

UNIVERZITET U NIŠU
PRIRODNO-MATEMATIČKI FAKULTET U NIŠU



Jelena D. Matejić

APROKSIMATIVNE BISIMULACIJE ZA TEŽINSKE AUTOMATE

DOKTORSKA DISERTACIJA

Niš, 2026.



UNIVERSITY OF NIŠ
FACULTY OF SCIENCES AND MATHEMATICS



Jelena D. Matejić

APPROXIMATE BISIMULATIONS FOR WEIGHTED AUTOMATA

DOCTORAL DISSERTATION

Niš, 2026.

Подаци о докторској дисертацији

Ментор:	Ивана З. Мицић, редовни професор, Природно-математички факултет у Нишу, Универзитет у Нишу
Наслов:	Апроксимативне бисимулације за тежинске аутомате
Резиме:	Ова дисертација проучава симулације и бисимулације за тежинске аутомате, са посебним фокусом на max-plus аутомате. Основни циљ рада је развој и анализа метода за поређење понашања аутомата у ситуацијама у којима стандардни поступци за одређивање симулација и бисимулација нису довољно ефикасни или не доводе до коначног решења. Разматрају се различити типови симулација и бисимулација, њихове слабе варијанте, као и карактеризације највећих решења помоћу система матричних неједначина. Посебна пажња посвећена је проблему бесконачности који се јавља при израчунавању слабих симулација и бисимулација. У том циљу уведене су релаксације засноване на ограничењу дужине речи и датума реализације, као и приступи засновани на доњем и горњем одсецању и померању тежина у max-plus структурама. На тај начин омогућено је контролисано приближно поређење понашања аутомата и ефикасније израчунавање одговарајућих релација. У завршном делу рада развијени приступ је проширен на фази аутомате над производ структуром, увођењем одсечене производ структуре и њој одговарајућих слабих симулација и бисимулација. Добијени резултати представљају допринос развоју метода за ефективно и апроксимативно поређење тежинских аутомата.
Научна област:	Рачунарске науке
Научна дисциплина:	Теоријско рачунарство
Кључне речи:	тежински аутомати, макс-плус аутомати, фази аутомати, симулације, бисимулације, слабе симулације, слабе бисимулације, апроксимативне симулације
УДК:	004.421:004.8
CERIF класификација:	P110, P160, P175, P176
Тип лиценце Креативне заједнице:	CC BY-NC-ND

Data on Doctoral Dissertation

Doctoral Supervisor:	Ivana Z. Micić, Full Professor, Faculty of Science and Mathematics, University of Niš
Title:	Approximate Bisimulations for Weighted Automata
Abstract:	This dissertation studies simulations and bisimulations for weighted automata, with particular emphasis on max-plus automata. The main objective is to develop and analyze methods for comparing the behavior of automata in situations where standard procedures for determining simulations and bisimulations are not sufficiently efficient or do not yield a finite solution. Different types of simulations and bisimulations, their weak variants, as well as characterizations of their greatest solutions by systems of matrix inequalities are considered. Particular attention is devoted to the problem of non-termination that arises in the computation of weak simulations and bisimulations. To address this issue, relaxations based on limiting the length of words and their execution dates are introduced, as well as approaches based on lower and upper truncation and shifting of weights in max-plus structures. These approaches enable controlled approximate comparison of automata behavior and more efficient computation of the corresponding relations. In the final part of the dissertation, the developed approach is extended to fuzzy automata over the product structure by introducing a truncated product structure and the corresponding weak simulations and bisimulations. The obtained results contribute to the development of methods for effective and approximate comparison of weighted automata.
Scientific Field:	Computer science
Scientific Discipline:	Theoretical computer science
Key Words:	weighted automata, max-plus automata, fuzzy automata, simulations, bisimulations, weak simulations, weak bisimulations, approximate simulations, approximate bisimulations
UDC:	004.421:004.8
CERIF Classification:	P110, P160, P175, P176
Creative Commons License Type:	CC BY-NC-ND

Zahvalnica

Najveću zahvalnost dugujem svojoj mentorki, prof. dr Ivani Micić, za nadahnuće, motivaciju i ljubav prema nauci koju je prenela na mene. Hvala joj na ukazanom poverenju, idejama, strpljenju, vremenu i razumevanju, ali pre svega na tome što je verovala u mene i u ono što je iza i ispred nas.

Od srca zahvaljujem prof. dr Jeleni Ignjatović i prof. dr Miroslavu Ćiriću na nesebičnoj pomoći i savetima, na spremnosti da podele svoje znanje i iskustvo, ali i na prilikama koje su mi pružili kroz učešće u različitim projektima i istraživanjima. Njihova podrška i prisustvo na ovom putu za mene imaju posebno mesto.

Veliku zahvalnost dugujem svojoj porodici, posebno roditeljima, sestri i partneru, na bezuslovnoj ljubavi, velikom razumevanju i podršci na svakom koraku. Hvala vam što ste razumeli sva odricanja, delili sa mnom i uspehe i teške trenutke i bili moje sigurno mesto tokom svih ovih godina.

Hvala i mojim prijateljima i kolegama, koji su ovaj put učinili lepšim, lakšim i ispunjenijim. Svaki razgovor, savet, osmeh i ohrabrenje ostavili su trag.

Ova disertacija jeste rezultat rada, istraživanja i istrajnosti, ali iza nje stoje ljudi koji su ovaj put učinili mogućim. Zato je ovo, pre svega, jedno veliko hvala svima vama.

Sadržaj

Spisak procedura, slika i tabela **1**

Predgovor **3**

1 Uvodni pojmovi **9**
1.1 Skupovi, relacije i uređenja 9
1.2 Osnovne algebarske strukture 16
1.3 Mreže i reziduirane mreže 22
1.4 Fazi skupovi i fazi relacije 27

2 Formalni jezici i težinski automati **31**
2.1 Slobodan monoid i formalni jezici 32
2.2 Težinski automati 34
2.3 Reprezentacija težinskih automata 37
2.4 Specifične klase težinskih automata 43

3 Simulacije i bisimulacije za max-plus automate **53**
3.1 Relacije poređenja automata 55
3.2 Simulacije i bisimulacije 57
3.3 Najveće simulacije i bisimulacije 63
3.4 Uticaj izbora max-plus strukture na određivanje simulacija i bisimulacija . 72

4 Slabe simulacije i bisimulacije za max-plus automate **77**
4.1 Presimulacije i prebisimulacije 78
4.2 Slabe simulacije i bisimulacije 79
4.3 Najveće slabe simulacije i bisimulacije 84
4.4 Procedura za računanje najvećih slabih rešenja 86

5 Relaksacija slabih simulacija i bisimulacija za max-plus automate **93**
5.1 Unapređeni postupci za ispitivanje inkluzije i ekvivalencije 94
5.2 Relaksirane slabe simulacije i bisimulacije 99
5.3 Relaksacije ograničene dužinom reči 103
5.4 Δ-odsečeni max-plus poluprsten. Gornje odsecanje poluprstena $\mathbb{P}_\infty$ 108
5.5 Δ-simulacije i Δ-bisimulacije. Relaksacije ograničene datumom 115

6 Odsecanje i pomeranje težina za max-plus automate **121**
6.1 Donje odsecanje i pomeranje težina na max-plus poluprstenu 122
6.2 Gornje odsecanje i pomeranje težina na max-plus poluprstenu 126

6.3 Odsecanje i pomeranje težina na max-plus automatima 128
6.4 Sim. i bisim. na odsečenim i pomerenim max-plus automatima 130
6.5 Slabe sim. i bisim. na odsečenim i pomerenim max-plus automatima . . . 138

7 Aproksimacije fazi automata nad proizvod strukturom 143
7.1 Odsečena proizvod struktura 144
7.2 Fazi automati nad odsečenom proizvod strukturom 149
7.3 ε-slabe simulacije i bisimulacije nad proizvod strukturom 151
7.4 Algoritam za računanje najvećih ε-slabih sim. i bisim. 159

Zaključak 165

Literatura 167

A Programski kod 175

B Biografija i bibliografija autora 203

Spisak procedura, slika i tabela

Spisak procedura

1 Postupak za određivanje najvećeg slabog rešenja između max-plus automata 87
2 Unapređeni postupak za određivanje najveće slabe simulacije/bisimulacije 97
3 Testiranje inkluzije i ekvivalencije između max-plus automata 98
4 Određivanje najveće ε-slabe simulacije i ε-slabe bisimulacije 161

Spisak slika

1 Venov dijagram operacija među skupovima 10
2 Grafovska i matrična reprezentacija binarne relacije R iz Primera 1.1 . . . 12
3 Primeri preslikavanja 13
4 Hasseov dijagram za Primer 1.2 15
5 Hasseov dijagram za Primer 1.3 15
6 Hijerarhija osnovnih algebarskih struktura 19
7 Usmerena grana u težinskom automatu 37
8 Putanja u težinskom automatu 38
9 Primer ciklusa (levo) i petlje (desno) 39
10 Hijerarhijski odnos klasičnih konačnih, fazi i težinskih automata 44
11 Grafovska reprezentacija automata iz primera 2.1 46
12 Grafovska reprezentacija max-plus automata iz primera 2.2 51
13 Klasifikacija simulacija i bisimulacija automata 56
14 Forward simulacija između max-plus automata 58
15 Odnosi između forward simulacija i odgovarajućih slabih relacija. 83
16 Stablo foldera programa 177
17 Početna stranica programa 178
18 Stranica za računanje (bi)simulacija nad komoletnim max-plus poluprstenom 179
19 Računanje slabih (bi)simulacija nad komoletnim max-plus poluprstenom . 180
20 Računanje odselenih (bi)simulacija nad komoletnim max-plus poluprstenom 181

Spisak tabela

1 Pregled osnovnih osobina algebarskih struktura 17
2 Operacije u Łukasiewiczevoj, Gödelovoj i Goguenovoj reziduiranoj mreži 26
3 Pregled osnovnih max-plus struktura i operacija 50
4 Pregled običnih i slabih simulacija i bisimulacija. 81
5 Proces računanja najveće slabe forward simulacije iz Primera 4.1. 90
6 Proces računanja najveće slabe forward simulacije iz Primera 4.3. 92
7 τ-parovi izračunati u Primeru 5.1 i njihovi reziduali. 106
8 Detalji izračunavanja iz primera 5.1 za $\Delta = 5$. 113
9 Pregled max-plus struktura i odgovarajućih operacija 142

Predgovor

Težinski automati predstavljaju značajno proširenje klasičnih konačnih automata, jer omogućavaju da se, pored informacije o postojanju prelaza između stanja, u model sistema uključe i kvantitativne informacije. U zavisnosti od izabrane algebarske strukture nad skupom težina, moguće je modelovati različite karakteristike sistema, kao što su trajanje događaja, troškovi, vremena izvršavanja, stepen pripadnosti i druge veličine od značaja za opis i analizu sistema. Težinski automati zbog toga predstavljaju prirodnu vezu između teorije automata, formalnih jezika, algebre i oblasti u kojima je potrebno opisati i analizirati kvantitativno ponašanje sistema [1, 2, 3].

Jedan od osnovnih problema u teoriji automata jeste poređenje ponašanja dva automata, odnosno ispitivanje da li je ponašanje jednog automata sadržano u ponašanju drugog ili su njihova ponašanja jednaka. Kod klasičnih automata ovi problemi svode se na ispitivanje inkluzije i jednakosti formalnih jezika, dok se kod težinskih automata ponašanje opisuje funkcijom koja svakoj reči pridružuje vrednost iz odgovarajuće algebarske strukture. Direktno rešavanje problema inkluzije i ekvivalencije tada zahteva poređenje ponašanja automata nad svim rečima iz slobodnog monoida, što u opštem slučaju može biti računarski zahtevno. Zbog toga se javlja potreba za relacijama koje omogućavaju da se ponašanje automata poredi posredno, preko odnosa između njihovih stanja i prelaza.

Simulacije i bisimulacije predstavljaju značajan alat za takvo poređenje. Prvobitno razvijene u teoriji tranzicionih sistema, koriste se za analizu odnosa između sistema, kao i za njihovu redukciju uz očuvanje određenih svojstava [4, 5]. U klasičnom pristupu simulacije i bisimulacije predstavljaju binarne relacije između skupova stanja. Međutim, takav pristup nije dovoljan za izražavanje kvantitativnih razlika između težinskih automata. Zbog toga su razvijene kvantitativne simulacije i bisimulacije, kod kojih se relacije između stanja predstavljaju matricama nad odgovarajućom algebarskom strukturom [6, 7, 8, 9]. Na taj način moguće je istovremeno uzeti u obzir strukturne odnose između stanja i kvantitativne informacije sadržane u težinama automata.

Posebno značajnu klasu težinskih automata predstavljaju max-plus automati. Max-plus algebra i max-plus automati imaju široku primenu u modelovanju diskretnih sistema događaja, uključujući proizvodne i transportne sisteme, komunikacione mreže i probleme raspoređivanja [10, 11, 12]. U takvim modelima težine se mogu interpretirati kao vremena izvršavanja, kašnjenja ili troškovi, dok se ponašanje automata određuje maksimalnom težinom putanja koje odgovaraju datoj ulaznoj reči [13, 14, 15]. Upravo zbog ove mogućnosti modelovanja kvantitativnog ponašanja, max-plus automati predstavljaju prirodan okvir za

proučavanje simulacija i bisimulacija u ovoj disertaciji.

Uvođenje simulacija i bisimulacija za max-plus automate omogućava da se problemi inkluzije i ekvivalencije ponašanja razmatraju preko sistema matričnih nejednačina. Za max-plus automate definisana su dva tipa simulacija i četiri tipa bisimulacija, pri čemu se njihovo postojanje i određivanje najvećih relacija mogu svesti na rešavanje odgovarajućih sistema matričnih nejednačina [9]. Ovakav pristup daje efikasan alat za poređenje automata, ali istovremeno otvara i pitanje terminacije postupaka za računanje najvećih rešenja. U opštem slučaju iterativni postupak se ne mora završiti nakon konačno mnogo koraka, što predstavlja jedno od osnovnih ograničenja ovog pristupa.

Dodatno ograničenje nastaje zbog činjenice da standardne simulacije i bisimulacije mogu biti suviše restriktivne. Može se dogoditi da dva automata imaju inkluzivna ili čak ekvivalentna ponašanja, a da između njih ne postoji simulacija ili bisimulacija odgovarajućeg tipa. Ovaj problem motivisao je uvođenje slabih simulacija i bisimulacija, kod kojih se uslovi za poređenje ne postavljaju neposredno za pojedinačne prelaze, već se uzima u obzir ponašanje automata nakon obrade reči. Takav pristup omogućava da se obuhvati šira klasa odnosa između automata [16, 17]. Za max-plus automate, slabe simulacije i bisimulacije predstavljaju prirodno proširenje standardnih relacija, pri čemu se i one mogu karakterisati odgovarajućim sistemima matričnih nejednačina [17].

Međutim, proširivanje relacija sa standardnih na slabe simulacije i bisimulacije dovodi do novog računarskog problema. Za razliku od sistema koji definišu standardne simulacije i bisimulacije, odgovarajući sistemi za slabe relacije u opštem slučaju mogu sadržati uslove za beskonačno mnogo reči iz skupa X^*. Zbog toga ni određivanje najvećih slabih simulacija i bisimulacija nije u opštem slučaju moguće svesti na postupak koji se garantovano završava nakon konačno mnogo koraka [17]. Ovaj problem predstavlja neposrednu motivaciju za razvoj aproksimativnih i relaksiranih pristupa.

Jedan način za prevazilaženje navedenog problema jeste relaksacija uslova tako da se umesto svih reči iz skupa X^* razmatra odgovarajući konačan podskup. Na taj način beskonačan sistem matričnih nejednačina može biti zamenjen konačnim sistemom, čime se omogućava efektivnije računanje najvećih rešenja. U ovoj disertaciji razmatraju se relaksacije zasnovane na ograničavanju dužine reči, kao i relaksacije zasnovane na ograničavanju datuma realizacije reči. Takav pristup omogućava da se nivo detaljnosti poređenja prilagodi konkretnom problemu.

Drugi pristup razmatran u disertaciji zasniva se na modifikovanju same algebarske strukture nad kojom se vrši računanje. Kod standardnih postupaka za određivanje najvećih simulacija i bisimulacija, iterativno računanje može proizvesti beskonačno opadajuće nizove elemenata koji konvergiraju ka $-\infty$. Odsecanjem kompletnog max-plus poluprstena i odgovarajućim pomeranjem težina moguće je u određenim slučajevima ograničiti takvo

ponašanje i omogućiti konačno računanje najvećih relacija [18]. Istovremeno, ovakav pristup nije bez ograničenja jer prelazak na odsečenu strukturu može promeniti svojstva relacija između automata, pa se zbog toga posebno mora analizirati odnos između simulacija i bisimulacija originalnih i transformisanih automata.

Navedeni problemi i pristupi predstavljaju osnovu za šire pitanje kojim se disertacija bavi - Kako konstruisati relacije poređenja koje istovremeno pružaju dovoljno preciznu informaciju o odnosu između ponašanja automata i mogu se efikasno izračunati? Odgovor se u radu razvija kroz više nivoa, počevši od standardnih simulacija i bisimulacija, preko slabih i relaksiranih relacija, do aproksimativnih simulacija i bisimulacija zasnovanih na modifikacijama algebarske strukture.

Osnovni cilj disertacije jeste razvoj i analiza novih pristupa za poređenje težinskih automata pomoću simulacija, bisimulacija i njihovih slabih i aproksimativnih varijanti. Poseban fokus stavljen je na max-plus automate, ali se razvijeni koncepti u završnom delu rada proširuju i na fazi automate nad Goguenovom, odnosno proizvod strukturom. Pri tome se posebna pažnja posvećuje karakterizaciji najvećih relacija, razvoju postupaka za njihovo računanje, analizi njihove terminacije i proučavanju načina na koje se problemi beskonačnosti mogu ublažiti relaksacijom ili aproksimacijom.

Jedan od doprinosa disertacije jeste sistematska analiza simulacija i bisimulacija za max-plus automate. U okviru ovog pristupa razmatraju se različiti tipovi simulacija i bisimulacija, njihova karakterizacija pomoću sistema matričnih nejednačina i postupci za određivanje njihovih najvećih rešenja. Pored toga, proučava se uticaj izbora max-plus strukture na dobijene relacije i na ponašanje postupaka za njihovo računanje. Ovi rezultati predstavljaju osnovu za dalji razvoj slabih i aproksimativnih oblika simulacija i bisimulacija [9].

Naredni doprinos odnosi se na slabe simulacije i bisimulacije za max-plus automate. Uvode se slabe presimulacije i prebisimulacije, kao i odgovarajuće slabe simulacije i bisimulacije, i karakterišu njihova najveća rešenja. Na osnovu dobijenih karakterizacija razvijaju se postupci za ispitivanje njihovog postojanja i za određivanje najvećih relacija kada one postoje. Time se proširuje skup situacija u kojima se pomoću simulacija i bisimulacija može zaključivati o inkluziji i ekvivalenciji ponašanja max-plus automata [17].

Dalje proširenje predstavlja uvođenje relaksiranih slabih simulacija i bisimulacija. Osnovna ideja je da se beskonačan skup uslova koji karakteriše slabe relacije ograniči na skup reči određenih svojstava. U disertaciji se razmatraju relaksacije ograničene dužinom reči i relaksacije ograničene datumom realizacije. Na ovaj način uvodi se kontrolisana aproksimacija slabih simulacija i bisimulacija koja omogućava da se problem računanja prilagodi unapred zadatom nivou posmatranja.

Poseban doprinos predstavlja proučavanje donjeg i gornjeg odsecanja i pomeranja težina na max-plus poluprstenu. Uvode se odgovarajuće odsečene strukture, proučavaju njihova algebrajska svojstva i analizira njihov uticaj na simulacije i bisimulacije max-plus automata. Pokazuje se da odsecanje u određenim slučajevima može omogućiti konačno određivanje najvećih relacija ili smanjiti broj potrebnih iteracija, ali se istovremeno identifikuju i situacije u kojima transformacija može dovesti do gubitka simulacije koja postoji između originalnih automata [18].

Završni doprinos odnosi se na aproksimativno poređenje fazi automata nad proizvod strukturom. Za razliku od nekih drugih fazi struktura, poluprstenski ostatak proizvod strukture nije lokalno konačan, što može dovesti do pojave beskonačnih sistema matričnih nejednačina pri određivanju slabih simulacija i bisimulacija. Uvođenjem ε-odsečene proizvod strukture ovaj problem se svodi na odgovarajuću lokalno konačnu strukturu. Na osnovu toga definišu se ε-slabe simulacije i ε-slabe bisimulacije, koje omogućavaju aproksimativno poređenje ponašanja originalnih fazi automata uz unapred kontrolisano odstupanje. Za ove relacije razvija se i postupak za ispitivanje njihovog postojanja i računanje najvećih relacija kada one postoje [19].

Na taj način doprinos disertacije nije usmeren samo na uvođenje pojedinačnih novih relacija, već i na razvoj jedinstvenog pristupa problemu poređenja težinskih automata u situacijama u kojima potpuno i direktno računanje nije moguće ili nije efikasno. Standardne simulacije i bisimulacije daju polazni model, slabe relacije proširuju njegovu primenljivost, relaksacije ograničavaju problem na konačan skup relevantnih reči, dok odsecanje i aproksimacija modifikuju strukturu nad kojom se računanje vrši. Time se uspostavlja prirodna veza između teorijskih svojstava relacija i njihove računarske obradivosti.

Disertacija je organizovana u sedam glava.

Glava 1 - Uvodni pojmovi, predstavlja matematičku osnovu rada. U njoj se uvode skupovi, relacije, preslikavanja i uređenja, osnovne algebarske strukture, mreže i reziduirane mreže, kao i osnovni pojmovi fazi skupova i fazi relacija. Posebna pažnja posvećena je strukturama koje će biti korišćene za definisanje težina i relacija između automata.

Glava 2 - Formalni jezici i težinski automati, uvodi slobodan monoid i formalne jezike, definišu se težinski automati i njihovo ponašanje, a zatim se prikazuju njihova grafovska i matrična reprezentacija. Na kraju se uvode klasični konačni automati, fazi automati i max-plus automati, koji predstavljaju osnovne klase automata razmatrane u nastavku rada.

Glava 3 - Simulacije i bisimulacije za max-plus automate, predstavlja polaznu tačku istraživanja. Uvode se različiti tipovi simulacija i bisimulacija za max-plus automate, karakterišu se pomoću sistema matričnih nejednačina i proučavaju njihova najveća rešenja. Razvijaju se postupci za njihovo računanje i analiziraju problemi terminacije iterativnih postupaka. Rezultati ove glave zasnovani su na istraživanju predstavljenom u radu [9].

Glava 4 - Slabe simulacije i bisimulacije za max-plus automate, proširuje standardne relacije na njihove slabe varijante. Uvode se slabe presimulacije, prebisimulacije, simulacije i bisimulacije, karakterišu njihova najveća rešenja i razvija postupak za ispitivanje njihovog postojanja. Posebno se analizira problem terminacije postupka koji nastaje zbog mogućnosti pojave beskonačno mnogo uslova. Rezultati ove glave zasnovani su na radu [17].

Glava 5 - Relaksacija slabih simulacija i bisimulacija za max-plus automate, bavi se problemom beskonačnosti sistema koji definišu slabe relacije. Uvodi se opšti pojam relaksiranih slabih simulacija i bisimulacija, a zatim se razmatraju relaksacije ograničene dužinom reči i relaksacije ograničene datumom realizacije. Na ovaj način razvija se aproksimativni pristup koji omogućava konačno računanje relacija za unapred određenu klasu relevantnih reči.

Glava 6 - Odsecanje i pomeranje težina za max-plus automate, proučava mogućnost rešavanja problema beskonačnosti modifikovanjem max-plus strukture. Uvode se donja i gornja odsecanja kompletnog max-plus poluprstena i odgovarajuće operacije, a zatim se analizira njihova primena na max-plus automate, simulacije i bisimulacije. U završnom delu glave razmatra se mogućnost primene odsecanja na slabe simulacije i bisimulacije i ukazuje na ograničenja takvog pristupa. Rezultati su zasnovani na radu [18].

Glava 7 - Aproksimacije fazi automata nad proizvod strukturom, proširuje razvijeni aproksimativni pristup na fazi automate nad Goguenovom, odnosno proizvod strukturom. Uvodi se odsečena proizvod struktura i razmatraju fazi automati nad njom, nakon čega se definišu ε-slabe simulacije i bisimulacije. Razvija se algoritam za ispitivanje njihovog postojanja i određivanje najvećih relacija kada one postoje. Na taj način se pokazuje da se ideja aproksimativnog poređenja može primeniti i u situacijama u kojima originalna struktura ne omogućava direktno efikasno rešavanje problema. Rezultati ove glave zasnovani su prvenstveno na radu [19].

Celokupna organizacija disertacije prati razvoj ideje da se od preciznog, ali često računarski zahtevnog poređenja ponašanja automata, dođe do relacija koje omogućavaju kontrolisanu relaksaciju ili aproksimaciju tog poređenja i time omogućavaju njegovo efektivnije računanje. Polazeći od standardnih simulacija i bisimulacija za max-plus automate, preko slabih i relaksiranih varijanti, pa sve do odsečenih struktura i aproksimativnih relacija za fazi automate, u radu se ispituju različiti načini prevazilaženja ograničenja koja nastaju pri direktnom rešavanju problema poređenja težinskih automata. Dobijeni rezultati doprinose razvoju teorije simulacija i bisimulacija za težinske automate i pružaju osnovu za dalje proučavanje aproksimativnih metoda za analizu njihovog ponašanja.

Glava 1

Uvodni pojmovi

Skupovi, relacije i preslikavanja predstavljaju osnovne matematičke pojmove koji se koriste u algebri, teoriji skupova i matematičkoj logici. U ovom poglavlju uvode se osnovni pojmovi i rezultati koji će biti korišćeni u nastavku rada.

U Sekciji 1.1 razmatraju se skupovi, relacije, preslikavanja i relacije uređenja, kao i osnovni pojmovi vezani za minimum, maksimum, infimum i supremum. U Sekciji 1.2 uvode se osnovne algebarske strukture, uključujući polugrupe, monoide, grupe, poluprstene, prstene i dioide. Sekcija 1.3 posvećena je mrežama i reziduiranim mrežama, sa posebnim osvrtom na algebarske strukture koje imaju značajnu primenu u teoriji fazi sistema i teoriji automata. Na kraju, u Sekciji 1.4 uvode se osnovni pojmovi fazi skupova i fazi relacija, koji predstavljaju matematičku osnovu za proučavanje fazi i težinskih automata u narednim poglavljima.

1.1 Skupovi, relacije i uređenja

U ovoj sekciji predstavljeni su osnovni pojmovi teorije skupova i teorije uređenja koji su neophodni za dalji rad. Najpre se razmatraju skupovi, relacije i preslikavanja kao fundamentalni matematički pojmovi na kojima se zasniva definisanje i analiza različitih algebarskih i uređenih struktura. Relacije i preslikavanja omogućavaju formalno opisivanje odnosa među elementima skupova i predstavljaju osnovu za uvođenje složenijih matematičkih koncepata [20, 21, 22, 23, 24].

Savremeno proučavanje teorije skupova započeli su Georg Cantor i Richard Dedekind krajem 19. veka. Cantor je u radu iz 1874. godine pokazao da skup realnih brojeva nije prebrojiv i time postavio temelje sistematskog proučavanja beskonačnih skupova [25]. U svojim kasnijim radovima razvio je teoriju kardinalnih i ordinalnih brojeva i uveo pojam dobro uređenih skupova [26], čime je značajno uticao na razvoj moderne matematike.

Pored osnovnih operacija nad skupovima, uvode se i binarne relacije, preslikavanja i relacije uređenja. Posebna pažnja posvećena je parcijalno uređenim skupovima, kao i pojmovima minimuma, maksimuma, infimuma i supremuma, koji imaju centralnu ulogu u teoriji uređenja. Ovi pojmovi predstavljaju osnovu za definisanje polumreža i mreža, koje će biti razmatrane u nastavku rada. Osnovni rezultati iz teorije skupova i uređenja mogu se pronaći u standardnoj literaturi [20, 21, 22, 23, 27, 28].

Skup je jedan od osnovnih matematičkih pojmova koji predstavlja kolekciju različitih objekata. Objekti koji pripadaju skupu nazivaju se elementi ili članovi skupa. Prema konvenciji, skupovi se označavaju velikim slovima abecede, dok se njihovi elementi označavaju malim slovima abecede. Za skup A i element x, zapis $x \in A$ označava da je x element skupa A (x pripada skupu A).

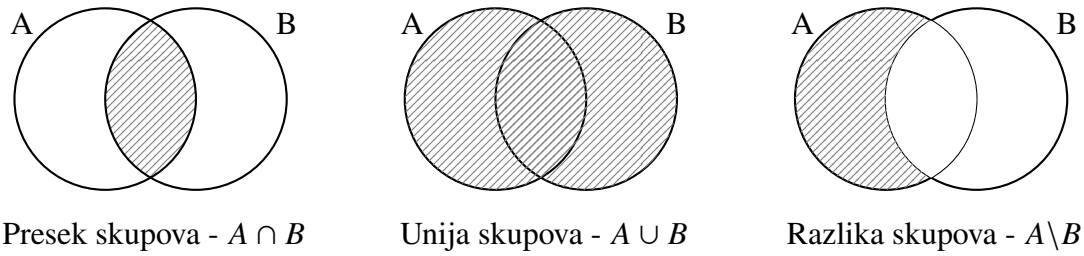
Za univernizalni skup U i skupove $A, B \subseteq U$, definišu se sledeće operacije:

Unija $A \cup B = \{x \mid x \in A \ \vee \ x \in B\}$,

Presek $A \cap B = \{x \mid x \in A \ \wedge \ x \in B\}$,

Razlika $A \setminus B = \{x \mid x \in A \ \wedge \ x \notin B\}$.

Osnovne operacije nad skupovima omogućavaju njihovo kombinovanje i poređenje. Kroz operacije poput unije, preseka, razlike i komplementa, možemo formirati nove skupove i analizirati odnose među njima. Skupove i operacije nad njima možemo vizuelizovati korišćenjem Venovih dijagrama (videti Sliku 1).



Presek skupova - $A \cap B$ Unija skupova - $A \cup B$ Razlika skupova - $A \setminus B$

Slika 1: Venov dijagram operacija među skupovima

Neka je U univerzalni skup takav da je $A \subseteq U$. ***Komplement*** skupa A definiše se sa $A^C = U \setminus A$. Drugim rečima, komplement skupa A čine svi elementi univerzalnog skupa U koji ne pripadaju skupu A.

Za skupove A i B definišemo relaciju ***inkluzije*** $A \subseteq B \iff (\forall x)(x \in A \Rightarrow x \in B)$.

Skup bez elemenata naziva se ***prazan skup*** i označava se $\emptyset$. Prazan skup je podskup svakog skupa, tj. $\emptyset \subseteq A$ za svaki skup A.

Jednakost skupova A i B važi ako oba skupa imaju iste elemente, što zapisujemo:

$$A = B \Leftrightarrow (\forall x)(x \in A \ \Leftrightarrow \ x \in B) \Leftrightarrow (A \subseteq B \ \wedge \ B \subseteq A).$$

Uređeni par elemenata je dvojka (a, b) kod koje je redosled komponenti od značaja. Dva uređena para su jednaka ako i samo ako su jednaka pokoordinatno, odnosno

$$(a, b) = (c, d) \ \Leftrightarrow \ a = c \ \wedge \ b = d.$$

Pojam uređenog para se prirodno proširuje na ***uređenu n-torku*** $(a_1, \ldots, a_n)$, kod koje je redosled navođenja elemenata takođe od značaja. Dve uređene n-torke jednake su ako i

samo ako su pokoordinatno jednake, odnosno

$$(a_1, a_2, \ldots, a_n) = (b_1, b_2, \ldots, b_n) \Leftrightarrow a_i = b_i, \text{ za svako } i \in \{1, \ldots, n\}.$$

Prema tome, uređene n-torke se razlikuju ukoliko se razlikuju u bar jednoj odgovarajućoj komponenti (koordinati).

Dekartov proizvod skupova A i B definiše se kao skup uređenih parova

$$A \times B = \{(a, b) \mid a \in A,\ b \in B\}.$$

U opštem slučaju, Dekartov proizvod skupova $A_1, \ldots, A_n$ definiše se kao

$$A_1 \times \cdots \times A_n = \{(a_1, \ldots, a_n) \mid a_i \in A_i,\ i = 1, \ldots, n\}.$$

Relacija dužine n (n-arna relacija) između skupova $A_1, \ldots, A_n$ je svaki podskup Dekartovog proizvoda $A_1 \times \cdots \times A_n$. Za $n = 2$ dobija se ***binarna relacija***.

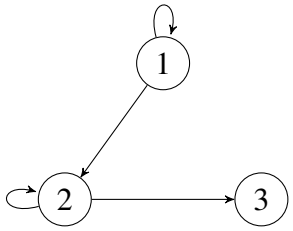
Ako je $R \subseteq A \times B$, pišemo xRy ako i samo ako je $(x, y) \in R$. Tada kažemo da su elementi x i y u relaciji R. Poseban slučaj predstavlja binarna relacija na skupu A, koja je svaki podskup skupa $A \times A$.

Neka je A neprazan skup. Za relacije na skupu A uvodimo sledeće pojmove:

Prazna relacija	$R_\emptyset = \emptyset$,
Identička relacija	$R_\Delta = \{(x, x) \mid x \in A\}$,
Univerzalna relacija	$R_\nabla = A \times A$,
Inverzna relacija	$R^{-1} = \{(y, x) \mid (x, y) \in R\}$,
Komplement relacije	$\overline{R} = (A \times A) \setminus R$,
Domen relacije	$\mathrm{dom}(R) = \{x \in A \mid \exists\, y \in A,\ (x, y) \in R\}$,
Rang relacije	$\mathrm{rang}(R) = \{y \in A \mid \exists\, x \in A,\ (x, y) \in R\}$.

Relacije se mogu predstavljati na različite načine. U nastavku će biti razmatrane grafovska i matrična reprezentacija relacija. Grafovska reprezentacija, zasnovana na usmerenim grafovima, pruža intuitivan prikaz odnosa između elemenata, dok matrična reprezentacija, zasnovana na binarnim matricama, omogućava efikasno izvođenje operacija nad relacijama i predstavlja osnovu za matrični pristup koji će biti korišćen u nastavku rada.

Primer 1.1 *Neka je dat skup* $A = \{1, 2, 3\}$*, i relacija* $R = \{(1, 1), (1, 2), (2, 2), (2, 3)\}$. *Na Slici 2 prikazane su dve ekvivalentne reprezentacije relacije R.*
U grafovskoj reprezentaciji čvorovi odgovaraju elementima skupa A, dok usmerene grane predstavljaju uređene parove koji pripadaju relaciji R. U matričnoj reprezentaciji element matrice u i*-toj vrsti i* j*-toj koloni ima vrednost* 1 *ako i samo ako uređeni par* (i, j) *pripada relaciji R, dok vrednost* 0 *označava da uređeni par* (i, j) *ne pripada relaciji.*



Grafovska reprezentacija

$$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

Matrična reprezentacija

Slika 2: Grafovska i matrična reprezentacija binarne relacije R iz Primera 1.1

U matričnoj reprezentaciji kompozicija relacija odgovara relacijskom množenju matrica. Za binarne relacije to je max–min kompozicija.

Neka je A neprazan skup, a $R, S \subseteq A \times A$ binarne relacije na skupu A. Kompozicija relacija R i S definiše se sa

$$R \circ S = \{(x, z) \in A \times A \mid (\exists\, y \in A)\big((x, y) \in R \wedge (y, z) \in S\big)\}.$$

Za skup $X \subseteq A$ i binarnu relaciju $R \subseteq A \times A$ definišemo

$$X \circ R = \{y \in A \mid (\exists\, x \in X)\, (x, y) \in R\},$$

$$R \circ X = \{x \in A \mid (\exists\, y \in X)\, (x, y) \in R\}.$$

Neka je A neprazan skup, $B, C \subseteq A$ njegovi podskupovi, $R, S, T \subseteq A \times A$ binarne relacije na skupu A i $\{S_i\}_{i \in I}$ familija binarnih relacija na skupu A, gde je I indeksni skup. Tada važe sledeća svojstva kompozicije relacija:

Asocijativnost	$(R \circ S) \circ T = R \circ (S \circ T) = R \circ S \circ T,$
	$(B \circ R) \circ S = B \circ (R \circ S) = B \circ R \circ S,$
	$(B \circ R) \circ C = B \circ (R \circ C) = B \circ R \circ C,$
	$(R \circ S) \circ B = R \circ (S \circ B) = R \circ S \circ B.$
Identička relacija	$R \circ R_\Delta = R_\Delta \circ R = R.$
Monotonost	$S_1 \subseteq S_2 \Rightarrow R \circ S_1 \subseteq R \circ S_2,$
	$S_1 \subseteq S_2 \Rightarrow S_1 \circ R \subseteq S_2 \circ R.$
Distributivnost u odnosu na uniju	$R \circ (\bigcup_{i \in I} S_i) = \bigcup_{i \in I} (R \circ S_i),$
	$(\bigcup_{i \in I} S_i) \circ R = \bigcup_{i \in I} (S_i \circ R).$
Inverz kompozicije	$(R \circ S)^{-1} = S^{-1} \circ R^{-1}.$

Funkcija ili preslikavanje iz nepraznog skupa A u neprazni skup B je binarna relacija $f \subseteq A \times B$, koju označavamo sa $f : A \to B$, za koju važe sledeći uslovi:

Dobra definisanost $(\forall x \in A)(\exists y \in B)$ tako da je $(x, y) \in f$,

Jednoznačnost ako je $(x, y_1) \in f$ i $(x, y_2) \in f$, tada je $y_1 = y_2$.

Drugim rečima, preslikavanje iz skupa A u skup B je pravilo koje svakom elementu $x \in A$ pridružuje tačno jedan element $y \in B$.

Preslikavanje može imati i sledeća svojstva:

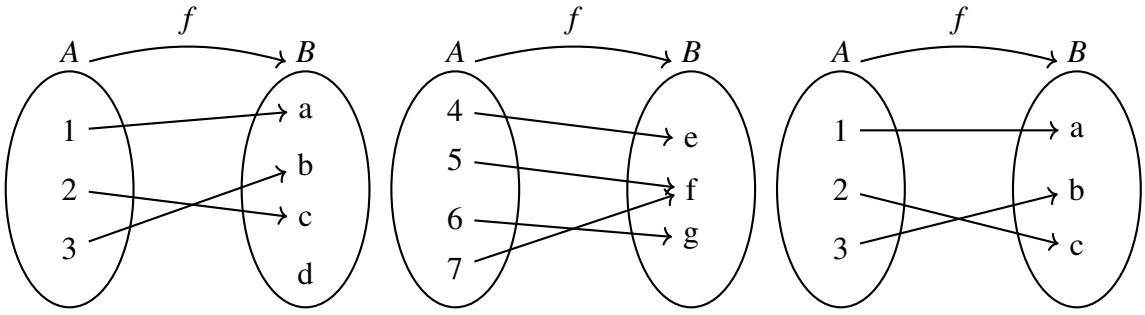
Injektivnost ako je $x_1 \neq x_2$, tada je $f(x_1) \neq f(x_2)$, $(\forall x_1, x_2 \in A)$,

Surjektivnost $(\forall y \in B)(\exists x \in A)$ tako da je $y = f(x)$,

Bijektivnost ako je preslikavanje istovremeno injektivno i surjektivno.

Preslikavanje koje ima svojstvo injektivnosti naziva se ***injekcija*** ili ***"1–1"***, preslikavanje koje ima svojstvo surjektivnosti naziva se ***surjekcija*** ili ***"na"***, dok se preslikavanje koje je istovremeno injektivno i surjektivno naziva ***bijekcija***. Vizuelni prikaz ovih svojstava dat je na Slici 3.

Za bijekciju $f : A \to B$ postoji jedinstveno preslikavanje $f^{-1} : B \to A$, koje nazivamo ***inverzno preslikavanje***, te važi $f^{-1}(f(x)) = x$ i $f(f^{-1}(y)) = y$, gde su $x \in A, y \in B$.



Injektivno preslikavanje Surjektivno preslikavanje Bijektivno preslikavanje

Slika 3: Primeri preslikavanja

Neka je data relacija $R \subseteq A \times A$ na skupu A. Kažemo da je relacija R:

Refleksivna ako za svako $x \in A$ važi $(x, x) \in R$;

Simetrična ako za sve $x, y \in A$ iz $(x, y) \in R$ sledi $(y, x) \in R$;

Antisimetrična ako za sve $x, y \in A$ iz $(x, y) \in R$ i $(y, x) \in R$ sledi $x = y$;

Tranzitivna ako za sve $x, y, z \in A$ iz $(x, y) \in R$ i $(y, z) \in R$ sledi $(x, z) \in R$.

Svojstva binarnih relacija se međusobno ne podrazumevaju. Određene kombinacije navedenih svojstava definišu važne klase binarnih relacija:

- ***Kvazi-uređenje (preorder)*** je refleksivna i tranzitivna relacija;
- ***Relacija ekvivalencije*** je refleksivna, simetrična i tranzitivna relacija;
- ***Parcijalno uređenje*** je refleksivna, antisimetrična i tranzitivna relacija.

Ako je relacija $\leq$ parcijalno uređenje na skupu A, tada se uređeni par $(A, \leq)$ naziva ***parcijalno uređen skup***. Elementi $x, y \in A$ su *uporedivi* ako važi $x \leq y$ ili $y \leq x$. Ako su svaka dva elementa skupa međusobno uporediva, tada je $(A, \leq)$ ***linearno uređen skup*** ili ***lanac***. Kaže se da skup zadovoljava ***uslov rastućeg lanca*** ako je svaki strogo rastući niz konačan, odnosno ***uslov opadajućeg lanca*** ako je svaki strogo opadajući niz konačan.

Za uređene skupove $(A, \leq)$ i $(B, \leq)$, preslikavanje $f : A \to B$ je:

Izotono (rastuće)	ako za sve $x, y \in A$ iz $x \leq y$ sledi $f(x) \leq f(y)$;
Antitono (opadajuće)	ako za sve $x, y \in A$ iz $x \leq y$ sledi $f(y) \leq f(x)$;
Izomorfno (izomorfizam)	ako je f bijekcija, a preslikavanja f i f^{-1} su izotona.

U parcijalno uređenom skupu $(A, \leq)$ element $x \in A$ je:

Minimalni element	ako za svako $y \in A$ iz $y \leq x$ sledi $y = x$;
Maksimalni element	ako za svako $y \in A$ iz $x \leq y$ sledi $y = x$;
Najmanji element	ako za svako $y \in A$ važi $x \leq y$;
Najveći element	ako za svako $y \in A$ važi $y \leq x$.

Za podskup $B \subseteq A$, element $x \in A$ naziva se ***donja granica*** skupa B ako za svako $y \in B$ važi $x \leq y$. Slično, element $x \in A$ naziva se ***gornja granica*** skupa B ako za svako $y \in B$ važi $y \leq x$.

Najveća donja granica skupa B, ukoliko postoji, naziva se ***infimum*** skupa B i označava se $\bigwedge B$ ili $\inf B$. Najmanja gornja granica skupa B, ukoliko postoji, naziva se ***supremum*** skupa B i označava se $\bigvee B$ ili $\sup B$. Infimum i supremum, ukoliko postoje, jedinstveni su. Ako skup B ima najmanji element (odnosno najveći element), tada je njegov najmanji element ujedno i infimum (odnosno njegov najveći element ujedno i supremum). Dakle, postojanje najmanjeg elementa (odnosno, najvećeg elementa) povlači postojanje infimuma (odnosno supremuma).

Primer 1.2 *Posmatrajmo skup $A = \{2, 3, 5, 6, 10, 15, 30\}$ uređen relacijom deljivosti. Definišimo relaciju uređenja $\leq$ na skupu A tako da za $a, b \in A$ važi $a \leq b$ ako i samo ako $a \mid b$ (a deli b). Ova relacija je refleksivna, antisimetrična i tranzitivna, te definiše parcijalno uređenje na A. Na Slici 4 prikazan je Hasseov dijagram ovog skupa.*

Elementi 2, 3 *i* 5 *su minimalni, jer nijedan od njih nije deljiv drugim elementom skupa osim samim sobom. Međutim, skup A nema najmanji element, pošto ne postoji element koji deli sve ostale elemente skupa.*

Element 30 *je najveći element. Budući da je svaki element skupa A delilac broja* 30*, element* 30 *je ujedno maksimum, pa samim tim i supremum skupa.*

Parcijalno uređen skup $(A, \leq)$ naziva se $\bigwedge$***-polumreža*** (***infimum polumreža***) ako svaka dva njegova elementa imaju infimum. Analogno, $(A, \leq)$ je $\bigvee$***-polumreža*** (***supremum polumreža***) ako svaka dva elementa imaju supremum. Ako svaki podskup skupa A ima infimum, tada se $(A, \leq)$ naziva ***kompletna*** $\bigwedge$***-polumreža*** (***kompletna infimum polumreža***). Analogno, ako svaki podskup skupa A ima supremum, tada se $(A, \leq)$ naziva ***kompletna*** $\bigvee$***-polumreža*** (***kompletna supremum polumreža***).

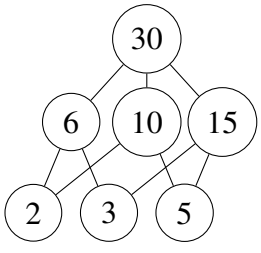
Mreža je parcijalno uređen skup u kome svaka dva elementa imaju infimum i supremum. ***Kompletna mreža*** je parcijalno uređen skup u kome za svaki podskup postoje infimum i supremum.

Svaka kompletna mreža poseduje najmanji i najveći element, koji predstavljaju infimum i supremum celog skupa. U opštem slučaju mreža ne mora biti ograničena, odnosno ne mora imati najmanji i najveći element. Ukoliko mreža poseduje najmanji i najveći element, naziva se ***ograničena mreža***.

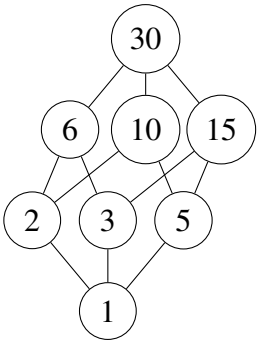
Primer 1.3 *Posmatrajmo ponovo skup iz Primera 1.2. Kako elementi* 2 *i* 3 *nemaju najveću donju granicu, infimum para* $\{2, 3\}$ *ne postoji (njihov jedini zajednički delilac je broj* 1*, ali* $1 \notin A$*). Stoga, parcijalno uređen skup A nije mreža.*

Dodajmo sada skupu A element 1 *i posmatrajmo skup* $A' = \{1, 2, 3, 5, 6, 10, 15, 30\}$*, uređen istom relacijom deljivosti. Hasseov dijagram dobijenog parcijalno uređenog skup prikazan je na Slici 5.*

Za svaki par elemenata skupa A' *postoje najveća donja granica i najmanja gornja granica. Prema tome, skup* A' *predstavlja mrežu. Štaviše, kako svaki podskup skupa* A' *ima infimum i supremum, skup* A' *predstavlja kompletn mrežu.*



Slika 4: Hasseov dijagram za Primer 1.2



Slika 5: Hasseov dijagram za Primer 1.3

Kao što je prethodno pokazano, pojam mreže može se definisati polazeći od parcijalno uređenih skupova. Međutim, mreže se mogu posmatrati i kao algebarske strukture definisane odgovarajućim binarnim operacijama. Ova dva pristupa su ekvivalentna, u smislu da svaka mreža definisana pomoću parcijalnog uređenja određuje odgovarajuću algebarsku strukturu i obrnuto. Algebarski pristup definisanju mreža biće predstavljen u Sekciji 1.3.

1.2 Osnovne algebarske strukture

U ovoj sekciji uvode se osnovne algebarske strukture koje će biti korišćene u nastavku rada. Najpre se razmatraju grupoidi, polugrupe, monoidi i grupe kao strukture definisane jednom binarnom operacijom, a zatim poluprsteni i prsteni kao strukture sa dve binarne operacije. Biće predstavljene i pojedine važne klase poluprstena, uključujući uređene poluprstene, dioide i kompletne dioide. Osnovni pojmovi i rezultati koji se navode u ovoj sekciji mogu se pronaći u standardnoj literaturi iz algebre i teorije poluprstena [2, 21, 22, 23, 27, 28, 29, 30, 31, 32, 33].

Neka je G neprazan skup i $n \in \mathbb{N}$. Preslikavanje $f : G^n \to G$ naziva se *n-arna operacija* ili sakraćeno ***operacija*** na skupu G. Specijalno, kada je $n = 1$, f je *unarna operacija*, a kada je $n = 2$ onda je u pitanju *binarna operacija*.

Za binarnu operaciju $*$ na skupu G kažemo da je:

Asocijativna ako je $(x * y) * z = x * (y * z)$, za svako $x, y, z \in G$;

Komutativna ako je $x * y = y * x$, za svako $x, y \in G$;

Idempotentna ako je $x * x = x$, za svako $x \in G$.

Dodavanjem novih svojstava binarnoj operaciji definišu se složenije algebarske strukture, kao što su polugrupe, monoidi i grupe.

Grupoid je uređeni par $(G, *)$ gde je G neprazan skup, a $* : G \times G \to G$ binarna operacija na skupu G. Operacija $*$ je zatvorena na G, tj. za sve $x, y \in G$ važi $x * y \in G$. Grupoid $(G, *)$ je ***komutativan grupoid*** (resp. ***idempotentan grupoid***) ako je operacija $*$ komutativna (resp. idempotentna).

Za element $e \in G$ kažemo da je levi (resp. desni) neutralni element grupoida ako i samo ako je $e * x = x$ (resp. $x * e = x$), za svako $x \in G$. Ako važi $e * x = x * e = x$ za svako $x \in G$, tada je e ***neutralni element*** grupoida $(G, *)$. Neutralni element, ukoliko postoji, jedinstven je.

Polugrupa je uređeni par $(G, *)$ gde je G neprazan skup, a $*$ zatvorena i asocijativna binarna operacija na G. Drugim rečima, polugrupa je asocijativni grupoid. Polugrupa $(G, *)$ je ***komutativna polugrupa*** (resp. ***idempotentna polugrupa***) ako je operacija $*$

komutativna (resp. idempotentna).

Element $z \in G$ je *leva nula* (resp. *desna nula*) ako važi $z * x = z$ (resp. $x * z = z$), za svako $x \in G$. Ako je $z * x = x * z = z$, za sve $x \in G$, tada je z ***nula polugrupe***. Ukoliko nula polugrupe postoji, jedinstvena je.

Ako se postojećem skupu G doda novi element $z \notin G$ koji zadovoljava zakon apsorpcije: $x * z = z * x = z$ za sve $x \in G$ i $z * z = z$, dobija se prošireni skup $G^0 = G \cup \{z\}$, a $(G^0, *)$ se naziva ***nulto proširenje*** polugrupe $(G, *)$, pri čemu je z nula te polugrupe.

Kao što je prethodno pokazano, $\bigwedge$-polumreže mogu se definisati kao parcijalno uređeni skupovi u kojima svaki par elemenata ima infimum, dok se $\bigvee$-polumreže definišu kao parcijalno uređeni skupovi u kojima svaki par elemenata ima supremum. Naredna definicija daje njihovu ekvivalentnu algebarsku karakterizaciju.

Polumreža je polugrupa $(G, *)$ u kojoj je binarna operacija $*$ komutativna i idempotentna. Polumreža $(G, *)$ predstavlja ***lanac*** ako je odgovarajuće prirodno uređenje linearno, ekvivalentno, za svaka dva elementa $x, y \in G$ važi $x * y = x$ ili $x * y = y$.

Pregled do sada uvedenih algebarskih struktura dat je u Tabeli 1.

Tabela 1: Pregled osnovnih osobina algebarskih struktura

Struktura $(G, *)$	**Zatvorena** $x * y \in G$	**Komutativna** $x * y = y * x$	**Idempotentna** $x * x = x$	**Asocijativna** $(x * y) * z = x * (y * z)$
Grupoid	DA			
Komutativni grupoid	DA	DA		
Idempotentni grupoid	DA		DA	
Polugrupa	DA			DA
Komutativna polugrupa	DA	DA		DA
Idempotentna polugrupa	DA		DA	DA
Polumreža	DA	DA	DA	DA

Ako polugrupa $(G, *)$ ima neutralni element, on se naziva ***jedinica polugrupe***. Svaka jedinica je idempotentna ($j * j = j$), a ukoliko postoji, jedinstvena je.

Monoid je uređena trojka $(G, *, 1)$ gde je G neprazan skup, $*$ binarna operacija na G i $1 \in G$ neutralan element (jedinica), takva da je $(G, *)$ polugrupa i $1 * x = x * 1 = x$, za

sve $x \in G$. Drugim rečima, monoid je polugrupa sa jediničnim (neutralnim) elementom.

Monoid $(G, *, 1)$ je ***komutativan monoid*** ako je $*$ komutativna operacija, dok je ***idempotentan monoid***, ako je $*$ idempotentna operacija.

Monoid se može konstruisati i *jediničnim proširenjem* polugrupe. Neka je data polugrupa $(G, *)$ i element $j \notin G$. Ako na proširenom skupu $G \cup \{j\}$ važi $j * a = a * j = a$ i $j * j = j$, za sve $a \in G$, tada $(G \cup \{j\}, *)$ predstavlja ***jedinično proširenje*** polugrupe $(G, *)$ i $(G \cup \{j\}, *, j)$ predstavlja monoid.

Neka je dat monoid $(G, *, 1)$. ***Podmonoid*** je podskup $P \subseteq G$ koji sadrži neutralni element i zatvoren je u odnosu na operaciju $*$, odnosno za svaka $x, y \in P$ važi $x * y \in P$.

Svaki podmonoid nekog monoida i sam je monoid. Najmanji podmonoid koji sadrži dati podskup $P \subseteq G$ dobija se kao presek svih podmonoida koji sadrže P. Ako se u preseku dobije ceo skup G, tada kažemo da je G *generisan* skupom P.

Poseban značaj u teoriji automata ima slobodni monoid, odnosno monoid reči. Ovoj strukturi biće posvećena posebna pažnja u Sekciji 2.1.

Grupa je uređena trojka $(G, *, e)$ gde je $(G, *, e)$ monoid i za svaki element $x \in G$ postoji element $x^{-1} \in G$ takav da važi $x * x^{-1} = x^{-1} * x = e$. Drugim rečima, grupa je monoid u kome svaki element ima inverzni element.

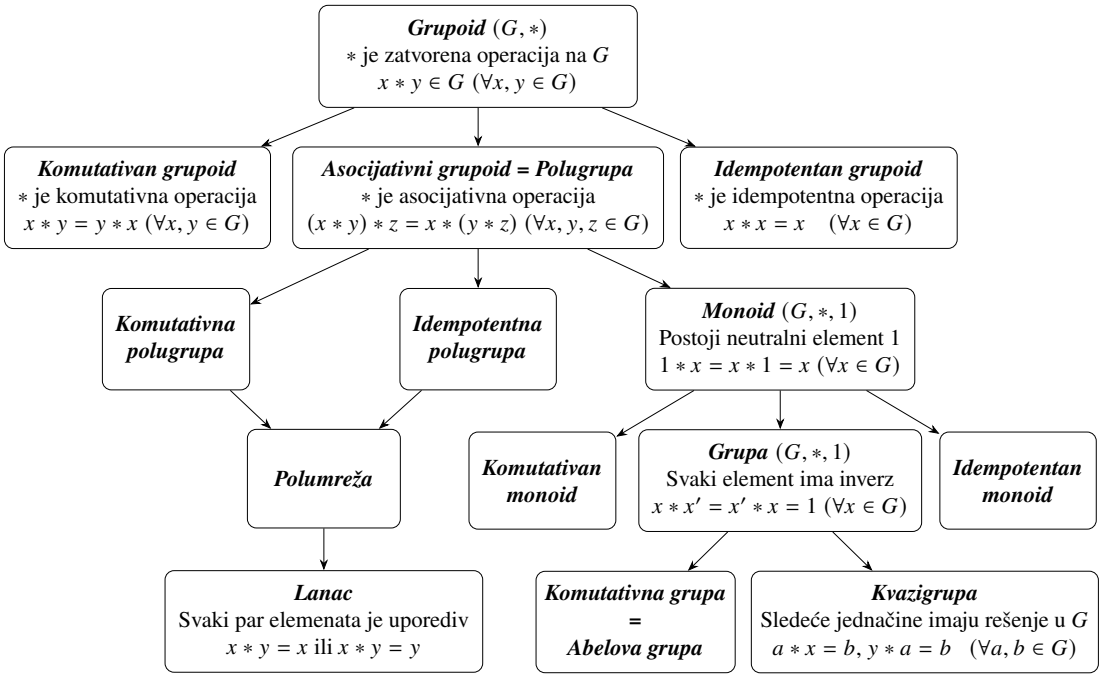
Grupa je ***komutativna grupa*** ili ***Abelova grupa*** ako je operacija $*$ komutativna.

Ako je skup G konačan (resp. beskonačan), tada je grupa *konačna* (resp. *beskonačna*).

Kvazigrupa je uređeni par $(G, *)$ gde je G neprazan skup, a $*$ binarna operacija na G takva da za sve $a, b \in G$ postoje jedinstveni elementi $x, y \in G$ za koje važi $x * a = b \quad \text{i} \quad a * y = b$.

Može se zaključiti da grupa predstavlja najbogatiju među do sada razmatranim algebarskim strukturama, jer objedinjuje osobine grupoida, polugrupa i monoida uz dodatni zahtev postojanja inverznog elementa. Međusobni odnosi između navedenih struktura prikazani su hijerarhijskim dijagramom na Slici 6.

U nastavku razmatramo poluprstene, algebarske strukture sa dve binarne operacije koje predstavljaju prirodno proširenje do sada uvedenih struktura. Poluprsteni imaju značajnu ulogu u savremenoj matematici i računarstvu, a posebno u teoriji težinskih automata, gde obezbeđuju matematički okvir za definisanje i kombinovanje težina pridruženih prelazima automata. Zbog svojih algebarskih osobina nalaze primenu i u teoriji formalnih jezika, optimizaciji, teoriji grafova i drugim oblastima [3, 31, 34, 35].



Slika 6: Hijerarhija osnovnih algebarskih struktura

Poluprsten je uređena petorka $(S, \oplus, \otimes, 0, 1)$, gde je S neprazan skup, $\oplus$ i $\otimes$ binarne operacije na S, a $0, 1 \in S$ elementi takvi da važe sledeći uslovi:

- $(S, \oplus, 0)$ je komutativan monoid.
- $(S, \otimes, 1)$ je monoid.
- Operacija množenja je distributivna u odnosu na sabiranje:
  $x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z)$
  $(x \oplus y) \otimes z = (x \otimes z) \oplus (y \otimes z)$, za sve $x, y, z \in S$.
- 0 je apsorbujući element:
  $x \otimes 0 = 0 \otimes x = 0$, za svako $x \in S$.

U zavisnosti od dodatnih osobina razlikuju se različite klase poluprstena. Najčešće razmatrane klase navedene su u nastavku.

Poluprsten je ***komutativan poluprsten*** ako je množenje komutativno tj. ukoliko je $(S, \otimes, 1)$ komutativan monoid.

Poluprsten je ***bez delilaca nule*** ako iz $x \otimes y = 0$ sledi $x = 0$ ili $y = 0$, za sve $x, y \in S$.

Primer 1.4 *Razmotrimo skup nenegativnih celih brojeva* $\mathbb{N}_0 = \{0, 1, 2, \dots\}$ *sa standardnim sabiranjem i množenjem. Tada struktura* $(\mathbb{N}_0, +, \cdot, 0, 1)$ *čini poluprsten.*

Zaista, $(\mathbb{N}_0, +, 0)$ *je komutativan monoid, dok je* $(\mathbb{N}_0, \cdot, 1)$ *monoid. Množenje je distributivno u odnosu na sabiranje, a nula je apsorbujući element.*

Dakle, $(\mathbb{N}_0, +, \cdot, 0, 1)$ *je komutativan poluprsten bez delilaca nule.*

Uređeni poluprsten je poluprsten $(S, \oplus, \otimes, 0, 1)$ na kome je definisano parcijalno uređenje $\leq$ kompatibilno sa operacijama sabiranja i množenja. Preciznije, poluprsten $(S, \oplus, \otimes, 0, 1)$ je uređen ukoliko postoji parcijalno uređenje $\leq$ na skupu S takvo da važi:

(c$\oplus$) za sve $x, y, z \in S$, iz $x \leq y$ sledi $x \oplus z \leq y \oplus z$ i $z \oplus x \leq z \oplus y$

(c$\otimes$) za sve $x, y, z \in S$, iz $x \leq y$ i $0 \leq z$ sledi $x \otimes z \leq y \otimes z$ i $z \otimes x \leq z \otimes y$.

Ako su svi elementi uređenog poluprstena pozitivni, odnosno ako važi $0 \leq x$ za svako $x \in S$, tada se takav poluprsten naziva ***pozitivan poluprsten***.

Poluprsten je ***aditivno-idempotentan poluprsten*** ili ***dioid*** ako je sabiranje idempotentna operacija, tako da važi $x \oplus x = x$ za svako $x \in S$, odnosno ako je $(S, \oplus)$ idempotentna polugrupa. Kod svakog dioida, operacija sabiranja indukuje prirodno parcijalno uređenje definisano sa

$$x \leq y \iff x \oplus y = y, \quad \text{za sve } x, y \in S,$$

što se u literaturi naziva ***kanoničko parcijalno uređenje*** (engl. *canonical partial order*) na S. Ovo uređenje kompatibilno je sa operacijama $\oplus$ i $\otimes$, a element 0 (nula poluprstena) predstavlja najmanji element u odnosu na ovo uređenje. Drugim rečima, kanonički uređen poluprsten je pozitivan poluprsten.

Operacija sabiranja u posmatranom dioidu odgovara supremumu dva elementa, odnosno važi $x \oplus y = \sup\{x, y\}$. Drugim rečima, struktura $(S, \oplus)$ predstavlja gornju polumrežu. Ako svaki podskup skupa S ima supremum u odnosu na prirodno uređenje, tada se $(S, \oplus)$ naziva ***kompletnom gornjom polumrežom***.

Za proizvoljnu familiju elemenata $\{a_i\}_{i \in I} \subseteq S$ njen supremum označava se sa

$$\bigvee_{i \in I} a_i = \bigoplus_{i \in I} a_i,$$

pri čemu se izraz na desnoj strani posmatra kao beskonačna suma elemenata a_i. Dioid u kome je za svaku familiju elemenata $\{a_i\}_{i \in I} \subseteq S$ definisana ovakva beskonačna suma naziva se ***dioid koji dopušta beskonačne sume***.

Kompletan dioid (kompletan aditivno-idempotentan poluprsten) je dioid koji dopušta beskonačne sume i u kome važi beskonačni distributivni zakon:

$$x \otimes \left(\bigoplus_{i \in I} a_i \right) = \bigoplus_{i \in I} (x \otimes a_i), \quad \text{i} \quad \left(\bigoplus_{i \in I} a_i \right) \otimes x = \bigoplus_{i \in I} (a_i \otimes x),$$

za svako $x \in S$ i svaku familiju $\{a_i\}_{i \in I} \subseteq S$.

Svaki kompletan aditivno-idempotentan poluprsten sa jedinicom naziva se ***kvantal sa jedinicom*** (engl. *unital quantale*).

Primetimo da poluprsten iz Primera 1.4 ne predstavlja dioid, jer operacija sabiranja nije idempotentna, odnosno u opštem slučaju ne važi $x \oplus x = x$. Takođe, ova struktura nije

kompletna, jer proizvoljne beskonačne sume elemenata iz $\mathbb{N}_0$ ne moraju pripadati skupu $\mathbb{N}_0$ (npr. $\bigoplus_{n=1}^{\infty} n$ divergira).

Neka je $\mathbb{S} = (S, \oplus, \otimes, 0, 1)$ kompletan dioid. Tada je, u odnosu na prirodno uređenje indukovano operacijom $\oplus$, struktura $(S, \leq)$ kompletna mreža. Drugim rečima, svaka familija elemenata $\{a_i\}_{i \in I} \subseteq S$ ima supremum i infimum, pri čemu važi

$$\bigvee_{i \in I} a_i = \bigoplus_{i \in I} a_i, \qquad \bigwedge_{i \in I} a_i.$$

Dakle, supremum familije elemenata poklapa se sa njihovom beskonačnom sumom. Prema tome, svaki kompletan dioid, pored strukture poluprstena, poseduje i strukturu kompletne mreže, pri čemu operacija sabiranja istovremeno predstavlja operaciju supremuma.

Reziduacija u kompletnim dioidima zasniva se na određivanju najvećih rešenja nejednačina definisanih pomoću operacije množenja.

Neka je $\mathbb{S} = (S, \oplus, \otimes, 0, 1)$ kompletan dioid i neka su $a, b \in S$.

Nejednačina $a \otimes x \leq b$, gde je $x \in S$ nepoznat element, ima najveće rešenje. To rešenje naziva se ***levi rezidual*** elementa b po elementu a i označava se sa $a \backslash b$.

Analogno, nejednačina $x \otimes a \leq b$ ima najveće rešenje koje se naziva ***desni rezidual*** elementa b po elementu a i označava se sa b/a.

Operacije $\backslash$ i $/$ nazivaju se ***operacije reziduacije***. One su povezane sa operacijom množenja ***zakonom adjunkcije***:

$$a \otimes b \leq c \iff b \leq a \backslash c \iff a \leq c/b, \quad \text{za sve } a, b, c \in S. \tag{1}$$

Ukoliko je operacija množenja $\otimes$ komutativna (kao kod komutativnih aditivno-idempotentnih poluprstena), tada se levi i desni rezidual poklapaju. Tada je dovoljno posmatrati jednu operaciju reziduacije, koja se najčešće označava sa $\rightarrow$, pri čemu važi

$$a \rightarrow b = a \backslash b = b/a, \quad \text{za sve } a, b \in S. \tag{2}$$

U tom slučaju zakon adjunkcije poprima oblik

$$a \otimes b \leq c \iff a \leq b \rightarrow c \iff b \leq a \rightarrow c, \quad \text{za sve } a, b, c \in S. \tag{3}$$

Operacija $\rightarrow$ naziva se operacija reziduacije pridružena operaciji $\otimes$.

U kompletnom aditivno-idempotentnom poluprstenu je $0 \backslash a = a/0 = \top$, za svako $a \in S$, gde je $\top$ najveći element skupa S, a $0 \in S$ je nula posmatranog poluprstena.

Prsten je petorka $\mathbb{S} = (S, \oplus, \otimes, 0, 1)$, gde je S neprazan skup, $\oplus$ i $\otimes$ binarne operacije na S, a $0, 1 \in S$ elementi takvi da važi:

- $(S, \oplus, 0)$ je komutativna grupa.
- $(S, \otimes, 1)$ je monoid.
- Množenje je distributivno u odnosu na sabiranje, odnosno za sve $x, y, z \in S$ važi:

$$x \otimes (y \oplus z) = x \otimes y \oplus x \otimes z,$$
$$(x \oplus y) \otimes z = x \otimes z \oplus y \otimes z.$$

Za razliku od poluprstena, uslov da je 0 apsorbujući element nije potrebno posebno navoditi, jer sledi iz ostalih aksioma prstena.

Prsten se može posmatrati kao specijalan slučaj poluprstena u kome svaki element poseduje aditivni inverz. Preciznije, svaki prsten je poluprsten, dok proizvoljan poluprsten ne mora biti prsten, jer struktura $(S, \oplus, 0)$ ne mora biti grupa.

Prsten je ***komutativan prsten*** ako je množenje komutativna operacija na S.

Ako iz $x \otimes y = 0$ sledi $x = 0$ ili $y = 0$, tada je prsten ***bez delilaca nule***.

Polje je komutativan prsten $(S, \oplus, \otimes, 0, 1)$ u kome svaki nenula element ima multiplikativni inverz, odnosno za svaki $x \in S \setminus \{0\}$ postoji $x^{-1} \in S$ takav da je $x \otimes x^{-1} = 1$. Drugim rečima, $(S, \oplus, 0)$ i $(S \setminus \{0\}, \otimes, 1)$ su komutativne grupe.

1.3 Mreže i reziduirane mreže

U ovoj sekciji razmatraju se algebarske strukture koje objedinjuju pojmove uređenja i binarnih operacija na skupovima. Nakon kratkog podsećanja na osnovne osobine polumreža i mreža, uvode se reziduirane mreže kao njihovo prirodno proširenje dodatnim algebarskim operacijama i relacijom reziduacije [21, 22, 23].

Reziduirane mreže imaju značajnu ulogu u teoriji fazi skupova, fazi relacija i viševrednosnih logika. Njihov značaj za ovaj rad ogleda se u tome što obezbeđuju algebarski okvir za definisanje operacija nad fazi relacijama i fazi automatima, uključujući operacije reziduacije, kao i različite oblike simulacija i bisimulacija koji će biti razmatrani u narednim poglavljima [36, 37, 38].

Polumreža je uređeni par $(L, *)$ gde je L neprazan skup, a $*$ binarna operacija na L koja je asocijativna, komutativna i idempotentna. Drugim rečima, polumreža je komutativna idempotentna polugrupa.

Ograničena polumreža je uređena trojka $(L, *, e)$ gde je $(L, *)$ polumreža, a $e \in L$ neutralni element za operaciju $*$, tj. $x * e = e * x = x$, za svako $x \in L$. Drugim rečima, ograničena polumreža je komutativan idempotentan monoid.

Mreže su već uvedene u Sekciji 1.1 u okviru teorije parcijalno uređenih skupova, gde su definisane pomoću postojanja supremuma i infimuma. U nastavku navodimo njihovu ekvivalentnu algebarsku karakterizaciju.

Mreža je uređena trojka $(L, \vee, \wedge)$ gde je L neprazan skup, a $\vee$ i $\wedge$ binarne operacije na L koje za sve $x, y, z \in L$ zadovoljavaju:

Idempotentnost $x \vee x = x$ i $x \wedge x = x$;

Komutativnost $x \vee y = y \vee x$ i $x \wedge y = y \wedge x$;

Asocijativnost $(x \vee y) \vee z = x \vee (y \vee z)$ i $(x \wedge y) \wedge z = x \wedge (y \wedge z)$;

Apsorptivnost $x \vee (x \wedge y) = x$ i $x \wedge (x \vee y) = x$;

Operacije $\vee$ i $\wedge$ odgovaraju operacijama supremuma i infimuma.

Za elemente $x, y \in L$ definišemo relaciju

$$x \leq y \iff x \vee y = y \iff x \wedge y = x.$$

Može se pokazati da je tako definisana relacija $\leq$ parcijalno uređenje na skupu L. Štaviše, za svaka dva elementa $x, y \in L$ operacije $\vee$ i $\wedge$ određuju njihov supremum i infimum u odnosu na ovo uređenje. Prema tome, uređeni skup $(L, \leq)$ predstavlja mrežu, pri čemu operacije $\vee$ i $\wedge$ određuju supremum i infimum svaka dva elementa. Na taj način algebarska definicija mreže $(L, \vee, \wedge)$ ekvivalentna je definiciji mreže pomoću parcijalno uređenih skupova.

Mreža $(L, \vee, \wedge)$ je ***distributivna*** ako za sve $x, y, z \in L$ važi

$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z) \quad \text{i} \quad x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z).$$

Element $0 \in L$ naziva se najmanjim elementom mreže (nulom mreže) ako za svako $x \in L$ važ $0 \vee x = x$ i $0 \wedge x = 0$. Analogno, poseduje jedinicu $1 \in L$, ako za sve $x \in L$ važi $1 \vee x = 1$ i $1 \wedge x = x$. Ako mreža poseduje i 0 i 1, tada se struktura $(L, \vee, \wedge, 0, 1)$ naziva ***ograničena mreža***.

Ograničena mreža $(L, \vee, \wedge, 0, 1)$ je ***komplementarna mreža*** ako za svaki $x \in L$ postoji $x' \in L$ takav da je $x \vee x' = 1$ i $x \wedge x' = 0$.

Primer 1.5 *Posmatrajmo skup* $B = \{0, 1\}$ *sa operacijama* $\vee$ *i* $\wedge$*, koje predstavljaju logičku disjunkciju i konjunkciju. Tada je struktura* $(B, \vee, \wedge, 0, 1)$ *distributivna komplementarna ograničena mreža, poznata kao* ***Bulova mreža****.*

U ovoj mreži element 0 *predstavlja najmanji, a element* 1 *najveći element. Pored toga, svaki element ima svoj komplement, pri čemu važi* $0' = 1$ *i* $1' = 0$*.*

Bulove mreže predstavljaju najjednostavniji primer distributivnih komplementarnih mreža i imaju značajnu ulogu u matematičkoj logici, teoriji skupova i teoriji algebarskih struktura. Mnoge njihove osobine mogu se posmatrati kao specijalni slučajevi osobina reziduiranih mreža, koje će biti uvedene u nastavku.

Reziduirane mreže predstavljaju prirodno proširenje distributivnih mreža uvođenjem dodatne operacije $\otimes$ i njoj odgovarajuće operacije reziduacije $\rightarrow$. Njihov međusobni odnos opisan je zakonom adjunkcije (videti (1) i (3)), koji predstavlja jedno od osnovnih svojstava reziduiranih mreža.

Neka je L neprazan skup sa binarnim operacijama $\vee$, $\wedge$, $\otimes$ i $\rightarrow$, i elementima $0, 1 \in L$. Struktura $\mathbb{L} = (L, \vee, \wedge, \otimes, \rightarrow, 0, 1)$ naziva se ***reziduirana mreža*** ako važi:

$(L, \vee, \wedge, 0, 1)$	je ograničena distributivna mreža;
$(L, \otimes, 1)$	je komutativan monoid;
$x \otimes y \leq z \iff x \leq y \rightarrow z$	za sve $x, y, z \in L$ važi zakon adjunkcije, gde je $\leq$ uređenje na mreži$(L, \vee, \wedge, 0, 1)$;

Reziduirana mreža naziva se ***kompletna reziduirana mreža*** ukoliko je $(L, \vee, \wedge, 0, 1)$ kompletna ograničena mreža.

Reziduirana mreža je ***idempotentna*** ako za sve $x \in L$ važi $x \otimes x = x$.

Za dokaz svojstava prikazanih u predstojećim propozicijama videti [38, 39].

Propozicija 1.1 *Neka je* $\mathbb{L} = (L, \vee, \wedge, \otimes, \rightarrow, 0, 1)$ *kompletna reziduirana mreža. Za sve* $x, y, z, w \in L$ *važe sledeće osobine:*

$$x \rightarrow x = 1; \quad 1 \rightarrow x = x; \quad x \otimes 0 = 0; \quad x \rightarrow 1 = 1; \quad 0 \rightarrow x = 1; \tag{4}$$

$$x \otimes (x \rightarrow y) \leq y; \quad y \leq x \rightarrow (x \otimes y); \quad x \leq (x \rightarrow y) \rightarrow y; \tag{5}$$

$$x \leq y \Rightarrow x \otimes z \leq y \otimes z; \quad x \leq y \Rightarrow z \otimes x \leq z \otimes y; \tag{6}$$

$$x \otimes y \leq x; \quad x \leq y \rightarrow x; \quad x \leq y \iff x \rightarrow y = 1; \tag{7}$$

$$(x \rightarrow y) \otimes (z \rightarrow w) \leq (x \otimes z) \rightarrow (y \otimes w) \tag{8}$$

$$x \otimes (x \rightarrow y) = y \text{ , ako i samo ako postoji } z \text{ takav da je } x \otimes z = y \tag{9}$$

Propozicija 1.2 *U svakoj kompletnoj reziduiranoj mreži* $(L, \vee, \wedge, \otimes, \rightarrow, 0, 1)$ *važi:*

(a) Operacija $\otimes$ *je rastuća po oba argumenta:*
ako je $x_1 \leq x_2$ *i* $y_1 \leq y_2$, *tada važi* $x_1 \otimes y_1 \leq x_2 \otimes y_2$

(b) Operacija $\rightarrow$ *je opadajuća po prvom i rastuća po drugom argumentu:*
ako je $x_1 \leq x_2$ *i* $y_1 \leq y_2$, *tada važi* $x_2 \rightarrow y_1 \leq x_1 \rightarrow y_2$.

Propozicija 1.3 *U svakoj kompletnoj reziduiranoj mreži* $(L, \vee, \wedge, \otimes, \rightarrow, 0, 1)$ *važe sledeća svojstva:*

$$x \rightarrow y = \bigvee \{z \in L \mid x \otimes z \leq y\} \tag{10}$$

$$x \otimes y = \bigwedge \{z \in L \mid x \leq y \rightarrow z\} \tag{11}$$

Propozicija 1.4 *U svakoj kompletnoj reziduiranoj mreži* $(L, \vee, \wedge, \otimes, \rightarrow, 0, 1)$*, važe sledeća svojstva, za svako* $\{x_i\}_{i\in I}, \{y_i\}_{i\in I} \subseteq L$*:*

$$x \otimes \bigvee_{i\in I} y_i = \bigvee_{i\in I} (x \otimes y_i), \qquad x \otimes \bigwedge_{i\in I} y_i \leq \bigwedge_{i\in I} (x \otimes y_i), \tag{12}$$

$$x \rightarrow \bigwedge_{i\in I} y_i = \bigwedge_{i\in I} (x \rightarrow y_i), \qquad \bigvee_{i\in I} x_i \rightarrow y = \bigwedge_{i\in I} (x_i \rightarrow y), \tag{13}$$

$$\bigvee_{i\in I} (x \rightarrow y_i) \leq x \rightarrow \bigvee_{i\in I} y_i, \qquad \bigvee_{i\in I} (x_i \rightarrow y) \leq \bigwedge_{i\in I} x_i \rightarrow y, \tag{14}$$

$$\bigwedge_{i\in I} (x_i \rightarrow y_i) \leq \bigwedge_{i\in I} x_i \rightarrow \bigwedge_{i\in I} y_i, \qquad \bigvee_{i\in I} (x_i \rightarrow y_i) \leq \bigvee_{i\in I} x_i \rightarrow \bigvee_{i\in I} y_i. \tag{15}$$

Neka je $(L, \vee, \wedge, \otimes, \rightarrow, 0, 1)$ reziduirana mreža. Binarna operacija ***birezidual*** na L predstavlja meru ekvivalencije između istinitosnih vrednosti i definiše se kao:

$$x \leftrightarrow y = (x \rightarrow y) \wedge (y \rightarrow x), \quad (\forall x, y \in L).$$

Negacija na skupu L modeluje dopunu (komplement) za datu istinitosnu vrednost i definiše se kao:

$$\neg x = x \rightarrow 0, \quad (\forall x \in L).$$

Propozicija 1.5 *U svakoj kompletnoj reziduiranoj mreži* $(L, \vee, \wedge, \otimes, \rightarrow, 0, 1)$*,osnovna svojstva bireziduala su:*

$$0 \leftrightarrow 1 = 1 \leftrightarrow 0 = 0, \quad 0 \leftrightarrow 0 = 1 \leftrightarrow 1 = 1 \tag{16}$$

$$x \leftrightarrow x = 1 \tag{17}$$

$$x \leftrightarrow y = y \leftrightarrow x \tag{18}$$

$$(x \leftrightarrow y) \otimes (y \leftrightarrow z) \leq x \leftrightarrow z \tag{19}$$

$$x \leftrightarrow 1 = x, \quad x \leftrightarrow 0 = \neg x \tag{20}$$

$$x \leftrightarrow y = (x \vee y) \rightarrow (x \wedge y) \tag{21}$$

$$\bigwedge_{i\in I} (x_i \leftrightarrow y_i) \leq (\bigwedge_{i\in I} x_i) \leftrightarrow (\bigwedge_{i\in I} y_i), \quad \bigwedge_{i\in I} (x_i \leftrightarrow y_i) \leq (\bigwedge_{i\in I} x_i) \leftrightarrow (\bigvee_{i\in I} y_i) \tag{22}$$

Jedna od najznačajnijih klasa reziduiranih mreža dobija se na intervalu $[0, 1]$, gde se operacija množenja definiše pomoću t-normi. T-norme predstavljaju osnovu za modelovanje konjunkcije u fazi logici i teoriji fazi skupova, dok reziduirane mreže koje one indukuju imaju značajnu ulogu u teoriji fazi relacija i fazi automata.

T-norma ili *trougaona norma* je binarna operacija $\otimes : [0, 1] \times [0, 1] \rightarrow [0, 1]$ koja zadovoljava sledeće uslove:

- $([0, 1], \otimes, 1)$ je komutativan monoid.
- $\otimes$ je monotona po oba argumenta, odnosno za sve $x_1, x_2, y_1, y_2 \in [0, 1]$ važi $x_1 \leq x_2$ i $y_1 \leq y_2 \Rightarrow x_1 \otimes y_1 \leq x_2 \otimes y_2$.

T-norme predstavljaju prirodnu generalizaciju operacije preseka skupova i često se interpretiraju kao operacije konjunkcije u odgovarajućim viševrednosnim logikama. U teoriji fazi skupova i fazi relacija, one omogućavaju kombinovanje stepena pripadnosti elemenata.

Poseban značaj imaju levo neprekidne T-norme, jer omogućavaju definisanje odgovarajuće operacije reziduacije. Za levo neprekidnu T-normu $\otimes$ rezidual se definiše sa

$$x \to y = \bigvee \{ z \in [0,1] \mid x \otimes z \leq y \},$$

pri čemu struktura $([0,1], \max, \min, \otimes, \to, 0, 1)$ predstavlja kompletnu reziduiranu mrežu.

Reziduirane mreže indukovane T-normama predstavljaju najčešće korišćene strukture vrednosti u teoriji fazi skupova, fazi relacija i fazi automata. Njihov značaj proizlazi iz činjenice da svaka kompletna reziduirana mreža prirodno indukuje odgovarajuću strukturu poluprstena. Naime, ako je $\mathbb{L} = (L, \vee, \wedge, \otimes, \to, 0, 1)$ kompletna reziduirana mreža, tada struktura $\mathbb{S} = (L, \vee, \otimes, 0, 1)$ predstavlja komutativan aditivno-idempotentan poluprsten.

Na taj način reziduirane mreže obezbeđuju prirodan algebarski okvir za definisanje i proučavanje težinskih i fazi automata.

Najčešće korišćene reziduirane mreže na intervalu $[0, 1]$ dobijaju se različitim izborom operacije množenja $\otimes$ i odgovarajućeg reziduala $\to$. Za svaku od ovih struktura operacije supremuma i infimuma definisane su standardno, odnosno kao maksimum i minimum: $x \vee y = \max\{x, y\}$ i $x \wedge y = \min\{x, y\}$.

U Tabeli 2 prikazane su operacije množenja $\otimes$, reziduala $\to$, bireziduala $\leftrightarrow$ i negacije $\neg$ za Łukasiewiczevu, Gödelovu i Goguenovu reziduiranu mrežu. Navedene reziduirane mreže predstavljaju najčešće korišćene strukture vrednosti u teoriji fazi skupova, fazi relacija i fazi automata. U narednoj sekciji biće uvedeni osnovni pojmovi fazi skupova i fazi relacija definisanih nad proizvoljnom kompletnom reziduiranom mrežom.

Tabela 2: Operacije u Łukasiewiczevoj, Gödelovoj i Goguenovoj reziduiranoj mreži

Operacija	**Łukasiewiczeva**	**Gödelova**	**Goguenova**
$x \otimes y$	$\max\{0, x + y - 1\}$	$\min\{x, y\}$	$x \cdot y$
$x \to y$	$\min\{1, 1 - x + y\}$	$\begin{cases} 1, & x \leq y \\ y, & x > y \end{cases}$	$\begin{cases} 1, & x \leq y \\ \frac{y}{x}, & x > y \end{cases}$
$x \leftrightarrow y$	$1 - \lvert x - y \rvert$	$\begin{cases} 1, & x = y \\ \min\{x, y\}, & x \neq y \end{cases}$	$\begin{cases} 1, & x = y = 0 \\ \frac{\min\{x, y\}}{\max\{x, y\}}, & \text{inače} \end{cases}$
$\neg x$	$1 - x$	$\begin{cases} 1, & x = 0 \\ 0, & x > 0 \end{cases}$	$\begin{cases} 1, & x = 0 \\ 0, & x > 0 \end{cases}$

1.4 Fazi skupovi i fazi relacije

Teoriju fazi skupova uveo je L. A. Zadeh 1965. godine kao proširenje klasične teorije skupova, u kojem pripadnost elementa skupu nije nužno binarna, već se izražava određenim stepenom pripadnosti [40]. Na taj način omogućeno je formalno modelovanje nepreciznosti i postepene pripadnosti, zbog čega je teorija fazi skupova našla široku primenu u matematici, logici, veštačkoj inteligenciji i računarstvu [37, 38, 41, 42].

Savremeni algebarski pristupi fazi skupovima i fazi logici zasnivaju se na kompletnim reziduiranim mrežama, gde se skup pripadnosti često modeluje pomoću kompletnih reziduiranih mreža. Takve strukture omogućavaju definisanje operacija konjunkcije, implikacije, negacije i drugih operacija nad fazi vrednostima, te pružaju prirodan matematički okvir za proučavanje fazi skupova i fazi relacija [38, 39, 43, 44].

