

УНИВЕРЗИТЕТ У БЕОГРАДУ
МАТЕМАТИЧКИ ФАКУЛТЕТ

Весна Маринковић

АУТОМАТСКО РЕШАВАЊЕ
КОНСТРУКТИВНИХ ПРОБЛЕМА
У ГЕОМЕТРИЈИ

докторска дисертација

Београд, 2015.

UNIVERSITY OF BELGRADE
FACULTY OF MATHEMATICS

Vesna Marinković

AUTOMATED SOLVING
OF CONSTRUCTION PROBLEMS
IN GEOMETRY

Doctoral Dissertation

Belgrade, 2015.

Ментор:

др Предраг ЈАНИЧИЋ, редовни професор
Универзитет у Београду, Математички факултет

Чланови комисије:

др Зоран ЛУЧИЋ, ванредни професор
Универзитет у Београду, Математички факултет

др Филип МАРИЋ, доцент
Универзитет у Београду, Математички факултет

др Pascal SCHRECK, професор
Универзитет у Стразбуру

Датум одбране: _____

Наслов дисертације: Аутоматско решавање конструктивних проблема у геометрији

Резиме: Проблеми геометријских конструкција уз помоћ лењира и шестара представљају један од најстаријих и најизазовнијих проблема у елементарној математици. Под решењем конструктивног проблема не подразумева се слика, већ процедура којом се, на основу задатих конструкцијских примитива, у низу корака даје “упутство” како треба конструисати тражени објекат. Главни проблем при решавању конструктивних проблема, и за човека и за рачунар, представља комбинаторна експлозија која се јавља у њима, као и процес доказивања исправности решења.

У овој дисертацији приказана је метода за аутоматско решавање једне класе конструктивних проблема, тзв. локацијских проблема, у којима је задатак конструисати троугао уколико су познате локације три значајне тачке троугла. Овом методом се успешно решавају скоро сви решиви проблеми из Верникове и Конелијеве листе. За сваки од проблема утврђује се да ли је симетричан са неким већ решаваним проблемом, а затим се утврђује његов статус: да ли је проблем редудантан, зависан од положаја, решив или га није могуће решити коришћењем постојећег знања. Такође, аутоматски се генерише опис конструкције на природном језику и на језику GCLC, заједно са пратећом илустрацијом и доказом исправности генерисане конструкције, уз списак услова под којим је конструкција исправна. Предложени метод имплементиран је у оквиру алата ArgoTriCS. За доказивање исправности аутоматски генерисаних конструкција, систем ArgoTriCS користи развијени доказивач ArgoCLP, аутоматске доказиваче теорема интегрисане у алате GCLC и OpenGeoProver, као и интерактивни доказивач теорема Isabelle. Демонстрирано је да добијени докази исправности могу бити машински провериви.

У оквиру ове тезе приказан је и систем ArgoCLP (развијен у сарадњи са Саном Стојановић) којим се аутоматски доказују теореме у кохерентној логици. Овај доказивач успешно је примењен на различите аксиоматске системе. Њиме се аутоматски генеришу читљиви докази на природном језику, као и машински провериви докази, чију је исправност могуће проверити коришћењем интерактивног доказивача теорема Isabelle. Важан део овог система чини модул за поједностављивање генерисаних доказа, којим се добијају краћи и читљивији докази.

Кључне речи: аутоматско резоновање, аутоматско доказивање теорема, ин-

теракивно доказивање теорема, аутоматско генерисање читљивих доказа, ко-
херентна логика, конструктивни проблеми у геометрији, конструкције лењиром
и шестаром

Научна област: рачунарство

Ужа научна област: вештачка интелигенција

УДК број: 004.832.3:514.181.8(043.3)

Dissertation title: Automated Solving of Construction Problems in Geometry

Abstract: The problems of geometry constructions using ruler and compass are one of the oldest and most challenging problems in elementary mathematics. A solution of construction problem is not an illustration, but a procedure that, using given construction primitives, gives a “recipe” how to construct the object sought. The main problem in solving construction problems, both for a human and a computer, is a combinatorial explosion that occurs along the solving process, as well as a procedure of proving solutions correct.

In this dissertation a method for automated solving of one class of construction problems, so-called location problems, is proposed. These are the problems in which the task is to construct a triangle if locations of three characteristic points are given. This method successfully proved most of the solvable problems from Wernick’s and Connelly’s list. For each of the problems it is checked if it is symmetric to some of the already solved problems, and then its status is determined: the problem can be found redundant, locus dependent, solvable, or not solvable using existing knowledge. Also, a description of the construction in natural-language form and in language GCLC is automatically generated, accompanied by a corresponding illustration, and correctness proof of the generated construction, followed by the list of conditions when the construction is correct. The proposed method is implemented within the tool ArgoTriCS. For proving generated constructions correct, the system ArgoTriCS uses a newly developed prover ArgoCLP, the automated theorem provers integrated within tools GCLC and OpenGeoProved, as well as the interactive theorem prover Isabelle. It is demonstrated that the proofs obtained can be machine verifiable.

Within this dissertation, the system ArgoCLP (developed in collaboration with Sana Stojanović) which is capable of automatically proving theorems in coherent logic is described. This prover is successfully applied to different axiomatic systems. It automatically generates proofs in natural-language form, as well as machine-verifiable proofs, whose correctness can be checked using interactive theorem prover Isabelle. The important part of this system is a module for simplification of generated proofs whereby shorter and readable proofs are obtained.

Keywords: automated reasoning, automated theorem proving, interactive theorem proving, automated generation of readable proofs, coherent logic, geometry construction problems, constructions by ruler and compass

Research area: Computer Science

Research sub-area: Artificial Intelligence

UDC number: 004.832.3:514.181.8(043.3)

Захвалница

Имала сам ретку срећу да ми и у основној и средњој школи математику предају професори који су били заљубљеници у математику, који су ми пренели своју љубав према математици. Од свих области математике одувек ме је највише привлачила геометрија, а посебно решавање конструктивних проблема. Област математичке логике и аутоматског резоновања упознала сам на часовима Математичке логике у рачунарству на четвртој години студија и одмах сам препознала да је то нешто што ми “лежи” и чиме бих волела да се бавим. Главни разлог за то била су изузетно надахнута и мотивишућа предавања професора Предрага Јаничића. Исти професор је крајем мојих студија био и један од оних који су ме убедили да останем на факултету, упишем докторске студије и бавим се научним радом. Током свих година израде докторске дисертације мотивисао ме је да што више научим, пружао ми је подршку и помоћ када ми је била потребна и указивао стрпљење када је за тим било потребе, те моја најискренија захвалност иде првенствено њему.

Свим члановима комисије захваљујем се на помоћи током израде докторске дисертације, као и на корисним саветима и коментарима који су допринели да она буде боља и потпунија. Желела бих да се захвалим и Харолду Конелију, инжењеру хемије у пензији, који се читавог живота из хобија бавио решавањем конструктивних проблема и који је са мном несебично поделио нека од својих решења конструктивних проблема. Захваљујем се и колегиници Сани са којом сам провела сате, дане и месеце радећи на заједничком раду који је био основа за даљи рад на дисертацији. И све остале многобројне колеге са катедре са којима сам дивно сарађивала како на наставном, тако и на научном плану, учиниле су ми рад на тези лакшим, а боравак и рад на факултету изузетно пријатним.

Ипак, највећу помоћ и подршку током читавог мог образовања, као и током израде ове докторске дисертације, добијала сам од своје породице: маме, тате, брата, бака, супруга и његових родитеља, и свима њима дугујем највећу захвалност, док је Стефан био највећа мотивација да се израда ове дисертације што пре приведе крају. Хвала!

Садржај

1	Увод	1
1.1	Мотивација и циљ тезе	2
1.2	Доприноси тезе	3
1.3	Организација тезе	5
1.4	Summary of Chapter <i>Introduction</i>	7
2	Преглед основних појмова	9
2.1	Заснивања геометрије	9
2.2	Конструкције лењиром и шестаром	11
2.2.1	Изазови у решавању конструктивних задатака	13
2.2.2	Кратак преглед историје конструктивних проблема	15
2.2.3	Алгебарски и геометријски приступ	16
2.2.4	Корпуси конструктивних проблема	17
2.3	Аутоматско и интерактивно доказивање теорема	20
2.3.1	Аутоматско доказивање теорема	22
2.3.2	Интерактивно доказивање теорема	22
2.3.3	Аутоматско и интерактивно доказивање теорема у геоме- трији	23
2.4	Кохерентна логика и аутоматско доказивање теорема у геометрији	25
2.5	Summary of Chapter <i>Background</i>	28
3	Доказивач ArgoCLP и проширења	30
3.1	Опис доказивача ArgoCLP	30
3.1.1	ArgoCLP процедуре доказивања	31
3.1.2	Имплементација и коришћење система ArgoCLP	37
3.2	Примери доказа	44
3.3	Модул за поједностављивање доказа	51
3.3.1	EIN трансформација	54

3.3.2	Својства EIN трансформације	61
3.3.3	LUP&ERN трансформација	63
3.3.4	Својства LUP&ERN трансформације	69
3.3.5	RAA трансформација	70
3.3.6	Својства RAA трансформације	73
3.3.7	Својства PST трансформације	74
3.4	Евалуација и примери поједностављених доказа	75
3.5	Преглед других приступа и система	85
3.5.1	Преглед других доказивача за кохерентну логику	85
3.5.2	Преглед постојећих система за презентовање и поједноста- вљивање доказа	86
3.6	Summary of Chapter <i>ArgoCLP Prover and its Extensions</i>	89
4	Аутоматско решавање конструктивних проблема	91
4.1	Преглед система ArgoTriCS	91
4.2	Репрезентација знања	92
4.2.1	Дефиниције	96
4.2.2	Леме	99
4.2.3	Примитивне конструкције	100
4.3	Преглед алгоритма	101
4.3.1	Генерисање проблема	101
4.3.2	Фаза препроцесирања	102
4.3.3	Фаза решавања	103
4.4	Имплементација	110
4.5	Пропратни резултати	113
4.6	Евалуација и примери	114
4.6.1	Резултати за разматране корпусе	114
4.6.2	Анализа коришћеног знања	134
4.6.3	Примери	134
4.7	Поређење са другим системима	148
4.8	Формализација	150
4.8.1	Анализа	152
4.8.2	Конструкција	156
4.8.3	Доказ	156
4.8.4	Дискусија	159
4.8.5	Аутоматизација процеса формализације	163
4.9	Summary of Chapter <i>Automated Solving of Construction Problems</i> . .	165

5	Закључци и даљи рад	167
5.1	Summary of Chapter <i>Conclusions and Future Work</i>	172
	Литература	182
A	Хилбертов аксиоматски систем	183
A.1	Дефиниције	183
A.2	Аксиоме	184
B	Списак лема и конструкцијских примитива	186
B.1	Леме	186
B.2	Примитивне конструкције	190
C	Примери решених задатака	192
C.1	Problem 582	192
C.1.1	Problem	192
C.1.2	Status	192
C.1.3	Solution	192
C.1.4	Correctness proof	197
C.2	Problem 589	200
C.2.1	Problem	200
C.2.2	Status	200
C.2.3	Solution	200
C.2.4	Correctness proof	204
C.3	Problem 42	207
C.3.1	Problem	207
C.3.2	Status	207
C.3.3	Solution	207
C.3.4	Correctness proof	208
	Биографија кандидата	212

1

Увод

Геометрија је одувек имала врло значајну улогу у математичком образовању због тога што захтева парадигматично резонување. Из истог разлога већ деценијама представља област пуну изазова за доказивање теорема уз помоћ рачунара, при чему је највише пажње посвећено еуклидској геометрији.

Проблеми геометријских конструкција уз помоћ лењира и шестара представљају један од најстаријих и најизазовнијих проблема у елементарној математици. Решење конструктивног проблема није слика, већ процедура којом се, на основу задатих конструкцијских примитива, у низу корака даје “упутство” како треба конструисати тражени објекат. Главни проблем при решавању конструктивних проблема, и за човека и за рачунар, представља огроман простор претраге, јер постоји велики број различитих начина на које је могуће применити конструкцијске примитиве.

Задатак конструктивних проблема је да се на конструктивистички начин докаже теорема следећег облика:

$$\forall X \exists Y. \Psi(X, Y) \quad (1.1)$$

при чему X означава вектор објеката који су задати (то могу бити тачке, праве, кругови итд.), Y вектор објеката које треба конструисати, а Ψ услове који треба да важе између ових објеката. На пример, ако је задатак да се за дате тачке A , B и G конструише троугао ABC тако да је тачка G тежиште троугла ABC , онда је задатак доказати тврђење:

$$\forall A, B, G. (\exists C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)))$$

при чему $\text{centroid}(G, A, B, C)$ означава да је тачка G тежиште троугла ABC , а

$collinear(A, B, C)$ да су тачке A , B и C колинеарне.

Овим су обухваћена два тврђења: да је проблем решив и да је конкретна конструкција (која представља сведока за тврђење $\exists Y. \Psi(X, Y)$) коректна. У оквиру описа проблема, за дате објекте X могу бити дата нека ограничења. Наиме, немају сви конструктивни проблеми решење: неки проблеми немају решење, а неки га имају само под неким додатних скупом услова, који нису унапред познати. Стога, уместо доказивања једначине (4.1), често је задатак открити $\Phi(X)$ (за дато $\Psi(X, Y)$) и доказати:

$$\forall X. (\Phi(X) \Rightarrow \exists Y. \Psi(X, Y)) \quad (1.2)$$

Овим се тврди да решење постоји под неким скупом услова. На пример, у претходном примеру решење постоји у случају да су тачке A , B и G неколинеарне. Али могуће је тврдити и више од овога, да решење постоји под неким скупом услова Φ , а да иначе решење не постоји. Проблем је што се често ови услови не могу изразити само у терминима датих објеката X , већ укључују и неке помоћне објекте (који се користе током конструкције).

1.1 Мотивација и циљ тезе

У последњих неколико десетина година развијени су бројни аутоматски доказивачи теорема засновани на логикама различите изражајне моћи. Њихов главни задатак представља аутоматска провера тачности неког тврђења и то што је ефикасније могуће. Као резултат, они често генеришу само одговор *да/не* на питање да ли је одређено тврђење теорема или не (нпр. резолуцијски доказивачи, алгебарски доказивачи у геометрији и слично). Неки доказивачи генеришу и одговарајуће трагове доказа или доказне објекте. Ипак, стил генерисаних доказа се разликује од стила који користе математичари – разликује се како језик којим се изражавају проблеми и решења, тако и природа решења. С обзиром на то да аутоматске генерисане доказе математичари обично не могу у потпуности разумети, често се сматра да њихова исправност у великој мери зависи од коришћеног софтвера. Чак и када су ови аутоматски генерисани докази читљиви, често се састоје из хиљада корака, од којих су многи ирелевантни, што математичарима ове доказе често чини бескорисним. На пример, може се десити да аутоматско доказивање тврђења које је једнако некој од аксиома у аксиоматском систему резултује трагом доказа који се састоји из десетина корака. У овом случају се очигледно траг доказа може скратити на дужину од само једног корака који одговара примени аксиоме којој је једнак. Све ово ука-

зује на то да би било од великог значаја аутоматски генерисати доказе који су кратки и читљиви, а чија је исправност формално потврђена.

Што се аутоматског доказивања теорема у геометрији тиче, још од педесетих година прошлог века јављају се занимљиви приступи, али је највећи успех постигнут у последњим декадама двадесетог века. Ипак, скоро сви постојећи приступи се фокусирају на универзално квантификоване теореме. С друге стране, постоји врло мало студија на тему логичких приступа геометријским конструкцијама. Као последица тога, аутоматско доказивање теорема облика $\forall\exists$, које одговарају проблемима геометријских конструкција, врло је мало проучавано.

Аутоматско решавање конструктивних проблема било би од користи милионима ученика који решавају ову врсту проблема у школама, у случајевима када поред себе немају наставника да им помогне. Такође, могло би бити од користи и у неким индустријским применама, рецимо у роботизици, када се неки проблем може свести на конструктивни проблем.

Основни циљ ове тезе је *развој рачунарски подржане методологије за генерисање формално верификованих решења конструктивних проблема у геометрији, коришћењем развијеног аутоматског решавача конструктивних проблема и развијеног аутоматског доказивача за кохерентну логику, а подржаног алгебарским и интерактивним доказивачима теорема.*

1.2 Доприноси тезе

У оквиру рада на докторској тези, предложена је нова метода за решавање конструктивних проблема и реализована његова имплементација; метода успешно решава скоро све проблеме из разматраних корпуса, а за одређени број проблема добијају се комплетно верификована решења. У наставку текста набројани су основни доприноси тезе.

- Развијена је метода за аутоматско доказивање у кохерентној логици којом се аутоматски могу генерисати докази на природном језику, као и одговарајући формални докази. Овај метод је генерички, па га је могуће применити на произвољну кохерентну теорију. Метода је имплементирана у виду доказивача ArgoCLP и доступан је са адресе <http://argo.matf.bg.ac.rs/>. Рад који описује ову методу и доказивач ArgoCLP приказан је на конференцији *Automated Deduction in Geometry 2010 (Munich, Germany)* и објављен у Springer-овој серији LNAI годину дана након тога [98].
- Развијена је метода за поједностављивање доказа у оквиру кохерентне ло-

гике, којим се из доказа добијених аутоматским доказивачима теорема елиминишу сви сувишни кораци. Ова метода је имплементирана у оквиру доказивача ArgoCLP. Рад који описује ову методу прихваћен је за објављивање у часопису *Computing and Informatics* у току 2015. године [67].

- Извршена је систематизација знања потребног за решавање једне велике класе конструктивних проблема у геометрији – проблема у којима је задатак конструисати троугао ако су задате локације три значајне тачке.
- Развијена је метода за аутоматско решавање локацијских конструктивних проблема. Решење укључује неформални опис конструкције на природном језику, формални опис конструкције коришћењем језика GCLC, доказ (коришћењем алгебарских доказивача теорема) да генерисана конструкција одговара спецификацији, као и услове недегенерисаности под којима решење постоји. Метода је имплементирана у виду система ArgoTriCS и доступан је са адресе <http://argo.matf.bg.ac.rs/>. Рад који описује ову методу, као и потребну систематизацију геометријског знања, приказан је на конференцији *Intelligent Computer Mathematics, CICM 2012 (Bremen, Germany)* и објављен у *Springer*-овој серији *LNAI* [68]. Такође, рад који детаљно описује систем ArgoTriCS послат је на разматрање за објављивање у часопис на *SCI* листи.
- Дат је (вероватно први) ригорозан, логички опис решења конструктивних проблема у геометрији и показано да се сви делови решења своде на доказивање теорема.
- За неке од аутоматски решених конструктивних проблема добијени су формални докази коришћењем интерактивног доказивача теорема Isabelle. Рад који описује формални логички оквир за описивање традиционалног четворофазног процеса решавања конструктивних проблема, заједно са примером генерисања формалног доказа комплетног решења конструктивног проблема, приказан је на конференцији *Automated Deduction in Geometry 2014 (Coimbra, Portugal)* и на конференцији *GC 2015 – International Seminar on Geometric Computation (Nanning, China)*. Такође, рад на ову тему прихваћен је за штампање у *Springer*-овој серији *LNAI* у току 2015. године.
- Аутоматски је генерисана електронска збирка решених конструктивних задатака за велики број проблема из разматраних корпуса. Збирка

је доступна са адресе: http://poincare.matf.bg.ac.rs/~vesnap/PhD_repository.zip.

1.3 Организација тезе

Преглед основних појмова (глава 2) У оквиру овог поглавља приказани су основни појмови, технике и алати који се користе у остатку тезе. Најпре је дат кратак преглед различитих заснивања геометрије, развијених од времена старе Грчке до данас (поглавље 2.1), а затим је дат кратак опис конструктивних проблема, као и преглед различитих приступа њиховом решавању (поглавље 2.2). Постоје два главна правца у доказивању теорема: аутоматско доказивање теорема и интерактивно доказивање теорема (поглавље 2.3). Природан оквир у коме се једноставно може формулисати велики број теорема из геометрије представља кохерентна логика (поглавље 2.4).

Доказивач ArgoCLP и проширења (глава 3) У оквиру овог поглавља приказан је аутоматски доказивач теорема за кохерентну логику ArgoCLP (Automated Reasoning GrOup Coherent Logic Prover). Овај доказивач користи различите технике доказивања којима је могуће побољшати ефикасност основне процедуре доказивања (поглавље 3.1). Генерисани докази (у форми низа корака) врло су слични доказима који су дати у стандардним геометријским уџбеницима (поглавље 3.2). Битан елемент доказивача представља модул за поједностављивање трагова доказа конструисаних у оквиру кохерентне логике. Процедура поједностављивања се састоји из три фазе: прва фаза се заснива на елиминацији корака који су ирелевантни за текући доказ, чиме се омогућава да се елиминишу и нека ирелевантна гранања, у другој фази се издижу кораци кроз гранајуће кораке, што је праћено елиминисањем поновљених корака, док се у трећој фази фрагменти доказа трансформишу у *reductio ad absurdum* форму, уколико је то могуће (поглавље 3.3). Поједностављени докази могу се извести у различите формате, од којих су најбитнији традиционални читљиви докази теорема на природном језику и формални докази у Isabelle/Isar формату (поглавље 3.4). На крају овог поглавља дато је поређење са сличним системима (поглавље 3.5).

Аутоматско решавање конструктивних проблема (глава 4) У овом поглављу предложена је метода за аутоматско решавање класе конструктив-

них проблема код којих је потребно, коришћењем лењира и шестара, конструисати троугао ако су дате локације произвољне три карактеристичне тачке у троуглу. Метода се заснива на идентификовању и систематизацији потребног геометријског знања (при чему се потребно знање разврстава на дефиниције, леме и конструкцијске примитиве), на специфичној усмереној претрази којом се ограничава скуп објеката који се конструише и обради редундантних инстанци проблема, као и проблема који су зависни од положаја. Предложена метода је имплементирана у оквиру система ArgoTriCS (Automated Reasoning GrOup Triangle Construction Solver) (поглавље 4.1) и текућа верзија имплементације може да реши велики број конструктивних проблема из разматраних корпуса (поглавље 4.6). Коректност генерисаних конструкција може се проверити коришћењем екстерних аутоматских доказивача теорема. Извршено је и поређење са постојећим системима за решавање конструктивних задатака (поглавље 4.7).

Развијен је и формалан логички оквир за описивање традиционалног процеса решавања конструктивних проблема у геометрији (поглавље 4.8). Дат је прецизан опис класичне форме решења конструктивних проблема у логици првог реда. Ово је према нашем сазнању први овакав опис и он служи као основа рачунарски подржане методологије за решавање конструктивних проблема и генерисање њихових решења, подржане аутоматским доказивачима теорема и формалним доказима унутар интерактивних доказивача теорема.

Закључци и даљи рад (глава 5) У оквиру овог поглавља сумирани су резултати тезе и дате идеје о могућим правцима даљег истраживања.

Хилбертов аксиоматски систем (додатак А) У оквиру овог поглавља дат је скуп дефиниција и аксиома Хилбертовог аксиоматског система.

Списак лема и конструкцијских примитива (додатак В) У оквиру овог поглавља дат је скуп лема и конструкцијских примитива које се користе приликом решавања проблема из разматраних корпуса.

Примери решених задатака (додатак С) У оквиру овог поглавља дат је приказ неких од решених конструктивних задатака. Решења задатака наликују традиционалним решењима и састоје се од аутоматски генерисане конструкције на природном језику, формалне спецификације конструкције коришћењем језика GCLC уз генерисање одговарајуће илустрације, извештаја да ли је неки од алгебарских доказивача доказао исправност

конструкције, као и услова недегенерисаности под којима постоје решења проблема.

1.4 Summary of Chapter *Introduction*

Construction problems in geometry are one of the oldest and the most challenging problems in elementary mathematics. A solution of a construction problem is not a figure, but the procedure which, using a set of given construction primitives, in a sequence of steps, gives a “recipe” how to construct the object sought. The main problem in solving construction problems, both for a human and a computer, is a huge search space, since there are many ways in which construction primitives can be applied.

The main task in solving a construction problem is to prove a theorem of the form:

$$\forall X \exists Y. \Psi(X, Y) \tag{1.3}$$

in constructive way, where X denotes a sequence of objects given (points, lines, circles, ...), Y a sequence of objects to be constructed, and Ψ conditions that have to hold among these objects.

There are very few studies concerning logical approaches to geometric constructions. As a consequence, automated proving of the theorems of the form $\forall \exists$, which correspond to geometric construction problems, has been rarely studied.

The main goal of this thesis is *development of semiautomated methodology for generation of formally verified solutions of construction problems in geometry, using a developed automated solver of construction problems and a developed automated prover for coherent logic, supported by algebraic and interactive theorem provers.*

The main contributions of this thesis are:

1. The method for automated proving in the framework of coherent logic, which can automatically generate proofs in natural language form, as well as formal proofs, is developed; it is implemented within ArgoCLP prover and is available from: <http://argo.matf.bg.ac.rs/>. The paper

describing this method and the prover ArgoCLP has been presented at conference *Automated Deduction in Geometry 2010 (Munich, Germany)* and published in *Springer's* series *LNAI* a year after [98].

2. A method for simplification of proofs in the framework of coherent logic, generated by automated theorem provers, is developed – this method is implemented within the ArgoCLP prover. The paper that describe this method is described in a paper to appear in the journal *Computing and Informatics* [67].
3. The knowledge needed for solving one big class of constructive problems in geometry has been systematized – these are the problems where the task is to construct a triangle if the locations of three significant points are known.
4. A method for automated solving of location triangle construction problems is developed – this method is implemented within a tool ArgoTriCS, which is available from <http://argo.matf.bg.ac.rs/>. The paper describing this method, as well as the systematization of geometrical knowledge needed, has been presented at the conference *Intelligent Computer Mathematics, CICM 2012 (Bremen, Germany)* and has been published in *Springer's* series *LNAI* [68]. Also, a paper which describes the system ArgoTriCS in more details has been submitted to the journal on *SCI* list.
5. Probably the first rigorous logical description of the solution of construction problem in geometry is given and it has been shown that all parts of the solution are reduced to theorem proving.
6. For a subset of the class of automatically solved problems, formal proofs of generated solutions are automatically generated. The paper describing the formal logical framework for describing traditional four-phase process solving of construction problems, as well as one example of generation of the formal proof of complete solution of construction problems, has been presented at conference *Automated Deduction in Geometry 2014 (Coimbra, Portugal)* and at conference *GC 2015 – International Seminar on Geometric Computation (Nanang, China)*. Also, a paper on this topic has been accepted for publication in *Springer's* series *LNAI*.
7. An electronic textbook consisting of solutions to problems from Wernick's and Connelly's corpora is automatically generated. It is available from http://poincare.matf.bg.ac.rs/~vesnap/PhD_repository.zip.

Преглед основних појмова

У овом поглављу биће дат кратак преглед основних области релевантних за ову тезу. На почетку ће бити укратко приказан преглед различитих заснивања геометрије, а затим ће бити дато кратко подсећање на проблеме конструкције троугла у геометрији. Након тога, биће укратко приказана два главна правца у доказивању теорема уз помоћ рачунара: аутоматско и интерактивно доказивање теорема, са посебним освртом на примену у геометрији. На крају овог поглавља биће укратко приказан један фрагмент логике првог реда, познат као кохерентна логика, погодан за доказивање теорема у геометрији.

2.1 Заснивања геометрије

Еуклид (*Ευκλείδης*) се сматра првим математичарем који је систематски приказао и користио аксиоматску методу [50]. Његово дело “Елементи”, написано око 300. године п.н.е., у 13 књига покрива области еуклидске геометрије и елементарне теорије бројева, као и једну класу алгебарских проблема коју су Грци називали геометријском алгебром¹; састоји се од дефиниција, аксиома, теорема и конструкција, као и математичких доказа ових тврђења. Еуклид је коришћењем чисто логичких правила успео да докаже многа геометријска тврђења која су била позната и давно пре њега. “Елементи” се сматрају најпознатијом и најутицајнијом расправом из геометрије свих времена и више од два миленијума служили су као уџбеник на коме се темељило више образовање [63]. Такође, сматрани су темељем математике све до деветнаестог века када је доведена у питање коректност Еуклидових доказа због коришћења дијаграма

¹У геометријској алгебри све величине се разматрају као дужине дужи, а тврђења, која се данас једноставно исказују алгебарским изразима, доказују се коришћењем теорема из геометрије.

у њима. Наиме, његови докази се ослањају на неке имплицитне претпоставке, док се у математичким доказима све претпоставке морају задати експлицитно. Такође, доведена је у питање исправност доказа у општем случају јер су се докази ослањали на закључке изведене за конкретне дијаграме.

Давид Хилберт (David Hilbert) је 1899. године, у својој књизи “Основе геометрије” (“Grundlagen der Geometrie”), предложио нови аксиоматски систем за еуклидску геометрију који је поправио многе недостатке и слабости Еуклидовог система [51]. Овај Хилбертов рад сматра се прекретницом у математици двадесетог века. Његов аксиоматски систем користи три врсте примитивних објеката: тачке, праве и равни, док је скуп аксиома подељен у пет група: аксиоме инциденције, аксиоме поретка, аксиоме конгруенције, аксиоме паралелности и аксиому непрекидности. Уз сваку групу аксиома дате су неке основне теореме које се могу доказати коришћењем само претходних аксиома. За разлику од Еуклида који је дефинисао и основе појмове као што су тачке, праве и равни, код Хилберта су тачке, праве и равни три примитивна појма која се не дефинишу. Чак и данас, и поред тога што је доста тога замерано овом систему, геометрија почива на принципима које је утемељио Хилберт. Такође, његов рад је значајан јер је иницирао истраживања која се односе на потпуност, непротивречност и независност аксиоматског система. Једну од савремених варијанти Хилбертовог система формулисали су Карол Борсук (Karol Borsuk) и Ванда Смилева (Wanda Szmielew) [11].

Након Хилбертовог рада, историјат развоја аксиоматских основа геометрија почиње да мења свој фокус – аксиоматски системи се више не конструишу са циљем да се попуне пропусти и отклоне недостаци претходних аксиоматских система, већ се креће у потрагу за најелегантнијим аксиоматским системом за дату теорију. Алфред Тарски (Alfred Tarski) је средином двадесетог века представио нову аксиоматизацију првог реда за *елементарну геометрију* (тачније, више различитих варијанти), заједно са процедуром одлучивања за ту теорију [94]. Под елементарном геометријом Тарски подразумева фрагмент еуклидске геометрије који се може формулисати у логици првог реда са једнакошћу без коришћења теорије скупова [101]. Аксиоматски систем Тарског врло је једноставан: заснован је на само једној врсти примитивних објеката – тачкама, користи само два предиката – *подударно* и *између* и једанаест аксиома. Све аксиоме су једноставне и изражене само у терминима примитивних појмова. Аксиоматика Тарског даје и својства непрекидности која су слабија од својства Хилбертове аксиоматике.

Развој нових аксиоматских система представља и даље активну област ис-

траживања. Недавно су Цереми Авигад (Jeremy Avigad), Едвард Дин (Edward Dean) и Џон Мума (John Mumma) предложили нову аксиоматизацију еуклидске геометрије која прилично верно прати основне идеје из Еуклидових “Елементата”, али их спроводи на стриктан и темељан начин [5]. Они полазе од тврђења да дијаграматско резонување које је користио Еуклид нема толико слабости као што се тврди и да је такав вид резонувања природнији од аксиоматизација геометрије које су формулисане у међувремену. Да би ово потврдили, аутори су увели формални аксиоматски систем Е којим се прецизно задаје који се закључци могу “прочитати” из дијаграма. Они су показали да је овај систем доказивања сагласан и потпун за модерну семантику конструкција лењиром и шестаром, која је одговарајућа приступу коришћеном у “Елементима” [5].

2.2 Конструкције лењиром и шестаром

Проблеми конструкције троугла (у еуклидској равни) су проблеми у којима је потребно коришћењем лењира² и шестара конструисати троугао који задовољава дата ограничења (најчешће три) [33, 70, 61]. Задатак у конструктивним проблемима је да се за дату декларативну спецификацију геометријске фигуре одреди одговарајућа, по могућству еквивалентна, процедурална спецификација заснована на расположивим конструктивним корацима (у том случају треба доказати оба смера ове еквиваленције). Процедурална спецификација треба да буде изражена у терминима алата које је дозвољено користити у конструкцији. Најчешће се као алати разматрају лењир и шестар, али је могуће користити и друге комбинације алата - посебне врсте лењира и шестара, оригами итд. У наставку ће од алата бити разматрани само лењир и шестар, и то лењир који на себи не садржи ознаке дужина. Ово је најстарија и најпроучаванија класа конструктивних проблема.

Конструктивна геометрија почива на првим трима Еуклидовим постулатима. Први и други Еуклидов постулат се односе на лењир, а трећи на шестар³.

²Подразумева се да на лењиру не постоје ознаке које се могу користити за мерења

³Први Еуклидов постулат гласи да се од сваке тачке ка свакој другој тачки може повући права линија. Други Еуклидов постулат гласи да се свака ограничена права може бесконачно продужити у праву. Трећи Еуклидов постулат гласи да се око сваког средишта сваким растојањем може описати круг [63].

