Univerza v Ljubljani

Fakulteta za računalništvo in informatiko
Ljubljana, Tržaška 25

2.seminarska naloga

Analiza orodja Kappa (model preklopnega stikala)

Sintezna biologija
[6.skupina]

OPTIČNE IN NANOTEHNOLOGIJE

Avtorji:

Borut Ajdič, Marko Gavrilovič,

Tine Ileršič, Simeon Puntar

Mentorja:

Miha Moškon, Miha Mraz

Ljubljana, Januar 2011

Kazalo

Uvod

Seminarska naloga je namenjena nam študentom, da se spoznamo z osnovami sintezne biologije. Marsikdo s tem pojmom sploh ni seznanjen in ne najde povezave z inženirskim delom. »Nova veda, ki se ukvarja z načrtovanjem in izdelavo novih bioloških delov, naprav in sistemov«. «Veda, ki raziskuje in odkriva kako preurejati naravne biološke sisteme v uporabne namene«. Poudarjeno je torej inženirsko delo, ki je verjetno najpomembnejša razlika v primerjavi z drugimi podobnimi vedami. Računalništvo tukaj nastopi kot veda, ki zna obvladovati dovolj abstraktno načrtovanje kompleksnih sistemov in ni obremenjena s podrobnim poznavanjem osnovnih gradnikov – genov. Posledica so konstrukti (logična vrata, celični oscilatorji, genska stikala,...), ki močno poenostavijo delo, kar vodi do večje učinkovitosti, ta pa do zahtevnejših rezultatov.

Mi smo se tokrat osredotočili na delovanje programske opreme Kappa, namenjene postavitvi in simulacijam modelov, ki izvirajo iz sintezne biologije. Poskušali smo relaizirati model preklopnega stikala(eng. Toggle switch).

Priprava programskega okolja

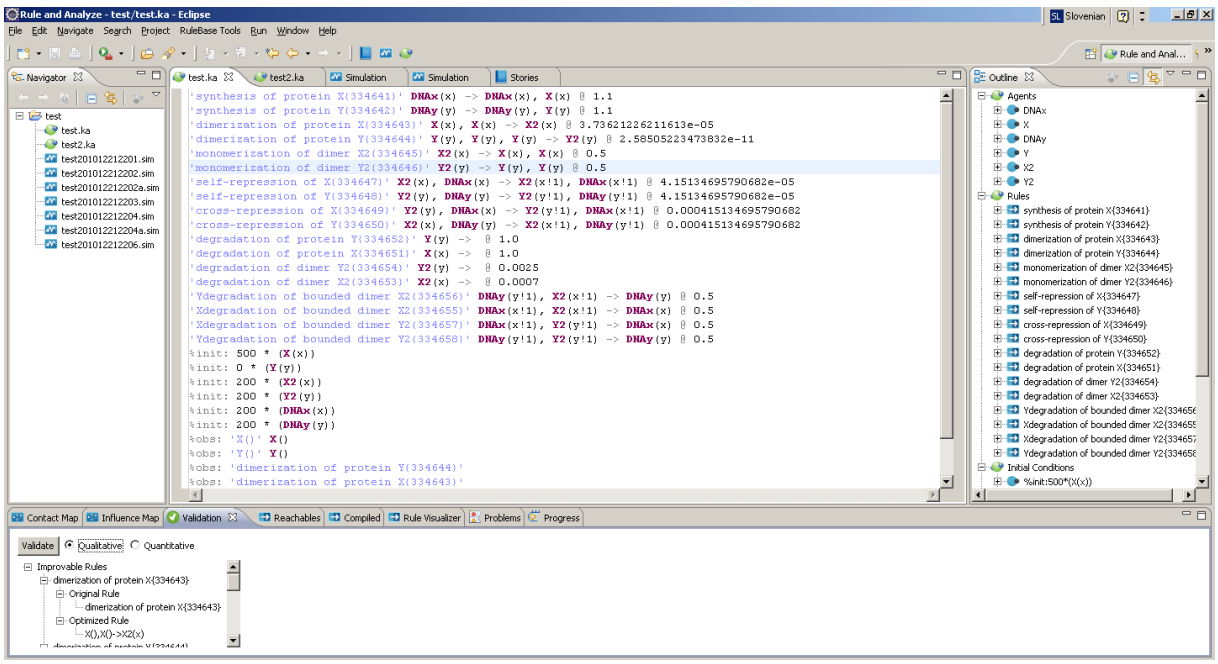
Programski jezik Kappo lahko uporabljamo na več načinov. Na voljo imamo po težavnosti razporejene, konzolno različico, dodatek za orodje Eclipse ali spletni portal. Ker je bil naš osnovni problem bolj posvečen postavitvi modela, se konzolni različici nismo posvetili. Za delo sta uporabniku bolj prijazni ostali dve.

Prednost dela v Eclipsu je zagotovo obarvanje sintakse in hitrejše delovanje. Dodatek nam omogoča tudi inteligentno podajanje ukazov, da je delo še hitrejše. Vse objekte, ki nastopajo v modelu, nam prikaže v pregledni drevesni strukturi. Omogoča validacijo pravilnosti sintakse. Preko menija pa lahko dostopamo tudi do ostalih priročnih orodij. Slabost takega načina je, da mora uporabnik biti seznanjem s sintakso Kappe in njenimi objekti. Moramo tudi imeti lokalno postavljeno okolje z orodjem Eclipse.

Spletni portal RuleBase pa nam omogoča vizualni pregled našega modela. Ena od prednosti je zagotovo internetni dostop. Tako je naše okolje vedno na voljo za delo preko brskalnika. Na voljo so nam tudi vsa grafično obarvana orodja, ki jih Kappa omogoča. Tako je delo uporabniku bolj prijazno, ni potrebno vedeti nič o Kappa sintaksi, ukvarja se lahko zgolj s problemom. Portal je sicer še v beta fazi in vsebuje veliko hroščev, zato je delo počasnejše – če dodamo temu še slabšo internetno povezavo zna biti delo kmalu prenaporno.

