KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
VERSLO INFORMATIKOS KATEDRA

Raimonda Vyšniauskaitė

Hibridinės sistemos imitacinio modelio sudarymas agregatinėje modeliavimo sistemoje

Magistro darbas

Darbo vadovas
doc. V. Pilkauskas

Kaunas, 2007

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
VERSLO INFORMATIKOS KATEDRA

Raimonda Vyšniauskaitė

Hibridinės sistemos imitacinio modelio sudarymas agregatinėje modeliavimo sistemoje

Magistro darbas

Recenzentas
Doc. E. Valakevičius
2007-05-25

Vadovas
doc. V. Pilkauskas
2007-05-28

Atliko
IFM – 1/1 gr. stud.
R.Vyšniauskaitė
2007-05-28

Kaunas, 2007

SUMMARY

Simulation of Hybrid Systems using aggregate method

Hybrid systems are one of the oldest modeling systems. They describe the area of models, where discrete and continuous variables are simulated. The main goal of hybrid systems is to describe system, where management and system performance depends on discrete and continuous variables. There are a lot of different hybrid system models designed.

For the design and analysis of dynamic systems simulation is a declared technique. The development of simulation model requires starting with formal specification that is independent from the simulation implementation details. A general specification of dynamic system can specify a simulation model in one of the four following paradigms: a set of differential equations, a discrete time formalism, discrete event formalism and difference equation formalism. In this work I will overlook such discrete events formalisms: discrete event behavior model specification, structural model specification, dynamic system modeling in DEVS GDEVS and PLA. PLA specification of hybrid system has been explored. It has been adjusted for barrel filling system.

TURINYS

Paveikslų sąrašas:

ĮVADAS

Šio darbo **tyrimo sritis –** hibridinių sistemų imitacinis modeliavimas, panaudojant agregatinį metodą, **tyrimo objektas** – agregatinės sistemos imitacinio modeliavimo algoritmo pritaikymas hibridinių sistemų modelių realizavimui.

Hibridinė sistema – tai dinaminė sistema, kuri atskleidžia tolydžiųjų ir diskrečiųjų dinamiškųjų įvykių elgseną. Ši viena seniausių modeliavimų sistemų aprašo sistemas, kuriose valdymas ir sistemos veikimas priklauso nuo diskrečiųjų ir tolydžiųjų kintamųjų. Hibridinių sistemų pavyzdžių nagrinėjimo tikslas - iliustruoti hibridinių reiškinių įvairovę ir jų įvykius skirtingose programose.

Dinaminių sistemų projektavimui ir analizei bus naudojama imitacinio modeliavimo metodika. Imitacinis modeliavimas dažniausiai naudojamas projektavimo modelių kūrimui arba jau egzistuojančioms sistemoms, kurios aprašo tikrovę tam tikrame abstrakcijos lygyje. Išskiriami du skirtingi imitacinio modeliavimo aspektai: imitacinis modeliavimas modelio kūrimui ir imitacinis modeliavimas sistemos analizei. Imitacinių modelių kūrimas pradedamas nuo formalių specifikacijų, kurios yra nepriklausomos nuo imitacijos įgyvendinimo detalių. Bendras dinaminių sistemų specifikavimas gali būti apibrėžiamas keturiomis paradigmomis: diferencialinių lygčių rinkiniu, diskretaus laiko formalizavimu, diskretaus įvykio formalizavimu bei skirtuminių lygčių formalizavimu.

Hibridinių sistemų imitacinis modeliavimas turi du pirmtakus: imitacinis modeliavimas tolydžiųjų sistemų ir diskrečiųjų sistemų. Tolydžiųjų sistemų imitacinis modeliavimas yra nusistovėjusi sritis praktikoje ir tyrimų srityje. Diskrečiųjų sistemų imitacinis modeliavimas labai svarbus kompiuterių moksle. Hibridinių sistemų imitaciniuose modeliuose turi būti naudojamos naujos procedūros ir metodikos, leidžiančios nagrinėti hibridinius reiškinius.

Darbe bus apžvelgiami tokie diskrečiųjų įvykių formalizmai: diskretaus įvykio elgsenos modelio specifikacija, struktūrinio modelio specifikacija, dinaminių sistemų modeliavimas DEVS, GDEVS ir PLA. Bus nagrinėjamas hibridinės sistemos PLA specifikacijos sudarymas. Jis bus taikomas rezervuarų užpildymo sistemai. Šis formalizmas yra kalba, skirta aprašyti atkarpomis tiesinius agregatus.

Darbo struktūra:

- Skyriuje „Hibridinių sistemų imitacinis modeliavimas“ bus trumpai aprašoma keletas hibridinių sistemų, aptariama hibridinių sistemų imitacinio modeliavimo metodika, apžvelgiamas diskrečiųjų ir tolydžiųjų sistemų

imitacinis modeliavimas, pateikiami hibridinių sistemų imitacinio modeliavimo metodai.

- Skyriuje „DEVS formalizmo panaudojimas hibridinių sistemų modeliavimui“ bus pateikiami keleto formalizmų aprašymai, apžvelgiama GDEVS imitacija, detaliau aptariami GDEVS integratoriai.
- Skyriuje „PLA formalizmas“ bus aprašomi atkarpomis-tiesiniai agregatai, aptariamas valdančiųjų sekų metodas ir jo panaudojimas agregatų funkcionavimo formalizmui, pateikiamas agregatinių sistemų formalizavimas.
- Skyriuje „Hibridinės sistemos PLA specifikacijos sudarymas“ bus pateikiamas rezervuarų užpildymo konceptualusis modelis bei modeliavimo rezultatai.

1. HIBRIDINIŲ SISTEMŲ IMITACINIS MODELIAVIMAS

Hibridinės sistemos - tai vienos iš seniausių modeliavimo sistemų. Hibridinės sistemos apima modelių sritį, kur modeliuojami diskretieji kintamieji kartu su tolydžiaisiais. Hibridinių sistemų tikslas - aprašyti realias sistemas, kuriose reikia ir diskrečiųjų kintamųjų ir tolydžiųjų, kur valdymas ir sistemos veikimas priklauso būtent nuo jų. Sukurta nemažai skirtingų hibridinių sistemų. Pateikiame keletą iš jų:

- Hibridinis automatas (*HA, hybrid automata*)
- Hibridinio veikimo automatas (*hybrid behavioural automaton*)
- Hibridinis I/O automatas (*hybrid input/output automaton*)
- Procesų algebra hibridiniams automatams (*process algebras for hybrid systems*)
- Hibridinio proceso elgsenos skaičiavimas (*behavioural hybrid process calculus*)
- Viena kitą papildančios sistemos (*complementary systems*)
- Atkarpomis panašios sistemos (*piecewise affine systems*)
- Modelica (*modelica*)
- Masaccio (*masaccio*)
- Charon (*charon*)
- Ryšių grafas (*bond graphs*)

Hibridinis automatas – tai viena iš sėkmingiausių hibridinių sistemų, kuriai yra išvystyta teorija ir sukurta programinė įranga (PĮ). Tai vienas iš labiausiai naudojamų modelių pasaulyje.

Hibridinio veikimo automatas ir *Hibridinis I/O automatas* – tai sistemos tiesiogiai gimusios iš hibridinio automato. Jos papildo Hibridinį automatą, tačiau nėra išbaigtos teorijos ir nėra sukurtos PĮ modeliams tirti automatiškai.

Procesų algebra hibridiniams automatams – tai plati šaka, kur apimama bendrinė matematika procesams aprašyti, tačiau nėra išvystyta PĮ šiems modeliams tirti.

Hibridinio proceso elgsenos skaičiavimas – tai sintezė geriausių statinių ir dinaminių modelių. Šis modelis atsirado visai neseniai ir taip pat neturi išvystytos PĮ modeliams tirti.

Viena kitą papildančios sistemos ir *Atkarpomis panašios sistemos* – labiau skirtos diskretinėms tiesinėms sistemoms aprašyti.

Hibridinės sistemos – *Ryšių grafas, Charon* ir *Masaccio* nėra labai populiarios ir naudojamos tik specializuotoms užduotims spręsti.

Modelica [17, 30, 31]– pasižymi lengvu naudojimu, modelių, sudarytų iš blokų panašių į lego, kūrimu, turi galimybę apibrėžti modelio bibliotekas su pakartotinio naudojimo komponentais ir palaiko sudėtingų programų, įtraukiančių kitų sričių programų dalis, modeliavimą ir imitavimą. Pagrindinės savybės:

- *Modelica* – pagrįsta lygtimis, o ne priskiriamaisiais sakiniais. Tai leidžia modeliavimą, priklausantį nuo būsimų įėjimų. Vadinasi galimas pakartotinis klasių panaudojimas, kol lygtys nespecifikuoja tikslių duomenų srovės krypčių. Taigi Modelica klasės gali būti pritaikytos daugiau nei vienam duomenų srovės kontekstui.
- *Modelica* turi keleto sričių modeliavimo galimybę, turima omenyje, kad modelio komponentai atitinkantys fizinius objektus iš kelių skirtingų sričių (tokių kaip pvz., elektros, mechanikos, termodinamikos, hidraulikos, biologijos ir valdymo programos) gali būti aprašomi ir sujungiami.
- *Modelica* yra objektiškai orientuota kalba su bendra klasių koncepcija, kuri suvienodina klases, bendrumus – žinomus kaip šablonai C++ kalboje, ir bendrų potipių išskaidymą į pavienes kalbos konstrukcijas. Tai palengvina pakartotinį komponentų panaudojimą ir modelio vystymąsi.
- Modelica turi stiprų programinės įrangos komponentų modelį, su konstrukcija, leidžiančia kurti ir sujungti komponentus. Taigi ši kalba idealiai tinka kaip architektūrinė aprašymų kalba sudėtingoms fizinėms sistemoms ir tam tikro laipsnio programinės įrangos sistemoms.

1.1. Hibrinių sistemų imitacinio modeliavimo metodika

Imitacinis modeliavimas yra pripažinta metodika dinaminių sistemų projektavimui ir analizei [27]. Ji plačiai naudojama pramonėje ir universitete. Sudėtingų integruotų sistemų, turinčių daugybę skirtingų komponentų, projektavimas sunkiai įmanomas be imitacinio modeliavimo įrankių. Dažnai imitacinis modeliavimas naudojamas projektavimo modelių kūrimui arba egzistuojančioms sistemoms, kurios atitinkamai modeliuoja tikrovę.

Dažnai tokios sistemos arba atitinkamos jų abstrakcijos demonstruoja hibridinius reiškinius. Čia ir pasireiškia hibridinių sistemų imitacinis modeliavimas. Kadangi jis nėra taip plačiai paplitęs, kaip tolydaus laiko sistemų imitacinis modeliavimas, tai greitai tampa bendru, ypač integruotų sistemų srityse. Atskirkime dvi svarbias imitacinio modeliavimo programos vietas:

- Imitacinis modeliavimas modelio kūrimui;
- Imitacinis modeliavimas sistemos analizei.

Šie du pratimai pabrėžia šiek tiek skirtingus imitacinio modeliavimo aspektus, bet bendri bruožai yra vienodi.

1.1.1. Imitacinis modeliavimas *modelio kūrimui*

Imitacinis modeliavimas dažnai naudojamas įrankis modelio kūrimui. Dažnai įdarbinamas tęsti kelias procedūras.

- Kol kuriamas modelis, imitacinis modeliavimas naudojamas įsitikinimui, kad modelio komponentai ir pats modelis elgiasi kaip tikimasi.
- Jei išduodamas klaidos ženklas, imitacinio modeliavimo įrankiai gali padėti surasti kelią iki klaidos ir išanalizuoti jos atsiradimo priežastis.
- Be to, kartais imitacinis modeliavimas panaudojamas kaip testavimo metodas.

1.1.2. Imitacinis modeliavimas *sistemos analizėje*

Imitacinis modeliavimas naudojamas kaip sistemos analizės įrankis. Atsižvelgiant į modelį, sistemos tyrimui sudaroma aibė eksperimentų.

- Savybės gali būti testuojamos pasirinktinai. Tokie testai neužtikrina, kad savybės išliks visada, tačiau jie suteikia naudingą supratimą apie sistemos elgseną.
- Eksperimentai veikimo analizei gali būti projektuojami ir vykdomi. Rezultatai suteikia gerą intuiciją modelio efektyvumui, dalyvauja aptinkant silpnas vietas ir neefektyvius komponentus.

1.2. Tolydžių ir diskrečių sistemų imitacinis modeliavimas

Hibridinių sistemų imitacinis modeliavimas turi du principinius pirmtakus: imitacinis modeliavimas tolydžių sistemų ir diskrečių sistemų. Tolydžių sistemų imitacinis modeliavimas yra gerai žinomas pramoninėje praktikoje ir tyrimų srityje. Imitacinis modeliavimas diskrečių sistemų yra dažnai keičiamas testavimu programinės įrangos pramonėje ir verifikavimu (pvz., modelių tikrinimu) kompiuterių moksle.

1.2.1. *Tolydžių sistemų* imitacinis modeliavimas

Imitacinis modeliavimas tolydžių sistemų yra nusistovėjusi sritis moksle ir reguliariai naudojama pramonėje. Yra gausybė pramoninių ir akademinių įrankių tolydžių sistemų imitaciniam modeliavimui. Kai kurie jų palaiko tiktai eilinių diferencialinių lygčių (*ODE – ordinary differential equations*) imitacinį modeliavimą, o įrankiai palaikantys diferencialines algebrines lygtis (*DAE – differential algebraic equations*) tampa šablonu. Hibridinių sistemų imitaciniame modeliavime, tolydžių sistemų imitacinis modeliavimas gali būti matomas kaip

imitacinis modeliavimas. Hibridinės sistemos su tam tikrais papildomais reikalavimais (pvz., tikslus įvykių aptikimas) tolydi dalis duoda rezultatą, kad šioje tyrimų srityje ji gali būti panaudota hibridinių sistemų imitaciniame modeliavime. Plačiau apie tolydžių sistemų imitacinį modeliavimą galima rasti [9, 43].

