ELEKTROTEHNIŠKI VESTNIK 84(5): 268–276, 2017
ORIGINAL SCIENTIFIC PAPER

Formalna specifikacija in verifikacija lastnosti uravnavanja laktoznega operona z orodjem EST

Rok Vogrin, Robert Meolic, Tatjana Kapus

Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, Koroška c. 46, 2000 Maribor, Slovenija

E-pošta: rok.vogrin@student.um.si, robert.meolic@um.si, tatjana.kapus@um.si

Povzetek. Zaradi velike kompleksnosti bioloških sistemov potekajo intenzivne raziskave uporabnosti formalnih metod za modeliranje in analizo reakcij v njih. Eden najpreprostejših organizmov, ki jih lahko proučujemo, je bakterija *Escherichia coli*. V zvezi z njo je bilo prvič uporabljeno poimenovanje operon za osnovno enoto prepisovanja genov. Eden najbolj raziskanih operonov je laktozni operon pri tej bakteriji. V članku je predstavljena uporaba orodja EST za formalno specifikacijo in verifikacijo kvalitativnih lastnosti sistema uravnavanja laktoznega operona. Sistem smo specificirali s procesno algebro. Glavni prispevek je specifikacija lastnosti s temporalno logiko ACTLW in njihova verifikacija z metodo preverjanja modelov.

Ključne besede: sistemska biologija, uravnavanje izražanja genov, formalne metode, procesna algebra, temporalna logika, preverjanje modelov

Formal specification and verification of properties of lactose operon regulation by using the EST toolbox

The growing interest in understanding biological systems has led to intensive research of the applicability of formal methods for modelling and analysis of reactions in these systems. One of the simplest organisms that can be investigated is bacterium *Escherichia coli*, in connection to which the term operon has first been coined to describe the basic unit of the gene transcription. One of the best-understood operons is the lactose operon of that bacterium. In this paper, a formal specification and verification of qualitative properties of the lactose-operon regulation system by using the EST toolbox are presented. The system is specified with a process algebra. The main contribution of this paper are the specification of the properties with temporal logic ACTLW and their verification by using model checking.

Keywords: systems biology, gene expression regulation, formal methods, process algebra, temporal logic, model checking

1 UVOD

Sistemska biologija obravnava biološke sisteme kot skupke komponent, ki med seboj komunicirajo, in raziskuje interakcije med komponentami ter njihov vpliv na lastnosti sistemov. V bioloških sistemih tipično opazimo verige sočasnih procesov in podprocesov. Zaradi kompleksnosti zahtevajo posebne metode za modeliranje in analizo [1]. Razvite so bile metode matematičnega modeliranja v obliki sistemov diferencialnih enačb, v katerih so sestavni deli procesov opisani s spremenljivkami [2]. V [3] navajajo, da je z modeli z enačbami težko ugotavljati lastnosti bioloških sistemov in primerjati različne sisteme med seboj.

Formalne metode za preverjanje pravilnosti obnašanja sistemov s sočasnostjo predpisujejo strogo matematično definiran jezik za specifikacijo sistema, prav tako strogo definiran jezik za specifikacijo pričakovanih lastnosti sistema in način preverjanja, ali specificirani sistem ima pričakovane lastnosti. Veliko formalnih metod temelji na modalnih in temporalnih logikah ter različnih procesnih algebrah [4]. V preteklosti so se uporabljale večinoma za verifikacijo računalniških programskih in strojnih sistemov, zdaj pa se uporabljajo tudi na netehničnih področjih, kot je modeliranje in analiza bioloških sistemov. Raziskovalci pričakujejo, da bo delo na tem področju pripeljalo do razvoja tehnik, ki bodo robustnejše, učinkovitejše in zanesljivejše [1]. V nasprotju z matematičnimi modeli z diferencialnimi enačbami, ki so v osnovi namenjeni analizi kvantitativnih lastnosti bioloških sistemov, so formalne metode posebej primerne za analizo kvalitativnih lastnosti le-teh [2]. Ena najbolj uveljavljenih formalnih metod za analizo kvalitativnih lastnosti sistemov s sočasnostjo je preverjanje modelov. Uporabljena je tudi v pričujočem članku. Ta metoda avtomatično preveri, ali sistem s končnim številom stanj ima lastnosti, podane v obliki formul temporalne logike.

EST (*Efficient Symbolic Tools*, `http://est.meolic.com/`) [5] je orodje za formalno specifikacijo in verifikacijo sistemov. Omogoča formalno verifikacijo sistemov z metodo preverjanja ekvivalence in z metodo preverjanja modelov. Algoritmi za verifikacijo so simbolični in delujejo s pomočjo binarnih odločitvenih grafov [4]. Zapis specifikacije sistema je mogoč z jezikom, podobnim procesni algebri *Calculus of Communicating*

Prejet 16. oktober, 2017
Odobren 21. november, 2017

Systems (CCS) [6], za specifikacijo lastnosti pa sta na voljo temporalni logiki ACTL (*Action-based Computation Tree Logic* – akcijska logika dreves izvajanj) in ACTLW (*ACTL with Unless operator* – ACTL z operatorjem *unless*) [7].

Bakterija *E. coli* (*Escherichia coli*) je eden najbolj raziskanih organizmov v biologiji, zato je postala model za preskušanje novih tehnologij [8]. V zvezi z njo je bilo prvič uporabljeno poimenovanje operon za osnovno enoto prepisovanja genov [3]. Za proučevanje novih metod modeliranja bioloških sistemov se pogosto uporablja uravnavanje laktoznega operona pri tej bakteriji. Za preskus metode preverjanja modelov je že bilo uporabljeno v [3], kjer je sistem specificiran v jeziku CCS, lastnosti pa z različico μ-računa, obogateno z operatorji CTL (*Computation Tree Logic* – logika dreves izvajanj), vendar je dvomljiva predvsem popolnost specifikacije nekaterih lastnosti. Cilj našega dela je bil za specifikacijo in verifikacijo uporabiti orodje EST in za specifikacijo lastnosti ACTLW. Skušali smo napisati popolne specifikacije lastnosti, predlaganih v [3], dodali pa smo tudi nove.

S problemom uravnavanja laktoznega operona pri bakteriji *E. coli* so se na podoben način ukvarjali tudi avtorji članka [9], ki so definirali nekoliko manjši model in zgolj nekaj lastnosti. Za specifikacijo lastnosti so uporabljali formule temporalne logike CTL z nekaterimi omejitvami v sintaksi. V [10] je predlagan spet drugačen kvalitativen model in izraženih je nekaj lastnosti z logiko CTL. V [11] je specificiran kvantitativen model uravnavanja z jezikom *Stochastic CLS* in opisana analiza dveh kvantitativnih lastnosti, izraženih z linearno temporalno logiko (LTL – *Linear Temporal Logic*), po prevodu modela v *Real-Time Maude*.

Članek je v nadaljevanju razdeljen takole. V razdelku 2 je okviren besedni opis delovanja uravnavanja laktoznega operona, privzetega v [3] in v tem članku. V razdelku 3 je predstavljena formalna specifikacija sistema uravnavanja. V razdelku 4 je najprej predstavljena logika ACTLW, nato pa specifikacija lastnosti in rezultati verifikacije. Razdelek 5 vsebuje komentar rezultatov in predloge za nadaljnje delo.