RuleStudio(Kappa) - Eclipse

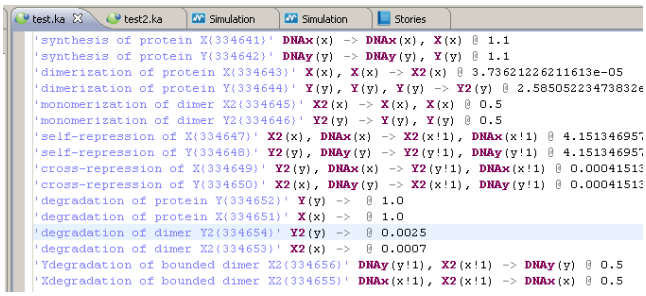
Delo z jezikom Kappa je lahko zelo poenostavljeno, če se odločimo pisati pravila v orodju Eclipse[1]. Obstaja dodatek za razvojno orodje Eclipse in je brezplačno dostopen na spletu[2]. Dodatek omogoča brezplačno in odprto kodno namizno okolje za delo s Kappa modeli.



Slika 1: Eclipse vmesnik za delo s Kappo

Dodatek vključuje naslednje funkcionalnosti:

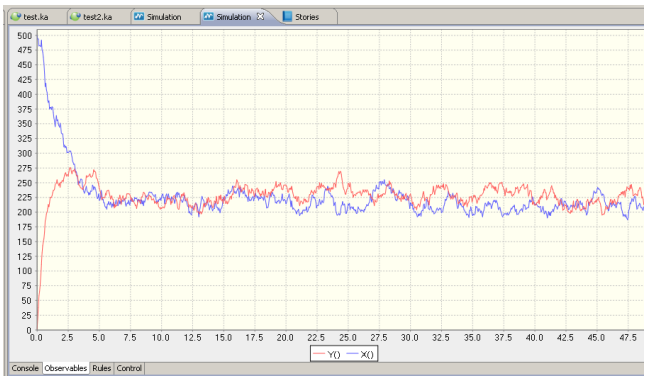
- Obarvanje Kappa sintakse



Slika 2: Obarvanje sintakse v Eclipsu

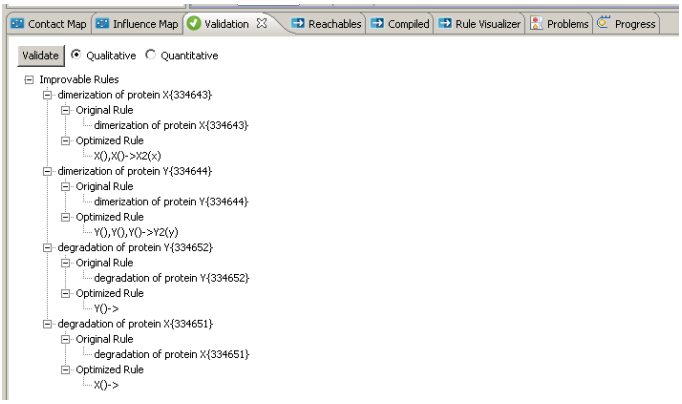
- Zagon simulacij in statičnih analiz

[1]http://www.eclipse.org/downloads/packages/release/galileo/r
[2]http://kappamodeler.github.com/rulestudio/.



Slika 3: Grafičen prikaz simulacij v Eclipsu

- Vgrajena vizualizacijska orodja za: rezultate simulacij, pravila, povezovalne mape, in mape medsebojnega vpliva pravil...
- Vgrajena validacija pravil



Slika 4: Validacija pravil v Eclipsu

- Možnost izvoza modelov direktno v RuleBase (več v nadaljevanju)

Instalacija je preprosta. Najprej potrebujemo instalacijo Eclipse orodja. Za instalacijo RuleStudia, ki vsebuje vsa potrebna orodja pa sledimo naslednjim korakom:

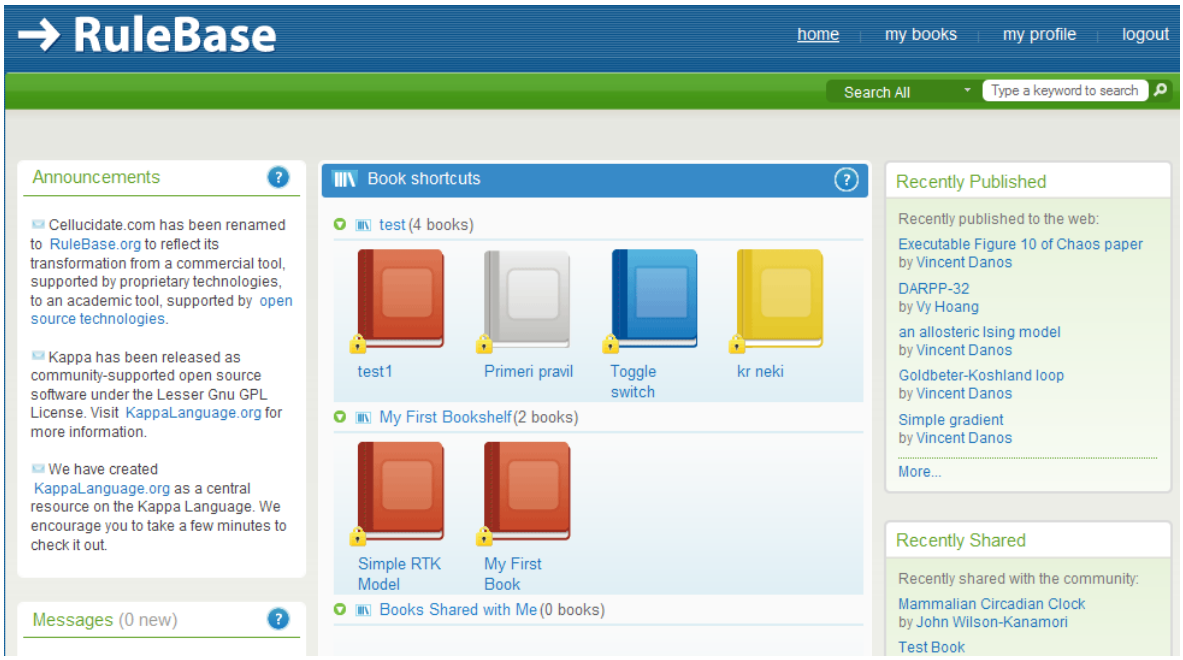
1. Zaženi Eclipse in pojdi na 'Help' -> 'Install New Software...'
2. Klikni 'Add...' za dodajanje nove strani.
3. Stran poimenuj npr. RuleStudio (dejansko ime ni pomembno).
4. Vpiši 'http://eclipse.rulebase.org" za lokacijo in pritisni OK.
5. Odkljukaj 'Group items by category', če je to potrebno.
6. Obkljukaj 'RuleStudio Update Site'.
7. Klikni Next.
8. Prikazati se morata 2 dodatka za instalacijo, klikni Next.
9. Sprejmi nove dodatke in ponovno zaženi Eclipse.