Дефиниција 2.1. *Под конструкцијом помоћу лењира и шестара подразумева се низ примитивних (елементарних) конструктивних корака, тако да је сваки од њих из наредног скупа корака:*

- *конструкција произвољне тачке (потенцијално различите од неких задатих тачака);*
- *конструкција (коришћењем лењира) праве која пролази кроз две дате тачке;*
- *конструкција (коришћењем шестара) круга који има центар у некој од задатих тачака и пролази кроз неку другу задату тачку;*
- *конструкција пресека (уколико постоји) два круга, две праве или праве и круга.*

Дефиниција 2.2. *Нека је задат коначан скуп тачака $\mathcal{B} = \{B_0, \dots, B_m\}$ у еуклидској равни. За тачку P кажемо да ју је могуће конструисати помоћу лењира и шестара почев од тачака $\mathcal{B}$ ако постоји коначан скуп тачака $\{P_0, \dots, P_n\}$ таквих да је $P = P_n$ и за сваку тачку $P_i (0 \leq i \leq n)$ важи да или припада скупу $\mathcal{B}$ или је добијена као пресек две праве, праве и круга или два круга, добијених на следећи начин:*

- *сваки разматрани круг има центар који је у скупу тачака $\{P_0, \dots, P_{i-1}\}$, а полупречник му је једнак растојању $P_j P_k$ за неко $j < i$ и $k < i$;*
- *свака разматрана права пролази кроз две тачке скупа $\{P_0, \dots, P_{i-1}\}$.*

Решење конструктивног проблема, традиционално, укључује наредне четири фазе/компоненте [33]:

Анализа: У анализи се обично креће од претпоставке да одређени геометријски објекти задовољавају спецификацију Ψ , а онда доказује да важе својства *Plans* која омогућавају конструкцију.

Конструкција: У овој фази треба описати конструкцију помоћу лењира и шестара, засновану на анализи (тј. на својствима *Plans* која се у оквиру ње одређују).

Доказ: У овој фази треба доказати да генерисана конструкција помоћу лењира и шестара задовољава дату спецификацију проблема, тј. услове Ψ .

Дискусија: У фази дискусије треба размотрити колико проблем има решења и под којим условима она постоје.

Доказ да је нека конструкција коректна се заправо састоји из две компоненте:

- доказа да конструисани објекат заиста представља решење полазног проблема и
- доказа да се конструкцијом долази до свих решења полазног проблема, као и дискусије којом се долази до полазних услова који доводе до различитих (неподударних) решења.

Анализа представља део доказа који се, из педагошких разлога, ставља на почетак решења, јер се њоме долази до мотивације за добијену конструкцију. У овом смислу доказ исправности конструкције обухвата, поред фазе доказа, и фазу анализе и фазу дискусије.

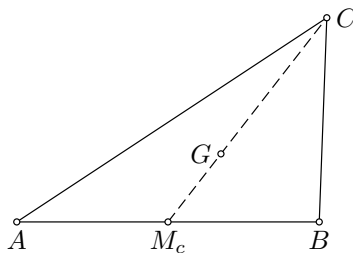
Решавање конструктивних проблема у геометрији повезује две важне области рачунарства примењене на геометрију:

- *аутоматско доказивање теорема*, где геометрија представља један од најуспешнијих домена почев од кључног Гелернтеровог рада [43];
- *динамички геометријски софтвер*, који је направио велике измене у образовној пракси геометрије коришћењем визуелизације и могућношћу експериментисања.

Обе ове области јако су повезане са конструктивним проблемима. Многе методе за аутоматско доказивање теорема у геометрији заснивају се на основним геометријским конструкцијама. Такође, конструкције које се изводе унутар динамичког геометријског алата обично одговарају конструкцијама помоћу лењира и шестара. Ипак, ове везе су до сада недовољно истражене.

2.2.1 Изазови у решавању конструктивних задатака

Главну потешкоћу при решавању конструктивних проблема, и за човека и за рачунар, представља огроман простор претраге: примитивни конструктивни кораци могу се применити на велики број начина, и број начина на који се могу применити расте како конструкција напредује. Као илустрацију, размотримо следећи, једноставан проблем: *Ако су задате тачке A , B и G , конструисати троугао ABC такав да је тачка G његово тежиште*. Једно од могућих решења овог проблема би било следеће: конструисати тачку M_c као средиште дужи AB , а затим конструисати тачку C тако да важи: $\overrightarrow{M_c G} / \overrightarrow{M_c C} = 1/3$ (слика 2.1).



Слика 2.1: Конструкција троугла ABC када су задата темена A и B и тежиште троугла G

Ово решење је веома једноставно и интуитивно. Ипак, ако бисмо хтели да опишемо систематски (аутоматски) начин на који би се дошло до решења проблема, морао би да се посматра широк опсег различитих могућности. На пример, након конструкције тачке M_c могла би се разматрати конструкција средишта дужи AG , BG , или чак дужи AM_c , BM_c , GM_c , па затим средишта дужи чији су крајеви међу овим тачкама, итд. Уместо корака којим се конструише тачка C , тако да је: $\overrightarrow{M_cG}/\overrightarrow{M_cC} = 1/3$, могло би се (непотребно) разматрати увођење тачке X за коју важи $\overrightarrow{AG}/\overrightarrow{AX} = 1/3$ или $\overrightarrow{BG}/\overrightarrow{BX} = 1/3$. Такође, уместо корака који уводи тачку C која задовољава $\overrightarrow{M_cG}/\overrightarrow{M_cC} = 1/3$ могло би се разматрати увођење тачке X за коју важи $\overrightarrow{M_cG}/\overrightarrow{M_cX} = k$, где је $k \neq 1/3$, итд. Дакле, и овако једноставан проблем показује да се сваку систематску методу за решавање конструктивних проблема може суочити са огромним простором претраге, чак и када се разматрају само два потребна конструктивна правила вишег нивоа. Додатни проблем представља чињеница да су неки од конструктивних проблема нерешиви, као што су, на пример, три античка геометријска проблема: квадратура круга, удвостручавање коцке и трисекција угла [96]. Ово је доказано алгебарском методом која се заснива на томе да се кругови и праве могу описати линеарним и квадратним једначинама, а једначине којима се описују квадратура круга, удвостручавање коцке и трисекција угла се не могу свести на линеарне и квадратне⁴. Ово не значи нужно да не постоји решење ових проблема у еуклидској равни, већ да не постоји решење које се може конструисати коришћењем само лењира и шестара. Иако је проблем конструктивности (коришћењем лењира и шестара) геометријске фигуре која се може описати алгебарским једначинама са коефицијентима у $\mathbb{Q}$ одлучив [41, 60, 18], не постоји једноставна и ефикасна, праволинијска процедура одлучивања, те је

⁴До овог се дошло тек почетком деветнаестог века, јер је било неопходно да се утемељи одговарајућа алгебарска теорија [63].

често јако тешко доказати да је проблем нерешив;⁵ углавном се докази нерешивости конструктивног проблема праве *ad-hoc* и нису изведени неким униформним алгоритмима.

2.2.2 Кратак преглед историје конструктивних проблема

Конструктивни проблеми се изучавају непрекидно још од времена античке Грчке и представљају област која је пуна изазова чак и за искусне математичаре. Од почетка двадесетог века “*геометрија троугла*”, која укључује и проблеме конструкције троугла, није сматрана примарном области математичких истраживања [29]. Ипак, конструктивни проблеми су задржали своју улогу на свим нивоима математичког образовања, скоро у истој форми, током више од два и по миленијума, што их врло вероватно чини проблемом који је најконстантније изучаван током историје математичког образовања. Почев од касних година двадесетог века геометријске конструкције поново постају предмет истраживања, али овај пут пре свега мета-математички и издвајају се два главна правца ових истраживања: истраживања аксиоматских основа геометријских конструкција и истраживања на тему развоја алгоритама за решавање конструктивних проблема.⁶

Према Виктору Памбучиану (Victor Pambuccian) и његовом прегледном раду на тему аксиоматизације геометријских конструкција, геометријске конструкције тек 1968. године постају део аксиоматизације геометрије [80]. У конструктивним аксиоматизацијама геометрије, према античком схватању, аксиоме су без квантификатора и укључују само функцијске симболе (којима се представљају конструктивни кораци) и индивидуалне константе, насупротив Хилбертовом приступу који укључује релационе симболе и код кога аксиоме нису универзална тврђења. Једну од аксиоматских теорија конструктивне геометрије — **ECG** (“Elementary Constructive Geometry”) недавно је предложио Мајкл Бисон (Michael Beeson) [7]. Конструктивне аксиоматизације представљају алтернативни приступ заснивању геометрије. Међутим, када је реч о решавању конкретног конструктивног проблема, оне не доносе значајнију предност у односу на Хилбертов приступ.

Упркос дугој традицији решавања конструктивних проблема, развијено је

⁵Тачније, рачун се изводи на проширењима над $\mathbf{Q}$: $\mathbf{Q}(a_1, a_2, \dots, a_k)$, где су са $a_1, a_2, \dots, a_k$ означене променљиве (параметри), те разматрани полиноми припадају $\mathbf{Q}(a_1, a_2, \dots, a_k)[X_1, X_2, \dots, X_m]$.

⁶Постоје и радови на тему истраживања сложености геометријских конструкција коришћењем лењира и шестара, који се заснивају или на пребројавању потребног броја операција или на статистичким методама [73].

тек неколико приступа за њихово аутоматско решавање [41, 44, 47, 89]. Највећи део њих, ако не и сви, концентрише се на процедуре претраге, а не на детекцију геометријског знања (аксиоме и леме) на којем се конструкције базирају, иако сви приступи имају стриктну листу расположивих примитивних конструктивних корака. Ранији ручни покушаји систематизовања конструктивних проблема такође нису дали малу и јасну (ако је могуће минималну у неком смислу) листу потребног геометријског знања [38, 39, 61]. Такође, већина постојећих приступа уопште не разматра доказивање коректности решења [47].

Постоје и истраживања конструктивних проблема при којима је на пример дозвољено користити само лењир, или је дозвољено користити само шестар или се забрањују неке од операција неке од датих алатки или се дозвољава употреба нових алатки као што је, на пример, оригами. Неки од ових скупова алата су еквивалентни лењиру и шестару (Мор-Маскеронијева теорема тврди да се било која геометријска конструкција која се може извести коришћењем лењира и шестара може извести и коришћењем само шестара [52]), неки су слабије моћи (на пример, није могуће конструисати средиште дате дужи коришћењем само лењира, без употребе произвољних тачака [6]), док је на пример оригами моћнији алат [2]. У овом раду бавићемо се искључиво проблемима конструкције троугла коришћењем лењира и шестара.

2.2.3 Алгебарски и геометријски приступ

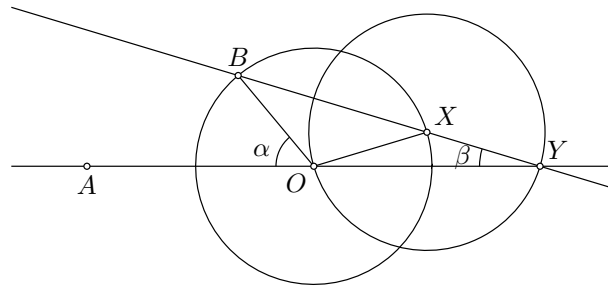
У данашње време разликујемо два приступа решавању конструктивних задатака: *алгебарски* и *геометријски приступ*. Напредак у области алгебре почетком деветнаестог века омогућио је решавање великог броја геометријских проблема који су били нерешени још од времена старих Грка. Приликом разматрања конструктивних проблема могу се анализирати два аспекта: конструктибилност и конструкција. У оба случаја алгебарске методе могу да играју значајну улогу, али већи успех и значај алгебарски алати имају у доказивању неконструктибилности. Теоријски је могуће екстраховати конструкцију из доказа неконструктибилности, али је то често непрактично, а чак и када је то ефективно могуће, нема никакву педагошку вредност.

Што се геометријског приступа тиче, за фигуру се може доказати да се не може конструисати коришћењем лењира и шестара (*i*) ако је сама спецификација неконзистентна (не постоји тражена фигура у еуклидској равни без обзира на то како би се произвела); (*ii*) ако се проблем може свести на неки други проблем (најчешће на неки *канонски* проблем) за који се зна да је нерешив коришћењем лењира и шестара.

У наставку је приказан једноставан пример примене геометријског приступа, који се базира на једној Архимедовој конструкцији⁷.

Пример 2.1. *Ако су задате три неколинеарне тачке A , B и O , конструисати тачке X и Y тако да важи (види слику 2.2):*

- $OX \cong OB$,
- $XY \cong OB$,
- тачке B , X и Y су колинеарне, и
- тачке A , O и Y су колинеарне.



Слика 2.2: Илустрација проблема који није могуће решити коришћењем лењира и шестара

Коришћењем елементарне геометрије може се доказати да је угао $\alpha = \angle AOB$ три пута већи од угла $\beta = \angle AYB$. Према томе, ако би овај проблем био решив, онда би био решив и проблем трисекције угла, а познато је да се у општем случају угао не може поделити на три једнака дела коришћењем само лењира и шестара.

2.2.4 Корпуси конструктивних проблема

Вилијем Верник (William Wernick) је 1982. године представио листу такозваних *локацијских проблема конструкције троугла* [108]. Код свих ових проблема задатак је конструисати троугао ако су дате позиције неке три тачке из следеће листе која садржи 16 карактеристичних тачака у троуглу:

⁷Архимед је показао да је могуће извршити трисекцију угла коришћењем лењира, шестара и траке папира

- A, B, C, O : три темена и центар описаног круга;
- M_a, M_b, M_c, G : средишта страница троугла и тежиште;
- H_a, H_b, H_c, H : подножја висина и ортоцентар;
- T_a, T_b, T_c, I : подножја бисектриса унутрашњих углова у троуглу и центар уписаног круга.

Постоји 560 тројки проблема које се могу описати коришћењем горе наведених тачака, али Верникова листа садржи само 139 значајно другачијих, нетривијалних проблема. На пример, проблем (A, B, C) је тривијалан, а проблеми (A, B, M_a) , (A, B, M_b) , (B, C, M_b) , (B, C, M_c) , (A, C, M_a) , и (A, C, M_c) се сматрају симетричним, тј. аналогним. Неке тројке проблема су редувантне. На пример, тројка (A, B, M_c) је редувантна јер ако су задате тачке A и B , тачка M_c је одређена на јединствен начин, па је њена појава редувантна у тројци (A, B, M_c) , али су ипак дате у Верниковој листи и означене су словом **R**. Неке тројке проблема су зависне од положаја. На пример, у тројци (A, B, O) тачка O мора да припада медијатриси дужи AB и у том случају постоји бесконачно много решења, а иначе проблем нема решења. У овим проблемима, у зависности од тога где се тачка налази, проблем има или бесконачно много решења или уопште нема решења. Ови проблеми у Верниковој листи означени су словом **L**. Постоји 113 проблема који не припадају групама означеним словом **R** или **L**. Преостали проблеми који се могу решити уз помоћ лењира и шестара означени су словом **S**, а проблеми који се не могу решити коришћењем лењира и шестара су означени словом **U**.

У оригиналној Верниковој листи, проблем 102 грешком је био означен са **S** уместо са **L** [74]. Такође, у првој верзији 41 проблема било је са недефинисаним статусом, али је 1996. година листа ажурирана тако да је остало 20 неklasификованих проблема [74]. У међувремену је доказано да су проблеми 90, 109, 110, 111 [95] и 138 [105] нерешиви. Паскал Шрек (Pascal Schreck) и Паскал Матис (Pascal Mathis) су недавно, коришћењем рачунарског алгебарског система Maple, за све преостале отворене проблеме утврдили да су нерешиви, осим за проблем 119 за који је доказано да је решив; такође су доказали да је проблем број 108 погрешно био класификован као нерешив⁸, а у ствари је решив [92]. За неке од проблема, као што су проблеми 43 [3, 28], 57 и 58 [26, 95], тражена су једноставнија решења. Наравно, многи од ових проблема су вековима разматрани

⁸Овај проблем је био класификован као нерешив, али нигде није био наведен доказ тог тврђења.

1. A, B, O	L	48. A, H_a, I	S	95. M_a, G, T_b	U [74]
2. A, B, M_a	S	49. A, H_b, H_c	S	96. M_a, G, I	S [74]
3. A, B, M_c	R	50. A, H_b, H	L	97. M_a, H_a, H_b	S
4. A, B, G	S	51. A, H_b, T_a	S	98. M_a, H_a, H	L
5. A, B, H_a	L	52. A, H_b, T_b	L	99. M_a, H_a, T_a	L
6. A, B, H_c	L	53. A, H_b, T_c	S	100. M_a, H_a, T_b	U [74]
7. A, B, H	S	54. A, H_b, I	S	101. M_a, H_a, I	S
8. A, B, T_a	S	55. A, H, T_a	S	102. M_a, H_b, H_c	L
9. A, B, T_c	L	56. A, H, T_b	U [74]	103. M_a, H_b, H	S
10. A, B, I	S	57. A, H, I	S [74]	104. M_a, H_b, T_a	S
11. A, O, M_a	S	58. A, T_a, T_b	S [74]	105. M_a, H_b, T_b	S
12. A, O, M_b	L	59. A, T_a, I	L	106. M_a, H_b, T_c	U [74]
13. A, O, G	S	60. A, T_b, T_c	S	107. M_a, H_b, I	U [74]
14. A, O, H_a	S	61. A, T_b, I	S	108. M_a, H, T_a	S [92]
15. A, O, H_b	S	62. O, M_a, M_b	S	109. M_a, H, T_b	U [95]
16. A, O, H	S	63. O, M_a, G	S	110. M_a, H, I	U [95]
17. A, O, T_a	S	64. O, M_a, H_a	L	111. M_a, T_a, T_b	U [95]
18. A, O, T_b	S	65. O, M_a, H_b	S	112. M_a, T_a, I	S
19. A, O, I	S	66. O, M_a, H	S	113. M_a, T_b, T_c	U [92]
20. A, M_a, M_b	S	67. O, M_a, T_a	L	114. M_a, T_b, I	U [74]
21. A, M_a, G	R	68. O, M_a, T_b	U [74]	115. G, H_a, H_b	U [74]
22. A, M_a, H_a	L	69. O, M_a, I	S	116. G, H_a, H	S
23. A, M_a, H_b	S	70. O, G, H_a	S	117. G, H_a, T_a	S
24. A, M_a, H	S	71. O, G, H	R	118. G, H_a, T_b	U [92]
25. A, M_a, T_a	S	72. O, G, T_a	U [74]	119. G, H_a, I	S [92]
26. A, M_a, T_b	U [74]	73. O, G, I	U [74]	120. G, H, T_a	U [74]
27. A, M_a, I	S [74]	74. O, H_a, H_b	U [74]	121. G, H, I	U [74]
28. A, M_b, M_c	S	75. O, H_a, H	S	122. G, T_a, T_b	U [92]
29. A, M_b, G	S	76. O, H_a, T_a	S	123. G, T_a, I	U [92]
30. A, M_b, H_a	L	77. O, H_a, T_b	U [92]	124. H_a, H_b, H_c	S
31. A, M_b, H_b	L	78. O, H_a, I	U [92]	125. H_a, H_b, H	S
32. A, M_b, H_c	L	79. O, H, T_a	U [74]	126. H_a, H_b, T_a	S
33. A, M_b, H	S	80. O, H, I	U [74]	127. H_a, H_b, T_c	U [92]
34. A, M_b, T_a	S	81. O, T_a, T_b	U [92]	128. H_a, H_b, I	U [92]
35. A, M_b, T_b	L	82. O, T_a, I	S [74]	129. H_a, H, T_a	L
36. A, M_b, T_c	S	83. M_a, M_b, M_c	S	130. H_a, H, T_b	U [74]
37. A, M_b, I	S	84. M_a, M_b, G	S	131. H_a, H, I	S [74]
38. A, G, H_a	L	85. M_a, M_b, H_a	S	132. H_a, T_a, T_b	U [92]
39. A, G, H_b	S	86. M_a, M_b, H_c	S	133. H_a, T_a, I	S
40. A, G, H	S	87. M_a, M_b, H	S [74]	134. H_a, T_b, T_c	U [92]
41. A, G, T_a	S	88. M_a, M_b, T_a	U [74]	135. H_a, T_b, I	U [92]
42. A, G, T_b	U [74]	89. M_a, M_b, T_c	U [74]	136. H, T_a, T_b	U [92]
43. A, G, I	S [74]	90. M_a, M_b, I	U [95]	137. H, T_a, I	U [92]
44. A, H_a, H_b	S	91. M_a, G, H_a	L	138. T_a, T_b, T_c	U [105]
45. A, H_a, H	L	92. M_a, G, H_b	S	139. T_a, T_b, I	S
46. A, H_a, T_a	L	93. M_a, G, H	S		
47. A, H_a, T_b	S	94. M_a, G, T_a	S		

Табела 2.1: Верникови проблеми и њихов статус

и решавани без контекста ове листе. Коначни статус проблема из Верникове листе дат је у табели 2.1: постоји 74 **S** проблема, 39 **U** проблема, 3 **R** проблема и 23 **L** проблема. За све новије резултате дата је референца на рад у коме је за тај проблем утврђен његов статус. Решења 59 решивих проблема могу се наћи на интернету [95].

Харолд Конели (Harold Connolly) је 2009. године представио и делимично решио проширену Верникову листу [25], која садржи четири додатне тачке:

- E_a, E_b, E_c : три Ојлерове тачке, које представљају средишта дужи одређених теменима A, B и C и ортоцентром H ;
- N : центар Ојлеровог круга троугла ABC (то је круг који садржи девет значајних тачака троугла: три средишта страница, три подножја висина и три Ојлерове тачке).

У међувремену су решена још четири проблема са проширене листе чији је статус био непознат [27]. Постоји 140 значајно различитих проблема који се могу описати додавањем нових тачака скупу тачака из Верникове листе. У табели 2.2 дат је тренутни статус проблема из Конелијеве листе: постоји 73 **S** проблема, 11 **U** проблема, 5 **R** проблема, 19 **L** проблема, као и 32 проблема са непознатим статусом⁹.

Постоје и друге варијације Верникове листе, на пример листа проблема у којима је задатак конструисати троугао ако су дата три елемента са наредне листе од 18 елемената: странице a, b, c ; углови α, β, γ ; висине h_a, h_b, h_c ; тежишне линије m_a, m_b, m_c ; бисектрисе углова t_a, t_b, t_c ; полупречник описаног круга троугла R ; полупречник уписаног круга троугла r и полуобим s . Постоји 186 значајно другачијих, нетривијалних проблема и показано је да 3 припадају групи **R**, 128 припадају групи **S**, 27 припадају групи **U**, док је статус осталих проблема још увек непознат [8]. Могу се разматрати и варијанте са задатим додатним елементима (полупречници спољашњих кругова r_a, r_b, r_c и површина троугла S) чиме се добија листа од 22 елемената и укупно 1540 тројки проблема. Луиз Лопез (Luis Lopes) је приказао решења 371 несиметричних проблема овог типа [61], а Фурсенко (V. B. Fursenko) листу од 350 проблема овог типа (неки су решиви, а неки нерешиви) [38, 39]. Постоји још радова на тему систематизације проблема конструкције троугла када су задати неки од горе поменутих елемената и реализације њихових решења у програму C.a.R. [103], као и решења разних конструктивних проблема јавно доступних на интернету [75].

2.3 Аутоматско и интерактивно доказивање теорема

У данашње време издвајају се два главна правца у доказивању теорема уз помоћ рачунара – аутоматско доказивање теорема и интерактивно доказивање

⁹Паскал Шрек и Паскал Матис тренутно раде на одређивању статуса проблема из Конелијеве листе чији је статус до сад био непознат и за већину проблема су показали да су нерешиви, за 3 проблема да су решиви, а за неке за сада нису успели да утврде статус.

1. A, B, E_a S	48. E_a, E_b, M_c S	95. E_a, I, T_b
2. A, B, E_c S	49. E_a, E_b, N L	96. E_a, M_a, M_b L
3. A, B, N S	50. E_a, E_b, O S	97. E_a, M_a, N R
4. A, E_a, E_b S	51. E_a, E_b, T_a	98. E_a, M_a, O S
5. A, E_a, G S	52. E_a, E_b, T_c U	99. E_a, M_a, T_a S [27]
6. A, E_a, H R	53. E_a, G, H S	100. E_a, M_a, T_b
7. A, E_a, H_a L	54. E_a, G, H_a S	101. E_a, M_b, M_c S
8. A, E_a, H_b L	55. E_a, G, H_b S	102. E_a, M_b, N L
9. A, E_a, I S	56. E_a, G, I	103. E_a, M_b, O S
10. A, E_a, M_a S	57. E_a, G, M_a S	104. E_a, M_b, T_a
11. A, E_a, M_b S	58. E_a, G, M_b S	105. E_a, M_b, T_b
12. A, E_a, N S	59. E_a, G, N S	106. E_a, M_b, T_c
13. A, E_a, O S	60. E_a, G, O S	107. E_a, N, O S
14. A, E_a, T_a S	61. E_a, G, T_a	108. E_a, N, T_a S [27]
15. A, E_a, T_b U	62. E_a, G, T_b	109. E_a, N, T_b
16. A, E_b, E_c S	63. E_a, H, H_a L	110. E_a, O, T_a
17. A, E_b, G S	64. E_a, H, H_b L	111. E_a, O, T_b
18. A, E_b, H S	65. E_a, H, I S	112. E_a, T_a, T_b
19. A, E_b, H_a S	66. E_a, H, M_a S	113. E_a, T_b, T_c
20. A, E_b, H_b L	67. E_a, H, M_b S	114. G, H, N R
21. A, E_b, H_c S	68. E_a, H, N S	115. G, H_a, N S
22. A, E_b, I	69. E_a, H, O S	116. G, I, N U
23. A, E_b, M_a S	70. E_a, H, T_a S	117. G, M_a, N S
24. A, E_b, M_b S	71. E_a, H, T_b U	118. G, N, O R
25. A, E_b, M_c S	72. E_a, H_a, H_b S	119. G, N, T_a U
26. A, E_b, N S	73. E_a, H_a, I S	120. H, H_a, N S
27. A, E_b, O S	74. E_a, H_a, M_a L	121. H, I, N U
28. A, E_b, T_a	75. E_a, H_a, M_b S	122. H, M_a, N S
29. A, E_b, T_b	76. E_a, H_a, N L	123. H, N, O R
30. A, E_b, T_c	77. E_a, H_a, O S	124. H, N, T_a U
31. A, G, N S	78. E_a, H_a, T_a L	125. H_a, H_b, N L
32. A, N, N S	79. E_a, H_a, T_b	126. H_a, I, N S
33. A, H_a, N S	80. E_a, H_b, H_c L	127. H_a, M_a, N L
34. A, H_b, N S	81. E_a, H_b, I	128. H_a, M_b, N L
35. A, I, N	82. E_a, H_b, M_a L	129. H_a, N, O S
36. A, M_a, N S	83. E_a, H_b, M_b S	130. H_a, N, T_a S [27]
37. A, M_b, N S	84. E_a, H_b, M_c S	131. H_a, N, T_b
38. A, N, O S	85. E_a, H_b, N L	132. I, M_a, N S
39. A, N, T_a	86. E_a, H_b, O S	133. I, N, O U
40. A, N, T_b	87. E_a, H_b, T_a	134. I, N, T_a
41. E_a, E_b, E_c S	88. E_a, H_b, T_b U	135. M_a, M_b, N L
42. E_a, E_b, G S	89. E_a, H_b, T_c	136. M_a, N, O S
43. E_a, E_b, H S	90. E_a, I, M_a S	137. M_a, N, T_a S [27]
44. E_a, E_b, H_a S	91. E_a, I, M_b	138. M_a, N, T_b
45. E_a, E_b, H_c S	92. E_a, I, N S	139. N, O, T_a U
46. E_a, E_b, I U	93. E_a, I, O	140. N, T_a, T_b
47. E_a, E_b, M_a L	94. E_a, I, T_a	

Табела 2.2: Конелијеви проблеми и њихов статус

теорема. Ова два правца се разликују у приступу и могућностима. Некада је неопходно користити интерактивно доказивање теорема зато што су неки проблеми превише тешки или је језик у ком су формулисани превише изражајан да би се могли решити аутоматским методама. Такође, ова два правца су мотивисана различитим циљевима: први тежи ка применама у образовању или индустрији, када је битније знати да је одређено тврђење тачно, него имати његов доказ, док други има за циљ формирање корпуса верификованог математичког знања.

2.3.1 Аутоматско доказивање теорема

Аутоматско доказивање теорема подразумева доказивање теорема коришћењем рачунара. Аутоматски доказивачи теорема имају примену у верификацији софтвера и хардвера, у комбинаторним проблемима, дизајну помоћу рачунара, проблемима планирања и распоређивања, али се такође често користе и као помоћ кориснику у спровођењу комплексних доказа.

Аутоматски доказивачи теорема су с годинама досегли јако висок ниво и данас се помоћу њих могу доказати или оповргнути нека јако тешка и комплексна тврђења. Постоје доказивачи општег типа, као што су на пример резолуцијски доказивачи [88] и доказивачи за конкретне теорије првог реда. Резолуцијски доказивачи се заснивају на основном принципу резолуције који датира још од 1965. године, али су у међувремену направљена многа унапређења. Неки од најуспешнијих резолуцијских доказивача данас су E [93], Spass [106], Vampire [87], Otter [71], итд.

Не постоји потпуно поверење у доказе или аргументе које дају аутоматски доказивачи теорема јер се без доказа претпоставља коректност њихове имплементације. Да би се повећала њихова поузданост требало би их повезати са неким интерактивним доказивачем теорема (видети поглавље 2.3.2), било да се провери код самог алата или да се покаже исправност генерисаних доказа.

2.3.2 Интерактивно доказивање теорема

Због учених грешака у објављеним доказима, последњих година, у свим областима математике и рачунарства, формални, машински-провериви докази (дати на објектном нивоу – у терминима аксиома и правила извођења) имају све већи значај. Формални докази имају велику улогу у одржавању математичког знања (на пример у дигитализацији математичког наслеђа), у образовању, али и у индустријским применама у којима је критична коректност неких алгоритама или израчунавања. Труд уложен у развој формалних доказа све је већи и многе тешке и значајне теореме су доказане на овај начин. Такође, конструисани су репозиторијуми доказаних теорема, као и многи софтверски алати за генерисање и проверу формалних доказа. *Интерактивни доказивачи теорема* су системи који имплементирају формалну логику и којима се генерисани докази аутоматски проверавају од стране рачунара. Данас најпопуларнији интерактивни доказивачи теорема су Isabelle [78], Coq [32], Mizar [104] и HOL-light [49]. Ниво аутоматизације у интерактивним доказивачима теорема је често врло ограничен, те се овакви докази углавном конструишу ручно и захтевају

искусног корисника; чак и тада њихово генерисање може да траје врло дуго. Такође, није лако искористити већ постојеће делове доказа у новом контексту. Штавише, с обзиром на то да је аутоматизација врло ограничене моћи, формулисање и доказивање исте теореме у незнатно модификованој теорији би у највећем броју случајева захтевало готово исту количину поново утрошеног времена. Све ово има за последицу да, упркос великом успеху у формализацији фрагмената математике и рачунарства, интерактивни доказивачи теорема још увек нису у широкој употреби.

Постоје два различита стила доказа које користе интерактивни доказивачи теорема: процедурални и декларативни. У процедуралним доказима корисник наводи систему шта треба да ради и добијени докази су кориснику најчешће нечитљиви и не одговарају доказима у логици. Isar (Intelligible semiautomated reasoning) приступ читљивим формалним доказима користи декларативни стил записа доказа и настао је са намером да се премости јаз између интерног појма доказа добијеног од интерактивних доказивача теорема и одговарајућег нивоа апстракције корисника [107]. Он представља алтернативу уобичајеним доказним скриптовима јер пружа интерфејс који је кориснику много читљивији. Isabelle/Isar систем има интерпретер за језик формалних доказа и читљиви Isar документи се конвертују и извршавају као низ Isabelle корака закључивања ниског нивоа. Стога Isar дозвољава кориснику да запише доказ на њему погодан начин, али се ти докази аутоматски проверавају од стране система Isabelle за доказивање који лежи у његовој основи.

Аутоматско и интерактивно доказивање теорема, иако настали са различитим мотивацијама, могу имати и заједничке циљеве: било би корисно, како за растући корпус верификованог математичког знања тако и у едукативне сврхе, имати формалне, машински провериве доказе који су аутоматски генерисани, што је могуће ефикасније, и то на традиционалан начин.

2.3.3 Аутоматско и интерактивно доказивање теорема у геометрији

Аутоматско доказивање теорема датира још од 1959. године када је Херберт Гелернтер (Herbert Gelernter) развио геометријски доказивач теорема који је могао да реши велики број проблема из средњошколских уџбеника равanske геометрије [43, 21]. Данас постоје три основне групе метода у аутоматском доказивању теорема у геометрији: *алгебарске*, *синтетичке* и *хибридне методе*. Алгебарске методе су ефикасне, али добијени докази не осликавају геометријску природу проблема. Синтетичке методе покушавају да опонашају

традиционалне доказе, те су докази које генеришу системи засновани на синтетичким методама читљивији, али су мање ефикасни од алгебарских. Хибридне методе укључују обимна израчунавања над различитим врстама геометријских величина, нешто су нешто мање ефикасне од алгебарских метода и њима се генеришу докази који могу бити читљиви, али се генерисани докази значајно разликују од традиционалних доказа.

Највећи успех у аутоматском доказивању теорема у геометрији имају алгебарске методе као што су *Вуова метода* [109] и *метода Гребнерових база* [15, 57]. Основна идеја алгебарских метода састоји се у томе да се геометријска својства трансформишу у једначине над координатама одговарајућих тачака и да се онда коришћењем алгебарских трансформација покаже да важе дата твђења. Предност ових метода је што се докази генеришу јако ефикасно. На пример доказивачи теорема за еуклидску геометрију који су засновани на Вуовој методи успевају да аутоматски докажу веома комплексне теореме (на пример многе задатке са математичких олимпијада) и ова метода се најчешће сматра најефикаснијом методом за аутоматско доказивање теорема. Недостатак ових метода је то што докази који се генеришу нису читљиви нити су налик традиционалним доказима.

