Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Martin Stražar

Paralelni evolucijski algoritem za odkrivanje znanja iz modela genskega regulatornega omrežja

DIPLOMSKA NALOGA
NA UNIVERZITETNEM ŠTUDIJU

prof. dr. Miha Mraz
MENTOR

Ljubljana, 2013

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01930/2013
Datum: 15.04.2013

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **MARTIN STRAŽAR**

Naslov: **PARALELNI EVOLUCIJSKI ALGORITEM ZA ODKRIVANJE ZNANJA IZ MODELA GENSKEGA REGULATORNEGA OMREŽJA**

PARALLEL EVOLUTIONARY ALGORITHM FOR KNOWLEDGE DISCOVERY FROM A MODEL OF A GENE REGULATORY NETWORK

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Kandidat naj v svojem delu skuša pospešiti metodo stohastičnega modeliranja vzorčnega genskega regulatornega omrežja bistabilnega stikala. Predlagana pospešitev lahko vsebuje tako način izvajanja algoritma (npr. ciljno strojno arhitekturo), kot tudi druge spremembe osnovnega stohastičnega algoritma, ki temelji na kemijski glavni enačbi. Pri tem naj se kandidat usmeri tudi na problem avtomatizacije prilagoditve vrednosti parametrov za izenačevanje simulacijskih in eksperimentalnih rezultatov.

Mentor:

prof. dr. Miha Mraz

Dekan:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

IZJAVA O AVTORSTVU DIPLOMSKEGA DELA

Spodaj podpisani izjavljam, da sem avtor dela, da slednje ne vsebuje materiala, ki bi ga kdorkoli predhodno že objavil ali oddal v obravnavo za pridobitev naziva na univerzi ali drugem visokošolskem zavodu, razen v primerih kjer so navedeni viri.
S svojim podpisom zagotavljam, da:

- sem delo izdelal samostojno pod mentorstvom prof. dr. Mihe Mraza,
- so elektronska oblika dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko in
- soglašam z javno objavo elektronske oblike dela v zbirki "Dela FRI".

— Martin Stražar, Ljubljana, junij 2013.

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Martin Stražar

Paralelni evolucijski algoritem za odkrivanje znanja iz modela genskega regulatornega omrežja

POVZETEK

Sintetična genska regulatorna omrežja predstavljajo enega izmed zadnjih dosežkov področja sintezne biologije ter so potencialne platforme za procesiranje informacij v prihodnosti. Pomembno vlogo pri načrtovanju takih omrežij igra matematično modeliranje, ki omogoča tako teoretičen opis in potrditev pravilnosti načrta sistema, kot tudi napovedovanje odzivov pri spremenjenih eksperimentalnih pogojih in topologijah omrežja.

Ocenjevanje vrednosti parametrov modelov je eden od glavnih problemov modeliranja bioloških sistemov. Pri uporabi modeliranja na realnih primerih navadno ne poznamo ali ne moremo izmeriti vseh vrednosti parametrov, zato so ti pogosto prilagojeni eksperimentalnim podatkom. Z uporabo optimizacijskih metod, ki minimizirajo napako med rezultati modela in dejanskimi eksperimentalnimi podatki, lahko pridobimo ocene za vrednosti parametrov.

Pristopi za opisovanje dinamike bioloških sistemov se razlikujejo po vrsti in natančnosti opisa ter posledično računski zahtevnosti. V pričujočem delu opišemo sistem, ki združi algoritem za stohastično simulacijo bioloških sistemov z evolucijskim algoritmom za ocenjevanje vrednosti parametrov. Dodatno predstavimo izboljšave algoritma, potrebne za optimalno izvajanje na paralelnem grafičnem procesorju. To nam omogoča natančno, kvantitativno napoved odziva sistema, kar dokažemo s primerjavo rezultatov modela in testnim eksperimentom.

Z omejitvijo začetne ocene vrednosti parametrov na biološko sprejemljive vrednosti, dobljeni rezultati z veliko verjetnostjo predstavljajo dovolj dobro oceno resničnih vrednosti. Slednji nam omogočajo vpogled v delovanje kompleksnega genskega regulatornega omrežja, stikala s pozitivno povratno zanko. Relativna razmerja med parametri lahko uporabimo za številsko oceno pogostosti izvajanja posameznih reakcij in tako pojasniti dobljene eksperimentalne podatke. Eksperimentalna potrditev pridobljenega znanja po-

meni korak bliže k razumevanju kompleksnih sistemov in grajenju bioloških struktur s predvidljivimi funkcijami.

Ključne besede: Gensko regulatorno omrežje, stohastični simulacijski algoritem (SSA), evolucijski algoritem, optimizacija, iskanje vrednosti parametrov, paralelno procesiranje, CUDA.

University of Ljubljana
Faculty of Computer and Information Science

Martin Stražar

Parallel Evolutionary Algorithm for Knowledge Discovery from a Model of a Gene Regulatory Network

ABSTRACT

Synthetic gene regulatory networks represent *state-of-the-art* achievements of synthetic biology and are promising candidates for information processing platforms in the future. Mathematical modelling plays an essential role in design and testing by enabling a theoretical insight and indentification of the key system features. It is also used as tool in predicting the gene regulatory network response under different experimental conditions or using various topologies.

Parameter estimation is one of the main problems in modelling biological systems in general. In real world modelling applications, not all parameter values are known or can be measured and are often fitted to experimental data, where optimization methods minimize some measure of error between predicted system response and actual data.

Paradigms in modeling biological systems differ in description form and precision. Intuitively, high precision of the results is proportional with computational complexity. In the present work, we present a system which merges a stochastic simulation algorithm with an evolutionary algorithm to optimize the model parameters. We made improvements of the standard simulation algorithm to optimize the execution on a parallel graphic processing unit. The results enable us to predict a precise quantitative system response, which we prove by confirming model prediction on a test experiment.

By limiting the initial guess of parameter values to valid biological estimates, the obtained parameter values reflect realistic values with high probability. These offer an insight into the machinery of a complex gene regulatory network, a bistable switch with positive feedback loop. The relative ratios between obtained parameters can be used to quantify the frequency of individual occuring reactions and explain the experimetnal results. By experimentally confirming obtained knowledge, we can get a step closer towards understanding complex biological systems and building structures with predictable

functions.

Key words: Gene regulatory networks, stochastic simulation algorithm (SSA), optimization, parameter estimation, parallel processing, CUDA.

ZAHVALA

Navdih in vsebinsko podlago diplomskega dela predstavlja projekt ekipe Univerze v Ljubljani na tekmovanju iz sintezne biologije iGEM 2012 na univerzi MIT. Za odlično vzdušje pred, med in po tekmovanju, v delovnem času ter izven njega, se zahvaljujem prijateljem članom ekipe. Hvala Urban Bezeljak, Lucija Kadunc, Dušan Vučko, Maja Somrak, Boštjan Pirš, Anja Golob, Miha Jerala, Uroš Zupančič, Fedja Pavlovec, Zala Lužnik ter mentorjem iz Laboratorija za biotehnologijo na Kemijskem inštitutu.

Za pomoč pri izdelavi dela in spodbudo za interdisciplinarno delo se zahvaljujem mentorju prof. dr. Mihi Mrazu ter somentorju dr. Mihi Moškonu. Marsikateri koristni nasvet, vir in delovno vzdušje pri izdelavi diplomskega dela so prispevali člani Laboratorija za računalniške strukture in sisteme, Mattia Petroni, Jure Demšar, Primož Pečar, Miran Koprivec, Domen Šoberl in Iztok Lebar Bajec.

Posebna zahvala za sodelovanje, eksperimentalne rezultate, kritično branje dela, uvajanje v laboratorijsko delo (z odpuščanjem napak) in druge zabavne trenutke pripada Tini Lebar.

Na koncu se za podporo v vseh pomenih skozi leta dodiplomskega študija zahvaljujem svoji družini Olgi, Božotu in Evi, brez katerih izdelava tega dela in dokončanje študija ne bi bilo mogoče.

— Martin Stražar, Ljubljana, junij 2013.

KAZALO

1 Uvod

Zaključeno dvajseto stoletje velja v znanosti za stoletje *razumevanja*. Količina novega znanja na področju naravnih zakonov se je v zadnjih stotih letih povečala kot nikoli dotlej. Smo na točki preloma, ko razpolagamo z veliko količino zbranega znanja v obliki teorij, dokazov ter prosto dostopnih podatkovnih virov. Na podlagi slednjih dejstev je trenutno stoletje označeno kot stoletje *sestavljanja*. Ključno vprašanje ni več, ali naravo lahko razumemo, temveč kaj lahko v njeni domeni z obstoječim znanjem sestavimo.

Z zgornjim sestavkom lahko povežemo vsako inženirsko naravnano znanost, med katere sodi tudi sintezna biologija. Z rojevanjem novih tehnologij smo ljudje sposobni opazovanja in posledično merjenja količin na vedno nižjem nivoju z vidika reda fizične velikosti. Razumevanje funkcije molekule DNA [11] in objava njene strukture [47], določanje DNA zaporedja genomov virusov [13], prvih prokariontskih organizmov [14] in človeka [50] zagotovo sodijo med večje prelomnice znanosti.

Po uspešni izvedbi sekveniranj prvih genomov organizmov so sledili poskusi razumevanja vloge genov ter posledično proteinov. Skladno s tem so se razvile baze podatkov (PDB, GeneOntology, GenBank, itd.), ki odkriti gen v organizmu povezujejo z nabo-

rom pojmov, ki označujejo pripadajoče lastnosti in funkcijo. Vinska mušica (*Drosophila melanogaster*) je v tem pogledu eden izmed najbolje razumljenih organizmov. V bazi GenBank je mogoče v času pisanja tega dela zanjo najti 15.867 pripadajočih označenih genov ter 27.752 proteinov.

Zaradi svojega velikega pomena je molekula DNA predmet mnogih raziskav, skladno s čimer so se razvile zanesljive tehnike njene manipulacije, kar omogoča nadzorovano in načrtovano izdelovanje umetnih zaporedij DNA. Mednje sodijo tudi taka, ki jih v naravi ne najdemo. Pri tem je vselej pomembno, da so sestavni deli dobro opisani (karakterizacija), kar je vodilo v pobudo za organizirane baze podatkov o znanih DNA konstruktih [41]. Iz samega zapisa DNA trenutno še ni mogoče enolično predvideti njene končne funkcije, zato se podatki nanašajo na gene in pripadajoče proteine ter njihovo interakcijo z drugimi proteini in DNA.

S standardizacijo in uvedbo principov inženirstva je nastala znanost *sintezna biologija*, ki s poznavanjem sestavnih delov in njihovih medsebojnih interakcij poizkuša sestavljati kompleksne sisteme za določen namen. Če si molekulo DNA predstavljamo kot program, organizem pa kot stroj, ki ta program izvaja, lahko s programiranjem dosežemo želeno obnašanje organizma. Možnosti uporabe so sprva vključevale zgolj proizvajanje produkta želenega gena, ki je lahko biološko zdravilo (npr. rastni faktor, zaviralec kopičenja prekomerne telesne mase), katalizator reakcij (encim), poročevalski protein, itd. V skladu z zgornjo prispodobo lahko danes sestavljamo kompleksnejše sisteme, ki so sposobni tako proizvajanja kompleksnejših produktov več genov (biogorivo ali hrana), kot tudi procesiranja informacij. Med primere slednjih sodijo primitivne računske strukture, kot so na primer logična vrata [12, 28, 35], pomnilni elementi [2], stikala [16, 48] ter oscilatorji [24, 45, 46], ki sodijo med zadnje dosežke področja (angl. *state-of-the-art*) sintezne biologije. Za vse omenjene sisteme velja, da so sestavljeni *iz gradnikov* (modulov) in bodo kot taki v prihodnosti uporabljeni *kot gradniki* za izgradnjo kompleksnejših sistemov. Če smo torej nekoč bili sposobni vplivati na produkcijo produkta le enega gena, lahko danes s poznavanjem njihovih interakcij zgradimo logična vezja (računske elemente), ki bodo v prihodnosti lahko združeno izvajala kompleksnejši program.

Eden od največjih virov idej za uporabo sintezne biologije je tekmovanje iGEM (angl. *International Genetically Engineered Machine*), ki se odvija vsako leto na univerzi MIT v Cambridgeu, Massachussets, ZDA. Tekmovanje sloni na pobudi o standardizaciji DNA konstruktov [27], iz katere je nastal tudi register gradnikov. Dober pregled nad po-

dročjem, možnostmi uporabe in izzivi v sintezni biologiji bralec najde v [25].

1.1 Genska regulatorna omrežja

Za razumevanje bistva, namena in priložnosti v sintezni biologiji, je potrebno poznavanje osnovnih principov delovanja celice, natančneje mehanizmov, ki to delovanje nadzorujejo. Celice so osnovne življenjske oblike, ki se združujejo v tkiva, ta nadalje v organe, slednji pa na koncu sestavljajo organizme.

Ker organi večceličnih organizmov opravljajo različne funkcije, so sestavljeni iz različnih tkiv, ta pa iz različnih vrst celic. V zgodnji fazi razvoja organizma (zarodku) se formira veliko število matičnih celic, ki jim funkcija še ni določena. Kasneje v razvoju organizma gredo matične celice skozi proces diferenciacije in tako privzamejo končno funkcijo.

Delovanje celice določa molekula deoksiribonukeinske kisline (DNA). Ta je sestavljena iz dveh fosfatnih verig, ki med seboj povezujeta pare baz (nukleotidov). V DNA najdemo štiri, gvanin (G), adenin (A), timin (T) in citozin (C), pri čemer se med seboj povezujeta bazi TA ter CG. Vzdolž fosfatne verige si tako sledi več ponovitev baznih parov (bp), ki tvorijo dve komplementarni verigi. Genom dobro opisane bakterije *Escherichie coli* je dolg približno $4.6 \cdot 10^6$ bp, genom vrste *homo sapiens* približno $3 \cdot 10^9$ bp, genom virusa SARS pa približno 29.750 bp.

Zaporedje baznih parov si lahko predstavljamo kot sestavne dele programov, ki se izvajajo v celicah. Osrednja dogma molekularne biologije uči, da se na molekulo DNA veže encim RNA polimeraza, ki podzaporedje molekule prepiše v sorodno molekulo informacijske RNA (angl. *messenger RNA*, mRNA), ki je prav tako sestavljena iz zaporedja nukleotidov. Slednji proces imenujemo *transkripcija*. Postopek nadaljujejo ribosomi, ki mRNA prevedejo v zaporedje aminokislin (proces translacije). Pri tem se trojice nukleotidov prevedejo v eno od dvajsetih vrst aminokislin, ki se povežejo v peptidno verigo. Ta se pri tem zaradi različnih interakcij med aminokislinami zvije v tridimenzionalno strukturo, ki na koncu določa funkcijo proteina.

Posamezni enoti DNA, ki se po opisanih postopkih prevede v protein, pravimo *gen*. Proteini so nosilci osnovnih funkcij, ki jih opravlja celica, kot so prenašanje signalov med celicami, delovanje kot substrat v kemijskih reakcijah, reagiranje na signale iz okolja (svetloba, temperatura, prisotnost hranilnih snovi itd.), vezava na druge proteine in molekule, itd.

Molekula DNA je dovzetna za vezavo nekaterih vrst proteinov. Pogost pojav je vezava na nukleotide v bližini določenih genov, kar lahko vpliva na transkripcijo. Takšnim proteinom pravimo *transkripcijski faktorji*, sistemu interakcij med proteini in geni pa *gensko regulatorno omrežje*. Ocena za velikost celotnega genskega regulatornega omrežja bakterije *E. coli* znaša 4279 genov, med katerimi je bilo odkritih 381 regulatornih povezav [15].

Uravnavanje izražanja genov je s stališča sintezne biologije izredno zanimivo, saj s poznavanjem tovrstnih povezav lahko ustvarimo popolnoma nova regulatorna omrežja, ki v naravi ne obstajajo. Transkripcijski faktor lahko na izražanje gena vpliva *negativno* (represija; intenzivnost transkripcije se zmanjša) ali *pozitivno* (aktivacija; intenzivnost transkripcije se poveča). Transkripcijske faktorje tako delimo na *represorje* in *aktivatorje*, oboji pa neposredno vplivajo na količino proizvedenega proteina. Eden najbolje opisanih regulatornih mehanizmov iz narave je zaznavanje pomanjkanja glukoze (osnovnega hranila) bakterije *E. coli*. Ob prisotnosti laktoze (alternativnega hranila) se tako hkrati aktivirajo geni *lac*Z, *lac*Y in *lac*A, katerih produkti katalizirajo prebavo laktoze. Slednji se do tedaj niso izražali zaradi aktivnega represorja LacI, ki ob prisotnosti laktoze preneha z delovanjem.

Vzporedni razvoj postopkov za nadzorovano manipulacijo molekule DNA[1] je omogočil sestavljanje umetnih genskih regulatornih omrežij. Ta so vedno kompleksnejša, zaradi česar se pri načrtovanju poslužujemo tehnik matematičnega modeliranja, ki nam omogoča lažjo, predvsem pa hitrejšo napoved predvidenih funkcij.

1.2 Modeliranje genskih regulatornih omrežij in problem iskanja parametrov

Skladno s sintezno biologijo so se razvili postopki za t. i. *in silico* (lat. *v siliciju*) preizkušanje različnih topologij in vrst gensko regulatornih omrežij. Nameni modeliranja v sintezni biologiji so predvsem teoretično dokazovanje pravilnosti delovanja predlagane topologije, odkrivanje novih znanj o procesih, ki jih skozi meritve ne zaznamo vedno in pomoč pri načrtovanju eksperimentalnih poizkusov na podlagi obstoječega znanja. Skladno s tem lahko iz rezultatov simulacij modelov izvlečemo dodatne informacije, ki jih

[1] npr. rezanja (restrikcija), lepljenja (ligacija), pridobivanja DNA iz organizma (izolacija), vstavljanja konstrukta (zaključenega podzaporedja DNA) v organizem (transfekcija, transformacija), pomnoževanja (kloniranje), branja (sekveniranje) in sestavljanja (sinteza).

nato upoštevamo pri eksperimentalnemu delu. Dodatno lahko z modeli preizkusimo in potrdimo hipoteze o še nepoznanih interakcijah med kemijskimi zvrstmi.

Da bi lahko s pomočjo simulacij dobili uporabne rezultate, je za opis modela ne glede na izbran pristop potrebno (a) dobro poznavanje opazovanih interakcij, pri čemer se nagibamo k upoštevanju zgolj bistvenih in (b) natančno poznavanje parametrov, ki številsko opisujejo pogostost in druge lastnosti izvajanja kemijskih reakcij.

Problem (a) se navezuje na dobro poznavanje biokemijskega procesa, kjer je za nizko računsko kompleksnost pomembna minimizacija modela, torej upoštevanje najmanjšega števila interakcij [20]. Jedro problema je dokaz, da lahko z minimalnim modelom dobimo rezultate z dovolj majhnim odstopanjem od rezultatov celotnega modela. Čeprav obstajajo študije o hkratnem modeliranju vseh procesov v celici, bodo zaradi velikega števila posplošitev in računske zahtevnosti taki modeli za relativno ozko področje, kot je sintezna biologija, manj uporabni od specifičnih modelov, postavljenih za točno določen problem [23].

Problem (b) zahteva karseda natančno karakterizacijo izbranih interakcij v obliki številskih vrednosti (parametrov), ki (ne glede na izbran pristop modeliranja) določajo pogostost ter kvantitativne lastnosti le-teh. Primeri parametrov v domeni genskih regulatornih omrežij so lahko pogostost (nagnjenost) reakcije, hitrost transkripcije, hitrost translacije, razpolovna doba proteina, zamik med transkripcijo in translacijo, itd. Pri določanju parametrov je nekatere mogoče eksperimentalno izmeriti, določene najti v literaturi, v praksi pa se redko zgodi, da so čisto vsi poznani od začetka. Dodatno težavo predstavlja opazovana nelinearnost v domeni gensko regulatornih omrežij, ki temelji na opažanjih, da sestavljen sistem iz okarakteriziranih konstruktov ne bo dal ekvivalentnega odziva kot vsota odzivov le-teh [33, 39].

Paradigme modeliranja v osnovi delimo v dve skupini in sicer na *deterministično* in *stohastično* modeliranje. Determinstično modeliranje predstavlja poenostavitev stohastičnega, vendar sta to dva najpogosteje uporabljana pristopa, ki predstavljata računsko obvladljiv opis sistema kemijskih reakcij.

Determinstični modeli so tipično sestavljeni iz diferencialnih enačb, ki določajo spreminjanje opazovanih kemijskih zvrsti v sistemu skozi čas [3]. Pri tem so upoštevane tudi koncentracije ostalih kemijskih zvrsti, ki na spreminjanje koncentracije opazovane kemijske zvrsti vplivajo bodisi pozitivno ali negativno. Primer pozitivnega vpliva je lahko pospeševanje transkripcije tarčnega gena skozi čas, primer negativnega pa vezava

represorja in posledično zaviranje transkripcije tarčnega gena. Najpogostejši primer simulacije je numerična integracija diferencialnih enačb, s čimer lahko opazujemo odziv sistema skozi čas.

Stohastično modeliranje obravnava vsako reakcijo posebej, pri čemer je potrebno poznati število in vrsto reaktantov na eni in produktov na drugi strani. Z vidika genskih regulatornih omrežij je ta princip nekoliko bolj obsežen, saj predpostavlja opis tako vezave proteinov na tarčne gene, kot tudi produkcijo in degradacijo proteinov, kar je pri deterministčnem modeliranju tipično povzeto z eno enačbo. Seznam reakcij nato simuliramo z izbrano metodo, ki je poenostavitev kemijske glavne enačbe (angl. *Chemical master equation*). Slednja da natančen rezultat, a zahteva eksponentno naraščajoči čas simuliranja, kar jo naredi neprimerno za praktično uporabo. Zato so se pojavile poenostavitve, med katerimi je najbolj znan Gillespijev algoritem, imenovan tudi SSA (angl. *Stochastic Simulation algorithm*, [6, 18, 49]), ki temelji na naključnem času med dvema reakcijama, pogostost izvajanja le-teh pa je sorazmerna s pripadajočo nagnjenostjo (angl. *propensity*), ki je predstavljena s številsko vrednostjo in je odvisna od prisotnosti reaktantov. Slednji algoritem bo uporabljen v jedru dela, zato bo v nadaljevanju natančneje opisan.

Oba pristopa imata prednosti in slabosti. Izkustveno pravilo pravi, da je za večje sisteme, kjer prisotnost ene same molekule določene kemijske zvrsti ne vpliva bistveno na sistem, bolj primerno deterministično modeliranje, ki kemijske zvrsti obravnava številsko v obliki koncentracij [20]. Poleg tega so nekatere diferencialne enačbe rešljive tudi analitično, kar nam omogoča enostavno določanje iskanih stabilnih ali nestabilnih stanj sistema ter iskanje točk bifurkacije (točk v prostoru parametrov, kjer se kvalitativni odziv sistema spremeni) [5].

Po drugi strani je za manjše sisteme, pri katerih je pomen posamezne molekule kemijske zvrsti večji, bolj primerno stohastično modeliranje. To prinaša dodatno računsko zahtevnost in potrebo po večjem številu ponovitev simulacij, da pridemo do želene statistike. Prednost pristopa pa so natančnejši rezultati, saj ti prikazujejo tudi šum, ki je pri eksperimentalnem testiranju neizbežen. To nam omogoča identifikacijo na šum občutljivih sistemov, ter določitev verjetnosti, da je sistem v določenem času v določenem stanju [20].

Za optimalno izvajanje SSA algoritma na paralelni procesni arhitekturi so potrebne izboljšave algoritma, ki ustrezno prilagodijo potek izvajanja. Tako lahko vzporedno iz-

vajamo več simulacij istega omrežja hkrati. Ker gre za podatkovno relativno zahteven problem, obstoječe implementacije [22, 31, 32, 40] pri tem nadzorujejo dostop do pomnilnika ter odpravljajo uporabo vejitev, kar bo obravnavano v nadaljevanju. V [31] avtorji dosežejo do 170-krat hitrejši časi izvajanja, vendar se omejijo na relativno majhen sistem treh reakcij. V [22] avtorji primerjajo čas izvajanja več sistemov, pri sistemu 61 reakcij in 28 kemijskih zvrsti, ki je po velikosti primerljiv z našim, avtorji dosežejo do 30-kratno pohitritev. Eden od naših prispevkov so izboljšave, ki še dodatno izboljšajo čas izvajanja v primerjavi s serijsko implementacijo.