1.2.2. *Diskrečių sistemų* imitacinis modeliavimas

Diskrečių sistemų imitacinis modeliavimas labai svarbus kompiuterių moksle taip pat kaip ir valdymo teorija. Dažniausiai tai yra naudojama įvairių tipų automatų [4, 5, 26] imitaciniam modeliavimui, algebros apdorojimui [6, 13, 37], kompiuterių moksle arba diskrečių įvykių sistemų imitaciniame modeliavime [43] valdymo teorijoje.

1.3. Hibridinių sistemų imitacinis modeliavimas

Hibridinių sistemų imitacinis modeliavimas jungia tolydaus laiko ir diskretaus funkcionavimo imitacinį modeliavimą. Deja, neužtenka sudėti tik šių tipų imituoklius, kadangi papildomos reikšmės reikalingos, nagrinėjant dviejų pasaulių sąveikas. Dėl to naujos procedūros ir metodika kuriama nagrinėjimui hibridinių reiškinių, kaip tolydaus laiko ir diskretaus funkcionavimo.

Egzistuoja keletas metodikų hibridinių sistemų imitaciniam modeliavimui. Pagrindinės [36]:

1.3.1. Glotnus metodas (*The Smoothing method*)

Hibridinis modelis transformuojamas į glotnų metodą, kuris suvienodina hibridinio modelio pasirinktus aspektus. Šis būdas sukuria įspūdį, kad jis yra besaikis (pvz., sistema modeliuojama kaip hibridinė ir tada vėl transformuojama į glotnią sistemą). Potenciali nauda yra ta, kad gali būti lengviau sumodeliuoti sistemą, kaip hibridinę. Detaliau galima rasti [36].

1.3.2. Įvykio sekimo metodas (*The event tracking method*)

Įvykio sekimo metodas yra paplitęs būdas hibridinių sistemų imitaciniam modeliavimui. Imitacinio modeliavimo seka [36] tokia:

1. Įvykio apdorojimas:
   - Inicialų (naujos) tolydžios būsenos pažymėjimas;
   - Apibrėžimas (naujo) metodo.
2. Tolydaus laiko dinamikos imitacinis modeliavimas su duotu metodu.
3. Įvykio sekimas.

Pirmiausia nustatomos pradinės sąlygos (pradiniai metodai/procesai ir pradinės signalų reikšmės). Tada tolydaus laiko funkcionavimas modeliuojamas, kol susekamas įvykis.

Jei įvykis atsiranda, tiksli (arba artima) įvykio laiko reikšmė ir atitinkama tolydžios būsenos (signalo) reikšmė apskaičiuojama. Tada naujos reikšmės ir metodas apibrėžiami, o imitacinio modeliavimo ciklas kartojamas.

1.3.3. Laiko žingsnio metodas (*The time-stepping method*)

Laiko žingsnio metode įvykiai nėra sekami. Pasirenkama tam tikra diskreti schema ir tuomet hibridinė sistema priartėja prie diskretizuotos sistemos. Šis būdas tinkamas hibridinių sistemų su atidžiai pasirinktomis diskretizavimo schemomis specifinėms klasėms. Detaliau galima rasti [36].

1.3.4. Įvykio sekimo algoritmas

Įvykio sekimo metodas yra hibridinių sistemų imitacinio modeliavimo standartas. Anderson [1] aptaria šios metodikos programas, skirtas objektiškai orientuotoms imitacinio modeliavimo hibridinėms sistemoms. Panaudojimas įvykio sekimo 20-sim įrankyje (Skyrius 7,9) aptariamas [7]. [32] aptaria programas su įvykio sekimo metodu, skirtas fizinių dinaminių sistemų imitaciniam modeliavimui. Hibridinių sistemų imitacinis modeliavimas Ptolemy II (skyrius 7,9) aptariamas [28]. [3] naudoja klasikinį būdą su mažais pakitimais hibridinių dinaminių sistemų modeliavimui ir imitacinio modeliavimo struktūrai.

Tuo tarpu Fabian et al. [15] perėmė imitacinio modeliavimo algoritmą Hybrid X (skyrius 5) pridėdamas procesų apdorojimą. Algoritmas veikia tokia seka:

1. Pažymimi sistemos (būsenos ir procesų) inicialai.

2. Kol yra aktyvių procesų ir imitacinio modeliavimo laikas nepasibaigęs:

(a) Kol aktyvūs procesai:

i. Vykdomi neblokuojami sakiniai aktyvių procesų;

ii. Pradinės būsenos apskaičiuojamos;

iii. Tikrinamos sąlygos ir procesai rūšiuojami į aktyvius ir pasyvius.

(b) Jei yra laiko pereikvojimai, kurie bus aktyvūs greičiau nei minimalus sprendimo žingsnis, procesai, kurie buvo blokuojami laiko, aktyvuojami ir ciklas žingsnyje (2) kartojamas.

(c) Jei minimalus sprendimo žingsnis yra trumpesnis nei laiko iki artimiausio laiko pereikvojimo DAE yra sprendžiama ir ciklas (2) kartojamas.

2. DEVS FORMALIZMO PANAUDOJIMAS HIBRIDINIŲ SISTEMŲ MODELIAVIMUI

2.1. Apibendrintos diskrečiųjų įvykių abstrakcijos tolydžiosiose sistemose

Imitacinių modelių kūrimas pradedamas nuo formalių specifikacijų, kurios yra nepriklausomos nuo imitacijos įgyvendinimo detalių [23]. Bendras dinaminių sistemų specifikavimas gali būti apibrėžiamas trimis paradigmomis: diferencialinių lygčių rinkiniu, diskretaus laiko formalizavimu ir diskretaus įvykio formalizavimu. Kruopščios analizės metu galima išskirti ir ketvirtą specifikavimo paradigmą – skirtingų lygčių formalizavimą.

Šių dienų skaitmeniniai kompiuteriai reikalauja diskretaus įvykio arba diskretaus laiko metodikos naudojimo. Pagrįstos diskretaus laiko metodika, tolydžiosios imitacijos gali reprezentuoti „švelnius pokyčius“ (*soft changes*) sistemos elgsenoje, o pagrįstos diskretaus įvykio metodika imitacijos – „šiurkščius pokyčius“ (*abrupt changes*) sistemos elgsenoje.

Klasikinės diskretaus įvykio abstrakcijos pagrindinė charakteristika [40, 41] yra ta, kad ji naudoja per visą atkarpą nekintančias įėjimo-išėjimo trajektorijas tam, kad suformuotų tolydžiųjų dinaminių sistemų diskrečiuosius įvykių modelius.

GDEVS (*Generalized Discrete Event Specification*) [21, 22] – tai naujas abstrakcijų apibendrinimo būdas. Jame įvesties, išvesties ir būsenos trajektorijos gaunamos iš atkarpų polinominių segmentų. Modeliuojant tolydžiuosius procesus, kaip klasikines diskrečiųjų įvykių abstrakcijas, kurios leidžia spartesnį vykdymą, sutartinių polinominių funkcijų panaudojimas segmentams užtikrina didesnį tikslumą. GDEVS abstrakcijos buvo panaudotos loginiams įėjimams su neapibrėžtais vėlinimais [25] modeliuoti ir imituoti. Be to, GDEVS paradigma siūlo galimybę formuoti tolygų būdą hibridinių (sudarytų iš tolydžiųjų ir diskrečiųjų komponentų) sistemų modeliavimui.

Aptarsime apibendrintus diskrečiųjų įvykių modelius integratoriams, naudojant atkarpomis tiesines įėjimų ir išėjimų trajektorijas. Demonstruosime būsenos perėjimus, atsižvelgdami į sistemos vidinius ir išorinius įvykius, kurie taip pat lengvai formalizuojami GDEVS‘e. Taip pat vaizduosime, kaip išėjimų klaidos lieka mažesnės GDEVS modelyje.

2.2. Keleto formalizmų apžvalga

2.2.1. Diskretaus įvykio elgsenos modelio specifikacija

Pagal DEVS [35, 40, 42] literatūrą, diskretaus įvykio modelio specifikacija yra struktūra, tarkim *M*:

$M =< X, S, Y, \delta_{\text{int}}, \delta_{ext}, l, D >$; čia

X – aibė įėjimų įvykių;

S – aibė nuoseklių būsenų;

Y – aibė išėjimų įvykių;

δ_{int} – vidinių perėjimų funkcijos, kurios apibrėžia būsenos pokyčius sukeltus vidinių įvykių;

δ_{ext} – išorinių perėjimų funkcijos, kurios apibrėžia būsenos pokyčius sukeltus išorinių įvykių;

l – išėjimo funkcijos;

D – nusako maksimalų būsenos ilgį ar gyvavimo trukmę.

[42] pristato koncepciją, nusakančią visas būsenas (*TS – total states*) kaip sistemą:

$TS = \{(s,e) s \in S, 0 < e < D(s)\}$, kur e nusako praėjusį laiką būsenoje s. Ši koncepcija yra fundamentali. Ji leidžia apibrėžti pagrįstą praėjusiu laiku būsimą būseną dabartinėje būsenoje. Potenciali nauda glūdi galimybėje filtruoti įvykius [10, 20], kuriuose planuojamas būsenos pakeitimas bus realizuojamas modeliu tik tada, kai laiko intervalas, skiriantis du pagrindinius įvykius, viršys iš anksto nustatytas reikšmes ir apims mechaninį įvykių filtravimą konceptualiame lygyje. Pagrindinis indėlis DEVS'o glūdi tradicinės perėjimo funkcijos susiskaidyme į vidinių ir išorinių perėjimų funkcijas. Vidinių perėjimų funkcija apibrėžta $\delta_{\text{int}} : S \to S$, leidžia užimti savarankišką modelio raidą. Kai modelis yra būsenoje s laiku t_i, tada galimas perėjimas į būseną $s' = \delta_{\text{int}}(s)$ laiku $t_i + D(s)$ numatant, kad jokio išorinio įvykio negali atsirasti tuo metu.

2.2.2. Struktūrinio modelio specifikacija

Struktūrinis modelis apibrėžiamas tokia struktūra:

$CM =< X, Y, M, D, E_{IC}, E_{OC}, I_C, Select >$, kur

X - išoriniai įvesties įvykiai;

Y - aibė išėjimų įvykių;

D - aibė komponentų vardų;

$M = Md \mid d \in D$ - aibė komponentų;

E_{IC} - išorinių įėjimų poravimas;

E_{OC} - išorinių išėjimų poravimas;

I_C - vidiniai poravimai;

Select papildoma funkcija, pvz. $Select : 2^D - \{\} \to D$.

2.2.3. Dinaminių sistemų modeliavimas DEVS

Dinaminių sistemų modeliavimas DEVS‘e [29, 38, 39] susideda iš laiko intervale nekintančių dinaminės sistemos segmentų, kurie gaunami per diskrečius įvykius sukuriant įėjimus, išėjimus ir būsenos trajektorijas.

Tolydžiosios dinaminės sistemos apibūdinamos laiko intervale nekintančiais įėjimų ir išėjimų segmentais. Kol išėjimai DEVS‘e yra tiesioginė vidinių būsenų funkcija, būsenos segmentas su unikalia reikšme gali būti siejamas su kiekvienu laiko intervale nekintančiu išėjimo segmentu, apibrėžtu per tą patį intervalą, kaip ir išėjimo segmentas. Be to, kaip išėjimo segmento reikšmė yra konstanta su intervalu apibrėžiančiu segmentą. Būsenos segmentas gali būti laikomas laiko intervale nekintančiu be jokių nuostolių. Laiko intervale nekintančios būsenos segmentai kartu sudaro būsenos trajektoriją. Tai svarbus žingsnis formuojant DEVS specifikaciją dinaminėms sistemoms.

2.3. GDEVS imitacija

2.3.1. GDEVS modelių imitacijos semantika

Detaliau apie elgsenos semantiką galima rasti [24, 41]. Elgsenos (atominių) GDEVS modelių imitacijos semantika yra duota GDEVS konceptualaus modeliavimo įrenginio. Jis yra identiškas DEVS modeliavimo įrenginiui, bet įvykis apibrėžiamas n-nariu vietoje unikalios reikšmės.

Modeliavimo įrenginys naudoja du laiko kintamuosius t_l ir t_n. Pirmasis nusako imitacijos laiką, kai paskutinis įvykis atsiranda, o antrasis – suplanuotą laiką, sekančio intervalo įvykiui (jis gali būti begalinis pastoviai būsenai).

Pagal gyvavimo laiko funkcijos apibrėžimą, seka, kad $t_n = t_l + t_a(s)$. Jei duotas globalus imitacijos laikas *t*, modeliavimo įrenginys gali skaičiuoti nuo šio kintamojo praėjusį laiką diskrečioje būsenoje nuo paskutinio įvykio $e = t - t_l$.

GDEVS modeliavimo įrenginys.

kintamieji:

t_1 // paskutinio įvykio laikas;

t_n // sekančio įvykio laikas;

y // veikiamojo išėjimo įvykio reikšmė, susietų modelių, kai vidinis įvykis, laiku *t*.

$y = \lambda(s)$ siunčia išėjimo įvykį:

$s = \delta_{\text{int}}(s)$

$t_1 = t$

$t_n = t_1 + ta(s)$

kai gaunamas įėjimo įvykis *(x,t)* laiku *t*, su įėjimo reikšme *x*

$e = t - t_1$

$s = \delta_{ext}(s, e, x)$

$t_1 = t$

$t_n = t_1 + ta(s)$

end.

