema je potrebno izvesti simulacijo delovanja takega sistema, da izvemo če naš model v biološkem sistemu pravilno deluje ali ne. Tako smo se odločili, da bomo v seminarski nalogi predstavili modeliranje stohastičnega preklopnega stikala s pomočjo orodja SGN Sim.

2. ANALIZA ORODJA SGN Sim

SGNSim[1] (Stohastic Gene Networks Simulator) je preprosto orodje za simulacijo genetsko-regulatornih omrežij. Posamezne transkripcije in translacije so modelirane kot skupek časovno zamaknjenih dogodkov, ki jih vodi algoritem SSA (stohastic simulation algoritm).

Sam program je konzolna aplikacija, kjer v konzoli poganjamo program skupaj z vhodno datoteko, ki vsebuje želene reakcije.

2.1 Vhodna Datoteka

Format

1. tip podatek

Prvi format se uporablja pri preprostih deklaracijah, kot začetek in konec simulacije, seme (za generator naključnih števil) ali za posamične deklaracije začetnih populacij:

```
population A = 1;
population B = 2;
```

2. tip { (podatek;) }*

Drugi format se uporablja kadar želimo predstaviti bloke podatkov, kot so na primer liste reakcij in začetnih populacij :

```
population {
        A = 1;
        B = 2;
}
```

3. tip !{ podatek }!

1 http://www.cs.tut.fi/~sanchesr/SGN/SGNSim.html

Tretji format dovoljuje, da podatki vsebujejo podpičja in oklepaje. Ta format se po večini uporablja za pisanje kode v programskem jeziku Lua[2] (v večini primerov v Lua bloke vpisujemo konstante ali enačbe za hitrost reakcije).

2.2 Zaganjanje Programa

Program zaženemo iz konzole (pred tem se premaknemo v imenik s sgns) z ukazom:

sgns --include *datoteka*

Datoteka mora vsebovati vse podatke potrebne za izpeljavo simulacije.

Napredek

Če pričakujemo dolgotrajno simulacijo, lahko z ukazom --progress prikažemo napredek naše simulacije.

2.3 Parametri Simulacije

Nekaj osnovnih parametrov za izvajanje simulacije:

time	*starttime*;	Začetna ura.
stop_time	*stoptime*;	Konec simulacije, ko se doseže *stoptime*.
seed	[*seed*];	Seme za generator naključnih števil.
readout_interval	[*interval*];	Interval med zapisi rezultatov v izhodno datoteko.
results_file	*filename*;	Ime izhodne datoteke.

Edini parameter, ki ga moramo obvezno napisati je stop_time, drugače simulacija za start_time in stop_time vzame 0 (simulacija se sploh ne izvede).

2.4 Začetne Populacije

Začetne populacije so predstavljene s tipom population. Primeri:

population A = 1;	Začetno populacijo A nastavimo na 1.
population A += 1;	Populaciji A povečamo za 1.
population A;	Če A že prej ni definiran potem A = 0, drugače A += 0.

2 http://www.lua.org/manual/5.1/

2.5 Reakcije

Reakcije so predstavljene s tipom reaction. Osnovna oblika reakcije:

substrati --[*hitrost_reakcije*]--> *produkti*

Primer združitve A in B pri čemer nastane C s hitrostjo reakcije 10:

reaction A + B --[10]--> C;

Reakcije brez parametrov pri *substratih* ali *produktih*:

--[10]--> C;	Nastajanje (iz ničesar) C s hitrostjo 10.
A --[2]--> ;	Degradacija (uničenje) A s hitrostjo 2.

Če potrebujemo več substratov za eno reakcijo, število napišemo pred ime substrata:

2A --[1]--> B;	Iz dveh A nastane B s hitrostjo 1.

V samo hitrost reakcije lahko zapišemo tudi enačbo, ki jo lahko interpretira Lua:

A + B --[1 + math.random(5)]--> C;	Iz A in B nastane C s hitrostjo 1 + naključna vrednost.

2.6 Čakalna vrsta

V čakalno vrsto lahko postavimo spojine, ki jih ne želimo imeti v naših reakcijah takoj na začetku simulacije ampak šele po določenem času. Če želimo po času 50 v sistem spustiti 10 enot A:

queue 10A(50);

2.7 Primer

Datoteka z imenom *primer.g*:

```
seed;
time 0;
stop_time 10;
readout_interval 0.1;
output_file izhod.xls;

//komentar

/*
se en komentar
*/

lua !{
   ka     = 1.1;          // hitrost sinteze A
   kc     = 2.2;          // hitrost sinteze C
   dc     = 3.3;          // hitrost degradacije C
}!

population {
        A = 100;
        B = 200;
        C;
}

reaction {
        --[ka]--> A;
        A + B --[kc]--> C;
        c + D --[dc]--> ;
}

queue [50]D(5);
```

Primer poženemo kot:

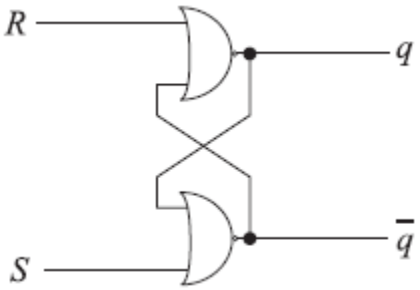
```
sgns --include datoteka          .g
```

Rezultate dobimo v datoteko izhod.xls.

3. STOHASTIČNI MODEL PREKLOPNEGA STIKALA

Preklopno stikalo (ang. toggle switch) je vezje, ki se v splošnem lahko nahaja v enem izmed dveh logičnih stanj, in sicer v aktivnem (visokem) in neaktivnem (nizkem) stanju. Izhod iz vezja predstavlja stanje preklopnega stikala, medtem ko zunanje vzbujanje predstavlja vhod v vezje. Vzbujanje povzroči preklop stikala iz enega stanja v drugo. Idealni model preklopnega stikala izvrši preklop takoj po detekciji spremembe vhodnega signala (vzbujanja). V bioloških in elektronskih sistemih preklop traja nek določen čas. V modernih elektronskih vezjih je ta čas običajno v rangu nekaj nanosekund ali mikrosekund, medtem ko je v bioloških sistemih bistveno večji, in sicer se meri v rangu sekund, minut ali celo ur, odvisno od hitrosti kemijskih reakcij.

Delovanje stohastičnega modela preklopnega stikala(ang. toggle switch) v biološkem sistemu je ekvivalentno delovanju RS-pomnilne celice. Logična shema RS-pomnilne celice in logična prevajalna funkcija RS-pomnilne celice sta prikazani na spodnji sliki:



(a) Logična shema

R	S	D^1q	$D^1\overline{q}$
0	0	q	$\overline{q}$
0	1	1	0
1	0	0	1
1	1	?	?