Fazi relacije predstavljaju prirodno proširenje klasičnih relacija na fazi okruženje i imaju značajnu ulogu u modelovanju odnosa između objekata sa različitim stepenima povezanosti. U teoriji fazi automata one se koriste za definisanje simulacija i bisimulacija, proučavanje ekvivalentnosti automata i konstrukciju postupaka za redukciju broja stanja [6, 9, 45, 46, 47]. Posebno značajne su fazi relacije definisane nad kompletnim reziduiranim mrežama, kao što su Gödelova, Łukasiewiczeva i proizvod struktura, koje će biti korišćene u nastavku rada [37, 38, 41].

U nastavku neka je struktura $\mathbb{L} = (L, \vee, \wedge, \otimes, \rightarrow, 0, 1)$ kompletna reziduirana mreža i neka je A neprazan skup.

Fazi skup na skupu A je svako preslikavanje $\alpha : A \rightarrow L$. Za $a \in A$, vrednost $\alpha(a)$ predstavlja stepen pripadnosti elementa a fazi skupu α. Skup svih fazi skupova nad A označava se sa L^A.

Klasični skupovi mogu se posmatrati kao specijalan slučaj fazi skupova. Naime, svakom podskupu $B \subseteq A$ odgovara karakteristična funkcija $\chi_B : A \rightarrow \{0, 1\}$ definisana

$$\chi_B(a) = \begin{cases} 1, & a \in B, \\ 0, & a \notin B. \end{cases}$$

Na taj način, klasična pripadnost elementa skupu je poseban slučaj fazi pripadnosti.

Za fazi skupove $\alpha, \beta \in L^A$ jednakost i inkluzija definišu se pokoordinatno:

$$\alpha = \beta \iff \alpha(a) = \beta(a), \qquad \text{za svako } a \in A;$$

$$\alpha \leq \beta \iff \alpha(a) \leq \beta(a), \qquad \text{za svako } a \in A;$$

Za fazi skupove $\alpha, \beta \in L^A$ skupovne operacije se definišu pokoordinatno, za svako $a \in A$:

Fazi unija $(\alpha \vee \beta)(a) = \alpha(a) \vee \beta(a)$;

Fazi presek $(\alpha \wedge \beta)(a) = \alpha(a) \wedge \beta(a)$;

Fazi negacija (komplement) $(\neg\alpha)(a) = \alpha(a) \to 0$;

Za proizvoljnu familiju fazi skupova $\{\alpha_i\}_{i\in I} \subseteq L^A$, njihova fazi unija i fazi presek (odnosno supremum i infimum na mreži L^A) definišu se takođe pokoordinatno:

$$\left(\bigvee_{i\in I} \alpha_i\right)(a) = \bigvee_{i\in I} \alpha_i(a), \qquad \left(\bigwedge_{i\in I} \alpha_i\right)(a) = \bigwedge_{i\in I} \alpha_i(a).$$

Fazi relacija između skupova A i B je svako preslikavanje $\varphi : A \times B \to L$, odnosno fazi podskup skupa $A \times B$. Vrednost $\varphi(a, b)$ predstavlja stepen u kome je element $a \in A$ u relaciji sa elementom $b \in B$. Skup svih fazi relacija između A i B označava se sa $L^{A\times B}$.

Za fazi relaciju $\varphi \in L^{A\times B}$ njen *inverz* je fazi relacija $\varphi^{-1} \in L^{B\times A}$, definisana sa $\varphi^{-1}(b, a) = \varphi(a, b)$, za svako $a \in A$ i $b \in B$.

Za fazi relacije $\varphi, \psi \in L^{A\times B}$ *jednakost* i *inkluzija* definišu se pokoordinatno:

$$\varphi = \psi \iff \varphi(a, b) = \psi(a, b) \quad \text{i} \quad \varphi \le \psi \iff \varphi(a, b) \le \psi(a, b),$$

za sve $a \in A$ i $b \in B$.

Neka su $\varphi \in L^{A\times B}$ i $\psi \in L^{B\times C}$ fazi relacije, $\alpha \in L^A$ i $\beta \in L^B$ fazi skupovi, a A, B i C neprazni skupovi. Kompozicije fazi relacija $\varphi \circ \psi \in L^{A\times C}$, $\alpha \circ \varphi \in L^B$, $\varphi \circ \beta \in L^A$ i kompozicija fazi skupova $\alpha \circ \beta \in L$ definišu se na sledeći način:

$$(\varphi \circ \psi)(a, c) = \bigvee_{b\in B} \varphi(a, b) \otimes \psi(b, c), \quad a \in A,\ c \in C, \tag{23}$$

$$(\alpha \circ \varphi)(b) = \bigvee_{a\in A} \alpha(a) \otimes \varphi(a, b), \quad b \in B, \tag{24}$$

$$(\varphi \circ \beta)(a) = \bigvee_{b\in B} \varphi(a, b) \otimes \beta(b), \quad a \in A, \tag{25}$$

$$\alpha \circ \beta = \bigvee_{a\in A} \alpha(a) \otimes \beta(a). \tag{26}$$

Ako su skupovi A, B i C konačni, tada se fazi relacije mogu predstaviti matricama sa elementima iz L, dok se fazi skupovi mogu posmatrati kao vektori nad istom strukturom. U tom slučaju kompozicija fazi relacija definisana izrazom (23) odgovara množenju matrica. Analogno tome, kompozicije fazi skupa i fazi relacije (24) i fazi relacije i fazi skupa (25) odgovaraju proizvodu vektora i matrice, odnosno matrice i vektora, dok kompozicija fazi skupova (26) predstavlja skalarni proizvod dva vektora. Inverzna fazi relacija može se posmatrati kao transponovana matrica odgovarajuće fazi relacije. Sve navedene operacije

definisane su pomoću operacija $\bigvee$ i $\otimes$ u posmatranoj strukturi L.

Nije teško pokazati da za proizvoljne fazi relacije $\varphi \in L^{A\times B}$, $\chi \in L^{B\times C}$ i $\psi \in L^{C\times D}$, kao i fazi skupove $\alpha \in L^A$, $\beta \in L^B$ i $\gamma \in L^C$, važe sledeće jednakosti:

$$(\varphi \circ \chi) \circ \psi = \varphi \circ (\chi \circ \psi), \qquad (\alpha \circ \varphi) \circ \chi = \alpha \circ (\varphi \circ \chi),$$
$$(\varphi \circ \chi) \circ \gamma = \varphi \circ (\chi \circ \gamma), \qquad (\alpha \circ \varphi) \circ \beta = \alpha \circ (\varphi \circ \beta).$$

Ove jednakosti izražavaju asocijativnost kompozicije fazi relacija i fazi skupova, te možemo zaključiti da se sve dobro definisane kompozicije fazi relacija i fazi skupova mogu zapisivati bez upotrebe zagrada.

Fazi relacija $\varphi \in L^{A\times A}$ naziva se:

Refleksivna ako je $\varphi(a, a) = 1$ za svako $a \in A$;

Simetrična ako je $\varphi = \varphi^{-1}$;

Tranzitivna ako je $\varphi \circ \varphi \leq \varphi$;

Fazi relacija koja je refleksivna i tranzitivna naziva se ***fazi kvaziuređenje (fazi preorder)***, dok se refleksivna, simetrična i tranzitivna fazi relacija naziva ***fazi ekvivalencija***.

Pomoću operacije reziduacije u strukturi L mogu se definisati posebne fazi relacije indukovane fazi skupovima. Neka su $\alpha \in L^A$ i $\beta \in L^B$. Fazi relacija $\alpha\backslash\beta \in L^{A\times B}$ naziva se ***levi rezidual*** fazi skupa β po fazi skupu α i definisana je sa

$$(\alpha\backslash\beta)(a, b) = \alpha(a) \to \beta(b), \qquad \text{za sve } a \in A,\ b \in B. \tag{27}$$

Analogno tome, fazi relacija $\alpha/\beta \in L^{A\times B}$ naziva se ***desni rezidual*** fazi skupa α po fazi skupu β i definisana je sa

$$(\alpha/\beta)(a, b) = \beta(b) \to \alpha(a), \qquad \text{za sve } a \in A,\ b \in B. \tag{28}$$

Pored toga, definiše se i fazi relacija $\alpha \mid \beta \in L^{A\times B}$ koja se naziva ***birezidual*** fazi skupova α i β i data je sa

$$(\alpha \mid \beta)(a, b) = \alpha(a) \leftrightarrow \beta(b), \qquad \text{za sve } a \in A,\ b \in B. \tag{29}$$

Lako se proverava da levi i desni rezidual zadovoljavaju sledeća svojstva adjunkcije:

$$\alpha \circ \varphi \leq \beta \iff \varphi \leq \alpha\backslash\beta, \qquad \varphi \circ \beta \leq \alpha \iff \varphi \leq \alpha/\beta, \tag{30}$$

za sve $\alpha \in L^A$, $\beta \in L^B$ i $\varphi \in L^{A\times B}$.

Levi i desni rezidual predstavljaju osnovne alate za definisanje aproksimativnih odnosa između fazi skupova, dok birezidual obezbeđuje prirodan način za simetrično poređenje elemenata. U teoriji težinskih automata rezidualne operacije koriste se za formalizaciju simulacija i bisimulacija između stanja automata, omogućavajući kvantitativno poređenje njihovog ponašanja. Ovi koncepti biće od ključnog značaja u nastavku rada.

Glava 2

Formalni jezici i težinski automati

Teorija automata predstavlja jednu od osnovnih oblasti teorijskog računarstva koja proučava matematičke modele diskretnih sistema čije se ponašanje opisuje pomoću stanja i prelaza između njih. Prve ideje o apstraktnim mašinama koje obrađuju ulazne simbole kroz konačan broj stanja, potiču iz rada McCullocha i Pittsa [48], u kome je predstavljen logički model neuronske aktivnosti. Kleene je kasnije uveo regularne izraze i pokazao njihovu povezanost sa konačnim automatima, čime su postavljeni temelji teorije regularnih jezika [49]. Rabin i Scott su potom formalizovali koncept nedeterminističkih automata i pokazali njihovu ekvivalenciju sa determinističkim automatima, što predstavlja jedan od osnovnih rezultata klasične teorije automata [50].

Da bi se formalno opisali ulazi koje automati obrađuju, neophodno je najpre uvesti pojam alfabeta i reči nad datim alfabetom. Skup svih konačnih reči nad datim alfabetom, zajedno sa operacijom konkatenacije i praznom reči kao neutralnim elementom, formira strukturu slobodnog monoida. Formalni jezici, definisani kao podskupovi slobodnog monoida, predstavljaju matematički okvir za opisivanje skupova reči koje određeni automat može da raspozna [42, 51, 52].

U klasičnoj teoriji automata, prelazi između stanja imaju binarnu prirodu: za dati ulazni simbol prelaz između dva stanja ili postoji ili ne postoji [51, 53]. Ovakav pristup omogućio je razvoj bogate teorije formalnih jezika, ali u mnogim primenama nije dovoljno opisati samo postojanje prelaza, već je potrebno izraziti i dodatne kvantitativne osobine sistema.

Prirodno proširenje klasičnog modela predstavljaju težinski automati, kod kojih se prelazima, inicijalnim i završnim stanjima pridružuju vrednosti iz izabrane algebarske strukture. Na taj način moguće je modelovati različite karakteristike sistema, kao što su cena izvršavanja, trajanje događaja, verovatnoća prelaza ili stepen pripadnosti. Težinski automati prvi put su sistematski proučavani u radovima Schützenbergera [1], dok je njihova opšta algebarska teorija razvijena u okviru teorije automata nad poluprstenima [2, 3]. Zbog toga se ponašanje težinskog automata ne opisuje samo skupom prihvaćenih reči, već funkcijom koja svakoj reči pridružuje odgovarajuću vrednost iz posmatranog poluprstena.

U zavisnosti od izbora strukture nad skupom težina dobijaju se različite klase težinskih automata. Klasični konačni automati mogu se posmatrati kao specijalan slučaj težinskih automata nad Bulovim poluprstenom. Daljim uopštavanjem skupa vrednosti razvijeni su fazi automati, zasnovani na teoriji fazi skupova koju je uveo Zadeh [40], kao i druge

klase kvantitativnih automata. Posebnu pažnju posvećujemo max-plus automatima, koji predstavljaju jednu od najznačajnijih klasa težinskih automata i koriste se za modelovanje sistema u kojima se težine interpretiraju kao vremena izvršavanja, troškovi ili druge optimizacione veličine [13, 14, 15, 54].

Za proučavanje strukture i ponašanja automata koriste se različiti načini reprezentacije, koji su međusobno ekvivalentni sa stanovišta opisivanja njihovog ponašanja. Grafovska reprezentacija omogućava interpretaciju automata pomoću usmerenih težinskih grafova i pruža prirodan opis stanja, prelaza, putanja i ciklusa. Sa druge strane, matrična reprezentacija opisuje prelaze pomoću matrica nad odgovarajućim poluprstenom, što omogućava primenu algebarskih metoda i predstavlja osnovu za veliki deo teorijskih razmatranja u nastavku rada.

U ovom poglavlju detaljnije se razmatraju svi prethodno pomenuti koncepti. Najpre se uvode osnovni pojmovi formalnih jezika, uključujući slobodni monoid i operacije nad jezicima. Potom se definišu težinski automati, njihovo ponašanje i jezici koje raspoznaju. U nastavku se prikazuju grafovska i matrična reprezentacija težinskih automata, dok se završni deo poglavlja posvećuje posebnim klasama automata koje će imati značajnu ulogu u nastavku rada: klasičnim konačnim automatima, fazi automatima i max-plus automatima.

2.1 Slobodan monoid i formalni jezici

Formalni jezici predstavljaju jedan od osnovnih pojmova teorije automata i teorije formalnih jezika. Oni omogućavaju matematičko opisivanje skupova reči koje određeni sistem može da generiše, prihvati ili raspozna. Za njihovo formalno definisanje najpre se uvode pojmovi alfabeta i reči nad datim alfabetom. Skup svih konačnih reči nad alfabetom, zajedno sa operacijom konkatenacije i praznom reči kao neutralnim elementom, obrazuje slobodni monoid, koji predstavlja osnovnu algebarsku strukturu u teoriji formalnih jezika [42, 51, 53].

Alfabet je neprazan konačan skup X. Elementi skupa X nazivaju se *simboli* ili *slova* alfabeta. ***Reč*** nad alfabetom X je konačan niz simbola $u = x_1x_2 \dots x_n$, gde je $n \in \mathbb{N}_0$ i $x_i \in X$ za svako $i \in \{1, \dots, n\}$. Dužina reči u označava se sa $|u| = n$ i predstavlja broj simbola u toj reči. Reč dužine nula naziva se ***prazna reč*** i označava se sa e, pri čemu je $|e| = 0$.

Dve reči $u = x_1x_2 \dots x_n$ i $v = y_1y_2 \dots y_m$ su jednake ako i samo ako je

$$n = m \quad \text{i} \quad x_i = y_i \ \text{, za svako } i \in \{1, \dots, n\}.$$

Skup svih nepraznih reči nad alfabetom X označava se sa X^+, dok se skup svih reči nad alfabetom X, uključujući praznu reč, označava sa X^*, pri čemu važi $X^* = X^+ \cup \{e\}$.

Za simbol $x \in X$ koriste se oznake

$$x^0 = e, \qquad x^1 = x, \qquad x^k = \underbrace{xx\ldots x}_{k \text{ puta}}, \qquad \text{pa je } |x^k| = k.$$

Konkatenacija ili nadovezivanje reči je binarna operacija nad rečima definisana sa

$$uv = x_1 \ldots x_n y_1 \ldots y_m, \qquad \text{gde su } u = x_1 \ldots x_n \text{ i } v = y_1 \ldots y_m.$$

Konkatenacija se označava simbolom tačke $\cdot$ ili bez navođenja ikakvog simbola.

Prazna reč e predstavlja neutralni element za operaciju konkatenacije, odnosno za svako $u \in X^*$ važi $u \cdot e = e \cdot u = u$.

Primetimo da je skup X^* zatvoren za operaciju konkatenacije, odnosno $u \cdot v \in X^*$ za sve $u, v \in X^*$. Kako je operacija konkatenacije asocijativna, a prazna reč e predstavlja neutralni element za ovu operaciju, struktura $(X^*, \cdot, e)$ predstavlja ***slobodni monoid*** ili ***monoid reči*** nad alfabetom X.

Operacija konkatenacije u opštem slučaju nije komutativna, odnosno za različite reči $u, v \in X^*$ važi $u \cdot v \neq v \cdot u$, te slobodan monoid nije komutativan.

Struktura $(X^+, \cdot)$ naziva se ***slobodna polugrupa*** ili ***polugrupa reči***.

Preslikavanje $f : X^* \to S$, gde je S proizvoljan neprazan skup, naziva se ***funkcija reči***. Ona svakoj reči nad alfabetom X pridružuje tačno jedan element skupa S. U teoriji težinskih automata, skup S je najčešće nosilac strukture poluprstena. Tada vrednost $f(u)$ određuje težinu pridruženu reči $u \in X^*$, pa se funkcija reči može posmatrati kao formalni opis ponašanja težinskog automata.

Ukoliko je skup S uređen $(S, \leqslant)$, tada se poredak može prirodno preneti i na funkcije reči. Naime, za dve funkcije reči $f, g \colon X^* \to S$ kažemo da je $f \leqslant g$ ako važi pokoordinatni poredak $f(u) \leqslant g(u)$, za svaku reč $u \in X^*$.

Jezik nad alfabetom X je proizvoljan podskup skupa svih reči, odnosno $\mathcal{L} \subseteq X^*$.

Za jezike $\mathcal{L}_1, \mathcal{L}_2 \subseteq X^*$ definišu se sledeće operacije:

Konkatenacija jezika	$\mathcal{L}_1\mathcal{L}_2 = \{uv \mid u \in \mathcal{L}_1,\ v \in \mathcal{L}_2\}$;
Unija jezika	$\mathcal{L}_1 \cup \mathcal{L}_2 = \{w \in X^* \mid w \in \mathcal{L}_1 \text{ ili } w \in \mathcal{L}_2\}$;
Presek jezika	$\mathcal{L}_1 \cap \mathcal{L}_2 = \{w \in X^* \mid w \in \mathcal{L}_1 \text{ i } w \in \mathcal{L}_2\}$;
Komplement jezika	$\overline{\mathcal{L}} = X^* \setminus \mathcal{L}$.

Stepeni jezika definišu se sa

$$\mathcal{L}^0 = \{e\}, \qquad \mathcal{L}^n = \mathcal{L} \cdot \mathcal{L}^{n-1}, \quad n \geq 1.$$

Na osnovu stepena jezika definišu se Kleeneovi operatori zatvorenja. Operator $\mathcal{L}^*$

naziva se *Kleeneova zvezda*, dok se operator $\mathcal{L}^+$ naziva *pozitivno zatvorenje* jezika $\mathcal{L}$:

$$\mathcal{L}^* = \bigcup_{n\geq 0} \mathcal{L}^n, \qquad \mathcal{L}^+ = \bigcup_{n\geq 1} \mathcal{L}^n.$$

Neka su u i v reči nad alfabetom X.

Reč u je ***prefiks*** reči v ako postoji reč $w \in X^*$ takva da važi $v = u \cdot w$. Ako je pritom $w \in X^+$, tada se u naziva ***pravim prefiksom*** reči v.

Analogno, reč u je ***sufiks*** reči v ako postoji reč $w \in X^*$ takva da je $v = w \cdot u$. Ako je $w \in X^+$, tada se u naziva ***pravim sufiksom*** reči v.

Dalje, reč u je ***infiks*** reči v ako postoje reči $w_1, w_2 \in X^*$ takve da važi $v = w_1 \cdot u \cdot w_2$. Ukoliko su $w_1, w_2 \in X^+$, tada se u naziva ***pravim infiksom*** reči v.

2.2 Težinski automati

Težinski automati predstavljaju prirodno proširenje klasičnih konačnih automata u kojem se prelazima između stanja, kao i inicijalnim i završnim stanjima, pridružuju težine iz odgovarajuće algebarske strukture. Za razliku od klasičnih automata, kod kojih je informacija o prelazu binarne prirode, težinski automati omogućavaju modelovanje različitih kvantitativnih karakteristika sistema, kao što su verovatnoće prelaza, troškovi, trajanja događaja ili stepeni pripadnosti u fazi modelima.

Težinski automati su prvi put sistematski proučavani u radovima Schützenbergera [1], a njihova opšta algebarska teorija razvijena je u okviru teorije automata nad poluprstenima [3, 55].

Neka je X konačan alfabet i neka je dat poluprsten $\mathbb{S} = (S, \oplus, \otimes, 0, 1)$. U kontekstu težinskih automata, operacija $\oplus$ koristi se za agregaciju težina različitih putanja, dok operacija $\otimes$ opisuje kombinovanje težina duž jedne putanje automata.

Težinski automat (engl. *Weighted Finite Automaton*, skraćeno WFA) nad poluprstenom $\mathbb{S}$ i alfabetom X definiše se kao četvorka $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$, gde je:

$n \in \mathbb{N}$	***broj stanja (dimenzija) automata***;
$\sigma^A \in S^{1\times n}$	***inicijalni težinski vektor***;
$\tau^A \in S^{n\times 1}$	***završni težinski vektor***;
$M_x^A \in S^{n\times n}$	***matrica prelaza*** *za svako slovo ulaznog alfabeta* $x \in X$.

Vektor σ^A posmatra se kao vektor reda (matrica dimenzija $1 \times n$), dok se τ^A posmatra kao vektor kolone (matrica dimenzija $n \times 1$). Matrica M_x^A opisuje prelaze automata koji se realizuju pri čitanju simbola x, za svako $x \in X$. Element $M_x^A(i, j)$ predstavlja težinu prelaza iz stanja i u stanje j, prilikom čitanja simbola x.

Na osnovu matrica prelaza definisanih za pojedinačne simbole mogu se definisati matrice prelaza za proizvoljne reči iz X^*.

Neka je $u = x_1 x_2 \dots x_k \in X^+$ neprazna reč nad alfabetom X. Matrica prelaza pridružena reči u definiše se sa

$$M_u^A = M_{x_1}^A \otimes M_{x_2}^A \otimes \cdots \otimes M_{x_k}^A. \tag{31}$$

Pri tome se operacija $\otimes$ nad matricama izvodi kao matrično množenje indukovano operacijama $\oplus$ i $\otimes$ poluprstena $\mathbb{S}$.

Za praznu reč definišemo

$$M_e^A = I_n, \tag{32}$$

gde I_n označava jediničnu matricu dimenzije $n \times n$. Ovakva definicija odražava činjenicu da čitanje prazne reči ne menja stanje automata.

Na ovaj način matrica M_u^A objedinjuje informacije o svim putanjama automata koje se realizuju pri čitanju reči u. Zbog asocijativnosti operacije $\otimes$ definicija matrice M_u^A je dobro određena.

Za proizvoljnu reč $u \in X^*$ definišemo i vektore

$$\sigma_u^A = \sigma^A \otimes M_u^A \qquad \text{i} \qquad \tau_u^A = M_u^A \otimes \tau^A. \tag{33}$$

Posebno, za praznu reč e važi

$$\sigma_e^A = \sigma^A \qquad \text{i} \qquad \tau_e^A = \tau^A. \tag{34}$$

Takođe, za svaku reč $u \in X^*$ i svako slovo $x \in X$ važi

$$\sigma_{ux}^A = \sigma_u^A \otimes M_x^A \qquad \text{i} \qquad \tau_{xu}^A = M_x^A \otimes \tau_u^A. \tag{35}$$

Vektor σ_u^A predstavlja rezultat primene prelaza određenih rečju u na inicijalni vektor, dok vektor τ_u^A predstavlja rezultat primene istih prelaza na završni vektor.

U slučajevima kada nema opasnosti od zabune, u oznakama za inicijalni i završni vektor, kao i za matrice prelaza, izostavljaćemo A u zapisu (eksponentu).

Matrice prelaza, zajedno sa vektorima σ_u^A i τ_u^A, omogućavaju kompaktan zapis ponašanja automata na proizvoljnim rečima, što će biti iskorišćeno u narednoj definiciji.

Ponašanje (engl. *behavior*) težinskog automata $\mathcal{A}$ definiše se kao preslikavanje

$$[\![\mathcal{A}]\!] : X^* \to S.$$

Preslikavanje $[\![\mathcal{A}]\!]$ naziva se i ***funkcija reči*** automata $\mathcal{A}$ i određuje ***težinski jezik*** (kraće *jezik*) koji automat $\mathcal{A}$ raspoznaje.

Za razliku od klasičnih automata, kod kojih jezik predstavlja podskup slobodnog monoida X^*, kod težinskih automata svakoj reči pridružuje se vrednost iz poluprstena S:

$$[\![\mathcal{A}]\!](u) \;=\; \sigma^A \otimes M_u^A \otimes \tau^A \;=\; \sigma^A \otimes M_{x_1}^A \otimes M_{x_2}^A \otimes \cdots \otimes M_{x_k}^A \otimes \tau^A, \tag{36}$$

za svako $u = x_1 x_2 \ldots x_k \in X^+$, gde su $x_1, x_2, \ldots, x_k \in X$.

Na osnovu definicije (33), za svaku reč $u \in X^*$ važi

$$[\![\mathcal{A}]\!](u) = \sigma^A \otimes M_u^A \otimes \tau^A = \sigma_u^A \otimes \tau^A = \sigma^A \otimes \tau_u^A. \tag{37}$$

Vrednost $[\![\mathcal{A}]\!](u)$ dobija se kombinovanjem inicijalnog vektora, matrice prelaza pridružene reči u i završnog vektora pomoću operacija poluprstena S, te predstavlja težinu koju automat $\mathcal{A}$ pridružuje reči u.

Za praznu reč $e \in X^*$ važi

$$[\![\mathcal{A}]\!](e) = \sigma^A \otimes M_e^A \otimes \tau^A = \sigma^A \otimes \tau^A. \tag{38}$$

Ako za svaku reč $u \in X^*$ važi $[\![\mathcal{A}]\!](u) = 0$, kažemo da automat $\mathcal{A}$ ima *nulto ponašanje* (engl. *zero behaviour*). Ovde 0 označava nulti element poluprstena $\mathbb{S}$.

Formula (36) može se interpretirati i preko svih putanja automata koje odgovaraju zadatoj ulaznoj reči. Za reč $u = x_1 x_2 \ldots x_k \in X^*$ važi:

$$[\![\mathcal{A}]\!](u) = \bigoplus_{i_1, i_2, \ldots, i_{k+1} \in \{1,\ldots,n\}} \sigma^A(i_1) \otimes M_{x_1}^A(i_1, i_2) \otimes \cdots \otimes M_{x_k}^A(i_k, i_{k+1}) \otimes \tau^A(i_{k+1}).$$

Drugim rečima, za svaku putanju označenu rečju $u = x_1 x_2 \ldots x_k$ vrednost

$$\sigma^A(i_1) \otimes M_{x_1}^A(i_1, i_2) \otimes \cdots \otimes M_{x_k}^A(i_k, i_{k+1}) \otimes \tau^A(i_{k+1})$$

predstavlja težinu te putanje.

Operacija $\otimes$ određuje način kombinovanja težina duž jedne putanje, dok operacija $\oplus$ agregira težine svih mogućih putanja označenih istom rečju. Na taj način različiti izbori poluprstena dovode do različitih interpretacija ponašanja automata.

Prema tome, ponašanje težinskog automata može se posmatrati kao težinski jezik koji automat raspoznaje, pri čemu vrednost $[\![\mathcal{A}]\!](u)$ predstavlja težinu pridruženu reči u.

Uvedeni pojmovi i oznake predstavljaju osnovu za formalno opisivanje i analizu težinskih automata. Iako se ovi automati mogu predstaviti na različite načine, u nastavku razmatramo njihove grafovske i matrične reprezentacije, koje omogućavaju pregledniji prikaz njihove strukture i efikasniji zapis ponašanja.

2.3 Reprezentacija težinskih automata

Težinski automati mogu se predstaviti na više međusobno ekvivalentnih načina. Najčešće korišćene reprezentacije su matrična i grafovska reprezentacija. Matrična reprezentacija omogućava kompaktan algebarski zapis automata i pogodna je za teorijsku analizu, dokazivanje osobina i razvoj algoritama. Sa druge strane, grafovska reprezentacija pruža intuitivnu vizuelnu interpretaciju stanja, prelaza, putanja i ciklusa u automatu.

Obe reprezentacije opisuju istu strukturu automata i nose istu informaciju, ali naglašavaju različite aspekte njegovog ponašanja. Matrični pristup posebno je pogodan za algebarsku analizu težinskih automata, dok grafovski pristup omogućava lakše razumevanje njihove strukture i interpretaciju izvršavanja automata na ulaznim rečima [2, 3, 42].

U nastavku će najpre biti predstavljena grafovska reprezentacija težinskih automata, a zatim i odgovarajuća matrična reprezentacija, uz objašnjenje veze između ova dva pristupa.

2.3.1 Grafovska reprezentacija težinskih automata

Težinski automati mogu se predstaviti grafički pomoću usmerenog težinskog grafa. U ovoj reprezentaciji čvorovi grafa odgovaraju stanjima automata, dok usmerene grane predstavljaju prelaze između stanja. Svaka grana označena je simbolom iz ulaznog alfabeta i odgovarajućom težinom prelaza.

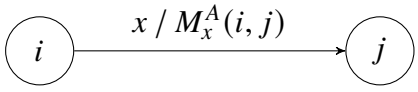
Posmatrajmo težinski automat $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x \in X}, \tau^A)$ definisan nad poluprstenom $\mathbb{S} = (S, \oplus, \otimes, 0, 1)$.

Graf automata može se posmatrati kao usmereni označeni težinski graf $G = (V, E)$, gde je $V = \{1, 2, \ldots, n\}$ skup čvorova, dok je E skup usmerenih grana definisan sa

$$E = \{(i, x, j) \in V \times X \times V \mid M_x^A(i, j) \neq 0\},$$

gde 0 označava nulu posmatranog poluprstena $\mathbb{S}$.

Svaka grana $(i, x, j) \in E$ nosi oznaku ulaznog simbola x i težinu $M_x^A(i, j)$. Grafički se takav prelaz predstavlja usmerenom granom od čvora i ka čvoru j označenom simbolom x i odgovarajućom težinom (videti Sliku 7).



Slika 7: Usmerena grana u težinskom automatu

U klasičnim krisp automatima težine se ne navode na granama, već grana jednostavno postoji ili ne postoji.

Stanje $i \in V$ je inicijalno ukoliko je $\sigma^A(i) \neq 0$. Grafički se označava strelicom koja ulazi u odgovarajući čvor, pri čemu se po potrebi navodi i inicijalna težina $\sigma^A(i)$.

Analogno, stanje j je finalno ukoliko je $\tau^A(j) \neq 0$. U grafičkoj reprezentaciji finalna stanja se najčešće označavaju dvostrukim krugom ili navođenjem odgovarajuće završne težine.

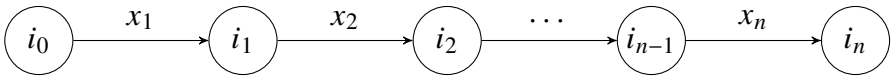
Putanja u grafu automata je niz čvorova $p = (i_1, i_2, \ldots, i_{k+1})$, takav da za svako $r \in \{1, \ldots, k\}$ postoji grana $(i_r, x_r, i_{r+1}) \in E$. U tom slučaju kažemo da je putanja p označena rečju $u = x_1 x_2 \ldots x_k \in X^+$.

Težina putanje p definiše se kao

$$w(p) = \sigma^A(i_1) \otimes M^A_{x_1}(i_1, i_2) \otimes M^A_{x_2}(i_2, i_3) \otimes \cdots \otimes M^A_{x_k}(i_k, i_{k+1}) \otimes \tau^A(i_{k+1}).$$

Drugim rečima, težina putanje dobija se kombinovanjem inicijalne težine početnog stanja, težina svih prelaza na putanji i završne težine finalnog stanja.

Dužina putanje odgovara dužini reči u (videti sliku 8).



Slika 8: Putanja u težinskom automatu

U grafu automata može postojati više različitih putanja koje realizuju istu reč u. Vrednost koju automat pridružuje reči u dobija se agregacijom težina svih takvih putanja, odnosno

$$\begin{aligned}[\![\mathcal{A}]\!](u) &= \bigoplus_{p \in P(u)} w(p) = \\ &= \bigoplus_{i_1, \ldots, i_{k+1} \in [1..n]} \sigma^A(i_1) \otimes M^A_{x_1}(i_1, i_2) \otimes \cdots \otimes M^A_{x_k}(i_k, i_{k+1}) \otimes \tau^A(i_{k+1}),\end{aligned}$$

pri čemu $P(u)$ označava skup svih putanja u grafu automata označenih rečju u.

Za praznu reč e posmatraju se putanje dužine nula, pa važi

$$[\![\mathcal{A}]\!](e) = \bigoplus_{i=1}^{n} \sigma^A(i) \otimes \tau^A(i).$$

Neka su $p_u = (i_1, i_2, \ldots, i_{k+1})$ i $p_v = (j_1, j_2, \ldots, j_{m+1})$ putanje označene rečima $u, v \in X^*$. Konkatenacija putanja p_u i p_v definisana je pod uslovom da je završno stanje prve putanje jednako početnom stanju druge, odnosno $i_{k+1} = j_1$. Tada je konkatenisana putanja data sa $p_{uv} = (i_1, i_2, \ldots, i_{k+1}, j_2, \ldots, j_{m+1})$, pri čemu njena težina zadovoljava

$$w(p_{uv}) = w(p_u) \otimes w(p_v).$$

Ciklus u grafu je putanja $p = (i_1, i_2, \ldots, i_{k+1})$ za koju važi $i_1 = i_{k+1}$. Dužina ciklusa definiše se kao broj prelaza u ciklusu.

Specijalan slučaj ciklusa predstavlja ***petlja***, koja nastaje kada grana počinje i završava se u istom čvoru (videti sliku 9).



Slika 9: Primer ciklusa (levo) i petlje (desno)

Grafovska reprezentacija pruža intuitivan uvid u strukturu automata, putanje i cikluse koji određuju njegovo ponašanje. Sa druge strane, matrična reprezentacija omogućava kompaktan algebarski zapis i pogodna je za formalnu analizu i algoritamsku obradu. U narednoj podsekciji biće predstavljena matrična reprezentacija težinskih automata.

2.3.2 Matrična reprezentacija težinskih automata

Matrična reprezentacija omogućava kompaktan algebarski zapis težinskih automata i predstavlja osnovu za primenu algebarskih metoda u njihovoj analizi. U ovom pristupu prelazi automata predstavljaju se matricama nad odgovarajućim poluprstenom, dok se inicijalna i završna stanja opisuju odgovarajućim vektorima. Ovakav zapis je posebno pogodan za formalno proučavanje ponašanja automata, kao i za definisanje različitih odnosa između automata, uključujući simulacije i bisimulacije koje predstavljaju jednu od centralnih tema ovog rada.

Neka je $\mathbb{S} = (S, \oplus, \otimes, 0, 1)$ poluprsten i neka su $m, n \in \mathbb{N}$.

Matrica dimenzije $m \times n$ nad poluprstenom $\mathbb{S}$ je preslikavanje $M : [1..m] \times [1..n] \to S$. Skup svih matrica dimenzije $m \times n$ nad poluprstenom $\mathbb{S}$ označava se sa $S^{m \times n}$. Vrednost $M(i, j)$ naziva se element matrice M na poziciji (i, j).

Matrica čiji su svi elementi jednaki nuli poluprstena naziva se *nula matrica* i označava se $O_{m \times n}$.

Slično, ***vektor*** dužine n nad poluprstenom $\mathbb{S}$ je preslikavanje $v : [1..n] \to S$. Vrednost $v(i)$ je element vektora v na poziciji i. Skup svih vektora dužine n označava se sa S^n.

Vektor vrste može se posmatrati kao matrica dimenzije $1 \times n$, dok se vektor kolone može posmatrati kao matrica dimenzije $n \times 1$.

Jedinična matrica reda n je matrica $I_n \in S^{n \times n}$ definisana sa

$$I_n(i, j) = \begin{cases} 1, & i = j, \\ 0, & i \neq j, \end{cases}$$

gde su 0 i 1 redom nula i jedinica posmatranog poluprstena $\mathbb{S}$.

Transponovana matrica matrice $M \in S^{m \times n}$ je matrica $M^T \in S^{n \times m}$ definisana sa $M^T(j, i) = M(i, j)$.

Za matrice $M, N \in S^{m\times n}$ sabiranje matrica definiše se koordinatno:

$$(M \oplus N)(i,j) = M(i,j) \oplus N(i,j).$$

Za matrice $M \in S^{m\times n}$ i $N \in S^{n\times p}$ proizvod matrica definiše se sa

$$(M \otimes N)(i,k) = \bigoplus_{j=1}^{n} \big(M(i,j) \otimes N(j,k)\big), \quad \text{za sve } i \in [1..m] \text{ i } k \in [1..p].$$

Proizvod matrica je asocijativan, odnosno

$$(M \otimes N) \otimes P = M \otimes (N \otimes P), \quad \text{za sve } M \in S^{m\times n},\ N \in S^{n\times p} \text{ i } P \in S^{p\times q}.$$

Za razliku od sabiranja, proizvod matrica u opštem slučaju nije komutativan, što se direktno može zaključiti iz definicije sabiranja i proizvoda matrica.

Za vektor vrste $\alpha \in S^{1\times m}$ i matricu $M \in S^{m\times n}$ definiše se ***vektorsko-matrični proizvod*** $\alpha \otimes M \in S^{1\times n}$ sa

$$(\alpha \otimes M)(j) = \bigoplus_{k=1}^{m} \big(\alpha(k) \otimes M(k,j)\big), \quad \text{za sve } j \in [1..n].$$

Analogno, za matricu $M \in S^{m\times n}$ i vektor kolonu $\beta \in S^{n\times 1}$ definiše se ***matrično-vektorski proizvod*** $M \otimes \beta \in S^{m\times 1}$ sa

$$(M \otimes \beta)(i) = \bigoplus_{k=1}^{n} \big(M(i,k) \otimes \beta(k)\big), \quad \text{za sve } i \in [1..m].$$

Poseban slučaj matričnog proizvoda predstavlja ***skalarni proizvod***. To je proizvod vektora vrste i vektora kolone, $\alpha \in S^{1\times n}$ i $\beta \in S^{n\times 1}$, gde je njihov proizvod element skupa S i definiše se sa

$$\alpha \otimes \beta = \bigoplus_{i=1}^{n} \big(\alpha(i) \otimes \beta(i)\big).$$

Za skalar $s \in S$ i matricu $M \in S^{m\times n}$ definiše se ***skalarno množenje matrice*** skalarom s kao matrica $s \otimes M \in S^{m\times n}$ čiji su elementi dati sa

$$(s \otimes M)(i,j) = s \otimes M(i,j), \quad \text{za sve } i \in [1..m] \text{ i } j \in [1..n].$$

Za skalare $s, t \in S$ i matrice odgovarajućih dimenzija važi

$$s \otimes (t \otimes M) = (s \otimes t) \otimes M, \qquad s \otimes (M \otimes N) = (s \otimes M) \otimes N.$$

Ukoliko je množenje u poluprstenu $\mathbb{S}$ komutativno, tada za sve $s, t \in S$, $M \in S^{m\times n}$, $N \in S^{n\times p}$, $\alpha \in S^{1\times m}$ i $\beta \in S^{n\times 1}$ važe sledeće relacije:

$$(s \otimes M) \otimes (t \otimes N) = (s \otimes t) \otimes (M \otimes N), \quad (s \otimes \alpha) \otimes (t \otimes \beta) = (s \otimes t) \otimes (\alpha \otimes \beta).$$

U prethodnim definicijama isti simbol $\otimes$ koristi se za množenje u poluprstenu, skalarno množenje matrica, matrični proizvod i proizvode vektora i matrica. Konkretna interpretacija operacije određena je dimenzijama njenih činilaca. Vektorsko-matrični, matrično-vektorski i skalarni proizvod predstavljaju posebne slučajeve matričnog proizvoda, dok se skalarno množenje matrice definiše koordinatno pomoću množenja u poluprstenu.

Ukoliko je na poluprstenu $\mathbb{S}$ definisano uređenje $\leq$, tada se na skupu matrica $S^{m\times n}$ definiše *koordinatno uređenje*:

$$M \leq N \iff M(i,j) \leq N(i,j), \quad \text{za sve } i \in [1..m] \text{ i } j \in [1..n].$$

Analogno, za vektore iste dužine definiše se koordinatno uređenje:

$$\alpha \leq \beta \iff \alpha(i) \leq \beta(i), \quad \text{za sve } i \in [1..n].$$

Ako je $\mathbb{S}$ kompletan dioid, tada je i skup matrica $S^{m\times n}$ kompletna mreža u odnosu na koordinatno uređenje.

Za proizvoljnu familiju matrica $\{M_k\}_{k\in K} \subseteq S^{m\times n}$ važi

$$\left(\bigoplus_{k\in K} M_k\right)(i,j) = \bigoplus_{k\in K} M_k(i,j), \qquad \left(\bigwedge_{k\in K} M_k\right)(i,j) = \bigwedge_{k\in K} M_k(i,j),$$

odnosno supremum i infimum proizvoljne familije matrica određuju se koordinatno.

Matrične operacije uvedene iznad omogućavaju kompaktan algebarski zapis težinskih automata i njihovog ponašanja.

Posmatrajmo težinski automat $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ nad poluprstenom $\mathbb{S}$. Svakom ulaznom simbolu $x \in X$ pridružuje se matrica prelaza $M_x^A \in S^{n\times n}$, čiji element $M_x^A(i,j)$ predstavlja težinu prelaza iz stanja i u stanje j pri čitanju simbola x. Pored matrica prelaza, automat je određen i inicijalnim i završnim težinskim vektorom.

Vektori inicijalnih i završnih stanja definišu se redom

$$\sigma^A = (\sigma^A(1), \sigma^A(2), \ldots, \sigma^A(n)) \in S^{1\times n}, \quad \tau^A = (\tau^A(1), \tau^A(2), \ldots, \tau^A(n))^T \in S^{n\times 1}.$$

Za nepraznu reč $u = x_1x_2\ldots x_k \in X^+$ odgovarajuća matrica prelaza definiše se proizvodom

$$M_u^A = M_{x_1}^A \otimes M_{x_2}^A \otimes \cdots \otimes M_{x_k}^A, \tag{39}$$

Za praznu reč e važi

$$M_e^A = I_n.$$

Na osnovu prethodno uvedenih pojmova, ponašanje težinskog automata može se izraziti u matričnom obliku:

$$[\![\mathcal{A}]\!](u) = \sigma^A \otimes M_u^A \otimes \tau^A.$$

Ako je poluprsten $\mathbb{S}$ kompletan dioid, tada se na matricama nad $\mathbb{S}$ mogu definisati i operacije reziduacije.

Reziduali se prirodno javljaju kao najveća rešenja odgovarajućih matričnih nejednačina. Za matrice $M \in S^{m\times n}$ i $N \in S^{m\times p}$ posmatrajmo matričnu nejednačinu $M \otimes U \leq N$, gde je U nepoznata matrica iz $S^{n\times p}$. Najveće rešenje ove nejednačine naziva se ***levi rezidual*** matrice N po matrici M i označava se sa $M\backslash N$. Elementi matrice $M\backslash N$ određeni su sa:

$$(M\backslash N)(i,j) = \bigwedge_{k=1}^{m} M(k,i)\backslash N(k,j), \qquad \text{za sve } i \in [1..n] \text{ i } j \in [1..p].$$

Analogno, za matrice $M \in S^{n\times p}$ i $N \in S^{m\times p}$ posmatrajmo nejednačinu $V \otimes M \leq N$, gde je V nepoznata matrica iz $S^{m\times n}$. Najveće rešenje ove nejednačine naziva se ***desni rezidual*** matrice N po matrici M i označava se sa N/M. Elementi matrice N/M određeni su sa

$$(N/M)(i,j) = \bigwedge_{k=1}^{p} N(i,k)/M(j,k), \qquad \text{za sve } i \in [1..m] \text{ i } j \in [1..n].$$

Osnovnu veza između matričnog množenja i reziduacije daje zakonom adjunkcije:

$$M \otimes N \leq P \iff N \leq M\backslash P \iff M \leq P/N$$

za sve matrice $M \in S^{m\times n}$, $N \in S^{n\times p}$ i $P \in S^{m\times p}$.

Pošto se vektori mogu posmatrati kao specijalan slučaj matrica, operacije reziduacije definišu se analogno i za vektore nad kompletnim dioidom.

Za vektore vrste $\alpha \in S^{1\times m}$ i $\beta \in S^{1\times n}$, ***levi rezidual*** $\alpha\backslash\beta$ predstavlja najveće rešenje nejednačine $\alpha \otimes U \leq \beta$, dok za vektore kolone $\alpha \in S^{m\times 1}$ i $\beta \in S^{n\times 1}$ ***desni rezidual*** β/α predstavlja najveće rešenje nejednačine $V \otimes \alpha \leq \beta$. Elementi ovih reziduala određeni su

$$(\alpha\backslash\beta)(i,j) = \alpha(i)\backslash\beta(j) \quad \text{i} \quad (\beta/\alpha)(j,i) = \beta(j)/\alpha(i), \qquad \text{za sve } i \in [1..m] \; i \; j \in [1..n].$$

Zbog komutativnosti operacije $\otimes$ važi

$$\beta/\alpha = (\alpha\backslash\beta)^T.$$

Takođe, važe osobine reziduacije:

$$\alpha \otimes U \leq \beta \iff U \leq \alpha\backslash\beta \quad \text{i} \quad V \otimes \alpha \leq \beta \iff V \leq \beta/\alpha. \tag{40}$$

Pored levog i desnog reziduala, često se koristi i ***birezidual***, koji omogućava simetrično poređenje vektora $\alpha \in S^{1\times m}$ i $\beta \in S^{1\times n}$, a označava sa $\alpha|\beta$ i predstavlja matricu iz $S^{m\times n}$ definisanu kao infimum levog i desnog reziduala:

$$\alpha|\beta = (\alpha\backslash\beta) \wedge (\beta/\alpha)^T.$$

Pošto se infimum matrica definiše koordinatno, za $i \in [1..m]$ i $j \in [1..n]$ važi

$$(\alpha|\beta)(i,j) = (\alpha\backslash\beta)(i,j) \wedge (\beta/\alpha)^T(i,j),$$

gde $\wedge$ na desnoj strani označava infimum u kompletnom dioidu $\mathbb{S}$.

Operacije nad matricama i vektorima, zajedno sa odgovarajućim rezidualima, predstavljaju važan alat za rešavanje nejednačina koje se pojavljuju pri definisanju i izračunavanju simulacija (inkluzije) i bisimulacija (ekvivalencije) težinskih automata [31, 2].

2.4 Specifične klase težinskih automata

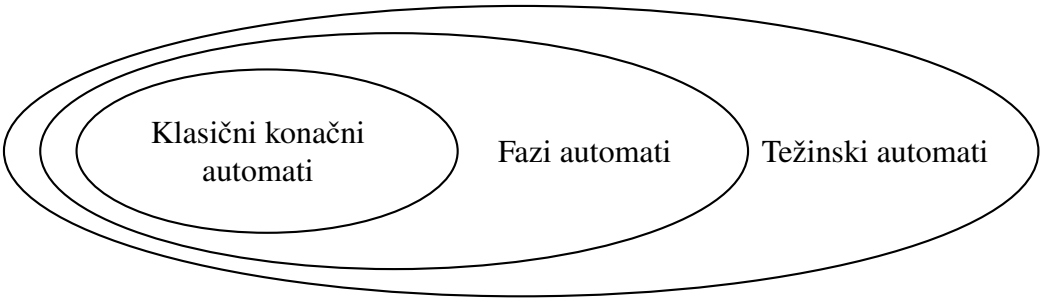
Kao što je prethodno prikazano, težinski automati predstavljaju opšti algebarski model u kome se težine prelaza, inicijalnih i završnih stanja uzimaju iz unapred zadatog poluprstena. Izbor algebarske strukture nad skupom težina određuje način kombinovanja prelaza, interpretaciju vrednosti koje automat pridružuje rečima, kao i oblast primene odgovarajućeg modela.

Klasični konačni automati, kod kojih prelazi imaju binarnu prirodu, mogu se posmatrati kao specijalan slučaj težinskih automata nad Bulovim poluprstenom. Njihova teorija predstavlja osnovu formalnih jezika i automata, razvijenu kroz radove čiji su autori Kleene, Rabin i Scott [49, 50].

Daljim uopštavanjem binarnih vrednosti prelaza razvijeni su modeli koji omogućavaju opisivanje sistema sa neodređenošću i kvantitativnim informacijama. Nakon uvođenja teorije fazi skupova od strane Zadeh-a [40], razvijeni su fazi automati, kod kojih vrednosti prelaza predstavljaju stepene pripadnosti ili stepene mogućnosti prelaza, a najčešće se posmatraju nad strukturama indukovanim kompletnim reziduiranim mrežama [56].

Sa druge strane, izborom drugih poluprstena dobijaju se različiti kvantitativni modeli automata. Za nas posebno značajnu klasu predstavljaju max-plus automati, odnosno težinski automati definisani nad max-plus poluprstenom. Oni se koriste za modelovanje vremenski diskretnih sistema događaja u kojima se javljaju sinhronizacija procesa, deljenje resursa i vremenska zavisnost aktivnosti. Max-plus automati imaju značajnu primenu u proizvodnim i transportnim sistemima, računarskim mrežama i drugim modelima kod kojih je potrebno analizirati vremenska ograničenja i optimizovati izvršavanje procesa. Njihov značaj proizilazi iz činjenice da se određeni nelinearni fenomeni mogu opisati linearnim modelima u okviru max-plus algebre [13, 15, 57, 58, 59].

U nastavku će biti predstavljene ove tri klase automata kao specijalni slučajevi težinskih automata. Posebna pažnja biće posvećena fazi automatima nad proizvod strukturom i max-plus automatima, budući da će oni predstavljati osnovne modele za proučavanje simulacija i bisimulacija u narednim glavama rada. Odnos između klasičnih konačnih, fazi i težinskih automata prikazan je na slici 10.



Slika 10: Hijerarhijski odnos klasičnih konačnih, fazi i težinskih automata

2.4.1 Klasični konačni automati

Klasični konačni automati predstavljaju osnovni model teorije automata kod kojih je ponašanje sistema opisano konačnim skupom stanja i prelazima između njih. Za razliku od opštih težinskih automata, kod kojih prelazi mogu imati proizvoljne vrednosti iz odgovarajućeg poluprstena, kod klasičnih automata prelazi imaju binarnu prirodu: prelaz između dva stanja ili postoji ili ne postoji.

Klasični konačni automati mogu se posmatrati kao specijalan slučaj težinskih automata nad Bulovim poluprstenom $\mathbb{B} = (\{0, 1\}, \vee, \wedge, 0, 1)$. U ovoj strukturi element 1 označava postojanje prelaza, dok element 0 označava njegovo odsustvo. Pri prirodnom uređenju skupa $\{0, 1\}$ element 0 predstavlja najmanji, a element 1 najveći element. Operacija $\vee$ odgovara logičkoj disjunkciji, dok operacija $\wedge$ odgovara logičkoj konjunkciji. Bulov poluprsten zasniva se na Bulovoj mreži, koja je razmatrana u Primeru 1.5.

Neka je dat automat $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x \in X}, \tau^A)$ nad Bulovim poluprstenom:

$$\sigma^A \in \{0, 1\}^{1 \times n}, \qquad \tau^A \in \{0, 1\}^{n \times 1}, \qquad M_x^A \in \{0, 1\}^{n \times n}.$$

Element matrice prelaza $M_x^A(i, j)$ jednak je 1 ako postoji prelaz iz stanja i u stanje j označen simbolom x, dok vrednost 0 označava da takav prelaz ne postoji.

Operacije Bulovog poluprstena određuju način računanja jezika raspoznatog automatom. Operacija $\wedge$ koristi se za proveru postojanja kompletne putanje označene određenom rečju, dok operacija $\vee$ agregira sve moguće putanje koje odgovaraju istoj reči.

Na taj način ponašanje automata $[\![\mathcal{A}]\!] : X^* \to \{0, 1\}$ za svaku reč $u \in X^*$ određuje da li je ona prihvaćena ili nije. Ako je $[\![\mathcal{A}]\!](u) = 1$, tada automat prihvata reč u, dok vrednost $[\![\mathcal{A}]\!](u) = 0$ znači da reč nije prihvaćena.

Skup svih reči koje automat $\mathcal{A}$ prihvata predstavlja jezik automata i definisan je sa

$$[\![\mathcal{A}]\!] = L(\mathcal{A}) = \{u \in X^* \mid [\![\mathcal{A}]\!](u) = 1\}.$$

Na ovaj način klasični konačni automati predstavljaju osnovni binarni slučaj težinskih automata. U nastavku će biti razmotrene i druge značajne klase težinskih automata, među kojima posebno mesto zauzimaju fazi automati i max-plus automati.

2.4.2 Fazi automati

Fazi automati predstavljaju posebnu klasu težinskih automata kod kojih težine prelaza, kao i težine inicijalnih i završnih stanja, uzimaju vrednosti iz skupa L kompletne reziduirane mreže.

Neka je data kompletna reziduirana mreža $(L, \vee, \wedge, \otimes, \rightarrow, 0, 1)$. Kako $(L, \vee, \otimes, 0, 1)$ predstavlja komutativan poluprsten, fazi automat se može posmatrati kao težinski automat $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ na poluprstenu L. Elementi početnog i završnog vektora predstavljaju stepene pripadnosti odgovarajućim fazi skupovima inicijalnih i završnih stanja, dok elementi matrica prelaza određuju stepene mogućnosti prelaza između stanja automata.

Vektor σ^A naziva se ***fazi skup inicijalnih stanja***, dok se vektor τ^A naziva ***fazi skup završnih stanja*** automata. Familija matrica prelaza $\{M_x^A\}_{x\in X}$ određuje ***fazi tranzicionu funkciju*** definisanu sa:

$$M^A : A \times X \times A \rightarrow L, \qquad M^A(a, x, b) = M_x^A(a, b), \text{ za sve } a, b \in A \text{ i } x \in X.$$

Drugim rečima, vrednost $M_x^A(i, j)$, odnosno $M^A(i, x, j)$, određuje stepen mogućnosti prelaza iz stanja i u stanje j prilikom čitanja simbola x.

Na taj način matrice prelaza mogu se posmatrati kao fazi relacije na skupu stanja automata. Zbog toga se operacije nad fazi relacijama uvedene u prethodnoj glavi prirodno primenjuju pri računanju jezika raspoznatog fazi automatom.

U standardnoj terminologiji fazi automata uobičajeno je da se automat zapisuje u obliku $\mathcal{A} = (A, X, \sigma^A, \delta^A, \tau^A)$, gde su σ^A i τ^A fazi skupovi inicijalnih i završnih stanja, δ^A fazi tranziciona funkcija, a X ulazni alfabet. Međutim, radi usklađenosti sa terminologijom koja je razvijena u radovima koji ulaze u sastav ove doktorske disertacije, koristićemo zapis $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$.

Za svaku reč $u \in X^*$ matrica M_u^A predstavlja fazi relaciju na skupu stanja automata. Pri tome za praznu reč važi $M_e^A = I_n$, dok se za reč $u = x_1x_2 \cdots x_k$ relacija M_u^A dobija kompozicijom fazi relacija

$$M_u^A = M_{x_1}^A \circ M_{x_2}^A \circ \cdots \circ M_{x_k}^A,$$

gde je kompozicija definisana operacijama kompletne reziduirane mreže.

Ponašanje fazi automata, odnosno ***fazi jezik*** koji automat raspoznaje, predstavlja specijalizaciju ponašanja težinskog automata nad poluprstenom $(L, \vee, \otimes, 0, 1)$ i definisano je preslikavanjem

$$[\![\mathcal{A}]\!] : X^* \rightarrow L.$$

Za svaku reč $u \in X^*$ vrednost $[\![\mathcal{A}]\!](u)$ daje stepen u kome automat $\mathcal{A}$ raspoznaje reč u.

Posebno značajnu klasu fazi automata čine automati definisani nad Goguenovom, odnosno proizvodnom strukturom. Ova klasa fazi automata biće korišćena u poslednjoj glavi disertacije, prilikom proučavanja aproksimativnih simulacija i bisimulacija između fazi automata.

Proizvod struktura definisana je na intervalu $[0, 1]$, gde su operacije supremuma i infimuma date sa

$$x \vee y = \max\{x, y\}, \qquad x \wedge y = \min\{x, y\}. \tag{41}$$

Operacija množenja definisana je standardnim proizvodom, dok je njoj odgovarajuća operacija reziduacije definisana na sledeći način:

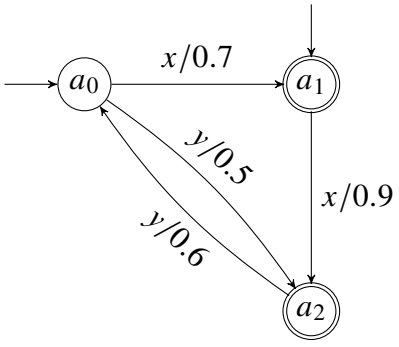
$$x \otimes y = x \cdot y, \qquad x \to y = \begin{cases} 1, & x \leq y, \\ \dfrac{y}{x}, & x > y. \end{cases} \tag{42}$$

Na ovaj način dobija se kompletna reziduirana mreža $([0, 1], \vee, \wedge, \otimes, \to, 0, 1)$. Fazi automati definisani nad ovom strukturom predstavljaju težinske automate nad poluprstenom $([0, 1], \vee, \otimes, 0, 1)$, pri čemu se kombinovanje težina vrši pomoću standardnog proizvoda, a agregacija pomoću maksimuma.

U narednom primeru prikazan je način izračunavanja stepena raspoznavanja reči nad Goguenovom (proizvod) strukturom.

Primer 2.1 *Neka je dat fazi automat $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x \in X}, \tau^A)$ nad Goguenovom (proizvod) strukturom. Neka je ulazni alfabet $X = \{x, y\}$ i neka automat ima $n = 3$ stanja. Inicijalni vektor, završni vektor i matrice prelaza dati su sa:*

$$\sigma^A = \begin{bmatrix} 1 & 0.6 & 0 \end{bmatrix}, \quad \tau^A = \begin{bmatrix} 0 \\ 0.8 \\ 0.5 \end{bmatrix}, \quad M_x^A = \begin{bmatrix} 0 & 0.7 & 0 \\ 0 & 0 & 0.9 \\ 0 & 0 & 0 \end{bmatrix}, \quad M_y^A = \begin{bmatrix} 0 & 0 & 0.5 \\ 0 & 0 & 0 \\ 0.6 & 0 & 0 \end{bmatrix}.$$



Slika 11: Grafovska reprezentacija automata iz primera 2.1

Stepen raspoznavanja prazne reči e računa se korišćenjem formule 38, gde se matrično množenje izvodi pomoću operacija $\vee$ i $\otimes$ iz Goguenove proizvod strukture:

$$[\![\mathcal{A}]\!](e) = \sigma^A \circ \tau^A = \begin{bmatrix} 1 & 0.6 & 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 0.8 \\ 0.5 \end{bmatrix} = \max\{1 \cdot 0,\ 0.6 \cdot 0.8,\ 0 \cdot 0.5\} = 0.48$$

Stepen raspoznavanja svake neprezne reči u automatom $\mathcal{A}$, računaćemo korišćenjem 36:

$$[\![\mathcal{A}]\!](x) = \sigma^A \otimes M_x^A \otimes \tau^A = \begin{bmatrix} 1 & 0.6 & 0 \end{bmatrix} \otimes \begin{bmatrix} 0 & 0.7 & 0 \\ 0 & 0 & 0.9 \\ 0 & 0 & 0 \end{bmatrix} \otimes \begin{bmatrix} 0 \\ 0.8 \\ 0.5 \end{bmatrix} = 0.56$$

Po istom principu mogu se izračunati stepeni raspoznavanja proizvoljnih reči iz skupa X^*, pri čemu se kompozicija prelaza određuje operacijama Goguenove proizvod strukture.

2.4.3 Max-plus automati

Max-plus automati predstavljaju posebnu klasu težinskih automata kod kojih su težine prelaza, inicijalnih i završnih stanja elementi max-plus poluprstena. Ovi automati imaju značajnu primenu u modelovanju diskretnih sistema događaja sa vremenom, optimizacionih problema i drugih sistema kod kojih težine predstavljaju trajanje, cenu ili količinu resursa potrebnih za izvršavanje određenih događaja.

Pre definisanja max-plus automata uvodimo osnovne max-plus strukture koje će se koristiti za dodeljivanje težina stanjima i prelazima automata.

U nastavku, $\mathbb{R}$ označava skup realnih brojeva, dok se prošireni skupovi realnih brojeva definišu sa

$$\mathbb{R}_{\max} = \mathbb{R} \cup \{-\infty\}, \qquad \mathbb{R}_{\infty} = \mathbb{R} \cup \{-\infty, +\infty\},$$

gde su $-\infty$ i $+\infty$ simboli koji ne pripadaju skupu $\mathbb{R}$ i predstavljaju negativnu, odnosno pozitivnu beskonačnost. Takođe, koristićemo skupove

$$\mathbb{R}^- = \{a \in \mathbb{R} \mid a < 0\}, \qquad \mathbb{R}^+ = \mathbb{P} = \{a \in \mathbb{R} \mid 0 \leq a\},$$

pri čemu $\leq$ označava uobičajeno linearno uređenje realnih brojeva, a $<$ njegovo strogo uređenje. Istim simbolom $\leq$ označavaćemo i proširenje ovog uređenja na skup $\mathbb{R}_{\infty}$, pri čemu je $-\infty$ najmanji, a $+\infty$ najveći element.

Max-plus poluprsten označava se sa $\mathbb{R}_{\max} = (\mathbb{R}_{\max}, \oplus, \otimes, -\infty, 0)$, pri čemu su za proizvoljne $a, b \in \mathbb{R}_{\max}$ operacije definisane sa

$$a \oplus b = \max\{a, b\} \quad \text{i} \quad a \otimes b = \begin{cases} a + b, & a, b \in \mathbb{R}, \\ -\infty, & a = -\infty \text{ ili } b = -\infty. \end{cases}$$

Element $-\infty$ predstavlja nulu poluprstena, dok je realni broj 0 jedinični element za operaciju $\otimes$. Operacija $\oplus$ odgovara supremumu u odnosu na uobičajeno uređenje realnih brojeva prošireno tako da je $-\infty$ najmanji element na skupu $\mathbb{R}_{\max}$.

Operacija $\otimes$ se može predstaviti i sledećom Caylel-evom tablicom:

$\otimes$	$b \in \mathbb{R}$	$-\infty$
$a \in \mathbb{R}$	$a + b$	$-\infty$
$-\infty$	$-\infty$	$-\infty$

Max-plus poluprsten $\mathbb{R}_{\max}$ je komutativan aditivno-idempotentan poluprsten (komutativan dioid). Takođe, on je pozitivan poluprsten, jer je $-\infty \leqslant a$ za svako $a \in \mathbb{R}_{\max}$. Dodatno, svaki neprazan podskup skupa $\mathbb{R}_{\max}$ ima infimum, a svaki konačan podskup ima supremum. Međutim, ova struktura nema najveći element i nije kompletna u smislu postojanja proizvoljnih supremuma, pa nije kompletna mreža. Zbog toga se uvodi proširena struktura dodavanjem elementa $+\infty$.

Kompletan max-plus poluprsten označava se sa $\mathbb{R}_\infty = (\mathbb{R}_\infty, \oplus, \otimes, -\infty, 0)$, gde je $\mathbb{R}_\infty = \mathbb{R} \cup \{-\infty, +\infty\}$. Pri tome je max-plus poluprsten $\mathbb{R}_{\max}$ podpoluprsten kompletnog max-plus poluprstena $\mathbb{R}_\infty$. Uobičajeno uređenje realnih brojeva proširuje se na skup $\mathbb{R}_\infty$ tako da važi $-\infty \leq a \leq +\infty$, za svako $a \in \mathbb{R}_\infty$.

Operacija sabiranja $\oplus$ i množenja $\otimes$ na skupu $\mathbb{R}_\infty$ definisane su na sledeći način:

$$a \oplus b = \max\{a, b\} \quad \text{i} \quad a \otimes b = \begin{cases} a + b, & a, b \in \mathbb{R}, \\ -\infty, & a = -\infty \text{ ili } b = -\infty, \\ +\infty, & \text{inače.} \end{cases}$$

Element $-\infty$ predstavlja nulu poluprstena, dok je realan broj 0 jedinični element za operaciju $\otimes$.

Kako svaki podskup skupa $\mathbb{R}_\infty$ ima supremum, u odnosu na uvedeno uređenje, struktura $\mathbb{R}_\infty = (\mathbb{R}_\infty, \oplus, \otimes, -\infty, 0)$ predstavlja kompletan komutativan dioid (kompletan aditivno-idempotentan poluprsten *engl. unital quantales*). Idempotentnost operacije sabiranja $\oplus$ neposredno sledi iz jednakosti $a \oplus a = \max\{a, a\} = a$, za svako $a \in \mathbb{R}_\infty$.

Kako je $\mathbb{R}_\infty$ kompletan dioid, na njemu je definisana operacija reziduacije. Zbog komutativnosti operacije $\otimes$, levi i desni rezidual se poklapaju, pa je dovoljno posmatrati jednu rezidualnu operaciju. U skladu sa opštom definicijom reziduacije u kompletnim dioidima, reziduum $a \backslash b$ predstavlja najveće rešenje nejednačine $a \otimes x \leq b$.

U kompletnoj max-plus strukturi važi:

$$a \backslash b = \begin{cases} b - a, & a, b \in \mathbb{R}, \\ +\infty, & a = -\infty \text{ ili } b = +\infty, \\ -\infty, & \text{inače.} \end{cases}$$

Operacije množenja $\otimes$ i reziduacije $\backslash$, na $\mathbb{R}_\infty$ poluprstenu, mogu se prikazati i predstojećim Cayley-evim tablicama:

$\otimes$	$b \in \mathbb{R}$	$-\infty$	$+\infty$
$a \in \mathbb{R}$	$a + b$	$-\infty$	$+\infty$
$-\infty$	$-\infty$	$-\infty$	$-\infty$
$+\infty$	$+\infty$	$-\infty$	$+\infty$

$\backslash$	$b \in \mathbb{R}$	$-\infty$	$+\infty$
$a \in \mathbb{R}$	$b - a$	$-\infty$	$+\infty$
$-\infty$	$+\infty$	$+\infty$	$+\infty$
$+\infty$	$-\infty$	$-\infty$	$+\infty$

Operacija reziduacije povezana je sa operacijom množenja zakonom adjunkcije:

$$a \otimes b \leq c \iff b \leq a \backslash c, \quad \text{za sve } a, b, c \in \mathbb{R}_\infty.$$

Na taj način kompletna max-plus struktura predstavlja reziduiranu strukturu, što omogućava rešavanje odgovarajućih nejednačina i definisanje rezidualnih relacija. Ove osobine imaju ključnu ulogu u narednim poglavljima prilikom proučavanja simulacija i bisimulacija max-plus automata.

Pored max-plus poluprstena i kompletnog max-plus poluprstena, u nastavku rada koristiće se i odgovarajući dioidi. Zbog toga je u Tabeli 3 dat objedinjeni pregled osnovnih max-plus struktura koje će se pojavljivati u ovoj disertaciji.

Primetimo da su skupovi $\mathbb{P}_{\max}$ i $\mathbb{P}_\infty$ podskupovi skupa $\mathbb{R}_\infty$, odnosno da je kompletan max-plus poluprsten $\mathbb{R}_\infty$ najopštija struktura koja obuhvata i max-plus dioid i kompletan max-plus dioid. Kod svih struktura iz Tabele 3 operacija sabiranja definisana je na isti način, $a \oplus b = \max(a, b)$, dok se operacije množenja i reziduacije razlikuju, odnosno zavise od posmatrane strukture. Operacija reziduacije definisana je samo na kompletnom max-plus poluprstenu i kompletnom max-plus dioidu.

Skupovi $\mathbb{Z}_\infty = \mathbb{Z} \cup \{-\infty, +\infty\}$ i $\mathbb{Z}_{\max} = \mathbb{Z} \cup \{-\infty\}$ predstavljaju podskupove skupa $\mathbb{R}_\infty$ koji su zatvoreni u odnosu na operacije $\oplus$ i $\otimes$, pri čemu je $\mathbb{Z}_\infty$ zatvoren i u odnosu na beskonačne sume i operaciju reziduacije.

Sa operacijama nasleđenim iz strukture $\mathbb{R}_\infty$, koje se označavaju istim simbolima kao odgovarajuće operacije na $\mathbb{R}_\infty$, struktura $\mathbb{Z}_\infty = (\mathbb{Z}_\infty, \oplus, \otimes, -\infty, 0)$ predstavlja *komutativan kompletan aditivno-idempotentan poluprsten*, koji se naziva ***kompletan max-plus poluprsten celih brojeva***. Sa druge strane, struktura $\mathbb{Z}_{\max} = (\mathbb{Z}_{\max}, \oplus, \otimes, -\infty, 0)$ predstavlja *aditivno-idempotentan poluprsten*, koji se naziva ***max-plus poluprsten celih brojeva***.

Tabela 3: Pregled osnovnih max-plus struktura i operacija

Max-plus struktura	$\mathbf{a} \otimes \mathbf{b}$	$\mathbf{a} \backslash \mathbf{b}$
Poluprsten $\mathbb{R}_{\max} = \mathbb{R} \cup \{-\infty\}$	$\begin{cases} a+b, & a, b \in \mathbb{R}, \\ -\infty, & a = -\infty \text{ ili } b = -\infty \end{cases}$	nije definisana
Dioid $\mathbb{P}_{\max} = \mathbb{R}^+ \cup \{-\infty\}$	$\begin{cases} a+b, & a, b \in \mathbb{R}^+, \\ -\infty, & a = -\infty \text{ ili } b = -\infty \end{cases}$	nije definisana
Kompletan poluprsten $\mathbb{R}_\infty = \mathbb{R} \cup \{-\infty, +\infty\}$	$\begin{cases} a+b, & a, b \in \mathbb{R}, \\ -\infty, & a = -\infty \text{ ili } b = -\infty, \\ +\infty, & \text{inače} \end{cases}$	$\begin{cases} b-a, & a, b \in \mathbb{R}, \\ +\infty, & a = -\infty \text{ ili } b = +\infty, \\ -\infty, & \text{inače} \end{cases}$
Kompletan dioid $\mathbb{P}_\infty = \mathbb{R}^+ \cup \{-\infty, +\infty\}$	$\begin{cases} a+b, & a, b \in \mathbb{R}^+, \\ -\infty, & a = -\infty \text{ ili } b = -\infty, \\ +\infty, & \text{inače} \end{cases}$	$\begin{cases} b-a, & a, b \in \mathbb{R}^+,\ b \geq a, \\ +\infty, & a = -\infty \text{ ili } b = +\infty, \\ -\infty, & \text{inače} \end{cases}$

Neka je dat težinski automat $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x \in X}, \tau^A)$ nad kompletnim max-plus poluprstenom. Tada je n broj stanja automata i važi

$$\sigma^A \in \mathbb{R}_\infty^{1 \times n}, \quad \tau^A \in \mathbb{R}_\infty^{n \times 1}, \quad M_x^A \in \mathbb{R}_\infty^{n \times n}.$$

Vrednost $M_x^A(i, j)$ predstavlja težinu prelaza iz stanja i u stanje j pri čitanju slova x, dok $\sigma^A(i)$ i $\tau^A(i)$ predstavljaju inicijalnu, odnosno završnu težinu stanja i.

U literaturi koja se bavi max-plus automatima i njihovom primenom u modelovanju vremenskih diskretnih sistema događaja, težine se često interpretiraju kao vremenske vrednosti. U tom slučaju inicijalni vektor σ^A naziva se ***vektor inicijalnih kašnjenja*** (engl. *initial delay vector*), završni vektor τ^A naziva se ***vektor završnih kašnjenja*** (engl. *final delay vector*), dok se matrice prelaza M_x^A, $x \in X$, nazivaju ***matrice vremena prelaza*** (engl. *transition time matrices*). Elementi alfabeta X tada se interpretiraju kao ***događaji*** (engl. *events*).

Ponašanje max-plus automata definisano je kao preslikavanje $[\![\mathcal{A}]\!] : X^* \to \mathbb{R}_\infty$, pri čemu za svaku reč $u \in X^*$ važi $[\![\mathcal{A}]\!](u) = \sigma^A \otimes M_u^A \otimes \tau^A$ (videti 36 i 38).

Drugim rečima, ponašanje max-plus automata predstavlja funkciju reči definisanu na slobodnom monoidu X^* sa vrednostima u skupu $\mathbb{R}_\infty$. U teoriji max-plus automata ovakve funkcije reči sa vrednostima u $\mathbb{R}_{\max}$ poznate su i pod engleskim nazivom ***daters***, budući da njihove vrednosti predstavljaju vremena izvršavanja odgovarajućih događaja. Konkretno, za datu sekvencu događaja $u \in X^+$, vrednost $[\![\mathcal{A}]\!](u)$ može se interpretirati kao trenutak u kome je sekvenca događaja u sigurno završena, dok vrednost $[\![\mathcal{A}]\!](u) = +\infty$ označava da ta sekvenca događaja nikada neće biti završena.

Ako za svaku reč $u \in X^*$ važi $[\![\mathcal{A}]\!](u) = -\infty$, odnosno ako je ponašanje automata jednako nultoj funkciji, kažemo da automat $\mathcal{A}$ ima ***nulto ponašanje***. U suprotnom, automat

ima *nenulto ponašanje*.

Za dati max-plus automat $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ definiše se njemu odgovarajući ***reverzni automat*** $\overline{\mathcal{A}} = (n, \overline{\sigma}^A, \{\overline{M}_x^A\}_{x\in X}, \overline{\tau}^A)$, gde je

$$\overline{\sigma}^A = (\tau^A)^T, \quad \overline{\tau}^A = (\sigma^A)^T, \quad \overline{M}_x^A = (M_x^A)^T, \text{ za svako } x \in X. \tag{43}$$

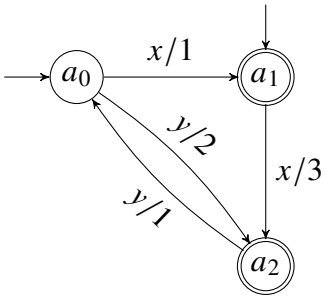
Drugim rečima, kod reverznog automata zamenjuju se inicijalni i završni vektor, dok se smerovi prelaza menjaju transponovanjem matrica prelaza.

Ako je operacija množenja u poluprstenu $\mathbb{S}$ komutativna, tada reverzni automat odgovara reverznoj reči. Naime, za svaku reč $u = x_1x_2\ldots x_k \in X^+$, gde je $\overline{u} = x_k\ldots x_2x_1$, važi $[\![\overline{\mathcal{A}}]\!](u) = [\![\mathcal{A}]\!](\overline{u})$.

Za razliku od fazi automata, gde vrednost $[\![\mathcal{A}]\!](u)$ predstavlja stepen raspoznavanja reči u, kod max-plus automata ona predstavlja maksimalnu težinu svih putanja označenih rečju u. Težina pojedinačne putanje dobija se primenom operacije $\otimes$, što odgovara sabiranju težina duž putanje, dok se među svim mogućim putanjama bira najveća vrednost primenom operacije $\oplus = \max$. Drugim rečima, ponašanje max-plus automata određeno je najvećom ukupnom težinom među svim putanjama označenim posmatranom reči.

Primer 2.2 *Neka je dat max-plus automat $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$. Neka je ulazni alfabet $X = \{x, y\}$ i neka automat ima $n = 3$ stanja. Inicijalni vektor, završni vektor i matrice prelaza dati su sa:*

$$\sigma^A = \begin{bmatrix} 0 \\ -1 \\ -\infty \end{bmatrix}^T, \quad \tau^A = \begin{bmatrix} -\infty \\ 2 \\ 1 \end{bmatrix}, \quad M_x^A = \begin{bmatrix} -\infty & 1 & -\infty \\ -\infty & -\infty & 3 \\ -\infty & -\infty & -\infty \end{bmatrix}, \quad M_y^A = \begin{bmatrix} -\infty & -\infty & 2 \\ -\infty & -\infty & -\infty \\ 1 & -\infty & -\infty \end{bmatrix}.$$



Slika 12: Grafovska reprezentacija max-plus automata iz primera 2.2

Stepen raspoznavanja prazne reči e automatom $\mathcal{A}$ računa se max-plus kompozicijom 38:

$$[\![\mathcal{A}]\!](e) = \sigma^A \otimes \tau^A = \begin{bmatrix} 0 & -1 & -\infty \end{bmatrix} \otimes \begin{bmatrix} -\infty \\ 2 \\ 1 \end{bmatrix} = \max(0 + (-\infty), -1 + 2, -\infty + 1) = 1.$$

Stepen raspoznavanja svake neprezne reči u automatom $\mathcal{A}$ računa se korišćenjem max-plus kompozicije matrica i vektora 36:

$$[\![\mathcal{A}]\!](x) = \sigma^A \otimes M_x^A \otimes \tau^A = \begin{bmatrix} 0 \\ -1 \\ -\infty \end{bmatrix}^T \otimes \begin{bmatrix} -\infty & 1 & -\infty \\ -\infty & -\infty & 3 \\ -\infty & -\infty & -\infty \end{bmatrix} \otimes \begin{bmatrix} -\infty \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} -\infty \\ 1 \\ 2 \end{bmatrix}^T \otimes \begin{bmatrix} -\infty \\ 2 \\ 1 \end{bmatrix} = 3.$$

$$[\![\mathcal{A}]\!](y) = \sigma^A \otimes M_y^A \otimes \tau^A = 3.$$

$$[\![\mathcal{A}]\!](xx) = \sigma^A \otimes M_x^A \otimes M_x^A \otimes \tau^A = 5.$$

$$[\![\mathcal{A}]\!](xy) = \sigma^A \otimes M_x^A \otimes M_y^A \otimes \tau^A = -\infty.$$

Drugim rečima, u zadatom max-plus automatu ne postoji putanja koja povezuje početno stanje sa završnim stanjem preko reči xy.

Po istom principu možemo izračunavati stepen raspoznavanja za svaku reč $u \in X^*$.

U ovoj glavi uvedeni su osnovni pojmovi iz teorije težinskih automata koji će predstavljati teorijsku osnovu za rezultate izložene u nastavku rada. Definisani su težinski automati, njihovo ponašanje i jezici koje raspoznaju, kao i njihove matrične i grafovske reprezentacije. Posebno su razmotrene dve značajne klase težinskih automata, fazi automati i max-plus automati, sa naglaskom na algebarske strukture nad kojima su definisani i interpretaciju njihovog ponašanja.

Max-plus automati, kao centralni objekat proučavanja u ovoj disertaciji, biće predmet detaljne analize u narednim poglavljima. Na osnovu pojmova uvedenih u ovoj glavi, u sledećem poglavlju uvode se simulacije i bisimulacije za max-plus automate, razmatraju njihova osnovna svojstva i proučavaju metode za njihovo određivanje.

Glava 3

Simulacije i bisimulacije za max-plus automate

U prethodnom poglavlju uvedeni su težinski automati kao uopštenje klasičnih konačnih automata kod kojih se prelazima, inicijalnim i završnim stanjima pridružuju vrednosti iz odgovarajuće algebarske strukture. Posebna pažnja posvećena je max-plus automatima, kod kojih se vrednosti prelaza, inicijalnih i završnih stanja uzimaju iz max-plus poluprstena, a ponašanje automata opisuje maksimalnu težinu svih putanja koje odgovaraju zadatoj ulaznoj reči [13, 15].

Max-plus automati imaju značajnu primenu u modelovanju diskretnih sistema događaja, kao što su proizvodni sistemi, transportne mreže, komunikacioni sistemi i problemi raspoređivanja poslova [10, 11, 12]. U takvim modelima težine prelaza najčešće predstavljaju trajanja, kašnjenja ili troškove, dok ponašanje automata opisuje najnepovoljniji (najduži) tok izvršavanja sistema.

Jedan od osnovnih problema pri analizi automata jeste poređenje njihovih ponašanja, odnosno ispitivanje da li su dva automata ekvivalentna ili da li je ponašanje jednog automata sadržano u ponašanju drugog. Kod max-plus automata, ovi problemi odgovaraju ispitivanju relacija između funkcija ponašanja automata, pri čemu se *inkluzija* (sadržavanje) može izraziti relacijom $[\![A]\!] \leqslant [\![B]\!]$, dok *ekvivalencija* odgovara jednakosti $[\![A]\!] = [\![B]\!]$. Direktno rešavanje ovih problema zahteva poređenje ponašanja automata nad svim rečima iz X^*, što u opštem slučaju može biti računarski zahtevno. Zbog toga se uvode simulacije i bisimulacije kao relacije između stanja automata koje omogućavaju da se svojstva njihovog ponašanja ispituju posredno, na nivou stanja i prelaza, umesto direktnim poređenjem ponašanja nad svim rečima.

Simulacije i bisimulacije prvobitno su razvijene u oblasti računarstva za analizu tranzicionih sistema, gde se koriste za poređenje sistema, kao i za njihovu analizu i redukciju broja stanja uz očuvanje određenih svojstava [4, 5]. Osnovna ideja jeste da se uspostavi relacija između stanja dva sistema, tako da se odgovarajući prelazi mogu međusobno pratiti. Na taj način, umesto ispitivanja svih mogućih izvršavanja sistema, moguće je razmatrati lokalne odnose između njihovih stanja i prelaza. U klasičnom pristupu, simulacije i bisimulacije posmatraju se kao binarne relacije između skupova stanja, koje se mogu predstaviti pomoću Bulovih matrica [6, 7, 8, 60, 61, 62, 63, 64]. Takav pristup opisuje postojanje veze između stanja, ali ne omogućava izražavanje kvantitativnih razlika u

težinama akumuliranim tokom izvršavanja, kao što su razlike u vremenima izvršavanja ili troškovima. Ograničenje ovog pristupa motivisalo je razvoj kvantitativnih proširenja simulacija i bisimulacija za različite klase težinskih automata.

U kontekstu max-plus automata, simulacije su prvi put razmatrane u radu [65], gde su definisane kao matrice nad max-plus poluprstenom koje zadovoljavaju odgovarajuće sisteme matričnih nejednačina. Kasnije su simulacije razmatrane i u kontekstu max-plus automata sa parcijalnim opažanjem [59], kao i u problemu determinacije max-plus automata [66]. Uslovi za postojanje simulacija između max-plus automata dodatno su proučavani u radu [67], dok su u radu [9] simulacije i bisimulacije sistematski razmatrane kao rešenja odgovarajućih sistema matričnih nejednačina, pri čemu su uvedeni postupci za određivanje njihovih najvećih rešenja.

U kvantitativnom pristupu, simulacije i bisimulacije predstavljaju se matricama čiji elementi pripadaju odgovarajućem max-plus poluprstenu i omogućavaju da se, pored postojanja relacije između stanja, u odnos uključe i kvantitativne informacije sadržane u težinama prelaza. Na taj način one daju kvantitativni opis odnosa između stanja različitih automata, za razliku od klasičnih binarnih relacija [9, 65, 66]. Za max-plus automate, dva tipa simulacija i četiri tipa bisimulacija definisana su kao rešenja odgovarajućih sistema matričnih nejednačina [9]. Time se ispitivanje postojanja simulacije ili bisimulacije određenog tipa, kao i određivanje najveće simulacije ili bisimulacije tog tipa, ukoliko postoji, svodi na rešavanje odgovarajućeg sistema matričnih nejednačina.

Isti koncept može se primeniti i na poređenje automata sa samim sobom. U tom slučaju govori se o autosimulacijama i autobisimulacijama, koje predstavljaju relacije simulacije, odnosno bisimulacije, između stanja jednog automata. Ove relacije mogu se koristiti za redukciju broja stanja automata i identifikovanje stanja koja su, u odgovarajućem smislu, međusobno povezana ili ekvivalentna.

Ove ideje razvijene su najpre za fazi automate [6, 68], a kasnije proširene na težinske automate nad aditivno-idempotentnim poluprstenima [8]. Iterativni pristup rešavanju ovih sistema, koji je primenjen u ovom poglavlju, uveden je u radu [9] i zasnovan je na metodama za rešavanje šire klase slabo linearnih sistema matričnih nejednačina razvijenim u radu [69].Rešavanje ovih sistema predstavlja značajan teorijski i algoritamski izazov, zbog čega se u nastavku posebna pažnja posvećuje postupcima za njihovo rešavanje i njihovim svojstvima [70, 71].