2.3.2. Pagrindinės charakteristikos GDEVS modeliavimo įrenginio

GDEVS modeliavimo įrenginys konceptualiai skiriasi nuo DEVS modeliavimo įrenginio tiktai „įvykio“ apibrėžimais ir „žinutės“ klasėmis. GDEVS‘e reikšmių sąrašas korektiškai gerai parengtas grandinės sąrašo formoje, atsakančioje į koeficiento-įvykį.

Tikslus numeris reikšmių sąraše apibrėžiamas polinomine tvarka. Panašios modifikacijos taikomos „žinučių“ klasei.

DiamSim - išvystytas programinės įrangos paketas, skirtas apibendrintų diskrečiųjų įvykių imitacijai. Šis paketas naudoja į įvykį orientuotą imitacijos branduolį [14] ir į vartotoją orientuotą imitacijos kalbą [12, 25].

2.3.3. GDEVS specifikacija ir hibridinių sistemų imitacija

Hibridinė sistema [35] yra tokia, kurioje modelio specifikacija įtraukia abu tolydžiuosius aprašymus diferencialinių lygčių formoje ir diskrečiuosius įvykius. H. Praehofer [35] suformavo formalizmą, skirtą hibridinės imitacijos modelių specifikavimui. Šis formalizmas naudoja tolydžiųjų ir diskrečiųjų įvykių poaibius. Taip pat H. Praehofer iliustravo idėją rezervuarų užpildymo sistemai. Pagal GDEVS pastoviai naudojant diskrečiuosius įvykius hibridinė sistema gali būti specifikuota ir duos panašius kokybės rezultatus.

Rezervuarų užpildymo pavyzdys yra geras įrankis intuityviai suprasti apibendrinimo įtaką, būdingą GDEVS būdui. Jis parodo, kad hibridinėms sistemoms galima įgyvendinti įvykio varomą imitaciją, vietoje diskretaus įvykio imituoklio (panaudojant tolydaus laiko pagrindą) jungimo su tolydžiu imituokiu (panaudojant diskretaus laiko pagrindą).

Rezervuarų užpildymo sistema charakterizuojama per tolydžiuosius įėjimus, *inflow*, tolydžiuosius išėjimus, *content*, ir diskrečiuosius įvykių išėjimus, *barrel*. *Content* taip pat

veikia kaip būsenos kintamasis. *Inflow* sukelia *content* reikšmės pakitimus ir pirma *content* išvestinė reikšmė yra lygi *inflow*. Rezervuaras laikomas pilnu kai *content* kintamasis pasiekia 10 reikšmę ir tada išvedama per diskrečiąją išėjimo jungtį *barrel.*

Kintamojo *content* reikšmė prilyginama nuliui. Duota, kad *content* pristato *inflow* įkomponavimą laike, jo elgsena yra atkarpomis tiesinė ir jo individualūs segmentai gali būti išreikšti GDEVS‘e diskrečiųjų kartu veikiančių įvykių formoje. Taigi rezervuarų generatoriaus sistema gali būti modeliuojama suvienodinant diskrečiuosius įvykius GDEVS‘e. *Barrel* ir *content* perduoda tradicinę hibridinę specifikaciją.

GDEVS imitacijos vykdymo procesas tiriamas per būsenos perėjimo vykdymą ir išėjimų funkcijas, naudojant GDEVS konceptualų modeliavimo įrenginį.

Tarkime, kad dvinaris *(ac, bc)* pristato tiesinės funkcijos koeficientą, *content*, ir kad *sigma* tiksliai nusako būsenos gyvavimo laiką, tada kitos funkcijos sukomplektuoja GDEVS modelio apibrėžimą.

$\delta_{ext}(((ac, bc), sigma, in), e, inflow)$

$bc := ac.e + bc$

$ac := inflow$

$in := true$

$sigma := 0$

$\delta_{\text{int}}((ac, bc), sigma, in)$

if in = false then $bc := 0$

$sigma := 10 / ac$

else if ac = 0 then $sigma := infinity$

else $sigma := (10 - bc) / ac$

endif

in:=false

endif

$\lambda((ac, bc), sigma, in)$

if in = false send out the barrel to output barrel

and send out (ac,0) to port-content

else send out (ac,bc) to port-content

2.4. Integratorius

Naudojant sujungtų modelių koncepciją, sudėtinių modelių charakterizavimui siūlomas pirmos eilės GDEVS integratoriaus modelis. Tai pagrindinis elementas

tolydžiuosiuose modeliuose [8]. Pamatysime, kaip GDEVS integratoriaus išėjimas pasilieka mažesnis, nei duotoji tolerancija ir kokia yra galimybė pakeisti šią toleranciją, atsižvelgiant į skirtingas strategines alternatyvas.

2.4.1. Pirmos eilės GDEVS integratoriaus abstrakcija

Tarkime, kad realaus laiko imitacinės sistemos įėjimo signalas, pavadintas integratoriumi, yra seka tiesinių segmentų. Jie nusakyti trinariu (t_i, s_i, X_i) , kur t_i - laikas, per kurį įvyksta įvykis, s_i - nuožulnumas naujos įėjimo atkarpos trajektorijos, ir X_i įėjimo reikšmė laiku t_i. Įėjimo trajektorijos segmentai apibrėžti kaip:

$$x(t) = s_i(t - t_i) + X_i,\ \forall t \in [t_i, t_{i+1}].$$

Jų integralas ir integratoriaus išėjimas yra parabolė:

$$y_v(t) = \frac{1}{2s_i(t-t)^2} + X_i(t - t_i) + y_v(t_i).$$

Praktikoje, realaus laiko situacijoje, sekančio įvykio atsitikimas, pvz. t_{i+1}, nėra žinomas. Tikslas yra apibūdinti išėjimo trajektorijos seką atkarpomis tiesinių segmentų $\{y_j(t)\}_j$, apibrėžtų laiko intervalų seka $\{[t_{i,j}, t_{i,j+1}]\}_j$ tokia, kad:

$[t_i, t_{i+1}] = E_{j=1}^{n}[t_{i,j}, t_{i,j+1}]$, kur $t_{i,1} \equiv t_i$, $t_{i,n_i+1} \equiv t_{i+1}$, $\exists n \in \mathrm{N} : t_{i+1} - t_i = n\tau_i$, su $\tau_i = t_{i,j+1} - t_{i,j}$ ir kur išėjimo signalas $y(t)$ tenkina:

$$\forall j \in \{1,2,...n_i\}\ \exists \beta_{i,j} \in R\ \ \forall t \in [t_{i,j}, t_{i,j+1}[,$$

$$y(t) \equiv y_j(t) = \beta_{i,j}(t - t_{i,j}) + y_v(t_{i,j});$$

$|y(t) - y_v(t)| < \varepsilon$ užtikrina mažesnį klaidų skaičių modelyje nei duotoji tolerancija ε.

Kai kurios parabolės savybės veda prie fakto, kad išėjimo klaida $y(t) - y_v(t)$ pasiekia savo limitą ε per laiką τ_i, duota iš: $\tau_i = \sqrt{\frac{2\varepsilon}{|s_i|}}$ ir laiko intervalai yra vienodo ilgio, pvz., $\tau_i = t_{i+1} - t_i$. Tuomet procedūra yra išėjimo reikšmės forsavimas prie tikrosios reikšmės $y_v(\tau_i)$, pridedant arba atimant esamą išėjimo reikšmę $y_j(\tau_i)$ prie tolerancijos ε, atsižvelgiant į įėjimo poslinkio požymį $y_v(\tau_i) = y_j(\tau_i) + sign(s_i) \cdot \varepsilon$.

Taigi segmentacija išėjimo trajektorijos gali būti vykdoma atsižvelgiant į $y(t) \equiv y_j(t) = \beta_{i,j}(t - t_{i,j}) + y_v(t_{i,j})$ išraišką laiko intervale $[t_i, t_{i+1}]$, kur $t_{i+1} = t_i + \tau_i$ [8].

Komentarai:

Pagrindinis tikslas yra pasiektas tuo požiūriu, kad pagal tinkamą tikslumą galima modeliuoti integratoriaus išėjimą. Naudojant tą patį formalizmą, su kuriuo įėjimo signalas yra duotas, t.y. su atkarpomis-tiesinių trajektorijų apibūdinimo parametrais. Tiksliau, galima palyginti išėjimo trinarį $(t_{i,j}, \beta_{i,j}, Y_{i,j})$, nusakantį išėjimo įvykius, ir atitinkamai įėjimo trinarį (t_i, s_i, X_i) , nusakantį įėjimo įvykius.

GDEVS modelius galima pasiūlyti realaus laiko imitacinėms sistemoms, aprašant taisykles, numatančias būsenos perėjimų aprašymus:

1. Išoriniai perėjimai, naudojant δ_{ext} funkciją, dėl pakeitimų į įėjimų įvykius (t_i, s_i, X_i) .

2. Vidiniai perėjimai, naudojant δ_{int} funkciją, dėl tikslumo apribojimo, t.y. ε, ir jo rezultato τ_i , vedant prie išėjimo įvykių $(t_{i,j}, \beta_{i,j}, Y_{i,j})$ išraiškų.

2.4.2. Aukštesnės eilės integratoriaus GDEVS abstrakcija

Nagrinėjamas atvejis, kai įėjimo ir išėjimo trajektorijos daugiau nebėra atkarpomis tiesiniai segmentai, o polinominės laiko tvarkos n funkcijos tokios, kad: at^n .

n-tos eilės polinomą $P_n(t) = a_n t^n + a_{n-1} t^{n-1} + ... + a_1 t^1 + a_0 t^0$ galima užrašyti trumpiau: $P_n(t) \equiv at^n$.

Tarkim, P yra aibė polinomų, apibrėžtų R^n, o $Ø : \mathrm{P} \to \mathrm{P}$ paskirstymas apibrėžtas P. P yra matrica, nuo kurios priklauso pagrindiniai pasikeitimai. Ji yra tokia, kad bet kuri polinomo funkcija n-tos eilės $x_n(t)$ susijusi su n-tos eilės polinomine funkcija $x_n'(t')$:

$$Ø(\mathrm{x_n}(t)) \equiv \mathrm{x_n'}(t')$$

$$x_n(t) = a_{n-1} t^{n-1} + ... + a_1 t + a_0$$

$$x_n'(t') \equiv at'^n$$

Taigi, problema dabar sumažėja šitaip:

duota įėjimo trajektorija *x(t)* nusakyta seka atkarpų n-tos eilės polinomų segmentų $\{x_i(t) = a_i t^n\}_i$ - suprastinimui numetame kablelius tarp kintamųjų *x,y,...t* – ar galima charakterizuoti integruotą signalą $y_v(t)$ seka atkarpų n-tos eilės polinominių segmentų $\{y_k(t) = b_k t^n\}_k$.

Tuomet pagrindinė problema:

duota įėjimo trajektorija $x(t) = at^n$ laiko intervale $[0, t_i]$, ar yra kokių nors baigtinių sekų $\{y_j(t)\}_j$, apibrėžtų laiko intervalų seka $\{[t_j, t_{j+1}]\}_j$, priartėjančių prie tikros integralinės funkcijos $y_v(t)$ nuo $x(t)$ tokios, kad:

- $[0, t_i] = \bigcup_{j=1}^{n} [t_j, t_{j+1}]$, $t_1 = 0$, $t_{n+1} = t_i$;
- $y_j(t) = b_j (t - t_j)^n + Y_j$, $\forall t \in [t_j, t_{j+1}]$, kur $Y_j \equiv y_v(t_j)$;
- Duotam realiam teigiamam ε, $|y_v(t) - y_j(t)| \le \varepsilon$, $\forall t \in [t_j, t_{j+1}]$.

Kaip ir $\tau_i = \sqrt{\frac{2\varepsilon}{|s_i|}}$, maksimali leidžiama klaida ε pasiekiama tuo pačiu laiku τ, paprastai susietu su įėjimo poslinkiu *a*, išėjimo poslinkiu *b* ir trajektorijos eile *n*: $\tau = n\frac{b}{a}$.

Leidžiamas išėjimo poslinkis nusakomas: $b^{n+1} = \frac{n+1}{n^n} \varepsilon |a|^n$. Kai $n = 1$, rezultatai gaunami kaip ir pirmos eilės integratoriaus. Parametrizuotam aprašymui įėjimo–išėjimo trajektorijos gali pristatyti sekančių parametrų, galutinai apibrėžiančių įėjimo ir išėjimo įvykius, vektorius:

$(t_i, a_{n,i}, a_{n-1,i}, ..., a_{1,i}, a_{0,i})$;

$(t_{i,j}, b_{n,i,j}, b_{n-1,i,j}, ..., b_{1,i,j}, b_{0,i,j})$;

atsakant į įvykių laiko pasireiškimą ir jų polinominius koeficientus, atitinkamai įėjimo ir išėjimo įvykiams.

Pasiūlytas integratorius pirmiausia pasikeis naudodamas paskirstymą Ø, įėjimo įvykį $(t_i, a_{n,i}, a_{n-1,i}, ..., a_{1,i}, a_{0,i})$ pagrindiniame įėjimo įvykyje (t_i', a_i') prieinamame laiko intervale $[t_i', t_{i+1}']$. Tuomet išėjimo segmentacija vykdoma atsižvelgiant į $\tau_i = \sqrt{\frac{2\varepsilon}{|s_i|}}$ ir $b^{n+1} = \frac{n+1}{n^n} \varepsilon |a|^n$, pateikiant pagrindinius išėjimo įvykius $(t_{i,j}', b_{i,j}')$. Galiausiai, naudojant atvirkštinį paskirstymą $Ø^{-1}$ gaunami $(t_{i,j}, b_{n,i,j}, b_{n-1,i,j}, ..., b_{1,i,j}, b_{0,i,j})$ išėjimo įvykiai. Ši procedūra yra visiškai suprantama vartotojui.

