VILNIAUS UNIVERSITETAS

Adomas Birštunas

SEKVENCINIAI SKAIČIAVIMAI BDI LOGIKOMS SU EFEKTYVIA CIKLŲ PAIEŠKA

Daktaro disertacijos santrauka
Fiziniai mokslai, Informatika (09P)

Vilnius, 2010

Disertacija rengta 2004-2009 metais Vilniaus universitete.

Mokslinis vadovas :
doc. habil. dr. Regimantas Pliuškevičius (Matematikos ir informatikos institutas, fiziniai mokslai, matematika - 01P)

Disertacija ginama Vilniaus universiteto Informatikos mokslo krypties taryboje :

Pirmininkas :
prof. habil. dr. Feliksas Ivanauskas (Vilniaus universitetas, fiziniai mokslai, informatika - 09P)

Nariai :
prof. dr. Romas Baronas (Vilniaus universitetas, fiziniai mokslai, informatika - 09P)
doc. habil. dr. Regimantas Pliuškevičius (Matematikos ir informatikos institutas, fiziniai mokslai, matematika - 01P)
doc. dr. Jūratė Sakalauskaitė (Matematikos ir informatikos institutas, fiziniai mokslai, matematika - 01P)
doc. dr. Rimantas Vaicekauskas (Vilniaus universitetas, fiziniai mokslai, informatika - 09P)

Oponentai :
prof. habil. dr. Henrikas Pranevičius (Kauno technologijos universitetas, fiziniai mokslai, informatika - 09P)
doc. dr. Stanislovas Leonas Norgėla (Vilniaus universitetas, fiziniai mokslai, informatika - 09P)

Disertacija bus ginama viešame Informatikos mokslo krypties tarybos posėdyje 2010 m. vasario 23 d. 14 val., Vilniaus universiteto Nuotolinių studijų centro salėje.
Adresas: Šaltinių g. 1A, LT-03225, Vilnius, Lietuva.

Disertacijos santrauka išsiuntinėta 2010 m. sausio mėn. 20 d.
Disertaciją galima peržiūrėti Vilniaus universiteto bibliotekoje.

VILNIUS UNIVERSITY

Adomas Birštunas

SEQUENT CALCULI WITH AN EFFICIENT LOOP-CHECK FOR BDI LOGICS

Summary of doctoral dissertation
Physical sciences, Informatics (09P)

Vilnius, 2010

This work was performed in 2004-2009 at Vilnius University, Lithuania.

Research supervisor :
Assoc. Prof. Habil. Dr. Regimantas Pliuškevičius (Institute of Mathematics and Informatics, phisical science, mathematics - 01P)

The thesis is defended at the council of Informatics of Vilnius University :

Chairman :
Prof. Habil. Dr. Feliksas Ivanauskas (Vilnius University, phisical science, informatics - 09P)

Members :
Prof. Dr. Romas Baronas (Vilnius University, phisical science, informatics - 09P)
Assoc. Prof. Habil. Dr. Regimantas Pliuškevičius (Institute of Mathematics and Informatics, phisical science, mathematics - 01P)
Assoc. Prof. Dr. Jūratė Sakalauskaitė (Institute of Mathematics and Informatics, phisical science, mathematics - 01P)
Assoc. Prof. Dr. Rimantas Vaicekauskas (Vilnius University, phisical science, informatics - 09P)

Opponents :
Prof. Habil. Dr. Henrikas Pranevičius (Kaunas University of Technology, phisical science, informatics - 09P)
Assoc. Prof. Dr. Stanislovas Leonas Norgėla (Vilnius University, phisical science, informatics - 09P)

The dissertation will be defended at the public meeting of the council of Informatics of Vilnius University on 23 th of Fabruary, 2010, in Vilnius University Remote Education Study Centre at 2 pm.
Address: Šaltinių g. 1A, LT-03225, Vilnius, Lithuania.

The summary of dissertation was distribued on 20 th of January, 2010.
The dissertation is available at the Library of Vilnius University.

Šioje disertacijoje yra pristatyti nauji sekvenciniai skaičiavimai, naudojantys efektyvią ciklų paiešką, skaidaus laiko logikai ir BDI logikoms, bei naujas beciklis sekvencinis skaičiavimas $KD45$ logikai.

Temos aktualumas

Šioje disertacijoje yra nagrinėjami dirbtinio intelekto uždaviniai susiję su agentų realizavimu. Agentai yra autonomiškos sistemos, kurios veikia kažkurioje aplinkoje ir siekia įvykdyti iš anksto apibrėžtus tikslus. Svarbiausia agento savybė yra sugebėjimas autonomiškai priimti sprendimus pagal informaciją gautą iš aplinkos. Vienas iš būdų realizuoti autonomišką sprendimų priėmimą yra modalinių logikų naudojimas.

Priklausomai nuo pritaikymo srities, naudojamos skirtingos modalinės logikos, naudojančios vieną ar kelis modalumus agento aprašymui. Pati populiariausia modalinė logika skirta agentų realizavimui yra BDI logika ([6]) pristatyta A.S. Rao ir M. Georgeff ([5]). BDI logikoje agentas aprašomas naudojant tris modalumus: įsitikinimus (beliefs), troškimus (desires) ir ketinimus (intentions), dažniausiai kombinuojant su laiko logika. BDI logikos populiarumą lėmė tai, kad daugeliu atveju ji yra tinkama aprašyti dalykinės srities agentus ir yra žinoma jos aksiomatizacija.

Sekvenciniai skaičiavimai yra pripažinti tinkamais automatinei išvedimo paieškai realizuoti. Yra žinomas sekvencinis skaičiavimas BDI logikai ([4]), kuris yra neprieštaringas ir pilnas. Tačiau šis skaičiavimas naudoja neefektyvią (tiesioginę) ciklų paiešką išsprendžiamumui gauti. Ciklų paieškos eliminavimas ir efektyvios ciklų paieškos sukūrimas įvairiems sekvenciniams skaičiavimams šiuo metu yra plačiai nagrinėjama sritis. Šiame darbe yra nagrinėjamas minėtas sekvencinis skaičiavimas BDI logikai ir sukuriamas kitas ekvivalentus skaičiavimas, kuriame dalis ciklų yra visiškai eliminuoti, o kitiems yra taikoma optimizuota ciklų paieška.

Tyrimų objektas

Becikliai sekvenciniai skaičiavimai ir sekvenciniai skaičiavimai su efektyvia ciklų paieška BDI logikoms ir jų fragmentams.

Darbo tikslas

Pagrindinis disertacijos tikslas sukonstruoti efektyvią neprieštaringą ir pilną išvedimo paieškos sistemą skirtą BDI logikoms, kuri būtų paremta sekvenciniu skaičiavimu, kuris nenaudotų ciklų paieškos apskritai arba naudotų tik apribotą ciklų paiešką.

Darbo uždaviniai

Disertacijos tikslui pasiekti reikėjo atlikti šiuos uždavinius:

1. Sukonstruoti pilną ir neprieštaringą beciklį arba naudojantį efektyvią ciklų paiešką sekvencinį skaičiavimą $KD45$ logikai bei įvertinti jo sudėtingumą.

2. Sukonstruoti pilną ir neprieštaringą beciklį arba naudojantį efektyvią ciklų paiešką sekvencinį skaičiavimą skaidaus laiko logikai bei įvertinti jo sudėtingumą.

3. Sukonstruoti pilną ir neprieštaringą beciklį arba naudojantį efektyvią ciklų paiešką sekvencinį skaičiavimą BDI logikai bei įvertinti jo sudėtingumą.

4. Apibendrinti rezultatus ir pateikti pilną ir neprieštaringą sekvencinį skaičiavimą naudojantį efektyvią ciklų paiešką daugiaagentinei BDI logikai.

Tyrimo metodai

Apverčiamos, pusiau apverčiamos išvedimo taisyklės, ir-taisyklių ir ar-taisyklių kombinavimas bei primarinės sekvencijos buvo naudotos skaičiavimų išsprendžiamumui gauti. Modalinių logikų analizė, ciklų paieškos metodas ir grafų teorija buvo naudota specifinėms savybėms, tam tikriems BDI logikų fragmentams, įrodyti. Tam tikro tipo sekvencijų istorijos (žymėti modalumo operatoriai, žymėtos sekvencijos ir modalumo operatoriai su specialiais indeksais) ir įrodytos savybės buvo panaudotos beciklių sekvencinių skaičiavimų ar sekvencinių skaičiavimų su efektyvia ciklų paieška BDI logikos fragmentams sukonstruoti. N. NIDE ir T. Shiro rezultatai ([4]) BDI logikoms ir rezultatai gauti BDI logikų fragmentams buvo naudoti neprieštaringų ir pilnų sistemų BDI logikoms sukūrimui.

Naujumas ir praktinė reikšmė

Darbe pateikti nauji sekvenciniai skaičiavimai: beciklis sekvencinis skaičiavimas $KD45$ logikai, sekvenciniai skaičiavimai skaidaus laiko ir BDI logikoms su efektyvia ciklų paieška.

Kuriant beciklį sekvencinį skaičiavimą $KD45$ logikai buvo panaudotas naujas požiūris į sekvencinio skaičiavimo išvedimo paieškos medžio konstravimą. Prie tam tikrų sąlygų kai kurios išvedimo paieškos medžio sekvencijos, nepriklausomai nuo to ar pačios atskirai yra išvedamos ar ne, gali būti laikomos neišvedamomis šakomis ir toliau nebeišvedinėjamos (pradinės sekvencijos išvedimą lems kitos ar-taisyklių šakos).

Konstruojant sekvencinį skaičiavimą laiko logikai buvo apibrėžtos esminės sekvenijos (jos sutinkamos visose sekvencijose ciklo viduje) leido stipriai apriboti ciklų paiešką. Tokių esminių formulių panaudojimas gali būti pritaikytas konstruojant ciklų paieškos apribojimus kitose logikose.

Daugiaagentinių sistemų realizavimui reikia efektyvių sprendimo paieškos procedūrų BDI logikoms. Viena iš tokių sprendimo paieškos procedūrų (sekvencinio skaičiavimo pagrindu) ir yra pateikta šiame darbe. Praktinį panaudojimą turi ne tik sekvenciniai skaičiavimai BDI logikoms, bet ir skaičiavimai jų fragmentams. Tiek $KD45$ logika, tiek skaidaus laiko logika yra naudojama ir ne BDI logikų kontekste.

Darbe pristatytas beciklis sekvencinis skaičiavimas $KD45$ logikai jau yra panaudotas $KD45$ logikos formulių išvedimo įrankio kūrime. Projektas "The Tableau Workbench (TWB)" ([2]) yra skirtas formulių išvedimo įrankiams skirtingoms modalinėms logikoms realizuoti. Projektui vadovauja žinomas logikas R. Gore (namų puslapis: http://users.rsise.anu.edu.au/ rpg/). Projekte yra realizuotas ir formulių išvedimo įrankis $KD45$ logikai, kuris yra realizuotas šiame darbe pateikto beciklio sekvencinio skaičiavimo $KD45$ logikai pagrindu ([1]).

Ginamieji teiginiai

Pagrindiniai darbo teiginiai, pateikiami gynimui, yra šie:

1. Naujas sukonstruotas sekvencinis skaičiavimas $KD45$ logikai yra beciklis skaičiavimas, kuris ne tik eliminuoja ciklų paiešką, bet ir sumažina pačios išvedimo paieškos sudėtingumą.

2. Naujas sukonstruotas sekvencinis skaičiavimas skaidaus laiko logikai su until operatoriumi naudoja apribojimus, kurie sumažina ciklų paieškos sudėtingumą.

3. Nauji sukonstruoti sekvenciniai skaičiavimai vienaagentinei ir daugiaagentinei BDI logikoms naudoja efektyvią ciklų paiešką visų tipų ciklams aptikti.

Rezultatų aprobavimas

Mokslinio darbo rezultatai paskelbti 7 publikacijose, iš kurių viena yra leidinyje įtrauktame į Mokslinės informacijos instituto (ISI) sąrašą, viena yra leidinyje įtrauktame į Lietuvos mokslo tarybos patvirtintą tarptautinių duomenų bazių sąrašą. Kitos publikacijos yra tarptautiniuose recenzuojamuose leidiniuose.

Tarpiniai rezultatai buvo pristatyti 5 konferencijose, bei Matematikos ir informatikos instituto logikos sekcijos organizuojamuose seminaruose.

Pristatytas sekvencinis skaičiavimas $KD45$ logikai buvo panaudotas "The Tableau Workbench (TWB)" projekte kuriant formulių išvedimo įrankį $KD45$ logikai.

Padėka

Esu dėkingas savo vadovui doc. Regimantui Pliuškevičiui už jo vertingą visokeriopą pagalbą darbo metu.

Džiaugiuosi galėdamas padėkoti doc. Stanislovui Leonui Norgėlai, kuris buvo mano ankstesnių darbų vadovas. Aš dėkingas daugeliui mano kolegų iš Universiteto už jų palaikymą ir padrąsinimą visų studijų metu. Matematikos ir informatikos instituto Logikos sekcijos personalui dėkoju už jų patarimus ir įdomias diskusijas.

Nuoširdžiai dėkoju savo tėvams ir seseriai už jų meilę ir palaikymą. Ypač dėkingas savo žmonai Vilmai ir dukteriai Milgei už jų kantrybę ir meilę.

Disertacijos struktūra

Įvade aprašomas darbo tikslas, temos aktualumas, uždaviniai bei esminiai rezultatai.

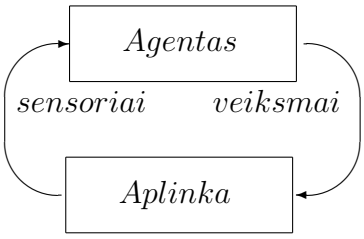
Pirmajame skyriuje aprašomi agentai ir agentinės sistemos. Pateikiama agento veikimo schema. Taip pat pristatoma agentinės sistemos architektūra paremta BDI agentais.

Antrasis skyrius skirtas pateikti esminius apibrėžimus ir sąvokas naudojamas disertacijoje. Taip pat supažindinama su ciklų paieškos metodu, jo privalumais bei trūkumais.

Trečiajame skyriuje yra pristatomas žinomas sekvencinis skaičiavimas $KD45$ logikai. Įrodomi teiginiai leidžiantys apriboti ciklų paiešką ir pateikiamas becikis sekvencinis skaičiavimas $KD45$ logikai, kuris naudoja žymėtus modalumo operatorius. Įrodomi teiginiai įvertinantys skaičiavimo sudėtingumą.