1.3 Določitev parametrov z evolucijskim računanjem

V računalništvu se srečujemo z velikim številom problemov, ki niso rešljivi ali jih ne znamo rešiti z analitičnim postopkom, ki bi v končnem času vrnil (enolično) optimalno rešitev. Kot enega od takih lahko obravnavamo določitev parametrov pri znanem odzivu kompleksnega sistema [34].

Naravni pojavi so bili pogosto navdih za izum novih algoritmov za reševanje optimizacijskih problemov. Tako se je razvila samostojna veja strojnega učenja, ki jo imenujemo *naravni algoritmi*. Znani so primeri razumevanja delovanja možganov, ki je vodilo v razvoj učnih algoritmov nevronskih mrež, samoorganizacija mravelj s pomočjo feromonov kot način za iskanje optimalnih poti v grafu, imunski sistem organizma kot način za razvoj varnostnih sistemov ali naravna selekcija kot primer izbiranja najboljših osebkov (rešitev) [37] .

Evolucijski algoritmi so pogost način reševanja problemov, pri katerih ne poznamo analitičnega postopka za pridobitev rešitve, imamo pa postopek za ocenjevanje kvalitete rešitve. V domeni genetskih algoritmov ta postopek imenujemo cenovna funkcija (angl. *cost function, fitness function*), katere definicija je bistvena za delovanje algoritma.

Temeljijo na paradigmi naravne selekcije, ki močnejšim osebkom v populaciji določa večjo verjetnost preživetja in s tem reprodukcije (angl. *survival of the fittest*). Kakor v naravi, kjer so osebki določeni z genskim zapisom, lahko tudi kandidata za rešitev predstavimo kot vektor parametrov. Ti so nato ocenjeni s cenovno funkcijo. Iz celotne populacije je nato izbrana podmnožica kandidatov za reprodukcijo, pri čemer imajo kandidati z boljšo vrednostjo večjo verjetnost izbire. S posebnimi postopki nato iz podmnožice ustvarimo novo populacijo, pri čemer sta uporabljena principa genetike, *križanje*

(ustvarjanje novih rešitev s kombinacijo dveh ali več obstoječih) in *mutacija* (spreminjanje vrednosti določenega parametra kot dela rešitve). Algoritem tako izvede veliko število iteracij selekcije, križanja in mutacij, ter skladno s principom naravne selekcije izboljšuje kandidate za optimalno rešitev, dokler ne doseže zadovoljive natančnosti, globalnega minimuma (konvergenca) ali maksimalnega dovoljenega števila iteracij.

Intuitivno sta jasni slabosti genetskih algoritmov. Ti sta velika računska zahtevnost (kvaliteta rešitve in hitrost konvergence sta odvisni od velikosti populacije [43]) ter možnost, da končna rešitev ne bo optimalna. Prvo slabost odpravimo z uporabo paralelnega procesiranja na grafičnem procesorju. Drugo navidezno slabost izkoristimo tako, da v začetku omejimo rešitve na biološke ocene za vrednosti parametrov. Posledica tega je, da vrednosti optimalnih parametrov na koncu optimizacije predstavljajo resnične vrednosti z veliko verjetnostjo. Z iskanjem rešitve torej algoritem prične *v bližini* lokalnega minimuma, ki nas zanima.

Evolucijski algoritmi so preprosti za razumevanje in implementacijo in primerni za reševanje splošnih problemov, predvsem če je prostor rešitev omejen (angl. *constrained optimization*). V pričujočem delu bomo raziskali ali lahko z evolucijskimi algoritmi razvijemo dobro metodo za ekstrakcijo znanja iz dobljenih odzivov kompleksnega sistema.

Za pregled nad razvojem in uporabo idej iz narave pri reševanju problemov je priporočeno branje vira [37]. Evolucijske algoritme je predlagal John Holland leta 1975 [19], definicijo algoritma, možnosti uporabe in izboljšav pa so opisane v [7].

1.4 Cilj diplomskega dela

V pričujočem delu želimo združiti tehnike modeliranja gensko regulatornih omrežij s tehnikami evolucijskih algoritmov ter dobljenimi eksperimentalnimi podatki in odkriti uporabno znanje, s pomočjo katerega lahko bodisi bolje opišemo odziv opazovanega sistema, ali vplivamo na odločitve pri načrtovanju nadaljnjih poizkusov.

Metodo bomo uporabili na primeru realizacije bistabilnega genskega stikala v sesalskih celicah, ki je sestavljeno iz množice umetno sestavljenih genskih konstruktov, ter s prisotnostjo enega izmed dveh zunanjih signalov omogoča izbiro med produkcijo dveh različnih proteinov, ob odsotnosti signalov pa ohrani trenutno stabilno stanje [1]. S pridobitvijo začetnih eksperimentalnih rezultatov prilagodimo parametre modela za napoved odziva s čim manjšim odstopanjem od izmerjenih meritev.

Pričakovani rezultati dela so:

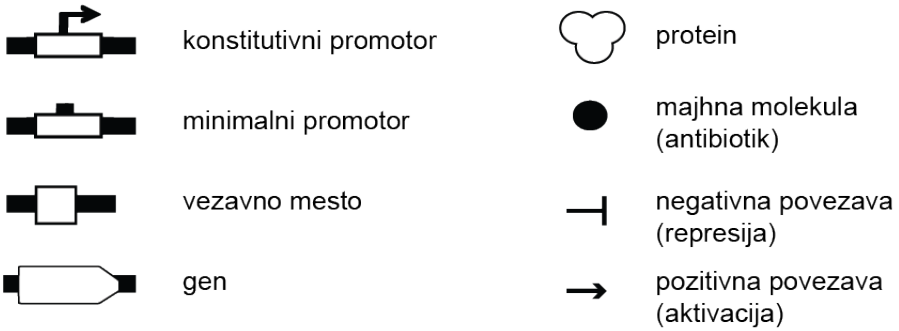
- vzpostavitev kvantitativnega stohastičnega modela bistabilnega genskega stikala z upoštevanjem kompromisa med natančnostjo in računsko učinkovitostjo,
- prilagoditev algoritma za stohastično simulacijo biokemijskih procesov (SSA) za učinkovito izvajanje na grafičnem procesorju ter iskanje parametrov za optimalno izkoriščenost računskih virov,
- implementacija optimizacijske metode za prilagoditev modela eksperimentalnim podatkom ter z rezultati pojasniti vzroke za rezultate meritev,
- identifikacija ključnih parametrov optimizacijske metode,
- identifikacija ključnih lastnosti sheme stikala, ki vodijo v želeni odziv sistema.

2 Metode

V pričujočem poglavju predstavimo računske in eksperimentalne metode, uporabljene v implementaciji sistema. Opisan je obravnavani sistem genskega regulatornega omrežja bistabilnega stikala s pozitivno povratno zanko. Zatem predstavimo matematične temelje modeliranja splošnih sistemov kemijskih reakcij, postopkov za primerjavo rezultatov modelov ali meritev, algoritma za iskanje optimalnih parametrov ter posebnosti ciljne arhitekture, za katero bo sistem implementiran.

2.1 Gensko regulatorno omrežje: bistabilno stikalo

Analiza genskih regulatornih omrežij živih organizmov je pokazala prisotnost motivov (vzorcev), ki se statistično pojavljajo dosti pogosteje, kot bi se v hipotetičnih, naključno zgrajenih omrežjih [4]. Skozi evolucijo se je tako oblikovala modularnost omrežij, kjer posamezni deli opravljajo določeno funkcijo. Za pravilno delovanje organizma ta pojav ni bistven (ekvivalentne funkcije bi v teoriji lahko opravljala naključno zgrajena omrežja). Ohranil se je zaradi manjšega potrebnega števila povezav [9], kar omrežje naredi manjše in manj potratno z vidika energije, potrebne za njegovo vzdrževanje.



Slika 2.1: Grafične podobe osnovnih elementov gensko regulatornih omrežij.

Pogost motiv, ki se pojavlja v organizmih, je regulacija dveh ali več tarčnih proteinov. Gre za natančen nadzor nad časovnim izražanjem različnih proteinov. Poenostavljeno si lahko to predstavljamo kot *stikalo* z dvema ali več *stanji*, ki se preklapljajo v odvisnosti od zunanjih signalov. Mednje sodijo naprimer svetloba ali prisotnost določenih molekul. Za organizme so tovrstna stikala med drugim pomembna za pravilen odziv na prehod med dnevom in nočjo, nadzorovano rast ali regulacijo metabolnih procesov. So tudi eden od mehanizmov, ki določajo usodo celice ob diferenciaciji, tj. procesu, kjer matična celica prevzame končno obliko in funkcijo.

V luči sintezne biologije smo si zamislili realizacijo bistabilnega stikala, s katerim bi nadzorovali izmenično produkcijo dveh terapevtskih proteinov v sesalskih celicah [1]. Stanje stikala, *ki je predstavljeno z izražanjem enega ali drugega proteina*, bi lahko nadzorovali s prisotnostjo dveh različnih antibiotikov. Potencialna uporabnost pristopa je zdravljenje bolezni v več fazah. Tako bi na primer ob vnetju tkiva celice v prisotnosti prvega antibiotika najprej proizvajale protein za odpravo vnetja, nato pa bi z dostavo drugega antibiotika zamenjali stanje celic ter sprožili produkcijo rastnega faktorja, ki bi pomagal pri regeneraciji tkiva.

2.1.1 Osnovni pojmi in definicije

V nadaljevanju se bomo sklicevali na osnovne elemente genskega omrežja, katerih grafične podobe so prikazane na sliki 2.1. Transkripcijski faktorji v omrežju opravljajo funkcijo regulacije svojega ali drugih genov. Transkripcijski faktorji s pozitivno regulacijo so aktivatorji, transkripcijski faktorji z negativno regulacijo pa represorji.

Pred genom se nahaja zaporedje *promotorja*, kamor se veže encim RNA polimeraza, ki sproži prepis DNA v molekulo mRNA. Ta se nato s pomočjo ribosomov prevede v zaporedje aminokislin, ki sestavljajo končni produkt - protein. Ločimo *minimalne in konstitutivne promotorje*, kjer prvi za izražanje gena potrebujejo prisotnost ustreznega aktivatorja, sicer do izražanja ne pride. Konstitutivni promotorji po drugi strani te omejitve nimajo in do prepisa prihaja tudi brez prisotnosti aktivatorjev.

Zaporedje DNA pred promotorjem, kamor se vežejo transkripcijski faktorji, imenujemo *vezavna mesta*. Ta so specifična za enoten del proteina, ki mu pravimo *DNA vezavna domena*. Tako se lahko na isto vezavno mesto vežejo različni proteini, če v svoji strukturi vsebujejo ustrezno vezavno domeno. Pri nekaterih proteinih se verjetnost vezave na DNA povečuje sorazmerno s količino že vezanega proteina (kooperativnost), kar je eden od razlogov za nelinearnost sistemov.

Funkcijo proteina lahko dodatno spreminjajo *majhne molekule*, ki se nanj neposredno vežejo. Tako pride do spremembe v proteinski strukturi, kar neposredno vodi v spremembo funkcije. Primer je izguba ali pridobitev sposobnosti vezave proteina na določeno vezavno mesto. Molekule, ki spreminjajo strukturo in funkcijo proteina, imenujemo *induktorji*, dostavi takih molekul v sistem pa indukcija.

Tako z ustreznim razporejanjem genov, promotorjev in vezavnih mest dosežemo predvidljivo delovanje sintetičnega genskega regulatornega omrežja, ki ga lahko nadzorujemo z dostavo induktorjev.

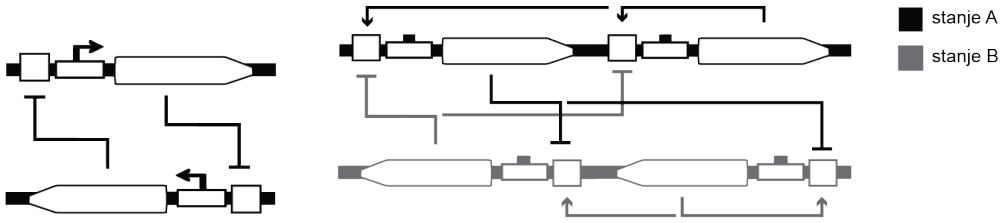
2.1.2 Logična shema bistabilnega stikala

Sintetična bistabilna stikala so že bila realizirana v različnih organizmih, tako v bakterijah [16], kot tudi v sesalskih celicah [29, 48]. Osnovni motiv delovanja stikala z dvema stanjema je prikazan na sliki 2.2a. Potreben pogoj za delovanje je prisotnost dveh različnih represorjev, ki medsebojno uravnavata transkripcijo drug drugega. Topologija v teoriji zagotavlja, da bo sistem v danem trenutku v enem izmed dveh možnih stanj, torej bo aktivna produkcija zgolj enega izmed dveh proteinov.

Teoretične študije z uporabo matematičnih modelov so pokazale, da tak sistem lahko opravlja funkcijo stikala le, če se izbrani represorji na ustrezna vezavna mesta vežejo kooperativno [8, 29], kar vodi v nelinearnost sistema. Ta lastnost v splošnem ne velja za vse transkripcijske faktorje, kar naredi delovanje takega omrežja odvisno od konkretne izbire represorskih proteinov.

Shema stikala, ki bi bila neodvisna od kooperativnosti regulacijskega proteina je bila predlagana v [36]. Ta predpostavlja, da protein lahko hkrati opravlja funkcijo aktivatorja in represorja, kar je v naravi težko najti.

Ena izmed ključnih idej projekta [1] je bila nadgradnja predlagane sheme v tako, ki bi zagotavljala robustnost delovanja, neodvisno od izbire regulacijskih elementov. Namesto da bi uporabili protein, ki opravlja vlogo tako aktivatorja kot represorja, smo uporabili vezavna mesta, za katere aktivatorji in represorji tekmujejo. Dodatno smo v sistem dodali pozitivno povratno zanko, ki ojači izražanje trenutnega stanja. Opisana nadgradnja v sistem vnese potrebno nelinearnost. Končna predlagana shema je prikazana na sliki 2.2b.



(a) (b)

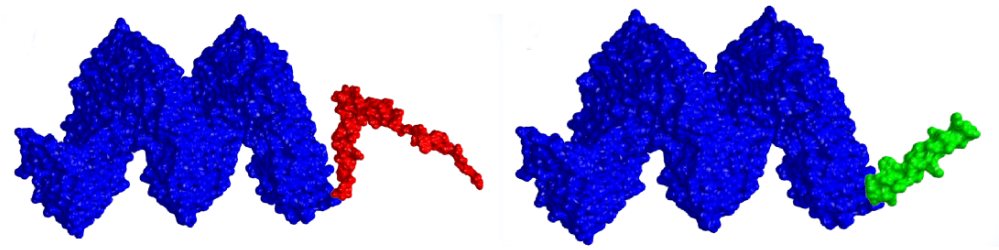
Slika 2.2: Shemi topologij bistabilnega stikala. a) Osnovna shema bistabilnega stikala z dvema recipročnima represorjema. b) Predlagana izboljšava osnovne sheme, ki poleg represorjev vsebuje še pozitivno povratno zanko, ki jo ustvarita aktivatorja.

2.1.3 Realizacija

V preostanku razdelka bo povzeta realizacija sistema bistabilnega stikala. Podrobnosti postopka presegajo obseg tega dela in so opisane v [1]. S pomočjo postopka, imenovanega transfekcija, je možno v sesalske celice vstaviti DNA v obliki plazmidov. Za vstavitev sintetičnih zaporedij, sestavljenih iz elementov, opisanih v prejšnjem razdelku, potrebujemo ustrezen vektor. Eni izmed najpogosteje uporabljanih vrst vektorjev za transfekcijo so plazmidi. Slednji so približno 10.000 baznih parov dolgi krožni fragmenti DNA. Ker so standardizirani, omogočajo enostavno manipulacijo v obliki vstavljanja poljubnih DNA zaporedij.

V študiji smo za realizacijo sistema stikala uporabili nedavno odkrite DNA vezavne proteine, efektorje TAL (angl. *Transcription activator like effector*). To so transkripcijski faktorji rastlinskih patogenov, ki prepoznajo specifično DNA zaporedje. Imajo značilno

strukturo, ki vsebuje ponavljajoče module, sestavljene iz 33-35 aminokislin. Vsak modul se veže na en nukleotid, pri čemer 12. in 13. aminokislina določata, kateri bazni par bo prepoznan. Z ustreznim sestavljanjem modulov lahko tako dosežemo vezavo na praktično katerokoli zaporedje DNA, kar nam omogoča pripravo zadostnega števila parov proteinov in pripadajočih vezavnih mest. Efektorji TAL so ortogonalni, saj jih lahko sestavljamo tako, da nimajo interakcij tako z genomom kot tudi drugimi proteini, kar so želene lastnosti proteinov za nadzorovano regulacijo.



Slika 2.3: Struktura TAL proteina s KRAB represorsko ali VP16 aktivatorsko domeno.

Poleg določitve vezavnega mesta lahko sestavljenemu TAL efektorju določimo tudi vpliv na intenzivnost transkripcije. Prek fuzije DNA vezavne domene TAL efektorja z efektorsko domeno lahko ustvarimo transkripcijske aktivatorje oziroma represorje, ki se vežejo na točno določena vezavna mesta. Uporabili smo VP16 aktivatorsko oziroma KRAB represorsko domeno za dosego želenega učinka. Tako lahko v principu ustvarimo aktivator in represor, ki se vežeta na (tekmujeta za) isto vezavno mesto in tako zadostimo pogojem iz razdelka 2.1.2. Struktura TAL proteina z obema vezavnima domenama je prikazana na sliki 2.3.

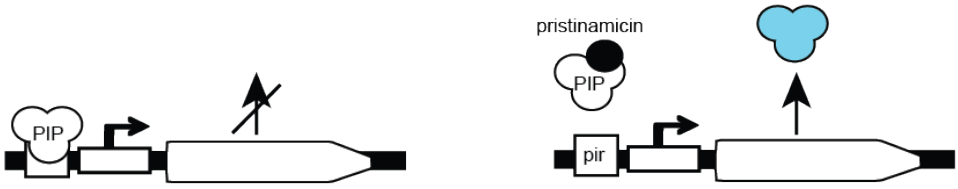
Za realizacijo sheme iz razdelka 2.1.2 smo ustvarili štiri transkripcijske faktorje (dva aktivatorja in dva represorja) ter dve vrsti vezavnih mest A in B, dva para aktivatorja in represorja za vsako stanje na sliki 2.2. V nadaljevanju bo uporabljena notacija

$$\text{TAL(vezavna domena):(efektorska domena)} \tag{2.1}$$

Zapis TALB:KRAB tako označuje TAL represor, ki prepoznava vezavno mesto tipa B, TALA:VP16 pa TAL aktivator, ki prepoznava vezavno mesto tipa A.

2.1.4 Inducibilni sistemi

Eden od možnih načinov nadzora sistema je uporaba majhnih induktorskih molekul (npr. antibiotikov), ki se vežejo na tarčni protein ter spremenijo njegovo strukturo. Uporabili smo pristinamicinski inducibilni protein (PIP) ter eritromicinski inducibilni protein (E), ki smo jih združili s KRAB represorsko domeno in tako ustvarili transkripcijska represorja PIP:KRAB in E:KRAB, ki prepoznata vezavni mesti tipa *pir* in tipa *etr*. Ob dodatku antibiotikov pristinamicina (PI) ali eritromicina (ER), ki se vežeta na ustrezni represorski protein, se slednji odveže z DNA ter omogoči transkripcijo, kot je prikazano na sliki 2.4.



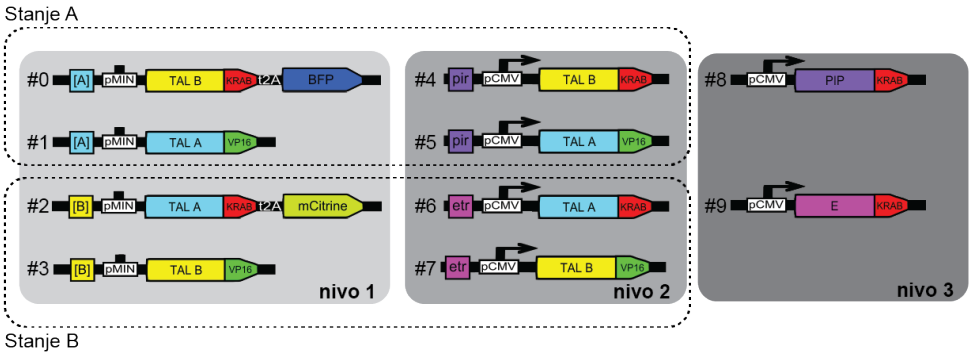
Slika 2.4: Princip delovanja inducibilnega sistema. Vezava majhne molekule (antibiotika) na tarčni protein spremeni njegovo strukturo ter negativno vpliva na sposobnost vezave. Prisotnost antibiotika tako neposredno omogoči transkripcijo.

2.1.5 Seznam plazmidov

Končni načrt sistema bistabilnega stikala, ki je logično predstavljen na sliki 2.2b, je sestavljen iz desetih plazmidov, ki jih v celice vstavimo s postopkom transfekcije.

Logično je mogoče načrt razdeliti na tri nivoje, kot je prikazano na sliki 2.5. *Prvi nivo* predstavlja predlagano shemo stikala z dvema paroma represorjev in aktivatorjev, kjer represor in aktivator nasprotnih stanj tekmujeta za ista vezavna mesta. Stanje A tako predstavlja par TALA:VP16 (aktivator stanja A) in TALB:KRAB (represor stanja B). Za večjo robustnost so v prvem nivoju uporabljeni minimalni promotorji. Da lahko spremljamo stanji sistema, smo konstruktom dodali dva fluorescentna proteina, modri fluorescentni protein (BFP) in rumeni fluorescentni protein (mCitrine), ki ju je mogoče spremljati z več biokemijskimi metodami in opravljata vlogo *poročevalskih proteinov*.

Drugi nivo predstavlja inducibilni sistem, ki ob dodatku pristinamicina ali eritromicina sproži produkcijo ustreznih TAL efektorjev. Ob tem se pripadajoči inducibilni pro-



Slika 2.5: Končna shema realizacije bistabilnega stikala s pozitivno povratno zanko. Stanji A in B označujeta izražanje označenih genov ter posledično proteinov ter se logično medsebojno izključujeta.

teini odvežejo z DNA. Tako na primer ob indukciji s pristinamicinom proteini PIP:KRAB izgubijo sposobnost vezave na DNA in sproži se produkcija TALA:VP16 ter TALB:KRAB, s čimer hkrati aktiviramo stanje A ter utišamo izražanje stanja B.

V *tretjem nivoju* se konstitutivno izražata transkripcijska faktorja PIP:KRAB in E:KRAB, ki zagotavljata, da sta ob odsotnosti obeh induktorjev inducibilna sistema (drugi nivo) za obe stanji zaprta.

2.2 Stohastična simulacija genskega regulatornega omrežja

Z vidika formalnega opisa lahko genska regulatorna omrežja opišemo kot sistem kemijskih reakcij. Ob poznavanju reakcij in njihovih verjetnosti je tako v teoriji možen natančen opis takega sistema [20]. Procesi v celicah so podvrženi spreminjajočemu okolju, različnosti znotraj populacije in naključnosti poteka kemijskih reakcij, ki slonijo na naključnemu trku med delci. Dolgo časa je veljalo prepričanje, da je šum v odzivu celic zgolj posledica genetske raznolikosti populacije ter sprememb v okolju. Tako naj bi se enake celice v enakih eksperimentalnih pogojih obnašale enako. Ta teza je bila kasneje ovržena, saj pri tem veliko vlogo igra tudi šum v poteku kemijskih reakcij. Naključnosti, kateri so podvržene reakcije znotraj celice, pravimo *notranji šum*, spreminjajočim se pogojem iz okolja, ki lahko za celico predstavljajo signal ter raznolikost znotraj populacije, pa *zunanji šum*. Verjetnost, da bosta dve naključno izbrani celici za opazovani sistem

ustvarili enak odziv, je tako zelo majhna.