Нешто мањи успех у доказивању теорема у геометрији имају *метода површина* [22] и *метода пуних углова* [23]. Основна идеја ових метода је да се претпоставка теореме формулише коришћењем скупа конструктивних корака, при чему сваки од њих уводи нову тачку или праву, а закључак теореме као једнакост израза над неким геометријским величинама (као што је означена површина троугла), без реферисања на декартовске координате, те се сматрају хибридним методама. Доказ се затим добија тако што се у обрнутом редоследу елиминишу конструисане тачке, односно праве. Након свих елиминација требало би да се добије тривијална једнакост неких израза који укључују само независне тачке. Овим методама се генеришу докази који често јесу читљиви, али су формулисани у терминима израза који укључују одређене геометријске величине, па се значајно разликују од традиционалних доказа који се могу наћи у уџбеницима. Читљивост ових доказа може бити додатно нарушена тиме што у доказима могу учествовати јако комплексни изрази које је тешко разбити на интуитивне изразе са јасним геометријским значењем.

Арт Кваифе (Art Quaiфе) је користио резолуцијски доказивач да докаже теореме геометрије Тарског [85]. Он је успео да докаже неке од тежих теорема, али нису добијени нити читљиви нити формални докази.

У последње време пуно труда је уложено у формализацију геометрије. Кри-

стоф Делинцер (Christophe Dehlinger), Жан Франсоа Дифур (Jean-Francois Dufourd) и Паскал Шрек су формализовали прве две групе Хилбертове аксиоматике у Coq -у у оквиру интуиционистичке логике и дошли до закључка да се многе значајне теореме не могу овако доказати [31]. Лора Мејкл (Laura Meikle) и Жак Флоријо (Jacques Fleuriot) су формализовали прве три групе Хилбертове аксиоматике у Isabelle/Isar-у [72]. Они су, насупрот општем мишљењу да су Хилбертови докази ригорозни, показали да су се неки од Хилбертових доказа ослањали на имплицитне претпоставке које нису биле јасно дате. На овај начин још једном је истакнута потреба за формално верификованим доказима теорема. Жилијен Нарбу (Julien Narboux) је формализовао у Coq -у [76, 14] првих дванаест поглавља књиге Тарског [94] – формално је извео многе геометријске особине, различите верзије Пашове аксиоме, особине релација *између* и *конгруентно*. Дошао је до закључка да је геометрија Тарског јако погодна за механизацију због своје једноставности и мањег броја дегенерисаних случајева. Сана Стојановић Ђурђевић, Жилијен Нарбу и Предраг Јаничић су аутоматски генерисали формалне доказе за велики проценат теорема књиге Тарског, коришћењем резолуцијских доказивача теорема и доказивача теорема за кохерентну логику.

Постоје и друге формализације геометрије развијене у Coq -у: Жил Кан (Gilles Kahn) је формализовао велики део фон Платоове конструктивне геометрије [56], Фредерик Гиљо (Frederic Guilhot) је формализовао велики део средњошколске геометрије [46], Жан Дупра (Jean Duprat) је формализовао аксиоматски систем за геометрију лењира и шестара [35], формализацију пројективне геометрије спровели су Николас Маго (Nicolas Magaud), Жилијен Нарбу и Паскал Шрек [64, 65], итд. Филип Марић и Данијела Петровић су у Isabelle/HOL-у формализовали геометрију комплексне равни. Све поменуте формализације урађене су у потпуности ручно.

2.4 Кохерентна логика и аутоматско доказивање теорема у геометрији

Кохерентна логика (CL), позната и као геометријска логика, и одговарајућа процедура доказивања првобитно је дефинисана од стране Торалфа Сколема (Thoralf Skolem), а у последње време привлачи пажњу због могућности аутоматизације [9, 10, 37]. Она представља погодно окружење за генерисање читљивих и формалних доказа јер има конструктивистички систем доказивања заснован на техници познатој као уланчавање унапред, што омогућава једноставно гене-

писање доказних објеката [9].

У CL је дозвољена ограничена употреба егзистенцијалног квантификатора, те се може сматрати проширењем логике резолуције. Насупрот методи резолуције, у кохерентној логици се тврђење које се доказује не мења (не користе се побијање, Сколемизација нити трансформисање у клаузалну форму) већ се само тврђење директно доказује. Ово је посебно важно у ситуацији када нас не интересује само истинитосна вредност неког тврђења већ и његов доказ. Докази у CL су природни и интуитивни и резонување је конструктивистичко (интуиционистичко).

Формално гледано, CL представља фрагмент логике првог реда који се састоји из формула облика:

$$A_1(\vec{x}) \wedge \dots \wedge A_n(\vec{x}) \Rightarrow \exists \vec{y}_1 \mathcal{B}_1(\vec{x}, \vec{y}_1) \vee \dots \vee \exists \vec{y}_m \mathcal{B}_m(\vec{x}, \vec{y}_m)$$

које су имплицитно универзално квантификоване и за које важи:

- $n \geq 0, m \geq 0$,
- $\vec{x}$ представља низ променљивих $x_1, x_2, \dots, x_k$, $A_i, 1 \leq i \leq n$ представљају атомичке формуле (које укључују неке од променљивих из $\vec{x}$),
- $\vec{y}_j, 1 \leq j \leq m$ представљају низ променљивих $y_1^j, y_2^j, \dots, y_{k_j}^j$
- $\mathcal{B}_j, 1 \leq j \leq m$ означавају конјункције атомичких формула (које у себи укључују неке од променљивих из $\vec{x}$ и $\vec{y}_j$)¹⁰.

У CL није дозвољена употреба функцијских симбола арности веће од 0. Нова константа, која се не јавља ни у до тада коришћеним аксиомама нити у тврђењу које доказујемо, назива се *сведоком*. Под термином константа подразумевају се и константе које су део сигнатуре и сведоци. Сви термови су или константе или променљиве. Атомичка формула је или $\perp$ или $p(t_1, t_2, \dots, t_n)$, где је p предикатски симбол арности n , а $t_i, 1 \leq i \leq n$ су термови. Атомичка формула над константама зове се *чињеница*.

У CL је дозвољено коришћење једино наредних правила извођења (у основи то је подскуп Генценових правила за природну дедукцију, и такође варијанта

¹⁰У даљем тексту ћемо увек са $\mathcal{B}_j$ означавати формулу која представља конјункцију атомичких формула.

правила датих у [10]¹¹):

$$\frac{A_1(\vec{a}) \wedge \dots \wedge A_n(\vec{a})}{A_1(\vec{a}), \dots, A_n(\vec{a})} \wedge E \quad \frac{A_1 \vee \dots \vee A_n \quad \frac{[A_1] \quad \vdots \quad c_1}{\vec{B}} \quad \dots \quad \frac{[A_n] \quad \vdots \quad c_n}{\vec{B}}}{B} \vee E \quad \frac{\perp}{A} efq$$

$$\frac{A_1(\vec{a}), \dots, A_n(\vec{a}) \quad A_1(\vec{x}) \wedge \dots \wedge A_n(\vec{x}) \Rightarrow \exists \vec{y}_1 \mathcal{B}_1(\vec{x}, \vec{y}_1) \vee \dots \vee \exists \vec{y}_m \mathcal{B}_m(\vec{x}, \vec{y}_m)}{\mathcal{B}_1(\vec{a}, \vec{w}_1) \vee \dots \vee \mathcal{B}_m(\vec{a}, \vec{w}_m)} ax$$

где:

- $\vec{a}$ означава вектор константи,
- $\vec{w}_j, 1 \leq j \leq m$ векторе сведока (тј. новоуведене константе),
- $c_i, 1 \leq i \leq n$ низ правила извођења.

Правила $(\wedge E)$, $(\vee E)$ и (efq) означавају стандардна правила за елиминацију конјункције, елиминацију дисјункције и правило *ex falso quodlibet*. Приликом примене правила $(\wedge E)$ изводи се $A_i(a)$ за свако i , $1 \leq i \leq n$. Правило (ax) примењује се само ако већ не постоје вектори $\vec{w}_j$ константи тако да важи: $\mathcal{B}_1(\vec{a}, \vec{w}_1) \vee \mathcal{B}_2(\vec{a}, \vec{w}_2) \vee \dots \vee \mathcal{B}_m(\vec{a}, \vec{w}_m)$. Овим се у некој мери смањује простор претраге.

Формула $A_1(\vec{x}) \wedge \dots \wedge A_n(\vec{x}) \Rightarrow \exists \vec{y}_1 \mathcal{B}_1(\vec{x}, \vec{y}_1) \vee \dots \vee \exists \vec{y}_m \mathcal{B}_m(\vec{x}, \vec{y}_m)$ је *теорема у кохерентној логици* ако се из премиса $A_1(\vec{a}), A_2(\vec{a}), \dots, A_n(\vec{a})$ (при чему $\vec{a}$ означава низ новоуведених константи) могу извести сви конјункти формуле $\mathcal{B}_j(\vec{a}, \vec{w})$ за неко $j, 1 \leq j \leq m$ и за неки вектор константи $\vec{w}$.

Постоји процедура доказивања заснована на претрази у ширину која је сагласна и потпуна: кохерентна формула F може се доказати ако и само ако је формула F тачна у свим моделима Тарског (са непразним доменима) скупа аксиома и чињеница $A_1(\vec{a}), \dots, A_n(\vec{a})$ [9].

Значај кохерентне логике лежи у томе што се велики број теорија и теорема може једноставно и директно формулисати у CL. С друге стране, може се доказати да се свака формула логике првог реда може трансформисати у CL, тако да је очувана њена задовољивост [83], и онда решити коришћењем доказивача за CL. Ова трансформација није увек конструктивистичка, већ може користити

¹¹Доказивач који користи само ова правила може бити сагласан, доказивач који додатно користи технику познату као итеративно продубљивање може бити потпун.

и неке кораке који укључују класичну логику. Последица овог превођења је да је CL неодлучива, али полуодлучива.

Према дефиницији кохерентне логики употреба негација није дозвољена. Појединачна чињеница $\neg A$ може се представити у форми $A \rightarrow \perp$, али ова репрезентација није применљива у општем случају. Да би ипак могло да се резонује о негираним чињеницама, за сваки предикатски симбол R обично се уводи нови симбол $\bar{R}$ који представља $\neg R$ и користе наредне аксиоме [83]: $R(\vec{x}) \wedge \bar{R}(\vec{x}) \rightarrow \perp$ и $R(\vec{x}) \vee \bar{R}(\vec{x})$.

Последњих година Марк Безем (Marc Bezem) и његови сарадници поново су оживели интересовање за кохерентну логику. Марк Безем и Тери Коқан (Thierry Coquand) развили су у PROLOG-у доказивач CL, заснован на процедури претраге у ширину, који генерише доказне објекте у Coq-у [9]. Штефан Бергхофер (Stefan Berghofer) и Марк Безем су у ML-у развили интерни доказивач за кохерентну логику у Isabelle-у. Ханс Де Нивел (Hans De Nivelle) је имплементирао доказивач теорема за логику блиску кохерентној логици који у себи има механизам за учење лема [30]. Први аутоматски доказивач теорема за кохерентну логику развијен је у PROLOG-у од стране Предрага Јаничића и Стевана Кордића [55] и коришћен је за решавање теорема из области еуклидске геометрије. Он је користио фиксирани скуп аксиома, сличан Борсуковом систему [11]. Његово проширење, али и значајно побољшање представља доказивач ArgoCLP који ће бити описан у наредном поглављу.

2.5 Summary of Chapter Background

In this section firstly the short overview of different axiomatizations of geometry is provided – from Euclid and his book “Elements” [50], through Hilbert’s seminal book “Foundations of Geometry” [51], Tarski’s simple axiomatization which uses only points and two predicate symbols [94, 101], up to the modern axiomatization, such as the work of Avigad and his coauthors [5].

A short remainder on geometric construction problems is given. The oldest and the most studied class of construction problems is the one where the task is to construct a figure using only ruler and compass [33, 70, 61]. Traditionally, a solution of construction problem goes in four phases: analysis, construction, proof and discussion [33]. Although solving construction problems has a long history, automation of generating their solutions has not been studied a lot [41, 44, 47, 89].

Two corpora of triangle location construction problems are presented: Wernick’s [108] and Connelly’s corpora [25]. In these problems a task is to construct a

triangle ABC if locations of three significant points are given. For the most of the problems from these two corpora the status is determined, manually or using algebraic methods [74, 95, 105, 92].

Two main directions in proving theorems using computers are presented – these are: automated theorem proving and interactive theorem proving. Automated theorem provers have reached very high level and some very complex theorems have been proved this way. However, the proofs or arguments they generate are not fully trusted. On the other hand, interactive theorem proving is used for generation of formal, machine verifiable proofs, using interactive theorem provers like Isabelle [78]. Concerning geometry, the most successful automated theorem provers are based on algebraic methods, such as Wu’s method [109] and Gröbner basis method [15, 57]. Also, much work has been put into formalization of different types of geometry and many geometry theorems has been formally proved [31, 72, 76, 14]. Most of these formalizations have been developed manually.

At the end of this chapter, one fragment of first order logic, called coherent logic is presented [9, 10, 37]. It presents suitable framework for generation readable and formal proofs, since it has a constructive proof system based on a forward chaining mechanism. A large number of theories and theorems can be easily formulated within coherent logic.

3

Доказивач ArgoCLP и проширења

Кохерентна логика, због свог природног, једноставног и конструктивистичког система доказивања, представља погодно окружење за развој алата којима се генеришу читљиви и формални докази. Један такав алат је доказивач ArgoCLP.

3.1 Опис доказивача ArgoCLP

ArgoCLP је генерички доказивач теорема за кохерентну логику који производи и читљиве и формалне, машински-провериве доказе у разним теоријама [98]. Ови докази могу се користити у образовне сврхе, а могу да представљају и допринос корпусу формализоване математике. Захваљујући аутоматизацији, доказивач ArgoCLP може бити користан алат за изградњу овог корпуса. Текућа верзија доказивача нема за циљ доказивање свих тешких теорема који се јављају у математичким уџбеницима, већ доказивање основних теорема умерене тежине, али на читљив, лако разумљив начин. Једна од могућих примена доказивача је и утврђивање да ли модификација неке аксиоме мења снагу аксиоматског система. Ово је важно питање јер постоји пуно аксиоматских система, који се понекад врло мало међусобно разликују, и утврђивање њиховог међусобног односа може бити веома захтевно. Додатно, доказивач ArgoCLP може се користити за доказивање погодно издвојених потциљева комплексних тврђења (као што је то био случај са доказивачем CL приликом доказивања Хесенбергове теореме [10]).

3.1.1 ArgoCLP процедуре доказивања

ArgoCLP је генерички доказивач теорема за кохерентну логику, те се може применити на произвољан скуп кохерентних аксиома и тврђење које је у кохерентној форми. У оквиру доказивача могу се употребљавати типови (сортe) или се, алтернативно, у те сврхе могу користити унарни предикати. У оквиру *сигнатуре* теорије над којом се ради неопходно је задати скуп свих типова теорије, скуп примитивних предикатских симбола, а за сваки од предикатских симбола задаје се његова арност и тип сваког од аргумената. Резоновање унутар доказивача је интуиционистичко. Као што је већ напоменуто у поглављу 2.4, у кохерентној логици није дозвољено користити негацију. Међутим, имајући у виду да докази многих тврђења захтевају примену инстанци аксиоматске схеме *tertium non datur* [31], за сваки предикатски симбол R обично се уводи предикатски симбол $\bar{R}$, који има улогу његове негације, као и одговарајуће аксиоме, наведене у поглављу 2.4. И са додатком ових аксиома резоновање остаје унутар интуиционистичке поставке.

Основна процедура доказивања.

Алтернативу процедуре претраге у ширину, поменутој у поглављу 2.4, представља једноставна процедура доказивања која се заснива на *улањавању унапред* и *итеративном продубљивању*. Аксиоме се примењују по принципу водопада: када се нека аксиома успешно примени, потрага за новом применљивом аксиомом креће испочетка, почев од прве аксиоме. Све константе се нумеришу и то на следећи начин: све универзално квантификоване променљиве из премиса тврђења које се доказује се инстанцирају константама почев од 1 до њиховог броја, а затим се свака следећа константа, која се конструише приликом доказивања, нумерише првом наредном вредношћу. Постоји наменски бројач s , кога ћемо звати *граничник*, који контролише примену аксиома према следећем принципу: аксиома се може применити само ако су све њене (универзално квантификоване) променљиве упарене са константама чији је редни број мањи од s и притом су до тад изведене све чињенице из премиса аксиоме, а не постоји вектор сведока тако да важи закључак аксиоме (у складу са (ax) правилом закључивања датим у поглављу 2.4). Расположиве дефиниције користе се као да су аксиоме. На почетку се вредност граничника иницијализује на број константи који се јавља у премисама тврђења које се доказује. У случају да се не може применити ниједна аксиома, вредност граничника се инкрементира и испочетка се врши провера да ли је могуће применити неку од аксиома.

$$\begin{array}{c}
\frac{A(a), B(b) \vdash \exists z.F(a, z)}{A(a) \rightarrow C(a) \text{ (A1)}} \\
\downarrow \\
C(a) \wedge B(b) \rightarrow \exists c.G(a, c) \text{ (A3)} \\
\downarrow \\
A(a) \wedge B(b) \rightarrow \exists d.I(a, b, d) \text{ (A4)} \\
\downarrow \\
G(a, c) \rightarrow F(a, c) \text{ (A2)}
\end{array}$$

Слика 3.1: Пример процедуре доказивања

Пример 3.1. Нека је дато тврђење $T : A(x) \wedge B(y) \rightarrow \exists z.F(x, z)$ и аксиоматски систем $A1 : A(x) \rightarrow C(x)$, $A2 : G(x, y) \rightarrow F(x, y)$, $A3 : C(x) \wedge B(y) \rightarrow \exists z.G(x, z)$, $A4 : A(x) \wedge B(y) \rightarrow \exists z.I(x, y, z)$. Вредност граничника s иницијализује се на број променљивих у премисама тврђења, а то је 2. Променљиве које се јављају у премисама (x и y) инстанцирају се редом константама a и b и нумеришу редом константама 1 и 2. Затим се покушава примена свих могућих аксиома, тако да су све универзално квантификоване променљиве у аксиоми нумерисане вредношћу мањом или једнаком 2. На тај начин успешно се примењују аксиома $A1$ на константу a и добија формула $A(a) \rightarrow C(a)$, аксиома $A3$ на константе a и b , чиме се уводи нова константа, на пример c , која се притом нумерише бројем 3 и добија формула $C(a) \wedge B(b) \rightarrow \exists c.G(a, c)$. Након тога се примењује и аксиома $A4$ на константе a и b , чиме се уводи још једна константа, на пример d , која се нумерише бројем 4, и добија се формула $A(a) \wedge B(b) \rightarrow \exists d.I(a, b, d)$. С обзиром на то да тврђење још увек није доказано, а примењене су све могуће аксиоме за дату вредност граничника, вредност граничника се инкрементира на 3, чиме је могуће променљиве из премиса аксиома инстанцирати и новоуведеном променљивом c . Сада је могућа примена и аксиоме $A2$ на константе a и c чиме се изводи формула $G(a, c) \rightarrow F(a, c)$, а из ње и закључак тврђења које се доказује $F(a, c)$. Доказ овог тврђења приказан је на слици 3.1.

Овим механизмом се постиже да се прво користе раније уведене константе коришћењем свих могућих аксиома, па тек онда оне које су касније уведене. На овај начин се избегава незауостављање приликом узастопне примене једне исте аксиоме сваки пут на новоконструисани објекат (на пример када се стално конструише средиште дужи са датим крајевима, где се за један крај дужи узима тачка конструисана у претходном кораку). Ако се више ниједна аксиома не може применити, а граничник s је достигао максималну вредност, и притом тврђење није доказано, то значи да тврђење није теорема у датој теорији (може

се, такође, десити да се за тврђење које није теорема у кохерентној логици процедура доказивања не зауставља). Резоновање у оквиру доказивача је базно, тј. формуле које се изводе су увек потпуно инстанциране.

Може се доказати (на сличан начин као што је доказано за процедуру претраге у ширину [9]) да је ова процедура доказивања сагласна и потпуна, тј. важи да се кохерентна формула F може доказати ако и само ако је формула F тачна у свим моделима Тарског (са непразним доменима) скупа аксиома и скупа чињеница $A_1(a), \dots, A_n(a)$. Може се доказати и да је процедура доказивања сагласна и потпуна у односу на систем извођења приказан у поглављу 2.4, тј. формула F може се доказати ако и само је теорема у кохерентној логици. И са додатком аксиома $R(x) \vee \bar{R}(x)$ не нарушава се потпуност процедуре доказивања. Међутим, и поред особине потпуности, приказаном процедуром је доказивање многих тврђења у неким теоријама практично немогуће (коришћењем ограничене меморије и ресурса).

Унапређење процедуре доказивања техникама које чувају потпуност

Ефикасност основне процедуре доказивања може се побољшати на велики број начина, а да се при том очува потпуност. Сва побољшања која ће бити наведена имају за циљ да на неки начин смање простор претраге и да умање ефекат комбинаторне експлозије (изазван извођењем многих чињеница које нису релевантне за доказ). Чак и коришћењем свих ових техника, многе тешке теореме не могу се доказати у разумном времену. Такође, генерисани докази могу да садрже многе ирелевантне кораке.

Поредак аксиома. Аксиоме су груписане у наредне групе (подразумева се да је $n \geq 0$ ($n > 0$ за трећу и четврту групу), $m > 1$ и да свака група аксиома искључује претходне групе као своје специјалне случајеве):

- *непродуктивне негранајуће аксиоме*
аксиоме облика: $A_1(x) \wedge \dots \wedge A_n(x) \Rightarrow \mathcal{B}(x)$
- *непродуктивне гранајуће аксиоме*
аксиоме облика: $A_1(x) \wedge \dots \wedge A_n(x) \Rightarrow \mathcal{B}_1(x) \vee \dots \vee \mathcal{B}_m(x)$
- *продуктивне негранајуће аксиоме*
аксиоме облика: $A_1(x) \wedge \dots \wedge A_n(x) \Rightarrow \exists y. \mathcal{B}(x, y)$
- *продуктивне гранајуће аксиоме*
аксиоме облика: $A_1(x) \wedge \dots \wedge A_n(x) \Rightarrow \exists y_1. \mathcal{B}_1(x, y_1) \vee \dots \vee \exists y_m. \mathcal{B}_m(x, y_m)$

- *јако продуктивне негранајуће аксиоме*
аксиоме облика: $\exists y. \mathcal{B}(y)$
- *јако продуктивне гранајуће аксиоме*
аксиоме облика: $\exists y_1 \mathcal{B}_1(y_1) \vee \dots \vee \exists y_m \mathcal{B}_m(y_m)$

Аксиомама се може аутоматски доделити група и он се користи у процесу до-казивања са приоритетима датим групама према горњем уређењу (прва група има максимални приоритет, а последња минимални). Циљ овог уређења је да се увођење нових константи приликом доказивања одложи што је могуће више, да би се смањио број непотребно уведених константи. У истом циљу прво се примењују негранајуће, па тек онда гранајуће аксиоме. Аксиоме унутар једне групе немају неко строго наметнуто уређење (иако њихов поредак унутар групе може значајно да утиче на ефикасност процедуре).

Рано одсецање. Приликом испитивања да ли је нека аксиома применљива није неопходно да се инстанцирају све њене променљиве па тек онда провери да ли су већ изведене све релевантне чињенице. Уместо тога, провера релевантних чињеница врши се што је раније могуће, да би се омогућило рано одбацивање неких инстанци аксиома и смањио простор претраге. На пример, приликом примене аксиоме:

$$\forall x : line \forall y : line \forall X : point \forall Y : point \\ (incident(X, x) \wedge incident(Y, x) \wedge incident(X, y) \wedge incident(Y, y) \wedge X = Y \\ \Rightarrow x = y)$$

уместо да се променљиве x, y, X и Y упаре са свим могућим константама, x и X ће бити прво унификовано са допустивим константама и ако инстанцирана чињеница $incident(X, x)$ није до тада изведена, процедура ће се вратити уназад.

Разбијање аксиома које уводе више сведока. Аксиома облика:

$$\forall x : line \exists X : point \exists Y : point (incident(X, x) \wedge incident(Y, x) \wedge X \neq Y)$$

неће бити примењена на конкретну праву a (којом се инстанцира променљива x) ако већ постоје константе $A : point$ и $B : point$ тако да важи $incident(A, a), incident(B, a)$ и $A \neq B$. Ипак, из разлога ефикасности, корисно је не примењивати горњу аксиому чак и у ситуацији када постоји константа $A : point$ за коју важи $incident(A, a)$, а не постоји константа $B : point$ тако да важи $incident(B, a)$ и $A \neq B$, јер би се тиме увеле две нове константе C и D . Уместо тога, требало би користити наредну варијанту горње аксиоме: $\forall x : line \forall X : point (incident(X, x) \Rightarrow \exists Y : point (incident(Y, x) \wedge X \neq Y))$.

Стога би се, уместо једне аксиоме, користиле две аксиоме, при чему би оригиналној аксиоми био дат мањи приоритет.

Исти механизам може се применити на све аксиоме које у себи садрже више од једног егзистенцијалног квантификатора. Разбијање таквих аксиома на више варијанти није увек једноставно као у горњем примеру. На пример, аксиому:

$$\exists X : point \exists Y : point \exists Z : point \exists U : point \text{noncoplanar}(X, Y, Z, U)$$

требало би разбити на четири различите варијанте, при чему би једна била:

$$\forall X : point \forall Y : point \forall Z : point \exists U : point \text{noncoplanar}(X, Y, Z, U).$$

Ипак, ово тврђење није тачно у еуклидској геометрији и потребна је додатна премиса $\text{noncolinear}(X, Y, Z)$. Према томе, ако аксиома може бити разбијена на више различитих аксиома, свака од нових аксиома мора се доказати (такође коришћењем доказивача за кохерентну логику и коришћењем полазне аксиоме) пре него што се буде користила. Ако се нека аксиома не може доказати (у оквиру неког временског периода) од корисника се може затражити да је модификује. Треба приметити да додатне аксиоме, уведене на овај начин, мењају оригинални аксиоматски систем, али с обзиром на то да је нови аксиоматски систем еквивалентан оригиналном (јер се свака од његових аксиома може доказати као теорема у другом систему, и обрнуто) ова модификација је легитимна (нове аксиоме могу се разматрати и као леме).

Једнакостно резонување. Код теорија које укључују једнакост аксиоме једнакости се не користе експлицитно. Уместо тога, одржавају се класе еквиваленција једнакости константи. Овај механизам омогућава да се, уместо над свим објектима, све операције извршавају само над канонским представницима класа. На почетку процеса доказивања сваки објекат представља своју класу еквиваленције, а онда се те класе одржавају коришћењем Тарџанове (Robert Tarjan) *union-find* структуре података [100]. На пример, ако постоје константе $A : point$, $p : line$, $q : line$ и $\alpha : plane$ тако да важи $\text{incident}(A, p)$, $\text{incident}(q, \alpha)$ и $p = q$, онда се може применити аксиома:

$$\forall X : point \forall x : line \forall \chi : plane (\text{incident}(X, x) \wedge \text{incident}(x, \chi) \Rightarrow \text{incident}(X, \chi))$$

за $X = A, x = p, \chi = \alpha$ и извести чињеница $\text{incident}(A, \alpha)$. Наиме, приликом провере да ли важи $\text{incident}(p, \alpha)$ прво се одређују канонски представници објеката p и α , на пример q и α и с обзиром на то да важи $\text{incident}(q, \alpha)$, аксиома се може применити.

Иако се аксиоме једнакости не користе експлицитно у току поступка доказивања, оне се користе приликом изградње *трага доказа*¹ (који садржи све

¹Овај појам одговара енглеском појму *proof-trace*, који би се још могао превести и као

потребне информације о примењеним корацима током доказивања у неком интерном формату) из кога се даље може конструисати (машински проверив) доказни објекат.

Обрада симетричних предикатских симбола. За предикат R каже се да је симетричан (на позицијама i и j) ако је формула: $R(x_1, \dots, x_i, \dots, x_j, \dots, x_n) \Leftrightarrow R(x_1, \dots, x_j, \dots, x_i, \dots, x_n)$ тачна, при чему се подразумева да је формула универзално квантификована. Код симетричних предикатских симбола могу се разматрати само канонски представници. На пример, уместо да се чува и чињеница $colinear(A, B, C)$ и $colinear(C, B, A)$, довољно је чувати само чињеницу $colinear(A, B, C)$. Представник класе чињеница може се одредити на следећи начин: коришћењем уређења константи (које је одређено њиховом нумерацијом), сортирају се аргументи на симетричним позицијама и минимални елемент бира за представника. Овај корак треба применити увек када је потребно проверити да ли важи нека чињеница над симетричним предикатом. Дати механизам може се користити у конјункцији са механизмом класа еквиваленција у односу на једнакост да би се додатно смањио број чињеница који се чува. Као и аксиоме једнакости, тврђења која обезбеђују да је предикат симетричан не користе се током процеса претраге, већ се користе приликом изградње трага доказа из кога се даље може конструисати (машински проверив) доказни објекат. На пример, ако постоје константе $A : point$, $B : point$, $C : point$ и $D : point$ са уређењем $A < B < C < D$ и изведеним чињеницама $noncolinear(C, B, D)$ и $colinear(A, D, C)$, ако је изведена чињеница $A = B$ класе еквиваленција ова два објекта биће спојене у једну класу и биће изведена контрадикција. Наиме, ако је A представник класе која садржи A и B , коришћењем класе еквиваленције, представник чињенице $noncolinear(C, B, D)$ је $noncolinear(C, A, D)$, а према својствима симетрије његов представник је $noncolinear(A, C, D)$. Аналогно, представник чињенице $colinear(A, D, C)$ је $colinear(A, C, D)$, те се из чињеница $noncolinear(A, C, D)$ и $colinear(A, C, D)$ може извести контрадикција.

Провера да ли је неки предикат симетричан може се извршити аутоматски (у фази претпроцесирања) тако што се генеришу сва релевантна тврђења а затим се покушавају да докажу. Уместо да се за све пермутације симетричних аргумената доказују тврђења, довољно је доказати тврђења за генераторе групе пермутација. На пример, када се покушава доказати да је предикат $coplanar$ симетричан у односу на све четири позиције аргумената довољно је доказати

костур доказа

тврђења само за генераторе групе пермутација (универзално квантификовање се подразумева):

$$\text{coplanar}(x_1, x_2, x_3, x_4) \Leftrightarrow \text{coplanar}(x_2, x_3, x_4, x_1)$$

$$\text{coplanar}(x_1, x_2, x_3, x_4) \Leftrightarrow \text{coplanar}(x_2, x_1, x_3, x_4)$$

Поновно коришћење доказаних теорема. Доказане теореме о симетричности предикатских симбола (заједно са њиховим доказима) могу се користити у ширем доказу. Такође, и друге теореме које су доказане у оквиру система могу се користити у даљем процесу доказивања као леме.

Унапређење процедуре доказивања техникама које не чувају потпуност

Да би се унапредила ефикасност доказивача, макар за нека тврђења, могу се користити и технике које не чувају потпуност, као што је, на пример, ограничење скупа аксиома које се могу користити приликом доказивања. Ако се на овај начин успешно докаже нека нова теорема, то што потпуност у општем случају није очувана ни на који начин не смета.

Ограничење гранајућих аксиома. Гранајуће аксиоме облика $R(x) \vee \overline{R}(x)$ генеришу се и користе само за примитивне (а не и за дефинисане) предикате. Штавише, може се доказати да у неким теоријама изостављање гранајућих аксиома за неке од дефинисаних предикатских симбола не нарушава потпуност система.

Ограничење аксиома. Приликом процеса доказивања, користе се само аксиоме које садрже искључиво предикате који се јављају у тврђењу које се доказује. Друга, мало слабија варијанта овог ограничења била би да се у процесу доказивања користе само аксиоме које садрже барем један предикат који се јавља у тврђењу.

3.1.2 Имплементација и коришћење система ArgoCLP

Доказивач ArgoCLP имплементиран је у програмском језику C++. Имплементација садржи око 5000 линија кода и организован је у 23 класе². У наредном поглављу биће описане главне класе, као и специфичности имплементације система.

²Извршни код доказивача доступан је на адреси: <http://argo.matf.bg.ac.rs/?content=downloads>.

Преглед имплементације

Апстрактна класа **Expression** представља базну класу за променљиве и атомичке формуле. Из класе **Expression** изводе се две класе:

- класа **Variable**, којом се представља променљива (на пример типа тачка),
- класа **Relation**, којом се задаје неко својство (на пример да одређена тачка припада одређеној правој).