Ketvirtajame skyriuje nagrinėjama skaidaus laiko logika. Apibrėžiamos esminės sekvencijos formulės. Įrodomi teiginiai leidžiantys apriboti naudojamą ciklų paiešką. Pateikiamas sekvencinis skaičiavimas skaidaus laiko logikai, kuris naudoja efektyvią ciklų paiešką realizuotą panaudojant indeksus.

Penktajame skyriuje pristatomi sekvenciniai skaičiavimai vienaagentinei ir daugiaagentinei BDI logikoms. Įrodomi teiginiai leidžiantys apribojimus, gautus BDI logikos fragmentams, pritaikyti ir pačioms BDI logikoms. Pateikiami sekvenciniai skaičiavimai vienaagentinei ir daugiaagentinei BDI logikai, kurie naudoja efektyvią ciklų paiešką.



1 pav.: Paprasta agento veikimo schema.
Agentas per sensorius surenka informaciją iš aplinkos ir atlieka veiksmus, siekdamas pakeisti aplinką.

Priede yra pateiktas ir paaiškintas pseudokodas skirtas beciklio sekvencinio skaičiavimo $KD45$ logikai realizavimui.

1 Skyrius. Agentai ir BDI logika

Šiame skyriuje yra aprašomi agentai ir agentinės sistemos. Pristatoma esminė agentų gyvavimo schema. Pateikiami pagrindiniai būdai kaip gali būti realizuojami agentai. Aptariamas BDI agentų modelis ir realizavimo aspektai.

Nuo seno žmonės stengėsi priversti mašinas dirbti jų labui. Šiais laikais, kai kompiuterizuotos sistemos plačiai naudojamos kasdieniniame gyvenime, tai dar aktualiau. Kai kurios tokių sistemų funkcijos gali būti vykdomos autonomiškai, be žmogaus įsikišimo. Mes norime, kad kompiuterizuotos sistemos dirbtų su kuo mažesniu žmogaus įsikišimu. Tam reikia, kad kompiuterizuotos sistemos ar mašinos galėtų priimti racionalius sprendimus be žmogaus įsikišimo. Tokios sistemos, kurios turi tokį sprendimų priėmimo mechanizmą tampa racionaliais agentais.

Mūsų gyvenime galime susidurti su begale skirtingų agentų - nuo stabdžių antiblokavimo sistemų, lėktuvų autopilotų, programų ieškančių informacijos internete iki mėnuleigių ir kitokių robotų. Agentai labai skirtingi ir jų svarba mūsų gyvenimui labai įvairi. Yra agentų, kurie nėra labai reikalingi, o yra ir tokių be kurių pagalbos kai kurių darbų apskritai vargu ar būtų įmanomi atlikti (pvz., mėnuleigio vairavimas tiesiogiai iš žemės praktiškai neįmanomas).

Agentai visada gyvuoja tam tikroje aplinkoje. Agentai turi savo tikslus ir stengiasi įtakoti aplinką taip, kad jų tikslai būtų pasiekti. Agentas surenka informaciją apie aplinką (per sensorius, daviklius ar specialias programas). Pati aplinka nėra pavaldi agentui ir gali keistis nepriklausomai nuo agento veiksmų. Agentas stengiasi įtakoti aplinką per savo veiksmus. Norint pasiekti savo tikslus agentas pagal surinktą informaciją turi atlikti tokius veiksmus, kurie leistų pasiekti tikslus. Pati paprasčiausia agento veikimo schema pateikta 1 Paveiksle.

Agentas gali būti apibrėžtas kaip esybė (mašina, programa, ...), kuri gali keisti aplinką per savo veiksmus, gali būti įtakojamas aplinkos, ir kuris tenkina tokias savybes kaip autonomiškumas, orientacija į tikslą, reaktyvumas ir socialumas.

Autonomiškumas yra svarbiausia ir sudėtingiausiai realizuojama agento savybė. Autonomiškumui užtikrinti reikia sukurti sprendimų priėmimo mechanizmą. Tai ir yra esminė dirbtinio intelekto dalis. Ši problema gali būti sprendžiama keliais būdais. Pats paprasčiausias būdas yra sprendimų medžių (ar sprendimų lentelių) naudojimas. Toks sprendimas tinkamas, kai agentas yra paprastas ir aplinka yra nuspėjama. Kai aplinka greitai keičiasi ar yra sudėtinga, tai sprendimų priėmimui realizuoti gali būti naudojami dirbtiniai neuroniniai tinklai ar matematinė logika.

Naudojant dirbtinius neuroninius tinklus, agentas visada darys klaidas. Jeigu klaidos tikimybė yra leistinose ribose, tai dirbtiniai neuroniniai tinklai gali būti tuo tinkamu būdu spendimų priėmimo mechanizmui realizuoti. Matematinė logika remiasi racionaliu mąstymų ir gali užtikrinti,

Paprasta BDI agento veikimo schema

1. $B := B_0$; /* B_0 pradiniai įsitikinimai */
2. $I := I_0$; /* I_0 pradiniai ketinimai */
3. while true do
4. get next percept p;
5. $B :=$ brf(B, p);
6. $D :=$ options(B, I);
7. $I :=$ filter(B, D, I);
8. $\pi :=$ plan(B, I);
9. execute(π);
10. end while

2 pav.: Paprasta BDI agento veikimo schema.
Čia p yra informacija surinkta iš aplinkos, B, D, I - agento įsitikinimų, troškimų ir ketinimų aibės, ir π yra tam tikras agento veiksmų planą.

kad tam tikros sąlygos niekada nebūtų pažeistos. Tai leidžia kurti agentus, kurie niekada nedarytų neleistinų klaidų, o tai yra be galo svarbu kai kuriuose taikymuose. Priklausomai nuo taikymo srities, kitokie ar kombinuoti sprendimai irgi gali būti prasmingi ir tinkami naudoti.

Disertacijoje yra nagrinėjami agentai, kurie sprendimo priėmimui naudoja matematinę logiką. Priklausomai nuo taikymo srities agentams realizuoti gali būti naudojamos skirtingos logikos, tokios kaip žinojimo, įsitikinimų, $KARO$, BDI logikos. Iš logikų tinkamų agentams realizuoti BDI išsiskiria savo išraiškingumu ir pritaikomumu skirtingoms sritims.

BDI logika grįsti agentai (ar tiesiog BDI agentai) buvo pristatyti A.S. Rao ir M. Georgeff ([5]). BDI logikoje agentas yra aprašomas per agento veiksmus, veiksmų planus, įsitikinimus (angl. Beliefs), troškimus (angl. Desires) ir ketinimus (angl. Intentions). Agento įsitikinimais yra aprašoma agento turima informacija apie jo aplinką. Agento troškimai nusako pagrindinius agento tikslus. Agento ketinimai nusako tai ką agentas planuoja padaryti. Agento veiksmai parodo, ką gali atlikti agentas, kaip jis gali paveikti aplinką. Veiksmų planai nusako sekas paprastų agento veiksmų, kurie, tikėtina, leis agentui pasiekti, kad aplinkoje įvyktų tam tikri pasikeitimai.

Pati paprasčiausia BDI agento veikimo schema pavaizduota 2 Paveiksle. BDI agentas surenka informaciją iš aplinkos ir pagal gautą informaciją atnaujina savo įsitikinimus. Vėliau agentas atnaujina savo troškimus ir ketinimus, bei pagal juos pasirenka veiksmų planą, kurį toliau ir vykdo.

Kai kurių schemoje pateiktų funkcijų realizavimui logika yra nenaudojama (pvz., funkcijai brf), kai kurioms gali būti naudojama (pvz., funkcijoms filter, plan). Jeigu formali logika yra naudojama kažkuriai funkcijai realizuoti, tai dažniausiai yra naudojamas prieš sąlygų tikrinimas. Jeigu tam tikra prieš sąlyga yra tenkinama, tai yra pasirenkamas atitinkamas veiksmų planas. Jeigu formali logika nėra naudojama nei vienoje iš minėtų funkcijų, tai tokį agentą nereiktų vadinti BDI logika grįstu agentu, nes BDI logika būtų naudojama tik kaip tam tikra notacija esamos agento būsenos aprašymui, o pats sprendimų priėmimas būtų realizuotas nenaudojant logikos.

BDI modelyje agento įsitikinimai, troškimai ir ketinimai yra išreiškiami naudojant BDI logikos modalumo operatorius. Yra sutariama, kad įsitikinimus geriausiai išreiškia $KD45$ modalinės logikos operatorius, o troškimus ir ketinimus KD modalinės logikos operatoriai. Dažniausiai BDI logikos nagrinėjamos su laiko išplėtimu. Priklausomai nuo taikymų, naudojama viena ar kita laiko logika, tačiau viena iš populiariausių yra skaidaus laiko logika. Taigi, BDI logika yra kombinuota logika, kuri yra gauta apjungus $KD45$, KD ir laiko logikas į vieną logiką. BDI logika yra multimodalinė logika, kuri naudoja specialius modalumo operatorius agento įsitikinimams, troškimams, ketinimams vaizduoti, bei laiko logikos operatorius laikui vaizduoti.

2 Skyrius. Sekvencinis skaičiavimas ir ciklų paieška

Šiame skyriuje yra pateikti pagrindiniai apibrėžimai naudojami kituose skyriuose (formulės, modalizuotos formulės, poformulės, išplėstinės poformulės, sekvencijos, išvedimo paieškos medžio, išvedimo medžio, formulių ir sekvencijų ilgio, modalumo gylio, apverčiamų pusiau apverčiamų taisyklių, ir- taisyklių, ar-taisyklių, ciklų ir kiti apibrėžimai). Aprašyta formali BDI logikos semantika. Pristatomi sekvenciniai skaičiavimai. Aptariamas ciklų paieškos metodas išsprendžiamumui gauti.

Disertacijoje yra pristatomi sekvenciniai skaičiavimai teiginių BDI logikoms ir jų fragmentams. Nagrinėjamuose skaičiavimuose formulė gali būti sudaryta naudojantis loginėmis operacijomis ($\neg, \vee, \&$) ir modalumo operatoriais. Modalumo operatorius $\mathbf{B}_i$ (ar vienaagentiniu atveju tiesiog $\mathbf{B}$) yra $KD45$ logikos modalumo operatorius skirtas i-ojo agento įsitikinimams išreikšti. Tuo atveju, kai nagrinėjama $KD45$ logika atskirai, vietoj $\mathbf{B}$ naudojamas įprastas modalumo operatorius $\Box$. Modalumo operatorius $\mathbf{D}_i$ (ar $\mathbf{D}$) yra KD logikos modalumo operatorius skirtas i-ojo agento troškimams išreikšti. Modalumo operatorius $\mathbf{I}_i$ (ar $\mathbf{I}$) yra KD logikos modalumo operatorius skirtas i-ojo agento ketinimams išreikšti.

Naudojamas diskretus laikas, kuris yra išreiškiamas tam tikra laiko momentų seka $t_1, t_2, \ldots$. Jeigu laikas yra tiesinis, tai ateitis yra išreiškiama viena laiko momentų seka. Jeigu laikas yra skaidus, tai tokių skirtingų laiko momentų sekų yra ne viena, ir kiekviena tokia seka išreiškia tam tikrą galimą ateitį. Laiko operatorius $\circ$ (angl. next) išreiškia sekantį laiko momentą. Disertacijoje nagrinėjama BDI logika su skaidaus laiko logika, kuri naudoja until operatorius. Until operatorius trumpai galima nusakyti taip:

- formulė $A(\phi \cup \psi)$ bus teisinga laiko momentu t, jeigu visose ateities laiko momentų sekose $t = t_1, t_2, \ldots$ yra toks laiko momentas t', kad formulė ϕ yra teisinga visais laiko momentais iki t', o formulė ψ yra teisinga laiko momentu t' (visose ateityse ϕ teisinga tol kol ψ tampa teisinga),

- formulė $E(\phi \cup \psi)$ bus teisinga laiko momentu t, jeigu bent vienoje ateities laiko momentų sekoje $t = t_1, t_2, \ldots$ yra toks laiko momentas t', kad formulė ϕ yra teisinga visais laiko momentais iki t', o formulė ψ yra teisinga laiko momentu t' (bent vienoje ateityje ϕ teisinga tol kol ψ tampa teisinga).

Darbe taip pat naudojami žymėti modalumo operatoriai ($\Box^*$, $\mathbf{B}^*$, $\mathbf{B}_i^*$) ir indeksuoti modalumo operatoriai ($A_\alpha^\beta(\phi \cup \psi)$, $E_\alpha^\beta(\phi \cup \psi)$), kurių reikšmė yra tokia pati kaip ir įprastų ir jie yra naudojami skaičiavimų efektyvumui gauti.

Bet kokį sekvencinį skaičiavimą nusako aksiomos ir išvedimo taisyklės. Aksiomos yra tokios sekvencijos, kurios yra laikomos išvedamomis. Paprastai aksioma yra sekvencija pavidalo $\phi, \Gamma \rightarrow \phi, \Delta$, tačiau šiame darbe kai kurie skaičiavimai naudoja ir ciklines aksiomas, kurios bus aptartos vėliau. Išvedimo taisyklės nusako, kaip iš vienų sekvencijų gauti kitas sekvencijas.

Sekvencijos S išvedimo paieškos medžiu vadinsime tokį medį, kurio šaknyje yra sekvencija S ir kurio visuose mazguose yra sekvencijos gautos iš mazgo vaikuose esančių sekvencijų pritaikius kažkurią sekvencinio skaičiavimo taisyklę. Jeigu visuose sekvencijos S išvedimo paieškos medžio lapuose yra aksiomos, tai toks medis vadinamas sekvencijos S išvedimo medžiu. Jei egzistuoja sekvencijos S išvedimo medis, tai sekvencija S yra išvedama.