U ovom poglavlju razmatraju se dva tipa simulacija i četiri tipa bisimulacija za max-plus automate. Najpre se uvode *forward* i *backward* simulacije i analiziraju njihova osnovna svojstva. Nakon toga definišu se različiti tipovi bisimulacija i razmatraju odnosi između njih. Pokazuje se da postojanje odgovarajuće simulacije predstavlja dovoljan uslov za inkluziju ponašanja, dok postojanje odgovarajuće bisimulacije obezbeđuje ekvivalenciju

ponašanja posmatranih automata. Posebna pažnja posvećena je karakterizaciji najvećih simulacija i bisimulacija pomoću sistema matričnih nejednačina, kao i algoritmima za njihovo računanje. Razmatra se i problem terminacije ovih postupaka za računanje, koji nastaje usled mogućnosti pojave beskonačnih nizova matrica tokom iterativnog postupka. Teorijski rezultati izloženi u ovom poglavlju potkrepljeni su odgovarajućim primerima.

Većina rezultata predstavljenih u ovom poglavlju zasnovana je na koautorskom radu proisteklom iz istraživanja sprovedenog u okviru ove doktorske disertacije [9].

3.1 Relacije poređenja automata

U prethodnom poglavlju pokazano je da svaki max-plus automat određuje težinski jezik, odnosno preslikavanje koje svakoj ulaznoj reči pridružuje odgovarajuću vrednost iz max-plus poluprstena. Na taj način ponašanje automata može se posmatrati kao funkcija koja opisuje maksimalne težine svih putanja označenih posmatranim rečima.

Neka su data dva max-plus automata nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$

$$\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A) \quad \text{i} \quad \mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B).$$

Njihova ponašanja opisana su funkcijama

$$[\![\mathcal{A}]\!] : X^* \to \mathbb{R}_\infty \quad \text{i} \quad [\![\mathcal{B}]\!] : X^* \to \mathbb{R}_\infty.$$

Za automate $\mathcal{A}$ i $\mathcal{B}$ kažemo da su ekvivalentni ako važi

$$[\![\mathcal{A}]\!](u) = [\![\mathcal{B}]\!](u)\ , \quad \text{za svaku reč } u \in X^*.$$

Kažemo da je ponašanje automata $\mathcal{A}$ sadržano u ponašanju automata $\mathcal{B}$ ako važi

$$[\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u)\ , \quad \text{za svaku reč } u \in X^*.$$

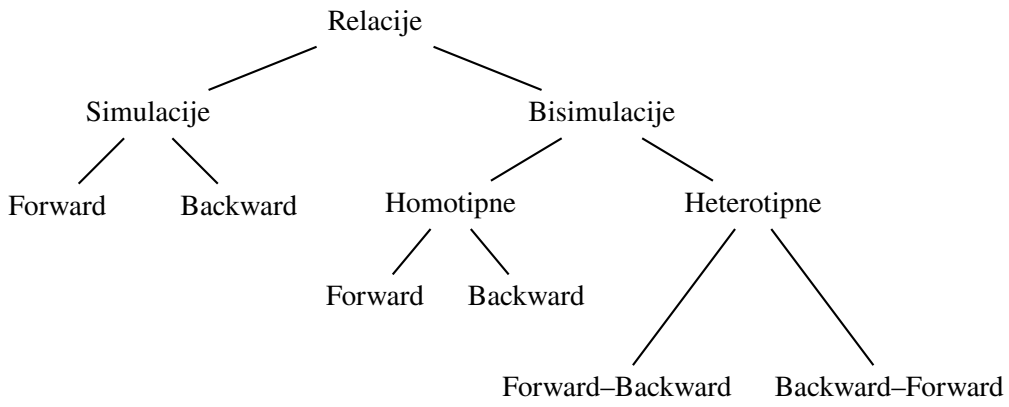
Direktna provera ovih uslova zahteva poređenje vrednosti koje automati pridružuju svim rečima iz skupa X^*. Kako je skup X^* u opštem slučaju beskonačan, ovakav pristup nije pogodan za efikasno poređenje ponašanja automata. Iz tog razloga se umesto direktnog poređenja težinskih jezika uvode relacije između stanja automata koje omogućavaju izvođenje zaključaka o njihovom ponašanju. Najznačajnije među njima su simulacije i bisimulacije, koje omogućavaju da se informacije o inkluziji ili ekvivalenciji ponašanja izvedu iz lokalnih uslova definisanih nad stanjima i prelazima automata.

Kod max-plus automata ovakve relacije nisu samo binarne, već nose dodatne kvantitativne informacije. Zato se relacija između skupa stanja automata $\mathcal{A}$ i skupa stanja automata $\mathcal{B}$ predstavlja matricom $U \in \mathbb{R}_\infty^{m\times n}$. Element $U(i, j)$ opisuje stepen povezanosti stanja i automata $\mathcal{A}$ sa stanjem j automata $\mathcal{B}$. Na taj način matrica U predstavlja max-plus relaciju između skupova stanja dva automata.

Intuitivno, simulacija između automata $\mathcal{A}$ i $\mathcal{B}$ omogućava da automat $\mathcal{B}$ prati ponašanje automata $\mathcal{A}$, dok bisimulacija omogućava praćenje u oba smera. Budući da su u max-plus slučaju predstavljene matricama, ove relacije, pored informacije o postojanju odgovarajuće veze između automata, daju i kvantitativnu informaciju o odnosu njihovih stanja.

U kontekstu max-plus automata, pojmovi ekvivalentnosti i sadržavanja mogu imati i prirodno praktično tumačenje, posebno u modelima u kojima težine predstavljaju trajanja, kašnjenja ili troškove. Tada sadržavanje ponašanja jednog automata u ponašanju drugog znači da vrednost koju prvi automat pridružuje svakoj ulaznoj reči nije veća od odgovarajuće vrednosti drugog automata. Kada težine predstavljaju vremena izvršavanja, to se može tumačiti kao zahtev da jedan sistem ni u jednom slučaju ne bude sporiji od drugog, posmatrano u okviru najnepovoljnijeg scenarija. Sa druge strane, ekvivalentnost ponašanja znači da dva automata za svaku ulaznu reč daju istu vrednost, odnosno da pri tumačenju težina kao vremena izvršavanja imaju isto ponašanje u najnepovoljnijem slučaju. Ovakvo tumačenje čini probleme sadržavanja i ekvivalentnosti značajnim za analizu performansi i poređenje modela diskretnih sistema događaja.

U nastavku se uvode relacije simulacije i bisimulacije između max-plus automata, koje predstavljaju osnovni alat za poređenje njihovog ponašanja. Za razliku od samog ispitivanja inkluzije i ekvivalencije jezika, simulacije i bisimulacije pružaju dodatnu informaciju o međusobnom odnosu stanja posmatranih automata. U zavisnosti od smera u kome se porede prelazi i stanja automata razlikuju se forward i backward simulacije. Na osnovu njih definišu se četiri tipa bisimulacija: forward, backward, forward-backward i backward-forward bisimulacije. Ove relacije omogućavaju formulisanje dovoljnih uslova za inkluziju, odnosno ekvivalenciju jezika koje automati prepoznaju. Postojanje više tipova simulacija i bisimulacija nije samo formalna razlika. U opštem slučaju može postojati bisimulacija jednog tipa između automata, dok bisimulacije ostalih tipova ne postoje. Međusobna povezanost i klasifikacija tipova simulacija i bisimulacija prikazana je na Slici 13.



Slika 13: Klasifikacija simulacija i bisimulacija automata

3.2 Simulacije i bisimulacije

U prethodnoj sekciji razmatrane su relacije poređenja max-plus automata zasnovane na inkluziji i ekvivalenciji njihovih ponašanja. Međutim, takve relacije ne pružaju neposredan uvid u strukturne veze između automata koje dovode do jednakosti ili poređenja njihovih jezika. Zbog toga se uvode simulacije i bisimulacije kao posebne relacije između stanja automata koje omogućavaju poređenje njihovog ponašanja na nivou prelaza i stanja.

Simulacije omogućavaju poređenje ponašanja automata u jednom smeru, odnosno utvrđivanje da li se ponašanje jednog automata može predstaviti ponašanjem drugog. Bisimulacije predstavljaju oblik poređenja kojim se zahteva međusobna usklađenost ponašanja posmatranih automata. Kao što će biti pokazano u nastavku, postojanje odgovarajuće simulacije ili bisimulacije omogućava izvođenje zaključaka o inkluziji ili ekvivalenciji ponašanja automata.

U nastavku će biti uvedeni različiti tipovi simulacija i bisimulacija za max-plus automate i razmotrena njihova osnovna svojstva.

3.2.1 Simulacije max-plus automata

Simulacije predstavljaju relacije između stanja automata koje omogućavaju izvođenje zaključaka o sadržavanju ponašanja automata bez direktnog poređenja vrednosti koje automati pridružuju svim rečima iz skupa X^*. Osnovna ideja simulacije jeste da automat može da prati ponašanje drugog automata uz očuvanje odgovarajuće strukture prelaza, kao i inicijalnih i završnih vrednosti.

Kod max-plus automata simulacije se posmatraju kao kvantitativne relacije između skupova stanja. Za razliku od klasičnih simulacija koje se predstavljaju Bulovim matricama i opisuju samo postojanje ili nepostojanje veze između stanja, max-plus simulacije svakom paru stanja pridružuju vrednost iz max-plus poluprstena. Na taj način dobija se kvantitativni opis odnosa između stanja posmatranih automata.

U zavisnosti od načina na koji se usklađuju inicijalni vektori, prelazi i završni vektori automata, razlikujemo dva tipa simulacija: *forward* i *backward simulacije*.

Neka su dati automati $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$ i neka je $U \in \mathbb{R}_\infty^{m\times n}$ nenulta matrica koja predstavlja max-plus relaciju između skupova stanja automata $\mathcal{A}$ i $\mathcal{B}$.

Forward simulacija predstavlja relaciju između stanja dva max-plus automata koja omogućava da automat $\mathcal{B}$ prati ponašanje automata $\mathcal{A}$. Intuitivno, svaki prelaz i svaka putanja automata $\mathcal{A}$ mogu biti predstavljeni odgovarajućim prelazima i putanjama automata $\mathcal{B}$, uz očuvanje odgovarajućih težinskih vrednosti.

Matrica U naziva se ***forward simulacija*** između automata $\mathcal{A}$ i $\mathcal{B}$ ako zadovoljava sledeći sistem matričnih nejednačina:

(1-fs) $\sigma^A \leqslant \sigma^B \otimes U^\top$

(2-fs) $U^\top \otimes M_x^A \leqslant M_x^B \otimes U^\top \quad (x \in X)$

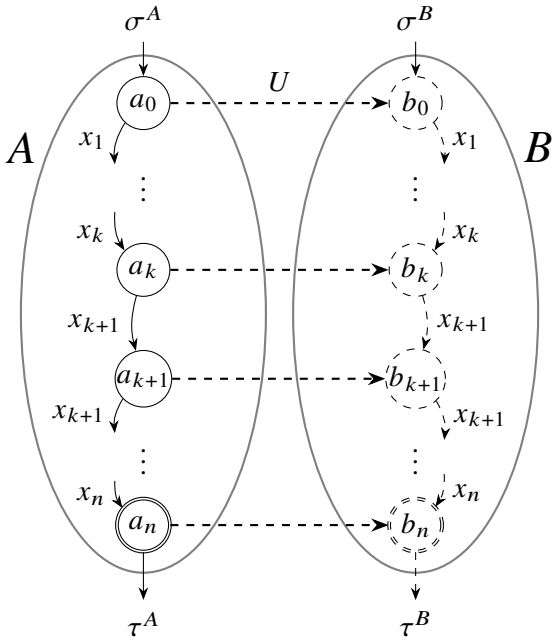
(3-fs) $U^\top \otimes \tau^A \leqslant \tau^B$

Prvi uslov povezuje inicijalne težinske vektore automata $\mathcal{A}$ i $\mathcal{B}$. On obezbeđuje da inicijalne vrednosti automata $\mathcal{A}$ budu povezane sa odgovarajućim inicijalnim vrednostima automata $\mathcal{B}$ preko relacije određene matricom U.

Drugi uslov predstavlja usklađenost prelaza automata. Za svaki simbol $x \in X$ zahteva se da prelazi automata $\mathcal{A}$, posmatrani kroz relaciju U, budu obuhvaćeni odgovarajućim prelazima automata $\mathcal{B}$, čime se obezbeđuje da se izvršavanje prelaza u automatu $\mathcal{A}$ može pratiti odgovarajućim izvršavanjem prelaza u automatu $\mathcal{B}$.

Treći uslov povezuje završne težinske vektore automata. Njime se obezbeđuje da završne vrednosti automata $\mathcal{A}$ budu očuvane prilikom njihovog predstavljanja pomoću automata $\mathcal{B}$.

Drugim rečima, forward simulacija omogućava da automat $\mathcal{B}$ prati ponašanje automata $\mathcal{A}$ u smeru izvršavanja prelaza, od inicijalnih ka završnim stanjima. Kao što će biti pokazano u nastavku, postojanje ovakve relacije predstavlja dovoljan uslov za sadržavanje (inkluziju) ponašanja automata $\mathcal{A}$ u ponašanju automata $\mathcal{B}$. Intuitivni prikaz forward simulacije između dva max-plus automata dat je na slici 14.



Slika 14: Forward simulacija između max-plus automata

Pored forward simulacija, kod max-plus automata razmatraju se i backward simulacije. Ova dva tipa simulacija zasnivaju se na istoj ideji poređenja ponašanja automata pomoću relacije između njihovih stanja, ali se razlikuju u načinu usklađivanja inicijalnih i završnih vektora, kao i prelaza automata.

Backward simulacija predstavlja relaciju između stanja dva max-plus automata koja omogućava poređenje njihovog ponašanja u obrnutom smeru. Intuitivno, usklađivanje se posmatra polazeći od završnih vrednosti i nastavlja se unazad kroz prelaze automata prema inicijalnim vrednostima.

Matrica U naziva se ***backward simulacija*** između automata $\mathcal{A}$ i $\mathcal{B}$ ako zadovoljava sledeći sistem matričnih nejednačina:

$$\begin{aligned}
&\text{(1-bs)} \quad \tau^A \leqslant U \otimes \tau^B \\
&\text{(2-bs)} \quad M_x^A \otimes U \leqslant U \otimes M_x^B \quad (x \in X) \\
&\text{(3-bs)} \quad \sigma^A \otimes U \leqslant \sigma^B
\end{aligned}$$

Dakle, dok forward simulacija usklađivanje vrši u smeru od inicijalnih ka završnim stanjima, backward simulacija to čini u suprotnom smeru, polazeći od završnih vrednosti ka inicijalnim.

Veza između forward i backward simulacija može se izraziti pomoću reverznih automata. Naime, matrica U je backward simulacija između max-plus automata $\mathcal{A}$ i $\mathcal{B}$ ako i samo ako je U forward simulacija između odgovarajućih reverznih automata $\overline{\mathcal{A}}$ i $\overline{\mathcal{B}}$.

3.2.2 Bisimulacije max-plus auomata

Bisimulacije predstavljaju simetričan oblik simulacije, kod koga se zahteva da svaki od posmatranih automata može simulirati ponašanje drugog automata. Kao i kod simulacija max-plus automata i bisimulacije se predstavljaju pomoću matrica nad max-plus poluprstenom.

Neka su dati automati $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$ i neka je $U \in \mathbb{R}_\infty^{m\times n}$ nenulta matrica koja predstavlja max-plus relaciju između njihovih skupova stanja. Kombinovanjem forward i backward simulacija između automata $\mathcal{A}$ i $\mathcal{B}$ dobijaju se četiri tipa bisimulacija.

Ako su i matrica U i njena transponovana matrica $U^\top$ *forward simulacije*, tada se U naziva ***forward bisimulacija*** između automata $\mathcal{A}$ i $\mathcal{B}$. Drugim rečima, matrica U je *forward bisimulacija* ako zadovoljava sledeći sistem matričnih nejednačina:

$$\begin{aligned}
&\text{(1-fb)} \quad \sigma^A \leqslant \sigma^B \otimes U^\top, \;\; \sigma^B \leqslant \sigma^A \otimes U \\
&\text{(2-fb)} \quad U^\top \otimes M_x^A \leqslant M_x^B \otimes U^\top, \;\; U \otimes M_x^B \leqslant M_x^A \otimes U \quad (x \in X) \\
&\text{(3-fb)} \quad U^\top \otimes \tau^A \leqslant \tau^B, \;\; U \otimes \tau^B \leqslant \tau^A
\end{aligned}$$

Analogno, ako su i matrica U i njena transponovana matrica $U^\top$ *backward simulacije*, tada se U naziva ***backward bisimulacija*** između automata $\mathcal{A}$ i $\mathcal{B}$. Drugim rečima, matrica U je *backward bisimulacija* ako zadovoljava sledeći sistem matričnih nejednačina:

$$\begin{aligned}
&\text{(1-bb)} \quad \tau^A \leqslant U \otimes \tau^B, \;\; \tau^B \leqslant U^\top \otimes \tau^A \\
&\text{(2-bb)} \quad M_x^A \otimes U \leqslant U \otimes M_x^B, \;\; M_x^B \otimes U^\top \leqslant U^\top \otimes M_x^A \quad (x \in X) \\
&\text{(3-bb)} \quad \sigma^A \otimes U \leqslant \sigma^B, \;\; \sigma^B \otimes U^\top \leqslant \sigma^A
\end{aligned}$$

Veza između forward i backward bisimulacija može se izraziti i pomoću reverznih automata. Matrica U je *backward bisimulacija* između automata $\mathcal{A}$ i $\mathcal{B}$ ako i samo ako je U *forward bisimulacija* između reverznih automata $\overline{\mathcal{A}}$ i $\overline{\mathcal{B}}$.

Pored forward i backward bisimulacija, razmatraju se i mešoviti oblici bisimulacija koji nastaju kombinovanjem različitih tipova simulacija.

Ako je matrica U forward simulacija, a njena transponovana matrica $U^\top$ backward simulacija između posmatranih automata, tada se U naziva ***forward-backward bisimulacija***. Ovaj tip bisimulacije određen je sledećim sistemom matričnih nejednačina:

$$\begin{aligned}
&\text{(1-fbb)} \quad \sigma^A \leqslant \sigma^B \otimes U^\top, \;\; \tau^B \leqslant U^\top \otimes \tau^A \\
&\text{(2-fbb)} \quad U^\top \otimes M_x^A \leqslant M_x^B \otimes U^\top, \;\; M_x^B \otimes U^\top \leqslant U^\top \otimes M_x^A \quad (x \in X) \\
&\text{(3-fbb)} \quad U^\top \otimes \tau^A \leqslant \tau^B, \;\; \sigma^B \otimes U^\top \leqslant \sigma^A,
\end{aligned}$$

Sa druge strane, ako je matrica U backward simulacija, a njena transponovana matrica $U^\top$ forward simulacija, tada se U naziva ***backward-forward bisimulacija***.

$$\begin{aligned}
&\text{(1-bfb)} \quad \tau^A \leqslant U \otimes \tau^B, \;\; \sigma^B \leqslant \sigma^A \otimes U \\
&\text{(2-bfb)} \quad M_x^A \otimes U \leqslant U \otimes M_x^B, \;\; U \otimes M_x^B \leqslant M_x^A \otimes U \quad (x \in X) \\
&\text{(3-bfb)} \quad \sigma^A \otimes U \leqslant \sigma^B, \;\; U \otimes \tau^B \leqslant \tau^A
\end{aligned}$$

Veza između forward-backward i backward-forward bisimulacija može se izraziti i pomoću reverznih automata $\overline{\mathcal{A}}$ i $\overline{\mathcal{B}}$. Matrica U je backward-forward bisimulacija između automata $\mathcal{A}$ i $\mathcal{B}$ ako i samo ako je U forward-backward bisimulacija između reverznih automata $\overline{\mathcal{A}}$ i $\overline{\mathcal{B}}$.

Na ovaj način dobijaju se četiri različita tipa bisimulacija za max-plus automate. Svaki od njih predstavlja poseban način međusobnog usklađivanja ponašanja dva automata, pri čemu se razlikuju prema smeru u kome se usklađuju prelazi, inicijalne i završne vrednosti.

Važno je napomenuti da navedeni tipovi bisimulacija u opštem slučaju nisu međusobno ekvivalentni. Postojanje jednog tipa bisimulacije između dva max-plus automata ne mora da implicira postojanje ostalih tipova. Zbog toga različiti oblici bisimulacija pružaju različite mogućnosti za poređenje ponašanja automata.

U daljem tekstu koristićemo sledeću skraćenu terminologiju. Ako je $\xi \in \{\text{fs}, \text{bs}\}$, tada svako nenulto rešenje sistema $(1-\xi)$, $(2-\xi)$ i $(3-\xi)$ nazivamo ***simulacija tipa*** ξ ili kraće ***ξ-simulacija***. Ako je $\xi \in \{\text{fb}, \text{bb}, \text{fbb}, \text{bfb}\}$, tada svako nenulto rešenje odgovarajućeg sistema $(1-\xi)$, $(2-\xi)$ i $(3-\xi)$ nazivamo ***bisimulacija tipa*** ξ ili ***ξ-bisimulacija***.

Uslovi kojima su definisane prethodno navedene simulacije i bisimulacije mogu se posmatrati kao sistemi matričnih nejednačina sa nepoznatom matricom U nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$.

Zbog toga se problem ispitivanja postojanja simulacija i bisimulacija između dva max-plus automata svodi na problem određivanja nenultih rešenja odgovarajućih sistema matričnih nejednačina. Ovakav pristup predstavlja osnovu algoritama za računanje najvećih simulacija i bisimulacija koji će biti razmatrani u nastavku.

Naredna teorema pokazuje osnovni značaj simulacija i bisimulacija u poređenju max-plus automata. Postojanje simulacije predstavlja dovoljan uslov za inkluziju ponašanja automata, odnosno za inkluziju težinskih jezika koje oni raspoznaju, dok postojanje bisimulacije predstavlja dovoljan uslov za njihovu ekvivalenciju.

Teorema 3.1 *Neka su $\mathcal{A}$ i $\mathcal{B}$ max-plus automati nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$. Tada važi:*

(a) *Ako postoji simulacija bilo kog tipa između automata $\mathcal{A}$ i $\mathcal{B}$, tada je $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!]$.*

(b) *Ako postoji bisimulacija bilo kog tipa između automata $\mathcal{A}$ i $\mathcal{B}$, tada je $[\![\mathcal{A}]\!] = [\![\mathcal{B}]\!]$.*

Dokaz 3.1 *Dokazaćemo najpre tvrđenje* (a) *za slučaj forward simulacije.*

Neka je U forward simulacija između automata $\mathcal{A}$ i $\mathcal{B}$. Tada važe (1-fs)*,* (2-fs) *i* (3-fs)*.*

Najpre ćemo indukcijom po dužini reči pokazati da za svaku reč $u \in X^*$ važi prošireni oblik uslova (2-fs)*, odnosno da je $U^\top \otimes M_u^A \leqslant M_u^B \otimes U^\top$.*

Tvrdnja očigledno važi za praznu reč, jer je $M_e^A = M_e^B = I$, kao i za svaku reč dužine jedan, $|u| = 1$, što je upravo uslov (2-fs)*.*

Pretpostavimo da za neku reč $u \in X^*$ važi $U^\top \otimes M_u^A \leqslant M_u^B \otimes U^\top$.

Neka je $v = u \cdot x$, gde je $x \in X$. Koristeći pretpostavku i uslov (2-fs)*, dobijamo:*

$$U^\top \otimes M_v^A = U^\top \otimes M_u^A \otimes M_x^A \leqslant M_u^B \otimes U^\top \otimes M_x^A \leqslant M_u^B \otimes M_x^B \otimes U^\top = M_v^B \otimes U^\top, \quad (44)$$

Time je pokazano da prošireni oblik uslova (2-fs) *važi za svaku reč $u \in X^*$. Na osnovu navedenog, pokažimo inkluziju jezika raspoznatih automatima.*

Za proizvoljnu reč $u \in X^*$, koristeći 36, 44, uslove (1-fs) *i* (3-fs)*, dobijamo:*

$$\begin{aligned}[\![\mathcal{A}]\!](u) &= \sigma^A \otimes M_u^A \otimes \tau^A \leqslant \sigma^B \otimes U^\top \otimes M_u^A \otimes \tau^A \leqslant \\ &\leqslant \sigma^B \otimes M_u^B \otimes U^\top \otimes \tau^A \leqslant \sigma^B \otimes M_u^B \otimes \tau^B = [\![\mathcal{B}]\!](u).\end{aligned}$$

Dakle, $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!]$*, čime je dokazano tvrđenje (a) za forward simulacije. Dokaz za backward simulacije zasniva se na istim idejama i sprovodi se analogno, čime je dokazano tvrđenje* (a).

Tvrđenje (b) *neposredno sledi iz tvrđenja* (a). *Naime, postojanje bisimulacije bilo kog tipa podrazumeva postojanje odgovarajućih simulacija u oba smera, pa važe inkluzije* $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!]$ *i* $[\![\mathcal{B}]\!] \leqslant [\![\mathcal{A}]\!]$*, odakle neposredno sledi* $[\![\mathcal{A}]\!] = [\![\mathcal{B}]\!]$. □

Pošto simulacije i bisimulacije omogućavaju zaključivanje o inkluziji i ekvivalenciji ponašanja automata, od posebnog značaja je određivanje najvećih simulacija i bisimulacija datog tipa. Ovom problemu posvećena je naredna sekcija.

3.2.3 Simulacije i bisimulacije na jednom automatu

Do sada su simulacije i bisimulacije posmatrane kao relacije između dva max-plus automata. Međutim, ove relacije mogu se razmatrati i između stanja jednog automata. U tom slučaju govori se o simulacijama i bisimulacijama na jednom automatu. Pošto je automat trivijalno ekvivalentan samom sebi, ovakve relacije nisu od značaja za utvrđivanje ekvivalencije dva automata, već pre svega za redukciju broja stanja datog automata.

Osnovna ideja redukcije jeste da se stanja koja imaju odgovarajuće, odnosno (dovoljno) slično ponašanje mogu grupisati i zameniti manjim brojem reprezentativnih stanja. Na taj način se vrši faktorizacija automata u odnosu na posmatranu simulaciju ili bisimulaciju, čime se dobija automat sa manjim brojem stanja. Postupak faktorizacije zasniva se na određivanju odgovarajućih grupa stanja i konstrukciji automata koji predstavlja redukovani oblik početnog automata. Pri tome se nastoji da se, uprkos smanjenju broja stanja, očuvaju relevantne karakteristike ponašanja početnog automata. Drugim rečima, redukovani automat treba da predstavlja adekvatnu aproksimaciju ili reprezentaciju originalnog automata u odnosu na relaciju koja se koristi za redukciju.

Ovakav pristup je posebno značajan u slučaju kalsičnih i slabih bisimulacija. Slabljenjem uslova koji definišu simulaciju ili bisimulaciju može se dozvoliti određeni stepen odstupanja između ponašanja povezanih stanja, čime se otvara mogućnost za značajniju redukciju broja stanja. Na taj način moguće je uspostaviti ravnotežu između složenosti automata i stepena očuvanja njegovog ponašanja.

Značaj ovakve redukcije posebno dolazi do izražaja kod automata sa velikim brojem stanja. Umesto analize veoma složene mreže sa velikim brojem čvorova, može se posmatrati njen redukovani oblik sa znatno manjim brojem stanja, čije ponašanje u odgovarajućoj meri predstavlja ponašanje originalnog automata.

3.3 Najveće simulacije i bisimulacije

U prethodnim sekcijama uvedeni su različiti tipovi simulacija i bisimulacija između max-plus automata i pokazano je da njihovo postojanje predstavlja potvrdu inkluzije, odnosno ekvivalencije jezika automata. Međutim, između dva automata u opštem slučaju može postojati više različitih simulacija ili bisimulacija istog tipa, zbog čega se prirodno postavlja pitanje određivanja najvećeg rešenja među njima. Određivanje najveće simulacije ili bisimulacije od posebnog je značaja, budući da svaka druga simulacija ili bisimulacija istog tipa predstavlja njenu podrelaciju. Stoga najveća simulacija, ukoliko postoji, sadrži svaku drugu simulaciju istog tipa, pa u potpunosti opisuje sve simulacije tog tipa između posmatranih automata. Drugim rečima, tada svaka druga simulacija ili bisimulacija istog tipa predstavlja matricu manju ili jednaku toj matrici u prirodnom uređenju na $\mathbb{R}_{\infty}^{m\times n}$. Zbog toga najveća simulacija, odnosno bisimulacija, sadrži potpunu informaciju o svim simulacijama i bisimulacijama datog tipa između posmatranih automata.

Kao što je pokazano u prethodnoj sekciji, svaki od šest tipova simulacija i bisimulacija definisan je odgovarajućim sistemom matričnih nejednačina. Zbog toga se problem određivanja simulacije ili bisimulacije može posmatrati kao problem pronalaženja nenultog rešenja odgovarajućeg sistema matričnih nejednačina. Pri tome, sistemi određeni uslovima (2-ξ) predstavljaju poseban slučaj slabo linearnih sistema matričnih nejednačina proučavanih u radu [69], dok uslovi (3-ξ) omogućavaju određivanje najveće matrice koja može biti rešenje tog sistema. Na osnovu ovih rezultata moguće je konstruisati iterativni postupak kojim se određuje najveće rešenje sistema sastavljenog od uslova (2-ξ) i (3-ξ). Nakon toga proverava se da li dobijeno rešenje zadovoljava i uslov (1-ξ), čime se odlučuje o postojanju simulacije ili bisimulacije odgovarajućeg tipa.

U nastavku će biti opisan postupak za određivanje najvećih simulacija i bisimulacija između dva max-plus automata. Osnovna ideja zasniva se na konstrukciji opadajućeg niza matrica čiji infimum predstavlja najveće rešenje odgovarajućeg sistema matričnih nejednačina.

Kao što je ranije rečeno, oznaka ξ predstavlja jedan od šest tipova simulacija i bisimulacija, odnosno $\xi \in \{\text{fs}, \text{bs}, \text{fb}, \text{bb}, \text{fbb}, \text{bfb}\}$. Za svaki od navedenih tipova konstruiše se odgovarajući opadajući niz matrica $\{U_s^{\xi}\}_{s\in\mathbb{N}}$, čiji infimum predstavlja najveće rešenje sistema određenog uslovima (2-ξ) i (3-ξ).

Naredna teorema predstavlja osnovu postupka za testiranje postojanja simulacija i bisimulacija, kao i za određivanje njihovih najvećih rešenja.

Teorema 3.2 *Neka su* $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ *i* $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ *max-plus automati nad kompletnim max-plus poluprstenom* $\mathbb{R}_{\infty}$.

Za proizvoljno $\xi \in \{\text{fs}, \text{bs}, \text{fb}, \text{bb}, \text{fbb}, \text{bfb}\}$ *neka je* $\{U_s^{\xi}\}_{s\in\mathbb{N}}$ *opadajući niz matrica iz* $\mathbb{R}_{\infty}^{m\times n}$ *definisan na sledeći način:*

$$
\begin{aligned}
U_1^{\mathrm{fs}} &= \tau^A \backslash \tau^B, & U_{s+1}^{\mathrm{fs}} &= U_s^{\mathrm{fs}} \wedge \bigwedge_{x \in X} \left(\left(M_x^B \otimes (U_s^{\mathrm{fs}})^\top \right) \not/ M_x^A \right)^\top, \\
U_1^{\mathrm{bs}} &= \sigma^A \backslash \sigma^B, & U_{s+1}^{\mathrm{bs}} &= U_s^{\mathrm{bs}} \wedge \bigwedge_{x \in X} M_x^A \backslash (U_s^{\mathrm{bs}} \otimes M_x^B), \\
U_1^{\mathrm{fb}} &= (\tau^A \backslash \tau^B) \wedge (\tau^A \not/ \tau^B), & U_{s+1}^{\mathrm{fb}} &= U_s^{\mathrm{fb}} \wedge \bigwedge_{x \in X} \left(\left(M_x^B \otimes (U_s^{\mathrm{fb}})^\top \right) \not/ M_x^A \right)^\top \wedge \left((M_x^A \otimes U_s^{\mathrm{fb}}) \not/ M_x^B \right), \\
U_1^{\mathrm{bb}} &= (\sigma^A \backslash \sigma^B) \wedge (\sigma^A \not/ \sigma^B), & U_{s+1}^{\mathrm{bb}} &= U_s^{\mathrm{bb}} \wedge \bigwedge_{x \in X} \left(M_x^A \backslash (U_s^{\mathrm{bb}} \otimes M_x^B) \right) \wedge \left(M_x^B \backslash \left((U_s^{\mathrm{bb}})^\top \otimes M_x^A \right) \right)^\top, \\
U_1^{\mathrm{fbb}} &= (\sigma^A \not/ \sigma^B) \wedge (\tau^A \backslash \tau^B), & U_{s+1}^{\mathrm{fbb}} &= U_s^{\mathrm{fbb}} \wedge \bigwedge_{x \in X} \left(\left(M_x^B \otimes (U_s^{\mathrm{fbb}})^\top \right) \not/ M_x^A \right)^\top \wedge \left(M_x^B \backslash \left((U_s^{\mathrm{fbb}})^\top \otimes M_x^A \right) \right)^\top, \\
U_1^{\mathrm{bfb}} &= (\sigma^A \backslash \sigma^B) \wedge (\tau^A \not/ \tau^B), & U_{s+1}^{\mathrm{bfb}} &= U_s^{\mathrm{bfb}} \wedge \bigwedge_{x \in X} \left(M_x^A \backslash (U_s^{\mathrm{bfb}} \otimes M_x^B) \right) \wedge \left((M_x^A \otimes U_s^{\mathrm{bfb}}) \not/ M_x^B \right),
\end{aligned}
$$

za svako $s \in \mathbb{N}$.

Neka je $\widehat{U}^\xi$ *infimum niza* $\{U_s^\xi\}_{s \in \mathbb{N}}$. *Tada važe sledeća tvrđenja:*

(a) *Matrica* $\widehat{U}^\xi$ *predstavlja najveće rešenje sistema matričnih nejednačina koji čine uslovi* (2-ξ) *i* (3-ξ).

(b) *Ako je, pored toga,* $\widehat{U}^\xi$ *nenulta matrica koja zadovoljava i uslove* (1-ξ), *tada je* $\widehat{U}^\xi$ *najveća simulacija, odnosno bisimulacija tipa* ξ *između automata* $\mathcal{A}$ *i* $\mathcal{B}$.

(c) *Ako je* $\widehat{U}^\xi$ *nulta matrica ili ne zadovoljava uslove* (1-ξ), *tada između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *ne postoji simulacija, odnosno bisimulacija tipa* ξ.

(d) *Ako za neko* $s \in \mathbb{N}$ *važi da je* $U_s^\xi = U_{s+1}^\xi$, *onda je* $\widehat{U}^\xi = U_s^\xi$.

Dokaz 3.2 *Dokazaćemo samo slučaj* $\xi =$ fb. *Preostali slučajevi se dokazuju analogno.*

(a) *Na osnovu osobine reziduacije 40, matrica* $U \in \mathbb{R}_\infty^{m \times n}$ *zadovoljava sistem nejednačina* (3-fb) *ako i samo ako važe:*

$$U^T \otimes \tau^A \leq \tau^B \iff U^T \leq \tau^B \not/ \tau^A \iff U \leq (\tau^B \not/ \tau^A)^\top \iff U \leq \tau^A \backslash \tau^B \tag{45}$$

$$U \otimes \tau^B \leq \tau^A \iff U \leq \tau^A \not/ \tau^B \tag{46}$$

Kako nejednačine 45 i 46 moraju biti istovremeno zadovoljene, važi:

$$U \leq (\tau^A \not/ \tau^B) \wedge (\tau^A \backslash \tau^B) = U_1^{fb}. \tag{47}$$

Dakle, matrica U zadovoljava sistem (3-fb) *ako i samo ako je manja ili jednaka matrici* U_1^{fb}, *tj.* $U \leq U_1^{fb}$. *Prema tome, najveće rešenje sistema koji čine uslovi* (2-fb) *i* (3-fb) *jednako je najvećem rešenju sistema* (2-fb) *koje je ograničeno matricom* U_1^{fb}.

Sa druge strane, primenom svojstva reziduacije 40 na sisteme nejednačina (2-fb)*, za svako $x \in X$ dobijamo:*

$$\begin{aligned} U^\top \otimes M_x^A \leqslant M_x^B \otimes U^\top &\iff U^\top \leqslant (M_x^B \otimes U^\top) \not/ M_x^A \\ &\iff U \leqslant \left((M_x^B \otimes U^\top) \not/ M_x^A\right)^\top \end{aligned} \tag{48}$$

$$U \otimes M_x^B \leqslant M_x^A \otimes U \iff U \leqslant M_x^A \setminus (U \otimes M_x^B) \tag{49}$$

Kako nejednačine 48 i 49 moraju biti istovremeno zadovoljene za svako $x \in X$ dobijamo:

$$U \leq \bigwedge_{x \in X} \left[\left((M_x^B \otimes U^\top) \not/ M_x^A\right)^\top \wedge \left(M_x^A \setminus (U \otimes M_x^B)\right) \right] \tag{50}$$

Prema tome, određivanje najvećeg rešenja sistema (2-fb) i (3-fb) svodi se na određivanje najvećeg rešenja sistema 47 i 50.

Prema rezultatu o rešavanju slabo linearnih sistema razmatranom u [69], sistem 50 predstavlja slabo linearni sistem matričnih nejednačina. Na osnovu prethodno određenog ograničenja 47, njegovo najveće rešenje može se odrediti iterativnim postupkom za rešavanje slabo linearnih sistema. Naime, za početnu matricu U_1^{fb} konstruiše se niz matrica definisan sa

$$\begin{aligned} &U_1^{fb} = U_1^{fb}, \\ &U_{s+1}^{fb} = U_s^{fb} \wedge \bigwedge_{x \in X} \left[\left((M_x^B \otimes (U_s^{fb})^\top) \not/ M_x^A\right)^\top \wedge \left(M_x^A \setminus (U_s^{fb} \otimes M_x^B)\right) \right], \ s \geq 1. \end{aligned} \tag{51}$$

Prema rezultatu iz [69], infimum konstruisanog niza predstavlja najveće rešenje sistema (2-fb) *uz ograničenje $U \leq U_1^{fb}$. Kako je prema 47 uslov $U \leq U_1^{fb}$ ekvivalentan sistemu nejednačina* (3-fb)*, sledi da je najveće rešenje sistema koji čine uslovi* (2-fb) *i* (3-fb)

$$\widehat{U}^{fb} = \bigwedge_{s \geq 1} U_s^{fb}. \tag{52}$$

Time je dokazano da je $\widehat{U}^{fb}$ najveće rešenje sistema (2-fb) *i* (3-fb)*. Prema tome, ukoliko matrica $\widehat{U}^{fb}$ zadovoljava i uslov* (1-fb)*, ona predstavlja najveću forward bisimulaciju između automata $\mathcal{A}$ i $\mathcal{B}$.*

(b) *Pretpostavimo da je $\widehat{U}^{\text{fb}}$ nenulta matrica koja zadovoljava uslov* (1-fb)*.*

Prema tvrđenju (a)*, matrica $\widehat{U}^{\text{fb}}$ predstavlja najveće rešenje sistema koji čine uslovi* (2-fb) *i* (3-fb)*. Kako, po pretpostavci, zadovoljava i uslov* (1-fb)*, sledi da ona zadovoljava sva tri uslova. Dakle, $\widehat{U}^{\text{fb}}$ predstavlja forward bisimulaciju između automata $\mathcal{A}$ i $\mathcal{B}$.*

Neka je sada U proizvoljna forward bisimulacija između automata $\mathcal{A}$ i $\mathcal{B}$. Tada matrica U zadovoljava sistem koji čine uslovi (1-fb)*,* (2-fb) *i* (3-fb)*, pa samim tim zadovoljava i sistem koji čine uslovi* (2-fb) *i* (3-fb)*. Kako je, prema tvrđenju* (a)*, $\widehat{U}^{\text{fb}}$ najveće rešenje tog sistema, sledi da važi $U \leqslant \widehat{U}^{\text{fb}}$. Prema tome, svaka forward bisimulacija između automata*

$\mathcal{A}$ i $\mathcal{B}$ sadržana je u matrici $\widehat{U}^{\mathrm{fb}}$, odnosno $\widehat{U}^{\mathrm{fb}}$ je najveća forward bisimulacija između ovih automata.

(c) *Pretpostavimo najpre da je $\widehat{U}^{\mathrm{fb}}$ nulta matrica. Ako bi postojala forward bisimulacija U između automata $\mathcal{A}$ i $\mathcal{B}$, onda bi matrica U zadovoljavala sistem nejednačina određen uslovima* (2-fb) *i* (3-fb)*. Kako je, prema tvrđenju* (a)*, $\widehat{U}^{\mathrm{fb}}$ najveće rešenje tog sistema, mora važiti $U \leqslant \widehat{U}^{\mathrm{fb}}$. Pošto je $\widehat{U}^{\mathrm{fb}}$ nulta matrica, sledi da je i U nulta matrica, što je u suprotnosti sa definicijom forward bisimulacije, prema kojoj matrica simulacije mora biti nenulta. Dakle, u ovom slučaju između automata $\mathcal{A}$ i $\mathcal{B}$ ne postoji forward bisimulacija.*

Pretpostavimo sada da je $\widehat{U}^{\mathrm{fb}}$ nenulta matrica, ali da ne zadovoljava uslov (1-fb)*. Pretpostavimo suprotno, da postoji forward bisimulacija U između automata $\mathcal{A}$ i $\mathcal{B}$. Tada, prema tvrđenju* (a)*, ponovo važi $U \leqslant \widehat{U}^{\mathrm{fb}}$. Kako matrica U zadovoljava uslov* (1-fb)*, dobijamo $\sigma^A \leqslant \sigma^B \otimes U^\top \leqslant \sigma^B \otimes (\widehat{U}^{\mathrm{fb}})^\top$ i $\sigma^B \leqslant \sigma^A \otimes U \leqslant \sigma^A \otimes \widehat{U}^{\mathrm{fb}}$. Odavde sledi da i matrica $\widehat{U}^{\mathrm{fb}}$ zadovoljava uslov* (1-fb)*, što je u suprotnosti sa polaznom pretpostavkom. Dakle, ni u ovom slučaju ne postoji forward bisimulacija između automata $\mathcal{A}$ i $\mathcal{B}$.*

(d) *Pretpostavimo da za neko $s \in \mathbb{N}$ važi $U_s^{\mathrm{fb}} = U_{s+1}^{\mathrm{fb}}$, tada je:*

$$\begin{aligned} U_{s+2}^{\mathrm{fb}} &= U_{s+1}^{\mathrm{fb}} \wedge \bigwedge_{x\in X} \left(\left(M_x^B \otimes (U_{s+1}^{\mathrm{fb}})^\top \right) \not/ M_x^A \right)^\top \wedge \left((M_x^A \otimes U_{s+1}^{\mathrm{fb}}) \not/ M_x^B \right) \\ &= U_s^{\mathrm{fb}} \wedge \bigwedge_{x\in X} \left(\left(M_x^B \otimes (U_s^{\mathrm{fb}})^\top \right) \not/ M_x^A \right)^\top \wedge \left((M_x^A \otimes U_s^{\mathrm{fb}}) \not/ M_x^B \right) \\ &= U_{s+1}^{\mathrm{fb}} = U_s^{\mathrm{fb}}. \end{aligned}$$

Jednostavnom indukcijom zaključujemo da je $U_t^{\mathrm{fb}} = U_s^{\mathrm{fb}}$, za svako $t \geqslant s$.

Prema tome, infimum posmatranog niza jednak je njegovom stabilizovanom članu, odnosno $\widehat{U}^{\mathrm{fb}} = U_s^{\mathrm{fb}}$. □

Teorema 3.2 daje postupak za ispitivanje postojanja simulacija i bisimulacija proizvoljnog tipa između dva max-plus automata nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$, kao i za određivanje njihove najveće simulacije, odnosno bisimulacije, ukoliko ona postoji.

Postupak se zasniva na sukcesivnom izračunavanju članova nerastućeg niza matrica $\{U_s^\xi\}_{s\in\mathbb{N}}$ i određivanju njegovog infimuma $\widehat{U}^\xi$. U slučaju kada se niz stabilizuje, odnosno kada za neko $s \in \mathbb{N}$ važi $U_s^\xi = U_{s+1}^\xi$, na osnovu tvrđenja (d) postupak se može zaustaviti, pri čemu poslednja izračunata matrica predstavlja infimum posmatranog niza. Na taj način najveće rešenje sistema može se odrediti u konačnom broju koraka.

Međutim, postupak ne mora uvek da se završi u konačnom broju koraka, budući da niz $\{U_s^\xi\}_{s\in\mathbb{N}}$ može biti beskonačan. U pojedinim situacijama infimum posmatranog niza može se odrediti eksplicitno, što će biti ilustrovano u narednom primeru.

Kada se opisani postupak ne završi u konačnom broju koraka, može se primeniti metod predložen u radu [8]. Ovaj metod se uvek završava nakon konačno mnogo koraka,

ali omogućava ispitivanje postojanja isključivo Bulovih simulacija i bisimulacija. Sa druge strane, Teorema 3.3 pruža mogućnost određivanja parcijalne inkluzije, odnosno parcijalne ekvivalencije ponašanja automata za sve reči čija je dužina manja od unapred zadate vrednosti.

Nakon određivanja matrice $\widehat{U}^{\xi}$ proverava se da li ona zadovoljava uslov (1-ξ). Ukoliko je taj uslov ispunjen, tada između posmatranih automata postoji simulacija, odnosno bisimulacija tipa ξ, pri čemu je matrica $\widehat{U}^{\xi}$ njihova najveća simulacija, odnosno bisimulacija. U suprotnom, prema tvrđenju (c), simulacija ili bisimulacija tog tipa ne postoji.

Primer 3.1 *Neka je* $X = \{x, y\}$ *i neka su dati max-plus automati* $\mathcal{A} = (2, \sigma^A, \{M_x^A, M_y^A\}, \tau^A)$ *i* $\mathcal{B} = (3, \sigma^B, \{M_x^B, M_y^B\}, \tau^B)$ *nad kompletnim max-plus poluprstenom* $\mathbb{R}_{\infty}$*, određeni sa*

$$M_x^A = \begin{bmatrix} 4 & 1 \\ -\infty & 2 \end{bmatrix}, \qquad M_y^A = \begin{bmatrix} -\infty & -\infty \\ -\infty & 1 \end{bmatrix}, \qquad \sigma^A = \begin{bmatrix} 1 & -\infty \end{bmatrix}, \qquad \tau^A = \begin{bmatrix} -\infty \\ 0 \end{bmatrix},$$

$$M_x^B = \begin{bmatrix} -\infty & 4 & -\infty \\ -\infty & 4 & -\infty \\ -\infty & -\infty & 2 \end{bmatrix}, \quad M_y^B = \begin{bmatrix} -\infty & -\infty & 0 \\ -\infty & -\infty & -2 \\ -\infty & -\infty & 1 \end{bmatrix}, \quad \sigma^B = \begin{bmatrix} 1 & -\infty & -\infty \end{bmatrix}, \quad \tau^B = \begin{bmatrix} -1 \\ -3 \\ 0 \end{bmatrix}.$$

Primenom postupka opisanog u Teoremi 3.2 dobijamo sledeće rezultate.

(a) *Postoji najveća forward simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$*.*
Indukcijom po s dobija se da važi

$$U_1^{\mathrm{fs}} = \begin{bmatrix} +\infty & +\infty & +\infty \\ -1 & -3 & 0 \end{bmatrix}, \qquad U_s^{\mathrm{fs}} = \begin{bmatrix} 0 & 0 & 5-2s \\ -1 & -3 & 0 \end{bmatrix}, \quad s \geqslant 2.$$

Prema Teoremi 3.2 (a), infimum ovog niza je najveće rešenje sistema (2-fs) *i* (3-fs)*:*

$$\widehat{U}^{\mathrm{fs}} = \begin{bmatrix} 0 & 0 & -\infty \\ -1 & -3 & 0 \end{bmatrix},$$

Kako matrica $\widehat{U}^{\mathrm{fs}}$ *zadovoljava uslov* (1-fs)

$$\sigma^A = \begin{bmatrix} 1 & -\infty \end{bmatrix} \leqslant \begin{bmatrix} 1 & 0 \end{bmatrix} = \sigma^B \otimes (\widehat{U}^{\mathrm{fs}})^{\top},$$

sledi da $\widehat{U}^{\mathrm{fs}}$ *predstavlja najveću forward simulaciju između automata* $\mathcal{A}$ *i* $\mathcal{B}$*.*

(b) *Ne postoji backward simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$*.*
Računanjem članova niza $\{U_s^{\mathrm{bs}}\}_{s\in\mathbb{N}}$ *dobija se*

$$U_1^{\mathrm{bs}} = \begin{bmatrix} 0 & -\infty & -\infty \\ +\infty & +\infty & +\infty \end{bmatrix}, \qquad U_s^{\mathrm{bs}} = \begin{bmatrix} -\infty & -\infty & -\infty \\ -\infty & -\infty & -\infty \end{bmatrix}, \quad s \geqslant 2.$$

Dakle, $\widehat{U}^{\text{bs}}$ je nulta matrica:

$$\widehat{U}^{\text{bs}} = \begin{bmatrix} -\infty & -\infty & -\infty \\ -\infty & -\infty & -\infty \end{bmatrix},$$

Na osnovu Teoreme 3.2 (c) zaključujemo da između automata $\mathcal{A}$ i $\mathcal{B}$ ne postoji backward simulacija.

(c) *Ne postoji forward bisimulacija između automata $\mathcal{A}$ i $\mathcal{B}$.*
Primenom postupka iz Teoreme 3.2 dobija se

$$U_1^{\text{fb}} = \begin{bmatrix} -\infty & -\infty & -\infty \\ -1 & -3 & 0 \end{bmatrix}, \qquad U_s^{\text{fb}} = \begin{bmatrix} -\infty & -\infty & -\infty \\ -2s-1 & -2s-1 & 0 \end{bmatrix}, \quad s \geqslant 2,$$

odakle sledi da je

$$\widehat{U}^{\text{fb}} = \begin{bmatrix} -\infty & -\infty & -\infty \\ -\infty & -\infty & 0 \end{bmatrix}.$$

Međutim, matrica $\widehat{U}^{\text{fb}}$ ne zadovoljava uslov (1-fb)*:*

$$\sigma^B \otimes (\widehat{U}^{\text{fb}})^\top = \begin{bmatrix} 1 & -\infty & -\infty \end{bmatrix} \otimes \begin{bmatrix} -\infty & -\infty \\ -\infty & -\infty \\ -\infty & 0 \end{bmatrix} = \begin{bmatrix} -\infty & -\infty \end{bmatrix},$$

stoga, prema Teoremi 3.2 (c), zaključujemo da između automata $\mathcal{A}$ i $\mathcal{B}$ ne postoji forward bisimulacija.

Međutim, ako se početni težinski vektori promene tako da

$$\sigma^A = \begin{bmatrix} -\infty & 1 \end{bmatrix}, \qquad \sigma^B = \begin{bmatrix} -\infty & -\infty & 1 \end{bmatrix},$$

onda ista matrica $\widehat{U}^{\text{fb}}$ zadovoljava i uslov (1-fb)*, te predstavlja najveću forward bisimulaciju između automata $\mathcal{A}$ i $\mathcal{B}$.*

Nakon određivanja najvećih simulacija i bisimulacija prirodno se postavlja pitanje da li se tokom same konstrukcije niza matrica $\{U_s^{\xi}\}_{s\in\mathbb{N}}$ mogu dobiti informacije o odnosu ponašanja posmatranih automata, i pre nego što se odredi njegovo granično rešenje $\widehat{U}^{\xi}$. Pokazaćemo da svaki član ovog niza koji zadovoljava prvi uslov odgovarajuće simulacije ili bisimulacije omogućava izvođenje zaključaka o inkluziji, odnosno ekvivalenciji ponašanja automata za sve reči čija je dužina manja od njegovog rednog broja. Drugim rečima, i kada postupak iz Teoreme 3.2 ne može da se završi u konačnom broju koraka, njegovi parcijalni rezultati pružaju informacije o ponašanju automata na rečima ograničene dužine. Naredna teorema formalizuje ovu činjenicu.

Teorema 3.3 *Neka su* $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ *i* $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ *max-plus automati nad kompletnim max-plus poluprstenom* $\mathbb{R}_\infty$.

Za proizvoljno $\xi \in \{\mathrm{fs}, \mathrm{bs}, \mathrm{fb}, \mathrm{bb}, \mathrm{fbb}, \mathrm{bfb}\}$ *neka je* $\{U_s^\xi\}_{s\in\mathbb{N}}$ *niz matrica definisan kao u Teoremi 3.2. Tada važe sledeća tvrđenja:*

(a) *Za svako* $k \in \mathbb{N}$ *i* $\xi \in \{\mathrm{fs}, \mathrm{bs}\}$*, ako matrica* U_k^ξ *zadovoljava uslov* (1-ξ)*, tada važi* $[\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u)$*, za svaku reč* $u \in X^*$ *takvu da je* $|u| < k$.

(b) *Za svako* $k \in \mathbb{N}$ *i* $\xi \in \{\mathrm{fb}, \mathrm{bb}, \mathrm{fbb}, \mathrm{bfb}\}$*, ako matrica* U_k^ξ *zadovoljava uslov* (1-ξ)*, tada važi* $[\![\mathcal{A}]\!](u) = [\![\mathcal{B}]\!](u)$*, za svaku reč* $u \in X^*$ *takvu da je* $|u| < k$.

Dokaz 3.3 *Dokazaćemo tvrđenje* (a) *u slučaju* $\xi = \mathrm{fs}$*. Preostali slučajevi dokazuju se na analogan način.*

Najpre ćemo pokazati da za svako $k \geqslant 2$ *i svaku reč* $u \in X^+$ *dužine* $|u| = l < k$ *važi*

$$(U_k^{\mathrm{fs}})^\top \otimes M_u^A \leqslant M_u^B \otimes (U_{k-l}^{\mathrm{fs}})^\top. \tag{53}$$

Ova nejednakost predstavlja proširenje uslova (2-fs) *sa pojedinačnih simbola na proizvoljne reči ograničene dužine.*

Za reči dužine jedan tvrđenje neposredno sledi iz definicije matrice U_k^{fs} *i osobine 40:*

$$\begin{aligned} U_k^{\mathrm{fs}} &\leqslant \left(\left(M_x^B \otimes (U_{k-1}^{\mathrm{fs}})^\top\right) \not/ M_x^A\right)^\top \\ &\iff (U_k^{\mathrm{fs}})^\top \leqslant \left(M_x^B \otimes (U_{k-1}^{\mathrm{fs}})^\top\right) \not/ M_x^A \\ &\iff (U_k^{\mathrm{fs}})^\top \otimes M_x^A \leqslant M_x^B \otimes (U_{k-1}^{\mathrm{fs}})^\top, \quad \text{za svako } x \in X. \end{aligned}$$

Pretpostavimo sada da nejednakost (53) *važi za sve reči dužine* $l < k-1$*, i neka je* $v = ux$*, gde je* $|u| = l$ *i* $x \in X$*. Korišćenjem* (53) *i 39, dobijamo:*

$$\begin{aligned} (U_k^{\mathrm{fs}})^\top \otimes M_v^A &= (U_k^{\mathrm{fs}})^\top \otimes M_u^A \otimes M_x^A \leqslant M_u^B \otimes (U_{k-l}^{\mathrm{fs}})^\top \otimes M_x^A \\ &\leqslant M_u^B \otimes M_x^B \otimes (U_{k-l-1}^{\mathrm{fs}})^\top = M_v^B \otimes (U_{k-(l+1)}^{\mathrm{fs}})^\top. \end{aligned}$$

Dalje, iz definicije matrice U_1^{fs} *sledi da je*

$$(U_1^{\mathrm{fs}})^\top = (\tau^A \backslash \tau^B)^\top = \tau^B \not/ \tau^A.$$

Kako za svako $l < k$ *važi* $U_{k-l}^{\mathrm{fs}} \leqslant U_1^{\mathrm{fs}}$*, dobijamo*

$$(U_{k-l}^{\mathrm{fs}})^\top \otimes \tau^A \leqslant (U_1^{\mathrm{fs}})^\top \otimes \tau^A \leqslant \tau^B.$$

Pretpostavimo sada da za neko $k \in \mathbb{N}$ *matrica* U_k^{fs} *zadovoljava uslov* (1-fs)*, odnosno*

$$\sigma^A \leqslant \sigma^B \otimes (U_k^{\mathrm{fs}})^\top.$$

Ako je $k = 1$*, tada neposredno sledi*

$$[\![\mathcal{A}]\!](\varepsilon) = \sigma^A \otimes \tau^A \leqslant \sigma^B \otimes (U_1^{\mathrm{fs}})^\top \otimes \tau^A \leqslant \sigma^B \otimes \tau^B = [\![\mathcal{B}]\!](\varepsilon).$$

Neka je sada $k \geqslant 2$ i $u \in X^*$ takva da je $|u| = l < k$. Tada, koristeći (53)*, dobijamo*

$$\begin{aligned}[\![\mathcal{A}]\!](u) &= \sigma^A \otimes M_u^A \otimes \tau^A \leqslant \sigma^B \otimes (U_k^{\mathrm{fs}})^\top \otimes M_u^A \otimes \tau^A \\ &\leqslant \sigma^B \otimes M_u^B \otimes (U_{k-l}^{\mathrm{fs}})^\top \otimes \tau^A \leqslant \sigma^B \otimes M_u^B \otimes \tau^B = [\![\mathcal{B}]\!](u).\end{aligned}$$

Time je dokazano tvrđenje (a) *u slučaju forward simulacija. Dokazi preostalih slučajeva zasnivaju se na istim idejama, čime je dokaz teoreme završen.* □

Primer 3.2 *Posmatrajmo automate iz Primera 3.1. Kao što je tamo pokazano, odgovarajući niz matrica $\{U_s^{\mathrm{fs}}\}_{s\in\mathbb{N}}$ se ne stabilizuje nakon konačno mnogo koraka, ali svaki njegov član pruža informacije o odnosu između jezika automata $\mathcal{A}$ i $\mathcal{B}$.*

Primenom Teoreme 3.3 može se odrediti za koje dužine reči već nakon konačno mnogo koraka sledi inkluzija ponašanja automata.

Kako U_s^{fs} zadovoljava uslov (1-fs)*, prema Teoremi 3.3, za svako $k \in \mathbb{N}$ važi*

$$[\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u), \qquad \text{za svaku reč } u \in X^* \text{ takvu da je } |u| < k.$$

Dakle, iako se najveća forward simulacija dobija tek kao infimum beskonačnog niza matrica, već nakon izračunavanja k-tog člana niza može se potvrditi inkluzija ponašanja automata za sve reči dužine manje od k.

Razmotrimo sada iste automate, ali neka početni težinski vektori budu

$$\sigma^A = \begin{bmatrix} -100 & 0 \end{bmatrix}, \qquad \sigma^B = \begin{bmatrix} -\infty & -\infty & 0 \end{bmatrix}.$$

Tada važi

$$\sigma^B \otimes (U_1^{\mathrm{fs}})^\top = \begin{bmatrix} -\infty & -\infty & 0 \end{bmatrix} \otimes \begin{bmatrix} +\infty & -1 \\ +\infty & -3 \\ +\infty & 0 \end{bmatrix} = \begin{bmatrix} +\infty & 0 \end{bmatrix} \geqslant \begin{bmatrix} -100 & 0 \end{bmatrix} = \sigma^A.$$

Za svako $k \in \mathbb{N}$, $k \geqslant 2$ biće:

$$\sigma^B \otimes (U_k^{\mathrm{fs}})^\top = \begin{bmatrix} -\infty & -\infty & 0 \end{bmatrix} \otimes \begin{bmatrix} 0 & -1 \\ 0 & -3 \\ 5-2k & 0 \end{bmatrix} = \begin{bmatrix} 5-2k & 0 \end{bmatrix},$$

pa je uslov (1-fs) *ispunjen ako i samo ako je $k \leqslant 52$, jer je $\begin{bmatrix} 5-2k & 0 \end{bmatrix} \geqslant \begin{bmatrix} -100 & 0 \end{bmatrix}$.*

Na osnovu Teoreme 3.3 zaključujemo da za sve reči $u \in X^*$ dužine manje od 52 važi $[\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u)$, iako inkluzija $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!]$ ne važi u opštem slučaju.

Ovaj primer pokazuje da članovi niza $\{U_s^{\mathrm{fs}}\}_{s\in\mathbb{N}}$ mogu pružiti značajne informacije o ponašanju automata i pre nego što se odredi njihov infimum ili ustanovi postojanje najveće simulacije.

U prethodnoj teoremi pokazano je da se postojanje simulacije ili bisimulacije odgovarajućeg tipa može utvrditi proverom da li infimum niza matrica zadovoljava uslov (1-ξ). Međutim, kada se niz $\{U_s^\xi\}_{s\in\mathbb{N}}$ ne stabilizuje nakon konačno mnogo koraka, određivanje njegovog infimuma može biti otežano. Zbog toga je važno ispitati da li se zaključci o ispunjenosti uslova (1-ξ) mogu donositi već na osnovu pojedinačnih članova niza.

Naredna teorema pokazuje da je odgovor potvrdan. Naime, uslov (1-ξ) zadovoljen je za infimum niza ako i samo ako je zadovoljen za svaki njegov član. Pored toga, ukoliko neki član niza zadovoljava ovaj uslov, tada ga zadovoljavaju i svi prethodni članovi niza.

Teorema 3.4 *Neka su* $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ *i* $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ *max-plus automati nad kompletnim max-plus poluprstenom* $\mathbb{R}_\infty$.

Za proizvoljno $\xi \in \{\text{fs}, \text{bs}, \text{fb}, \text{bb}, \text{fbb}, \text{bfb}\}$ *neka je* $\{U_s^\xi\}_{s\in\mathbb{N}}$ *niz matrica definisan kao u Teoremi 3.2, a* $\widehat{U}^\xi$ *njegov infimum. Tada važe sledeća tvrđenja:*

(a) *Matrica* $\widehat{U}^\xi$ *zadovoljava uslov* (1-ξ) *ako i samo ako svaka matrica* U_s^ξ, $s \in \mathbb{N}$, *zadovoljava isti uslov.*

(b) *Ako za neko* $s \in \mathbb{N}$ *matrica* U_s^ξ *zadovoljava uslov* (1-ξ)*, onda taj uslov zadovoljava i svaka matrica* U_t^ξ *za* $t < s$.

Dokaz 3.4 *Dokazaćemo tvrđenje samo za slučaj* $\xi = \text{fs}$. *Preostali slučajevi dokazuju se na analogan način.*

(a) *Pretpostavimo najpre da matrica* $\widehat{U}^{\text{fs}}$ *zadovoljava uslov* (1-fs).

Kako je $\widehat{U}^{\text{fs}} \leqslant U_s^{\text{fs}}$, *za svako* $s \in \mathbb{N}$, *dobijamo* $\sigma^A \leqslant \sigma^B \otimes (\widehat{U}^{\text{fs}})^\top \leqslant \sigma^B \otimes (U_s^{\text{fs}})^\top$, *odnosno svaka matrica* U_s^{fs} *zadovoljava uslov* (1-fs).

Pretpostavimo sada obrnuto, da svaka matrica U_s^{fs} *zadovoljava uslov* (1-fs). *Tada za svako* $s \in \mathbb{N}$ *važi* $\sigma^A \leqslant \sigma^B \otimes (U_s^{\text{fs}})^\top$.

Prema Lemi 4.1 iz rada [69], množenje matrice sa leve strane čuva infimume i kako transponovanje komutira se infimumom 12, dobijamo

$$\sigma^A \leqslant \bigwedge_{s\in\mathbb{N}} \left(\sigma^B \otimes (U_s^{\text{fs}})^\top\right) = \sigma^B \otimes \left(\bigwedge_{s\in\mathbb{N}} (U_s^{\text{fs}})^\top\right) = \sigma^B \otimes \left(\bigwedge_{s\in\mathbb{N}} U_s^{\text{fs}}\right)^\top = \sigma^B \otimes (\widehat{U}^{\text{fs}})^\top.$$

Dakle, $\widehat{U}^{\text{fs}}$ *zadovoljava uslov* (1-fs)*, čime je dokazano tvrđenje* (a).

(b) *Pretpostavimo da za neko* $s \in \mathbb{N}$ *matrica* U_s^{fs} *zadovoljava uslov* (1-fs).

Kako je niz $\{U_s^{\text{fs}}\}_{s\in\mathbb{N}}$ *nerastući, za svako* $t < s$ *važi* $U_s^{\text{fs}} \leqslant U_t^{\text{fs}}$.

Otuda dobijamo $\sigma^A \leqslant \sigma^B \otimes (U_s^{\text{fs}})^\top \leqslant \sigma^B \otimes (U_t^{\text{fs}})^\top$, *što znači da i matrica* U_t^{fs} *zadovoljava uslov* (1-fs). □

Na osnovu Teoreme 3.4 postupak za ispitivanje postojanja simulacija i bisimulacija, opisan nakon Teoreme 3.2, može se dodatno unaprediti. Naime, tokom sukcesivnog izračunavanja članova niza matrica $\{U_s^\xi\}_{s\in\mathbb{N}}$ nije neophodno odrediti njegov infimum. Ukoliko

se za neki član U_s^ξ ustanovi da ne zadovoljava uslov (1-ξ), tada, prema Teoremi 3.4, isti uslov ne zadovoljava nijedan naredni član niza, kao ni njegov infimum $\widehat{U}^\xi$. Stoga se postupak može odmah prekinuti i zaključiti da između posmatranih automata ne postoji simulacija, odnosno bisimulacija tipa ξ.

Na taj način, u slučaju nepostojanja simulacije ili bisimulacije, zaključak se može doneti bez određivanja infimuma niza. Sa druge strane, ukoliko svi do tada izračunati članovi zadovoljavaju uslov (1-ξ), postupak se ne mora završiti nakon konačno mnogo koraka, jer niz $\{U_s^\xi\}_{s\in\mathbb{N}}$ u opštem slučaju može biti beskonačan.

Teoreme 3.3 i 3.4 ukazuju i na mogućnost proučavanja parcijalne inkluzije i parcijalne ekvivalencije ponašanja automata. Umesto poređenja ponašanja automata za sve reči iz skupa X^*, može se posmatrati njihovo ponašanje samo na rečima čija je dužina manja od unapred zadate vrednosti k.

Za dati prirodan broj k, parcijalna inkluzija određena je uslovom

$$[\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u), \qquad \text{za svaku reč } u \in X^* \text{ takvu da je } |u| < k,$$

dok je parcijalna ekvivalencija određena uslovom

$$[\![\mathcal{A}]\!](u) = [\![\mathcal{B}]\!](u), \qquad \text{za svaku reč } u \in X^* \text{ takvu da je } |u| < k.$$

Prema Teoremama 3.3 i 3.4, dovoljno je odrediti najveći član niza $\{U_s^\xi\}_{s\in\mathbb{N}}$ koji zadovoljava uslov (1-ξ). Tada se može zaključiti da važi parcijalna inkluzija, odnosno parcijalna ekvivalencija, ponašanja automata za sve reči čija je dužina manja od odgovarajućeg broja k, bez potrebe za neposrednim poređenjem vrednosti koje automati pridružuju svim takvim rečima.

3.4 Uticaj izbora max-plus strukture na određivanje simulacija i bisimulacija

Rezultati dobijeni u prethodnim sekcijama izvedeni su za max-plus automate nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$. Međutim, u mnogim primenama težine automata ne pripadaju skupu svih realnih brojeva, već nekoj njegovoj podstrukturi, kao što su skup celih ili racionalnih brojeva. Zbog toga se prirodno postavlja pitanje u kojoj meri se prethodno razvijeni postupci mogu primeniti i na takvim strukturama, kao i da li izbor osnovne max-plus strukture utiče na tok i rezultat algoritama za određivanje simulacija i bisimulacija.

Kada su težine automata celobrojne, prirodno se razmatra *max-plus poluprsten nad celim brojevima* $\mathbb{Z}_{\max} = (\mathbb{Z}_{\max}, \oplus, \otimes, -\infty, 0)$, gde je $\mathbb{Z}_{\max} = \mathbb{Z} \cup \{-\infty\}$. Kako je $\mathbb{Z}_{\max}$ podpoluprsten strukture $\mathbb{R}_{\max}$, svaki automat nad $\mathbb{Z}_{\max}$ može se posmatrati i kao automat nad $\mathbb{R}_{\max}$, pri čemu se njegovo ponašanje ne menja.

Slično, *kompletan max-plus poluprsten* $\mathbb{Z}_\infty = (\mathbb{Z}_\infty, \oplus, \otimes, -\infty, 0)$, $\mathbb{Z}_\infty = \mathbb{Z} \cup \{-\infty, +\infty\}$, predstavlja kompletan dioid sa istim operacijama sabiranja, množenja i reziduacije kao i $\mathbb{R}_\infty$. Zbog toga se algoritmi za određivanje simulacija i bisimulacija mogu primeniti na isti način kao i u slučaju strukture $\mathbb{R}_\infty$, pri čemu se dobijaju isti rezultati. Potpuno analogni zaključci važe i kada se skup celih brojeva zameni skupom racionalnih brojeva.

Za razliku od prethodno navedenih struktura, situacija se menja ukoliko se posmatra skup nenegativnih realnih brojeva $\mathbb{R}^+ = \mathbb{P} = \{r \in \mathbb{R} \mid r \geq 0\}$. U tom slučaju definišu se strukture $\mathbb{P}_{\max}$ i $\mathbb{P}_\infty$, koje imaju iste operacije sabiranja i množenja kao odgovarajuće max-plus strukture nad skupom svih realnih brojeva. Međutim, operacija reziduacije u $\mathbb{P}_\infty$ nije određena na isti način kao u $\mathbb{R}_\infty$, što je prikazano sledećom Cayley-evom tablicom:

	$b \in \mathbb{R}^+$			
$\backslash$	$b \geqslant a$	$b < a$	$-\infty$	$+\infty$
$a \in \mathbb{R}^+$	$b - a$	$-\infty$	$-\infty$	$+\infty$
$-\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$
$+\infty$	$-\infty$	$-\infty$	$-\infty$	$+\infty$

Može se primetiti da se reziduacija u strukturama $\mathbb{R}_\infty$ i $\mathbb{P}_\infty$ razlikuje upravo u slučajevima kada bi rezultat oduzimanja bio negativan. U strukturi $\mathbb{P}_\infty$ takve vrednosti zamenjuju se elementom $-\infty$. Ova naizgled mala razlika utiče na konstrukciju nizova matrica $\{U_s^\xi\}_{s\in\mathbb{N}}$, a samim tim i na tok algoritma za određivanje simulacija i bisimulacija, što će biti analizirano u nastavku. Tabelarni prikaz operacija nad karakterističnim max-plus strukturama dat je u Tabeli 3.

Primetimo da za proizvoljne elemente $a, b \in \mathbb{P}_\infty$ važi

$$a \backslash b = \bigvee \{x \in \mathbb{P}_\infty \mid a \otimes x \leqslant b\} = \top \{x \in \mathbb{P}_\infty \mid a \otimes x \leqslant b\}, \tag{54}$$

gde oznaka $\top X$ predstavlja najveći element skupa X.

Teorema 3.2 zasniva se na opštim osobinama kompletnih dioida. Zbog toga se isti postupak za određivanje najvećih simulacija i bisimulacija može primeniti ne samo nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$, već i nad kompletnim max-plus dioidom $\mathbb{P}_\infty$.

Drugim rečima, algoritam ostaje formalno isti, dok se jedina promena odnosi na operaciju reziduacije. Sve ostale operacije zadržavaju isti oblik kao u strukturi $\mathbb{R}_\infty$. Iako se ova izmena na prvi pogled može činiti neznatnom, ona može značajno uticati na tok izvršavanja algoritma. Naime, svaki max-plus automat nad $\mathbb{P}_\infty$ može se posmatrati i kao automat nad $\mathbb{R}_\infty$, pa se postojanje simulacija i bisimulacija može ispitivati primenom oba postupka. Međutim, zbog različito definisane reziduacije, nizovi matrica $\{U_s^\xi\}_{s\in\mathbb{N}}$ dobijeni na ove dve strukture u opštem slučaju nisu isti.

Primetimo da se pri računanju nad $\mathbb{P}_\infty$ svaka negativna vrednost koja bi se u $\mathbb{R}_\infty$ pojavila kao rezultat reziduacije odmah zamenjuje elementom $-\infty$. Jednom dobijena vrednost $-\infty$

više ne može postati konačna vrednost u narednim iteracijama, što predstavlja bitnu razliku u odnosu na postupak nad $\mathbb{R}_\infty$. Zbog toga se niz matrica nad $\mathbb{P}_\infty$ često stabilizuje brže nego odgovarajući niz nad $\mathbb{R}_\infty$, a u pojedinim slučajevima stabilizacija nastupa i kada postupak nad $\mathbb{R}_\infty$ proizvodi beskonačan niz matrica.

Sa druge strane, postupak nad $\mathbb{R}_\infty$ ima važnu prednost. Pošto dozvoljava i negativne vrednosti u matricama relacije, rešenja se traže u široj klasi matrica nego u slučaju strukture $\mathbb{P}_\infty$. Zbog toga može postojati simulacija ili bisimulacija nad $\mathbb{R}_\infty$, iako odgovarajuća simulacija ili bisimulacija nad $\mathbb{P}_\infty$ ne postoji.

Obe navedene pojave, brža stabilizacija niza i mogućnost postojanja simulacija u široj klasi matrica, biće ilustrovane u narednom primeru.

Primer 3.3 *Neka je dat ulazni alfabet* $X = \{x, y\}$ *i neka su* $\mathcal{A} = (2, \sigma^A, \{M_x^A, M_y^A\}, \tau^A)$ *i* $\mathcal{B} = (3, \sigma^B, \{M_x^B, M_y^B\}, \tau^B)$ *max-plus automati zadati sa*

$$M_x^A = \begin{bmatrix} 4 & 1 \\ -\infty & 2 \end{bmatrix}, \qquad M_y^A = \begin{bmatrix} -\infty & -\infty \\ -\infty & 1 \end{bmatrix}, \qquad \sigma^A = \begin{bmatrix} 5 & -\infty \end{bmatrix}, \qquad \tau^A = \begin{bmatrix} -\infty \\ 0 \end{bmatrix},$$

$$M_x^B = \begin{bmatrix} -\infty & 4 & -\infty \\ -\infty & 4 & -\infty \\ -\infty & -\infty & 2 \end{bmatrix}, \quad M_y^B = \begin{bmatrix} -\infty & -\infty & 0 \\ -\infty & -\infty & 2 \\ -\infty & -\infty & 1 \end{bmatrix}, \quad \sigma^B = \begin{bmatrix} 1 & -\infty & -\infty \end{bmatrix}, \quad \tau^B = \begin{bmatrix} 1 \\ 3 \\ 0 \end{bmatrix}.$$

Ove automate možemo posmatrati kao max-plus automate nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$*, ali i kao max-plus automate nad kompletnim max-plus dioidom* $\mathbb{P}_\infty$*. U nastavku ćemo razmotriti oba slučaja.*

a) *Slučaj kada se automati* $\mathcal{A}$ *i* $\mathcal{B}$ *posmatraju nad* $\mathbb{R}_\infty$*.*
Primenom postupka za izračunavanje članova sekvence $\{U_s^{\mathrm{fs}}\}_{s\in\mathbb{N}}$ *iz Teoreme 3.2, indukcijom po s dobijamo*

$$U_1^{\mathrm{fs}} = \begin{bmatrix} +\infty & +\infty & +\infty \\ 1 & 3 & 0 \end{bmatrix}, \quad U_2^{\mathrm{fs}} = \begin{bmatrix} 6 & 6 & 1 \\ -1 & 1 & 0 \end{bmatrix}, \quad U_s^{\mathrm{fs}} = \begin{bmatrix} 4 & 4 & 5-2s \\ -1 & 1 & 0 \end{bmatrix}, \quad s \geqslant 3,$$

a infimum ove sekvence je

$$\widehat{U}^{\mathrm{fs}} = \begin{bmatrix} 4 & 4 & -\infty \\ -1 & 1 & 0 \end{bmatrix}.$$

Dalje, proverom uslova (1-fs) uviđamo da je $\widehat{U}^{\mathrm{fs}}$ *najveća forward simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *nad* $\mathbb{R}_\infty$*:*

$$\sigma^A = \begin{bmatrix} 5 & -\infty \end{bmatrix} \leqslant \begin{bmatrix} 5 & 0 \end{bmatrix} = \begin{bmatrix} 1 & -\infty & -\infty \end{bmatrix} \otimes \begin{bmatrix} 4 & -1 \\ 4 & 1 \\ -\infty & 0 \end{bmatrix} = \sigma^B \otimes (\widehat{U}^{\mathrm{fs}})^\top.$$

b) *Slučaj kada se automati $\mathcal{A}$ i $\mathcal{B}$ posmatraju nad $\mathbb{P}_\infty$.*

Primenom odgovarajućeg iterativnog postupka dobijamo

$$U_1^{\mathrm{fs}} = \begin{bmatrix} +\infty & +\infty & +\infty \\ 1 & 3 & 0 \end{bmatrix}, \quad U_2^{\mathrm{fs}} = \begin{bmatrix} 6 & 6 & 1 \\ -\infty & 1 & 0 \end{bmatrix}, \quad U_3^{\mathrm{fs}} = U_4^{\mathrm{fs}} = \widehat{U}^{\mathrm{fs}} = \begin{bmatrix} 4 & 4 & -\infty \\ -\infty & 1 & 0 \end{bmatrix}.$$

Kako je i ovde ispunjen uslov (1-fs), $\widehat{U}^{\mathrm{fs}}$ je najveća forward simulacija između automata $\mathcal{A}$ i $\mathcal{B}$:

$$\sigma^A = \begin{bmatrix} 5 & -\infty \end{bmatrix} \leqslant \begin{bmatrix} 5 & -\infty \end{bmatrix} = \begin{bmatrix} 1 & -\infty & -\infty \end{bmatrix} \otimes \begin{bmatrix} 4 & -\infty \\ 4 & 1 \\ -\infty & 0 \end{bmatrix} = \sigma^B \otimes (\widehat{U}^{\mathrm{fs}})^\top.$$

Prednost postupka nad $\mathbb{P}_\infty$ ogleda se u tome što daje konačnu sekvencu matrica, za razliku od postupka nad $\mathbb{R}_\infty$ koji u ovom primeru generiše beskonačnu sekvencu. Sa druge strane, postupak nad $\mathbb{R}_\infty$ pretražuje širu klasu matrica, pa može utvrditi postojanje forward simulacije i u situacijama kada ona ne postoji nad $\mathbb{P}_\infty$. Naime, ako početni vektor automata $\mathcal{A}$ zamenimo sa $\sigma^A = \begin{bmatrix} 5 & 0 \end{bmatrix}$, tada forward simulacija između automata postoji nad $\mathbb{R}_\infty$, dok nad $\mathbb{P}_\infty$ ne postoji.

Rezultati predstavljeni u ovoj glavi pokazuju da simulacije i bisimulacije predstavljaju efikasan alat za ispitivanje inkluzije i ekvivalencije ponašanja max-plus automata. Pokazano je da se njihovo određivanje može svesti na rešavanje odgovarajućih sistema matričnih nejednačina, kao i da se najveće simulacije i bisimulacije mogu konstruisati primenom odgovarajućih iterativnih postupaka. Takođe je analiziran uticaj izbora max-plus strukture na tok i rezultate ovih postupaka.

U narednoj glavi biće uvedene slabe simulacije i bisimulacije, koje predstavljaju prirodno proširenje razmatranih pojmova. Za razliku od simulacija i bisimulacija uvedenih u ovoj glavi, one omogućavaju poređenje ponašanja automata uz veću fleksibilnost, zanemarujući određene prelaze koji ne utiču na posmatrano ponašanje automata.

Glava 4

Slabe simulacije i bisimulacije za max-plus automate

U prethodnoj glavi uvedeni su različiti tipovi simulacija i bisimulacija za max-plus automate i pokazano je da njihovo postojanje predstavlja dovoljan uslov za potvrdu inkluzije, odnosno ekvivalencije ponašanja automata. Takođe su razvijeni postupci za određivanje najvećih simulacija i bisimulacija zasnovani na rešavanju odgovarajućih sistema matričnih nejednačina.

Međutim, može se dogoditi da između dva automata ne postoji simulacija ili bisimulacija odgovarajućeg tipa, iako su njihova ponašanja inkluzivna ili ekvivalentna. Razlog za to leži u činjenici da standardne simulacije i bisimulacije nameću stroge uslove koji se proveravaju na nivou pojedinačnih prelaza automata. Zbog toga se prirodno postavlja pitanje da li je moguće oslabiti ove uslove, a da se pritom i dalje očuva mogućnost zaključivanja o inkluziji ili ekvivalenciji ponašanja automata.

Jedan od odgovora na ovo pitanje predstavljaju *slabe simulacije* i *slabe bisimulacije*. Osnovna ideja ovih relacija jeste da se uslovi koji se kod standardnih simulacija i bisimulacija proveravaju na nivou pojedinačnih prelaza zamene uslovima koji se odnose na ponašanje automata na nivou ulaznih reči. Na taj način slabe simulacije i bisimulacije proširuju mogućnosti poređenja max-plus automata, iako njihovo određivanje u opštem slučaju može zahtevati razmatranje beskonačno mnogo parova vektora.

Slabe simulacije i bisimulacije prvi put su uvedene u teoriji fazi automata [16], gde su se, pored potvrde inkluzije i ekvivalencije, pokazale kao uspešan alat za redukciju broja stanja i determinizaciju automata [19, 46, 72, 73, 74, 75, 76]. U ovoj glavi navedeni pojmovi biće prošireni na max-plus automate.

Kao i kod standardnih simulacija, razmatraju se dva tipa slabih simulacija, *forward* i *backward*, kao i dva odgovarajuća tipa slabih bisimulacija. Pored njih uvode se i relacije nazvane *slabe presimulacije* i *slabe prebisimulacije*. Pojmovi presimulacije i prebisimulacije prvi put su uvedeni u [77, 78], dok se njihove slabe varijante u ovoj glavi uvode za max-plus automate. Biće pokazano da svaka standardna presimulacija predstavlja slabu presimulaciju odgovarajućeg tipa, kao i da analogni odnosi važe između standardnih i slabih simulacija, odnosno između standardnih i slabih prebisimulacija i bisimulacija [17].

Kao i standardne simulacije i bisimulacije, slabe simulacije i bisimulacije karakterišu se kao rešenja odgovarajućih sistema matričnih nejednačina. Pokazaće se da sistemi kojima su definisane slabe presimulacije i prebisimulacije uvek imaju najveće rešenje, koje predstavlja najveću slabu presimulaciju, odnosno najveću slabu prebisimulaciju odgovarajućeg tipa. Na osnovu tih rešenja razvijaju se postupci za ispitivanje postojanja slabih simulacija i bisimulacija. Time se dobija mogućnost poređenja šire klase automata, budući da postoje slučajevi u kojima standardna simulacija ili bisimulacija ne postoji, dok odgovarajuća slaba simulacija ili bisimulacija postoji. Sa druge strane, postojanje standardne simulacije ili bisimulacije uvek povlači postojanje odgovarajuće slabe simulacije ili bisimulacije.

U ovoj glavi najpre se uvode slabe presimulacije i prebisimulacije, a zatim se karakterišu njihova najveća rešenja i razvija postupak za ispitivanje postojanja slabih simulacija i bisimulacija. Posebna pažnja posvećena je problemu terminacije ovog postupka, budući da se najveća slaba rešenja u opštem slučaju određuju razmatranjem beskonačno mnogo reči iz skupa X^*. Takođe se razmatraju slučajevi u kojima se najveća rešenja mogu odrediti eksplicitno, dok su teorijski rezultati ilustrovani odgovarajućim primerima.

Rezultati predstavljeni u ovom poglavlju zasnovani su na koautorskom radu proisteklom iz istraživanja sprovedenog u okviru ove doktorske disertacije [17].

4.1 Presimulacije i prebisimulacije

U prethodnoj glavi uvedeni su pojmovi *forward* i *backward* simulacija, kao i odgovarajući pojmovi bisimulacija između max-plus automata. Pokazano je da postojanje simulacije predstavlja dovoljan uslov za inkluziju ponašanja automata, dok postojanje bisimulacije predstavlja dovoljan uslov za njihovu ekvivalenciju.

Pored ovih pojmova, značajnu ulogu imaju i *presimulacije* i *prebisimulacije*. One predstavljaju prirodno slabljenje odgovarajućih simulacija i bisimulacija i dobijaju se izostavljanjem uslova koji povezuje inicijalne težinske vektore automata. Njihov osnovni značaj ogleda se u tome što predstavljaju polaznu osnovu za određivanje najvećih simulacija i bisimulacija. Takođe, rezultati prikazani u ovoj glavi poslužiće kao osnova za kasnije uvođenje slabih simulacija i bisimulacija, kao i za konstrukciju njihovih najvećih rešenja.

U daljem tekstu koristićemo sledeću skraćenu terminologiju. Ako je $\xi \in \{\mathrm{fs}, \mathrm{bs}\}$, tada svako rešenje sistema $(2-\xi)$ i $(3-\xi)$ nazivamo ***presimulacija tipa*** ξ ili kraće ***ξ-presimulacija***.