2.4.3. GDEVS ir hibridinės sistemos

Hibridinio metodo imitacija yra diskretaus įvykio modelių ir diferencialinių lygčių modelių imitacija[2, 11, 16, 33, 34]. Hibridinio metodo imitacija įtraukia diskretaus įvykio imitatoriaus (su tolydaus laiko pagrindu) ir tolydaus imitatoriaus (su diskretaus laiko pagrindu) sujungimą. Tolydieji imitatoriai naudoja laiko varomą mechanizmą (laiko progresas

žingsniais), tuo tarpu diskretieji įvykio imitatoriai naudoja įvykio varomą paradigmą (laiko progresas šuoliais). Efektyvus metodas sinchronizacijai tarp dviejų laiko pristatymų lieka sunkiai suprantamas. Ši sinchronizavimo problema neegzistuoja GDEVS paradigmoje, kadangi laiko atvaizdavimas yra unikalus ir visi imitatoriai yra įvykiu varomi.

Esminė problema lygiagrečios ir paskirstytos imitacijos yra sinchronizacija. *Time Warp* algoritmas [18, 19] yra dažniausiai naudojamas sinchronizacijos protokolas, diskretaus įvykio modelių paskirstytosioms imitacijoms. Bet paskirstytos hibridinių modelių imitacijos numato specifinių protokolų panaudojimą, kurie gali vadovauti tolydaus ir diskretaus laiko atvaizdavimams.

Naudojant GDEVS paradigmą hibridiniam modeliavimui, tereikia panaudoti Time Warp algoritmą, kuris padėtų išvengti specifinių sinchronizacijos protokolų panaudojimo.

3. PLA FORMALIZMAS

3.1. Atkarpomis-tiesiniai agregatai

Aprašant sistemą, sistemos būsenų aibėje S yra išskiriama baigtinė pagrindinių būsenų aibė $I = \{0, 1, 2, \ldots, s\}$. Šios aibės elementai $\nu \in I$ yra pagrindinės agregato būsenos. Kiekvienai pagrindinei būsenai yra priskiriamas sveikas neneigiamas skaičius $\|\nu\|$, kuris vadinamas būsenos rangu, bei išgaubtas daugiakampis $Z^{(\nu)}$, nusakytas $\|\nu\|$ išmatavimų Euklido erdvėje. Sakoma, kad būsenų aibę $Z = \bigcup_{\nu \in I} Z^{(\nu)}$ sudaro poros $(\nu, z^{(\nu)})$, čia $\nu \in I$, o $z^{(\nu)} \in Z^{(\nu)}$. $z^{(\nu)}$ – vadinamos papildomomis agregato koordinatėmis.

Pradiniu laiko momentu t_0 agregatas yra būsenoje $z(t_0) = (\nu, z^{(\nu)}(0))$, čia $z^{(\nu)}(0) \in Z^{(\nu)}$. Jei nėra įėjimo signalo, kai $t > t_0$, taškas $z^{(\nu)}(t)$ juda srityje $Z^{(\nu)}$ tol, kol pasieks šios srities kontūrą. Laiko momentas t_1, kai pasiekiamas kontūras, vadinamas *atraminiu*.

Daugiakampio *Z* kontūras yra aprašomas lygtimis:

$$\sum_{i=1}^{\|\nu\|} \gamma_{ji}^{(\nu)} z_i^{(\nu)} + \gamma_{j0}^{(\nu)} = 0, \; j = 1, \ldots, m(\nu);$$

čia m(ν) – kontūrų skaičius;

$z_i^{(\nu)}$ – vektoriaus $z^{(\nu)}$ komponentės, $i = 1, \ldots, \|\nu\|$;

$\gamma_{ji}^{(\nu)}$ – sistemos parametrais apibūdinami dydžiai.

Atraminiu laiko momentu agregato pagrindinė būsena kinta iš ν į ν', o agregato papildomos koordinatės įgyja reikšmę $z^{(\nu')}(t_1) \in Z^{(\nu')}$.

Papildomos koordinatės kinta srityje $Z^{(\nu')}$ tol, kol nepatenka ant šios srities kontūro. Tam įvykus, agregatas vėl keičia būseną.

Atkarpomis-tiesinio agregato būsenai patekus į srities kontūrą, yra generuojamas išėjimo signalas $y \in Y$, čia Y – išėjimo signalų aibė, kuri yra analogiška aibei *Z*. Išėjimo signalo struktūra

$$y = (\lambda, y^{(\lambda)});$$

čia λ – išėjimo signalo diskreti dalis;

$y^{(\lambda)}$ – išėjimo signalo papildomų koordinačių vektorius, priklausantis nuo λ:

$$y^{(\lambda)} = (y_1^{(\lambda)}, \ldots, y_{r(\lambda)}^{(\lambda)}).$$

Jei laiko momentu t^* į agregatą yra paduodamas įėjimo signalas, tai agregato papildomos koordinatės nustoja kitusios ir agregato būsena akimirksniu pereina į kitą tos pačios ar kitos srities $Z^{(\nu'')}$ tašką.

Įėjimo signalas turi struktūrą, analogišką išėjimo signalo struktūrai, t.y.: $x=\left(\mu, x^{(\mu)}\right)$.

Įėjimo signalo atėjimo momentu agregatas išduoda išėjimo signalą ir šis momentas taip pat yra atraminis. Toliau taškas $z^{(\nu'')}(t)$ srityje $Z^{(\nu'')}$ juda taip pat, kaip buvo aprašyta anksčiau.

Kai nėra įėjimo signalų, atkarpomis-tiesinio agregato būsenos kitimas aprašomas tokiomis lygtimis:

$$\nu(t)=\nu=const;\ \frac{dz^{(\nu)}}{dt}=-\alpha^{(\nu)};$$

čia $\alpha^{(\nu)}$ – pastovus vektorius, turintis pavidalą $\alpha^{(\nu)}=\left(\alpha_1^{(\nu)},\ldots,\alpha_{m(\nu)}^{(\nu)}\right)$.

Vektorinėje formoje diferencialinės lygties sprendinys $z^{(\nu)}(t)$ gali būti užrašytas

$$z^{(\nu)}(t)=z^{(\nu)}(0)+\alpha^{(\nu)}(t-t_0).$$

Spręsdami papildomų koordinačių kitimo ir sričių kontūrų lygtis, kartu galime apskaičiuoti ir laiko momentus, kai papildomų koordinačių reikšmės patenka į kontūrus. Pvz., pirmas atraminis laiko momentas:

$$t_1=\min_j\left[t: z^{(\nu)}(0)+\alpha^{(\nu)}(t-t_0)\in\bigcup_{i=1}^{m(\nu)}Z_j^{(\nu)},\ t>t_0\right]$$

arba

$$t_1=\min_j\left\{t:\ t>t_0,\ \sum_{i=1}^{\|\nu\|}\gamma_{ji}^{(\nu)}\left[z_{ji}^{(\nu)}(0)+\alpha_i^{(\nu)}(t-t_0)\right]+\gamma_{j0}^{(\nu)}=0\right\}.$$

Minimumas yra ieškomas indeksų $j=1,\ldots,m(\nu)$ aibėje.

Jeigu pažymėsime: $\tau_j=-\left(\sum_{i=1}^{\|\nu\|}\gamma_{ji}^{(\nu)}z_i^{(\nu)}(0)+\gamma_{j0}^{(\nu)}\right)\Big/\sum_{i=1}^{\|\nu\|}\gamma_{ji}^{(\nu)}\alpha_i^{(\nu)}$ ir $\tau=\min\{\tau_j:\tau_j>0\}$,

tai laiko momentas, kai pirmą kartą pasiekiamas kontūras, yra: $t_1=t_0+\tau$.

Patekus į kontūrą, nauja pagrindinė būsena ν' apibrėžiama tikimybių skirstiniu P_1, priklausančiu tik nuo būsenos $z(t_1)$. Siekiant apibrėžti papildomų koordinačių vektorių $z^{(\nu')}$, sudaromas pagalbinis vektorius η, kurio skirstinys nepriklauso nuo proceso istorijos, o priklauso tik nuo ν ir $z^{(\nu)}$.

Papildomų koordinačių vektorius apskaičiuojamas: $z^{(\nu')}=\left(z_*^{(\nu)},\eta\right)\times L_z^{(\nu\nu')}$;

Čia $L_z^{(\nu\nu')}$ – matrica, kurios elementai priklauso nuo ν, ν' ir $z_*^{(\nu)}$; $z_*^{(\nu)}=z^{\nu}(\tau)$.

Kai ateina įėjimo signalas $\left(\mu, x^{(\mu)}\right)$, naują pagrindinę būseną ν'' nusako tikimybinis skirstinys P2, kuris priklauso nuo ν, $z_*^{(\nu)}$ ir μ.

Papildomų koordinačių vektoriaus $z^{(\nu'')}$ apibrėžimui sudaromas papildomas vektorius ξ, kurio pasiskirstymas nepriklauso nuo proceso istorijos, o priklauso nuo ν, $z_*^{(\nu)}$ ir μ.

$$z^{(\nu'')} = \left(z_*^{(\nu)}, \xi, x^{(\mu)}\right) \times L_x^{(\nu,\nu'')},$$

čia $L_x^{(\nu,\nu'')}$ – matrica, kurios elementai priklauso nuo ν, $z_*^{(\nu)}$, ν'' ir μ.

Išėjimo signalas formuojamas, kai $z^{(\nu)}$ pasiekia kontūrą arba ateina įėjimo signalas (laiko momentu t').

Pirmu atveju:

$$\lambda = \lambda\left(\nu, \nu', z_*^{(\nu)}\right);$$
$$y^{(\lambda)} = \left(z_*^{(\nu)}, \eta\right) \times L_y^{(\nu\nu')}.$$

Matricos $L_y^{(\nu\nu')}$ elementai priklauso nuo ν, ν' ir $z_*^{(\nu)}$.

Antru atveju:

$$\lambda = \lambda\left(\nu, \nu'', z^{(\nu)}(t'), \mu\right);$$
$$y^{(\lambda)} = \left(z^{(\nu)}(t'), \xi, x^{(\mu)}\right) \times L_y^{(\nu\nu'')}.$$

Matricos $L_y^{(\nu\nu'')}$ elementai priklauso nuo ν, ν'', $z^{(\nu)}(t')$ ir μ.

3.2. Valdančiųjų sekų metodo panaudojimas agregatų funkcionavimo formalizavimui

Atkarpomis-tiesiniai agregatai priklauso automatų modelių klasei. Kaip ir automatas, atkarpomis-tiesinis agregatas aprašomas nurodant būsenų aibę Z, įėjimo signalų aibę Y bei perėjimo operatorių (atvaizdavimą) H ir išėjimo operatorių G. Tačiau agregatas turi nemažai ypatybių, skiriančių šį modelį nuo automatinių modelių.

Agregato funkcionavimas stebimas aibėje laiko momentų $t \in T$, t.y. agregato būsena $z \in Z$ yra laiko funkcija $z(t)$. Atkarpomis-tiesinio agregato būsenos struktūra yra tokia pat, kaip ir atkarpomis-tiesinio Markovo proceso, t.y.: $z(t) = (\nu(t), z_\nu(t))$;

čia $\nu(t)$ – diskreti būsenos komponentė;

$z_\nu(t)$ – tolydi būsenos komponentė.

Bendru atveju,

$$\nu(t) = \{\nu_1(t), \nu_2(t), \ldots, \nu_m(t)\}, \qquad z_\nu(t) = \{z_{\nu 1}(t), z_{\nu 2}(t), \ldots, z_{\nu k}(t)\},$$

čia $\nu_i(t)$ – i-ji diskrečios komponentės koordinatė;

$z_{\nu i}(t)$ – i-ji tolydžios komponentės koordinatė.

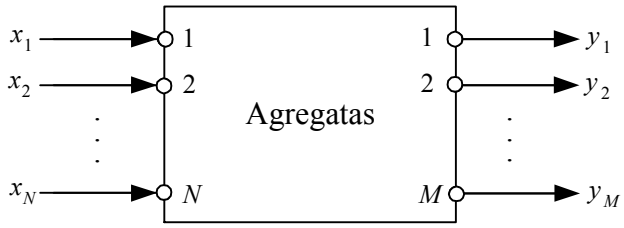
Kai nėra įėjimo signalų, agregato būsenos kinta taip:

$$\nu(t) = \text{const}, \ \frac{dz_\nu(t)}{dt} = -\alpha_\nu;$$

čia $\alpha_\nu = (\alpha_{\nu 1}, \alpha_{\nu 2}, \ldots, \alpha_{\nu k})$ – pastovus vektorius.

Agregato būsena gali pakisti tik dviem atvejais: kai į agregatą yra paduodamas įėjimo signalas arba kai viena iš tolydžios komponentės koordinačių įgyja tam tikrą reikšmę.

Tarkime, turime agregatą, su N įėjimo ir M išėjimo sąveikavimo taškų (ST)



1 pav. Agregato schematinis atvaizdavimas

Įėjimo signalai $x_1, x_2, ..., x_w \in X$ yra paduodami į įėjimo ST. Signalai $x_i \in X_i, \ i = \overline{1, N}$, yra vadinami elementariais signalais, o aibė X_i yra vadinama elementarių signalų aibe. Bendru atveju elementarus signalas yra vektorius, t.y. $x_i = (x_i^1, x_i^2, \ldots, x_i^{r_i})$, be to, šio vektoriaus koordinačių reikšmės priklauso atitinkamai aibei, t.y. $x_i^j \in X_i^j, \ j = \overline{1, r_i}$.