Pavyzdys 1 Sekvencijos $S = \phi_1 \vee (\phi_2 \& \neg\phi_3) \rightarrow (\phi_2 \& \neg\phi_3) \vee \phi_1$ išvedimo paieškos medis gali atrodyti taip:

$$\dfrac{\dfrac{\overset{\oplus}{\phi_1 \rightarrow \neg\phi_2 \& \neg\phi_3, \phi_1}}{\phi_1 \rightarrow (\neg\phi_2 \& \neg\phi_3) \vee \phi_1}\,(\vee R) \qquad \dfrac{\dfrac{\dfrac{\dfrac{\overset{\oplus}{\phi_2 \rightarrow \phi_3, \phi_2, \phi_1} \quad \dfrac{\overset{\oplus}{\phi_2, \phi_3 \rightarrow \phi_3, \phi_1}}{\phi_2 \rightarrow \phi_3, \neg\phi_3, \phi_1}\,(\neg R)}{\phi_2 \rightarrow \phi_3, \phi_2 \& \neg\phi_3, \phi_1}\,(\& R)}{\phi_2, \neg\phi_3 \rightarrow \phi_2 \& \neg\phi_3, \phi_1}\,(\neg L)}{\phi_2, \neg\phi_3 \rightarrow (\phi_2 \& \neg\phi_3) \vee \phi_1}\,(\vee R)}{\phi_2 \& \neg\phi_3 \rightarrow (\phi_2 \& \neg\phi_3) \vee \phi_1}\,(\& L)}{\phi_1 \vee (\phi_2 \& \neg\phi_3) \rightarrow (\phi_2 \& \neg\phi_3) \vee \phi_1}\,(\vee L)$$

Tai yra išvedimo medis, nes jis yra išvedimo paieškos medis ir visuose lapuose yra aksiomos. Taigi, duotoji sekvencija S yra išvedama.

Sekvencinio skaičiavimo taisyklė gali būti apverčiama, pusiau apverčiama arba neapverčiama. Jeigu taisyklės prielaidos išvedamos, tai išvedama ir taisyklės išvada. Tai galioja visoms sekvencinių skaičiavimų taisyklėms. Taisyklė vadinama apverčiama, jeigu visos jos prielaidos yra išvedamos tada ir tik tada, kai išvedama taisyklės išvada. Taisyklė vadinama pusiau apverčiama, jeigu egzistuoja tokia taisyklės prielaida, kuri yra išvedama, kai išvedama taisyklės išvada. Modalinėse logikose dažnai naudojamos neapverčiamos arba pusiau apverčiamos modalinės taisyklės.

Sekvenciniuose skaičiavimuose taisyklės yra taikomos iš apačios į viršų (pagal sekvenciją ieškomos prielaidos iš kurių galima gauti sekvenciją, tol kol gaunamos aksiomos). Jeigu visos skaičiavimo taisyklės yra apverčiamos, tai konstruojant sekvencijos išvedimo medį niekada nereikia grįžti atgal. Jeigu sekvenciniame skaičiavime yra pusiau apverčiamų taisyklių, tai nepavykus sukonstruoti išvedimo medžio tenka vykdyti paiešką su grįžimu (angl. backtracking) ir pasirinkti kitą taisyklės prielaidą. Jeigu skaičiavime yra neapverčiamų taisyklių, ir jeigu pavyksta sukonstruoti sekvencijos išvedimo medį, tai žinome, kad sekvencija išvedama. Jeigu nepavyksta, tai, bendru atveju, nežinoma ar sekvencija išvedama ar ne.

Šioje disertacijoje, greta įprastų sekvencinio skaičiavimo ir-taisyklių yra naudojamos ar-taisyklės. Ar-taisyklė skiriasi nuo ir-taisyklės tuom, kad išvada yra išvedama tada ir tik tada kai yra išvedama bent viena iš taisyklės prielaidų. Naudojamos ar-taisyklės pakeičia įprastą paiešką su grįžimu ir leidžia atsižvelgti į gretimas alternatyvias šakas vykdant sekvencijos išvedimo paiešką.

Norint turėti skaičiavimą, kurį būtų galima panaudoti agentų ar agentinių sistemų kūrimui būtina, kad jis pasižymėtų tokiomis savybėmis:

- skaičiavimas turi būti pilnas ir neprieštaringas nagrinėjamai logikai,
- išvedimo paieška turi būti baigtinė,
- išvedimo paieška turi būti efektyvi (išvedimą galima būtų atlikti per laiką, kuris yra priimtinas agento realizavimui).

Išvedimo paieška gali būti begalinė, jeigu yra naudojamos neapverčiamos taisyklės arba yra taisyklių, kurių prielaidos yra sudėtingesnės už išvadas. Taigi, reikia konstruoti sekvencinius skaičiavimus, kurie turi tik apverčiamas, arba bent jau pusiau apverčiamas taisykles. Tačiau vien tik apverčiamų ar pusiau apverčiamų taisyklių naudojimo gali nepakakti baigtinumui gauti.

Taisyklių su sudėtingesnėmis prielaidom už išvadas problema gali būti sprendžiama naudojant ciklų paieškos metodą (angl. loop-check). Ciklų paieškos metodas buvo pristatytas M. Fittingo darbe [3]. Paprastai taisyklių prielaidos yra sudarytos tik iš formulių kurios yra tam tikros poformulės formulių esančių taisyklės išvadoje. Kadangi pradinė sekvencija turi baigtinį kiekį skirtingų

poformulių, tai taikant taisykles bet kurioje išvedimo paieškos medžio šakoje arba gausime aksiomą, arba galutinę sekvenciją (tokią, kuriai jokia taisyklė nebegali būti taikoma), arba gausime sekvenciją, kuri sutaps su kažkuria tėvine sekvencija. Tokiu atveju sakome, kad radome ciklą. Dažniausiai ciklo egzistavimas reiškia, kad atitinkama sekvencija yra neišvedama. Kai kuriuose skaičiavimuose yra naudojamos ciklinės aksiomos, kur ciklo egzistavimas reiškia, kad atitinkama sekvencija yra išvedama.

Paprastai sekvencijos išvedimas naudojant ciklų paiešką užima labai daug laiko, nes po kiekvieno taisyklės pritaikymo yra tikrinamos visos tėvinės sekvencijos, kol randamas ciklas arba pasiekiama medžio šaknis. Jeigu ciklas aptiktas, tai toliau taisyklės nebetaikomos, jeigu ciklas nėra aptiktas, tai taisyklės taikomos toliau.

Tiesmukiška ciklų paieška yra labai neefektyvi, todėl reikia konstruoti sekvencinius skaičiavimus, kuriems ciklų paieškos nereiktų visiškai (beciklius sekvencinius skaičiavimus), arba sekvencinius skaičiavimus, kurie naudotų efektyvių ciklų paiešką.

Efektyviai ciklų paieškai gauti gali būti naudojamos sekvencijos su istorijomis. Istorijose yra saugoma papildoma informacija, kuri leidžia patikrinti ar dabartinė sekvencija yra ciklo pabaigos sekvencija netikrinant visų tėvinių sekvencijų. Istorijose saugoma informacija apie ankščiau medyje taikytas taisykles. Istorijos gali būti realizuotos formulių aibėmis, indeksais ar specialiai žymėtomis sekvencijomis, žymėtomis formulėmis ar žymėtais operatoriais.

Indeksai ir žymės yra paprastesnis istorijų tipas ir dėl to efektyvesnis. Šioje disertacijoje kaip tam tikros istorijos yra naudojami specialiai žymėti operatoriai, žymėtos sekvencijos ir indeksuotos poformulės.

3 Skyrius. Beciklis sekvencinis skaičiavimas $KD45$ logikai

Šiame skyriuje nagrinėjama $KD45$ logika. Modalinė $KD45$ logika yra naudojama agento įsitikinimams išreikšti ir yra viena iš pagrindinių BDI logikų sudedamųjų dalių. Skyriuje yra pristatytas naujas pilnas ir neprieštaringas beciklis sekvencinis skaičiavimas $KD45$ logikai ir įvertintas jo sudėtingumas.

Pradžioje yra pateikiama $KD45$ logikos aksiomatizacija remiantis Hilbertiniu skaičiavimu. Pristatomas žinomas pilnas ir neprieštaringas sekvencinis skaičiavimas $KD45_{init}$. Šis skaičiavimas, be loginių taisyklių naudoja silpninimo taisyklė $(Weak)$ ir modalinę taisyklę $(\Box)$:

$$\frac{\Gamma, \Box\Gamma \to \Theta, \Box\Theta, \Box\Delta}{\Box\Gamma \to \Box\Theta, \Box\Delta} \quad (\Box) \quad \Theta \text{ - yra viena formulė arba tuščia.}$$

N. NIDE and T. Shiro darbe [4] parodė, kad sekvencinis skaičiavimas $KD45_{init}$ yra pilnas ir neprieštaringas skaičiavimas $KD45$ logikai. Autoriai pateikia ir baigtinę procedūrą, kuri remiasi sekvenciniu skaičiavimu $KD45_{init}$ ir specialia taktika, ir kuri leidžia nustatyti ar duota sekvencija yra išvedama ar ne. Šioje procedūroje yra naudojama neefektyvi, tiesmukiška ciklų paieška. Taip pat šio skaičiavimo taisyklė $(Weak)$ yra neapverčiama.

Toliau yra pateikiamas tarpinis sekvencinis skaičiavimas $KD45$ logikai $KD45_{wf}$, kuris nebenaudoja neapverčiamos taisyklės $(Weak)$. Šis skaičiavimas vietoj modalinės taisyklės $(\Box)$ ir neapverčiamos taisyklės $(Weak)$ naudoją nauja pusiau apverčiamą taisyklę $(\Box^W)$:

$$\frac{\Gamma, \Box\Gamma \to \phi_1, \Box\phi_1, \ldots, \Box\phi_n \quad || \quad \ldots \quad || \quad \Gamma, \Box\Gamma \to \phi_n, \Box\phi_1, \ldots, \Box\phi_n}{\Sigma, \Box\Gamma \to \Pi, \Box\phi_1, \ldots, \Box\phi_n} \quad (\Box^W)$$

- Atvejis, kai $n = 0$ yra leistinas ir tuomet taisyklė atrodo taip: $\dfrac{\Gamma, \Box\Gamma \to}{\Sigma, \Box\Gamma \to \Pi} \quad (\Box^W)$.
- Σ, Π - baigtinės loginių kintamųjų aibės, $\Sigma \cap \Pi = \emptyset$.

Taikant taisyklę ($\Box$) reikia pasirinkti vieną formulę iš sekvencijos. Naujoji taisyklė ($\Box^W$) yra ar-taisyklė ir joje formulės pasirinkimas atliekamas sukuriant keletą ar-šakų. Taikant šią taisyklę išvada bus išvedama, jeigu bus išvedama bent vien iš prielaidų (atskirtų simboliais $||$).

Įrodoma, kad sekvencinis skaičiavimas $KD45_{wf}$ yra ekvivalentus sekvenciniam skaičiavimui $KD45_{init}$, todėl jis irgi yra pilnas ir neprieštaringas skaičiavimas. Skaičiavime $KD45_{wf}$ nebėra neapverčiamų taisyklių, todėl atliekant išvedimo paiešką nebereikia naudoti jokios taktikos, tačiau vis dar yra reikalinga ciklų paieška.

Vėliau nagrinėjamas skaičiavimas $KD45_{wf}$ atkreipiant dėmesį į ciklų paiešką. Įrodomos lemos, kurios tvirtina, kad jeigu sekvencijos išvedimo paieškos medyje turime ciklą $S \rightsquigarrow S'$, tada tarp S ir S' egzistuoja bent vienas taisyklės ($\Box^W$) taikymas ir visos sekvencijos ciklo viduje turi tokias pačias modalizuotas formules (formules pavidalo $\Box\phi$).

Paskui suformuluojama ir įrodoma lema, kuri teigia:

Tarkime $\overline{S}_1, \overline{S}_2$ yra sekvencijos išvedimo paieškos medyje ir tenkina savybes:

(a) $\overline{S}_1$ yra kažkuri sekvencijos $\overline{S}_2$ tėvinė sekvencija,

(b) sekvencijose $\overline{S}_1$ ir $\overline{S}_2$ yra tokios pačios modalizuotos formulės,

(c) išvedimo paieškos medyje tarp sekvencijų $\overline{S}_1$ ir $\overline{S}_2$ yra lygiai vienas taisyklės ($\Box^W$) taikymas ir šio taisyklės taikymo išvada yra sekvencija $\overline{S}_1$,

(d) Sekvencijos $\underline{S}_{1,0}, \ldots \underline{S}_{1,n}$ yra sekvencijos $\overline{S}_1$ vaikai, ir $\underline{S}_{1,r}$ yra toks iš jų, kad kelias iš $\overline{S}_1$ į $\overline{S}_2$ eina per $\underline{S}_{1,r}$: $\overline{S}_1 \rightarrow \underline{S}_{1,r} \rightarrow \ldots \rightarrow \overline{S}_2$.

Tuomet $\overline{S}_1$ yra išvedama tada ir tik tada jeigu egzistuoja išvedama sekvencija $\underline{S}_{1,i} \neq \underline{S}_{1,r}$.

Pagal šią lemą, jeigu turime sekvenciją $\overline{S}_2$ ir ji tenkina minėtas sąlygas, tai nuo jos visiškai nepriklauso pradinės sekvencijos išvedimas ir ją galime traktuoti kaip neišvedamą. Jeigu $\overline{S}_2$ išvedama, tai mes vistiek galime ją traktuoti kaip neišvedamą, nes tokiu atveju sekvencija $\overline{S}_1$ bus išvedama vien jau dėl kitos išvedamos ar-šakos (sekvencijos $\underline{S}_{1,i}$).

Jeigu sekvencijos išvedimo paieškos medyje gavome sekvenciją S tai toliau jos nebereikia išvedinėti, jeigu:

(a) S yra aksioma (ir ši šaka laikoma išvedama),

(b) S yra galutinė sekvencija (ir ši šaka laikoma neišvedama),

(c) S yra ciklo pabaigos sekvencija (ir ši šaka laikoma neišvedama),

(d) S yra primarinė sekvencija (tokia, kuriai tik taisyklė ($\Box^W$) gali būti taikoma) ir yra tokia tėvinė sekvencija $\overline{S}'$, kuri turi tokias pačias modalizuotas formules (tuomet ši šaka laikoma neišvedama).

Atvejams 3 ir 4 aptikti yra reikalinga ciklų paieška. Tačiau atvejui 4 nustatyti reikia tik lokalaus patikrinimo - reikia patikrinti ar sekvencija, kuriai paskutinį kartą buvo taikyta taisyklė ($\Box^W$) turi tokias pačias modalizuotas formules, o atvejui 3 nustatyti prireiks tiesmukiškos ciklų paieškos.

Vėliau yra įrodoma, kad atvejis 3 yra perteklinis ir nereikalingas. Tam kad atsiminti, kokios modalizuotos formulės buvo gautos paskutinio taisyklės ($\Box^W$) taikymo metu, yra įvedamas žymėtas modalumo operatorius ($\Box^*$). Modalizuotos formulės $\Box\phi$ išorinis modalumo operatorius yra pažymimas žvaigždute, jeigu tokia formulė buvo gauta paskutinio taisyklės ($\Box^W$) taikymo metu. Tokiu būdu, jeigu išvedimo paieškos medyje gavau primarinę sekvenciją ir joje yra tik žymėtos modalizuotos formulės $\Box^*\phi$, tai žinome, kad gavome atvejį 4 ir tokią sekvenciją galime traktuoti kaip neišvedamą.