(b) Prevajalna funkcija

Logična shema na sliki pod (a) ima dva vhoda in dva izhoda. Vhod RESET(R) v aktivnem stanju(logična vrednost 1) postavi izhod (q) na logično vrednost 0, vhod SET (S) v aktivnem stanju pa postavi izhod (q) na logično vrednost 1. Če ni aktiven noben od vhodov, se stanje izhodov ohranja, kar je zagotovljeno s povratno vezavo. Prehajanje med stanji na podlagi obeh aktivnih vhodov ni definirano. Delovanje pomnilne celice RS ponazarja prevajalna funkcija na sliki pod (b).

Dinamika v bioloških sistemih

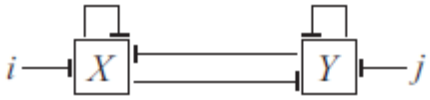
Za postavitev stohastičnega modela preklopnega stikala v bioloških sistemih si najprej poglejmo dinamiko bioloških sistemov. V bioloških sistemih je informacija predstavljena s koncentracijo specifičnega proteina. Koncentracijo proteina merimo v nanomolih na liter(nM). V gensko regulatorskih omrežjih simuliramo koncentracijo proteina s pomočjo procesov degradacije in sinteze.

Sinteza proteina je sestavljena iz dveh procesov, in sicer transkripcije in translacije. V procesu transkripcije se DNA prepiše v sporočilno RNA (mRNA). Protein je sintetiziran v procesu translacije na osnovi zaporedja mRNA. Stopnjo transkripcije upravljamo z transkripcijskimi faktorji, ki se vežejo na specifično DNA zaporedje, kot so na primer promotorji. Vezava transkripcijskih faktorjev z promotorjem lahko aktivira/poveča ali represira/zmanjša stopnjo transkripcije. Transkripcijski faktorji, ki povečajo stopnjo transkripcije, se imenujejo aktivatorji in transkripcijski faktorji, ki zmanjšajo transkripcijsko stopnjo, se imenujejo represorji. Predpostavimo lahko, da je proces translacije izveden brezpogojno. Sinteza proteina je tako odvisna od odsotnosti oziroma prisotnosti transkripcijskih faktorjev. Sintetiziran protein lahko nastopa kot transkripcijski faktor za sintezo nekega drugega proteina ali celo za svojo lastno sintezo. Z spremembami v zaporedju promotorja lahko definiramo kateri protein bo deloval kot transkripcijski faktorji in kako bodo ti vplivali na transkripcijo (aktiviranje ali represiranje). Na tak način zgradimo sistem, ki je konstruiran z različnim DNA zaporedjem, ki medsebojno vpliva drug na drugega. Omrežje takega DNA zaporedja imenujemo gensko regulatorsko omrežje, saj je prav v genih zapisan (kodiran) način kako in kdaj je potrebno neko beljakovino sintetizirati. Primer takega gensko regulatorskega omrežja je omrežje dveh medsebojno represiranih proteinov, kjer prvi protein represira sintezo drugega proteina in obratno.

Degradacija povzroči zmanjšanje koncentracije proteina in je tako nasproten proces sintezi. Stopnjo degradacije upravljamo z zunanjimi vplivi(UV svetloba) in jo predstavimo z določeno količino.

Model preklopnega stikala v biološkem sistemu

Model preklopnega stikala(ang. toggle switch) v biološkem sistemu je prikazan na spodnji sliki:



Model preklopnega stikala v biološkem sistemu

Zgornja slika prikazuje biološko shemo gensko regulatorskega omrežja preklopnega stikala, kjer X in Y predstavljata opazovane proteine, ter i in j predstavljata prisotnost zunanjih vhodov(vzbujanj), ki vplivajo na degradacijo vsakega opazovanega proteina.

V modelu predpostavljamo, da nizka koncentracija proteina predstavlja logično vrednost 0 in visoka koncentracija logično vrednost 1. Prisotnost proteina X in Y lahko interpretiramo kot izhod iz sistema, ki je ekvivalenten signaloma q in q/ izhodu RS-pomnilne celice. Zunanji vpliv(UV svetloba) poveča določeno stopnjo degradacije proteina, zato sta i in j interpretirana kot RESET(R) in SET(S) vhoda RS-pomnilne celice. Tako je model preklopnega stikala biološkega sistema ekvivalenten logičnemu obnašanju RS-pomnilne celice.

Preklopno stikalo (ang. toggle switch) v biološkem sistemu lahko realiziramo kot gensko regulatorsko omrežje dveh medsebojno represiranih proteinov(X in Y). Zato da bi omejili nekontrolirano rast njihovih koncentracij vsak od njih represira še samega sebe(povratna povezava-negative feedback), ampak samo če je njihova koncentracija zelo visoka.

Stabilno stanje sistema je predstavljeno z visoko koncentracijo natanko enega opazovanega proteina. V našem modelu imamo dve možni stabilni stanji, zato imamo opravka z nestabilnim sistemom, ki ga lahko uporabimo kot element pomnjenja. Preklopi med stabilnimi stanji so proženi z zunanjim vhodom podobno kot v RS-pomnilni celici. Za zunanji vhod lahko uporabimo sevanje z UV žarki, ali kateri drugi katalizator, ki poveča degradacijo enega opazovanega proteina. Z povečano degradacijo dosežemo, da koncentracija proteina drastično pade. Represija drugega proteina je ustavljena in stopnja sinteze se poveča. Stanje sistema, ki je definiran z visoko koncentracijo natanko enega opazovanega proteina je zaradi tega preklopljen, če je degradacija proteina z visoko prisotnostjo povečana, ali osvežen, če je degradacija proteina z majhno prisotnostjo povečana. Iz tega sklepamo, da imamo na razpolago dva različna zunanja vpliva(dva različna svetlobna spektra), kjer vsak od njih vpliva samo na enega od opazovanih proteinov.

Postavitev modela preklopnega stikala z orodjem SGN Sim

Stohastični model preklopnega stikala postavimo tako, da na začetku določimo seme za generator naključnih števil (seed), čas trajanja simulacije (time in stop_time), interval med zapisi rezultatov (readout_interval) in ime datoteke kamor izpisujemo rezultate (output_file). Nato ustvarimo lua !{….} blok, kjer vpišemo vrednost konstant, ki se bodo uporabljale pri kemičnih reakcijah. Konstante, ki jih uporabljamo so podane v spodnji tabeli:

Oznaka	Pomen
kx	hitrost sinteze proteina X
ky	hitrost sinteze proteina Y
kdx	hitrost degradacije proteina X
kdy	hitrost degradacije proteina Y
kdx_f	hitra degradacija proteina X
kdy_f	hitra degradacija proteina Y
k1r	hitrost dimerizacije proteina X
k1c	hitrost dimerizacije proteina Y
k_1r	cepitev dimera X2 na 2 X-a
k_1c	cepitev dimera X2 na 2 Y-a
kdr	degradacija dimera X2
kdc	degradacija dimera Y2
kf	hitrost vezave proteina na drug DNK
kb	cepitev proteina iz nasprotnega DNK
kfs	hitrost vezave proteina na svoj DNK
kbs	cepitev proteina iz svojega DNK