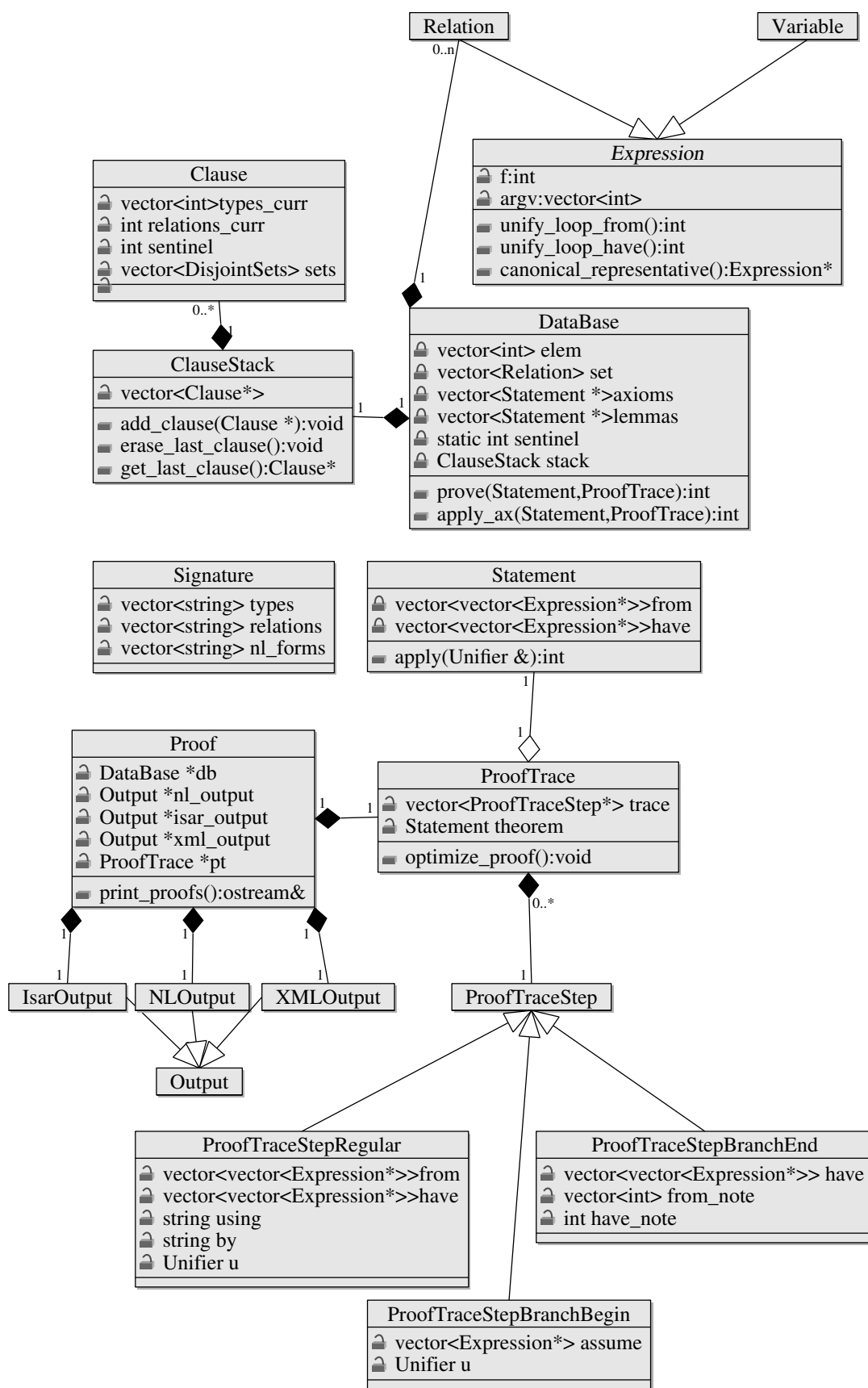
Тврђење које се доказује, као и аксиоме и леме које се користе приликом доказивања, представљају се инстанцом класе **Statement**. Тврђење се задаје скуповима премиса и закључака, као векторима вектора објеката типа **Expression** (чиме се задаје дисјункција конјункција атомичких формула), и идентификатором тврђења.

У случају примене гранајуће аксиоме потребно је пре уласка у прву грану запамтити текуће стање базе знања, на које се треба вратити на крају обраде сваке од грана. То се постиже тиме што се чува стек клауза (који се представља класом **ClauseStack**). Свака клауза садржи:

- редни број дисјункта до којег се стигло,
- тренутну вредност граничника,
- број изведених објеката сваког од типова,
- списак изведених релација.

За чување трага доказа користи се класа **ProofTrace**. Ова класа садржи и информације које су од значаја у процесу поједностављивања доказа. Траг доказа се састоји из вектора корака. Базна класа за све врсте корака је класа **ProofTraceStep**, а из ње се изводе класе којима се представља регуларан корак (**ProofTraceStepRegular**), корак почетка гранања (**ProofTraceStepBranchBegin**), корак краја гранања (**ProofTraceStepBranchEnd**) итд. Класа **Proof** интегрише базу знања, траг доказа и излазе у различите формате.

Приликом потраге за доказом, кључне методе су рекурзивне методе **unify_loop_from** и **unify_loop_have** којима се врши провера да ли је нека аксиома применљива на текуће стање базе знања. Њима се обрађује један по један израз из аксиоме, и у зависности од тога да ли је у питању променљива или релација и да ли се стигло до премиса или закључка аксиоме има дефинисано различито понашање.



Слика 3.2: Дијаграм најбитнијих класа доказивача ArgoCLP

Класа `Output` је базна класа за све подржане врсте излаза:

- излаз у природан језик (задат класом `NLOutput`, која садржи методе за испис тврђења које се доказује, као и комплетног доказа у природно-језичком формату),
- излаз у Isabelle/Isar формат (задат класом `IsarOutput`, која садржи методе за испис Isabelle доказа који укључује методе за штампање сигнатуре, скупа аксиома у одговарајућем формату, тврђења теореме која се доказује, као и самог доказа)
- излаз у XML формат (задат класом `XMLOutput`).

На слици [3.2](#) приказан је однос између најбитнијих класа у систему.

Улаз

Постоје два начина на које је могуће задати улаз у доказивач *ArgoCLP*. Један начин је да се и сигнатура и скуп аксиома увезу у програм коришћењем датотека у интерном формату. Овим се омогућава да се доказивач може користити за различите кохерентне теорије. У овом случају доказивачу је као улаз потребно обезбедити:

- сигнатуру теорије – имена типова наводе се након кључне речи `types`, на пример:

```
types point line plane
```

након чега се наводи листа предикатских симбола заједно са листом типова за сваки од аргумената. Сваки предикатски симбол задаје се у посебном реду након кључне речи `datatype`. На пример, предикат који описује да тачка припада правој може се задати као:

```
datatype inc_po_1 point line
```

Подразумева се да предикат `eq_type` означава једнакост два објекта типа `type`.

- скуп аксиома – аксиоме су задате у формату `premises Prem conclusions Concl`, при чему су са `Prem` означене премисе аксиоме, задате низом предикатских симбола у којима су променљиве представљене природним бројевима, а са `Concl` закључци аксиоме у истом формату као и премисе и у којима се евентуална појава дисјункција означава са `|`. Све променљиве које се јављају са леве стране импликације подразумевано су универзално

квантификоване, а променљиве које се јављају само са десне стране импликације су егзистенцијално квантификоване. Пример једне аксиоме би био:

```
premises
point(1)
point(2)
~eq_point(1,2)

conclusions
line(3)
inc_po_l(1,3)
inc_po_l(2,3)
```

Симбол $\sim R$ означава предикат који игра улогу негације предикатског симбола R , уведен у поглављу [3.1.1](#).

- скуп дефиниција – ради угодности могу се користити и дефиниције и оне имају исти формат као аксиоме, на пример:

```
premises
point(1)
point(2)
point(3)
line(4)
inc_po_l(1,4)
inc_po_l(2,4)
inc_po_l(3,4)

conclusions
colinear(1,2,3)
```

- скуп лема – некада је потребно нека тврђења, за која је претходно доказано да су тачна, додати као леме да би се могла користити приликом процеса доказивања; често су то нека једноставна тврђења, као, на пример, својства симетричности предикатских симбола. Леме се задају у истом формату као и аксиоме, на пример:

```
premises
point(1)
point(2)
point(3)
colinear(1,3,2)

conclusions
colinear(1,2,3)
```

- теорему – која је дата у истом формату као и аксиоме, на пример:

```
premises
point(1)
point(2)
point(3)
line(4)
inc_po_l(1,4)
inc_po_l(2,4)
bet(1,2,3)

conclusions
inc_po_l(3,4)
```

Други могући формат у коме је могуће задати улаз у доказивач је коришћењем TRTP (Thousands of Problems for Theorem Provers) формата [99, 97]. TRTP садржи библиотеку проблема за аутоматске доказиваче теорема којима је могуће вршити њихово тестирање, евалуацију и међусобно поређење. Стога представља пожељан формат улаза за све аутоматске доказиваче теорема. На овај начин поједностављено је поређење доказивача *ArgoCLP* са другим доказивачима за кохерентну логику, а такође и са другим аутоматским доказивачима теорема, као што су, на пример, резолуцијски доказивачи.

Конфигурисање доказивача

Сигнатура теорије над којом се ради задаје се листом дозвољених типова, као и расположивих предикатских симбола, при чему је за сваки од предикатских симбола задата његова арност и тип сваког од аргумената. За сваки од предикатских симбола задата је и његова формулација на природном језику.

На основу вредности параметара које се налазе у конфигурационој датотеци, постављају се вредности одговарајућих параметара сигнатуре који ће бити од значаја током процеса доказивања. Највећи број техника наведених у поглављу 3.1.1 имплементиран је у оквиру доказивача ArgoCLP. Корисник у конфигурационој датотеци може да наведе које од техника жели да користи током потраге за доказом:

- *параметар за једнакост* контролише да ли корисник жели да користи уграђено једнакостно резонување
- *параметри за искључење трећег* контролишу да ли корисник жели да користи аксиоме искључења трећег и да ли жели да користи аксиоме искључења трећег само за примитивне предикатске симболе (а не и за дефинисане)
- *параметри за технике које које не чувају потпуност* контролишу да ли корисник жели да се у процесу доказивања користе само аксиоме које садрже само предикате који се појављују у оквиру теореме (ово се не односи на једнакост, ако је постављен параметар за једнакост); да ли жели да се користе само аксиоме које садрже барем један предикат који се јавља у оквиру теореме (такође се не односи на једнакост, ако је постављен параметар за једнакост); да ли граничник треба инкрементирати пре него што се покуша примена неке јако продуктивне аксиоме.

Излаз

Током процеса доказивања, ArgoCLP генерише траг доказа са свим релевантним информацијама. Доказивач може да изведе траг доказа у Isabelle/Isar формат и у природно-језички формат. Постоји грана развоја доказивача, чији су аутори Сана Стојановић Ђурђевић, Предраг Јаничић и Жилијен Нарбу, којом је постигнуто да се траг доказа може превести у међуформат у XML формату³ из кога се могу генерисати формални докази за интерактивне доказиваче теорема Isabelle/Isar и Coq, а такође и докази у природном језику (на енглеском језику, форматирани у L^AT_EX-у или HTML-у) [97].

Као излаз добија се и списак аксиома, дефиниција и лема који коришћених у процесу доказивања.

Развијен је и механизам за поједностављивање доказа којим се добија “чист”, често доста краћи доказ и он ће бити детаљно приказан у поглављу 3.3. Поједностављени докази се такође могу извести у различите излазне формате.

³<http://www.w3.org/XML/>

Излаз у Isabelle/Isar и Coq. У првој верзији доказивача излаз у Isabelle/Isar формату се конструисао тако што се за сваки од корака у трагу доказа генерисао одговарајући корак у Isabelle/Isar формату. Понекад је било потребно реконструисати Isabelle доказ због једнакосног резонувања и *union-find* алгоритма имплементираних у оквиру процедуре потраге за доказом. У оквиру Isabelle-a (и Coq-a) користи се уграђена негација уместо дефинисане негације у кохерентној логици. Нова верзија доказивача може да генерише доказе у Isabelle/Isar и Coq формату на основу XML формата доказа, коришћењем скупа XSL ⁴ трансформација.

Излаз у природни језик. Генерисани докази у Coq и Isabelle/Isar формату су прилично разумљиви. Ипак, за потребе генерисања читљивих доказа, који би што више наликовали доказима из математичких уџбеника, много боље решење представља превођење у природно-језички формат.

Доказивач ArgoCLP на основу трага доказа генерише низ нумерисаних корака у природно-језичкој форми у L^AT_EX формату. Ово се постиже тако што се као улаз у доказивач обезбеђује датотека са формулацијама на природном језику свих предикатских симбола из сигнатуре, при чему су знаком `_` означена места где треба ставити имена променљивих, нпр:

```
inc_po_l: the point _ is incident to the line _
```

Такође, да би генерисани докази на природном језику што више наликовали стандардним математичким доказима, објектима се у зависности од њиховог типа придружују уобичајена имена, на пример, тачке се именују $A, B, C, \dots$, праве $p, q, r, \dots$, равни $\alpha, \beta, \gamma, \dots$.

Излаз у природном језику структуриран је блоковски. Ово се остварује назубљивањем корака на следећи начин: увек када се уводи нека претпоставка, пасус се додатно увуче, а када се грана која одговара тој претпоставци затвори онда се вредност увлачења враћа на претходну вредност. На овај начин делови доказа који припадају поддоказу не ограђују се линијом нити правоугаоником, већ се визуелно, увлачењем одвајају од остатка доказа.

3.2 Примери доказа

У наставку је дат један пример генерисаног доказа у природно-језичком формату (коришћењем природно-језичких формулација предикатских симбола

⁴<http://www.w3.org/Style/XSL/>

сигнатуре теорије), заједно са аутоматски генерисаном спецификацијом тврђења на природном језику.

Пример 3.2. Доказати да ако је тачка B између тачака A и C и важи да је дуж AB подударна дужи AD и дуж CB подударна дужи CD , онда се тачке B и D поклапају⁵.

Theorem TH_geo:

Assuming that $\text{bet}(A,B,C)$, $(A, B) \cong (A, D)$, and $(C, B) \cong (C, D)$, show that $B = D$.

Proof:

1. *It holds that $\text{bet}(B,A,A)$ (by axiom th_3_1).*
2. *From the facts that $\text{bet}(A,B,C)$, it holds that $\text{col}(C,A,B)$ (by axiom ax_4_10_3).*
3. *From the facts that $(A, B) \cong (A, D)$, it holds that $(A, D) \cong (A, B)$ (by axiom th_2_2).*
4. *It holds that $A = B$ or $A \neq B$ (by axiom ax_g1).*
5. *Assume that $A = B$.*
6. *From the facts that $(A, D) \cong (A, B)$ and $A = B$ it holds that $(A, D) \cong (A, A)$.*
7. *From the facts that $(A, D) \cong (A, A)$, it holds that $A = D$ (by axiom ax_3).*
8. *From the facts that $A = A$, $A = B$ and $A = D$ it holds that $B = D$.*

This proves the conjecture.

9. *Assume that $A \neq B$.*

Let us prove that $A \neq C$ by reductio ad absurdum.

10. *Assume that $A = C$.*
11. *From the facts that $\text{bet}(A,B,C)$ and $A = C$ it holds that $\text{bet}(A,B,A)$.*
12. *From the facts that $\text{bet}(A,B,A)$, and $\text{bet}(B,A,A)$, it holds that $A = B$ (by axiom th_3_4).*
13. *From the facts that $A \neq B$, and $A = B$ we get a contradiction.*

Contradiction.

Therefore, it holds that $A \neq C$.

14. *From the fact that $A \neq C$, it holds that $C \neq A$ (by the equality axioms).*

⁵Ово је један од примера који прате рад “A Vernacular for Coherent Logic” [97] и који су доступни на адреси argo.matf.bg.ac.rs/downloads/software/clvernacular.zip.

15. From the facts that $C \neq A$, $\text{col}(C, A, B)$, $(C, B) \cong (C, D)$, and $(A, B) \cong (A, D)$, it holds that $B = D$ (by axiom *th_4_18*).

This proves the conjecture.

Theorem proved in 15 steps and in 2.67 s.

У наставку следи главни део генерисаног доказа истог тврђења у Isabelle/Isar формату⁶.

```
lemma TH_geo:
  assumes "bet P01 P02 P03" and "cong P01 P02 P01 P04"
  and "cong P03 P02 P03 P04"
  shows " (P02 = P04)"
  proof -

    have "bet P02 P01 P01"
    using th_3_1 [of "P02" "P01"]
    by auto

    from 'bet P01 P02 P03'
    have "col P03 P01 P02"
    using ax_4_10_3 [of "P01" "P02" "P03"]
    by auto

    from 'cong P01 P02 P01 P04'
    have "cong P01 P04 P01 P02"
    using th_2_2 [of "P01" "P02" "P01" "P04"]
    by auto

    have "P01 = P02 \<or> P01 ~= P02"
    using ax_g1 [of "P01" "P02"]
    by auto

    { assume "P01 = P02"
    from 'cong P01 P04 P01 P02' and 'P01 = P02'
    have "cong P01 P04 P01 P01"
    by auto

    from 'cong P01 P04 P01 P01'
    have "P01 = P04"
    using ax_3 [of "P01" "P04" "P01"]
    by auto
```

⁶На много места у доказу могуће је релативно једноставно избацити аутоматизацију (тј. позиве тактике *auto*).

```
from 'P01 = P02' and 'P01 = P04'
have "P02 = P04"
by auto

from 'P02 = P04'
have ?thesis by auto
} note note1 = this

{ assume "P01 ~= P02"
have "P01 = P03 \<or> P01 ~= P03"
using ax_g1 [of "P01" "P03"]
by auto

{ assume "P01 = P03"
from 'bet P01 P02 P03' and 'P01 = P03'
have "bet P01 P02 P01"
by auto

from 'bet P01 P02 P01' and 'bet P02 P01 P01'
have "P01 = P02"
using th_3_4 [of "P01" "P02" "P01"]
by auto

from 'P01 ~= P02' and 'P01 = P02'
have False by auto
} note note2 = this

{ assume "P01 ~= P03"
from 'P01 ~= P03'
have "P03 ~= P01"
by auto

from 'P03 ~= P01' and 'col P03 P01 P02'
and 'cong P03 P02 P03 P04' and 'cong P01 P02 P01 P04'
have "P02 = P04"
using th_4_18 [of "P03" "P01" "P02" "P04"]
by auto

from 'P02 = P04'
have ?thesis by auto
} note note3 = this

from note2 and note3 and 'P01 = P03 \<or> P01 ~= P03'
have ?thesis
by auto
```

```
} note note4 = this

from note1 and note4 and 'P01 = P02 \<or> P01 ~= P02'
have ?thesis by auto
from this
show ?thesis by auto
qed
```

Доказивач ArgoCLP примењен је на четири аксиоматска система за еуклидску геометрију на униформан начин. То су Хилбертов аксиоматски систем [51], аксиоматски систем Тарског [101, 94], систем који су формулисали Борсук и Смилева [11] и аксиоматски систем из уџбеника Зорана Лучића [62]. Да би се покушала доказати иста тврђења у оквиру различитих аксиоматских система, за све системе коришћена је иста сигнатура. Она се добија као унија свих типова и предикатских симбола коришћених у сваком од ових система. Ако неки систем не користи неки од предикатских симбола, он се не може користити за доказивање његових својстава. На пример, аксиоматски систем Тарског не може се користити за доказивање својстава релација инциденције, јер ради само над тачкама. Све аксиоме ових система, са изузетком аксиома непрекидности због њихове тежине, искодирана су у датом систему. Велики фрагмент геометрије може се изградити и без њиховог коришћења. Неке од аксиома су преформулисане да би се избегло увођење комплексних појмова као што су полуправа, полураван, унутрашњи угао итд., али је очувано њихово оригинално значење.

Кодирање аксиома само по себи није тривијалан задатак, јер су оригиналне формулације често непрецизне и притом се нека својства имплицитно претпостављају. На пример, када Хилберт у својим аксиомама користи синтагму “две тачке”, он претпоставља да су оне различите (иако то нигде експлицитно не тврди). На овај проблем су указали и Мејкл и Флоријо [72]. Постоји велики број проблема ове врсте и понекада није тривијално показати да ли би нека модификација променила снагу система. Овде нам није за циљ темељно поређење ова четири система, већ илустровање доказивача ArgoCLP. Доказивач се такође може користити за утврђивање које се аксиоме и како могу модификовати.

Доказивач ArgoCLP примењен је на ове аксиоматске системе и доказан је одређен број теорема. Као што је и очекивано, резултати доста зависе од скупа аксиома који се користи. Као пример, приказан је списак од 14 теорема (укључујући и неке које нису доказане од стране доказивача у оквиру временског лимита од 30 секунди) и добијени резултати за четири аксиоматска система у

табели 3.1. Сви резултати добијени су за једну фиксирану конфигурацију до-казивача у којој се користе само аксиоме које укључују само предикате који се користе у тврђењу које се доказује и такође користе се само аксиоме искључења трећег за примитивне предикате.

Теорема 1: За две праве које се секу важи да постоји раван која их садржи.

$$\forall p : line \ \forall q : line \ (int(p, q) \Rightarrow \exists \alpha : plane \ (inc(p, \alpha) \wedge inc(q, \alpha)))$$

Теорема 2: Ако две различите праве имају заједничку тачку, онда је она јединствена.

$$\forall p : line \ \forall q : line \ \forall A : point \ \forall B : point \ (p \neq q \wedge inc(A, p) \wedge inc(A, q) \wedge inc(B, p) \wedge inc(B, q) \Rightarrow A = B)$$

Теорема 3: За раван и праву која не лежи у тој равни важи да ако имају заједничку тачку онда је она јединствена.

$$\forall p : line \ \forall \alpha : plane \ \forall A : point \ \forall B : point \ (\neg inc(p, \alpha) \wedge inc(A, p) \wedge inc(A, \alpha) \wedge inc(B, p) \wedge inc(B, \alpha) \Rightarrow A = B)$$

Теорема 4: Три неколинеарне тачке су две по две различите.

$$\forall A : point \ \forall B : point \ \forall C : point \ (\neg col(A, B, C) \Rightarrow A \neq B \wedge A \neq C \wedge B \neq C)$$

Теорема 5: За три неколинеарне тачке важи да постоји раван која их садржи.

$$\forall A : point \ \forall B : point \ \forall C : point \ (\neg col(A, B, C) \Rightarrow \exists \alpha : plane (inc(A, \alpha) \wedge inc(B, \alpha) \wedge inc(C, \alpha)))$$

Теорема 6: За праву и тачку која не припада тој правој важи да постоји раван која их садржи.

$$\forall A : point \ \forall p : line \ (\neg inc(A, p) \Rightarrow \exists \alpha : plane \ (inc(A, \alpha) \wedge inc(p, \alpha)))$$

Теорема 7: За четири компланарне тачке важи да ако три од њих нису колинеарне и припадају некој равни, онда и четврта тачка припада тој равни.

$$\forall A : point \ \forall B : point \ \forall C : point \ \forall D : point \ \forall \alpha : plane \ (comp(A, B, C, D) \wedge \neg col(A, B, C) \wedge inc(A, \alpha) \wedge inc(B, \alpha) \wedge inc(C, \alpha) \Rightarrow inc(D, \alpha))$$

Теорема 8: За три праве p , q и r од којих су p и q , и q и r различите и раван α која их садржи важи да ако се праве p и q не секу и праве q и r не секу и ако постоји тачка A која припада равни α и правама p и r , онда се праве p и r поклапају.

$$\forall p : line \ \forall q : line \ \forall r : line \ \forall A : point \ \forall \alpha : plane \ (p \neq q \wedge q \neq r \wedge inc(p, \alpha) \wedge inc(q, \alpha) \wedge inc(r, \alpha) \wedge \neg int(p, q) \wedge \neg int(q, r) \wedge inc(A, \alpha) \wedge inc(A, p) \wedge inc(A, r) \Rightarrow p = r)$$

Теорема 9: Ако тачке A и B припадају истој правој и постоји тачка C тако да је тачка B између тачака A и C онда и тачка C припада тој правој.

$$\forall A : point \quad \forall B : point \quad \forall C : point \quad \forall p : line \quad (inc(A, p) \wedge inc(B, p) \wedge bet(A, B, C) \Rightarrow inc(C, p))$$

Теорема 10: За три тачке A , B и C такве да тачка B лежи између тачака A и C важи да тачка C не лежи између тачака A и B .

$$\forall A : point \quad \forall B : point \quad \forall C : point \quad (bet(A, B, C) \Rightarrow \neg bet(A, C, B))$$

Теорема 11: За две различите тачке постоји тачка која лежи између њих.

$$\forall A : point \quad \forall B : point \quad (A \neq B \Rightarrow \exists C : point \quad bet(A, C, B))$$

Теорема 12: Релација конгруенције је рефлексивна.

$$\forall A : point \quad \forall B : point \quad cong(A, B, A, B)$$

Теорема 13: Релација конгруенције је симетрична.

$$\forall A : point \quad \forall B : point \quad \forall C : point \quad \forall D : point \quad (cong(A, B, C, D) \Rightarrow cong(C, D, A, B))$$

Теорема 14: Релација конгруенције је симетрична по паровима.

$$\forall A : point \quad \forall B : point \quad \forall C : point \quad \forall D : point \quad (cong(A, B, C, D) \Rightarrow cong(B, A, D, C))$$

#	ARGO систем	Систем Тарског	Борсуков систем	Хилбертов систем
1	-	NA	-	-
2	0.01/5	NA	0.01/5	0.01/5
3	0.01/5	NA	0.01/5	0.01/5
4	-	NA	-	-
5	0.01/27	NA	0.03/28	-
6	-	NA	16.07/524	-
7	11.08/125	NA	8.09/119	-
8	0.01/12	NA	0.01/12	0.01/12
9	-	NA	-	-
10	0.01/2	-	0.01/2	-
11	-	-	0.07/71	-
12	0.01/5	0.01/6	0.01/6	-
13	0.25/13	0.16/24	0.22/24	-
14	1.26/26	0.52/30	0.57/30	-

Табела 3.1: Перформансе доказивача; вредности у табели су дате у формату време/ n_1 , где је са n_1 означен број корака у доказу; '-' означава да је прекорачен временски лимит, NA означава да теорема не припада језику теорије; експерименти су извођени на PC Core 2Quad 2.4GHz са 4GB RAM, под Linux-ом.

3.3 Модул за поједностављивање доказа

Докази које генеришу аутоматски доказивачи теорема по много чему се разликују од доказа које пишу математичари, а пре свега по самој природи решења и језику који се у њима користи. Аутоматски генерисани докази често су потпуно нечитљиви и неупотребљиви за корисника, а чак и када су читљиви неретко се састоје из хиљада корака, од којих су многи ирелевантни, чиме постају бескорисни. На пример, може се десити да доказивање теореме која је једнака некој од аксиома генерише доказ који се састоји из десетина корака; у овом случају доказ би се могао скратити на дужину од само једног корака који би био примена аксиоме којој је теорема једнака. Све ово указује на то да је екстраховање кратких и читљивих доказа од круцијалне важности за многе примене као што су:

образовна примена – “чисти”, формално верификовани докази могли би се искористити за аутоматско генерисање доказа сличних онима који се могу наћи у математичким уџбеницима. На овај начин не би се добили само докази који представљају потврду истинитости тврђења, већ и докази који су човеку разумљиви.

формализација математичког знања – поред тога што је важно имати формално верификовано математичко знање, битно је да је ово формализовано знање разумљиво човеку. Аутоматски генерисани докази, који се често састоје из хиљада корака, нису најпожељнији за изградњу корпуса верификованог математичког знања.

примене у индустрији – иако је главни циљ индустријских примена само утврђивање да ли је неко тврђење тачно или не, кратки и читљиви докази би могли да користе и индустрији. На пример, читљив доказ би могао да пружи увид у то где се појавио неки проблем било у софтверу било у хардверу.

Занимљиво је то да се чак Хилберт у свом двадесетчетвртом проблему [102] бавио критеријумом једноставности математичких доказа и развојем теорије која би имала моћ да докаже да је неки доказ најједноставнији могући под неким претпоставкама⁷. У међувремену, проблем поједностављивања доказа у различитим врстама логики препознат је као врло важан и развијене су различите технике за његово решавање, као што су одсецање [19], сажимање доказа

⁷Из неког разлога Хилберт га није укључио у листу 23 важна математичка проблема.

[58] итд. За неке од логика имплементирани су и неке додатне напредне технике као што је екстракција лема [58].

За потребе доказивача ArgoCLP развијена је процедура поједностављивања доказа која трансформише доказ у кохерентној логици у доказ истог тврђења који је често доста краћи (али не нужно и најкраћи могући) [67]. За разлику од општих процедура поједностављивања, развијена метода је специфична за кохерентну логику и једноставна за имплементацију те се доказни објекти једноставно генеришу. Кораци који се примењују током ове процедуре нису ни сложени нити неочекивани. Заправо, они обављају посао који би човеку био заморан или чак и немогућ када су у питању дуги докази.

Пре описа саме процедуре трансформисања доказа, потребно је увести неке појмове на које ће се реферисати у даљем тексту.

Дефиниција 3.1. *Кохерентна формула:*

$$A_1(\vec{x}) \wedge \dots \wedge A_n(\vec{x}) \Rightarrow \exists \vec{y}_1 \mathcal{B}_1(\vec{x}, \vec{y}_1) \vee \dots \vee \exists \vec{y}_m \mathcal{B}_m(\vec{x}, \vec{y}_m)$$

назива се гранајућом (са m грана) ако важи $m \geq 2$, а иначе се назива негранајућом (ово је у складу са дефиницијом гранајуће и негранајуће аксиоме, датој у поглављу 3.1.1)⁸.

Дефиниција 3.2. Стабло доказа у кохерентној логици је стабло за које важи:

(P1) сваком чвору придружен је скуп изведених базних атомичких формула (изведених чињеница);

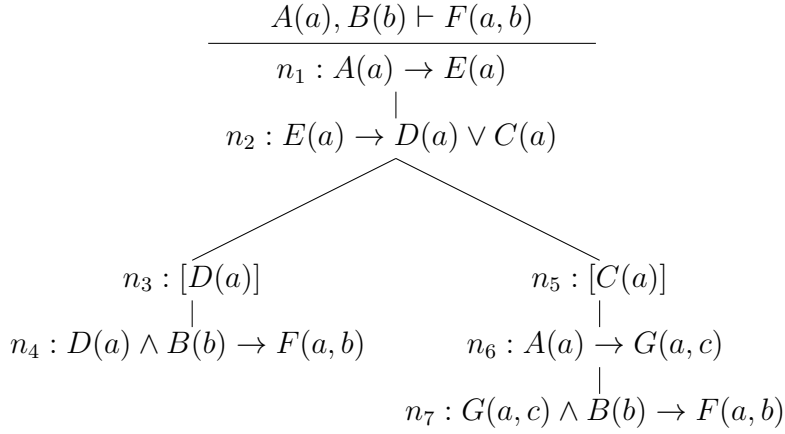
(P2) сваком чвору, изузев листова, придружена је примена аксиоме, тако да се све чињенице из њених премиса налазе у скупу изведених чињеница тог чвора. Скуп изведених чињеница неког чвора једнак је унији скупа изведених чињеница његовог родитељског чвора и чињеница из закључка примене аксиоме родитељског чвора (ово одговара правилу закључивања (ax), датом у поглављу 2.4). Ако је чвору придружена гранајућа аксиома са m грана, онда тај чвор има m синова и чињенице из i -те гране закључка аксиоме налазе се у скупу изведених чињеница i -тог сина.

Важи да је кохерентна формула:

$$A_1(\vec{x}) \wedge \dots \wedge A_n(\vec{x}) \Rightarrow \exists \vec{y}_1 \mathcal{B}_1(\vec{x}, \vec{y}_1) \vee \dots \vee \exists \vec{y}_m \mathcal{B}_m(\vec{x}, \vec{y}_m)$$

теорема у кохерентној логици ако постоји стабло доказа за које важи:

⁸Негранајућа формула одговара ономе што се уобичајено назива Хорнова формула, а гранајућа одговара не-Хорновој формули.



Слика 3.3: Пример стабла доказа

(T1) скуп изведених чињеница корена стабла састоји се од инстанцираних премиса $A_1(\vec{a}), \dots, A_n(\vec{a})$ тврђења које се доказује, при чему $\vec{a}$ означава вектор новоуведених константи;

(T2) скуп изведених чињеница сваког од листова садржи или све конјункте формуле $B_j(\vec{a}, \vec{w}_j)$, за неко j ($1 \leq j \leq m$) и за неки вектор константи $\vec{w}_j$, или контрадикцију.

Пример 3.3. На слици 3.3 приказано је стабло доказа тврђења $A(x) \wedge B(y) \rightarrow F(x, y)$. Ради читљивости, за сваки чвор приказана је само примена одговарајуће аксиоме, док је изостављен скуп изведених чињеница. У корену стабла доказа налазе се инстанциране премисе тврђења које се доказује коришћењем две константе a и b . Чворови су нумерисани у редоследу обиласка стабла доказа у дубину. За сваки чвор важи да је његов скуп изведених чињеница једнак унији скупа изведених чињеница родитељског чвора и чињеница из закључка аксиоме придружене родитељском чвору. На пример, скуп изведених чињеница чвора n_2 садржи чињенице: $A(a), B(b), E(a)$. Чвор n_2 је гранајући и скуп изведених чињеница његовог левог сина (чвора n_3) садржи додатно и чињеницу $D(a)$, а скуп изведених чињеница његовог другог сина (чвора n_5) чињеницу $C(a)$.

Ради једноставнијег записа, уводе се и следеће дефиниције:

Дефиниција 3.3. $proof(\phi, \Delta)$ означава да је Δ доказ теореме ϕ .

Дефиниција 3.4. Величина доказа Δ , у ознаци $size(\Delta)$, једнака је броју чворова у његовом стаблу.

Процедура поједностављивања састоји се из три корака:

1. уклањања ирелевантних чворова – *EIN (Eliminating Irrelevant Nodes)* трансформација
2. подизања чворова кроз гранајуће чворове и уклањања поновљених чворова – *LUP&ERN (Lifting-up and Eliminating Repeated Nodes)* трансформација
3. конвертовања у *reductio ad absurdum* форму – *RAA (Reductio ad Absurdum)* трансформација

Процедура поједностављивања доказа – *PST (Proof Simplification Transformation)* трансформација дефинише се као композиција *RAA*, *LUP&ERN* и *EIN* трансформације:

$$PST = RAA \circ LUP\&ERN \circ EIN$$

Она се примењује *post-festum*, тек када је тврђење иницијално доказано.