Remiantis įrodytomis lemomis sukonstruotas beciklis sekvencinis skaičiavimas $KD45_{lcf}$, kuris naudoja žymėtą modalumo operatorių. Skaičiavimas $KD45_{lcf}$ vietoj taisyklės $(\Box^W)$, naudoja pusiau apverčiamą taisyklę $(\Box^{LCF})$:

$$\frac{\Gamma, \Gamma_1, \Box^*\Gamma, \Box^*\Gamma_1 \to \phi_1, \Box^*\phi_1, \ldots, \Box^*\phi_n || \ldots || \Gamma, \Gamma_1, \Box^*\Gamma, \Box^*\Gamma_1 \to \phi_n, \Box^*\phi_1, \ldots, \Box^*\phi_n}{\Sigma, \Box\Gamma, \Box^*\Gamma_1 \to \Pi, \Box\phi_1, \ldots, \Box\phi_k, \Box^*\phi_{k+1}, \ldots, \Box^*\phi_n}$$

Taisyklė $(\Box^{LCF})$ gali būti taikoma tik jeigu $\Box\Gamma \cup \Box\phi_1 \cup \ldots \cup \Box\phi_k \neq \emptyset$.

- Atvejis, kai $n = 0$ yra leistinas ir tuomet taisyklė atrodo taip:

  $\frac{\Gamma, \Gamma_1, \Box^*\Gamma, \Box^*\Gamma_1 \to}{\Sigma, \Box\Gamma, \Box^*\Gamma_1 \to \Pi}$ $(\Box^{LCF})$.

- Σ, Π - baigtinės loginių kintamųjų aibės, $\Sigma \cap \Pi = \emptyset$.

Paprastai pasakius, taisyklė $(\Box^{LCF})$ yra tokia pati kaip ir taisyklė $(\Box^W)$, tik jos negalima taikyti sekvencijai, kurioje visos modalizuotos formulės turi žymėtą modalumo operatorių. Tokios sekvencijos (atitinkančios 4 atvejį) tampa galutinėmis sekvencijomis, nes nėra tokių taisyklių, kurias galima būtų joms pritaikyti, ir todėl jos yra laikomos neišvedamomis.

Įrodoma, kad sekvencinis skaičiavimas $KD45_{lcf}$ yra ekvivalentus sekvenciniam skaičiavimui $KD45_{wf}$, todėl jis irgi yra pilnas ir neprieštaringas skaičiavimas. Skaičiavime $KD45_{lcf}$ nebėra neapverčiamų taisyklių ir jame nebereikia vykdyti ciklų paieškos. Išvedimo paieškos medžio konstravimas visada yra baigtinis, nes po kiekvieno taisyklės $(\Box^{LCF})$ pritaikymo mes gauname bent viena žymėta formule daugiau ir nėra jokios taisyklės, kurios galėtų sumažinti žymėtų formulių kiekį. Taigi, gausime sekvenciją, kurioje visos modalizuotos formulės turės žymėtą modalumo operatorių, o tokiom sekvencijoms taisyklė $(\Box^{LCF})$ negali būti taikoma.

Pavyzdys 2 Pateiksime sekvencijos $S =\!\!\to \neg\Box(\psi \vee \Box\phi) \vee \Box\Box\phi$ išvedimo paieškos medį sukonstruotą becikliame sekvenciniame skaičiavime $KD45_{lcf}$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Box^*(\psi \vee \Box\phi) \overset{\ominus}{\to} \Box^*\phi, \Box^*\Box\phi}{\Box^*(\psi \vee \Box\phi) \to \Box\phi, \Box^*\phi, \Box^*\Box\phi} \;||\; \dfrac{\Box^*(\psi \vee \Box\phi) \overset{\ominus}{\to} \phi, \Box^*\Box\phi, \Box^*\phi}{}\,(\Box^{LCF})}{\Box^*(\psi \vee \Box\phi), \psi \to \Box\phi, \Box^*\Box\phi}}{\uparrow}}{\uparrow}}{\uparrow \quad \Box^*(\psi \vee \Box\phi), \Box\phi \overset{\oplus}{\to} \Box\phi, \Box^*\Box\phi}\,(\vee L)}{\Box^*(\psi \vee \Box\phi), \psi \vee \Box\phi \to \Box\phi, \Box^*\Box\phi}\,(\Box^{LCF})}{\Box(\psi \vee \Box\phi) \to \Box\Box\phi}}{\to \neg\Box(\psi \vee \Box\phi), \Box\Box\phi}\,(\neg R)}{\to \neg\Box(\psi \vee \Box\phi) \vee \Box\Box\phi}\,(\vee R)$$

Sekvencijos S išvedimo paieškos medyje yra dvi ar-šakos (sekvencijos $\Box^*(\psi \vee \Box\phi) \to \Box^*\phi, \Box^*\Box\phi$ ir $\Box^*(\psi \vee \Box\phi) \to \phi, \Box^*\Box\phi, \Box^*\phi$). Abiejose ar-šakose gauname galutines sekvencijas (pažymėtos $\ominus$ simboliu), nes joms jokia sekvencinio skaičiavimo $KD45_{lcf}$ taisyklė negali būti pritaikoma (visos modalizuotos formlės turi žymėta modalumo operatorių $\Box^*$). Simboliu $\oplus$ yra pažymėta aksioma. Šiame pavyzdyje neišvedamos sekvencijos yra ar-šakose. Jeigu viena iš šakų būtų išvedama, tai ir pradinė sekvencija būtų išvedama. Šiuo atveju abi ar-šakos yra neišvedamos, todėl ir pradinė sekvencija S yra neišvedama.

Vėliau yra nagrinėjamas naujojo beciklio sekvencinio skaičiavimo $KD45_{lcf}$ sudėtingumas. Parodoma, kad išvedimo paieškos medžiai sukonstruoti naudojant skaičiavimą $KD45_{lcf}$ yra mažesni arba tokio pat dydžio, kaip ir išvedimo paieškos medžiai gauti naudojant skaičiavimą $KD45_{wf}$ ($KD45_{init}$).

Įrodoma, kad sekvencijos S išvedimo paieškos medžio aukštis negali viršyti $m \cdot k$, kur m yra skirtingų modalizuotų poformulių (turinčių pavidalą $\Box\phi$) skaičius, o k loginių operacijų skaičius pradinėje sekvencijoje. Šis rezultatas leido įrodyti, kad sekvencinio skaičiavimo $KD45_{lcf}$ sudėtingumas laiko atžvilgiu yra EXPTIME.

Pasinaudojus priede pateiktu pseudokodu (skirtu išvedimo paieškai realizuoti) ir formulės skirtingų poformulių indeksavimo lentele yra įrodyta, kad sekvencinio skaičiavimo sudėtingumas yra PSPACE. Svarbu paminėti, kad gautas įverti yra pakankamai mažas - $O(l^3)$, kur l pradinės sekvencijos ilgis.

Šiame skyriuje yra pateikiamas netrivialus išvedimo paieškos medžio konstravimo pavyzdys, kuris iliustruoja kiek sekvencinis skaičiavimas $KD45_{lcf}$ yra efektyvesnis už sekvencinį skaičiavimą $KD45_{wf}$ ($KD45_{init}$).

Skyriaus pabaigoje yra įrodoma lema, kuri teigia, kad egzistuoja tokios sekvencijos, kurios išvedimas sekvenciniame skaičiavime $KD45_{lcf}$ reikalauja $O(l^3)$ atminties. Paprasčiausių tokių sekvencijų išvedimo paieškos medžių fragmentai pademonstruoti pavyzdyje.

4 Skyrius. Sekvencinis skaičiavimas skaidaus laiko logikai su efektyvia ciklų paieška

Šiame skyriuje yra nagrinėjami sekvenciniai skaičiavimai skaidaus laiko logikai. BDI logikoje laikui išreikšti dažniausiai yra naudojama skaidaus laiko logika. Šiame skyriuje yra pateiktas naujas pilnas ir neprieštaringas sekvencinis skaičiavimas skaidaus laiko logikai, kuris naudoja efektyvią ciklų paiešką. Skyriaus pabaigoje yra pateikiamas naudojamos efektyvios ciklų paieškos sudėtingumas.

Skyriaus pradžioje aprašomos populiariausios laiko logikos, bei joms naudojami modalumo operatoriai. Pateikiama skaidaus laiko logikos su until operatoriumi aksiomatizacija remiantis Hilberto tipo skaičiavimu. Pristatomas žinomas sekvencinis skaičiavimas PTL_{init} skaidaus laiko logikai su until operatoriumi.

Sekvencinis skaičiavimas PTL_{init} be loginių taisyklių ir silpninimo taisyklės $(Weak)$ dar naudoja 5 modalines taisykles:

$$\frac{\Gamma \to \Theta}{\circ\Gamma \to \circ\Theta} \quad (\circ) \quad \Theta \text{ - yra viena formulė arba tuščia.}$$

$$\frac{\psi, \Gamma \to \Delta \quad \phi, \circ A(\phi \cup \psi), \Gamma \to \Delta}{A(\phi \cup \psi), \Gamma \to \Delta} \quad \text{(AU-L)}$$

$$\frac{\Gamma \to \psi, \phi, \Delta \quad \Gamma \to \psi, \circ A(\phi \cup \psi), \Delta}{\Gamma \to A(\phi \cup \psi), \Delta} \quad \text{(AU-R)}$$

$$\frac{\psi, \Gamma \to \Delta \quad \phi, \neg \circ \neg E(\phi \cup \psi), \Gamma \to \Delta}{E(\phi \cup \psi), \Gamma \to \Delta} \quad \text{(EU-L)}$$

$$\frac{\Gamma \to \psi, \phi, \Delta \quad \Gamma \to \psi, \neg \circ \neg E(\phi \cup \psi), \Delta}{\Gamma \to E(\phi \cup \psi), \Delta} \quad \text{(EU-R)}$$

Sekvencinis skaičiavimas PTL_{init} greta įprastos aksiomos ($\phi, \Gamma \rightarrow \phi, \Delta$) naudoja ir ciklinę aksiomą. Sekvencija S' yra ciklinė aksioma, jeigu ji yra ciklo $S \rightsquigarrow S'$ pabaigos sekvencija ir ciklo viduje yra toks taisyklės (AU-L) ar (EU-L) taikymas, kad dešinioji prielaida yra cikle $S \rightsquigarrow S'$.

Taisyklių (AU-L), (AU-R), (EU-L), (EU-R) prielaidos yra sudėtingesnės už išvadas, todėl išsprendžiamumui gauti yra reikalinga ciklų paieška. Šiuo atveju yra naudojami dviejų tipų ciklai: paprasti, kurie reiškia, kad sekvencija neišvedama ir ciklinės aksiomos, kurios reiškia, kad sekvencija išvedama. Šioje disertacijoje pateikti apribojimai ciklų paieškai yra vienodai pritaikomi abiejų tipų ciklams aptikti.

N. NIDE and T. Shiro darbe [4] parodė, kad sekvencinis skaičiavimas PTL_{init} yra pilnas ir neprieštaringas skaičiavimas ir pateikė baigtinę procedūrą sekvencijos išvedamumui nustatyti. Ši procedūra naudoja ciklų paiešką ir specialią taktiką neapverčiamos taisyklės $(Weak)$ naudojimui apriboti.

Kadangi sekvencinis skaičiavimas PTL_{init} naudoja neapverčiamą taisyklę, tai disertacijoje pirmiausiai yra pateikiamas tarpinis sekvencinis skaičiavimas PTL_{wf} , kuriame vietoj taisyklės $(\circ)$ ir neapverčiamos taisyklės $(Weak)$ yra naudojama viena pusiau apverčiama taisyklė $(\circ^W)$:

$$\frac{\Gamma \rightarrow \phi_1 \quad || \quad \Gamma \rightarrow \phi_2 \quad || \ldots || \quad \Gamma \rightarrow \phi_n}{\Sigma, \circ\Gamma \rightarrow \Pi, \circ\phi_1, \circ\phi_2, \ldots, \circ\phi_n} \quad (\circ^W)$$

- Atvejis, kai $n = 0$ yra leistinas ir tuomet taisyklė atrodo taip: $\frac{\Gamma \rightarrow}{\Sigma, \circ\Gamma \rightarrow \Pi} \quad (\circ^W)$
- Σ, Π - baigtinės loginių kintamųjų aibės, $\Sigma \cap \Pi = \emptyset$.

Įrodoma, kad sekvencinis skaičiavimas PTL_{wf} yra ekvivalentus sekvenciniam skaičiavimui PTL_{init} , todėl jis irgi yra pilnas ir neprieštaringas skaičiavimas. Skaičiavime PTL_{wf} nebėra neapverčiamų taisyklių, todėl atliekant išvedimo paiešką nebereikia naudoti jokios taktikos, tačiau vis dar yra reikalinga tiesmukiška ciklų paieška.

Toliau yra apibrėžiamas $\circ$-ciklas. Ciklas $S \rightsquigarrow S'$ yra vadinamas $\circ$-ciklu, jeigu sekvencijos S ir S' yra taisyklės $(\circ^W)$ taikymo prielaidos: $\frac{S}{S_1}(\circ^W), \quad \frac{S'}{S_2}(\circ^W)$. Atitinkamai apibrėžiama ir $\circ$-ciklinė aksioma (tai ciklinė aksioma, kuri yra $\circ$-ciklo pabaigos sekvencija.

Vėliau pateikiamas sekvencinis skaičiavimas PTL_{ol} , kuris nuo sekvencinio skaičiavimo PTL_{wf} skiriasi tik tuom, kad vietoj įprastų ciklų yra naudojami tik $\circ$-ciklai (tiek ciklinėms aksiomoms, tiek ciklams naudojamiems sekvencijos neišvedamumui parodyti). Sekvenciniame skaičiavime PTL_{ol} naudojama ciklų paieška yra stipriai apribota, nes jame ciklų paieška yra vykdoma ne visoms sekvencijoms, o tik toms, kurios yra taisyklės $(\circ^W)$ taikymo prielaidos. Be to, ciklų paieškos metu yra tikrinamos tik sekvencijos, kurios irgi yra taisyklės $(\circ^W)$ taikymo prielaidos. Taip yra stipriai sumažinamas kiekis sekvencijų, kurioms atliekama ciklų paieška, ir kiekis pačių tikrinamų sekvencijų.