V blok population {…} vpišemo začetne vrednosti kemičnih zvrsti, ki nastopajo v kemičnih reakcijah. Vse kemične zvrsti, ki jih uporabljamo so podane v spodnji tabeli:

Kemična zvrst	Vloga v sistemu
DNK_X	promotor DNK kodiranega proteina X
DNK_Y	promotor DNK kodiranega proteina Y
X	protein X
Y	protein Y
X_2	dimer proteina X
Y_2	dimer proteina Y
X_2DNK_X	DNK_X je vezan z proteinom dimera X_2
Y_2DNK_Y	DNK_Y je vezan z proteinom dimera Y_2
Y_2DNK_X	DNK_X je vezan z proteinom dimera Y_2
X_2DNK_Y	DNK_Y je vezan z proteinom dimera X_2
I	dodatna kemična spojina za proženje preklopa

V blok reaction {…} vpišemo vse kemične reakcije, ki vplivajo na dinamiko našega modela. Kemične reakcije so podane v spodnji tabeli:

Kemična reakcija	Opis
$DNK_X \xrightarrow{kx} X + DNK_X$	sinteza proteina X
$DNK_Y \xrightarrow{ky} Y + DNK_Y$	sinteza proteina Y
$X + X \xrightarrow{k1r} X_2$	dimerizacija proteina X
$Y + Y \xrightarrow{k1c} Y_2$	dimerizacija proteina Y
$X_2 \xrightarrow{k\_1r} X + X$	monomerizacija dimera X_2
$Y_2 \xrightarrow{k\_1c} Y + Y$	monomerizacija dimera Y_2
$X_2 + DNK_X \xrightarrow{kfs} X_2DNK_X$	samo-represija X
$Y_2 + DNK_Y \xrightarrow{kfs} Y_2DNK_Y$	samo-represija Y
$Y_2 + DNK_X \xrightarrow{kf} Y_2DNK_X$	križna-represija X
$X_2 + DNK_Y \xrightarrow{kf} X_2DNK_Y$	samo-represija Y
$X \xrightarrow{kdx} Ø$	degradacija proteina X
$Y \xrightarrow{kdy} Ø$	degradacija proteina Y
$X_2 \xrightarrow{kdr} Ø$	degradacija dimera X_2
$Y_2 \xrightarrow{kdc} Ø$	degradacija dimera Y_2
$X_2DNK_X \xrightarrow{kdr} DNK_X$	degradacija vezanega dimera X_2
$X_2DNK_Y \xrightarrow{kdr} DNK_Y$	degradacija vezanega dimera X_2
$Y_2DNK_X \xrightarrow{\boldsymbol{kdc}} DNK_X$	degradacija vezanega dimera Y_2
$Y_2DNK_Y \xrightarrow{\boldsymbol{kdc}} DNK_Y$	degradacija vezanega dimera Y_2
$X_2DNK_X \xrightarrow{\boldsymbol{kbs}} X_2 + DNK_X$	cepitev dimera X_2 iz svojega DNK
$Y_2DNK_Y \xrightarrow{\boldsymbol{kbs}} Y_2 + DNK_Y$	cepitev dimera Y_2 iz svojega DNK
$X_2DNK_Y \xrightarrow{\boldsymbol{kb}} X_2 + DNK_Y$	cepitev dimera X_2 iz nasprotnega DNK
$Y_2DNK_X \xrightarrow{\boldsymbol{kb}} Y_2 + DNK_X$	cepitev dimera Y_2 iz nasprotnega DNK
$X + I \xrightarrow{\boldsymbol{kdx\_f}} Ø$	hitra degradacija proteina X vezanega z dodatno spojino I

Na koncu ustvarimo čakalno vrsto(queue [ ]I( )), s katero dosežemo da se sprosti določena spojina po nekem času od začetka simulacije. Tako dobimo preklop po času, ki smo ga določili v čakalni vrsti, saj se takrat sprosti spojina.

Primer našega modela preklopnega stikala postavljenega z orodjem SGNSim:

```
seed;
time 0;
stop_time 5000;
readout_interval 1;
output_file toggleswitch2.xls;
save_interval;


lua !{
   // parametri za x
   kx      = 1.1;          // hitrost sinteze proteina X
   kdx     = 0.0007;       // hitrost degradacije proteina X
   k1r     = 0.09;         // hitrost dimerizacije proteina X
   k_1r    = 0.5;          // cepitev dimera X2 na 2 X-a
   kdr     = 0.0007;       // degradacija dimera X2

   // parametri za y
   ky      = 1.1;          // hitrost sinteze proteina Y
   kdy     = 0.0007;       // hitrost degradacije proteina Y
   k1c     = 0.09;         // hitrost dimerizacije proteina Y
   k_1c    = 0.5;          // cepitev dimera Y2 na 2 Y-a
   kdc     = 0.0007;       // degradacija dimera Y2

   // skupni parametri
   kf      = 1;            // hitrost vezave proteina na drug DNK
   kb      = 0.5;          // cepitev proteina iz nasprotnega DNK
   kfs     = 0.1;          // hitrost vezave proteina na svoj DNK
   kbs     = 0.5;          // cepitev proteina iz svojega DNK

   // hitra degradacija za x
   kdx_f   = 1;            // hitra degradacija monomera
   kdr_f   = 1;            // hitra degradacija dimera

   // hitra degradacija za y
   kdy_f   = 1;            // hitra degradacija monomera
   kdc_f   = 1;            // hitra degradacija dimera

}!
```

```
population {
	DNK_X=1;
	DNK_Y=0;
	X=112;
	Y=28;
	X_2=558;
	Y_2=35;
	X_2.DNK_Y=199;
	Y_2.DNK_X=75;
	X_2.DNK_X=124;
	Y_2.DNK_Y=1;
	I = 10;
}

reaction {
	DNK_X --[kx]--> X + DNK_X;
	DNK_Y --[ky]--> Y + DNK_Y;
	X+X --[k1r]--> X_2;
	Y+Y --[k1c]--> Y_2;
	X_2 --[k_1r]--> X+X;
	Y_2 --[k_1c]--> Y+Y;
	X_2+DNK_X --[kfs]--> X_2.DNK_X;
	Y_2+DNK_Y --[kfs]--> Y_2.DNK_Y;
	Y_2+DNK_X --[kf]--> Y_2.DNK_X;
	X_2+DNK_Y --[kf]--> X_2.DNK_Y;
	X --[kdx]--> ;
	Y --[kdy]--> ;
	X_2 --[kdr]--> ;
	Y_2 --[kdc]--> ;
	X_2.DNK_X --[kdr]--> DNK_X;
	X_2.DNK_Y --[kdr]--> DNK_Y;
	Y_2.DNK_X --[kdc]--> DNK_X;
	Y_2.DNK_Y --[kdc]--> DNK_Y;
	X_2.DNK_X --[kbs]--> X_2 + DNK_X;
	Y_2.DNK_Y --[kbs]--> Y_2 + DNK_Y;
	X_2.DNK_Y --[kb]--> X_2 + DNK_Y;
	Y_2.DNK_X --[kb]--> Y_2 + DNK_X;
	X+I --[kdx_f]--> ;

}
queue [2000]I(2000);
```

4. REZULTATI SIMULACIJE

Testiranja smo izvedli za tri različne parametre, ki so podani v tabeli 4.1 za dve različni populaciji. Parametri populacije **'A'** in **'B'** so podani v tabeli 4.2. Časovno smo se omejili na 5000 sekund, pri čemer sprožimo preklop po 2000 sekundah. To storimo tako, da ob času 2000 sekund dodamo dodatno spojino za hitro degradacijo. Čas dovajanja te spojine mora biti enak času,ki je potreben, da koncentracija dimera pade na nič. Ta čas je enak času padca(fall time). Za testiranje šuma in ustaljenih mej, pa smo se časovno omejili na 500000 sekund. Čas preklopa je ostal enak.