Principi modeliranja sistema kemijskih reakcij izhajajo iz *kemijske glavne enačbe* (angl. *Chemical master equation*), ki opazovanim kemijskim reakcijam priredi ustrezne verjetnosti za sprožitev v naslednjem opazovanem infinitezimalnem časovnem koraku. Ustrezno s potekom reakcij se spreminjajo količine opazovanih kemijskih zvrsti, ki so predstavljene z naključnimi celoštevilskimi spremenljivkami. Posamezna reakcija je opisana s pravilom (formulo) oblike

$$A + B \xrightarrow{k} C, \tag{2.2}$$

kjer A in B predstavljata število molekul reaktantov, C pa število molekul produktov. Števila molekul določene kemijske zvrsti so tako celoštevilske spremenljivke in skupaj s parametrom k določajo verjetnost, da se reakcija izvede v naslednjem časovnem koraku. Ta je premo sorazmerna s količinami reaktantov.

Opazujemo sistem kemijskih reakcij v prostoru z volumnom Ω, ki vsebuje N opazovanih kemijskih zvrsti $x_1, ..., x_N$. Te so med seboj povezane v sistemu M kemijskih reakcij $R_1, ..., R_M$, pri čemer predpostavljamo uniformno porazdelitev kemijskih zvrsti v prostoru in termično ravnovesje. Reakcije so torej podvržene konstantni temperaturi, ki povzroči premik molekul zaradi toplotne energije.

Naj bo $\vec{\boldsymbol{X}}(t) = [x_1(t), ..., x_N(t)]^T$ vektor stanj sistema, ki opisuje število molekul kemijske zvrsti x_i v danem časovnem trenutku. Reakcije, ki spreminjajo vektor $\vec{\boldsymbol{X}}(t)$ so z vidika reaktantov lahko monomolekularne ali bimolekularne. Seznam reakcij in kemijskih zvrsti lahko preuredimo v matriko velikosti $M \times N$, kjer stolpci predstavljajo kemijske zvrsti, vrstice pa spremembe stanj za posamezno reakcijo. Dobljeno *stehiometrično matriko* lahko obravnavamo kot matriko prehajanja stanj končnega avtomata:

$$S_{M \times N} = [s_{ij}] \tag{2.3}$$

kjer s_{ij} predstavlja spremembo kemijske zvrsti x_i, ki jo povzroči reakcija R_j. Vsaka reakcija tako definira prehod iz trenutnega stanja v $\vec{\boldsymbol{X}}(t+1) = \vec{\boldsymbol{X}}(t) + s_j$, pri čemer je povezana s funkcijo *nagnjenosti* $w_j(x)$, ki je odvisna od trenutnega stanja $X(t)$ in definira verjetnost, da se bo reakcija R_j izvedla v časovnem intervalu $[t, t+dt)$. Osnovne vrste reakcij in pripadajoče funkcije nagnjenosti so podane v tabeli 2.1.

Vrsta reakcije	Formula	Funkcija nagnjenosti w
monomolekularna	$x_i \rightharpoonup$ produkti	$Cx_i, C = c_i$
bimolekularna	$x_i + x_j \rightharpoonup$ produkti	$Cx_ix_j, C = c_i/\Omega$
bimolekularna	$x_i + x_i \rightharpoonup$ produkti	$Cx_i(x_i - 1), C = 2c_i/\Omega$

Tabela 2.1: Osnovne vrste kemijskih reakcij ter pripadajoče funkcije nagnjenosti.

2.2.1 Kemijska glavna enačba

V danem opisu konteksta kemijska glavna enačba opisuje časovno odvisnost verjetnosti, da je sistem v stanju $\vec{\boldsymbol{X}}(t) = \vec{\boldsymbol{x}}$. Opis ustreza Markovski verigi (naslednje stanje je odvisno samo od prejšnjega). Naj bo sistem v stanju $\vec{\boldsymbol{x}}$ v času t. V okviru ocene napake reda velikost $O(dt^2)$ veljajo naslednje trditve:

- Verjetnost za sprožitev dane reakcije R_j natančno enkrat v časovnem intervalu $[t, t + dt)$ je enaka $w_j(\vec{\boldsymbol{x}})dt$.
- Verjetnost, da se ne sproži nobena od reakcij v časovnem intervalu $[t, t + dt)$, je enaka $1 - \sum_j (w_j(\vec{\boldsymbol{x}}))dt$.
- Verjetnost, da se v časovnem intervalu$[t, t+dt)$ sproži več kot ena reakcija je enaka nič.

Kemijska glavna enačba je podana z izrazom

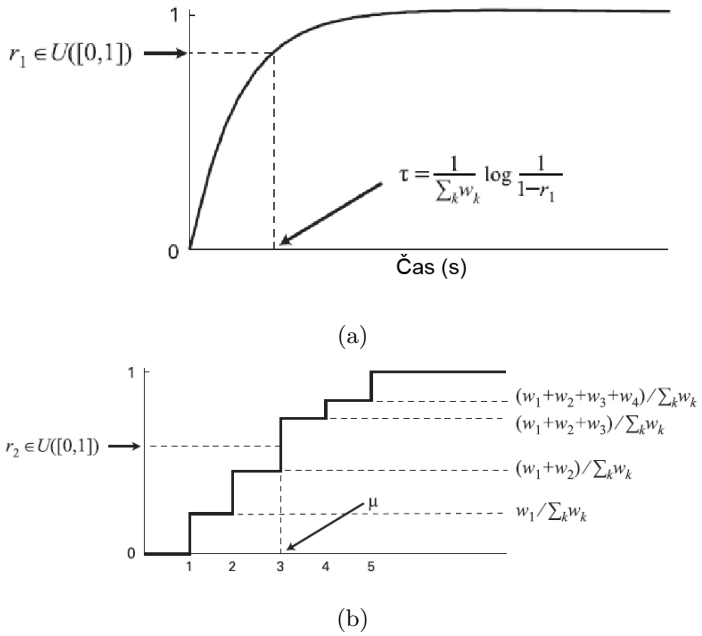
$$\frac{P(\vec{\boldsymbol{x}}, t)}{dt} = \sum_j^M w_j(\vec{\boldsymbol{x}} - s_j)P(\vec{\boldsymbol{x}} - s_j, t) - w_j(\vec{\boldsymbol{x}})P(\vec{\boldsymbol{x}}, t), \tag{2.4}$$

ki predstavlja spremembo verjetnosti stanja $\vec{\boldsymbol{x}}$ v času t. Prvi člen predstavlja verjetnosti vseh stanj, ki preko matrike S lahko vodijo v stanje $\vec{\boldsymbol{x}}$, drugi člen pa verjetnost, da stanje zapustimo.

Z drugimi besedami, kemijska glavna enačba izračuna odklon vrednosti naključne spremenljivke od njene povprečne vrednosti. Ko števila molekul postanejo dovolj velika, postanejo odkloni relativno majhni in jih lahko zanemarimo. V limiti odziv sistema kemijskih reakcij torej konvergira proti enolično določeni krivulji v skladu s centralnim limitnim izrekom, kar pa lahko določimo s prevedbo na deterministični sistem diferencialnih enačb [20].

Lastnost predstavlja kompromis med natančnostjo opisa in računsko zahtevnostjo. Za majhne sisteme v majhnem volumnu je tako bolj primeren stohastični opis, kjer spremljamo vsako molekulo posebej, saj majhna sprememba na nivoju posamezne naključne spremenljivke lahko pomeni veliko spremembo z vidika celotnega sistema. Po drugi strani lahko večje sisteme z zadovoljivo natančnostjo enolično zapišemo kot sistem diferencialnih enačb, kar omogoča manjšo računsko zahtevnost in možnost analitične razčlenitve ali rešitve diferencialnih enačb.

V našem primeru je osrednja lastnost sistema omejeno število vezavnih mest ter tekmovanje represorja in aktivatorja za vezavna mesta pred pripadajočim promotorjem. Tekom eksperimentalnega dela smo opazili, da je odziv stikala zelo občutljiv (z vidika formalnega opisa torej relativno majhen) na relativna razmerja količin konstruktov zaradi česar bomo izbrali stohastični način simulacije, ki bo v nadaljevanju natančneje opisan.



Slika 2.6: (a) Graf zvezno porazdelitve spremenljivke časa naslednje reakcije τ. (b) Graf diskretne porazdelitve spremenljivke indeksa naslednje reakcije μ.

2.2.2 Gillespijev algoritem (SSA)

Kemijska glavna enačba v splošnem nima analitične rešitve, zato v praksi uporabljamo aproksimacijske algoritme. Eden izmed takih je Gillespiev algoritem [18], poznan tudi

pod kratico SSA (angl. *Stochastic simulation algorithm*).

Vsak korak stohastične simulacije se prične v času t in v stanju $\vec{\boldsymbol{X}}(t) = \vec{\boldsymbol{x}}$ ter sestoji iz treh glavnih korakov:

1. Izračunaj čas do naslednje reakcije.
2. Na podlagi rezultata posamezne funkcije nagnjenosti s seznama izberi reakcijo, ki se bo izvedla.
3. Ažuriraj čas in stanje sistema kot odraz vpliva korakov 1 in 2.

Za vsako reakcijo R_j v množici $R_1, ..., R_M$ določimo naključno spremenljivko τ_j, ki označuje čas do naslednje sprožitve reakcije R_j. Ključno dejstvo je, da je τ_j ustreza eksponentni porazdelitvi s parametrom w_j. Definirajmo dve dodatni naključni spremenljivki, eno zvezno ter eno diskretno:

$$\tau = \min_j \tau_j \text{ (čas do naslednje reakcije)}, \tag{2.5}$$

$$\mu = \arg\min_j \tau_j \text{ (indeks naslednje reakcije)}, \tag{2.6}$$

pri čemer je τ eksponentno porazdeljena s parametrom $\sum_j w_j$, μ pa ustreza diskretni porazdelitvi $P(\mu = j) = w_j / \sum_i^M w_i$. Grafa porazdelitev sta prikazana na sliki 2.6.

Potek algoritma SSA (pot med stanji avtomata) je neposredno odvisen od naključnih spremenljivk τ in μ. Psevdokoda algoritma je prikazana v kodi 2.1.

Opisani pristop je natančen v smislu izračuna porazdelitve vsake naključne spremenljivke v skladu z rešitvijo pripadajoče kemijske glavne enačbe. Posamezni zagon simulacije izračuna zgolj eno od možnih trajektorij, zato ga je potrebno ponoviti večkrat, da dobimo ustrezno statistiko za posamezno kemijsko zvrst.

2.3 Aproksimacija stohastičnih procesov

Z napredkom v razumevanju procesov v celici lahko postane natančno modeliranje celotnega sistema zapleteno. Potreben je kompromis med zadostno kompleksnostjo modela, da lahko iz rezultatov razberemo koristno znanje, a dovolj preprost, da ostane računsko obvladljiv.

Velika količina dejavnikov vodi v dolg in kompleksen matematični opis sistema, ki z vključevanjem naključnosti lahko privede v kombinatorično eksplozijo, formalna analiza

```
while(t < T):
   t := 0;
   X := x;

  for each j:
    w[j] := w(X,R[j]);

  tau := vzorec iz porazdelitve tau(w);
  mi :=  vzorec iz porazdelitve r(w);

  t := t + tau;
  X := X + S[mi];
```

Koda 2.1: Potek algoritma za stohastično simulacijo sistema kemijskih reakcij.

takega sistema pa postane neobvladljiva. Zato se poslužujemo tako imenovane redukcije modelov, s katero poizkušamo zmanjšati število kemijskih zvrsti in reakcij. To lahko počnemo z združevanjem več reakcij ali kemijskih zvrsti v podmnožico le teh, ali pa zanemarjanjem reakcij in zvrsti, ki na odziv sistema bistveno ne vplivajo. Pri tem se lahko poslužujemo različnih nivojev abstrakcij, s katerimi opisujemo procese. Cilj postopka je zmanjšati število spremenljivk v modelu, a ohraniti ključne in zanimive lastnosti odziva sistema.

Da bi lahko ovrednotili rezultate minimizacije, potrebujemo kriterije oz. metrike, s katerimi lahko določimo razdaljo med dvema odzivoma. Znana primera sta srednja kvadratna napaka (angl. *mean squared error*), ki jo uporabljamo za primerjavo dveh funkcij iste spremenljivke ter Kullback-Leiblerjeva divergenca, s katero primerjamo dve porazdelitvi opazovane naključne spremenljivke. Za primerjavo dveh stohastičnih biokemijskih procesov bomo v nadaljevanju opisali razred *Wassersteinovih psevdometrik* (angl. *Wasserstein pseudometrics*), ki temelji na primerjavi porazdelitev spremenljivk, v splošnem pridobljenih z različnimi načini merjenja. V našem primeru bomo uporabili postopek opisan v [20].

2.3.1 Izpeljava verjetnostnih porazdelitev

Stohastična omrežja reakcij so množice naključnih spremenljivk, definirane v prostoru $(\mathcal{O}, \mathcal{F}, \mathcal{P})$, kjer $\mathcal{O}$ predstavlja prostor vzorcev, $\mathcal{P}$ verjetnostno porazdelitev, $\mathcal{F}$ pa domeno (definicijsko območje) verjetnostne porazdelitve. Ta opisuje množico trajektorij, med katerimi lahko razlikujemo z izbranim načinom merjenja.

Rezultat simulacije omrežja reakcij je trajektorija, ki se spreminja s časom. Ta je definirana s pravili končnega avtomata, ki za vsako diskretno časovno točko $t > 0$ določi notranje stanje $\vec{\boldsymbol{X}}(t) = \vec{\boldsymbol{x}}$ in vrednost izhodne funkcije $h(\vec{\boldsymbol{x}})\ \epsilon\ Y$. Posamezna instanca trajektorije tako podaja funkcijo $\omega : \mathbb{R}^+ \to Y$, ki določa izhodno vrednost za vsako časovno točko. Prostor vzorcev $\mathcal{O}$ je tako množica vseh trajektorij ω.

Radi bi definirali razdaljo d med dvema vzorcema v prostoru $\mathcal{O}$, tako da je $d : \mathcal{O} \times \mathcal{O} \to \mathbb{R}+$ vedno nenegativna ter zadovoljuje trikotniško neenakost $d(\omega, \phi) + d(\phi, \mu) \geq d(\omega, \mu)$. V praksi zahtevamo, da mora biti izmerljiva z obzirom na domeno $\mathcal{F}$.

Stohastični proces je metoda, ki definira verjetnostno porazdelitev neodvisne spremenljivke v prostoru vzorcev $\mathcal{O}$. Verjetnostno porazdelitev si lahko predstavljamo kot histogram, ki je rezultat n simulacij stohastičnega procesa, kjer $n \to \infty$. Za definicijo razdalje med dvema stohastičnima procesoma, zadošča razdalja med dobljenima verjetnostnima porazdelitvama, ki sta posledici procesov.

2.3.2 Razred Wassersteinovih psevdometrik

Zanimajo nas metrike oblike $d(\omega, \mu) = |Z(\omega) – Z(\mu)|$, kjer je $Z : \mathcal{O} \to \mathbb{R}$ poljubna spremenljivka, ki opisuje trajektorijo ω. Z imenujemo poročevalska spremenljivka, saj opisuje opazovani zanimiv vidik dobljene trajektorije, ki jo lahko izmerimo. V splošnem lahko za izbiro funkcije Z izberemo katerokoli funkcijo nad $\mathcal{O}$. Za boljšo primerjavo lahko psevdometriko izračunamo tudi za več različnih definicij poročevalskih spremenljivk. Primeri definicij poročevalskih spremenljivk:

- Z lahko predstavlja količino proteina v času t:

$$Z(\omega) = \omega(t), \tag{2.7}$$

- Z lahko predstavlja prvo časovno točko, ob kateri je prisotnih vsaj N molekul proteina:

$$Z(\omega) = min(\omega^{-1}(N)), \tag{2.8}$$

- Z lahko predstavlja povprečno količino proteina v časovnem intervalu $[t_1, t_2]$:

$$Z(\omega) = 1/(t_2 - t_1) \cdot \int_{t_1}^{t_2} \omega(t)dt, \tag{2.9}$$

- Z lahko predstavlja pojav več od N molekul proteina v času celotne simulacije:

$$Z(\omega) = \begin{cases} 1 & \exists t, \omega(t) \geq N, \\ 0 & \text{sicer.} \end{cases}$$

Vsaka verjetnostna porazdelitev definira funkcijo kumulativne porazdelitve spremenljivke Z:

$$F_{\mathcal{P},Z} = P(Z < z). \tag{2.10}$$

Inverz funkcije kumulativne porazdelitve je:

$$F^{-1}_{\mathcal{P},Z} = \inf\left(z : F_{\mathcal{P},Z}(z) \geq y|\right). \tag{2.11}$$

Naj bosta P_1 in P_2 verjetnostni porazdelitvi nad $\mathcal{O}$. Z uporabo inverzov $F_{\mathcal{P}_1,Z}$ in $F_{\mathcal{P}_2,Z}$ lahko definiramo naslednjo psevdometriko za kvantifikacijo medsebojne razdalje.

Definicija. Za vsak $p > 0$, je Wassersteinova psevdometrika W_d^p med dvema verjetnostnima porazdelitvama $\mathcal{P}_1, \mathcal{P}_2$ nad prostorom vzorcev $\mathcal{O}$ definirana z:

$$W_d^p(\mathcal{P}_1, \mathcal{P}_2) = (\int_0^1 |F^{-1}_{\mathcal{P}_1,Z}(y) - F^{-1}_{\mathcal{P}_2,Z}(y)|^p \; dy)^{1/p}. \tag{2.12}$$

2.3.3 Algoritem za izračun Wassersteinovih psevdometrik

Markovski proces, ki izhaja iz aproksimacije biokemijskega procesa ima v splošnem lahko neskončno mnogo število stanj, zaradi česar je izračun razdalje med dvema porazdelitvama težek problem. Verjetnostna porazdelitev, ki je rezultat takega procesa, je v splošnem zelo kompleksna in je ni mogoče izračunati analitično. Ker je tudi Markovski proces aproksimacija dejanskega procesa, problem rešimo z upoštevanjem empirične porazdelitve $\mathcal{P}_i$, ki jo pridobimo z vzorčenjem n neodvisnih vzorcev iz $\mathcal{O}$ z obzirom na $\mathcal{P}$.

Z izračunom Wassersteinove psevdometrike nad dvema empiričnima porazdelitvama ocenimo razdaljo med dejanskima verjetnostnima porazdelitvama. V splošnem lahko

podatke pridobimo iz različnih virov, bodisi z izvajanjem dejanskih eksperimentov ali zagonom stohastičnega simulacijskega algoritma, opisanega v razdelku 2.2.2.

Enačbo lahko posplošimo za diskretne empirične porazdelitve. Naj bosta $\mathcal{P}_1$ in $\mathcal{P}_2$ neznani verjetnostni porazdelitvi, pri čemer vzorčimo n neodvisnih vzorcev $\omega_1, \omega_2, ..., \omega_n$ iz $\mathcal{P}_1$ ter $l \cdot n$ neodvisnih vzorcev $\mu_1, \mu_2, ..., \mu_{l \cdot n}$ iz $\mathcal{P}_2$, kjer $l \, \epsilon \, \mathbb{N}$. Empirični kumulativni funkciji porazdelitve sta

$$F_{\mathcal{P}'_{1,n},Z}(z) = \frac{1}{n}|\omega : Z(\omega) < z|, \tag{2.13}$$

$$F_{\mathcal{P}'_{2,l \cdot n},Zn}(z) = \frac{1}{l \cdot n}|\omega : Z(\omega) < z|, \tag{2.14}$$

kjer $\mathcal{P}'_{1,n}$ in $\mathcal{P}'_{2,l \cdot n}$ označujeta vzorčeni porazdelitvi neznanih porazdelitev $\mathcal{P}_1$ in $\mathcal{P}_2$.

Vzorce razvrstimo tako da $Z(\omega_1) \leq Z(\omega_2), ... \leq Z(\omega_n)$ in $Z(\mu_1) \leq Z(\mu_2), ... \leq Z(\mu_{l \cdot n})$. Inverz empiričnih kumulativnih funkcij porazdelitve je tako:

$$F_{\mathcal{P}_n,Z}(y) = Z(\omega_i), \tag{2.15}$$

kjer $(i-1)/n < y < i/n$.

Izrek. Naj bo matematično upanje $E_{\mathcal{P}_i}(|Z|) < \infty$ za $i = 1, 2$. Wassersteinova psevdometrika $W_d^1(\mathcal{P}_1, \mathcal{P}_2)$ med dvema verjetnostnima porazdelitvama $\mathcal{P}_1$ in $\mathcal{P}_2$ z upoštevanjem funkcije psevdorazdalje $d(\omega, \mu) = |Z(\omega) - Z(\mu)|$ nad $\mathcal{O}$ je enaka

$$W_d^1(\mathcal{P}_1, \mathcal{P}_2) = \lim_{n \to \infty} \frac{1}{l \cdot n} \sum_{1}^{l \cdot n} |Z(\omega_{\lceil i/l \cdot n \rceil} - Z(\mu_i)|. \tag{2.16}$$

Časovna odvisnost algoritma je odvisna od razvrščanja vzorcev in je enaka $O(l \cdot n \; log(l \cdot n))$. Ker v praksi generiranje vzorcev vključuje bodisi izvajanje fizičnih eksperimentov ali izračun velike količine simulacij, je čas za izračun Wassersteinove psevdometrike in primerjavo dveh modelov bistveno krajši in odvisen od časa jemanja vzorcev.

2.4 Osnovni model bistabilnega stikala s pozitivno povratno zanko

V nadaljevanju bo opisan osnovni model bistabilnega stikala s pozitivno povratno zanko, ki je shematsko prikazan na sliki 2.5. Osnovni model služi kot iztočnica za postopek minimizacije, ki je podrobneje opisan v razdelku 3.3. Izpostavljene bodo ključne posebnosti modela, celoten seznam reakcij in kemijskih zvrsti je podan v prilogah A.1 in A.2.

2.4.1 Tekmovanje aktivatorjev in represorjev za vezavna mesta

Osnovna ideja stikala temelji na vezavi dveh različnih proteinov na istis tip vezavnega mesta, s čimer zadostimo pogoju, potrebnemu za doseg dveh stabilnih stanj z uporabo nekooperativnih elementov [36]. Za eksperimentalne poskuse smo določili 10 vezavnih mest pred vsakim minimalnim promotorjem, pri čemer je v vsakem trenutku mesto lahko bodisi zasedeno s proteinom (aktivatorjem ali represorjem) bodisi prosto. Število predstavlja kompromis med učinkovitostjo delovanja in kompleksnostjo konstrukta [1]. Vezavna mesta in njihovo trenutno stanje zasedenosti imenujemo *stanje promotorja*.

Prehajanje med stanji promotorja si lahko predstavljamo kot končni avtomat, ki je prikazan na sliki 2.7a. Ker bi upoštevanje vseh kombinacij stanj pri desetih vezavnih mestih ter prehodi med njimi povzročilo kombinatorično eksplozijo, uporabimo ključno posplošitev. *Namesto desetih vezavnih mest uporabimo abstrakcijo zasedenosti, kjer z dvema hipotetičnima vezavnima mestoma predstavimo šest možnih posplošitev za stanje promotorja, ki prehajajo v odvisnosi od konstant vezave TAL proteina k_a, k_b ter konstantama disociacije k_{da}, k_{db}.*

Eksperimentalni rezultati so pokazali učinkovitejše delovanje represorskega proteina v primerjavi z aktivatorskim, zaradi česar vpeljemo konstanto asimetričnosti $k_{\mathrm{KRAB}} > 1$. Konstanta predvideva intenzivnejšo vezavo represorskih proteinov[1], ter je uporabljena za prilagoditev modela začetnim meritvam za tekmovanje med aktivatorjem in represorjem.