Elementarūs signalai, kurie gali ateiti į i-jį ST, sudaro aibę:

$$X_i = X_i^1 \times X_i^2 \times \ldots \times X_i^{r_i}, \ i = \overline{1, N}.$$

Agregatų įėjimo signalų aibė yra lygi aibių X_i sąjungai, t.y.:

$$X = \bigcup_{i=1}^{N} X_i.$$

Analogiškai yra apibrėžiama išėjimo signalų aibė:

$$Y = \{y_1, y_2, \ldots, y_M\}, \ \ y_1 = \{y_l^1, y_l^2, \ldots, y_l^{s_i}\} \in Y_l;$$

$$y_l^k \in Y_l^k, \ \ l = \overline{1, M}, \ \ k = \overline{1, s_l}.$$

Elementarių signalų aibė išeinanti iš l-jo ST yra lygi:

$$Y_l = Y_l^1 \times Y_l^2 \times \ldots \times Y_l^{s_l}, \ \ l = \overline{1, M}.$$

Agregato išėjimo signalų aibė:

$$Y = \bigcup_{l=1}^{M} Y_l.$$

Agregato funkcionavimas yra nagrinėjamas diskrečiais laiko momentais, kurie priklauso aibei $T = \{t_0, t_1, ..., t_m, ...\}$. Tais laiko momentais gali įvykti vienas ar keli įvykiai,

kurie sukelia agregato būsenos pasikeitimą. Agregato įvykių aibę $E = E' \cup E''$ sudaro du nepersikertantys poaibiai $E' \cap E'' = \varnothing$. Aibę $E' = \{e'_1, e'_2, ..., e'_N\}$ sudaro įvykiai, kurie įvyksta dėl įėjimo signalų atėjimo. Tarp aibių X ir E' elementų yra funkcinis ryšys. Poaibis $E'' = \{e''_1, e''_2, ..., e''_f\}$ yra vadinamas vidinių įvykių poaibiu, čia $e''_i = \{e''_{ij},\ j = 1, 2, 3, \ldots\}$, $i = \overline{1, f}$, yra agregato vidiniai įvykiai, f – operacijų, kurios gali vykti agregate, skaičius. Aibės E'' įvykiai fiksuoja operacijų pabaigą.

Laiko momentų aibė T susideda iš dviejų poaibių:

$$T = T' \cup T'';$$

čia T' – laiko momentų poaibis, kurį sudaro laiko momentai, kada ateina įėjimo signalai;

T'' – vidinių įvykių įvykimo laiko momentai.

Kiekvienam vidiniam įvykiui $e_i \in E''$ yra priskiriama valdanti seka, t.y.:

$$e''_i \mapsto \left\{\xi_j^{(i)}\right\},\ j = \overline{1, \infty};$$

čia $\xi_j^{(i)}$ – operacijos trukmė, kuriai pasibaigus įvyksta vidinis įvykis.

Taip pat nurodoma skaitliukų aibė:

$$\{r(e''_i, t_m)\},\quad i = \overline{1, f};$$

čia $r(e''_i, t_m)$ – e''_i įvykių skaičius, kuris įvyko laiko intervale $[t_0, t_m]$.

Tam, kad būtų galima apibrėžti operacijų pradžios ir pabaigos momentus, yra naudojamos aibės valdančiųjų sumų:

$$\{s(e''_i, t_m)\},\ \{w(e''_i, t_m)\},\ \ i = \overline{1, f};$$

čia $s(e''_i, t_m)$ – operacijos pradžios momentas, kuriai pasibaigus įvyksta e''_i įvykis;

$w(e''_i, t_m)$ – operacijos pabaigos momentas.

Neprioritetinių operacijų atveju, valdanti suma $w(e''_i, t_m)$ yra apibrėžiama taip:

$$w(e''_i, t_m) = \begin{cases} s(e''_i, t_m) + \xi_{r(e''_i, t_m)+1}, & \text{jei laiko momentu } t_m \text{ vyksta operacija,} \\ & \text{kuriai pasibaigus, įvyks įvykis } e''_i, \\ \infty, & \text{priešingu atveju.} \end{cases}$$

Begalybės simbolis (∞) yra naudojamas pažymėti, kad laiko momentas, kai pasibaigs operacija, yra nežinomas.

Pateiktas valdančios sumos apibrėžimas yra naudojamas sudarant imitacinius modelius. Kai agregatinis modelis yra naudojamas sistemų formalizavimui ir sistemos funkcionavimo korektiškumo analizei, valdanti suma apibrėžiama taip:

$$w(e_i'', t_m) = \begin{cases} < \infty, & \text{jei laiko momentu } t_m \text{ vyksta operacija, kuriai} \\ & \text{pasibaigus, įvyks įvykis } e_i^{"}, \\ \infty, & \text{priešingu atveju.} \end{cases}$$

Įvedus valdančias sumas, agregato būsenos tolydi komponentė įgauna pavidalą:

$$z_v(t_m) = \{w(e_1'', t_m), w(e_2'', t_m), \ldots, w(e_f'', t_m)\}.$$

Tolydžios komponentės koordinatės apibrėžia laiko momentus, kada agregate gali įvykti įvykiai. Be to, visada $w(e_i'', t_m) \geq t_m$.

Agregato būsena $z(t_m)$ kinta diskrečiais laiko momentais t_m, m = 1, 2, …, ir išlieka pastovi laiko intervalais $[t_m, t_{m+1})$, m = 0, 1, 2, …, čia t_0 – pradinis sistemos funkcionavimo momentas.

Kai yra žinoma agregato būsena $z(t_m)$, m = 0, 1, 2, …, laiko momentas t_{m+1}, kai įvyksta sekantis įvykis, paskaičiuojamas taip:

$$t_{m+1} = \min_i \{w(e_i'', t_m)\},\ 1 \leq i \leq f.$$

Operatorius H apibrėžia naują agregato būseną:

$$z(t_{m+1}) = H[z(t_m, e_i)],\quad e_i \in E' \cup E''.$$

Operatorius G apibrėžia išėjimo signalus:

$$y = G[z(t_m, e_i)],\quad e_i \in E' \cup E'',\quad y \in Y.$$

3.3. Agregatinių sistemų formalizavimas

Nagrinėsime sistemą, susidedančią iš K agregatų (įskaitant ir išorinės aplinkos agregatus). Tarkime, k-sis agregatas turi N_k įėjimų ir M_k išėjimų. L – bendras skaičius ryšio kanalų, kuriais perduodami signalai tarp agregatų.

Apibrėšime agregatų įėjimų matricą

$$R = (r_{ij}),\ i = \overline{1, L},\ j = \overline{1,2},$$

čia r_{1i} – numeris agregato, priimančio įėjimo signalą iš i - jo ryšio kanalo, r_{1i} =1,2,…,K;

r_{2i} – numeris r_{1i} - jo agregato įėjimo, į kurį ateina signalas i - ju kanalu, $1 \leq r_{2i} \leq Nr$.

Agregatų išėjimų matrica aprašoma taip:

$$H = (h_{ij}),\ i = \overline{1, K},\ j = \overline{1, \max_{1 \leq k \leq K} \{M_k\}},$$

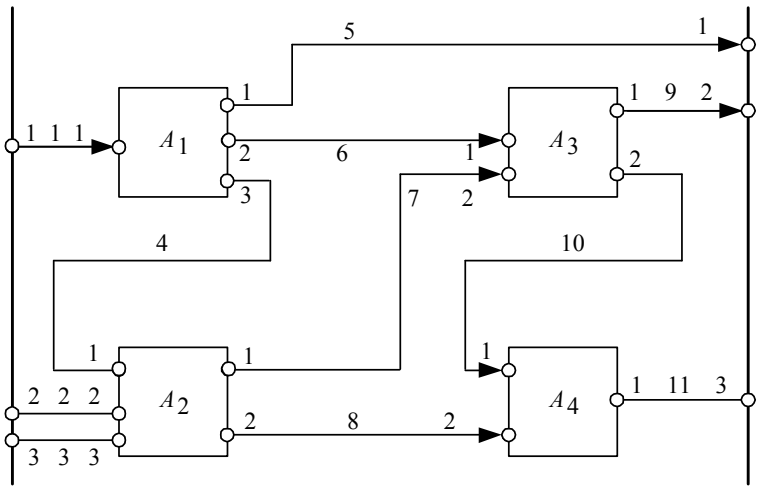
čia h_{ij} – numeris kanalo, į kurį persiduoda i - jo agregato j - jo išėjimo signalas,

$1 \le h_{2i} \le L$;

R ir H matricos vienareikšmiškai apibrėžia, kur yra perduodami agregatų išėjimo signalai. Toliau pateikiamos matricos R ir H skirtos agregatinei sistemai, pateiktai [1pav. Agregato schematinis atvaizdavimas]:

$$R = \begin{pmatrix} 1 & 2 & 2 & 2 & 0 & 3 & 3 & 4 & 0 & 4 & 0 \\ 1 & 2 & 3 & 1 & 1 & 1 & 2 & 2 & 2 & 1 & 3 \end{pmatrix}, \quad H = \begin{pmatrix} 1 & 2 & 3 \\ 5 & 6 & 4 \\ 7 & 8 & 0 \\ 9 & 10 & 0 \\ 11 & 0 & 0 \end{pmatrix}.$$

Kiekvienu kanalu gali būti perduodamas tik vienas signalas, kuris yra išėjimo signalas vienam sistemos agregatui ir įėjimo – kitam. Atsižvelgiant į tai, jog agregatinė sistema yra uždara, t.y. į ją nepatenka jokie išoriniai signalai ir iš jos neperduodami jokie signalai, tai įėjimo signalų atsiradimas agregatuose įmanomas tik vidinių įvykių rezultate, viename iš agregatinės sistemos agregatų.



2pav. Agregatinė sistemos schema

Aibė įvykių, kurie gali įvykti agregatinėje sistemoje:

$$E = \bigcup_{k=1}^{K} E_k'';$$

čia E_k'' – aibė įvykių, vykstančių k-me agregate.

Aibė laiko momentų, kurių metu įvyksta įvykiai iš aibės E

$$T = \bigcup_{k=1}^{K} T_k'';$$

čia T_k'' – aibė laiko momentų, kurių metu vyksta vidiniai įvykiai k-me agregate.

Kitas laiko momentas t_{m+1}, kurio metu įvyksta įvykis iš aibės E, aprašomas taip:

$$t_{m+1} = \min_{1 \le k \le K} \left(\min_{\substack{w_k(e''_r, t_m) \in z_{\upsilon_k}(t_m) \\ e''_r \in E''_k}} w_k\left(e''_r, t_m\right) \right).$$

Išėjimo signalai gali būti formuojami tik įvykių iš aibės *E* įvykimo momentais ir todėl įėjimo signalų atsiradimo momentai apibrėžiami aukščiau pateikta išraiška.

Išėjimo signalų aibė

$$Y = \bigcup_{k=1}^{K} Y_k;$$

čiaY_k –k-jo agregato išėjimo signalų aibė. Naudojant matricą *H* kiekvienam išėjimo signalui iš aibės *Y* paskiriamas kanalas, kuriuo jis turi būti perduodamas. Matrica *R* apibrėžia agregatą ir įėjimo numerį, į kurį paskirtu kanalu perduodamas išėjimo signalas.

Žemiau yra pateikiamas agregatinės sistemos modeliavimo algoritmas:

1. Formuojamos agregatų pradinės būsenos:

$$z_k(t_0) = \left\{ \nu_k(t_0); r_k(t_0); z_\nu^k(t_0) \right\}, \quad k = \overline{1, K}.$$

2. Nustatomas laiko momentas t_{m+1}, kada įvyks sekantis vidinis įvykis, ir numeris k-jo agregato, kuriame tas įvykis įvyks.

3. Apibrėžiama nauja k-jo agregato būsena $z_k(t_{m+1})$ pagal išraišką:

$z_k(t_{m+1}) = H[z_k(t_m), e_i]$, $e_i \in E''$ ir formuojami išėjimo signalai, priklausantys aibei $\widetilde{Y}$;

$y = G[z_k(t_m), e_i]$, $e_i \in E''$, $y \in \widetilde{Y}$.

Visiems likusiems agregatams fiksuojama nauja būsena: $z_i(t_{m+1}) = z_i(t_m)$, $i = \overline{1, K}$, $i \neq K$.

Agregatų, kurių numeris nelygus k, tiek diskrečių, tiek ir tolydžių būsenų dedamosios išlieka nepakitę.

4. Tikrinama aibė $\widetilde{Y}$. Jei aibėje $\widetilde{Y}$ yra bent vienas elementas $y_i \in Y_k$, $1 \le i \le M_k$, $1 \le k \le K$, tuomet, naudojantis matrica *H*, nustatomas kanalo numeris h_{ki}, kuriuo turi būti perduodamas išėjimo signalas. Naudojantis matrica *R* nustatomas agregato numeris $r_{1h_{k_i}}$ ir įėjimo poliaus $r_{2h_{k_i}}$, kur perduodamas įėjimo signalas y_i Pereinama prie 5 žingsnio. Jei $\widetilde{Y} = \varnothing$, tai pereinama prie 2 žingsnio.

5. Apibrėžiama nauja agregato $k = r_{1h_{k_i}}$ būsena, kai ateina įėjimo signalas x_i,

čia $i = r_{2h_{k_i}}$ ir formuojamas išėjimo signalas, kuris fiksuojamas aibėje $\widetilde{Y}$.

Elementas y_i pašalinamas iš aibės $\widetilde{Y}$. Pereinama prie 4 žingsnio.

4. HIBRIDINĖS SISTEMOS PLA SPECIFIKACIJOS SUDARYMAS

4.1. Naftos produktų rezervuarų užpildymo konceptualusis modelis

Naftos produktai pilami iš laivo į rezervuarų sistemą. Išpylus naftos produktus iš laivo, laukiama, kol atplauks kitas laivas. Išpylimas atliekamas 3 etapais:

1. Atplaukė laivas, pradedami pilti naftos produktai. Pylimo greitis šiame etape didinamas tiesiškai nuo 0 iki nustatyto maksimalaus greičio.