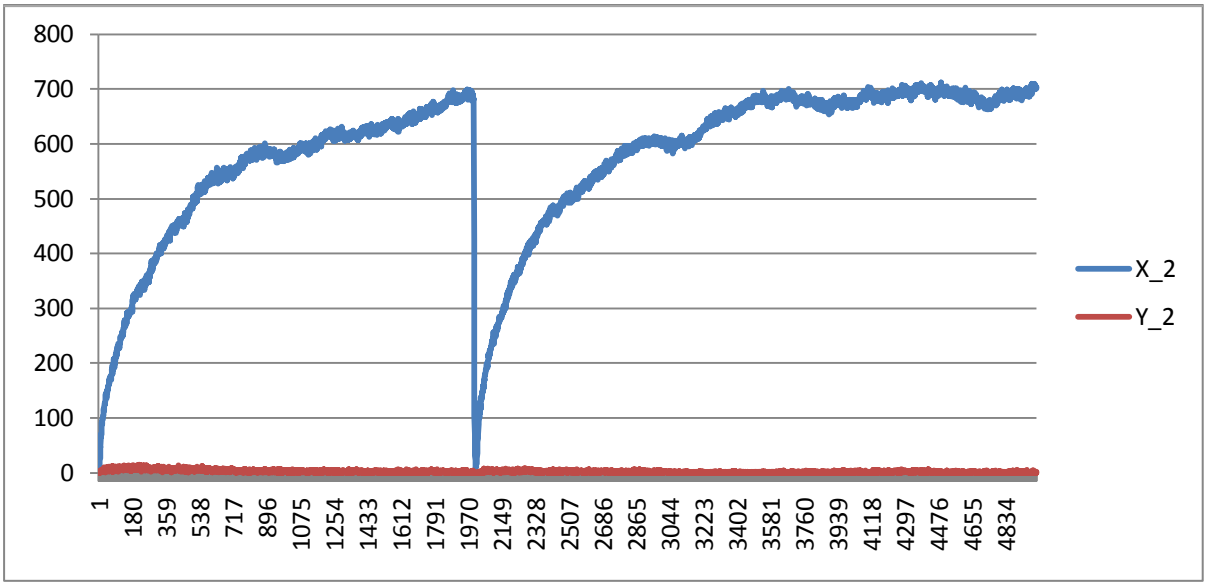
Oznaka	pomen	vrednost 1	vrednost 2	vrednost 3
kx	hitrost sinteze proteina X	1.1 nM/s	1.1nM/s	0.5nM/s
ky	hitrost sinteze proteina Y	0.5 nM/s	1.1nM/s	0.5nM/s
kdx	hitrost degradacije proteina X	0.0007 s^{-1}	0.0007 s^{-1}	0.0025 s^{-1}
kdy	hitrost degradacije proteina Y	0.0025 s^{-1}	0.0007 s^{-1}	0.0025 s^{-1}
kdxf	hitra degradacija proteina X	1.0 s^{-1}	1.0 s^{-1}	1.0 s^{-1}
kdyf	hitra degradacija proteina Y	1.0 s^{-1}	1.0 s^{-1}	1.0 s^{-1}
k1r	hitrost dimerizacije proteina X	0.09 $(nMs)^{-1}$	0.09 $(nMs)^{-1}$	0.0015 $(nMs)^{-1}$
k1c	hitrost dimerizacije proteina Y	0.0015 $(nMs)^{-1}$	0.09 $(nMs)^{-1}$	0.0015 $(nMs)^{-1}$
k-1r	cepitev dimera X2 na 2 X-a	0.5 s^{-1}	0.5s^-1	0.5s^-1
k-1c	cepitev dimera X2 na 2 Y-a	0.5 s^{-1}	0.5s^-1	0.5s^-1
kdr	degradacija dimera X2	0.0007 s^{-1}	0.0007 s^{-1}	0.0025 s^{-1}
kdc	degradacija dimera Y2	0.0025 s^{-1}	0.0007 s^{-1}	0.0025 s^{-1}
kf	hitrost vezave proteina na drug DNK	1.0 $(nMs)^{-1}$	1.0 $(nMs)^{-1}$	1.0 $(nMs)^{-1}$
kb	cepitev proteina iz nasprotnega DNK	0.5 s^{-1}	0.5 s^{-1}	0.5 s^{-1}
kfs	hitrost vezave proteina na svoj DNK	0.1*kf	0.1*kf	0.1*kf
kbs	cepitev proteina iz svojega DNK	kb	kb	kb

Tabela 4.1 Tri različne vrednosti parametrov pri simuliranju preklopnega modela

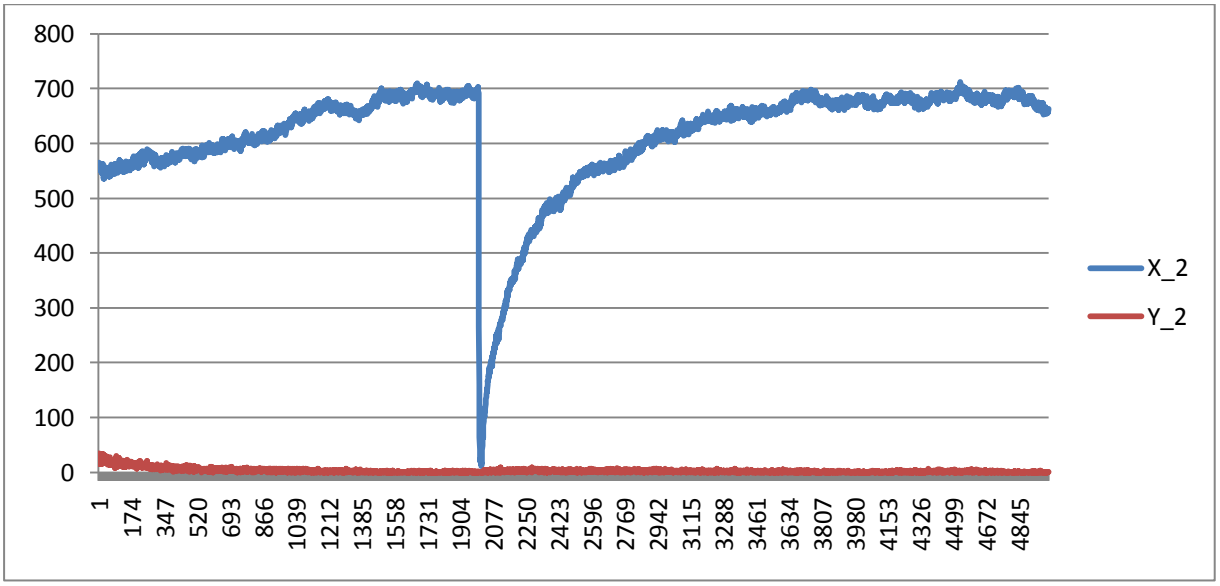
	'A'	*'B'*
DNK_X	200	1
DNK_Y	200	0
X	0	112
Y	0	28
X_2	0	558
Y_2	0	35
X_2.DNK_Y	0	199
Y_2.DNK_X	0	75
X_2.DNK_X	0	124
Y_2.DNK_Y	0	1