2 URAVNAVANJE LAKTOZNEGA OPERONA

Bakterija *E. coli* dobro raste na gojišču z glukozo [3]. Vzemimo, da je na gojišču samo laktoza. V tem primeru bakterija tvori glukozo iz laktoze.

Laktozni operon bakterije *E. coli* sestavljajo geni Y, Z in A, laktozni promotor ter primarni operator in dva sekundarna [3]. Gen Y nosi zapis za encim galaktozid permeaza, gen Z pa za encim β-galaktozidaza. Prvi encim omogoča, da laktoza iz okolja prehaja v celico, drugi pa skrbi, da se iz laktoze tvorijo izomer laktoze alolaktoza ter glukoza in galaktoza. Promotor je del, ki lahko spodbudi prepisovanje genov, operatorji pa so segmenti, na katere se lahko veže laktozni represor. Pri uravnavanju laktoznega operona sodeluje še ciklični adenozin monofosfat (cAMP), ki deluje kot soaktivator cAMP-receptorskega proteina (CRP), ki je aktivatorski protein.

V odsotnosti alolaktoze v celici se represor veže na primarni operator in na enega od sekundarnih ter preprečuje prepisovanje genov oziroma geni se prepisujejo in tako encimi tvorijo samo na osnovni ravni. Promotor je reprimiran. Promotor lahko spodbuja prepisovanje genov, samo če ni reprimiran in če je hkrati aktiviran. Galaktozid permeaza poskrbi, da laktoza vstopa v celico, β-galaktozidaza pa, da se iz nje tvori alolaktoza, ki se poveže na represor in s tem povzroči, da se le-ta odcepi od primarnega in sekundarnega operatorja. Promotor v tem primeru ni reprimiran, vendar se prepisovanje genov ne izvaja na najvišji ravni, če ni aktiviran. Njegovo aktiviranje je odvisno od količine glukoze v celici.

Kadar je vsebnost glukoze v celici majhna oziroma ničelna, je količina cAMP velika. To privede do vezave CRP s cAMP in kompleksa CRP–cAMP prek CRP na mesto blizu promotorja ("CRP-mesto"). Promotor se aktivira. Če ni reprimiran, spodbudi prepisovanje genov. S tem se poveča raven β-galaktozidaze ter tako glukoze v celici. β-galaktozidaza pa se pri razgradnji laktoze porablja in s tem se njena raven manjša.

Kadar količina glukoze v celici naraste, količina cAMP pade. CRP se odcepi od cAMP in z mesta pri promotorju. Posledica je dezaktiviranje promotorja. Raven β-galaktozidaze pade in tvorjenje glukoze iz laktoze je ustavljeno, vsaj dokler raven glukoze ne pade.

3 SPECIFIKACIJA SISTEMA V STILU JEZIKA CCS

3.1 Definicija jezika, podobnega CCS

CCS [6] je procesna algebra, ki omogoča opisovanje in analiziranje obnašanja sistema, sestavljenega iz več komponent. Te predstavimo s procesi, ki med seboj komunicirajo. Njihovo obnašanje je podano z akcijami, ki jih lahko izvedejo. Ločimo zunanje akcije, ki so lahko vhodne (a) ali izhodne ($\overline{a}$), in notranjo akcijo τ. Pravimo, da sta akciji a in $\overline{a}$ med seboj komplementarni. Pri pisanju specifikacije sistema smo v osnovi uporabili formalizem z naslednjo sintakso, ki je podobna sintaksi jezika CCS [12]:

$$P ::= 0 \mid a.P_1 \mid A \mid P_1 + P_2 \mid P_1|P_2 \mid P_1\backslash a$$

Zapis $a.P_1$ pomeni proces, v katerem se lahko najprej izvede akcija a, nato pa proces P_1. 0 je prazen proces, to je tak, ki ne naredi ničesar. Namesto $a.0$ bomo pisali kar a. Z zapisom $A \triangleq P$ definiramo, da je A ime procesa P. V procesu P lahko nastopa A, s čimer dosežemo, da se P ne konča, ampak se izvede ponovno. Torej lahko imamo rekurzijo. Kot primer lahko zapišemo $P \triangleq a.\overline{b}.P$, kar pomeni, da proces P najprej izvede

$$
\begin{aligned}
Ent &= !ELAC.React && (1)\\
React &= !ILAC.React2 + !rbeta.React && (2)\\
React2 &= !ELAC.React + !rbeta.React2 && (3)\\
Beta\_galactosidase\_low &= ?rbeta.!RBETA.!iallo.!IALLO.Beta\_galactosidase\_low\\
&\quad + ?ibeta.!IBETA.Beta\_galactosidase\_high && (4)\\
Beta\_galactosidase\_high &= ?rbeta.!RBETA.!iglu.!IGLU.Beta\_galactosidase\_high\\
&\quad + ?dbeta.!DBETA.Beta\_galactosidase\_low && (5)\\
Allolactose\_none &= ?iallo.Allolactose\_low && (6)\\
Allolactose\_low &= ?iallo.Allolactose\_low + ?ballo.Allolactose\_none && (7)\\
Neg &= !BO1.(TAU.!BO2.!rep.!REP.!ballo.!BALLO.!UBO1.!UBO2.!urep.!UREP.Neg\\
&\quad + TAU.!BO3.!rep.!REP.!ballo.!BALLO.!UBO1.!UBO3.!urep.!UREP.Neg) && (8)\\
Promoter\_iu &= ?rep.Promoter\_ir + ?act.!ibeta.Promoter\_au && (9)\\
Promoter\_ir &= ?urep.Promoter\_iu + ?act.Promoter\_ar && (10)\\
Promoter\_au &= ?rep.!dbeta.Promoter\_ar + ?iact.!dbeta.Promoter\_iu && (11)\\
Promoter\_ar &= ?urep.!ibeta.Promoter\_au + ?iact.Promoter\_ir && (12)\\
Pos &= !lev.(?low.!L\_to\_H.!BC.!BS.!act.!ACT.Act + ?high.Pos) && (13)\\
Act &= !lev.(?low.Act + ?high.!H\_to\_L.!UBC.!UBS.!iact.!IACT.Pos) && (14)\\
Glucose\_high &= ?iglu.Glucose\_high + ?lev.!HIGH.!high.Glucose\_high\\
&\quad + TAU.!DGLU.Glucose\_low && (15)\\
Glucose\_low &= ?iglu.(TAU.Glucose\_low + TAU.!GLU\_L\_to\_H.Glucose\_high)\\
&\quad + ?lev.!LOW.!low.Glucose\_low && (16)\\
\\
raw\ net\ SpecFull &= |(Ent, Beta\_galactosidase\_low, Allolactose\_none, Neg,\\
&\quad Promoter\_iu, Pos, Glucose\_high)\\
&\quad \backslash rbeta\backslash iallo\backslash ibeta\backslash iglu\backslash dbeta\backslash ballo\backslash rep\backslash urep\backslash act\backslash iact\backslash lev\backslash low\backslash high && (17)
\end{aligned}
$$

Slika 1: Abstraktna specifikacija sistema ($SpecFull$)

vhodno akcijo a, za njo izhodno akcijo $\overline{b}$, nato pa se ponovno obnaša kot P.