Ako je $\xi \in \{\mathrm{fb}, \mathrm{bb}, \mathrm{fbb}, \mathrm{bfb}\}$, tada svako rešenje sistema $(2-\xi)$ i $(3-\xi)$ nazivamo ***prebisimulacija tipa*** ξ ili kraće ***ξ-prebisimulacija***. Dakle, presimulacije i prebisimulacije dobijaju se iz odgovarajućih simulacija i bisimulacija izostavljanjem uslova $(1-\xi)$ koji se odnosi na inicijalne težinske vektore automata.

Kao i simulacije i bisimulacije, presimulacije i prebisimulacije mogu se posmatrati kao rešenja odgovarajućih sistema matričnih nejednačina. Naročito je značajno to što sistemi kojima su definisane presimulacije i prebisimulacije uvek imaju najveće rešenje, koje predstavlja najveću presimulaciju, odnosno najveću prebisimulaciju odgovarajućeg tipa. Ova najveća rešenja mogu se zatim koristiti za ispitivanje postojanja simulacija i bisimulacija odgovarajućeg tipa.

Najveća forward presimulacija može se odrediti iterativnim postupkom zasnovanim na opadajućem nizu matrica $\{U_s\}_{s\in\mathbb{N}}$ definisanom sa

$$U_1 = \tau^A \backslash \tau^B = (\tau^B \not/ \tau^A)^\top, \qquad U_{s+1} = U_s \wedge \left(\bigwedge_{x\in X} \left((M_x^B \otimes U_s^\top) \not/ M_x^A \right)^\top \right). \tag{55}$$

Infimum ovog niza predstavlja najveću forward presimulaciju između automata $\mathcal{A}$ i $\mathcal{B}$. Ukoliko dobijena matrica zadovoljava i uslov (1-fs), ona predstavlja najveću forward simulaciju. U suprotnom, između posmatranih automata ne postoji forward simulacija.

Analogni postupci važe i za ostale tipove presimulacija i prebisimulacija.

U narednoj sekciji biće uvedene slabe simulacije i bisimulacije. Pokazaće se da se njihova konstrukcija prirodno nadovezuje na pojmove presimulacija i prebisimulacija uvedene u ovoj sekciji.

4.2 Slabe simulacije i bisimulacije

U ovom odeljku uvode se slabe simulacije i slabe bisimulacije za max-plus automate. Ovi pojmovi predstavljaju uopštenje odgovarajućih simulacija i bisimulacija razmatranih u prethodnoj glavi, pri čemu zadržavaju njihovu osnovnu ulogu u potvrđivanju relacija inkluzije i ekvivalencije između automata. Kao i kod običnih simulacija i bisimulacija, uvode se odgovarajući pojmovi slabih presimulacija i slabih prebisimulacija, koji omogućavaju određivanje najvećih slabih simulacija i bisimulacija.

Da bi se definisale slabe simulacije i bisimulacije, najpre je potrebno uvesti vektore koji opisuju ponašanje automata nakon izvršavanja proizvoljne reči.

Neka je $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ max-plus automat nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$ i alfabetom X. Za svaku reč $u \in X^*$ definišu se vektori $\sigma_u^A \in \mathbb{R}_\infty^{1\times n}$ i $\tau_u^A \in \mathbb{R}_\infty^{n\times 1}$ sa

$$\sigma_u^A = \sigma^A \otimes M_u^A \quad \text{i} \quad \tau_u^A = M_u^A \otimes \tau^A. \tag{56}$$

Ovi vektori mogu se ekvivalentno definisati induktivno. Za sve $u \in X^*$ i $x \in X$ važi

$$\sigma_\varepsilon^A = \sigma^A,\ \sigma_{ux}^A = \sigma_u^A \otimes M_x^A \quad \text{i} \quad \tau_\varepsilon^A = \tau^A,\ \tau_{xu}^A = M_x^A \otimes \tau_u^A. \tag{57}$$

Vektori σ_u^A, za $u \in X^*$, nazivaju se ***σ-vektori***, dok se vektori τ_u^A, za $u \in X^*$, nazivaju ***τ-vektori***. U literaturi se σ-vektori često nazivaju i *generalized daters* ili *state vectors* [57, 66, 67, 79]. Obe familije vektora imaju značajnu ulogu u teoriji max-plus automata, naročito u postupcima njihove determinizacije.

Za dva max-plus automata $\mathcal{A}$ i $\mathcal{B}$, uređeni par (τ_u^A, τ_u^B), za $u \in X^*$, koji se sastoji od odgovarajućih τ-vektora automata $\mathcal{A}$ i $\mathcal{B}$, naziva se ***τ-par*** za automate $\mathcal{A}$ i $\mathcal{B}$. Analogno, uređeni par (σ_u^A, σ_u^B) naziva se ***σ-par*** za automate $\mathcal{A}$ i $\mathcal{B}$.

Neka su sada $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (m, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ max-plus automati nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$ i alfabetom X.

Matrica $U \in \mathbb{R}_\infty^{n\times m}$ naziva se ***slaba forward simulacija*** između automata $\mathcal{A}$ i $\mathcal{B}$ ako zadovoljava sledeći sistem matričnih nejednačina

$$\begin{aligned} &\text{(1-wfs)} \quad \sigma^A \leqslant \sigma^B \otimes U^\top, \\ &\text{(2-wfs)} \quad U^\top \otimes \tau_u^A \leqslant \tau_u^B, \qquad (u \in X^*). \end{aligned}$$

Matrica U naziva se ***slaba forward presimulacija*** između automata $\mathcal{A}$ i $\mathcal{B}$ ukoliko zadovoljava uslov (2-wfs), pri čemu nije neophodno da bude ispunjen uslov (1-wfs).

Analogno se definiše ***slaba forward bisimulacija***. Matrica $U \in \mathbb{R}_\infty^{n\times m}$ predstavlja slabu forward bisimulaciju između automata $\mathcal{A}$ i $\mathcal{B}$ ako zadovoljava sistem

$$\begin{aligned} &\text{(1-wfb)} \quad \sigma^A \leqslant \sigma^B \otimes U^\top, \qquad \sigma^B \leqslant \sigma^A \otimes U, \\ &\text{(2-wfb)} \quad U^\top \otimes \tau_u^A \leqslant \tau_u^B, \qquad U \otimes \tau_u^B \leqslant \tau_u^A, \qquad (u \in X^*). \end{aligned}$$

Ako je ispunjen samo uslov (2-wfb), tada se U naziva ***slaba forward prebisimulacija***.

Slabe backward simulacije i bisimulacije, kao i odgovarajuće slabe presimulacije i slabe prebisimulacije, definišu se na potpuno analogan način. Radi lakšeg poređenja, pregled svih tipova običnih i slabih simulacija i bisimulacija dat je u Tabeli 4.

Kod svih navedenih relacija odgovarajuće presimulacije, odnosno prebisimulacije dobijaju se izostavljanjem prve grupe uslova.

Iz Tabele 4 vidi se da se slabe simulacije i bisimulacije od odgovarajućih običnih simulacija i bisimulacija razlikuju jedino u uslovima kojima se opisuje ponašanje automata. Dok se kod običnih relacija usklađenost ponašanja izražava preko matrica prelaza za svaki simbol alfabeta, kod slabih relacija ona se izražava preko σ-vektora i τ-vektora određenih svim rečima iz skupa X^*. Na taj način poređenje ponašanja automata nije ograničeno na pojedinačne prelaze, već obuhvata određene sekvence prelaza.

Kao i u slučaju običnih simulacija i bisimulacija, odgovarajuće slabe simulacije, bisimulacije, presimulacije i prebisimulacije mogu se posmatrati kao rešenja odgovarajućih sistema matričnih nejednačina sa nepoznatom matricom U.

Tabela 4: Pregled običnih i slabih simulacija i bisimulacija.

Relacija	**Tip**	**Obične**	**Slabe**
Simulacije	Forward	(1-fs) $\sigma^A \leqslant \sigma^B \otimes U^\top$ (2-fs) $U^\top \otimes M_x^A \leqslant M_x^B \otimes U^\top$ (3-fs) $U^\top \otimes \tau^A \leqslant \tau^B$	(1-wfs) $\sigma^A \leqslant \sigma^B \otimes U^\top$ (2-wfs) $U^\top \otimes \tau_u^A \leqslant \tau_u^B$
	Backward	(1-bs) $\tau^A \leqslant U \otimes \tau^B$ (2-bs) $M_x^A \otimes U \leqslant U \otimes M_x^B$ (3-bs) $\sigma^A \otimes U \leqslant \sigma^B$	(1-wbs) $\tau^A \leqslant U \otimes \tau^B$ (2-wbs) $\sigma_u^A \otimes U \leqslant \sigma_u^B$
Bisimulacije	Forward	(1-fb) $\sigma^A \leqslant \sigma^B \otimes U^\top$, $\sigma^B \leqslant \sigma^A \otimes U$ (2-fb) $U^\top \otimes M_x^A \leqslant M_x^B \otimes U^\top$, $U \otimes M_x^B \leqslant M_x^A \otimes U$ (3-fb) $U^\top \otimes \tau^A \leqslant \tau^B$, $U \otimes \tau^B \leqslant \tau^A$	(1-wfb) $\sigma^A \leqslant \sigma^B \otimes U^\top$, $\sigma^B \leqslant \sigma^A \otimes U$ (2-wfb) $U^\top \otimes \tau_u^A \leqslant \tau_u^B$, $U \otimes \tau_u^B \leqslant \tau_u^A$
	Backward	(1-bb) $\tau^A \leqslant U \otimes \tau^B$, $\tau^B \leqslant U^\top \otimes \tau^A$ (2-bb) $M_x^A \otimes U \leqslant U \otimes M_x^B$, $M_x^B \otimes U^\top \leqslant U^\top \otimes M_x^A$ (3-bb) $\sigma^A \otimes U \leqslant \sigma^B$, $\sigma^B \otimes U^\top \leqslant \sigma^A$	(1-wbb) $\tau^A \leqslant U \otimes \tau^B$, $\tau^B \leqslant U^\top \otimes \tau^A$ (2-wbb) $\sigma_u^A \otimes U \leqslant \sigma_u^B$, $\sigma_u^B \otimes U^\top \leqslant \sigma_u^A$

Napomena: U svim izrazima u tabeli podrazumeva se da je $x \in X$, dok je $u \in X^*$.

Naredna teorema pokazuje da slabe simulacije i bisimulacije predstavljaju prirodno uopštenje odgovarajućih običnih simulacija i bisimulacija.

Teorema 4.1 *Neka su $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (m, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ max-plus automati nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$ i alfabetom X. Tada važe tvrđenja:*

(a) *Svaka forward (odnosno backward) presimulacija između automata $\mathcal{A}$ i $\mathcal{B}$ predstavlja slabu forward (odnosno slabu backward) presimulaciju između istih automata.*

Prema tome, svaka forward (odnosno backward) simulacija između automata $\mathcal{A}$ i $\mathcal{B}$ predstavlja slabu forward (odnosno slabu backward) simulaciju između istih automata.

(b) *Svaka forward (odnosno backward) prebisimulacija između automata $\mathcal{A}$ i $\mathcal{B}$ predstavlja slabu forward (odnosno slabu backward) prebisimulaciju između istih automata.*

Prema tome, svaka forward (odnosno backward) bisimulacija između automata $\mathcal{A}$ i $\mathcal{B}$ predstavlja slabu forward (odnosno slabu backward) bisimulaciju između istih automata.

Dokaz 4.1 *Dovoljno je dokazati prvo tvrđenje za* forward *presimulacije. Tvrđenje za* forward *simulacije neposredno sledi, budući da se* forward *simulacije i slabe* forward *simulacije od odgovarajućih presimulacija razlikuju jedino po uslovima* (1-fs) *i* (1-wfs)*, koji su identični. Ostala tvrđenja dokazuju se na potpuno analogan način.*

Neka je U forward presimulacija između automata $\mathcal{A}$ i $\mathcal{B}$. Tada, prema definiciji, važe uslovi (2-fs) *i* (3-fs)*.*

Indukcijom po dužini reči $u \in X^*$, pokažimo da važi uslov (2-wfs) $U^\top \otimes \tau_u^A \leqslant \tau_u^B$.

Za praznu reč ε tvrdnja neposredno sledi iz uslova (3-fs)*, jer je* $\tau_\varepsilon^A = \tau^A$ *i* $\tau_\varepsilon^B = \tau^B$.

Za reč dužine jedan, $|u| = 1$, tvrdnja sledi primenom 56, uslova (2-fs) *i* (3-fs)

$$U^T \otimes \tau_x^A = U^T \otimes M_x^A \otimes \tau^A \leq M_x^B \otimes U^T \otimes \tau^A \leq M_x^B \otimes \tau^B = \tau_x^B$$

Pretpostavimo da za svaku reč $u \in X^*$ dužine n važi tvrđenje $U^\top \otimes \tau_u^A \leqslant \tau_u^B$.

Neka je $v = xu$, gde je $x \in X$ i $|u| = n$. Tada, koristeći 56, uslov (2-fs) *i induktivnu pretpostavku, dobijamo*

$$U^\top \otimes \tau_v^A = U^\top \otimes M_x^A \otimes \tau_u^A \leqslant M_x^B \otimes U^\top \otimes \tau_u^A \leqslant M_x^B \otimes \tau_u^B = \tau_v^B.$$

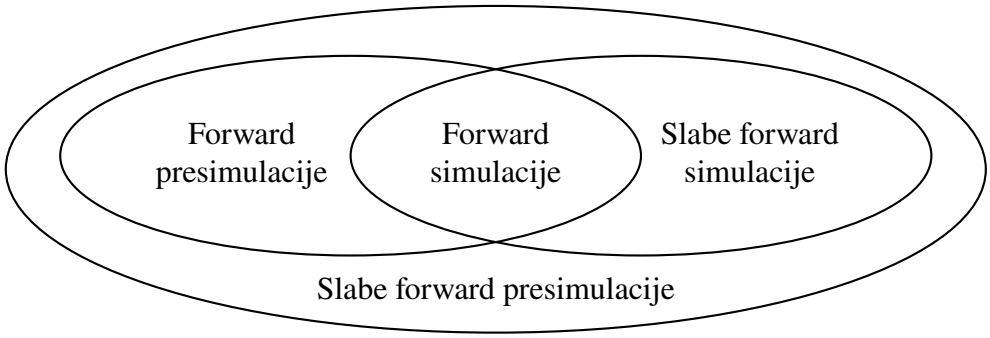
Time je indukcija završena, pa važi uslov (2-wfs) *za svaku reč $u \in X^*$. Dakle, U je slaba* forward *presimulacija.* □

Prethodna teorema pokazuje da slabe simulacije i bisimulacije predstavljaju prirodno uopštenje odgovarajućih simulacija i bisimulacija uvedenih u prethodnoj glavi. Naime, svaka simulacija, odnosno bisimulacija, istovremeno predstavlja i slabu simulaciju, odnosno slabu bisimulaciju istog tipa. Kao što je pokazano u prethodnoj glavi, postojanje simulacije između dva max-plus automata predstavlja potvrdu inkluzije jezika koje oni raspoznaju, dok postojanje bisimulacije predstavlja potvrdu njihove ekvivalencije. Zbog toga simulacije imaju značajnu ulogu u utvrđivanju inkluzije, a bisimulacije u utvrđivanju ekvivalencije max-plus automata.

Na osnovu prethodne teoreme, između klasa forward simulacija, forward presimulacija, slabih forward simulacija i slabih forward presimulacija važe sledeće relacije:

$$\mathsf{FS} \subseteq \mathsf{WFS} \subseteq \mathsf{WFPS}, \qquad \mathsf{FS} \subseteq \mathsf{FPS} \subseteq \mathsf{WFPS}, \qquad \mathsf{FS} = \mathsf{WFS} \cap \mathsf{FPS}, \tag{58}$$

gde FS, WFS, FPS i WFPS označavaju, redom, klase forward simulacija, slabih forward simulacija, forward presimulacija i slabih forward presimulacija. Ovi odnosi su ilustrovani na slici 15.



Slika 15: Odnosi između forward simulacija i odgovarajućih slabih relacija.

U narednoj teoremi biće pokazano da slabe simulacije i bisimulacije zadržavaju ista svojstva. Štaviše, imajući u vidu prethodnu teoremu, one u pojedinim slučajevima omogućavaju uspešnije utvrđivanje inkluzije i ekvivalencije nego odgovarajuće obične simulacije i bisimulacije.

Teorema 4.2 *Neka su* $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ *i* $\mathcal{B} = (m, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ *max-plus automati nad kompletnim max-plus poluprstenom* $\mathbb{R}_\infty$ *i alfabetom* X*. Tada važe sledeća tvrđenja:*

(a) *Postojanje slabe simulacije bilo kog tipa između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *implicira relaciju inkluzije, odnosno* $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!]$.

(b) *Postojanje slabe bisimulacije bilo kog tipa između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *implicira ekvivalenciju, odnosno* $[\![\mathcal{A}]\!] = [\![\mathcal{B}]\!]$.

Dokaz 4.2 *Dovoljno je dokazati tvrđenje* (a) *za slučaj slabe* forward *simulacije, dok se ostali slučajevi dokazuju na analogan način.*

Neka je U *slaba* forward *simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$*. Tada za proizvoljnu reč* $u \in X^*$*, na osnovu 36, 56, uslova* (1-wfs) *i* (2-wfs)*, dobijamo*

$$[\![\mathcal{A}]\!](u) = \sigma^A \otimes M_u^A \otimes \tau^A = \sigma^A \otimes \tau_u^A \leqslant \sigma^B \otimes U^\top \otimes \tau_u^A \leqslant \sigma^B \otimes \tau_u^B = \sigma^B \otimes M_u^B \otimes \tau^B = [\![\mathcal{B}]\!](u).$$

Kako je $u \in X^*$ *bila proizvoljno izabrana reč, sledi da je* $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!]$.

Tvrđenje (b) *neposredno sledi iz tvrđenja* (a)*, budući da slaba bisimulacija predstavlja slabu simulaciju u oba smera.* □

Prva od naredne dve teoreme uspostavlja vezu između slabih forward i slabih backward simulacija i bisimulacija, dok se druga odnosi na zatvorenost ovih relacija u odnosu na transponovanje, množenje i sabiranje matrica.

Teorema 4.3 *Neka su $\mathcal{A}$ i $\mathcal{B}$ max-plus automati nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$ i alfabetom X. Tada važe sledeća tvrđenja:*

(a) *Matrica $U \in \mathbb{R}_\infty^{n\times m}$ je slaba forward simulacija između automata $\mathcal{A}$ i $\mathcal{B}$ ako i samo ako je slaba backward simulacija između reverznih automata $\overline{\mathcal{A}}$ i $\overline{\mathcal{B}}$.*

(b) *Matrica $U \in \mathbb{R}_\infty^{n\times m}$ je slaba forward bisimulacija između automata $\mathcal{A}$ i $\mathcal{B}$ ako i samo ako je slaba backward bisimulacija između reverznih automata $\overline{\mathcal{A}}$ i $\overline{\mathcal{B}}$.*

Dokaz 4.3 *Tvrđenje (a) neposredno sledi iz definicija slabe forward i slabe backward simulacije, kao i iz osobine reverznog automata date u (43). Naime, za svaku reč $u \in X^*$ važi $\tau_u^{\overline{\mathcal{A}}} = \sigma_{\overline{u}}^{A}$, $\tau_u^{\overline{\mathcal{B}}} = \sigma_{\overline{u}}^{B}$, pa se uslovi* (1-wfs) *i* (2-wfs) *za automate $\mathcal{A}$ i $\mathcal{B}$ transformišu upravo u uslove* (1-wbs) *i* (2-wbs) *za reverzne automate $\overline{\mathcal{A}}$ i $\overline{\mathcal{B}}$.*

Analogno se pokazuje i tvrđenje (b).

Teorema 4.4 *Neka su $\mathcal{A}$, $\mathcal{B}$ i $\mathcal{C}$ max-plus automati nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$ i alfabetom X. Tada važe sledeća tvrđenja:*

(a) *Proizvod slabe forward simulacije između automata $\mathcal{A}$ i $\mathcal{B}$ i slabe forward simulacije između automata $\mathcal{B}$ i $\mathcal{C}$ jeste slaba forward simulacija između automata $\mathcal{A}$ i $\mathcal{C}$.*

(b) *Zbir proizvoljne familije slabih forward simulacija između automata $\mathcal{A}$ i $\mathcal{B}$ jeste slaba forward simulacija između automata $\mathcal{A}$ i $\mathcal{B}$.*

(c) *Transponovana matrica slabe forward (odnosno backward) bisimulacije između automata $\mathcal{A}$ i $\mathcal{B}$ predstavlja slabu forward (odnosno backward) bisimulaciju između automata $\mathcal{B}$ i $\mathcal{A}$.*

Tvrđenja (a) i (b) prethodne teoreme važe i kada se izraz *forward* zameni izrazom *backward* i/ili kada se izraz *simulacija* zameni izrazom *bisimulacija*.

U nastavku će biti razmotrena karakterizacija najvećih slabih simulacija i bisimulacija, kao i postupci za njihovo određivanje.

4.3 Najveće slabe simulacije i bisimulacije

Kao i u slučaju običnih simulacija i bisimulacija, problem određivanja najvećih slabih simulacija i bisimulacija svodi se na određivanje najvećih slabih presimulacija i prebisimulacija odgovarajućeg tipa.

U narednoj teoremi izvedeni su eksplicitni izrazi za najveće slabe forward i backward presimulacije i prebisimulacije. Na osnovu ovih izraza može se utvrditi postojanje odgovarajućih slabih simulacija i bisimulacija, kao i odrediti njihova najveća rešenja kada ona postoje.

Teorema 4.5 *Neka su* $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ *i* $\mathcal{B} = (m, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ *max-plus automati nad kompletnim max-plus poluprstenom* $\mathbb{R}_\infty$ *i alfabetom* X*, i neka su matrice*

$$U^{\mathrm{wfs}} = \bigwedge_{u\in X^*} \tau_u^A \backslash \tau_u^B, \qquad U^{\mathrm{wbs}} = \bigwedge_{u\in X^*} \sigma_u^A \backslash \sigma_u^B,$$
$$U^{\mathrm{wfb}} = \bigwedge_{u\in X^*} \tau_u^A \mathbin{\backslash\!\!\backslash} \tau_u^B, \qquad U^{\mathrm{wbb}} = \bigwedge_{u\in X^*} \sigma_u^A \mathbin{\backslash\!\!\backslash} \sigma_u^B.$$

Tada važe sledeća tvrđenja:

(a) *Matrica* U^{wfs} *je najveća slaba forward presimulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$*.*

Ukoliko U^{wfs} *zadovoljava uslov* (1-wfs)*, tada predstavlja najveću slabu forward simulaciju između automata* $\mathcal{A}$ *i* $\mathcal{B}$*. U suprotnom, između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *ne postoji slaba forward simulacija.*

(b) *Matrica* U^{wbs} *je najveća slaba backward presimulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$*.*

Ukoliko U^{wbs} *zadovoljava uslov* (1-wbs)*, tada predstavlja najveću slabu backward simulaciju između automata* $\mathcal{A}$ *i* $\mathcal{B}$*. U suprotnom, između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *ne postoji slaba backward simulacija.*

(c) *Matrica* U^{wfb} *je najveća slaba forward prebisimulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$*.*

Ukoliko U^{wfb} *zadovoljava uslov* (1-wfb)*, tada predstavlja najveću slabu forward bisimulaciju između automata* $\mathcal{A}$ *i* $\mathcal{B}$*. U suprotnom, između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *ne postoji slaba forward bisimulacija.*

(d) *Matrica* U^{wbb} *je najveća slaba backward prebisimulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$*.*

Ukoliko U^{wbb} *zadovoljava uslov* (1-wbb)*, tada predstavlja najveću slabu backward bisimulaciju između automata* $\mathcal{A}$ *i* $\mathcal{B}$*. U suprotnom, između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *ne postoji slaba backward bisimulacija.*

Dokaz 4.4 *Dovoljno je dokazati tvrđenje* (a)*, jer se preostala tvrđenja dokazuju na analogan način.*

Prema osobini reziduacije (40)*, za matricu* $U \in \mathbb{R}_\infty^{n\times m}$ *uslov* (2-wfs) *ekvivalentan je uslovima* $U \leqslant \tau_u^A \backslash \tau_u^B$*,* $u \in X^*$*. Otuda sledi da je* $U^{\mathrm{wfs}} = \bigwedge_{u\in X^*} \tau_u^A \backslash \tau_u^B$ *najveće rešenje uslova* (2-wfs)*, odnosno najveća slaba forward presimulacija između* $\mathcal{A}$ *i* $\mathcal{B}$*.*

Pretpostavimo sada da matrica U^{wfs} *zadovoljava i uslov* (1-wfs)*. Tada ona predstavlja slabu forward simulaciju između automata* $\mathcal{A}$ *i* $\mathcal{B}$*. Kako je svaka slaba forward simulacija ujedno i slaba forward presimulacija, za svaku slabu forward simulaciju* U *važi inkluzija* $U \leqslant U^{\mathrm{wfs}}$*. Prema tome,* U^{wfs} *predstavlja najveću slabu forward simulaciju između automata* $\mathcal{A}$ *i* $\mathcal{B}$*.*

Pretpostavimo sada da U^{wfs} *ne zadovoljava uslov* (1-wfs)*. Ako bi postojala slaba forward simulacija U između automata* $\mathcal{A}$ *i* $\mathcal{B}$*, onda bi ona ujedno bila i slaba forward presimulacija, pa bi važilo* $U \leqslant U^{\text{wfs}}$*. Pošto U zadovoljava uslov* (1-wfs)*, dobijamo*

$$\sigma^A \leqslant \sigma^B \otimes U^\top \leqslant \sigma^B \otimes (U^{\text{wfs}})^\top,$$

što pokazuje da i matrica U^{wfs} *zadovoljava uslov* (1-wfs)*, što je u suprotnosti sa pretpostavkom. Dakle, između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *ne postoji slaba* forward *simulacija.* □

Prethodna teorema pokazuje da se problem određivanja slabih simulacija i bisimulacija svodi na određivanje najvećih slabih presimulacija i prebisimulacija odgovarajućeg tipa. Naime, najveće slabe simulacije i bisimulacije mogu se dobiti neposredno iz najvećih slabih presimulacija i prebisimulacija proverom odgovarajućih uslova koji povezuju inicijalne, odnosno završne težinske vektore automata. Ukoliko su ti uslovi ispunjeni, dobijena rešenja predstavljaju najveće slabe simulacije, odnosno bisimulacije, dok u suprotnom odgovarajuće slabe simulacije, odnosno bisimulacije ne postoje. Stoga se problem određivanja najvećih slabih simulacija i bisimulacija u suštini svodi na efikasno računanje najvećih slabih presimulacija i prebisimulacija. U narednom odeljku biće opisan postupak za njihovo određivanje.

4.4 Procedura za računanje najvećih slabih rešenja

Prethodna teorema daje karakterizaciju najvećih slabih presimulacija i prebisimulacija odgovarajućeg tipa. Međutim, dobijeni izrazi zadati su u obliku infimuma po svim rečima iz skupa X^*, zbog čega se ne mogu neposredno koristiti za njihovo izračunavanje.

U ovoj sekciji biće opisan postupak za ispitivanje postojanja slabih simulacija i bisimulacija, kao i za određivanje njihovih najvećih rešenja kada ona postoje. Postupak se zasniva na određivanju najveće slabe presimulacije, odnosno prebisimulacije odgovarajućeg tipa, nakon čega se proverava da li dobijena matrica zadovoljava dodatni uslov kojim se definiše odgovarajuća slaba simulacija, odnosno bisimulacija. Na kraju sekcije biće prikazani primeri koji ilustruju primenu predloženog postupka, kao i situacije u kojima se njegovo izvršavanje ne završava nakon konačno mnogo koraka.

U nastavku će biti objašnjen rad Procedure 1 u slučaju kada je *simTip* = wfs, odnosno kada se određuje najveća slaba *forward* simulacija. U ostalim slučajevima postupak je analogan.

Algoritam započinje formiranjem početnog para $(\pi_0^A, \pi_0^B) = (\tau^A, \tau^B)$ i inicijalizacijom matrice $U = \pi_0^A \backslash \pi_0^B$. Nakon toga se sukcesivno obrađuju svi generisani parovi τ-vektora. Za svaki obrađeni par vektora (π^A, π^B) i svaki simbol $x \in X$ formira se novi par vektora $(M_x^A \otimes \pi^A,\ M_x^B \otimes \pi^B)$. Svaki novodobijeni par obrađuje se najviše jednom, dok se matrica U

Procedura 1 Postupak za određivanje najvećeg slabog rešenja između max-plus automata

Ulaz: Max-plus automati $\mathcal{A} = (n, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (m, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$;
ulazni alfabet X;
simType $\in$ {wfs, wbs, wfb, wbb} – tip slabe simulacije/bisimulacije.

Izlaz: Najveća slaba simulacija/bisimulacija U, ukoliko postoji, u suprotnom `null`.

Inicijalizacija praznog reda *toProcess* i praznog skupa *processedPairs*;
Inicijalizacija para

$$(\pi_0^A, \pi_0^B) \leftarrow \begin{cases} (\tau^A, \tau^B), & \text{if } \mathit{simType} \in \{\text{wfs}, \text{wfb}\}, \\ (\sigma^A, \sigma^B), & \text{if } \mathit{simType} \in \{\text{wbs}, \text{wbb}\}; \end{cases}$$

Inicijalizacija matrice

$$U \leftarrow \begin{cases} \pi_0^A \backslash \pi_0^B, & \text{if } \mathit{simType} \in \{\text{wfs}, \text{wbs}\}, \\ \pi_0^A \mathbin{/} \pi_0^B, & \text{if } \mathit{simType} \in \{\text{wfb}, \text{wbb}\}. \end{cases}$$

processedPairs.add((π_0^A, π_0^B));
toProcess.enqueue((π_0^A, π_0^B));
while *toProcess* is not empty **do**
    $(\pi^A, \pi^B) \leftarrow$ *toProcess*.dequeue();
    for each $x \in X$ **do**

$$(\pi_*^A, \pi_*^B) \leftarrow \begin{cases} (M_x^A \otimes \pi^A,\ M_x^B \otimes \pi^B), & \text{if } \mathit{simType} \in \{\text{wfs}, \text{wfb}\}, \\ (\pi^A \otimes M_x^A,\ \pi^B \otimes M_x^B), & \text{if } \mathit{simType} \in \{\text{wbs}, \text{wbb}\}. \end{cases}$$

        if $(\pi_*^A, \pi_*^B) \notin$ *processedPairs* **then**
            processedPairs.add((π_*^A, π_*^B));
            toProcess.enqueue((π_*^A, π_*^B));

$$U \leftarrow \begin{cases} U \wedge (\pi_*^A \backslash \pi_*^B), & \text{if } \mathit{simType} \in \{\text{wfs}, \text{wbs}\}, \\ U \wedge (\pi_*^A \mathbin{/} \pi_*^B), & \text{if } \mathit{simType} \in \{\text{wfb}, \text{wbb}\}. \end{cases}$$

        end if
    end for
end while

$$\mathit{condition} \leftarrow \begin{cases} \sigma^A \leqslant \sigma^B \otimes U^\top, & \text{if } \mathit{simType} = \text{wfs}, \\ \tau^A \leqslant U \otimes \tau^B, & \text{if } \mathit{simType} = \text{wbs}, \\ \left(\sigma^A \leqslant \sigma^B \otimes U^\top\right) \wedge \left(\sigma^B \leqslant \sigma^A \otimes U\right), & \text{if } \mathit{simType} = \text{wfb}, \\ \left(\tau^A \leqslant U \otimes \tau^B\right) \wedge \left(\tau^B \leqslant U^\top \otimes \tau^A\right), & \text{if } \mathit{simType} = \text{wbb}. \end{cases}$$

if *condition* **then**
    return U
else
    return `null`
end if

u svakoj iteraciji ažurira određivanjem infimuma sa odgovarajućim rezidualom. Postupak se završava kada više nije moguće generisati nove parove τ-vektora.

Ukoliko algoritam završi nakon konačno mnogo koraka, dobijena matrica U predstavlja *najveću slabu forward presimulaciju* između posmatranih automata. Nakon toga proverava se da li je zadovoljen uslov (1-wfs). Ukoliko je taj uslov ispunjen, matrica U predstavlja *najveću slabu forward simulaciju* između automata $\mathcal{A}$ i $\mathcal{B}$. U suprotnom se zaključuje da između posmatranih automata ne postoji slaba forward simulacija.

Iako algoritam u opštem slučaju ne garantuje završetak nakon konačno mnogo koraka, moguće je proceniti složenost jedne njegove iteracije. Neka su n i m brojevi stanja automata $\mathcal{A}$ i $\mathcal{B}$, respektivno, a $l = |X|$ kardinalnost ulaznog alfabeta. U svakoj iteraciji obrađuje se jedan par vektora (π^A, π^B), pri čemu se za svaki simbol $x \in X$ računaju proizvodi $M_x^A \otimes t^A$ i $M_x^B \otimes t^B$, a zatim se, ukoliko generisani par nije ranije obrađen, ažurira matrica U. S obzirom na dimenzije matrica $M_x^A \in \mathbb{R}_\infty^{n\times n}$, $M_x^B \in \mathbb{R}_\infty^{m\times m}$ i $U \in \mathbb{R}_\infty^{n\times m}$, složenost obrade jednog simbola iznosi $O(n^2 + m^2 + nm)$. Budući da se u svakoj iteraciji razmatra svih $l = |X|$ simbola ulaznog alfabeta, ukupna vremenska složenost jedne iteracije iznosi

$$O\Big(l \cdot (n^2 + m^2 + nm)\Big) = O\Big(l \cdot \max(n^2, m^2)\Big)\,.$$

Pri tome se zanemaruje složenost provere da li je generisani par već obrađen, koja zavisi od konkretne implementacije strukture podataka za čuvanje obrađenih parova.

Pošto broj generisanih parova vektora u opštem slučaju nije ograničen, ukupan broj iteracija ne mora biti konačan. Zbog toga završetak algoritma nije garantovan. U praktičnim implementacijama često se uvodi dodatni kriterijumi zaustavljanja, kao što je maksimalan broj iteracija ili vremensko ograničenje izvršavanja.

U nastavku su dati primeri koji ilustruju primenu opisanog postupka, kao i situacije u kojima se postupak ne završava nakon konačno mnogo koraka.

Primer 4.1 *Posmatrajmo max-plus automate nad kompletnim max-plus poluprstenom, $\mathcal{A} = (2, \sigma^A, \{M_x^A, M_y^A\}, \tau^A)$ i $\mathcal{B} = (3, \sigma^B, \{M_x^B, M_y^B\}, \tau^B)$, nad alfabetom $X = \{x, y\}$, zadate matricama i vektorima*

$$M_x^A = \begin{bmatrix} 0 & -\infty \\ -\infty & -\infty \end{bmatrix}, \qquad M_y^A = \begin{bmatrix} -\infty & 2 \\ -\infty & -\infty \end{bmatrix}, \qquad \sigma^A = \begin{bmatrix} 1 & 2 \end{bmatrix}, \qquad \tau^A = \begin{bmatrix} 1 \\ -2 \end{bmatrix},$$

$$M_x^B = \begin{bmatrix} 0 & -\infty & -\infty \\ -\infty & -\infty & 1 \\ -\infty & -\infty & 0 \end{bmatrix}, \quad M_y^B = \begin{bmatrix} -\infty & -\infty & 2 \\ -\infty & 0 & -\infty \\ -\infty & -\infty & 0 \end{bmatrix}, \quad \sigma^B = \begin{bmatrix} 0 & 1 & -1 \end{bmatrix}, \quad \tau^B = \begin{bmatrix} 2 \\ -1 \\ 0 \end{bmatrix}.$$

Prikažimo primenu postupka iz Procedure 1 za određivanje najveće slabe forward *simulacije između automata $\mathcal{A}$ i $\mathcal{B}$.*

Automat $\mathcal{A}$ ima četiri različita τ-vektora:

$$\tau^A_\varepsilon = \begin{bmatrix} 1 \\ -2 \end{bmatrix}; \quad \tau^A_{x^k} = \begin{bmatrix} 1 \\ -\infty \end{bmatrix},\ k \in \mathbb{N}; \quad \tau^A_{x^k y} = \begin{bmatrix} 0 \\ -\infty \end{bmatrix},\ k \in \mathbb{N}_0; \quad \tau^A_u = \begin{bmatrix} -\infty \\ -\infty \end{bmatrix},\ \textit{za ostale } u \in X^*.$$

S druge strane, automat $\mathcal{B}$ ima samo dva različita τ-vektora:

$$\tau^B_\varepsilon = \begin{bmatrix} 2 \\ -1 \\ 0 \end{bmatrix} = \tau^B_{y^k}, \quad \tau^B_u = \begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix},\ \textit{gde je } k \in \mathbb{N},\ \textit{odnosno } u \in X^+ \textit{ takvo da } |u|_x \neq 0.$$

Prema tome, tokom izvršavanja algoritma pojavljuje se ukupno šest različitih parova τ-vektora.

Tabela 5 prikazuje tok rada algoritma. Za svaki generisani par τ-vektora određuje se odgovarajući rezidual i ažurira trenutni infimum. Simbol $\boxtimes$ označava da je odgovarajući par već ranije generisan, zbog čega se iz njega ne formiraju novi parovi niti utiče na dalje računanje infimuma.

Na osnovu Tabele 5 dobija se da je najveća slaba forward presimulacija između automata $\mathcal{A}$ i $\mathcal{B}$:

$$U = \begin{bmatrix} 1 & -2 & -1 \\ 4 & 1 & 2 \end{bmatrix}.$$

Dalje, kako je ispunjena sledeća nejednakost, zaključujemo da je matrica U najveća slaba forward simulacija između automata $\mathcal{A}$ i $\mathcal{B}$:

$$\sigma^B \otimes U^\top = \begin{bmatrix} 0 & 1 & -1 \end{bmatrix} \otimes \begin{bmatrix} 1 & 4 \\ -2 & 1 \\ -1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 4 \end{bmatrix} \geq \begin{bmatrix} 1 & 2 \end{bmatrix} = \sigma^A.$$

Analogna primena algoritma za određivanje najveće slabe backward presimulacije daje:

$$U = \begin{bmatrix} -1 & -\infty & -2 \\ -\infty & -\infty & -3 \end{bmatrix}.$$

Kako uslov (1-wbs) *nije ispunjen, zaključujemo da između posmatranih automata ne postoji slaba backward simulacija:*

$$U \otimes \tau^B = \begin{bmatrix} -1 & -\infty & -2 \\ -\infty & -\infty & -3 \end{bmatrix} \otimes \begin{bmatrix} 2 \\ -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ -3 \end{bmatrix} \not\geq \begin{bmatrix} 1 \\ -2 \end{bmatrix} = \tau^A.$$

Ovaj primer pokazuje da postojanje slabe forward simulacije ne povlači postojanje slabe backward simulacije. Naime, izračunavanjem se može utvrditi da između posmatranih automata postoji slaba forward simulacija, dok slaba backward simulacija ne postoji.

Tabela 5: Proces računanja najveće slabe forward simulacije iz Primera 4.1.

Br. koraka	**Dužina reči**	**Parovi τ-vektora koji su izračunati**	**Reziduali izračunatih parova**	**Infimum nakon tekućeg koraka**
1	0	$(\tau^A_\varepsilon, \tau^B_\varepsilon) = \left(\begin{bmatrix}1\\-2\end{bmatrix}, \begin{bmatrix}2\\-1\\0\end{bmatrix}\right)$	$\tau^A_\varepsilon \backslash\!\!\!\backslash \tau^B_\varepsilon = \begin{bmatrix}1 & -2 & -1\\4 & 1 & 2\end{bmatrix}$	$U := \begin{bmatrix}1 & -2 & -1\\4 & 1 & 2\end{bmatrix}$
2	1	$(\tau^A_x, \tau^B_x) = \left(\begin{bmatrix}1\\-\infty\end{bmatrix}, \begin{bmatrix}2\\1\\0\end{bmatrix}\right)$	$\tau^A_x \backslash\!\!\!\backslash \tau^B_x = \begin{bmatrix}1 & 0 & -1\\+\infty & +\infty & +\infty\end{bmatrix}$	$U := \begin{bmatrix}1 & -2 & -1\\4 & 1 & 2\end{bmatrix}$
3		$(\tau^A_y, \tau^B_y) = \left(\begin{bmatrix}0\\-\infty\end{bmatrix}, \begin{bmatrix}2\\-1\\0\end{bmatrix}\right)$	$\tau^A_y \backslash\!\!\!\backslash \tau^B_y = \begin{bmatrix}2 & -1 & 0\\+\infty & +\infty & +\infty\end{bmatrix}$	$U := \begin{bmatrix}1 & -2 & -1\\4 & 1 & 2\end{bmatrix}$
4	2	$(\tau^A_{x^2}, \tau^B_{x^2}) = \left(\begin{bmatrix}1\\-\infty\end{bmatrix}, \begin{bmatrix}2\\1\\0\end{bmatrix}\right)$ ⊠		
5		$(\tau^A_{xy}, \tau^B_{xy}) = \left(\begin{bmatrix}0\\-\infty\end{bmatrix}, \begin{bmatrix}2\\1\\0\end{bmatrix}\right)$	$\tau^A_{xy} \backslash\!\!\!\backslash \tau^B_{xy} = \begin{bmatrix}2 & 1 & 0\\+\infty & +\infty & +\infty\end{bmatrix}$	$U := \begin{bmatrix}1 & -2 & -1\\4 & 1 & 2\end{bmatrix}$
6		$(\tau^A_{yx}, \tau^B_{yx}) = \left(\begin{bmatrix}-\infty\\-\infty\end{bmatrix}, \begin{bmatrix}2\\1\\0\end{bmatrix}\right)$	$\tau^A_{yx} \backslash\!\!\!\backslash \tau^B_{yx} = \begin{bmatrix}+\infty & +\infty & +\infty\\+\infty & +\infty & +\infty\end{bmatrix}$	$U := \begin{bmatrix}1 & -2 & -1\\4 & 1 & 2\end{bmatrix}$
7		$(\tau^A_{y^2}, \tau^B_{y^2}) = \left(\begin{bmatrix}-\infty\\-\infty\end{bmatrix}, \begin{bmatrix}2\\-1\\0\end{bmatrix}\right)$	$\tau^A_{y^2} \backslash\!\!\!\backslash \tau^B_{y^2} = \begin{bmatrix}+\infty & +\infty & +\infty\\+\infty & +\infty & +\infty\end{bmatrix}$	$U := \begin{bmatrix}1 & -2 & -1\\4 & 1 & 2\end{bmatrix}$
8	3	$(\tau^A_{x^2y}, \tau^B_{x^2y}) = \left(\begin{bmatrix}0\\-\infty\end{bmatrix}, \begin{bmatrix}2\\1\\0\end{bmatrix}\right)$ ⊠		
9		$(\tau^A_{yxy}, \tau^B_{yxy}) = \left(\begin{bmatrix}-\infty\\-\infty\end{bmatrix}, \begin{bmatrix}2\\1\\0\end{bmatrix}\right)$ ⊠		
10		$(\tau^A_{xyx}, \tau^B_{xyx}) = \left(\begin{bmatrix}-\infty\\-\infty\end{bmatrix}, \begin{bmatrix}2\\1\\0\end{bmatrix}\right)$ ⊠		
11		$(\tau^A_{y^2x}, \tau^B_{y^2x}) = \left(\begin{bmatrix}-\infty\\-\infty\end{bmatrix}, \begin{bmatrix}2\\1\\0\end{bmatrix}\right)$ ⊠		
12		$(\tau^A_{xy^2}, \tau^B_{xy^2}) = \left(\begin{bmatrix}-\infty\\-\infty\end{bmatrix}, \begin{bmatrix}2\\1\\0\end{bmatrix}\right)$ ⊠		
13		$(\tau^A_{y^3}, \tau^B_{y^3}) = \left(\begin{bmatrix}-\infty\\-\infty\end{bmatrix}, \begin{bmatrix}2\\-1\\0\end{bmatrix}\right)$ ⊠		

Analogno, razmatranjem reverznih automata dobija se primer u kome postoji slaba backward simulacija, dok slaba forward simulacija ne postoji. Prema tome, slabe forward i backward simulacije su međusobno nezavisni pojmovi, pa pri ispitivanju relacije inkluzije nijedan od ova dva tipa simulacije nema prednost u odnosu na drugi. Ukoliko jedan tip simulacije ne postoji, potrebno je ispitati postojanje drugog.

Primer 4.2 *Razmotrimo automate iz Primera 3.1, za koje je u prethodnom poglavlju određena najveća forward simulacija.*

Za iste automate može se odrediti i najveća slaba forward simulacija. Iako oba automata imaju beskonačno mnogo τ-vektora, analizom svih mogućih parova dobija se da je najveća slaba forward presimulacija

$$U = \bigwedge_{u\in X^*} \tau_u^A \backslash \tau_u^B = \begin{bmatrix} +\infty & +\infty & +\infty \\ -1 & -3 & 0 \end{bmatrix} \wedge \left(\bigwedge_{h(u)=x} \begin{bmatrix} 0 & 0 & -2\ell(u)+3 \\ 2\ell(u)-3 & 2\ell(u)-3 & 0 \end{bmatrix} \right)$$
$$= \begin{bmatrix} +\infty & +\infty & +\infty \\ -1 & -3 & 0 \end{bmatrix} \wedge \begin{bmatrix} 0 & 0 & -\infty \\ -1 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -\infty \\ -1 & -3 & 0 \end{bmatrix}.$$

Kako važi sledeća nejednakost, zaključujemo da je U najveća slaba forward simulacija:

$$\sigma^B \otimes U^\top = \begin{bmatrix} 1 & -\infty & -\infty \end{bmatrix} \otimes \begin{bmatrix} 0 & -1 \\ 0 & -3 \\ -\infty & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} > \begin{bmatrix} 1 & -\infty \end{bmatrix} = \sigma^A.$$

Pri tome, najveća slaba forward simulacija poklapa se sa najvećom forward simulacijom određenom u Primeru 3.1. Time se pokazuje da uvođenje slabih simulacija ne mora nužno dovesti do novih simulacionih relacija, već u pojedinim slučajevima daje isti rezultat kao i odgovarajuća obična simulacija.

Primer 4.3 *Za alfabet $X = \{x\}$, neka su dati max-plus automati $\mathcal{A} = (3, \sigma^A, \{M_x^A\}, \tau^A)$ i $\mathcal{B} = (2, \sigma^B, \{M_x^B\}, \tau^B)$ određeni matricama i vektorima*

$$M_x^A = \begin{bmatrix} 3 & 5 & -\infty \\ 8 & 6 & -\infty \\ 4 & 2 & 0 \end{bmatrix}, \qquad \sigma^A = \begin{bmatrix} 5 & 4 & -\infty \end{bmatrix}, \qquad \tau^A = \begin{bmatrix} -\infty \\ -\infty \\ 1 \end{bmatrix},$$

$$M_x^B = \begin{bmatrix} 2 & -\infty \\ -\infty & -\infty \end{bmatrix}, \qquad \sigma^B = \begin{bmatrix} -\infty & 1 \end{bmatrix}, \qquad \tau^B = \begin{bmatrix} -\infty \\ 1 \end{bmatrix}.$$

Postupak iz Algoritma 1 prikazan je u Tabeli 6.

Dobija se da je najveća slaba forward presimulacija

$$U = \begin{bmatrix} +\infty & +\infty \\ +\infty & +\infty \\ -\infty & -\infty \end{bmatrix}.$$

Kako važi sledeća nejednakost, zaključujemo da je U najveća slaba forward simulacija između automata $\mathcal{A}$ i $\mathcal{B}$:

$$\sigma^B \otimes U^\top = \begin{bmatrix} -\infty & 1 \end{bmatrix} \otimes \begin{bmatrix} +\infty & +\infty & -\infty \\ +\infty & +\infty & -\infty \end{bmatrix} = \begin{bmatrix} +\infty & +\infty & -\infty \end{bmatrix} > \begin{bmatrix} 5 & 4 & -\infty \end{bmatrix} = \sigma^A.$$

Tabela 6: Proces računanja najveće slabe forward simulacije iz Primera 4.3.

Br. koraka	**Dužina reči**	**Parovi τ-vektora koji su izračunati**	**Reziduali različitih parova**	**Infimum nakon tekućeg koraka**
1	0	$(\tau_\varepsilon^A, \tau_\varepsilon^B) = \left(\begin{bmatrix}-\infty\\-\infty\\1\end{bmatrix}, \begin{bmatrix}-\infty\\1\end{bmatrix}\right)$	$\tau_\varepsilon^A \backslash \tau_\varepsilon^B = \begin{bmatrix}+\infty & +\infty\\+\infty & +\infty\\-\infty & 0\end{bmatrix}$	$U := \begin{bmatrix}+\infty & +\infty\\+\infty & +\infty\\-\infty & 0\end{bmatrix}$
2	1	$(\tau_x^A, \tau_x^B) = \left(\begin{bmatrix}-\infty\\-\infty\\1\end{bmatrix}, \begin{bmatrix}-\infty\\-\infty\end{bmatrix}\right)$	$\tau_x^A \backslash \tau_x^B = \begin{bmatrix}+\infty & +\infty\\+\infty & +\infty\\-\infty & -\infty\end{bmatrix}$	$U := \begin{bmatrix}+\infty & +\infty\\+\infty & +\infty\\-\infty & -\infty\end{bmatrix}$
3	2	$(\tau_{x^2}^A, \tau_{x^2}^B) = \left(\begin{bmatrix}-\infty\\-\infty\\1\end{bmatrix}, \begin{bmatrix}-\infty\\-\infty\end{bmatrix}\right)$ $\boxtimes$		

Sa druge strane, primenom postupka iz Teoreme 4.1 rada [9] dobija se niz matrica čija je granična vrednost najveća forward presimulacija između automata $\mathcal{A}$ i $\mathcal{B}$:

$$U_1 = \begin{bmatrix}+\infty & +\infty\\+\infty & +\infty\\-\infty & 0\end{bmatrix}, \qquad U_s = \begin{bmatrix}+\infty & -\infty\\+\infty & -\infty\\-\infty & -\infty\end{bmatrix}, \quad s \geqslant 2, \qquad \widehat{U} = \begin{bmatrix}+\infty & -\infty\\+\infty & -\infty\\-\infty & -\infty\end{bmatrix}.$$

Međutim, $\widehat{U}$ ne zadovoljava uslov, pa između automata $\mathcal{A}$ i $\mathcal{B}$ ne postoji forward simulacija:

$$\sigma^B \otimes (\widehat{U})^\top = \begin{bmatrix}-\infty & 1\end{bmatrix} \otimes \begin{bmatrix}+\infty & +\infty & -\infty\\-\infty & -\infty & -\infty\end{bmatrix} = \begin{bmatrix}-\infty & -\infty & -\infty\end{bmatrix} \not\geqslant \begin{bmatrix}5 & 4 & -\infty\end{bmatrix} = \sigma^A.$$

Ovaj primer pokazuje da slaba forward simulacija može postojati čak i kada odgovarajuća forward simulacija ne postoji, čime se potvrđuje da slabe simulacije predstavljaju strogo opštiji pojam od običnih simulacija.

Rezultati predstavljeni u ovoj glavi pružaju teorijsku osnovu za određivanje najvećih slabih simulacija i bisimulacija između max-plus automata. Međutim, pošto završetak opisanog postupka u opštem slučaju nije garantovan, njegova praktična primena može biti ograničena. Zbog toga će u narednoj glavi biti uvedene relaksacije slabih simulacija i bisimulacija, koje predstavljaju osnovu za razvoj efikasnijih postupaka čije je izvršavanje garantovano nakon konačno mnogo koraka.

Glava 5

Relaksacija slabih simulacija i bisimulacija za max-plus automate

U prethodnoj glavi uvedeni su pojmovi slabih simulacija i bisimulacija za max-plus automate i pokazano je da njihovo postojanje predstavlja dovoljan uslov za potvrdu inkluzije, odnosno ekvivalencije ponašanja automata. Takođe je pokazano da se određivanje najvećih slabih simulacija i bisimulacija svodi na rešavanje odgovarajućih sistema matričnih nejednačina.

Međutim, za razliku od običnih simulacija i bisimulacija, sistemi koji definišu slabe simulacije i bisimulacije u opštem slučaju sadrže beskonačno mnogo matričnih nejednačina, budući da se odgovarajući uslovi formulišu za sve reči slobodnog monoida X^* [17, 80]. Zbog toga se procedura razvijena u prethodnoj glavi ne mora završiti nakon konačno mnogo koraka, pa određivanje najvećih slabih simulacija i bisimulacija predstavlja računarski zahtevan problem za koji u opštem slučaju nije garantovan završetak postupka. Usled toga, u opštem slučaju nije poznat efikasan postupak za određivanje najvećih slabih simulacija i bisimulacija između dva proizvoljna max-plus automata.

Jedan od mogućih načina za prevazilaženje ovog problema jeste relaksacija sistema matričnih nejednačina kojima su definisane slabe simulacije i bisimulacije [19, 80, 81, 82]. Osnovna ideja sastoji se u tome da se, umesto svih reči iz skupa X^*, razmatra samo odgovarajući podskup reči. Na taj način beskonačan sistem matričnih nejednačina zamenjuje se konačnim sistemom, čime se omogućava efektivnije određivanje njihovih najvećih rešenja. Tako se dobijaju relaksirane slabe simulacije i bisimulacije, koje više ne potvrđuju inkluziju i ekvivalenciju ponašanja automata u potpunosti, već samo u okviru posmatranog skupa reči. Iako predstavljaju aproksimaciju slabih simulacija i bisimulacija, ove relacije zadržavaju njihovu osnovnu interpretaciju u pogledu poređenja ponašanja automata, a istovremeno omogućavaju efikasniju računarsku obradu i praktičnu primenu. Pored toga, u mnogim praktičnim primenama nije neophodno porediti ponašanje automata za sve reči iz skupa X^*, već samo za određenu klasu reči relevantnih za konkretnu primenu. Relaksirane slabe simulacije i bisimulacije omogućavaju upravo takav oblik parcijalnog poređenja ponašanja automata.

U ovoj glavi uvodi se opšti pojam relaksiranih slabih simulacija i bisimulacija za max-plus automate, kao i odgovarajući pojmovi relaksiranih presimulacija i prebisimulacija. Izvode se karakterizacije njihovih najvećih rešenja i ispituje njihova veza sa slabim

simulacijama i bisimulacijama uvedenim u prethodnoj glavi.

Posebna pažnja posvećena je dvema prirodnim klasama relaksacija. Prva se zasniva na ograničavanju dužine posmatranih reči, pri čemu se razmatraju samo reči čija dužina ne prelazi unapred zadatu vrednost. Druga se zasniva na ograničavanju datuma realizacije reči, pri čemu se posmatraju samo reči čija je vrednost u ponašanju bar jednog od dva posmatrana automata manja od unapred zadate vremenske granice. Kada se reči posmatraju kao sekvence događaja, ova relaksacija odgovara razmatranju samo onih sekvenci događaja koje se u bar jednom od automata realizuju do zadatog datuma. Za obe klase relaksacija razvijeni su postupci za određivanje najvećih relaksiranih slabih simulacija i bisimulacija, a njihova primena ilustrovana je odgovarajućim primerima.

Rezultati prikazani u ovoj glavi predstavljaju nastavak istraživanja simulacija i bisimulacija za max-plus automate započetih u prethodnim poglavljima i zasnovani su na do sada neobjavljenim rezultatima proisteklim iz istraživanja sprovedenog u okviru ove doktorske disertacije.

5.1 Unapređeni postupci za ispitivanje inkluzije i ekvivalencije

Kao što je pokazano u prethodnoj glavi, određivanje najvećih slabih presimulacija i prebisimulacija između dva max-plus automata svodi se na rešavanje odgovarajućih sistema matričnih nejednačina. Takođe je razvijena procedura za određivanje najvećih slabih presimulacija i prebisimulacija, kao i za ispitivanje da li dobijena relacija zadovoljava dodatne uslove potrebne da bude slaba simulacija, odnosno bisimulacija odgovarajućeg tipa. Efikasnost ovih postupaka neposredno zavisi od konačnosti skupa svih σ-parova, odnosno τ-parova posmatranih automata [17].

Rezultati predstavljeni u ovoj sekciji, kao i odgovarajući postupci, zasnivaju na svojstvima koja nisu specifična samo za max-plus poluprsten. Naime, analogni rezultati važe i za težinske automate nad proizvoljnim kompletnim aditivno-idempotentnim poluprstenom. U nastavku se, međutim, ograničavamo na max-plus automate.

U Proceduri 1 pokazano je da se u postupku za određivanje najveće slabe simulacije, odnosno bisimulacije, sukcesivno generišu σ-parovi ili τ-parovi automata, pri čemu se za svaki generisani par određuju odgovarajući uslovi i ažurira kandidat za najveće rešenje. Međutim, ukoliko je cilj samo ispitivanje inkluzije ili ekvivalencije ponašanja automata, nije neophodno određivati samu slabu simulaciju, odnosno bisimulaciju. U tom slučaju postupak se može pojednostaviti tako što se za svaki generisani par neposredno proverava odgovarajući uslov koji izražava inkluziju, odnosno ekvivalenciju funkcija jezika automata.

Budući da skup svih σ-parova, odnosno skup svih τ-parova, dva max-plus automata u opštem slučaju može biti beskonačan, sistemi matričnih nejednačina kojima su definisane slabe simulacije i bisimulacije mogu sadržati beskonačno mnogo uslova. Zbog toga se nji-

hova najveća rešenja u opštem slučaju ne mogu odrediti postupkom koji garantuje završetak nakon konačno mnogo koraka. Zbog toga je potrebno razmotriti odgovarajuće relaksacije ovih sistema, kojima se beskonačan sistem uslova zamenjuje konačnim sistemom.

Osnovna ideja relaksacije sastoji se u tome da se, umesto svih reči slobodnog monoida X^*, razmatra samo unapred zadati skup reči $L \subseteq X^*$. Na taj način dobijaju se relaksirani sistemi matričnih nejednačina čija se rešenja nazivaju *relaksirane slabe simulacije* i *relaksirane slabe bisimulacije*. Ove relacije predstavljaju aproksimaciju slabih simulacija i bisimulacija, jer potvrđuju inkluziju i ekvivalenciju ponašanja automata samo u okviru skupa reči L.

Neka je u nastavku $L \subseteq X^*$ proizvoljan skup reči nad alfabetom X. Za max-plus automate $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$, umesto sistema matričnih nejednačina koji definišu slabe simulacije i bisimulacije, razmatraćemo odgovarajuće relaksirane sisteme u kojima se uslovi koji uključuju σ- i τ-parove zahtevaju samo za reči koje pripadaju skupu L. Pre nego što pređemo na razmatranje konkretnih relaksacija sistema matričnih nejednačina, pokazaćemo da konačnost skupa svih σ-parova ili τ-parova omogućava efikasno rešavanje problema inkluzije i ekvivalencije ponašanja max-plus automata.

Teorema 5.1 *Ako je skup svih τ-parova ili skup svih σ-parova max-plus automata $\mathcal{A}$ i $\mathcal{B}$ konačan, tada se problemi inkluzije i ekvivalencije ponašanja automata $\mathcal{A}$ i $\mathcal{B}$ mogu efikasno rešiti.*

Dokaz 5.1 *Dokaz izvodimo za slučaj kada je konačan skup svih τ-parova. Dokaz za slučaj σ-parova je analogan.*

Pretpostavimo da je skup svih τ-parova konačan i da je dat sa

$$\{(\tau_{u_1}^A, \tau_{u_1}^B), (\tau_{u_2}^A, \tau_{u_2}^B), \ldots, (\tau_{u_k}^A, \tau_{u_k}^B)\}, \text{ za neko } k \in \mathbb{N}.$$

Tada za svaku reč $u \in X^*$ postoji indeks $i \in [1..k]$ takav da važi $(\tau_u^A, \tau_u^B) = (\tau_{u_i}^A, \tau_{u_i}^B)$, odnosno $\tau_u^A = \tau_{u_i}^A$ i $\tau_u^B = \tau_{u_i}^B$.

Otuda sledi da su

$$[\![\mathcal{A}]\!](u) = \sigma^A \otimes \tau_u^A = \sigma^A \otimes \tau_{u_i}^A = [\![\mathcal{A}]\!](u_i) \quad i \quad [\![\mathcal{B}]\!](u) = [\![\mathcal{B}]\!](u_i).$$

Prema tome, svaka vrednost funkcije jezika automata $\mathcal{A}$ (odnosno $\mathcal{B}$) jednaka je jednoj od vrednosti

$$[\![\mathcal{A}]\!](u_1), [\![\mathcal{A}]\!](u_2), \ldots, [\![\mathcal{A}]\!](u_k) \quad i \quad [\![\mathcal{B}]\!](u_1), [\![\mathcal{B}]\!](u_2), \ldots, [\![\mathcal{B}]\!](u_k),$$

koje se dobijaju množenjem odgovarajućih τ-parova vektorima σ^A i σ^B sa leve strane.

Na osnovu toga može se formulisati efikasan postupak za rešavanje problema inkluzije i ekvivalencije. Naime, nakon određivanja pojedinačnog τ-para $(\tau_{u_i}^A, \tau_{u_i}^B)$, umesto izraču-

navanja odgovarajućeg reziduala i parcijalnog infimuma, dovoljno je izračunati proizvode $\sigma^A \otimes \tau^A_{u_i}$ *i* $\sigma^B \otimes \tau^B_{u_i}$ *i proveriti da li važi odgovarajuća nejednakost. U slučaju ispitivanja inkluzije proverava se nejednakost 59 (a) , odnosno u slučaju ispitivanja ekvivalencije proverava se 59 (b).*

$$\sigma^A \otimes \tau^A_{u_i} \leqslant \sigma^B \otimes \tau^B_{u_i} \quad \text{(a)}, \qquad \sigma^A \otimes \tau^A_{u_i} = \sigma^B \otimes \tau^B_{u_i} \quad \text{(b)}. \tag{59}$$

Potpuno analogan postupak dobija se korišćenjem σ*-parova umesto* τ*-parova.* □

Prethodna teorema pokazuje da se, ukoliko je skup svih σ-parova, odnosno τ-parova, konačan, problemi inkluzije i ekvivalencije mogu rešavati proverom konačnog broja uslova. Ova činjenica omogućava i određena poboljšanja postupka za određivanje najvećih slabih simulacija i bisimulacija uvedenog u Proceduri 1. Osnovna ideja sastoji se u tome da se tokom generisanja σ-parova, odnosno τ-parova, istovremeno proverava da li trenutno dobijeni kandidat za najveću slabu simulaciju ili bisimulaciju zadovoljava odgovarajuće početne ili završne uslove. Ukoliko se utvrdi da neki od ovih uslova nije zadovoljen, postupak se može prekinuti bez daljeg generisanja novih parova. Na taj način ne menja se rezultat postupka, već se u određenim slučajevima omogućava ranije otkrivanje da odgovarajuća slaba simulacija ili bisimulacija ne postoji. Ideja ovakvog unapređenja prikazana je u Proceduri 2.

Međutim, u mnogim situacijama nije potrebno određivati najveću slabu simulaciju ili bisimulaciju (što predstavlja izlaz Procedura 1 i 2), već je dovoljno ispitati da li između posmatranih automata postoji inkluzija, odnosno ekvivalencija ponašanja. U tom slučaju nije potrebno računati reziduale i parcijalne infimume, niti određivati matricu koja predstavlja kandidata za najveću slabu simulaciju ili bisimulaciju. Umesto toga, inkluzija ili ekvivalencija može se ispitati direktno, proverom vrednosti ponašanja automata na generisanim σ-parovima, odnosno τ-parovima.

Na osnovu Teoreme 5.1 može se formulisati postupak za direktno ispitivanje inkluzije i ekvivalencije korišćenjem σ-parova ili τ-parova. Osnovna ideja sastoji se u tome da se za svaki generisani par neposredno proverava odgovarajući uslov na vrednosti funkcija jezika automata. U slučaju korišćenja τ-parova proveravaju se uslovi

$$\sigma^A \otimes \tau^A_u \leqslant \sigma^B \otimes \tau^B_u, \qquad \sigma^A \otimes \tau^A_u = \sigma^B \otimes \tau^B_u$$

dok se u slučaju korišćenja σ-parova proveravaju uslovi

$$\sigma^A_u \otimes \tau^A \leqslant \sigma^B_u \otimes \tau^B, \qquad \sigma^A_u \otimes \tau^A = \sigma^B_u \otimes \tau^B.$$

Ukoliko odgovarajući uslov nije zadovoljen za neki generisani par, postupak se odmah zaustavlja i zaključuje se da inkluzija, odnosno ekvivalencija, ne važi. U suprotnom, generisanje i obrada novih parova se nastavljaju. Ako je skup svih σ-parova, odnosno τ-parova, konačan, svaki različiti par može biti obrađen najviše jednom, pa se postupak

Procedura 2 Unapređeni postupak za određivanje najveće slabe simulacije/bisimulacije

Ulaz: Max-plus automati $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$;
ulazni alfabet X;
simType $\in$ {wfs, wbs, wfb, wbb} – specificira tip slabe simulacije/bisimulacije.

Izlaz: Najveća slaba simulacija/bisimulacija U, ukoliko postoji, ili null

Inicijalizacija praznog reda *toProcess* i praznog skupa *uniquePairs*;
Inicijalizacija para

$$(\pi_0^A, \pi_0^B) \leftarrow \begin{cases} (\tau^A, \tau^B), & \text{if } simType \in \{\text{wfs, wfb}\} \\ (\sigma^A, \sigma^B), & \text{if } simType \in \{\text{wbs, wbb}\} \end{cases};$$

Inicijalizacija matrice

$$U \leftarrow \begin{cases} \pi_0^A \backslash \pi_0^B & \text{if } simType \in \{\text{wfs, wbs}\} \\ \pi_0^A \phi \pi_0^B & \text{if } simType \in \{\text{wfb, wbb}\} \end{cases};$$

function CheckCondition($U, \mathcal{A}, \mathcal{B}, simType$)

$$\textbf{return } condition \;=\; \begin{cases} \sigma^A \leqslant \sigma^B \otimes U^T & \text{if } simType = \text{wfs} \\ \tau^A \leqslant U \otimes \tau^B & \text{if } simType = \text{wbs} \\ \sigma^A \leqslant \sigma^B \otimes U^\top \text{ and } \sigma^B \leqslant \sigma^A \otimes U & \text{if } simType = \text{wfb} \\ \tau^A \leqslant U \otimes \tau^B \text{ and } \tau^B \leqslant U^\top \otimes \tau^A & \text{if } simType = \text{wbb} \end{cases};$$

end function
condition $\leftarrow$ CheckCondition($U, \mathcal{A}, \mathcal{B}, simType$);
if *not condition* **then**
    return null
end if
uniquePairs.add((π_0^A, π_0^B));
toProcess.enqueue((π_0^A, π_0^B));
while *toProcess* is not empty **do**
    $(\pi^A, \pi^B) \leftarrow$ *toProcess*.dequeue();
    for each $x \in X$ **do**
        $$(\pi_*^A, \pi_*^B) \leftarrow \begin{cases} (M_x^A \otimes \pi^A, M_x^B \otimes \pi^B) & \text{if } simType \in \{\text{wfs, wfb}\} \\ (\pi^A \otimes M_x^A, \pi^B \otimes M_x^B) & \text{if } simType \in \{\text{wbs, wbb}\} \end{cases};$$
        if $(\pi_*^A, \pi_*^B) \notin$ *uniquePairs* **then**
            uniquePairs.add((π_*^A, π_*^B));
            toProcess.enqueue((π_*^A, π_*^B));
            $$U \leftarrow \begin{cases} U \wedge (\pi_*^A \backslash \pi_*^B) & \text{if } simType \in \{\text{wfs, wbs}\} \\ U \wedge (\pi_*^A \phi \pi_*^B) & \text{if } simType \in \{\text{wfb, wbb}\} \end{cases};$$
            condition $\leftarrow$ CheckCondition($U, \mathcal{A}, \mathcal{B}, simType$);
            if *not condition* **then**
                return null
            end if
        end if
    end for
end while
return U

završava nakon konačno mnogo koraka. Na taj način dobija se efikasan postupak za direktno ispitivanje inkluzije i ekvivalencije ponašanja automata. Ideja opisanog postupka prikazana je u Proceduri 3.

Procedura 3 Testiranje inkluzije i ekvivalencije između max-plus automata

Ulaz: Max-plus automati $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$;
Ulazni alfabet X;
$purp \in \{\text{cont}, \text{equiv}\}$ (specificira namenu – da li testiramo inkluziju ili ekvivalenciju);
$appr \in \{\sigma, \tau\}$ (specificira pristup – da li koristimo σ-parove ili τ-parove).

Izlaz: YES, ukoliko postoji inkluzija/ekvivalencija između $\mathcal{A}$ i $\mathcal{B}$, ili null, ukoliko ne postoji

Inicijalizuje prazan red *toProcess* i prazan skup *uniquePairs*;
Dodeljuje vrednost inicijalnom paru:

$$(\pi_0^A, \pi_0^B) \leftarrow \begin{cases} (\tau^A, \tau^B), & \text{if } appr = \tau \\ (\sigma^A, \sigma^B), & \text{if } appr = \sigma \end{cases};$$

uniquePairs.add((π_0^A, π_0^B));
toProcess.enqueue((π_0^A, π_0^B));
while *toProcess* is not empty **do**
    $(\pi^A, \pi^B) \leftarrow$ *toProcess*.dequeue();

$$condition = \begin{cases} \sigma^A \otimes \pi^A \leqslant \sigma^B \otimes \pi^B & \text{if } purp = \text{cont},\ appr = \tau \\ \pi^A \otimes \tau^A \leqslant \pi^B \otimes \tau^B & \text{if } purp = \text{cont},\ appr = \sigma \\ \sigma^A \otimes \pi^A = \sigma^B \otimes \pi^B & \text{if } purp = \text{equiv},\ appr = \tau \\ \pi^A \otimes \tau^A = \pi^B \otimes \tau^B & \text{if } purp = \text{equiv},\ appr = \sigma \end{cases};$$

    if *not condition* **then**
        return null
    end if
    for each $x \in X$ **do**
        $(\pi_*^A, \pi_*^B) \leftarrow \begin{cases} (M_x^A \otimes \pi^A, M_x^B \otimes \pi^B) & \text{if } appr = \tau \\ (\pi^A \otimes M_x^A, \pi^B \otimes M_x^B) & \text{if } appr = \sigma \end{cases};$
        if $(\pi_*^A, \pi_*^B) \notin$ *uniquePairs* **then**
            uniquePairs.add((π_*^A, π_*^B));
            toProcess.enqueue((π_*^A, π_*^B));
        end if
    end for
end while
return YES

Iako se u slučaju konačnog skupa σ-parova, odnosno τ-parova, inkluzija i ekvivalencija mogu ispitati direktno, određivanje slabih simulacija i bisimulacija ima širi značaj. Naime, za razliku od postupka koji samo utvrđuje da li određena inkluzija ili ekvivalencija

važi, slabe simulacije i bisimulacije daju dodatnu informaciju o odnosima između stanja posmatranih automata koji dovode do takvog odnosa njihovih ponašanja. Ove informacije mogu biti od značaja i u postupcima redukcije broja stanja i determinizacije max-plus automata.

Iako se uvođenjem Procedure 3 izbegava računanje najveće slabe simulacije ili bisimulacije, njena primena i dalje zavisi od konačnosti skupa svih σ-parova, odnosno τ-parova. U opštem slučaju ta konačnost nije garantovana. Zbog toga ćemo u nastavku razmotriti relaksacije kojima se ograničava skup reči za koje se zahtevaju odgovarajući uslovi. Na taj način broj relevantnih σ-parova, odnosno τ-parova, postaje konačan, čime se obezbeđuje završetak postupka nakon konačno mnogo koraka.

5.2 Relaksirane slabe simulacije i bisimulacije

U prethodnoj glavi uvedene su slabe simulacije i bisimulacije za max-plus automate kao rešenja odgovarajućih sistema matričnih nejednačina definisanih za sve reči slobodnog monoida X^*. U ovoj sekciji razmatramo njihove relaksacije dobijene ograničavanjem tih uslova na unapred zadati skup reči.

Neka je $L \subseteq X^*$ proizvoljan jezik nad alfabetom X. U kontekstu max-plus automata, jezik L može se interpretirati kao skup sekvenci događaja za koje se proveravaju odgovarajući uslovi simulacije ili bisimulacije.

Neka su $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ max-plus automati nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$. Umesto sistema matričnih nejednačina koji definišu slabe simulacije i bisimulacije za sve reči iz X^*, razmatraćemo odgovarajuće relaksirane sisteme u kojima se uslovi koji zavise od reči zahtevaju samo za reči $u \in L$.

Za slabe forward simulacije razmatraćemo sistem

$$\begin{aligned} &\text{(1-wfs)} \quad \sigma^A \leqslant \sigma^B \otimes U^\top, \\ &\text{(2_L-wfs)} \quad U^\top \otimes \tau_u^A \leqslant \tau_u^B, \quad \text{za svako } u \in L, \end{aligned}$$

čija se rešenja nazivaju ***L-relaksirane slabe forward simulacije*** između posmatranih automata $\mathcal{A}$ i $\mathcal{B}$.

Za slabe backward simulacije razmatraćemo sistem

$$\begin{aligned} &\text{(1-wbs)} \quad \tau^A \leqslant U \otimes \tau^B, \\ &\text{(2_L-wbs)} \quad \sigma_u^A \otimes U \leqslant \sigma_u^B, \quad \text{za svako } u \in L, \end{aligned}$$

čija se rešenja nazivaju ***L-relaksirane slabe backward simulacije*** za automate $\mathcal{A}$ i $\mathcal{B}$.

Za slabe forward bisimulacije razmatraćemo sistem

$$\begin{aligned}
&\text{(1-wfb)} \quad \sigma^A \leqslant \sigma^B \otimes U^\top, \quad \sigma^B \leqslant \sigma^A \otimes U,\\
&\text{(2_L-wfb)} \quad U^\top \otimes \tau_u^A \leqslant \tau_u^B, \quad U \otimes \tau_u^B \leqslant \tau_u^A, \quad \text{za svako } u \in L,
\end{aligned}$$

čija se rešenja nazivaju ***L-relaksirane slabe forward bisimulacije*** za automate $\mathcal{A}$ i $\mathcal{B}$.

Konačno, za slabe backward bisimulacije razmatraćemo sistem

$$\begin{aligned}
&\text{(1-wbb)} \quad \tau^A \leqslant U \otimes \tau^B, \quad \tau^B \leqslant U^\top \otimes \tau^A,\\
&\text{(2_L-wbb)} \quad \sigma_u^A \otimes U \leqslant \sigma_u^B, \quad \sigma_u^B \otimes U^\top \leqslant \sigma_u^A, \quad \text{za svako } u \in L,
\end{aligned}$$

čija se rešenja nazivaju ***L-relaksirane slabe backward bisimulacije*** između $\mathcal{A}$ i $\mathcal{B}$.

Radi jednostavnijeg zapisa, za svaki $\xi \in \{\text{wfs}, \text{wbs}, \text{wfb}, \text{wbb}\}$, rešenja sistema sastavljenog od uslova (1-ξ) i (2_L-ξ) nazivaćemo L-relaksiranim slabim simulacijama, odnosno bisimulacijama, tipa ξ. Analogno, rešenja sistema (2_L-ξ) nazivaćemo L-relaksiranim slabim presimulacijama, odnosno prebisimulacijama, tipa ξ.

Neka su $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ max-plus automati nad poluprstenom $\mathbb{R}_\infty$ i neka je $L \subseteq X^*$. Relaciju $\leqslant_L$ između njihovih ponašanja definišemo

$$[\![\mathcal{A}]\!] \leqslant_L [\![\mathcal{B}]\!] \quad \Longleftrightarrow \quad [\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u), \quad \text{za svako } u \in L.$$

Ukoliko važi $[\![\mathcal{A}]\!] \leqslant_L [\![\mathcal{B}]\!]$, kažemo da je ponašanje automata $\mathcal{A}$ *L-inkluzivno* u ponašanju automata $\mathcal{B}$ (ponašanje automata $\mathcal{A}$ je L-sadržano u ponašanju automata $\mathcal{B}$). Relaciju $\leqslant_L$ nazivamo ***L-inkluzijom***. Relaciju $\leqslant_L$ možemo posmatrati i kao relaciju između automata, pa ćemo pisati $\mathcal{A} \leqslant_L \mathcal{B}$ ako i samo ako važi $[\![\mathcal{A}]\!] \leqslant_L [\![\mathcal{B}]\!]$.

Na analogan način definišemo relaciju $\equiv_L$ sa

$$[\![\mathcal{A}]\!] \equiv_L [\![\mathcal{B}]\!] \quad \Longleftrightarrow \quad [\![\mathcal{A}]\!](u) = [\![\mathcal{B}]\!](u), \quad \text{za svako } u \in L.$$

Relaciju $\equiv_L$ nazivamo ***L-ekvivalencijom***. Ukoliko važi $[\![\mathcal{A}]\!] \equiv_L [\![\mathcal{B}]\!]$, kažemo da su automati $\mathcal{A}$ i $\mathcal{B}$ *L-ekvivalentni* i pišemo $\mathcal{A} \equiv_L \mathcal{B}$.

Za svaki $\xi \in \{\text{wfs}, \text{wbs}, \text{wfb}, \text{wbb}\}$ definišimo matricu $U_L^\xi \in \mathbb{R}_\infty^{m\times n}$ na sledeći način:

$$U_L^{\text{wfs}} = \bigwedge_{u\in L} \tau_u^A \backslash \tau_u^B, \qquad U_L^{\text{wbs}} = \bigwedge_{u\in L} \sigma_u^A \backslash \sigma_u^B, \tag{60}$$

$$U_L^{\text{wfb}} = \bigwedge_{u\in L} \tau_u^A \oblong \tau_u^B, \qquad U_L^{\text{wbb}} = \bigwedge_{u\in L} \sigma_u^A \oblong \sigma_u^B. \tag{61}$$

Kada je $L = X^*$, umesto oznake $U_{X^*}^\xi$ koristićemo kraću oznaku U^ξ.

U narednoj teoremi pokazujemo da postojanje L-relaksirane slabe simulacije predstavlja dovoljan uslov za L-inkluziju, dok postojanje L-relaksirane slabe bisimulacije predstavlja dovoljan uslov za L-ekvivalenciju.

Teorema 5.2 *Neka su* $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ *i* $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ *max-plus automati nad* $\mathbb{R}_\infty$ *i neka je* $L \subseteq X^*$. *Tada važi:*

(a) *Ako postoji L-relaksirana slaba simulacija bilo kog tipa između automata* $\mathcal{A}$ *i* $\mathcal{B}$, *tada važi* $[\![\mathcal{A}]\!] \leqslant_L [\![\mathcal{B}]\!]$.

(b) *Ako postoji L-relaksirana slaba bisimulacija bilo kog tipa između automata* $\mathcal{A}$ *i* $\mathcal{B}$, *tada važi* $[\![\mathcal{A}]\!] \equiv_L [\![\mathcal{B}]\!]$.

Dokaz 5.2 *Dokazaćemo tvrđenje pod* (a) *za slučaj L-relaksirane slabe forward simulacije. Neka je U jedna takva simulacija. Za proizvoljno* $u \in L$, *na osnovu 37 i uslova* (1-wfs), (2_L-wfs) *dobijamo*

$$[\![\mathcal{A}]\!](u) = \sigma^A \otimes \tau_u^A \leqslant \sigma^B \otimes U^\top \otimes \tau_u^A \leqslant \sigma^B \otimes \tau_u^B = [\![\mathcal{B}]\!](u).$$

Kako je $u \in L$ *proizvoljno, sledi* $[\![\mathcal{A}]\!] \leqslant_L [\![\mathcal{B}]\!]$.

Dokazi za slučaj L-relaksirane slabe backward simulacije, kao i za forward i backward L-relaksirane slabe bisimulacije, izvode se na analogan način. □

Prethodna teorema pokazuje da postojanje L-relaksirane slabe simulacije, odnosno bisimulacije, predstavlja dovoljan uslov za L-inkluziju, odnosno L-ekvivalenciju ponašanja automata. U nastavku ćemo pokazati da se najveće L-relaksirane presimulacije i prebisimulacije mogu direktno karakterisati pomoću matrica U_L^ξ definisanih preko odgovarajućih reziduala. Na taj način dobijamo i kriterijum za postojanje najvećih L-relaksiranih slabih simulacija i bisimulacija.

Teorema 5.3 *Neka su* $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ *i* $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ *max-plus automati nad* $\mathbb{R}_\infty$, *neka je* $L \subseteq X^*$ *i neka je* $\xi \in \{\text{wfs}, \text{wbs}\}$ $\big(\xi \in \{\text{wfb}, \text{wbb}\}\big)$. *Tada važi:*

(a) U_L^ξ *je najveća L-relaksirana slaba presimulacija (prebisimulacija) između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *tipa* ξ.

(b) *Ako je* U_L^ξ *ne nula matrica i zadovoljava uslov* (1-ξ), *tada je* U_L^ξ *najveća L-relaksirana slaba simulacija (bisimulacija) između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *tipa* ξ.

(c) *Ako je* U_L^ξ *nula-matrica ili ne zadovoljava uslov* (1-ξ), *tada između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *ne postoji L-relaksirana slaba simulacija (bisimulacija) tipa* ξ.

Dokaz 5.3 *Dokazaćemo tvrđenje za slučaj* $\xi = \text{wfs}$. *Dokaz za slučaj* $\xi = \text{wbs}$, *kao i za slučajeve* $\xi \in \{\text{wfb}, \text{wbb}\}$, *izvodi se na analogan način.*

(a) *Prema definiciji matrice* U_L^{wfs}, *za proizvoljno* $u \in L$ *važi* $U_L^{\text{wfs}} \leqslant \tau_u^A \backslash \tau_u^B$.

Otuda sledi

$$\left(U_L^{\text{wfs}}\right)^\top \leqslant \left(\tau_u^A \backslash \tau_u^B\right)^\top = \tau_u^B \not/ \tau_u^A.$$

Na osnovu svojstva reziduala za $\not/$ *dobijamo da* U_L^{wfs} *predstavlja rešenje sistema* (2_L-wfs),

$$\left(U_L^{\text{wfs}}\right)^\top \otimes \tau_u^A \leqslant \tau_u^B.$$

Neka je $U \in \mathbb{R}_\infty^{m\times n}$ *proizvoljno rešenje sistema* (2_L-wfs). *Tada za svako* $u \in L$ *važi*

$$U^\top \otimes \tau_u^A \leqslant \tau_u^B.$$

Ponovnom primenom svojstva reziduala za $\not/$ *dobijamo*

$$U^\top \leqslant \tau_u^B \not/ \tau_u^A, \quad \textit{odnosno } U \leqslant \left(\tau_u^B \not/ \tau_u^A\right)^\top = \tau_u^A \backslash \tau_u^B.$$

Kako ovo važi za svako $u \in L$, *dobijamo da je* U_L^{wfs} *najveće rešenje sistema* (2_L-wfs)*:*

$$U \leqslant \bigwedge_{u\in L} \tau_u^A \backslash \tau_u^B = U_L^{\text{wfs}}.$$

(b) *Pretpostavimo da je* U_L^{wfs} *matrica različita od nula-matrice i da zadovoljava uslov* (1-wfs). *Prema delu (a),* U_L^{wfs} *je rešenje sistema* (2_L-wfs), *pa zajedno sa uslovom* (1-wfs) *predstavlja L-relaksiranu slabu forward simulaciju između automata* $\mathcal{A}$ *i* $\mathcal{B}$.

Sa druge strane, svaka L-relaksirana slaba forward simulacija predstavlja rešenje sistema (2_L-wfs). *Kako je* U_L^{wfs} *najveće rešenje ovog sistema, sledi da je* U_L^{wfs} *najveća L-relaksirana slaba forward simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$.

(c) *Pretpostavimo da je* U_L^{wfs} *nula-matrica ili da ne zadovoljava uslov* (1-wfs). *Pretpostavimo, suprotno, da postoji L-relaksirana slaba forward simulacija U između automata* $\mathcal{A}$ *i* $\mathcal{B}$. *Prema delu (a),* U_L^{wfs} *je najveće rešenje sistema* (2_L-wfs), *pa važi* $U \leqslant U_L^{\text{wfs}}$.

Kako U zadovoljava uslov (1-wfs), *biće*

$$\sigma^A \leqslant \sigma^B \otimes U^\top \leqslant \sigma^B \otimes \left(U_L^{\text{wfs}}\right)^\top.$$

Ukoliko je U_L^{wfs} *nula-matrica, poslednja nejednakost nije moguća, dok u slučaju kada* U_L^{wfs} *ne zadovoljava uslov* (1-wfs) *dobijamo kontradikciju sa pretpostavkom da je*

$$\sigma^A \leqslant \sigma^B \otimes \left(U_L^{\text{wfs}}\right)^\top.$$

Dakle, ni u jednom od posmatranih slučajeva ne postoji L-relaksirana slaba forward simulacija između automata $\mathcal{A}$ *i* $\mathcal{B}$. □

Prethodno definisani uslovi za slabe simulacije i bisimulacije zavise od izbora jezika $L \subseteq X^*$. Zbog toga je prirodno ispitati na koji način promena jezika utiče na njihovu zadovoljenost. Sledeća teorema pokazuje da se zadovoljenost uslova (1-ξ) prenosi sa jezika na njegove podjezike, kao i da se odgovarajući uslov na jeziku X^* može karakterisati preko

uslova definisanih na njegovim podskupovima.