Tabela 4.2 Začetni populaciji pri zagonu simulacij

4.1 Rezultati testiranj parametrov 'vrednost1'

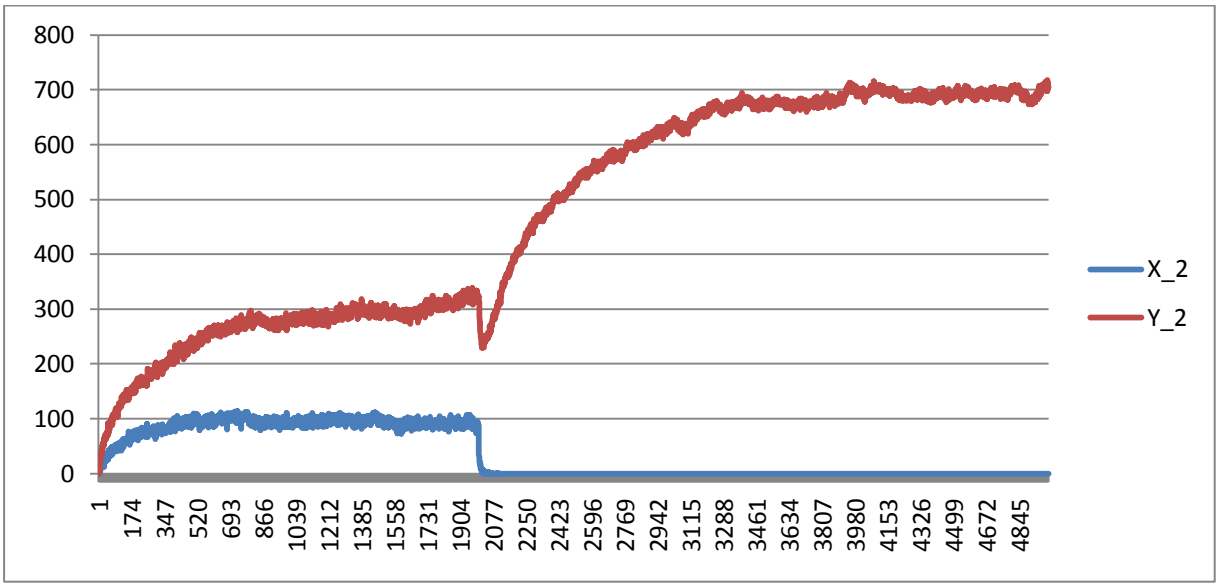


Graf 1 Testiranje pri populaciji **A**. Prikazuje nam količino proteinov dimera $\boldsymbol{X_2}$ in dimera $\boldsymbol{Y_2}$. Vidimo, da kljub preklopu dimera $\boldsymbol{X_2}$ v času 2000 sekund, dimer $\boldsymbol{Y_2}$ ne reagira. Tako do preklopa ne pride in sistem ni ustrezen.
Rezultati teh testiranj so v datoteki: 'parametri1-a.xlsx'

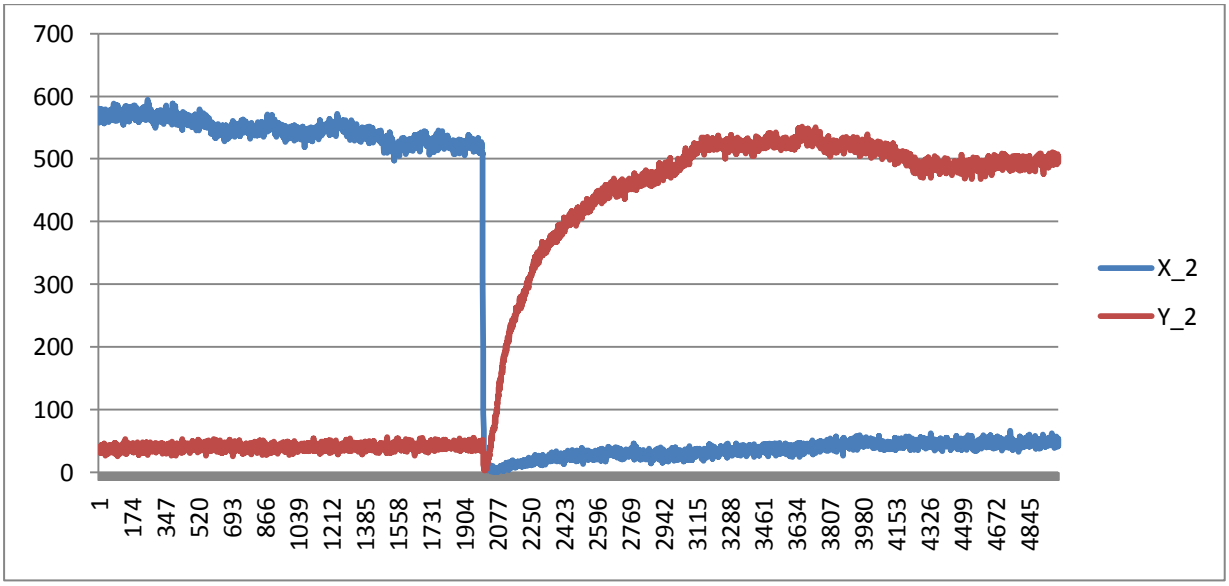


Graf 2 Testiranje pri populaciji **B**. Prikazuje nam količino proteinov dimera $\boldsymbol{X_2}$ in dimera $\boldsymbol{Y_2}$. Vidimo, da kljub preklopu dimera $\boldsymbol{X_2}$ v času 2000 sekund, dimer $\boldsymbol{Y_2}$ ne reagira. Tako do preklopa ne pride in sistem zopet ni ustrezen.
Rezultati teh testiranj so v datoteki: 'parametri1-b.xlsx'