Disertacijoje įrodoma, kad sekvencinis skaičiavimas PTL_{ol} yra ekvivalentus sekvenciniam skaičiavimui PTL_{wf} , todėl jis irgi yra pilnas ir neprieštaringas skaičiavimas skaidaus laiko logikai. Šiame įrodyme yra naudojama kita (taip pat darbe įrodyta) lema, kuri teigia, kad bet kokiame cikle egzistuoja bent vienas taisyklės $(\circ^W)$ taikymas.

Vėliau yra nagrinėjami $\circ$-ciklai sutinkami skaičiavime PTL_{ol} . Apibrėžiama kas yra sekvencijos atraminė formulė ir pateikiamos kelios lemos, kurios parodo $\circ$-ciklų ir atraminių formulių ryšį.

Æ formulėms ($A(\phi \cup \psi)$ ir $E(\phi \cup \psi)$) apibrėžiama išplėstinė formulės aibė $Ext(F)$:
$Ext(A(\phi \cup \psi)) = Ext(\circ A(\phi \cup \psi)) = \{A(\phi \cup \psi), \circ A(\phi \cup \psi)\}$,
$Ext(E(\phi \cup \psi)) = Ext(\neg E(\phi \cup \psi)) = Ext(\circ\neg E(\phi \cup \psi)) = Ext(\neg \circ \neg E(\phi \cup \psi)) =$
$\{E(\phi \cup \psi), \neg E(\phi \cup \psi), \circ\neg E(\phi \cup \psi), \neg \circ \neg E(\phi \cup \psi)\}$.

Disertacijoje yra naudojama išplėstinė poformulės sąvoka, pagal kurią, jeigu formulės $F_1, F_2 \in Ext(G)$, tai F_1 yra F_2 poformulė ir F_2 yra F_1 poformulė. Visos įprastinės poformulės irgi yra poformulės. F yra tikroji formulės G poformulė ($F \subset_{sf} G$), jei F yra formulės G poformulė ($F \subseteq_{sf} G$) ir formulė F turi mažesnį ilgį už formulę G.

Formulė vadinama atramine sekvencijoje S, jeigu ji nėra jokios kitos formulės, esančios sekvencijoje S, tikroji poformulė.

Vėliau yra įrodomos kelios pagalbinės lemos, kurios reikalingos įrodyti lemą, kuri sako, kad kiekviename $\circ$-cikle $S \rightsquigarrow S'$ visos atraminės formulės yra išplėstinės *AE* formulės (tokios, kurios priklauso $Ext(F)$) ir jos (ar jų išplėstinės poformulės) visos yra sutinkamos visose sekvencijose esančiose $\circ$-cikle.

Remiantis šia lema yra padaroma išvada: jeigu sekvencijos išvedimo paieškos medyje yra toks taisyklės R taikymas, kad T' yra prielaida, or T yra išvada, ir sekvencijoje T egzistuoja atraminė formulė F, bet sekvencijoje T' nėra tokios atraminės formulės $F' \in Ext(F)$, tai sekvencija T nepriklauso jokiam $\circ$-ciklui.

Ši išvada mums sako: jei taikome taisyklę, kuri nuvalo atraminę (išplėstinę *AE*) formulę, tai ciklų paieškos žemiau šio taisyklės taikymo vykdyti nebereikia.

Vėliau yra pristatomos indeksuotos *AE* formulės. Indeksai yra sukurti taip, kad leistų išvedimo metu nustatyti tokius taisyklių taikymus, kurie negali būti jokio $\circ$-ciklo viduje.

Pagal apibrėžtą indeksavimą, kiekviena *AE* formulė turi 2 indeksus - apatinį ir viršutinį. Kiekvienai skirtingai *AE* poformulei yra suteikiamas savas indeksas (natūralusis skaičius). Apatinis indeksas yra aibė, kurios elementai yra natūralieji skaičiai (viršutiniai indeksai). Indeksas i yra įtraukiamas į apatinį poformulės $A^j_U(\phi \cup \psi)$ indeksą U, jeigu $A^j_U(\phi \cup \psi)$ yra poformulė tokios formulės, kuri turi viršutinį indeksą i. Kitaip tariant, *AE* poformulės apatinis indeksas nurodo kokių, sekvencijoje esančių, *AE* formulių poformulė ji yra.

Pavyzdys 3 Tegu sekvencija S yra:
$\circ\phi_1 \& \circ A(\circ\neg E(\phi_2 \cup \circ\phi_3) \cup (\phi_2 \& \circ \neg A(\phi_3 \cup \phi_1))) \rightarrow A(\phi_2 \cup \circ E(\phi_2 \cup \circ\phi_3))$.

Tuomet indeksuota sekvencija bus:
$\circ\phi_1 \& \circ A^1_\emptyset(\circ\neg E^2_{\{1,4\}}(\phi_2 \cup \circ\phi_3) \cup (\phi_2 \& \circ \neg A^3_{\{1\}}(\phi_3 \cup \phi_1))) \rightarrow A^4_\emptyset(\phi_2 \cup \circ E^2_{\{1,4\}}(\phi_2 \cup \circ\phi_3))$.

Vėliau yra pateikiamas sekvencinis skaičiavimas PTL_{rlc} , kuris sekvencinio skaičiavimo PTL_{ol} modalines taisykles pakeičia tokiomis taisyklėmis:

$$\frac{[\Gamma \xrightarrow{\circ} \phi_1]^+ \quad || \quad [\Gamma \xrightarrow{\circ} \phi_2]^+ \quad || \ldots || \quad [\Gamma \xrightarrow{\circ} \phi_n]^+}{\Sigma, \circ\Gamma \rightarrow \Pi, \circ\phi_1, \ldots, \circ\phi_n} \quad (\circ^{RLC})$$

- Atvejis, kai $n = 0$ yra leistinas ir tuomet taisyklė atrodo taip: $\dfrac{[\Gamma \xrightarrow{\circ}]^+}{\Sigma, \circ\Gamma \rightarrow \Pi} \quad (\circ^{RLC})$
- Σ, Π - baigtinės loginių kintamųjų aibės, $\Sigma \cap \Pi = \emptyset$.

$$\frac{[\psi, \Gamma \rightarrow \Delta]^+ \quad \phi, \circ A^j_U(\phi \cup \psi), \Gamma \rightarrow \Delta}{A^j_U(\phi \cup \psi), \Gamma \rightarrow \Delta} \quad (\text{AU-L}^+)$$

$$\frac{[\Gamma \rightarrow \psi, \phi, \Delta]^+ \quad \Gamma \rightarrow \psi, \circ A^j_U(\phi \cup \psi), \Delta}{\Gamma \rightarrow A^j_U(\phi \cup \psi), \Delta} \quad (\text{AU-R}^+)$$

$$\frac{[\psi, \Gamma \rightarrow \Delta]^+ \quad \phi, \neg \circ \neg E^j_U(\phi \cup \psi), \Gamma \rightarrow \Delta}{E^j_U(\phi \cup \psi), \Gamma \rightarrow \Delta} \quad (\text{EU-L}^+)$$

$$\frac{[\Gamma \rightarrow \psi, \phi, \Delta]^+ \quad \Gamma \rightarrow \psi, \neg \circ \neg E^j_U(\phi \cup \psi), \Delta}{\Gamma \rightarrow E^j_U(\phi \cup \psi), \Delta} \quad (\text{EU-R}^+)$$

Jeigu visi i, priklausantys kažkurios poformulės apatiniam indeksui, yra sutinkami kaip kažkurios poformulės viršutinis indeksas, tai sekvencija $[\Gamma \to \Delta]^+ = \Gamma \to \Delta$.

Jeigu yra toks i, priklausantis kažkurios poformulės apatiniam indeksui, bet nėra sutinkamas kaip viršutinis indeksas, tai sekvencija $[\Gamma \to \Delta]^+ = \Gamma' \xrightarrow{+} \Delta'$, kur Γ', Δ' gautos iš Γ, Δ išbraukus visus tokius indeksus i.

Šios taisyklės iš esmės nesiskiria nuo sekvencinio skaičiavimo PTL_{ol} taisyklių, tačiau naujosios taisyklės yra skirtos indeksuotoms formulėms ir, be to, taikymo metu pažymi sekvencijas:

- su $\circ$ - tokias sekvencijas, kurioms gali būti vykdoma ciklų paieška ir kurios pačios gali būti tikrinamos ciklų paieškos metu,
- su $+$ - tokias sekvencijas, kurios negali patekti į jokį $\circ$-ciklą.

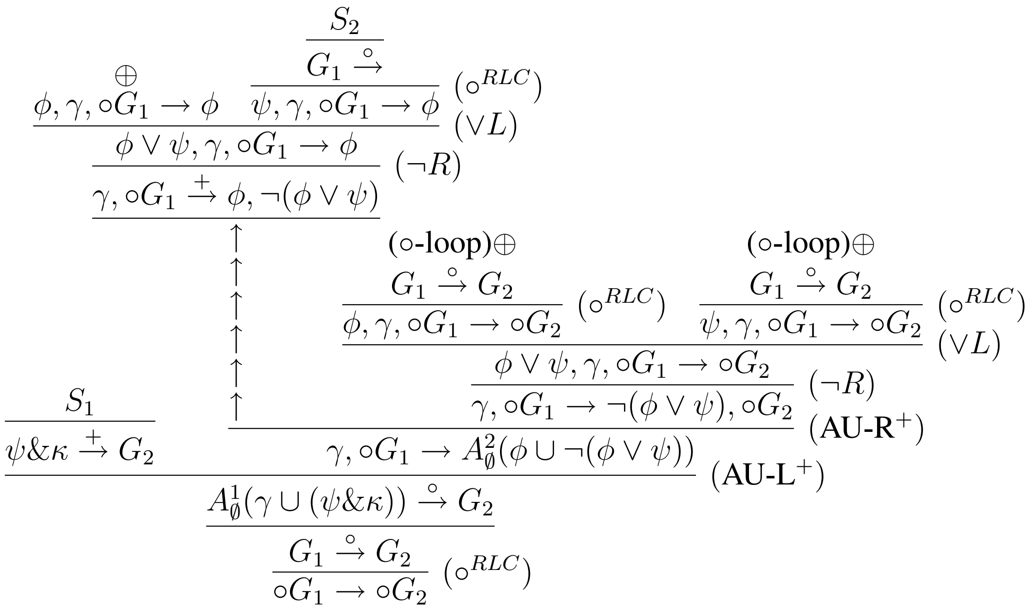
Simboliu $+$ pažymėtomis sekvencijomis mes užtikriname, kad ciklų paieška bus atliekama tik iki kol bus sutikta sekvencija pažymėta su $+$, žemiau esančios sekvencijos nebebus tikrinamos. Tai dar vienas apribojimas, kuris stipriai sumažina ciklų paiešką (ciklų paieškos metu tikrinamų sekvencijų kiekį).

Pavyzdys 4 Turime sekvenciją: $S = \circ A(\gamma \cup (\psi \& \kappa)) \to \circ A(\phi \cup \neg(\phi \vee \psi))$.

Indeksuota sekvencija bus: $S = \circ A^1_\emptyset(\gamma \cup (\psi \& \kappa)) \to \circ A^2_\emptyset(\phi \cup \neg(\phi \vee \psi))$.

Pažymėkime: $G_1 = A^1_\emptyset(\gamma \cup (\psi \& \kappa))$, $G_2 = A^2_\emptyset(\phi \cup \neg(\phi \vee \psi))$.

Išvedimo paieškos medžio fragmentas sekvenciniame skaičiavime PTL_{rlc} yra:



Šiame išvedimo paieškos medžio fragmente yra keletas sekvencijų pažymėtu su $\circ$ ($G_1 \xrightarrow{\circ}$, $G_1 \xrightarrow{\circ} G_2$). Tik tokioms sekvencijoms yra atliekama ciklų paieška. Be to, atliekant ciklų paiešką, išvedimo paieškos medyje aukščiau esančiai sekvencijai, bus tikrinamos tik šios sekvencijos.

Fragmente yra dvi sekvencijos pažymėtos $+$ (S_1 ir $\gamma, \circ G_1 \xrightarrow{+} \phi, \neg(\phi \vee \psi)$). Tai reiškia, kad atliekant ciklų paiešką, sekvencijoms esančioms virš S_1 ir S_2, bus tikrinamos tik $\circ$ pažymėtos sekvencijos ir tik tos, kurios medyje sutinkamos virš $+$ pažymėtų sekvencijų.

Sekvencijos $G_1 \xrightarrow{\circ} G_2$ yra $\circ$-ciklinės aksiomos, nes jos yra $\circ$-ciklų pabaigos sekvencijos ir tuose cikluose yra toks taisyklės (AU-L$^+$) taikymas, kad ciklas eina per dešinę prielaidą.

Disertacijoje įrodoma, kad sekvencinis skaičiavimas PTL_{rlc} yra ekvivalentus sekvenciniam skaičiavimui PTL_{ol} , todėl jis irgi yra pilnas ir neprieštaringas skaičiavimas skaidaus laiko logikai. Gavome pilną ir neprieštaringą sekvencinį skaičiavimą skaidaus laiko logikai su efektyvia ciklų paieška.

Vėliau disertacijoje pateikiamas sudėtingas ir išsamus pavyzdys, demonstruojantis kaip stipriai yra sumažinamas kiekis sekvencijų, kurioms yra atliekama ciklų paieška, ir kiekis sekvencijų tikrinamų ciklų paieškos metu.

Skyriaus pabaigoje yra įrodytos kelios lemos, kurios parodo kiek daugiausiai kartų gali būti pritaikyta taisyklė $(\circ^{RLC})$ vienoje šakoje, koks yra maksimalus išvedimo paieškos medžio aukštis, kiek daugiausiai sekvencijų gali prireikti patikrinti vienos ciklų paieškos metu (sekvenciniame skaičiavime PTL_{rlc}).

Taip pat pateikiamas visų ciklų paieškos tikrinamų sekvencijų kiekio viršutinis įvertis tiek skaičiavimui PTL_{rlc} , tiek ir PTL_{wf} .

5 Skyrius. Sekvenciniai skaičiavimai BDI logikoms su efektyvia ciklų paieška

Šiame skyriuje pristatomi nauji sekvenciniai skaičiavimai vienaagentinei ir daugiaagentinei BDI logikoms. BDI logikoje yra naudojami skirtingi modalumo operatoriai agento įsitikinimams, troškimams ir ketinimams išreikšti, bei modalumo operatoriai laikui išreikšti. BDI logika yra keletos modalinių logikų kombinacija. Šiame skyriuje, sekvenciniai skaičiavimai BDI logikos fragmentams (pateikti ankstesniuose skyriuose) yra apjungiami į naujus sekvencinius skaičiavimus BDI logikoms, kurie naudoja efektyvią ciklų paiešką.