Teorema 5.4 *Neka su* $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ *i* $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ *max-plus automati nad* $\mathbb{R}_\infty$*, neka je* $L \subseteq X^*$ *i neka je* $\xi \in \{wfs, wbs, wfb, wbb\}$*. Tada važe sledeća svojstva:*

(a) *Ako* U_L^ξ *zadovoljava uslov* (1-ξ) *za neki* $L \subseteq X^*$*, tada* U_K^ξ *zadovoljava uslov* (1-ξ) *za svaki* $K \subseteq L$*;*

(b) U^ξ *zadovoljava uslov* (1-ξ) *ako i samo ako* U_L^ξ *zadovoljava uslov* (1-ξ) *za svaki* $L \subseteq X^*$*.*

Dokaz 5.4 *Teoremu ćemo dokazati za slučaj* $\xi = wfs$*. Dokazi u ostalim slučajevima su analogni.*

(a) Neka U_L^{wfs} zadovoljava uslov (1-wfs) i neka je $K \subseteq L$. Tada važi

$$U_L^{\mathrm{wfs}} = \bigwedge_{u\in L} \tau_u^A \backslash \tau_u^B \leqslant \bigwedge_{u\in K} \tau_u^A \backslash \tau_u^B = U_K^{\mathrm{wfs}}, \text{ odakle sledi}$$

$$\sigma^A \leqslant \sigma^B \otimes (U_L^{\mathrm{wfs}})^\top \leqslant \sigma^B \otimes (U_K^{\mathrm{wfs}})^\top.$$

Prema tome, U_K^{wfs} takođe zadovoljava uslov (1-wfs).

(b) Prema delu (a), ako U^{wfs} zadovoljava uslov (1-wfs), tada U_L^{wfs} zadovoljava uslov (1-wfs) za svaki $L \subseteq X^*$. Obratno tvrđenje je očigledno. □

Teoreme dokazne u ovoj podsekciji pokazuju da se relaksirane slabe simulacije i bisimulacije mogu posmatrati kao prirodno ograničenje odgovarajućih uslova na proizvoljan skup reči $L \subseteq X^*$. U nastavku ćemo razmotriti jednu značajnu klasu takvih relaksacija, dobijenu ograničavanjem dužine posmatranih reči.

5.3 Relaksacije ograničene dužinom reči

U ovoj sekciji razmatraćemo posebnu klasu L-relaksiranih slabih simulacija i bisimulacija, koja se zasniva na ograničavanju dužine posmatranih reči. Preciznije, za dato $k \in \mathbb{N}_0$, neka je $L_k = \{u \in X^* \mid |u| \leqslant k\}$ skup svih reči nad alfabetom X čija dužina nije veća od k.

Za $\xi \in \{\mathrm{wfs}, \mathrm{wbs}, \mathrm{wfb}, \mathrm{wbb}\}$, L_k-relaksiranu slabu simulaciju, odnosno bisimulaciju, tipa ξ nazivaćemo ***slabom k-simulacijom***, odnosno ***slabom k-bisimulacijom***, tipa ξ.Sistem (2_{L_k}-ξ) kraće ćemo označavati sa (2_k-ξ), a njegova rešenja nazivaćemo *slabim k-presimulacijama*, odnosno *slabim k-prebisimulacijama*, tipa ξ.

Relacije L_k-inkluzije i L_k-ekvivalencije nazivaćemo redom ***k-inkluzijom*** i ***k-ekvivalencijom***, a označavaćemo ih sa $\leqslant_k$ i $\equiv_k$.

Specijalizacijom definicija (60) i (61) za slučaj $L = L_k$ dobijamo matrice

$$U_k^{\mathrm{wfs}} = \bigwedge_{|u|\leqslant k} \tau_u^A \backslash \tau_u^B, \qquad U_k^{\mathrm{wbs}} = \bigwedge_{|u|\leqslant k} \sigma_u^A \backslash \sigma_u^B, \tag{62}$$

$$U_k^{\mathrm{wfb}} = \bigwedge_{|u|\leqslant k} \tau_u^A \mathbin{\oblong} \tau_u^B, \qquad U_k^{\mathrm{wbb}} = \bigwedge_{|u|\leqslant k} \sigma_u^A \mathbin{\oblong} \sigma_u^B. \tag{63}$$

Na osnovu Teoreme 5.3 neposredno dobijamo sledeći rezultat.

Teorema 5.5 *Neka su $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ max-plus automati nad $\mathbb{R}_\infty$ i neka je $\xi \in \{\mathrm{wfs}, \mathrm{wbs}\}$ $\big(\xi \in \{\mathrm{wfb}, \mathrm{wbb}\}\big)$. Tada važi:*

(a) *U_k^ξ je najveća slaba k-presimulacija (k-prebisimulacija) između automata $\mathcal{A}$ i $\mathcal{B}$ tipa ξ.*

(b) *Ako je U_k^ξ matrica različita od nula-matrice i zadovoljava uslov* (1-ξ)*, tada je U_k^ξ najveća slaba k-simulacija (k-bisimulacija) između automata $\mathcal{A}$ i $\mathcal{B}$ tipa ξ.*

(c) *Ako je U_k^ξ nula-matrica ili ne zadovoljava uslov* (1-ξ)*, tada između automata $\mathcal{A}$ i $\mathcal{B}$ ne postoji slaba k-simulacija (k-bisimulacija) tipa ξ.*

Dokaz 5.5 *Dokaz neposredno sledi iz Teoreme 5.3 izborom $L = L_k$.* □

Relaksacije zasnovane na ograničavanju dužine reči predstavljaju prirodan način redukovanja uslova koji definišu slabe simulacije i bisimulacije. Za dato $k \in \mathbb{N}_0$, razmatranjem skupa $L_k = \{u \in X^* \mid |u| \leqslant k\}$ dobija se konačan skup relevantnih σ-parova, odnosno τ-parova, čime se beskonačan sistem uslova zamenjuje konačnim sistemom. Na osnovu Teoreme 5.5 moguće je na taj način odrediti najveću slabu k-simulaciju, odnosno slabu k-bisimulaciju odgovarajućeg tipa, ukoliko ona postoji.

Ograničavanje dužine reči omogućava i neposredno razmatranje k-inkluzije i k-ekvivalencije ponašanja max-plus automata. Naime, umesto poređenja ponašanja automata za sve reči iz skupa X^*, može se ispitivati njihov odnos samo za reči čija je dužina najviše k. Na taj način dobija se konačan problem koji predstavlja prirodnu aproksimaciju problema inkluzije, odnosno ekvivalencije ponašanja automata na celom skupu reči.

Ovakav pristup je posebno značajan kada se u konkretnoj primeni razmatraju samo ulazne sekvence ograničene dužine. Istovremeno, ograničavanje dužine reči predstavlja samo jedan od mogućih načina redukovanja beskonačnog skupa uslova koji se javljaju kod slabih simulacija i bisimulacija. U nastavku ćemo razmotriti drugačiji pristup, zasnovan na ograničavanju datuma realizacije reči.

Primer 5.1 *Neka su dati automati* $\mathcal{A} = (2, \sigma^A, \{M_x^A, M_y^A\}, \tau^A)$ *i* $\mathcal{B} = (3, \sigma^B, \{M_x^B, M_y^B\}, \tau^B)$ *nad max-plus poluprstenom* $\mathbb{P}_\infty$ *i alfabetom* $X = \{x, y\}$*, zadati sa*

$$M_x^A = \begin{bmatrix} 4 & 1 \\ -\infty & 2 \end{bmatrix}, \qquad M_y^A = \begin{bmatrix} -\infty & -\infty \\ -\infty & 1 \end{bmatrix}, \qquad \sigma^A = \begin{bmatrix} 1 & -\infty \end{bmatrix}, \qquad \tau^A = \begin{bmatrix} -\infty \\ 0 \end{bmatrix},$$

$$M_x^B = \begin{bmatrix} -\infty & 4 & -\infty \\ -\infty & 4 & -\infty \\ -\infty & -\infty & 2 \end{bmatrix}, \quad M_y^B = \begin{bmatrix} -\infty & -\infty & 0 \\ -\infty & -\infty & 2 \\ -\infty & -\infty & 1 \end{bmatrix}, \quad \sigma^B = \begin{bmatrix} 1 & -\infty & -\infty \end{bmatrix}, \quad \tau^B = \begin{bmatrix} 1 \\ 3 \\ 0 \end{bmatrix}.$$

Jasno je da se automati $\mathcal{A}$ *i* $\mathcal{B}$ *mogu posmatrati istovremeno kao automati nad* $\mathbb{R}_\infty$ *i kao automati nad* $\mathbb{P}_\infty$*. Kada je reč o računanju* τ*-parova, nema razlike između rada nad* $\mathbb{R}_\infty$ *i* $\mathbb{P}_\infty$*, budući da se koriste samo operacije sabiranja i množenja, a obe operacije nad* $\mathbb{P}_\infty$ *predstavljaju restrikcije odgovarajućih operacija nad* $\mathbb{R}_\infty$*. Razlika se, međutim, javlja pri računanju reziduala* τ*-parova, jer se operacije reziduacije nad* $\mathbb{R}_\infty$ *i* $\mathbb{P}_\infty$ *bitno razlikuju. U nastavku ćemo pokazati da je u ovom primeru povoljnije računanje sprovesti nad* $\mathbb{P}_\infty$*, pa ćemo najpre razmotriti taj slučaj.*

(a) Određivanje postojanja i izračunavanje najveće slabe forward simulacije nad $\mathbb{P}_\infty$**.**
Oba automata imaju beskonačno mnogo τ*-vektora, pa samim tim postoji i beskonačno mnogo* τ*-parova. Ovi parovi, zajedno sa odgovarajućim rezidualima, prikazani su u Tabeli 7. Formule kojima su predstavljeni* τ*-parovi* (τ_u^A, τ_u^B) *mogu se dokazati indukcijom po složenosti reči u. Podsetimo da* $|u|_x$ *i* $|u|_y$ *označavaju redom broj pojavljivanja slova x i y u reči u.*

Prema tome, najveća slaba forward presimulacija između automata $\mathcal{A}$ *i* $\mathcal{B}$ *dobija se:*

$$U = U^{\mathrm{wfs}} = \bigwedge_{u \in X^*} \tau_u^A \backslash \tau_u^B = \begin{bmatrix} +\infty & +\infty & +\infty \\ 1 & 3 & 0 \end{bmatrix} \wedge \begin{bmatrix} +\infty & +\infty & +\infty \\ -\infty & 1 & 0 \end{bmatrix} \wedge \begin{bmatrix} 4 & 4 & 1 \\ 3 & 3 & 0 \end{bmatrix} \wedge \begin{bmatrix} 6 & 6 & 1 \\ 5 & 5 & 0 \end{bmatrix} \wedge$$

$$\wedge \left(\bigwedge_{s \geqslant 2} \begin{bmatrix} 4 & 4 & -\infty \\ 2s+1 & 2s+1 & 0 \end{bmatrix} \right) \wedge \left(\bigwedge_{s \geqslant 2} \begin{bmatrix} 6 & 6 & -\infty \\ 2s+3 & 2s+3 & 0 \end{bmatrix} \right) =$$

$$= \begin{bmatrix} +\infty & +\infty & +\infty \\ 1 & 3 & 0 \end{bmatrix} \wedge \begin{bmatrix} +\infty & +\infty & +\infty \\ -\infty & 1 & 0 \end{bmatrix} \wedge \begin{bmatrix} 4 & 4 & 1 \\ 3 & 3 & 0 \end{bmatrix} \wedge \begin{bmatrix} 6 & 6 & 1 \\ 5 & 5 & 0 \end{bmatrix} \wedge \begin{bmatrix} 4 & 4 & -\infty \\ 5 & 5 & 0 \end{bmatrix} \wedge \begin{bmatrix} 6 & 6 & -\infty \\ 7 & 7 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 4 & 4 & -\infty \\ -\infty & 1 & 0 \end{bmatrix}.$$

Zaključujemo da je U *najveća slaba forward simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$*, budući da je:*

$$\sigma^B \otimes U^\top = \begin{bmatrix} 1 & -\infty & -\infty \end{bmatrix} \otimes \begin{bmatrix} 4 & -\infty \\ 4 & 1 \\ -\infty & 0 \end{bmatrix} = \begin{bmatrix} 5 & -\infty \end{bmatrix} > \begin{bmatrix} 1 & -\infty \end{bmatrix} = \sigma^A.$$

Tabela 7: τ-parovi izračunati u Primeru 5.1 i njihovi reziduali.

Par (τ_u^A, τ_u^B) **τ-vektora**	**Rezidual** $\tau_u^A \backslash \tau_u^B$	
$\left(\begin{bmatrix}-\infty\\0\end{bmatrix}, \begin{bmatrix}1\\3\\0\end{bmatrix}\right)$	$\begin{bmatrix}+\infty & +\infty & +\infty\\1 & 3 & 0\end{bmatrix}$	$u = e$
$\left(\begin{bmatrix}-\infty\\2\lvert u\rvert_x + \lvert u\rvert_y\end{bmatrix}, \begin{bmatrix}2\lvert u\rvert_x + \lvert u\rvert_y - 1\\2\lvert u\rvert_x + \lvert u\rvert_y + 1\\2\lvert u\rvert_x + \lvert u\rvert_y\end{bmatrix}\right)$	$\begin{bmatrix}+\infty & +\infty & +\infty\\-\infty & 1 & 0\end{bmatrix}$	$u = yv,\ v \in X^*$
$\left(\begin{bmatrix}2\lvert u\rvert_x + \lvert u\rvert_y - 1\\2\lvert u\rvert_x + \lvert u\rvert_y\end{bmatrix}, \begin{bmatrix}2\lvert u\rvert_x + \lvert u\rvert_y + 3\\2\lvert u\rvert_x + \lvert u\rvert_y + 3\\2\lvert u\rvert_x + \lvert u\rvert_y\end{bmatrix}\right)$	$\begin{bmatrix}4 & 4 & 1\\3 & 3 & 0\end{bmatrix}$	$u = xyv,\ v \in X^*$
$\left(\begin{bmatrix}2\lvert u\rvert_x + \lvert u\rvert_y + 2s - 3\\2\lvert u\rvert_x + \lvert u\rvert_y\end{bmatrix}, \begin{bmatrix}2\lvert u\rvert_x + \lvert u\rvert_y + 2s + 1\\2\lvert u\rvert_x + \lvert u\rvert_y + 2s + 1\\2\lvert u\rvert_x + \lvert u\rvert_y\end{bmatrix}\right)$	$\begin{bmatrix}4 & 4 & -\infty\\2s+1 & 2s+1 & 0\end{bmatrix}$	$u = x^s yv,\ s \geqslant 2,\ v \in X^*$
$\left(\begin{bmatrix}1\\2\end{bmatrix}, \begin{bmatrix}7\\7\\2\end{bmatrix}\right)$	$\begin{bmatrix}6 & 6 & 1\\5 & 5 & 0\end{bmatrix}$	$u = x$
$\left(\begin{bmatrix}4s-3\\2s\end{bmatrix}, \begin{bmatrix}4s+3\\4s+3\\2s\end{bmatrix}\right)$	$\begin{bmatrix}6 & 6 & -\infty\\2s+3 & 2s+3 & 0\end{bmatrix}$	$u = x^s,\ s \geqslant 2$

Primetićemo da se, pri primeni Procedure 2, infimum dostiže već nakon šest koraka, odnosno nakon generisanja šest τ-parova koji odgovaraju rečima e, x, y, x^2, yx i xy. Međutim, postupak se nastavlja, jer u okviru postojeće procedure ne postoji način da se u ovom trenutku detektuje da je infimum već dostignut, na osnovu čega bi se postupak mogao zaustaviti. Ovo pitanje ostaje otvoreno za dalja istraživanja.

(b) Ispitivanje postojanja inkluzije i ekvivalencije nad $\mathbb{P}_\infty$**.**
U delu (a) ovog primera videli smo da, iako je skup svih τ-parova beskonačan, najveću slabu forward simulaciju ipak možemo odrediti, mada ne pomoću Procedure 2, već teorijskim pristupom, korišćenjem indukcije po reči u. Na osnovu formula kojima su predstavljeni τ-parovi, možemo pokazati i da između max-plus automata $\mathcal{A}$ i $\mathcal{B}$ postoji inkluzija:

$$[\![\mathcal{A}]\!](u) = \sigma^A \otimes \tau_u^A = \begin{cases} -\infty & \text{ako je } u = e,\\ -\infty & \text{ako je } u = yv,\ v \in X^*,\\ 2|u|_x + |u|_y + 2s - 3 & \text{ako je } u = x^s yv,\ s \geqslant 1,\ v \in X^*,\\ 4s - 3 & \text{ako je } u = x^s,\ s \geqslant 1 \end{cases}$$

$$\leqslant \begin{cases} 1 & \text{ako je } u = e,\\ 2|u|_x + |u|_y - 1 & \text{ako je } u = yv,\ v \in X^*,\\ 2|u|_x + |u|_y + 2s + 1 & \text{ako je } u = x^s yv,\ s \geqslant 1,\ v \in X^*,\\ 4s + 3 & \text{ako je } u = x^s,\ s \geqslant 1 \end{cases} = \sigma^B \otimes \tau_u^B = [\![\mathcal{B}]\!](u).$$

Iz prethodnog se takođe vidi da automati $\mathcal{A}$ i $\mathcal{B}$ nisu ekvivalentni.

(c) Određivanje najveće slabe k-forward simulacije nad $\mathbb{P}_\infty$.

Budući da se najveća slaba forward simulacija U^{wfs} između automata $\mathcal{A}$ i $\mathcal{B}$ dobija već nakon određivanja τ-parova koji odgovaraju rečima e, x, y, x^2, yx i xy, kao i njihovih reziduala, sledi da za svako $k \in \mathbb{N}$, $k \geqslant 2$, najveća slaba k-forward simulacija U_k^{wfs} jednaka je sa najvećom slabom forward simulacijom U^{wfs}, odnosno:

$$U_k^{\mathrm{wfs}} = U^{\mathrm{wfs}} = \begin{bmatrix} 4 & 4 & -\infty \\ -\infty & 1 & 0 \end{bmatrix}.$$

S druge strane, za $k = 0$ i $k = 1$ dobijamo

$$U_0^{\mathrm{wfs}} = \tau_e^A \backslash \tau_e^B = \begin{bmatrix} +\infty & +\infty & +\infty \\ 1 & 3 & 0 \end{bmatrix}, \qquad U_1^{\mathrm{wfs}} = \tau_e^A \backslash \tau_e^B \wedge \tau_x^A \backslash \tau_x^B \wedge \tau_y^A \backslash \tau_y^B = \begin{bmatrix} 6 & 6 & 1 \\ -\infty & 1 & 0 \end{bmatrix}.$$

(d) Ispitivanje postojanja i određivanje najveće slabe forward simulacije nad $\mathbb{R}_\infty$.

Kao što smo već napomenuli, kada se proračuni vrše nad $\mathbb{R}_\infty$, dobijaju se isti τ-parovi kao i u slučaju kada se proračuni vrše nad $\mathbb{P}_\infty$. Međutim, razlike se javljaju pri izračunavanju reziduala τ-parova. Naime, reziduali τ-parova izračunati nad $\mathbb{R}_\infty$ dati su sa

$$\tau_u^A \backslash \tau_u^B = \begin{cases} \begin{bmatrix} +\infty & +\infty & +\infty \\ 1 & 3 & 0 \end{bmatrix} & \text{ako je } u = e, \\ \begin{bmatrix} +\infty & +\infty & +\infty \\ -1 & 1 & 0 \end{bmatrix} & \text{ako je } u = yv,\ v \in X^*, \\ \begin{bmatrix} 4 & 4 & -2s+3 \\ 2s+1 & 2s+1 & 0 \end{bmatrix} & \text{ako je } u = x^s yv,\ s \geqslant 1,\ v \in X^*, \\ \begin{bmatrix} 6 & 6 & -2s+3 \\ 2s+3 & 2s+3 & 0 \end{bmatrix} & \text{ako je } u = x^s,\ s \geqslant 1. \end{cases}$$

Prema tome, u ovom slučaju dobijamo

$$U^{\mathrm{wfs}} = \bigwedge_{u \in X^*} \tau_u^A \backslash \tau_u^B = \begin{bmatrix} 4 & 4 & -\infty \\ -1 & 1 & 0 \end{bmatrix}.$$

Dakle, najveća slaba forward simulacija izračunata nad $\mathbb{R}_\infty$ razlikuje se samo neznatno od one izračunate nad $\mathbb{P}_\infty$. Međutim, značajna razlika javlja se u njihovom određivanju primenom Procedure 2. Naime, u delu (a) videli smo da se infimum dostiže već nakon šest koraka, što se pokazalo značajnim za delove (c) i (d). Nasuprot tome, za prethodno navedene matrice, kada se proračuni vrše nad $\mathbb{R}_\infty$, infimum se ne može dostići nakon konačno mnogo koraka. Razlog za to je činjenica da se primenom reziduala nad $\mathbb{P}_\infty$ izbegavaju beskonačne sekvence vrednosti koje konvergiraju ka $-\infty$.

(e) Ispitivanje postojanja inkluzije i ekvivalencije nad $\mathbb{R}_\infty$***.***
Budući da se u ovom slučaju koriste samo operacije sabiranja i množenja, dobijaju se isti rezultati kao u delu (b).

(f) Određivanje najveće slabe k***-forward simulacije nad*** $\mathbb{R}_\infty$***.***
Kada se proračuni vrše nad $\mathbb{R}_\infty$*, dobijamo*

$$U_0^{\text{wfs}} = \tau_\varepsilon^A \backslash \tau_\varepsilon^B = \begin{bmatrix} +\infty & +\infty & +\infty \\ 1 & 3 & 0 \end{bmatrix},$$

a za proizvoljno $k \in \mathbb{N}$ *dobijamo*

$$K_1 = \{u \in X^* \mid h(u) = y,\ |u| \leqslant k\},$$
$$K_2 = \{u \in X^* \mid u = x^s yv,\ s \geqslant 1,\ v \in X^*,\ |u| \leqslant k\},$$
$$K_3 = \{u \in X^* \mid u = x^s,\ s \geqslant 1,\ |u| \leqslant k\},$$

$$U_k^{\text{wfs}} = \bigwedge_{|u| \leqslant k} \tau_u^A \backslash \tau_u^B = U_0^{\text{wfs}} \wedge \left(\bigwedge_{u \in K_1} \tau_u^A \backslash \tau_u^B \right) \wedge \left(\bigwedge_{u \in K_2} \tau_u^A \backslash \tau_u^B \right) \wedge \left(\bigwedge_{u \in K_3} \tau_u^A \backslash \tau_u^B \right) =$$
$$= \begin{bmatrix} +\infty & +\infty & +\infty \\ 1 & 3 & 0 \end{bmatrix} \wedge \begin{bmatrix} +\infty & +\infty & +\infty \\ -1 & 1 & 0 \end{bmatrix} \wedge \begin{bmatrix} 4 & 4 & -2k+5 \\ 3 & 3 & 0 \end{bmatrix} \wedge \begin{bmatrix} 6 & 6 & -2k+3 \\ 5 & 5 & 0 \end{bmatrix} =$$
$$= \begin{bmatrix} 4 & 4 & -2k+3 \\ -1 & 1 & 0 \end{bmatrix}.$$

5.4 Δ-odsečeni max-plus poluprsten. Gornje odsecanje poluprstena $\mathbb{P}_\infty$

U prethodnoj sekciji razmatrali smo relaksacije zasnovane na ograničavanju dužine posmatranih reči. U nastavku ćemo razmotriti relaksacije dobijene ograničavanjem datuma realizacije sekvenci događaja, koje ćemo nazivati *relaksacijama ograničenim datumom*. U tu svrhu biće nam potrebna sledeća modifikacija kompletnog max-plus poluprstena $\mathbb{P}_\infty$, koju ćemo nazivati ***gornjim odsecanjem*** ovog poluprstena.

Teorema 5.6 *Za dato* $\Delta \in \mathbb{R}$*,* $\Delta > 0$*, neka je* $\mathbb{P}_\Delta = [0, \Delta) \cup \{-\infty, +\infty\}$*, neka je* $\oplus_\Delta$ *restrikcija operacije* $\oplus$ *sa* $\mathbb{P}_\infty$ *na skup* $\mathbb{P}_\Delta$*, a* $\otimes_\Delta$ *binarna operacija na* $\mathbb{P}_\Delta$ *definisana sledećom Cayley-evom tablicom:*

	$b \in [0, \Delta)$			
$\otimes_\Delta$	$a \otimes b < \Delta$	$a \otimes b \geqslant \Delta$	$-\infty$	$+\infty$
$a \in [0, \Delta)$	$a \otimes b$	$+\infty$	$-\infty$	$+\infty$
$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
$+\infty$	$+\infty$	$+\infty$	$-\infty$	$+\infty$

Tada važe sledeća tvrđenja:

(a) *Petorka* $\mathbb{P}_\Delta = (\mathbb{P}_\Delta, \oplus_\Delta, \otimes_\Delta, -\infty, 0)$ *čini komutativan kompletan aditivno-idempotentan poluprsten.*

(b) *Operacija reziduala* $\backslash_\Delta$ *na* $\mathbb{P}_\Delta$ *predstavlja restrikciju operacije reziduala* $\backslash$ *na* $\mathbb{P}_\infty$.

(c) $\mathbb{P}_\Delta$ *je lokalno konačan poluprsten.*

Dokaz 5.6 (a) *Najpre uočimo da je* $(\mathbb{P}_\Delta, \oplus_\Delta)$ *polugrupa, jer je* $\oplus_\Delta$ *restrikcija operacije* $\oplus$ *na skup* $\mathbb{P}_\Delta$*, a element* $-\infty$ *predstavlja neutralni element za ovu operaciju.*

Posmatrajmo skup $\mathbb{P}'_\Delta = [0, \Delta) \cup \{+\infty\}$ *sa operacijom* $\otimes'_\Delta$*, dobijenom restrikcijom operacije* $\otimes_\Delta$ *na* $\mathbb{P}'_\Delta$*. Tada je* $(\mathbb{P}'_\Delta, \otimes'_\Delta)$ *Reesov faktor polugrupe* $(\mathbb{P}'_\infty, \otimes')$*, gde je* $\mathbb{P}'_\infty = \mathbb{R}^+ \cup \{+\infty\} = \mathbb{P}_\infty \setminus \{-\infty\}$*, po idealu* $\mathbb{I}_\Delta = [\Delta, +\infty]$*. Otuda sledi da je* $(\mathbb{P}'_\Delta, \otimes'_\Delta)$ *polugrupa. Kako se* $(\mathbb{P}_\Delta, \otimes_\Delta)$ *dobija njenim proširenjem nulom* $-\infty$*, zaključujemo da je i* $(\mathbb{P}_\Delta, \otimes_\Delta)$ *polugrupa. Pri tome je* 0 *neutralni element za operaciju* $\otimes_\Delta$*.*

Neposrednom proverom utvrđuje se da važe distributivni zakoni i da je $-\infty$ *apsorbujući element za množenje. Prema tome,* $\mathbb{P}_\Delta = (\mathbb{P}_\Delta, \oplus_\Delta, \otimes_\Delta, -\infty, 0)$ *predstavlja poluprsten. Pošto je* $\oplus_\Delta$ *restrikcija operacije* $\oplus = \max$*, sledi da je* $\mathbb{P}_\Delta$ *kompletan aditivno-idempotentan komutativan poluprsten.*

(b) *Neka su* $a, b \in \mathbb{P}_\Delta$*.*

Ako je $a = -\infty$ *ili* $b = +\infty$*, tada je svaki element skupa* $\mathbb{P}_\Delta$ *rešenje nejednačine* $a \otimes_\Delta x \leqslant b$*, pa važi*

$$a \backslash_\Delta b = +\infty = a \backslash b. \tag{64}$$

Pretpostavimo sada da je $a \neq -\infty$ *i* $b \neq +\infty$*.*

Ako je $b = -\infty$*, tada je jedino rešenje nejednačine* $a \otimes_\Delta x \leqslant b$ *element* $-\infty$*, pa dobijamo*

$$a \backslash_\Delta b = -\infty = a \backslash b. \tag{65}$$

Ako je $a = +\infty$*, tada za svako* $x \in \mathbb{P}_\Delta$ *važi*

$$+\infty \otimes_\Delta x = \begin{cases} -\infty, & x = -\infty, \\ +\infty, & x \in [0, \Delta) \cup \{+\infty\}, \end{cases}$$

pa je za svako $b \in [0, \Delta) \cup \{-\infty\}$ *ponovo jedino rešenje nejednačine* $a \otimes_\Delta x \leqslant b$ *element* $-\infty$*. Dakle, i u ovom slučaju važi* (65)*.*

Preostaje slučaj $a, b \in [0, \Delta)$*. Uočimo da je* $-\infty$ *rešenje i nejednačine* $a \otimes_\Delta x \leqslant b$ *u poluprstenu* $\mathbb{P}_\Delta$ *i nejednačine* $a \otimes x \leqslant b$ *u poluprstenu* $\mathbb{P}_\infty$*, dok* $+\infty$ *nije rešenje nijedne od njih.*

Dalje, za svako $c \in [0, \Delta)$ *važi:*

$$a \otimes c \leqslant b \iff a \otimes_\Delta c \leqslant b,$$

jer je $b < \Delta$ *i u tom slučaju* $a \otimes_\Delta c = a \otimes c$.

Prema tome, obe nejednačine imaju isti skup rešenja, pa imaju i isto najveće rešenje. Dakle, $a \backslash_\Delta b = a \backslash b$.

Time je dokazano da je $\backslash_\Delta$ *restrikcija operacije* $\backslash$ *na skup* $\mathbb{P}_\Delta$.

(c) *Poznato je da je poluprsten lokalno konačan ako i samo ako su njegovi aditivni i multiplikativni redukti lokalno konačni monoidi.*

Aditivna struktura $(\mathbb{P}_\Delta, \oplus_\Delta)$ *lokalno je konačna, jer predstavlja komutativnu idempotentnu polugrupu.*

Zato je dovoljno pokazati da je i multiplikativna struktura $(\mathbb{P}_\Delta, \otimes_\Delta)$ *lokalno konačna.*

Neka je S *proizvoljan konačan podskup skupa* $\mathbb{P}_\Delta$. *Polugrupa generisana skupom* S *ima noseći skup*

$$\bigcup_{k\in\mathbb{N}} S^k, \quad gde\ je \quad S^k = \{a_1 \otimes_\Delta \cdots \otimes_\Delta a_k \mid a_1, \ldots, a_k \in S\}.$$

Bez umanjenja opštosti možemo pretpostaviti da $\{-\infty, 0\} \notin S$, *jer njihovo prisustvo ne utiče na konačnost generisane polugrupe.*

Kako je S *konačan skup, postoji njegov najmanji element* a. *Po Arhimedovom aksiomu postoji* $k_0 \in \mathbb{N}$ *takav da je* $k_0 a \geqslant \Delta$.

Tada za svako $k \geqslant k_0$ *i proizvoljne elemente* $a_1, \ldots, a_k \in S$ *važi*

$$a_1 \otimes \cdots \otimes a_k \ \geqslant\ ka \ \geqslant\ k_0 a \ \geqslant\ \Delta.$$

Otuda sledi, $a_1 \otimes_\Delta \cdots \otimes_\Delta a_k = +\infty$.

Dakle,

$$\bigcup_{k\in\mathbb{N}} S^k = \bigcup_{k\leqslant k_0} S^k \cup \{+\infty\}.$$

Pošto je svako S^k *konačno, zaključujemo da je i generisana polugrupa konačna. Prema tome,* $(\mathbb{P}_\Delta, \otimes_\Delta)$ *je lokalno konačna polugrupa.*

Sledi da je i poluprsten $\mathbb{P}_\Delta$ *lokalno konačan.* □

Neka je parametar $\Delta \in \mathbb{R}$, $\Delta > 0$. Poluprsten $\mathbb{P}_\Delta$ definisan Teoremom 5.6 nazivaćemo Δ***-odsecanjem*** poluprstena $\mathbb{P}_\infty$. Takođe ćemo reći da je $\mathbb{P}_\Delta$ Δ***-odsečeni poluprsten***.

U skladu sa prethodnom definicijom, svakoj matrici $M \in \mathbb{P}_\infty^{m\times n}$ pridružujemo matricu $M_\Delta \in \mathbb{P}_\Delta^{m\times n}$ definisanu sa

$$M_\Delta(i,j) = \begin{cases} M(i,j), & \text{ako je } M(i,j) < \Delta, \\ +\infty, & \text{ako je } M(i,j) \geqslant \Delta, \end{cases} \tag{66}$$

za svako $i \in [1..m]$ i $j \in [1..n]$. Ovu matricu nazivamo Δ-***odsecanjem*** matrice M.

Ako vektor $\alpha \in \mathbb{P}_\infty^n$ posmatramo kao matricu dimenzije $1 \times n$ ili $n \times 1$, prethodna definicija daje Δ-odsecanje $\alpha_\Delta \in \mathbb{P}_\Delta^n$ vektora α.

Neposredno se može proveriti da za svaku matricu M sa elementima iz $\mathbb{P}_\infty$ važi $(M^\top)_\Delta = (M_\Delta)^\top$. Bez opasnosti od zabune, Δ-odsecanje matrice $M^\top$ označavaćemo jednostavno sa $M_\Delta^\top$.

Takođe, očigledno je da za svaku matricu M sa elementima iz $\mathbb{P}_\infty$ i svako $\Delta \in \mathbb{R}$, $\Delta > 0$, važi $M \leqslant M_\Delta$. Isto svojstvo važi i za svaki vektor sa elementima iz $\mathbb{P}_\infty$.

Za matrice $G \in \mathbb{P}_\Delta^{m\times n}$ i $H \in \mathbb{P}_\Delta^{n\times p}$ proizvod nad poluprstenom $\mathbb{P}_\Delta$ označavaćemo sa $G \otimes_\Delta H$, odnosno $G \otimes_\Delta H \in \mathbb{P}_\Delta^{m\times p}$ definisaćemo sa

$$(G \otimes_\Delta H)(i,k) = \bigoplus_{j=1}^{n} G(i,j) \otimes_\Delta H(j,k), \qquad \text{za svako } i \in [1..m] \text{ i } k \in [1..p].$$

Operacija sabiranja u ovom izrazu jeste operacija sabiranja u $\mathbb{P}_\Delta$, ali se može posmatrati i kao operacija sabiranja u $\mathbb{P}_\infty$, budući da je $\oplus_\Delta$ njena restrikcija.

Pokazaćemo da važi sledeće tvrđenje.

Propozicija 5.1 *Neka je $\Delta \in \mathbb{R}$, $\Delta > 0$, neka su $M \in \mathbb{P}_\infty^{m\times n}$ i $N \in \mathbb{P}_\infty^{n\times p}$ proizvoljne matrice, a $\alpha \in \mathbb{P}_\infty^{1\times m}$, $\beta \in \mathbb{P}_\infty^{m\times 1}$ i $\gamma \in \mathbb{P}_\infty^{n\times 1}$ proizvoljni vektori. Tada važi*

$$M_\Delta \otimes_\Delta N_\Delta = (M \otimes N)_\Delta, \quad \alpha_\Delta \otimes_\Delta M_\Delta = (\alpha \otimes M)_\Delta, \quad M_\Delta \otimes_\Delta \gamma_\Delta = (M \otimes \gamma)_\Delta, \tag{67}$$

$$\alpha_\Delta \otimes_\Delta \beta_\Delta = \begin{cases} \alpha \otimes \beta, & \text{ako je } \alpha \otimes \beta < \Delta, \\ +\infty, & \text{ako je } \alpha \otimes \beta \geqslant \Delta. \end{cases} \tag{68}$$

Dokaz 5.7 *Neka su $i \in [1..m]$ i $j \in [1..p]$ proizvoljni. Pokazaćemo da za svako $k \in [1..n]$ važi*

$$M_\Delta(i,k) \otimes_\Delta N_\Delta(k,j) < \Delta \iff M(i,k) \otimes N(k,j) < \Delta. \tag{69}$$

Pretpostavimo najpre da je $M_\Delta(i,k) \otimes_\Delta N_\Delta(k,j) < \Delta$.

Tada je $M_\Delta(i,k) < \Delta$ i $N_\Delta(k,j) < \Delta$, pa iz definicije Δ-odsecanja sledi

$$M_\Delta(i,k) = M(i,k), N_\Delta(k,j) = N(k,j).$$

Prema tome, $M(i,k) \otimes_\Delta N(k,j) = M_\Delta(i,k) \otimes_\Delta N_\Delta(k,j) < \Delta$.

Na osnovu definicije operacije $\otimes_\Delta$ dobijamo $M(i,k) \otimes_\Delta N(k,j) = M(i,k) \otimes N(k,j)$, odakle sledi $M(i,k) \otimes N(k,j) < \Delta$.

Obratno, neka je $M(i,k) \otimes N(k,j) < \Delta$. Tada je nužno $M(i,k) < \Delta$, $N(k,j) < \Delta$, pa važi $M_\Delta(i,k) = M(i,k)$, $N_\Delta(k,j) = N(k,j)$.

Stoga dobijamo $M(i,k) \otimes N(k,j) = M(i,k) \otimes_\Delta N(k,j) = M_\Delta(i,k) \otimes_\Delta N_\Delta(k,j)$, te je $M_\Delta(i,k) \otimes_\Delta N_\Delta(k,j) < \Delta$.

Na osnovu ekvivalencije (69) *važi*

$$\{k \in [1..n] \mid M_\Delta(i,k) \otimes_\Delta N_\Delta(k,j) < \Delta\} = \{k \in [1..n] \mid M(i,k) \otimes N(k,j) < \Delta\}.$$

Označimo ovaj skup sa $K = \{k \in [1..n] \mid M_\Delta(i,k) \otimes_\Delta N_\Delta(k,j) < \Delta\}$.

Neka je najpre $K \subsetneq [1..n]$. *Tada postoji* $l \in [1..n] \setminus K$, *pa je* $M_\Delta(i,l) \otimes_\Delta N_\Delta(l,j) = +\infty$. *Kako je operacija* $\oplus$ *maksimum, sledi*

$$(M_\Delta \otimes_\Delta N_\Delta)(i,j) = \bigoplus_{k=1}^{n} M_\Delta(i,k) \otimes_\Delta N_\Delta(k,j) = +\infty.$$

Sa druge strane, iz $l \notin K$ *i* (69) *dobijamo* $M(i,l) \otimes N(l,j) \geqslant \Delta$, *pa je*

$$(M \otimes N)(i,j) = \bigoplus_{k=1}^{n} M(i,k) \otimes N(k,j) \geqslant \Delta.$$

Otuda je $(M \otimes N)_\Delta(i,j) = +\infty$, *te važi* $(M_\Delta \otimes_\Delta N_\Delta)(i,j) = (M \otimes N)_\Delta(i,j)$.

Pretpostavimo sada da je $K = [1..n]$. *Tada za svako* $k \in [1..n]$ *važi*

$$M_\Delta(i,k) \otimes_\Delta N_\Delta(k,j) = M(i,k) \otimes N(k,j),$$

pa dobijamo

$$\begin{aligned}(M_\Delta \otimes_\Delta N_\Delta)(i,j) &= \bigoplus_{k=1}^{n} M_\Delta(i,k) \otimes_\Delta N_\Delta(k,j) \\ &= \bigoplus_{k=1}^{n} M(i,k) \otimes N(k,j) \\ &= (M \otimes N)(i,j).\end{aligned}$$

Kako je $K = [1..n]$, *svi sabirci su manji od* Δ, *pa je i* $(M \otimes N)(i,j) < \Delta$.

Zbog toga je $(M \otimes N)_\Delta(i,j) = (M \otimes N)(i,j)$, *te ponovo važi* $(M_\Delta \otimes_\Delta N_\Delta)(i,j) = (M \otimes N)_\Delta(i,j)$.

Kako su i *i* j *bili proizvoljni, zaključujemo da je* $M_\Delta \otimes_\Delta N_\Delta = (M \otimes N)_\Delta$.

Druga i treća jednakost u (67) *neposredno slede iz prve, dok se jednakost* (68) *dokazuje na potpuno analogan način.* □

Primer 5.2 *U ovom primeru ponovo razmatramo automate* $\mathcal{A}$ *i* $\mathcal{B}$ *iz Primera 5.1. Za proizvoljno* $\Delta \in \mathbb{R}$, $\Delta > 0$, *posmatrajmo njihove* Δ*-odsečene automate* $\mathcal{A}_\Delta$ *i* $\mathcal{B}_\Delta$. *U skladu sa* (72), τ*-parovi automata* $\mathcal{A}_\Delta$ *i* $\mathcal{B}_\Delta$ *dobijaju se* Δ*-odsecanjem odgovarajućih* τ*-vektora automata* $\mathcal{A}$ *i* $\mathcal{B}$.

Za razliku od slučaja bez odsecanja, skup svih τ*-parova automata* $\mathcal{A}_\Delta$ *i* $\mathcal{B}_\Delta$ *je konačan. Isto važi i za skup svih* σ*-parova. Stoga se ispitivanje inkluzije i ekvivalencije automata* $\mathcal{A}_\Delta$ *i* $\mathcal{B}_\Delta$, *kao i određivanje njihovih najvećih slabih forward presimulacija i prebisimulacija, može sprovesti primenom konačnog postupka.*

Za $\Delta = 5$*, postupak određivanja najveće slabe forward presimulacije između automata* $\mathcal{A}_\Delta$ *i* $\mathcal{B}_\Delta$*, primenom Procedure 2, prikazan je u Tabeli 8. Napominjemo da se pod brojem koraka podrazumeva ukupan broj* τ*-parova generisanih tokom izvršavanja procedure, pri čemu generisani* τ*-parovi ne moraju biti međusobno različiti. Kao što se vidi iz tabele, za* $\Delta = 5$ *procedura se izvršava u ukupno* 27 *koraka. Nakon* 5 *iteracija generisano je* 13 *različitih* τ*-parova, a najveća slaba forward simulacija između automata* $\mathcal{A}_\Delta$ *i* $\mathcal{B}_\Delta$ *je:*

$$\begin{bmatrix} +\infty & +\infty & -\infty \\ -\infty & 1 & 0 \end{bmatrix}.$$

Neposrednom proverom može se utvrditi da ova matrica zadovoljava uslov (1-wfs), odakle sledi da ona predstavlja najveću slabu forward simulaciju između max-plus automata $\mathcal{A}_\Delta$ *i* $\mathcal{B}_\Delta$*.*

Za $\Delta = 6$*, u izvršavanju procedure generiše se ukupno* 33 τ*-parova, pri čemu je potrebno* 5 *iteracija* `for` *petlje i dobija se* 16 *različitih* τ*-parova. Najveća slaba forward simulacija dobijena u ovom slučaju jednaka je onoj koja se dobija za* $\Delta = 5$*.*

Sa druge strane, za $\Delta = 7$*, procedura se izvršava u ukupno* 39 *koraka, kroz* 6 *iteracija* `for` *petlje, pri čemu se generiše* 19 *različitih* τ*-parova. U ovom slučaju dobijena najveća slaba forward simulacija jednaka je onoj koja je određena u delu (a) Primera 5.1, odnosno simulaciji dobijenoj u slučaju kada se ne primenjuje* Δ*-odsecanje.*

Tabela 8: Detalji izračunavanja iz primera 5.1 za $\Delta = 5$.

Broj koraka	**Ciklus**	**Par** $(\tau_u^{A_\Delta}, \tau_u^{B_\Delta})$ **τ-vectors**	**Rezidual** $\tau_u^{A_\Delta} \backslash_\Delta \tau_u^{B_\Delta}$	**Trenutni infimum**	
1	0	$\left(\begin{bmatrix} -\infty \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 3 \\ 0 \end{bmatrix}\right)$	$\begin{bmatrix} +\infty & +\infty & +\infty \\ 1 & 3 & 0 \end{bmatrix}$	$\begin{bmatrix} +\infty & +\infty & +\infty \\ 1 & 3 & 0 \end{bmatrix}$	$u = e$
2	1	$\left(\begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} +\infty \\ +\infty \\ 2 \end{bmatrix}\right)$	$\begin{bmatrix} +\infty & +\infty & 1 \\ +\infty & +\infty & 0 \end{bmatrix}$	$\begin{bmatrix} +\infty & +\infty & 1 \\ 1 & 3 & 0 \end{bmatrix}$	$u = x$
3		$\left(\begin{bmatrix} -\infty \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}\right)$	$\begin{bmatrix} +\infty & +\infty & +\infty \\ -\infty & 1 & 0 \end{bmatrix}$	$\begin{bmatrix} +\infty & +\infty & 1 \\ -\infty & 1 & 0 \end{bmatrix}$	$u = y$
4	2	$\left(\begin{bmatrix} +\infty \\ 4 \end{bmatrix}, \begin{bmatrix} +\infty \\ +\infty \\ 4 \end{bmatrix}\right)$	$\begin{bmatrix} +\infty & +\infty & -\infty \\ +\infty & +\infty & 0 \end{bmatrix}$	$\begin{bmatrix} +\infty & +\infty & -\infty \\ -\infty & 1 & 0 \end{bmatrix}$	$u = x^2$
5		$\left(\begin{bmatrix} 2 \\ 3 \end{bmatrix}, \begin{bmatrix} +\infty \\ +\infty \\ 3 \end{bmatrix}\right)$	$\begin{bmatrix} +\infty & +\infty & 1 \\ +\infty & +\infty & 0 \end{bmatrix}$	$\begin{bmatrix} +\infty & +\infty & -\infty \\ -\infty & 1 & 0 \end{bmatrix}$	$u = xy$

Broj koraka	**Ciklus**	**Par** $(\tau_u^{A_\Delta}, \tau_u^{B_\Delta})$ **τ-vectors**	**Rezidual** $\tau_u^{A_\Delta} \blacktriangleright_\Delta \tau_u^{B_\Delta}$	**Trenutni infimum**	
6		$\left(\begin{bmatrix}-\infty\\3\end{bmatrix}, \begin{bmatrix}2\\4\\3\end{bmatrix}\right)$	$\begin{bmatrix}+\infty & +\infty & +\infty\\ -\infty & 1 & 0\end{bmatrix}$	$\begin{bmatrix}+\infty & +\infty & -\infty\\ -\infty & 1 & 0\end{bmatrix}$	$u = yx$
7		$\left(\begin{bmatrix}-\infty\\2\end{bmatrix}, \begin{bmatrix}1\\3\\2\end{bmatrix}\right)$	$\begin{bmatrix}+\infty & +\infty & +\infty\\ -\infty & 1 & 0\end{bmatrix}$	$\begin{bmatrix}+\infty & +\infty & -\infty\\ -\infty & 1 & 0\end{bmatrix}$	$u = y^2$
8	3	$\left(\begin{bmatrix}+\infty\\+\infty\end{bmatrix}, \begin{bmatrix}+\infty\\+\infty\\+\infty\end{bmatrix}\right)$	$\begin{bmatrix}+\infty & +\infty & +\infty\\ +\infty & +\infty & +\infty\end{bmatrix}$	$\begin{bmatrix}+\infty & +\infty & -\infty\\ -\infty & 1 & 0\end{bmatrix}$	$u = x^3$
9		$\left(\begin{bmatrix}+\infty\\+\infty\end{bmatrix}, \begin{bmatrix}+\infty\\+\infty\\+\infty\end{bmatrix}\right)$ ⊠			$u = x^2y$
10		$\left(\begin{bmatrix}4\\+\infty\end{bmatrix}, \begin{bmatrix}+\infty\\+\infty\\+\infty\end{bmatrix}\right)$	$\begin{bmatrix}+\infty & +\infty & +\infty\\ +\infty & +\infty & +\infty\end{bmatrix}$	$\begin{bmatrix}+\infty & +\infty & -\infty\\ -\infty & 1 & 0\end{bmatrix}$	$u = xyx$
11		$\left(\begin{bmatrix}3\\4\end{bmatrix}, \begin{bmatrix}+\infty\\+\infty\\4\end{bmatrix}\right)$	$\begin{bmatrix}+\infty & +\infty & 1\\ +\infty & +\infty & 0\end{bmatrix}$	$\begin{bmatrix}+\infty & +\infty & -\infty\\ -\infty & 1 & 0\end{bmatrix}$	$u = xy^2$
12		$\left(\begin{bmatrix}-\infty\\+\infty\end{bmatrix}, \begin{bmatrix}4\\+\infty\\+\infty\end{bmatrix}\right)$	$\begin{bmatrix}+\infty & +\infty & +\infty\\ -\infty & +\infty & +\infty\end{bmatrix}$	$\begin{bmatrix}+\infty & +\infty & -\infty\\ -\infty & 1 & 0\end{bmatrix}$	$u = yx^2$
13		$\left(\begin{bmatrix}-\infty\\4\end{bmatrix}, \begin{bmatrix}3\\+\infty\\4\end{bmatrix}\right)$	$\begin{bmatrix}+\infty & +\infty & +\infty\\ -\infty & +\infty & 0\end{bmatrix}$	$\begin{bmatrix}+\infty & +\infty & -\infty\\ -\infty & 1 & 0\end{bmatrix}$	$u = yxy$
14		$\left(\begin{bmatrix}-\infty\\4\end{bmatrix}, \begin{bmatrix}3\\+\infty\\4\end{bmatrix}\right)$ ⊠			$u = y^2x$
15		$\left(\begin{bmatrix}-\infty\\3\end{bmatrix}, \begin{bmatrix}2\\4\\3\end{bmatrix}\right)$ ⊠			$u = y^3$
16-20	4	$\left(\begin{bmatrix}+\infty\\+\infty\end{bmatrix}, \begin{bmatrix}+\infty\\+\infty\\+\infty\end{bmatrix}\right)$ ⊠			$u \in \{x^4, x^2yx, x^2y^2, xyx^2, xyxy\}$
21		$\left(\begin{bmatrix}-\infty\\+\infty\end{bmatrix}, \begin{bmatrix}+\infty\\+\infty\\+\infty\end{bmatrix}\right)$	$\begin{bmatrix}+\infty & +\infty & +\infty\\ +\infty & +\infty & +\infty\end{bmatrix}$	$\begin{bmatrix}+\infty & +\infty & -\infty\\ -\infty & 1 & 0\end{bmatrix}$	$u = yx^3$

Broj koraka	Ciklus	Par $(\tau_u^{A_\Delta}, \tau_u^{B_\Delta})$ τ-vectors	Rezidual $\tau_u^{A_\Delta} \backslash_\Delta \tau_u^{B_\Delta}$	Trenutni infimum
22, 24		$\left(\begin{bmatrix}-\infty\\+\infty\end{bmatrix}, \begin{bmatrix}+\infty\\+\infty\\+\infty\end{bmatrix}\right)$ ⊠		$u \in \{yxyx, y^2x^2\}$
23, 25		$\left(\begin{bmatrix}-\infty\\+\infty\end{bmatrix}, \begin{bmatrix}4\\+\infty\\+\infty\end{bmatrix}\right)$ ⊠		$u \in \{yxy^2, y^2xy\}$
26	5	$\left(\begin{bmatrix}+\infty\\+\infty\end{bmatrix}, \begin{bmatrix}+\infty\\+\infty\\+\infty\end{bmatrix}\right)$ ⊠		$u = xyx^3$
27		$\left(\begin{bmatrix}-\infty\\+\infty\end{bmatrix}, \begin{bmatrix}+\infty\\+\infty\\+\infty\end{bmatrix}\right)$ ⊠		$u = y^2x^3$

5.5 Δ-simulacije i Δ-bisimulacije. Relaksacije ograničene datumom

Težinski automati nad poluprstenom $\mathbb{P}_\Delta$ takođe će se nazivati max-plus automatima, odnosno preciznije, *max-plus automatima nad* $\mathbb{P}_\Delta$.

Neka je $\Delta \in \mathbb{R}$, $\Delta > 0$, i neka je $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ max-plus automat nad $\mathbb{P}_\infty$. Definišimo max-plus automat $\mathcal{A}_\Delta = (m, \sigma^{A_\Delta}, {M_x^{A_\Delta}}_{x\in X}, \tau^{A_\Delta})$ nad $\mathbb{P}_\Delta$ na sledeći način:

$$\sigma^{A_\Delta} = (\sigma^A)_\Delta, \qquad M_x^{A_\Delta} = (M_x^A)_\Delta, \text{ za svako } x \in X, \qquad \tau^{A_\Delta} = (\tau^A)_\Delta.$$

Automat $\mathcal{A}_\Delta$ nazivamo Δ-*odsecanjem automata* $\mathcal{A}$. U slučajevima kada ne postoji opasnost od zabune, koristićemo pojednostavljenu oznaku $\mathcal{A}_\Delta = (m, \sigma^\Delta, \{M_x^\Delta\}_{x\in X}, \tau^\Delta)$.

Na osnovu Propozicije 5.1, kompozitne matrice prelaza, kao i σ- i τ-vektora automata $\mathcal{A}_\Delta$ koji odgovaraju reči $u = x_1x_2\ldots x_k \in X^+$, $x_1, x_2, \ldots, x_k \in X$, dati su sa

$$M_u^{A_\Delta} = M_{x_1}^{A_\Delta} \otimes_\Delta M_{x_2}^{A_\Delta} \otimes_\Delta \cdots \otimes_\Delta M_{x_k}^{A_\Delta} = \left(M_{x_1}^A \otimes M_{x_2}^A \otimes \cdots \otimes M_{x_k}^A\right)_\Delta = (M_u^A)_\Delta \quad (70)$$

$$\sigma_u^{A_\Delta} = \sigma^{A_\Delta} \otimes_\Delta M_u^{A_\Delta} = \left(\sigma^A \otimes M_u^A\right)_\Delta = (\sigma_u^A)_\Delta, \quad (71)$$

$$\tau_u^{A_\Delta} = M_u^{A_\Delta} \otimes_\Delta \tau^{A_\Delta} = \left(M_u^A \otimes \tau^A\right)_\Delta = (\tau_u^A)_\Delta. \quad (72)$$

Za praznu reč važi

$$M_\varepsilon^{A_\Delta} = I_m = (I_m)_\Delta = (M_\varepsilon^A)_\Delta, \quad (73)$$

$$\sigma_\varepsilon^{A_\Delta} = \sigma^{A_\Delta} = (\sigma^A)_\Delta, \quad (74)$$

$$\tau_\varepsilon^{A_\Delta} = \tau^{A_\Delta} = (\tau^A)_\Delta. \quad (75)$$

Za proizvoljnu funkciju reči $f : X^* \to \mathbb{P}_\infty$ definišemo njeno Δ-*odsecanje* kao funkciju

reči $f_\Delta : X^* \to \mathbb{P}_\Delta$ datu za svaku reč $u \in X^*$ sa

$$f_\Delta(u) = \begin{cases} f(u), & \text{ako je } f(u) < \Delta, \\ +\infty, & \text{ako je } f(u) \geqslant \Delta. \end{cases} \tag{76}$$

Teorema 5.7 *Neka je $\Delta \in \mathbb{R}$, $\Delta > 0$, i neka je $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ max-plus automat nad $\mathbb{P}_\infty$. Tada važi $[\![\mathcal{A}_\Delta]\!] = [\![\mathcal{A}]\!]_\Delta$.*

Dokaz 5.8 *Pokažimo da tvrđenje važi za svaku reč $u \in X^*$,*

Na osnovu Propozicije 5.1, za praznu reč $u = \varepsilon$ biće

$$\begin{aligned} [\![\mathcal{A}_\Delta]\!](\varepsilon) &= \sigma^{A_\Delta} \otimes_\Delta \tau^{A_\Delta} \\ &= \begin{cases} \sigma^A \otimes \tau^A, & \textit{ako je } \sigma^A \otimes \tau^A < \Delta, \\ +\infty, & \textit{ako je } \sigma^A \otimes \tau^A \geqslant \Delta \end{cases} \\ &= \begin{cases} [\![\mathcal{A}]\!](\varepsilon), & \textit{ako je } [\![\mathcal{A}]\!](\varepsilon) < \Delta, \\ +\infty, & \textit{ako je } [\![\mathcal{A}]\!](\varepsilon) \geqslant \Delta \end{cases} \\ &= [\![\mathcal{A}]\!]_\Delta(\varepsilon). \end{aligned}$$

Za proizvoljnu nepraznu reč $u \in X^+$, na osnovu Propozicije 5.1 i 75 dobijamo

$$\begin{aligned} [\![\mathcal{A}_\Delta]\!](u) &= \sigma^{A_\Delta} \otimes_\Delta M_u^{A_\Delta} \otimes_\Delta \tau^{A_\Delta} = \\ &= \left(\sigma^A \otimes M_u^A\right)_\Delta \otimes_\Delta (\tau^A)_\Delta = \\ &= \begin{cases} \sigma^A \otimes M_u^A \otimes \tau^A, & \textit{ako je } \sigma^A \otimes M_u^A \otimes \tau^A < \Delta, \\ +\infty, & \textit{ako je } \sigma^A \otimes M_u^A \otimes \tau^A \geqslant \Delta \end{cases} \\ &= \begin{cases} [\![\mathcal{A}]\!](u), & \textit{ako je } [\![\mathcal{A}]\!](u) < \Delta, \\ +\infty, & \textit{ako je } [\![\mathcal{A}]\!](u) \geqslant \Delta \end{cases} \\ &= [\![\mathcal{A}]\!]_\Delta(u). \end{aligned}$$

Prema tome, za svako $u \in X^*$ važi $[\![\mathcal{A}_\Delta]\!](u) = [\![\mathcal{A}]\!]_\Delta(u)$. □

Neka je $\Delta \in \mathbb{R}$, $\Delta > 0$, i neka su $\mathcal{A}$ i $\mathcal{B}$ max-plus automati nad $\mathbb{P}_\infty$. Definišimo jezik $L_\Delta(\mathcal{A}, \mathcal{B}) \subseteq X^*$ sa

$$L_\Delta(\mathcal{A}, \mathcal{B}) = \left\{u \in X^* \mid [\![\mathcal{A}]\!](u) < \Delta \text{ ili } [\![\mathcal{B}]\!](u) < \Delta\right\}. \tag{77}$$

Prethodni rezultati omogućavaju da se relacija inkluzije između Δ-odsečenih automata izrazi preko Δ-odsečenja njihovih funkcija ponašanja, kao i preko L_Δ-inkluzije polaznih automata. Preciznije, važi sledeća teorema.

Teorema 5.8 *Neka je* $\Delta \in \mathbb{R}$, $\Delta > 0$, *i neka su* $\mathcal{A}$ *i* $\mathcal{B}$ *proizvoljni max-plus automati nad* $\mathbb{P}_\infty$. *Tada su sledeća tvrđenja ekvivalentna:*

(a) $[\![\mathcal{A}_\Delta]\!] \leqslant [\![\mathcal{B}_\Delta]\!]$;

(b) $[\![\mathcal{A}]\!]_\Delta \leqslant [\![\mathcal{B}]\!]_\Delta$;

(c) $[\![\mathcal{A}]\!] \leqslant_L [\![\mathcal{B}]\!]$, *gde je* $L = L_\Delta(\mathcal{A}, \mathcal{B})$.

Dokaz 5.9 *Ekvivalentnost tvrđenja* (a) *i* (b) *neposredno sledi iz Teoreme 5.7.*

(b) $\Rightarrow$ (c) *Pretpostavimo da važi* $[\![\mathcal{A}]\!]_\Delta \leqslant [\![\mathcal{B}]\!]_\Delta$ *i posmatrajmo proizvoljnu reč* $u \in L$. *Prema definiciji skupa* $L = L_\Delta(\mathcal{A}, \mathcal{B})$ *imamo da je* $[\![\mathcal{A}]\!](u) < \Delta$ *ili* $[\![\mathcal{B}]\!](u) < \Delta$.

Ako je $[\![\mathcal{A}]\!](u) < \Delta$ *i* $[\![\mathcal{B}]\!](u) < \Delta$, *tada važi*

$$[\![\mathcal{A}]\!](u) = [\![\mathcal{A}]\!]_\Delta(u) \leqslant [\![\mathcal{B}]\!]_\Delta(u) = [\![\mathcal{B}]\!](u).$$

Ako je $[\![\mathcal{A}]\!](u) < \Delta$ *i* $[\![\mathcal{B}]\!](u) \geqslant \Delta$, *tada je*

$$[\![\mathcal{A}]\!](u) < \Delta \leqslant [\![\mathcal{B}]\!](u).$$

Slučaj $[\![\mathcal{A}]\!](u) \geqslant \Delta$ *i* $[\![\mathcal{B}]\!](u) < \Delta$ *nije moguć, jer bi to bilo u suprotnosti sa polaznom pretpostavkom:*

$$[\![\mathcal{A}]\!]_\Delta(u) = +\infty > \Delta > [\![\mathcal{B}]\!](u) = [\![\mathcal{B}]\!]_\Delta(u).$$

Prema tome, $[\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u)$ *za svako* $u \in L$, *odakle sledi* $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!]$.

(c) $\Rightarrow$ (b) *Pretpostavimo da važi* $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!]$ *i posmatrajmo proizvoljnu reč* $u \in X^*$. *Ako je* $u \in L$, *tada prema pretpostavci važi* $[\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u)$.
Ako je pritom $[\![\mathcal{B}]\!](u) < \Delta$, *tada je*

$$[\![\mathcal{A}]\!]_\Delta(u) = [\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u) = [\![\mathcal{B}]\!]_\Delta(u).$$

U suprotnom, ako je $[\![\mathcal{B}]\!](u) \geqslant \Delta$, *tada je*

$$[\![\mathcal{A}]\!]_\Delta(u) \leqslant +\infty = [\![\mathcal{B}]\!]_\Delta(u).$$

Sa druge strane, ako $u \notin L$, *tada prema definiciji skupa* $L = L_\Delta(\mathcal{A}, \mathcal{B})$ *važi* $[\![\mathcal{A}]\!](u) \geqslant \Delta$ *i* $[\![\mathcal{B}]\!](u) \geqslant \Delta$, *pa sledi*

$$[\![\mathcal{A}]\!]_\Delta(u) = +\infty = [\![\mathcal{B}]\!]_\Delta(u).$$

Prema tome, za svako $u \in X^*$ *važi* $[\![\mathcal{A}]\!]_\Delta(u) \leqslant [\![\mathcal{B}]\!]_\Delta(u)$, *odnosno* $[\![\mathcal{A}]\!]_\Delta \leqslant [\![\mathcal{B}]\!]_\Delta$. □

Na potpuno analogan način dobijamo odgovarajuću karakterizaciju L_Δ-ekvivalencije.

Posledica 5.1 *Neka je* $\Delta \in \mathbb{R}$, $\Delta > 0$, *i neka su* $\mathcal{A}$ *i* $\mathcal{B}$ *proizvoljni max-plus automati nad* $\mathbb{P}_\infty$. *Tada su sledeća tvrđenja ekvivalentna:*

(a) $[\![\mathcal{A}_\Delta]\!] = [\![\mathcal{B}_\Delta]\!]$;

(b) $[\![\mathcal{A}]\!]_\Delta = [\![\mathcal{B}]\!]_\Delta$;

(c) $[\![\mathcal{A}]\!] \equiv_L [\![\mathcal{B}]\!]$, *gde je* $L = L_\Delta(\mathcal{A}, \mathcal{B})$.

Sledeća tvrđenja takođe predstavljaju direktne posledice prethodnih rezultata.

Posledica 5.2 *Neka su* $\Delta, \Delta_1, \Delta_2 \in \mathbb{R}$ *takvi da je* $\Delta > 0$ *i* $\Delta_2 > \Delta_1 > 0$, *i neka su* $\mathcal{A}$ *i* $\mathcal{B}$ *proizvoljni max-plus automati nad* $\mathbb{P}_\infty$. *Tada važe sledeća tvrđenja:*

(a) $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{A}_\Delta]\!]$ *i* $[\![\mathcal{A}]\!] \equiv_L [\![\mathcal{A}_\Delta]\!]$, *gde je* $L = L_\Delta(\mathcal{A})$;

(b) $[\![\mathcal{A}_{\Delta_2}]\!] \leqslant [\![\mathcal{A}_{\Delta_1}]\!]$ *i* $[\![\mathcal{A}_{\Delta_2}]\!] \equiv_{L_1} [\![\mathcal{A}_{\Delta_1}]\!]$, *gde je* $L_1 = L_{\Delta_1}(\mathcal{A})$;

(c) $L_{\Delta_1}(\mathcal{A}, \mathcal{B}) \subseteq L_{\Delta_2}(\mathcal{A}, \mathcal{B})$;

(d) $[\![\mathcal{A}]\!] \leqslant_{L_2} [\![\mathcal{A}]\!]$ *implicira* $[\![\mathcal{A}]\!] \leqslant_{L_1} [\![\mathcal{A}]\!]$,
gde je $L_1 = L_{\Delta_1}(\mathcal{A}, \mathcal{B})$, $L_2 = L_{\Delta_2}(\mathcal{A}, \mathcal{B})$;

(e) $[\![\mathcal{A}]\!] \equiv_{L_2} [\![\mathcal{A}]\!]$ *implicira* $[\![\mathcal{A}]\!] \equiv_{L_1} [\![\mathcal{A}]\!]$,
gde je $L_1 = L_{\Delta_1}(\mathcal{A}, \mathcal{B})$, $L_2 = L_{\Delta_2}(\mathcal{A}, \mathcal{B})$.

Osnovni razlog za uvođenje Δ-odsečenog poluprstena jeste činjenica da njegova lokalna konačnost omogućava da se dobije konačnost skupova svih τ-parova i σ-parova max-plus automata. Preciznije, važi sledeća teorema.

Teorema 5.9 *Neka je* $\Delta \in \mathbb{R}$, $\Delta > 0$, *i neka su* $\mathcal{A}$ *i* $\mathcal{B}$ *max-plus automati nad* Δ*-odsečenim poluprstenom* $\mathbb{P}_\Delta$. *Tada su skup svih* τ*-parova i skup svih* σ*-parova automata* $\mathcal{A}$ *i* $\mathcal{B}$ *konačni.*

Dokaz 5.10 *Dokazaćemo samo tvrđenje koje se odnosi na* τ*-parove, budući da se tvrđenje koje se odnosi na* σ*-parove dokazuje na analogan način.*

Neka je E *skup svih elemenata koji se pojavljuju u završnim težinskim vektorima* τ^A *i* τ^B, *kao i u matricama težina prelaza* M_x^A *i* M_x^B, *za svako* $x \in X$, *i neka je* $\mathbb{G}$ *podpoluprsten poluprstena* $\mathbb{P}_\Delta$ *generisan skupom* E. *Budući da je* $\mathbb{P}_\Delta$ *lokalno konačan poluprsten, a skup* E *konačan, sledi da je poluprsten* $\mathbb{G}$ *konačan.*

Svi elementi τ*-vektora automata* $\mathcal{A}$ *i* $\mathcal{B}$ *pripadaju poluprstenu* $\mathbb{G}$, *pa zaključujemo da su skupovi svih* τ*-vektora automata* $\mathcal{A}$ *i* $\mathcal{B}$ *konačni. Odavde neposredno sledi da je i skup svih* τ*-parova automata* $\mathcal{A}$ *i* $\mathcal{B}$ *konačan.* □

Neka je $\Delta \in \mathbb{R}$, $\Delta > 0$, i neka su $\mathcal{A}$ i $\mathcal{B}$ max-plus automati nad $\mathbb{P}_\infty$. Pod ***slabom forward*** (odnosno ***backward***) Δ***-simulacijom*** između automata $\mathcal{A}$ i $\mathcal{B}$ podrazumevaćemo L-relaksiranu slabu forward (odnosno backward) simulaciju između $\mathcal{A}$ i $\mathcal{B}$, gde je $L = L_\Delta(\mathcal{A}, \mathcal{B})$. Analogno, pod ***slabom forward*** (odnosno ***backward***) Δ***-bisimulacijom*** podrazumevaćemo L-relaksiranu slabu forward (odnosno backward) bisimulaciju između $\mathcal{A}$ i $\mathcal{B}$ za $L = L_\Delta(\mathcal{A}, \mathcal{B})$.

Na osnovu Posledice 5.1, postupak za ispitivanje postojanja i određivanje najveće slabe Δ-simulacije, odnosno Δ-bisimulacije datog tipa između max-plus automata $\mathcal{A}$ i $\mathcal{B}$ nad $\mathbb{P}\infty$ može se svesti na odgovarajući postupak za ispitivanje postojanja i određivanje najveće obične slabe simulacije, odnosno bisimulacije istog tipa između njihovih Δ-odsečenih automata $\mathcal{A}_\Delta$ i $\mathcal{B}_\Delta$ nad $\mathbb{P}_\Delta$. Na taj način problem određivanja slabih Δ-simulacija i Δ-bisimulacija postaje konačan, jer automati $\mathcal{A}_\Delta$ i $\mathcal{B}_\Delta$ imaju težine iz konačnog poluprstena $\mathbb{P}_\Delta$. Stoga se najveće slabe Δ-simulacije i Δ-bisimulacije mogu efikasno određivati primenom postupaka razvijenih za obične slabe simulacije i bisimulacije nad konačnim max-plus strukturama.

Glava 6

Odsecanje i pomeranje težina za max-plus automate

U prethodnim glavama razmatrani su sistemi matričnih nejednačina čija rešenja predstavljaju relacije poređenja između max-plus automata, pri čemu se najveće presimulacije i prebisumulacije mogu određivati iterativnim postupkom. Kada se ovaj postupak stabilizuje nakon konačno mnogo koraka, odgovarajuće najveće rešenje može se efikasno izračunati. Međutim, u opštem slučaju niz matrica dobijen ovim postupkom može biti beskonačan, pri čemu se njegove komponente mogu strogo smanjivati i konvergirati ka $-\infty$. U takvom slučaju infimum niza, a samim tim ni najveća presimulacija ili prebisumulacija, u opštem slučaju ne može se odrediti standardnim iterativnim postupkom u konačno mnogo koraka [9].

Ovaj problem može se pojaviti čak i kada su sve realne težine koje se javljaju u prelaznim matricama, kao i u vektorima početnih i završnih težina posmatranih automata, pozitivne. Naime, primena operacije reziduacije u konstrukciji niza matrica može dovesti do pojave negativnih elemenata, pri čemu se na pojedinim pozicijama mogu formirati beskonačno opadajući nizovi realnih brojeva koji konvergiraju ka $-\infty$. Zbog toga se prirodno postavlja pitanje da li se struktura nad kojom se vrši računanje može modifikovati tako da se ovakvi beskonačni procesi svedu na konačan broj koraka.

Jedan od odgovora na ovo pitanje predstavlja *odsecanje* kompletnog max-plus poluprstena $\mathbb{R}_\infty$ odozdo, u odnosu na proizvoljan realan broj λ. Osnovna ideja jeste da se iz skupa $\mathbb{R}_\infty$ uklone svi elementi manji od λ, uz zadržavanje elementa $-\infty$. Na taj način dobija se skup $\mathbb{R}_\infty^\lambda = [\lambda, +\infty) \cup \{-\infty, +\infty\}$.

Da bi ovako definisana struktura zadržala svojstva potrebna za razmatranje težinskih automata i sistema matričnih nejednačina, neophodno je odgovarajuće modifikovati i operaciju množenja. Modifikovano množenje dobija se pomeranjem proizvoda u $\mathbb{R}_\infty$ za $-\lambda$, što povlači za sobom i odgovarajuću promenu operacije reziduala. Detaljnije o donjem odsecanju i odgovarajućim operacijama biće reči u prvoj sekciji ove glave.

Pored donjih odsecanja, u nastavku se razmatraju i *gornja odsecanja* kompletnog max-plus poluprstena. U zavisnosti od toga da li se element $+\infty$ zadržava u skupu nosaču, uvode se dve varijante gornjeg odsecanja, koje ćemo označavati sa $\mathbb{R}_\infty^\upsilon$ i $\mathbb{R}_{\max}^\upsilon$. I u ovim strukturama operacija množenja dobija se pomeranjem rezultata proizvoda u $\mathbb{R}_\infty$ za $-\upsilon$. Pokazuje se da obe strukture predstavljaju komutativne kompletne aditivno-idempotentne

poluprstene, pri čemu se njihove operacije reziduacije bitno razlikuju u zavisnosti od prisustva, odnosno odsustva elementa $+\infty$.

Nakon uvođenja odgovarajućih struktura, razmatra se njihova primena na max-plus automate. Za automat $\mathcal{A}$ čije se težine nalaze u $\mathbb{R}_{\infty}^{\lambda}$ definiše se njegovo *donje λ-pomeranje* $\mathcal{A}^{\lambda}$, kao automat nad $\mathbb{R}_{\infty}^{\lambda}$ sa istim osnovnim prelaznim matricama i vektorima početnih i završnih težina. Pokazuje se da prelazak na donja pomeranja može omogućiti konačno izračunavanje najvećih presimulacija i prebisumulacija, kao i da se dobijene relacije mogu povezati sa odgovarajućim relacijama između originalnih automata

Potom se razmatra primena odsecanja na simulacije i bisimulacije između max-plus automata. Numerički primeri ilustruju prednosti i ograničenja ovog pristupa. Biće pokazano da se u pojedinim slučajevima najveća presimulacija, koja se nad kompletnim max-plus poluprstenom ne može odrediti u konačno mnogo koraka, može efikasno izračunati nakon prelaska na odgovarajuće donje odsecanje, kao i da se u nekim slučajevima broj potrebnih koraka može dodatno smanjiti. Sa druge strane, odsecanje može dovesti do gubitka simulacije između pomerenih automata iako ona postoji između originalnih automata.

U završnom delu poglavlja razmatra se primena odsecanja na slabe simulacije i bisimulacije. Za razliku od standardnih simulacija i bisimulacija, slabe simulacije i bisimulacije omogućavaju poređenje šire klase automata, ali njihovo određivanje zahteva razmatranje sistema koji može sadržati beskonačno mnogo matričnih nejednačina [17]. Pokazaće se da u ovom slučaju donja i gornja odsecanja kompletnog max-plus poluprstena ne pružaju analogno efikasan način za obezbeđivanje konačnosti postupka. Dobijeni rezultati ukazuju na ograničenja primene odsecanja u ovom kontekstu, ali istovremeno ukazuju i na moguće pravce daljeg istraživanja.

Rezultati predstavljeni u ovoj glavi zasnovani su na koautorskom radu proisteklom iz istraživanja sprovedenog u okviru ove doktorske disertacije [18].

6.1 Donje odsecanje i pomeranje težina na max-plus poluprstenu

Za proizvoljno $\lambda \in \mathbb{R}$ definišemo skupove

$$\mathbb{R}_{\infty}^{\lambda} = [\lambda, +\infty) \cup \{-\infty, +\infty\} \qquad \text{i} \qquad \mathbb{R}_{\max}^{\lambda} = [\lambda, +\infty) \cup \{-\infty\}.$$

Na skupovima $\mathbb{R}_{\infty}^{\lambda}$ i $\mathbb{R}_{\max}^{\lambda}$ posmatramo restrikcije poretka definisanog na $\mathbb{R}_{\infty}$, koje ćemo, kao i originalni poredak, označavati sa $\leqslant$.

Teorema 6.1 *Za proizvoljno $\lambda \in \mathbb{R}$, neka je $\oplus_\lambda$ operacija sabiranja na $\mathbb{R}_\infty^\lambda$ istovetna operaciji sabiranja na $\mathbb{R}_\infty$, a neka je $\otimes_\lambda$ operacija množenja na $\mathbb{R}_\infty^\lambda$ definisana tablicom:*

$\otimes_\lambda$	$b \in [\lambda, +\infty)$	$-\infty$	$+\infty$
$a \in [\lambda, +\infty)$	$a + b - \lambda$	$-\infty$	$+\infty$
$-\infty$	$-\infty$	$-\infty$	$-\infty$
$+\infty$	$+\infty$	$-\infty$	$+\infty$

Tada je $(\mathbb{R}_\infty^\lambda, \oplus_\lambda, \otimes_\lambda, -\infty, \lambda)$ komutativan kompletan aditivno-idempotentan poluprsten sa nulom $-\infty$ i jedinicom λ, a operacija reziduacije $\rightarrow_\lambda$ pridružena operaciji množenja $\otimes_\lambda$ na $\mathbb{R}_\infty^\lambda$ definisana je na sledeći način:

	$b \in [\lambda, +\infty)$			
$\rightarrow_\lambda$	$a \leqslant b$	$a > b$	$-\infty$	$+\infty$
$a \in [\lambda, +\infty)$	$b - a + \lambda$	$-\infty$	$-\infty$	$+\infty$
$-\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$
$+\infty$	$-\infty$	$-\infty$	$-\infty$	$+\infty$

Dokaz 6.1 *Za proizvoljne $a, b \in [\lambda, +\infty)$, iz $a \geqslant \lambda$ i $b \geqslant \lambda$ sledi $a + b \geqslant 2\lambda$, pa je $a + b - \lambda \geqslant \lambda$, što znači da je $\otimes_\lambda$ dobro definisana operacija na $\mathbb{R}_\infty^\lambda$.*

Uočimo da za sve $a, b \in [\lambda, +\infty)$ važi

$$a \otimes_\lambda b = (a \otimes b) - \lambda = (a - \lambda) \otimes b = a \otimes (b - \lambda).$$

Takođe, skup $\mathbb{R}_\infty^\lambda$ zatvoren je za konačne i beskonačne sume, pri čemu su beskonačne sume u $\mathbb{R}_\infty^\lambda$ nasleđene iz $\mathbb{R}_\infty$. U nastavku ćemo za beskonačne sume u $\mathbb{R}_\infty^\lambda$ koristiti istu oznaku kao za beskonačne sume u $\mathbb{R}_\infty$, što neće dovoditi do zabune, jer se ove sume podudaraju kada se primenjuju na elemente skupa $\mathbb{R}_\infty^\lambda$.

Jasno je da je operacija $\otimes_\lambda$ komutativna, a neposrednom proverom dobija se da je i asocijativna, kao i da je λ njena jedinica. Kako je operacija $\oplus_\lambda$ nasleđena iz $\mathbb{R}_\infty$, ona je komutativna i asocijativna, a $-\infty$ predstavlja njen neutralni element.

Pokažimo sada da važi beskonačna distributivnost.

Najpre neka je $a \in [\lambda, +\infty)$ i $\{b_i\}_{i \in I} \subseteq [\lambda, +\infty)$. Tada važi

$$a \otimes_\lambda \left(\bigoplus_{i \in I} b_i \right) = (a - \lambda) \otimes \left(\bigoplus_{i \in I} b_i \right) = \bigoplus_{i \in I} \big((a - \lambda) \otimes b_i \big) = \bigoplus_{i \in I} (a \otimes_\lambda b_i).$$

Ako je $a = -\infty$, tada za proizvoljnu familiju $\{b_i\}_{i \in I} \subseteq \mathbb{R}_\infty^\lambda$ važi

$$a \otimes_\lambda \left(\bigoplus_{i \in I} b_i \right) = -\infty = \bigoplus_{i \in I} (a \otimes_\lambda b_i).$$

Dalje, neka je $a \in \mathbb{R}_\infty^\lambda$, $a \neq -\infty$, *i neka* $\{b_i\}_{i\in I} \subseteq \mathbb{R}_\infty^\lambda$ *sadrži element* $+\infty$, *tj. neka je* $b_j = +\infty$ *za neko* $j \in I$. *Tada je* $a \otimes_\lambda b_j = +\infty$, *pa važi*

$$a \otimes_\lambda \left(\bigoplus_{i\in I} b_i \right) = a \otimes_\lambda (+\infty) = +\infty = \bigoplus_{i\in I} (a \otimes_\lambda b_i).$$

Konačno, neka je $a \in \mathbb{R}_\infty^\lambda$, $a \neq -\infty$, *i* $\{b_i\}_{i\in I} \subseteq [\lambda, +\infty) \cup \{-\infty\}$. *Tada imamo*

$$a \otimes_\lambda \left(\bigoplus_{i\in I} b_i \right) = a \otimes_\lambda \left(\bigoplus_{j\in J} b_j \right) = \bigoplus_{j\in J} (a \otimes_\lambda b_j) = \bigoplus_{i\in I} (a \otimes_\lambda b_i),$$

gde je $J = \{j \in I \mid b_j \neq -\infty\}$.

Ovde smo iskoristili činjenicu da beskonačne sume predstavljaju supremume, pa se svi članovi jednaki najmanjem elementu, u ovom slučaju $-\infty$, *mogu izostaviti. Ako je skup J prazan, koristi se uobičajena konvencija prema kojoj je supremum praznog skupa jednak najmanjem elementu.*

Time je dokazano da je $\mathbb{R}_\infty^\lambda$ *komutativan kompletan aditivno-idempotentan poluprsten sa nulom* $-\infty$ *i jedinicom* λ.

Preostaje da pokažemo da je $\to_\lambda$ *operacija reziduacije na* $\mathbb{R}_\infty^\lambda$ *pridružena operaciji množenja* $\otimes_\lambda$. *U tu svrhu pokazaćemo da je* $a \to_\lambda b$ *najveće rešenje nejednačine* $a \otimes_\lambda x \leqslant b$, *za proizvoljne* $a, b \in \mathbb{R}_\infty^\lambda$.

Pre nego što razmotrimo pojedinačne slučajeve, uočimo da za $a, b \in [\lambda, +\infty)$ *takve da je* $b \geqslant a$ *važi* $b - a \geqslant 0$, *pa je* $b - a + \lambda \geqslant \lambda$. *Dakle,* $\to_\lambda$ *je dobro definisana operacija na* $\mathbb{R}_\infty^\lambda$.

Ako su $a, b \in [\lambda, +\infty)$ *i* $b \geqslant a$, *tada važi*

$$a \otimes_\lambda x \leqslant b \quad \Longleftrightarrow \quad a + x - \lambda \leqslant b \quad \Longleftrightarrow \quad x \leqslant b - a + \lambda. \tag{78}$$

Prema tome, $b - a + \lambda = a \to_\lambda b$ *predstavlja najveće rešenje nejednačine* $a \otimes_\lambda x \leqslant b$.

Ako su $a, b \in [\lambda, +\infty)$ *i* $b < a$, *tada je* $b - a + \lambda < \lambda$, *pa* $b - a + \lambda \notin [\lambda, +\infty) \cup \{+\infty\}$. *Iz* (78) *sledi da je* $-\infty = a \to_\lambda b$ *jedino rešenje nejednačine* $a \otimes_\lambda x \leqslant b$, *te je samim tim i njeno najveće rešenje.*

Ako je $a \in [\lambda, +\infty)$ *i* $b = -\infty$, *na sličan način zaključujemo da je* $-\infty = a \to_\lambda b$ *jedino rešenje nejednačine* $a \otimes_\lambda x \leqslant b$, *pa je ono i njeno najveće rešenje.*

Dalje, ako je $a = -\infty$, *tada je* $a \otimes_\lambda x = -\infty \leqslant b$ *za sve* $x, b \in \mathbb{R}_\infty^\lambda$. *Prema tome, svaki element* $x \in \mathbb{R}_\infty^\lambda$ *predstavlja rešenje nejednačine* $a \otimes_\lambda x \leqslant b$, *pa je njeno najveće rešenje* $+\infty = a \to_\lambda b$.

Sa druge strane, ako je $b = +\infty$, *tada je za svako* $a \in \mathbb{R}_\infty^\lambda$ *svaki* $x \in \mathbb{R}_\infty^\lambda$ *rešenje nejednačine* $a \otimes_\lambda x \leqslant b$. *Stoga je njeno najveće rešenje* $+\infty = a \to_\lambda b$.

Konačno, neka je $a = +\infty$ i $b \in \mathbb{R}_{\infty}^{\lambda}$, $b \neq +\infty$. Tada je

$$a \otimes_{\lambda} x = \begin{cases} +\infty, & \text{ako je } x \neq -\infty, \\ -\infty, & \text{ako je } x = -\infty. \end{cases}$$

Odavde sledi da je $-\infty = a \to_{\lambda} b$ jedino rešenje nejednačine $a \otimes_{\lambda} x \leqslant b$, pa je ono i njeno najveće rešenje.

Obuhvativši sve moguće slučajeve, zaključujemo da je $\to_{\lambda}$ operacija reziduacije na $\mathbb{R}_{\infty}^{\lambda}$ pridružena operaciji množenja $\otimes_{\lambda}$. □

Poluprsten $\mathbb{R}_{\infty}^{\lambda}$ nazivamo ***donjim λ-odsecanjem*** poluprstena $\mathbb{R}_{\infty}$, pri čemu je operacija množenja pomerena u odnosu na λ. $\mathbb{R}_{\infty}^{\lambda}$ još nazivamo i *donjim odsečenim kompletnim max-plus poluprstenom*.