Primer. Reakcije prehajanja stanj za konstrukte z vezavnimi mesti tipa A:

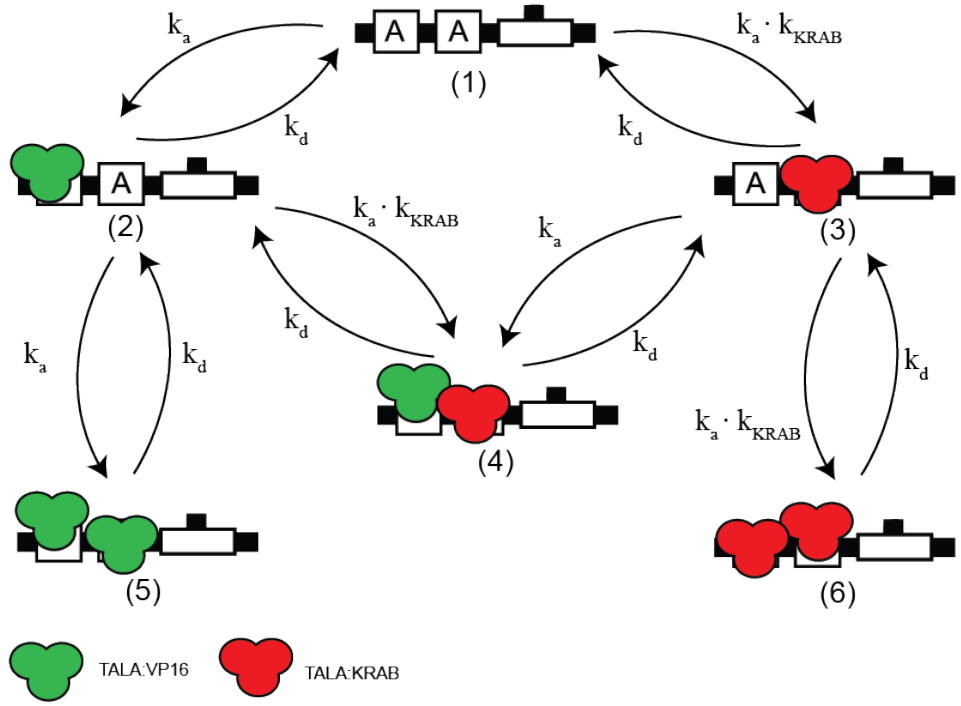
$$[P_{min\#i}] + [T_{AV}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}}}{\rightleftharpoons}} [T_{AV}\text{-}P_{min\#i}] \tag{2.17}$$

$$[P_{min\#i}] + [T_{AK}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}}\cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{AK}\text{-}P_{min\#i}] \tag{2.18}$$

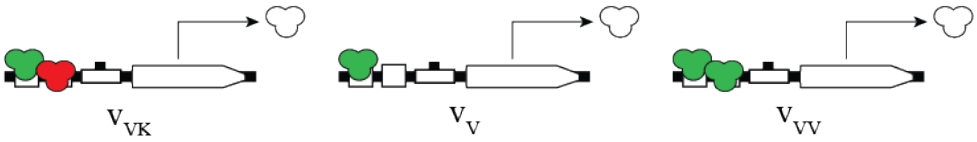
$$[T_{AK}\text{-}P_{min\#i}] + [T_{AK}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}}\cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{AK}\text{-}T_{AK}\text{-}P_{min\#i}], \tag{2.19}$$

$$[T_{AV}\text{-}P_{min\#i}] + [T_{AK}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}}\cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{AV}\text{-}T_{AK}\text{-}P_{min\#i}], \tag{2.20}$$

[1] V resnici je verjetnost vezave TAL proteinov odvisna le od vezavne domene, torej za aktivator in represor z enako vezavno domeno obstaja enaka verjetnost vezave. Predpostavka je ključna pri posplošitvi predstavitve desetih vezavnih mest na dve hipotetični vezavni mesti ter omogoča prilagoditev modela začetnim meritvam. Do navidezne razlike v intenzivnosti vezave pride zaradi različnih biokemijskih mehanizmov aktivatorja in represorja.



(a)



(b)

Slika 2.7: Opis minimalnega promotorja kot končnega avtomata. (a) Prikaz stanj minimalnega promotorja v prvem nivoju v obliki končnega avtomata. Opis je analogen za obe strani stikala, tako za konstrukte z vezavnimi mesti tipa A, kot konstrukti z vezavnimi mesti tipa B. (b) Intenzivnost transkripcije v odvisnosti od stanja promotorja, pri čemer velja $v_{\mathrm{VK}} < v_{\mathrm{V}} < v_{\mathrm{VV}}$.

$$[T_{AV}\_P_{min\#i}] + [T_{AV}] \underset{k_{da}}{\overset{k_a}{\rightleftharpoons}} [T_{AV}\_T_{AV}\_P_{min\#i}]. \tag{2.21}$$

$$i \,\epsilon\, \{0, 1\};$$

2.4.2 Transkripcija

Za sprožitev transkripcije pri minimalnih promotorji je nujna prisotnost aktivatorjev. Transkripcija je torej možna v treh stanjih, ki vključujejo aktivator, prikazanimi na sliki 2.7b. Pri tem je transkripcija najintenzivnejša ob polno aktiviranemu promotorju, najmanj pa pri kombinaciji zasedenosti z aktivatorji in represorji, kar so potrdili tudi eksperimentalni rezultati (podatki niso prikazani).

Primer. Transkripcija v odvisnosti od zasedenosti promotorja v konstruktu #0:

$$[T_{AV}\_P_{min\#0}] \overset{v_V}{\longrightarrow} [T_{AV}\_P_{min\#0}] + [T_{BK}] + [\mathrm{BFP}], \tag{2.22}$$

$$[T_{AV}\_T_{AK}\_P_{min\#0}] \overset{v_{VK}}{\longrightarrow} [T_{AV}\_T_{AK}\_P_{min\#0}] + [T_{BK}] + [\mathrm{BFP}], \tag{2.23}$$

$$[T_{AV}\_T_{AV}\_P_{min\#0}] \overset{v_{VV}}{\longrightarrow} [T_{AV}\_T_{AV}\_P_{min\#0}] + [T_{BK}] + [\mathrm{BFP}]. \tag{2.24}$$

2.4.3 Inducibilni sistem

Inducibilni sistem deluje na podlagi pristinamicinskega inducibilnega proteina (PIP) ter eritromicinskega inducibilnega proteina (E). Oba smo združili s KRAB represorsko domeno, s čimer dobita represijski učinek. Ker se izražata konstitutivno, ob odsotnosti zunanjih signalov oba reprimirata ekspresijo transkripcijskih faktorjev v nivoju 2. Ob prisotnosti signala v obliki antibiotikov pristinamicina (PI) oziroma eritromicina (ER) pa izgubita funkcijo vezave, ter sprožita transkripcijo tarčnih transkripcijskih faktorjev.

Primer. Ob dodatku pristinamicina se sprosti transkripcija proteinov TALA:VP16 in TALB:KRAB, kar povzroči hkratno aktivacijo stanja A ter represijo stanja B.

$$[\mathrm{PIP}_K\_P_{\mathrm{CMV}\#i}] + [PI] \overset{k_1}{\longrightarrow} [P_{\mathrm{CMV}\#i}] + [PI\_\mathrm{PIP}_K], \tag{2.25}$$

$$i \,\epsilon\, \{4, 5\};$$

$$[P_{\mathrm{CMV}\#4}] \overset{v_{\mathrm{CMV}}}{\longrightarrow} [P_{\mathrm{CMV}\#4}] + [T_{BK}], \tag{2.26}$$

$$[P_{\mathrm{CMV}\#5}] \overset{v_{\mathrm{CMV}}}{\longrightarrow} [P_{\mathrm{CMV}\#5}] + [T_{AV}]. \tag{2.27}$$

2.4.4 Degradacija

Proteini TAL, poročevalski proteini ter inducibilni proteini se v celici po določenem času razgradijo. Antibiotiki so del hranilnega medija, v kateremu gojimo celice. Število molekul antibiotika je neprimerljivo večje s številom proteinov v posamezni celici, zato zanje ne upoštevamo razpada. Konstante degradacije so različne med proteini TAL ter poročevalskimi proteini, kar lahko bistveno vpliva na odziv sistema.

Primer. Degradacija proteina TALA:VP16 ter poročevalca BFP.

$$[T_{AV}] \xrightarrow{d_{\mathrm{T}}} \emptyset, \tag{2.28}$$

$$[\mathrm{BFP}] \xrightarrow{d_{\mathrm{R}}} \emptyset. \tag{2.29}$$

2.5 Parametri reakcij

Verjetnost sprožitve posamezne reakcije je odvisna od količine zvrsti, ki nastopajo kot reaktanti ter parametra pogostosti reakcije. Slednjih ne poznamo natančno, saj so reakcije odvisne od velikega števila faktorjev, zato bi morali tovrstne meritve opraviti za točno določen ciljni organizem ob točno določenih eksperimentalnih pogojih.

Z optimizacijsko metodo želimo poiskati take parametre, ki bodo najbolje ustrezali dobljenim eksperimentalnim meritvam. Da pa bi ohranili biološki smisel, se opremo na literaturo, kjer najdemo referenčne vrednosti (časovne okvire) posameznih vrst reakcij (tabela 2.2). Sklepamo, da se reakcije, ki potekajo v manjših časovnih okvirih, izvajajo pogosteje, ter imajo posledično večjo vrednost pogostosti. *Če optimizacijsko metodo omejimo tako, da se lahko parametri spreminjajo le znotraj določenih meja, lahko dobimo natančen kvantitativni model sistema in ohranimo parametre pogostosti reakcij biološko smiselne.* V nadaljevanju bo vektor referečnih vrednosti označen s $\vec{\boldsymbol{c}}$.

2.6 Eksperimentalne meritve

Za potrebe preliminarnih eksperimentalnih meritev stikala smo omenjena poročevalska proteina (BFP in mCitrine) začasno zamenjali z encimom luciferazo[2]. Prisotna je v veliko organizmih, med katerimi je najbolj znana žuželka kresnica (*Photuris lucicrescens*). Je

[2]Encim luciferaza katalizira pretvorbo substrata luciferina v oksiluciferin, ki se nahaja v elektronsko vzbujenem stanju. Ob vračanju oksiluciferina v osnovno stanje se sprosti foton svetlobe.

Proces	Red velikosti	Parameter	Ref. vrednost ($\vec{c}$)	Vir
Čas trankripcije gena in čas translacije proteina	60 min	v_{CMV}	10^{-5}	[3]
		v_{V}	10^{-4}	[3]
		v_{VK}	10^{-6}	[3]
		v_{VV}	10^{-4}	[3]
Vezava majhne molekule na protein	1 s	k_{PI}	1.0	[3]
		k_{ER}	1.0	[3]
Vezava trankripcijskega faktorja na DNA	1 s	k_{a}, k_{b}	1.0	[3]
		k_{I}	1.0	[3]
Čas vezanega transkripcijskega faktorja na DNA	20 min	k_{da}, k_{db}	10^{-4}	[17]
		k_{dI}	10^{-4}	[17]
Degradacija proteina	10 h	d_{T}	10^{-5}	[26]
		d_{I}	10^{-5}	[26]
		d_{R}	10^{-5}	[26]

Tabela 2.2: Vrste parametrov reakcij ter pripadajoči časovni okviri in začetne vrednosti.

komercialno dobavljiva, ter pogosto uporabljana v laboratorijskih poizkusih. Njen učinek je mogoče enostavno kvantificirati z uporabo luminometra, saj je intenziteta oddane svetlobe proporcionalna količini encima v vzorcu.

V študiji se bomo osredotočili na eksperiment, v katerem preizkušamo sposobnost preklopa stikala. Ob tem celice, ki vsebujejo stikalo, sprva dva dni hranimo z medijem, ki vsebuje antibiotik enega stanja, nato pa po dveh dneh zamenjamo medij, ki vsebuje drugi antibiotik. Pričakujemo, da se najprej vzpostavi prvo stanje, po menjavi medija pa drugo. Zaradi omejene življenjske dobe celic in njihove občutljivosti na stres smo izvedli pet meritev. Prvi vzorec vzamemo ob času preklopa (2. dan) nato pa vzorčimo vsakih 48 ur do desetega dneva. V nadaljevanju bomo odziv modela (nivoja obeh poročevalskih proteinov) s spreminjanjem parametrov prilagodili nivojem petih vzorcev ter s tem pridobili natančen, kvantitativni model, katerega rezultati so interpretirani v razdelku 3.5.

2.7 Optimizacija parametrov reakcij

V domeni modeliranja bioloških sistemov pogosto poznamo procese (reakcije), ki so ključni za delovanje opazovanega sistema. Eden izmed ključnih problemov pri poizkusu matematičnega modeliranja je kvantitativna določitev vrednosti parametrov. Med razloge za to sodijo raznolikost organizmov, variacija v eksperimentalnih pogojih, kompleksnost meritve ali težavnost opazovanja posameznega parametra. Zaradi opazovane nelinearnosti bioloških sistemov težavo predstavlja tudi sklepanje iz posameznih meritev preprostih sistemov na odziv sestavljenih, kompleksnih sistemov.

V pričujočem delu bomo ubrali hibridni pristop. Z uporabo literature poiščemo karseda natančne referenčne vrednosti parametrov (vektor $\vec{c}$) za opazovane tipe kemijskih procesov, ki jih opazujemo, pri čemer je ključna natančna določitev razmerij med redi velikosti posameznih tipov kemijskih procesov. Model nato s preiskovanjem omejenega prostora parametrov poskušamo čim bolje prilagoditi eksperimentalnim meritvam, pri čemer ohranimo medsebojna razmerja v redih velikosti. Z dobljenimi vrednostmi parametrov nato pojasnimo razloge za odziv stikala ter smernice za izboljšave v delovanju.

Evolucijski algoritmi so popularna metoda za reševanje optimizacijskih problemov z več parametri, za katere ne poznamo analitične rešitve. Osrednja ideja sloni na simulaciji naravne selekcije, ki iterativno ocenjuje, spreminja in kombinira množico kandidatov za rešitev. Glavne lastnosti metod lahko strnemo v [7]:

- Evolucijski algoritmi uporabljajo kolektivno učno sposobnost populacije kandidatov. Vsak *kandidat* predstavlja točko (vektor) v prostoru iskanja in s tem oceno končne rešitve.
- Populacija se spreminja v skladu z naključnima procesoma *mutacije*, ki z majhno verjetnostjo spremeni komponento vektorja kandidata, ter *rekombinacije*, ki predstavlja križanje dveh vektorjev (staršev) v novi vektor (potomec).
- Za ocenjevanje posameznih kandidatov za rešitev je potrebna definicija enotne *cenovne funkcije* (angl. *fitness function*). Skladno z naravno selekcijo imajo na podlagi vrednosti cenovne funkcije kandidati z nižjo vrednostjo večjo verjetnost za preživetje in generiranje novih kandidatov za rešitev.

Naj bo $\mathcal{I}$ prostor kandidatov $\vec{a} \in \mathcal{I}$, ter $F : \mathcal{I} \to \mathbb{R}$ cenovna funkcija, ki vsakemu kandidatu določi vrednost glede na kvaliteto približka iskani rešitvi optimizacije. Naj bo

```
Vhod:
  mi, lambda, theta[];
Izhod:
  P*; Optimalna populacija;

Algoritem:

  P(t) = inicializacija populacije;
  F(t) = oceni(P(t));

  while (t < T)
    Pr(t) := rekombinacija (P(t));
    Pm(t) := mutacija (Pr(t));
    F(t) := oceni(Pr(t));
    P(t + 1) := selekcija(Pr(t),F(t),mi,theta[s]);
    t = t + 1;
```

Koda 2.2: Psevdokoda poteka osnovnega evolucijskega algoritma.

μ velikost množice staršev in λ velikost množice potomcev. $\vec{\boldsymbol{P}}(t) = (\vec{\boldsymbol{a}}_1(t), ..., \vec{\boldsymbol{a}}_\mu) \; \epsilon \; \mathcal{I}$ tako predstavlja populacijo v času t. Definirane so naslednje operacije:

- Selekcija $s : I^\lambda \rightarrow I^\mu$ predstavlja prehod v novo generacijo,
- Mutacija $m : I \overset{k}{\rightarrow} I$, kjer k predstavlja celo število dogodkov mutacij na posamezno generacijo,
- Rekombinacija $r : I^\mu \rightarrow I^\lambda$, ki iz populacije staršev določi populacijo potomcev,

pri čemer ima vsak operator množico lastnih nastavitev Θ_r, Θ_m, Θ_s, ki okarakterizirajo lastnosti algoritma. Osnovni evolucijski algoritem je opisan v kodi 2.2.

Z iterativnim izvajanjem tako algoritem preiskuje prostor parametrov do izpolnitve kriterija za končanje, ki je lahko dovolj majhna vrednost cenovne funkcije (odstopanja), ali iztek maksimalnega števila iteracij. V nadaljevanju bodo predstavljene konkretne odločitve pri implementaciji evolucijskega algoritma, ki bodo ocenjene v razdelku 3.4.

2.7.1 Predstavitev kandidatov za rešitev ($\vec{a}$)

Kandidata za rešitev predstavlja matrika faktorjev neznanih parametrov reakcij modela

$$A = [a_{ij}], \vec{\boldsymbol{a}}_i = [\vec{\boldsymbol{a}}_{i1}, ..., \vec{\boldsymbol{a}}_{iL}], \tag{2.30}$$

kjer vrstica $\vec{\boldsymbol{a}}_i$ predstavlja vektorje kandidatov za rešitev, stolpec $j \in 1, ..., L$ pa ustrezen faktor, s katerim pomnožimo referenčno vrednost parametra. Povedano drugače, elementi vektorja kandidata za rešitev predstavljajo faktorje, ki zmanjšajo ali povečajo referenčno vrednost. Velikost populacije je končna in se s časom ne spreminja.

2.7.2 Cenovna funkcija (F)

Cenovna funkcija za ocenjevanje kandidata za rešitev je sestavljena iz korakov:

1. Množenje soležnih elementov vektorja z referenčnimi vrednostmi $\vec{\boldsymbol{c}}$ (tabela 2.2):

$$\vec{\boldsymbol{a}}' = \vec{\boldsymbol{a}} \circ \vec{\boldsymbol{c}}. \tag{2.31}$$

2. Zagon n_{SSA} stohastičnih simulacij s parametri $\vec{\boldsymbol{a}}'$, kjer vsaka vrne odstopanje od eksperimentalno izmerjenih vrednosti v petih časovnih točkah za oba poročevalska proteina. Posamezna instanca simulacije izračuna srednjo kvadratno napako simulacijskih vrednosti r_{SSA} obeh poročevalskih proteinov v petih eksperimentalno izmerjenih točkah vzorčenja t_s:

$$f_{\vec{\boldsymbol{a}}} = \sum_{t_s} \sqrt{(r_s - r_{\text{SSA}})^2}, \quad t_s \in \{t_1, t_2, t_3, t_4, t_5\}, \tag{2.32}$$

3. Srednjo kvadratno napako povprečimo za oba poročevalca po zagonu n_{SSA} instanc simulacij:

$$\epsilon_b = 1/n_{\text{SSA}} \cdot \sum_{i=1}^{n_{\text{SSA}}} f_{\vec{\boldsymbol{a}}}(\text{BFP}), \tag{2.33}$$

$$.\epsilon_m = 1/n_{\text{SSA}} \cdot \sum_{i=1}^{n_{\text{SSA}}} f_{\vec{\boldsymbol{a}}}(\text{mCT}). \tag{2.34}$$

4. Končno vrednost cenovne funkcije predstavlja večja izmed kvadratnih napak vrednosti obeh poročevalcev:

$$F = \max.\{\epsilon_b, \epsilon_m\} \tag{2.35}$$

2.7.3 Definicija procesa mutacije (Θ_m)

Pri postopku mutacije gradimo na ideji opisani v [30], ki temelji na prilagajanju števila dogodkov mutacij pri posameznemu kandidatu glede na vrednost cenovne funkcije kandidata in standardni odklon oz. informativno vrednost posamezne komponente vektorja. Boljše ocenjeni kandidati so tako podvrženi manj mutacijam, pogosteje pa so mutirani parametri, ki imajo manjšo informativno vrednost. Kumulativna funkcija verjetnosti za opazovanega kandidata i je definirana kot

$$C(F(\vec{\boldsymbol{a}}_i)) = 1/N \sum_{f \geq F} N(f) \tag{2.36}$$

in predstavlja delež kandidatov z večjo vrednostjo cenovne funkcije. Verjetnost, da pri kandidatu $\vec{\boldsymbol{a}}_i$ pride do mutacije, je tako enaka $\alpha(\vec{\boldsymbol{a}}_i) = 1–C(F(\vec{\boldsymbol{a}}_i))$. Število dogodkov mutacij pri opazovanem kandidatu je tako enako

$$N_m = \alpha(\vec{\boldsymbol{a}}_i) \cdot L. \tag{2.37}$$

Za vsako posamezno komponento a_{ij} vektorja $\vec{\boldsymbol{a}}_i$ kandidata rešitve izračunamo standardni odklon σ_j preko celotne populacije. Komponente z manjšim standardnim odklonom intuitivno upoštevamo kot bolj informativne, zato je verjetnost, da bodo del končne rešitve, večja. Pri posameznem kandidatu $\vec{\boldsymbol{a}}_i$ *začasno* razporedimo komponente vektorja po pravilu

$$\sigma(\vec{\boldsymbol{a}}'_{i,j=1}) \geq \sigma(\vec{\boldsymbol{a}}'_{i,j=2}) \geq ... \geq \sigma(\vec{\boldsymbol{a}}'_{i,j=L}), i = 1, ..., N, \tag{2.38}$$

ter izvedemo mutacije na prvih N_m komponentah začasnega vektorja. V našem primeru novo vrednost komponente vzorčimo s pomočjo naključne spremenljivke porazdeljene po normalni porazdelitvi s parametroma $\mu = 0$ in $\sigma = 1.0$. Začasni vektor s spremenjenimi komponentami priredimo v originalni vektor tako, da komponente razporedimo v originalni vrstni red.

2.7.4 Definicija procesa rekombinacije (Θ_r)

Parametri reakcij modela za stohastično simulacijo so med seboj neodvisni, relativna pogostost reakcij pa je odvisna od relativnih razmerij med parametri. Ideja križanja je ohranitev dobrih podzaporedij v kandidatu vektorja. Odločili smo se za neodvisno križanje vsake komponente vektorja (angl. *N-point mutation*). Potomec dveh kandidatov $\vec{\boldsymbol{a}}_i$ in $\vec{\boldsymbol{a}}'_i$, je tako definiran z

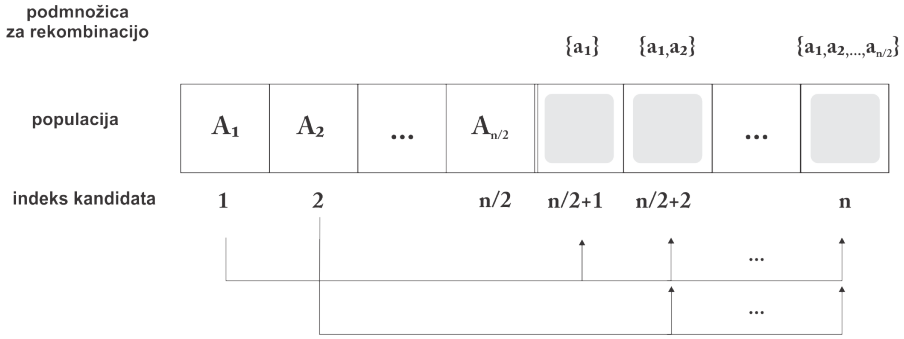
$$\vec{\boldsymbol{a}_k} = [r(\vec{\boldsymbol{a}}_{i1}, \vec{\boldsymbol{a}}_{i'1}), r(\vec{\boldsymbol{a}}_{i2}, \vec{\boldsymbol{a}}_{i'2}), ...r(\vec{\boldsymbol{a}}_{i'L}, \vec{\boldsymbol{a}}_{jL})], \tag{2.39}$$

kjer funkcija r izbere eno izmed komponent a_{il} ali $a_{i'l}$ z enako verjetnostjo.