Pradžioje yra pristatoma vienaagentinė BDI logika, kuri naudoją skaidaus laiko logiką su until operatoriumi. BDI logikoje yra naudojami tokie modalumo operatoriai:

- $\mathbf{B}$ - yra $KD45$ logikos modalumo operatorius, kuris skirtas agento įsitikinimams išreikšti,
- $\mathbf{D}$ - yra KD logikos modalumo operatorius, kuris skirtas agento troškimams išreikšti,
- $\mathbf{I}$ - yra KD logikos modalumo operatorius, kuris skirtas agento ketinimams išreikšti,
- $\circ$ - skaidaus laiko logikos operatorius ir $\circ\phi$ reiškia, kad ϕ teisinga kitu laiko momentu,
- A - skaidaus laiko logikos operatorius ir $A(\phi \cup \psi)$ reiškia, kad visose ateityse ϕ teisingas tol, kol ψ tampa teisingu,
- E - skaidaus laiko logikos operatorius ir $E(\phi \cup \psi)$ reiškia, kad bent vienoje ateityje ϕ teisingas tol, kol ψ tampa teisingu.

Vėliau pateikiama pilna BDI logikos aksiomatizacija remiantis Hilbertiniu skaičiavimu ir pristatomas pilnas ir neprieštaringas sekvencinis skaičiavimas BDI_{init} . Sekvencinis skaičiavimas BDI_{init} naudoja jau pristatytas taisykles laiko operatoriams ($(\circ)$, (AU-L), (AU-R), (EU-L), (EU-R)), silpninimo taisyklę $(Weak)$ ir tokias modalines taisykles:

$$\frac{\Gamma, \mathbf{B}\Gamma \to \Theta, \mathbf{B}\Theta, \mathbf{B}\Delta}{\mathbf{B}\Gamma \to \mathbf{B}\Theta, \mathbf{B}\Delta}\ (BEL) \qquad \frac{\Gamma \to \Theta}{\mathbf{I}\Gamma \to \mathbf{I}\Theta}\ (INT) \qquad \frac{\Gamma \to \Theta}{\mathbf{D}\Gamma \to \mathbf{D}\Theta}\ (DES)$$

Θ - yra viena formulė arba tuščia.

Reikia paminėti, kad 3 skyriuje naudotas modalumo operatorius $\Box$ yra tas pats modalumo operatorius $\mathbf{B}$ tik šiame skyriuje jis žymimas kitaip dėl to, kad naudojami modalumo operatoriai

ir agento troškimams, bei įsitikinimams. Taip pat ir naudota taisyklė ($\Box$) yra ta pati taisyklė (BEL).

N. NIDE and T. Shiro darbe [4] parodė, kad sekvencinis skaičiavimas BDI_{init} yra pilnas ir neprieštaringas skaičiavimas BDI logikai. Yra žinoma ir baigtinė procedūra leidžia nustatyti ar duota sekvencija yra išvedama ar ne. Šiai procedūrai yra naudojama taktika, skirta taisyklės ($Weak$) taikymams apriboti. Be to, ji naudoja tiesmukišką ciklų paiešką.

Vėliu yra pristatomas sekvencinis skaičiavimas BDI_{wf} , kuris vietoj neapverčiamos taisyklės ($Weak$) naudoja pusiau apverčiamą taisyklę ($Weak^*$). Taip pat taisykles (BEL), (DES) ir (INT) pakeičia taisyklėmis (BEL^{or}), (INT^{or}), (DES^{or}), kurios naudoja ar-šakas. Naujos taisyklės yra tokios:

$$\frac{\mathbf{B}\Gamma_1 \to \mathbf{B}\Delta_1 \quad || \quad \mathbf{D}\Gamma_2 \to \mathbf{D}\Delta_2 \quad || \quad \mathbf{I}\Gamma_3 \to \mathbf{I}\Delta_3 \quad || \quad \circ\Gamma_4 \to \circ\Delta_4}{\Sigma, \mathbf{B}\Gamma_1, \mathbf{D}\Gamma_2, \mathbf{I}\Gamma_3, \circ\Gamma_4 \to \Pi, \mathbf{B}\Delta_1, \mathbf{D}\Delta_2, \mathbf{I}\Delta_3, \circ\Delta_4} \quad (Weak^*)$$

- Σ, Π - baigtinės loginių kintamųjų aibės, $\Sigma \cap \Pi = \emptyset$.

$$\frac{\Gamma, \mathbf{B}\Gamma \to \phi_1, \mathbf{B}\phi_1, \ldots, \mathbf{B}\phi_n \quad || \quad \ldots \quad || \quad \Gamma, \mathbf{B}\Gamma \to \phi_n, \mathbf{B}\phi_1, \ldots, \mathbf{B}\phi_n}{\mathbf{B}\Gamma \to \mathbf{B}\phi_1, \ldots, \mathbf{B}\phi_n} \quad (BEL^{or})$$

$$\frac{\Gamma \to \phi_1 \quad || \quad \Gamma \to \phi_2 \quad || \quad \ldots \quad || \quad \Gamma \to \phi_n}{\mathbf{I}\Gamma \to \mathbf{I}\phi_1, \ldots, \mathbf{I}\phi_n} \quad (INT^{or})$$

$$\frac{\Gamma \to \phi_1 \quad || \quad \Gamma \to \phi_2 \quad || \quad \ldots \quad || \quad \Gamma \to \phi_n}{\mathbf{D}\Gamma \to \mathbf{D}\phi_1, \ldots, \mathbf{D}\phi_n} \quad (DES^{or})$$

- Atvejis, kai $n = 0$ yra leistinas ir tuomet taisyklės atrodo taip:

$$\frac{\Gamma, \mathbf{B}\Gamma \to}{\mathbf{B}\Gamma \to} \quad (BEL^{or}), \quad \frac{\Gamma \to}{\mathbf{I}\Gamma \to} \quad (INT^{or}) \quad \text{ir} \quad \frac{\Gamma \to}{\mathbf{D}\Gamma \to} \quad (DES^{or})$$

Įrodoma, kad sekvencinis skaičiavimas BDI_{wf} yra ekvivalentus sekvenciniam skaičiavimui BDI_{init} , todėl jis irgi yra pilnas ir neprieštaringas skaičiavimas. Skaičiavime BDI_{wf} yra tik apverčiamos arba pusiau apverčiamos taisyklės ir yra naudojama tiesmukiška ciklų paieška.

BDI logika savyje apima modalumo operatorius iš skirtingų modalinių logikų ($KD45$, KD ir skaidaus laiko logikų). Modalumo logikai KD yra žinomas beciklis sekvencinis skaičiavimas. Šioje disertacijoje yra pateiktas beciklis sekvencinis skaičiavimas modalinei logikai $KD45$ ir sekvencinis skaičiavimas skaidaus laiko logikai, kuris naudoja efektyvią ciklų paiešką.

Darbe yra pateikta N. NIDE and T. Shiro įrodyta lema, kuri teigia, kad bet koks ciklas sekvenciniame skaičiavime BDI_{init} (BDI_{wf}) yra arba Belief tipo, arba Until tipo ciklas, bet ne abiejų tipų kartu. Ciklą vadiname Belief tipo ciklu, jeigu jame yra taikoma taisyklė (BEL) (ar (BEL^*), (BEL_i^{or}), (BEL_i^*)). Ciklą vadiname Until tipo ciklu, jeigu jame yra taikom taisyklė ($\circ^{or}$) (ar ($\circ^+$)).

Pateikta lema teigia, kad bet kokiame cikle yra taikoma arba taisyklė (BEL), arba taisyklė ($\circ^{or}$) ir tikrai nėra taikomos taisyklės (DES) ir (INT). Taigi, visi apribojimai ir įrodymai (pateikti 3 skyriuje) skirti Belief tipo ciklams, yra teisingi ir BDI logikai. Taip pat visi apribojimai ir įrodymai (pateikti 4 skyriuje) skirti Until tipo ciklams, yra teisingi ir BDI logikai.

Pasinaudojus gautais rezultatais Belief ir Until tipo ciklams yra pateikiamas sekvencinis skaičiavimas BDI_{elc} , kuris eliminuoja Belief tipo ciklus, o Until tipo ciklams naudoja efektyvią ciklų paiešką.

Sekvencinis skaičiavimas naudoja logines taisykles, šiame darbe ankščiau apibrėžtas taisykles (DES^{or}), (INT^{or}), ($\circ^+$), (AU-L$^+$), (AU-R$^+$), (EU-L$^+$), (EU-R$^+$) ir dvi modifikuotas taisykles ($Weak^{*+}$), (BEL^*):

$$\frac{\mathbf{B}\Gamma_1, \mathbf{B}^*\Gamma_1' \xrightarrow{+} \mathbf{B}\Delta_1, \mathbf{B}^*\Delta_1' \quad || \quad \mathbf{D}\Gamma_2 \xrightarrow{+} \mathbf{D}\Delta_2 \quad || \quad \mathbf{I}\Gamma_3 \xrightarrow{+} \mathbf{I}\Delta_3 \quad || \quad \circ\Gamma_4 \to \circ\Delta_4}{\Sigma, \mathbf{B}\Gamma_1, \mathbf{B}^*\Gamma_1', \mathbf{D}\Gamma_2, \mathbf{I}\Gamma_3, \circ\Gamma_4 \to \Pi, \mathbf{B}\Delta_1, \mathbf{B}^*\Delta_1', \mathbf{D}\Delta_2, \mathbf{I}\Delta_3, \circ\Delta_4}$$

- Σ, Π - baigtinės loginių kintamųjų aibės, $\Sigma \cap \Pi = \emptyset$.

$$\frac{\Gamma, \Gamma_1, \mathbf{B}^*\Gamma, \mathbf{B}^*\Gamma_1 \to \phi_1, \mathbf{B}^*\phi_1, \ldots, \Box^*\phi_n || \ldots || \Gamma, \Gamma_1, \mathbf{B}^*\Gamma, \mathbf{B}^*\Gamma_1 \to \phi_n, \mathbf{B}^*\phi_1, \ldots, \mathbf{B}^*\phi_n}{\mathbf{B}\Gamma, \mathbf{B}^*\Gamma_1 \to \mathbf{B}\phi_1, \ldots, \mathbf{B}\phi_k, \mathbf{B}^*\phi_{k+1}, \ldots, \mathbf{B}^*\phi_n}$$

Taisyklė (BEL^*) gali būti taikoma tik jeigu $\mathbf{B}\Gamma \cup \mathbf{B}\phi_1 \cup \ldots \cup \mathbf{B}\phi_k \neq \emptyset$.

- Atvejis, kai $n = 0$ yra leistinas ir tuomet taisyklė atrodo taip:

$$\frac{\Gamma, \Gamma_1, \mathbf{B}^*\Gamma, \mathbf{B}^*\Gamma_1 \to}{\mathbf{B}\Gamma, \mathbf{B}^*\Gamma_1 \to} \quad (BEL^*).$$

Įrodoma, kad sekvencinis skaičiavimas BDI_{wf} yra ekvivalentus sekvenciniam skaičiavimui BDI_{wf} , todėl jis irgi yra pilnas ir neprieštaringas skaičiavimas. Skaičiavimas BDI_{elc} naudoja ciklų paiešką tik Until tipo ciklams ir naudojama ciklų paieška yra stipriai apribota.

Tokiu būdu gavome pilną ir neprieštaringą sekvencinį skaičiavimą vienaagentinei BDI logikai, kuris naudoja tik efektyvią ciklų paiešką.

Vėliau yra nagrinėjama daugiaagentinė BDI logika. Daugiaagentinė BDI logika naudoja skirtingus operatorius, skirtingų agentų įsitikinimams, troškimam ir ketinimams išreikšti. i-ojo agento įsitikinimai išreiškiami $KD45$ logikos operatoriumi $\mathbf{B}_i$, troškimai ir ketinimai išreiškiami KD logikos operatoriais $\mathbf{D}_i$ ir $\mathbf{I}_i$. Kitaip pasakius, kiekvienam agentui išreikšti naudojamas atskiras modalumo operatorių trejetas. Yra sutariama, kad i-ojo ir j-ojo agentų modalumo operatoriai neturi tarpusavio priklausomybių.

Daugiaagentinei BDI logikai yra pateikiama aksiomatizacija remiantis Hilberto tipo skaičiavimu. Pristatomas sekvencinis skaičiavimas BDI^n_{init} daugiaagentinei BDI logikai, kuris yra pilnas ir neprieštaringas, bet naudoja neefektyvią ciklų paiešką.

Analogiškai kaip ir vienaagentiniu atveju, pateikiamas sekvencinis skaičiavimas BDI^n_{wf} , kuris yra ekvivalentus skaičiavimui BDI^n_{init} ir kuriame yra naudojamos tik apverčiamos arba pusiau apverčiamos taisyklės.

Įrodoma lema, kuri parodo, kad bet koks Belief tipo ciklas yra vieno agento Belief tipo ciklas (ciklo viduje taisyklės yra taikomos tik pagal vieno agento $\mathbf{B}_i$ operatorius). Taigi, skirtingų agentų Belief tipo ciklai nepersidengia ir visiems jiems galima taikyti tokius pačius apribojimus, kaip ir vienaagentiniu atveju.

Remiantis šiuo faktu yra sukonstruotas sekvencinis skaičiavimas BDI^n_{elc} , kuris naudoja logines taisykles, ankščiau pristatytas taisykles $(\circ^+)$, (AU-L$^+$), (AU-R$^+$), (EU-L$^+$), (EU-R$^+$), bei taisykles $(Weak_i^{*+})$, (BEL_i^*), (DES_i^{or}), (INT_i^{or}):

$$\frac{S_1^1 || \ldots || S_1^n \quad || \quad S_2^1 || \ldots || S_2^n \quad || \quad S_3^1 || \ldots || S_3^n \quad || \quad \circ\Gamma_4 \to \circ\Delta_4}{\Sigma, \mathbf{B}_{1\ldots n}\Gamma_1, \mathbf{B}^*_{1\ldots n}\Gamma_1', \mathbf{D}_{1\ldots n}\Gamma_2, \mathbf{I}_{1\ldots n}\Gamma_3, \circ\Gamma^4 \to \Pi, \mathbf{B}_{1\ldots n}\Delta_1, \mathbf{B}^*_{1\ldots n}\Delta_1', \mathbf{D}_{1\ldots n}\Delta_2, \mathbf{I}_{1\ldots n}\Delta_3, \circ\Delta_4}$$

- $S_1^i = \mathbf{B}_i\Gamma_1^i, \mathbf{B}_i^*\Gamma_1'^i \xrightarrow{+} \mathbf{B}_i\Delta_1^i, \mathbf{B}_i^*\Delta_1'^i$; $\quad S_2^i = \mathbf{D}_i\Gamma_2^i \xrightarrow{+} \mathbf{D}_i\Delta_2^i$; $\quad S_3^i = \mathbf{I}_i\Gamma_3^i \xrightarrow{+} \mathbf{I}_i\Delta_3^i$, visiems $i = 1, 2, \ldots n$.