Ako je $\lambda \in \mathbb{Z}$, tada je $\mathbb{Z}_{\infty}^{\lambda}$ podskup skupa $\mathbb{R}_{\infty}^{\lambda}$ zatvoren za operacije $\oplus_{\lambda}$, $\otimes_{\lambda}$ i $\to_{\lambda}$, kao i za beskonačne sume. Prema tome,

$$\mathbb{Z}_{\infty}^{\lambda} = (\mathbb{Z}_{\infty}^{\lambda}, \oplus_{\lambda}, \otimes_{\lambda}, -\infty, \lambda), \quad \mathbb{Z}_{\infty}^{\lambda} = \{z \in \mathbb{Z} \mid \lambda \leqslant z\} \cup \{-\infty, +\infty\}$$

predstavlja komutativni kompletan aditivno-idempotentan poluprsten, koji nazivamo *donjim λ-odsecanjem* poluprstena $\mathbb{Z}_{\infty}$ sa pomerenim množenjem.

Posebno je značajno donje 0-odsecanje kompletnog max-plus poluprstena. Poluprsten $\mathbb{R}_{\infty}^{0}$ proučavan je u Glavi 3, kao i u radu [9], gde je označavan sa $\mathbb{P}_{\infty}$ odnosno $\mathbb{R}_{\infty}^{+}$. Operacija množenja $\otimes_0$ na $\mathbb{R}_{\infty}^{0}$ predstavlja restrikciju operacije množenja $\otimes$ na $\mathbb{R}_{\infty}$, dok se operacija reziduacije $\to_0$ razlikuje od operacije reziduacije $\to$ na $\mathbb{R}_{\infty}$ i zapravo se podudara sa operacijom reziduacije na poluprstenu $\mathbb{P}_{\infty}$. Radi lakšeg praćenja daljeg teksta, ovu operaciju navodimo i u sledećoj tabeli:

	$b \in [0, +\infty)$			
$\to_0$	$a \leqslant b$	$a > b$	$-\infty$	$+\infty$
$a \in [0, +\infty)$	$b - a$	$-\infty$	$-\infty$	$+\infty$
$-\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$
$+\infty$	$-\infty$	$-\infty$	$-\infty$	$+\infty$

Primetimo da ako je $\lambda < 0$, tada interval $[\lambda, +\infty)$ nije zatvoren za operaciju $\otimes$. Naime, za negativne $a, b \in [\lambda, +\infty)$ može se dogoditi da $a \otimes b = a + b < \lambda$, zbog čega je neophodno pomeriti proizvod $a \otimes b = a + b$ za $-\lambda$ kako bi se dobio element iz $[\lambda, +\infty)$.

Ako je $\lambda \geqslant 0$, interval $[\lambda, +\infty)$ jeste zatvoren za operaciju $\otimes$, pa bi se moglo učiniti da nema potrebe za ovim pomeranjem. Međutim, u tom slučaju λ ne bi bila jedinica za operaciju množenja. Zbog toga se, i za $\lambda \geqslant 0$, proizvod $a \otimes b = a + b$ pomera za $-\lambda$, čime se obezbeđuje da nova operacija ima jedinicu λ. Operacija $\otimes$ ostaje nepromenjena na $[\lambda, +\infty)$ samo u slučaju $\lambda = 0$.

Takođe, iako je λ najmanji element intervala $[\lambda, +\infty)$, element $-\infty$ mora biti zadržan kao najmanji element u posmatranoj strukturi, odnosno kao nula poluprstena. U suprotnom, bez elementa $-\infty$, λ bi istovremeno bila nula i jedinica, što je za poluprsten moguće samo kada on ima jedan element.

Posmatrajmo skup $\mathbb{R}^{\lambda}_{\max}$, sa operacijama sabiranja i množenja istovetnim kao u $\mathbb{R}^{\lambda}_{\infty}$. $\mathbb{R}^{\lambda}_{\max}$ u tom slučaju predstavlja komutativan aditivno-idempotentan poluprsten sa nulom $-\infty$ i jedinicom λ. Međutim, ovaj poluprsten nije kompletan, jer nema najveći element. Zbog toga operacija reziduacije nije definisana za svaki par elemenata iz $\mathbb{R}^{\lambda}_{\max}$, budući da nejednačina $a \otimes_{\lambda} x \leqslant b$ nema najveće rešenje $\mathbb{R}^{\lambda}_{\max}$ za sve $a, b \in \mathbb{R}^{\lambda}_{\max}$. Preciznije, to se događa kada je $a = -\infty$, iako je tada svaki element iz $\mathbb{R}^{\lambda}_{\max}$ rešenje navedene nejednačine.

6.2 Gornje odsecanje i pomeranje težina na max-plus poluprstenu

Za proizvoljno $\upsilon \in \mathbb{R}$ definišemo skupove

$$\mathbb{R}^{\upsilon}_{\max} = (-\infty, \upsilon] \cup \{-\infty\} \qquad \text{i} \qquad \mathbb{R}^{\upsilon}_{\infty} = (-\infty, \upsilon] \cup \{-\infty, +\infty\}.$$

Na skupovima $\mathbb{R}^{\upsilon}_{\infty}$ i $\mathbb{R}^{\upsilon}_{\max}$ posmatramo restrikcije poretka definisanog na $\mathbb{R}_{\infty}$, koje ćemo, kao i originalni poredak, označavati sa $\leqslant$.

Teorema 6.2 *Za proizvoljno $\upsilon \in \mathbb{R}$, neka je $\oplus_{\upsilon}$ operacija sabiranja na $\mathbb{R}^{\upsilon}_{\max}$ nasleđena od operacije sabiranja na $\mathbb{R}_{\max}$, a neka je $\otimes_{\upsilon}$ operacija množenja na $\mathbb{R}^{\upsilon}_{\max}$ definisana sledećom tabelom:*

$\otimes_{\upsilon}$	$b \in (-\infty, \upsilon]$	$-\infty$
$a \in (-\infty, \upsilon]$	$a + b - \upsilon$	$-\infty$
$-\infty$	$-\infty$	$-\infty$

Tada je $(\mathbb{R}^{\upsilon}_{\max}, \oplus_{\upsilon}, \otimes_{\upsilon}, -\infty, \upsilon)$ komutativan kompletan aditivno-idempotentan poluprsten sa nulom $-\infty$ i jedinicom υ, a operacija reziduacije $\rightarrow_{\upsilon}$ na $\mathbb{R}^{\upsilon}_{\max}$ pridružena operaciji množenja $\otimes_{\upsilon}$ data je sledećom tabelom:

	$b \in (-\infty, \upsilon]$		
$\rightarrow_{\upsilon}$	$a \geqslant b$	$a < b$	$-\infty$
$a \in (-\infty, \upsilon]$	$b - a + \upsilon$	υ	$-\infty$
$-\infty$	υ	υ	υ

Dokaz 6.2 *Ovu teoremu možemo dokazati na istovetan način kao i teoremu 6.1, te će njen dokaz biti izostavljen.* □

Poluprsten $\mathbb{R}^{\upsilon}_{\max}$ nazivamo ***gornjim υ-odsecanjem*** poluprstena $\mathbb{R}_{\max}$, pri čemu je operacija množenja pomerena za $-\upsilon$.

Za razliku od poluprstena $\mathbb{R}_{\infty}^{\lambda}$, kod kojeg je bilo neophodno zadržati $-\infty$ kao najmanji element, pri konstrukciji poluprstena $\mathbb{R}_{\max}^{\upsilon}$ ne postoji opasnost da se izostavljanjem elementa $+\infty$ dobije struktura u kojoj bi υ istovremeno predstavljala jedinicu i najveći element.

Ovako definisano $\mathbb{R}_{\max}^{\upsilon}$ predstavlja kompletnu reziduiranu mrežu, odnosno komutativni kvantal čija se jedinica podudara sa najvećim elementom. Element $+\infty$ možemo zadržati i na skupu $\mathbb{R}_{\infty}^{\upsilon}$, definisati operaciju množenja na analogan način kao za $\mathbb{R}_{\infty}^{\lambda}$, što ćemo formulisati u sledećoj teoremi.

Teorema 6.3 *Za proizvoljno $\upsilon \in \mathbb{R}$, neka je $\oplus_{\upsilon}$ operacija sabiranja na $\mathbb{R}_{\infty}^{\upsilon}$ nasleđena od operacije sabiranja na $\mathbb{R}_{\infty}$, a neka je $\otimes_{\upsilon}$ operacija množenja na $\mathbb{R}_{\infty}^{\upsilon}$ definisana sledećom tabelom:*

$\otimes_{\upsilon}$	$b \in (-\infty, \upsilon]$	$-\infty$	$+\infty$
$a \in (-\infty, \upsilon]$	$a + b - \upsilon$	$-\infty$	$+\infty$
$-\infty$	$-\infty$	$-\infty$	$-\infty$
$+\infty$	$+\infty$	$-\infty$	$+\infty$

Tada je $(\mathbb{R}_{\infty}^{\upsilon}, \oplus_{\upsilon}, \otimes_{\upsilon}, -\infty, \upsilon)$ komutativan kompletan aditivno-idempotentan poluprsten sa nulom $-\infty$ i jedinicom υ, a operacija reziduacije $\rightarrow_{\upsilon}$ na $\mathbb{R}_{\infty}^{\upsilon}$ pridružena operaciji množenja $\otimes_{\upsilon}$ data je sledećom tabelom:

	$b \in (-\infty, \upsilon]$			
$\rightarrow_{\upsilon}$	$a \geqslant b$	$a < b$	$-\infty$	$+\infty$
$a \in (-\infty, \upsilon]$	$b - a + \upsilon$	υ	$-\infty$	$+\infty$
$-\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$
$+\infty$	$-\infty$	$-\infty$	$-\infty$	$+\infty$

Dokaz 6.3 *Ovu teoremu takođe možemo dokazati na sličan način kao Teoremu 6.1, te će njen dokaz biti izostavljen.* □

Poluprsten $\mathbb{R}_{\infty}^{\upsilon}$ nazivamo ***gornjim υ-odsecanjem*** poluprstena $\mathbb{R}_{\infty}$, pri čemu je operacija množenja pomerena za $-\upsilon$.

Radi jednostavnije terminologije, poluprsten $\mathbb{R}_{\max}^{\upsilon}$ posmatraćemo ne samo kao gornje υ-odsecanje poluprstena $\mathbb{R}_{\max}$, već i kao gornje υ-odsecanje poluprstena $\mathbb{R}_{\infty}$.

Za proizvoljno $\upsilon \in \mathbb{Z}$, skupovi

$$\mathbb{Z}_{\infty}^{\upsilon} = \{z \in \mathbb{Z} \mid z \leqslant \upsilon\} \cup \{-\infty, +\infty\} \quad i \quad \mathbb{Z}_{\max}^{\upsilon} = \{z \in \mathbb{Z} \mid z \leqslant \upsilon\} \cup \{-\infty\}$$

predstavljaju podskupove skupova $\mathbb{R}_{\infty}^{\upsilon}$, odnosno $\mathbb{R}_{\max}^{\upsilon}$, zatvorene za operacije $\oplus_{\upsilon}$, $\otimes_{\upsilon}$ i odgovarajuće operacije reziduacije, kao i za beskonačne sume.

Prema tome, $\mathbb{Z}^{\upsilon}_{\infty} = (\mathbb{Z}^{\upsilon}_{\infty}, \oplus_{\upsilon}, \otimes_{\upsilon}, -\infty, \upsilon)$ i $\mathbb{Z}^{\upsilon}_{\max} = (\mathbb{Z}^{\upsilon}_{\max}, \oplus_{\upsilon}, \otimes_{\upsilon}, -\infty, \upsilon)$ predstavljaju komutativne kompletne aditivno-idempotentne poluprstene.

Iako su operacije množenja na $\mathbb{R}^{\lambda}_{\infty}$ i $\mathbb{R}^{\upsilon}_{\infty}$ definisane na analogan način, a operacija množenja na $\mathbb{R}^{\upsilon}_{\max}$ predstavlja restrikciju operacije množenja na $\mathbb{R}^{\upsilon}_{\infty}$, Teoreme 6.1, 6.2 i 6.3 pokazuju da se odgovarajuće operacije reziduacije na $\mathbb{R}^{\lambda}_{\infty}$, $\mathbb{R}^{\upsilon}_{\infty}$ i $\mathbb{R}^{\upsilon}_{\max}$ bitno razlikuju.

Na kraju ove glave, u tabeli 9, dat je pregled svih razmatranih max-plus struktura.

6.3 Odsecanje i pomeranje težina na max-plus automatima

Pre razmatranja max-plus automata nad odsečenim max-plus poluprstenovima, potrebno je precizirati način množenja matrica nad ovim strukturama. U tu svrhu najpre uvodimo oznaku za multiplikativne stepene elemenata poluprstena $\mathbb{R}_{\infty}$. Za proizvoljno $r \in \mathbb{R}$ i $k \in \mathbb{N}_0$, k-ti multiplikativni stepen elementa r u poluprstenu $\mathbb{R}_{\infty}$ označavaćemo sa $r^{[k]}$, kako bismo ga razlikovali od k-tog multiplikativnog stepena elementa $r \in \mathbb{R}$.

Teorema 6.4 *Za proizvoljno $\xi \in \mathbb{R}$ neka je $\mathbb{R}^{\xi}_{\infty}$ donje ili gornje ξ-odsecanje poluprstena $\mathbb{R}_{\infty}$, a za proizvoljno $s \in \mathbb{N}$ neka su $M_1, M_2, \ldots, M_s$ matrice čiji su elementi iz $\mathbb{R}^{\xi}_{\infty}$ i za koje postoji proizvod matrica $M_1 \otimes_{\xi} M_2 \otimes_{\xi} \cdots \otimes_{\xi} M_s$. Tada važi*

$$M_1 \otimes_{\xi} M_2 \otimes_{\xi} \cdots \otimes_{\xi} M_s = (-\xi)^{[s-1]} \otimes (M_1 \otimes M_2 \otimes \cdots \otimes M_s), \tag{79}$$

gde $(-\xi)^{[s-1]}$ označava $(s-1)$-vi stepen elementa $-\xi$ u multiplikativnom monoidu $\mathbb{R}^{\xi}_{\infty}$.

Dokaz 6.4 *Najpre ćemo pokazati da za sve $a, b \in \mathbb{R}^{\xi}_{\infty}$ važi $a \otimes_{\xi} b = (-\xi) \otimes a \otimes b$.*

Za $a, b \in [\xi, +\infty)$ biće $a \otimes_{\xi} b = a + b - \xi = (-\xi) + a + b = (-\xi) \otimes a \otimes b$.

U svim ostalim slučajevima važi $a \otimes_{\xi} b = a \otimes b = (-\xi) \otimes a \otimes b$, jer je tada $a \otimes b$ jednako $-\infty$ ili $+\infty$, pri čemu $(-\xi) \otimes (-\infty) = -\infty$ i $(-\xi) \otimes (+\infty) = +\infty$.

Sada tvrdnju teoreme dokazujemo indukcijom po s. Za $s = 1$ jednakost (79) *očigledno važi, jer je $(-\xi)^{[1-1]} \otimes M_1 = (-\xi)^{[0]} \otimes M_1 = 0 \otimes M_1 = M_1$.*

Radi preglednosti, najpre ćemo eksplicitno pokazati slučaj $s = 2$. Neka su $M_1 \in \mathbb{S}^{m \times n}$ i $M_2 \in \mathbb{S}^{n \times p}$. Tada za proizvoljne $i \in [1..m]$ i $k \in [1..p]$ važi

$$\begin{aligned}
(M_1 \otimes_{\xi} M_2)(i,k) &= \bigoplus_{j=1}^{n} \big(M_1(i,j) \otimes_{\xi} M_2(j,k)\big) \\
&= \bigoplus_{j=1}^{n} \big((-\xi) \otimes M_1(i,j) \otimes M_2(j,k)\big) \\
&= (-\xi) \otimes \left(\bigoplus_{j=1}^{n} \big(M_1(i,j) \otimes M_2(j,k)\big)\right) \\
&= (-\xi) \otimes \big((M_1 \otimes M_2)(i,k)\big) = \big((-\xi) \otimes M_1 \otimes M_2\big)(i,k).
\end{aligned}$$

Dakle, $M_1 \otimes_\xi M_2 = (-\xi) \otimes M_1 \otimes M_2$*, čime je dokazana tvrdnja za* $s = 2$*.*

Pretpostavimo sada da tvrdnja važi za neko $s \in \mathbb{N}$*,* $s \geqslant 2$*, odnosno da važi* (79)*. Neka je* M_{s+1} *matrica sa elementima iz* $\mathbb{R}_\infty^\xi$ *takva da proizvod* $M_1 \otimes_\xi M_2 \otimes_\xi \cdots \otimes_\xi M_s \otimes_\xi M_{s+1}$ *postoji. Na osnovu indukcijske pretpostavke i svojstava skalarnog množenja dobijamo*

$$\begin{aligned}
&M_1 \otimes_\xi M_2 \otimes_\xi \cdots \otimes_\xi M_s \otimes_\xi M_{s+1} \\
&\quad = (M_1 \otimes_\xi M_2 \otimes_\xi \cdots \otimes_\xi M_s) \otimes_\xi M_{s+1} \\
&\quad = (-\xi) \otimes \big((M_1 \otimes_\xi M_2 \otimes_\xi \cdots \otimes_\xi M_s) \otimes M_{s+1}\big) \\
&\quad = (-\xi) \otimes \Big(\big((-\xi)^{[s-1]} \otimes (M_1 \otimes M_2 \otimes \cdots \otimes M_s)\big) \otimes M_{s+1}\Big) \\
&\quad = (-\xi) \otimes \big((-\xi)^{[s-1]} \otimes (M_1 \otimes M_2 \otimes \cdots \otimes M_s \otimes M_{s+1})\big) \\
&\quad = \big((-\xi) \otimes (-\xi)^{[s-1]}\big) \otimes (M_1 \otimes M_2 \otimes \cdots \otimes M_s \otimes M_{s+1}) \\
&\quad = (-\xi)^{[s]} \otimes (M_1 \otimes M_2 \otimes \cdots \otimes M_s \otimes M_{s+1}).
\end{aligned}$$

Time je dokazana jednakost (79) *za* $s + 1$*, pa je teorema dokazana.* □

Za max-plus automat $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$, sa $\min(\mathcal{A})$ označavaćemo minimum svih elemenata vektora početnih i završnih težina i osnovnih prelaznih matrica automata $\mathcal{A}$, dok će $\max(\mathcal{A})$ označavati njihov maksimum. Dakle,

$$\begin{aligned}
\min(\mathcal{A}) &= \min\big(\{\sigma^A(i) \mid i \in [1..m]\} \cup \{M_x^A(i,j) \mid i,j \in [1..m],\ x \in X\} \cup \{\tau^A(i) \mid i \in [1..m]\}\big), \\
\max(\mathcal{A}) &= \max\big(\{\sigma^A(i) \mid i \in [1..m]\} \cup \{M_x^A(i,j) \mid i,j \in [1..m],\ x \in X\} \cup \{\tau^A(i) \mid i \in [1..m]\}\big).
\end{aligned}$$

Za dva max-plus automata $\mathcal{A}$ i $\mathcal{B}$ definišemo takođe

$$\min(\mathcal{A},\mathcal{B}) = \min\{\min(\mathcal{A}), \min(\mathcal{B})\} \quad \text{i} \quad \max(\mathcal{A},\mathcal{B}) = \max\{\max(\mathcal{A}), \max(\mathcal{B})\}.$$

Neka je $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ max-plus automat nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$ i neka je $\mathbb{R}_\infty^\lambda$ donje λ-odsecanje poluprstena $\mathbb{R}_\infty$, gde je $\lambda \in \mathbb{R}$ takav da je $\lambda \leqslant \min(\mathcal{A})$. Tada svi elementi prelaznih matrica M_x^A, $x \in X$, kao i vektora početnih i završnih težina σ^A i τ^A, pripadaju skupu $\mathbb{R}_\infty^\lambda$. Max-plus automat nad $\mathbb{R}_\infty^\lambda$ sa istim matricama prelaza M_x^A, $x \in X$, i istim vektorima početnih i završnih težina σ^A i τ^A kao automat $\mathcal{A}$ označavaćemo sa $\mathcal{A}^\lambda$ i nazivati njegovim *donjim* λ*-pomeranjem.*

Analogno, neka je $\mathbb{R}_\infty^\upsilon$ gornje υ-odsecanje poluprstena $\mathbb{R}_\infty$, gde je $\upsilon \in \mathbb{R}$ takav da $\upsilon \geqslant \max(\mathcal{A})$. Tada svi elementi prelaznih matrica M_x^A, $x \in X$, kao i vektora σ^A i τ^A, pripadaju skupu $\mathbb{R}_\infty^\upsilon$. Max-plus automat nad $\mathbb{R}_\infty^\upsilon$ koji zadržava iste prelazne matrice, iste vektore početnih i završnih težina označavaćemo sa $\mathcal{A}^\upsilon$ i nazivati njegovim *gornjim* υ*-pomeranjem.*

Gornje υ-pomeranje $\mathcal{A}^\upsilon$ može se, na isti način, posmatrati i kao automat nad $\mathbb{R}_{\max}^\upsilon$.

U nastavku, kada kažemo da je $\mathcal{A}^\lambda$ donje λ-pomeranje automata $\mathcal{A}$, podrazumevaćemo da je ono dobro definisano, odnosno da važi $\lambda \leqslant \min(\mathcal{A})$. Analogno, kada kažemo da je

$\mathcal{A}^{\upsilon}$ gornje υ-pomeranje automata $\mathcal{A}$, podrazumevaćemo da je dobro definisano, odnosno da važi $\upsilon \geqslant \max(\mathcal{A})$.

Prethodno dobijeni rezultat omogućava da se proizvodi matrica nad odsečenim max-plus poluprstenovima izraze pomoću odgovarajućih proizvoda nad kompletnim max-plus poluprstenom, uz odgovarajući skalarni faktor. Budući da se ponašanje max-plus automata dobija množenjem početnog vektora, prelaznih matrica i završnog vektora, prirodno je ispitati kako je ponašanje automata povezano sa ponašanjem njegovog donjeg, odnosno gornjeg pomeranja. Odgovor na ovo pitanje daje naredna teorema.

Teorema 6.5 *Neka je $\mathcal{A}$ max-plus automat nad kompletnim max-plus poluprstenom $\mathbb{R}_{\infty}$ i neka je $\xi \in \mathbb{R}$. Ako je $\mathcal{A}^{\xi}$ donje ili gornje ξ-pomeranje automata $\mathcal{A}$, tada važi*

$$[\![\mathcal{A}^{\xi}]\!](u) = (-\xi)^{[s+1]} \otimes [\![\mathcal{A}]\!](u), \quad \text{za svaku reč } u \in X^* \text{ dužine } s. \tag{80}$$

Dokaz 6.5 *Dokaz ove teoreme neposredno sledi iz definicije ponašanja težinskih automata i Teoreme 6.4.* □

Teorema 6.6 *Neka su $\mathcal{A}$ i $\mathcal{B}$ max-plus automati nad kompletnim max-plus poluprstenom $\mathbb{R}_{\infty}$ i neka je $\xi \in \mathbb{R}$. Ako su $\mathcal{A}^{\xi}$ i $\mathcal{B}^{\xi}$ donja ili gornja ξ-pomeranja automata $\mathcal{A}$ i $\mathcal{B}$, respektivno, tada važi:*

(a) $[\![\mathcal{A}]\!] \leqslant [\![\mathcal{B}]\!]$ *ako i samo ako* $[\![\mathcal{A}^{\xi}]\!] \leqslant [\![\mathcal{B}^{\xi}]\!]$;

(b) $[\![\mathcal{A}]\!] = [\![\mathcal{B}]\!]$ *ako i samo ako* $[\![\mathcal{A}^{\xi}]\!] = [\![\mathcal{B}^{\xi}]\!]$.

Dokaz 6.6 *Prema* (80)*, za svaku reč $u \in X^*$ dužine s važi $[\![\mathcal{A}^{\xi}]\!](u) = (-\xi)^{[s+1]} \otimes [\![\mathcal{A}]\!](u)$ i analogno za automat $\mathcal{B}$.*

Budući da je $(-\xi)^{[s+1]}$ isti skalarni faktor u oba izraza, iz $[\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u)$ sledi $[\![\mathcal{A}^{\xi}]\!](u) \leqslant [\![\mathcal{B}^{\xi}]\!](u)$.

Obrnuto, iz (80) *dobijamo $[\![\mathcal{A}]\!](u) = (-\xi)^{[s+1]} \otimes [\![\mathcal{A}^{\xi}]\!](u)$, i analogno za $\mathcal{B}$, pa iz $[\![\mathcal{A}^{\xi}]\!](u) \leqslant [\![\mathcal{B}^{\xi}]\!](u)$ sledi $[\![\mathcal{A}]\!](u) \leqslant [\![\mathcal{B}]\!](u)$.*

Time je dokazana tvrdnja (a). Tvrdnja (b) neposredno sledi iz tvrdnje (a). □

6.4 Simulacije i bisimulacije na odsečenim i pomerenim max-plus automatima

Jedan od osnovnih problema koji se javlja pri praktičnoj primeni iterativnog postupka za određivanje najveće forward presimulacije $\hat{U}$ jeste njegovo efikasno zaustavljanje. Naime, ukoliko postoji $s \in \mathbb{N}$ takav da važi $U_s = U_{s+1}$, tada se niz stabilizuje i najveća forward presimulacija jednaka je matrici U_s. Zbog toga je nakon konstrukcije matrice U_{s+1}

na osnovu U_s dovoljno proveriti da li su ove dve matrice jednake. Ukoliko postoji najmanji indeks s za koji je $U_s = U_{s+1}$, postupak se završava i dobija se $\hat{U} = U_s$.

Međutim, u opštem slučaju može se dogoditi da su svi članovi niza različiti i da se iterativni postupak ne završi nakon konačno mnogo koraka. Na primer, za neke $i \in [1..m]$ i $j \in [1..n]$ niz $U_s(i,j)_{s\in\mathbb{N}}$ može biti beskonačan i strogo opadajući, pri čemu njegovi članovi konvergiraju ka $-\infty$. Upravo mogućnost pojave ovakvih nizova pokazuje zašto donja λ-odsecanja mogu biti korisna u izračunavanju najvećih presimulacija i prebisimulacija. Njihovom primenom vrednosti manje od λ svode se na $-\infty$, čime se potencijalno beskonačan iterativni postupak može svesti na konačan.

U nastavku ćemo pokazati da je ovakav pristup naročito pogodan za max-plus automate čiji vektori početnih i završnih težina, kao i prelazne matrice, imaju celobrojne elemente, odnosno za max-plus automate nad $\mathbb{Z}_\infty$.

Upravo u ovom slučaju može se garantovati konačnost iterativnog postupka. Osnovni rezultat koji to omogućava zasniva se na činjenici da donje λ-odsecanje $\mathbb{Z}_\infty^\lambda$, za $\lambda \in \mathbb{Z}$, zadovoljava uslov opadajućih lanaca.

Teorema 6.7 *Za proizvoljno $\lambda \in \mathbb{Z}$ i proizvoljne automate $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ definisane nad max-plus strukturom i donjim λ-odsecanjem $\mathbb{Z}_\infty^\lambda$ poluprstena $\mathbb{Z}_\infty$, niz* (55) *se stabilizuje nakon konačnog broja koraka.*

Dokaz 6.7 *Najpre primetimo da poluprsten $\mathbb{Z}_\infty^\lambda$ zadovoljava uslov opadajućih lanaca, odnosno da je svaki opadajući niz elemenata iz $\mathbb{Z}_\infty^\lambda$ konačan. Stoga je, za svaki par $(i,j) \in [1..m] \times [1..n]$, opadajući niz $\{U_s(i,j)\}_{s\in\mathbb{N}}$ stabilizuje nakon konačno mnogo koraka. Prema tome, postoji najmanji broj $k_{i,j} \in \mathbb{N}$ takav da je $U_{k_{i,j}}(i,j)$ najmanji element niza $\{U_s(i,j)\}_{s\in\mathbb{N}}$. Kako je niz opadajući, važi $U_{k_{i,j}}(i,j) = U_l(i,j)$, za svaki $l \in \mathbb{N}$, $l \geqslant k_{i,j}$.*

Sada neka je $k = \max\{k_{i,j} \mid (i,j) \in [1..m] \times [1..n]\}$.

Tada za svaki $(i,j) \in [1..m] \times [1..n]$ i svaki $l \in \mathbb{N}$, $l \geqslant k$, važi $U_k(i,j) = U_l(i,j)$, pa je $U_k = U_l$, za svaki $l \in \mathbb{N}$, $l \geqslant k$. Dakle, niz (55) *se stabilizuje nakon konačno mnogo koraka, čime je dokazano da je konačan.* □

Primer 6.1 *Neka je dat ulazni alfabet $X = \{x, y\}$ i automati $\mathcal{A} = (2, \sigma^A, \{M_x^A, M_y^A\}, \tau^A)$ i $\mathcal{B} = (3, \sigma^B, \{M_x^B, M_y^B\}, \tau^B)$ nad $\mathbb{Z}_\infty$, dati matricama i vektorima*

$$M_x^A = \begin{bmatrix} 4 & 1 \\ -\infty & 2 \end{bmatrix}, \qquad M_y^A = \begin{bmatrix} -\infty & -\infty \\ -\infty & 1 \end{bmatrix}, \qquad \sigma^A = \begin{bmatrix} -\infty & 2 \end{bmatrix}, \qquad \tau^A = \begin{bmatrix} -\infty \\ 0 \end{bmatrix},$$

$$M_x^B = \begin{bmatrix} -\infty & 4 & -\infty \\ -\infty & 4 & -\infty \\ -\infty & -\infty & 2 \end{bmatrix}, \quad M_y^B = \begin{bmatrix} -\infty & -\infty & 0 \\ -\infty & -\infty & -2 \\ -\infty & -\infty & 1 \end{bmatrix}, \quad \sigma^B = \begin{bmatrix} 1 & -\infty & 2 \end{bmatrix}, \quad \tau^B = \begin{bmatrix} -1 \\ -3 \\ 0 \end{bmatrix}.$$

(a) Računanje najveće forward simulacije nad kompletnim max-plus poluprstenom $\mathbb{Z}_\infty$. Indukcijom po s dobijamo da su članovi niza $\{U_s\}_{s\in\mathbb{N}}$ dati sa

$$U_1 = \begin{bmatrix} +\infty & +\infty & +\infty \\ -1 & -3 & 0 \end{bmatrix}, \qquad U_s = \begin{bmatrix} 0 & 0 & 5-2s \\ -1 & -3 & 0 \end{bmatrix}, \quad s \in \mathbb{N},\ s \geqslant 2.$$

Možemo zaključit da se niz ne stabilizuje nakon konačno mnogo koraka, jer njegov element $U_s(1,3) = 5-2s$ strogo opada i konvergira ka $-\infty$. Njegov infimum je

$$\widehat{U} = \begin{bmatrix} 0 & 0 & -\infty \\ -1 & -3 & 0 \end{bmatrix}.$$

Kao što je pokazano u prethodnim poglavljima, $\widehat{U}$ predstavlja najveću forward presimulaciju između $\mathcal{A}$ i $\mathcal{B}$, a proverom uslova (1-fs) utvrđujemo da predstavlja najveću forward simulaciju između $\mathcal{A}$ i $\mathcal{B}$:

$$\sigma^B \otimes (\widehat{U})^\top = \begin{bmatrix} 1 & -\infty & 2 \end{bmatrix} \otimes \begin{bmatrix} 0 & -1 \\ 0 & -3 \\ -\infty & 0 \end{bmatrix} = \begin{bmatrix} 1 & 2 \end{bmatrix} > \begin{bmatrix} -\infty & 2 \end{bmatrix} = \sigma^A.$$

(b) Računanje najveće forward simulacije nad donjim λ-odsecanjem. Posmatrajmo donje λ-odsecanje $\mathbb{Z}_\infty^\lambda$ poluprstena $\mathbb{Z}_\infty$, za proizvoljno $\lambda \leqslant -3$. Primenom formula iz (55) *dobijamo članove niza $\{U_s^\lambda\}_{s\in\mathbb{N}}$:*

$$U_1^\lambda = \begin{bmatrix} +\infty & +\infty & +\infty \\ -\infty & -\infty & \lambda \end{bmatrix}, \quad U_2^\lambda = \begin{bmatrix} +\infty & +\infty & 1+\lambda \\ -\infty & -\infty & \lambda \end{bmatrix}, \quad U_3^\lambda = U_4^\lambda = \widehat{U}^\lambda = \begin{bmatrix} -\infty & -\infty & -\infty \\ -\infty & -\infty & \lambda \end{bmatrix},$$

gde $\widehat{U}^\lambda$ označava infimum ovog niza. Za razliku od slučaja (a), niz se ovde stabilizuje nakon konačno mnogo koraka. Takođe, zadovoljen je uslov (1-fs) i $\widehat{U}^\lambda$ predstavlja najveću forward simulaciju između donjih λ-pomeranja $\mathcal{A}^\lambda$ i $\mathcal{B}^\lambda$.

$$\sigma^B \otimes_\lambda (\widehat{U}^\lambda)^\top = \begin{bmatrix} 1 & -\infty & 2 \end{bmatrix} \otimes_\lambda \begin{bmatrix} -\infty & -\infty \\ -\infty & -\infty \\ -\infty & \lambda \end{bmatrix} = \begin{bmatrix} -\infty & 2 \end{bmatrix} = \sigma^A.$$

(c) Računanje najveće forward simulacije nad gornjim υ-odsecanjem. Posmatrajmo gornje υ-odsecanje $\mathbb{Z}_\infty^\upsilon$ poluprstena $\mathbb{Z}_\infty$, za $\upsilon = 4$. Ako $\mathcal{A}$ i $\mathcal{B}$ posmatramo kao automate nad $\mathbb{Z}_\infty^\upsilon$, za $\upsilon = 4$ dobijamo beskonačan niz

$$U_1^\upsilon = \begin{bmatrix} +\infty & +\infty & +\infty \\ 3 & 1 & 4 \end{bmatrix}, \qquad U_s^\upsilon = \begin{bmatrix} 4 & 4 & 8-2s \\ 3 & 1 & 4 \end{bmatrix}, \quad s \in \mathbb{N},\ s \geqslant 2, \qquad \widehat{U}^\upsilon = \begin{bmatrix} 4 & 4 & -\infty \\ 3 & 1 & 4 \end{bmatrix}.$$

Lako se proverava da $\widehat{U}^\upsilon$ zadovoljava uslov (1-fs), pa predstavlja najveću forward simulaciju između gornjih υ-pomeranja $\mathcal{A}^\upsilon$ i $\mathcal{B}^\upsilon$. Dakle, u ovom primeru gornje odsecanje ne dovodi do stabilizacije iterativnog postupka nakon konačno mnogo koraka.

(d) Promena vektora početnih težina

Sada promenimo vektore početnih težina automata $\mathcal{A}$ i $\mathcal{B}$ tako da važi

$$\sigma^A = \begin{bmatrix} 1 & -\infty \end{bmatrix}, \qquad \sigma^B = \begin{bmatrix} 1 & -\infty & -\infty \end{bmatrix}.$$

Budući da vektori početnih težina ne utiču na izračunavanje članova niza $\{U_s\}_{s\in\mathbb{N}}$, dobijamo isti niz kao u delu (a), a samim tim i isti infimum $\widehat{U}$. Sada, međutim, važi

$$\sigma^B \otimes (\widehat{U})^\top = \begin{bmatrix} 1 & -\infty & -\infty \end{bmatrix} \otimes \begin{bmatrix} 0 & -1 \\ 0 & -3 \\ -\infty & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} > \begin{bmatrix} 1 & -\infty \end{bmatrix} = \sigma^A.$$

Prema tome, $\widehat{U}$ i dalje zadovoljava uslov (1-fs), pa predstavlja najveću forward simulaciju između $\mathcal{A}$ i $\mathcal{B}$.

S druge strane, pri računanju nad $\mathbb{Z}_\infty^\lambda$ za automate $\mathcal{A}^\lambda$ i $\mathcal{B}^\lambda$ dobija se isti niz kao u delu (b), a njegov infimum je $\widehat{U}^\lambda$. Međutim, $\widehat{U}^\lambda$ sada ne zadovoljava uslov (1-fs), jer

$$\sigma^B \otimes_\lambda (\widehat{U}^\lambda)^\top = \begin{bmatrix} 1 & -\infty & -\infty \end{bmatrix} \otimes_\lambda \begin{bmatrix} -\infty & -\infty \\ -\infty & -\infty \\ -\infty & \lambda \end{bmatrix} = \begin{bmatrix} -\infty & -\infty \end{bmatrix} \not\geq \begin{bmatrix} 1 & -\infty \end{bmatrix} = \sigma^A.$$

Stoga u ovom slučaju između automata $\mathcal{A}^\lambda$ i $\mathcal{B}^\lambda$ ne postoji forward simulacija.

Prethodni primer pokazuje da se, u slučaju kada se najveća forward presimulacija između dva max-plus automata nad $\mathbb{R}_\infty$ ne može izračunati u konačno mnogo koraka, problem može preneti na njihova donja pomeranja, gde se najveća forward presimulacija može dobiti nakon konačno mnogo koraka. Za max-plus automate sa celobrojnim težinama takva konačnost je garantovana Teoremom 6.7. U skladu sa Teoremom 6.6, prelazak na donja pomeranja pri tome omogućava očuvanje relacije inkluzije ponašanja automata.

Međutim, primer ukazuje i na ograničenje ovog pristupa. Modifikovana operacija reziduacije na $\mathbb{R}_\infty^\lambda$ može dovesti do toga da najveća forward presimulacija između donjih pomeranja bude isuviše mala da bi zadovoljila uslov (1-fs). Stoga se može dogoditi da između originalnih automata postoji forward simulacija, dok između njihovih donjih pomeranja forward simulacija ne postoji, iako se najveća forward presimulacija nad donjim odsecanjem može izračunati u konačno mnogo koraka.

Takođe, prethodni primer pokazuje da gornje odsecanje ne daje iste prednosti kao donje odsecanje, jer ne mora dovesti do konačnosti niza (55).

Naredni primer pokazuje da primena donjih odsecanja može biti korisna čak i u situacijama kada se najveća forward presimulacija između dva max-plus automata nad poluprstenom $\mathbb{R}_\infty$ može izračunati u konačno mnogo koraka. Naime, i u takvim slučajevima donje odsecanje može dovesti do efikasnijeg izračunavanja, jer se najveća forward presimulacija može dobiti nakon manjeg broja iteracija.

Primer 6.2 *Za ulazni alfabet* $X = \{x, y\}$*, neka su dati* $\mathcal{A} = (3, \sigma^A, \{M_x^A, M_y^A\}, \tau^A)$ *i* $\mathcal{B} = (2, \sigma^B, \{M_x^B, M_y^B\}, \tau^B)$ *max-plus automati nad poluprstenom* $\mathbb{Z}_\infty$ *zadati sa*

$$M_x^A = \begin{bmatrix} 10 & 3 & 4 \\ 5 & 10 & 3 \\ 4 & 6 & 7 \end{bmatrix}, \qquad M_y^A = \begin{bmatrix} 5 & 6 & 2 \\ 3 & 3 & 4 \\ 7 & 7 & 10 \end{bmatrix}, \qquad \sigma^A = \begin{bmatrix} 5 & 0 & 0 \end{bmatrix}, \qquad \tau^A = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix},$$

$$M_x^B = \begin{bmatrix} 10 & 6 \\ 6 & 7 \end{bmatrix}, \qquad M_y^B = \begin{bmatrix} 6 & 6 \\ 7 & 10 \end{bmatrix}, \qquad \sigma^B = \begin{bmatrix} 5 & 0 \end{bmatrix}, \qquad \tau^B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

(a) Računanje najveće forward simulacije nad poluprstenom $\mathbb{Z}_\infty$*.*
Primenom formula iz (55) *dobijamo članove niza* $\{U_s\}_{s\in\mathbb{N}}$*:*

$$U_1 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \qquad U_2 = \begin{bmatrix} 0 & -3 \\ 0 & -3 \\ -4 & 0 \end{bmatrix}, \qquad U_3 = U_4 = \widehat{U} = \begin{bmatrix} 0 & -4 \\ 0 & -4 \\ -4 & 0 \end{bmatrix}.$$

Pošto matrica $\widehat{U}$ *zadovoljava uslov (1-fs), ona predstavlja najveću forward simulaciju između automata* $\mathcal{A}$ *i* $\mathcal{B}$*.*

(b) Računanje najveće forward simulacije nad donjim λ*-odsecanjem* $\mathbb{Z}_\infty^\lambda$*.*
Za poluprsten $\mathbb{Z}_\infty$*, donje* λ*-odsecanje* $\mathbb{Z}_\infty^\lambda$ *za* $\lambda = 0$*, daje niz* $\{U_s^\lambda\}_{s\in\mathbb{N}}$ *čiji su članovi*

$$U_1^\lambda = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \qquad U_2^\lambda = U_3^\lambda = \widehat{U}^\lambda = \begin{bmatrix} 0 & -\infty \\ 0 & -\infty \\ -\infty & 0 \end{bmatrix}.$$

Matrica $\widehat{U}^\lambda$ *takođe zadovoljava uslov* (1-fs)*, pa predstavlja najveću forward simulaciju između automata* $\mathcal{A}^\lambda$ *i* $\mathcal{B}^\lambda$*. Primećujemo da se u ovom slučaju najveća forward simulacija očuvala nakon primene donjeg odsecanja, dok je broj iteracija potreban za njeno određivanje smanjen.*

Pri izboru donjeg praga λ jedini neophodan uslov jeste da važi $\lambda \leqslant \min(\mathcal{A}, \mathcal{B})$, gde su $\mathcal{A}$ i $\mathcal{B}$ max-plus automati za koje se vrši izračunavanje. Ovaj uslov obezbeđuje da donji λ-pomaci $\mathcal{A}^\lambda$ i $\mathcal{B}^\lambda$ imaju iste vektore početnih i završnih težina, kao i iste osnovne matrice prelaza kao polazni automati, dok se razlike pojavljuju samo u složenim prelaznim matricama i ponašanju automata.

Prirodan izbor jeste $\lambda = \min(\mathcal{A}, \mathcal{B})$. Međutim, iako bi se moglo očekivati da manja vrednost praga dovede do manjeg broja iteracija potrebnih za određivanje najveće forward presimulacije, numerički primeri razmatrani u ovom radu ne ukazuju na postojanje takve pravilnosti.

Kao što je već napomenuto u uvodu, sistem nejednačina (2-fs)–(3-fs), kojim su definisane forward presimulacije, predstavlja poseban slučaj tzv. slabolinearnih sistema (*engl.*

weakly linear systems – WLS) razmatranih u radu [69]. Iterativni postupak za određivanje najveće forward presimulacije predstavlja specijalizaciju opšteg postupka iz [69] za određivanje najvećeg rešenja koje je manje ili jednako zadatoj matrici.

Zajednička karakteristika oba postupka jeste da odgovarajući opadajući niz matrica ne mora da se stabilizuje nakon konačno mnogo iteracija. Međutim, kada se računanja izvode nad donjim λ-odsecanjem $\mathbb{Z}_\infty^\lambda$, gde je $\lambda \in \mathbb{Z}$, Teorema 6.7 garantuje konačnost postupka.

Pored toga, u slučaju kada matrica $U_1 = \tau^A \backslash_\lambda \tau^B$ ne sadrži elemente $+\infty$, može se dobiti i gruba gornja ocena broja iteracija. Naime, svaka komponenta niza $(U_s)_{s\in\mathbb{N}}$ može se smanjiti samo konačno mnogo puta, pa broj iteracija ne prelazi vrednost $mn(d+1)$, gde je $d = \max\{\tau^B(j) - \tau^A(i) \mid (i,j) \in [1..m] \times [1..n],\ \tau^A(i) \leqslant \tau^B(j)\}$.

Određivanje sličnih ocena u slučaju kada matrica U_1 sadrži elemente $+\infty$ znatno je složenije i zahteva posebno istraživanje.

Takođe je prirodno postaviti pitanje da li primena donjih odsecanja utiče na računsku složenost iterativnog postupka za određivanje najveće forward presimulacije.

Neka su $\mathcal{A}$ i $\mathcal{B}$ max-plus automati dimenzija m i n nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$, neka je $k = |X|$ broj slova ulaznog alfabeta i neka je $\lambda \in \mathbb{R}$ takvo da važi $\lambda \leqslant \min(\mathcal{A}, \mathcal{B})$. Posmatrajmo jednu iteraciju postupka u kojoj se matrica U_{s+1} izračunava na osnovu matrice U_s.

Za svako slovo $x \in X$, izračunavanje proizvoda $M_x^B \otimes U_s^\top$ zahteva vreme reda $O(mn^2)$, dok je za određivanje reziduala $(M_x^B \otimes U_s^\top)/M_x^A$ i njegove transponovane matrice potrebno vreme reda $O(m^2 n)$. Prema tome, za izračunavanje matrice $\left((M_x^B \otimes U_s^\top)/M_x^A\right)^\top$ potrebno je vreme reda $O\big(mn(m+n)\big)$.

Kako se ovakav postupak sprovodi za svako slovo iz alfabeta, za određivanje svih matrica koje učestvuju u definiciji matrice U_{s+1} potrebno je vreme reda $O\big(kmn(m+n)\big)$.

Nakon toga ostaje još da se izračuna njihov infimum i infimum sa matricom U_s, što ne menja red veličine ukupne složenosti. Stoga je vremenska složenost jedne iteracije postupka određivanja najveće forward presimulacije jednaka $O\big(kmn(m+n)\big)$.

Ako se najveća forward presimulacija određuje između donjih λ-pomeraja automata $\mathcal{A}^\lambda$ i $\mathcal{B}^\lambda$, tada asimptotska složenost jedne iteracije ostaje nepromenjena. Naime, operacije $\otimes_\lambda$ i $\to_\lambda$ u odnosu na operacije $\otimes$ i $\to$ zahtevaju samo konstantan broj dodatnih aritmetičkih operacija i poređenja, što ne utiče na red veličine vremenske složenosti.

Dakle, primena donjih odsecanja ne smanjuje složenost pojedinačne iteracije, ali može značajno smanjiti ukupan broj iteracija potrebnih za stabilizaciju niza $(U_s)_{s\in\mathbb{N}}$.

Rezultati dobijeni u prethodnim primerima pokazuju da se najveće forward presimulacije i simulacije često mogu efikasnije određivati na odsečenim i pomerenim automatima. Prirodno se postavlja pitanje u kojoj meri se tako dobijene relacije mogu preneti na polazne automate. Sledeća teorema pokazuje da se forward presimulacije i simulacije između

max-plus automata nad poluprstenom $\mathbb{R}_\infty$ mogu neposredno dovesti u vezu sa forward presimulacijama i simulacijama između njihovih donjih i gornjih pomeraja.

Teorema 6.8 *Neka su $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ i $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ max-plus automati nad kompletnim max-plus poluprstenom $\mathbb{R}_\infty$. Neka je $\xi \in \mathbb{R}$, i neka su $\mathcal{A}^\xi$ i $\mathcal{B}^\xi$ odgovarajući donji ili gornji ξ-pomaci automata $\mathcal{A}$ i $\mathcal{B}$. Dalje, neka je U matrica dimenzije $m \times n$ sa elementima iz $\mathbb{R}_\infty^\xi$. Tada važi:*

(a) U je forward presimulacija između $\mathcal{A}^\xi$ i $\mathcal{B}^\xi$ ako i samo ako je $(-\xi) \otimes U$ forward presimulacija između $\mathcal{A}$ i $\mathcal{B}$.

(b) U je forward simulacija između $\mathcal{A}^\xi$ i $\mathcal{B}^\xi$ ako i samo ako je $(-\xi) \otimes U$ forward simulacija između $\mathcal{A}$ i $\mathcal{B}$.

Međutim, čak i kada je U najveća forward presimulacija (respektivno najveća forward simulacija) između $\mathcal{A}^\xi$ i $\mathcal{B}^\xi$, matrica $(-\xi)\otimes U$ ne mora biti najveća forward presimulacija (respektivno najveća forward simulacija) između $\mathcal{A}$ i $\mathcal{B}$.

Dokaz 6.8 *(a) Na osnovu Teoreme 6.4 i skalarnog množenja, za proizvoljno $x \in X$ važi*

$$\begin{aligned} U^\top \otimes_\xi M_x^A \leqslant M_x^B \otimes_\xi U^\top &\iff (-\xi) \otimes U^\top \otimes M_x^A \leqslant (-\xi) \otimes M_x^B \otimes U^\top \\ &\iff \big((-\xi) \otimes U^\top\big) \otimes M_x^A \leqslant M_x^B \otimes \big((-\xi) \otimes U^\top\big) \\ &\iff \big((-\xi) \otimes U\big)^\top \otimes M_x^A \leqslant M_x^B \otimes \big((-\xi) \otimes U\big)^\top. \end{aligned}$$

Slično,

$$\begin{aligned} U^\top \otimes_\xi \tau^A \leqslant \tau^B &\iff (-\xi) \otimes U^\top \otimes \tau^A \leqslant \tau^B \\ &\iff \big((-\xi) \otimes U^\top\big) \otimes \tau^A \leqslant \tau^B \\ &\iff \big((-\xi) \otimes U\big)^\top \otimes \tau^A \leqslant \tau^B. \end{aligned}$$

Prema tome, uslovi (2-fs) i (3-fs) važe za matricu U ako i samo ako važe za matricu $(-\xi) \otimes U$, pa zaključujemo da je U forward presimulacija između automata $\mathcal{A}^\xi$ i $\mathcal{B}^\xi$ ako i samo ako je $(-\xi) \otimes U$ forward presimulacija između automata $\mathcal{A}$ i $\mathcal{B}$.

Posmatrajmo automate iz Primera 6.1 i neka je $\xi = \lambda$. U delu (b) tog primera pokazano je da je najveća forward presimulacija između automata $\mathcal{A}^\lambda$ i $\mathcal{B}^\lambda$ jednaka U, te je:

$$(-\lambda) \otimes U = (-\lambda) \otimes \begin{bmatrix} -\infty & -\infty & -\infty \\ -\infty & -\infty & \lambda \end{bmatrix} = \begin{bmatrix} -\infty & -\infty & -\infty \\ -\infty & -\infty & 0 \end{bmatrix},$$

Prema delu (a) istog primera, najveća forward presimulacija između automata $\mathcal{A}$ i $\mathcal{B}$ data matricom $\widehat{U}$, pa važi:

$$(-\lambda) \otimes U = \begin{bmatrix} -\infty & -\infty & -\infty \\ -\infty & -\infty & 0 \end{bmatrix} < \begin{bmatrix} 0 & 0 & -\infty \\ -1 & -3 & 0 \end{bmatrix} = \widehat{U},$$

Odavde zaključujemo da $(-\lambda) \otimes U$ *nije najveća forward presimulacija između* $\mathcal{A}$ *i* $\mathcal{B}$.

(b) Forward simulacije su forward presimulacije koje dodatno zadovoljavaju uslov (1-fs). Ponovo koristeći Teoremu 6.4, dobijamo

$$\begin{aligned} \sigma^A \leqslant \sigma^B \otimes_\xi U^\top &\iff \sigma^A \leqslant (-\xi) \otimes \sigma^B \otimes U^\top \\ &\iff \sigma^A \leqslant \sigma^B \otimes \big((-\xi) \otimes U^\top\big) \\ &\iff \sigma^A \leqslant \sigma^B \otimes \big((-\xi) \otimes U\big)^\top. \end{aligned}$$

Zajedno sa delom (a), ovo pokazuje da je U forward simulacija između automata $\mathcal{A}^\xi$ *i* $\mathcal{B}^\xi$ *ako i samo ako je* $(-\xi) \otimes U$ *forward simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$.

Poslednja tvrdnja sledi ponovo id Primera 6.1 gde je najveća forward presimulacija istovremeno i najveća forward simulacija. □

Poslednja teorema u ovoj sekciji daje dovoljne uslove pod kojima se najveća forward presimulacija između donjih λ-pomaka automata može preneti u najveću forward presimulaciju između polaznih automata. Preciznije, pokazujemo da u slučaju kada najveća forward presimulacija između automata $\mathcal{A}^\lambda$ i $\mathcal{B}^\lambda$ nema elemente jednake $-\infty$, njeno pomeranje za $-\lambda$ daje najveću forward presimulaciju između automata $\mathcal{A}$ i $\mathcal{B}$.

Teorema 6.9 *Neka su* $\mathcal{A} = (m, \sigma^A, \{M_x^A\}_{x\in X}, \tau^A)$ *i* $\mathcal{B} = (n, \sigma^B, \{M_x^B\}_{x\in X}, \tau^B)$ *max-plus automati nad kompletnim max-plus poluprstenom* $\mathbb{R}_\infty$. *Neka su* $\mathcal{A}^\lambda$ *i* $\mathcal{B}^\lambda$ *donji* λ*-pomaci automata* $\mathcal{A}$ *i* $\mathcal{B}$, *gde je* $\lambda \in \mathbb{R}$ *i* $\lambda \leqslant \min(\mathcal{A}, \mathcal{B})$, *i neka je U matrica dimenzije* $m \times n$ *sa elementima iz* $\mathbb{R}_\infty^\lambda$ *koja nema elemente jednake* $-\infty$. *Tada važi:*

(a) Ako je U najveća forward presimulacija između $\mathcal{A}^\lambda$ *i* $\mathcal{B}^\lambda$, *onda je* $(-\lambda) \otimes U$ *najveća forward presimulacija između* $\mathcal{A}$ *i* $\mathcal{B}$.

(b) Ako je U najveća forward simulacija između $\mathcal{A}^\lambda$ *i* $\mathcal{B}^\lambda$, *onda je* $(-\lambda) \otimes U$ *najveća forward simulacija između* $\mathcal{A}$ *i* $\mathcal{B}$.

Dokaz 6.9 *(a) Neka su* $\widehat{U}$ *i* $\widehat{U}^\lambda$ *najveće forward presimulacije između automata* $\mathcal{A}$ *i* $\mathcal{B}$, *odnosno između njihovih donjih* λ*-pomaka* $\mathcal{A}^\lambda$ *i* $\mathcal{B}^\lambda$, *gde su nizovi* $\{U_s\}_{s\in\mathbb{N}}$ *i* $\{U_s^\lambda\}_{s\in\mathbb{N}}$ *definisani formulom* (55) *nad poluprstenima* $\mathbb{R}_\infty$ *i* $\mathbb{R}_\infty^\lambda$, *respektivno:*

$$\widehat{U} = \bigwedge_{s\in\mathbb{N}} U_s \quad i \quad \widehat{U}^\lambda = \bigwedge_{s\in\mathbb{N}} U_s^\lambda$$

Pretpostavimo da je $U = \widehat{U}^\lambda$ *i da matrica U nema elemente jednake* $-\infty$. *Pošto je niz* $\{U_s^\lambda\}_{s\in\mathbb{N}}$ *opadajući, odatle sledi da ni jedna matrica* U_s^λ *nema elemente jednake* $-\infty$.

Pokazaćemo indukcijom po s da važi $U_s^\lambda = \lambda \otimes U_s$, $s \in \mathbb{N}$.

Za $s = 1$ *važi* $U_1^\lambda = \tau^A \backslash_\lambda \tau^B$.

Po definiciji reziduala u poluprstenu $\mathbb{R}_{\infty}^{\lambda}$ i činjenici da U_1^{λ} nema elemente jednake $-\infty$, neposredno se dobija

$$U_1^{\lambda} = \lambda \otimes (\tau^A \backslash \tau^B) = \lambda \otimes U_1.$$

Pretpostavimo sada da važi $U_s^{\lambda} = \lambda \otimes U_s$.

Koristeći definiciju niza (55)*, indukcionu pretpostavku i činjenicu da su sve vrednosti matrica U_s^{λ} veće ili jednake od λ, iz Teoreme 6.4 i osobina operacija $\otimes_{\lambda}$ i $\backslash_{\lambda}$ sledi da se skalarni faktor λ može "izvući"kroz sve operacije koje se pojavljuju u formuli za U_{s+1}^{λ}:*

$$U_{s+1}^{\lambda} = \lambda \otimes U_{s+1}.$$

Time je indukcija završena, pa važi $U_s^{\lambda} = \lambda \otimes U_s,\ s \in \mathbb{N}$.

Kako je množenje skalarom distributivno u odnosu na infimum, dobijamo

$$\widehat{U}^{\lambda} = \bigwedge_{s \in \mathbb{N}} (\lambda \otimes U_s) = \lambda \otimes \left(\bigwedge_{s \in \mathbb{N}} U_s \right) = \lambda \otimes \widehat{U}.$$

Odavde sledi da je $(-\lambda) \otimes U$ najveća forward presimulacija između $\mathcal{A}$ i $\mathcal{B}$:

$$\widehat{U} = (-\lambda) \otimes \widehat{U}^{\lambda} = (-\lambda) \otimes U.$$

(b) Neka je U najveća forward simulacija između $\mathcal{A}^{\lambda}$ i $\mathcal{B}^{\lambda}$. Po Teoremi 6.8, matrica $(-\lambda) \otimes U$ jeste forward simulacija između $\mathcal{A}$ i $\mathcal{B}$.

Kako je, prema delu (a), matrica $(-\lambda) \otimes U$ ujedno najveća forward presimulacija između $\mathcal{A}$ i $\mathcal{B}$, ona zadovoljava i uslov (1-fs), pa predstavlja najveću forward simulaciju između $\mathcal{A}$ i $\mathcal{B}$. □

Prethodna teorema pokazuje da, ukoliko najveća forward presimulacija između donjih λ-pomaka max-plus automata $\mathcal{A}$ i $\mathcal{B}$ nema elemente jednake $-\infty$, tada se najveća forward presimulacija između automata $\mathcal{A}$ i $\mathcal{B}$ može dobiti jednostavnim pomeranjem za $-\lambda$.

Posebno, u slučaju max-plus automata nad poluprstenom $\mathbb{Z}_{\infty}$, Teoreme 6.7 i 6.9 zajedno pokazuju da, ukoliko postoji forward presimulacija bez elemenata $-\infty$, tada se najveća forward presimulacija može izračunati u konačnom broju iteracija. Na taj način dobija se blago uopštenje rezultata.

6.5 Slabe simulacije i bisimulacije na odsečenim i pomerenim max-plus automatima

U prethodnim glavama uvedene su slabe simulacije i bisimulacije između max-plus automata, kao i odgovarajuće slabe presimulacije i prebisimulacije. Kako se svi navedeni pojmovi definišu pomoću generalizovanih vektora stanja i njima pridruženih sistema matričnih nejednačina, efekti odsecanja i pomeranja mogu se analizirati već na nivou slabih

forward presimulacija. Dobijeni zaključci se analogno prenose i na ostale tipove slabih simulacija i bisimulacija.

U prethodnim poglavljima pokazano je da se najveća slaba forward presimulacija između max-plus automata $\mathcal{A}$ i $\mathcal{B}$ određuje formulom 55. Ako je matrica $\widehat{U}$ različita od nulte matrice i zadovoljava uslov (1-wfs), tada ona predstavlja najveću slabu forward simulaciju između $\mathcal{A}$ i $\mathcal{B}$. U suprotnom, slaba forward simulacija između ovih automata ne postoji.

Međutim, praktična primena 55 može biti otežana činjenicom da skup reči X^*, pa samim tim i skup parova (τ_u^A, τ_u^B), $u \in X^*$, može biti beskonačan. Zbog toga se pri računanju najveće slabe presimulacije može javiti potreba za generisanjem beskonačno mnogo τ-parova.

Prirodno se postavlja pitanje da li se ovaj problem može prevazići prelaskom na odsečene ili pomerene max-plus automate. Sledeći primer pokazuje da ni donje ni gornje odsecanje ne garantuju konačnost skupa τ-parova, odnosno da se i u tim slučajevima može dobiti beskonačna familija takvih parova.

Primer 6.3 *Neka je $X = \{x\}$ i neka su dati max-plus automati $\mathcal{A} = (2, \sigma^A, M_x^A, \tau^A)$ i $\mathcal{B} = (3, \sigma^B, M_x^B, \tau^B)$ nad poluprstenom $\mathbb{Z}_\infty$, određeni sa*

$$M_x^A = \begin{bmatrix} 4 & -\infty \\ -\infty & 2 \end{bmatrix}, \qquad \sigma^A = \begin{bmatrix} -\infty & 2 \end{bmatrix}, \qquad \tau^A = \begin{bmatrix} -\infty \\ 0 \end{bmatrix},$$

$$M_x^B = \begin{bmatrix} -\infty & -\infty & 1 \\ -\infty & 4 & -\infty \\ 1 & -\infty & 2 \end{bmatrix}, \qquad \sigma^B = \begin{bmatrix} 1 & -\infty & 2 \end{bmatrix}, \qquad \tau^B = \begin{bmatrix} 1 \\ -3 \\ 0 \end{bmatrix}.$$

(a) Posmatrajmo najpre izračunavanje najveće slabe forward simulacije nad kompletnim max-plus poluprstenom $\mathbb{Z}_\infty$.

Za svako $k \in \mathbb{N}_0$ dobijamo

$$(\tau_{x^k}^A, \tau_{x^k}^B) = \left(\begin{bmatrix} -\infty \\ 2k \end{bmatrix}, \begin{bmatrix} 2k-1 \\ 4k-3 \\ 2k \end{bmatrix} \right), \quad \tau_{x^k}^A \backslash \tau_{x^k}^B = \begin{bmatrix} +\infty & +\infty & +\infty \\ -1 & 2k-3 & 0 \end{bmatrix}.$$

Pošto je niz $(2k-3)_{k\in\mathbb{N}_0}$ rastući, infimum se dostiže već za $k = 0$, te je najveća slaba forward presimulacija

$$\widehat{U} = \begin{bmatrix} +\infty & +\infty & +\infty \\ -1 & -3 & 0 \end{bmatrix}.$$

Kako matrica $\widehat{U}$ zadovoljava uslov (1-wfs), ona predstavlja i najveću slabu forward simulaciju između automata $\mathcal{A}$ i $\mathcal{B}$.

(b) Neka je sada $\lambda = -3$ i posmatrajmo donje odsecanje $\mathbb{Z}_\infty^\lambda$.

Tada za svako $k \in \mathbb{N}_0$ važi

$$(\tau_{x^k}^A, \tau_{x^k}^B) = \left(\begin{bmatrix} -\infty \\ 5k \end{bmatrix}, \begin{bmatrix} 5k-1 \\ 7k-3 \\ 5k \end{bmatrix} \right).$$

Odgovarajući reziduali imaju oblik

$$\tau_\varepsilon^A \backslash_\lambda \tau_\varepsilon^B = \begin{bmatrix} +\infty & +\infty & +\infty \\ -1 & -\infty & -3 \end{bmatrix}, \quad \tau_x^A \backslash_\lambda \tau_x^B = \begin{bmatrix} +\infty & +\infty & +\infty \\ -\infty & -\infty & -3 \end{bmatrix}, \quad \tau_{x^k}^A \backslash_\lambda \tau_{x^k}^B = \begin{bmatrix} +\infty & +\infty & +\infty \\ -\infty & 2k-6 & -3 \end{bmatrix}, k \geq 2.$$

Otuda sledi da je najveća slaba forward presimulacija

$$\widehat{U}^\lambda = \begin{bmatrix} +\infty & +\infty & +\infty \\ -\infty & -\infty & -3 \end{bmatrix},$$

a pošto zadovoljava uslov (1-wfs), ona je i najveća slaba forward simulacija između automata $\mathcal{A}^\lambda$ i $\mathcal{B}^\lambda$.

(c) Neka je $\upsilon = 4$ i posmatrajmo gornje odsecanje $\mathbb{Z}_\infty^\upsilon$. Tada dobijamo

$$(\tau_{x^k}^A, \tau_{x^k}^B) = \left(\begin{bmatrix} -\infty \\ -2k \end{bmatrix}, \begin{bmatrix} -2k-1 \\ -3 \\ -2k \end{bmatrix} \right), \qquad k \in \mathbb{N}_0.$$

Reziduali su dati sa

$$\tau_\varepsilon^A \backslash_\upsilon \tau_\varepsilon^B = \begin{bmatrix} +\infty & +\infty & +\infty \\ 4 & 1 & 4 \end{bmatrix}, \tau_x^A \backslash_\upsilon \tau_x^B = \begin{bmatrix} +\infty & +\infty & +\infty \\ 3 & 3 & 4 \end{bmatrix}, \tau_{x^k}^A \backslash_\upsilon \tau_{x^k}^B = \begin{bmatrix} +\infty & +\infty & +\infty \\ 3 & 4 & 4 \end{bmatrix}, k \geq 2.$$

Prema tome,

$$\widehat{U}^\upsilon = \begin{bmatrix} +\infty & +\infty & +\infty \\ 3 & 1 & 4 \end{bmatrix},$$

predstavlja najveću slabu forward presimulaciju, a ujedno i najveću slabu forward simulaciju između automata $\mathcal{A}^\upsilon$ i $\mathcal{B}^\upsilon$.

Ovaj primer pokazuje da prelazak na donja i gornja odsecanja poluprstena ne mora dovesti do konačnog skupa τ-parova. Drugim rečima, za razliku od običnih simulacija i bisimulacija, odsecanje samo po sebi ne garantuje konačnost postupka izračunavanja slabih simulacija i bisimulacija.

Prethodni primer ukazuje i na jednu dodatnu pojavu. U slučaju izračunavanja nad poluprstenom $\mathbb{Z}_\infty$ infimum se dostiže već nakon prvog koraka, dok se kod donjeg i gornjeg odsecanja dostiže nakon drugog koraka. Nakon toga dalja generisanja τ-parova i

odgovarajućih reziduala ne utiču na vrednost već izračunatog infimuma. Drugim rečima, iako skup τ-parova može biti beskonačan, najveća slaba forward presimulacija može biti određena nakon konačno mnogo koraka.

Ovo otvara pitanje postojanja kriterijuma za zaustavljanje postupka izračunavanja koji bi omogućio da se prepozna trenutak kada je infimum dostignut, čime bi se izbeglo dalje generisanje novih τ-parova. Ispitivanje takvih kriterijuma, kao i konstrukcija novih tipova odsecanja koji bi garantovali konačnost skupova τ-parova, predstavlja pravac budućih istraživanja.

Dobijeni rezultati pokazuju da se uticaj donjih i gornjih odsecanja na slabe simulacije i bisimulacije značajno razlikuje od njihovog uticaja na obične simulacije i bisimulacije. Za razliku od postupaka razmatranih u prethodnim sekcijama, prelazak na odsečene ili pomerene max-plus automate ne garantuje konačnost skupa τ-parova, pa samim tim ni konačnost postupka za izračunavanje najvećih slabih simulacija i bisimulacija. Ipak, prikazani primer ukazuje na mogućnost ranije stabilizacije infimuma kojim se određuje najveća slaba presimulacija, što otvara prostor za razvoj efikasnijih kriterijuma zaustavljanja i predstavlja zanimljiv pravac budućih istraživanja.

Tabela 9: Pregled max-plus struktura i odgovarajućih operacija

Max-plus struktura	**Skup vrednosti**	**Operacija $\oplus$**	**Operacija $\otimes$**	**Operacija reziduacije $\rightarrow$**
Poluprsten $\mathbb{R}_{\max}$	$\mathbb{R}_{\max} = \mathbb{R} \cup \{-\infty\}$	$a \oplus b = \max\{a, b\}$	$a \otimes b = \begin{cases} a+b, & a, b \in \mathbb{R}, \\ -\infty, & a = -\infty \text{ ili } b = -\infty \end{cases}$	Nije definisana
Dioid $\mathbb{P}_{\max}$	$\mathbb{P}_{\max} = \mathbb{R}^+ \cup \{-\infty\}$		$a \otimes b = \begin{cases} a+b, & a, b \in \mathbb{R}^+, \\ -\infty, & a = -\infty \text{ ili } b = -\infty \end{cases}$	Nije definisana
Kompletan poluprsten $\mathbb{R}_{\infty}$	$\mathbb{R}_{\infty} = \mathbb{R} \cup \{-\infty, +\infty\}$		$a \otimes b = \begin{cases} a+b, & a, b \in \mathbb{R}, \\ -\infty, & a = -\infty \text{ ili } b = -\infty, \\ +\infty, & \text{inače} \end{cases}$	$a \rightarrow b = \begin{cases} b-a, & a, b \in \mathbb{R}, \\ +\infty, & a = -\infty \text{ ili } b = +\infty, \\ -\infty, & \text{inače} \end{cases}$
Kompletan dioid $\mathbb{P}_{\infty}$	$\mathbb{P}_{\infty} = \mathbb{R}^+ \cup \{-\infty, +\infty\}$		$a \otimes b = \begin{cases} a+b, & a, b \in \mathbb{R}^+, \\ -\infty, & a = -\infty \text{ ili } b = -\infty, \\ +\infty, & \text{inače} \end{cases}$	$a \rightarrow b = \begin{cases} b-a, & a, b \in \mathbb{R}^+,\ b \geqslant a, \\ +\infty, & a = -\infty \text{ ili } b = +\infty, \\ -\infty, & \text{inače} \end{cases}$
Donje λ-odsecanje $\mathbb{R}_{\max}^{\lambda}$	$\mathbb{R}_{\max}^{\lambda} = [\lambda, +\infty) \cup \{-\infty\}$		$a \otimes_{\lambda} b = \begin{cases} a+b-\lambda, & a, b \in [\lambda, +\infty), \\ -\infty, & a = -\infty \text{ ili } b = -\infty, \\ +\infty, & \text{inače} \end{cases}$	Nije definisana
Donje λ-odsecanje $\mathbb{R}_{\infty}^{\lambda}$	$\mathbb{R}_{\infty}^{\lambda} = [\lambda, +\infty) \cup \{-\infty, +\infty\}$		$a \otimes_{\lambda} b = \begin{cases} a+b-\lambda, & a, b \in [\lambda, +\infty), \\ -\infty, & a = -\infty \text{ ili } b = -\infty, \\ +\infty, & \text{inače} \end{cases}$	$a \rightarrow_{\lambda} b = \begin{cases} b-a+\lambda, & a, b \in [\lambda, +\infty),\ b \geqslant a, \\ +\infty, & a = -\infty \text{ ili } b = +\infty, \\ -\infty, & \text{inače} \end{cases}$
Gornje υ-odsecanje $\mathbb{R}_{\max}^{\upsilon}$	$\mathbb{R}_{\max}^{\upsilon} = (-\infty, \upsilon] \cup \{-\infty\}$		$a \otimes_{\upsilon} b = \begin{cases} a+b-\upsilon, & a, b \in (-\infty, \upsilon], \\ -\infty, & a = -\infty \text{ ili } b = -\infty \end{cases}$	$a \rightarrow_{\upsilon} b = \begin{cases} b-a+\upsilon, & a, b \in (-\infty, \upsilon],\ a \geqslant b, \\ -\infty, & a \in (-\infty, \upsilon],\ b = -\infty, \\ \upsilon, & \text{inače} \end{cases}$
Gornje υ-odsecanje $\mathbb{R}_{\infty}^{\upsilon}$	$\mathbb{R}_{\infty}^{\upsilon} = (-\infty, \upsilon] \cup \{-\infty, +\infty\}$		$a \otimes_{\upsilon} b = \begin{cases} a+b-\upsilon, & a, b \in (-\infty, \upsilon], \\ -\infty, & a = -\infty \text{ ili } b = -\infty, \\ +\infty, & \text{inače} \end{cases}$	$a \rightarrow_{\upsilon} b = \begin{cases} b-a+\upsilon, & a, b \in (-\infty, \upsilon],\ a \geqslant b, \\ \upsilon, & a, b \in (-\infty, \upsilon],\ a < b, \\ +\infty, & a = -\infty \text{ ili } b = +\infty, \\ -\infty, & \text{inače} \end{cases}$

Glava 7

Aproksimacije fazi automata nad proizvod strukturom

U prethodnim glavama razmatrane su simulacije i bisimulacije za težinske automate, kao i njihove slabe varijante, pri čemu je posebna pažnja posvećena postupcima za njihovo određivanje i problemima koji nastaju usled mogućnosti pojave beskonačnih sistema matričnih nejednačina. U prethodnoj glavi razmatrana je i primena odsecanja max-plus poluprstena kao načina za ograničavanje ovih problema.

U ovoj glavi navedeni koncepti biće primenjeni na fazi automate nad Goguenovom, odnosno proizvod strukturom. Oslanjaćemo se na osnovne pojmove o težinskim automatima uvedene u Sekcijama 2.2 Težinski automati i 2.4.2 Fazi automati.

Za fazi automate nad proizvod strukturom određivanje slabih simulacija i bisimulacija može biti otežano zbog mogućnosti pojave beskonačno mnogo matričnih nejednačina [83]. Ovaj problem je povezan sa svojstvom lokalne konačnosti odgovarajuće algebarske strukture. Algebra se naziva *lokalno konačnom* ukoliko je svaka njena konačno generisana podalgebra konačna. Analogno tome, poluprsten je lokalno konačan ako je svaki njegov konačno generisani podpoluprsten konačan. Intuitivno, lokalna konačnost znači da se iz konačnog skupa elemenata primenom operacija posmatrane strukture može dobiti samo konačno mnogo različitih elemenata.

Za razliku od Gödelove i Łukasiewiczeve strukture, čiji su poluprstenski ostaci lokalno konačni, poluprstenski ostatak proizvod strukture nije lokalno konačan. Ova razlika ima neposredan uticaj na razmatranje slabih simulacija i bisimulacija. U slučaju proizvod strukture iz konačnog skupa početnih elemenata može nastati beskonačno mnogo različitih elemenata, pa odgovarajući sistem matričnih nejednačina može sadržati beskonačno mnogo uslova. Ovakva svojstva proizvod strukture, zajedno sa osnovnim svojstvima fazi relacija i fazi funkcija koja se koriste u teoriji fazi sistema [84, 85, 86], motivišu uvođenje aproksimativnog pristupa kojim bi se beskonačnost skupa relevantnih elemenata ograničila [72, 87]. U nastavku glave ovaj pristup biće ostvaren uvođenjem odsečene proizvod strukture [19, 80].

Osnovna ideja ove glave zasniva se na uvođenju odsečene proizvod strukture. Za proizvoljno $\varepsilon \in (0, 1)$ konstruiše se ε-odsečena proizvod struktura, koja zadržava svojstva kompletne reziduirane mreže, dok njen poluprstenovski ostatak postaje lokalno konačan. Na taj način svaki fazi automat nad proizvod strukturom može se zameniti odgovarajućom

ε-kopijom definisanom nad odsečenom strukturom, čime se beskonačni sistemi matričnih nejednačina koji određuju slabe simulacije i bisimulacije svode na konačne sisteme.

Na osnovu ove konstrukcije uvode se pojmovi ε-slabih simulacija i ε-slabih bisimulacija između fazi automata nad proizvod strukturom. Pokazuje se da postojanje takvih relacija omogućava aproksimativno poređenje ponašanja automata, pri čemu odstupanje od ekvivalencije, odnosno inkluzije raspoznatih fazi jezika, ne prelazi zadatu vrednost ε.

Pored teorijskih rezultata, u ovoj glavi biće razvijen i algoritam za ispitivanje postojanja ε-slabih simulacija i ε-slabih bisimulacija između fazi automata nad proizvod strukturom, kao i za izračunavanje najvećih relacija tog tipa kada one postoje.

Rezultati predstavljeni u ovoj glavi zasnovani su prvenstveno na koautorskom radu proisteklom iz istraživanja sprovedenog u okviru ove doktorske disertacije [19], uz oslanjanje na prethodne rezultate o slabim simulacijama i bisimulacijama za fazi automate [80].

7.1 Odsečena proizvod struktura

Ideja koja će biti razvijena u ovoj sekciji sastoji se u modifikaciji proizvod strukture uvođenjem praga $\varepsilon \in (0, 1)$. Na taj način dobija se nova kompletna reziduirana mreža, nazvana *ε-odsečena proizvod struktura*, čiji je poluprstenovski ostatak lokalno konačan. Ova osobina omogućava da se sistemi nejednačina kojima su definisane slabe simulacije i bisimulacije svedu na konačne sisteme, čime njihovo određivanje postaje algoritamski izvodljivo. Nakon uvođenja ε-odsečene proizvod strukture, biće definisan pojam ε-odsecanja fazi skupova i fazi relacija, kao i njihove osnovne osobine.

Pođimo od ranije najavljene modifikacije proizvod strukture koja je definisana u Sekciji 2.4.2. Neka je $\varepsilon \in [0, 1)$ proizvoljno izabrana vrednost i posmatrajmo interval $\mathbb{I}_\varepsilon = [\varepsilon, 1]$. Cilj je da se na skupu $\mathbb{I}_\varepsilon$ definišu operacije izvedene iz operacija proizvod strukture tako da se ponovo dobije kompletna reziduirana mreža. Na taj način dobija se nova struktura koja zadržava osnovne osobine proizvod strukture, ali poseduje dodatna svojstva značajna za konstrukciju aproksimativnih simulacija i bisimulacija.

Sledećom teoremom uvodimo operacije množenja i reziduiranja na skupu $\mathbb{I}_\varepsilon$.

Teorema 7.1 *Neka je $\varepsilon \in [0, 1)$. Na skupu $\mathbb{I}_\varepsilon = [\varepsilon, 1]$, za sve $a, b \in \mathbb{I}_\varepsilon$, definišimo binarne operacije $*_\varepsilon$ i $\to_\varepsilon$ sa*

$$a *_\varepsilon b = \begin{cases} a \cdot b, & \text{ako je } a \cdot b \geqslant \varepsilon, \\ \varepsilon, & \text{ako je } a \cdot b < \varepsilon, \end{cases} \qquad a \to_\varepsilon b = \begin{cases} 1, & \text{ako je } a \leqslant b, \\ \dfrac{b}{a}, & \text{ako je } a > b. \end{cases}$$

Tada je algebra $\mathbb{I}_\varepsilon = (\mathbb{I}_\varepsilon, \wedge, \vee, *_\varepsilon, \to_\varepsilon, \varepsilon, 1)$ kompletna reziduirana mreža.

Dokaz 7.1 *Jasno je da je* $(\mathbb{I}_\varepsilon, \wedge, \vee, \varepsilon, 1)$ *kompletna mreža.*

Neka je $\mathbb{J}_\varepsilon = [0, \varepsilon]$. *Skup* $\mathbb{J}_\varepsilon$ *predstavlja ideal polugrupe* $(\mathbb{I}, \cdot)$, *a polugrupa* $(\mathbb{I}_\varepsilon, *_\varepsilon)$ *izomorfna je Risovoj faktor-polugrupi polugrupe* $(\mathbb{I}, \cdot)$ *po idealu* $\mathbb{J}_\varepsilon$. *Otuda sledi da je* $(\mathbb{I}_\varepsilon, *_\varepsilon)$ *komutativna polugrupa sa neutralnim elementom* 1 *i apsorbujućim elementom* ε.

Dalje, pokažimo da je operacija $\to_\varepsilon$ *dobro definisana. Iz* $\varepsilon \leqslant b < a \leqslant 1$ *sledi* $\varepsilon \leqslant b \leqslant \frac{b}{a} < 1$, *pa je* $\frac{b}{a} \in \mathbb{I}_\varepsilon$. *Takođe, očigledno je da je* $\to_\varepsilon$ *restrikcija operacije reziduiranja* $\to$ *na skupu* $\mathbb{I}_\varepsilon \times \mathbb{I}_\varepsilon$.

Preostaje da se pokaže da operacije $*_\varepsilon$ *i* $\to_\varepsilon$ *zadovoljavaju osobinu reziduacije. Neka su* $a, b, c \in \mathbb{I}_\varepsilon$ *proizvoljni.*

Pretpostavimo da važi $a *_\varepsilon b \leqslant c$.

Ako je $b \leqslant c$, *tada je* $b \to_\varepsilon c = 1$, *pa odatle sledi* $a \leqslant 1 = b \to_\varepsilon c$.

Neka je sada $b > c$. *Tada je* $b \to_\varepsilon c = \frac{c}{b}$.

Ako je $a \cdot b \geqslant \varepsilon$, *onda je* $a *_\varepsilon b = a \cdot b \leqslant c$, *odakle dobijamo* $a \leqslant \frac{c}{b} = b \to_\varepsilon c$.

U suprotnom, $a \cdot b < \varepsilon \leqslant c$, *pa važi* $a < \frac{c}{b} = b \to_\varepsilon c$.

Dakle, $a *_\varepsilon b \leqslant c \quad \Longrightarrow \quad a \leqslant b \to_\varepsilon c$.

Obratno, pretpostavimo da je $a \leqslant b \to_\varepsilon c$.

Ako je $a *_\varepsilon b = \varepsilon$, *onda neposredno važi* $a *_\varepsilon b \leqslant c$.

Neka je sada $a *_\varepsilon b \neq \varepsilon$. *Tada je* $a *_\varepsilon b = a \cdot b > \varepsilon$.

Ako je $b \leqslant c$, *dobijamo* $a *_\varepsilon b = a \cdot b \leqslant b \leqslant c$.

Ako je $b > c$, *tada je* $b \to_\varepsilon c = \frac{c}{b}$, *pa iz* $a \leqslant \frac{c}{b}$ *sledi* $a *_\varepsilon b = a \cdot b \leqslant c$.

Prema tome, $a \leqslant b \to_\varepsilon c \quad \Longrightarrow \quad a *_\varepsilon b \leqslant c$.

Pokazali smo da za sve $a, b, c \in \mathbb{I}_\varepsilon$ *važi* $a *_\varepsilon b \leqslant c \iff a \leqslant b \to_\varepsilon c$, *odnosno da je zadovoljena osobina reziduiranja. Stoga je* $(\mathbb{I}_\varepsilon, \wedge, \vee, *_\varepsilon, \to_\varepsilon, \varepsilon, 1)$ *kompletna reziduirana mreža.* □

Kompletnu rezidiranu mrežu $\mathbb{I}_\varepsilon = (\mathbb{I}_\varepsilon, \wedge, \vee, *_\varepsilon, \to_\varepsilon, \varepsilon, 1)$ nazivaćemo ***ε-odsečena proizvod struktura***. Napominjemo da za $\varepsilon = 0$ dobijamo standardnu proizvod strukturu.

Operacija reziduiranja $\to_\varepsilon$ na skupu $\mathbb{I}_\varepsilon$ definisana je na isti način kao operacija $\to$ na skupu $\mathbb{I}$, ali se ove operacije razlikuju po svojim domenima. Kao što je već napomenuto, operacija $\to_\varepsilon$ predstavlja restrikciju operacije $\to$ na skupu $\mathbb{I}_\varepsilon \times \mathbb{I}_\varepsilon$. Radi jasnog razlikovanja koristićemo oznaku sa indeksom ε.

Prema definiciji operacije $\to_\varepsilon$, za svako $a \in \mathbb{I}_\varepsilon$ važi $a \to_\varepsilon 1 = 1$ i $a \to_\varepsilon \varepsilon = \frac{\varepsilon}{a}$. Posebno, $\varepsilon \to_\varepsilon \varepsilon = \varepsilon \to_\varepsilon 1 = 1 \to_\varepsilon 1 = 1$ i $1 \to_\varepsilon \varepsilon = \varepsilon$.

Takođe, za sve $a, b \in \mathbb{I}_\varepsilon$ važi

$$a \to_\varepsilon b = \bigvee \{ x \in \mathbb{I}_\varepsilon \mid x *_\varepsilon a \leqslant b \},$$

odnosno, $a \to_\varepsilon b$ predstavlja najveći element skupa svih $x \in \mathbb{I}_\varepsilon$ za koje je $x *_\varepsilon a \leqslant b$.

Bireziduum na strukturi $\mathbb{I}_\varepsilon$ definiše se sa $a \leftrightarrow_\varepsilon b = (a \rightarrow_\varepsilon b) \wedge (b \rightarrow_\varepsilon a)$, za proizvoljne $a, b \in \mathbb{I}_\varepsilon$.

Naredna teorema daje jednu od najvažnijih osobina ε-odsečene proizvod strukture.

Teorema 7.2 *Poluprsten* $(\mathbb{I}_\varepsilon, \vee, *_\varepsilon, \varepsilon, 1)$ *je lokalno konačan za svako* $\varepsilon \in (0, 1)$.

Dokaz 7.2 *Poznato je da je poluprsten lokalno konačan ako i samo ako su i njegov aditivni i njegov multiplikativni monoid lokalno konačni. Kako je* $(\mathbb{I}_\varepsilon, \vee, \varepsilon)$ *polumreža, a svaka polumreža je lokalno konačna, dovoljno je pokazati da je multiplikativni monoid* $(\mathbb{I}_\varepsilon, *_\varepsilon, 1)$ *lokalno konačan.*

Dalje, monoid $(\mathbb{I}_\varepsilon, *_\varepsilon, 1)$ *je lokalno konačan ako i samo ako je polugrupa* $(\mathbb{I}_\varepsilon \setminus \{1\}, *_\varepsilon)$ *lokalno konačna. Stoga je dovoljno dokazati lokalnu konačnost ove polugrupe.*

Neka je S *proizvoljan konačan podskup skupa* $\mathbb{I}_\varepsilon \setminus \{1\}$. *Pošto je* S *konačan, postoji njegov najveći element* t, *pri čemu je* $t < 1$.

Tada je svaki proizvod od n *elemenata skupa* S *manji ili jednak od* t^n.

Kako je $t < 1$, *postoji prirodan broj* n_0 *takav da važi* $t^{n_0} < \varepsilon$.

Prema definiciji operacije $*_\varepsilon$, *za svako* $n \geqslant n_0$ *proizvod bilo kojih* n *elemenata iz skupa* S *u polugrupi* $(\mathbb{I}_\varepsilon, *_\varepsilon)$ *jednak je elementu* ε.