2. Pasiekus maksimalų naftos produktų pylimo greitį, toliau pylimas atliekamas šiuo maksimaliu greičiu.

3. Prieš baigiant produktų išpylimą, pylimo greitis mažinamas nuo maksimalaus iki 0 taip pat tiesiškai.

Užsipildžius vieną rezervuarą, iš karto pradedamas pildyti kitas.

Sistemos matematinio modelio sudarymui buvo naudojama rezervuaro, užpildyto naftos produktais, aukščio (h) priklausomybė nuo rezervuaro pagrindo ploto (S), pylimo greičio (ρ) ir pylimo laiko (t):

$$h(t) = \frac{1}{S}\int_0^t \rho(t)dt \text{ , čia } h(0) = 0$$

4.1.1. Agregato *LAIVAS* matematinis aprašymas

1. Įėjimo signalų aibė: Ø.

2. Išėjimo signalų aibė: $Y = \{y_1, y_2, y_3, y_4\}$;

čia y_1 - pradedamas pylimas;

y_2 - pasiektas maksimalus pylimo greitis;

y_3 - pradedamas pylimo stabdymas;

y_4 - pylimas baigtas.

3. Išorinių įvykių aibė: $E' = Ø$.

4. Vidinių įvykių aibė: $E'' = \{e_1'', e_2'', e_3'', e_4''\}$;

čia e_1'' - pradedamas pylimas;

e_2'' - pasiektas maksimalus pylimo greitis;

e_3'' - pylimas pradedamas stabdyti;

e_4'' - pylimas baigtas.

5. Valdymo sekos: $e_1'' \to \{\eta_j\}_{j=0}^{\infty}$;

čia η_j - laiko intervalas tarp dviejų laivų atplaukimo;

$\eta_j > \tau_1 + \tau_2 + \tau_3$.

6. Diskrečioji būsenos dedamoji Ø.

7. Tolydžioji būsenos dedamoji:

$\omega(e_1'', t_m)$ - laiko momentas, kada pradedamas pylimas;

$\omega(e_2'', t_m)$ - laiko momentas, kada pasiektas maksimalus pylimo greitis;

$\omega(e_3'', t_m)$ - laiko momentas, kada pylimas pradedamas stabdyti;

$\omega(e_4'', t_m)$ - laiko momentas, kada pylimas baigiamas.

8. Parametrai:

ρ_0 - maksimalus pylimo greitis;

τ_1 - laiko intervalas nuo pylimo pradžios iki maksimalaus pylimo greičio;

τ_2 - laiko intervalas nuo maksimalaus pylimo greičio iki pylimo stabdymo;

τ_3 - laiko intervalas nuo pylimo stabdymo iki pylimo pabaigos.

9. Pradinė būsena:

$\omega(e_1'', t_0) = \eta_1$.

10. Perėjimo ir išėjimo parametrai:

$H[e_1'']$: / *pradedamas pylimas* /

$\omega(e_1'', t_{m+1}) = t_m + \eta_j$.

$G[e_1'']$: $Y = \{y_1\}$; čia $y_1 = \dfrac{\rho_0}{\tau_1}$.

$H[e_2'']$: / *maksimalus pylimo greitis* /

$\omega(e_2'', t_{m+1}) = t_m + \tau_1$.

$G[e_2'']$: $Y = \{y_2\}$; čia $y_2 = \rho_0$.

$H[e_3'']$: / *stabdomas pylimas* /

$\omega(e_3'', t_{m+1}) = t_m + \tau_2$.

$G[e_3'']$: $Y = \{y_3\}$; čia $y_3 = -\dfrac{\rho_0}{\tau_3}$.

$H[e_4'']$: / *pylimas baigtas* /

$\omega(e_4'', t_{m+1}) = t_m + \tau_3$.

$G[e_4'']$: $Y = \{y_4\}$.

4.1.2. Agregato *REZERVUARAS* matematinis aprašymas

1. Įėjimo signalų aibė: x_1 - atiteka naftos produktai iš laivo.

2. Išėjimo signalų aibė: Ø.

3. Išorinių įvykių aibė: $E^{'} = \{e_1^{'}, e_2^{'}, e_3^{'}, e_4^{'}\}$;

čia $e_1^{'}$ - naftos produktai pradėjo tekėti iš laivo;

$e_2^{'}$ - pasiekiamas maksimalus naftos produktų pylimo greitis;

$e_3^{'}$ - lėtėja naftos produktų pylimo greitis;

$e_4^{'}$ - naftos produktai nebeteka.

4. Vidinių įvykių aibė: $E^{''} = \{e_1^{''}, e_2^{''}, e_3^{''}\}$;

čia $e_1^{''}$ - rezervuaras užsipildė, kol nebuvo pasiektas maksimalus pylimo greitis;

$e_2^{''}$ - rezervuaras užsipildė, kai jau buvo pasiektas maksimalus pylimo greitis;

$e_3^{''}$ - rezervuaras užsipildė, kai naftos produktų pylimas pradedamas stabdyti.

5. Valdymo sekos Ø.

6. Diskrečioji būsenos dedamoji:

$l_1(t_m)$ - laiko momentas, kada pradėta pilti;

$l_2(t_m)$ - laiko momentas, kada pradėta stabdyti;

$r_1(t_m)$ - laiko momentas, kada pradėta pilti į naują rezervuarą, kol nebuvo pasiektas maksimalus pylimo greitis;

$r_2(t_m)$- laiko momentas, kada pradėta pilti į naują rezervuarą, kai jau buvo pasiektas maksimalus pylimo greitis;

$r_3(t_m)$- laiko momentas, kada pradėta pilti į naują rezervuarą, kai naftos produktų pylimas pradedamas stabdyti;

$h(t_m)$ - rezervuarų užpildymo aukštis laiko momentu t_m;

$N(t_m)$ - užpildytų rezervuarų skaičius.

7. Tolydžioji būsenos dedamoji:

$\omega(e_1^{''}, t_m)$ - rezervuaras užpildytas, kol nebuvo pasiektas maksimalus pylimo greitis;

$\omega(e_2^{''}, t_m)$ - rezervuaras užpildytas, kai jau buvo pasiektas maksimalus pylimo greitis;

$\omega(e_3^{''}, t_m)$ - rezervuaras užpildytas, kai naftos produktų pylimas pradedamas stabdyti.

8. Parametrai:

S – rezervuaro plotis;

h_0 - rezervuaro aukštis;

ρ_0 - maksimalus pylimo greitis;

9. Pradinė būsena:

$h(t_0) = 0$;

$N(t_0) = 0$.

10. Perėjimo ir išėjimo parametrai:

$H[e_1^{'}]$**:**

$l_1(t_{m+1}) = t_m$;

$r_1(t_m) = t_m$;

$\omega(e_1^{''}, t_{m+1}) = t_m + \sqrt{\frac{2S(h_0 - h(t_m))}{k_1}}$;

čia $\omega(e_1^{''}, t_{m+1})$ išraiška gaunama iš tokios priklausomybės:

$$\frac{k_1}{S}\int_{t_m}^{\omega(e_1^{''},t_{m+1})}(t - t_m)dt = h_0 - h(t_m);$$

$$\int_{t_m}^{\omega(e_1^{''},t_{m+1})}(t - t_m)dt = \frac{S(h_0 - h(t_m))}{k_1};$$

$$\left(\frac{t^2}{2} - t \cdot t_m\right)\Bigg|_{t_m}^{\omega(e_1^{''},t_{m+1})} = \frac{S(h_0 - h(t_m))}{k_1};$$

$$\frac{\omega^2(e_1^{''}, t_{m+1})}{2} - \omega(e_1^{''}, t_{m+1}) \cdot t_m - \frac{t_m^2}{2} + t_m^2 = \frac{S(h_0 - h(t_m))}{k_1};$$

$$\frac{\omega^2(e_1^{''}, t_{m+1})}{2} - \omega(e_1^{''}, t_{m+1}) \cdot t_m + \frac{t_m^2}{2} = \frac{S(h_0 - h(t_m))}{k_1};$$

$$\omega^2(e_1^{''}, t_{m+1}) - 2 \cdot \omega(e_1^{''}, t_{m+1}) \cdot t_m + t_m^2 = \frac{2S(h_0 - h(t_m))}{k_1};$$

$$(\omega(e_1^{''}, t_{m+1}) - t_m)^2 = \frac{2S(h_0 - h(t_m))}{k_1};$$

$$\omega(e_1^{''}, t_{m+1}) - t_m = \sqrt{\frac{2S(h_0 - h(t_m))}{k_1}};$$

$H[e_1^{''}]$:

$h(t_{m+1}) = 0$;

$$r_1(t_m) = t_m;$$

$$\omega(e_1'', t_{m+1}) = l_1(t_m) + \sqrt{l_1^2(t_m) - 2l_1(t_m)t_m + t_m^2 - 2\frac{Sh_0}{k_1}};$$

čia $\omega(e_1'', t_{m+1})$ išraiška gaunama iš tokios priklausomybės:

$$\frac{k_1}{S}\int_{t_m}^{\omega(e_1'', t_{m+1})}(t - l_1(t_m))dt = h_0;$$

$$\int_{t_m}^{\omega(e_1'', t_{m+1})}(t - l_1(t_m))dt = \frac{S \cdot h_0}{k_1};$$

$$\left(\frac{t^2}{2} - l_1(t_m) \cdot t\right)\Big|_{t_m}^{\omega(e_1'', t_{m+1})} = \frac{S \cdot h_0}{k_1};$$

$$\frac{\omega^2(e_1'', t_{m+1})}{2} - l_1(t_m) \cdot \omega(e_1'', t_{m+1}) - \frac{t_m^2}{2} + l_1(t_m) \cdot t_m = \frac{S \cdot h_0}{k_1};$$

$$\frac{\omega^2(e_1'', t_{m+1})}{2} - l_1(t_m) \cdot \omega(e_1'', t_{m+1}) = \frac{S \cdot h_0}{k_1} + \frac{t_m^2}{2} - l_1(t_m) \cdot t_m;$$

$H[e_2']$**:**

$$r_2(t_m) = t_m;$$

$$\omega(e_1'', t_{m+1}) = \infty;$$

$$\omega(e_2'', t_{m+1}) = \frac{S(h_0 - h(t_m))}{a} + t_m - \frac{k_1}{a}\left(\frac{t_m^2}{2} - l_1(t_m) \cdot t_m - \frac{r_1^2(t_m)}{2} + l_1(t_m) \cdot r_1(t_m)\right);$$

čia $\omega(e_2'', t_{m+1})$ išraiška gaunama iš tokios priklausomybės:

$$\frac{k_1}{S}\int_{r_1(t_m)}^{t_m}(t - l_1(t_m))dt + \frac{1}{S}\int_{t_m}^{\omega(e_2'', t_{m+1})} a dt = h_0 - h(t_m);$$

$$k_1\int_{r_1(t_m)}^{t_m}(t - l_1(t_m))dt + \int_{t_m}^{\omega(e_2'', t_{m+1})} a dt = S(h_0 - h(t_m));$$

$$k_1\left(\left(\frac{t^2}{2} - l_1(t_m) \cdot t\right)\Big|_{r_1(t_m)}^{t_m}\right) + at\Big|_{t_m}^{\omega(e_2'', t_{m+1})} = S(h_0 - h(t_m));$$

$$k_1\left(\frac{t_m^2}{2} - l_1(t_m) \cdot t_m - \frac{r_1^2(t_m)}{2} + l_1(t_m) \cdot r_1(t_m)\right) + a\,\omega(e_2'', t_{m+1}) - at_m = S(h_0 - h(t_m));$$

$$\omega\left(e_2^{''}, t_{m+1}\right) = \frac{S\left(h_0 - h\left(t_m\right)\right)}{a} + t_m - \frac{k_1}{a}\left(\frac{t_m^2}{2} - l_1\left(t_m\right)\cdot t_m - \frac{r_1^{\,2}\left(t_m\right)}{2} + l_1\left(t_m\right)\cdot r_1\left(t_m\right)\right);$$

$H[e_2^{''}]$**:**

$$h\left(t_{m+1}\right) = 0;$$

$$r_1\left(t_m\right) = t_m;$$

$$r_2\left(t_m\right) = t_m;$$

$$\omega\left(e_2^{''}, t_{m+1}\right) = \frac{Sh_0}{a} + t_m;$$

čia $\omega\left(e_2^{''}, t_{m+1}\right)$ išraiška gaunama iš tokios priklausomybės:

$$\frac{a}{S}\int_{t_m}^{\omega\left(e_2^{''}, t_{m+1}\right)} dt = h_0;$$

$$t\Big|_{t_m}^{\omega\left(e_2^{''}, t_{m+1}\right)} = \frac{S\cdot h_0}{a};$$

$$\omega\left(e_2^{''}, t_{m+1}\right) - t_m = \frac{S\cdot h_0}{a};$$

$H[e_3^{'}]$**:**

$$r_3\left(t_m\right) = t_m;$$

$$l_2\left(t_{m+1}\right) = t_m;$$

$$\omega\left(e_2^{''}, t_{m+1}\right) = \infty;$$

$$\omega\left(e_3^{''}, t_m\right) = \frac{-\left(k_2\cdot t_m + a\right) + \sqrt{\left(k_2\cdot t_m + a\right)^2 + 2\cdot k_2\cdot C}}{-k_2};$$

čia $C = \frac{k_1}{2}\cdot\left(r_2^2\left(t_m\right) - r_1^2\left(t_m\right)\right) - k_1\cdot l_1\left(t_m\right)\cdot\left(r_2\left(t_m\right) - r_1\left(t_m\right)\right); -ar_2\left(t_m\right) - S\left(h_0 - h\left(t_m\right)\right);$