2.7.5 Definicija procesa selekcije (Θ_s)

Po izračunu cenovne funkcije populacije kandidatov slednjo vstavimo v podatkovno strukturo kopice [10], ki za kriterij uporablja vrednost cenovne funkcije v naraščajočem redu. Implementirana je s pomočjo seznama.

V procesu selekcije (slika 2.8) spodnjo polovico razvrščene populacije zavržemo, nastala prosta mesta pa bodo zapolnjena s kandidati, ki bodo rezultat rekombinacije. Vsako prosto mesto obravnavamo posebej, nove kandidate pa ustvarimo z rekombinacijo iz naključnega para iz podmnožice za rekombinacijo. V slednjo z vsakim novim prostim mestom dodamo enega izmed kandidatov iz zgornje polovice populacije po vrstnem redu naraščajoče cenovne funkcije. Boljši kandidati imajo tako večjo verjetnost reprodukcije, naključna izbira parov pa zagotavlja možnost preživetja tudi neoptimalnim rešitvam.



Slika 2.8: Shematski prikaz procesa selekcije.

2.7.6 Kriterij za zaključitev

Iteracije evolucijskega algoritma se zaključijo po izteku vnaprej določenega maksimalnega števila iteracij.

2.8 CUDA Arhitektura

Tako algoritem SSA kot evolucijski algoritem zahtevata mnogo izračunov, ki so med seboj neodvisni. To lastnost lahko s pridom izkoristimo tako, da omenjene dele programa izračunamo paralelno. V zadnjih letih so paralelni računalniki vedno dostopnejši. Naravna potreba po paralelnem računanju se med drugim pojavi pri procesiranju grafike,

```
__global__ void add(int *a, int *b, inc*){
  int tid = threadIdx.x;
  c[tid] = a[tid] + b[tid];
```

Koda 2.3: Paralelno seštevanje dveh vektorjev. Funkcija kot parametre prejme naslova izvornih in ponornih vektorjev. Glede na svojo zaporedno številko na ustrezen naslov vrne rezultat izračuna.

kjer je pogost problem računanje neodvisnih operacij nad slikovnimi pikami, zato so domala vsi grafični procesorji danes zasnovani kot paralelni procesorji. Koncept se je razširil tudi na splošno namensko procesiranje (angl. *general purpose computing*), kjer je eden najbolj uporabljanih produktov arhitektura CUDA podjetja NVIDIA, ki je sestavljena iz posebne arhitekture vzporednega SIMD[3] računalnika ter pripadajočega programskega vmesnika.

Da bi razumeli implementacijo prilagojenega SSA algoritma, moramo poznati osnove CUDA arhitekture ter nekatere njene posebnosti, saj te prinašajo dodatne omejitve pri programiranju.

2.8.1 Programski model

Programiranje za paralelne naprave, ki podpirajo arhitekturo CUDA se od programiranja za običajne procesorje ne razlikuje veliko. Program, ki se izvaja na centralni procesni enoti (angl. *Central processing unit*, CPU), ter program, ki se izvaja na grafičnem procesorju (angl. *Graphics processing unit*, GPU) si delita skupni navidezni pomnilniški prostor. Vanj preko kazalcev prenašamo podatkovne strukture.

Programska koda, ki se izvaja na GPU, je pisana v posebnih metodah, ki jih imenujemo *jedra* (angl. *kernel*). Te se v pomnilnik GPU prenesejo skupaj s podatki v skupinah, imenovanih *bloki* (angl. *block*). Vsak blok je nadalje razdeljen na *niti* (angl. *thread*). Vsaka izmed slednjih je naslovljena z enotno kombinacijo zaporedne številke bloka ter zaporedne številke niti, kar omogoča dostop do različnih naslovov. Primer uporabe za seštevanje dveh vektorjev v jeziku C/C++ je prikazan v kodi 2.3.

[3] *Single instruction, multiple data.* Arhitektura računalnika, ki lahko isti ukaz izvede hkrati nad več neodvisnimi podatki.

2.8.2 Opis arhitekture

Fizična realizacija in logični programski model se v realizaciji CUDA arhitekture zelo razlikujeta, zato je potrebno poznati oba. Grafične kartice NVIDIA imajo število *multiprocesorjev*, kjer vsak od njih paralelno izvaja program. Multiprocesor vsebuje skupino *jeder* (angl. *core*), ki izvajajo niti programa. Le-te se izvajajo v načinu SIMD; vsaka skupina niti (angl. *warp*) izvaja isti ukaz (v splošnem nad različnimi podatki). Shema CUDA naprave je prikazana na sliki 2.9.

V našem primeru[4] grafični procesor vsebuje 7 multiprocesorjev, kjer vsak vsebuje 48 jeder, skupno torej 336 jeder. Posamezni multiprocesor lahko hkrati izvaja dve skupini po 32 niti, kjer se v enem ciklu izvedeta dva ukaza. V teoriji je tako mogoče doseči maksimalno 4 ukaze na urin cikel. Da se torej lahko hkrati izvaja čim večje število niti je nujno, da je tok ukazov pri vseh nitih enak. Primer intuitivne omejitve tovrstne arhitekture je prikazana v kodah 2.4 in 2.5. V opisanem primeru se odseka kode v primeru različnih tokov ukazov ne izvedeta paralelno. Odsek 2 (v niti 2) se tako ne izvede, dokler se Odsek 1 (v niti 1) ne zaključi[5]. Dobra praksa tako narekuje izogibanje vejitvam, kjer je to možno.

```
...
if (pogoj)  Odsek 1;
else ...
...
```

Koda 2.4: Program niti 1.

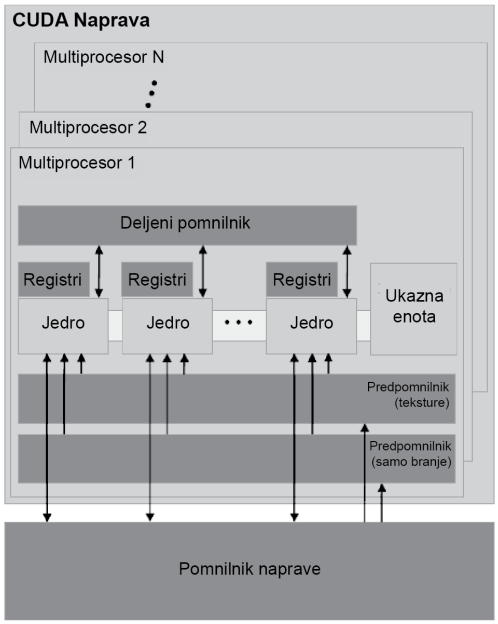
```
...
if (pogoj)   ...
else  Odsek 2;
...
```

Koda 2.5: Program niti 2

Vsako jedro vsebuje svoji enoti za celoštevilsko računanje ter računanje v plavajoči vejici z enkratno natančnostjo. Posebne, transcendentne matematične funkcije (logaritem, kotne funkcije, itd.) so izračunane v osmih posebnih, deljenih funkcijskih enotah. Izvajanje takih ukazov se na posameznem multiprocesorju tako upočasni za 6-krat, saj morajo niti čakati na prosto enoto.

[4] V študiji uporabljamo Grafično kartico NVIDIA GeForce 460 GTX. Specifikacije so opisane v prilogi A.3.

[5] Vrstni red izvajanja niti z različnim tokom ukazov je v splošnem odvisen od algoritma razvrščanja, za potrebe ilustracije smo privzeli, da se nit 1 izvede pred nitjo 2.



Slika 2.9: Shema CUDA naprave.

2.8.3 Organizacija pomnilnika

Podobno kot posebne funkcijske enote, si jedra delijo tudi enote za operacije branja in pisanja v pomnilnik. Vsak multiprocesor ima 16 bralno/pisalnih enot, torej si v povprečju eno enoto delijo po tri niti, zaradi česar je potrebno minimizirati število bralno pisalnih operacij.

Dostop do globalnega pomnilnika GPU naprave je zaradi visoke latence (400-800 urinih ciklov) in v splošnem naključnih vzorcev dostopov počasen. V našem primeru je uporabljena Fermi arhitektura z dvema nivojema predpomnilnika in hitrejšim deljenim bralnim pomnilnikom z latenco nekaj ciklov. Med čakanjem na dostop do pomnilnika je v splošnem možno izvajanje ostalih niti, a pri nitih z enakim zaporedjem ukazov to dejstvo opazno ne izboljša hitrosti izvajanja.

Jedra v posameznem multiprocesorju si delijo tudi 32.768 32-bitnih registrov, do katerih je dostop najhitrejši. V registre je možno zapisovati posamezne spremenljivke in ne seznamov, kar je pomembno dejstvo, ki bo obravnavano v nadaljevanju [38].

3 Rezultati

V pričujočem poglavju bodo predstavljene ugotovitve in rezultati dela. V začetku bomo opisali posebnosti pri implementacije rešitve, primerjavo paralelne in serijske implementacije algoritma ter na koncu izpostavili pridobljeno znanje iz simulacij genskega regulatornega omrežja stikala s pozitivno povratno zanko.

3.1 Implementacija SSA algoritma na paralelni arhitekturi

Zaradi omejitev arhitekture opisanih v razdelku 2.8, bomo v nadaljevanju izpostavili nekatere ključne izboljšave pri implementaciji SSA algoritma, ki v praksi močno zmanjšajo čas potreben za izvajanje posamezne simulacije, pri čemer pa je pomembno, da ne spremenijo rezultata.

3.1.1 Odprava seznamov

Niti, ki se izvajajo na GPU, si delijo večstopenjski skupni pomnilnik. Ker je število enot za dostop do pomnilnika manjše od števila niti, ki se lahko hkrati izvajajo, je pomembno, da je karseda veliko število spremenljivk shranjenih v registrih, do katerih je dostop naj-

hitrejši. Seznami števil, pri katerih je pomemben vrstni red, se v najboljšem primeru shranjujejo v prvi nivo predpomnilnika, zaradi česar jih nadomestimo z navadnimi spremenljivkami (koda 3.1), saj se le te lahko shranjujejo v registre.

```
int mu; //Indeks izvedene reakcije
float species0, species1,...species28; //Koncentracije kemijskih zvrsti
float c0,c1,...c45; // Koeficienti pogostosti reakcij
float w0,w1,...w45; // Koeficienti nagnjenosti reakcij
```

Koda 3.1: Pomembnejše spremenljivke v algoritmu SSA.

3.1.2 Odprava vejitev

Ker vse niti hkrati izvajajo enak ukaz (v splošnem nad različnimi operandi) je pomembno, da se izognemo vejitvam (pogojnim stavkov), kadar je to mogoče, tudi za ceno večjega števila ukazov. Primer je posodabljanje koncentracij kemijskih zvrsti, kjer pri vsaki oblikujemo enačbo, v kateri je kot faktor upoštevana vsaka reakcija, ki spremeni njeno koncentracijo. Tako v vsakem koraku algoritma izračunamo spremembo vseh zvrsti, ne glede na trenutno reakcijo (koda 3.2).

```
...
  species0 = species0 + (mu == 24) * 1 + (mu == 25) * 1 + (mu == 40) * -1 ;
  species1 = species1 + (mu == 28) * 1 + (mu == 29) * 1 + (mu == 41) * -1 ;
...
```

Koda 3.2: Primer odprave pogojnih stavkov in seznamov pri posodabljanju koncentracij kemijskih zvrsti.

3.1.3 Število iteracij

Podobno kot v prejšnjem razdelku je pomembno, da vse niti izvedejo enako število iteracij. Originalni SSA algoritem se zaključi ob dosegu določene časovne točke. Ker se čas povečuje po pravilu $t = t + \tau$, pri čemer je τ naključno porazdeljena spremenljivka po enačbi 2.5, se v splošnem dejansko število časovnih korakov od simulacije do simulacije razlikuje. To omejitev odpravimo tako, da pri vsaki ponovitvi simulacije izvedemo enako število korakov (reakcij). Slednje je empirično določeno in enako $18 \cdot 10^4$.

3.1.4 Izračun naravnega logaritma

Ne glede na odpravljeno omejitev v prejšnjem razdelku je pomembno, da ohranimo izračun časa skozi simulacijo. Enačba 2.5 predvideva izračun naravnega logaritma, ki predstavlja transcendenčno funkcijo. Slednje se na CUDA napravi računajo v posebnih enotah, zato se jim izognemo, kadar je to mogoče. V našem primeru uporabimo numerični približek [44]

$$\log(x) = \frac{\pi}{2M(1, 2^{2-m}/x)} - m\log(2), \tag{3.1}$$

pri čemer $M(x, y)$ predstavlja aritmetično-geometrično sredino števil x in y, število m pa izpolnjuje pogoj

$$x \cdot 2^m > 2^{p/2}, \tag{3.2}$$

kjer je p željena natančnost, merjena v bitih.

3.2 Analiza časovne zahtevnosti

Zastavljen problem je računsko zelo zahteven, saj evolucijski algoritem pri vsaki od iteracij predpostavlja izračun cenovne funkcije (odstopanje od izmerjenih vrednosti), ki jo v našem primeru predstavlja rezultat posamezne instance simulacije genskega regulatornega omrežja. Zaradi stohastične narave algoritma je potrebno posamezno instanco simulacije ponoviti večkrat, da dobimo boljšo oceno cenovne funkcije. Na koncu vsake iteracije evolucijskega algoritma je populacijo potrebno še razporediti glede na naraščajočo vrednost cenovne fukcije. Predvidena časovna zahtevnost serijske implementacije algoritma je tako

$$T(n) = O(I_{MAX} \cdot N \cdot (n_{SSA} + logN)), \tag{3.3}$$

kjer I_{MAX} predstavlja največje dovoljeno število iteracij evolucijskega algoritma, N velikost populacije, n_{SSA} pa število ponovitev SSA algoritma na posameznem kandidatu za rešitev. Uporabljeno sortiranje s kopico (angl. *heapsort*) ima časovno zahtevnost $O(N) = N \log N$ [10].

Izračuni cenovnih funkcij so pri posameznih kandidatih neodvisni, zato lahko arhitekturo izkoristimo za paralelni izračun le-teh. Tako v enem koraku izračunamo n_{SSA} ponovitev izračuna cenovne funkcije za N kandidatov. Pri tem posmeznega kandidata predstavlja *blok*, kjer je vsakemu bloku dodeljenih n_{SSA} *niti* za ponovitev izračuna cenovne

funkcije. Zaradi relativno majhnega števila populacije (glej razdelek 3.5), se razvrščanje vrši na CPU. Predvidena izboljšana časovna zahtevnost je tako

$$T(n) = O(I_{MAX} \cdot N \cdot (\epsilon + logN)), \tag{3.4}$$

kjer $\epsilon << n_{SSA}$, kar v praksi pomeni

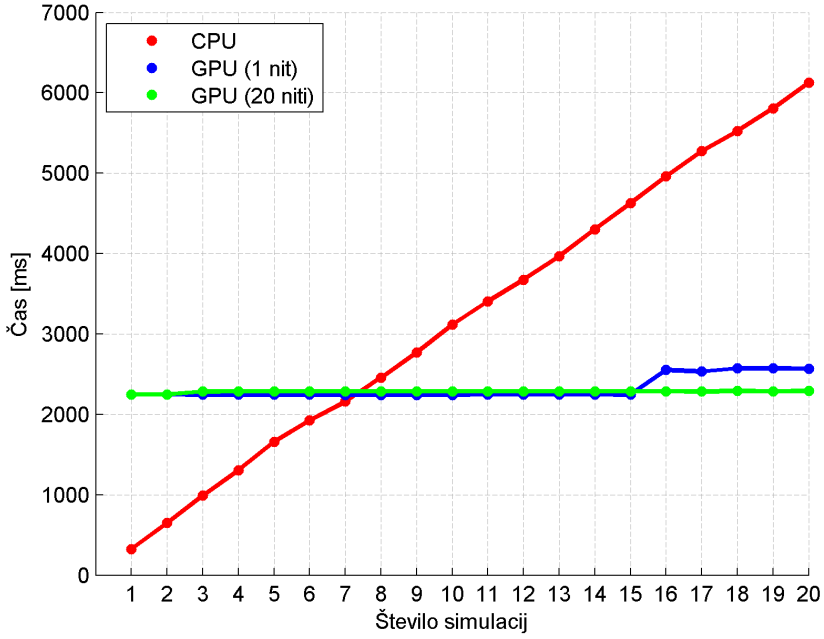
$$T(n) = O(I_{MAX} \cdot N \log N). \tag{3.5}$$

Ker smo razvrščanje v obeh primerih izvajali na CPU, bomo v nadaljevanju primerjali le čas izvajanja v odvisnosti od števila simulacij. Po pričakovanjih časovna zahtevnost izvajanja na CPU raste premo sorazmerno s številom simulacij (slika 3.1a). Presenetljivo se točka, kjer je čas izvajanja približno enak tako na CPU kot GPU, pojavi že pri relativno nizki vrednosti 7 simulacij, kar nakazuje na relativno majhen prispevek k času s strani prenašanja podatkov med GPU in CPU. Čas izvajanja na GPU ostaja približno konstanten. Opazna je strojna omejitev, saj se čas pri enakem številu simulacij pri večjem številu blokov od niti nekoliko poveča. Vzporedno se tako lahko izvaja večje število niti kot blokov.

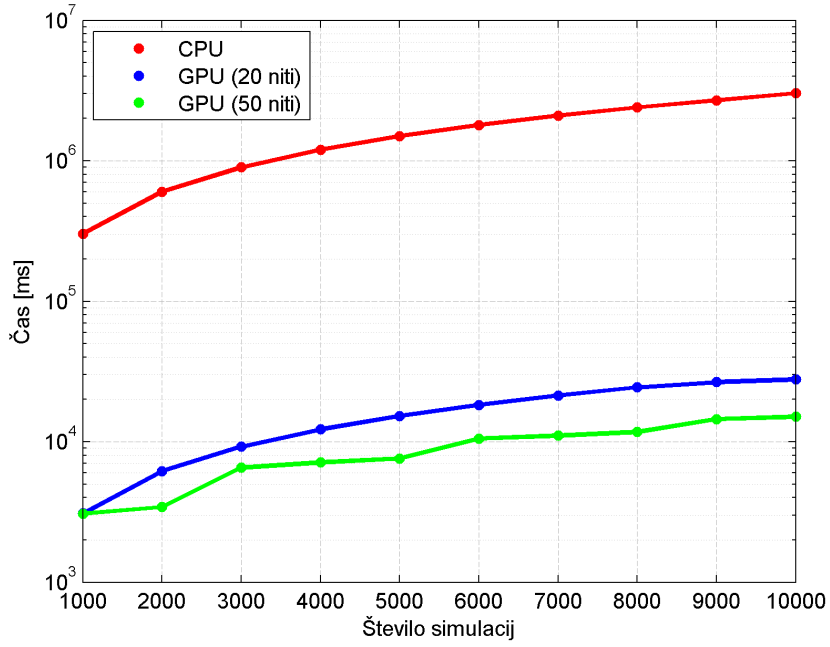
Pri večjih vrednostih števila simulacij, ki jih uporabljamo v praksi (nekaj 1000), so opazne tudi strojne omejitve GPU, a vendar s številom simulacij čas narašča počasneje in je v praksi za dva razreda velikosti manjši od izvajanja na CPU (slika 3.1b). Z našo implementacijo tako dosežemo največjo pohitritev za faktor 204.2, kar je 18% več v primerjavi z rezultati v [31] ter 625% v primerjavi z modelom primerljive velikosti v [22].

3.3 Minimizacija in predpriprava modela

Po uporabi optimizacij na nivoju programske kode smo se poslužili minimizacije modela. Ker bistvo delovanja stikala podajata koncentraciji dveh poročevalskih proteinov, lahko nekatere reakcije odstranimo [20]. Ker je časovna zahtevnost izvajanja SSA algoritma odvisna predvsem od števila kemijskih zvrsti in reakcij [6, 18, 49], lahko na ta način bistveno zmanjšamo čas izvajanja. Pri opisu izhajamo iz *osnovnega modela* stikala, ki ga prilagodimo v več korakih in na koncu dobimo *minimalni model*. Pri tem je pomembno, da primerjana modela ohranita enak odziv do izbrane vrednosti napake. Reakcije in kemijske zvrsti obeh modelov so podane v prilogah A.1 in A.2.



(a)



(b)

Slika 3.1: (a) Primerjava časov izvajanja števila simulacij na CPU (rdeča) ter GPU. Tako v primeru uporabe konstantnega števila blokov (zelena) ali niti (modra) čas izvajanja ostane približno konstanten. (b) Primerjava časov izvajanja števila simulacij na CPU (rdeča) ter GPU (zelena, modra).

3.3.1 Inducibilni sistem

Za delovanje bistabilnega stikala sta bistveni prisotnost dveh recipročnih represorjev in pozitivna povratna zanka [28, 36]. Naloga inducibilnega sistema, ki ga sestavljajo konstrukti v drugem in tretjem nivoju (razdelek 2.1.5), je zunanji nadzor izražanja enega ali drugega stanja. To dosežemo s povečano produkcijo ustreznih aktivatorjev in represorjev, opis česar v osnovnem modelu zahteva 22 reakcij ter 16 kemijskih zvrsti (glej prilogo A.1).

Naša teza je, da lahko inducibilni sistem opišemo z manj potrebnimi elementi tako, da *ob prisotnosti antibiotika pristinamicina (PI) ali eritromicina (ER) sprožimo produkcijo ustreznih represorjev in aktivatorjev v drugem nivoju.* To lahko storimo *brez upoštevanja konstruktov tretjega nivoja in inducibilnih proteinov.* Tako lahko indukcijo stanja A opišemo s povečano transkripcijo konstruktov v drugem nivoju ob prisotnosti pristinamicina:

$$[P_{\mathrm{CMV}\#4}] + [PI] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#4}] + [T_{BK}], \tag{3.6}$$

$$[P_{\mathrm{CMV}\#5}] + [PI] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#5}] + [T_{AV}]. \tag{3.7}$$

Analogno opišemo indukcijo stanja B:

$$[P_{\mathrm{CMV}\#6}] + [ER] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#6}] + [T_{AK}], \tag{3.8}$$

$$[P_{\mathrm{CMV}\#7}] + [ER] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#7}] + [T_{BV}]. \tag{3.9}$$

Omenjena primera torej ne povzameta vseh reakcij, ki se zgodijo pri indukciji, vendar zadoščata za opis želenega kvalitativnega obnašanja inducibilnega sistema.

Puščanje inducibilnega sistema

Pri eksperimentalnem delu smo veliko pozornosti namenili testiranju inducibilnih sistemov, ki je pokazalo asimetrični odziv indukcije z enim ali drugim antibiotikom. Razlog je *puščanje*, t.j. transkripcija proteinov tudi pri predvidenem represiranem promotorju. V praksi to pomeni manjši odstotek transkripcije, ki je v idealnih pogojih ne bi bilo. Da bi v nadaljevanju z optimizacijo odziv modela bolje prilagodili eksperimentalno dobljenim reazultatom (glej razdelek 2.6), upoštevamo še reakcije za puščanje inducibilnega sistema, ki s parametroma l_{ER} in l_{PI} povzameta asimetričnost delovanja:

$$[P_{\mathrm{CMV}\#4}] \xrightarrow{l_{\mathrm{PI}} \cdot v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#4}] + [T_{BK}], \tag{3.10}$$

$$[P_{\mathrm{CMV}\#5}] \xrightarrow{l_{\mathrm{PI}} \cdot v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#5}] + [T_{AV}], \tag{3.11}$$

$$[P_{\mathrm{CMV}\#6}] \xrightarrow{l_{\mathrm{ER}} \cdot v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#6}] + [T_{AK}], \tag{3.12}$$

$$[P_{\mathrm{CMV}\#7}] \xrightarrow{l_{\mathrm{ER}} \cdot v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#7}] + [T_{BV}]. \tag{3.13}$$

Ker je transkripcija v primeru puščanja manjša od transkripcije neinduciranih promotorjev, upoštevamo $l_{\mathrm{PI}},\ l_{\mathrm{ER}} < 1$. Na ta način smo opis inducibilnega sistema zmanjšali na 8 reakcij in 6 kemijskih zvrsti.