Odavde sledi da je podpolugrupa generisana skupom S *jednaka*

$$\bigcup_{k\in\mathbb{N}} S^k = \bigcup_{k\leqslant n_0} S^k, \quad \text{gde je } S^k = \{\, a_1 *_\varepsilon \cdots *_\varepsilon a_k \mid a_1, \ldots, a_k \in S \,\}.$$

Pošto je skup S *konačan, svaki skup* S^k *je konačan, pa je i unija* $\bigcup_{k\leqslant n_0} S^k$ *konačna.*

Dakle, svaka konačno generisana podpolugrupa polugrupe $(\mathbb{I}_\varepsilon\setminus\{1\}, *_\varepsilon)$ *je konačna, što znači da je ova polugrupa lokalno konačna. Odatle sledi da je i poluprsten* $(\mathbb{I}_\varepsilon, \vee, *_\varepsilon, \varepsilon, 1)$ *lokalno konačan.* □

Jedan od ključnih pojmova koji uvodimo u ovoj glavi jeste pojam ε-odsecanja fazi skupova i fazi relacija.

Definicija 7.1 Neka je $\varepsilon \in \mathbb{I}$, $\varepsilon < 1$, i neka je $\alpha \in \mathbb{I}^A$ fazi skup na skupu A. Za svako $a \in A$, ***ε-odsecanje*** fazi skupa α definiše se kao fazi skup $\alpha_\varepsilon \in \mathbb{I}_\varepsilon^A$ zadat sa

$$\alpha_\varepsilon(a) = \begin{cases} \alpha(a), & \text{ako je } \alpha(a) > \varepsilon, \\ \varepsilon, & \text{ako je } \alpha(a) \leqslant \varepsilon. \end{cases}$$

Za $\varepsilon = 0$ važi $\mathbb{I}_0 = \mathbb{I}$ i $\alpha_0 = \alpha$.

Analogno, za fazi relaciju $\phi \in \mathbb{I}^{A\times B}$, ***ε-odsecanje*** relacije ϕ definiše se kao ε-odsecanje odgovarajućeg fazi skupa na skupu $A \times B$.

Naredna propozicija daje osnovne osobine ε-odsecanja fazi skupova.

Propozicija 7.1 *Neka je A neprazan skup, neka su $\xi, \eta \in \mathbb{I}^A$ fazi skupovi na skupu A, i neka su $\varepsilon, \varepsilon_1, \varepsilon_2 \in \mathbb{I} \setminus \{1\}$ takvi da je $\varepsilon_1 \leqslant \varepsilon_2$. Tada važe sledeća tvrđenja:*

(i) $\xi \leqslant \xi_\varepsilon$;

(ii) $\xi \leqslant \eta$ *implicira* $\xi_\varepsilon \leqslant \eta_\varepsilon$;

(iii) $(\xi_\varepsilon)_\varepsilon = \xi_\varepsilon$;

(iv) $\xi_{\varepsilon_1} \leqslant \xi_{\varepsilon_2}$;

(v) $(\xi_{\varepsilon_1})_{\varepsilon_2} = (\xi_{\varepsilon_2})_{\varepsilon_1} = \xi_{\varepsilon_2}$.

Dokaz 7.3 *Tvrdnja (i) i jednakost (iii) neposredno slede iz definicije ε-odsecanja.*
(ii) Neka je $\xi \leqslant \eta$ i neka je $a \in A$ proizvoljno.
Ako je $\xi(a) > \varepsilon$, tada je $\xi_\varepsilon(a) = \xi(a) \leqslant \eta(a) \leqslant \eta_\varepsilon(a)$.
Ako je $\xi(a) \leqslant \varepsilon$, tada je $\xi_\varepsilon(a) = \varepsilon \leqslant \eta_\varepsilon(a)$.
Dakle, $\xi_\varepsilon \leqslant \eta_\varepsilon$.
(iv) Neka je $a \in A$ proizvoljno.
Ako je $\xi(a) > \varepsilon_2$, tada je $\xi_{\varepsilon_1}(a) = \xi(a) = \xi_{\varepsilon_2}(a)$.
Ako je $\varepsilon_1 < \xi(a) \leqslant \varepsilon_2$, tada je $\xi_{\varepsilon_1}(a) = \xi(a) \leqslant \varepsilon_2 \leqslant \xi_{\varepsilon_2}(a)$.
Ako je $\xi(a) \leqslant \varepsilon_1$, tada je $\xi_{\varepsilon_1}(a) = \varepsilon_1 \leqslant \varepsilon_2 \leqslant \xi_{\varepsilon_2}(a)$.
Prema tome, $\xi_{\varepsilon_1} \leqslant \xi_{\varepsilon_2}$.
(v) Iz tvrdnji (i) i (iv) sledi $\xi \leqslant \xi_{\varepsilon_1} \leqslant \xi_{\varepsilon_2}$.
Primenom tvrdnje (ii) dobijamo $\xi_{\varepsilon_2} \leqslant (\xi_{\varepsilon_1})_{\varepsilon_2} \leqslant (\xi_{\varepsilon_2})_{\varepsilon_2} = \xi_{\varepsilon_2}$, pa je $(\xi_{\varepsilon_1})_{\varepsilon_2} = \xi_{\varepsilon_2}$.
Dalje, za svako $a \in A$ važi $\xi_{\varepsilon_2}(a) \geqslant \varepsilon_2 \geqslant \varepsilon_1$, pa se novim odsecanjem na nivou ε_1 ništa ne menja. Dakle, $(\xi_{\varepsilon_2})_{\varepsilon_1} = \xi_{\varepsilon_2}$.
Odavde zaključujemo da važi $(\xi_{\varepsilon_1})_{\varepsilon_2} = (\xi_{\varepsilon_2})_{\varepsilon_1} = \xi_{\varepsilon_2}$, čime je dokaz završen. □

Napomenimo da tvrđenja Propozicije 7.1 ostaju važeća i kada su ξ i η fazi relacije na skupu A.

U nastavku ćemo za množenje u proizvod strukturi koristiti uobičajenu oznaku "$\cdot$", dok ćemo za operaciju reziduiranja i sve iz nje izvedene pojmove zadržati oznake uvedene u Glavi 2. Posebno, sve kompozicije fazi relacija, fazi skupova, kao i fazi skupova i fazi relacija označavaćemo simbolom "$\circ$".

Sa druge strane, za odgovarajuće pojmove definisane nad ε-odsečenom proizvod strukturom koristićemo iste oznake sa indeksom ε. Jedini izuzetak predstavlja množenje u strukturi $\mathbb{I}_\varepsilon$, koje označavamo sa "$*_\varepsilon$", u skladu sa Teoremom 7.1. Tako ćemo kompozicije definisane nad $\mathbb{I}_\varepsilon$ označavati sa "$\circ_\varepsilon$", dok ćemo levi rezidual, desni rezidual i birezidual označavati redom sa $\backslash_\varepsilon$, $/_\varepsilon$ i $|_\varepsilon$.

Sledeća propozicija daje osnovne osobine kompozicija fazi relacija i fazi skupova u vezi sa operacijom ε-odsecanja.

Propozicija 7.2 *Neka je $\varepsilon \in \mathbb{I}$, $\varepsilon < 1$, i neka je A neprazan skup. Tada za proizvoljne fazi skupove $\alpha, \beta \in \mathbb{I}^A$ i proizvoljne fazi relacije $\phi, \psi \in \mathbb{I}^{A\times A}$ važi*

$$\phi_\varepsilon \circ_\varepsilon \psi_\varepsilon = (\phi \circ \psi)_\varepsilon, \qquad \alpha_\varepsilon \circ_\varepsilon \phi_\varepsilon = (\alpha \circ \phi)_\varepsilon, \qquad \phi_\varepsilon \circ_\varepsilon \beta_\varepsilon = (\phi \circ \beta)_\varepsilon \tag{81}$$

$$\alpha_\varepsilon \circ_\varepsilon \beta_\varepsilon = \begin{cases} \alpha \circ \beta & \text{ako je } \alpha \circ \beta \geqslant \varepsilon, \\ \varepsilon & \text{u suprotnom.} \end{cases} \tag{82}$$

Dokaz 7.4 *Dokazaćemo samo prvu jednakost, jer se preostale dokazuju na isti način.*

Neka su $a, b \in A$ proizvoljni. Pokazaćemo da za svako $c \in A$ važi

$$\phi_\varepsilon(a,c) *_\varepsilon \psi_\varepsilon(c,b) > \varepsilon \iff \phi(a,c) \cdot \psi(c,b) > \varepsilon.$$

Pretpostavimo najpre da je $\phi_\varepsilon(a,c) *_\varepsilon \psi_\varepsilon(c,b) > \varepsilon$.
Tada je $\phi_\varepsilon(a,c) > \varepsilon$ i $\psi_\varepsilon(c,b) > \varepsilon$, pa važi $\phi_\varepsilon(a,c) = \phi(a,c)$ i $\psi_\varepsilon(c,b) = \psi(c,b)$.
Prema definiciji operacije $*_\varepsilon$ dobijamo

$$\phi(a,c) \cdot \psi(c,b) = \phi(a,c) *_\varepsilon \psi(c,b) = \phi_\varepsilon(a,c) *_\varepsilon \psi_\varepsilon(c,b) > \varepsilon.$$

Obratno, neka je $\phi(a,c) \cdot \psi(c,b) > \varepsilon$.
Tada su i $\phi(a,c)$ i $\psi(c,b)$ veći od ε, pa je $\phi_\varepsilon(a,c) = \phi(a,c)$, $\psi_\varepsilon(c,b) = \psi(c,b)$.
Odatle sledi $\phi_\varepsilon(a,c) *_\varepsilon \psi_\varepsilon(c,b) = \phi(a,c) *_\varepsilon \psi(c,b) = \phi(a,c) \cdot \psi(c,b) > \varepsilon$.
Time je dokazana ekvivalencija, a u slučaju kada su njene obe strane zadovoljene važi i

$$\phi_\varepsilon(a,c) *_\varepsilon \psi_\varepsilon(c,b) = \phi(a,c) \cdot \psi(c,b).$$

Sada definišimo skup $A' = \{\, c \in A \mid \phi_\varepsilon(a,c) *_\varepsilon \psi_\varepsilon(c,b) > \varepsilon \,\}$.

Na osnovu prethodno dokazane ekvivalencije važi i $A' = \{\, c \in A \mid \phi(a,c) \cdot \psi(c,b) > \varepsilon \,\}$.

Ako je $A' \neq \emptyset$, tada dobijamo

$$\begin{aligned}(\phi_\varepsilon \circ_\varepsilon \psi_\varepsilon)(a,b) &= \bigvee_{c\in A} \phi_\varepsilon(a,c) *_\varepsilon \psi_\varepsilon(c,b) = \bigvee_{c\in A'} \phi_\varepsilon(a,c) *_\varepsilon \psi_\varepsilon(c,b) \\ &= \bigvee_{c\in A'} \phi(a,c) \cdot \psi(c,b) = \bigvee_{c\in A} \phi(a,c) \cdot \psi(c,b) \\ &= (\phi \circ \psi)(a,b) = (\phi \circ \psi)_\varepsilon(a,b).\end{aligned}$$

Pri tome je $(\phi \circ \psi)(a,b) > \varepsilon$, pa sledi $(\phi_\varepsilon \circ_\varepsilon \psi_\varepsilon)(a,b) = (\phi \circ \psi)_\varepsilon(a,b)$.

Ako je $A' = \emptyset$, tada je $(\phi_\varepsilon \circ_\varepsilon \psi_\varepsilon)(a,b) = \varepsilon$, dok iz definicije skupa A' sledi da je $(\phi \circ \psi)(a,b) \leqslant \varepsilon$, pa je $(\phi \circ \psi)_\varepsilon(a,b) = \varepsilon$.

Dakle, i u ovom slučaju važi $(\phi_\varepsilon \circ_\varepsilon \psi_\varepsilon)(a,b) = (\phi \circ \psi)_\varepsilon(a,b)$.

Pošto je ovo tačno za sve $a, b \in A$, zaključujemo da je $\phi_\varepsilon \circ_\varepsilon \psi_\varepsilon = (\phi \circ \psi)_\varepsilon$. □

7.2 Fazi automati nad odsečenom proizvod strukturom

Nakon uvođenja odsečene proizvod strukture i operacije ε-odsecanja, prirodno se postavlja pitanje na koji način se fazi automati definisani nad proizvod strukturom mogu posmatrati u okviru odgovarajuće odsečene proizvod strukture. U ovoj sekciji uvodi se pojam *ε-kopije fazi automata*, koji omogućava da se fazi automat nad proizvod strukturom na prirodan način transformiše u fazi automat nad ε-odsečenom proizvod strukturom.

Biće pokazano da se osnovni objekti koji određuju ponašanje automata, kao što su relacije prelaza, dostižna i ko-dostižna stanja, na prirodan način povezuju sa odgovarajućim ε-odsecanjima. Kao posledica toga, dokazaćemo da je jezik koji raspoznaje ε-kopija automata upravo ε-odsecanje jezika polaznog automata. Takođe ćemo pokazati da se ponašanje originalnog automata može aproksimirati proizvoljno precizno izborom dovoljno male vrednosti parametra ε.

Neka je $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ fazi konačni automat nad proizvod strukturom i neka je $\varepsilon \in (0, 1)$. Definišemo novi fazi konačni automat $\mathcal{A}_\varepsilon = (A, \sigma^{A_\varepsilon}, \delta^{A_\varepsilon}, \tau^{A_\varepsilon})$, gde je $A_\varepsilon = A$ i $\sigma^{A_\varepsilon} = (\sigma^A)_\varepsilon, \quad \delta_x^{A_\varepsilon} = (\delta_x^A)_\varepsilon,\ x \in X, \quad \tau^{A_\varepsilon} = (\tau^A)_\varepsilon.$

Jasno je da je $\mathcal{A}_\varepsilon$ fazi konačni automat nad ε-odsečenom proizvod strukturom $\mathbb{I}_\varepsilon$. Automat $\mathcal{A}_\varepsilon$ nazivaćemo ***ε-kopijom automata*** $\mathcal{A}$.

Iako smo do sada dopuštali da ε bude proizvoljan element iz skupa $\mathbb{I}\setminus\{1\}$, u primenama je najprirodnije birati ε kao pozitivan broj koji je manji od svih nenultih vrednosti koje uzimaju funkcije σ^A, δ^A i τ^A. U tom slučaju funkcije σ^A, δ^A i τ^A razlikuju se od funkcija σ^{A_ε}, δ^{A_ε} i τ^{A_ε} jedino u onim tačkama u kojima automat $\mathcal{A}$ ima vrednost 0, dok je u automatu $\mathcal{A}_\varepsilon$ ta vrednost zamenjena brojem ε.

Složenu fazi relaciju prelaza automata $\mathcal{A}_\varepsilon$ određenu rečju $u \in X^*$ označavaćemo sa $\delta_u^{A_\varepsilon}$, dok ćemo odgovarajuće dostižne i ko-dostižne fazi skupove označavati sa $\sigma_u^{A_\varepsilon}$ i $\tau_u^{A_\varepsilon}$.

Ako je $u = x_1 x_2 \cdots x_n$, gde su $x_1, x_2, \ldots, x_n \in X$, tada važi

$$\delta_u^{A_\varepsilon} = \delta_{x_1}^{A_\varepsilon} \circ_\varepsilon \delta_{x_2}^{A_\varepsilon} \circ_\varepsilon \cdots \circ_\varepsilon \delta_{x_n}^{A_\varepsilon}, \quad \sigma_u^{A_\varepsilon} = \sigma^{A_\varepsilon} \circ_\varepsilon \delta_u^{A_\varepsilon}, \quad \tau_u^{A_\varepsilon} = \delta_u^{A_\varepsilon} \circ_\varepsilon \tau^{A_\varepsilon}.$$

Propozicija 7.3 *Neka je $\varepsilon \in (0, 1)$, neka je $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ fazi konačni automat nad proizvod strukturom i neka je $\mathcal{A}_\varepsilon = (A, \sigma^{A_\varepsilon}, \delta^{A_\varepsilon}, \tau^{A_\varepsilon})$ ε-kopija automata $\mathcal{A}$. Tada za svaku reč $u \in X^*$ važi*

$$\delta_u^{A_\varepsilon} = (\delta_u^A)_\varepsilon, \qquad \sigma_u^{A_\varepsilon} = (\sigma_u^A)_\varepsilon, \qquad \tau_u^{A_\varepsilon} = (\tau_u^A)_\varepsilon.$$

Dokaz 7.5 *Najpre dokazujemo prvu jednakost. Dokaz sprovodimo indukcijom po dužini reči u.*

Jasno je da jednakost važi za praznu reč. Takođe, prema definiciji automata $\mathcal{A}_\varepsilon$, ona važi za svako slovo iz skupa X.

Pretpostavimo da za neku reč $u \in X^*$ važi $\delta_u^{A_\varepsilon} = (\delta_u^A)_\varepsilon$, i neka je $x \in X$ proizvoljno. Na osnovu Propozicije 7.2 dobijamo

$$\delta_{ux}^{A_\varepsilon} = \delta_u^{A_\varepsilon} \circ_\varepsilon \delta_x^{A_\varepsilon} = (\delta_u^A)_\varepsilon \circ_\varepsilon (\delta_x^A)_\varepsilon = (\delta_u^A \circ \delta_x^A)_\varepsilon = (\delta_{ux}^A)_\varepsilon.$$

Time je dokazana prva jednakost i ostaje pokazati preostale dve jednakosti.
Neka je sada $u \in X^*$ proizvoljna reč. Ponovo primenom Propozicije 7.2 dobijamo:

$$\sigma_u^{A_\varepsilon} = \sigma^{A_\varepsilon} \circ_\varepsilon \delta_u^{A_\varepsilon} = (\sigma^A)_\varepsilon \circ_\varepsilon (\delta_u^A)_\varepsilon = (\sigma^A \circ \delta_u^A)_\varepsilon = (\sigma_u^A)_\varepsilon.$$

$$\tau_u^{A_\varepsilon} = \delta_u^{A_\varepsilon} \circ_\varepsilon \tau^{A_\varepsilon} = (\delta_u^A)_\varepsilon \circ_\varepsilon (\tau^A)_\varepsilon = (\delta_u^A \circ \tau^A)_\varepsilon = (\tau_u^A)_\varepsilon.$$

Time su dokazane i preostale dve jednakosti, čime je dokaz završen. □

Teorema 7.3 *Neka je $\varepsilon \in (0, 1)$, neka je $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ fazi automat nad proizvod strukturom i neka je $\mathcal{A}_\varepsilon = (A, \sigma^{A_\varepsilon}, \delta^{A_\varepsilon}, \tau^{A_\varepsilon})$ ε-kopija automata $\mathcal{A}$. Tada je ponašanje automata $\mathcal{A}_\varepsilon$ jednako ε-odsecanju ponašanja automata $\mathcal{A}$, odnosno $[\![\mathcal{A}_\varepsilon]\!] = [\![\mathcal{A}]\!]_\varepsilon$.*

Dokaz 7.6 *Neka je $u \in X^+$ proizvoljna neprazna reč. Na osnovu Propozicija 7.2 i 7.3 dobijamo*

$$\begin{aligned}
[\![\mathcal{A}_\varepsilon]\!](u) &= \sigma^{A_\varepsilon} \circ_\varepsilon \delta_u^{A_\varepsilon} \circ_\varepsilon \tau^{A_\varepsilon} = \sigma_u^{A_\varepsilon} \circ_\varepsilon \tau^{A_\varepsilon} = (\sigma_u^A)_\varepsilon \circ_\varepsilon (\tau^A)_\varepsilon \\
&= \begin{cases} \sigma_u^A \circ \tau^A & \text{ako je } \ \sigma_u^A \circ \tau^A \geq \varepsilon \\ \varepsilon & \text{u suprotnom} \end{cases} \\
&= \begin{cases} [\![\mathcal{A}]\!](u) & \text{ako je } \ [\![\mathcal{A}]\!](u) \geq \varepsilon \\ \varepsilon & \text{u suprotnom} \end{cases} = [\![\mathcal{A}]\!]_\varepsilon(u).
\end{aligned}$$

Na isti način za praznu reč e dobijamo

$$[\![\mathcal{A}_\varepsilon]\!](e) = \sigma^{A_\varepsilon} \circ_\varepsilon \tau^{A_\varepsilon} = (\sigma^A \circ \tau^A)_\varepsilon = ([\![\mathcal{A}]\!](e))_\varepsilon.$$

Stoga, za svaku reč $u \in X^*$ važi $[\![\mathcal{A}_\varepsilon]\!](u) = ([\![\mathcal{A}]\!](u))_\varepsilon$, odnosno $[\![\mathcal{A}_\varepsilon]\!] = ([\![\mathcal{A}]\!])_\varepsilon$. □

Radi poređenja ponašanja fazi automata i njegove ε-kopije koristićemo meru različitosti definisanu na sledeći način. Neka je A neprazan skup. *Łukašjevičeva mera različitosti* je preslikavanje definisano sa

$$\mathrm{Diss} : \mathbb{I}^A \times \mathbb{I}^A \to \mathbb{I}, \quad \mathrm{Diss}(\alpha, \beta) = \bigvee_{a \in A} |\alpha(a) - \beta(a)|,$$

za proizvoljne fazi skupove $\alpha, \beta \in \mathbb{I}^A$.

Ova mera izražava maksimalno odstupanje između fazi skupova α i β. Naziv *Łukašjevičeva mera različitosti* potiče od dualno definisane mere sličnosti indukovane Łukašjevičevom operacijom reziduacije (videti [88]).

Teorema 7.4 *Neka je $\varepsilon \in (0,1)$, neka je $\mathcal{A}$ fazi automat nad proizvod strukturom i neka je $\mathcal{A}_\varepsilon$ ε-kopija automata $\mathcal{A}$. Tada važi* $\mathrm{Diss}(\llbracket \mathcal{A} \rrbracket, \llbracket \mathcal{A}_\varepsilon \rrbracket) \leqslant \varepsilon$.

Dokaz 7.7 *Neka je $u \in X^*$ proizvoljna reč.*
Ako je $\llbracket \mathcal{A} \rrbracket(u) \geqslant \varepsilon$, onda prema Teoremi 7.3 važi

$$\llbracket \mathcal{A}_\varepsilon \rrbracket(u) = \llbracket \mathcal{A} \rrbracket(u) \text{ , pa je } \quad \left| \llbracket \mathcal{A} \rrbracket(u) - \llbracket \mathcal{A}_\varepsilon \rrbracket(u) \right| = 0.$$

Ako je $\llbracket \mathcal{A} \rrbracket(u) < \varepsilon$, tada je

$$\llbracket \mathcal{A}_\varepsilon \rrbracket(u) = \varepsilon \text{ , odakle sledi } \quad \left| \llbracket \mathcal{A} \rrbracket(u) - \llbracket \mathcal{A}_\varepsilon \rrbracket(u) \right| = \varepsilon - \llbracket \mathcal{A} \rrbracket(u) \leqslant \varepsilon.$$

Prema tome, za svaku reč $u \in X^*$ važi $\left| \llbracket \mathcal{A} \rrbracket(u) - \llbracket \mathcal{A}_\varepsilon \rrbracket(u) \right| \leqslant \varepsilon$, pa na osnovu definicije mere različitosti dobijamo

$$\mathrm{Diss}(\llbracket \mathcal{A} \rrbracket, \llbracket \mathcal{A}_\varepsilon \rrbracket) = \bigvee_{u \in X^*} \left| \llbracket \mathcal{A} \rrbracket(u) - \llbracket \mathcal{A}_\varepsilon \rrbracket(u) \right| \leqslant \varepsilon.$$

Time je dokaz završen. □

Prethodna teorema pokazuje da se ponašanje fazi automata može aproksimirati proizvoljno precizno izborom dovoljno male vrednosti parametra ε.

7.3 ε-slabe simulacije i bisimulacije nad proizvod strukturom

U Glavi 4 pokazano je da su slabe simulacije i bisimulacije između težinskih automata $\mathcal{A}$ i $\mathcal{B}$ definisane pomoću sistema nejednačina određenih familijama $\{\tau_u^A\}_{u \in X^*}$ i $\{\tau_u^B\}_{u \in X^*}$. Pošto se fazi automati mogu posmatrati kao specijalan slučaj težinskih automata, isti pristup može se primeniti i na fazi automate. U tom slučaju slabe simulacije i bisimulacije određene su odgovarajućim sistemima nejednačina fazi relacija.

Za fazi automate nad Gedelovom i Łukašjevičevom strukturom navedene familije su uvek konačne, pa odgovarajući sistemi sadrže konačno mnogo nejednačina i mogu se efikasno rešavati postupcima predloženim u radu [16]. Međutim, u slučaju fazi automata nad proizvod strukturom, familije kodostižnih skupova mogu biti beskonačne, zbog čega odgovarajući sistemi mogu sadržati beskonačno mnogo nejednačina i ne mogu se efikasno rešavati primenom pomenutih postupaka.

U ovoj glavi predlaže se pristup kojim se navedeni problem prevazilazi korišćenjem odsečenih proizvod struktura i odgovarajućih ε-kopija automata. Na taj način sistemi koji određuju slabe simulacije i bisimulacije mogu se aproksimirati konačnim sistemima nejednačina.

Ključnu ulogu u daljem razmatranju ima sledeća lema.

Lema 7.1 *Neka je $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ fazi automat nad proizvod strukturom i neka je $\varepsilon \in (0,1)$. Tada je familija $\left\{(\tau_u^A)_\varepsilon \mid u \in X^*\right\} = \left\{\tau_u^{A_\varepsilon} \mid u \in X^*\right\}$ konačna.*

Dokaz 7.8 *Neka je $V \subseteq \mathbb{I}_\varepsilon$ skup svih vrednosti koje su pojavljuju u fazi skupi $\tau^{A_\varepsilon} = (\tau^A)_\varepsilon$ i fazi relacije $\delta_x^{A_\varepsilon} = (\delta_x^A)_\varepsilon$, za svako $x \in X$, i neka je $\langle V \rangle$ podpoluprsten poluprstena $(\mathbb{I}_\varepsilon, \vee, *_\varepsilon, \varepsilon, 1)$ generisan skupom V. Kako je $\mathcal{A}$ konačan fazi automat, skup V je konačan, pa je i $\langle V \rangle$ konačno generisan podpoluprsten.*

Prema Teoremi 7.2, svaki konačno generisan podpoluprsten poluprstena $(\mathbb{I}_\varepsilon, \vee, *_\varepsilon, \varepsilon, 1)$ je konačan. Dakle, konačan je i podpoluprsten $\langle V \rangle$. Odatle sledi da je konačan i skup $\langle V \rangle^A$ svih preslikavanja iz A u $\langle V \rangle$. Dalje, prema Propoziciji 7.3 važi

$$\left\{(\tau_u^A)_\varepsilon \mid u \in X^*\right\} = \left\{\tau_u^{A_\varepsilon} \mid u \in X^*\right\} \subseteq \langle V \rangle^A.$$

Prema tome, familija $\{(\tau_u^A)_\varepsilon \mid u \in X^*\}$ je konačna. □

Neka su $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ i $\mathcal{B} = (B, \sigma^B, \delta^B, \tau^B)$ fazi automati i neka je $\varepsilon \in (0, 1)$.

Za fazi relaciju $\varphi \in \mathbb{I}^{A \times B}$ kažemo da je ***ε-slaba forward simulacija*** između automata $\mathcal{A}$ i $\mathcal{B}$ ukoliko je njeno ε-odsecanje φ_ε slaba forward simulacija između automata $\mathcal{A}_\varepsilon$ i $\mathcal{B}_\varepsilon$, odnosno ukoliko važe

$$\sigma^{A_\varepsilon} \leqslant \sigma^{B_\varepsilon} \circ_\varepsilon \varphi_\varepsilon^{-1}, \tag{1-ε-wfs}$$

$$\varphi_\varepsilon^{-1} \circ_\varepsilon \tau_u^{A_\varepsilon} \leqslant \tau_u^{B_\varepsilon}, \qquad \text{za svako } u \in X^*. \tag{2-ε-wfs}$$

Dakle, svojstvo ε-slabe forward simulacije ne posmatra se direktno na celoj fazi relaciji φ, već na njenom ε-odsecanju. Na taj način vrednosti fazi relacije koje su manje od praga ε ne utiču na ispitivanje simulacionog svojstva. Drugim rečima, relacija φ se najpre svodi na relaciju φ_ε koja predstavlja njen deo relevantan za zadati nivo ε, a zatim se nad tako dobijenom relacijom proveravaju uslovi slabe forward simulacije.

Za fazi relaciju φ kažemo da je ***ε-slaba forward bisimulacija*** između automata $\mathcal{A}$ i $\mathcal{B}$ ukoliko je φ_ε slaba forward bisimulacija između automata $\mathcal{A}_\varepsilon$ i $\mathcal{B}_\varepsilon$, odnosno ukoliko, pored (1-ε-wfs) i (2-ε-wfs), važe i

$$\sigma^{B_\varepsilon} \leqslant \sigma^{A_\varepsilon} \circ_\varepsilon \varphi_\varepsilon, \tag{1-ε-wfb}$$

$$\varphi_\varepsilon \circ_\varepsilon \tau_u^{B_\varepsilon} \leqslant \tau_u^{A_\varepsilon}, \qquad \text{za svako } u \in X^*. \tag{2-ε-wfb}$$

U nastavku ćemo izostavljati izraz "između automata $\mathcal{A}$ i $\mathcal{B}$" kada su automati $\mathcal{A}$ i $\mathcal{B}$ jasni iz konteksta. Na analogan način mogu se definisati i ε-slabe backward simulacije i ε-slabe backward bisimulacije, ali se njima nećemo baviti, pa ćemo pridev „forward" izostavljati i pisaćemo jednostavno ε-slaba simulacija i ε-slaba bisimulacija.

Vredi primetiti da, prema Lemi 7.1, sistemi nejednačina (2-ε-wfs) i (2-ε-wfb) sadrže konačno mnogo nejednačina.

ε-slaba simulacija (respektivno, ε-slaba bisimulacija) između fazi automata $\mathcal{A}$ i njega samog naziva se ε-slaba simulacija (respektivno, ε-slaba bisimulacija) na automatu $\mathcal{A}$ ili se još može nazivati ε-slaba autosimulacija (respektivno, ε-slaba autobisimulacija).

Propozicija 7.4 *Neka su* $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ *i* $\mathcal{B} = (B, \sigma^B, \delta^B, \tau^B)$ *fazi automati nad proizvod strukturom i neka su* $\varepsilon_1, \varepsilon_2 \in \mathbb{I}$ *takvi da važi* $\varepsilon_1 < \varepsilon_2 < 1$. *Tada važi:*

a) *Svaka* ε_1*-slaba simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *jeste i* ε_2*-slaba simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$.

b) *Svaka* ε_1*-slaba bisimulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *jeste i* ε_2*-slaba bisimulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$.

Dokaz 7.9 *(a) Neka je* φ ε_1*-slaba simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$. *Tada važe uslovi (1-*ε_1*-wfs) i (2-*ε_1*-wfs). Analogno dokazu Propozicije 7.2, ove nejednačine mogu se zapisati u obliku*

$$(\sigma^A)_{\varepsilon_1} \leqslant (\sigma^B \circ \varphi^{-1})_{\varepsilon_1}, \tag{83}$$

$$(\varphi^{-1} \circ \tau_u^A)_{\varepsilon_1} \leqslant (\tau_u^B)_{\varepsilon_1}, \qquad \text{za svako } u \in X^*. \tag{84}$$

Posmatrajmo proizvoljno $a \in A$. *Ako je* $\sigma^A(a) \leqslant \varepsilon_2$, *tada je* $(\sigma^A)_{\varepsilon_2}(a) = \varepsilon_2$, *pa očigledno važi*

$$(\sigma^A)_{\varepsilon_2}(a) \leqslant (\sigma^B \circ \varphi^{-1})_{\varepsilon_2}(a).$$

Ako je $\sigma^A(a) > \varepsilon_2$, *onda je i* $\sigma^A(a) > \varepsilon_1$, *pa iz* (83) *dobijamo*

$$(\sigma^A)_{\varepsilon_2}(a) = (\sigma^A)_{\varepsilon_1}(a) \leqslant (\sigma^B \circ \varphi^{-1})_{\varepsilon_1}(a) \leqslant (\sigma^B \circ \varphi^{-1})_{\varepsilon_2}(a).$$

Prema tome, $(\sigma^A)_{\varepsilon_2} \leqslant (\sigma^B \circ \varphi^{-1})_{\varepsilon_2}$.
Na potpuno isti način, za proizvoljno $u \in X^*$ *i* $a \in A$, *iz* (84) *dobijamo*

$$(\varphi^{-1} \circ \tau_u^A)_{\varepsilon_2} \leqslant (\tau_u^B)_{\varepsilon_2}.$$

Sada, primenom Propozicije 7.2, sledi

$$\sigma^{A_{\varepsilon_2}} = (\sigma^A)_{\varepsilon_2} \leqslant (\sigma^B)_{\varepsilon_2} \circ_{\varepsilon_2} \varphi_{\varepsilon_2}^{-1} = \sigma^{B_{\varepsilon_2}} \circ_{\varepsilon_2} \varphi_{\varepsilon_2}^{-1},$$

$$\varphi_{\varepsilon_2}^{-1} \circ_{\varepsilon_2} \tau_u^{A_{\varepsilon_2}} = (\varphi^{-1} \circ \tau_u^A)_{\varepsilon_2} \leqslant (\tau_u^B)_{\varepsilon_2} = \tau_u^{B_{\varepsilon_2}}, \qquad u \in X^*.$$

Dakle, važe (1-ε_2*-wfs) i (2-*ε_2*-wfs), pa je* φ ε_2*-slaba simulacija između* $\mathcal{A}$ *i* $\mathcal{B}$.
(b) Tvrđenje se dokazuje na isti način, preimenom (a) na uslove (1-ε*-wfb) i (2-*ε*-wfb).* □

Kako su obične slabe simulacije (bisimulacije) zapravo 0-slabe simulacije (bisimulacije), iz prethodne propozicije neposredno sledi da postojanje slabe simulacije (bisimulacije) između dva fazi automata nad proizvod strukturom povlači postojanje ε-slabe simulacije (bisimulacije) za svako $\varepsilon \in (0, 1)$.

Međutim, obrnuto tvrđenje u opštem slučaju ne važi. Štaviše, može se dogoditi da je pitanje postojanja slabe bisimulacije teško ili praktično neizvodljivo proveriti, dok se postojanje ε-slabe bisimulacije za zadatu vrednost parametra ε može utvrditi veoma efikasno. Naredni primer ilustruje upravo takvu situaciju.

Primer 7.1 *Neka su* $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ *i* $\mathcal{B} = (B, \sigma^B, \delta^B, \tau^B)$ *fazi automati nad proizvod strukturom i azbukom* $X = \{x\}$*, gde je*

$$\sigma^A = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}, \qquad \delta^A_x = \begin{bmatrix} 1 & 0 & 1 \\ 0 & \frac{1}{5} & 0 \\ 1 & 0 & 1 \end{bmatrix}, \qquad \tau^A = \begin{bmatrix} 1 \\ \frac{1}{5} \\ 1 \end{bmatrix},$$

$$\sigma^B = \begin{bmatrix} 1 & 1 \end{bmatrix}, \qquad \delta^B_x = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{5} \end{bmatrix}, \qquad \tau^B = \begin{bmatrix} 1 \\ \frac{7}{10} \end{bmatrix}.$$

Lako se izračunavaju familije $\{\tau^A_u\}_{u\in X^*}$ *i* $\{\tau^B_u\}_{u\in X^*}$*, pri čemu za svako* $n \in \mathbb{N}$ *važi*

$$\tau^A_{x^n} = \begin{bmatrix} 1 \\ \frac{1}{5^{n+1}} \\ 1 \end{bmatrix}, \qquad \tau^B_{x^n} = \begin{bmatrix} 1 \\ \frac{1}{5^n}\frac{7}{10} \end{bmatrix}.$$

Dakle, ukoliko želimo da utvrdimo da li između automata $\mathcal{A}$ *i* $\mathcal{B}$ *postoji slaba forward bisimulacija, potrebno je proveriti beskonačno mnogo nejednačina za svako* $n \in \mathbb{N}$

$$\varphi^{-1} \circ \begin{bmatrix} 1 \\ \frac{1}{5^{n+1}} \\ 1 \end{bmatrix} \leq \begin{bmatrix} 1 \\ \frac{1}{5^n}\frac{7}{10} \end{bmatrix}, \qquad \varphi \circ \begin{bmatrix} 1 \\ \frac{1}{5^n}\frac{7}{10} \end{bmatrix} \leq \begin{bmatrix} 1 \\ \frac{1}{5^{n+1}} \\ 1 \end{bmatrix}.$$

U ovom primeru članove familija $\{\tau^A_u\}_{u\in X^*}$ *i* $\{\tau^B_u\}_{u\in X^*}$ *uspeli smo da predstavimo eksplicitno pomoću opštih formula. Takve situacije su retke. U najvećem broj slučajeva veoma je teško, a često i praktično nemoguće, dobiti eksplicitan opis ovih familija.*

Posmatrajmo sada slučaj kada je $\varepsilon = 0.0001$*. Tada su familije* $\{(\tau^A_u)_\varepsilon\}_{u\in X^*}$ *i* $\{(\tau^B_u)_\varepsilon\}_{u\in X^*}$ *konačne i imaju oblik*

$$(\tau^A_{x^n})_\varepsilon = \begin{bmatrix} 1 \\ \frac{1}{5^{n+1}} \\ 1 \end{bmatrix} \text{for } n \in \{0, 1, .., 4\}, \qquad (\tau^B_{x^n})_\varepsilon = \begin{bmatrix} 1 \\ \frac{1}{5^n}\frac{7}{10} \end{bmatrix} \text{for } n \in \{0, 1, .., 5\},$$

$$(\tau^A_{x^n})_\varepsilon = \begin{bmatrix} 1 \\ \varepsilon \\ 1 \end{bmatrix} \text{for } n > 4, \qquad (\tau^B_{x^n})_\varepsilon = \begin{bmatrix} 1 \\ \varepsilon \end{bmatrix} \text{for } n > 5.$$

Prema tome, provera postojanja ε*-slabe forward bisimulacije za* $\varepsilon = 0.0001$ *svodi se*

na rešavanje sledećeg konačnog sistema nejednačina:

$$\begin{bmatrix} 1 & \varepsilon & 1 \end{bmatrix} \leq \begin{bmatrix} 1 & 1 \end{bmatrix} \circ_\varepsilon \varphi^{-1}, \quad \begin{bmatrix} 1 & 1 \end{bmatrix} \leq \begin{bmatrix} 1 & \varepsilon & 1 \end{bmatrix} \circ_\varepsilon \varphi,$$

$$\varphi^{-1} \circ_\varepsilon \begin{bmatrix} 1 \\ \frac{1}{5^{n+1}} \\ 1 \end{bmatrix} \leq \begin{bmatrix} 1 \\ \frac{1}{5^n}\,\frac{7}{10} \end{bmatrix}, \qquad \varphi \circ_\varepsilon \begin{bmatrix} 1 \\ \frac{1}{5^n}\,\frac{7}{10} \end{bmatrix} \leq \begin{bmatrix} 1 \\ \frac{1}{5^{n+1}} \\ 1 \end{bmatrix}, \quad za\ \ n \in \{0, .., 4\},$$

$$\varphi^{-1} \circ_\varepsilon \begin{bmatrix} 1 \\ \varepsilon \\ 1 \end{bmatrix} \leq \begin{bmatrix} 1 \\ \frac{1}{5^5}\,\frac{7}{10} \end{bmatrix}, \qquad \varphi \circ_\varepsilon \begin{bmatrix} 1 \\ \frac{1}{5^5}\,\frac{7}{10} \end{bmatrix} \leq \begin{bmatrix} 1 \\ \varepsilon \\ 1 \end{bmatrix},$$

$$\varphi^{-1} \circ_\varepsilon \begin{bmatrix} 1 \\ \varepsilon \\ 1 \end{bmatrix} \leq \begin{bmatrix} 1 \\ \varepsilon \end{bmatrix}, \qquad \varphi \circ_\varepsilon \begin{bmatrix} 1 \\ \varepsilon \end{bmatrix} \leq \begin{bmatrix} 1 \\ \varepsilon \\ 1 \end{bmatrix}.$$

Na početku ove sekcije definisali smo pojam ε-slabe bisimulacije između fazi automata $\mathcal{A}$ i $\mathcal{B}$ kao fazi relacije čije je ε-odsecanje obična slaba bisimulacija između ε-kopija automata $\mathcal{A}$ i $\mathcal{B}$. Drugim rečima, postojanje ε-slabe bisimulacije između automata $\mathcal{A}$ i $\mathcal{B}$ povlači ekvivalenciju njihovih ε-kopija $\mathcal{A}_\varepsilon$ i $\mathcal{B}_\varepsilon$, ali ne nužno i ekvivalenciju samih automata $\mathcal{A}$ i $\mathcal{B}$.

Zbog toga se prirodno postavlja pitanje koliko se ponašanja automata $\mathcal{A}$ i $\mathcal{B}$ razlikuju. U prethodnoj sekciji razlika između ponašanja automata i njegove ε-kopije merena je pomoću Lukasijevičeve mere sličnosti. U nastavku koristimo istu meru kako bismo kvantifikovali razliku između ponašanja automata $\mathcal{A}$ i $\mathcal{B}$.

Teorema 7.5 *Neka su $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ i $\mathcal{B} = (B, \sigma^B, \delta^B, \tau^B)$ fazi konačni automati nad proizvod strukturom. Ako između automata $\mathcal{A}$ i $\mathcal{B}$ postoj ε-slaba bisimulacija, za neko $\varepsilon \in (0, 1)$, tada važi* $\mathrm{Diss}([\![\mathcal{A}]\!], [\![\mathcal{B}]\!]) \leqslant \varepsilon$.

Dokaz 7.10 *Pretpostavimo da između automata $\mathcal{A}$ i $\mathcal{B}$ postoji ε-slaba bisimulacija. Tada, po definiciji, između automata $\mathcal{A}_\varepsilon$ i $\mathcal{B}_\varepsilon$ postoji slaba bisimulacija, pa važi $[\![\mathcal{A}_\varepsilon]\!] = [\![\mathcal{B}_\varepsilon]\!]$.*

Posmatrajmo proizvoljnu reč $u \in X^*$.

Ako je $[\![\mathcal{A}_\varepsilon]\!](u) = [\![\mathcal{B}_\varepsilon]\!](u) > \varepsilon$, onda je

$$[\![\mathcal{A}_\varepsilon]\!](u) = [\![\mathcal{A}]\!]_\varepsilon(u) = [\![\mathcal{A}]\!](u) \quad i \quad [\![\mathcal{B}_\varepsilon]\!](u) = [\![\mathcal{B}]\!]_\varepsilon(u) = [\![\mathcal{B}]\!](u),$$

odakle sledi $[\![\mathcal{A}]\!](u) = [\![\mathcal{B}]\!](u)$, pa je $\left|[\![\mathcal{A}]\!](u) - [\![\mathcal{B}]\!](u)\right| = 0$.

U suprotnom, neka je $[\![\mathcal{A}_\varepsilon]\!](u) = [\![\mathcal{B}_\varepsilon]\!](u) = \varepsilon$, odnosno

$$[\![\mathcal{A}]\!]_\varepsilon(u) = [\![\mathcal{B}]\!]_\varepsilon(u) = \varepsilon.$$

Tada važi $[\![\mathcal{A}]\!](u) \leqslant \varepsilon$ i $[\![\mathcal{B}]\!](u) \leqslant \varepsilon$, pa je $\left|[\![\mathcal{A}]\!](u) - [\![\mathcal{B}]\!](u)\right| \leqslant \varepsilon$.

Prema tome,

$$\mathrm{Diss}(\llbracket \mathcal{A} \rrbracket, \llbracket \mathcal{B} \rrbracket) = \bigvee_{u \in X^*} \left| \llbracket \mathcal{A} \rrbracket(u) - \llbracket \mathcal{B} \rrbracket(u) \right| \leqslant \varepsilon.$$

Time je dokaz završen. □

Prema definiciji ε-slabih simulacija i Teoreme 4.2, postojanje ε-slabe simulacije između automata $\mathcal{A}$ i $\mathcal{B}$ povlači uključenje njihovih ε-kopija, odnosno $\llbracket \mathcal{A}_\varepsilon \rrbracket \leqslant \llbracket \mathcal{B}_\varepsilon \rrbracket$.

Prirodno se postavlja pitanje kakav je odnos između jezika $\llbracket \mathcal{A} \rrbracket$ i $\llbracket \mathcal{B} \rrbracket$. Jasno je da iz postojanja ε-slabe simulacije ne mora slediti uključenje $\llbracket \mathcal{A} \rrbracket \leqslant \llbracket \mathcal{B} \rrbracket$, pa se nameće pitanje u kojoj meri je ovo uključenje narušeno. Da bismo odgovorili na to pitanje, uvodimo nekoliko novih pojmova.

Neka su $\alpha, \beta \in \mathbb{I}^A$ fazi skupovi. ***Razlika fazi skupova*** α i β definiše se kao fazi skup $\alpha \ominus \beta \in \mathbb{I}^A$ zadat sa

$$(\alpha \ominus \beta)(a) = \max\{\alpha(a) - \beta(a), 0\}, \text{ za svako } a \in A. \tag{85}$$

Dalje, ***mera isključenja*** definiše se kao preslikavanje $\mathrm{Excl} : \mathbb{I}^A \times \mathbb{I}^A \to \mathbb{I}$, zadato sa

$$\mathrm{Excl}(\alpha, \beta) = \bigvee_{a \in A} (\alpha \ominus \beta)(a), \text{ za sve } \alpha, \beta \in \mathbb{I}^A. \tag{86}$$

Kažemo da je $\mathrm{Excl}(\alpha, \beta)$ *mera isključenja skupa* α *iz skupa* β. Ova mera kvantifikuje odstupanje fazi skupa α od uključenja u fazi skup β.

Teorema 7.6 *Neka su* $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ *i* $\mathcal{B} = (B, \sigma^B, \delta^B, \tau^B)$ *fazi konačni automati nad proizvod strukturom. Ako između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *postoj* ε*-slaba simulacija, za neko* $\varepsilon \in (0, 1)$*, tada važi* $\mathrm{Excl}(\llbracket \mathcal{A} \rrbracket, \llbracket \mathcal{B} \rrbracket) \leqslant \varepsilon$.

Dokaz 7.11 *Pretpostavimo da između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *postoji* ε*-slaba simulacija. Tada, po definiciji, između automata* $\mathcal{A}_\varepsilon$ *i* $\mathcal{B}_\varepsilon$ *postoji slaba simulacija. Na osnovu Teoreme 3.1 sledi da je* $\llbracket \mathcal{A}_\varepsilon \rrbracket \leqslant \llbracket \mathcal{B}_\varepsilon \rrbracket$*, odnosno* $\llbracket \mathcal{A} \rrbracket_\varepsilon \leqslant \llbracket \mathcal{B} \rrbracket_\varepsilon$.

Posmatrajmo proizvoljnu reč $u \in X^*$.

Ako je $\llbracket \mathcal{A} \rrbracket(u) \leqslant \llbracket \mathcal{B} \rrbracket(u)$*, onda je* $(\llbracket \mathcal{A} \rrbracket \ominus \llbracket \mathcal{B} \rrbracket)(u) = 0 < \varepsilon$.

Pretpostavimo sada da je $\llbracket \mathcal{A} \rrbracket(u) > \llbracket \mathcal{B} \rrbracket(u)$*. Pošto važi* $\llbracket \mathcal{A} \rrbracket_\varepsilon \leqslant \llbracket \mathcal{B} \rrbracket_\varepsilon$*, ovo je moguće jedino ako je* $\llbracket \mathcal{B} \rrbracket(u) < \llbracket \mathcal{A} \rrbracket(u) \leqslant \varepsilon$*. Tada dobijamo* $0 < \llbracket \mathcal{A} \rrbracket(u) - \llbracket \mathcal{B} \rrbracket(u) \leqslant \varepsilon$*, odnosno* $(\llbracket \mathcal{A} \rrbracket \ominus \llbracket \mathcal{B} \rrbracket)(u) \leqslant \varepsilon$*. Prema tome,*

$$\mathrm{Excl}(\llbracket \mathcal{A} \rrbracket, \llbracket \mathcal{B} \rrbracket) = \bigvee_{u \in X^*} (\llbracket \mathcal{A} \rrbracket \ominus \llbracket \mathcal{B} \rrbracket)(u) \leqslant \varepsilon. \; \square$$

Sledeća teorema daje postupak za ispitivanje postojanja ε-slabih simulacija i ε-slabih bisimulacija između fazi automata, kao i za određivanje najvećih takvih relacija u slučajevima kada one postoje.

Teorema 7.7 *Neka su* $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ *i* $\mathcal{B} = (B, \sigma^B, \delta^B, \tau^B)$ *fazi konačni automati nad proizvod strukturom i neka je* $\varepsilon \in (0, 1)$. *Tada važi:*

a) *Ako fazi relacija*

$$\phi_{\varepsilon\text{wfs}} = \bigwedge_{u \in X^*} (\tau_u^A)_\varepsilon \backslash_\varepsilon (\tau_u^B)_\varepsilon$$

zadovoljava uslov (1-ε*-wfs), onda je ona najveća* ε*-slaba simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$. *U suprotnom, između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *ne postoji* ε*-slaba simulacija.*

b) *Ako fazi relacija*

$$\phi_{\varepsilon\text{wfb}} = \bigwedge_{u \in X^*} (\tau_u^A)_\varepsilon \mid_\varepsilon (\tau_u^B)_\varepsilon$$

zadovoljava uslove (1-ε*-wfs), i (1-*ε*-wfb), onda je ona najveća* ε*-slaba bisimulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$. *U suprotnom, između automata* $\mathcal{A}$ *i* $\mathcal{B}$ *ne postoji* ε*-slaba bisimulacija.*

Dokaz 7.12 *(a) Najpre uočimo da važi* $\phi_{\varepsilon\text{wfs}} \in (\mathbb{I}_\varepsilon)^{A \times B}$, *pa je* $(\phi_{\varepsilon\text{wfs}})_\varepsilon = \phi_{\varepsilon\text{wfs}}$.

Prema Teoremi 3.7 iz rada [16], ukoliko relacija $\phi_{\varepsilon\text{wfs}}$ *zadovoljava uslov (1-*ε*-wfs), tada je ona najveća slaba simulacija između automata* $\mathcal{A}_\varepsilon$ *i* $\mathcal{B}_\varepsilon$.

Kako važi $(\phi_{\varepsilon\text{wfs}})_\varepsilon = \phi_{\varepsilon\text{wfs}}$, *sledi da je* $\phi_{\varepsilon\text{wfs}}$ *i* ε*-slaba simulacija između* $\mathcal{A}$ *i* $\mathcal{B}$.

Neka je sada φ *proizvoljna* ε*-slaba simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$. *Tada je* φ_ε *slaba simulacija između automata* $\mathcal{A}_\varepsilon$ *i* $\mathcal{B}_\varepsilon$, *pa zbog maksimalnosti relacije* $\phi_{\varepsilon\text{wfs}}$ *važi* $\varphi_\varepsilon \leqslant \phi_{\varepsilon\text{wfs}}$. *Pošto je očigledno* $\varphi \leqslant \varphi_\varepsilon$, *dobijamo* $\varphi \leqslant \varphi_\varepsilon \leqslant \phi_{\varepsilon\text{wfs}}$, *odakle sledi da je* $\phi_{\varepsilon\text{wfs}}$ *najveća* ε*-slaba simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$.

Ako relacija $\phi_{\varepsilon\text{wfs}}$ *ne zadovoljava uslov (1-*ε*-wfs), tada prema [16, Teorema 3.7] ne postoji slaba simulacija između automata* $\mathcal{A}_\varepsilon$ *i* $\mathcal{B}_\varepsilon$. *Prema tome, ne postoji ni* ε*-slaba simulacija između automata* $\mathcal{A}$ *i* $\mathcal{B}$.

(b) Tvrđenje se dokazuje na potpuno analogan način. □

Posledica 7.1 *Neka je* $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ *fazi automat nad proizvod strukturom i neka je* $\varepsilon \in (0, 1)$. *Tada važi:*

a) *Fazi relacija*

$$\psi_{\varepsilon\text{wfs}} = \bigwedge_{u \in X^*} (\tau_u^A)_\varepsilon \backslash_\varepsilon (\tau_u^A)_\varepsilon$$

jeste fazi predurađenje i najveća ε*-slaba simulacija na automatu* $\mathcal{A}$;

b) *Fazi relacija*

$$\psi_{\varepsilon\text{wfb}} = \bigwedge_{u \in X^*} (\tau_u^A)_\varepsilon \leftrightarrow_\varepsilon (\tau_u^A)_\varepsilon$$

jeste fazi ekvivalencija i najveća ε*-slaba bisimulacija na automatu* $\mathcal{A}$.

Dokaz 7.13 *Za proizvoljan fazi skup važi da njegovi levi i desni rezidul sa samim sobom predstavljaju fazi pre-oreder, dok odgovarajući birezidul predstavlja fazi ekvivalenciju. Stoga je relacija $\psi_{\varepsilon\text{wfs}}$ fazi preduređenje, kao presek familije fazi pre-oreder, a relacija $\psi_{\varepsilon\text{wfb}}$ fazi ekvivalencija, kao presek familije fazi ekvivalencija.*

Preostali deo tvrđenja neposredno sledi iz Teoreme 7.7, kao i iz rezultata dokazanih u [46, Theorem 5.6] i [16, Theorem 3.7]. □

Primer 7.2 *Posmatrajmo fazi automate iz Primera 7.1.*

Primenom Teoreme 7.7, za $\varepsilon = 0.0001$ dobijamo da je najveća ε-slaba simulacija između automata $\mathcal{A}$ i $\mathcal{B}$ jednaka

$$\begin{aligned}\varphi &= \bigwedge_{u\in X^*} (\tau_u^A)_\varepsilon \backslash_\varepsilon (\tau_u^B)_\varepsilon \\ &= \bigwedge_{n\in\{0,1,..,4\}} \begin{bmatrix} 1 \\ \frac{1}{5^{n+1}} \\ 1 \end{bmatrix} \backslash_\varepsilon \begin{bmatrix} 1 \\ \frac{1}{5^n}\ \frac{7}{10} \end{bmatrix} \wedge \begin{bmatrix} 1 \\ \varepsilon \\ 1 \end{bmatrix} \backslash_\varepsilon \begin{bmatrix} 1 \\ \frac{1}{5^5}\ \frac{7}{10} \end{bmatrix} \wedge \bigwedge_{n>5} \begin{bmatrix} 1 \\ \varepsilon \\ 1 \end{bmatrix} \backslash_\varepsilon \begin{bmatrix} 1 \\ \varepsilon \end{bmatrix} = \begin{bmatrix} 1 & \varepsilon \\ \varepsilon & \frac{2}{7} \\ 1 & \varepsilon \end{bmatrix}.\end{aligned}$$

Pošto ova fazi relacija zadovoljava uslov (1-ε-wfs), sledi da je ona najveća ε-slaba simulacija između automata $\mathcal{A}$ i $\mathcal{B}$.

Primer 7.3 *Neka su $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ i $\mathcal{B} = (B, \sigma^B, \delta^B, \tau^B)$ fazi automati nad proizvod strukturom i alfabetom $X = \{x\}$, gde je*

$$\sigma^A = \begin{bmatrix} 1 & 0 & 1 \end{bmatrix}, \qquad \delta_x^A = \begin{bmatrix} 1 & 0 & 1 \\ 0 & \frac{1}{5} & 0 \\ 1 & 0 & 1 \end{bmatrix}, \qquad \tau^A = \begin{bmatrix} 1 \\ \frac{1}{5} \\ 1 \end{bmatrix},$$

$$\sigma^B = \begin{bmatrix} 1 & \frac{1}{10} \end{bmatrix}, \qquad \delta_x^B = \begin{bmatrix} 1 & 0 \\ 0 & \frac{1}{5} \end{bmatrix}, \qquad \tau^B = \begin{bmatrix} 1 \\ \frac{7}{10} \end{bmatrix}.$$

Za $\varepsilon = 10^{-1}$ postoji ε-slaba bisimulacija između automata $\mathcal{A}$ i $\mathcal{B}$, i data je matricom

$$\varphi = \begin{bmatrix} 1 & \varepsilon \\ \varepsilon & \frac{2}{7} \\ 1 & \varepsilon \end{bmatrix}.$$

Međutim, za $\varepsilon = 10^{-2}$ ne postoji ε-slaba bisimulacija između ovih automata.

Sledeći primer ilustruje primenu Teoreme 7.7 za izračunavanje najveće ε-slabe autobisimulacije na datom fazi automatu. Za posmatrani automat nije moguće direktno primeniti poznati postupak za određivanje najveće slabe forward autobisimulacije, jer je familija kodostižnih skupova $\{\tau_u\}_{u\in X^*}$ beskonačna. Međutim, nakon ε-odsecanja dobija se konačna familija, pa je najveću ε-slabu autobisimulaciju moguće efikasno izračunati.

Dobijena ε-slaba autobisimulacija predstavlja netrivijalnu fazi ekvivalenciju, različitu od identičke fazi relacije, te se može koristiti za redukciju broja stanja posmatranog fazi automata.

Primer 7.4 Neka je $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ fazi automat nad proizvod strukturom i alfabetom $X = \{x, y\}$, gde je

$$\sigma^A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0.7 \end{bmatrix}, \qquad (\tau^A)^T = \begin{bmatrix} 0 & 1 & 0.8 & 1 & 1 \end{bmatrix},$$

$$\delta_x^A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 \\ 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \qquad \delta_y^A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.3 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Za $\varepsilon = 10^{-1}$ najveća ε-slaba forward autobisimulacija dobija se nakon generisanja 7 različitih vektora oblika $(\tau_u)_\varepsilon$, gde je $u \in X^*$. U tom slučaju dobija se fazi relacija

$$\varphi = \begin{bmatrix} 1 & \varepsilon & \frac{1}{8} & \varepsilon & \varepsilon \\ \varepsilon & 1 & \frac{1}{5} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{8} & \frac{1}{5} & 1 & \varepsilon & \varepsilon \\ \varepsilon & \frac{1}{2} & \varepsilon & 1 & 1 \\ \varepsilon & \frac{1}{2} & \varepsilon & 1 & 1 \end{bmatrix}$$

Sa druge strane, za $\varepsilon = 10^{-3}$ najveća ε-slaba forward autobisimulacija dobija se nakon generisanja 21 različitog vektora oblika $(\tau_u)_\varepsilon$, a odgovarajuća relacija je

$$\varphi = \begin{bmatrix} 1 & \varepsilon & \frac{1}{800} & \varepsilon & \varepsilon \\ \varepsilon & 1 & \frac{1}{500} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{800} & \frac{1}{500} & 1 & \varepsilon & \varepsilon \\ \varepsilon & \frac{1}{2} & \varepsilon & 1 & 1 \\ \varepsilon & \frac{1}{2} & \varepsilon & 1 & 1 \end{bmatrix}$$

7.4 Algoritam za računanje najvećih ε-slabih simulacija i bisimulacija

U Teoremi 7.7 izvedeni su izrazi za najveću ε-slabu simulaciju i najveću ε-slabu bisimulaciju između dva fazi automata nad proizvod strukturom. Na osnovu tih rezultata može se konstruisati algoritam za njihovo određivanje.

Za razliku od običnih slabih simulacija i bisimulacija, kod kojih familije kodostižnih skupova mogu biti beskonačne, u slučaju ε-odsecanja, prema Lemi 7.1, posmatrane familije su konačne. Zbog toga se najveće ε-slabe simulacije i bisimulacije mogu izračunati generisanjem svih različitih parova odsečenih kodostižnih skupova i postepenim formiranjem odgovarajuće relacije.

Algoritam 4, zasnovan na Teoremi 7.7, daje postupak za određivanje najveće ε-slabe simulacije, odnosno najveće ε-slabe bisimulacije između fazi automata $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ i $\mathcal{B} = (B, \sigma^B, \delta^B, \tau^B)$ nad alfabetom X. Pored automata $\mathcal{A}$ i $\mathcal{B}$, alfabeta X i parametra $\varepsilon \in (0, 1)$, algoritam kao ulaz koristi i logičku promenljivu *bis*, koja određuje da li se računa simulacija ili bisimulacija.

Algoritam održava skup *AllPairs* svih do tada generisanih parova odsečenih kodostižnih skupova $((\tau_u^A)_\varepsilon, (\tau_u^B)_\varepsilon)$, a parovi koji još nisu obrađeni čuvaju u redu *Unprocessed*. Pri tome se same reči $u \in X^*$ ne pamte, već samo odgovarajući parovi vektora. Svaki generisani par obrađuje se najviše jednom, što predstavlja ključnu ideju algoritma.

Polazeći od početnog para $((\tau^A)_\varepsilon, (\tau^B)_\varepsilon)$, algoritam sukcesivno generiše nove parove primenom odsečenih prelaznih relacija. Istovremeno se formira relacija φ kao infimum odgovarajućih reziduala, odnosno bireziduala u slučaju bisimulacija. Po završetku generisanja svih različitih parova proveravaju se uslovi (1-ε-wfs), a u slučaju bisimulacija i (1-ε-wfb). Ukoliko su oni zadovoljeni, dobijena relacija predstavlja najveću ε-slabu simulaciju, odnosno ε-slabu bisimulaciju; u suprotnom se zaključuje da takva relacija ne postoji.

Procedura 4 Određivanje najveće ε-slabe simulacije i ε-slabe bisimulacije

Ulaz: Fazi automati $\mathcal{A} = (A, \sigma^A, \delta^A, \tau^A)$ i $\mathcal{B} = (B, \sigma^B, \delta^B, \tau^B)$ nad proizvod strukturom;
ulazni alfabet X;
$\varepsilon \in (0, 1)$;
logička promenljiva *bis* koja određuje da li se računa
simulacija (*bis* = False) ili bisimulacija (*bis* = True).

Izlaz: najveća ε-slaba simulacija, odnosno ε-slaba bisimulacija
između automata $\mathcal{A}$ i $\mathcal{B}$, ukoliko postoji,
odnosno *null* ukoliko ne postoji.

Inicijalizuje prazan skup *allPairs* i prazan red *unprocessed*;

$$(\tau_0^A, \tau_0^B) = ((\tau^A)_\varepsilon, (\tau^B)_\varepsilon);$$

allPairs.add(τ_0^A, τ_0^B);
unprocessed.enqueue(τ_0^A, τ_0^B);

$$\varphi \leftarrow \begin{cases} \tau_0^A \mid_\varepsilon \tau_0^B, & \text{ako je } bis = \text{True}, \\ \tau_0^A \backslash_\varepsilon \tau_0^B, & \text{inače.} \end{cases}$$

while *unprocessed* is not empty **do**
    $(\tau_1, \tau_2) \leftarrow$ *unprocessed.dequeue*();
    for each $x \in X$ **do**
        $\tau_1' \leftarrow (\delta_x^A)_\varepsilon \circ_\varepsilon \tau_1$;
        $\tau_2' \leftarrow (\delta_x^B)_\varepsilon \circ_\varepsilon \tau_2$;

$$\varphi \leftarrow \varphi \wedge \begin{cases} \tau_1' \mid_\varepsilon \tau_2', & \text{ako je } bis = \text{True}, \\ \tau_1' \backslash_\varepsilon \tau_2', & \text{inače.} \end{cases}$$

        if $(\tau_1', \tau_2') \notin$ *allPairs* **then**
            allPairs.add(τ_1', τ_2');
            unprocessed.enqueue(τ_1', τ_2');
        end if
    end for
end while
if $(\sigma^A)_\varepsilon \not\leqslant (\sigma^B) * \varepsilon \circ_\varepsilon \varphi^{-1}$ *or* (*bis and* $(\sigma^B)_\varepsilon \not\leqslant (\sigma^A) * \varepsilon \circ_\varepsilon \varphi$) **then**
    return *null*
else
    return φ
end if

Teorema 7.8 *Algoritam 4 se uvek završava u konačno mnogo koraka i kao rezultat daje najveću slabu (bi)simulaciju, ukoliko ona postoji.*

Dokaz 7.14 *Posmatrajmo glavnu petlju Algoritma 4. Njen invarijant je da skup AllPairs sadrži samo parove oblika* $((\tau_u^A)_\varepsilon, (\tau_u^B)_\varepsilon)$*, za neka* $u \in X^*$*, dok red Unprocessed sadrži samo parove iz skupa AllPairs koji još nisu obrađeni.*

Novi par (τ_1', τ_2') *dodaje se u skup AllPairs i red Unprocessed samo ukoliko se već ne nalazi u skupu AllPairs, dok se elementi iz skupa AllPairs nikada ne uklanjaju. Prema Lemi 7.1, familije* $\{(\tau_u^A)_\varepsilon \mid u \in X^*\}$ *i* $\{(\tau_u^B)_\varepsilon \mid u \in X^*\}$ *su konačne. Otuda postoji konačno mnogo različitih parova koji se mogu pojaviti u skupu AllPairs, pa algoritam može izvršiti samo konačno mnogo umetanja novih parova. Kako se svaki par obrađuje najviše jednom, sledi da se glavna petlja završava nakon konačno mnogo koraka, odnosno da se algoritam uvek zaustavlja.*

Dalje, u unutrašnjoj petlji imamo

$$\varphi = \left(bis\ ?\ \bigwedge_{(\tau_1,\tau_2)\in AllPairs} \tau_1|_\varepsilon \tau_2\ :\ \bigwedge_{(\tau_1,\tau_2)\in AllPairs} \tau_1 \backslash_\varepsilon \tau_2 \right).$$

Indukcijom po dužini reči $u \in X^*$ *jednostavno se pokazuje da će svaki par* $((\tau_u^A)_\varepsilon, (\tau_u^B)_\varepsilon)$ *biti generisan i na kraju dodat u skup AllPairs. Prema tome, po završetku glavne petlje važi*

$$\varphi = \left(bis\ ?\ \bigwedge_{u\in X^*} (\tau_u^A)_\varepsilon|_\varepsilon (\tau_u^B)_\varepsilon\ :\ \bigwedge_{u\in X^*} (\tau_u^A)_\varepsilon \backslash_\varepsilon (\tau_u^B)_\varepsilon \right).$$

Sada ispravnost algoritma neposredno sledi iz Teoreme 7.7, jer završna provera uslova (1-ε-wfs), odnosno i (1-ε-wfb) u slučaju bisimulacija, tačno odgovara uslovima iz tog teorema. □

Nakon što smo pokazali korektnost Algoritma 4, razmotrićemo njegovu vremensku složenost.

Neka je D skup svih stepeni pripadnosti iz intervala $(0, 1)$ koji se pojavljuju u matricama prelaza i vektorima završnih stanja automata $\mathcal{A}$ i $\mathcal{B}$. Označimo sa $n = \max(|A|, |B|)$ maksimalan broj stanja posmatranih automata, a sa $m = |D|$ broj različitih stepena pripadnosti koji se u njima javljaju. Ako je $m = 0$, stavljamo $l = 1$. U suprotnom, neka je $l = \lfloor \log_\lambda \varepsilon \rfloor$, $\lambda = \max(D)$. Tada za svako $k > l$ važi $\lambda^k < \varepsilon$, pa se nakon najviše l uzastopnih kompozicija svaki stepen pripadnosti spušta ispod praga ε i u ε-odsecanju postaje jednak vrednosti ε. Posmatrajmo skup

$$E = \{(\tau_u^A)_\varepsilon \varepsilon(a), (\tau_u^B)_\varepsilon \varepsilon(b) \mid u \in X^*,\ a \in A,\ b \in B\}.$$

Lako se vidi da važi $|E| \leq m^l + 2$. Otuda sledi da broj različitih parova odsečenih kodostižnih skupova koje algoritam može generisati ne prelazi $(m^l + 2)^{2n}$, pa se glavna petlja algoritma izvršava najviše toliko puta. Cena jedne iteracije glavne petlje reda je $O\left(|X|\left(n^2 + n\log_2((m^l+2)^{2n})\right)\right)$, što je sadržano u $O(|X|\, l\, n^2 \log n)$, jer je $m = O(n^2)$.

Kako je složenost naredbi van glavne petlje zanemarljiva u odnosu na složenost same petlje, dobijamo sledeći rezultat.

Propozicija 7.5 *Vremenska složenost Algoritma 4 je*

$$O\left(|X|(m^l+2)^{2n} l n^2 \log n\right),$$

gde su l, m i n definisani kao u prethodnom tekstu.

U ovoj glavi uveden je pojam ε-slabih simulacija i bisimulacija kao aproksimativne verzije slabih simulacija i bisimulacija za fazi automate nad proizvod strukturom. Osnovna ideja zasniva se na ε-odsecanju fazi skupova, čime se beskonačne familije kodostižnih skupova svode na konačne, a problemi određivanja simulacija i bisimulacija postaju algoritamski rešivi.

Pokazano je da postojanje slabih simulacija i bisimulacija povlači postojanje odgovarajućih ε-slabih simulacija i bisimulacija za svako $\varepsilon \in (0, 1)$, kao i da se približna ekvivalencija i približna inkluzija između ponašanja automata mogu kvantitativno proceniti. Preciznije, dokazano je da postojanje ε-slabe bisimulacije ograničava meru različitosti jezika automata vrednošću ε, dok postojanje ε-slabe simulacije ograničava odstupanje od inkluzije takođe vrednošću ε.

Posebno značajan rezultat predstavlja izvođenje zatvorenih izraza za najveće ε-slabe simulacije i bisimulacije, kao i algoritma za njihovo određivanje. Za razliku od običnih slabih simulacija i bisimulacija, za koje u opštem slučaju nije poznat algoritam za izračunavanje najvećih rešenja, najveće ε-slabe simulacije i bisimulacije mogu se efektivno izračunati primenom predloženog postupka.

Na taj način, ε-slabe simulacije i bisimulacije predstavljaju praktičan i teorijski zasnovan pristup aproksimaciji slabih simulacija i bisimulacija, koji zadržava njihovo osnovno semantičko značenje, a istovremeno omogućava efikasnu računarsku obradu. Kao prirodan pravac daljih istraživanja nameće se razvoj sličnih aproksimativnih metoda za druge strukture istinitosnih vrednosti koje nisu lokalno konačne.

Zaključak

U ovoj disertaciji proučavane su simulacije i bisimulacije za težinske automate, sa posebnim fokusom na max-plus automate. Razmatrani su različiti tipovi standardnih i slabih simulacija i bisimulacija, njihova karakterizacija pomoću sistema matričnih nejednačina, kao i postupci za određivanje njihovih najvećih rešenja. Posebna pažnja posvećena je problemima koji otežavaju njihovo efektivno računanje, pre svega mogućnosti neprekidnog iteriranja i pojavu beskonačno mnogo uslova kod slabih relacija. U cilju prevazilaženja ovih ograničenja uvedene su relaksacije slabih simulacija i bisimulacija, kao i pristupi zasnovani na odsecanju i pomeranju težina u max-plus strukturama. Razvijeni pristupi prošireni su i na aproksimativno poređenje fazi automata nad proizvod strukturom.

Dobijeni rezultati pokazuju da se relaksacijom uslova i odgovarajućom modifikacijom strukture nad kojom se vrši računanje mogu prevazići određena ograničenja standardnih postupaka, uz istovremeno očuvanje kontrolisanog nivoa informacija o ponašanju automata. Na taj način simulacije i bisimulacije mogu predstavljati ne samo teorijski alat za poređenje automata, već i osnovu za razvoj efektivnih aproksimativnih metoda.

U okviru disertacije simulacije i bisimulacije su, najvećim delom, posmatrane kao relacije između dva automata. U jednom delu rada samo je ukratko pomenuta mogućnost njihovog posmatranja između stanja jednog automata. Ovakav pristup otvara dodatnu mogućnost primene simulacija i bisimulacija, jer se relacije između stanja mogu koristiti za njihovo grupisanje, odnosno faktorizaciju i redukciju broja stanja automata, uz odgovarajući stepen očuvanja njegovog ponašanja.

Jedan od pravaca daljeg istraživanja biće upravo proučavanje uticaja simulacija i bisimulacija, kao i njihovih aproksimativnih verzija, na redukciju broja stanja max-plus automata. Posebna pažnja može biti posvećena istraživanju konstrukcije redukovanih automata na osnovu odgovarajućih relacija između njihovih stanja, kao i ispitivanju odnosa između stepena relaksacije simulacije ili bisimulacije i ostvarene redukcije. Dalja istraživanja mogu obuhvatiti i razvoj efikasnih algoritama za pronalaženje takvih relacija i analizu dobijenih redukovanih automata. Na taj način prirodan nastavak istraživanja predstavlja povezivanje teorije simulacija i bisimulacija sa problemom redukcije težinskih automata, čime bi se razvijeni koncepti mogli primeniti ne samo na poređenje automata, već i na njihovo pojednostavljivanje i efikasniju analizu.

Literatura

[1] M.-P. Schützenberger, On the definition of a family of automata, Information and Control 4 (2-3) (1961) 245–270. doi:https://doi.org/10.1016/S0019-9958(61)80020-X.

[2] W. Kuich, A. Salomaa, Semirings, Automata, Languages, Springer Berlin, Heidelberg, 2012. doi:10.1007/978-3-642-69959-7.

[3] M. Droste, W. Kuich, H. Vogler, Handbook of Weighted Automata, Springer, 2009.

[4] R. Milner, A Calculus of Communicating Systems, Lecture Notes in Computer Science, 1980.

[5] D. M. R. Park, Concurrency and automata on infinite sequences, in: Theoretical Computer Science, Vol. 104 of Lecture Notes in Computer Science, Springer, 1981, pp. 167–183. doi:https://doi.org/10.1007/BFb0017309.

[6] M. Ćirić, J. Ignjatović, N. Damljanović, M. Bašić, Bisimulations for fuzzy automata, Fuzzy Sets and Systems 186 (2012) 100–139. doi:https://doi.org/10.1016/j.fss.2011.07.003.

[7] P. Buchholz, Bisimulation relations for weighted automata, Theoretical Computer Science 393 (2008) 109–123. doi:https://doi.org/10.1016/j.tcs.2007.11.018.

[8] N. Damljanović, M. Ćirić, J. Ignjatović, Bisimulations for weighted automata over an additively idempotent semiring, Theoretical Computer Science 534 (2014) 86–100. doi:https://doi.org/10.1016/j.tcs.2014.02.032.

[9] M. Ćirić, I. Micić, J. Matejić, A. Stamenković, Simulations and bisimulations for max-plus automata, Discret. Event Dyn. Syst. 34 (2024) 269–295. doi:https://doi.org/10.1007/s10626-024-00395-1.

[10] F. Baccelli, G. Cohen, G. J. Olsder, J.-P. Quadrat, Synchronization and Linearity: An Algebra for Discrete Event Systems, Wiley, 1992.

[11] S. Lahaye, J. Komenda, J.-L. Boimond, Supervisory control of (max,+) automata: extensions towards applications, International Journal of Control 88 (12) (2015) 2523–2537. doi:10.1080/00207179.2015.1048295.

[12] I. Necoara, T. J. van den Boom, B. De Schutter, J. Hellendoorn, Stabilization of max-plus-linear systems using receding horizon control — the unconstrained case,

IFAC Proceedings Volumes 39 (5) (2006) 148–153. doi:https://doi.org/10.3182/20060607-3-IT-3902.00029.

[13] S. Gaubert, Performance evaluation of (max,+) automata, IEEE Transactions on Automatic Control 40 (12) (1995) 2014–2025. doi:https://doi.org/10.1109/9.478227.

[14] S. Gaubert, Methods and applications of (max, +) linear algebra, Lecture Notes in Computer Science 1200 (1997) 261–282. doi:https://doi.org/10.1007/BFb0023465.

[15] P. Butkovič, Max-linear Systems: Theory and Algorithms, Springer, 2010.

[16] I. Jančić, Weak bisimulations for fuzzy automata, Fuzzy Sets and Systems 249 (2014) 49–72. doi:https://doi.org/10.1016/j.fss.2013.10.006.

[17] J. Matejić, I. Micić, M. Ćirić, Weak simulations and bisimulations for max-plus automata, Filomat 40 (10) (2026).

[18] J. Matejić, M. Ćirić, J. Ignjatović, I. Micić, Truncating and shifting weights for max-plus automata, Axioms 15 (1) (2026). doi:10.3390/axioms15010079.

[19] I. Micić, M. Ćirić, J. Matejić, S. Stanimirović, L. A. Nguyen, Approximate weak simulations and bisimulations for fuzzy automata over the product structure, Fuzzy Sets and Systems 485 (2024) 108959. doi:https://doi.org/10.1016/j.fss.2024.108959.

[20] S. S. Epp, Discrete Mathematics with Applications, 5th Edition, Cengage Learning, 2019.

[21] G. Birkhoff, Lattice theory, American Mathematical Society, 1964.

[22] B. A. Davey, H. A. Priestley, Introduction to Lattices and Order, Cambridge University Press, 2005.

[23] T. S. Blyth, Lattices and Ordered Algebraic Structures, Springer, 2005.

[24] G. Birkhoff, T. C. Bartee, Modern applied algebra, The American Mathematical Monthly 79 (5) (1972) 529–530. doi:https://doi.org/10.2307/2317587.

[25] G. Cantor, On a property of the class of all real algebraic numbers, Crelle's Journal for Mathematics 77 (1874) 258–262.

[26] G. Cantor, A contribution to the theory of sets, Crelle's Journal for Mathematics 84 (1878) 119–133.

[27] B. Šešelja, A. Tepavčević, Algebra 1, Univerzitet u Novom Sadu, 2000.

[28] G. Grätzer, Universal Algebra, Springer, 1979.

[29] S. Bogdanović, M. Ćirić, Polugrupe, Prosveta Niš, 1993.

[30] J. M. Howie, Fundamentals of Semigroup Theory, Oxford University Press, 1995.

[31] M. Gondran, M. Minoux, Graphs, Dioids and Semirings: New Models and Algorithms, Springer New York, NY, 2008. doi:https://doi.org/10.1007/978-0-387-75450-5.

[32] K. Glazek, A Guide to the Literature on Semirings and their Applications in Mathematics and Information Sciences: With Complete Bibliography, Springer, 2002.

[33] U. Hebisch, H. J. Weinert, Semirings: Algebraic Theory and Applications in Computer Science, World Scientific Publishing Company, 1998.

[34] J. S. Baras, G. Theodorakopoulos, Path Problems in Networks, Springer, 2010.

[35] J. S. Golan, Semirings and their Applications, Springer, 1999.

[36] N. Galatos, P. Jipsen, T. Kowalski, H. Ono, Residuated Lattices: An Algebraic Glimpse at Substructural Logics, Elsevier, 2007.

[37] P. Hájek, Metamathematics of Fuzzy Logic, Springer Dordrecht, 1998.

[38] R. Belohlávek, Fuzzy Relational Systems: Foundations and Principles, Springer, 2002.

[39] R. Belohlávek, V. Vychodil, Fuzzy Equational Logic, Springer, 2005.

[40] L. A. Zadeh, Fuzzy sets, Information and Control 8 (1965) 338–353. doi:https://doi.org/10.1016/S0019-9958(65)90241-X.

[41] G. J. Klir, B. Yuan, Fuzzy Sets and Fuzzy Logic: Theory and Applications, Prentice Hall PTR, 1995.

[42] J. Ignjatović, M. Ćirić, Automati i formalni jezici, Prirodno-matematički fakultet, Niš, 2016.

[43] L. Wu, D. Qiu, Automata theory based on complete residuated lattice-valued logic: Reduction and minimization, Fuzzy Sets and Systems 161 (2010) 1635–1656. doi:https://doi.org/10.1016/j.fss.2009.12.011.

[44] M. Gondran, M. Minoux, Dioïds and semirings: Links to fuzzy sets and other applications, Fuzzy Sets and Systems 158 (12) (2007) 1273–1294. doi:https://doi.org/10.1016/j.fss.2007.01.016.

[45] J. Ignjatović, M. Ćirić, S. Bogdanović, Determinization of fuzzy automata with membership values in complete residuated lattices, Information Sciences 178 (2008) 164–180. doi:https://doi.org/10.1016/j.ins.2007.08.003.

[46] A. Stamenković, M. Ćirić, J. Ignjatović, Reduction of fuzzy automata by means of fuzzy quasi-orders, Information Sciences 275 (2014) 168–198. doi:https://doi.org/10.1016/j.ins.2014.02.028.

[47] Z. Jančić, J. Ignjatović, M. Ćirić, An improved algorithm for determinization of weighted and fuzzy automata, Information Sciences 181 (7) (2011) 1358–1368. doi:https://doi.org/10.1016/j.ins.2010.12.008.

[48] W. S. McCulloch, W. Pitts, A logical calculus of the ideas immanent in nervous activity, The Bulletin of Mathematical Biophysics 5 (4) (1943) 115–133. doi:https://doi.org/10.1007/bf02478259.

[49] S. C. Kleene, Representation of Events in Nerve Nets and Finite Automata, Princeton University Press, 1956. doi:https://doi.org/10.1515/9781400882618-002.

[50] M. O. Rabin, D. S. Scott, Finite automata and their decision problems, IBM J. Res. Dev. 3 (1959) 114–125. doi:https://doi.org/10.1147/rd.32.0114.

[51] D. C. Kozen, Automata and Computability, Springer, 1997.

[52] G. Rozenberg, A. Salomaa, Handbook of Formal Languages, Springer, 1997.

[53] J. E. Hopcroft, J. D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, 1979.

[54] F. Baccelli, G. Cohen, G. J. Olsder, J.-P. Quadrat, Synchronization and linearity - an algebra for discrete event systems, The Journal of the Operational Research Society 45 (1) (1994) 118–119. doi:https://doi.org/10.2307/2583959.

[55] M. Droste, P. Gastin, Weighted automata and weighted logics, Theoretical Computer Science 380 (1) (2007) 69–86. doi:https://doi.org/10.1016/j.tcs.2007.02.055.

[56] J. N. Mordeson, D. S. Malik, Fuzzy Automata and Languages: Theory and Applications, Chapman and Hall/CRC., 2002.

[57] S. Gaubert, J. Mairesse, Modeling and analysis of timed petri nets using heaps of pieces, IEEE Transactions on Automatic Control 44 (4) (1999) 683–698. doi:https://doi.org/10.1109/9.754807.

[58] J. Komenda, S. Lahaye, J.-L. Boimond, Supervisory control of (max,+) automata: A behavioral approach, Discrete Event Dynamic Systems 19 (4) (2009) 525–549. doi:https://doi.org/10.1007/s10626-009-0083-6.

[59] J. Komenda, S. Lahaye, J. L. Boimond, (max,+)-automata with partial observations, IFAC-PapersOnLine 51 (7) (2018) 192–197. doi:https://doi.org/10.1016/j.ifacol.2018.06.300.

[60] N. Lynch, F. Vaandrager, Forward and backward simulations, Information and Computation 121 (1995) 214–233. doi:https://doi.org/10.1006/inco.1995.1134.

[61] K. G. Larsen, C. Schilling, J. Srba, Simulation relations and applications in formal methods, in: Principles of Systems Design, Vol. 13660, Springer, 2022, pp. 373–397. doi:10.1007/978-3-031-22337-2_13.

[62] M. Ćirić, J. Ignjatović, P. Stanimirović, Bisimulations for weighted finite automata over semirings, Research Square (2022). doi:https://doi.org/10.21203/rs.3.rs-2386298/v1.

[63] M. Ćirić, J. Ignjatović, M. Bašić, I. Jančić, Nondeterministic automata: Equivalence, bisimulations, and uniform relations, Information Sciences 261 (2014) 185–218. doi:https://doi.org/10.1016/j.ins.2013.07.029.

[64] M. Holzer, M. Kutrib, Nondeterministic finite automata—recent results on the descriptional and computational complexity, Implementation and Applications of Automata 5148 (2008) 1–16. doi:10.1007/978-3-540-70844-5_1.

[65] N. Urabe, I. Hasuo, Quantitative simulations by matrices, Information and Computation 252 (2017) 110–137. doi:https://doi.org/10.48550/arXiv.1810.09146.

[66] S. Lahaye, A. Lai, J. Komenda, J.-L. Boimond, A contribution to the determinization of max-plus automata, Discrete Event Dynamic Systems 30 (2020) 155–174. doi:https://doi.org/10.1007/s10626-019-00309-6.

[67] B. Daviaud, S. Lahaye, M. Lhommeau, J. Komenda, On the existence of simulations for max-plus automata, IEEE Control Systems Letters 8 (2024) 694–699. doi:https://doi.org/10.1109/LCSYS.2024.3401415.

[68] M. Ćirić, J. Ignjatović, I. Jančić, N. Damljanović, Computation of the greatest simulations and bisimulations between fuzzy automata, Fuzzy Sets and Systems 208 (2012) 22–42. doi:https://doi.org/10.1016/j.fss.2012.05.006.

[69] A. Stamenković, M. Ćirić, D. Đurđanović, Weakly linear systems for matrices over the max-plus quantale, Discrete Event Dynamic Systems 32 (2022) 1–25. doi:https://doi.org/10.1007/s10626-021-00342-4.