$\omega\left(e_3^{''}, t_{m+1}\right)$ išraiška gaunama iš tokios priklausomybės:

$$\frac{k_1}{S}\int_{r_1\left(t_m\right)}^{r_2\left(t_m\right)}\left(t - l_1\left(t_m\right)\right)dt + \frac{1}{S}\int_{r_2\left(t_m\right)}^{t_m} a\,dt + \frac{1}{S}\int_{t_m}^{\omega\left(e_3^{''}, t_{m+1}\right)}\left(a - k_2\left(t - t_m\right)\right)dt = h_0 - h\left(t_m\right);$$

$$k_1\int_{r_1\left(t_m\right)}^{r_2\left(t_m\right)} t\,dt - k_1\int_{r_1\left(t_m\right)}^{r_2\left(t_m\right)} l_1\left(t_m\right)dt + \int_{r_2\left(t_m\right)}^{t_m} a\,dt + \int_{t_m}^{\omega\left(e_3^{''}, t_{m+1}\right)}\left(a - k_2\left(t - t_m\right)\right)dt = S\left(h_0 - h\left(t_m\right)\right);$$

$$\frac{k_1}{2}\cdot t^2\Big|_{r_1(t_m)}^{r_2(t_m)} - k_1\cdot l_1(t_m)\cdot t\Big|_{r_1(t_m)}^{r_2(t_m)} + at\Big|_{r_2(t_m)}^{t_m} + at\Big|_{t_m}^{\omega\left(e_3^{''},t_{m+1}\right)} - k_2\cdot\frac{t^2}{2}\Big|_{t_m}^{\omega\left(e_3^{''},t_{m+1}\right)} +$$

$$+ k_2\cdot t_m\cdot t\Big|_{t_m}^{\omega\left(e_3^{''},t_{m+1}\right)} = S\left(h_0 - h\left(t_m\right)\right);$$

$$\frac{k_1}{2}\cdot\left(r_2^2(t_m) - r_1^2(t_m)\right) - k_1\cdot l_1(t_m)\cdot\left(r_2(t_m) - r_1(t_m)\right) + at_m - ar_2(t_m) + a\,\omega\left(e_3^{''},t_{m+1}\right) - at_m -$$

$$-\frac{k_2}{2}\cdot\left(\omega^2\left(e_3^{''},t_{m+1}\right) - t_m^2\right) + k_2\cdot t_m\cdot\left(\omega\left(e_3^{''},t_{m+1}\right) - t_m\right) = S\left(h_0 - h\left(t_m\right)\right);$$

$$-\frac{k_2}{2}\cdot\omega^2\left(e_3^{''},t_{m+1}\right) + \frac{k_2}{2}\cdot t_m^2 + k_2\cdot t_m\cdot\omega\left(e_3^{''},t_{m+1}\right) - k_2\cdot t_m^2 + a\,\omega\left(e_3^{''},t_{m+1}\right) - ar_2\left(t_m\right) -$$

$$- S\left(h_0 - h\left(t_m\right)\right) + \frac{k_1}{2}\cdot\left(r_2^2\left(t_m\right) - r_1^2\left(t_m\right)\right) - k_1\cdot l_1(t_m)\cdot\left(r_2(t_m) - r_1\left(t_m\right)\right);$$

$$-\frac{k_2}{2}\cdot\omega^2\left(e_3^{''},t_{m+1}\right) + \omega\left(e_3^{''},t_{m+1}\right)\left(k_2\cdot t_m + a\right) - \frac{k_2}{2}\cdot t_m^2 - ar_2\left(t_m\right) - S\left(h_0 - h\left(t_m\right)\right) +$$

$$+\frac{k_1}{2}\cdot\left(r_2^2\left(t_m\right) - r_1\left(t_m\right)\right) - k_1\cdot l_1(t_m)\cdot\left(r_2(t_m) - r_1\left(t_m\right)\right);$$

$$\frac{k_1}{2}\cdot\left(r_2^2\left(t_m\right) - r_1^2\left(t_m\right)\right) - k_1\cdot l_1(t_m)\cdot\left(r_2(t_m) - r_1\left(t_m\right)\right); -ar_2\left(t_m\right) - S\left(h_0 - h\left(t_m\right)\right) = C;$$

$$-\frac{k_2}{2}\cdot\omega^2\left(e_3^{''},t_{m+1}\right) + \left(k_2\cdot t_m + a\right)\cdot\omega\left(e_3^{''},t_{m+1}\right) + C = 0;$$

$$D = \left(k_2\cdot t_m + a\right)^2 + 4\cdot\frac{k_2}{2}\cdot C;$$

$$\omega\left(e_3^{''},t_m\right) = \frac{-\left(k_2\cdot t_m + a\right) \pm \sqrt{\left(k_2\cdot t_m + a\right)^2 + 2\cdot k_2\cdot C}}{-k_2};$$

$H[e_3^{''}]$**:**

$$h\left(t_{m+1}\right) = 0;$$

$$r_1\left(t_m\right) = t_m;$$

$$r_2\left(t_m\right) = t_m;$$

$$r_3\left(t_m\right) = t_m;$$

$$\omega\left(e_3^{''},t_{m+1}\right) = \frac{Sh_0 + a\cdot t_m - k_2\left(t_m - l_2\left(t_m\right)\right)\cdot t_m}{a - k_2\left(t_m - l_2\left(t_m\right)\right)};$$

čia $\omega(e_3^{''}, t_{m+1})$ išraiška gaunama iš tokios priklausomybės:

$$\frac{1}{S}\int_{t_m}^{\omega(e_3^{''},t_{m+1})}(a - k_2(t_m - l_2(t_m)))dt = h_0;$$

$$\int_{t_m}^{\omega(e_3^{''},t_{m+1})}(a - k_2(t_m - l_2(t_m)))dt = Sh_0;$$

$$(a \cdot t - k_2(t_m - l_2(t_m)) \cdot t)\Big|_{t_m}^{\omega(e_3^{''},t_{m+1})} = Sh_0;$$

$$a \cdot \omega(e_3^{''}, t_{m+1}) - k_2(t_m - l_2(t_m)) \cdot \omega(e_3^{''}, t_{m+1}) - a \cdot t_m + k_2(t_m - l_2(t_m)) \cdot t_m = Sh_0;$$

$$\omega(e_3^{''}, t_{m+1})(a - k_2(t_m - l_2(t_m))) = Sh_0 + a \cdot t_m - k_2(t_m - l_2(t_m)) \cdot t_m;$$

$H[e_4^{'}]$:

$$h(t_{m+1}) = \frac{G}{S};$$

čia $G = \frac{k_1}{2} \cdot (r_2^2(t_m) - r_1^2(t_m)) - k_1 \cdot l_1(t_m) \cdot (r_2(t_m) - r_1(t_m)) + a \cdot r_3(t_m) - ar_2(t_m) + a\omega(e_3^{''}, t_{m+1}) -$

$$- a \cdot r_3(t_m) - \frac{k_2}{2} \cdot (\omega^2(e_3^{''}, t_{m+1}) - r_3^2(t_m)) + k_2 \cdot t_m \cdot (\omega(e_3^{''}, t_{m+1}) - r_3(t_m));$$

čia $h(t_{m+1})$ išraiška gaunama iš tokios priklausomybės:

$$\frac{k_1}{S}\int_{r_1(t_m)}^{r_2(t_m)}(t - l_1(t_m))dt + \frac{1}{S}\int_{r_2(t_m)}^{r_3(t_m)}adt + \frac{1}{S}\int_{r_3(t_m)}^{t_m}(a - k_2(t - t_m))dt = h(t_{m+1});$$

$$k_1\int_{r_1(t_m)}^{r_2(t_m)}tdt - k_1\int_{r_1(t_m)}^{r_2(t_m)}l_1(t_m)dt + \int_{r_2(t_m)}^{r_3(t_m)}adt + \int_{r_3(t_m)}^{\omega(e_3^{''},t_{m+1})}(a - k_2(t - t_m))dt = S \cdot h(t_{m+1});$$

$$\frac{k_1}{2} \cdot t^2\Big|_{r_1(t_m)}^{r_2(t_m)} - k_1 \cdot l_1(t_m) \cdot t\Big|_{r_1(t_m)}^{r_2(t_m)} + at\Big|_{r_2(t_m)}^{r_3(t_m)} + at\Big|_{r_3(t_m)}^{\omega(e_3^{''},t_{m+1})} - k_2 \cdot \frac{t^2}{2}\Big|_{r_3(t_m)}^{\omega(e_3^{''},t_{m+1})} + k_2 \cdot t_m \cdot t\Big|_{r_3(t_m)}^{\omega(e_3^{''},t_{m+1})} =$$

$$= S \cdot h(t_{m+1});$$

$$\frac{k_1}{2} \cdot (r_2^2(t_m) - r_1^2(t_m)) - k_1 \cdot l_1(t_m) \cdot (r_2(t_m) - r_1(t_m)) + a \cdot r_3(t_m) - ar_2(t_m) + a\omega(e_3^{''}, t_{m+1}) - a \cdot r_3(t_m) -$$

$$- \frac{k_2}{2} \cdot (\omega^2(e_3^{''}, t_{m+1}) - r_3^2(t_m)) + k_2 \cdot t_m \cdot (\omega(e_3^{''}, t_{m+1}) - r_3(t_m)) = S \cdot h(t_{m+1});$$

4.2. Standartinis agregatinių sistemų imitacinio modeliavimo algoritmas

Sudarytas hibridinės sistemos agregatinis aprašymas gali būti programiškai realizuotas, panaudojant standartinį agregatinių sistemų imitacinio modeliavimo algoritmą, kuris pateiktas 3pav.:

```
types
  record STATE
    k : EXTERNALEVENTQUEUE;
    w : INTERNALEVENTSORTEDQUEUE;
    s : AGGREGATESSTATE;
  end
algorithm Simulation(SimTime:TIME)
var
  state : STATE;
  t : TIME;
begin
  state :=Initialize();
  t := 0;
  while t < SimTime do
    e' := NextEXTERNALEVENT(state);
    while e' ≠ null do
      state := TakeEXTERNALTRANSITION (e', state);
      e' :=NextEXTERNALEVENT (state);
    end
    e'' := NextINTERNALEVENT (state);
    t := GetINTERNALEVENTTIME (e'');
    state := TakeINTERNALTRANSITION (e'', state);
  end
end
```

3pav. Standartinis agregatinių sistemų imitacinio modeliavimo algoritmas

4.3. Standartinio imitacinio modeliavimo algoritmo išplėtimas hibridinėms sistemoms

Darbe pasiūlytas imitacinio modeliavimo algoritmo išplėtimas hibridinėms sistemoms, tam, kad būtų galima paprasčiau sudaryti hibridinės sistemos agregatinę specifikaciją. Šis algoritmas pateiktas 4 pav.

```
types
  record STATE
    k : EXTERNALEVENTQUEUE;
    w : INTERNALEVENTSORTEDQUEUE;
    c : CONTINUOUSEVENTSET;
    s : AGGREGATESSTATE;
  end
algorithm SimulationContinuous(SimTime:TIME)
var
  state : STATE;
  t : TIME;
begin
  state :=Initialize();
  t := 0;
  while t < SimTime do
    e' := NextEXTERNALEVENT(state);
    while e' ≠ null do
      state := TakeEXTERNALTRANSITION (e', state);
      e' :=NextEXTERNALEVENT (state);
    end
    e'' := NextINTERNALEVENT (state);
    t1 := GetINTERNALEVENTTIME (e'');
    e''' := NextCONTINUOUSEVENT (state);
    t2 := GetCONTINUOUSEVENTTIME (e''');
    if (t1 < t2) then
      t := t1;
      state := TakeINTERNALTRANSITION(e'', state);
    else
      t := t2;
    state := TakeCONTINUOUSTRANSITION (e''', state);
    end
  end
end
```

4 pav. Standartinio imitacinio modeliavimo algoritmo išplėtimas hibridinėms sistemoms

Šiame algoritme įvesti nauji specializuoti vidiniai įvykiai e''', kurie pavadinti „Continous“. Šių vidinių įvykių įvykimo laiko momentas $\omega(e_i''', t_{m+1})$ paskaičiuojamas iš tokios lygties :

$$\int_{t_m}^{w(e_i''',t_{m+1})} (ta(e_i''',t_m) + b(e_i''',t_m))dt = c(e_i''',t_m);$$

Mūsų atveju $c(e_i'',t_m)$ nustatomas pradinėje būsenoje ir toliau išlieka toks pat.

čia $a(e_i'',t_m)$, $b(e_i'',t_m)$ ir $c(e_i'',t_m)$ šio specializuoto vidinio įvykio parametrai apibrėžiami modelio specifikacijoje.

Turint šio algoritmo programinę realizaciją, rezervuarų užpildymo pavyzdžio agregatinė specifikacija bus tokia:

1. Įėjimo signalų aibė: x_1 - atiteka naftos produktai iš laivo.

2. Išėjimo signalų aibė: Ø.

3. Išorinių įvykių aibė: $E' = \{e_1', e_2', e_3', e_4'\}$;

čia e_1' - naftos produktai pradėjo tekėti iš laivo;

e_2' - pasiekiamas maksimalus naftos produktų pylimo greitis;

e_3' - lėtėja naftos produktų pylimo greitis;

e_4' - naftos produktai nebeteka.

4. Vidinių įvykių aibė: $E''' = \{e_1'''\}$;

čia e_1''' - rezervuaras užsipildė;

5. Valdymo sekos Ø.

6. Diskrečioji būsenos dedamoji:

$N(t_m)$ - užpildytų rezervuarų skaičius.

7. Tolydžioji būsenos dedamoji:

$\omega(e_1'',t_m)$ - rezervuaras užpildytas

8. Parametrai:

S - rezervuaro plotis;

h_0 - rezervuaro aukštis;

9. Pradinė būsena:

$N(t_0) = 0$.