3.3.2 Zmanjšanje števila opazovanih stanj promotorja

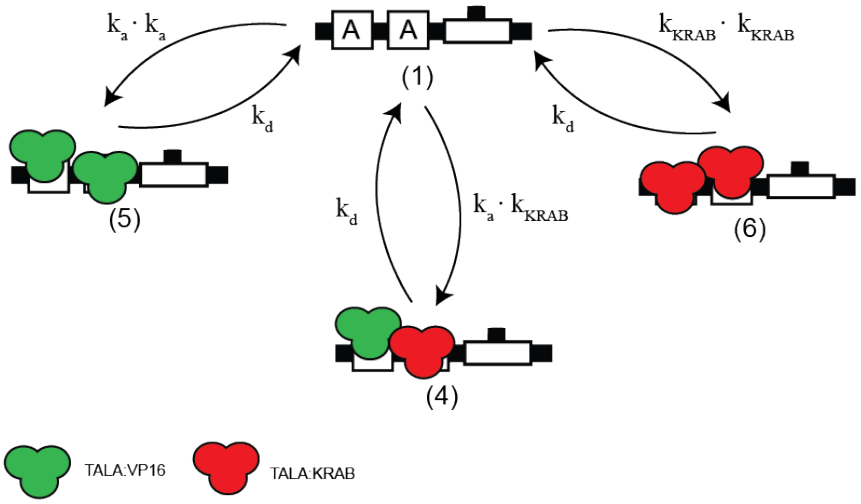
Opis vseh možnih stanj za 4 minimalne promotorje zahteva opazovanje 28 kemijskih zvrsti ter 52 reakcij. S simulacijo osnovnega modela lahko ugotovimo, katera stanja bistveno vplivajo na kvalitativni odziv sistema.

Predvidene referenčne vrednosti parametrov iz literature (tabela 2.2), predvidevajo večjo verjetnost *vezave* TAL proteina na DNA v primerjavi z *disociacijo* z DNA

$$k_{\mathrm{a}} << k_{\mathrm{da}},\ k_{\mathrm{b}} << k_{\mathrm{db}}. \tag{3.14}$$

V tem primeru pričakujemo relativno majhno frekvenco stanj s prostimi vezavnimi mesti (slika 2.7), kar smo preverili s pomočjo simulacije osnovnega modela. Izkaže se, da je *povprečna relativna frekvenca stanj promotorja s prostimi vezavnimi mesti bistveno nižja od frekvence stanj, kjer so vsa vezavna mesta zasedena.* Slednje dejstvo je mogoče razbrati iz tabele 3.1, kjer je relativna frekvenca stanj z vsaj enim prostim vezavnim mestom vsaj dva velikostna razreda nižja od polno zasedenih stanj.

Skupna vsota relativnih frekvenc pojavljanja stanj promotorja s prostimi vezavnimi mesti (stanja $\emptyset$, V, K) je enaka 0.4%. Delež predpostavimo kot zanemarljiv, zato sistem poenostavimo tako, da opazujemo le tri možna stanja, v katerih je promotor v celoti zaseden (KK, VK in VV), kar je prikazano na Sliki 3.2. Transkripcija je torej možna v dveh stanjih. Z uporabo opisane poenostavitve prehajanja stanje novega avtomata opišemo z 20 kemijskimi zvrstmi ter 32 reakcijami.



Slika 3.2: Prikaz stanj minimalnega promotorja v prvem nivoju v obliki končnega avtomata minimalnega modela. Opis je analogen za obe strani stikala, tako za konstrukte z vezavnimi mesti tipa A, kot konstrukti z vezavnimi mesti tipa B.

Zasedenost vezavnih mest	$P_{min\#0}$	$P_{min\#1}$	$P_{min\#2}$	$P_{min\#3}$
$\emptyset$	0.00050	0.00011	0.00049	0.00009
V	0.00090	0.00019	0.00080	0.00015
K	0.00090	0.00016	0.00026	0.00005
KK	0.19473	0.03953	0.27761	0.05464
VK	0.01621	0.00242	0.02707	0.00617
VV	0.20334	0.04089	0.11042	0.02222

Tabela 3.1: Relativne frekvence stanj zasedenosti promotorja v simulaciji stikala, kjer simuliramo preklop iz stanja A v stanje B in obratno. Zasedenost vezavnih mest (z represorji ali aktivatorji) je zakodirana kot množica kemijskih zvrsti promotorja na naslednji način: $\emptyset = \{P_{min\#0}, P_{min\#1}, P_{min\#2}, P_{min\#0}\}$, K $= \{T_{AK}\text{-}P_{min\#i}, T_{BK}\text{-}P_{min\#j}\}$, V $= \{T_{AV}\text{-}P_{min\#i}, T_{BV}\text{-}P_{min\#j}\}$, KK $= \{T_{AK}\text{-}T_{AK}\text{-}P_{min\#i}, T_{BK}\text{-}T_{BK}\text{-}P_{min\#j}\}$, VK $= \{T_{AV}\text{-}T_{AK}\text{-}P_{min\#i}, T_{BV}\text{-}T_{BK}\text{-}P_{min\#j}\}$, VV $= \{T_{AV}\text{-}T_{AV}\text{-}P_{min\#i}, T_{BV}\text{-}T_{BV}\text{-}P_{min\#j}\}$, pri čemer velja $i \in \{0,1\}, j \in \{2,3\}$.

3.3.3 Primerjava časov izvajanja in napake odziva osnovnega in minimalnega modela

Pravilnost rezultatov minimalnega modela smo preverili z uporabo treh Wassersteinovih psevdometrik, ki so opisane v razdelku 2.3. Odziv sistema smo preverili z uporabo štirih vrst simulacij:

- **Stabilno stanje**; V začetku sprožimo indukcijo stanja A ali stanja B ter v času t_1 odstranimo antibiotik. Sistem tako ostane v stanju, določenem v času t_0.
- **Preklop**; V začetku sprožimo indukcijo stanja A ali stanja B ter v času t_1 *zamenjamo* antibiotik. Sistem najprej privzame prvo stanje, ob menjavi induktorja pa ga zamenja.

Rezultati so prikazani v tabeli 3.2. Opazimo majhno odstopanje osnovnega in minimalnega modela pri uporabi dveh različnih metrik in štirih vrst simulacij. Relativne napake (ε_d) minimalnega modela je reda velikosti 1.0%. Zaključimo, da lahko minimalni model opiše (za praktično uporabo) ekvivalenten odziv kot osnovni model.

S postopki minimazacije smo iz začetnih 46 kemijskih zvrsti in 90 reakcij v osnovnem modelu opis modela zmanjšali na 28 kemijskih zvrsti in 46 reakcij. Prednost uporabe minimalnega modela smo dodatno preverili s primerjavo časa izvajanja (Tabela 3.3). V vseh primerih se simulacija minimalnega modela izvede v približno 25% krajšem času.

3.3.4 Asimetričnost sistema

Poleg asimetričnega puščanja inducibilnega sistema v Razdelku 3.3.1 smo po končani minimizaciji sprejeli dodatne predpostavke za boljše prileganje asimetričnim eksperimentalnim rezultatom:

- razliko v verjetnosti vezave med proteini z vezavnimi domenami TALA ali TALB določata ločena parametra k_{a} in k_{b},
- razliko v verjetnosti odvezave med proteini z vezavnimi domenami TALA ali TALB določata ločena parametra k_{da} in k_{db},
- razliko v transkripciji pri aktivaciji s TALA:VP16 ali TALB:VP16 določa parameter a_{T}.

Vrsta simulacije	$Z(\omega)$, ω = [BFP]		$Z(\omega)$, ω = [mCT]	
	$W_d^1(\mathcal{P}_1, \mathcal{P}_2)$	$\varepsilon_d(\mathcal{P}_1 \| \mathcal{P}_2)$	$W_d^1(\mathcal{P}_1, \mathcal{P}_2)$	$\varepsilon_d(\mathcal{P}_1 \| \mathcal{P}_2)$
Stabilno stanje (A)	3.61	0.020	0.00	0.00
Stabilno stanje (B)	0.00	0.00	2.540	0.013
Preklop (A → B)	0.45	0.0027	2.84	0.015
Preklop (B → A)	3.52	0.019	0.27	0.0017

(a)

Vrsta simulacije	$Z(\omega)$, ω = [BFP]		$Z(\omega)$, ω = [mCT]	
	$W_d^1(\mathcal{P}_1, \mathcal{P}_2)$	$\varepsilon_d(\mathcal{P}_1 \| \mathcal{P}_2)$	$W_d^1(\mathcal{P}_1, \mathcal{P}_2)$	$\varepsilon_d(\mathcal{P}_1 \| \mathcal{P}_2)$
Stabilno stanje (A)	1.37	0.0075	0.00	0.00
Stabilno stanje (B)	0.00	0.00	2.60	0.014
Preklop (A → B)	1.18	0.0064	0.00	0.00
Preklop (B → A)	0.0004	0.0002	2.66	0.014

(b)

Tabela 3.2: Izračunane vrednosti treh različnih Wassersteinovih psevdometrik za dobljeni porazdelitvi poročevalskih spremenljivk osnovnega ($\mathcal{P}_1$) in minimalnega modela ($\mathcal{P}_2$) in pripadajoče napake $\varepsilon_d(\mathcal{P}_1||\mathcal{P}_2) = W_d^1(\mathcal{P}_1, \mathcal{P}_2)/\max(\mathcal{P}_1)$. (a) Metrika predstavlja količino proteina ob zaključku simulacije (zadnja točka vzorčenja), $Z(\omega) = \omega(t_5)$. (b) Metrika predstavlja količino proteina v času menjave ali odstranitve antibiotika v drugem dnevu (prva točka vzorčenja), $Z(\omega) = \omega(t_1)$.

Vrsta simulacije	Osnovni model [ms]	Minimalni model [ms]
Stabilno stanje (A)	0.44045	0.33064
Stabilno stanje (B)	0.43873	0.33482
Preklop (A → B)	0.44055	0.33418
Preklop (B → A)	0.44064	0.33382

Tabela 3.3: Primerjava časov izvajanja osnovnega in minimalnega modela za posamezno simulacijo pri različnih vrstah indukcije in enakem številu simulacijskih korakov. Zaradi primerjave pohitritve kot posledice minimizacije modela sta bili obe različici testirani na CPU.

3.4 Izbira parametrov evolucijskega algoritma

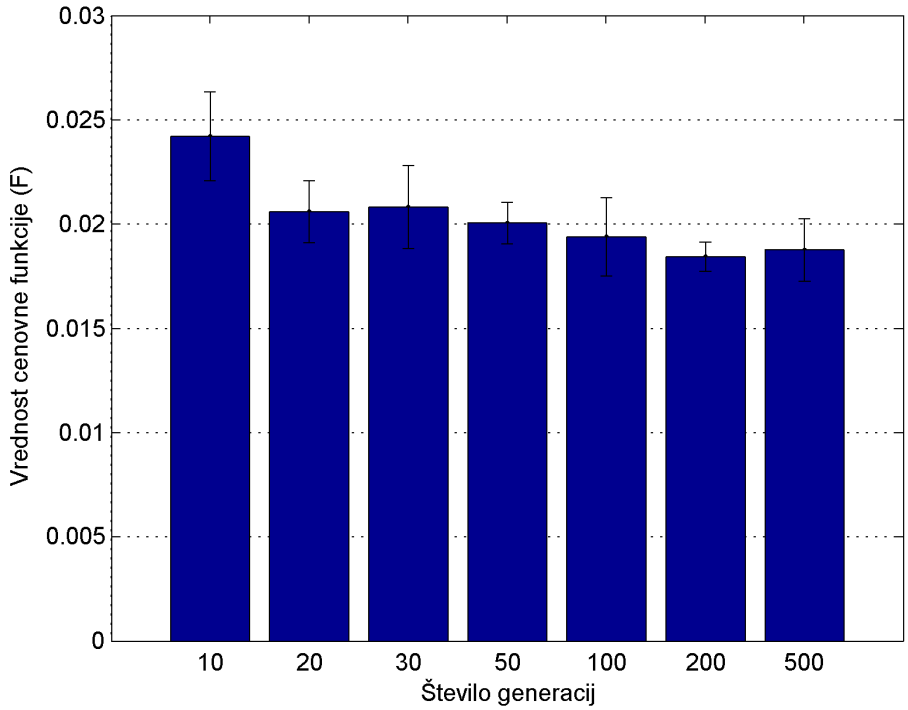
Optimalno delovanje evolucijskega algoritma je odvisno od nastavitev delovanja. V nadaljevanju se bomo osredotočili na nekatere ključne parametre, ter vsakega izmed njih preizkusili ločeno. Vsak preizkus algoritma je bil ponovljen štirikrat.

Algoritem zagotavlja rešitev, ki predstavlja lokalni minimum, k kateremu konvergira po določenem številu generacij (iteracij) [7]. Na sliki 3.3 smo zagon algoritma prezkusili pri različnih številih generacij. Izkaže se, da cenovna funkcija v povprečju doseže najnižjo vrednost pri zagonu med 100 in 200 generacij. Večje število generacij ne ponudi bistveno boljše rešitve.

Hitrost konvergence evolucijskega algoritma je tudi nelinearno odvisna od velikosti populacij (števila kandidatov za rešitev) [43]. Na sliki 3.4a je prikazana konvergenca najboljšega kandidata za rešitev skozi čas v odvisnosti od velikosti populacije po 100 generacijah. Po pričakovanju je vrednost cenovne funkcije sorazmerna z velikostjo populacije. Najboljšo vrednost doseže pri velikosti 500, pri čemer ni velike razlike v velikostih 200 ali 500. Pri vseh vrednostih rešitve dosežejo konvergenco po približno 50 generacijah.

Zaradi stohastičnosti algoritma je potrebno simulacijo istega sistema kemijskih reakcij ponoviti večkrat. Tako dobimo približek resnične verjetnostne porazdelitve posamezne kemijske zvrsti [20]. Pri izračunu cenovne funkcije za vsakega kandidata algoritem SSA poženemo večkrat (n_{SSA}). Na sliki 3.4b je prikazana odvisnost končne rešitve od števila zagonov, kjer je razvidno, da manjše število ponovitev po določenem času doseže boljše rešitve. Ta lastnost je odvisna od specifičnega sistema kemijskih reakcij. Rezultat na-

miguje, da je opazovani sistem genskega stikala zelo robusten, saj se pri večjem številu ponovitev vrednost cenovne funkcije ne spremeni bistveno.

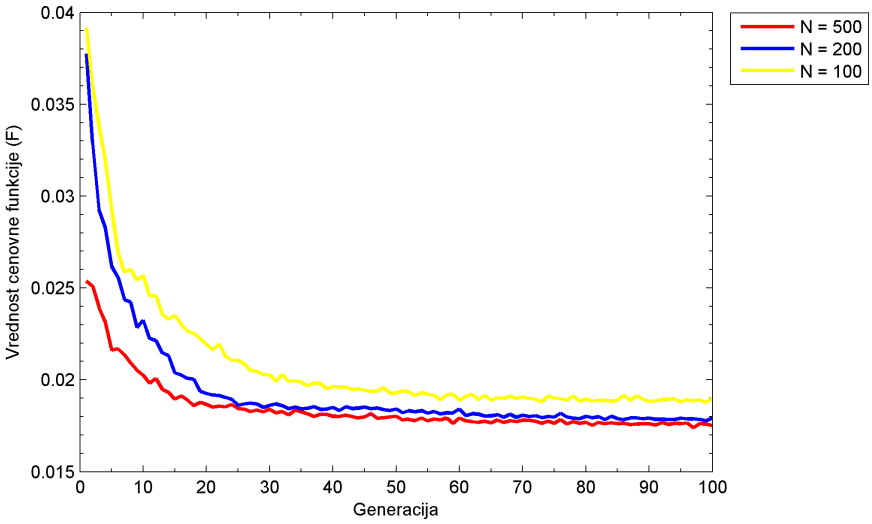


Slika 3.3: Vrednost cenovne funkcije najboljšega kandidata za rešitev na koncu evolucije v odvisnosti od števila generacij ($N = 100,\ n_{SSA} = 20$).

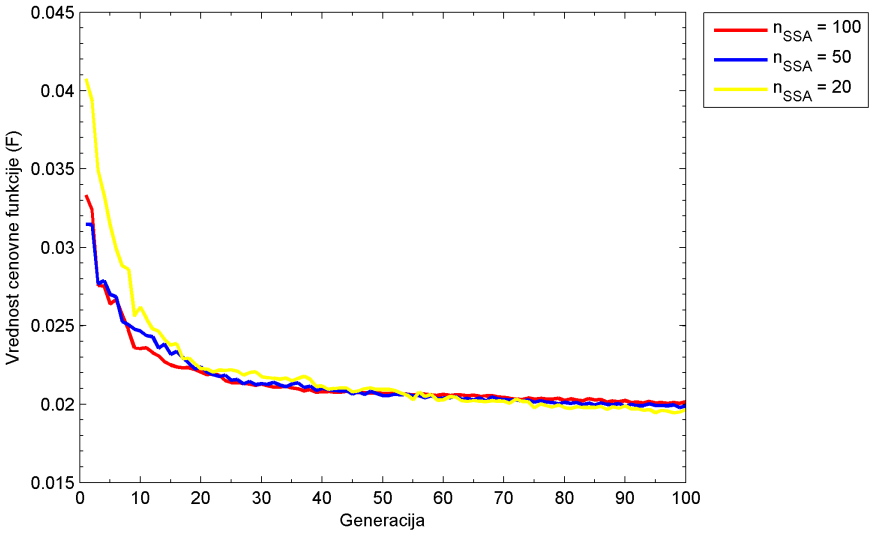
3.5 Optimizacija parametrov modela

Po določitvi optimalnih nastavitev evolucijskega algoritma smo minimalni model bistabilnega stikala prilagodili eksperimentalnim podaktom (razdelek 2.6), kjer smo namesto poročevalskih fluorescentnih proteinov za opazovanje sistema uporabili encim luciferazo. Model smo prilagodili dvem eksperimentom preklopa, ki se razlikujeta v časovnem poteku indukcije. V prvem eksperimentu preizkušamo preklop iz stanja A (indukcija s PI) v stanje B (indukcija z ER), pri drugem pa preklop v obratno smer. Oba eksperimenta sta potekala 10 dni, pri čemer je bil antibiotik za prvo stanje dodan takoj, menjava antibiotika pa je bila opravljena po drugem dnevu.

Rezultati optimizacije z evolucijskim algoritmnom so prikazani na sliki 3.5, kjer je



(a)



(b)

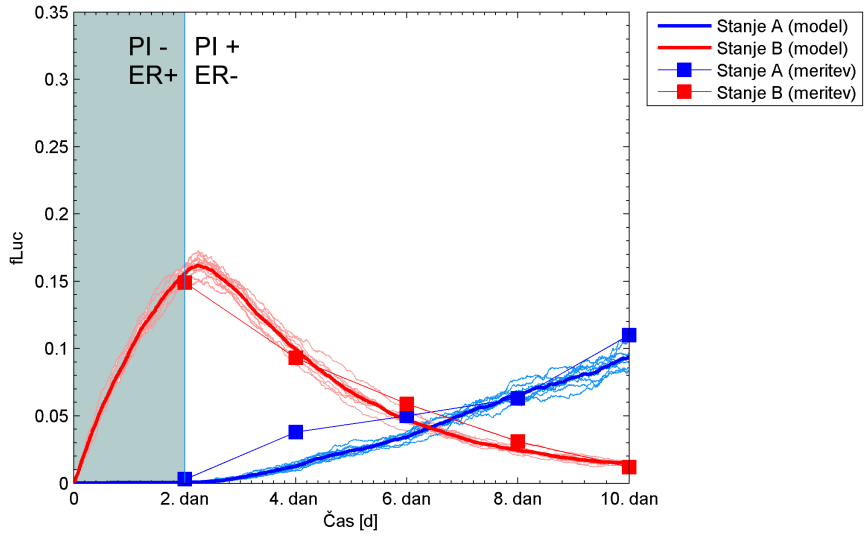
Slika 3.4: (a) Vrednost cenovne funkcije najboljšega kandidata za rešitev skozi čas v odvisnosti od velikosti populacije ($I_{max} = 100,\ n_{SSA} = 20$). (b) Vrednost cenovne funkcije najboljšega kandidata za rešitev skozi čas v odvisnosti od števila ponovitev izračuna cenovne funkcije (zagona algoritma SSA) ($I_{max} = 100,\ N = 100$).

opazno kvantitativno ujemanje med modelom in meritvami z relativno majhno srednjo kvadratno napako (vrednost cenovne funkcije 0.02 fLuc), kar dodatno potrjuje pravilnost opisa modela.

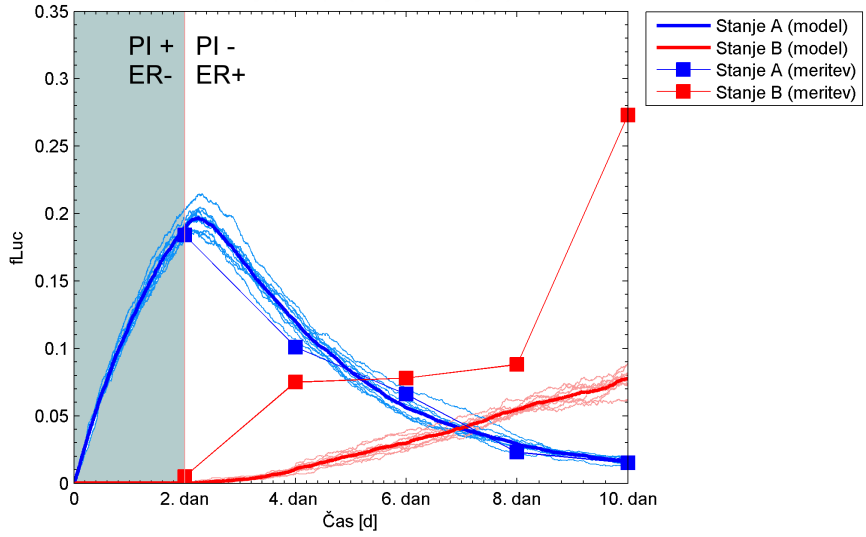
Oznaka	$\vec{a_1}$	$\vec{\mu}$	$\vec{\sigma}$	$\vec{c_v}$	$\vec{c}$	$\vec{c} \circ \vec{\mu}$
k_{da}	2.168	2.233	1.158	0.518	1.0	**2.233**
k_{da}	0.825	1.319	0.642	0.486	$1.0 \cdot 10^{-4}$	$\mathbf{1.319 \cdot 10^{-4}}$
k_{b}	4.061	3.590	2.671	0.744	1.0	**3.590**
k_{db}	0.602	1.145	0.793	0.692	$1.0 \cdot 10^{-4}$	$\mathbf{1.145 \cdot 10^{-4}}$
d_{T}	4.858	4.264	3.704	0.868	$1.0 \cdot 10^{-5}$	$\mathbf{4.264 \cdot 10^{-5}}$
v_{VV}	0.925	1.093	0.476	0.435	$1.0 \cdot 10^{-4}$	$\mathbf{1.093 \cdot 10^{-4}}$
d_{R}	0.118	0.127	0.068	0.538	$1.0 \cdot 10^{-5}$	$\mathbf{1.272 \cdot 10^{-6}}$
l_{PI}	15.013	9.929	6.609	0.665	$1.0 \cdot 10^{-1}$	$\mathbf{9.929 \cdot 10^{-1}}$
l_{ER}	3.541	7.613	5.451	0.716	$1.0 \cdot 10^{-c}$	$\mathbf{7.613 \cdot 10^{-1}}$
f_{R}	0.257	0.299	0.150	0.501	$1.0 \cdot 10^{-3}$	$\mathbf{2.992 \cdot 10^{-4}}$
v_{CMV}	5.157	4.125	1.798	0.436	$5.0 \cdot 10^{-2}$	$\mathbf{2.256 \cdot 10^{-5}}$
a_{T}	0.879	0.753	0.090	0.119	1.0	$\mathbf{7.537 \cdot 10^{-1}}$
v_{VK}	29.813	30.816	5.838	0.189	$1.00 \cdot 10^{-2}$	$\mathbf{3.371 \cdot 10^{-5}}$
$a_{\mathrm{PI/ER}}$	1.052	1.401	0.718	0.513	1.0	**1.401**
k_{KRAB}	0.519	0.532	0.038	0.071	4.0	**2.131**

Tabela 3.4: Rezultati optimizacije parametrov modela iz množice 20 % najboljših rešitev v 50 poizkusih, kjer stolpec $\vec{a_1}$ predstavlja najboljšo najdeno rešitev, $\vec{\mu}$ povprečne vrednosti, $\vec{\sigma}$ standardni odklon, $\vec{c_v}$ korelacijski coeficient ($c_v = \sigma/\mu$), $\vec{c}$ vektor začetnih vrednosti ter $\vec{c} \circ \vec{\mu}$ predvideno resnično vrednost parametra.