4.2 Rezultati testiranj parametrov 'vrednost2'

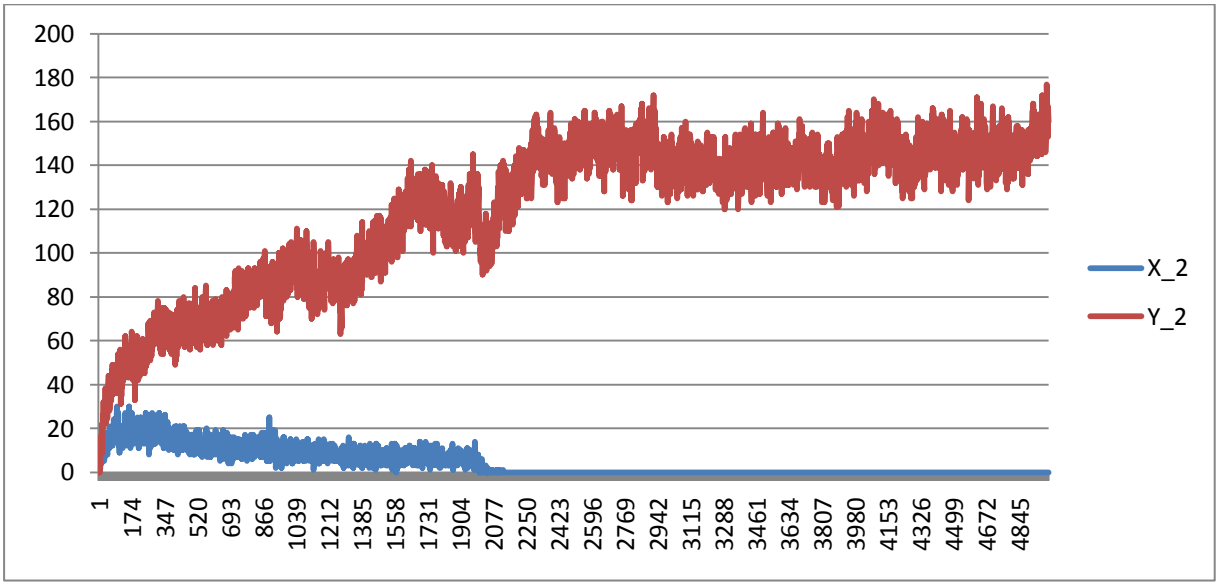


Graf 3 Testiranje pri populaciji **A**. Prikazuje nam količino proteinov dimera X_2 in dimera Y_2. Dimer Y_2 nam tu hitreje narašča kot dimer X_2. V času 2000 sekund vidimo da začneta z degredacijo oba dimera, nato pa začne glede na nizko količino dimera X_2 dimer Y_2 strmo naraščati. Do preklopa ne pride ker je pred preklopom količina obeh dimerov napačna in sistem zato ni ustrezen.
Rezultati teh testiranj so v datoteki: 'parametri2-a.xlsx'

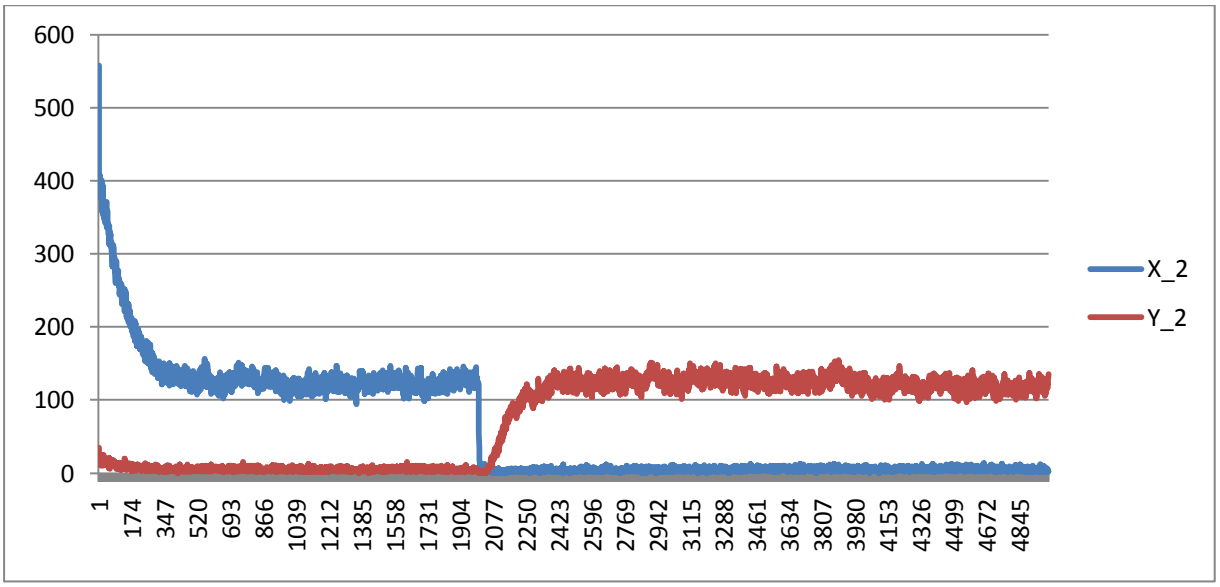


Graf 4 Testiranje pri populaciji **B**. Prikazuje nam količino proteinov dimera X_2 in dimera Y_2. V tem primeru imamo prvič pravi preklop obeh količin dimerov. Točno ob preklopu se poleg degradacije dimera X_2 sicer zgodi tudi degredacija dimera Y_2 , vendar se takoj zatem začne večati količina dimera Y_2 glede na nizko količino dimera X_2 kar tudi pričakujemo. Tu je preklop uspešen.
Rezultati teh testiranj so v datoteki: 'parametri2-b.xlsx'