$c(e_1''',t_0) = h_0 S$;

10. Perėjimo ir išėjimo parametrai:

$H[e_1']$**:**

$a(e_1''',t_{m+1}) = y_1$;

$H[e_2']$**:**

$$a\left(e_1''', t_{m+1}\right) = 0;$$

$$b\left(e_1''', t_{m+1}\right) = y_2;$$

$H[e_3']$:

$$a\left(e_1''', t_{m+1}\right) = y_3;$$

$H[e_4']$:

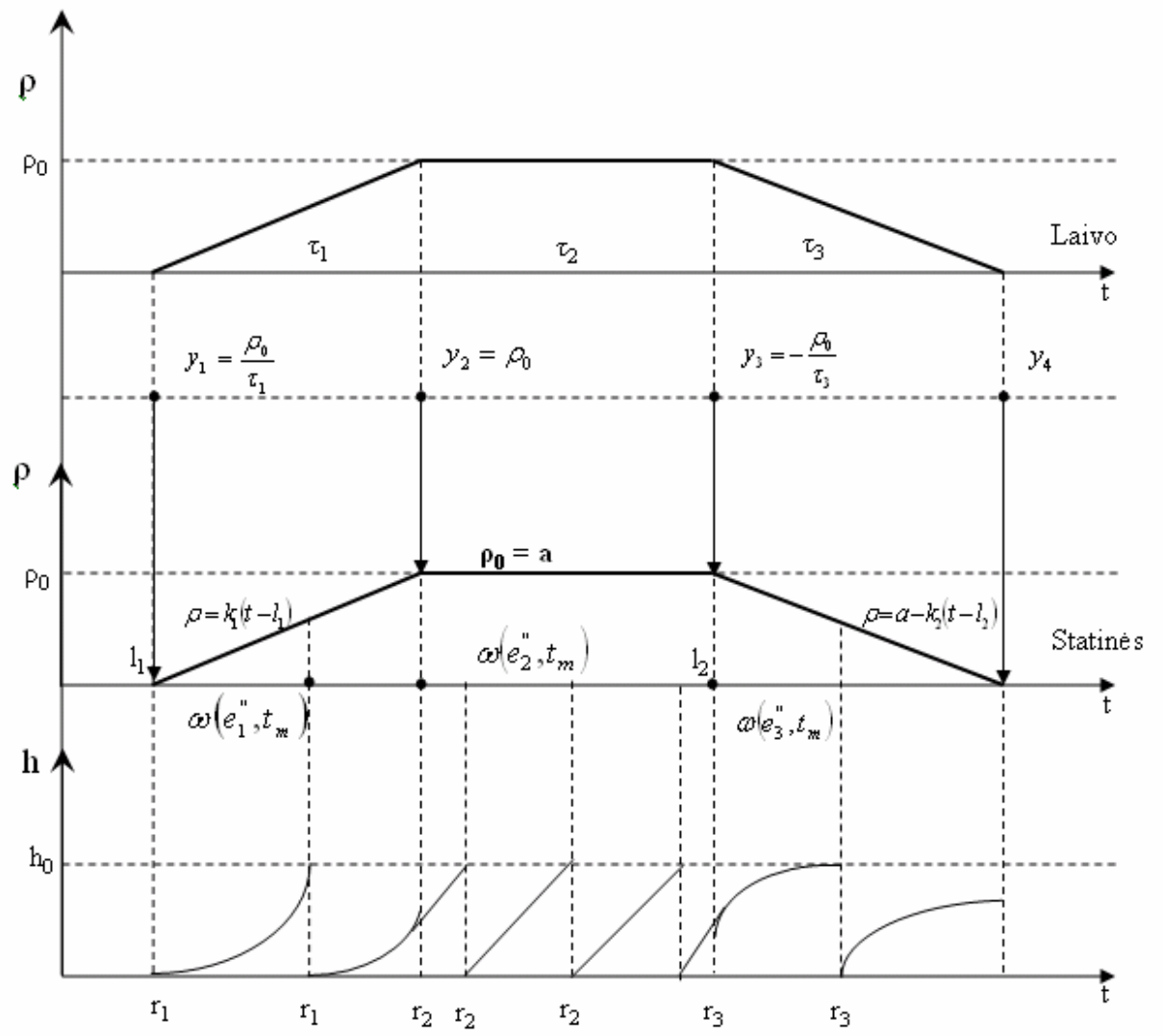
$$a\left(e_1''', t_{m+1}\right) = 0;$$

$$b\left(e_1''', t_{m+1}\right) = 0;$$

$H[e_1''']$:

$$N\left(t_{m+1}\right) = N\left(t_m\right) + 1;$$

Modeliavimo rezultatai pateikti 5 pav.



5pav. Rezervuarų užpildymo modelis

IŠVADOS

1. Buvo apžvelgti diskrečiųjų įvykių formalizmai: diskretaus įvykio elgsenos modelio specifikacija, struktūrinio modelio specifikacija, dinaminių sistemų modeliavimas DEVS, GDEVS ir PLA.

2. Iš atliktos literatūros analizės apie hibridinių sistemų formalizmą galime daryti išvadą, kad hibridinių sistemų imitacinis modeliavimas sudėtingas procesas, nes naudojamos integralinės, diferencialinės arba skirtuminės lygtys.

3. Taip pat buvo išnagrinėtas PLA metodo panaudojimas hibridinių sistemų imitacinio modelio sudarymui. Parodyta, kad sudarant tokios sistemos imitacinio modelio agregatinę specifikaciją reikia išspręsti atitinkamas lygtis, tame tarpe ir apskaičiuojant apibrėžtinius integralus. Atliekant šiuos veiksmus galimos klaidos.

4. Darbe yra pasiūlytas agregatinės sistemos imitacinio modeliavimo algoritmo išplėtimas su specializuotais vidiniais įvykiais, kas leido žymiai supaprastinti imitacinio modelio specifikacijos sudarymą, t.y. nereikia spręsti sudėtingų lygčių

5. Imitacinio modeliavimo algoritmo išplėtimas buvo pritaikytas rezervuarų užpildymo imitacinio modelio agregatinei specifikacijai sudaryti.

LITERATŪRA

1. Anderson, M. *Object-Oriented Modeling and Simulation of Hybrid Systems.* PhD thesis, Department of Automatic Control, Lund Inst. of Technology, Sweden, December 1994.
2. Astron, K.; Elmquist, H.; ir Mattson, S. *Evolution of continuous time modeling and simulation.* In: 12th ESM, Manchester, June 1998.
3. Barton, P.I.; ir Lee, C.K. *Modeling, simulation, sensitivity analysis, and optimization of hybrid systems. ACM Trans. Madel. Comput. Simul., 12(4):256-289,* 2002. ISSN 1049-3301.
4. Behrmann, G.; David, A.; ir Larsen, K.G. A tutorial on UPPAAL. In Bernardo, M. and Corradini, F., editors, *Formal Methods for the Design of Real-Time Systems: 4th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM-RT 2004,* number 3185 in LNCS. Springer, September 2004. 200-236 p.
5. Bengtsson, J., et al. New generation of UPPAAL. In *Int. Workshop on Software Tools for Technology Transfer*, June 1998.
6. Bolognesi, T.; ir Brinksma, H. Introduction to the ISO specification language LOTOS. *Computer Networks,* 14:25-59, 1987.
7. Broenink, J.F.; ir Weustink, P.B.T. A combined system simulator for mechatronics systems. In *Proc. of Modeling and Simulation (ESM'96).* Budapest, Hungary, 1996. 225-229 p. Publishing SCS Europe, Ghent. ISBN 1-56555-097-8.
8. Carmona, J.C.; Giambiasi, N.; ir Naamane, A. *Generalized discrete event abstraction of continuous systems: application to an integrator.* J. Int. Rob. Syst. 41, September 2004. 37–64 p.
9. Cellier, F.E. *Continuous System Modeling*. Springer, 1991. ISBN 0-387-97502-0.
10. Chicoix, C.; Giambiasi, N.; ir Clapier, J. *An accurate time delay model for large digital network simulation, in: Design Automation Conference.* San Francisco, June 1976. [12].
11. Clark, D.; ir Handhelds, D. *Drive mixed signal chip development*. IEEE Comput. 33 (11), 2000. 2–4 p.
12. Damiba, A. *Mode´lisation et Simulation a` e´ve´nements discrets de Bond Graph*. Ph.D Thesis, Universite´ d_Aix-Marseille 3, October 2000.
13. Eertink, H. *Simulation Techniques for Validation of LOTOS Specifications.* PhD thesis, University of Twente, 1994.
14. Escude, B. *Mode´lisation et Simulation a` e´ve´nements discrets de syste`mes hybrids.* Ph.D. Thesis, Univesite´ d_Aix-Marseille, 3, January 2000.
15. Fabian,G.; van Beck, D.; ir Rooda, J. Integration of the discrete and the continuous behavior in the hybrid chi simulator. In *Proc*. 1998 *Eur. Simulation Mu1ticonf*, 1998. 252-257 p.

16. Frey, P.; Carter, H.W.; ir Wilsey, P.A. *Parallel synchronization of continuous time and discrete event simulators.* In: Proceedings of 1997 International Conference on Parallel Processing, August 1997. 227–231 p.
17. Fritzon, P.; ir Bunus, P. Modelica – a general object oriented language for continuous and discrete-event system modeling and simulation. Linkoping, Sweden.
18. Ghosh, S.; Fujimoto, R.M.; ir Schwan, K. *Time warp simulation in time constrained systems*. In:Proceedings of 7th Workshop on Parallel and Distributed Simulation, San Diego, California, USA, 1993. 163–166 p.
19. Ghosh, S.; ir Giambiasi, N. *Breakthrough in modeling and simulation of mixed-signal electronic designs in VHDL.* Modeling and Simulation, May 2001.
20. Ghosh, S.; ir Giambiasi, N. *On the need for consistency between the VHDL language constructs and the underlying hardware design.* ESS 1996, Genoa, Italy, 1996.
21. Giambiasi, N.; Escude, B.; ir Ghosh, S. *GDEVS: A Generalized Discrete Event specification for accurate modelling of dynamic systems*. Trans. S.C.S.I. 17–23, 2000. 120–134 p.
22. Giambiasi, N.; Escude´, B.; ir Ghosh, S. *GDEVS: A Generalized Discrete Event Specification for Accurate Modeling of Dynamic Systems, in: AIS_2000*. Tucson, USA, March 2000.
23. Giambiasi, N.; ir Carmona; J.C. Generalized discrete event abstraction of continuous systems: GDEVS formalism. France, 2005.
24. Giambiasi, N.; Naamane, A.; ir Damiba, A. *Discrete simulation of bond graph, simulation, SCSI.* vol. 77-1,2, July–August 2001. 4–22 p.
25. Giambiasi, N.; Smaili, M.; ir Frydman, C. *Discrete event simulation with fuzzy times, in: European Simulation Symposium.* Turkey, October 1994. [9]
26. Kaynar, D.K., et al. The IOA Simulator. Technical Report MIT-LCS-TR-843, MIT Laboratory for Computer Science, Cambridge, MA, July 2002.
27. Krilavičius, T. *Hybrid Techniques for Hybrid Systems.* The Enschede, Netherlands, 2006. ISBN: 90-365-2397-4.
28. Liu, J.; ir Lee, E.A. A component-based approach to modeling and simulating mixedsignal and hybrid systems. *ACM Trans. Model. Comput. Simul., 12(4):343-368,* 2002. ISSN 1049-3301.
29. Luh, C.J.; ir Zeigler, B. *Abstracting event-based control models for high autonomy systems.* IEEE Trans. Syst. Man Cybernet. 23, 1993. 42–54p.
30. Modelica Association. *Modelica – A Unified Object-Oriented Language for Physical Systems Modeling – Tutorial and Design Rationale Version 1.4.* December 15, 2000.

31. Modelica Association. *Modelica – A Unified Object-Oriented Language for Physical Systems Modeling – Language Specification Version 1.4*. December 15, 2000.

32. Mosterman, P.J.; ir Biswas, G. A Hybrid Modeling and SimuIation Methodology for Dynamic Physical Systems. *SIMULATION*, *78(1):5-17,* 2002. URL http://sim. sagepub.com/cgi/content/abstract/78/l/5.

33. Naamane, A.; Giambiasi, N.; ir Djadja, M. *Discrete event models for mechatronic systems, in: 30th ISATA*. Florence, Italy, June 1997.

34. Otter, M.; Elmquist, H.; ir Mattson, S. *Hybrid modeling in Modelica based on the synchronous data flow Principle.* In: CASCD_99, August 22–26, 1999, Hawaı, USA.

35. Praehofer, H. *System theoretic formalisms for combined discret-continuous system simulation*. Int. J. Gen. Syst. 19 (1991) 219–240.

36. van der Schaft, A.J.; ir Schumacher, J.M. *An Introduction to Hybrid Dynamical Systems,* volume 251 of LNCIS. Springer, London 2000.

37. van Eijk, P. *Software tools for the specification language LOTOS.* PhD thesis, University of Twente, 1988.

38. Wang, Q.; ir Cellier, F. *Time windows: automated abstraction of continuous time models into discrete event models in high autonomy systems*. Int. J. Gen. Syst. 19 (1991) 241–262.

39. Zeigler, B. *DEVS representation of dynamical systems, in: Proceedings of the IEEE*. vol. 77, 1989, pp. 72–80.

40. Zeigler, B. *Multifaceted Modelling and Discrete Event Simulation*. Academic Press, London, 1984.

41. Zeigler, B. *Object-oriented Simulation with Hierarchical, Modular Models*. Academic Press, London, 1990.

42. Zeigler, B.; *Theory of Modeling and Simulation*. John Wiley & Sons, New York, 1976.

43. Zeigler, B.P.; Praenhofer, H.; ir Kim, T.G., *Theory of Modelling and Simulation*. Academic Press, second edition, 2000. ISBN 0-12-778455-1.