- Σ, Π - baigtinės loginių kintamųjų aibės, $\Sigma \cap \Pi = \emptyset$.

- $\mathbf{B}_{1\ldots n}\Gamma_1 = \mathbf{B}_1\Gamma_1^1, \ldots, \mathbf{B}_n\Gamma_1^n$; $\qquad \mathbf{B}_{1\ldots n}\Delta_1 = \mathbf{B}_1\Delta_1^1, \ldots, \mathbf{B}_n\Delta_1^n$;
  $\mathbf{B}^*_{1\ldots n}\Gamma_1' = \mathbf{B}_1^*\Gamma_1'^1, \ldots, \mathbf{B}_n^*\Gamma_1'^n$; $\qquad \mathbf{B}^*_{1\ldots n}\Delta_1' = \mathbf{B}_1^*\Delta_1'^1, \ldots, \mathbf{B}_n^*\Delta_1'^n$.

- $\mathbf{D}_{1\ldots n}\Gamma_2 = \mathbf{D}_1\Gamma_2^1, \ldots, \mathbf{D}_n\Gamma_2^n; \qquad \mathbf{D}_{1\ldots n}\Delta_2 = \mathbf{D}_1\Delta_2^1, \ldots, \mathbf{D}_n\Delta_2^n.$
- $\mathbf{I}_{1\ldots n}\Gamma_3 = \mathbf{I}_1\Gamma_3^1, \ldots, \mathbf{I}_n\Gamma_3^n; \qquad \mathbf{I}_{1\ldots n}\Delta_3 = \mathbf{I}_1\Delta_3^1, \ldots, \mathbf{I}_n\Delta_3^n.$

$$\frac{\Gamma, \Gamma_1, \mathbf{B}_i^*\Gamma, \mathbf{B}_i^*\Gamma_1 \to \phi_1, \mathbf{B}_i^*\phi_1, \ldots, \mathbf{B}_i^*\phi_n || \ldots || \Gamma, \Gamma_1, \mathbf{B}_i^*\Gamma, \mathbf{B}_i^*\Gamma_1 \to \phi_n, \mathbf{B}_i^*\phi_1, \ldots, \mathbf{B}_i^*\phi_n}{\mathbf{B}_i\Gamma, \mathbf{B}_i^*\Gamma_1 \to \mathbf{B}_i\phi_1, \ldots, \mathbf{B}_i\phi_k, \mathbf{B}_i^*\phi_{k+1}, \ldots, \mathbf{B}_i^*\phi_n}$$

Taisyklė (BEL_i^*) gali būti taikoma tik jeigu $\mathbf{B}_i\Gamma \cup \mathbf{B}_i\phi_1 \cup \ldots \cup \mathbf{B}_i\phi_k \neq \emptyset$.

$$\frac{\Gamma \to \phi_1 \quad || \quad \Gamma \to \phi_2 \quad || \quad \ldots \quad || \quad \Gamma \to \phi_n}{\mathbf{D}_i\Gamma \to \mathbf{D}_i\phi_1, \ldots, \mathbf{D}_i\phi_n} \quad (DES_i^{or})$$

$$\frac{\Gamma \to \phi_1 \quad || \quad \Gamma \to \phi_2 \quad || \quad \ldots \quad || \quad \Gamma \to \phi_n}{\mathbf{I}_i\Gamma \to \mathbf{I}_i\phi_1, \ldots, \mathbf{I}_i\phi_n} \quad (INT_i^{or})$$

$$(i = 1, 2, \ldots, n)$$

- Atvejai, kai $n = 0$ yra leistinas ir tuomet taisyklės atrodo taip:

$$\frac{\Gamma, \Gamma_1, \mathbf{B}_i^*\Gamma, \mathbf{B}_i^*\Gamma_1 \to}{\mathbf{B}_i\Gamma, \mathbf{B}_i^*\Gamma_1 \to} \quad (BEL_i^*) \quad , \quad \frac{\Gamma \to}{\mathbf{D}_i\Gamma \to} \quad (DES_i^{or}) \quad \text{ir} \quad \frac{\Gamma \to}{\mathbf{I}_i\Gamma \to} \quad (INT_i^{or}).$$

Įrodoma, kad sekvencinis skaičiavimas BDI_{elc}^n yra ekvivalentus skaičiavimui BDI_{init}^n ir todėl yra pilnas ir neprieštaringas sekvencinis skaičiavimas daugiaagentinei BDI logikai. Be to naujasis skaičiavimas naudoja tik efektyvią ciklų paiešką ir tik Until tipo ciklams rasti.

Pavyzdys 5 Turime sekvenciją: $S = \circ A(\mathbf{B}_1\phi \cup \mathbf{B}_2\psi) \to \circ\mathbf{B}_2(\psi \vee \phi)$.
Indeksuota sekvencija yra: $S = \circ A_\emptyset^1(\mathbf{B}_1\phi \cup \mathbf{B}_2\psi) \to \circ\mathbf{B}_2(\psi \vee \phi)$.

Išvedimo paieškos medis sekvenciniame skaičiavime BDI_{elc}^n yra:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\text{(final)}\ \mathbf{B}_1^*\phi \xrightarrow{+}}{\phi, \mathbf{B}_1^*\phi \to}\,(Weak_i^{*+})}{\mathbf{B}_1\phi \xrightarrow{+}}\,(BEL_1^*) \quad || \quad \dfrac{\text{(∘-loop)}\oplus\ A_\emptyset^1(\mathbf{B}_1\phi \cup \mathbf{B}_2\psi) \xrightarrow{\circ}}{\circ A_\emptyset^1(\mathbf{B}_1\phi \cup \mathbf{B}_2\psi) \to}\,(\circ^+)}{\mathbf{B}_1\phi, \circ A_\emptyset^1(\mathbf{B}_1\phi \cup \mathbf{B}_2\psi) \to}\,(Weak_i^{*+}) \quad \dfrac{\dfrac{\text{(final)}\ \mathbf{B}_2^*\psi \xrightarrow{+}}{\psi, \mathbf{B}_2^*\psi \to}\,(Weak_i^{*+})}{\mathbf{B}_2\psi \to}\,(BEL_2^*)}{\ldots}}{\ldots}}{\ldots}$$

$$\dfrac{\dfrac{\dfrac{\text{(final)}\ \mathbf{B}_1^*\phi \xrightarrow{+}}{\phi, \mathbf{B}_1^*\phi \to}\,(Weak_i^{*+})}{\mathbf{B}_1\phi \xrightarrow{+}}\,(BEL_1^*) \quad || \quad \dfrac{\dfrac{\dfrac{\text{(final)} \xrightarrow{+} \mathbf{B}_2^*(\psi \vee \phi)}{\to \psi, \phi, \mathbf{B}_2^*(\psi \vee \phi)}\,(Weak_i^{*+})}{\to \psi \vee \phi, \mathbf{B}_2^*(\psi \vee \phi)}\,(\vee R)}{\xrightarrow{+} \mathbf{B}_2(\psi \vee \phi)}\,(BEL_2^*) \quad || \quad \dfrac{\dfrac{\ldots}{A_\emptyset^1(\mathbf{B}_1\phi \cup \mathbf{B}_2\psi) \xrightarrow{\circ}}\,(\text{AU-L}^+)}{\circ A_\emptyset^1(\mathbf{B}_1\phi \cup \mathbf{B}_2\psi) \to}\,(\circ^+)}{\mathbf{B}_1\phi, \circ A_\emptyset^1(\mathbf{B}_1\phi \cup \mathbf{B}_2\psi) \to \mathbf{B}_2(\psi \vee \phi)}\,(Weak_i^{*+})$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\oplus\ \psi, \mathbf{B}_2^*\psi \to \psi, \phi, \mathbf{B}_2^*(\psi \vee \phi)}{\psi, \mathbf{B}_2^*\psi \to \psi \vee \phi, \mathbf{B}_2^*(\psi \vee \phi)}\,(\vee R)}{\mathbf{B}_2\psi \to \mathbf{B}_2(\psi \vee \phi)}\,(BEL_2^*) \quad \ldots}{A_\emptyset^1(\mathbf{B}_1\phi \cup \mathbf{B}_2\psi) \xrightarrow{\circ} \mathbf{B}_2(\psi \vee \phi)}\,(\text{AU-L}^+)}{\circ A_\emptyset^1(\mathbf{B}_1\phi \cup \mathbf{B}_2\psi) \to \circ\mathbf{B}_2(\psi \vee \phi)}\,(\circ^+)$$

Sekvencija S nėra išvedama sekvenciniame skaičiavime BDI^n_{elc} . Sekvencijos S išvedimo paieškos medyje yra kelios ar-šakos. Viename išvedimo paieškos medžio lape yra aksioma, vianame yra ciklinė aksioma, o visuose kituose yra galutinės (neišvedamos) sekvencijos.

Skyriaus pabaigoje yra įrodytos kelios lemos sekvenciniam skaičiavimui BDI^n_{elc} , kurios parodo maksimalų vienos ciklų paieškos metu tikrinamų sekvencijų kiekį, bei maksimalų išvedimo paieškos medžio aukštį.

Gautas ciklų paieškoje tikrinamų sekvencijų įvertis yra ne didesnis, negu atitinkamas įvertis skaidaus laiko logikai. Taip yra todėl, nes bet kurios iš taisyklių (BEL^*_i), (DES^{or}_i), (INT^{or}_i) taikymas išvedimo medyje užkerta kelią ciklų atsiradimui (prieš šių taisyklių taikymą bus taisyklės $(Weak^{*+}_i)$ taikymas, kuris atitinkamas šakas pažymės su $+$). Jeigu tokios taisyklės nenaudojamos, tai praktiškai gausime išvedimą skaidaus laiko logikai.

Išvados

Šioje disertacijoje yra pristatyti keli nauji sekvenciniai skaičiavimai BDI logikoms ir jų fragmentams. Pateikti skaičiavimai yra arba beciklai ($KD45$ logikai) arba naudojantys efektyvią ciklų paiešką (skaidaus laiko ir BDI logikoms). BDI logikos ir jų fragmentai yra plačiai naudojami daugiaagentinių sistemų kūrime. Šioje disertacijoje pateikti rezultatai yra labai naudingi tokių daugiaagentinių sistemų kūrimui.

Rezultatai gauti BDI logikų fragmentams (tiksliau, $KD45$ ir skaidaus laiko logikoms) yra svarbūs ne tik BDI logikomis grįstų agentinių sistemų realizavimui. $KD45$ logika yra naudojama agento įsitikinimams išreikšti. Agento įsitikinimai yra naudojami įvairiose agentinėse sistemose ir yra vieni iš dažniausiai agentui aprašyti naudojamų modalumų. Skaidaus laiko logika yra dar svarbesnė, nes laiko logikos (tiesinio ir skaidaus laiko logikos) gali būti naudojamos beveik visose agentinėse sistemose.

Pagrindiniai disertacijos rezultatai yra šie:

(a) Sukurtas beciklis sekvencinis skaičiavimas $KD45$ logikai, kuri yra naudojama išreikšti agento įsitikinimus BDI architektūroje. Panaudotos ar-taisyklės eliminavo paiešką su grįžimu ir, kartu su įrodytomis savybėmis $KD45$ logikai, įgalino pritaikyti naują požiūrį į išvedimo paieškos medžio formavimą. Prie tam tikrų sąlygų kai kurios išvedimo paieškos medžio sekvencijos gali būti traktuojamos kaip neišvedamos, nepaisant to ar jos, paimtos atskirai, yra išvedamos skaičiavime ar ne. Tokiu atveju, atitinkamos išvedimo paieškos medžio šakos gali būti toliau nebeišvedinėjamos. Tai, ar sekvencija yra traktuojama kaip išvedama ar ne, priklauso ne tik nuo jos pačios, bet ir nuo sekvencijų esančių kitose ar-šakose. Šis naujas požiūris ir panaudotas žymėtas modalumo operatorius leido mums sukonstruoti beciklį sekvencinį skaičiavimą $KD45$ logikai.

(b) Sukurtas sekvencinis skaičiavimas skaidaus laiko logikai, kuris naudoja efektyvią ciklų paiešką. Laiko logikos yra plačiai naudojamos, tačiau sudėtingos logikos. Išvedimo paieškos medžiai sukonstruoti remiantis pristatytu sekvenciniu skaičiavimu yra tokio pat dydžio, kaip ir išvedimo paieškos medžiai sukonstruoti remiantis iki šiol žinomu sekvenciniu skaičiavimu, tačiau taikoma ciklų paieška yra stipriai apribota. Paprastai ciklų paieška yra atliekama visoms medžio sekvencijoms ir jos metu yra tikrinamos visos tėvinės sekvencijos iki pat medžio šaknies. Naujame sekvenciniame skaičiavime ciklų paieškai yra taikomi apribojimai. Pagrindinė apribojimų idėja - ciklų paieškai naudoti ne visas, o tik tam tikras specialiai pažymėtas tėvines sekvencijas. Kitas apribojimas yra tas, kad sekvencijos išvedimo paieškos medžio formavimo metu kai kurios sekvencijos yra pažymimos $+$. Ciklų paieškos metu

yra tikrinamos tik tos sekvencijos, kurios išvedimo paieškos medyje yra aukščiau negu sekvencijos pažymėtos $+$. Dėl šių apribojimų ciklų paieška tampa efektyvia.

(c) Sukurtas sekvencinis skaičiavimas vienaagentinei BDI logikai. BDI logika yra tam tikra kitų modalinių logikų kombinacija. Rezultatai gauti BDI logikų fragmentams buvo pritaikyti sukuriant sekvencinį skaičiavimą pačiai BDI logikai.

(d) Sukurtas sekvencinis skaičiavimas daugiaagentinei BDI logikai. Visi tarpiniai rezultatai buvo apjungti tam, kad gauti pilną išvedimo paieškos sistemą daugiaagentinei BDI logikai, ir kuri remtųsi sekvenciniu skaičiavimu su efektyvia ciklų paieška. Sukurtas sekvencinis skaičiavimas yra efektyvus, pilnas ir neprieštaringas skaičiavimas skirtas daugiaagentinių sistemų realizavimui.