Da ustvarimo novo Kappa datoteko, najprej kreiramo novo mapo za model (Model Folder). Vanjo preko menija dodamo Kappa datoteko (Kappa File). Vse potrebne komponente modela določimo s

Kappa sintakso. Kappa objekti, ki jih deklariramo preko kode se nam tudi drevesno prikažejo. Ob pisanju nam je na voljo tudi priročen inteligentni izbirnik kode.

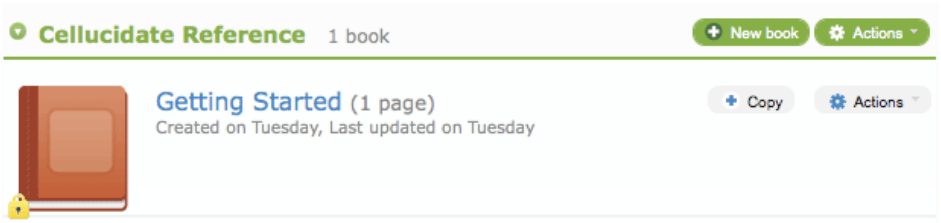
RuleBase

RuleBase smo nakratko omenili že v prejšnjem razdelku. Je spletni portal, v katerega se brezplačno registriramo. Portal je zasnovan kot grafično razvojno okolje v katerem upravljamo z jezikom Kappa. Da bi naše delo potekalo kar se da hitro, smo se tudi mi poslužili tega portala. V nadaljevanju bomo na kratko opisali, kako si okolje pripravimo za delo.



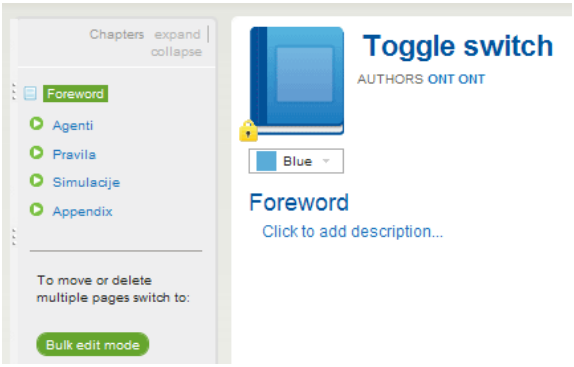
Slika 5: Izgled vmesnika RuleBase in seznam že kreiranih knjig

Celotno delo, ki ga opravimo, se shranjuje v obliki knjige. Na Slika 5 je prikazana knjižna polica z že obstoječimi knjigami. Prvi korak pri delu je kreiranje nove knjige s klikom na »New book«.



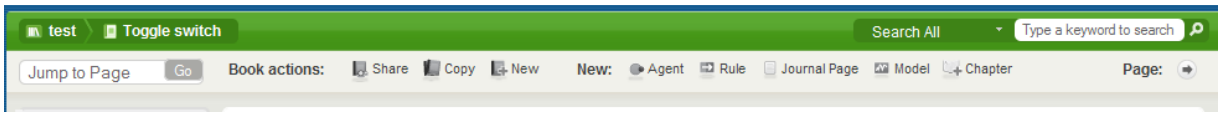
Slika 6: Vmesnik za kreiranje nove knjige in dodajanje akcij

Ko ustvarimo novo knjigo, lahko dodamo svoja poglavja. Poglavje je namenjeno bolj strukturirani razvrstitvi naših elementov, ki bodo nastopali v našem končnem modelu. Mi smo razdelili poglavja na Agente, Pravila in Simulacijo, kar je vidno na Slika 7 .



Slika 7: Seznam poglavij, ki sestavljajo knjigo

Nove elemente vstavljamo v poglavja preko orodne vrstice, ki je prikazana na Slika 8. Na voljo imamo **Agente**, **Pravila**, **Poglavja**, **Model** (simulacija) ali strani za splošen tekst.



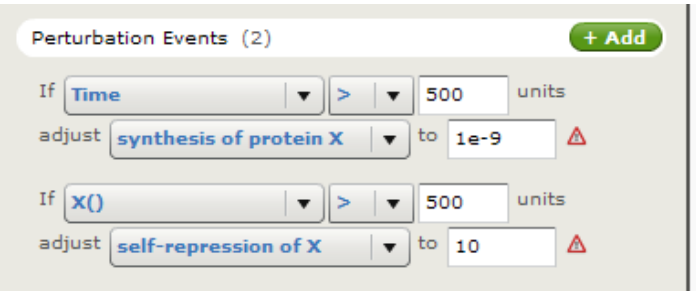
Slika 8: Orodna vrstica za vstavljanje agentov, pravil, modelov, poglavij...