V procesu se lahko izbere, katera akcija se bo izvedla. To omogoča operator "+". Primer je proces $P \triangleq a + \overline{b}$, kjer se lahko izvede vhodna ali izhodna akcija (nato pa se v obeh primerih proces konča).

Izvedba komponent sistema je lahko vzporedna. Vzporednost izrazimo z znakom "|" za paralelno kompozicijo, na primer $(a.P)|(\overline{a}.Q)$. Komunikacija med procesoma v paralelni kompoziciji je mogoča prek akcij, iz katerih sta sestavljena. Procesa lahko izvedeta komunikacijo, če sta v njiju hkrati omogočeni med seboj komplementarni akciji, kakor v našem primeru. Procesa lahko hkrati izvedeta akcijo a in $\overline{a}$, vendar se ta komunikacija označi z notranjo akcijo τ, in prvi proces nadaljuje kot P, drugi pa kot Q. Komunikacija pa ni edina možnost. Levi proces lahko sam izvede akcijo a in nadaljuje kot P, pa tudi desni proces lahko sam izvede akcijo $\overline{a}$ in nadaljuje kot Q. Če hočemo doseči, da se v našem primeru lahko izvede samo komunikacija, to storimo z operatorjem "\" za prepoved akcij.

Z zapisom $P_1\backslash a$ prepovemo, da bi se kdaj izvedla akcija a ali njej komplementarna akcija v procesu P_1. S tem tudi preprečimo, da bi proces P_1 komuniciral s katerimkoli procesom prek akcije a ali $\overline{a}$. Torej zapis $((a.P)|(\overline{a}.Q))\backslash a$ pomeni proces, ki najprej izvede notranjo akcijo τ, nato pa nadaljuje kot proces $(P|Q)\backslash a$. Rečemo tudi, da je akcija a (in njen komplement) v procesu $((a.P)|(\overline{a}.Q))\backslash a$ skrita, saj je edina možnost, da nastopa v notranji komunikaciji.

3.2 Specifikacija sistema uravnavanja laktoznega operona

V [3] sta podani specifikacija sistema $SystemFull$ in abstraktna specifikacija $SpecFull$. Specifikacija sistema uravnavanja, ki smo jo napisali v EST-u in jo uporabili za preverjanje večine lastnosti, je skoraj enaka specifikaciji $SystemFull$ iz [3]. Zaradi njene dolžine je na sliki 1 namesto nje prikazana abstraktna specifikacija $SpecFull$, napisana za EST. Je paralelna kompozicija procesov Ent, $Beta\_galactosidase\_low$, $Allolactose\_none$, Neg, $Promoter\_iu$, Pos in $Glucose\_high$. $SystemFull$ se od nje razlikuje v tem, da namesto vsakega od procesov Ent, Neg in Pos vsebuje dva procesa ali več. Procesi $Beta\_galactosidase\_low$, $Allolactose\_none$, $Promoter\_iu$ in $Glucose\_high$ predstavljajo β-galaktozidazo, alolaktozo, promotor oziroma glukozo iz neformalnega opisa. Proces Ent pov-

zema skupno obnašanje procesov, ki predstavljajo laktozo na gojišču, laktozo v celici in galaktozid permeazo v specifikaciji $SpecFull$. Neg ("negativno uravnavanje") opisuje skupni učinek obnašanja procesov, ki v zadnji predstavljajo laktozni represor in vsakega od treh operatorjev (te v nadaljevanju označimo z O1, O2 in O3). Pos ("pozitivno uravnavanje") pa specificira skupni učinek obnašanja procesov, ki v njej predstavljata CRP in cAMP. Imena procesov v paralelni kompoziciji so hkrati začetna stanja teh procesov in tako vidimo, kakšno začetno stanje je privzeto za katero snov v sistemu. Na sliki 1 vidimo, da opis procesa $Glucose\_high$ sestoji iz štirih vrstic in da ima poleg začetnega še stanje $Glucose\_low$. Privzeto je torej, da je v začetku izvajanja raven glukoze v celici visoka. Vzeto je še, da je količina β-galaktozidaze majhna, da v celici ni alolaktoze, da je promotor neaktiviran in nereprimiran, da represor ni povezan na operatorje in da so torej prosti, iz $SystemFull$ [3] pa je razvidno, da je privzeto, da je laktoza ves čas na gojišču in da obstaja nespremenljiva količina galaktozid permeaze, da na začetku ni laktoze v celici ter da je malo cAMP, da ni povezan s CRP in da CRP ni povezan na promotor.

Orodje EST za branje uporablja razpoznavalnik za specifikacije CCS z nekoliko spremenjeno sintakso. Vhodno akcijo označimo z "?", izhodno pa s "!". Primer izhodne akcije na sliki 1 je $!iglu$, vhodne pa $?iglu$. Za notranjo akcijo τ uporabimo TAU. Za tvorjenje paralelne kompozicije uporabimo pri njenem imenu rezervirano besedo net. Z besedo raw pa povemo EST-u, da za kompozicijo ni treba izračunati pomožnih struktur za nadaljnje sestavljanje.

V specifikaciji nastopajo akcije za komunikacijo znotraj sistema, ki jih pišemo z malimi črkami (na primer $iallo$), akcija τ in zunanje akcije sistema (na primer $\overline{IALLO}$). Na sliki 1 vidimo, da so vse akcije za komunikacijo znotraj sistema prepovedane in se torej skrijejo za akcijami τ. Zunanje akcije so, kot običajno v specifikacijah CCS, uporabljene kot sonde, ki povedo, kaj se je kdaj zgodilo v notranjosti sistema, saj sicer ne bi mogli specificirati pričakovanih lastnosti sistema. Tako kot v [3] so izhodne. Akcija $\overline{IALLO}$ se na primer v sistemu lahko zgodi takoj za $\overline{iallo}$ v procesu $Beta\_galactosidase\_low$ in tako takoj za komunikacijo le-tega s procesom $Allolactose\_none$ prek akcije $iallo$ ter pove, da je nastala alolaktoza. Akcija τ je v procesih uporabljena za modeliranje obnašanja, ki ni odvisno od drugih procesov ali je nedeterministično. V procesu $Glucose\_high$ na primer vidimo, da ko je raven glukoze visoka, se lahko samodejno zniža. To predstavlja porabo glukoze v celici. Ko pa je raven glukoze nizka, se lahko zgodi akcija $iglu$, ki na splošno pomeni zvišanje ravni, povzročeno z β-galaktozidazo (akcija $\overline{iglu}$), vendar je s pomočjo notranje akcije specificirano, da ostane raven nespremenjena ali pa se poviša. Tako je narejeno zato, ker je količina glukoze modelirana kvalitativno samo z dvema ravnema in naj bi bilo specificirano, da raven še ni tako narasla, da bi bila že označena kot visoka ("high") in ne kot nizka ("low").