3.3.1 EIN трансформација

Нека је дат траг доказа конструисан коришћењем правила извођења датих у поглављу 2.4. У наставку, уместо да се траг доказа разматра као низ правила закључивања, разматраће се одговарајуће стабло доказа. За сваки чвор у стаблу доказа биће утврђено да ли је релевантан за доказ и сви ирелевантни чворови биће уклоњени из доказа.

Дефиниција 3.5. Чвор у стаблу доказа је релевантан за доказ ако примена аксиоме придружене том чвору изводи бар једну чињеницу (у свакој од грана ако је чвор гранајући) која је потребна за извођење закључка тврђења које се доказује (или контрадикције) у свим одговарајућим листовима стабла. Иначе је чвор ирелевантан.

У циљу утврђивања релевантности чворова у доказу, одржаваће се скуп R релевантних објеката у скупу тренутно обрађених чворова. У скупу R могу се наћи само базне атомичке формуле (чињенице). Скуп R се иницијализује на празан скуп. С обзиром на то да је за сваки чвор потребно утврдити да ли се чињенице које изводи користе у његовим подстаблима, неопходно је пре обраде чвора сакупити све информације о његовим подстаблима. У складу са тим *EIN* трансформација креће од последњег чвора у стаблу доказа (крајњег

десног листа у стаблу). Након тога, стабло се обрађује у смеру супротном од смера обиласка стабла у дубину. У зависности од типа чвора, разликује се и његова обрада:

Обрада негранијућег чвора: примена аксиоме негранијућег чвора је облика:

$$A_1(\vec{a}) \wedge \dots \wedge A_n(\vec{a}) \Rightarrow B^1(\vec{a}, \vec{w}) \wedge \dots \wedge B^k(\vec{a}, \vec{w})$$

при чему важи:

- $n \geq 0, k \geq 0$,
- $\vec{a}$ означава вектор константи,
- $\vec{w}$ означава вектор сведока (који може бити и празан вектор),
- $A_i, 1 \leq i \leq n$ и $B^j, 1 \leq j \leq k$ означавају атомичке формуле.

У случају негранијућег чвора, ако за неко j важи да се чињеница $B^j(\vec{a}, \vec{w})$ налази у скупу релевантних објеката R у тренутку обиласка чвора, онда се тај чвор сматра релевантним и скуп R се ажурира на следећи начин: све чињенице из закључка $(B^j(\vec{a}, \vec{w}), 1 \leq j \leq k)$ бришу се из скупа R (ако се тамо налазе), док се све чињенице из премиса $(A_i(\vec{a}), 1 \leq i \leq n)$ додају скупу R . Иначе се чвор сматра ирелевантним, при чему скуп R остаје неизмењен.

Чим се теорема докаже у некој од грана, та грана се затвара. Стога се сви листови стабла доказа (они који изводе закључак тврђења или контрадикцију) *a priori* означавају као релевантни и у складу са тим све чињенице из њихових премиса додају се скупу R у моменту њиховог обиласка.

Пример 3.4. Доказ теореме $T : A(x) \wedge B(y) \rightarrow F(x, y)$ коришћењем аксиоматског система: $A1 : A(x) \rightarrow C(x)$, $A2 : A(x) \rightarrow D(x) \wedge E(x)$, $A3 : C(x) \wedge B(y) \rightarrow F(x, y)$ приказан је на слици 3.4a. Као и у претходном примеру, ради боље читљивости, за сваки чвор приказана је само примена одговарајуће аксиоме. Тврђење је иницијално инстанцирано увођењем две нове константе a и b и заменом универзално квантификованих променљивих x и y овим константама. Циљ је извести закључак $F(a, b)$ коришћењем датог скупа аксиома и претпоставки $A(a)$ и $B(b)$.

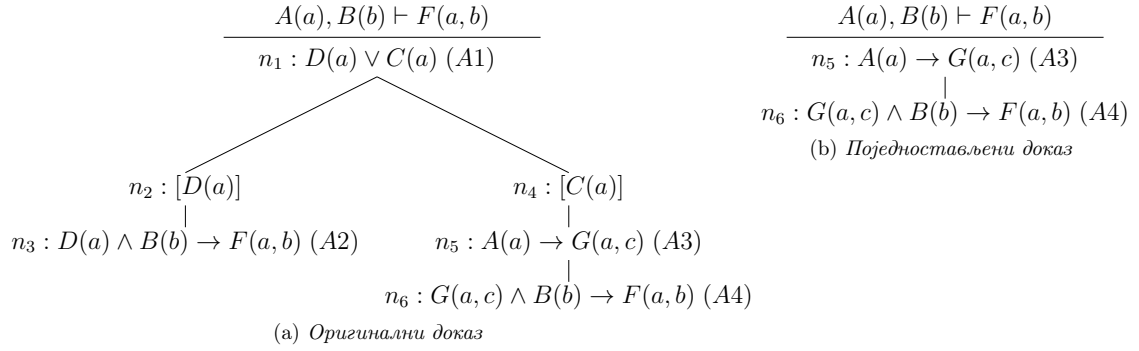
EIN трансформација креће од листа n_3 и скуп R се иницијализује на $\{C(a), B(b)\}$. Након тога обрађује се чвор n_2 и он се означава као ирелевантан јер се ниједна од чињеница из његовог закључка не налази у скупу R . Последњи

Слика 3.4: Пример уклањања неграничућег чвора

Обрада гранајућег чвора: примена аксиоме гранајућег чвора је облика:

при чѣмъ важи:

- Гранајући чвор има онолико синова колико је дисјункта у њему придруженој аксиоми, и сваки од његових синова назива се *чвором претпоставке* (јер “претпоставља” да важи B_j), а свако B_j *претпоставком гранања*. Гранајући чвор је релевантан само ако је за извођења спроведена у свакој од грана потребна њена претпоставка гранања (B_j) за извођење закључка тврђења (или контрадикције). Ово заправо значи да за сваку од грана важи да се барем једна од чињеница из њене претпоставке гранања налази у скупу R у тренутку обраде чвора претпоставке те гране. Ако то није случај, онда је гранање сувишно, гранајући чвор



Слика 3.5: Пример уклањања гранајућег чвора

заједно са осталим гранама може се уклонити из стабла доказа и од те тачке у стаблу доказа се може задржати само извођење спроведено у тој грани⁹. У случају када је та грана затворена контрадикцијом, може се закључити да је почетно тврђење противречно. У овом случају, скуп R остаје неизмењен.

С друге стране, ако је гранајући чвор означен као релевантан, онда се из скупа R бришу све чињенице из његовог закључка (за свако $B_j(\vec{a}, \vec{w}_j)$, $1 \leq j \leq m$), а додају све чињенице из његових премиса ($A_i(\vec{a})$, $1 \leq i \leq n$).

Пример 3.5. Нека је дат доказ исте теореме $T : A(x) \wedge B(y) \rightarrow F(x, y)$ као у претходном примеру, у оквиру другог аксиоматског система $A1 : D(x) \vee C(x)$, $A2 : D(x) \wedge B(y) \rightarrow F(x, y)$, $A3 : A(x) \rightarrow G(x, y)$, $A4 : G(x, y) \wedge B(z) \rightarrow F(x, z)$, који је приказан на слици 3.5a. На почетку се R поставља на скуп $\{G(a, c), B(b)\}$ и током обилазак чвора n_5 , чињеница $G(a, c)$ из закључка се брише, а чињеница $A(a)$ додаје скупу R . Може се приметити да се у тренутку обраде чвора претпоставке n_4 , његова претпоставка гранања $C(a)$ не налази у скупу R , те се стога целокупан доказ може заменити краћим, негранајућим доказом који се састоји само из чворова n_5 и n_6 .

Када се обрада стабла заустави (када се обради и корен стабла), скуп R садржи само инстанциране премисе тврђења које се доказује (што води томе да су све неопходне информације за доказивање тврђења дате у његовим премисама). Прецизније, скуп R једнак је подскупу инстанцираних премиса; ако је његов прави подскуп, онда су неке од премиса тврђења редундантне што значи да се тврђење може доказати и без њиховог коришћења.

Псеудокод *EIN* трансформације дат је алгоритмом 1. Претпоставља се да сваки чвор има поље *children_num* које садржи број синова тог чвора, као и век-

⁹Ако постоји више од једне гране која не користи своју претпоставку гранања, било која од грана може се избрати да остане у доказу.

тор *child* који садржи све његове синове. *R* је глобална променљива иницијално постављена на празан скуп.

EIN трансформацијом се у “чистом” доказу задржавају само чворови за које је утврђено да су релевантни. У поглављу 3.3.2 биће показано да је новоконструисани објекат и даље доказ истог тврђења. Додатно, када се трансформација заустави, стабло доказа не садржи ниједан ирелевантан чвор.

Algorithm 1 *EIN*(*r*)

Require: *r* - корен стабла доказа

Ensure: корен поједностављеног стабла доказа истог тврђења

```

children_num  $\leftarrow r.children\_num$ 
 $A \leftarrow \bigcup_{i=1..n} A_i(a)$  { $A_i(a)$ ,  $1 \leq i \leq n$  су премисе аксиоме придружене чвору r}
if children_num = 0 then {лист}
     $R \leftarrow R \cup A$  {ажурирање скупа релевантних објеката}
    return r {сви листови се иницијално означавају као релевантни}
else if children_num = 1 then {негранајући чвор}
     $r_0 \leftarrow EIN(r.child[0])$  { $r_0$  је корен поједностављеног стабла}
     $B \leftarrow \bigcup_{j=1..k} B^j(a, w)$  { $B = \bigwedge_{1 \leq j \leq k} B^j(a, w)$  је закључак аксиоме придружене чвору r}
    if  $R \cap B \neq \emptyset$  then {чвор r је релевантан}
         $r.child[0] \leftarrow r_0$ 
         $R \leftarrow (R \setminus B) \cup A$  {ажурирање скупа релевантних објеката}
        return r
    else {чвор r је ирелевантан}
        return  $r_0$ 
    end if
else {гранајући чвор}
    for  $j = children\_num - 1$  downto 0 do
         $R \leftarrow \emptyset$ 
         $B_j \leftarrow \bigcup_{k=1..k_j} B_j^k(a, w_j)$  { $B_j = \bigwedge_{1 \leq k \leq k_j} B_j^k(a, w_j)$  је j-та претпоставка гранања аксиоме придружене чвору r}
         $r_j \leftarrow EIN(r.child[j])$  { $r_j$  је корен поједностављеног j-тог подстабла}
        if  $R \cap B_j \neq \emptyset$  then {ова грана користи своју претпоставку}
             $(r.child[j]) \leftarrow r_j$ 
             $R_j \leftarrow R$  {скуп релевантних чињеница у овој грани памти се у променљивој  $R_j$ }
        else {ова грана не користи своју претпоставку}
            return  $r_j$  {остале гране и гранајући чвор бришу се из стабла доказа}
        end if
    end for
     $R \leftarrow \bigcup_{j=1..children\_num} R_j \setminus A$ 
    return r {све гране су релевантне, гранајући чвор се задржава у поједностављеном доказу}
end if

```

Пример 3.6. Нека су дате аксиоме (A1)–(A9) и тврђење (C):

$$(A1): \forall x A(x) \rightarrow E(x)$$

$$(A2): \forall x A(x) \rightarrow I(x)$$

$$(A3): \forall x B(x) \rightarrow H(x)$$

$$(A4): \forall x C(x) \wedge I(x) \rightarrow J(x)$$

$$(A5): \forall x \forall y \forall z G(x, y) \wedge B(x) \wedge B(z) \rightarrow F(x, z)$$

$$(A6): \forall x \forall y D(x) \wedge H(y) \rightarrow F(x, y)$$

$$(A7): \forall x D(x) \vee C(x)$$

$$(A8): \forall x (B(x) \wedge K(x)) \vee D(x)$$

$$(A9): \forall x A(x) \rightarrow \exists y G(x, y)$$

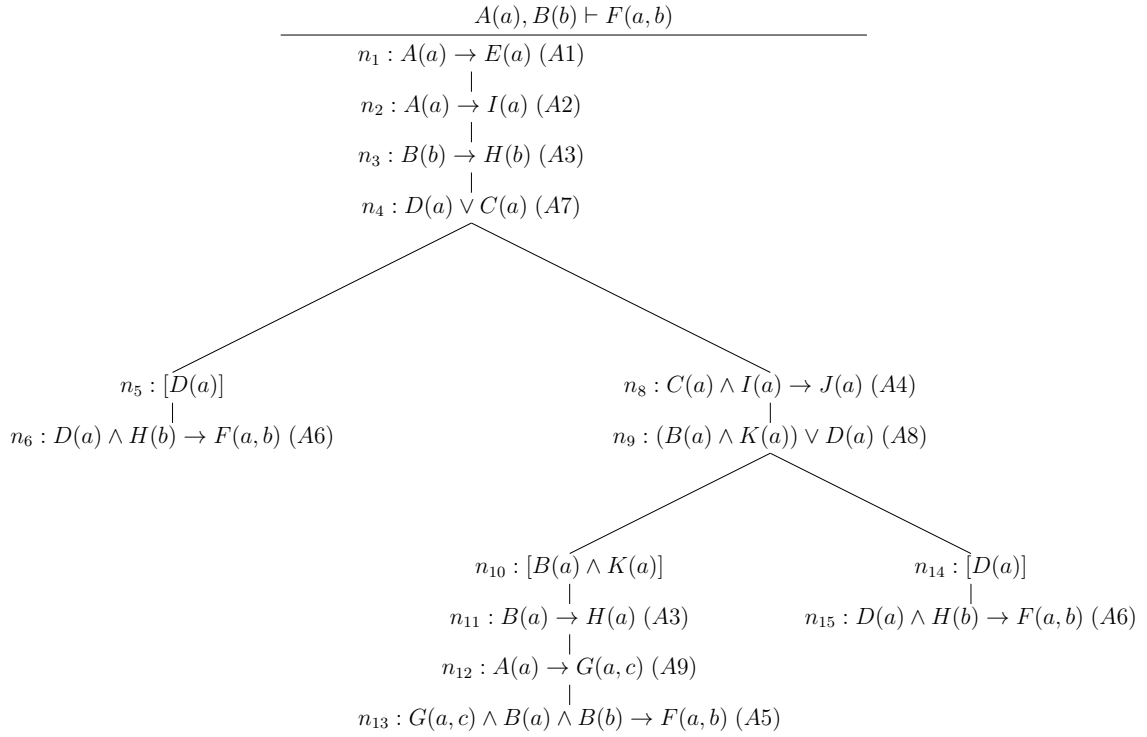
$$(C): \forall x \forall y A(x) \wedge B(y) \rightarrow F(x, y)$$

Доказ тврђења C , коришћењем датог аксиоматског система, приказан је на слици 3.6. У овом доказу, све гране су затворене извођењем закључка тврђења $(F(a, b))$. Може се приметити да корен n_1 генерисаног стабла доказа није релевантан за доказ јер изводи само једну чињеницу $E(a)$ која се нигде даље не користи у доказу. Слично, чвор n_8 није релевантан јер се чињеница коју изводи $(J(a))$ не користи даље у тој грани. Стога се доказ може поједноставити.

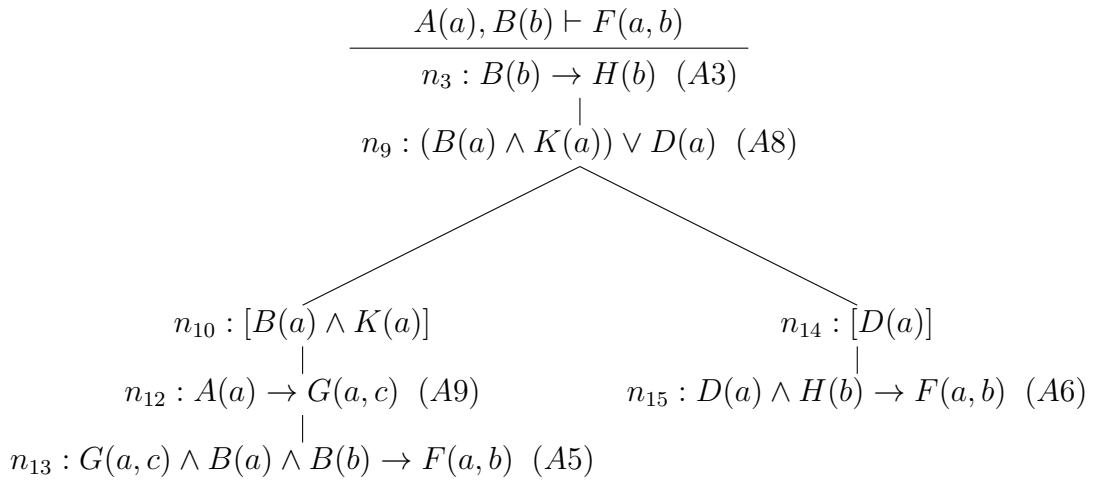
EIN трансформација би утврдила да је први гранајући чвор n_4 ирелевантан, јер се претпоставка гранања $C(a)$ не користи за извођење закључка у тој грани (иако се користи у чвору n_8 , али он није релевантан за доказ). Други гранајући чвор n_9 био би означен као релевантан јер у обе претпоставке гранања постоји чињеница која се користи за извођење закључка тврђења у тој грани. Стога се у поједностављеном доказу задржава само друга грана првог гранања, при чему су уклоњени сви ирелевантни негранајући чворови (n_1, n_2, n_8, n_{11}) . Поједностављени доказ приказан је на слици 3.7.

Извођење спроведено у поједностављеном доказу може се приказати и у стилу природне дедукције, на следећи начин:

$$\frac{(B(a) \wedge K(a)) \vee D(a) \quad (3.2) \quad (3.3)}{F(a, b)} \vee E \quad (3.1)$$



Слика 3.6: Оригинални доказ



Слика 3.7: Доказ након EIN трансформације

при чему су са (3.2) и (3.3) означени наредни фрагменти доказа:

$$\frac{\frac{A(a) \quad \frac{[B(a) \wedge K(a)]}{B(a)} \wedge E}{G(a, c)} \quad \frac{A(x) \wedge B(x) \Rightarrow G(x, y)}{ax} \quad \frac{B(b) \quad G(x, y) \wedge B(z) \Rightarrow F(x, z)}{ax}}{F(a, b)} \quad ax \quad (3.2)$$

$$\frac{[D(a)] \quad \frac{B(b) \quad B(x) \Rightarrow H(x)}{H(b)} ax \quad D(x) \wedge H(y) \Rightarrow F(x, y)}{F(a, b)} ax \quad (3.3)$$

3.3.2 Својства EИN трансформације

У овом поглављу биће доказано да EИN трансформација као излаз даје валидан доказ истог тврђења, чија је величина мања или једнака величини оригиналног доказа и који нема ирелевантних корака¹⁰.

Лема 3.1 (Коректност EИN трансформације). *Ако важи $proof(\phi, \Delta)$, онда важи и $proof(\phi, EИN(\Delta))$.*

Доказ. За почетак треба показати да новоконструисани објекат $EИN(\Delta)$ јесте доказ, тј. да задовољава својство (P1) којим се тврди да је сваком чвору придружен скуп базних атомичких формула и (P2) којим се тврди да за аксиому придружену чвору важи да се све њене премисе налазе у скупу изведених чињеница тог чвора, а закључци у скупу изведених чињеница његових синова. Својство (P1) тривијално важи, с обзиром на то да важи за доказ Δ , а скуп чворова од $EИN(\Delta)$ је подскуп скупа чворова доказа Δ .

Нека $EИN(\Delta)$ садржи чвор n за који важи да постоји чињеница F из премиса њему придружене аксиоме, која се не налази у скупу изведених чињеница тог чвора. С обзиром на то да чвор n припада $EИN(\Delta)$, чињеница F је, након обраде тог чвора, додата скупу релевантних објеката. У складу са тим, предак чвора n који садржи примену аксиоме која изводи чињеницу F у доказу Δ (а он мора да постоји у Δ , јер је у питању валидно стабло доказа) био би додат $EИN(\Delta)$. Стога се чињеница F налази у скупу изведених чињеница тог чвора што је контрадикција. Дакле, почетна претпоставка је била погрешна, односно за сваки чвор у $EИN(\Delta)$ важи да се све чињенице из премиса аксиоме налазе у скупу изведених чињеница тог чвора. Стога, $EИN(\Delta)$ задовољава и својство (P2). Следи да је $EИN(\Delta)$ валидан доказ.

Остаје питање да ли је то доказ истог тврђења ϕ ? Треба показати и да својства (T1) и (T2) важе за доказ $EИN(\Delta)$ и формулу ϕ . Подсетимо се да

¹⁰Сви наведени докази у ствари представљају само грубе скице строгих доказа.

својство (T1) каже да се у корену стабла налазе инстанциране премисе тврђења које се доказује, а својство (T2) да се у сваком од листова налази или неки од дисјунктата из закључка теореме или контрадикција.

Нека је корен стабла $EIN(\Delta)$ означен са n_1 , а корен стабла Δ са n_2 , и нека скуп изведених чињеница чвора n_1 садржи чињеницу F која није из скупа премиса тврђења ϕ . С обзиром на то да скуп изведених чињеница чвора n_2 не садржи F , за доказ Δ важи да је чвор n_1 потомак чвора n_2 и постоји предак чвора n_1 који изводи чињеницу F . Али у том случају, према спецификацији трансформације, тај чвор би био додат стаблу $EIN(\Delta)$, и с обзиром на то да је он предак чвора n_1 , чвор n_1 не би могао да буде корен стабла $EIN(\Delta)$, што је контрадикција. Стога, корен стабла $EIN(\Delta)$ садржи само чињенице из премиса тврђења ϕ , те $EIN(\Delta)$ задовољава својство (T1).

Према спецификацији трансформације, скуп листова стабла $EIN(\Delta)$ је подскуп скупа листова стабла Δ (јер је сваки лист иницијално постављен да буде релевантан и може се уклонити из поједностављеног доказа само заједно са целом граном). Према томе, $EIN(\Delta)$ задовољава и својство (T2), те $EIN(\Delta)$ представља доказ истог тврђења ϕ .

□

Лема 3.2 (Заустављање EIN трансформације). *Процедура елиминисања ирелевантних чворова се зауставља.*

Доказ. EIN трансформација се заснива на обиласку стабла доказа и обрада сваког чвора се изводи тачно једном, у константном времену. Штавише, у случају када у стаблу доказа постоји гранајући чвор који није релевантан и у доказу треба оставити грану која није прва по реду, чворови који се налазе у гранама које јој претходе неће ни бити обрађени. Прецизније, ако се са $T(n)$ означи укупан број рекурзивних позива за стабло које садржи n чворова, онда важи:

$$T(1) = 1,$$

$$T(n) \leq \sum_i T(n_i) + 1, \quad j \sum n_i = n - 1.$$

Индукцијом се може показати да важи $T(n) \leq n - 1$. С обзиром на то да се сваки чвор обрађује у константном времену, следи да је сложеност целе процедуре $O(n)$. Стога важи да се EIN трансформација зауставља и линеарне је сложености (у односу на величину улазног стабла доказа). □

Лема 3.3 (Идемпотентност EIN трансформације). *Процедура елиминисања ирелевантних чворова не садржи ирелевантне чворове, тј. важи: $EIN(EIN(\Delta)) = EIN(\Delta)$*

Доказ. Нека је $EIN(\Delta)$ краће означено са Δ_1 . Треба показати да важи $EIN(\Delta_1) = \Delta_1$, тј. да су сви чворови доказа Δ_1 релевантни за доказ Δ_1 . Ово се може показати коришћењем математичке индукције по висини чвора у стаблу доказа.

1. базни случај: сви листови (чворови висине 0) стабла Δ_1 су релевантни, према спецификацији процедуре.
2. индуктивни корак: под претпоставком да ово својство важи за све чворове висине мање или једнаке од k , треба показати да важи за чвор n висине $k + 1$. Скуп R се не мења током обраде ирелевантних чворова и релевантност чвора зависи само од његових потомака. Сви потомци чвора n су релевантни према индуктивној хипотези (јер је њихова висина у стаблу мања), што нам даје да током конструкције стабла $EIN(\Delta_1)$ у моменту обиласка чвора n скуп R садржи исте чињенице које је садржао током конструкције стабла Δ_1 . С обзиром на то да је чвор n тада проглашен за релевантан, исто се може закључити и сад. Стога је чвор n (висине $k + 1$) релевантан.

Према томе, сви кораци у доказу Δ_1 су релевантни за доказ Δ_1 , те важи: $EIN(\Delta_1) = \Delta_1$. $\square$

Лема 3.4 (EIN трансформацијом се не повећава величина стабла доказа). $size(EIN(\Delta)) \leq size(\Delta)$.

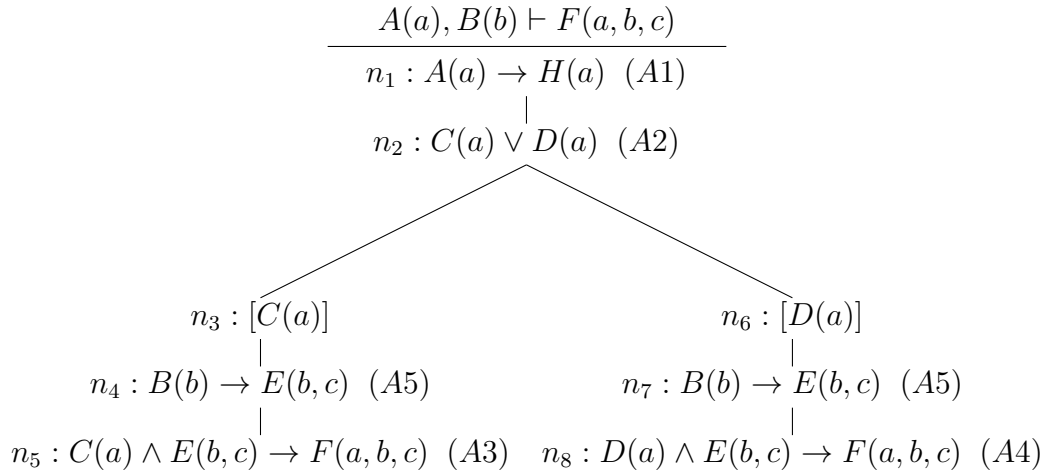
Доказ. EIN трансформација мења доказ тако што евентуално брише неке од чворова (понекад и читаве гране), али се њоме никада не додаје нови чвор. Стога важи: $size(EIN(\Delta)) \leq size(\Delta)$. $\square$

3.3.3 LUP&ERN трансформација

Дефиниција 3.6. За подстабло $n_i, n_{i+1}, \dots, n_k$ стабла $n_1, n_2, \dots, n_s$ каже се да је линеарно ако ниједан од чворова подстабла није гранајући. Линеарно подстабло је максимално ако се не може проширити ниједним од суседних чворова стабла (n_{i-1} или n_{k+1}) тако да и даље остане линеарно.

Дефиниција 3.7. За доказ ћемо рећи да је штедљив ако ни у ком случају нетривијално извођење не изводи нешто што је већ било изведено или нешто што је било претпостављено да важи¹¹.

¹¹ Овај појам уведен је у складу са термином parsimony [4]

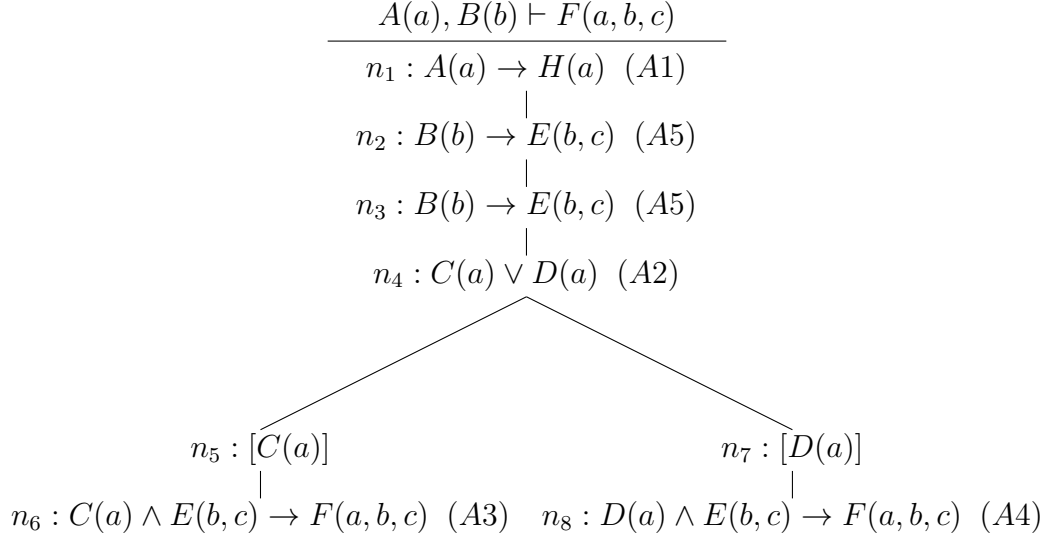


Слика 3.8: Доказ након EIN трансформације

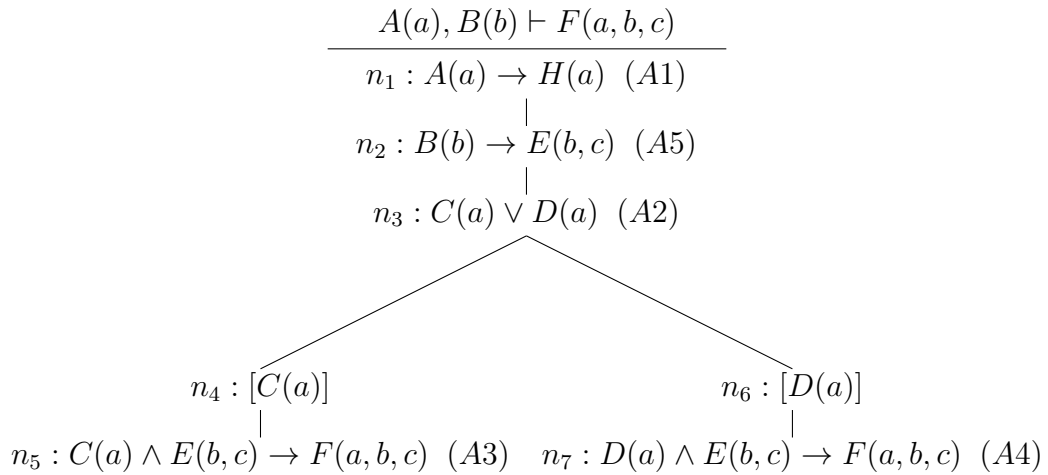
Кад се *EIN* трансформација заустави, у стаблу доказа неће бити ниједног ирелевантног чвора. Ипак, доказ не мора бити штедљив јер иста чињеница може бити изведена у различитим гранама истог гранања. У случају када чвор који изводи ту чињеницу не користи нити претпоставку гранања, нити неки од закључака изведен на путу од чвора претпоставке до тог чвора, тај чвор се може издигнути изнад гранајућег чвора. Ово је посебно корисно у ситуацијама када постоји више појављивања истог чвора у различитим гранама јер, када се издигну изнад гранања, њихово вишеструко појављивање може се заменити само једним чвором, смањујући на тај начин величину доказа.

Пример 3.7. Нека је дат доказ теореме $T : A(x) \wedge B(y) \rightarrow F(x, y, z)$ у оквиру аксиоматског система: $A1 : A(x) \rightarrow H(x)$, $A2 : C(x) \vee D(x)$, $A3 : C(x) \wedge E(y, z) \rightarrow F(x, y, z)$, $A4 : D(x) \wedge E(y, z) \rightarrow F(x, y, z)$, $A5 : B(x) \rightarrow E(x, y)$, који је приказан на слици 3.8. Чвор n_4 би се могао издигнути на позицију изнад гранајућег чвора n_2 , јер његов скуп инстанцираних премиса ($B(b)$) не зависи нити од претпоставке гранања ($C(a)$), нити од неког од чворова између чвора претпоставке и самог чвора n_4 . Слично важи и за чвор n_7 . Стога би се као међукорак ове трансформације добио доказ приказан на слици 3.9. Након тога, два појављивања истог чвора (n_2 и n_3 према новим ознакама) била би замењена само једним. Крајњи доказ приказан је на слици 3.10.

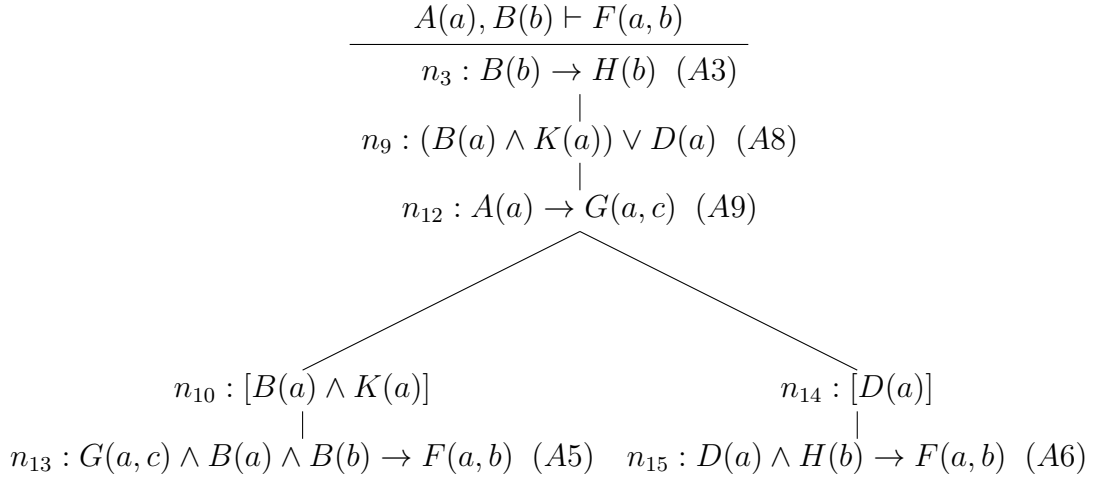
Чвор се увек издиже, без обзира на то да ли постоји или не исти такав чвор у још некој грани, јер ако би се издизао само у случају када постоје вишеструка појављивања, алгоритам би морао да води рачуна и о врло удаљеним гранањима и то би повећало сложеност алгоритма. Лоша страна приступа у коме се увек



Слика 3.9: Доказ након LUP трансформације



Слика 3.10: Доказ након ERN трансформације



Слика 3.11: Доказ након LUP&ERN трансформације; пример када се чвор који се издиже јавља само једном

врши издизање чвора је то да се чвор издиже чак и онда када има само једно појављивање.