Ko si opremimo model s potrebnimi pravili in začetnimi nastavitvami, lahko definiramo tudi novo simulacijo. Zgornji del Slika 9 predstavlja vmesnik za nastavitev parametrov simulacije (tip simulacije, dolžina simulacije, število zbranih končnih vzorcev). Rezultat simulacije je prikazan v grafični obliki na spodnjem delu Slika 9. Slika 11

Slika 9: Nastavljanje parametrov in rezultat simulacije

RuleBase omogoča dve vrsti dinamične simulacije – stohastične in deterministične (NDE). Deterministične simulacije se izvajajo preko numeričnega vrednotenja s pomočjo navadnih diferencialnih enačb (NDE). Deterministične simulacije so pogosto veliko hitrejše kot stohastične, ampak ne upoštevajo naključnosti kemijskih sistemov, ki lahko močno vplivajo na obnašanje modela. Ta naključnost je izrecno zajeta v stohastičnih simulacijah, ki uporabljajo metodo Gillespie za simulacijo naključnega pojavljanja posameznih reakcij v kemičnih sistemih.

V razdelku »Perturbation Events« lahko nastavimo sprožilce dogodkov med samim potekom simulacije. Za sprožilec lahko izberemo čas ali količino posameznega agenta. Sprožilec določimo v prvi vrstici in priredbo določimo v drugi vrstici. Nastavljanje je zelo jasno in enostavno, kar je razvidno iz Slika 10.



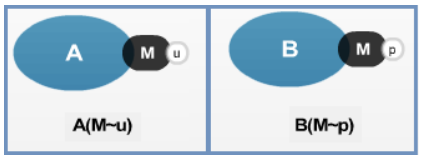
Slika 10: Peturbation events

Kappa

Za modeliranje preklopnega stikala smo uporabili agent-pravilo (eng. agent- and rule-based) jezik imenovan Kappa. Sistem v Kappi je sestavljen iz agentov in pravil.

Agenti

Agenti so funkcionalne komponente, ki omogočajo predstavitev skorajda katerekoli celične komponente, od enega atoma kalcija, pa do največjih makromolekulskih struktur kot ribosom ali kot celotna celica. Lahko si jih predstavljamo tudi kot poimenovano entiteto z množico mest. Mesta imajo notranja stanja in se lahko povezujejo z mesti na drugih agentih. Notranja stanja so ponavadi fosforilizirana ali pa nefosforilzirana. Prikaz agentov z mestom v obeh stanjih je na Slika 11.

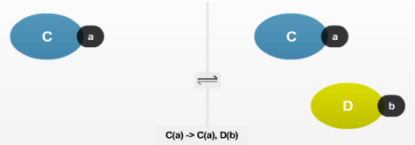


Slika 11: Levo je agent A z nefosforiliziranim mestom M in na desni agent B z fosforiliziranim mestom M.

Pravila

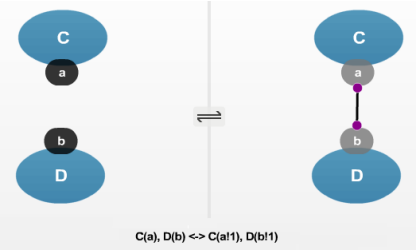
Pravilo je rekonfiguracija agentov, ki predstavlja kemijsko reakcijo. Leva stran pravila določa agente potrebne za sprožitev pravila oziroma reakcije. Desna stran pravila določa spremenjene strukture agentov, ki nastanejo kot rezultat izvršitve pravila. Pravila opisujejo pogoje, pod katerimi se agenti združujejo, razkrojujejo ali spreminjajo. Prav tako lahko pravila gradijo nove agente ali izbrišejo obstoječe. Utežitev pravila v simulacijah je odvisna od kinetične stopnje in koncentracije ujemajočih se vzorcev agentov, ki so potrebni za njegovo sprožitev. Obstaja pet različnih tipov pravil: kreacija, povezovanje, razpad, sprememba in uničenje. Pravil so prikazana na slikah 12-16.

Kreacija



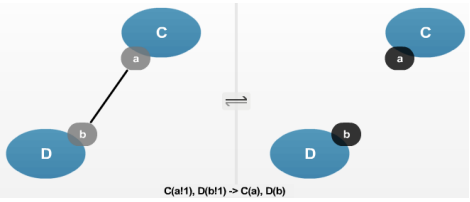
Slika 12: Kappa pravilo za kreacijo: C(a) -> C(a), D(b)

Povezovanje



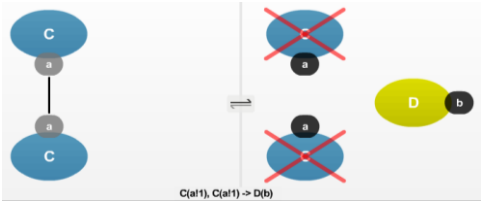
Slika 13: Kappa pravilo za povezovanje: C(a), D(b) <-> C(a!1), D(b!1)

Razpad



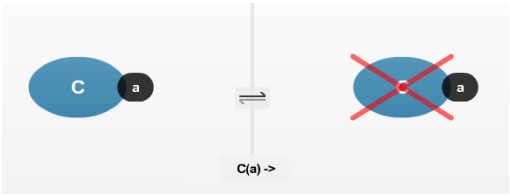
Slika 14: Kappa pravilo za razpad: C(a!1), D(b!1) <-> C(a), D(b)

Sprememba



Slika 15: Kappa pravilo za spremembo: C(a!1), C(a!1) -> D(b!1)

Uničenje

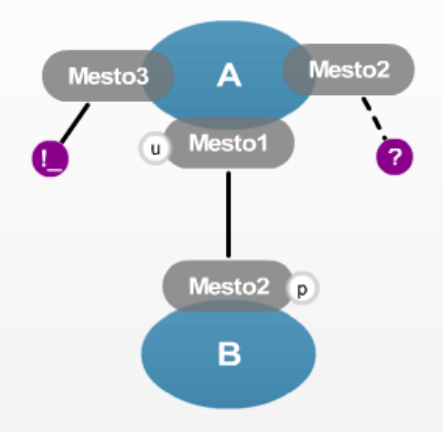


Slika 16: Uničenje

Vezi (bonds)

Vezi ali stanja lahko dodamo mestom na agentih. Obstajajo tri vrste vezi in so prikazane na Slika 17:

- Mesto-mesto: fizična povezava med dvema mestoma, ki označuje končni točki povezave. Vsako mesto se lahko povezuje z natanko enim mestom
- Katerakoli vez: povezano mesto, kjer ni določeno mesto s katerim se povezuje
- »Wildcard« vez: tako mesto je lahko povezano ali pa ne in se ponavadi uporablja na mestih z notranjimi stanji.