V specifikaciji sistema tako kot v [3] nastopajo naslednje zunanje akcije:

$\overline{ELAC}$	vstop laktoze v celico
$\overline{ILAC}$	povečanje količine laktoze v celici
$\overline{RBETA}$	zmanjšanje količine laktoze v celici
$\overline{IALLO}$	povečanje količine alolaktoze
$\overline{IBETA}$	povečanje količine β-galaktozidaze
$\overline{DBETA}$	zmanjšanje količine β-galaktozidaze
$\overline{IGLU}$	tvorjenje glukoze
$\overline{BO1}$	vezava represorja na O1
$\overline{BO2}$	vezava represorja na O2
$\overline{BO3}$	vezava represorja na O3
$\overline{REP}$	represor vezan na operatorja
$\overline{UREP}$	represor odcepljen od operatorjev
$\overline{BALLO}$	vezava alolaktoze na represor
$\overline{UBO1}$	odcep represorja od O1
$\overline{UBO2}$	odcep represorja od O2
$\overline{UBO3}$	odcep represorja od O3
$\overline{BC}$	vezava cAMP na CRP
$\overline{BS}$	vezava CRP–cAMP na CRP-mesto
$\overline{ACT}$	aktiviranje promotorja
$\overline{UBC}$	odcep cAMP od CRP
$\overline{UBS}$	odcep CRP s CRP-mesta
$\overline{IACT}$	dezaktiviranje promotorja
$\overline{L\_to\_H}$	povečanje količine cAMP
$\overline{H\_to\_L}$	zmanjšanje količine cAMP
$\overline{DGLU}$	zmanjšanje količine glukoze
$\overline{GLU\_L\_to\_H}$	povečanje količine glukoze

Za potrebe preverjanja lastnosti smo dodali še akciji $\overline{HIGH}$ in $\overline{LOW}$, ki kažeta raven glukoze v celici.

Orodje EST ne omogoča izrazov oblike $a.(A_1 + A_2)$, kjer sta A_1 in A_2 imeni procesov. Takšna izraza sta v specifikaciji $SystemFull$ v [3] uporabljena za specificiranje, da po neki akciji a proces, ki predstavlja laktozo v celici, sam odloči, ali se bo njena raven spremenila ali ne. Namesto tega smo uporabili zapis $a.A_1 + a.A_2$, kar samo prestavi trenutek odločanja in ne vpliva na preverjane lastnosti.

4 PREVERJANJE MODELOV Z ACTLW

4.1 Definicija ACTLW

ACTLW [7] je izjavna temporalna logika razvejanega časa. Definirana je nad označenim sistemom prehajanja stanj (LTS – *Labelled Transition System*). LTS je četverica $\mathcal{M} = (S, Act_\tau, D, s_{init})$, v kateri je S množica vseh stanj, Act_τ množica, ki vsebuje vsaj notranjo akcijo, lahko pa še zunanje, $D \subseteq S \times Act_\tau \times S$ prehajalna relacija in $s_{init} \in S$ začetno stanje. LTS $\mathcal{M}$ je končen, če in samo če sta množici S in Act_τ

$a \models \alpha$, če in samo če $a = \alpha$;
$a \models \neg\chi$, če in samo če $a \not\models \chi$;
$a \models \chi \vee \chi'$, če in samo če $a \models \chi$ ali $a \models \chi'$;
$s \models_{\mathcal{M}} true$ vedno;
$s \models_{\mathcal{M}} \neg\varphi$, če in samo če $s \not\models_{\mathcal{M}} \varphi$;
$s \models_{\mathcal{M}} \varphi \vee \varphi'$, če in samo če $s \models_{\mathcal{M}} \varphi$ ali $s \models_{\mathcal{M}} \varphi'$;
$s \models_{\mathcal{M}} \mathbf{EE}\,\gamma$, če in samo če v $\mathcal{M}$ obstaja polna pot π, za katero velja $\pi(0) = s$ in $\pi \models_{\mathcal{M}} \gamma$;
$s \models_{\mathcal{M}} \mathbf{AA}\,\gamma$, če za vse polne poti π v $\mathcal{M}$ velja $\pi(0) = s \Longrightarrow \pi \models_{\mathcal{M}} \gamma$;
$\pi \models_{\mathcal{M}} [\{\chi\}\varphi\,\mathbf{U}\,\{\chi'\}\varphi']$, če in samo če $len(\pi) > 1$ in obstaja tak i, $1 \leq i < len(\pi)$,
da $\pi(i) \models_{\mathcal{M}} \varphi'$ in $\pi(i) \in D_{/\chi'/}(\pi(i-1))$
ter za vsak $1 \leq j \leq i-1$ velja $\pi(j) \models_{\mathcal{M}} \varphi$ in $\pi(j) \in D_{/\chi/}(\pi(j-1))$;
$\pi \models_{\mathcal{M}} [\{\chi\}\varphi\,\mathbf{W}\,\{\chi'\}\varphi']$, če in samo če $\pi \models_{\mathcal{M}} [\{\chi\}\varphi\,\mathbf{U}\,\{\chi'\}\varphi']$
ali pa za vsak i, $1 \leq i < len(\pi)$, velja $\pi(i) \models_{\mathcal{M}} \varphi$ in $\pi(i) \in D_{/\chi/}(\pi(i-1))$.

Slika 2: Definicija pomena operatorjev pri temporalni logiki ACTLW; za podane izraze velja $a \in Act_\tau$ in $/\chi/ = \{\alpha \mid \alpha \models \chi\}$.

končni. Trojka $(s, \alpha, s') \in D$ se imenuje prehod z akcijo α iz stanja s v stanje s', pri čemer se prehod z akcijo τ imenuje tudi notranji prehod. Pot π v LTS-ju je končno ali neskončno zaporedje stanj in akcij $s_0, a_1, s_1, a_2, s_2, ...$, ki se začne v stanju s_0 in v katerem za vsaki zaporedni stanji s_{i-1} in s_i velja $(s_{i-1}, a_i, s_i) \in D$. Eno stanje na poti π označimo s $\pi(i)$, pri čemer je $\pi(0)$ prvo stanje na poti, stanje $\pi(i+1)$ pa tisto, ki neposredno sledi stanju $\pi(i)$. Dolžina poti je število stanj na poti π in jo označimo z $len(\pi)$. Dolžina neskončne poti je ω.

Če obstaja $(s, \alpha, s') \in D$, potem je stanje s' naslednik stanja s. Z oznako $D_A(s)$ označimo množico vseh naslednikov stanja s, ki so dosegljiva s prehodom, ki vsebuje akcijo iz množice A. Stanje brez naslednikov se imenuje zagatno stanje. Neskončne poti in končne poti, ki se končajo v zagatnem stanju, imenujemo polne poti.