Ponovili smo 50 poizkusov zagona evolucijskega algoritma, jih razvrstili po naraščajoči vrednosti cenovne funkcije najboljšega kandidata ter med seboj primerjali zgornjih (najboljših) 20%. Rezultati so zbrani v tabeli 3.4. Izpostavljena je najboljša rešitev $\vec{a_1}$, katere vrednosti se v povprečju nahajajo v intervalu $\mu \pm 0.2 \cdot \mu$, kar nakazuje, da evolucijski algoritem najde (isti) lokalni minimum z veliko verjetnostjo. Dobljene številske vrednosti lahko pomagajo pojasniti podprocese, ki sestavljajo sistem bistabilnega genskega stikala. Dodatno informacijo podaja korelacijski koeficient, ki ocenjuje napovedno vrednost posameznega parametra, tj. nižja vrenost nakazuje večjo verjetnost, da je dobljena vrednost



(a)



(b)

Slika 3.5: Rezultati optimizacije parametrov, ki minimizira razdaljo med petimi eksperimentalnimi podatki za posamezno stanje in rezultati modela v pripadajočih točkah (parametri evolucijskega algoritma: $I_{max} = 100$, $N = 150$, $n_{SSA} = 20$). (a) Preklop iz stanja A v stanje B. (b) Preklop iz stanja B v stanje A.

parametra dejansko resnična. Na podlagi dobljenih vrednosti lahko postavimo naslednje hipoteze:

- Delovanje TAL efektorjev z represorsko domeno je približno dvakrat boljše od aktivatorjev, $k_{\mathrm{KRAB}} = 2.131$ ($c_v = 0.071$). Pri vseh kombinacijah razporeditev aktivatorjev in represorjev na 10 vezavnih mestih, bi bilo torej $\frac{2}{3}$ takih, kjer ima represor večji vpliv od aktivatorja. Predvidena določitev vpliva pri kombinacijah aktivatorja (V) in represorja (K) na desetih vezavnih mestih brez upoštevanja vrstnega reda je prikazana v tabeli 3.5. Pri tem upoštevamo, da je v resnici *afiniteta vezave proteinov TAL ne glede na efektorsko domeno enaka*, ter z oceno navidezno intenzivnejše vezave ocenimo vpliv posamezne kombinacije.

- Opazna je manjša razlika v vezavi TAL efektorjev z vezavno domeno A ($k_{\mathrm{a}} = 2.233$, $c_v = 0.518$) v primerjavi s TAL efektorji z vezavno domeno B ($k_{\mathrm{b}} = 3.590$, $c_v = 0.744$), ki predvideva intenzivnejšo vezavo proteinov TAL A. Po drugi strani je verjetnost disociacije za proteine TAL B manjša ($k_{\mathrm{da}} = 1.319 \cdot 10^{-4}$, $k_{\mathrm{db}} = 1.145 \cdot 10^{-4}$). Razlika v delovanju TAL efektorjev je bila zaznana tudi v eksperimentalnem delu (podatki niso prikazani).

- Razgradnja TAL efektorjev je intenzivnejša od poročevalskih proteinov $d_{\mathrm{T}} = 4.264 \cdot 10^{-5}$ ($c_v = 0.435$) $> d_{\mathrm{R}} = 1.272 \cdot 10^{-6}$ ($c_v = 0.538$). Ta zaključek je vprašljiv v primeru encima luciferaze, saj ta sodi med proteine z nizkim razgradnim časom (red velikosti nekaj ur, [21]), medtem ko razgradni čas za TAL proteine ni znan. Zaključek je spodbuden v primeru uporabe fluorescentnih poročevalskih proteinov, saj je razgradni čas le-teh reda velikosti nekaj dni v primeru BFP [42].

- Odkrito je puščanje ob inducibilnih sistemov ($l_{\mathrm{PI}} = 9.929 \cdot 10^{-1}$, $c_v = 0.665$ in $l_{\mathrm{ER}} = 7.613 \cdot 10^{-1}$, $c_v = 0.716$), kar potrjuje robustnost topologije stikala, saj preklop deluje tudi ob puščanju. Po štetju števila sprožitve reakcij, ki so posledica puščanja inducibilnega sistema v primerjavi z običajno indukcijo smo ugotovili 1.9% puščanje pristinamicinskega in 2.5% puščanje eritromicinskega inducibilnega sistema.

- Hitrost izražanja pri konstitutivnem promotorju je pričakovano manjša od hitrosti izražanja aktiviranega minimalnega promotorja, tj. $v_{\mathrm{CMV}} < v_{\mathrm{VK}} < v_{\mathrm{VV}}$.

Kombinacija	Vpliv na izražanje	Kombinacija	Vpliv na izražanje
VVVVVVVVVV	Aktivacija	VVVVKKKKKK	Represija
VVVVVVVVVK	Aktivacija	VVVKKKKKKK	Represija
VVVVVVVVKK	Aktivacija	VVKKKKKKKK	Represija
VVVVVVVKKK	Aktivacija	VKKKKKKKKK	Represija
VVVVVVKKKK	Represija	KKKKKKKKKK	Represija
VVVVVKKKKK	Represija		

Tabela 3.5: Možne kombinacije aktivatorjev in represorjev na desetih vezavnih mestih pred promotorjem brez upoštevanja vrstnega reda in predviden vpliv na izražanje pripadajočega gena na podlagi izračunanega parametra $k_{\mathrm{KRAB}} = 2.131$ ($c_v = 0.071$).

Številske vrednosti parametrov torej omogočajo neposredno primerjavo različnih aspektov sistema in predstavljajo podlago za nadaljno eksperimentalno delo, ki bi potrdilo pravilnost domnev in s tem razumevanje delovanja kompleksnega genskega regulatornega omrežja v obliki bistabilnega stikala s pozitivno povratno zanko. Poleg tega je mogoče na podlagi dobljenih rezultatov načrtovati nove poizkuse, s katerimi bi delovanje stikala še dodatno izboljšali (razdelek 3.6). Ne glede na specifičnost obravnavanega primera je pristop univerzalen in ga je mogoče uporabiti za katerokoli gensko regulatorno omrežje ali poljuben sistem kemijskih reakcij.

3.6 Napovedna vrednost modela

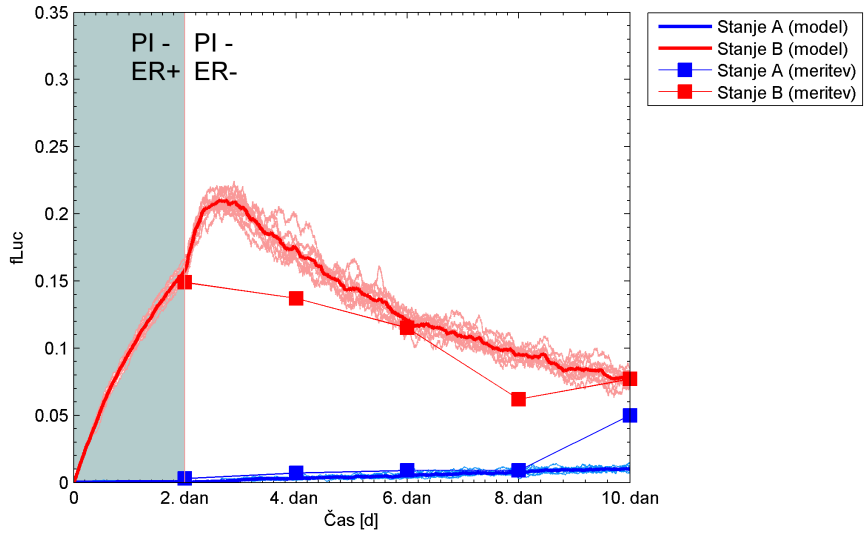
Dobljene vrednosti parametrov smo uporabili pri preizkusu napovedi dodatnega poizkusa na istem genskem regulatornem omrežju z drugačnim časovnim potekom indukcije. Tako smo po drugem dnevu indukcije celice prestavili v medij, ki ne vsebuje antibiotika, s čimer bi pokazali stabilnost enega ali drugega stanja. Glede na začetni antibiotik torej pričakujemo izražanje ustreznega poročevalskega proteina, ob ostranitvi pa ohranitev stabilnega stanja. Na sliki 3.6a je prikazana *napoved* modela s predhodno naučenimi parametri, ki se ujema z dejansko izmerjenimi vrednostmi (vrednost srednje kvadratne napake je enaka 0.019 fLuc), ter pravilno napove postopno padanja izražanja s časom.

Delovanje genskega regulatornega omrežja je neposredno odvisno od količin konstruktov, ki jih preko postopka transfekcije vstavimo v celice. Te so podane neposredno v masi DNA. Za delovanje stikala nas tako zanimajo optimalna razmerja količin posameznih

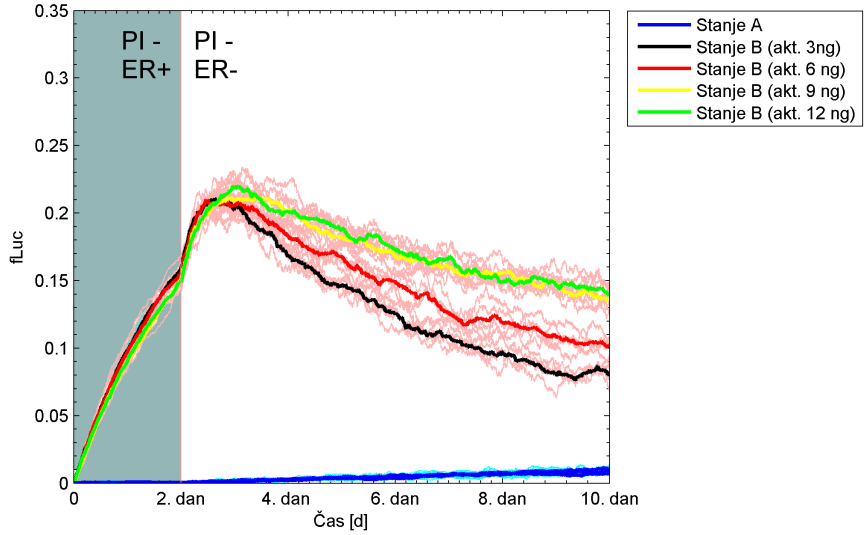
konstruktov stikala (slika 2.5). Vsi prikazani eksperimentalni podatki so rezultat istega poizkusa, pri katerem so bile uporabljene količine $P_{min\#0} = 15$ ng, $P_{min\#1} = 2.5$ ng, $P_{min\#2} = 15$ ng, $P_{min\#3} = 2.5$ ng, $P_{CMV\#4} = 15$ ng, $P_{CMV\#5} = 15$ ng, $P_{CMV\#6} = 15$ ng, $P_{CMV\#7} = 15$ ng, $P_{CMV\#8} = 60$ ng, $P_{CMV\#9} = 60$ ng. Pri eksperimentalnem delu se je za pomemben faktor za pravilno delovanje stikala izkazalo *razmerje med aktivatorskimi in represorskimi konstrukti v prvem nivoju stikala.* Konkretno to pomeni količine DNA, ki jih lahko prilagajamo ter neposredno določajo količino plazmidov, ki bodo na koncu prišli v celice.

Na sliki 3.6b preizkusimo različne vrednosti konstruktov, ki vključujejo aktivator, torej količine $P_{min\#1}$ in $P_{min\#3}$, ostale vrednosti pa pustimo enake začetnim. Izkaže se, da pri večjih količinah aktivatorja stanje ostane stabilno dalj časa, vendar pa je preklop v tem primeru počasnejši. Model je torej mogoče uporabljati za načrtovanje nadaljnih poizkusov ter napovedovanje odziva sistema pri različnih pogojih.

Kljub ujemanju modela z eksperimentalnimi podatki ter omejitvami na biološko relevantne vrednosti so dobljeni rezultati približek dejanskim parametrom, vendar pa predstavljajo dobro oceno dejanskega stanja ter možnost za postavljanje novih hipotez o opazovanem biokemijskem procesu. Z več vhodnimi podatki (eksperimenti) bi bila v principu možna še boljša napoved in opis delovanja sistema.



(a)



(b)

Slika 3.6: Stabilno stanje B, kjer po drugem dnevu odstranimo antibiotik. (a) Primerjava napovedi modela z eksperimentalnimi podatki. Srednja kvadratna napaka modela znaša 0.019 fLuc. (b) Napovedi modela pri različnih količinah konstruktov $P_{min\#1}$ ter $P_{min\#3}$. Količine ostalih konstruktov so enake kot v razdelku 3.6.

4 Zaključek

V pričujočem delu smo opisali model kompleksnejšega sintetičnega genskega regulatornega omrežja s ciljem ugotovitve ključnih lastnosti opazovanega sistema. Predstavili smo princip, ki upošteva tekmovanje dveh različnih proteinov z enako DNA vezavno domeno za vezavna mesta iste vrste ter slednje opiše z minimalnim številom reakcij. S postopki analize odziva modela smo uspeli izpostaviti ključne reakcije in kemijske zvrsti, ter tako opazno zmanjšali prostorsko in časovno zahtevnost.

Algoritem za stohastično simulacijo smo združili z evolucijskim algoritmom za optimizacijo parametrov, ter celoten sistem prilagodili za izvanjanje na paralelnem grafičnem procesorju. Z ustrezno implementacijo ter optimizacijo programske kode smo dosegli vsaj dvakrat večje pohitritve v primerjavi z dosedanjimi implementacijami. S pridom smo izkoristili navidezno slabo lastnost evolucijskega algoritma, ki v splošnem zagotavlja lokalni minimum cenovne funkcije. Z vnaprejšnjo določitvijo začetne ocene reda velikosti parametrov posameznih reakcij smo z optimizacijo izračunali parametre, skladne z biološko intuicijo.

Z optimizacijo parametrov smo opisani model uspešno prilagodili dvem eksperimen-

talnim meritvam, s čimer smo potrdili pravilnost opisa. Dodatno smo s pomočjo modela lahko z majhno napako ocenili kvantitativni odziv tretjega eksperimenta, ki ni bil del optimizacije, kar nam omogoča načrtovanje nadaljnih poizkusov. Rezultat nakazuje približevanje sposobnosti natančnega opisa kompleksnega biološkega sistema.

Dobljeni rezultati omogočajo oceno relativnih razmerij v pogostosti izvajanja posameznih reakcij. Na ta način nam omogočajo odkriti znanje, ki delno pojasni dobljene eksperimentalne podatke. Z morebitnim odkritjem novih lastnosti sistema lahko v prihodnosti z dopolnitvijo modela pridobimo ali potrdimo nova, dodatna znanja, ki jih v danem trenutku še ne poznamo.

Opisani princip je v splošnem mogoče razširiti na poljubno gensko regulatorno omrežje. Z uporabo dodatnih eksperimentalnih podatkov bi bilo v principu mogoče dobiti še boljšo oceno za odziv sistema. To vključuje več instanc izvedenih eksperimentov ali opazovanje dodatnih aspektov sistema.

Rezultati ponujajo vpogled v procese na nižjih nivojih od opazovanega ter predstavljajo izhodišča za nadaljne eksperimentalno delo, ki bi potrdilo napovedi modela. S cikli izmenjave podatkov med eksperimentalnim in teoretičnim delom se tako približujemo boljšemu razumevanju opazovanega biološkega procesa.

A Dodatek

A.1 Osnovni model bistabilnega stikala s pozitivno povratno zanko

V tem dodatku sledi opis modela, ki ustreza shemi v razdelku 2.1.3. Opazovane kemijske zvrsti so opisane v tabeli A.1. Zaradi krajšega opisa smo nekatere reakcije opisali združeno, saj so enake za združene konstrukte. Indeksi oštevilčevanja se sklicujejo na sliko 2.5. Skupno sistem sestavlja 46 entitet (kemijskih zvrsti) in 90 reakcij.

Nivo 1: Vezava aktivatorjev in represorjev

Konstrukta z vezavnimi mesti tipa A

$$[P_{min\#i}] + [T_{AV}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}}}{\rightleftharpoons}} [T_{AV}\_P_{min\#i}] \tag{A.1}$$

$$[P_{min\#i}] + [T_{AK}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}} \cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{AK}\_P_{min\#i}] \tag{A.2}$$

$$[T_{AK}\_P_{min\#i}] + [T_{AK}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}} \cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{AK}\_T_{AK}\_P_{min\#i}] \tag{A.3}$$

$$[T_{AV}\_P_{min\#i}] + [T_{AK}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}} \cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{AV}\_T_{AK}\_P_{min\#i}] \tag{A.4}$$

Vrsta	Oznaka	Kemijska zvrst	Opomba
Proteini in antibiotiki	T_{AK}	Represor TALA:KRAB	
	T_{AV}	Aktivator TALA:VP16	
	T_{BK}	Represor TALB:KRAB	
	T_{BV}	Aktivator TALB:VP16	
	PIP_K	Pristinamicinski inducibilni represor	
	E_K	Eritromicinski inducibilni represor	
	PI	Antibiotik pristinamicin	
	ER	Antibiotik eritromicin	
	PI_PIP_K	Kompleks pristinamicin-PIP_K	
	ER_E_K	Kompleks eritromicin-E_K	
	BFP	Poročevalski protein BFP	
	mCT	Poročevalski protein mCitrine	
Promotorji (nivo 1)	$P_{\min\#i}$	Minimalni promotor	$i \in 0,1,2,3$
	$T_{AK}\_P_{min\#i}$	Delno zasedeni min. promotor	$i \in 0,1$
	$T_{BK}\_P_{min\#i}$	Delno zasedeni min. promotor	$i \in 2,3$
	$T_{AV}\_P_{min\#i}$	Delno zasedeni min. promotor	$i \in 0,1$
	$T_{BV}\_P_{min\#i}$	Delno zasedeni min. promotor	$i \in 2,3$
	$T_{AV}\_T_{AV}\_P_{min\#i}$	Polno zasedeni min. promotor	$i \in 0,1$
	$T_{AK}\_T_{AK}\_P_{min\#i}$	Polno zasedeni min. promotor	$i \in 0,1$
	$T_{AV}\_T_{AK}\_P_{min\#i}$	Polno zasedeni min. promotor	$i \in 0,1$
	$T_{BV}\_T_{BV}\_P_{min\#i}$	Polno zasedeni min. promotor	$i \in 2,3$
	$T_{BK}\_T_{BK}\_P_{min\#i}$	Polno zasedeni min. promotor	$i \in 2,3$
	$T_{BV}\_T_{BK}\_P_{min\#i}$	Polno zasedeni min. promotor	$i \in 2,3$
Promotorji (nivo 2)	$P_{\mathrm{CMV}\#i}$	Konstitutivni promotor	$i \in 4,5,6,7$
	$PIP_K\_P_{\mathrm{CMV}\#i}$	Zasedeni kon. promotor	$i \in 4,5$
	$E_K\_P_{\mathrm{CMV}\#i}$	Zasedeni kon. promotor	$i \in 6,7$
Promotorji (nivo 3)	$P_{\mathrm{CMV}\#i}$	Konstitutivni promotor	$i \in 8,9$

Tabela A.1: Opazovane kemijske zvrsti v osnovnem modelu.

$$[T_{AV}\_P_{min\#i}] + [T_{AV}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}}}{\rightleftharpoons}} [T_{AV}\_T_{AV}\_P_{min\#i}] \tag{A.5}$$

$$i \in \{0, 1\};$$

Konstrukta z vezavnimi mesti tipa B

$$[P_{min\#j}] + [T_{BV}] \underset{k_{\mathrm{db}}}{\overset{k_{\mathrm{b}}}{\rightleftharpoons}} [T_{BV}\_P_{min\#j}] \tag{A.6}$$

$$[P_{min\#j}] + [T_{BK}] \underset{k_{\mathrm{db}}}{\overset{k_{\mathrm{b}} \cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{BK}\_P_{min\#j}] \tag{A.7}$$

$$[T_{BK}\_P_{min\#j}] + [T_{BK}] \underset{k_{\mathrm{db}}}{\overset{k_{\mathrm{b}} \cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{BK}\_T_{BK}\_P_{min\#j}] \tag{A.8}$$

$$[T_{BV}\_P_{min\#j}] + [T_{BK}] \underset{k_{\mathrm{db}}}{\overset{k_{\mathrm{b}} \cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{BV}\_T_{BK}\_P_{min\#j}] \tag{A.9}$$

$$[T_{BV}\_P_{min\#j}] + [T_{BV}] \underset{k_{\mathrm{db}}}{\overset{k_{\mathrm{b}}}{\rightleftharpoons}} [T_{BV}\_T_{BV}\_P_{min\#j}] \tag{A.10}$$

$$j \in \{2, 3\};$$

Nivo 1: Transkripcija in translacija

Konstrukt #0

$$[T_{AV}\_P_{min\#0}] \overset{v_{\mathrm{V}}}{\rightharpoonup} [T_{AV}\_P_{min\#0}] + [T_{BK}] + [BFP] \tag{A.11}$$

$$[T_{AV}\_T_{AK}\_P_{min\#0}] \overset{v_{\mathrm{VK}}}{\rightharpoonup} [T_{AV}\_T_{AK}\_P_{min\#0}] + [T_{BK}] + [BFP] \tag{A.12}$$

$$[T_{AV}\_T_{AV}\_P_{min\#0}] \overset{v_{\mathrm{VV}}}{\rightharpoonup} [T_{AV}\_T_{AV}\_P_{min\#0}] + [T_{BK}] + [BFP] \tag{A.13}$$

Konstrukt #1

$$[T_{AV}\_P_{min\#1}] \overset{v_{\mathrm{V}}}{\rightharpoonup} [T_{AV}\_P_{min\#1}] + [T_{AV}] \tag{A.14}$$

$$[T_{AV}\_T_{AK}\_P_{min\#1}] \overset{v_{\mathrm{VK}}}{\rightharpoonup} [T_{AV}\_T_{AK}\_P_{min\#1}] + [T_{AV}] \tag{A.15}$$

$$[T_{AV}\_T_{AV}\_P_{min\#1}] \overset{v_{\mathrm{VV}}}{\rightharpoonup} [T_{AV}\_T_{AV}\_P_{min\#1}] + [T_{AV}] \tag{A.16}$$

Konstrukt #2

$$[T_{BV}\_P_{min\#2}] \overset{v_{\mathrm{V}}}{\rightharpoonup} [T_{BV}\_P_{min\#2}] + [T_{AK}] + [mCT] \tag{A.17}$$

$$[T_{BV}\_T_{BK}\_P_{min\#2}] \overset{v_{\mathrm{VK}}}{\rightharpoonup} [T_{BV}\_T_{BK}\_P_{min\#2}] + [T_{AK}] + [mCT] \tag{A.18}$$

$$[T_{BV}\_T_{BV}\_P_{min\#2}] \overset{v_{\mathrm{VV}}}{\rightharpoonup} [T_{BV}\_T_{BV}\_P_{min\#2}] + [T_{AK}] + [mCT] \tag{A.19}$$