Slika 17: Mesto1 je povezano z Mestom2 z navadno povezavo, Mesto2 ima katerokoli povezavo, Mesto3 pa »Wildcard«.

Sintaksa

Sintakso Kappe je najenostavnejše razložiti na primeru:

Pravilo: Unphosphorylated Mesto1 of A binds to Mesto1 of B
Pravilo v Kappi: A(Mesto1~u),B(Mesto1) ->A(Mesto1~u!1),B(Mesto1!1)

Agenti, ki so ponavadi proteini, imajo poljubna imena, ki morajo biti iz alfa-numeričnih znakov. V zgornjem zgledu sta A in B dva različna agenta, ki se pojavita v pravilu.

Mesta (sites) na agentu omogočajo interakcijo in so zapisana v oklepajih, ki sledijo imenu agenta. Več mest lahko med sabo razdelimo z vejico (,), pri poimenovanju uporabljamo alfa-numerične znake. V primeru agenta A, je Mesto1 eno od mest na agentu A, enako velja za B. Upoštevati moramo, da dve mesti na istem agentu ne morejo imeti enakega imena, prav tako v pravilu ni obvezujoče uporabiti vseh mest. Tudi če je mesto omenjeno, lahko njegovo notranje stanje ostane nedefinirano.

Leva stran pravila določa pogoje za sprožitev pravila, medtem ko desna stran določa spremembe agentov na levi.

!: označuje fizično vez, ki uporablja skupne oznake za označitev dveh koncev povezave med agenti. V zgornjem primeru sta A in B nepovezana na levi strani in postaneta povezana na desni strani. Vezava je označena z »!1« na vsakem mestu vsebovanem v vezavi. Naslednja vezava bi bila označena z »!2« in tako naprej. Vsako mesto je lahko povezano natanko z enim mestom.

!_ označuje povezano mesto, vendar ni določeno s katerim mestom se povezuje. Tak tip povezave se imenuje katerakoli povezava (ang. any bond).

!? označuje stran, pri kateri ni pomembno, če je povezana ali ne. Tak tip povezave se imenuje »wildcard« povezava.

-> označuje enosmerno pravilo. V zgornjem primeru je reakcija enosmerna in ne pride do razkroja.

<-> označuje dvostransko pravilo.

~ označuje *vrednost*, ki določa notranje stanje agentovega mesta. Vrednost je lahko poljubna alfa-numerična beseda.

~u notacija nefosforiliziranega mesta. V zgornjem primeru ima mesto1 agenta A vrednost »~u«.

~p notacija fosforiliziranega mesta.

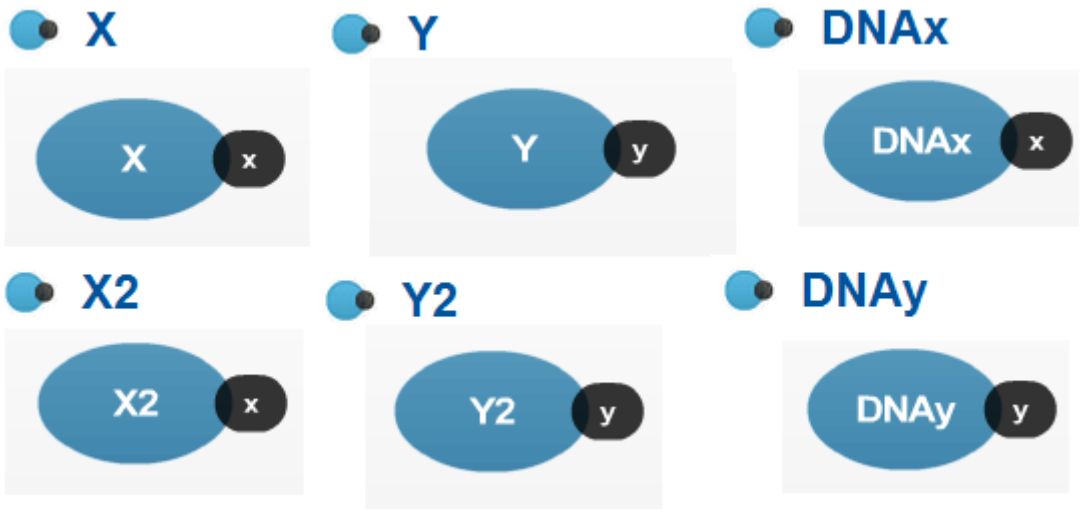
Sintaksa določuje naslednje objekte:

%init: number * agent list	Inicializacija začetnih količin agentov.
%mod: perturbation expression	Določitev medsimulacijskih sprožilcev. Sprožilec je lahko čas ali agent.
%obs: rule or agent observable	Izbor pravil oz. agentov, ki jih opazujemo tekom simulacije.
%story: rule	
'rule label' agent list	Deklaracija pravil v našem modelu.

Vzpostavitev modela

Predstavitev agentov

V modelu smo uporabili šest agentov, ki so predstavljeni na Slika 18.