Skladnja formul ACTLW φ (imenovanih tudi formule stanja) nad $\mathcal{M}$-jem je definirana z naslednjo slovnico, pri čemer je χ formula akcije, $\alpha \in Act_\tau$, γ formula poti, **EE** in **AA** sta kvantifikatorja poti, **U** in **W** pa temporalna operatorja *until* in *unless*:

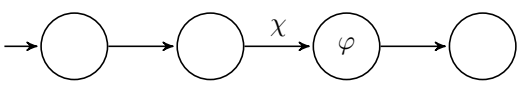
$$\begin{array}{lcl} \chi & ::= & \alpha \mid \neg\chi \mid \chi \vee \chi \\ \varphi & ::= & true \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{EE}\,\gamma \mid \mathbf{AA}\,\gamma \\ \gamma & ::= & [\{\chi\}\varphi\,\mathbf{U}\,\{\chi\}\varphi] \mid [\{\chi\}\varphi\,\mathbf{W}\,\{\chi\}\varphi] \end{array}$$

Pomen formul ACTLW je induktivno definiran s pravili na sliki 2. Z $a \models \chi$ označimo, da akcija a zadošča formuli akcije χ, s $s \models_{\mathcal{M}} \varphi$, da formula stanja φ velja v stanju s, s $\pi \models_{\mathcal{M}} \gamma$ pa, da formula poti γ velja na polni poti π. Formulo $\alpha \vee \neg\alpha$, v kateri je $\alpha \in Act_\tau$ poljubno izbrana akcija, krajše zapišemo kot $true$. Namesto $\neg true$ v formulah akcije in formulah stanja pišemo $false$. Operator $\wedge$ je definiran kot $\chi \wedge \chi' = \neg(\neg\chi \vee \neg\chi')$ oziroma $\varphi \wedge \varphi' = \neg(\neg\varphi \vee \neg\varphi')$. Če $\alpha \models \chi$, prehod (s, α, s') imenujemo χ-prehod, če $\alpha \models \chi$ in $s' \models_{\mathcal{M}} \varphi$, pa prehod (s, α, s') imenujemo (χ, φ)-prehod. Formula ACTLW φ velja v LTS-ju $\mathcal{M}$ (to označimo z $\mathcal{M} \models \varphi$), če in samo če $s_{init} \models_{\mathcal{M}} \varphi$.

Iz definicije skladnje formul ACTLW sledijo operatorji **EEU**, **EEW**, **AAU** in **AAW**. Dodatno definiramo izpeljane operatorje ACTLW **EEF**, **AAF**, **EEG** in **AAG**:

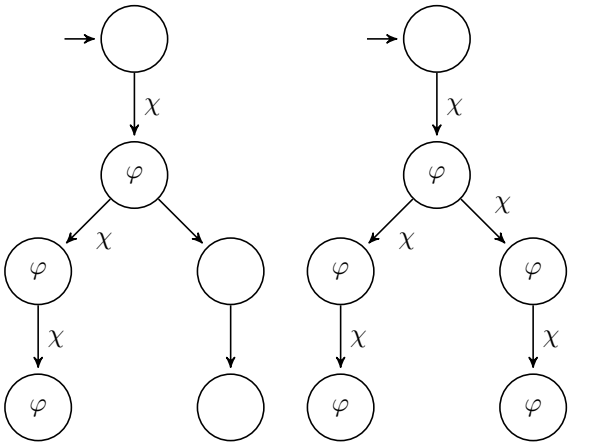
$$\begin{array}{l} \mathbf{EEF}\{\chi\}\varphi \triangleq \mathbf{EE}[\{true\}true\,\mathbf{U}\,\{\chi\}\varphi] \\ \mathbf{AAF}\{\chi\}\varphi \triangleq \mathbf{AA}[\{true\}true\,\mathbf{U}\,\{\chi\}\varphi] \\ \mathbf{EEG}\{\chi\}\varphi \triangleq \mathbf{EE}[\{\chi\}\varphi\,\mathbf{W}\,\{false\}false] \\ \mathbf{AAG}\{\chi\}\varphi \triangleq \mathbf{AA}[\{\chi\}\varphi\,\mathbf{W}\,\{false\}false] \end{array}$$

Na splošno je vsak operator ACTLW sestavljen tako, da kvantifikatorju poti **EE** ali **AA** pridružimo temporalni operator **F**, **G**, **U** ali **W**. Pomen teh neformalno razložimo takole. Za formulo poti $\mathbf{F}\{\chi\}\varphi$ velja, da je zadoščena na polni poti, če in samo če na njej obstaja (χ, φ)-prehod (slika 3). Formula $\mathbf{G}\{\chi\}\varphi$ je zadoščena na polni poti, če in samo če so vsi prehodi na njej (χ, φ)-prehodi. To ilustrira slika 4, ki hkrati prikazuje primera veljavnosti formul ACTLW v LTS-jih. V prvem velja formula $\mathbf{EEG}\{\chi\}\varphi$, v drugem pa tudi $\mathbf{AAG}\{\chi\}\varphi$. Formula $[\{\chi\}\varphi\,\mathbf{U}\,\{\chi'\}\varphi']$ je zadoščena na polni poti, če in samo če se ta začne s končnim zaporedjem (χ, φ)-prehodov, ki mu sledi (χ', φ')-prehod. Formula $[\{\chi\}\varphi\,\mathbf{W}\,\{\chi'\}\varphi']$ je zadoščena na polni poti, če in samo če je na tej poti zadoščena formula $[\{\chi\}\varphi\,\mathbf{U}\,\{\chi'\}\varphi']$ ali formula $\mathbf{G}\{\chi\}\varphi$.



Slika 3: Pomen formule poti $\mathbf{F}\{\chi\}\varphi$

Zapis formul ACTLW lahko dodatno okrajšamo s skladenjskimi okrajšavami, pri katerih namesto $\{\chi\}true$ zapišemo samo $\{\chi\}$, namesto $\{true\}\varphi$ pa samo φ. Tukaj je nekaj primerov:



Slika 4: Pomen formul ACTLW **EEG**$\{\chi\}\varphi$ in **AAG**$\{\chi\}\varphi$ v LTS-ju

$\mathbf{EEF}\{\chi\} \triangleq \mathbf{EEF}\{\chi\}true$
$\mathbf{EEF}\ \varphi \triangleq \mathbf{EEF}\{true\}\varphi$
$\mathbf{EEG}\{\chi\} \triangleq \mathbf{EEG}\{\chi\}true$
$\mathbf{EEG}\ \varphi \triangleq \mathbf{EEG}\{true\}\varphi$
$\mathbf{EE}[\{\chi\}\ \mathbf{U}\ \{\chi'\}] \triangleq \mathbf{EE}[\{\chi\}true\ \mathbf{U}\ \{\chi'\}true]$
$\mathbf{EE}[\varphi\ \mathbf{U}\ \varphi'] \triangleq \mathbf{EE}[\{true\}\varphi\ \mathbf{U}\ \{true\}\varphi']$

4.2 Formalna verifikacija uravnavanja laktoznega operona

Pričakovane lastnosti sistema, opisanega z našo specifikacijo *SystemFull*, smo specificirali s formulami ACTWL in z metodo preverjanja modelov v EST-u ugotovili, ali jim sistem zadošča. V EST-u se opis sistema s procesno algebro najprej pretvori v LTS. Preverjanje modelov za izbrano formulo ACTLW deluje tako, da se poišče množica vseh stanj, v katerih formula velja, nato pa preveri, ali je začetno stanje LTS-ja v tej množici. V [3] poročajo, da jim z orodjem CWB-NC in s 4 GB RAM-a ni uspelo tvoriti LTS-ja. Po več kot dveh urah je orodju zmanjkalo pomnilniškega prostora. V EST-u nam ga je uspelo tvoriti v trenutku, pa tudi potrebe po večji količini pomnilnika ni bilo. Naslednja tabela prikazuje število stanj in prehodov dobljenega LTS-ja brez akcij $\overline{HIGH}$ in $\overline{LOW}$ (S1) ter z njima (S2) (čas tvorjenja je bil izmerjen na računalniku s procesorjem Intel Core i5-4250U pri frekvenci 1,2 GHz):

sistem	stanja	prehodi	čas tvorjenja	porabljeni pomnilnik
S1	440700	2156768	1,15 s	55 MiB
S2	460164	2235304	1,24 s	55 MiB