[70] A. Dovier, C. Piazza, A. Policriti, A fast bisimulation algorithm, Lecture Notes in Computer Science (2001) 79–90doi:https://doi.org/10.1007/3-540-44585-4_8.

[71] A. Dovier, C. Piazza, A. Policriti, An efficient algorithm for computing bisimulation equivalence, Theoretical Computer Science 311 (1) (2004) 221–256. doi:https://doi.org/10.1016/S0304-3975(03)00361-X.

[72] Z. Jančić, I. Micić, M. Ćirić, J. R. González de Mendívil, S. Stanimirović, Approximate determinization of fuzzy automata over the product structure by means of approximate weak simulations, Fuzzy Sets and Systems 523 (2026) 109625. doi:https://doi.org/10.1016/j.fss.2025.109625.

[73] Z. Jančić, I. Micić, J. Ignjatović, M. Ćirić, Further improvements of determinization methods for fuzzy finite automata, Fuzzy Sets and Systems 301 (2016) 79–102. doi:https://doi.org/10.1016/j.fss.2015.11.019.

[74] Z. Jančić, J. R. G. de Mendívil, I. Micić, M. Ćirić, Approximate children fuzzy automata over the product structure, Filomat 40 (1) (2026) 309–320. doi:https://doi.org/10.2298/FIL2601309J.

[75] I. Micić, S. Stanimirović, J. R. González de Mendívil, M. Ćirić, Z. Jančić, Finite determinization of fuzzy automata using a parametric product-based t-norm, Fuzzy Sets and Systems 488 (2024) 108990. doi:https://doi.org/10.1016/j.fss.2024.108990.

[76] S. Stanimirović, M. Ćirić, J. Ignjatović, Determinization of fuzzy automata by factorizations of fuzzy states and right invariant fuzzy quasi-orders, Information Sciences 469 (2018) 79–100. doi:https://doi.org/10.1016/j.ins.2018.08.033.

[77] M. Stanković, M. Ćirić, J. Ignjatović, Simulations and bisimulations for fuzzy multimodal logics over heyting algebras, Filomat 37 (3) (2023) 711–743. doi:https://doi.org/10.2298/FIL2303711S.

[78] I. Micić, Z. Jančić, M. Ćirić, Simulations for fuzzy automata over the product structure: From approximation to exact solution, Fuzzy Sets and Systems 539 (2026) 109923. doi:https://doi.org/10.1016/j.fss.2026.109923.

[79] S. Lahaye, J. Komenda, J.-L. Boimond, Compositions of (max,+) automata, IFAC Proceedings Volumes 45 (29) (2012) 61–66. doi:https://doi.org/10.3182/20121003-3-MX-4033.00013.

[80] I. Micić, J. Matejić, S. Stanimirović, L. A. Nguyen, Towards new types of weak bisimulations for fuzzy automata using the product t-norm, in: Fuzzy Logic and Technology, and Aggregation Operators, Springer Nature Switzerland, 2023, pp. 567–578. doi:https://doi.org/10.1007/978-3-031-39965-7_47.

[81] L. A. Nguyen, I. Micić, S. Stanimirović, Depth-bounded fuzzy simulations and bisimulations between fuzzy automata, Fuzzy Sets and Systems 473 (2023) 108729. doi:https://doi.org/10.1016/j.fss.2023.108729.

[82] S. Stanimirović, I. Micić, M. Ćirić, Approximate bisimulations for fuzzy automata over complete heyting algebras, IEEE Transactions on Fuzzy Systems 30 (2022) 437–447. doi:https://doi.org/10.1109/TFUZZ.2020.3039968.

[83] L. A. Nguyen, Fuzzy simulations and bisimulations between fuzzy automata, International Journal of Approximate Reasoning 155 (2023) 113–131. doi:https://doi.org/10.1016/j.ijar.2023.02.002.

[84] M. Ćirić, J. Ignjatović, S. Bogdanović, Fuzzy equivalence relations and their equivalence classes, Fuzzy Sets and Systems 158 (2007) 1295–1313. doi:https://doi.org/10.1016/j.fss.2007.01.010.

[85] M. Ćirić, J. Ignjatović, S. Bogdanović, Uniform fuzzy relations and fuzzy functions, Fuzzy Sets and Systems 160 (2009) 1054–1081. doi:https://doi.org/10.1016/j.fss.2008.07.006.

[86] M. B. Demirci, J. R. Ferrés, Fuzzy groups, fuzzy functions and fuzzy equivalence relations, Fuzzy Sets and Systems 144 (3) (2004) 441–458. doi:https://doi.org/10.1016/S0165-0114(03)00301-4.

[87] I. Micić, L. A. Nguyen, S. Stanimirović, Characterization and computation of approximate bisimulations for fuzzy automata, Fuzzy Sets and Systems 442 (2022) 331–350. doi:https://doi.org/10.1016/j.fss.2022.05.003.

[88] Y. Li, K. Qin, X. He, Some new approaches to constructing similarity measures, Fuzzy Sets and Systems 234 (2014) 46–60. doi:https://doi.org/10.1016/j.fss.2013.03.008.

Dodatak A

Program za računanje simulacija i bisimulacija nad različitim težinskim strukturama

U okviru istraživanja predstavljenih u ovoj disertaciji razvijen je program za računanje različitih vrsta simulacija i bisimulacija nad težinskim automatima. Program predstavlja praktičnu implementaciju teorijskih rezultata razmatranih u prethodnim poglavljima i omogućava izračunavanje odgovarajućih relacija za zadate težinske automate.

Program podržava računanje simulacija i bisimulacija nad različitim algebarskim strukturama, uključujući kompletan max-plus poluprsten, kompletan max-plus dioid i proizvod strukturu, kao i odgovarajuće odsečene strukture. U zavisnosti od izabrane težinske strukture, moguće je računati različite vrste klasičnih i slabih simulacija i bisimulacija, u skladu sa teorijskim rezultatima predstavljenim u prethodnim poglavljima.

Program podržava računanje simulacija i bisimulacija nad različitim algebarskim strukturama. Početna stranica programa organizovana je prema težinskim strukturama, pri čemu su trenutno podržane dve osnovne grupe: max-plus struktura (kompletan max-plus poluprsten i kompletan max-plus dioid) i proizvod struktura. Za pojedine strukture dostupne su i odgovarajuće odsečene varijante. Upravo izbor strukture predstavlja prvi korak u postupku računanja, nakon čega korisnik bira odgovarajući tip simulacije ili bisimulacije.

U okviru kompletnog max-plus poluprstena omogućeno je računanje klasičnih simulacija i bisimulacija, kao i slabih simulacija i bisimulacija. Pored toga, u okviru odsečenih struktura dostupne su donje i gornje odsečene varijante, za koje je omogućeno računanje odgovarajućih vrsta simulacija i bisimulacija. Za kompletnu max-plus poluprstenu implementirano je i računanje slabih simulacija i bisimulacija ograničene dužine.

Za kompletan max-plus dioid program trenutno omogućava računanje klasičnih simulacija i bisimulacija, dok je u okviru gornje odsečene varijante omogućeno odgovarajuće računanje nad gornje odsečenim kompletnim max-plus dioidom.

Posebna grupa na početnoj stranici programa odnosi se na proizvod strukturu. Nad ovom strukturom omogućeno je računanje klasičnih i slabih simulacija i bisimulacija, dok je u okviru odsečene varijante implementirano računanje ε-slabih simulacija i bisimulacija. Na predstavljen način, organizacija programa prati različite algebarske strukture i vrste relacija razmatrane u teorijskom delu disertacije.

Korisnički interfejs programa organizovan je tako da korisnik najpre bira težinsku strukturu, a zatim odgovarajuću vrstu računanja (Slika 17). Nakon izbora, učitavaju se ulazni podaci koji predstavljaju težinske automate (Slike 18, 19, 20). Na osnovu unetih podataka program formira odgovarajuće matrice i, primenom implementiranih algoritama,

izračunava tražene relacije. Dobijeni rezultati prikazuju se u matričnom obliku, čime se omogućavaju njihova neposredna analiza i poređenje.

Program je implementiran korišćenjem programskog jezika Python na serverskoj strani i JavaScript-a na klijentskoj strani. Serverska aplikacija razvijena je korišćenjem programskog okvira FastAPI, dok se za pokretanje serverskog procesa koristi Uvicorn. Za generisanje HTML stranica koriste se Jinja2 šabloni, a za prihvatanje podataka i datoteka korišćen je paket python-multipart. Matematički deo implementacije realizovan je u Python-u, bez korišćenja dodatnih biblioteka za računanje.

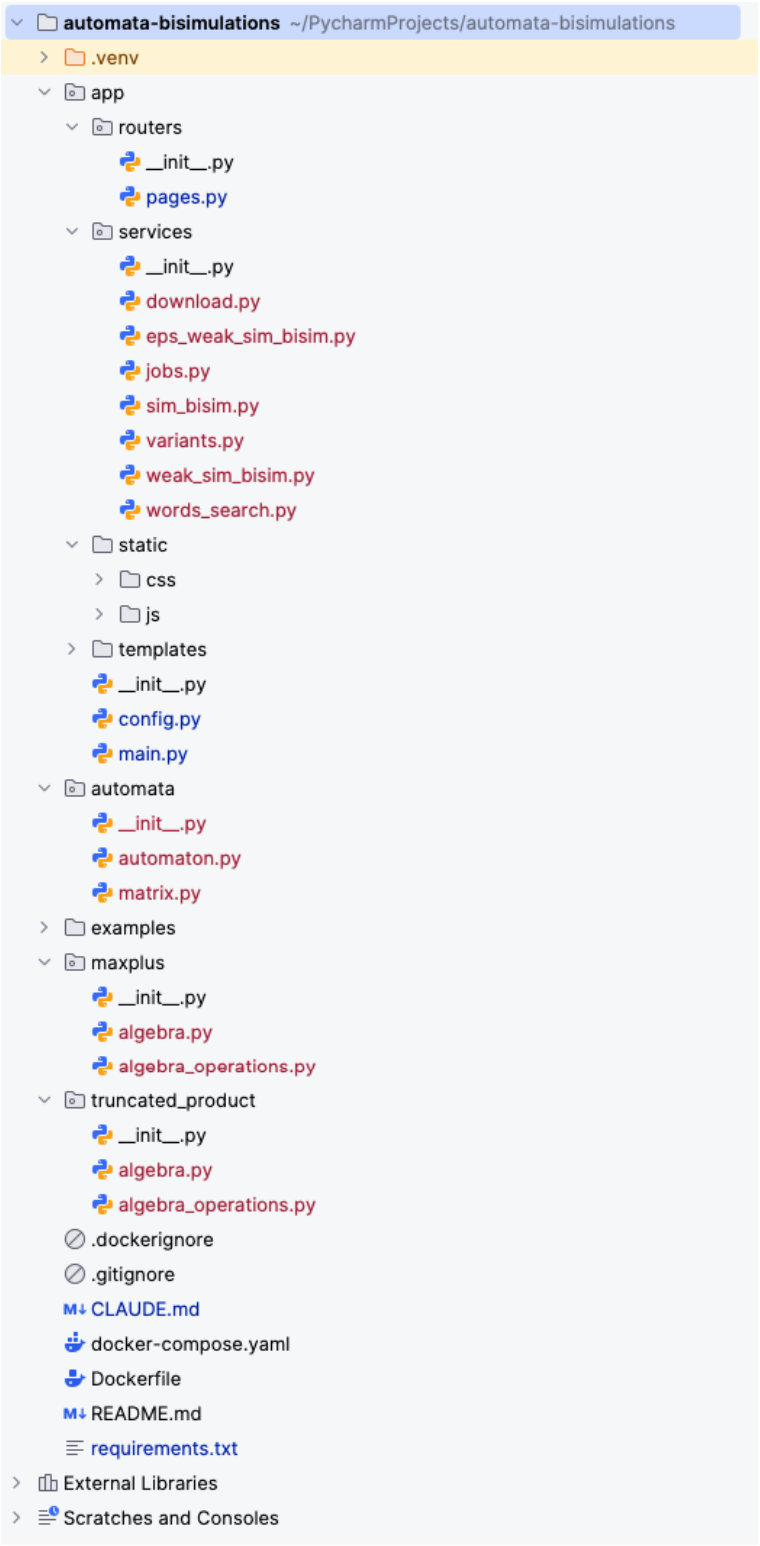
Klijentski deo aplikacije zasnovan je na Jinja2 šablonima koji se generišu na serverskoj strani. Za oblikovanje korisničkog interfejsa korišćen je Bootstrap 5, za asinhronu komunikaciju sa serverskom stranom jQuery, dok je za neke funkcionalnosti korišćen standardni JavaScript. Aplikacija ne koristi frontend okvire poput React-a, niti zahteva proces izgradnje (build) ili kompajliranje klijentskog koda, već se statički fajlovi direktno plasiraju korisniku.

Za distribuciju i pokretanje aplikacije korišćena je tehnologija Docker. Konfiguracija projekta obuhvata datoteku `Dockerfile`, kojom se definiše okruženje potrebno za pokretanje aplikacije, i datoteku docker-compose.yaml, kojom se definiše način pokretanja i povezivanja potrebnih servisa.

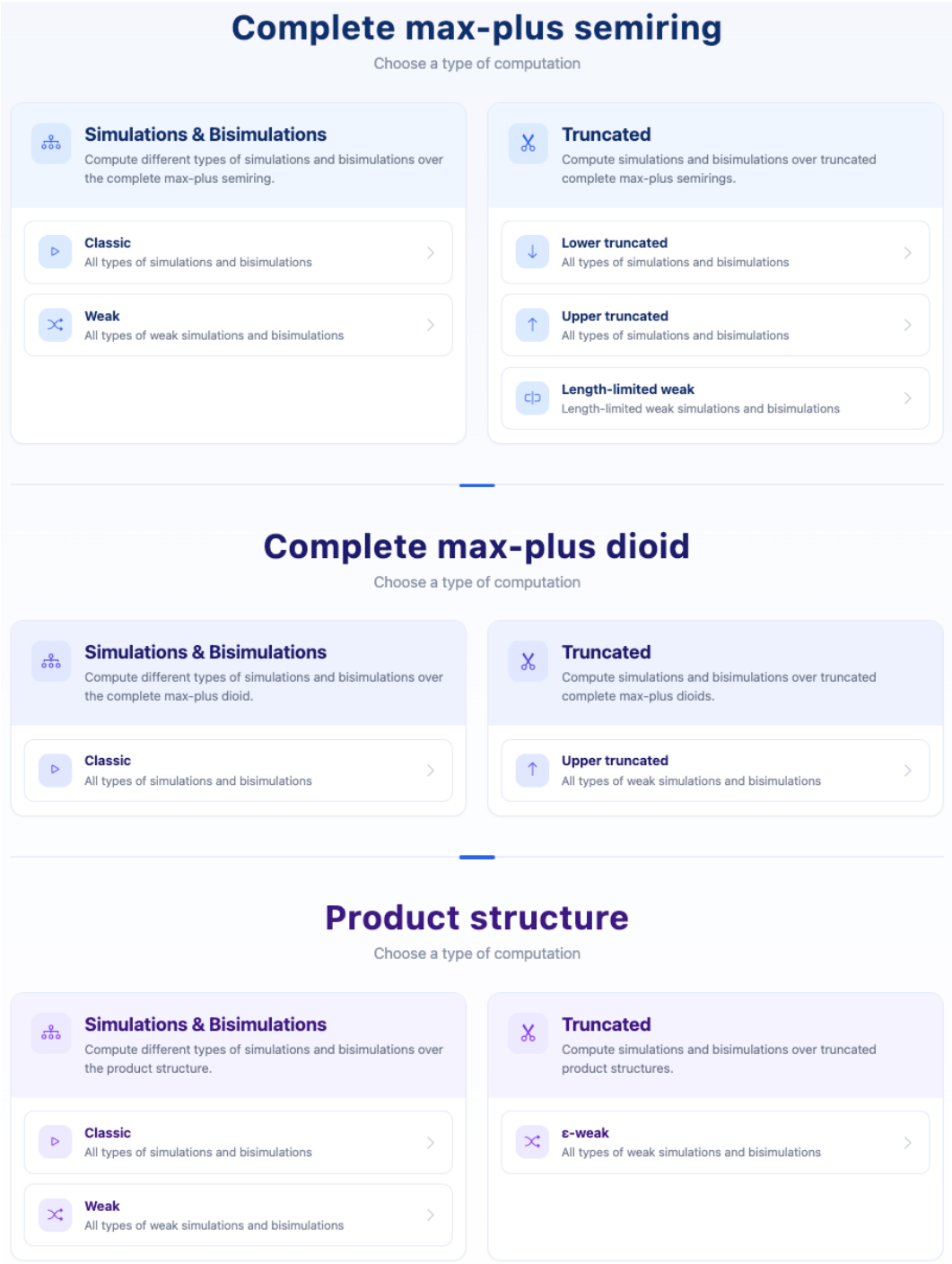
Klijentski deo aplikacije zasnovan je na Jinja2 šablonima koji se generišu na serverskoj strani. Za oblikovanje korisničkog interfejsa korišćen je Bootstrap 5, dok se za asinhronu komunikaciju sa serverskom stranom koristi jQuery. Dodatne funkcionalnosti realizovane su korišćenjem standardnog JavaScript-a.

Celokupan programski kod projekta javno je dostupan na GitHub repozitorijumu `https://github.com/JelenaMatejic/automata-bisimulations`. Verzija koda koja odgovara stanju projekta u trenutku pisanja ove doktorske disertacije, septembra 2026. godine, data je i u nastavku ovog dodatka.

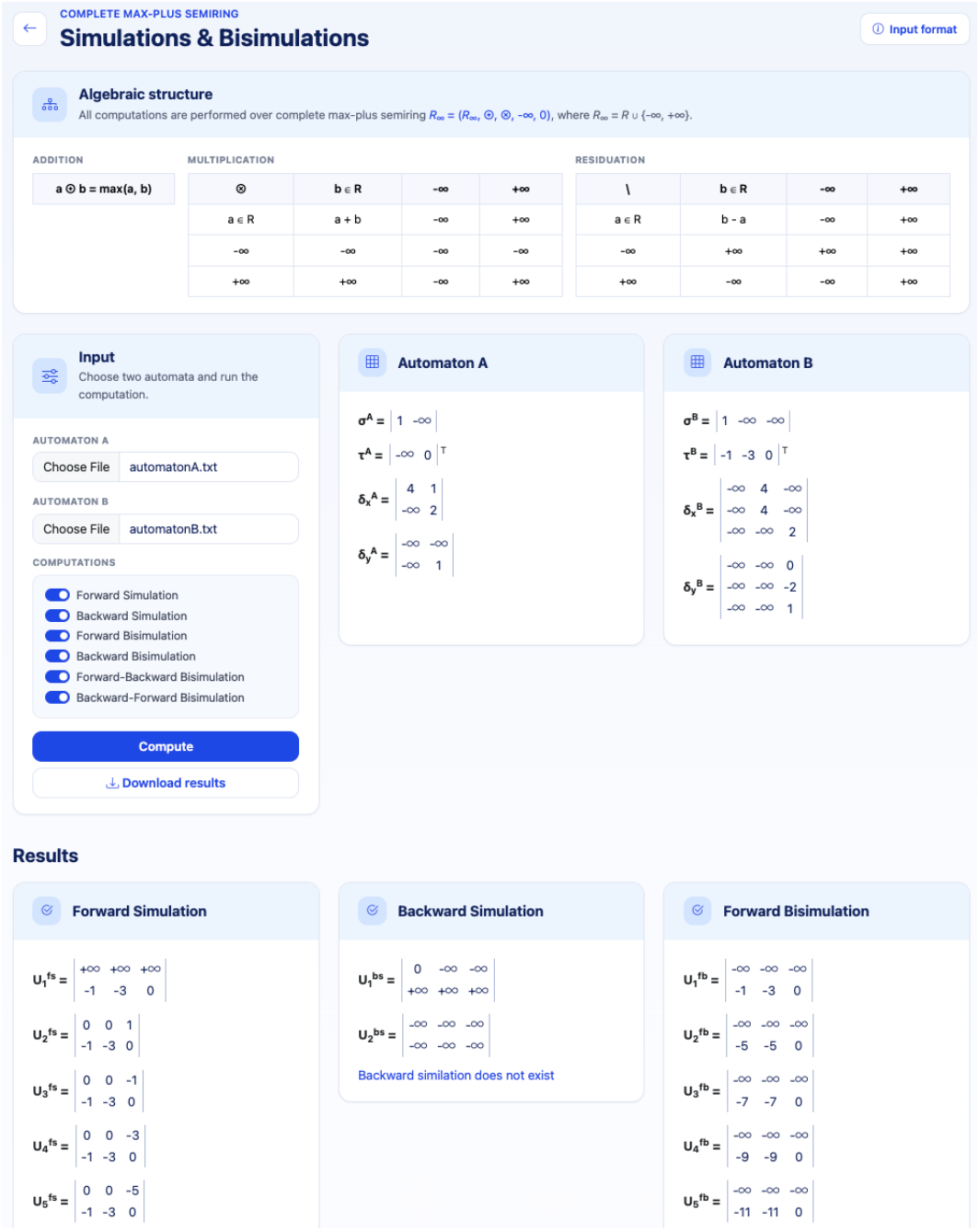
Na slici 16 prikazana je struktura foldera i datoteka projekta. Programska implementacija (u nastavku) nije predstavljena redosledom kojim se datoteke pojavljuju u projektu, već je organizovana prema logičkim celinama. Najpre su prikazane osnovne komponente programa, nakon čega slede implementacije operacija nad matricama i težinskim strukturama, a zatim algoritmi za računanje pojedinačnih vrsta simulacija i bisimulacija. Ovakva organizacija omogućava postupno upoznavanje sa implementacijom i lakše povezivanje programskog koda sa teorijskim rezultatima predstavljenim u prethodnim poglavljima.



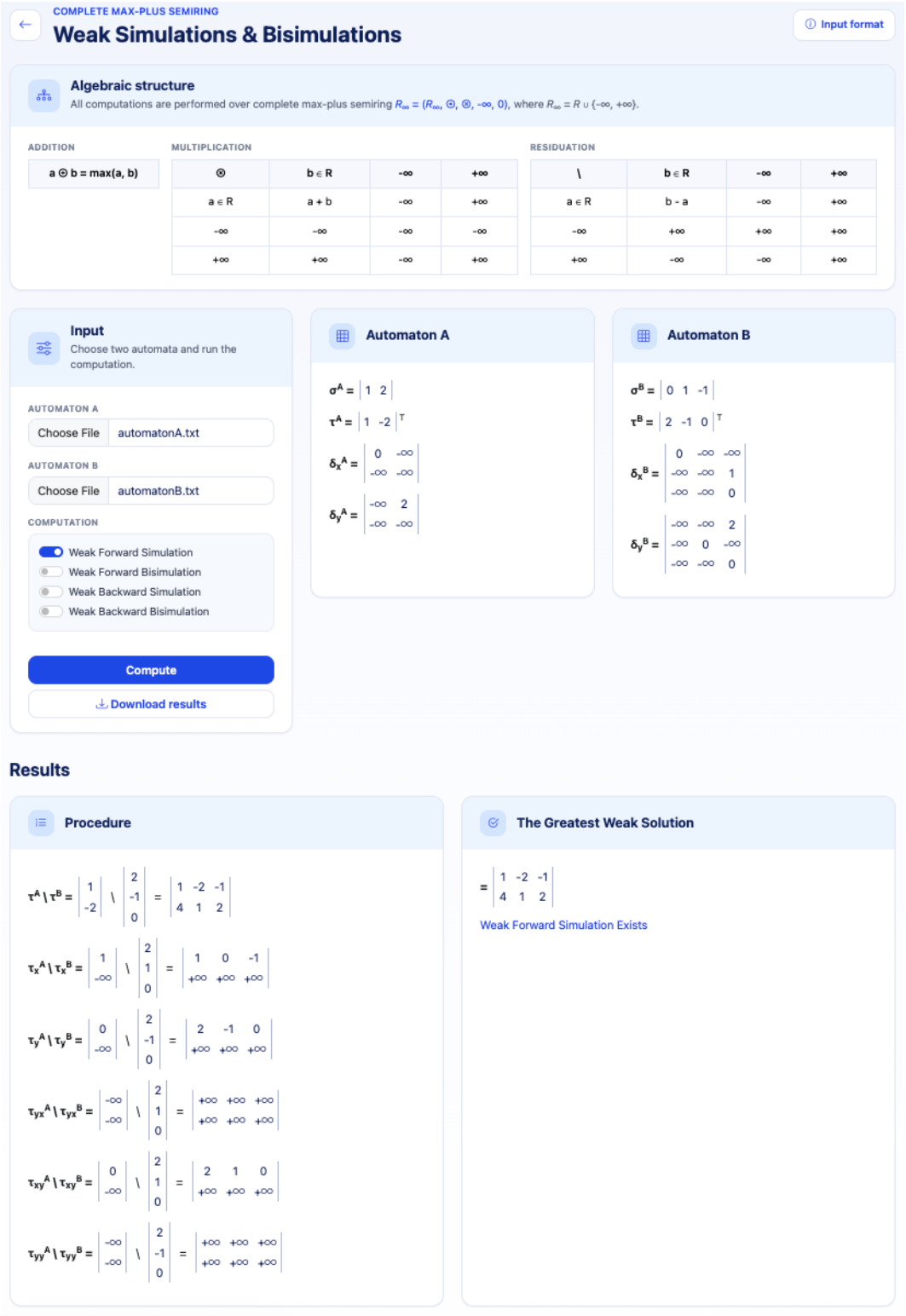
Slika 16: Stablo foldera programa



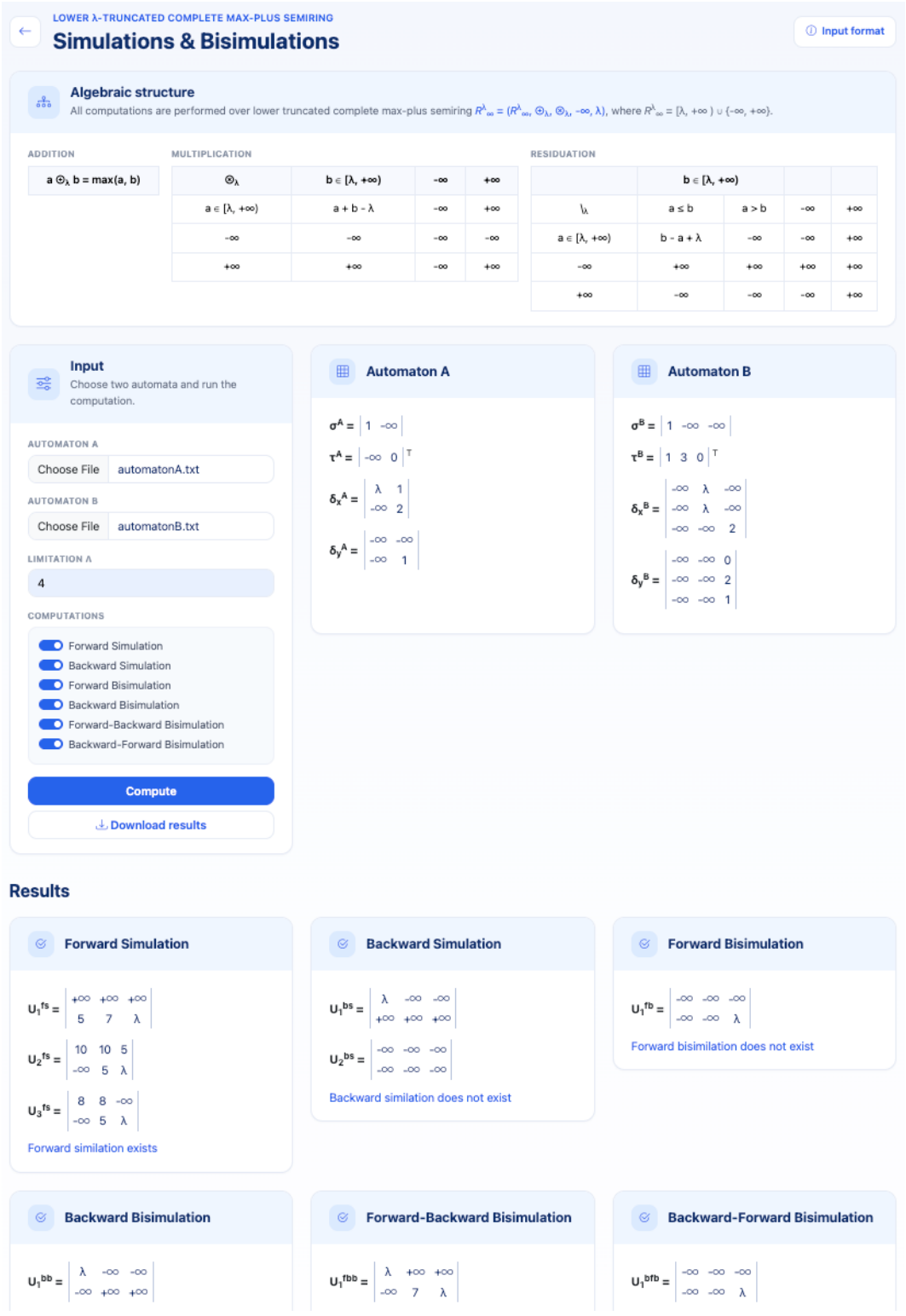
Slika 17: Početna stranica programa



Slika 18: Stranica za računanje (bi)simulacija nad komoletnim max-plus poluprstenom



Slika 19: Računanje slabih (bi)simulacija nad komoletnim max-plus poluprstenom



Slika 20: Računanje odselenih (bi)simulacija nad komoletnim max-plus poluprstenom

Izvorni kod 1: automata/matrix.py

```
class Matrix:
    def __init__(self, name, value):
        self.name = name
        self.value = value
        self.rows = len(self.value)  # n - number of rows in matrix
        self.cols = len(self.value[0])  # m - number of cols in matrix

    def transpose(self):
        value_tmp = [[self.value[j][i] for j in range(self.rows)] for i in range(self.cols)]
        return Matrix(self.name + "^T", value_tmp)

    @staticmethod
    def convert_inf_to_str(data):
        # +/i infty is not valid JSON, so it travels as a string.
        if isinstance(data, list):
            return [Matrix.convert_inf_to_str(item) for item in data]
        return str(data) if data == -float('inf') or data == float('inf') else data

    def get_matrix(self):
        my_dict = self.__dict__.copy()
        my_dict['value'] = Matrix.convert_inf_to_str(my_dict['value'])
        return my_dict

    def print_matrix_in_file(self, filepath, floor=None):
        with open(filepath, 'a') as file:
            file.write(self.name + '\n')
            for row in self.value:
                cells = ['epsilon' if floor is not None and element == floor else str(element)
                         for element in row]
                file.write(','.join(cells) + '\n')
            file.write('-------------------------------' + '\n')

def matrix_min(matrix_n, matrix_m):
    # Element-wise minimum - Matrix N of dimensions n x m; Matrix M of dimensions n x m;
    if matrix_n.cols != matrix_m.cols or matrix_n.rows != matrix_m.rows:
        raise ValueError(
            "The number of rows and cols in the first matrix must match the number of rows and cols
     in the second matrix.")

    result = [[0] * matrix_n.cols for _ in range(matrix_n.rows)]

    for i in range(matrix_n.rows):
        for j in range(matrix_n.cols):
            result[i][j] = min(matrix_n.value[i][j], matrix_m.value[i][j])

    return Matrix('matrix_min', result)

def matrix_ordering(matrix_n, matrix_m):
    # Is N <= M element-wise?
    for i in range(matrix_n.rows):
        for j in range(matrix_n.cols):
            if (matrix_n.value[i][j] > matrix_m.value[i][j]):
                return False
    return True
```

Izvorni kod 2: automata/automaton.py

```
from automata.matrix import Matrix

class InvalidFileFormatError(Exception):
    pass

class Automaton:

    def __init__(self, filepath, floor=None):
        self.n = None
        self.sigma = None
        self.tau = None
        self.transitions = []
        self.floor = floor # for structures whose smallest value is not 0

        self.read_file(filepath)

    def read_file(self, filepath):
        with open(filepath, 'r') as file:
            # Read the entire file, Remove newline characters, Check if the file format is valid
            lines = file.readlines()
            lines = [line.strip() for line in lines if line.strip()]

            if len(lines) < 7 or lines[0] != 'states' or lines[2] != 'start' or lines[4] != 'end':
                print('Invalid file format')
                raise InvalidFileFormatError("Invalid file format")

            # Read: number of states, start vector, end vector, transitions
            self.n = int(lines[1])

            start_vector = [self._parse(x) for x in lines[3].split(',')]
            self.sigma = Matrix('start', [start_vector])

            end_vector = [self._parse(x) for x in lines[5].split(',')]
            self.tau = Matrix('end', [end_vector])

            current_transition = None
            for line in lines[6:]:
                if line.startswith('transition'):
                    current_transition = line.split(' ')[1][-1]
                    matrix = Matrix(current_transition, [[]])
                    matrix.value = []
                    matrix.rows = 0
                    self.transitions.append(matrix)
                else:
                    if current_transition is not None:
                        row = [self._parse(x) for x in line.split(',')]
                        matrix.value.append(row)
                        matrix.rows += 1
                        matrix.cols = len(row)

    def _parse(self, value):
        parsed = Automaton.parse_value(value)
        if self.floor is not None and parsed == 0:
            return self.floor
        return parsed
```

```

    @staticmethod
    def parse_value(value):
        if value == '-' or value == '-inf':
            return float('-inf')
        elif value == '+' or value == 'inf':
            return float('inf')
        else:
            return float(value)

def getResponseAutomaton(automaton):
    responses = {}
    r = automaton.sigma.get_matrix()
    responses[r['name']] = r
    r = automaton.tau.get_matrix()
    responses[r['name']] = r
    for i in range(len(automaton.transitions)):
        r = automaton.transitions[i].get_matrix()
        responses[r['name']] = r
    return responses
```

Izvorni kod 3: maxplus/algebra.py

```
from cmath import inf

class Algebra:

    def __init__(self, name, times, residual, zero, top):
        self.name = name
        self.times = times
        self.residual = residual
        self.zero = zero
        self.top = top

    def __repr__(self):
        return "Algebra(%r)" % self.name

# --- scalar operations: max-plus -----------------------------------------------------

def _maxplus_times(a, b):
    if a == -inf or b == -inf:
        return -inf
    elif a == inf or b == inf:
        return inf
    else:
        return a + b

def _semiring_residual(a, b):
    if a == -inf or b == inf:
        return inf
    elif a == inf or b == -inf:
        return -inf
    else:
        return b - a

def _dioid_residual(a, b):
    if a == -inf or b == inf:
```

```
        return inf
    elif a == inf or b == -inf or a > b:
        return -inf
    else:
        return b - a

# --- scalar operations: product -----------------------------------------------------

def _product_times(a, b):
    return a * b

def _product_residual(a, b):
    if a <= b:
        return 1
    return b / a

# --- scalar operations: truncated max-plus ------------------------------------------

def _truncated_times(limitation):
    def times(a, b):
        if a == -inf or b == -inf:
            return -inf
        elif a == inf or b == inf:
            return inf
        else:
            return a + b - limitation

    return times

def _delta_truncated_times(delta):
    def times(a, b):
        if a == -inf or b == -inf:
            return -inf
        elif a + b < delta:
            return a + b
        return inf

    return times

def _lower_residual(limitation):
    def residual(a, b):
        if a == -inf or b == inf:
            return inf
        elif a == inf or b == -inf or a > b:
            return -inf
        else:
            return b - a + limitation

    return residual

def _upper_residual(limitation):
    def residual(a, b):
        if a == -inf or b == inf:
            return inf
        elif a == inf or b == -inf:
            return -inf
        elif a < b:
```

```
            return limitation
        else:
            return b - a + limitation

    return residual

# --- the structures ------------------------------------------------------

SEMIRING = Algebra("semiring", _maxplus_times, _semiring_residual, -inf, inf)
DIOID = Algebra("dioid", _maxplus_times, _dioid_residual, -inf, inf)
PRODUCT = Algebra("product", _product_times, _product_residual, 0, 1)

def lower(limitation):
    return Algebra("lower", _truncated_times(limitation), _lower_residual(limitation), -inf, inf)

def upper(limitation):
    return Algebra("upper", _truncated_times(limitation), _upper_residual(limitation), -inf, inf)

def upper_dioid(delta):
    return Algebra("upper_dioid", _delta_truncated_times(delta), _dioid_residual, -inf, inf)

_BY_NAME = {"semiring": SEMIRING, "dioid": DIOID, "product": PRODUCT}

def of(name):
    try:
        return _BY_NAME[name]
    except KeyError:
        raise ValueError("Unknown structure: %r" % name)
```

Izvorni kod 4: max_plus/algebra_operations.py

```
from automata.matrix import Matrix

def multiply(matrix_m, matrix_n, algebra):
    # Matrix M of dimensions m x n; Matrix N of dimensions n x p; Matrix M * N dimensin m x p;
    if matrix_m.cols != matrix_n.rows:
        raise ValueError(
            "The number of columns in the first matrix must equal the number of rows in the second.")

    times = algebra.times
    result = []
    for i in range(matrix_m.rows):
        row = []
        for j in range(matrix_n.cols):
            element = algebra.zero
            for k in range(matrix_n.rows):
                a = matrix_m.value[i][k]
                b = matrix_n.value[k][j]
                element = max(element, times(a, b))
            row.append(element)
        result.append(row)

    return Matrix("Result", result)

def right_residual(matrix_m, matrix_n, algebra):
    # Matrix M of dimensions m x n; Matrix N of dimensions m x p; Matrix M\N dimensin n x p;
    if matrix_m.rows != matrix_n.rows:
```

```
        raise ValueError("The number of rows in the first matrix must equal the number of rows in the second.")

    residual = algebra.residual
    result = [[algebra.top] * matrix_n.cols for _ in range(matrix_m.cols)]

    for j in range(matrix_m.cols):
        for k in range(matrix_n.cols):
            for i in range(matrix_m.rows):
                a = matrix_m.value[i][j]
                b = matrix_n.value[i][k]
                result[j][k] = min(result[j][k], residual(a, b))

    return Matrix('Right Residual', result)

def left_residual(matrix_n, matrix_m, algebra):
    # Matrix N of dimensions m x p; Matrix M of dimensions n x p; Matrix N/M dimensin m x n = M -> N;
    if matrix_n.cols != matrix_m.cols:
        raise ValueError("The number of rows in the first matrix must match the number of rows in the second matrix.")

    residual = algebra.residual
    result = [[algebra.top] * matrix_m.rows for _ in range(matrix_n.rows)]

    for i in range(matrix_n.rows):
        for j in range(matrix_m.rows):
            for k in range(matrix_m.cols):
                m = matrix_m.value[j][k]
                n = matrix_n.value[i][k]
                result[i][j] = min(result[i][j], residual(m, n))

    return Matrix('Right Residual', result)
```

Izvorni kod 5: truncated_product/algebra.py

```
class Algebra:
    def __init__(self, name, times, sim, floor):
        self.name = name
        self.times = times
        self.sim = sim
        self.floor = floor

    def __repr__(self):
        return "Algebra(%r)" % self.name

def _product_times(a, b):
    return a * b

def _similarity(x, y):
    if x == y:
        return 1.0
    return min(x, y) / max(x, y)


def truncated(epsilon):
    return Algebra("truncated_product", _product_times, _similarity, epsilon)
```

Izvorni kod 6: truncated_product/algebra_operations.py

```
from cmath import inf
from automata.matrix import Matrix

def multiply(matrix_m, matrix_n, algebra):
    # Matrix M of dimensions m x n; Matrix N of dimensions n x p; Matrix M * N dimensin m x p;
    if matrix_m.cols != matrix_n.rows:
        raise ValueError(
            "The number of columns in the first matrix must equal the number of rows in the second.")

    times, floor = algebra.times, algebra.floor
    result = []
    for i in range(matrix_m.rows):
        row = []
        for j in range(matrix_n.cols):
            element = -inf
            for k in range(matrix_n.rows):
                a = matrix_m.value[i][k]
                b = matrix_n.value[k][j]
                element = max(element, times(a, b))
            # A coordinate this low means a very low acceptance probability.
            if element <= floor and element != 0:
                element = floor
            row.append(element)
        result.append(row)

    return Matrix("Result", result)
```

Izvorni kod 7: services/jobs.py

```
import os
import shutil
import uuid

from app import config

VARIANT_FILE = ".variant"

def job_dir(job_id):
    return os.path.join(config.WORK_ROOT, job_id)

def selected_options(flags):
    # The six sim/bisim booleans
    return [name for name in config.SIM_BISIM_OPTIONS if flags.get(name)]

def _prepare_job(file1_bytes, file2_bytes):
    job_id = uuid.uuid4().hex
    directory = job_dir(job_id)
    for sub in config.RESULT_SUBDIRS:
        os.makedirs(os.path.join(directory, sub), exist_ok=True)

    path_a = os.path.join(directory, "automatonA.txt")
    path_b = os.path.join(directory, "automatonB.txt")
    with open(path_a, "wb") as f:
        f.write(file1_bytes)
    with open(path_b, "wb") as f:
```

```
        f.write(file2_bytes)
    return job_id, directory, path_a, path_b

def _append_conclusions(directory, variant_key, result):
    # Append each conclusion to its result file, for the download.
    sources = config.DOWNLOAD_SOURCES.get(variant_key, [])
    for key, message in result.items():
        filename = config.RESULT_FILES.get(key)
        if not filename or not isinstance(message, str) or not message.strip():
            continue
        target = next(
            (src for src in reversed(sources)
             if os.path.isfile(os.path.join(directory, src, filename))),
            None,
        )
        if target is not None:
            with open(os.path.join(directory, target, filename), "a", encoding="utf-8") as f:
                f.write("\n" + message.strip() + "\n")

def run(variant_key, file1_bytes, file2_bytes, *, flags=None, limitation=None, mode=None,
        word_length=None, delta=None, threshold=None):
    variant = config.VARIANTS[variant_key]
    compute = variant["compute"]
    kind = variant["kind"]

    job_id, directory, path_a, path_b = _prepare_job(file1_bytes, file2_bytes)

    # Remember which page produced this job
    with open(os.path.join(directory, VARIANT_FILE), "w") as f:
        f.write(variant_key)

    # All 10 processes are divided into 6 categories (i.e. ismbisim: max-plus, max-plus-dioid,
     product ...)
    if kind == "simbisim":
        result = compute(directory, path_a, path_b, selected_options(flags or {}))
    elif kind == "simbisim_limit":
        result = compute(directory, path_a, path_b, selected_options(flags or {}), limitation)
    elif kind == "weak_length":
        result = compute(directory, path_a, path_b, mode, word_length)
    elif kind == "weak_delta":
        result = compute(directory, path_a, path_b, mode, delta)
    elif kind == "weak_threshold":
        result = compute(directory, path_a, path_b, mode, threshold)
    else:
        result = compute(directory, path_a, path_b, mode)

    _append_conclusions(directory, variant_key, result)

    result["jobId"] = job_id
    return result

def reset_work_root():
    shutil.rmtree(config.WORK_ROOT, ignore_errors=True)
    os.makedirs(config.WORK_ROOT, exist_ok=True)
```

Izvorni kod 8: services/variants.py

```
from maxplus import algebra
from app.services import eps_weak_sim_bisim, sim_bisim, weak_sim_bisim

_MAXPLUS = {
    'forwardSimulation': "Forward similation does not find in 10 iterations.",
    'backwardSimulation': "Backward similation not found in 10 iterations.",
    'forwardBisimulation': "Forward bisimilation not found in 10 iterations.",
    'backwardBisimulation': "Backward bisimilation not found in 10 iterations.",
    'forwardBackwardBisimulation': "Forward-backward bisimilation not found in 10 iterations.",
    'backwardForwardBisimulation': "Backward-forward bisimilation not found in 10 iterations",
}

_TRUNCATED = {
    'forwardSimulation': "Forward similation does not find in 10 iterations.",
    'backwardSimulation': "Backward similation does not find in 10 iterations.",
    'forwardBisimulation': "Forward bisimilation does not find in 10 iterations.",
    'backwardBisimulation': "Backward bisimilation does not find in 10 iterations.",
    'forwardBackwardBisimulation': "Forward-backward bisimilation does not exist",
    'backwardForwardBisimulation': "Backward-forward bisimilation does not exist",
}

# --- classic (bi)simulations ------------------------------------------------

def _run(structure, output_dir, messages, base_dir, file1, file2, selected_options):
    return sim_bisim.compute(base_dir, file1, file2, selected_options,
                             structure, output_dir, messages)

def semiring(base_dir, file1, file2, selected_options):
    return _run(algebra.SEMIRING, 'results_max_plus', _MAXPLUS,
                base_dir, file1, file2, selected_options)

def dioid(base_dir, file1, file2, selected_options):
    return _run(algebra.DIOID, 'results_max_plus', _MAXPLUS,
                base_dir, file1, file2, selected_options)

def product(base_dir, file1, file2, selected_options):
    return _run(algebra.PRODUCT, 'results_product', _MAXPLUS,
                base_dir, file1, file2, selected_options)

def lower_truncation(base_dir, file1, file2, selected_options, limitation):
    return _run(algebra.lower(limitation), 'results_max_plus_lower_truncation', _TRUNCATED,
                base_dir, file1, file2, selected_options)

def upper_truncation(base_dir, file1, file2, selected_options, limitation):
    return _run(algebra.upper(limitation), 'results_max_plus_upper_truncation', _TRUNCATED,
                base_dir, file1, file2, selected_options)

# --- weak (bi)simulations ---------------------------------------------------

def max_plus_weak(base_dir, file1, file2, mode):
    return weak_sim_bisim.compute(base_dir, file1, file2, mode)

def length_limited(base_dir, file1, file2, mode, word_length):
    return weak_sim_bisim.compute(base_dir, file1, file2, mode, max_length=word_length,
                                  output_dir="results_length_limited")
```

```

def upper_truncated_dioid(base_dir, file1, file2, mode, delta):
    return weak_sim_bisim.compute(base_dir, file1, file2, mode,
                                  output_dir="results_upper_truncated_dioid",
                                  structure=algebra.upper_dioid(delta))

def product_weak(base_dir, file1, file2, mode):
    return weak_sim_bisim.compute(base_dir, file1, file2, mode,
                                  output_dir="results_product_weak",
                                  structure=algebra.PRODUCT, max_iterations=500)

def truncated_product(base_dir, file1, file2, mode, threshold):
    return eps_weak_sim_bisim.compute(base_dir, file1, file2, mode, threshold)
```

Izvorni kod 9: services/download.py

```
import os
import zipfile

from app import config
from app.services import jobs

def _read_variant(directory):
    # Which page produced this job - scopes the download to its own folders.
    path = os.path.join(directory, jobs.VARIANT_FILE)
    if os.path.isfile(path):
        with open(path) as f:
            return f.read().strip()
    return None

def zip_job(job_id):
    # Zip this page's result files and the two inputs
    directory = jobs.job_dir(job_id)
    if not os.path.isdir(directory):
        return None

    sources = config.DOWNLOAD_SOURCES.get(_read_variant(directory), config.RESULT_SUBDIRS)

    zip_path = os.path.join(config.WORK_ROOT, job_id + "_results.zip")
    with zipfile.ZipFile(zip_path, "w", zipfile.ZIP_DEFLATED) as archive:
        for input_file in ("automatonA.txt", "automatonB.txt"):
            input_path = os.path.join(directory, input_file)
            if os.path.isfile(input_path):
                archive.write(input_path, input_file)

        filenames = []
        for src in sources:
            src_dir = os.path.join(directory, src)
            if os.path.isdir(src_dir):
                for fname in sorted(os.listdir(src_dir)):
                    if os.path.isfile(os.path.join(src_dir, fname)) and fname not in filenames:
                        filenames.append(fname)

        for fname in filenames:
            parts = []
            for src in sources:
                fpath = os.path.join(directory, src, fname)
```

```
                if os.path.isfile(fpath):
                    with open(fpath, "r", encoding="utf-8") as f:
                        parts.append(f.read())
            if parts:
                archive.writestr(fname, "".join(parts))
    return zip_path
```

Izvorni kod 10: services/words_search.py

```
def findByKey(arr, key):
    for item in arr:
        if item.get("word") == key:
            return item.get("vector")
    return False

def findByValue(arr, value):
    keys = []
    for item in arr:
        if item.get("vector").value == value:
            x = item.get("word")
            keys.append(x)
    if not keys:
        return False
    else:
        return keys

def compareArrays(arr1, arr2):
    for a1 in arr1:
        for a2 in arr2:
            if a1 == a2:
                return True
    return False
```

Izvorni kod 11: services/sim_bisim.py

```
import os

from fastapi import UploadFile
from automata.automaton import Automaton, getResponseAutomaton
from automata.matrix import Matrix
from automata.matrix import matrix_min, matrix_ordering
from maxplus.algebra_operations import left_residual, multiply, right_residual

MAX_ITERATIONS = 10

class Vectors:

    def __init__(self, automaton1, automaton2):
        self.automaton1 = automaton1
        self.automaton2 = automaton2
        self.sigma_A, self.tau_A = automaton1.sigma, automaton1.tau  # row vectors
        self.sigma_B, self.tau_B = automaton2.sigma, automaton2.tau  # row vectors
        self.sigma_A_T, self.tau_A_T = self.sigma_A.transpose(), self.tau_A.transpose()
        self.sigma_B_T, self.tau_B_T = self.sigma_B.transpose(), self.tau_B.transpose()


```

```
class Option:

    def __init__(self, key, suffix, filename, label, seed, step, holds):
        self.key = key
        self.suffix = suffix
        self.filename = filename
        self.label = label
        self.seed = seed
        self.step = step
        self.holds = holds

OPTIONS = [
    Option(
        "forwardSimulation", "fs", "fs.txt", "Forward similation",
        seed=lambda s, v: right_residual(v.tau_A, v.tau_B, s),
        step=lambda s, U, M_A, M_B: left_residual(multiply(M_B, U.transpose(), s), M_A, s).transpose
    (),
        holds=lambda s, v, U: matrix_ordering(v.sigma_A, multiply(v.sigma_B, U.transpose(), s)),
    ),
    Option(
        "backwardSimulation", "bs", "bs.txt", "Backward similation",
        seed=lambda s, v: right_residual(v.sigma_A, v.sigma_B, s),
        step=lambda s, U, M_A, M_B: right_residual(M_A, multiply(U, M_B, s), s),
        holds=lambda s, v, U: matrix_ordering(v.tau_A_T, multiply(U, v.tau_B_T, s)),
    ),
    Option(
        "forwardBisimulation", "fb", "fb.txt", "Forward bisimilation",
        seed=lambda s, v: matrix_min(right_residual(v.tau_A, v.tau_B, s),
            left_residual(v.tau_A_T, v.tau_B_T, s)),
        step=lambda s, U, M_A, M_B: matrix_min(
            left_residual(multiply(M_B, U.transpose(), s), M_A, s).transpose(),
            left_residual(multiply(M_A, U, s), M_B, s)),
        holds=lambda s, v, U:
            matrix_ordering(v.sigma_A, multiply(v.sigma_B, U.transpose(), s))
            and matrix_ordering(v.sigma_B, multiply(v.sigma_A, U, s)),
    ),
    Option(
        "backwardBisimulation", "bb", "bb.txt", "Backward bisimilation",
        seed=lambda s, v: matrix_min(right_residual(v.sigma_A, v.sigma_B, s),
                                     left_residual(v.sigma_A_T, v.sigma_B_T, s)),
        step=lambda s, U, M_A, M_B: matrix_min(
            right_residual(M_A, multiply(U, M_B, s), s),
            right_residual(M_B, multiply(U.transpose(), M_A, s), s).transpose()),
        holds=lambda s, v, U:
            matrix_ordering(v.tau_A_T, multiply(U, v.tau_B_T, s))
            and matrix_ordering(v.tau_B_T, multiply(U.transpose(), v.tau_A_T, s)),
    ),
    Option(
        "forwardBackwardBisimulation", "fbb", "fbb.txt", "Forward-backward bisimilation",
        # (sigma_A / sigma_B) i (tau_A \ tau_B)
        seed=lambda s, v: matrix_min(left_residual(v.sigma_A_T, v.sigma_B_T, s),
                                     right_residual(v.tau_A, v.tau_B, s)),
        step=lambda s, U, M_A, M_B: matrix_min(
            left_residual(multiply(M_B, U.transpose(), s), M_A, s).transpose(),
            right_residual(M_B, multiply(U.transpose(), M_A, s), s).transpose()),
        holds=lambda s, v, U:
            matrix_ordering(v.sigma_A, multiply(v.sigma_B, U.transpose(), s))
```

```
            and matrix_ordering(v.tau_B_T, multiply(U.transpose(), v.tau_A_T, s)),
    ),
    Option(
        "backwardForwardBisimulation", "bfb", "bfb.txt", "Backward-forward bisimilation",
        seed=lambda s, v: matrix_min(right_residual(v.sigma_A, v.sigma_B, s),
                                     left_residual(v.tau_A_T, v.tau_B_T, s)),
        step=lambda s, U, M_A, M_B: matrix_min(
            right_residual(M_A, multiply(U, M_B, s), s),
            left_residual(multiply(M_A, U, s), M_B, s)),
        holds=lambda s, v, U:
            matrix_ordering(v.tau_A_T, multiply(U, v.tau_B_T, s))
            and matrix_ordering(v.sigma_B, multiply(v.sigma_A, U, s)),
    ),
]

_BY_KEY = {option.key: option for option in OPTIONS}

# Section titles written at the top of every result file.
_TITLES = [('fs.txt', 'Forward Simulation'), ('bs.txt', 'Backward Simulation'),
           ('fb.txt', 'Forward Bisimulation'), ('bb.txt', 'Backward Bisimulation'),
           ('fbb.txt', 'Forward-Backward Bisimulation'),
           ('bfb.txt', 'Backward-Forward Bisimulation')]

def _solve(option, structure, output_dir, messages, vectors, zero_matrix, base_dir, result):
    responses = {}
    file_path = os.path.join(base_dir, output_dir, option.filename)

    def record(matrix):
        matrix.print_matrix_in_file(file_path)
        r = matrix.get_matrix()
        responses[r['name']] = r

    U_1 = option.seed(structure, vectors)
    U_1.name = "U_1_" + option.suffix
    record(U_1)

    n = 1
    while True:
        n += 1
        name = "U_" + str(n) + "_" + option.suffix

        # infimum by word length
        local_min = Matrix(name, [[structure.top] * U_1.cols for _ in range(U_1.rows)])
        for i in range(len(vectors.automaton1.transitions)):
            M_x_A = vectors.automaton1.transitions[i]  # transition matrix for automaton A
            M_x_B = vectors.automaton2.transitions[i]  # transition matrix for automaton B
            local_min = matrix_min(local_min, option.step(structure, U_1, M_x_A, M_x_B))

        U_2 = matrix_min(local_min, U_1) # infimum by previous solution

        if n == MAX_ITERATIONS:
            result[option.key + 'Result'] = messages[option.key]
            break
        elif U_1.value == U_2.value:
            # Chech 1-xi condition
            holds = (U_2.value != zero_matrix.value and option.holds(structure, vectors, U_2))
            result[option.key + 'Result'] = option.label + (" exists" if holds else " does not exist"
```

```
    )
            break
        else:
            # Continue further search for the greatest solution
            U_2.name = name
            U_1 = U_2
            record(U_1)

    result[option.key] = responses

def compute(base_dir, file1: UploadFile, file2: UploadFile, selected_options, structure, output_dir,
     messages):
    result = {}
    automaton1 = Automaton(file1)
    automaton2 = Automaton(file2)

    result['automatonA'] = getResponseAutomaton(automaton1)
    result['automatonB'] = getResponseAutomaton(automaton2)

    # Insert titles in files
    for filename, title in _TITLES:
        file_path = os.path.join(base_dir, output_dir, filename)
        with open(file_path, 'w') as file:
            file.write("----- " + title + " -----\n")

    zero_matrix = Matrix("zero", [[structure.zero] * automaton2.n for _ in range(automaton1.n)])
    vectors = Vectors(automaton1, automaton2)

    for checkbox_text in selected_options:
        _solve(_BY_KEY[checkbox_text], structure, output_dir, messages,
               vectors, zero_matrix, base_dir, result)

    return result
```

Izvorni kod 12: services/weak_sim_bisim.py

```
import os

from automata.matrix import matrix_min, matrix_ordering
from maxplus import algebra_operations
from maxplus import algebra
from app.services.words_search import compareArrays, findByKey, findByValue
from automata.automaton import Automaton, getResponseAutomaton
from automata.matrix import Matrix

MAX_ITERATIONS = 100

_BACKWARD = {"weakBackwardSimulation", "weakBackwardBisimulation"}
_BISIMULATION = {"bisimulation", "weakBackwardBisimulation"}

_DEFAULT_STRUCTURE = algebra.SEMIRING


def _bisim(v1, v2, structure):
    """Residual matrix U[i][j] = v1_i \\ v2_j (the residual matrix behind one word)."""
    value = [
        [structure.residual(v1.value[0][i], v2.value[0][j]) for j in range(v2.cols)]
```

```
        for i in range(v1.cols)
    ]
    return Matrix(v1.name + " <-> " + v2.name, value)


def _sim_cond_sigma(sigma1, sigma2, phi, structure):
    return matrix_ordering(algebra_operations.multiply(sigma2, phi.transpose(), structure), sigma1)


def _bisim_cond_sigma(sigma1, sigma2, phi, structure):
    return matrix_ordering(sigma2, algebra_operations.multiply(sigma1, phi, structure))


def _conclusion(a1, a2, max_bisim, backward, bisimulation, structure):
    if backward:
        tau_a = a1.tau.transpose()
        tau_b = a2.tau.transpose()
        holds = matrix_ordering(tau_a, algebra_operations.multiply(max_bisim, tau_b, structure))
        if bisimulation:
            holds = holds and matrix_ordering(
                tau_b, algebra_operations.multiply(max_bisim.transpose(), tau_a, structure)
            )
        label = "Weak Backward Bisimulation" if bisimulation else "Weak Backward Simulation"
        return label + (" Exists" if holds else " Does Not Exist")

    # Forward: "does not exist" when the condition holds.
    fails = _sim_cond_sigma(a1.sigma, a2.sigma, max_bisim, structure)
    if bisimulation:
        fails = fails and _bisim_cond_sigma(a1.sigma, a2.sigma, max_bisim, structure)
    label = "Weak Forward Bisimulation" if bisimulation else "Weak Forward Simulation"
    return label + (" Does Not Exist" if fails else " Exists")


def compute(base_dir, file1, file2, mode, *, max_length=None,
            output_dir="results_max_plus_weak", structure=None,
            max_iterations=MAX_ITERATIONS):
    structure = structure or _DEFAULT_STRUCTURE
    backward = mode in _BACKWARD
    bisimulation = mode in _BISIMULATION

    automaton1 = Automaton(file1)
    automaton2 = Automaton(file2)

    result = {
        "automatonA": getResponseAutomaton(automaton1),
        "automatonB": getResponseAutomaton(automaton2),
    }

    out_dir = os.path.join(base_dir, output_dir)
    all_calculations = os.path.join(out_dir, "all_calculations.txt")
    max_calculation = os.path.join(out_dir, "max_calculation.txt")

    if backward:
        seed1, seed2 = automaton1.sigma, automaton2.sigma
        symbol, prefix, seed_word = "sigma", "s_", "sigma_"
    else:
        seed1, seed2 = automaton1.tau, automaton2.tau
```

```
        symbol, prefix, seed_word = "tau", "t_", "tau_"

    responses = {}
    vectors = {}
    words = [seed_word]
    vecs1 = [{"word": seed_word, "vector": seed1}]
    vecs2 = [{"word": seed_word, "vector": seed2}]

    start_bisim = _bisim(seed1, seed2, structure)
    start_bisim.name = seed_word + "  <-> " + seed_word + "  "
    start_bisim.print_matrix_in_file(all_calculations)
    responses[start_bisim.name] = start_bisim.get_matrix()
    vectors[start_bisim.name] = {"A": seed1.get_matrix(), "B": seed2.get_matrix()}

    max_bisim = _bisim(seed1, seed2, structure)
    # matrix_min returns a fresh matrix, so the fold loses this name; it is put
    # back once the loop is done, before the solution is written out.
    solution_name = max_bisim.name
    break_flag = False
    length_capped = False  # True if the word-length cap actually stopped expansion
    count = 1

    for w in words:
        # Stop expanding once the word reaches the requested length.
        if max_length is not None and len(w) - 2 >= max_length:
            length_capped = True
            continue

        v1 = findByKey(vecs1, w)
        v2 = findByKey(vecs2, w)

        for d1, d2 in zip(automaton1.transitions, automaton2.transitions):
            if backward:
                new_w = prefix + w[2:] + d1.name          # sigma \otimes M_u
                p1 = algebra_operations.multiply(v1, d1, structure)
                p2 = algebra_operations.multiply(v2, d2, structure)
            else:
                new_w = prefix + d1.name + w[2:]           # M_u \otimes tau
                p1 = algebra_operations.multiply(d1, v1.transpose(), structure).transpose()
                p2 = algebra_operations.multiply(d2, v2.transpose(), structure).transpose()
            p1.name = new_w
            p2.name = new_w

            search1 = findByValue(vecs1, p1.value)
            search2 = findByValue(vecs2, p2.value)

            if not (search1 and search2 and compareArrays(search1, search2)):
                vecs1.append({"word": new_w, "vector": p1})
                vecs2.append({"word": new_w, "vector": p2})
                words.append(new_w)

                b = _bisim(p1, p2, structure)
                b.print_matrix_in_file(all_calculations)
                responses[b.name] = b.get_matrix()
                vectors[b.name] = {"A": p1.get_matrix(), "B": p2.get_matrix()}

                max_bisim = matrix_min(max_bisim, b)
```

```
                count += 1

                if count == max_iterations:
                    break_flag = True
                    result["weakSimulationResult"] = (
                        "Procedure did not finish in %d iterations" % max_iterations
                    )

            if break_flag:
                break
        if break_flag:
            break

    if "weakSimulationResult" not in result:
        message = _conclusion(
            automaton1, automaton2, max_bisim, backward, bisimulation, structure
        )
        # If it only "Exists" up to the requested word length, say
        if length_capped and message.endswith("Exists"):
            message += " for Word Length " + str(max_length)
        result["weakSimulationResult"] = message

    max_bisim.name = solution_name
    max_bisim.print_matrix_in_file(max_calculation)

    result["weakSimulation"] = responses
    result["weakBisimulation"] = responses
    result["weakVectors"] = vectors
    result["max_solution"] = {max_bisim.name: max_bisim.get_matrix()}
    result["different_tau_vectors"] = count
    result["symbol"] = symbol
    return result
```

Izvorni kod 13: services/eps_weak_sim_bisim.py

```
import os

from automata.automaton import Automaton, getResponseAutomaton
from automata.matrix import Matrix
from automata.matrix import matrix_min, matrix_ordering
from app.services.words_search import compareArrays, findByKey, findByValue
from truncated_product import algebra
from truncated_product.algebra_operations import multiply

# Upper bound on the procedure: the set of tau vector pairs is not guaranteed
# to stabilise, so stop after this many newly discovered pairs.
ITERATION_LIMIT = 500

RESULTS_DIR = 'results_truncated_product'
ALL_CALCULATIONS = 'all_calculations.txt'
MAX_CALCULATION = 'max_calculation.txt'

LABELS = {
    "simulation": "epsilon-weak Simulation",
    "bisimulation": "epsilon-weak Bisimulation",
}

```

```
23
24 class InvalidThresholdError(Exception):
25     pass
26
27
28 def _start_files(base_dir, threshold, simulation):
29     # Open both result files with this run's heading; return their paths.
30     label = "Simulation" if simulation else "Bisimulation"
31     heading = "\n\nEpsilon = " + str(threshold) + "\n\n"
32     paths = {}
33     for name, title in ((ALL_CALCULATIONS, label + "s"), (MAX_CALCULATION, "Max " + label)):
34         paths[name] = os.path.join(base_dir, RESULTS_DIR, name)
35         with open(paths[name], 'w') as file:
36             file.write("----- " + title + " -----" + heading)
37     return paths
38
39
40 def _sim(v1, v2, structure):
41     # How similar the two vectors are, coordinate by coordinate.
42     value = [
43         [structure.sim(v1.value[0][i], v2.value[0][j]) for j in range(v2.cols)]
44         for i in range(v1.cols)
45     ]
46     return Matrix(v1.name + ' <-> ' + v2.name, value)
47
48
49 def _sim_cond_sigma(sigma1, sigma2, phi, structure):
50     return matrix_ordering(sigma1, multiply(sigma2, phi.transpose(), structure))
51
52
53 def _sim_cond_tau(tau1, tau2, phi, structure):
54     return matrix_ordering(multiply(phi.transpose(), tau1.transpose(), structure),
55                            tau2.transpose())
56
57
58 def _bisim_cond_sigma(sigma1, sigma2, phi, structure):
59     return matrix_ordering(sigma2, multiply(sigma1, phi, structure))
60
61
62 def _bisim_cond_tau(tau1, tau2, phi, structure):
63     return matrix_ordering(multiply(phi, tau2.transpose(), structure), tau1.transpose())
64
65
66 def _relation_holds(simulation, automaton1, automaton2, t1, t2, phi, structure):
67     # Does the sigma/tau condition still hold for the current solution?
68     if simulation:
69         return (_sim_cond_sigma(automaton1.sigma, automaton2.sigma, phi, structure)
70                 and _sim_cond_tau(t1, t2, phi, structure))
71     return (_bisim_cond_sigma(automaton1.sigma, automaton2.sigma, phi, structure)
72             and _bisim_cond_tau(t1, t2, phi, structure))
73
74
75 def compute(base_dir, file1, file2, mode, threshold):
76     if(threshold > 1 or threshold <= 0 or isinstance(threshold, float) == False):
77         raise InvalidThresholdError("Invalid threshold Error")
78
79     simulation = mode != "bisimulation"
```

```
    structure = algebra.truncated(float(threshold))

    automaton1 = Automaton(file1, threshold)
    automaton2 = Automaton(file2, threshold)
    paths = _start_files(base_dir, threshold, simulation)

    result = {
        'automatonA': getResponseAutomaton(automaton1),
        'automatonB': getResponseAutomaton(automaton2),
    }
    responses = {}

    words = ['tau_']                                          # words array
    vectors_A = [{'word': 'tau_', 'vector': automaton1.tau}]    # word -> vector, automaton A
    vectors_B = [{'word': 'tau_', 'vector': automaton2.tau}]    # word -> vector, automaton B

    # The greatest solution so far; matrix_min folds each new pair into it.
    max_bisim = _sim(automaton1.tau, automaton2.tau, structure)
    max_bisim.name = "tau_  <-> tau_  "
    solution_name = max_bisim.name
    max_bisim.print_matrix_in_file(paths[ALL_CALCULATIONS], threshold)
    r = max_bisim.get_matrix()
    responses[r['name']] = r

    holds = True
    stopped_after = None
    break_flag = False
    iterations = 0

    for w in words:
        t1 = findByKey(vectors_A, w)  # vector in automaton A for selected word w
        t2 = findByKey(vectors_B, w)  # vector in automaton B for selected word w

        for d1, d2 in zip(automaton1.transitions, automaton2.transitions):
            new_w = "t_" + d1.name + w[2:]
            t1_prod = multiply(d1, t1.transpose(), structure).transpose()
            t1_prod.name = new_w
            t2_prod = multiply(d2, t2.transpose(), structure).transpose()
            t2_prod.name = new_w

            search_A = findByValue(vectors_A, t1_prod.value)
            search_B = findByValue(vectors_B, t2_prod.value)

            # Already seen this pair of vectors? Then this word adds nothing.
            if (not (search_A and search_B and compareArrays(search_A, search_B))):
                vectors_A.append({'word': new_w, "vector": t1_prod})
                vectors_B.append({'word': new_w, "vector": t2_prod})
                words.append(new_w)
                iterations = iterations + 1

                step = _sim(t1_prod, t2_prod, structure)
                step.print_matrix_in_file(paths[ALL_CALCULATIONS], threshold)
                r = step.get_matrix()
                responses[r['name']] = r

                max_bisim = matrix_min(max_bisim, step)

```

```
                if not _relation_holds(simulation, automaton1, automaton2,
                                       t1_prod, t2_prod, max_bisim, structure):
                    holds = False
                    break_flag = True

                if not break_flag and iterations >= ITERATION_LIMIT:
                    stopped_after = ITERATION_LIMIT
                    break_flag = True

            if break_flag:
                break

        if break_flag:
            break

    with open(paths[ALL_CALCULATIONS], 'a') as file:
        file.write("Number of tau vectors: " + str(len(words)) + '\n')
    max_bisim.name = solution_name
    max_bisim.print_matrix_in_file(paths[MAX_CALCULATION], threshold)

    if stopped_after is not None:
        # Hit the iteration cap, so neither answer has been established.
        message = "Procedure stopped after %d iterations" % stopped_after
    else:
        label = LABELS["simulation" if simulation else "bisimulation"]
        message = label + (" Exists" if holds else " Does Not Exist")

    result['weakSimulation'] = responses
    result['weakBisimulation'] = responses
    result['weakSimulationResult'] = message
    result['weakVectors'] = _tau_vectors(file1, file2, responses, structure, threshold)
    result['max_solution'] = {max_bisim.name: max_bisim.get_matrix()}
    result['different_tau_vectors'] = len(words)
    result['symbol'] = "tau"
    return result


def _suffix_from_name(name):
    # The word behind one result name, e.g. "t_xy <-> t_xy" -> "xy".
    token = name.split("<->")[0].strip()
    idx = token.find("_")
    return token[idx + 1:].strip() if idx >= 0 else ""


def _tau_for_suffix(automaton, suffix, structure):
    # Fold the word's transitions over tau, right to left.
    transitions = {t.name: t for t in automaton.transitions}
    vector = automaton.tau
    for letter in reversed(suffix):
        vector = multiply(transitions[letter], vector.transpose(), structure).transpose()
    return vector


def _tau_vectors(path_a, path_b, responses, structure, threshold):
    # The two tau vectors behind every result, for display alongside it.
    automaton_a = Automaton(path_a, threshold)
    automaton_b = Automaton(path_b, threshold)
```

```
    vectors = {}
    for name in responses:
        suffix = _suffix_from_name(name)
        try:
            vec_a = _tau_for_suffix(automaton_a, suffix, structure)
            vec_b = _tau_for_suffix(automaton_b, suffix, structure)
        except (KeyError, ValueError):
            continue  # unknown letter / shape mismatch: just skip the vectors
        vectors[name] = {"A": vec_a.get_matrix(), "B": vec_b.get_matrix()}
    return vectors
```

Dodatak B

Biografija i bibliografija autora

Jelena Matejić rođena je 18. novembra 1994. godine. Osnovnu školu „Njegoš" u Nišu završila je 2009. godine. Prirodno-matematički smer Gimnazije „Svetozar Marković" u Nišu završila je 2013. godine.

Svoje akademsko obrazovanje stekla je na Prirodno-matematičkom fakultetu Univerziteta u Nišu, na Departmanu za računarske nauke. Osnovne akademske studije upisala je 2013. godine, a završila 2016. godine. Iste godine upisala je master akademske studije, modul Upravljanje informacijama, koje je završila 2018. godine odbranom master rada pod nazivom „Razvoj aplikacija za elektronsko učenje korišćenjem Laravel radnog okruženja". Nakon toga, 2018. godine upisala je doktorske akademske studije.

Od aprila 2019. godine zaposlena je na Departmanu za računarske nauke Prirodno-matematičkog fakulteta Univerziteta u Nišu, gde je najpre bila angažovana u zvanju istraživača-pripravnika, a od septembra 2021. godine u zvanju asistenta. U nastavnom radu bila je angažovana na izvođenju vežbi iz više predmeta u oblasti računarstva. Pored angažovanja na fakultetu, radno iskustvo sticala je i u školama programiranja namenjenim deci i odraslima.

Učestvovala je u realizaciji više međunarodnih i nacionalnih naučnoistraživačkih i obrazovnih projekata. Bila je član koordinatorskog tima Erasmus+ projekta *Strengthening Teaching Competences in Higher Education in Natural and Mathematical Sciences* (2019–2022), član tima projekta Fonda za nauku Republike Srbije *Quantitative Automata Models: Fundamental Problems and Applications* (2022–2024), član partnerskog tima Erasmus+ projekta *Master Degree in Integrating Innovative STEM Strategies in Higher Education* (2024–2026), kao i koordinatorskog tima projekta Centra za promociju nauke *Hab mladih istraživača* (2026–).

Kao autor i koautor objavila je više naučnih radova u međunarodnim i domaćim časopisima. Rezultate svojih istraživanja prezentovala je na više međunarodnih naučnih konferencija.

Pored naučnoistraživačkog i nastavnog rada, aktivno učestvuje u brojnim projektima i aktivnostima usmerenim ka promociji nauke i obrazovanja. Za postignute rezultate u nastavnom radu dobitnica je Nagrade za nastavnu delatnost Prirodno-matematičkog fakulteta Univerziteta u Nišu za 2024. godinu.

Bibliografija autora

Radovi objavljeni u međunarodnim i domaćim časopisima

- J. Matejić, I. Micić, M. Ćirić, *Weak simulations and bisimulations for max-plus automata*, Filomat, prihvaćen za objavljivanje, 2026.
- J. Matejić, M. Ćirić, J. Ignjatović, I. Micić, *Truncating and Shifting Weights for Max-Plus Automata*, Axioms, vol. 15, 2026 (ar. 79).
- M. Ćirić, I. Micić, J. Matejić, S. Stanimirović, *Simulations and bisimulations for max-plus automata*, Discrete Event Dynamic Systems, vol. 34, 2024 (pp. 269–295).
- I. Micić, M. Ćirić, J. Matejić, S. Stanimirović, L. A. Nguyen, *Approximate weak simulations and bisimulations for fuzzy automata over the product structure*, Fuzzy Sets and Systems, vol. 485, 2024 (ar. 108959).
- J. Matejić, A. Milenković, *Impact of Peer Feedback in a Web Programming Course on Students' Achievement*, International Journal of Cognitive Research in Science, Engineering and Education, vol. 13, no. 1, 2025 (pp. 33–49).
- J. Matejić, *Students' perceptions of peer assessment in a Web programming course: Reflections on feedback, learning, and professional improvement*, Journal of Educational Studies in Mathematics and Computer Science, vol. 2, no. 1, 2025 (pp. 56–66).
- J. Matejić, J. Ignjatović, *Kvizovi novog doba: pristupi u testiranju znanja*, Nastava matematike, vol. LXX, no. 1–2, 2025 (pp. 62–68).

Radovi objavljeni u zbornicima međunarodnih i domaćih naučnih skupova

- I. Micić, J. Matejić, S. Stanimirović, L. A. Nguyen, *Towards New Types of Weak Bisimulations for Fuzzy Automata Using the Product T-Norm*, Fuzzy Logic and Technology, and Aggregation Operators, 2023 (pp. 567–578).
- A. Milenković, J. Matejić, *Vršnjačko vrednovanje radova studenata matematike na temu digitalnih nastavnih materijala*, Beograd: Pedagoško društvo Srbije, 2023 (pp. 167–172).

Saopštenja na međunarodnim i domaćim naučnim skupovima objavljena u izvodu

- J. Matejić, I. Micić, M. Ćirić, S. Stanimirović, *Approximating Language Equivalence: Computation of ε-Weak Bisimulations for Fuzzy Automata*, The 14th International Conference on Social and Technological Development (STED 2025), Trebinje, Bosnia and Herzegovina, June 12–15, 2025.

- J. Matejić, I. Micić, M. Ćirić, S. Stanimirović, L. A. Nguyen, *Efficient Computation of Epsilon-Weak Bisimulations for Fuzzy Automata*, International Conference on Discrete Applied Mathematics and Mathematical Programming (ICDAMMP 2024), Lisbon, Portugal, August 27–28, 2024

- J. Matejić, I. Micić, M. Ćirić, S. Stanimirović, L. A. Nguyen, *Algorithmic Insights into Computing ε-Weak Bisimulations for Fuzzy Automata*, 11th International Workshop on Weighted Automata: Theory and Applications (WATA 2023), Leipzig, Germany, October 4–7, 2023.

- J. Matejić, I. Micić, M. Ćirić, S. Stanimirović, *Computing ε-weak bisimulations for fuzzy automata over truncated product structures: algorithms and complexity analysis*, International Mathematical Conference Analysis, Approximations, and Applications (AAA 2023), Faculty of Science, University of Kragujevac, Vrnjačka Banja, Serbia, June 21–24, 2023.

- J. Matejić, J. Ignjatović, *Vršnjačko ocenjivanje u visokom obrazovanju: percepcije studenata i uticaj na učenje*, Druga konferencija o nastavi matematike i informatike (TEMATCOM 2026), Kragujevac, Serbia, May 29–30, 2026.

ИЗЈАВА О АУТОРСТВУ

Изјављујем да је докторска дисертација, под насловом

АПРОКСИМАТИВНЕ БИСИМУЛАЦИЈЕ ЗА ТЕЖИНСКЕ АУТОМАТЕ

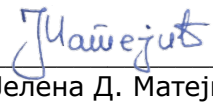
која је одбрањена на Природно-математичком факултету Универзитета у Нишу:

- резултат сопственог истраживачког рада;
- да ову дисертацију, ни у целини, нити у деловима, нисам пријављивала на другим факултетима, нити универзитетима;
- да нисам повредила ауторска права, нити злоупотребила интелектуалну својину других лица.

Дозвољавам да се објаве моји лични подаци, који су у вези са ауторством и добијањем академског звања доктора наука, као што су име и презиме, година и место рођења и датум одбране рада, и то у каталогу Библиотеке, Дигиталном репозиторијуму Универзитета у Нишу, као и у публикацијама Универзитета у Нишу.

У Нишу, 13. 10. 2026. године.

Потпис аутора дисертације:



Јелена Д. Матејић

ИЗЈАВА О ИСТОВЕТНОСТИ ЕЛЕКТРОНСКОГ И ШТАМПАНОГ ОБЛИКА ДОКТОРСКЕ ДИСЕРТАЦИЈЕ

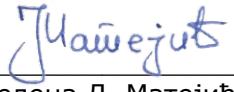
Наслов дисертације:

АПРОКСИМАТИВНЕ БИСИМУЛАЦИЈЕ ЗА ТЕЖИНСКЕ АУТОМАТЕ

Изјављујем да је електронски облик моје докторске дисертације, коју сам предала за уношење у **Дигитални репозиторијум Универзитета у Нишу,** истоветан штампаном облику.

У Нишу, 13. 10. 2026. године.

Потпис аутора дисертације:



Јелена Д. Матејић

ИЗЈАВА О КОРИШЋЕЊУ

Овлашћујем Универзитетску библиотеку „Никола Тесла" да у Дигитални репозиторијум Универзитета у Нишу унесе моју докторску дисертацију, под насловом

АПРОКСИМАТИВНЕ БИСИМУЛАЦИЈЕ ЗА ТЕЖИНСКЕ АУТОМАТЕ

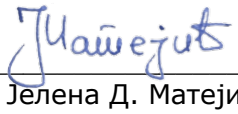
Дисертацију са свим прилозима предала сам у електронском облику, погодном за трајно архивирање.

Моју докторску дисертацију, унету у Дигитални репозиторијум Универзитета у Нишу, могу користити сви који поштују одредбе садржане у одабраном типу лиценце Креативне заједнице (Creative Commons), за коју сам се одлучила.

1. Ауторство **(CC BY)**
2. Ауторство – некомерцијално **(CC BY-NC)**
3. Ауторство – некомерцијално – без прераде **(CC BY-NC-ND)**
4. Ауторство – некомерцијално – делити под истим условима **(CC BY-NC-SA)**
5. Ауторство – без прераде **(CC BY-ND)**
6. Ауторство – делити под истим условима **(CC BY-SA)**

У Нишу, 13. 10. 2026. године.

Потпис аутора дисертације:



Јелена Д. Матејић

ПРИРОДНО - МАТЕМАТИЧКИ ФАКУЛТЕТ
НИШ

КЉУЧНА ДОКУМЕНТАЦИЈСКА ИНФОРМАЦИЈА

Редни број, **РБР**:	
Идентификациони број, **ИБР**:	
Тип документације, **ТД**:	монографска
Тип записа, **ТЗ**:	текстуални / графички
Врста рада, **ВР**:	докторска дисертација
Аутор, **АУ**:	Јелена Матејић
Ментор, **МН**:	Ивана Мицић
Наслов рада, **НР**:	Апроксимативне бисимулације за тежинске аутомате
Језик публикације, **ЈП**:	српски
Језик извода, **ЈИ**:	српски
Земља публиковања, **ЗП**:	Србија
Уже географско подручје, **УГП**:	Србија
Година, **ГО**:	2026.
Издавач, **ИЗ**:	ауторски репринт
Место и адреса, **МА**:	Ниш, Вишеградска 33.
Физички опис рада, **ФО**: (поглавља/страна/ цитата/табела/слика/графика/прилога)	7 поглавља / 205 страна / 88 цитата / 9 табела / 20 слика / 0 графика / 2 прилога
Научна област, **НО**:	рачунарске науке
Научна дисциплина, **НД**:	теоријско рачунарство
Предметна одредница/Кључне речи, **ПО**:	тежински аутомати, макс-плус аутомати, фази аутомати, симулације, бисимулације, слабе симулације, слабе бисимулације, апроксимативне симулације
УДК	004.421:004.8
Чува се, **ЧУ**:	библиотека
Важна напомена, **ВН**:	

Извод, **ИЗ**:		Ова дисертација проучава симулације и бисимулације за тежинске аутомате, са посебним фокусом на max-plus аутомате. Основни циљ рада је развој и анализа метода за поређење понашања аутомата у ситуацијама у којима стандардни поступци за одређивање симулација и бисимулација нису довољно ефикасни или не доводе до коначног решења. Разматрају се различити типови симулација и бисимулација, њихове слабе варијанте, као и карактеризације највећих решења помоћу система матричних неједначина. Посебна пажња посвећена је проблему бесконачности који се јавља при израчунавању слабих симулација и бисимулација. У том циљу уведене су релаксације засноване на ограничењу дужине речи и датума реализације, као и приступи засновани на доњем и горњем одсецању и померању тежина у max-plus структурама. На тај начин омогућено је контролисано приближно поређење понашања аутомата и ефикасније израчунавање одговарајућих релација. У завршном делу рада развијени приступ је проширен на фази аутомате над производ структуром, увођењем одсечене производ структуре и њој одговарајућих слабих симулација и бисимулација. Добијени резултати представљају допринос развоју метода за ефективно и апроксимативно поређење тежинских аутомата.
Датум прихватања теме, **ДП**:		08. 12. 2025.
Датум одбране, **ДО**:		
Чланови комисије, **КО**:	Председник:	
	Члан:	
	Члан:	
	Члан:	
	Члан, ментор:	

ПРИРОДНО - МАТЕМАТИЧКИ ФАКУЛТЕТ
НИШ

KEY WORDS DOCUMENTATION

Accession number, **ANO**:	
Identification number, **INO**:	
Document type, **DT**:	monograph
Type of record, **TR**:	textual / graphic
Contents code, **CC**:	doctoral dissertation
Author, **AU**:	Jelena Matejić
Mentor, **MN**:	Ivana Micić
Title, **TI**:	Approximate bisimulations for weighted automata
Language of text, **LT**:	Serbian
Language of abstract, **LA**:	Serbian
Country of publication, **CP**:	Serbia
Locality of publication, **LP**:	Serbia
Publication year, **PY**:	2026
Publisher, **PB**:	author's reprint
Publication place, **PP**:	Niš, Višegradska 33.
Physical description, **PD**: (chapters/pages/ref./tables/pictures/graphs/appendixes)	7 chapters / 205 pages / 88 ref / 9 tables / 20 pictures / 0 graphs / 2 appendixes
Scientific field, **SF**:	Computer science
Scientific discipline, **SD**:	Theoretical computer science
Subject/Key words, **S/KW**:	weighted automata, max-plus automata, fuzzy automata, simulations, bisimulations, weak simulations, weak bisimulations, approximate simulations, approximate bisimulations
UC	004.421:004.8
Holding data, **HD**:	library
Note, **N**:	

Abstract, **AB**:		This dissertation studies simulations and bisimulations for weighted automata, with particular emphasis on max-plus automata. The main objective is to develop and analyze methods for comparing the behavior of automata in situations where standard procedures for determining simulations and bisimulations are not sufficiently efficient or do not yield a finite solution. Different types of simulations and bisimulations, their weak variants, as well as characterizations of their greatest solutions by systems of matrix inequalities are considered. Particular attention is devoted to the problem of non-termination that arises in the computation of weak simulations and bisimulations. To address this issue, relaxations based on limiting the length of words and their execution dates are introduced, as well as approaches based on lower and upper truncation and shifting of weights in max-plus structures. These approaches enable controlled approximate comparison of automata behavior and more efficient computation of the corresponding relations. In the final part of the dissertation, the developed approach is extended to fuzzy automata over the product structure by introducing a truncated product structure and the corresponding weak simulations and bisimulations. The obtained results contribute to the development of methods for effective and approximate comparison of weighted automata.
Accepted by the Scientific Board on, **ASB**:		08. 12. 2025.
Defended on, **DE**:		
Defended Board, **DB**:	President:	
	Member:	
	Member:	
	Member:	
	Member, Mentor:	