Slika 18: Vsi agenti uporabljeni v modelu

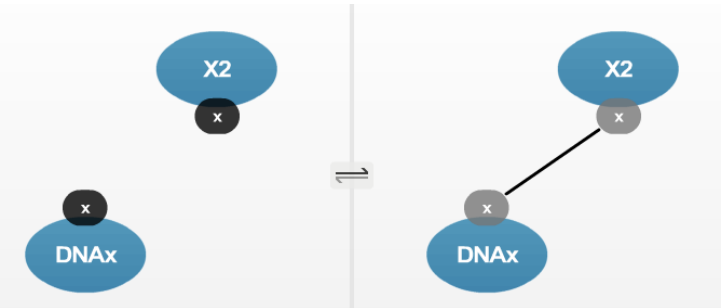
Predstavitev reakcij

Samo-represija proteina X

Poteka po enačbi DNAX + X2 -> DNAx*X2. Agenta, ki sodelujeta pri samo-represiji sta X2 in DNAx. Na levi strani Slika 19 sta nepovezana DNAx in X2, po poteku reakcije pa se med seboj povežeta in nastane kompleks DNAx*X2.

Koda v jeziku Kappa, kjer x!1 označuje povezanost mest x na obeh agentih:

'self-repression of X{334647}' X2(x), DNAx(x) -> X2(x!1), DNAx(x!1) @ 4.15134695790682e-05



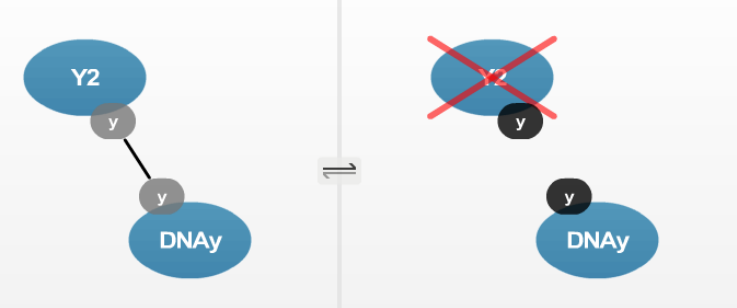
Slika 19: Samo-represija proteina X

Y degradacija povezanega dimera Y2

Degradacija poteka po enačbi Y2*DNAy -> DNAy. Sodelujoča agenta sta Y2 in DNAy. Na levi strani Slika 20 sta med seboj še povezana, po degradaciji pa se Y2 uniči in ostane samo DNAy.

Koda v jeziku Kappa, kjer y!1 označuje povezanost mest y na obeh agentih:

'Ydegradation of bounded dimer Y2{334658}' DNAy(y!1), Y2(y!1) -> DNAy(y) @ 0.5.



Slika 20: Y degradacija dimera Y2

Sinteza proteina Y

Poteka po enačbi DNAy -> Y + DNAy. Agent DNAy je osnova za sintezo, agent Y pa je produkt sinteze. Na Slika 21 je na levi strani DNAy, na desni strani pa po poteku sinteze DNAy in protein Y.

Koda v jeziku Kappa:

'synthesis of protein Y{334642}' DNAy(y) -> DNAy(y), Y(y) @ 1.1

Slika 21: Sinteza proteina Y

Opis programske kode modela

```
#
# Programska koda Kappa za postavitev modela Toggle Switch
#
```

```
#agentov, ki jih uporabljamo v pravilih ni potrebno ločeno deklarirati!
#deklaracija pravil nad agenti (naziv  pravila, pravilo, konstanta)
'synthesis of protein X{334641}'DNAx(x) ->DNAx(x), X(x) @ 1.1
'synthesis of protein Y{334642}'DNAy(y) ->DNAy(y), Y(y) @ 0.5
'dimerization of protein X{334643}'X(x), X(x) ->X2(x) @ 9.0e-02
'dimerization of protein Y{334644}'Y(y), Y(y), Y(y) ->Y2(y) @ 1.50e-3
'monomerization of dimer X2{334645}'X2(x) ->X(x), X(x) @ 0.5
'monomerization of dimer Y2{334646}'Y2(y) ->Y(y), Y(y) @ 0.5

#pravila povezovanja
'self-repression of X{334647}'X2(x), DNAx(x) ->X2(x!1), DNAx(x!1) @0.1,0.5
'self-repression of Y{334648}'Y2(y), DNAy(y) ->Y2(y!1), DNAy(y!1) @  0.1,0.5
'cross-repression of X{334649}'Y2(y), DNAx(x) ->Y2(y!1), DNAx(x!1) @  1,0.5
'cross-repression of Y{334650}'X2(x), DNAy(y) ->X2(x!1), DNAy(y!1) @1,0.5

#pravilo uničenja
'degradation of protein Y{334652}'Y(y) ->  @ 2.50e-3
'degradation of protein X{334651}'X(x) ->  @7.0e-4
'degradation of dimer Y2{334654}'Y2(y) ->  @2.50e-3
'degradation of dimer X2{334653}' X2(x) ->  @7.0e-4

#pravilo razpada
'Ydegradation of bounded dimer X2{334656}'DNAy(y!1), X2(x!1) ->DNAy(y) @ 0.5
'Xdegradation of bounded dimer X2{334655}'DNAx(x!1), X2(x!1) ->DNAx(x) @ 0.5
'Xdegradation of bounded dimer Y2{334657}'DNAx(x!1), Y2(y!1) ->DNAx(x) @ 0.5
'Ydegradation of bounded dimer Y2{334658}'DNAy(y!1), Y2(y!1) ->DNAy(y) @ 0.5




#inicializacija začetnih pogojev ( %init:, količina * ( agent ) )
%init: 200 * (X(x))
%init: 0 * (Y(y))
%init: 200 * (X2(x))
%init:0 * (Y2(y))
%init: 200 * (DNAx(x))
%init: 200 * (DNAy(y))

#opazovanje
#nastavitev agentov za opazovanje
%obs:'X()'X()
%obs:'Y()'Y()
#nastavitev pravil za opazovanje (%obs:, naziv pravila)
%obs:'dimerization of protein Y{334644}'
%obs:'dimerization of protein X{334643}'

#sprožilci vmesnih dogodkov, ki vplivajo na potek simulacije
%mod: $T>100.0 do'degradation of protein X{334651}':=100.0
%mod: $T>100.0 do'degradation of dimer X2{334653}':=100.0

%story:'synthesis of protein X{334641}'
```