Пример 3.8. Вратимо се примеру 3.6 и доказу добијеном након EIN трансформације. Може се уочити да се у стаблу доказа добијеном након EIN трансформације, приказаним на слици 3.7, чвор n_{12} може издигнути изнад гранајућег чвора n_9 и стабло би након LUP&ERN трансформације имало структуру приказану на слици 3.11. Тиме се врши издизање чвора кроз гранајући чвор, иако се он јавља у само једној грани. Видимо да се у овом сценарију LUP&ERN трансформацијом не смањује величина доказа.

Да би се извршило издизање чворова кроз гранајуће чворове користи се матрица зависности M_d , која је квадратна и димензије једнаке величини стабла доказа добијеног након EIN трансформације. Она се иницијално поставља на нула матрицу. Нека су чворови стабла доказа нумерисани у редоследу обиласка стабла у дубину. За сваки чвор n_i и за сваку чињеницу из скупа премиса тражи се чвор n_j који му претходи а изводи ту чињеницу и елемент $M_d(i, j)$ матрице зависности се поставља на 1. Овим се у матрици памти да ли је за извођење у чвору n_i потребно извођење у чвору n_j . Након тога, за сваку врсту r матрице M_d тражи се максимални индекс s колоне која садржи вредност 1, као индекс последњег чвора у стаблу који садржи извођење чињенице неопходно за примену аксиоме у том чвору (да би стабло остало исправно стабло доказа, мора се водити рачуна о томе да све инстанциране премисе аксиоме у датом чвору буду претходно већ изведене). У случају да постоји гранајући чвор n_g између чворова n_c и n_r , чвор n_r се брише и умеће на позицију која претходи чвору

n_g . Овај поступак се понавља све док постоје чворови кандидати за замену позиција у доказу. Коначно, за свако максимално линеарно подстабло стабла доказа идентификују се вишеструка појављивања истих чворова и оставља само једна инстанца сваког чвора.

Пример 3.9. *Илуструјмо употребу матрице зависности на доказу добијеном након EIN трансформације, приказаном на слици 3.8. С обзиром на то да ово стабло има осам чворова, димензија матрице ће бити 8×8 и иницијално је постављена на наредну вредност:*

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

Из матрице се види да за чворове n_2 , n_4 и n_7 није потребна никаква премиса, те се могу издигнути произвољно високо. Изнад чворова n_4 и n_7 постоји гранајући чвор n_2 , те се они могу издигнути изнад њега, а сам чвор n_2 је гранајући и изнад њега не постоји неки други гранајући чвор, те се он не издиже.

Псеудокод *LUPERN* трансформације дат је алгоритмом 2.

Algorithm 2 $LUP\&ERN(\Delta)$

Require: Δ - стабло доказа у коме су чворови нумерисани у редоследу обиласка стабла у дубину

Ensure: поједностављено стабло Δ

$N \leftarrow size(\Delta)$ {стабло Δ има N чворова}

$M_d \leftarrow 0_N$ {матрицу M_d иницијализујемо на нула матрицу димензије N }

for $i = 1$ to N **do** {за сваки чвор и сваку његову премису}

for $j = 1$ to $premise_num(i)$ **do** { $premise_num(i)$ означава број премиса чвора n_i }

for $k = i - 1$ downto 1 **do** {за све претходне чворове}

$B_k \leftarrow \bigcup_{l=1..m} B_k^l(a, w)$ { $B_k = \bigwedge_{1 \leq l \leq m} B_k^l(a, w)$ је закључак аксиоме придружене чвору n_k }

if $A_i^j \in B_k$ **then** { A_i^j је j -та премиса чвора n_i }

$M_d(i, k) \leftarrow 1$

end if

end for

end for

end for

repeat

$ind \leftarrow 0$

for $r = 1$ to N **do** {за сваку врсту r матрице M_d }

$found_branching \leftarrow 0$

for $c = N$ downto 1 **do** {за сваку колону c матрице M_d }

if $M_d(r, c) == 1$ **and** $found_branching == 0$ **then** {ако постоји гранајући чвор између чворова n_c и n_r }

$delete(\Delta, r)$ {бришемо чвор на позицији r из доказа}

$insert(\Delta, n_r, found_branching - 1)$ {умећемо чвор n_r пре гранајућег чвора}

$ind \leftarrow 1$;

else if $branching(n_c)$ **then** {ако је чвор гранајући, памтимо његову позицију}

$found_branching \leftarrow c$

end if

end for

end for

until $ind == 0$

for $i = 1$ to N **do** {за сваки чвор у стаблу}

for $j = i + 1$ to N **do** {гледамо чворове у истом фрагменту доказа}

if $branching(n_j)$ **or** $assuming(n_j)$ **then** {ако је чвор гранајући или чвор претпоставке}

break {завршен је линеарни фрагмент доказа}

end if

if $n_i == n_j$ **then**

$delete(\Delta, j)$ {бришемо чвор на позицији j из доказа}

end if

end for

end for

3.3.4 Својства LUP&ERN трансформације

Иста својства која важе за *EIN* трансформацију, важе и за *LUP&ERN* трансформацију.

Лема 3.5 (Коректност *LUP&ERN* трансформације). *Ако важи $proof(\phi, \Delta)$ онда важи $proof(\phi, LUP\&ERN(\Delta))$.*

Доказ. Треба показати да важе својства (P1) и (P2). Подсетимо се да својство (P1) тврди да је сваком чвору придружен скуп чињеница, а својство (P2) да за аксиому придружену неком чвору важи да се све њене премисе налазе у скупу изведених чињеница тог чвора, а закључци у скупу изведених чињеница његових синова. Сви чворови новог стабла су чворови оригиналног стабла доказа, тако да својство (P1) тривијално важи. Својство (P2) важи јер сви чворови оригиналног стабла доказа који су користили чињенице које тај чвор изводи су његови потомци у новом стаблу и све чињенице из његовог скупа премиса изведене су од стране чворова који су и даље изнад његове нове позиције у стаблу (што је осигурано алгоритмом *LUP* трансформације). Стабло које се добија након *ERN* трансформације јесте стабло доказа јер су све чињенице изведене обрисаним (поновљеним) чворовима изведене и у новом стаблу (али овај пут само једном, уместо потенцијално више пута). $\square$

Лема 3.6 (Заустављање *LUP&ERN* трансформације). *Процедура издизања и уклањања поновљених чворова се зауставља.*

Доказ. *LUP&ERN* трансформација се зауставља с обзиром на то да је број чворова у стаблу доказа коначан. *LUP* процедура премешта чворове који се јављају испод гранајућег чвора на позицију изнад гранајућег чвора, те се може применити највише онолико пута колико има негрاناјућих чворова у стаблу, што је у најгорем случају једнако $O(size(\Delta))$. У свакој итерацији *LUP* процедуре рачуна се матрица зависности M_d , што је сложености $O(size(\Delta)^2)$. Затим се за сваку врсту матрице M_d тражи максимални индекс колоне елемента са вредношћу 1 и уколико се пре тога наиђе на гранајући чвор, одговарајући чвор издиже се на позицију изнад тог гранајућег чвора – ово је сложености $O(size(\Delta))$. Након што се *LUP* процедура заустави, поновљени чворови се уклањају тако што се за свако максимално линеарно подстабло доказа Δ сви парови корака се тестирају на једнакост и ако се нађу два једнака један од њих се брише. Ова операција је сложености $O(size(\Delta)^2)$. Укупна сложеност *LUP&ERN* трансформације је $O(size(\Delta)^3)$ у најгорем случају. $\square$

Лема 3.7 (Идемпотентност $LUP\&ERN$ трансформације). *Процедура издизања и уклањања поновљених чворова је идемпотентна, тј. важи:*

$$LUP\&ERN(LUP\&ERN(\Delta)) = LUP\&ERN(\Delta)$$

Доказ. Процедура $LUP\&ERN$ се зауставља када више нема чворова који би могли бити издигнути кроз гранајуће чворове и када су сви поновљени чворови у линеарним фрагментима доказа обрисани. Стога, поновна примена процедуре не би изменила стабло доказа, јер не би било више нити чворова кандидата за издизање нити поновљених чворова. $\square$

Лема 3.8 ($LUP\&ERN$ трансформацијом се не повећава величина стабла доказа).

$$size(LUP\&ERN(\Delta)) \leq size(\Delta).$$

Доказ. Процедура издизања чворова не мења величину стабла доказа, док уклањање поновљених чворова може да смањи његову величину, те је, према томе, нова величина стабла доказа мања или једнака величини оригиналног стабла доказа. $\square$

3.3.5 RAA трансформација

Као што је већ поменуто у поглављу 2.4, у кохерентној логици није дозвољена употреба негације, али се она може симулирати увођењем новог симбола $\bar{R}$ који има улогу $\neg R$, подржану аксиомама $R(\vec{x}) \wedge \bar{R}(\vec{x}) \Rightarrow \perp$ и $R(\vec{x}) \vee \bar{R}(\vec{x})$, при чему је ова друга аксиома само конкретна инстанца *tertium non datur* аксиоматске схеме.

RAA трансформација састоји се из конвертовања фрагмената доказа у *reductio ad absurdum* форму. Биће коришћена класична верзија RAA трансформације, код које важи $(\bar{R} \rightarrow \perp) \rightarrow R$, где је R атомичка формула. Ова трансформација је посебно значајна за доказе формулисане у терминима природне дедукције, јер докази у *reductio ad absurdum* форми наликују традиционалним доказима који се могу наћи у математичким уџбеницима. Ово је важно јер се докази који су добијени у оквиру кохерентне логике могу једноставно превести у природно-језичку форму.

Из практичних разлога, скуп правила извођења проширен је правилима:

$$\frac{\begin{array}{c} [R] \\ \vdots \\ \perp \end{array}}{R} \neg I_1 \qquad \frac{\begin{array}{c} [\bar{R}] \\ \vdots \\ \perp \end{array}}{R} \neg I_2$$

при чему R означава атомичку формулу. Ова два правила су изведена из аксиоме $R(x) \vee \bar{R}(x)$ и правила $(\vee E)$. Прво правило је правило увођења негације, а друго правило ослобађања од негације (под негацијом подразумевамо нови предикат уведен да симулира негирани предикат).

У контексту ове трансформације разматра се и *проширени доказ* за који, уместо својства (P2), важи својство:

(P2') за сваки чвор или важи својство (P2) или је чвору придружена примена правила $(\neg I_1)$ или $(\neg I_2)$ – у том случају скуп изведених чињеница датог чвора садржи $\perp$, док је скуп изведених чињеница његовог сина једнак скупу изведених чињеница његовог најближег претка којем је придружена нека претпоставка, са изменом да је чињеница која одговара претпоставци замењена њеном негацијом, односно у случају када је претпоставка негација чињенице, самом том чињеницом.

RAA трансформација је применљива у случају када траг доказа садржи примену аксиоме облика $R(\vec{x}) \vee \bar{R}(\vec{x})$ на неки вектор константи $\vec{a}$, и када је једна од две гране затворена контрадикцијом. Уколико је прва грана затворена контрадикцијом, уместо примене аксиоме $R(\vec{x}) \vee \bar{R}(\vec{x})$, може се увести претпоставка прве гране $R(\vec{a})$, искористити исти низ корака који је коришћен у првој грани, извести контрадикција, и закључити, коришћењем новоуведеног правила $\neg I_1$, да важи $\bar{R}(\vec{a})$. Ако је друга грана затворена контрадикцијом, може се увести претпоставка друге гране $\bar{R}(\vec{a})$, искористити исти низ корака закључивања који је коришћен у другој грани, извести контрадикција и закључити, према новоуведеном правилу $\neg I_2$, да важи $R(\vec{a})$.

$$\begin{array}{c}
 \begin{array}{c}
 G \\
 \vdots \\
 c_1 \\
 R(\vec{a}) \vee \bar{R}(\vec{a})
 \end{array}
 \quad
 \begin{array}{c}
 [R(\vec{a})] \\
 \vdots \\
 c_2 \\
 \perp \\
 F
 \end{array}
 \quad
 \begin{array}{c}
 [\bar{R}(\vec{a})] \\
 \vdots \\
 c_3 \\
 F
 \end{array}
 \\
 \hline
 F \quad \vee E
 \end{array}
 \rightarrow
 \begin{array}{c}
 G \\
 \vdots \\
 c_1 \\
 [R(\vec{a})] \\
 \vdots \\
 c_2 \\
 \perp \\
 \hline
 \bar{R}(\vec{a}) \\
 \vdots \\
 c_3 \\
 F
 \end{array}
 \neg I_1
 \quad (3.4)$$

$$\begin{array}{c}
 \begin{array}{ccc}
 \begin{array}{c} G \\ \vdots \\ c_4 \\ R(\vec{a}) \vee \bar{R}(\vec{a}) \end{array} & \begin{array}{c} [R(\vec{a})] \\ \vdots \\ c_5 \\ F \end{array} & \begin{array}{c} [\bar{R}(\vec{a})] \\ \vdots \\ c_6 \\ \perp \\ F \end{array} \\
 \hline
 F & & \vee E
 \end{array}
 \end{array}
 \rightarrow
 \begin{array}{c}
 \begin{array}{c} G \\ \vdots \\ c_4 \\ [\bar{R}(\vec{a})] \\ \vdots \\ c_6 \\ \perp \\ R(\vec{a}) \end{array} \\
 \hline
 F \\
 \vdots \\
 c_5
 \end{array}
 \neg I_2
 \quad (3.5)$$

RAA трансформација приказана је извођењима датим у (3.4) и (3.5). Ланци корака закључивања означени су са $c_i, 1 \leq i \leq 6$, G означава премисе, а F закључак тврђења које се доказује.

Треба приметити да, због увођења правила $(\neg I_1)$ и $(\neg I_2)$, структура стабла добијеног применом RAA трансформације више не задовољава својство (P2), већ својство (P2'). Стога је добијени доказ проширени доказ.

Псеудокод RAA трансформације дат је алгоритмом 3.

Algorithm 3 $RAA(\Delta, 0, N)$

Require: Δ - стабло доказа у коме су чворови нумерисани у редоследу обиласка стабла у дубину, N - број чворова у полазном стаблу Δ

Ensure: стабло Δ трансформисано у RAA формат

for $i = N$ **downto** 0 **do**

if $i = candidates[0]$ **then**

 delete(Δ, i) {бришемо гранајући чвор на позицији i из доказа}

if $candidates[0].contradiction = 0$ **then** {ако је прва грана затворена контрадикцијом}

$RAA(\Delta, candidates[0].branching + 1, candidates[0].end_first);$

 insert($\Delta, candidates[0].end_first + 1, (\perp \Rightarrow \Delta(candidates[0].end_first + 1), \neg I_1)$) {додајемо корак који каже да онда важи претпоставка друге гране}

$RAA(\Delta, candidates[0].end_first + 1, candidates[0].end_second);$

end if

if $candidates[0].contradiction = 1$ **then** {ако је друга грана затворена контрадикцијом}

$RAA(\Delta, candidates[0].end_first + 1, candidates[0].end_second);$

 insert($\Delta, candidates[0].branching + 1, (\perp \Rightarrow \Delta(candidates[0].end_second + 1), \neg I_1)$) {додајемо корак који каже да онда важи претпоставка прве гране}

$RAA(\Delta, candidates[0].branching + 1, candidates[0].end_first);$

end if

 delete($candidates, 0$) {бришемо из скупа кандидата први елемент}

end if

end for

У коду се користи вектор *candidates* чији је сваки елемент структура са пољима:

- *branching* које садржи позицију гранајућег чвора који је кандидат за *RAA* трансформацију,
- *end_first* које садржи позицију краја прве гране тог гранања,
- *end_second* које садржи позицију краја друге гране тог гранања.
- *contradiction* које садржи индекс гране (0 или 1) која је завршена контрадикцијом.

Претпоставка је да је овај вектор пре уласка у дату трансформацију иницијализован на гранања која треба реструктурирати у доказу.

3.3.6 Својства *RAA* трансформације

У овом поглављу биће доказано да је *RAA* трансформација коректна, да се зауставља, да задовољава својство идемпотентности и да не повећава величину стабла доказа.

Лема 3.9 (Коректност *RAA* трансформације). *Ако важи $proof(\phi, \Delta)$ онда важи $proof(\phi, RAA(\Delta))$.*

Доказ. Ово својство следи директно из спецификације трансформације. Као што је већ речено, добијени доказ је проширени доказ. $\square$

Лема 3.10 (*RAA* трансформација се зауставља). *Процедура конвертовања доказа у $reductio\ ad\ absurdum$ форму се зауставља.*

Доказ. Број инстанци аксиома облика $P \vee \bar{P}$ у доказу је коначан, те се процедура зауставља. Током једног обилазак стабла доказа, свим гранајућим чворовима придружује се информација о томе да ли је он кандидат за *RAA* трансформацију (ако су сви листови у једној од две гране затворени контрадикцијом). Након тога потребан је још један обилазак стабла доказа да би се стабло трансформисало у *reductio ad absurdum* форму. Из овога следи да је временска сложеност *RAA* процедуре линеарна у односу на величину оригиналног стабла доказа. $\square$

Лема 3.11 (Идемпотентност *RAA* трансформације). *Процедура конвертовања доказа у $reductio\ ad\ absurdum$ форму је идемпотентна, тј. $RAA(RAA(\Delta)) = RAA(\Delta)$.*

Доказ. Нека је за $RAA(\Delta)$ уведен скраћени запис Δ_1 ; циљ је доказати да важи $RAA(\Delta_1) = \Delta_1$. RAA трансформација из доказа уклања све инстанце аксиома облика $P \vee \overline{P}$, за које важи да је једна грана затворена контрадикцијом. Према томе, Δ_1 не садржи ниједан чвор који је кандидат за RAA трансформацију, чиме се добија: $RAA(\Delta_1) = \Delta_1$. $\square$

Лема 3.12 (RAA трансформацијом се не повећава величина доказа). $size(RAA(\Delta)) = size(\Delta)$.

Доказ. RAA трансформација се састоји у једноставном реструктурирању доказа. Оригиналнo и трансформисано стабло имају различите структуре, али исти број чворова. Стога важи: $size(RAA(\Delta)) = size(\Delta)$. $\square$

3.3.7 Својства PST трансформације

Из својстава доказаних за EIN , $LUP\&ERN$ и RAA трансформацију у претходним поглављима, може се показати да је PST трансформација коректна, да се зауставља и да се њоме не повећава величина доказа.

Теорема 3.1 (Коректност PST трансформације). *Ако важи $proof(\phi, \Delta)$ онда важи: $proof(\phi, PST(\Delta))$.*

Доказ. Коректност PST трансформације је директна последица дефиниције PST трансформације, леме 3.1, леме 3.5 и леме 3.9. $\square$

Теорема 3.2 (PST трансформација се зауставља). *Трансформација поједностављивања доказа се зауставља.*

Доказ. Према лема 3.2, лема 3.6 и лема 3.10 EIN , $LUP\&ERN$ и RAA трансформација се заустављају, те се и PST трансформација, као њихова композиција, такође зауставља. $\square$

Теорема 3.3 (PST трансформацијом се не повећава величина доказа). $size(PST(\Delta)) \leq size(\Delta)$.

Доказ. Коришћењем леме 3.4, леме 3.8 и леме 3.12 добија се тражена неједнакост:

$$\begin{aligned} size(PST(\Delta)) &= size(RAA(LUP\&ERN(EIN(\Delta)))) \\ &= size(LUP\&ERN(EIN(\Delta))) \leq size(EIN(\Delta)) \leq size(\Delta) \end{aligned}$$

$\square$

Може се приметити да PST трансформација нема својство идемпотентности. Ово је зато што је доказ у *reductio ad absurdum* форми проширени доказ, а он не задовољава нужно својство (P2) и, због тога, није могуће поново применити EIN трансформацију на њега. Међутим, ако би се појам релевантног чвора дефинисао и за проширени доказ, на начин аналоган оном за стандардни доказ, могло би се лако доказати да $PST(\Delta)$ не садржи ирелевантне чворове. Такође не би биле могуће ни поновне примене правила $(\neg I_1)$ и $(\neg I_2)$.

3.4 Евалуација и примери поједностављених доказа

Доказивач са модулом за поједностављивање тестиран је на теоремама из области еуклидске геометрије у оквиру четири различита аксиоматска система (истим оним теоремама из поглавља 3.2).¹² Од укупно 56 тврђења (14 теорема у 4 различита аксиоматска система), успешно је доказано њих 26, генерисано је 26 трагова доказа и они су након тога поједностављени коришћењем модула за поједностављивање. Постигнута скраћења приказана су у табели 3.2. Четири теореме су идентификоване као аксиоме у одговарајућим аксиоматским системима (јер иако су оригинални докази имали и преко 20 корака, поједностављени докази су имали дужину 1 – а то је примена аксиоме којој су еквивалентне). Ова четири трага доказа елиминисана су из даље анализе, а за преостала 22, израчунат је просечни проценат редувантних корака у доказу. За овај корпус он износи 61.8%, што значи да је више од половине корака у аутоматски генерисаним доказима било редувантно. Такође је примећено да што је доказ дужи то је веће његово процентуално скраћење.

Као што је већ напоменуто, поједностављени доказ може се експортирати у Isabelle/Isar формат и његова коректност може се проверити у оквиру интерактивног доказивача теорема Isabelle. Ово је јако важно јер се тиме добија гаранција да и поједностављени доказ јесте доказ исте теореме.¹³

У наставку следи пет поједностављених аутоматски генерисаних доказа: три примера из области геометрије, који су доказани коришћењем Хилбертовог аксиоматског система (датог у додатку А), један из области презаписивања термова и један из области метричких простора, дати у природно-језичком формату.

¹² RAA трансформација примењена је само на доказе на природном језику.

¹³ Ипак, овим се не гарантује да је предложена процедура поједностављивања коректна – ово би се могло показати само верификацијом саме процедуре унутар интерактивног доказивача теорема.

#	ARGO систем	Систем Тарског	Борсуков систем	Хилбертов систем
1	-	NA	-	-
2	5/3	NA	5/3	5/3
3	5/3	NA	5/3	5/3
4	-	NA	-	-
5	27/1	NA	28/1	-
6	-	NA	524/59	-
7	125/4	NA	119/4	-
8	12/9	NA	12/9	12/9
9	-	NA	-	-
10	2/1	-	2/1	-
11	-	-	71/8	-
12	5/2	6/2	6/2	-
13	13/3	24/3	24/3	-
14	26/7	30/7	30/7	-

Табела 3.2: Перформансе доказивача; вредности су дате у форми n_1/n_2 , где је n_1 број корака у аутоматски генерисаном доказу, а n_2 је број корака у поједностављеном доказу у природно-језичком формату; '-' означава прекорачен временски лимит, а NA да теорема не припада језику теорије

Оригиналан аутоматски генерисан доказ прве теореме дуг је 54 корака, а након поједностављивања има само 8 корака.

Пример 3.10. Доказати да за сваке две равни важи да се не секу или постоји права која им припада обема.

Ова теорема доказана је коришћењем Хилберовог аксиоматског система.

Theorem TH_geol1:

Show that the plane α and the plane β do not intersect or there exist a line p such that the line p is incident to the plane α and the line p is incident to the plane β .

Proof:

1. It holds that the plane α and the plane β intersect or the plane α and the plane β do not intersect (by axiom ax_D8a).
2. Assume that the plane α and the plane β intersect.
3. From the facts that the plane α and the plane β intersect, there exist a point A such that the point A is incident to the plane α , the point A is incident to the plane β and $\alpha \neq \beta$ (by axiom ax_D8).
4. From the facts that $\alpha \neq \beta$, the point A is incident to the plane α , and the point A is incident to the plane β , there exist a point B such that $A \neq B$, the

point B is incident to the plane α and the point B is incident to the plane β (by axiom ax_I7).

5. From the facts that $A \neq B$, there exist a line p such that the point A is incident to the line p and the point B is incident to the line p (by axiom ax_I1).

6. From the facts that the point A is incident to the line p , $A \neq B$, the point B is incident to the line p , the point A is incident to the plane α , and the point B is incident to the plane α , it holds that the line p is incident to the plane α (by axiom ax_I6).

7. From the facts that the point A is incident to the line p , $A \neq B$, the point B is incident to the line p , the point A is incident to the plane β , and the point B is incident to the plane β , it holds that the line p is incident to the plane β (by axiom ax_I6).

This proves the conjecture.

8. Assume that the plane α and the plane β do not intersect.

This proves the conjecture.

Theorem proved in 8 steps and in 0.6 s.

За другу теорему важи да је процедура поједностављивања скратила доказ са 219 на само 10 корака.

Пример 3.11. Доказати да ако две различите равни садрже тачку и праву, онда дата тачка мора да припада тој правој.

И ова теорема је, као и претходна, доказана коришћењем Хилбертовог аксиоматског система.

Theorem TH_geo2:

Assuming that $\alpha \neq \beta$, the line p is incident to the plane α , the line p is incident to the plane β , the point A is incident to the plane α , and the point A is incident to the plane β , show that the point A is incident to the line p .

Proof:

Let us prove that the point A is incident to the line p by *reductio ad absurdum*.

1. Assume that the point A is not incident to the line p .

2. There exist a point B and a point C such that the point B is incident to the line p , $B \neq C$ and the point C is incident to the line p (by axiom ax_I3a).

3. From the facts that the line p is incident to the plane α , and the point B is incident to the line p , it holds that the point B is incident to the plane α (by axiom ax_D11).

4. From the facts that the line p is incident to the plane β , and the point B is incident to the line p , it holds that the point B is incident to the plane β (by axiom ax_D11).

5. From the facts that $B \neq C$, the point B is incident to the line p , the point C is incident to the line p , and the point A is not incident to the line p , it holds that the points B , C and A are not collinear (by axiom ax_D1a).

6. From the facts that the line p is incident to the plane α , and the point C is incident to the line p , it holds that the point C is incident to the plane α (by axiom ax_D11).

7. From the facts that the line p is incident to the plane β , and the point C is incident to the line p , it holds that the point C is incident to the plane β (by axiom ax_D11).

8. From the facts that the points B , C and A are not collinear, it holds that the points A , B and C are not collinear (by axiom ax_ncol_231).

9. From the facts that the points A , B and C are not collinear, the point A is incident to the plane α , the point B is incident to the plane α , the point C is incident to the plane α , the point A is incident to the plane β , the point B is incident to the plane β , and the point C is incident to the plane β , it holds that $\alpha = \beta$ (by axiom ax_I5).

10. From the facts that $\alpha = \beta$, and $\alpha \neq \beta$ we get a contradiction.

Contradiction.

Therefore, it holds that the point A is incident to the line p .

This proves the conjecture.

Theorem proved in 10 steps and in 4.56 s.

Аутоматски генерисани доказ треће теореме има 62 корака, а његова поједностављена верзија само 16 корака. У овом доказу идентификовани су неки поновљени кораци који су, након *EIN* трансформације, елиминисани из доказа, при чему је у доказу остала само по једна појава тих корака.

Пример 3.12. Доказати да ако је тачка A ван дужи BC , онда је тачка A и ван дужи CB .

Ова теорема доказана је коришћењем аксиоматског система Тарског¹⁴

¹⁴Ово је један од примера који прате рад “A Vernacular for Coherent Logic” [97] и који су доступни на адреси argo.matf.bg.ac.rs/downloads/software/clvernacular.zip, где се може наћи и скуп аксиома Тарског.

Theorem TH_geo3:

Assuming that $out(A,B,C)$, show that $out(A,C,B)$.

Proof:

1. *There exist a point G such that $bet(B,A,G)$ and $A \neq G$ (by axiom th_3_14).*
2. *From the fact that $A \neq G$, it holds that $G \neq A$ (by the equality axioms).*
3. *It holds that $A = B$ or $A \neq B$ (by axiom ax_g1).*
4. *Assume that $A = B$.*
 5. *From the facts that $out(A,B,C)$ and $A = B$ it holds that $out(A,A,C)$.*
 6. *From the facts that $out(A,A,C)$, there exist a point I such that $A \neq A$, $C \neq A$, $I \neq A$, $bet(A,A,I)$ and $bet(C,A,I)$ (by axiom $th_6_3_1$).*
 7. *From the facts that $A \neq A$, and $A = A$ we get a contradiction.*

This proves the conjecture.
8. *Assume that $A \neq B$.*
 9. *From the fact that $A \neq B$, it holds that $B \neq A$ (by the equality axioms).*

Let us prove that $A \neq C$ by reductio ad absurdum.

 10. *Assume that $A = C$.*
 11. *From the facts that $out(A,B,C)$ and $A = C$ it holds that $out(A,B,A)$.*
 12. *From the facts that $out(A,B,A)$, there exist a point I such that $B \neq A$, $A \neq A$, $I \neq A$, $bet(B,A,I)$ and $bet(A,A,I)$ (by axiom $th_6_3_1$).*
 13. *From the facts that $A \neq A$, and $A = A$ we get a contradiction.*

Contradiction.

Therefore, it holds that $A \neq C$.
 14. *From the fact that $A \neq C$, it holds that $C \neq A$ (by the equality axioms).*
 15. *From the facts that $B \neq A$, $C \neq A$, $G \neq A$, $bet(B,A,G)$, and $out(A,B,C)$, it holds that $bet(C,A,G)$ (by axiom $th_6_2_2$).*
 16. *From the facts that $C \neq A$, $B \neq A$, $G \neq A$, $bet(C,A,G)$, and $bet(B,A,G)$, it holds that $out(A,C,B)$ (by axiom $th_6_3_2$).*

This proves the conjecture.

Theorem proved in 16 steps and in 6.68 s.

Доказ четврте теореме који је генерисан аутоматски има 35 корака, а “чисти” доказ 21 корак. У презаписивању термова, није дозвољено користити негације те је могуће применити само EIN и $LUP\&ERN$ трансформације.

Пример 3.13. Својство дијаманта у теорији презаписивања гласи да ако се терм p може презаписати и у терм q и у терм r , онда се термови q и r могу презаписати у исти терм s .

Доказати да ако нека релација презаписивања задовољава својство дијаманта, онда и њено рефлексивно затворење задовољава својство дијаманта.¹⁵.

Овај пример доказан је коришћењем наредног скупа аксиома:

Аксиома ax_ref_e : $\forall RE : term \Rightarrow e(RE, RE)$

Аксиома ax_sym_e : $\forall RE1 : term \forall RE2 : term e(RE1, RE2) \Rightarrow e(RE2, RE1)$

Аксиома ax_congl : $\forall RE1 : term \forall RE2 : term \forall RE3 : term e(RE1, RE2) \wedge re(RE2, RE3) \Rightarrow re(RE1, RE3)$

Аксиома $ax_e_in_re$: $\forall RE1 : term \forall RE2 : term e(RE1, RE2) \Rightarrow re(RE1, RE2)$

Аксиома $ax_r_in_re$: $\forall RE1 : term \forall RE2 : term r(RE1, RE2) \Rightarrow re(RE1, RE2)$

Аксиома $ax_e_or_r$: $\forall RE1 : term \forall RE2 : term re(RE1, RE2) \Rightarrow e(RE1, RE2) \vee r(RE1, RE2)$

Аксиома ax_dp_r : $\forall RE1 : term \forall RE2 : term \forall RE3 : term r(RE1, RE2) \wedge r(RE1, RE3) \wedge RE2 \neq RE3 \Rightarrow \exists RE4 : term r(RE2, RE4) \wedge r(RE3, RE4)$

Theorem TH_rew:

Assuming that $RE0$ rewrites to or is equal to $RE1$, and $RE0$ rewrites to or is equal to $RE2$, show that there exist a reword $RE3$ such that $RE1$ rewrites to or is equal to $RE3$ and $RE2$ rewrites to or is equal to $RE3$.

Proof:

1. *It holds that $RE1$ is equal to $RE1$ (by axiom ax_ref_e).*
2. *It holds that $RE2$ is equal to $RE2$ (by axiom ax_ref_e).*
3. *From the facts that $RE1$ is equal to $RE1$, it holds that $RE1$ rewrites to or is equal to $RE1$ (by axiom $ax_e_in_re$).*
4. *From the facts that $RE2$ is equal to $RE2$, it holds that $RE2$ rewrites to or is equal to $RE2$ (by axiom $ax_e_in_re$).*
5. *From the facts that $RE0$ rewrites to or is equal to $RE1$, it holds that $RE0$ is equal to $RE1$ or $RE0$ rewrites to $RE1$ (by axiom $ax_e_or_r$).*
6. *Assume that $RE0$ is equal to $RE1$.*
7. *From the facts that $RE0$ is equal to $RE1$, it holds that $RE1$ is equal to $RE0$ (by axiom ax_sym_e).*

¹⁵Ово је један од проблема из Беземове колекције проблема из кохерентне логике која је расположива на адреси <http://www.ii.uib.no/~bezem/GL/>.