V osnovi ločimo varnostne in živostne lastnosti sistemov. Varnostna lastnost pove, da se na nobeni poti v LTS-ju nikoli ne zgodi nič slabega, živostna pa, da se na vsaki poti nekoč zgodi nekaj dobrega. Najpomembnejše je preveriti varnostne lastnosti. Najprej smo preverili, da v sistemu ni zagatnih stanj, saj pričakujemo, da se nikoli ne ustavi. To smo preverili s formulo **EEX**$\{true\}$ *AND* **AAG** **EEX**$\{true\}$. Formula **EEX**$\{true\}$ je definirana kot **EE**$[\{false\}false\ \mathbf{U}\ \{true\}true]$ in v stanju velja, če in samo če ima to naslednika. Če EST ugotovi, da sistem ne zadošča formuli, lahko izpiše protiprimer, tj. zaporedje akcij, ki krši trditev, opisano s formulo. Sledi besedni opis pomembnih lastnosti uravnavanja, podanih na sliki 5 skupaj z rezultati preverjanja:

(A) S formulami pokažemo odnos med količino glukoze in cAMP.
(A21) Na vseh poteh se vedno, kadar je količina cAMP velika, zmanjša, samo če prej naraste količina glukoze.
V nasprotju s pričakovanji ta formula ni veljavna. Tvorjeni protiprimer je pokazal, da se lahko zgodi, da proces, ki predstavlja cAMP (na sliki 1 je skrit v procesu Pos [3]), izvede akcijo lev in nato low skupaj s procesom glukoze, ko je ta nizka, nato raven glukoze naraste, tj. izvede se $\overline{GLU\_L\_to\_H}$, šele zatem pa proces cAMP izvede akcijo $\overline{L\_to\_H}$ (tj. raven cAMP naraste, ker je proces prej dobil informacijo, da je glukoza nizka). Nato cAMP izvede akcijo lev, dobi informacijo, da je glukoza visoka, in izvede akcijo $\overline{H\_to\_L}$. Vzrok, da formula A21 ne velja, je v tem, da poizvedovanje o ravni glukoze in reakcija na informacijo ne tvorita nedeljive akcije. Tako sonda $\overline{GLU\_L\_to\_H}$ ne izraža nujno ravni glukoze, ki je bila upoštevana pri spremembi ravni cAMP. Zato smo v procesu, ki predstavlja glukozo, uvedli sondažno akcijo $\overline{HIGH}$ (slika 1), ki jo izraža, saj se zgodi po poizvedovanju procesa cAMP po trenutni ravni glukoze prek akcije lev in pred akcijo $\overline{high}$, ki cAMP-ju da odgovor. Tako lahko izrazimo naslednjo lastnost, ki je veljavna.
(A23) Na vseh poteh vedno, kadar je količina cAMP velika (tj. potem ko naraste), se zmanjša, samo če je količina glukoze velika.
To je varnostna formula in ne pove, ali bo raven cAMP padla. To izrazimo z živostjo.
(A40) Na vseh poteh vedno, kadar je količina cAMP velika, ko raven glukoze naraste, se zmanjša količina cAMP.
Ta formula ne velja, ker obstajajo tudi poti, na katerih bi se lahko izvedla akcija $\overline{H\_to\_L}$, vendar so omogočene tudi druge akcije, kot na primer vezava represorja na operatorja in nato odcep, kar se lahko nenehno izvaja. Zagotovo pa obstaja kakšna pot, na kateri se bo cAMP zmanjšal. To izrazimo s formulo, ki jo dobimo iz živostne z nadomestitvijo drugega operatorja

A	property $A21 ==$ *NOT* **EEF** {!L_to_H} **EE**[{*NOT* !GLU_L_to_H} **U** {!H_to_L}]	FALSE
	property $A23 ==$ *NOT* **EEF** {!L_to_H} **EE**[{*NOT* !HIGH} **U** {!H_to_L}]	TRUE
	property $A40 ==$ *NOT* **EEF** {!L_to_H} **EEF** {!HIGH} **EEG** {*NOT* !H_to_L}	FALSE
	property $A42 ==$ *NOT* **EEF** {!L_to_H} **EEF** {!HIGH} **AAG** {*NOT* !H_to_L}	TRUE
B	property $B23 ==$ *NOT* **EEF** {!H_to_L} **EE**[{*NOT* !LOW} **U** {!L_to_H}]	TRUE
	property $B2z ==$ **AA**[{*NOT* !L_to_H} **W** {!DGLU}]	TRUE
C	property $C1 ==$ **AAG EEF** {!BC}	TRUE
D	property $D1 ==$ *NOT* **EEF** {!H_to_L} **EE**[{*NOT* !L_to_H} **U** {!BC}]	TRUE
	property $D2 ==$ *NOT* **EEF** {!L_to_H} **EE**[{*NOT* !H_to_L} **U** {!UBC}]	TRUE
E	property $E14 ==$ *NOT* **EEF** {!L_to_H} **AAG** {*NOT* !BC}	TRUE
	property $E15 ==$ *NOT* **EEF** {!UBS} **EE**[{*NOT* !BC} **U** {!BS}]	TRUE
F	property $F11 ==$ **AAG EEF** {!BS}	TRUE
	property $F12 ==$ *NOT* **EEF** {!BC} **AAG** {*NOT* !BS}	TRUE
G	property $G2 ==$ *NOT* **EEF** {!IACT} **EE**[{*NOT* !BS} **U** {!ACT}]	TRUE
K	property $K1z ==$ **AA**[{*NOT* !IALLO} **W** {!RBETA}]	TRUE
	property $K13 ==$ *NOT* **EEF**{!BALLO} **AAG** {*NOT* !IALLO}	TRUE
L	property $L12 ==$ **AA**[{*NOT* !IALLO} **W** {!ELAC}]	TRUE
M	property $M12 ==$ *NOT* **EEF** {!IALLO} **AAG** {*NOT* !BALLO}	TRUE
N	property $N1 ==$ *NOT* **EEF** {!REP} **EE**[{*NOT* !BALLO} **U** {!UREP}]	TRUE
O	property $O11 ==$ **AAG EEF** {!BO1}	TRUE
	property $O12 ==$ **AAG EEF** {!UBO1}	TRUE
U	property $U11 ==$ *NOT* **EEF** {!BO2} **EE**[{*NOT* !UBO2} **U** {!BO3}]	TRUE
	property $U21 ==$ *NOT* **EEF** {!BO1} **EE**[{*NOT* !BO2 *AND NOT* !BO3} **U** {!UBO1}]	TRUE
V	property $V1z ==$ **AA**[{*NOT* !IBETA} **W** {!BS}]	TRUE
	property $V2z ==$ **AA**[{*NOT* !BS} **W** {!BC}]	TRUE
	property $V7 ==$ *NOT* **EEF** {!IBETA} **EE**[{*NOT* !IACT *AND NOT* !REP} **U** {!DBETA}]	FALSE
	property $V8 ==$ *NOT* **EEF** {!DBETA} **EE**[{*NOT* !ACT *AND NOT* !UREP} **U** {!IBETA}]	FALSE
W	property $W3 ==$ *NOT* **EEF** {!DBETA} **EE**[{*NOT* !IBETA} **U** {!IGLU}]	TRUE
Y	property $Y12 ==$ *NOT* **EEF** {!GLU_L_to_H} **AAG** {*NOT* !DBETA}	TRUE
Z	property $Z1 ==$ *NOT* **EEF** {!DGLU} **EE**[{*NOT* !IGLU} **U** {!GLU_L_to_H}]	FALSE

Slika 5: Formule ACTLW za lastnosti uravnavanja

EE z operatorjem **AA**. Takšnim lastnostim pravimo lastnosti možnosti [13].