Konstrukt #3

$$[T_{BV}\_P_{min\#3}] \xrightarrow{v_{\mathrm{V}}} [T_{BV}\_P_{min\#3}] + [T_{BV}] \tag{A.20}$$

$$[T_{BV}\_T_{BK}\_P_{min\#3}] \xrightarrow{v_{\mathrm{VK}}} [T_{BV}\_T_{BK}\_P_{min\#3}] + [T_{BV}] \tag{A.21}$$

$$[T_{BV}\_T_{BV}\_P_{min\#3}] \xrightarrow{v_{\mathrm{VV}}} [T_{BV}\_T_{BV}\_P_{min\#3}] + [T_{BV}] \tag{A.22}$$

Nivo 2: Vezava inducibilnih proteinov

$$[P_{\mathrm{CMV}\#i}] + [PIP_K] \underset{k_{\mathrm{dI}}}{\overset{k_{\mathrm{I}}}{\rightleftharpoons}} [PIP_K\_P_{\mathrm{CMV}\#i}] \tag{A.23}$$

$$i \in \{4, 5\};$$

$$[P_{\mathrm{CMV}\#j}] + [E_K] \underset{k_{\mathrm{dI}}}{\overset{k_{\mathrm{I}}}{\rightleftharpoons}} [E_K\_P_{\mathrm{CMV}\#j}] \tag{A.24}$$

$$j \in \{6, 7\};$$

Nivo 2: Interakcija molekul antibiotikov z inducibilnimi proteini

$$[PIP_K\_P_{\mathrm{CMV}\#i}] + [PI] \xrightarrow{I_{\mathrm{I}}} [P_{\mathrm{CMV}\#i}] + [PI\_PIP_K] \tag{A.25}$$

$$i \in \{4, 5\};$$

$$[E_K\_P_{\mathrm{CMV}\#j}] + [ER] \xrightarrow{I_{\mathrm{I}}} [E_{\mathrm{CMV}\#j}] + [ER\_E_K] \tag{A.26}$$

$$j \in \{6, 7\};$$

Nivo 2: Transkripcija in translacija

$$[P_{\mathrm{CMV}\#4}] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#4}] + [T_{BK}] \tag{A.27}$$

$$[P_{\mathrm{CMV}\#5}] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#5}] + [T_{AV}] \tag{A.28}$$

$$[P_{\mathrm{CMV}\#6}] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#6}] + [T_{AK}] \tag{A.29}$$

$$[P_{\mathrm{CMV}\#7}] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#7}] + [T_{BV}] \tag{A.30}$$

Nivo 3: Transkripcija in translacija

$$[P_{\mathrm{CMV}\#8}] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#8}] + [PIP_K] \tag{A.31}$$

$$[P_{\mathrm{CMV}\#9}] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#9}] + [E_K] \tag{A.32}$$

Degradacija kemijskih zvrsti

$$[T_{AK}] \xrightarrow{d_{\mathrm{T}}} \emptyset \tag{A.33}$$

$$[T_{AV}] \xrightarrow{d_{\mathrm{T}}} \emptyset \tag{A.34}$$

$$[T_{BK}] \xrightarrow{d_{\mathrm{T}}} \emptyset \tag{A.35}$$

$$[T_{BV}] \xrightarrow{d_{\mathrm{T}}} \emptyset \tag{A.36}$$

$$[PIP_K] \xrightarrow{d_{\mathrm{I}}} \emptyset \tag{A.37}$$

$$[E_K] \xrightarrow{d_{\mathrm{I}}} \emptyset \tag{A.38}$$

$$[BFP] \xrightarrow{d_{\mathrm{R}}} \emptyset \tag{A.39}$$

$$[mCT] \xrightarrow{d_{\mathrm{R}}} \emptyset \tag{A.40}$$

A.2 Minimalni model bistabilnega stikala s pozitivno povratno zanko

V tem dodatku sledi opis modela bistabilnega stikala, pri katerem so bile določene reakcije odstranjene. Opazovane kemijske zvrsti so opisane v tabeli A.2. Zaradi krajšega opisa smo nekatere reakcije opisali združeno, saj so enake za združene konstrukte. Indeksi oštevilčevanja se sklicujejo na sliko 2.5. Za potrebe prilagajanja modela eksperimentalnim meritvam smo dodali parameter b za asimetrično vezavo TALA in TALB proteinov ter reakcije puščanja inducibilnega sistema. Skupno sistem sestavlja 28 entitet (kemijskih zvrsti) in 46 reakcij.

Nivo 1: Vezava aktivatorjev in represorjev

Konstrukta z vezavnimi mesti tipa A

$$[P_{min\#i}] + 2[T_{AK}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}} \cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{AK}\_T_{AK}\_P_{min\#i}] \tag{A.41}$$

$$[P_{min\#i}] + 2[T_{AV}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}}}{\rightleftharpoons}} [T_{AV}\_T_{AV}\_P_{min\#i}] \tag{A.42}$$

$$[P_{min\#i}] + [T_{AK}] + [T_{AV}] \underset{k_{\mathrm{da}}}{\overset{k_{\mathrm{a}} \cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{AV}\_T_{AK}\_P_{min\#i}] \tag{A.43}$$

$$i \, \epsilon \, \{0, 1\};$$

Vrsta	Oznaka	Kemijska zvrst	Opomba
Proteini in antibiotiki	T_{AK}	Represor TALA:KRAB	
	T_{AV}	Aktivator TALA:VP16	
	T_{BK}	Represor TALB:KRAB	
	T_{BV}	Aktivator TALB:VP16	
	BFP	Poročevalski protein BFP	
	mCT	Poročevalski protein mCitrine	
	PI	Antibiotik pristinamicin	
	ER	Antibiotik eritromicin	
Promotorji (nivo 1)	$P_{\mathrm{min}\#i}$	Minimalni promotor	$i \in 0,1,2,3$
	$T_{AV}\_T_{AV}\_P_{min\#i}$	Polno zasedeni min. promotor	$i \in 0,1$
	$T_{AK}\_T_{AK}\_P_{min\#i}$	Polno zasedeni min. promotor	$i \in 0,1$
	$T_{AV}\_T_{AK}\_P_{min\#i}$	Polno zasedeni min.promotor	$i \in 0,1$
	$T_{BV}\_T_{BV}\_P_{min\#i}$	Polno zasedeni min. promotor	$i \in 2,3$
	$T_{BK}\_T_{BK}\_P_{min\#i}$	Polno zasedeni min. promotor	$i \in 2,3$
	$T_{BV}\_T_{BK}\_P_{min\#i}$	Polno zasedeni min. promotor	$i \in 2,3$
Promotorji (nivo 2)	$P_{\mathrm{CMV}\#i}$	Konstitutivni promotor	$i \in 4,5,6,7$

Tabela A.2: Opazovane kemijske zvrsti v minimalnem modelu.

Konstrukta z vezavnimi mesti tipa B

$$[P_{min\#i}] + 2[T_{BK}] \underset{k_{\mathrm{db}}}{\overset{k_{\mathrm{b}} \cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{BK}\_T_{BK}\_P_{min\#j}] \tag{A.44}$$

$$[P_{min\#i}] + 2[T_{BV}] \underset{k_{\mathrm{db}}}{\overset{k_{\mathrm{b}}}{\rightleftharpoons}} [T_{BV}\_T_{BV}\_P_{min\#j}] \tag{A.45}$$

$$[P_{min\#i}] + [T_{BK}] + [T_{BV}] \underset{k_{\mathrm{db}}}{\overset{b \cdot k_{\mathrm{b}} \cdot k_{\mathrm{KRAB}}}{\rightleftharpoons}} [T_{BV}\_T_{BK}\_P_{min\#j}] \tag{A.46}$$

$$j \in \{2,3\};$$

Nivo 1: Transkripcija in translacija

Konstrukt #0

$$[T_{AV}\_T_{AK}\_P_{min\#0}] \xrightarrow{v_{\mathrm{VK}}} [T_{AV}\_T_{AK}\_P_{min\#0}] + [T_{BK}] + [BFP] \tag{A.47}$$

$$[T_{AV}\_T_{AV}\_P_{min\#0}] \xrightarrow{v_{\mathrm{VV}}} [T_{AV}\_T_{AV}\_P_{min\#0}] + [T_{BK}] + [BFP] \tag{A.48}$$

Konstrukt #1

$$[T_{AV}\_T_{AK}\_P_{min\#1}] \xrightarrow{v_{\mathrm{VK}}} [T_{AV}\_T_{AK}\_P_{min\#1}] + [T_{AV}] \tag{A.49}$$

$$[T_{AV}\_T_{AV}\_P_{min\#1}] \xrightarrow{v_{\mathrm{VV}}} [T_{AV}\_T_{AV}\_P_{min\#1}] + [T_{AV}] \tag{A.50}$$

Konstrukt #2

$$[T_{BV}\_T_{BK}\_P_{min\#2}] \xrightarrow{v_{\mathrm{VK}}\cdot v_{\mathrm{b}}} [T_{BV}\_T_{BK}\_P_{min\#2}] + [T_{AK}] + [mCT] \tag{A.51}$$

$$[T_{BV}\_T_{BV}\_P_{min\#2}] \xrightarrow{v_{\mathrm{VV}}\cdot v_{\mathrm{b}}} [T_{BV}\_T_{BV}\_P_{min\#2}] + [T_{AK}] + [mCT] \tag{A.52}$$

Konstrukt #3

$$[T_{BV}\_T_{BK}\_P_{min\#3}] \xrightarrow{v_{\mathrm{VK}}\cdot v_{\mathrm{b}}} [T_{BV}\_T_{BK}\_P_{min\#3}] + [T_{BV}] \tag{A.53}$$

$$[T_{BV}\_T_{BV}\_P_{min\#3}] \xrightarrow{v_{\mathrm{VV}}\cdot v_{\mathrm{b}}} [T_{BV}\_T_{BV}\_P_{min\#3}] + [T_{BV}] \tag{A.54}$$

Nivo 2: Transkripcija in translacija ob indukciji

$$[P_{\mathrm{CMV}\#4}] + [PI] \xrightarrow{v_{\mathrm{CMV}}\cdot a_{\mathrm{PI/ER}}} [P_{\mathrm{CMV}\#4}] + [T_{BK}] \tag{A.55}$$

$$[P_{\mathrm{CMV}\#5}] + [PI] \xrightarrow{v_{\mathrm{CMV}}\cdot a_{\mathrm{PI/ER}}} [P_{\mathrm{CMV}\#5}] + [T_{AV}] \tag{A.56}$$

Analogno opišemo indukcijo stanja B:

$$[P_{\mathrm{CMV}\#6}] + [ER] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#6}] + [T_{AK}] \tag{A.57}$$

$$[P_{\mathrm{CMV}\#7}] + [ER] \xrightarrow{v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#7}] + [T_{BV}] \tag{A.58}$$

Nivo 2: Puščanje inducibilnega sistema

$$[P_{\mathrm{CMV}\#4}] \xrightarrow{l_{\mathrm{PI}}\cdot v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#4}] + [T_{BK}] \tag{A.59}$$

$$[P_{\mathrm{CMV}\#5}] \xrightarrow{l_{\mathrm{PI}}\cdot v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#5}] + [T_{AV}] \tag{A.60}$$

$$[P_{\mathrm{CMV}\#6}] \xrightarrow{l_{\mathrm{ER}}\cdot v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#6}] + [T_{AK}] \tag{A.61}$$

$$[P_{\mathrm{CMV}\#7}] \xrightarrow{l_{\mathrm{ER}}\cdot v_{\mathrm{CMV}}} [P_{\mathrm{CMV}\#7}] + [T_{BV}] \tag{A.62}$$

Degradacija kemijskih zvrsti

$$[T_{AK}] \xrightarrow{d_{\mathrm{T}}} \emptyset \tag{A.63}$$

$$[T_{AV}] \xrightarrow{d_{\mathrm{T}}} \emptyset \tag{A.64}$$

$$[T_{BK}] \xrightarrow{d_{\mathrm{T}}} \emptyset \tag{A.65}$$

$$[T_{BV}] \xrightarrow{d_{\mathrm{T}}} \emptyset \tag{A.66}$$

$$[BFP] \xrightarrow{d_{\mathrm{R}}} \emptyset \tag{A.67}$$

$$[mCT] \xrightarrow{d_{\mathrm{R}}} \emptyset \tag{A.68}$$

A.3 Specifikacija CUDA naprave

```
CUDA Device Query (Runtime API) version (CUDART static linking)
Detected 1 CUDA Capable device(s)
Device 0: "GeForce GTX 460"
  CUDA Driver Version / Runtime Version          5.0 / 5.0
  CUDA Capability Major/Minor version number:    2.1
  Total amount of global memory:                 768 MBytes (805306368 bytes)
  ( 7) Multiprocessors x ( 48) CUDA Cores/MP:    336 CUDA Cores
  GPU Clock rate:                                1350 MHz (1.35 GHz)
  Memory Clock rate:                             1800 Mhz
  Memory Bus Width:                              192-bit
  L2 Cache Size:                                 393216 bytes
  Max Texture Dimension Size (x,y,z)             1D=(65536),
                       2D=(65536,65535), 3D=(2048,2048,2048)
  Max Layered Texture Size (dim) x layers        1D=(16384) x 2048,
                       2D=(16384,16384) x 2048
  Total amount of constant memory:               65536 bytes
  Total amount of shared memory per block:       49152 bytes
  Total number of registers available per block: 32768
  Warp size:                                     32
  Maximum number of threads per multiprocessor:  1536
  Maximum number of threads per block:           1024
  Maximum sizes of each dimension of a block:    1024 x 1024 x 64
```

```
  Maximum sizes of each dimension of a grid:     65535 x 65535 x 65535
  Maximum memory pitch:                          2147483647 bytes
  Texture alignment:                             512 bytes
  Concurrent copy and kernel execution:          Yes with 1 copy engine(s)
  Run time limit on kernels:                     Yes
  Integrated GPU sharing Host Memory:            No
  Support host page-locked memory mapping:       Yes
  Alignment requirement for Surfaces:            Yes
  Device has ECC support:                        Disabled
  Device supports Unified Addressing (UVA):      No
  Device PCI Bus ID / PCI location ID:           1 / 0
  Compute Mode:
     < Default (multiple host threads can use
        ::cudaSetDevice() with device simultaneously) >

deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 5.0,
CUDA Runtime Version = 5.0, NumDevs = 1, Device0 = GeForce GTX 460
```

Koda A.1: Lastnosti CUDA naprave (izpis proloženega programa deviceQuery.).

A.4 Izvorna koda dela

Izvorna koda celotnega pričujočega dela je napisana v jezikih C (z uporabo knjižnice CUDA), Python, Octave/Matlab in Bash. Delo je prosto dostopno na naslovu http://code.google.com/p/igem-craggz/.

Literatura

[1] Team Slovenia, Switch-IT Inducible Therapeutics, projekt na tekmovanju iGEM 2012. (Nov. 2012).
url: http://2012.igem.org/Team:Slovenia

[2] C. M. Ajo-Franklin, D. a. Drubin, J. a. Eskin, E. P. S. Gee, D. Landgraf, I. Phillips, P. a. Silver, Rational design of memory in eukaryotic cells., Genes & development 21 (18) (2007) 2271–6.

[3] U. Alon, An introduction to systems biology: design principles of biological circuits, Chapman & Hall/CRC, 2007.

[4] U. Alon, Biological networks: the tinkerer as an engineer., Science (New York, N.Y.) 301 (5641) (2003) 1866–7.

[5] D. Angeli, J. E. Ferrell, E. D. Sontag, Detection of multistability, bifurcations, and hysteresis in a large class of biological positive-feedback systems., Proceedings of the National Academy of Sciences of the United States of America 101 (7) (2004) 1822–7.

[6] Y. Cao, D. T. Gillespie, L. R. Petzold, The slow-scale stochastic simulation algorithm., The Journal of chemical physics 122 (1) (2005) 14116.

[7] L. D. Chambers, The Practical Handbook of Genetic Algorithms: Applications, Chapman and Hall/CRC, Perth, 2000.

[8] J. L. Cherry, F. R. Adler, How to make a biological switch., Journal of theoretical biology 203 (2) (2000) 117–33.

[9] J. Clune, J.-b. Mouret, H. Lipson, The evolutionary origins of modularity, Arxiv preprint arXiv12072743 (2012) 1–17.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, Introduction to Algorithms, Vol. 7 of The Mit Electrical Engineering and computer Science Series, MIT Press, 2001.

[11] F. Crick, Central Dogma of Molecular, Nature 227 (6) (1970) 6–8.

[12] J. E. Dueber, B. J. Yeh, R. P. Bhattacharyya, W. a. Lim, Rewiring cell signaling: the logic and plasticity of eukaryotic protein circuitry., Current opinion in structural biology 14 (6) (2004) 690–9.

[13] W. Fiers, R. Contreras, G. Haegemann, R. Rogiers, A. Van De Voorde, H. Van Heuverswyn, J. Van Herreweghe, G. Volckaert, M. Ysebaert, Complete nucleotide sequence of SV40 DNA., Nature 273 (5658) (1978) 113–120.

[14] R. D. Fleischmann, M. D. Adams, O. White, R. A. Clayton, E. F. Kirkness, A. R. Kerlavage, C. J. Bult, J. F. Tomb, B. A. Dougherty, J. M. Merrick, Whole-genome random sequencing and assembly of Haemophilus influenzae Rd., Science 269 (5223) (1995) 496–512.

[15] Y. Fu, L. R. Jarboe, J. a. Dickerson, Reconstructing genome-wide regulatory network of E. coli using transcriptome data and predicted transcription factor activities., BMC bioinformatics 12 (1) (2011) 233.

[16] T. S. Gardner, C. R. Cantor, J. J. Collins, Construction of a genetic toggle switch in Escherichia coli., Nature 403 (6767) (2000) 339–42.

[17] J. C. M. Gebhardt, D. M. Suter, R. Roy, Z. W. Zhao, A. R. Chapman, S. Basu, T. Maniatis, X. S. Xie, Single-molecule imaging of transcription factor binding to DNA in live mammalian cells, Nature Methods (SeptembeR 2012).

[18] D. T. Gillespie, Numerical Simulation for Biochemical Kinetics Physical Chemists ' View of Biology , and Systems Biology.

[19] J. H. Holland, Adaptation in Natural and Artificial Systems, Vol. Ann Arbor, University of Michigan Press, 1975.

[20] P. Iglesias, B. Ingalls, Control theory and systems biology, 2010.

[21] J. M. Ignowski, D. V. Schaffer, Kinetic analysis and modeling of firefly luciferase as a quantitative reporter gene in live mammalian cells., Biotechnology and bioengineering 86 (7) (2004) 827–34.

[22] D. D. Jenkins, G. D. Peterson, GPU Accelerated Stochastic Simulation (3) 3–5.

[23] J. R. Karr, J. C. Sanghvi, D. N. Macklin, M. V. Gutschow, J. M. Jacobs, B. Bolival, N. Assad-Garcia, J. I. Glass, M. W. Covert, A whole-cell computational model predicts phenotype from genotype., Cell 150 (2) (2012) 389–401.

[24] J. Kim, E. Winfree, Synthetic in vitro transcriptional oscillators., Molecular systems biology 7 (465) (2011) 465.

[25] R. Kitney, P. Freemont, Synthetic Biology – The State of Play, FEBS Letters 586 (15) (2012) 2029–36.

[26] E. Knecht, C. Aguado, J. Cárcel, I. Esteban, J. M. Esteve, G. Ghislat, J. F. Moruno, J. M. Vidal, R. Sáez, Intracellular protein degradation in mammalian cells: recent developments., Cellular and molecular life sciences : CMLS 66 (15) (2009) 2427–43.

[27] T. Knight, Idempotent Vector Design for Standard Assembly of Biobricks Standard Biobrick Sequence Interface 1–11.

[28] B. P. Kramer, C. Fischer, M. Fussenegger, BioLogic gates enable logical transcription control in mammalian cells., Biotechnology and bioengineering 87 (4) (2004) 478–84.

[29] B. P. Kramer, A. U. Viretta, M. Daoud-El-Baba, D. Aubel, W. Weber, M. Fussenegger, An engineered epigenetic transgene switch in mammalian cells., Nature biotechnology 22 (7) (2004) 867–70.

[30] N. Law, K. Szeto, Adaptive genetic algorithm with mutation and crossover matrices, in: Proceedings of the 20th International Joint Conference on Artificial Intelligence, no. i, Hyderabad, India, 2007, pp. 2330–2333.

[31] H. Li, L. R. Petzold, Stochastic Simulation of Biochemical Systems on the Graphics Processing Unit, Bioinformatics 0 (x) (2005) 1–5.

[32] H. Li, L. Petzold, Efficient Parallelization of Stochastic Simulation Algorithm for Chemically Reacting Systems on the Graphics Processing Unit (2008) 1–25.

[33] G. Lillacci, M. Khammash, A distribution matching method for parameter estimation and model selection in computational biology, International Journal of Robust and Nonlinear Control.

[34] G. Lillacci, M. Khammash, Parameter estimation and model selection in computational biology., PLoS computational biology 6 (3) (2010) 696.

[35] J. J. Lohmueller, T. Z. Armel, P. a. Silver, A tunable zinc finger-based framework for Boolean logic computation in mammalian cells., Nucleic acids research (2012) 1–8.

[36] J. Macía, S. Widder, R. Solé, Why are cellular switches Boolean? General conditions for multistable genetic circuits., Journal of theoretical biology 261 (1) (2009) 126–35.

[37] R. W. Morris, C. A. Bean, G. K. Farber, D. Gallahan, E. Jakobsson, Y. Liu, P. M. Lyster, G. C. Y. Peng, F. S. Roberts, M. Twery, J. Whitmarsh, K. Skinner, Digital biology: an emerging and promising discipline., Trends in biotechnology 23 (3) (2005) 113–117.

[38] K. Mueller, General Purpose Computing on Graphics Hardware. Spletni tečaj. url: http://ci.nii.ac.jp/naid/110002673332

[39] L. Pasotti, N. Politi, S. Zucca, M. G. Cusella De Angelis, P. Magni, Bottom-up engineering of biological systems through standard bricks: a modularity study on basic parts and devices., PloS one 7 (7) (2012) e39407.

[40] L. Petzold, Efficient Parallelization of the Stochastic Simulation Algorithm for Chemically Reacting Systems On the Graphics Processing Unit, International Journal of High Performance Computing Applications 24 (2) (2009) 107–116.

[41] R. P. Shetty, D. Endy, T. F. Knight, Engineering BioBrick vectors from BioBrick parts., Journal of biological engineering 2 (1) (2008) 5.

[42] O. M. Subach, P. J. Cranfill, M. W. Davidson, V. V. Verkhusha, An enhanced monomeric blue fluorescent protein with the high chemical stability of the chromophore., PLoS ONE 6 (12) (2011) e28674.

[43] I. G. Szendro, J. Franke, J. A. G. M. de Visser, J. Krug, Predictability of evolution depends nonmonotonically on population size., Proceedings of the National Academy of Sciences of the United States of America 110 (2) (2013) 571–6.

[44] Y. K. Tateaki Sasaki, Practically Fast Multiple-Precision Evaluation of LOG(X), Biotechnology and bioengineering 5 (4) (1982) 247–250.

[45] M. Tigges, T. T. Marquez-Lago, J. Stelling, M. Fussenegger, A tunable synthetic mammalian oscillator., Nature 457 (7227) (2009) 309–12.

[46] M. Tigges, N. Dénervaud, D. Greber, J. Stelling, M. Fussenegger, A synthetic low-frequency mammalian oscillator., Nucleic acids research 38 (8) (2010) 2702–11.

[47] J. D. Watson, F. H. Crick, Molecular structure of nucleic acids; a structure for deoxyribose nucleic acid., Nature 171 (4356) (1953) 737–738.

[48] W. Weber, M. Fussenegger, Novel gene switches., Handbook of experimental pharmacology (178) (2007) 73–105.

[49] H. El Samad, M. Khammash, L. Petzold, D. Gillespie, Stochastic modelling of gene regulatory networks, International Journal of Robust and Nonlinear Control 15 (15) (2005) 691–711.

[50] Venter, J C et al., The sequence of the human genome., Science 291 (5507) (2001) 1304–1351.