8. From the facts that $RE1$ is equal to $RE0$, and $RE0$ rewrites to or is equal to $RE2$, it holds that $RE1$ rewrites to or is equal to $RE2$ (by axiom ax_congl).

This proves the conjecture.

9. Assume that $RE0$ rewrites to $RE1$.

10. From the facts that $RE0$ rewrites to or is equal to $RE2$, it holds that $RE0$ is equal to $RE2$ or $RE0$ rewrites to $RE2$ (by axiom $ax_e_or_r$).

11. Assume that $RE0$ is equal to $RE2$.

12. From the facts that $RE0$ is equal to $RE2$, it holds that $RE2$ is equal to $RE0$ (by axiom ax_sym_e).

13. From the facts that $RE2$ is equal to $RE0$, and $RE0$ rewrites to or is equal to $RE1$, it holds that $RE2$ rewrites to or is equal to $RE1$ (by axiom ax_congl).

This proves the conjecture.

14. Assume that $RE0$ rewrites to $RE2$.

15. It holds that $RE1 = RE2$ or $RE1 \neq RE2$ (by axiom ax_g1).

16. Assume that $RE1 = RE2$.

17. From the facts that $RE1$ rewrites to or is equal to $RE1$ and $RE1 = RE2$ it holds that $RE2$ rewrites to or is equal to $RE1$.

This proves the conjecture.

18. Assume that $RE1 \neq RE2$.

19. From the facts that $RE0$ rewrites to $RE1$, $RE1 \neq RE2$, and $RE0$ rewrites to $RE2$, there exist a rewtterm $RE3$ such that $RE1$ rewrites to $RE3$ and $RE2$ rewrites to $RE3$ (by axiom ax_dp_r).

20. From the facts that $RE1$ rewrites to $RE3$, it holds that $RE1$ rewrites to or is equal to $RE3$ (by axiom $ax_r_in_re$).

21. From the facts that $RE2$ rewrites to $RE3$, it holds that $RE2$ rewrites to or is equal to $RE3$ (by axiom $ax_r_in_re$).

This proves the conjecture.

Theorem proved in 21 steps and in 0.32 s.

Процедура поједностављивања је доказ пете теореме скратила са 141 корака на само 7 корака.

Пример 3.14. У теорији метричких простора важи да ако је метрички простор $SP1$ комплетан и $SP2$ је његов затворени подскуп, онда је и метрички простор $SP2$ комплетан. Метрички простор $SP2$ је комплетан ако и само ако сваки Кошијев низ у $SP2$ конвергира у $SP2$.

Доказати да ако је метрички простор $SP1$ комплетан, метрички простор $SP2$ затворен, а $AR0$ је Кошијев низ у $SP2$ који не конвергира у $SP2$, онда метрички простор $SP2$ није потпростор метричког простора $SP1$ ¹⁶.

За доказивање ове теореме биће коришћен наредни скуп аксиома:

Аксиома ax_1 : $\forall SE : sequence \quad \forall SP1 : space \quad \forall SP2 : space$
 $belong_seq(SE, SP1) \wedge subspace(SP1, SP2) \Rightarrow belong_seq(SE, SP2)$

Аксиома ax_2 : $\forall SE : sequence \quad \forall SP : space \quad Cauchy(SE) \wedge$
 $belong_seq(SE, SP) \wedge complete(SP) \Rightarrow converges(SE, SP)$

Аксиома ax_3 : $\forall SE : sequence \quad \forall SP : space \quad converges(SE, SP) \Rightarrow \exists EL :$
 $element (belong_el(EL, SP) \wedge converges_to(SE, EL))$

Аксиома ax_4 : $\forall SP1 : space \quad \forall SP2 : space \quad \forall SE : sequence \quad \forall EL :$
 $element \quad closed(SP1) \wedge subspace(SP1, SP2) \wedge belong_seq(SE, SP1) \wedge$
 $converges_to(SE, EL) \Rightarrow belong_el(EL, SP1)$

Аксиома ax_5 : $\forall SP : space \quad \forall SE : sequence \quad \forall EL :$
 $element \quad converges_to(SE, EL) \wedge belong_el(EL, SP) \Rightarrow converges(SE, SP)$

Theorem TH_mp:

Assuming that the metric space $SP0$ is complete, the metric space $SP1$ is closed, the sequence $SE0$ is Cauchy, the sequence $SE0$ belongs to the metric space $SP1$, and the sequence $SE0$ does not converge in the metric space $SP1$, show that the metric space $SP1$ is not a subspace of the metric space $SP0$.

Proof:

Let us prove that the metric space $SP1$ is not a subspace of the metric space $SP0$ by reductio ad absurdum.

1. Assume that the metric space $SP1$ is a subspace of the metric space $SP0$.
2. From the facts that the sequence $SE0$ belongs to the metric space $SP1$, and the metric space $SP1$ is a subspace of the metric space $SP0$, it holds that the sequence $SE0$ belongs to the metric space $SP0$ (by axiom ax_1).
3. From the facts that the sequence $SE0$ is Cauchy, the sequence $SE0$ belongs to the metric space $SP0$, and the metric space $SP0$ is complete, it holds that the sequence $SE0$ converges in the metric space $SP0$ (by axiom ax_2).

¹⁶Ово је један од примера из рада Ганешалингама и Гауерса [40]

4. From the facts that the sequence $SE0$ converges in the metric space $SP0$, there exist a element $EL0$ such that the value $EL0$ belongs to the metric space $SP0$ and the sequence $SE0$ converges to the value $EL0$ (by axiom ax_3).

5. From the facts that the metric space $SP1$ is closed, the metric space $SP1$ is a subspace of the metric space $SP0$, the sequence $SE0$ belongs to the metric space $SP1$, and the sequence $SE0$ converges to the value $EL0$, it holds that the value $EL0$ belongs to the metric space $SP1$ (by axiom ax_4).

6. From the facts that the sequence $SE0$ converges to the value $EL0$, and the value $EL0$ belongs to the metric space $SP1$, it holds that the sequence $SE0$ converges in the metric space $SP1$ (by axiom ax_5).

7. From the facts that the sequence $SE0$ converges in the metric space $SP1$, and the sequence $SE0$ does not converge in the metric space $SP1$ we get a contradiction.

Contradiction.

Therefore, it holds that the metric space $SP1$ is not a subspace of the metric space $SP0$.

This proves the conjecture.

Theorem proved in 7 steps and in 0.65 s.

Постоје докази које није могуће поједноставити без измене самог аксиоматског система. На пример у аксиоматском систему Тарског постоји аксиома $col(PO1, PO2, PO3) \Rightarrow bet(PO1, PO2, PO3) \vee bet(PO2, PO3, PO1) \vee bet(PO3, PO1, PO2)$ која, с обзиром на то да не постоји услов да тачке $PO1$, $PO2$ и $PO3$ морају бити различите, може да буде примењена за $PO1 = A, PO2 = A, PO3 = A$. У том случају добија се гранајући корак облика $col(A, A, A) \Rightarrow bet(A, A, A) \vee bet(A, A, A) \vee bet(A, A, A)$ у коме су све три гране исте у формули са десне стране импликације. Иако се он генерално може свести на случај једне гране, то није могуће урадити у кохерентној логици. Из тог разлога овакав доказ не може се поједноставити. Овај случај илустрован је наредним примером.

Пример 3.15. Доказати да ако не важи да је тачка A између тачака A и A , онда не важи ни да су тачке A , A и A колинеарне.

Овај пример доказан је коришћењем аксиоматике Тарског.

Theorem TH_geo4:

Assuming that not $bet(A, A, A)$, show that not $col(A, A, A)$.

Proof:

Let us prove that not $col(A,A,A)$ by reductio ad absurdum.

1. *Assume that $col(A,A,A)$.*

2. *From the facts that $col(A,A,A)$, it holds that $bet(A,A,A)$ or $bet(A,A,A)$ or $bet(A,A,A)$ (by axiom $ax_def_col_1$).*

3. *Assume that $bet(A,A,A)$.*

4. *From the facts that not $bet(A,A,A)$, and $bet(A,A,A)$ we get a contradiction.*

Contradiction.

5. *Assume that $bet(A,A,A)$.*

6. *From the facts that not $bet(A,A,A)$, and $bet(A,A,A)$ we get a contradiction.*

Contradiction.

7. *Assume that $bet(A,A,A)$.*

8. *From the facts that not $bet(A,A,A)$, and $bet(A,A,A)$ we get a contradiction.*

Contradiction.

Therefore, it holds that not $col(A,A,A)$.

This proves the conjecture.

Theorem proved in 8 steps and in 0.09 s.

Да би се доказ учинио још више налик доказима у математичким уџбеницима, могуће је урадити неке једноставне модификације: из доказа се може избацити нумерација корака и уместо форме у којој је сваки корак дат у посебној линији, кораке би требало груписати у пасусе. Требало би, такође, избацити неке тривијалне кораке, као и синтагме као што је на пример “by axiom”. Доказ теореме TH_mp , трансформисан ручно у овакву форму, гласи:

Proof:

Assume that the metric space $SP1$ is a subspace of the metric space $SP0$. From the facts that the sequence $SE0$ belongs to the metric space $SP1$, and the metric space $SP1$ is a subspace of the metric space $SP0$, it holds that the sequence $SE0$ belongs to the metric space $SP0$. From the facts that the sequence $SE0$ is Cauchy, the sequence $SE0$ belongs to the metric space $SP0$, and the metric space $SP0$ is complete, it holds that the sequence $SE0$ converges in the metric space $SP0$. From

the facts that the sequence $SE0$ converges in the metric space $SP0$, there exist a element $EL0$ such that the value $EL0$ belongs to the metric space $SP0$ and the sequence $SE0$ converges to the value $EL0$. From the facts that the metric space $SP1$ is closed, the metric space $SP1$ is a subspace of the metric space $SP0$, the sequence $SE0$ belongs to the metric space $SP1$, and the sequence $SE0$ converges to the value $EL0$, it holds that the value $EL0$ belongs to the metric space $SP1$. From the facts that the sequence $SE0$ converges to the value $EL0$, and the value $EL0$ belongs to the metric space $SP1$, it holds that the sequence $SE0$ converges in the metric space $SP1$. Contradiction. Therefore, it holds that the metric space $SP1$ is not a subspace of the metric space $SP0$.

Ово је још увек далеко од доказа који се могу наћи у математичким уџбеницима. Проблем генерисања течних реченица у природно-језичкој форми изузетно је тежак и изазован задатак, чак и за веома софистициране системе.

3.5 Преглед других приступа и система

У овом поглављу биће дат преглед других доказивача за кохерентну логику, као и преглед других система за поједностављивање доказа.

3.5.1 Преглед других доказивача за кохерентну логику

Први аутоматски доказивач теорема за кохерентну логику Euklid развијен је од стране Предрага Јаничића и Стевана Кордића [55, 53]. Доказивач је користио фиксиран скуп геометријских аксиома, близак Борсуковом аксиоматском систему и успешно је доказивао на десетине основних теорема из стандардних геометријских уџбеника, али није могао да ради над произвољном кохерентном теоријом. Доказивач Euklid је користио исту основну процедуру доказивања као доказивач ArgoCLP (у његовом опису је по први пут уведен појам граничника и усмеравање претраге поретком аксиома), али није коришћена ниједна од напредних техника. Такође, доказивач је генерисао доказе на природном језику који су блиски људској интуицији, али није генерисао формалне доказе, нити су генерисани докази поједностављивани. Доказивач ArgoCLP на неки начин представља наследника овог доказивача, али и његово значајно проширење и унапређење.

Наставак рада на доказивачу ArgoCLP представља доказивач ArgoCaLyPso, развијен од стране Младена Николића и Предрага Јаничића, и у оквиру њега имплементирају се технике из CDCL SAT решавача као што су скок уназад

и учење лема [77]. Овим се омогућава да, за разлику од других доказивача за кохерентну логику, резонување не мора да буде базно. Доказивач може да генерише доказе у TPTP формату.

Први доказивач за кохерентну логику који у себи има уграђен механизам учења лема је Geo и развио га је Ханс Де Невил [30]. Учење лема омогућава да се приликом повратка у претрази, уместо да се иде на последње гранање, иде на последње гранање које је релевантно за затварање те гране (ово је механизам који је у SAT терминологији познат као скок уназад).

Доказивач CL, развијен у PROLOG-у од стране Марка Безема и Терија Кокана, заснован је на процедури претраге у ширину и генерише доказне објекте у Coq-у [9]. Докази у природном језику се не генеришу.

Доказивач који су Штефан Бергхофер и Марк Безем развили у ML-у као интерни доказивач за кохерентну логику у Isabelle-у има неколико предности у односу на “екстерне” доказиваче: користи се постојећа инфраструктура система Isabelle и искључује потреба за конвертовањем из/у “екстерне” формате. Декларативни програмски језици као што су PROLOG и ML су веома погодни за ову врсту проблема, али могу да резултују спорим извршним кодом. Такође, с обзиром на то да је доказивач интегрисан у Isabelle, овај доказивач не генерише доказе који се могу проверити у неком другом интерактивном доказивачу теорема (на пример Coq), нити доказе у природном језику.

3.5.2 Преглед постојећих система за презентовање и поједностављивање доказа

У последње време проблем поједностављивања доказа добија на значају у различитим контекстима. Главна мотивација за то је екстраховање “чистог” доказа из трага доказа генерисаног од стране аутоматских доказивача теорема. Приступи који се баве овим проблемом фокусирају се или на елиминацију редувантних делова доказа одсецањем [84], сажимањем доказа, елиминацијом заједничких подизраза [58] или представљањем доказа на компактнији начин, коришћењем, на пример, екстракције лема [86].

Општа техника најсличнија предложеној процедури поједностављивања назива се одсецање. Одсецање се састоји у избацивању свих редувантних разликовања случајева и поддоказа у којима се елиминише егзистенцијални квантификатор. Одсецање је иницијално уведено од стране Дага Правица (Dag Prawitz) [84] и оригинална варијанта је концептуално слична елиминацији гранајућих чворова. Такође, свако извођење које садржи увођење неког логичког везника, које је директно праћено његовом елиминацијом може се претворити у

еквивалентно извођење без овог “заобиласка”. Овим се проблем поједностављивања доказа своди на проблем нормализације λ -термова. Логике које су домени ове две процедуре су различите: кохерентна логика, са својим конструктивистичким системом доказивања, омогућава једноставнију и директнију процедуру поједностављивања трага доказа.

Један од начина за приказ објектних доказа у математичким уџбеницима је коришћењем графичког формата структурираног блоковски, који је иницијално предложен од стране Станислава Јашковског (Stanislaw Jaśkowski) [81]. У овом стилу приказа доказа сваки пут када се нешто претпоставља почиње нови фрагмент доказа и он се ограђује правоугаоником (означавајући на тај начин поддоказ). Фредерик Фич (Frederic Fitch) је модификовао овај начин приказа изменама као што је цртање само левих страница правоугаоника око поддоказа и подвлачењем претпоставки, и данас је овај стил приказа познат као Фичов стил. Константин Аркоудас (Konstantine Arkoudas) је развио алгоритам за поједностављивање доказа у природној дедукцији у логици првог реда датих у Фичовом стилу [4]. Он се користи за “чишћење” доказа добијених од стране аутоматских доказивача теорема заснованих на природној дедукцији у Фичовом стилу, као и од стране резолуцијских доказивача, њиховим претварањем у доказе у Фичовом стилу. Алгоритам као улаз може да узме доказ који чак и није штедљив, а затим се примењују неке технике као што су елиминисање редундантних делова, понављања и заобилазака и као излаз се добија поједностављени доказ који јесте штедљив. Његов систем прво мења структуру доказа (чиме се не враћа нужно подстабло стабла доказа), а након тога имплементирају се трансформације скраћивања којима се елиминишу неки успутни закључци који се не користе даље у доказу. Насупрот Аркоудасовој општој процедури поједностављивања, предложена процедура поједностављивања специфична је за кохерентну логику и једноставнија за имплементацију.

Још један стил доказа који је потекао из формата Јашковског је Лемонов (John Lemmon) стил [24]. Он користи формат доказа који се састоји из листе нумерисаних корака, док је за сваки од корака дат редни број тврђења која се у том кораку користе, као и назив правила које се користе. Овај формат јако је близак формату који се користи за приказ доказа на објектном нивоу у неким математичким уџбеницима.

Мијуки Кошимура (Miyuki Koshimura) и Риузо Хасегава (Ryuzo Hasegawa) су развили процедуру поједностављивања доказа за генерисање модела у доказивању теорема [58]. Њихов приступ сличан је предложеној методи по томе да се у оба приступа идентификују непотребни делови доказа, а затим елиминишу

из доказа. Ово се постиже израчунавањем релевантних атома који доприносе затварању поддоказа. Међутим, предложена метода разликује се од њиховог у следећем: прво, логика у основи њиховог система не укључује егзистенцијални квантификатор, те су логике на којима се методе заснивају различите; затим, дефиниције релевантних атома се разликују, као и метода за елиминацију ирелевантних поддоказа. У њиховом приступу, стабла поддоказа бришу се само ако су сви листови подстабла означени са $\perp$. Они, пак, користе скуп релевантних атома не само за поједностављивања доказа када се доказ комплетира, већ и током доказивања, као и за генерисање лема.

Највећи број аутоматских доказивача теорема не ставља акценат на генерисање доказа које је лако разумети. Ипак, постоје системи као што је Theorema који наглашавају значај добијања доказа који имају лепу презентацију [82]. Систем Theorema креиран је са намером да обезбеди рачунарску подршку свим аспектима циклуса математичког истраживања (укључујући доказивање, решавање и израчунавање) у оквиру једне униформне логике. Значајан аспект система чини поједностављивање доказа које систем Theorema генерише. Постоје сличности у процедури за поједностављивање доказа, али се разликује начин на који се генеришу доказни објекти, што је последица различитих логика у основи. И Моан Ганесалингем (Mohan Ganesalingam) и Тимоти Говерс (Timothy Gowers) промовишу добре стране читљивих доказа и у те сврхе такође разматрају кохерентну логику, мада не под тим именом [40]. Они су развили аутоматски решавач проблема чији је излаз јако тешко разликовати од решења које би написао човек.

Постоје, такође, многе ситуације у којима доказе треба сачувати или послати, и када би сваки вид поједностављивања доказа, као и начин компактнијег презентовања, био од користи. Шри Пракаш Раул (Shree Prakash Rahul) и Џорџ Некула (George Necula) су развили методу за оптимизацију доказа, заједно са методом за екстракцију лема којим се слични поддокази замењују инстанцама општијих лема [86]. Ови методи раде над доказима у логици првог реда и постижу значајно смањење величине доказа.

Ниједан од поменутих система не ради трансформацију доказа у *reductio ad absurdum* форму.

3.6 Summary of Chapter *ArgoCLP Prover and its Extensions*

In this chapter of the thesis, the theorem prover for coherent logic ArgoCLP is presented [98]. It is a generic theorem prover which can be used for any coherent theory and it generates readable proofs, as well as formal proofs. Its proof procedure is based on forward chaining mechanism and iterative deepening. There are many techniques that preserve completeness and that can improve efficiency of the prover. These are: ordering of the axioms (first non-productive axioms are used, then productive, and at the end, strongly productive ones), early pruning of inapplicable axioms, breaking of the axioms that introduce several witnesses and mechanisms for dealing with equality and symmetric predicate symbols. There are also some techniques that do not preserve completeness, but can improve efficiency significantly.

ArgoCLP is implemented in C++. The signature of the theory and the set of axioms are imported using files in internal format, or using TPTP syntax [99, 97]. The prover as an output generates a proof in natural language form, in L^AT_EX format, and a formal proof in Isabelle/Isar format, which is also quite readable. The prover has been applied to four different axiomatic systems for Euclidean geometry and to the set of theorems from elementary textbooks and most of the theorems have been proved.

A module for simplifying proofs in the framework of coherent logic has been developed [67]. It is mostly used for proof traces generated by different theorem provers. Simplification of proofs could be useful for educational purposes, for formalization of mathematical knowledge, and also for industry applications. This simplification procedure consists from three transformations:

1. elimination of irrelevant nodes,
2. lifting-up nodes through non-branching nodes and elimination of repeated nodes,
3. transformation into *reductio ad absurdum* form.

For all these transformations it is proved that they terminate, that they generate proof of the same theorem, whose size is at most equal to the size of original proof, and that they are idempotent. ArgoCLP prover with the module for proof simplification has been tested on the set of theorems from different theories and significant shortenings have been obtained.

At the end of this chapter, examples of generated proofs from different theories: geometry, term rewriting and metric spaces have been given. Also, a survey on existing systems for proving theorems in coherent logic [55, 53, 77, 30, 9] and existing systems for proof simplification [84, 81, 4, 58, 82] is presented.

4

Аутоматско решавање конструктивних проблема

У овом поглављу биће описан систем ArgoTriCS за аутоматско решавање конструктивних проблема у геометрији. Након тога биће дати примери генерисаних доказа, евалуација система на проблемима из Верниковог и Конелијевог корпуса, као и поређење са другим сличним системима.

4.1 Преглед система ArgoTriCS

ArgoTriCS је алат којим се, на основу неког расположивог геометријског знања, аутоматски решава конструктивни проблем. Проблеми које је могуће решити припадају класи проблема у којима је циљ конструисати троугао ако су задате локације три дате тачке. Кључна својства система ArgoTriCS су:

- идентификовање релевантног геометријског знања,
- погодна репрезентација знања која омогућава ефикасну претрагу,
- релативно једноставан систем претраге са уланчавањем унапред,
- навођење којим се постиже да су сви нови објекти који се креирају током конструкције потенцијално релевантни за ту конструкцију,
- механизми за препознавање симетричних проблема, редундантних проблема и проблема зависних од положаја,
- механизми који омогућавају везу са аутоматским доказивачима теорема и проверавање исправности генерисаних конструкција,

- механизми који омогућавају везу са интерактивним доказивачима теорема и формално верификовање генерисаних решења.

Решавање конструктивног проблема системом ArgoTriCS подразумева:

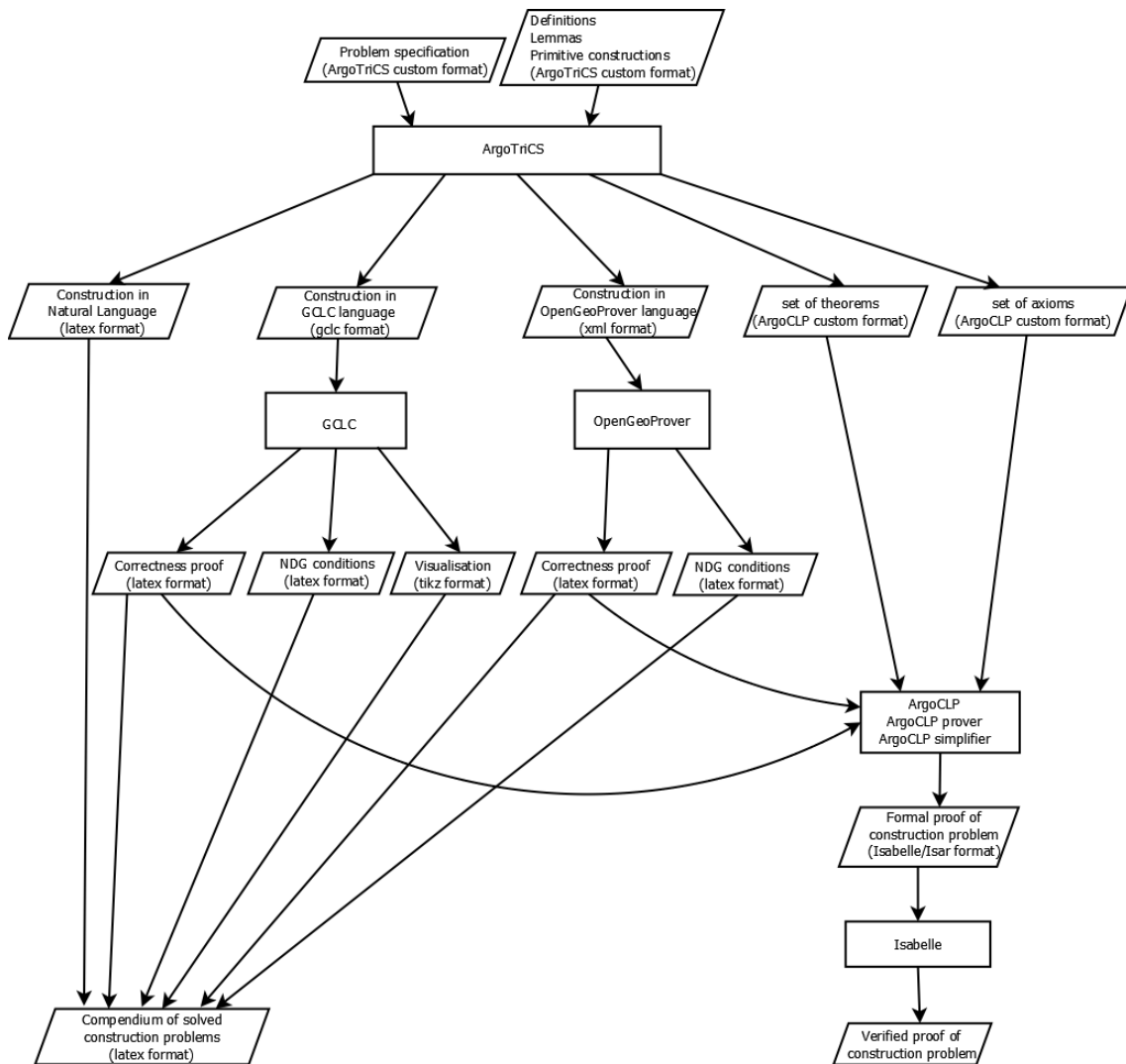
- аутоматско генерисање неформалног описа конструкције на природном језику,
- аутоматско генерисање формалне спецификације конструкције коришћењем језика GCLC [54] уз генерисање одговарајуће илустрације,
- доказивање исправности конструкције аутоматским доказивачима (доказивачем OpenGeoProver [66] и доказивачима који постоје у оквиру алата GCLC),
- генерисање улаза за фазу дискусије у којој се одређује при којим условима постоји решење проблема, као и колико решења проблем има,
- формализацију комплетног решења конструктивног проблема.

Главна намена овог алата јесте решавање што већег броја проблема из задатог корпуса проблема (под решавањем подразумевамо генерисање конструкције, као и комплетан доказ исправности). Да би се ово постигло, алат ArgoTriCS комуницира са алгебарским аутоматским и синтетичким доказивачима теорема, динамичким геометријским алатом за визуелизацију, као и са интерактивним доказивачима теорема. Систем ArgoTriCS би могао да буде од користи у математичком образовању, приликом учења уз помоћ рачунара, милионима студената који решавају ову врсту проблема, у ситуацијама када поред себе немају професора. Такође, на овај рад се може гледати и као на допринос управљању геометријским знањем и на први покушај идентификовања знања потребног за решавање проблема из задатог корпуса.

Схема веза система ArgoTriCS са алгебарским аутоматским доказивачима теорема, синтетичким аутоматским доказивачима теорема, интерактивним доказивачима теорема, као и динамичким геометријским алатима за визуелизацију дата је на слици ??.

4.2 Репрезентација знања

Као што је већ напоменуто у поглављу 2.2, главни проблем приликом решавања конструктивних проблема, и за човека и за рачунар, представља огроман



Слика 4.1: *Схема система ArgoTriCS*

простор претраге – постоји велики број расположивих конструктивних корака, а сваки од њих је могуће применити на велики број начина. Значајно је логирати, разумети и систематизовати знање које је релевантно за решавање конструктивних проблема или неке њихове поткласе. То би било од значаја како наставницима и студентима, тако и формирању базе математичког знања. Оваква систематизација би такође требало да води ка систему који аутоматски решава ове врсте проблема и требало би да буде од користи у образовању.

Предмет детаљне анализе ове тезе су Верников и Конелијев корпус, описаних у поглављу 2.2.4. У циљу што ефикаснијег решавања конструктивних проблема, детектовано је мало језгро геометријског знања потребно за решавање скоро свих проблема из ова два корпуса. Детектовано знање разврстано је на скуп дефиниција, лема и примитивних конструктивних корака и развијен је и једноставни алгоритам за аутоматско решавање проблема из ових корпуса. Пронађено знање минимално је у смислу да је сваки од елемената (дефиниција/лема/примитивних конструктивних корака) неопходан за решавање неког проблема из корпуса. Ово не значи да не постоји неко мање језгро знања које би било довољно за решавање истог скупа проблема.

У циљу генерисања што једноставнијих и интуитивнијих решења, која би била изражена на вишем нивоу апстракције, уместо елементарних операција помоћу лењира и шестара, биће коришћене сложене конструкцијске примитиве, као што је на пример конструкција средишта дужи. Све конструкцијске примитиве се, релативно једноставно, могу превести у термине елементарних операција помоћу лењира и шестара.

Пример 4.1. Вратимо се на мотивациони пример из поглавља 2.2 у којем је задатак био да се конструише троугао ABC ако су дата темена A и B и тежиште троугла G . Разматрано решење је следеће: конструисати средиште M_c дужи AB , а затим тачку C тако да важи $\overrightarrow{M_c G} : \overrightarrow{M_c C} = 1 : 3$. У овом решењу искоришћено је наредно знање:

- тачка M_c је средиште дужи AB (дефиниција тачке M_c),
- тачка G је тежиште троугла ABC (дефиниција тачке G),
- важи да је: $\overrightarrow{M_c G} = 1/3 \overrightarrow{M_c C}$ (лема),
- ако су дате тачке X и Y , могуће је конструисати средиште дужи XY (конструкцијска примитива),
- ако су дате тачке X и Y , могуће је конструисати тачку Z , тако да важи: $\overrightarrow{XY} : \overrightarrow{XZ} = 1 : k$ (конструкцијска примитива).

Приликом решавања конструктивних задатака углавном се не истиче приroda ових својстава. Ипак, раздвајање концепата на дефиниције, леме и конструкцијске примитиве требало би да буде од значаја не само за систематизацију математичког знања, већ и приликом процеса аутоматизације. Раздвајање концепата је од значаја и за развој добре педагошке стратегије која би се користила у образовном софтверу који се бави геометријским конструкцијама.

Постоје два начина на која је могуће задати компоненте знања (дефиниције, леме и конструкцијске примитиве): један је приступ у коме су компоненте у потпуности *инстанциране*, док је идеја другог приступа да не буду инстанциране, већ да буду *генеричке* и да се инстанцирају током фазе потраге за конструкцијом. На пример, могуће је задати инстанцирано својство да центар описаног круга троугла ABC припада медијатриси дужи AB , као $inc(circumcenter([a, b], c), side_bis(a, b))$ или задати генеричко својство да центар круга припада медијатриси произвољне тетиве тог круга, као $inc(circumcenter([X, Y], Z), side_bis(X, Y))$. У случају неинстанцираних компоненти знања током контрукције се врши инстанцирање променљивих константним термовима који одговарају конкретним објектима за фиксирани троугао ABC . На пример, могуће је инстанцирати променљиве X , Y и Z редом константним термовима a , $midpoint(b, c)$ и $foot(a, [b, c])$. Предност првог приступа је то што се инстанцираним знањем постиже висок степен контроле над објектима који се уводе током конструкције, али оваквог инстанцираног знања може бити превише и није проблем само у томе како набројати све дефиниције, леме и примитивне конструкције и како их систематизовати, већ и како их претраживати. Предност неинстанцираних компоненти знања је њихов мањи број и лакше одржавање. За сваку од компоненти знања разматране су обе могућности и направљен је одговарајући избор.

Што се задавања дефиниција тиче, да би се ограничио број објеката којима се током конструкције рукује, у систему ArgoTriCS одабран је приступ у коме су дефиниције свих тачака инстанциране. Праве и кругови се дефинишу над скупом тачака (као, на пример, права кроз две тачке или круг са центром у датој тачки кроз другу задату тачку), те се они дефинишу генеричким дефиницијама.

Разликујемо две врсте лема:

- оне које се односе на тачно један конкретан објекат, на пример, да тежиште G троугла ABC припада дужи CM_c – овакве леме је логично дати као *инстанциране*,
- оне које важе за све објекте који задовољавају неки предуслов – оваквих објеката може бити пуно, те је овакве леме погодније дати као *генеричке*.

Овакве леме су дате као универзално квантификована тврђења, у форми импликације.

Што се примитивних конструкција тиче, разматрана су оба приступа. Пример потпуно инстанциране конструкцијске примитиве би био:

- “ако је дато средиште странице AB – тачка M_c и тежиште троугла ABC – тачка G , онда се може конструисати теме C троугла ABC (јер су ове три тачке колинеарне и важи $\overrightarrow{CG} : \overrightarrow{CM_c} = 2 : 3$).

Јасно је да би оваквих конструкцијских примитива било јако пуно. Стога је разматрано како се ова инстанцирана конструкцијска примитива може извести и то је могуће урадити коришћењем:

- леме $\overrightarrow{M_cG} = 1/3 \overrightarrow{M_cC}$ и
- неинстанциране конструкцијске примитиве “ако су дате тачке X и Y , може се конструисати тачка Z , тако да важи: $\overrightarrow{XY} : \overrightarrow{XZ} = 1 : r$ ”.

Приступ у коме су све конструкцијске примитиве неинстанциране се чини једноставнијим и елегантнијим, те је он прихваћен у оквиру система ArgoTriCS. Проблем са ограничавањем скупа објеката који се контруишу генеричким конструкцијским примитивама решен је на начин описан у поглављу 4.3.3.