(A42) Na vseh poteh vedno, kadar je količina cAMP velika, ko raven glukoze naraste, je mogoče, da se količina cAMP zmanjša.

(B) **(B23)** pravi nasprotno kot A23. Na vseh poteh vedno, potem ko količina cAMP pade, naraste, samo če je količina glukoze majhna.

Ker je na začetku izvajanja sistema raven cAMP že nizka, pa ta formula ne zajame začetnega dogajanja. To izraža naslednja formula.

(B2z) Na nobeni poti na začetku ne naraste količina cAMP, razen če prej ne pade raven glukoze.

(C) Na vseh poteh je vedno mogoče, da se cAMP veže na CRP.

(D) **(D1)** Na vseh poteh se cAMP vedno veže na CRP samo, če je raven cAMP visoka.

(D2) Na vseh poteh se cAMP vedno odcepi od CRP samo, če je raven cAMP nizka.

(E) **(E14)** Na vseh poteh je vedno, kadar se količina cAMP poveča, mogoče, da se cAMP veže na CRP.

(E15) Na nobeni poti se nikoli, kadar se CRP odcepi s CRP-mesta, ne veže nanj, dokler se cAMP ne veže na CRP.

(F) Na vseh poteh je vedno mogoče, da se CRP–cAMP veže na CRP-mesto.

(G) Na vseh poteh se promotor vedno aktivira samo, če se prej CRP–cAMP veže na CRP-mesto.

(K) S formulami preverimo lastnosti alolaktoze.

(K1z) Na nobeni poti se na začetku ne zviša raven alolaktoze, dokler ne nastopi reakcija

laktoze in β-galaktozidaze, ki zmanjša količino laktoze.
(K13) Na vseh poteh je vedno, kadar se raven alolaktoze zmanjša, mogoče, da se nekoč poveča.

(L) Na nobeni poti se na začetku ne poveča količina alolaktoze, če v celico ne vstopi laktoza.

(M) Na vseh poteh je vedno, če obstaja alolaktoza v celici, mogoče, da se veže na laktozni represor.

(N) Na vseh poteh, kadarkoli je promotor reprimiran, ne postane nereprimiran, razen če se na represor ne veže alolaktoza.

(O) S formulami preverimo lastnosti vezave laktoznega represorja na operatorje. Enako kot za O1 lahko zapišemo še za O2 in O3.
(O11) Na vseh poteh je vedno mogoče, da se laktozni represor veže na O1.
(O12) Na vseh poteh je vedno mogoče, da se laktozni represor odcepi od O1.

(U) S formulami preverimo lastnosti operatorjev.
(U11) Na vseh poteh se vedno, kadar se represor veže na O2, od njega odcepi, preden se veže na O3.
(U21) Na nobeni poti se nikoli, kadar se laktozni represor veže na O1, ne odcepi, dokler se ne veže na O2 ali na O3.

(V) S formulami preverimo lastnosti β-galaktozidaze.
(V1z) Na nobeni poti količina β-galaktozidaze ne naraste, če se CRP–cAMP ne veže na CRP-mesto.
(V2z) Na nobeni poti se CRP–cAMP ne veže na CRP-mesto, dokler se cAMP ne veže na CRP.
(V7) Na vseh poteh vedno, kadar je količina β-galaktozidaze velika, pade, samo če promotor postane dezaktiviran ali reprimiran.
(V8) Na vseh poteh vedno, kadar je količina β-galaktozidaze majhna, naraste, samo če promotor postane aktiviran ali nereprimiran.

(W) Na vseh poteh se vedno, če je raven β-galaktozidaze nizka, glukoza lahko tvori, samo če raven β-galaktozidaze postane visoka.

(Y) Na vseh poteh je vedno, kadar količina glukoze naraste, mogoče, da se zmanjša količina β-galaktozidaze.

(Z) Na vseh poteh vedno, kadar raven glukoze pade, ne naraste, dokler se iz laktoze v celici ne tvori glukoza.

Samo na prvi pogled je presenetljiva neveljavnost formul V7, V8 in Z1 na sliki 5. Vzrok je podobno kot pri A21 to, da v modelu ni zagotovljeno, da se sondažna akcija izvede takoj po komunikaciji, za katero igra vlogo sonde. Da bi to vsaj delno zagotovili, smo v specifikaciji $SystemFull$ uporabili podoben princip kot pri akcijah $\overline{HIGH}$ in $\overline{LOW}$. Na sliki 1 vidimo, da je pri njiju akcija $high$ oziroma low nekakšna potrditev zahteve lev, ki poskrbi, da se komunicirajoča procesa ne nadaljujeta, dokler se poizvedovanje, vključno z izvedbo sonde, ne konča. Tako smo za vsako zunanjo nesondažno akcijo, takoj za katero je v procesu napisana sondažna akcija, uvedli potrditveno zunanjo nesondažno akcijo. To je, za vse prepovedane akcije specifikacije $SystemFull$ razen za $ubo23e$, $acte$ in $iacte$ ter lev, low in $high$ smo uvedli še akcijo z enakim imenom z dodano predpono ack in tudi potrditvene akcije pri kompoziciji prepovedali. V procesih pa smo jih uporabili takole. Vzemimo, da v enem procesu nastopa zaporedje akcij $a.s$, pri čemer je a zunanja vhodna nesondažna akcija, s pa sondažna, v drugem pa akcija $\overline{a}$, komplementarna akciji a. Potem smo v prvem procesu namesto $a.s$ napisali zaporedje akcij $a.s.\overline{acka}$, pri čemer je $acka$ potrditvena akcija akcije a, v drugem pa $a.acka$ namesto a. Analogno smo storili, če je bil a v prvem procesu izhodna, v drugem pa vhodna akcija (dejansko ni pomembno, v katerem od obeh procesov je potrditvena akcija vhodna, v katerem pa izhodna). Tako se je na primer zaporedje $?dbeta.!DBETA$ v procesu $Beta\_galactosidase\_high$ (slika 1) spremenilo v $?dbeta.!DBETA.!ackdbeta$, na obeh mestih v procesu $Promoter\_au$ pa smo za akcijo $!dbeta$ dodali $?ackdbeta$. Tudi LTS za novo specifikacijo $SystemFull$ se je tvoril hitro (3,55 s), imel pa je 1379904 stanj in 6405300 prehodov. EST je za tvorjenje LTS-ja porabil 106 MiB pomnilnika. Preverjanje lastnosti je za novo specifikacijo trajalo 8,12 s, v sistemu S1 in S2 pa 3,92 s oziroma 4,51 s. EST za preverjanje lastnosti ne zahteva rezervacije dodatnega pomnilnika. V novi specifikaciji so formule V7, V8 in Z1 veljavne, veljavnost preostalih formul s slike 5 pa se ne spremeni.