Rezultati simulacije

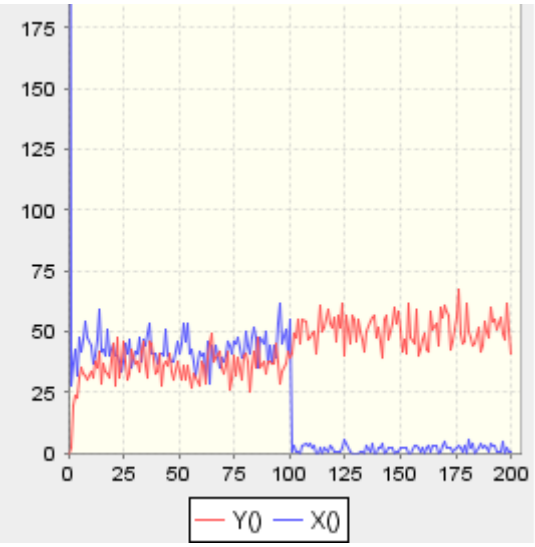
Vhodni parametri

Oznaka	Pomen	Vrednost 1	Vrednost 2	Vrednost 3
kx	hitrost sinteze proteina X	1.1 nM/s	1.1 nM/s	0.5 nM/s
ky	hitrost sinteze proteina Y	0.5 nM/s	1.1 nM/s	0.5 nM/s
kdx	hitrost degradacije proteina X	7.0*10^(-4) s^-1	7.0*10^(-4) s^-1	2.5*10^(-3) s^-1
kdy	hitrost degradacije proteina Y	2.5*10^(-3) s^-1	7.0*10^(-4) s^-1	2.5*10^(-3) s^-1
kdxf	hitra degradacija proteina X	1 s^-1	1 s^-1	1 s^-1
kdyf	hitra degradacija proteina Y	1 s^-1	1 s^-1	1 s^-1
k1r	hitrost dimerizacije proteina X	9.0 * 10^(-2) nM^-1 s^-1	9.0 * 10^(-2) nM^-1 s^-1	1.5*10^(-3) nM^-1 s^-1
k1c	hitrost dimerizacije proteina Y	1.5*10^(-3) nM^-1 s^-1	9.0 * 10^(-2) nM^-1 s^-1	1.5*10^(-3) nM^-1 s^-1
k-1r	cepitev dimera X2 na 2 X-a	0.5 s^-1	0.5 s^-1	0.5 s^-1
k-1c	cepitev dimera X2 na 2 Y-a	0.5 s^-1	0.5 s^-1	0.5 s^-1
kdr	degradacija dimera X2	7*10^(-4) s^-1	7*10^(-4) s^-1	2.5*10^(-3) s^-1
kdc	degradacija dimera Y2	2.5*10^(-3) s^-1	7*10^(-4) s^-1	2.5*10^(-3) s^-1
kf	hitrost vezave proteina na drug DNK	1 nM^-1 s^-1	1 nM^-1 s^-1	1 nM^-1 s^-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
n	število DNK	200	200	200

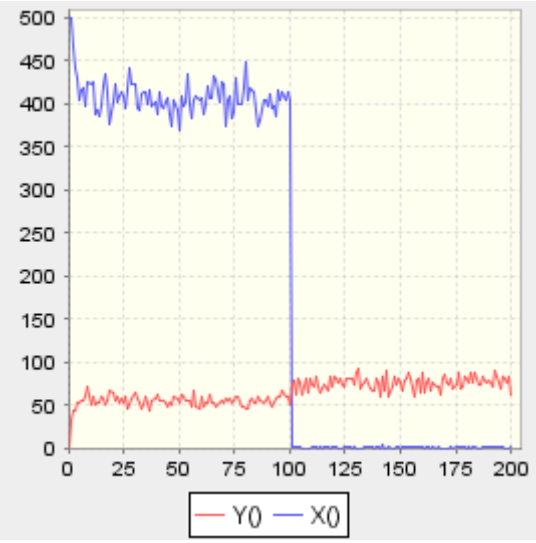
Slika 22: Vhodni parametri uporabljeni v simulacijah

Stohastični model

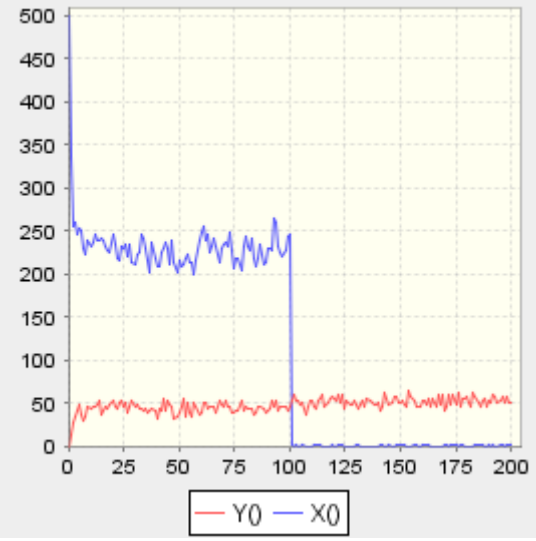
X() in Y() predstavljata koncentracijo proteinov X in Y.