Summary of Dissertation

In this thesis, research on the tasks of the artificial intelligence related to agent implementation is presented. Agents are autonomous systems, those acts in some environment and aspire to achieve preassigned goals. Agents main property is an ability to decide that to do autonomous according to the information obtained from the environment. One of the possible solution for autonomous decision making implementation is the modal logic applications.

For different scopes, there are used different modal logics, those uses one or several modalities to express agent. The most popular modal logic used for agents implementation is BDI logic ([6]), which express agent using three modalities: beliefs, desires and intentions; usually, combined together with temporal logic.

In this thesis, sequent calculi for BDI logics and their fragments are introduced. There is known sequent calculus for BDI logic ([4]), which is sound and complete. However this calculus uses inefficient (direct) loop-check to get decidability. In this thesis, new loop-check free sequent calculus for $KD45$ logic, sequent calculi with an efficient loop-check for branching time and BDI logics are presented.

Aim of the Work and Defending Statements

The main aim of the thesis is to construct an efficient sound and complete derivation search system for BDI logics, based on the sequent calculus, which do not use loop-check or uses only restricted loop-check.

Statements presented for defence are the following:

(a) New constructed sequent calculus for $KD45$ logic is a loop-check free calculus, which not only eliminates loop-check, but also decrease the complexity of the derivation search itself.

(b) New constructed sequent calculus for branching time logic with until operator uses restrictions those decrease complexity of the used loop-check.

(c) New constructed sequent calculi for monoagent and multiagent BDI logics use efficient loop-check for detection of the loops of all the types.

Results of the scientific research are publicated in 7 articles. Two publications are in the journal included in the international databases under acceptation of the Science Council of Lithuania. Intermediate results presented in 5 conference and in the seminars organized by the Logical section of the Institute of Mathematics and Informatics.

Introduced sequent calculus for $KD45$ logic was used to create a prover for $KD45$ logic in the "The Tableau Workbench (TWB)" project.

Outline of the Thesis

The thesis consists of the introduction, five chapters, the concluding remarks, and the appendix.

Chapter 1 presents agents and BDI model for agent implementation. Agent can be defined as entity (machine, software system ...), which can change environment via its actions, can be influenced by the environment, and has the following properties: autonomy, proactiveness, reactivity and social ability. Agent tries to achieve its goals in the changing environment. Agent gets information from the environment via input sensors and performs actions those may change environment.

The autonomy property is the most important agent property. If we want to construct an agent we have to create some decision making mechanism. Decision making mechanism may be based on decision trees (or decision tables), artificial neural networks or some mathematical logic. The most popular logic used for agent implementation is BDI logic presented by A.S. Rao and M. Georgeff ([5]). BDI logic uses three main modalities to describe agent: belief (modality of the $KD45$ logic), desire and intention (both are modalities of the KD logic). Usually, BDI logics are combined with some temporal logic. Agent architecture, based on the BDI logics is also presented in this chapter.

Chapter 2 is devoted for presentation of main definitions used through the thesis. First of all, defintions related to formula are given. Further sequent calculus is described. Invertable, semi-invertable rules, and-rules, or-rules are highlighted. Rule is invertable if it satisfies condition: all premises are derivable if and only if its conclusion is derivable. Rule is semi-invertable if it satisfies condition: if conclusion is derivable then at least one premise is derivable. Or-rules may be used as an alternative for the backtracking. Or-rule contains several premises separated by special symbol ($||$), and its conclusion is derivable if at least one of the premises is derivable.

Semantics of the used logics and loop-check description are given. To get decision procedure we need sequent calculus, those has only invertable or semi-invertable rules. If there are rules those premises are bigger then conclusion, then loop-check technique may be used to get finite decision procedure. Simple loop-check tests all the sequents placed bellow the current sequent to find repeating sequent. Simple loop-check is very inefficient and must be restricted to get useful procedure. Therefore, we need loop-check free sequent calculus or at least sequent calculus with an efficient loop-check.

Sequents with histories are also described. Sequents with histories may be used to get loop-check free sequent calculus, or sequent calculus with an efficient loop-check. In this thesis, special types of histories (marked modal operators, marked sequents and indexes) are also used.

Chapter 3 presents new loop-check free sequent calculus for $KD45$ logic. $KD45$ logic is a fragment of the BDI logics and is used to describe agents beliefs. In this chapter, Hilbert style axiomatization and known sound and complete sequent calculus $KD45_{init}$ for $KD45$ logic are presented. There is known decision procedure based on the sequent calculus $KD45_{init}$, but it uses direct (inefficient) loop-check.

The intermediate sequent calculus $KD45_{wf}$, which has only invertable and semi-invertable rules, but still uses loop-check, is introduced. Some proven lemmas enable us to use a new approach to the inference tree construction. If particular conditions are satisfied, some sequents on the inference tree, irrespective of the fact that the sequent is derivable or not by itself, may be treated as non derivable branches and their derivation are not proceeded (initial sequent derivability will be determined by other branches of the or-rules).

Proven lemmas and used or-rules help us to construct loop-check free sequent calculus $KD45_{lcf}$ for $KD45$ logic. Marked modal operator is also used. Marked modal operator is a special type of the histories, since it stores information about the previously applied rules in the current branch. Marked modal operator is very simple and, therefore, very efficient type of histories. The equivalence of $KD45_{init}$, $KD45_{wf}$ and $KD45_{lcf}$ are proven.

At the end of the chapter, main complexity results are given. Maximum inference tree height is calculated. As a result, we prove that sequent calculus $KD45_{lcf}$ complexity is PSPACE. Moreover, sequent calculus $KD45_{lcf}$ space complexity is only $O(l^3)$ (l - length of an initial sequent). An example of the sequent, which requires $O(l^3)$ space is also given.

Chapter 4 is used to present sequent calculus for branching time logic which uses efficient loop-check. Usually, BDI logics are combined with temporal logics. In this chapter, branching time logic with until operators is researched as a fragment of the BDI logic. Hilbert style axiomatization and known sound and complete sequent calculus PTL_{init} for branching time logic are presented. Sequent calculus PTL_{init} uses inefficient loop-check to detect loops of two types. Usually, loop existence means that current branch is non derivable. Sequent calculus PTL_{init} also uses loop-axioms - loops those means, that current branch is derivable. Restrictions presented in this chapter are applicable for the both loop types.

Two intermediate sequent calculi (PTL_{wf} and PTL_{ol}) are introduced. Sequent calculus PTL_{wf} uses only invertable and semi-invertable rules. Sequent calculus PTL_{ol} uses $\circ$-loops instead normal loops. $\circ$-loop is a loop between sequents, those are premises of the rule $((\circ^W))$ applications. Sequent calculus PTL_{ol} restricts loop-check, since we need to apply loop-check not for every but only for some special sequents and only special sequents are tested.

Further, sequent ground formula is presented. Formula is ground in some sequent if it is not a proper subformula of any other formula in that sequent. It is proven, that if some ground formula is deleted in some rule application, then this rule application is outside any loop. It means, that during loop-check we do not need to test sequents bellow such a rule application. These results, marked sequents and indexes are used to get sequent calculus PTL_{rlc} . Sequent calculus PTL_{rlc} uses efficient loop-check, since only special marked (by $\circ$) sequents are used for loop-check and loop-check is performed only for the marked sequents placed above the sequents marked by $+$. The equivalence of PTL_{init} , PTL_{wf} , PTL_{ol} and PTL_{rlc} are proven. Complexity results of the used loop-check are also given.

Chapter 5 presents sequent calculi for monoagent and multiagent BDI logics. Researched BDI logics are combination of $KD45$, KD and branching time logics. In the previous chapters, loop-check free sequent calculus for $KD45$ logic and sequent calculus with an efficient loop-check for branching time logic were presented. Known sequent calculus for KD logic is loop-check free calculus.

N. NIDE and T. Shiro have proven, that every loop in the monoagent BDI logic is either Belief type loop (loop for $KD45$ logic) or Until type loop (loop for branching time logic), and not both types. Therefore, all restrictions, placed in the previous chapters, are also applicable for BDI logic. We prove, that the same fact is hold for multiagent case.

These facts enable us to construct sequent calculi BDI_{elc} (for monoagent BDI logic) and BDI^n_{elc} (for multiagent BDI logic), those use only efficient loop-check. Moreover, loop-check is used only for Until type loops, since Belief type loops are totally eliminated. Complexity results of the used loop-check is also presented.

Conclusion

In this thesis, there are presented some new sequent calculi for BDI logics and their fragments. Presented calculi are either loop-check free sequent calculi (for $KD45$ logic) or sequent calculi with an efficient loop-check (for branching time and BDI logics). BDI logics and their fragments are widely used in multiagent agent systems implementation. Results, presented in this thesis, are very useful for such a multiagent systems implementation.

Results obtained for the fragments of the BDI logics (namely $KD45$ logic and branching time logic) are important not only for agent systems based on the BDI logics. $KD45$ logic is used to represent agents belief modality, which is one of the most popular modalities used in various agent systems. Branching time temporal logic is even more important, since temporal logics (Linear and branching time logics) are used almost in any agent system.

These are the main contribution of the thesis:

(a) Created loop-check free sequent calculus for $KD45$ logic, which is used to represent agents beliefs in BDI architecture. Or-rules usage instead of the simple backtracking and some proven properties for $KD45$ logic enables us to use a new approach to the inference tree construction. If particular conditions are hold, some sequents on the inference tree, irrespective of the fact that the sequent is derivable or not by itself, may be treated as non derivable branches and their derivation are not proceeded. In this case, sequent derivability depends not only on the sequent itself, but on the other or-branches of the inference tree. This approach together with used marked modal operators enables us to construct loop-check free sequent calculus for $KD45$ logic.

(b) Created sequent calculus with an efficient loop-check for branching time logic. Temporal logics are widely used, but complex logics. Inference trees in introduced sequent calculus have the same size, but applied loop-check was strongly restricted. Usual loop-check tests all the ancestor sequents till the root. The main idea of the used restrictions is to use not all but several special marked sequents. Loop-check is performed only for these special sequents and only such a sequents are tested. Moreover, during inference tree construction, some sequents are marked by $+$. During loop-check, only the marked sequents placed above the sequents marked by $+$ must be tested. Indexes are used to determine sequents those must be marked by $+$. These restrictions leads to the efficient loop-check.

(c) Created sequent calculus with an efficient loop-check for monoagent BDI logic. Since BDI logic is combination of some other modal logics, we applied obtained results for the fragments of the BDI logics to get sequent calculus for BDI logic.

(d) Created sequent calculus with an efficient loop-check for multiagent BDI logic. All intermediate results were summarized to construct full system for multiagent BDI logic, based on the sequent calculus with an efficient loop-check. Introduced sequent calculus is an efficient, sound and complete system applicable for the multiagent systems implementation.

Autoriaus publikacijos

Tarpiniai darbo rezultatai, pateikiami disertacijoje, buvo paskelbti šiose publikacijose:

- Periodiniuose tarptautiniuose recenzuojamuose mokslo lediniuose, įtrauktuose į Lietuvos mokslo tarybos patvirtintą tarptautinių duomenų bazių sąrašą:

  1. A. Birštunas, Efficient loop-check for $KD45$ logic, Lithuanian Mathematical Journal, vol 46, No. 1, 2006, pp. 44–53, Springer, New York.

  2. A. Birštunas, PSPACE complexity of modal logic $KD45_n$, Lithuanian Mathematical Journal, vol 48, No. 2, 2008, pp. 174–187, Springer, New York.

- Tarptautiniuose recenzuojamuose mokslo leidiniuose:

  3. A. Birštunas, Efficient decision procedure for Belief modality, Lithuanian Mathematical Journal, vol 45, spec. issue, 2005, pp. 321–325.

  4. A. Birštunas, Sequent calculus usage for BDI agent implementation, Lithuanian Mathematical Journal, vol 46, spec. issue, 2006, pp. 232–237.

  5. A. Birštunas, Efficient loop-check for multimodal $KD45_n$ logic, Lithuanian Mathematical Journal, vol 47, spec. issue, 2007, pp. 351–355.

  6. A. Birštunas, Restrictions for loop-check in sequent calculus for temporal logic, Lithuanian Mathematical Journal, LMD works, vol 48-49, 2008, pp. 269–274.

  7. A. Birštunas, Restrictions for loop-check in sequent calculus for temporal logic with until operator, Lithuanian Mathematical Journal, LMD works, vol 50, 2009, pp. 247–252.

Žinios apie autorių

Adomas Birštunas gimė 1980 m. kovo 13 d. Vilniuje.

1986 - 1995 m.	M. Daukšos vidurinė mokykla, Vilnius.
1995 - 1998 m.	Vilniaus Licėjus.
1998 - 2002 m.	Bakalauro studijos Vilniaus universitete, Matematikos ir informatikos fakultete. Informatikos studijų programa.
2002 - 2004 m.	Magistrantūros studijos Vilniaus universitete, Matematikos ir informatikos fakultete. Informatikos studijų programa.
2004 - 2007 m.	Asistento pareigos Vilniaus universitete, Matematikos ir informatikos fakultete.
2004 - 2008 m.	Doktorantūros studijos Vilniaus universitete, Matematikos ir informatikos fakultete. Informatikos mokslų kryptis.
2007 - 2010 m.	Lektoriaus pareigos Vilniaus universitete, Matematikos ir informatikos fakultete.

Literatūra

[1] P. Abate, R.Gore, The Tableau WorkBench (TWB),
www [date: 2009-12-05]: http://twb.rsise.anu.edu.au,
Prover for propositional modal logic $KD45$,
www [date: 2009-12-05]:
http://twb.rsise.anu.edu.au/propositional_modal_logic_kd45.

[2] P. Abate, R.Gore, The Tableau WorkBench (TWB), Electronic Notes in Theoretical Computer Science, 2003

[3] M. Fitting, Proof Methods for Modal and Intuitionistic Logics, 1983, D. Reidel, Dordrecht, Holland: Synthese Library.

[4] N. Nide and S. Takata, Deducton systems for BDI logic using sequent calculus, in: Proc. AAMAS'02, 2002, pp. 928–935.

[5] A.S. Rao, M. Georgeff, BDI agents: from theory to practice, in: Proc. of the First International Conference on Multi-Agent Systems (ICMAS-95), 1995, pp. 312–319, S. Francisco, CA.

[6] M. Wooldridge, Reasoning about Rational Agents, The MIT Press, 2000.