4.3 Rezultati testiranj parametrov 'vrednost3'

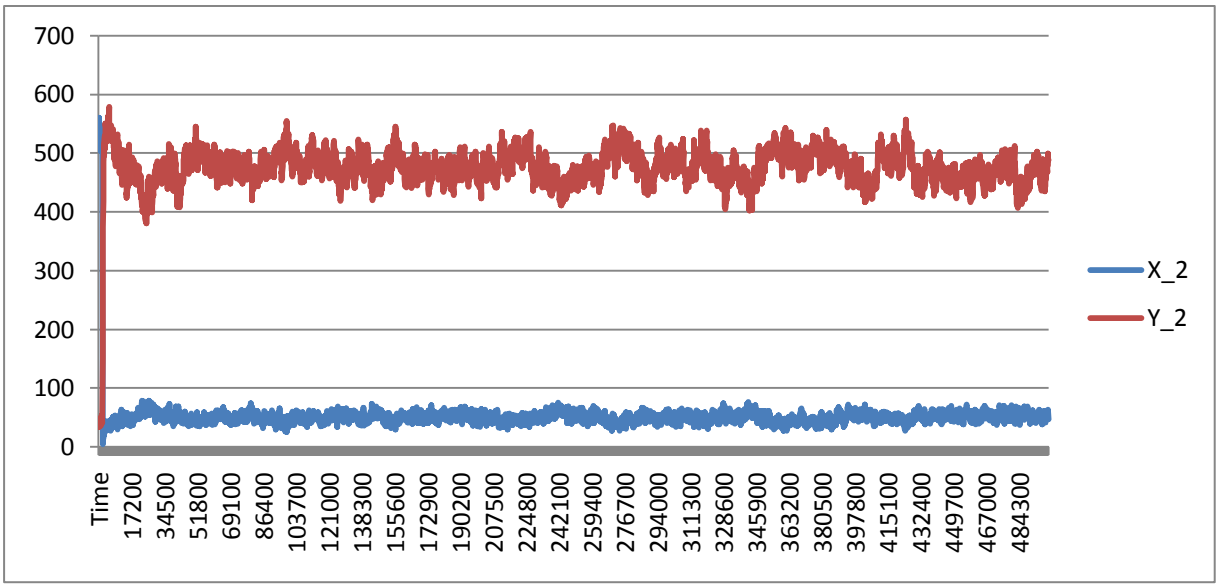


Graf 5 Testiranje pri populaciji **A**. Prikazuje nam količino proteinov dimera X_2 in dimera Y_2. Kot smo opazili pri primeru s parametri 'vrednost2', populacija **A**, je tudi tukaj dimer Y_2 količinsko narastel veliko več kot dimer X_2. Zaradi tega po času 2000 sekund ne pride do preklopa.
Rezultati teh testiranj so v datoteki: 'parametri3-a.xlsx'

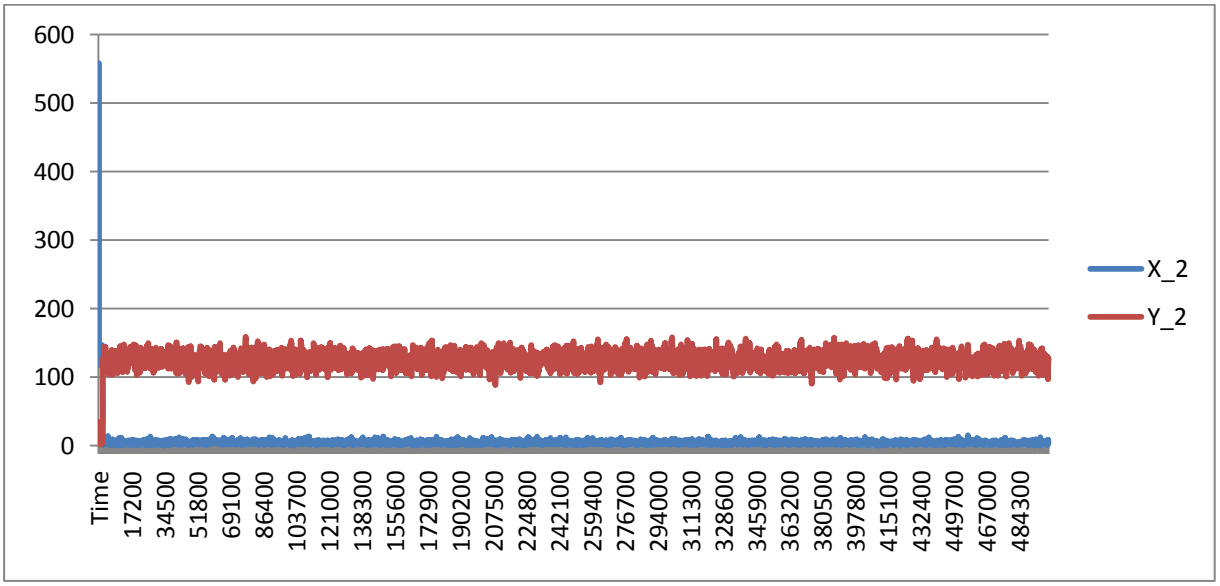


Graf 6 Testiranje pri populaciji **B**. Prikazuje nam količino proteinov dimera X_2 in dimera Y_2. Najprej opazimo da se nam je količina dimera X_2 po 280 sekundah spustila na mejo med 100 in 150, ter se tam tudi ustali, dokler ne pridemo do preklopa pri času 2000 sekund. Po preklopu se dimer X_2 postavi v nizko stanje, dimer Y_2 pa se postavi v visoko stanje z ustaljenim razponom med tistimi vrednostmi pri katerih je bil pred preklopom dimer X_2.
Rezultati teh testiranj so v datoteki: 'parametri3-b.xlsx'

4.4 Rezultati testiranj šuma parametrov 'vrednost2' in 'vrednost3'(uspešnih sistemov)



Graf 7 Testiranje pri populaciji **B**, za parametre'vrednost2' Prikazuje nam količino proteinov dimera $\boldsymbol{X_2}$ in dimera $\boldsymbol{Y_2}$. V tem primeru gre za to da računamo ali je osveževanje potrebno in kolikšen je razpon šuma. Opazimo da je je razpon šuma dimera $\boldsymbol{Y_2}$ že na pogled od 2-3 krat večji kot šum dimera $\boldsymbol{X_2}$. Podrobnejši izračun je v razdelku 4.5.
Rezultati teh testiranj so v datoteki: 'parametri2-b-dolg.xlsx'



Graf 8 Testiranje pri populaciji **B**, za parametre'vrednost3'. Prikazuje nam količino proteinov dimera $\boldsymbol{X_2}$ in dimera $\boldsymbol{Y_2}$. Opazimo da nam količina dimera $\boldsymbol{Y_2}$ ne preseže vrednosti 200. Vprimerjavi z grafom 7 tudi vidimo da je šum veliko bolj enakomeren in manjši. Vendar pa je tu šum dimera $\boldsymbol{Y_2}$ od 3-4 krat večji kot je šum dimera $\boldsymbol{X_2}$, kar pa nas ne moti. Dejansko je bolje imeti bolj enakomeren in majši šum, kot pa imeti manjši količnik med šumoma dimerov $\boldsymbol{X_2}$ $\boldsymbol{in}$ $\boldsymbol{Y_2}$.
Rezultati teh testiranj so v datoteki: 'parametri3-b-dolg.xlsx'

4.5 Metrike, ki smo jih uporabljali pri opazovanju bioloških sistemov

1. Odločitev, katero spojino bomo izbirali kot nosilca informacij na podlagi izmerjenega šuma

Kot nosilca informacij smo izbrali dimera X_2 in Y_2, saj smo iz opazovanj grafov že takoj opazili, da imata najmanjši šum, čeprav bi podobne rezultate dobili tudi z opazovanjem koncentracije X_2DNK_x in Y_2DNK_y.

2. Ocenjevanje šuma

Šum smo ocenjevali tako, da smo po preklopu poiskali minimalno in maksimalno koncentracijo, ki sta jo dosegla dimera X_2 in Y_2. Razlika med maksimalno in minimalno koncentracijo nam je podala velikost šuma:

$$šum(X_2) = \max(X_2) - \min(X_2)$$
$$šum(Y_2) = \max(Y_2) - \min(Y_2)$$

Pri tem velja še omeniti, da so ocene šuma na nizkem nivoju(logična 0) za dimer X_2 in na visokem nivoju(logična 1) za dimer Y_2.