Slika 23: Simulacija modela za vrednosti parametrov 1



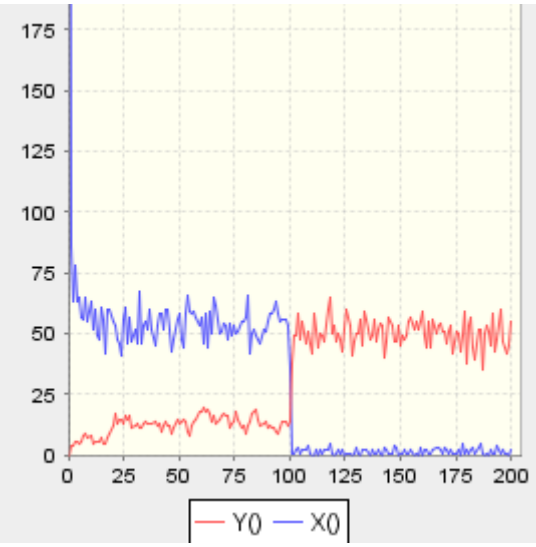
Slika 24: Simulacija modela za vrednosti parametrov 2



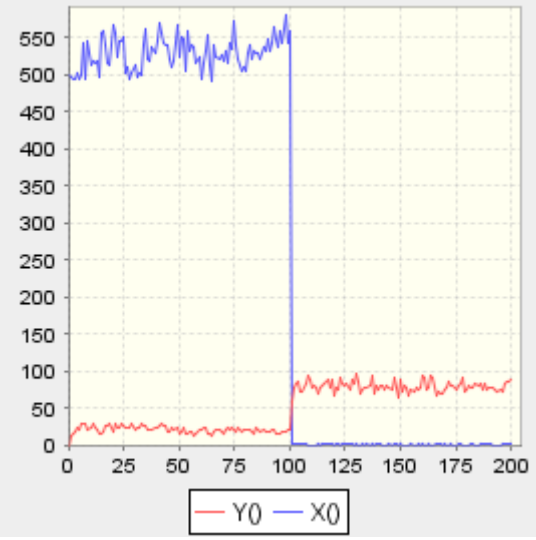
Slika 25: Simulacija modela za vrednosti parametrov 3

Rezultati simulacij s spremenjenim vhodnim parametrom:

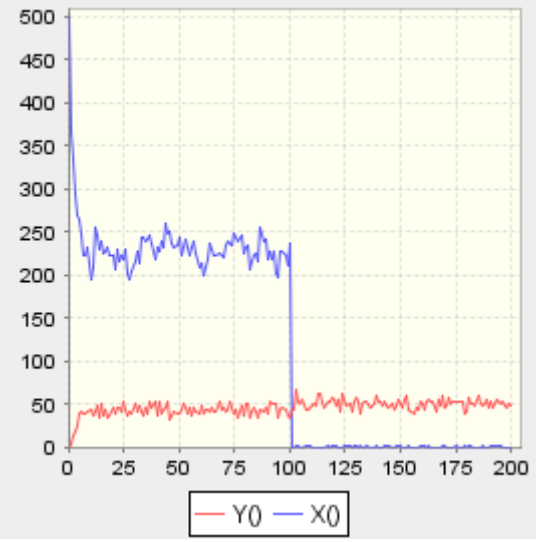
kfs	hitrost vezave proteina na svoj DNK	0.001*kf	0.001*kf	0.001*kf



Slika 26: Simulacija modela za vrednosti parametrov 1



Slika 27: Simulacija modela za vrednosti parametrov 2



Slika 28: Simulacija modela za vrednosti parametrov 3

Deterministični model

Deterministčnega modela na spletnem portalu RuleBase nismo uspeli realizirati. Obstaja sicer možnost izbire med stohastičnim in deterministčnim sistemom v omenjenem portalu, vendar stvar ne deluje. Pri zagonu simulacije se pojavljajo napake, prav tako ne izpiše Kappa kode.

Analiza

Rezultati simulacij niso ravno v skladu s pričakovanji. Pri simulacijah z osnovnimi vhodnimi parametri nismo uspeli doseči željenega rezultata. V prvem primeru (Rezultati simulacije –Slika 23) sta koncentraciji proteinov X in Y pred povečanjem degradacije proteina Xpreblizu skupaj, v drugem (Rezultati simulacije – Slika 24) in tretjem (Rezultati simulacije – Slika 25) primeru pa nenadno zmanjšanje koncentracije proteina X zelo malo vpliva na koncentracijo proteina Y.

Šele po spremembi vhodnih parametrov nam je uspelo realizirati preklopno stikalo (Rezultati simulacije – Slika 26). Delovanje preklopnega stikala je najboljše v primeru, ko so parametri za X in Y različni.

Zaključek

V seminarski nalogi smo spoznali in raziskali jezik in orodja za delo s Kappo. Samo delo s Kappo je dokaj zanimivo, predvsem delo z grafičnim vmesnikom na spletnem portalu RuleBase. RuleBase omogoča hitro razumevanje osnovnih pristopov vzpostavitve modela, generiranja agentov, sestavljanja pravil, itd.. Priporočamo ga začetnikom, ki še nimajo osnovnega znanja za delo s Kappo. Zagotovo pa se bo portal tekom let tudi uspešno razvijal in posodavljal, da bo delo bolj kakovostno in zanesljivo. Uspešno smo postavili stohastični model preklopnega stikala, čeprav za nekatere parametre ni deloval v skladu s pričakovanji. Z deterministčnim modelom smo imeli nemalo težav, saj se na portalu RuleBase pojavljajo napake, simulacija se ne izvede, prav tako ne izdela Kappa kode, kot je to pri stohastičnemu modelu. Prav pri deterministčnemu modelu opazimo, da ima orodje še veliko možnosti izboljšav. Druga možna izboljšava je sama hitrost delovanja spletnega portala, ki deluje zelo počasi ter sama stabilnost delovanja. Nemalokrat pride do situacije, da se portal enostavno zruši. V okolju Eclipse je samo delovanje veliko bolj stabilno in hitro, vendar tudi tukaj pogrešamo možnost postavitve deterministčnega modela.

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
- Miha Moškon, Miha Mraz: Analysing the information processing capabilities of biological systems. November , 2010.
- Michael B. Elowitz, Stanislas Leibler: A synthetic oscillatory network of transcriptional regulators. *Nature*, January 2000.
- Ron Weiss, Subhayu Basu: The device physics of cellular logic gates, 2002.

- http://2010.igem.org/Team:Edinburgh/Modelling/Kappa
- http://support.cellucidate.com/forums/92334/entries/84208
- http://support.cellucidate.com/entries/81966-creating-rules
- http://support.cellucidate.com/entries/81963-creating-agents
- http://www.di.ens.fr/~feret/teaching/MPRI/local_views_MPRI.pdf