5 SKLEP

V članku smo izvedli formalno verifikacijo uravnavanja laktoznega operona. Neformalno smo opisali obnašanje in lastnosti biološkega procesa v bakteriji *E. coli* na gojišču z laktozo. Uporabili smo formalno specifikacijo sistema, opisano v članku [3]. Njegovim avtorjem je pomenila težavo kompleksnost specifikacije, zato smo preverili, kako lahko z njo opravi orodje EST. Ugotovili smo, da EST uspešno in v zelo kratkem času tvori označen sistem prehajanja stanj celo za specifikacijo, ki vsebuje še več akcij kot prvotna. Formalno smo specificirali lastnosti uravnavanja laktoznega operona s formulami ACTLW. Z metodo preverjanja modelov smo ugotavljali, ali jim sistem zadosti. Z začetnimi primeri lastnosti smo skušali prikazati, da je pomembno, da formule pa tudi njihov neformalni opis izražajo značilnosti vseh mogočih poti, da za določeno značilnost posebej izrazimo varnost in živost oziroma da se vedno znova nekaj lahko zgodi in da je treba morda z dodatnimi formulami poskrbeti, da bodo značilnosti izražene tudi za začetek poti. Vse to je v [3] le delno upoštevano.

Rezultati verifikacije se skladajo s pričakovanimi lastnostmi laktoznega operona, vendar smo za nekatere lastnosti to dosegli šele z uvedbo dodatnih akcij. Na podlagi rezultatov lahko z veliko verjetnostjo trdimo, da so lastnosti pravilno specificirane s formulami ACTLW in da specifikacija sistema zvesto opisuje realni sistem, kar se tiče lastnosti, ki so nas zanimale, razen živosti. Za različne snovi nam je uspelo potrditi samo veljavnost lastnosti možnosti. Da bi lahko potrdili tudi veljavnost pričakovanih lastnosti živosti, bi bilo treba preverjanje modelov v EST-u razširiti z možnostjo upoštevanja poštenostnih omejitev. Z njimi bi lahko dosegli, da bi se pregledovale samo tiste poti, na katerih se ne bi ves čas izvajale samo nekatere akcije, druge, ki bi se v realnosti zagotovo občasno izvedle, pa nikoli.

Logika ACTLW in orodje EST sta se sicer izkazala za dobro kombinacijo pri pisanju formalne specifikacije lastnosti in izvajanju verifikacije. Zelo nam je pomagala možnost tvorjenja protiprimerov. Težave so nastale zaradi jezika, podobnega CCS, ki zahteva uporabo sond. V prihodnje bi bilo jezik v EST-u dobro dopolniti tako, da bi lahko izbrali, naj se komplementarni akciji pri komunikaciji ne skrijeta za notranjo akcijo, ampak naj nova akcija odraža ime sodelujočih akcij. Tako bi za specifikacijo lastnosti lahko uporabili kar imena novih akcij in sonde ne bi bile potrebne. Verifikacija pa ni nujno namenjena samo preverjanju že znanih lastnosti. Želimo si, da bi za sistem odkrili lastnosti, ki so bile do zdaj neznane.

ZAHVALA

Raziskovalno delo Tatjane Kapus je sofinancirala Javna agencija za raziskovalno dejavnost Republike Slovenije (ARRS) v okviru programa P2-0069.

LITERATURA

[1] T. A. Basuki, "Model-checking biological systems using calculi of looping sequences," Ph.D. Thesis proposal.

[2] G. Bernot, J.-C. Comet, A. Richard, M. Chaves, J.-L. Gouzé, and F. Dayan , "Modeling and analysis of gene regulatory networks," in *Modeling in Computational Biology and Biomedicine: A Multidisciplinary Endeavour*, F. Cazals and P. Kornprobst, Eds. Springer, 2013, pp. 47–80.

[3] M. C. Pinto, L. Foss, J. C. M. Mombach, L. Ribeiro, "Modelling, property verification and behavioural equivalence of lactose operon regulation," *Computers in Biology and Medicine*, vol. 37, no. 2, pp. 134–148, Feb. 2007.

[4] R. Meolic, "Preverjanje pravilnosti obnašanja sistemov s sočasnostjo," Master's Thesis, Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, 1999.

[5] R. Meolic, "Notes on specifying systems in EST," in *Zbornik ERK 2006*, Portorož, Slovenia, vol. B, pp. 23–26, Sept. 2006.

[6] R. Milner, *Communication and Concurrency*, New York, NY, USA: Prentice-Hall, 1989.

[7] R. Meolic, "Akcijska logika dreves izvajanj z operatorjem unless," Ph.D. Dissertation, Univerza v Mariboru, Fakulteta za elektrotehniko, računalništvo in informatiko, 2005.

[8] J. C. Silva, R. Denny, C. Dorschel, M. V. Gorenstein, G. Z. Li, K. Richardson, D. Wall, and S. J. Geromanos, "Simultaneous qualitative and quantitative analysis of the Escherichia coli proteome: a sweet tale," *Molecular & Cellular Proteomics*, vol. 5, no. 4, pp. 589–607, 2006.

[9] P. Drábik, P. Maggiolo-Schettini, P. Milazzo, "Modular verification of interactive systems with an application to biology," *Scientific Annals of Computer Science*, vol. 21, no. 1, pp. 39–72, 2011.

[10] G. Bernot and J.-P. Comet, "On the use of temporal formal logic to model gene regulatory networks," in *Proc. Int. Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics*, LNBI 6160, Berlin: Springer-Verlag, 2010, pp. 112–138.

[11] T. A. Basuki, A. Cerone, P. Milazzo, "Translating stochastic CLS into Maude," *Electronic Notes in Theoretical Computer Science*, vol. 227, pp. 37–58, Feb. 2009.

[12] Wikipedia, "Calculus of communicating systems," Available: https://en.wikipedia.org/wiki/Calculus_of_communicating_systems. Accessed on: July 7, 2017.

[13] L. Lamport, "Proving possibility properties," *Theoretical Computer Science*, vol. 206, no. 1–2, pp. 341–352, Oct. 1998.

Rok Vogrin je diplomiral leta 2015 na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru, kjer je trenutno v absolventskem stažu magistrskega študijskega programa Telekomunikacije.

Robert Meolic je docent na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Njegovo raziskovalno področje so simbolične metode za formalno verifikacijo sistemov, še zlasti metoda preverjanja modelov. Je avtor orodja EST.

Tatjana Kapus je redna profesorica na Fakulteti za elektrotehniko, računalništvo in informatiko Univerze v Mariboru. Njeno glavno raziskovalno področje je formalna specifikacija in verifikacija reaktivnih sistemov, kot so na primer komunikacijski protokoli, sekvenčna digitalna vezja in biološki sistemi.