3. Ocenjevanje potrebnosti osveževanja

Ko naš sistem naredi preklop, se koncentraciji X_2 in Y_2 gibljejo v območju šuma in ohranjajo vrednost, tako da sistema ni potrebno osveževati.

4. Določanje mej logične 0 in 1

Mejo za logično 0 smo določili tako, da smo poiskali maksimalno vrednost koncentracije dimera, ki je bil v nizkem stanju(v naših primerih dimer X_2) in tej vrednosti prišteli še 5nM.

$$meja_0 = \max(X_{2nizki\ nivo}) + 5$$

Obratno smo poiskali mejo za logično 1 tako, da smo poiskali minimalno vrednost koncentracije dimera v visokem stanju(v naših primerih Y_2)in odšteli 5nM.

$$meja_1 = \min(Y_{2visoki\ nivo}) - 5$$

5. Ocenjevanje časa vzpona (rise time)

Čas vzpona smo določili tako, da smo izračunali čas od prožitve preklopa (t=2000s za naš primer) do časa, ko je koncentracija dimera Y_2 dvignila nad izračunano mejo za logično 1 in ni več padla pod njo.

6. Ocenjevanje časa padca (fall time)

Čas padca smo določili tako, da smo izračunali čas od prožitve preklopa (t=2000s za naš primer) do časa, ko je koncentracija dimera X_2 padla pod izračunano mejo za logično 0 in ni več zrasla nad njo.

7. Ocenjevanje časa preklopa

Čas preklopa smo določili tako, da smo izbrali maksimalno vrednost med časom vzpona in časom padca:

$$čas\ preklopa = \max(t_r, t_f)$$

4.6 Dobljene metrike pri testiranju uspešnih sistemov

1. Dobljene metrike pri uporabi parametrov 'vrednost2' in populaciji b:

- Šum:
  - nizki nivo:
    $$šum(X_2) = \max(X_2) - \min(X_2) = 79 - 5 = 74nM$$
  - visoki nivo:
    $$šum(Y_2) = \max(Y_2) - \min(Y_2) = 579 - 380 = 199nM$$

- Meje:
  - Logična 0:
    $$meja_0 = \max(X_{2nizki\ nivo}) + 5 = 79 + 5 = 84nM$$
  - Logična 1:
    $$meja_1 = \min(Y_{2visoki\ nivo}) - 5 = 380 - 5 = 375nM$$

- Čas vzpona:
  $$t_r = t_{prehod\_zgornje\_meje} - t_{dodatne\_spojine} = 2371 - 2000 = 371s$$

  Mi smo izvedli več testiranj istega sistema in smo dobili povprečen čas vzpona 368s.

- Čas padca:
  $$t_r = t_{prehod\_spodnje\_meje} - t_{dodatne\_spojine} = 2007 - 2000 = 7s$$

  Mi smo izvedli več testiranj istega sistema in smo dobili povprečen čas padca 7s.

- Čas preklopa:
  $$čas\ preklopa = \max(t_r, t_f) = \max(368,7) = 368s$$

2. **Dobljene metrike pri uporabi parametrov 'vrednost3' in populaciji b:**

- Šum:
  - nizki nivo:
    $$šum(X_2) = \max(X_2) - \min(X_2) = 15 - 0 = 15nM$$
  - visoki nivo:
    $$šum(Y_2) = \max(Y_2) - \min(Y_2) = 159 - 89 = 70nM$$

- Meje:
  - Logična 0:
    $$meja_0 = \max(X_{2nizki\ nivo}) + 5 = 15 + 5 = 20nM$$
  - Logična 1:
    $$meja_1 = \min(Y_{2visoki\ nivo}) - 5 = 89 - 5 = 84nM$$

- Čas vzpona:
  $$t_r = t_{prehod\_zgornje\_meje} - t_{dodatne\_spojine} = 2206 - 2000 = 206s$$

  Mi smo izvedli več testiranj istega sistema in smo dobili povprečen čas vzpona 194s.

- Čas padca:
  $$t_r = t_{prehod\_spodnje\_meje} - t_{dodatne\_spojine} = 2007 - 2000 = 7s$$

  Mi smo izvedli več testiranj istega sistema in smo dobili povprečen čas padca 7s.

- Čas preklopa:
  $$čas\ preklopa = \max(t_r, t_f) = \max(194,7) = 194s$$

4.7 Komentar k rezultatom testiranj

Tekom testiranj smo ugotovili, da postaviti uspešen biološki sistem ni enostavno, saj morajo biti vsi parametri podani zelo natančno in tudi začetna populacija mora biti ustrezna. Tako smo s testiranjem pokazali kar nekaj neuspešnih sistemov in le dva uspešna.

Če primerjamo oba uspešna sistema vidimo, da je sistem s parametri 3 boljši, saj dobimo manjši čas preklopa, pa tudi šum je veliko manjši. Za potrebe preklopnega stikala bi bila oba uspešna sistema primerna.

5. ZAKLJUČEK

Reči skupini štirim študentom računalništva, da se bodo ukvarjali z njim nepoznanim področjem se nam je že sprva zdel zanimiv izziv, ki smo ga morali sprejeti.
Takoj na začetku smo se spopadli z osnovnimi pojmi kot so protein, spojina, dimer… Pojmi, ki povečini niso bili v našem besednem zakladu. Tako smo dobili veliko člankov od asistenta Miha Moškona, ki nam je bil tekom izdelave seminarja v veliko pomoč, ter se pridno lotili prebiranja in spoznavanja z nam neznanim področjem. Ko smo prebrodili začetne težave, smo se lotili rokovanja z orodjem SGN Sim. Ugotovili smo, da je rokovanje z njim dokaj enostavno, in hitro smo imeli postavljen začetni sistem, ki pa ni deloval, saj sta nam manjkali dve enačbi reakcij. Ko smo odpravili še te težave, smo se lotili testiranj in se lotili pisanje seminarske naloge, ki je pred vami.
Na koncu smo se vsi strinjali, da imajo biološki sistemi v prihodnosti velik potencial, vendar pa če primerjamo čase preklopa elektronskih stikal in stikala v biološkem sistemu vidimo, da nas čaka še dolga pot do postavitve biološkega sistema, ki bi nadomestil elektronskega.

6. LITERATURA:

- [1]Miha Moškon and Miha Mraz, "Analysing the information processing capabilities of biological systems," to be published in Mathematical and Computer Modeling.
- [2]Timothy S. Gardner, Charles R. Cantor, and James J. Collins, "Construction of a genetic toggle switch in Escherichia coli, *Nature*, vol. 403, pp. 339-342, 2000.
- [3]Andre S. Ribeiro and Jason Lloyd-Price, "SGN Sim, a Stochastic Genetic Networks
- Simulator," *Bioinformatics*, vol. 23, pp. 777--779, 2007.
- http://www.cs.tut.fi/~sanchesr/SGN/SGNSim.html