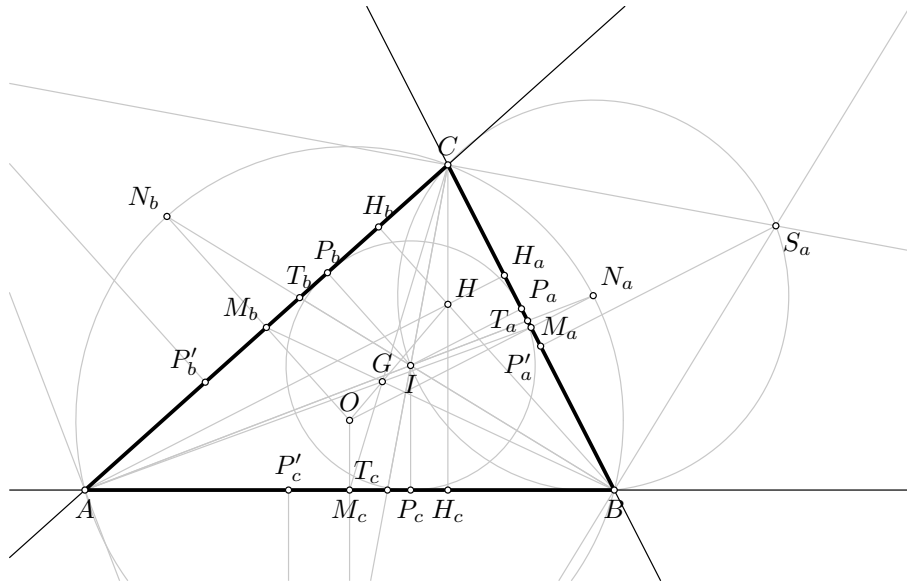
Дакле, за изградњу система ArgoTriCS, одабран је приступ у коме важи:

- све дефиниције тачака су инстанциране (док се неке праве и кругови уводе инстанцираним, а неке генеричким дефиницијама),
- постоје и инстанциране и генеричке леме,
- све конструкцијске примитиве су генеричке.

У циљу решавања проблема из Верниковог корпуса детектовано је знање потребно за решавање великог броја проблема из тог корпуса. Очекује се да је текућем знању потребно додати врло мало новог знања да би се успешно решили проблеми из неке нове фамилије конструктивних проблема. Ово се потврдило на примеру Конелијевог корпуса.

4.2.1 Дефиниције

Пре него што буде наведен скуп дефиниција које се користе у оквиру приказаног система, треба поменути да стандардна дефиниција тежишта (и сличних појмова) није одговарајућа. Наиме, тежиште се уобичајено дефинише као



Слика 4.2: Тачке које се користе у решењима проблема из Верниковог корпуса

пресечна тачка тежишних линија, али овај тип дефиниција у себи крије нетривијално својство да се све три тежишне линије секу у једној тачки. У оквиру система ArgoTriCS тежиште G троугла ABC дефинише се као пресек тежишних линија AM_a и BM_b (из Пашове аксиоме директно следи да овај пресек постоји), а својство да и трећа тежишна линија CM_c пролази кроз тежиште G дато је као лема. Дефиниције сличних појмова формулисане су у истом духу.

У Верниковој листи тројке проблема (A, B, G) и (A, C, G) сматране су аналогним и из тог разлога друга тројка није ни била приказана у листи. За разлику од Верникове листе, због приступа који ми следимо, тројке проблема (A, B, G) и (A, C, G) биће сматране различитим. У оквиру система ArgoTriCS за сваки од проблема биће разматрано да ли је симетрична у односу на дефиниције (у ознаци S_d) са неким од претходних проблема или у односу на дефиниције и леме (у ознаци S_{dl}). На пример, проблеми (B, M_a, G) и (A, M_b, G) симетрични су у односу на дефиниције, а проблеми (A, B, G) и (A, C, G) симетрични су у односу на дефиниције и леме.

За решавање проблема из Верникове листе у оквиру система ArgoTriCS користи се 25 значајних тачака троугла илустрованих на слици 4.2, а које су набројане у наставку, заједно са својим дефиницијама (запис одговара Верниковој нотацији):

D1 O (центар описаног круга): пресек медијатриса дужи BC и AC ;

D2 I (центар уписаног круга): пресек бисектриса унутрашњих углова AT_a и

BT_b ;

D3 H (ортоцентар): пресек висина AH_a и BH_b ;

D4 G (тежиште): пресек тежишних дужи AM_a и BM_b ;

D5–D7 H_a, H_b, H_c : пресеци висина троугла са наспрамним страницама троугла;

D20–D22 M_a, M_b, M_c (средишта страница): тачке за које важи $\overrightarrow{BM_a}/\overrightarrow{BC} = 1/2, \overrightarrow{CM_b}/\overrightarrow{CA} = 1/2, \overrightarrow{AM_c}/\overrightarrow{AB} = 1/2$;

D23–D25 T_a, T_b, T_c : пресеци бисектриса унутрашњих углова са наспрамним страницама троугла;

D47–D49 N_a, N_b, N_c : пресеци описаног круга троугла и бисектриса унутрашњих углова AT_a, BT_b и CT_c , редом;

D91–D93 T'_a, T'_b, T'_c : пресеци бисектриса спољашњих углова са наспрамним страницама троугла;

D94–D96 P_a, P_b, P_c : подножја из центра уписаног круга I редом на странице BC, AC и AB ;

D97–D99 P'_a, P'_b, P'_c : подножја из центара споља описаних кругова редом на странице BC, AC и AB .

Све наведене дефиниције су у потпуности инстанциране.

Поред њих постоје и наредне генеричке дефиниције:

GD01 XY означава праву која садржи тачке X и Y ;

GD02 $k(X, Y)$ означава круг са центром у тачки X који садржи тачку Y .

Из списка дефиниција изостављене су инстанциране дефиниције правих и кругова који се користе у програму. Такође, нису наведене ни (инстанциране, конкретне) тачке за које се обично сматра да нису значајне, али су потребне током конструкције, као што су, на пример, друга пресечна тачка висине и описаног круга у троуглу (различита од темена) ¹.

За потребе решавања проблема из Конелијевог корпуса потребне су још и следеће тачке:

¹Ове дефиниције одговарају бројевима који недостају у нумерацији горе наведених дефиниција

D28–D30 E_a, E_b, E_c (Ојлерове тачке): тачке за које важи $\overrightarrow{AE_a}/\overrightarrow{AH} = 1/2$, $\overrightarrow{BE_b}/\overrightarrow{BH} = 1/2$, $\overrightarrow{CE_c}/\overrightarrow{CH} = 1/2$;

D31 N (центар Ојлеровог круга): тачка за коју важи $\overrightarrow{HN}/\overrightarrow{HO} = 1/2^2$;

D100–D102 P_a^1, P_b^1, P_c^1 : пресеци висина са кругом који има центар у одговарајућој Ојлеровој тачки и који садржи средиште одговарајуће странице троугла.

4.2.2 Леме

Нека релевантна геометријска својства проистичу из дефиниција, а нека не – оваква својства зваћемо лемама.

Систем ArgoTriCS за решавање проблема из Верникове листе користи 96 инстанцираних и 20 неинстанцираних лема и њихов комплетан списак дат је у додатку В. Следе примери инстанцираних лема:

- тачка G припада дужи CM_c ;
- тачке A и B припадају кругу $k(O, C)$;
- $\overrightarrow{AG}/\overrightarrow{AM_a} = 2/3$, $\overrightarrow{BG}/\overrightarrow{BM_b} = 2/3$, $\overrightarrow{CG}/\overrightarrow{CM_c} = 2/3$;
- $\angle HAI = \angle IAO$, $\angle HBI = \angle IBO$, $\angle HCI = \angle ICO$;
- праве AH , BH и CH су бисектрисе унутрашњих углова троугла $H_aH_bH_c$.

Примери неинстанцираних лема које се користе за решавање проблема из Верниковог корпуса су:

- Центар круга припада медијатриси произвољне тетиве тог круга;
- Ако важи $\overrightarrow{XY}/\overrightarrow{XZ} = r$, тачка Z припада правој p , а тачка Y јој не припада, онда тачка X припада правој која је слика праве p при хомотетији са центром у тачки Y са коефицијентом $r/(r-1)$;
- Ако важи $\alpha = \beta/2^{L_1}$ и $\gamma = K_1 \cdot \beta/2^{K_2} + K_3 \cdot \pi/2^{K_4}$ онда важи и $\gamma = K_1 \cdot \alpha/2^{K_2-L_1} + K_3 \cdot \pi/2^{K_4}$;

²Неке од овде наведених дефиниција нису дефиниције које се срећу у уџбеницима геометрије, већ представљају еквивалентне карактеризације тих појмова, подесније за контекст рачунарски подржаног решавања конструктивних проблема.

За потребе решавања проблема из Конелијевог корпуса потребно је додати још 23 инстанциране леме и оне су наведене у додатку В.

Укупно, за решавање проблема из Верниковог и Конелијевог корпуса, потребно је 119 инстанцираних и 20 неинстанцираних лема. Многе од ових лема су тривијалне, а веома често постоје међусобно сличне леме³. Оне се најчешће уче у основној или средњој школи на првим лекцијама из “геометрије троугла” и могу се доказати коришћењем Хилбертових аксиома геометрије а и аутоматски - коришћењем аутоматских доказивача за геометрију.

Један од планова за даљи рад је да се све леме које се користе у оквиру система ArgoTriCS докажу коришћењем неког аксиоматског система.

4.2.3 Примитивне конструкције

Као што је већ напоменуто, све примитивне конструкције су задате у неинстанцираној форми и оне се инстанцирају приликом потраге за конструкцијом. За сваку од примитивних конструкција дат је:

- скуп *услова недегенерисаности* – то су услови који обезбеђују да конструисани објекти заиста постоје,
- скуп *услова одређености* – то су услови при којима конструисани објекти јединствено постоје.

Примитивних конструкција потребних за решавање проблема из Верниковог корпуса има 20 и набројане су у додатку В, а неке од њих су:

- Ако су дате тачке X , Z и W , и рационалан број r могуће је конструисати тачку Y тако да важи: $\overrightarrow{XY}/\overrightarrow{ZW} = r$;
- Ако су дати права и круг, могуће је конструисати њихове заједничке тачке; услов недегенерисаности је да се права и круг секу;
- Ако су дати права и круг и једна њихова заједничка тачка, могуће је конструисати њихову другу заједничку тачку; услов недегенерисаности је да се права и круг секу;
- Ако су дате тачке X и Y и угао α могуће је конструисати скуп тачака из којих се дуж $\overline{XY}$ види под углом $A \cdot \alpha/2^B + C \cdot \pi/2^D$.

³Ако би требало да класификујемо леме по тежини, могли бисмо да кажемо да их има око 30 тежих, око 30 средње тежине, док су остале тривијалне. Од ових 30 тежих лема многе су међусобно повезане и често једна следи из друге. Теже леме се углавном тичу односа углова који важе у троуглу, на пример да важи: $\angle T_c I T_b = \angle BAC/2 + \pi/2$.

Подразумева се да за примитивну конструкцију не постоје услови недегенерисаности и услови одређености, уколико услови нису експлицитно наведени.

За потребе решавања проблема из Конелијевог корпуса потребне су још 3 примитивне конструкције.

До датог списка потребних дефиниција, лема и примитивних конструкција дошло се анализом решења појединачних проблема. Анализа је спроведена у итерацијама. Ручно су анализирана решења првих неколико проблема из датог корпуса пронађена у литератури и знање потребно за њихово решавање додато је у систем. Коришћењем тог знања системом ArgoTriCS покушано је да се реше сви проблеми из корпуса. Од проблема које на овај начин нису решене проучавана су решења њих неколико и додатно детектовано знање додато је у систем, итд.

4.3 Преглед алгоритма

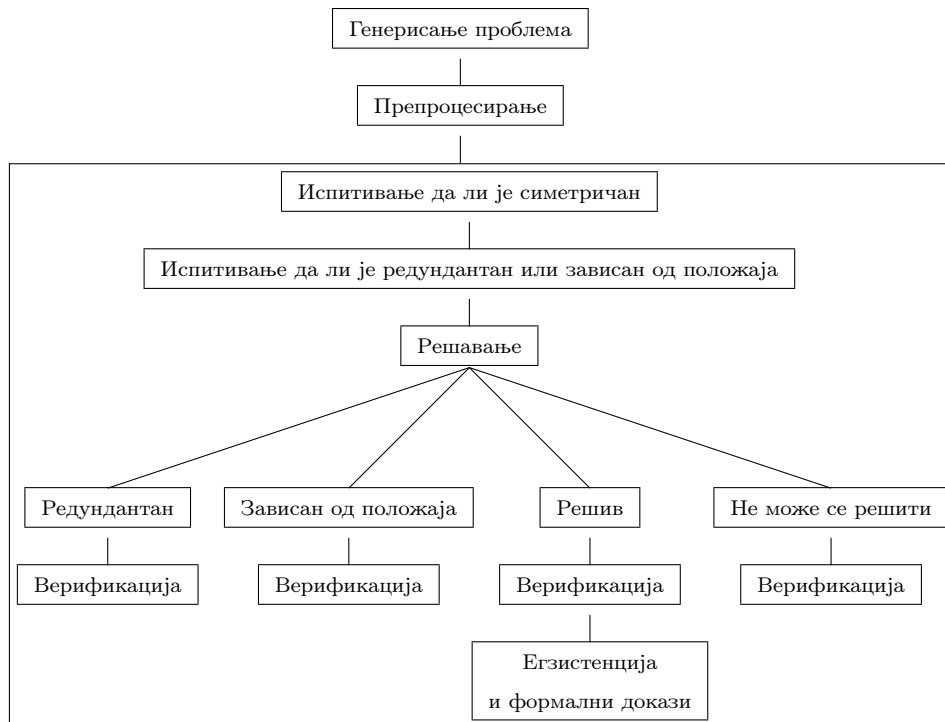
Нека је дата фамилија конструктивних проблема које треба решити. Решавање проблема из ове фамилије састоји из наредних етапа:

- генерисање свих проблема на основу скупа улазних елемената,
- фаза препроцесирања,
- фаза решавања појединачних проблема из ове фамилије која укључује:
 - испитивање да ли је проблем симетричан,
 - испитивање да ли је проблем редундантан,
 - испитивање да ли је проблем зависан од положаја,
 - генерисање конструкције,
 - доказивање исправности конструкције,
 - утврђивање када проблем има решење и колико има решења.

На слици [4.3](#) приказан је ток извршавања алгоритма.

4.3.1 Генерисање проблема

За сваки од корпуса чије проблеме треба решити задаје се скуп улазних елемената, а затим се над тим скупом формирају све различите тројке проблема. Скуп улазних елемената Конелијевог корпуса представља проширење



Слика 4.3: Алгоритам

скупа улазних елемената Верниковог корпуса (пored тачака из Верниковог корпуса укључује и тачке E_a, E_b, E_c, N), али се, у складу са Конелијевом листом проблема [25], све тројке проблема које постоје у Верниковом корпусу изузимају из Конелијевог корпуса.

4.3.2 Фаза препроцесирања

Пре почетка процеса решавања проблема, извршава се фаза препроцесирања. Она се извршава само једном и то увек пре фазе решавања проблема.

Током фазе препроцесирања врши се инстанцирање генеричких дефиниција свим релевантним објектима и оне се додају у базу знања као изведене дефиниције. Током ове фазе врши се и инстанцирање генеричких лема објектима који задовољавају предуслове дате леме. Притом се инстанцирани закључци додају у базу знања као изведене леме. Инстанциране дефиниције се растављају на појединачна својства, при чему се чува информација из које дефиниције потиче својство.

У току фазе препроцесирања се, такође, на основу расположивог скупа дефиниција, формирају дефиниције свих појмова у GCLC формату, као и у XML формату који је потребан за OpenGeoProver, а такође и аксиоме у ArgoCLP формату потребне за фазу формалне верификације добијених решења.

4.3.3 Фаза решавања

Након фазе препроцесирања прелази се на решавање проблема из дате фамилије.

Испитивање да ли је проблем симетричан

За сваки од проблема се, пре него што се покуша његово решавање, прво врши испитивање да ли је симетричан са неким већ решаваним проблемом (при томе проблем може бити симетричан само у односу на дефиниције или у односу на дефиниције и леме, као што је објашњено у поглављу 4.2.1). Два проблема су симетрична у односу на дефиниције уколико се применом супституције на скуп задатих објеката првог проблема за неку од пермутација скупа темена $\{A, B, C\}$ добија скуп задатих објеката другог проблема. На пример, проблеми: $A, B, M_a = \text{midpoint}([B, C])$ и $A, B, M_b = \text{midpoint}([A, C])$ су симетрични у односу на дефиниције јер се супституцијом скупа задатих објеката првог проблема за пермутацију (B, A, C) добијају исти скупови објеката. Два проблема су симетрична у односу на дефиниције и леме уколико се применом супституције на скуп задатих објеката првог проблема за неку од пермутација скупа темена $\{A, B, C\}$ добија скуп објеката који је једнак у односу на дефиниције и леме. На пример, проблеми: $A, B, O = \text{circumcenter}([A, B], C)$ ⁴ и $A, C, O = \text{circumcenter}([A, B], C)$ су симетрични у односу на дефиниције и леме јер се применом супституције за пермутацију (A, C, B) на први скуп објеката добија скуп објеката $A, B, \text{circumcenter}([A, C], B)$, а за тачке $\text{circumcenter}([A, C], B)$ и $\text{circumcenter}([A, B], C)$ се једноставно може показати да су једнаке јер припадају двома истим правама. Ако је проблем симетричан са неким већ решаваним проблемом, та информација се памти. Тренутно се сви проблеми решавају изнова, али би било могуће искористити информацију са којим проблемом је дати проблем симетричан и његово решење у циљу ефикаснијег решавања нових проблема.

Испитивање да ли је проблем редундантан

Затим се испитује да ли је проблем редундантан. Проблем се сматра *редундантним* ако се један од задатих елемената може конструисати на основу друга два. Ако се утврди да је проблем редундантан, неко од темена које још увек

⁴У запису овог предиката различито су задате тачке A и B од тачке C јер се центар описаног круга дефинише као пресек медијатриса страница BC и AC , а својство да медијатриса треће странице AB пролази кроз ову тачку је дато као лема.

није конструисано конструише се на произвољан начин, а затим се, уколико проблем још увек није решен, наставља његово решавање на уобичајен начин.

Испитивање да ли је проблем зависан од положаја

Уколико проблем није редундантан, онда се врши провера да ли неки од задатих елемената мора да припада неком скупу тачака (правој или кругу) који је одређен са друга два улазна елемента. Ако ово важи, онда се тај проблем сматра *зависним од положаја* и конструкција се наставља на следећи начин: тај елемент се на произвољан начин бира тако да припада том скупу тачака. У ове сврхе користимо конструктивне кораке облика: “одабери на случајан начин тачку дате праве” и “одабери на случајан начин тачку датог круга”. Затим се процес решавања полазног проблема наставља на уобичајен начин.

Генерисање конструкције

Ако проблем није нити редундантан нити зависан од положаја, почиње потрага за конструкцијом. Циљ процедуре претраге јесте да се конструишу сви објекти које се траже. На пример, за све проблеме из Верникове листе циљ је исти: конструисати троугао ABC , тј. тачке A , B и C . Као што је већ поменуто, примитивне конструкције су дате у општој, неинстанцираној форми и инстанцирају се током потраге за конструкцијом на следећи начин: ако постоји инстанцирање тако да су сви објекти из премиса примитивне конструкције већ конструисани (или дати спецификацијом проблема) онда се конструишу инстанцирани објекти из закључка, уколико већ нису конструисани. Ипак, дозвољено је конструисати само ограничен скуп објеката, као што ће бити објашњено у наставку, у поглављу 4.3.3. Процедура претраге је итеративна – примитивне конструкције су дате у неком редоследу и у сваком кораку покушава се да се нека од њих примени на познате објекте (дате спецификацијом проблема или већ конструисане) и ако се у томе успе, потрага креће испочетка од прве примитивне конструкције, по принципу водопада. Ако се нека примитивна конструкција не може применити, покушава се примена наредне примитивне конструкције. Ако се сви тражени објекти конструишу, претрага се зауставља и полазни проблем је *решив*. Иначе, уколико се не може применити ниједна примитивна конструкција, процедура се зауставља са неуспехом и проблем *није решив* коришћењем датог геометријског знања.

Током решавања проблема одржава се скуп свих коришћених дефиниција, лема и примитивних конструкција у генерисаним конструкцијама, те се након решавања свих проблема из задатог корпуса може прецизно утврдити које је

тачно знање било потребно за решавање свих решених проблема из тог корпуса. Такође, уз генерисану конструкцију чува се и скуп услова недегенерисаности употребљених примитивних конструкција.

Ефикасност решавања, а такође и само решење, може да зависи од редоследа у коме су задате примитивне конструкције и ово ће бити илустровано у поглављу 4.6. Ефикасност основне процедуре може се побољшати коришћењем неке од додатних техника, које су дате у наставку. Чак и са овако ограниченом и усмереном потрагом, током потраге за конструкцијом генеришу се неки ирелевантни конструкцијски кораци. Када се тражена конструкција пронађе, сви ирелевантни кораци се елиминишу из крајње “чисте” конструкције.

Ограничавање објеката који се могу конструисати Објекти које је могуће конструисати су ограничени на скуп објеката који су релевантни за неки од објеката који још увек није конструисан, тј. на објекте који се јављају у дефиницији објекта који још увек није конструисан или у лема у којој се јавља објекат који још увек није конструисан.

На пример, размотримо примитивну конструкцију која гласи да је за две дате тачке могуће конструисати средиште дужи коју оне одређују. Ако не би било никаквих ограничења, средиште дужи би се конструисало за сваке две могуће тачке, док се многа од њих не би користила нигде даље; штавише конструисана средишта би се даље разматрала као крајеви дужи чије средиште треба конструисати. Да би се избегао овај нежељени ефекат, ово правило биће примењено само на дужи за које важи да се њихово средиште јавља у некој од дефиниција или лема које укључују објекат који до тада није конструисан, а дефинисан је у оквиру почетног, фиксног скупа објеката релевантних за неки од проблема из корпуса.

На пример, ако би се тачка M_c јављала само у дефиницији медијатрисе дужи AB , у оквиру својства $inc(M_c, side_bis(A, B))$, и ако би медијатриса ове дужи већ била конструисана, онда не би било користи од конструисања тачке M_c . Уколико би се тачка M_c јављала и у оквиру леме $sratio(A, M_c, A, B, 1, 2)$ и нека од тачака A и B не би била конструисана, онда би могло бити од користи конструисати тачку M_c .

С друге стране, средиште дужи AM_c се не користи ни у једној од дефиниција и лема и зато га никада не конструирамо. Оправдање за увођење оваквог ограничења је добијено као резултат анализе извесног броја проблема и касније је потврђено успешним решавањем великог броја проблема (иако нема гаранције да, на пример, средиште дужи AM_c не може бити

од користи у неком решењу неког конструктивног задатка).

Ово ограничење може значајно смањити време претраге, као и дужину генерисане конструкције.

Поредак примитивних конструкција Приликом утврђивања редоследа у коме треба задати примитивне конструкције користи се следећа хеуристика: примитивне конструкције су поређане у таквом редоследу да се конструкцијске примитиве које се чешће користе налазе пре оних које се користе само за неколико инстанци конструктивних проблема. Конструкцијске примитиве које се чешће користе су идентификоване прелиминарним експериментисањем са различитим могућим уређењима и пребројавањем броја пута колико се свака од њих користила. Наравно, у финалном систему, ово пребројавање не мора да да исти резултат, али нам служи да раздвоји она правила која се често користе од оних која се користе врло ретко.

Рано одсецање неприменљивих примитивних конструкција Приликом потраге за применљивом примитивном конструкцијом и тестирања да ли је применљива нека од примитивних конструкција која се у листи примитивних конструкција налази испред оне која је искоришћена у претходном конструкцијском кораку, користи се наредно ограничење које омогућава рано одсецање неких грана претраге, пре него што се потпуно провери применљивост неке конструкцијске примитиве: барем један од објеката из листе познатих објеката те примитивне конструкције треба да буде инстанциран објектом који је конструисан у претходном конструкцијском кораку. Овакво закључивање је исправно јер те примитивне конструкције нису биле применљиве пре претходног конструкцијског корака (на објекте који су до тада били конструисани), те су могле да постану применљиве само за новоконструисане објекте.

На пример, ако је последња примењена примитивна конструкција била конструкција C_i којом се конструише пресечна тачка два позната круга и ако овој конструкцијској примитиви претходи конструкцијска примитива C_j којом се конструише пресечна тачка две познате праве, онда након примене конструкцијске примитиве C_i нема потребе испитивати применљивост конструкцијске примитиве C_j , с обзиром на то да она није била применљива пре последњег корака конструкције, а ниједна нова права није у међувремену конструисана.

Подршка различитим начинима задавања истог објекта Приликом испитивања да ли инстанцирани објекат из закључка примитивне конструкције није већ конструисан, проверавају се различите репрезентације тог објекта. На пример, ако се премисе примитивне конструкције могу задовољити и њом се може конструисати права која је слика праве AB при хомотетији са центром у тачки C са коефицијентом $1/2$, а права M_aM_b је већ конструисана, с обзиром на то да је слика тачке A при датом пресликавању тачка M_b , а слика тачке B тачка M_a , процедура ће закључити да ова примитивна конструкција није применљива.

Доказивање исправности конструкције

Након генерисања конструкције за дати проблем следи фаза верификације, којом се показује да ако се објекти конструишу на дати начин, онда они задовољавају спецификацију проблема.

Често се за нека решења конструктивних проблема тврди да су “исправна на основу конструкције”. У неким случајевима то може бити тачно, али то не важи у општем случају. У наставку ће бити приказана два решења истог конструктивног проблема на којима ћемо илустровати ову тврдњу.

Вратимо се поново на мотивациони пример из поглавља 2.2. Задатак је конструисати троугао ABC ако су дата темена A и B и тежиште троугла G . Размотрићемо два различита решења овог проблема, илустрована на сликама 4.4 и 4.5.

Конструкција 1:

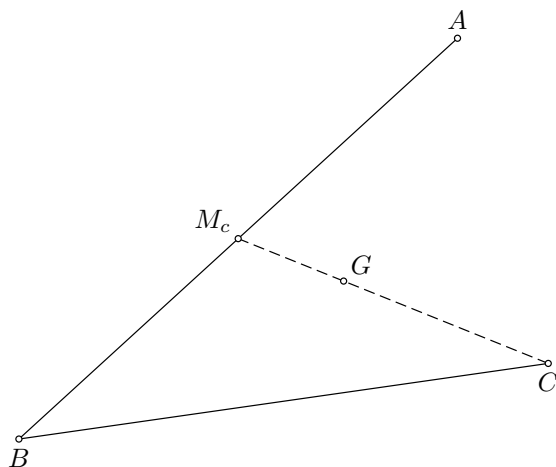
1. конструисати тачку M_c тако да важи $\overrightarrow{AM_c} : \overrightarrow{AB} = 1 : 2$,
2. конструисати тачку C тако да важи $\overrightarrow{M_cG} : \overrightarrow{M_cC} = 1 : 3$.

Доказ 1:

Треба показати да је тачка G тежиште троугла ABC . На основу конструкције тачка M_c је средиште дужи AB . На основу конструкције и познате леме, с обзиром на то да важи $\overrightarrow{M_cG} : \overrightarrow{M_cC} = 1 : 3$, тачка G је тежиште троугла ABC .

Конструкција 2:

1. конструисати тачку M_a тако да важи $\overrightarrow{AM_a} : \overrightarrow{AG} = 3 : 2$,
2. конструисати тачку M_b тако да важи $\overrightarrow{BM_b} : \overrightarrow{BG} = 3 : 2$,
3. конструисати тачку C као пресечну тачку правих AM_b и BM_a .

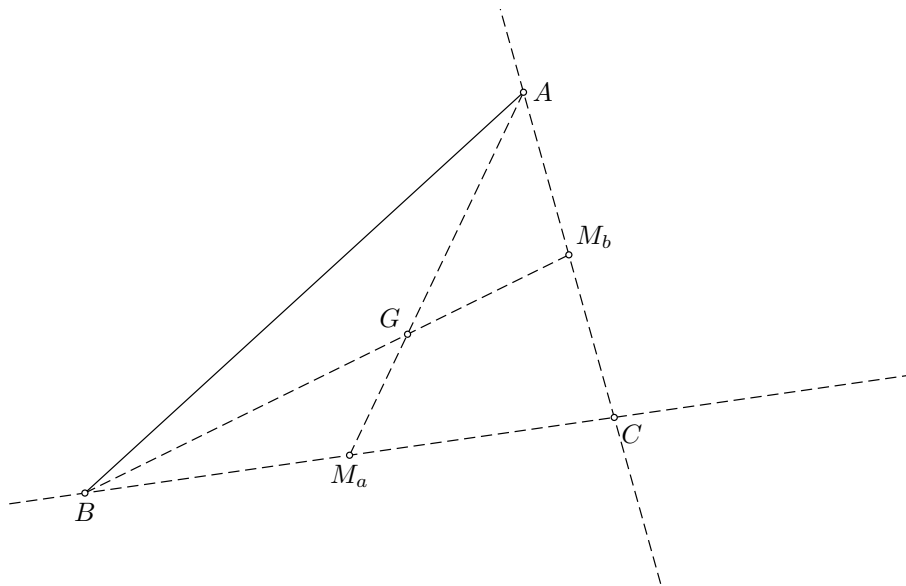


Слика 4.4: Илустрација једне могуће конструкције за проблем A, B, G

Доказ 2:

Треба показати да је тачка G тежиште троугла ABC . За разлику од претходне конструкције, из ове не следи директно да је тачка M_a средиште дужи BC , нити да је тачка M_b средиште дужи AC . Нека је тачка G' тежиште троугла ABC . На основу познате леме важи да тежиште припада правој која се добија као слика праве AC при хомотетији са центром у тачки B са коефицијентом $2/3$. Слично важи да тачка G' припада правој која се добија као слика праве BC при хомотетији са центром у тачки A са коефицијентом $2/3$. На основу конструкције тачка G такође задовољава ове услове. Ако тачке A , B и C нису колинеарне (то јест тачке A , B и G нису колинеарне), ова тачка је јединствено одређена, па важи да су тачке G и G' једнаке, односно тачка G је заиста тежиште троугла ABC .

Прво решење датог конструктивног проблема може се сматрати исправним на основу конструкције, док се за друго то не може тврдити. Проблем се може илустровати решавањем једначина у којем се примењује резоновање према еквиваленцији, тј. изводи се низ једначина $E_0, E_1, \dots, E_n$, при чему је E_0 једначина коју треба решити, а E_n канонска једначина коју је лако решити, при чему важи: $E_0 \Leftrightarrow E_1 \Leftrightarrow \dots \Leftrightarrow E_n$. У овом случају решења су исправна на основу конструкције. У великом броју случајева приликом решавања конструктивних проблема у геометрији не може се применити овакво резоновање. У неком тренутку може се резонovati само према потребним условима, односно добија се низ импликација: $E_0 \Rightarrow E_1 \Rightarrow \dots \Rightarrow E_n$, при чему је једначина E_n лака за



Слика 4.5: Илустрација друге могуће конструкције за проблем A, B, G

решавање. Стога су нека од решења за E_n “лажна решења” јер нису решења и почетног проблема E_0 .

Из овога следи да је за генерисане конструкције у општем случају неопходно показати њихову исправност. У ове сврхе користе се доказивачи који су имплементирани у оквиру алата GCLC (постоји подршка за три методе за аутоматско доказивање теорема у геометрији: метода површина, Вуова метода и метода Гребнерових база) [54] и Вуова метода имплементирана у оквиру доказивача OpenGeoProver [66]. Ови доказивачи као излаз дају аргументе тачности тврђења (који не дају увек потпуни доказ), као и услове недегенерисаности. Тврђење које се доказује јесте да су тачке које су задате поставком проблема (различите од темена троугла ABC) заиста одговарајуће тачке конструисаног троугла ABC . Одатле следи да конструисани троугао задовољава спецификацију датог конструктивног проблема.

Фаза дискусије

За решиве проблеме се, након верификације, врши испитивање да ли увек (а уколико то није случај када) постоје конструисани објекти и колико проблем има решења. Као што је већ напоменуто, уз конструкцију се одржава и скуп услова недегенерисаности примитивних конструкција и он се користи за утврђивање када конструисани објекти заиста постоје. Што се броја решења тиче, он се рачуна као производ броја могућих начина на које је могуће конструисати сваки од објеката који је конструисан током конструкције. На пример, ако се

тачка конструише као пресечна тачка два круга који се секу, онда постоје два различита начина како је конструисати. У случају када се тачка конструише као произвољна тачка на правој или кругу, постоји бесконачан број могућности.

4.4 Имплементација

Систем ArgoTriCS имплементиран је у програмском језику PROLOG и, заједно са делом кода којим се задаје полазно геометријско знање, има укупно око 3000 линија кода, организованих у 10 изворних датотека⁵. Код је организован на следећи начин: у једној датотеци дат је скуп свих дефиниција, у другој датотеци скуп свих лема, а у посебној датотеци скуп свих примитивних конструкција. На овај начин лако се може изменити полазно геометријско знање. Постоји модул који задаје све потребне предикате за фазу препроцесирања, а посебно је издвојен модул за утврђивање статуса једног конкретног конструктивног проблема. Такође је, у посебној датотеци, издвојен део кода који се тиче генерисања свих потребних фајлова за улаз у GCLC, а посебно део који одговара генерисању улаза у OpenGeoProver. Такође, генерисање свих потребних фајлова за добијање формалних доказа у Isabelle формату издвојено је у посебну датотеку.

Полазно геометријско знање задаје се скупом инстанцираних и генеричких дефиниција, скупом инстанцираних и генеричких лема, као и скупом примитивних конструкција. Инстанциране дефиниције као аргументе имају потпуно инстанцирани објекат који се дефинише, листу својстава којима се објекат дефинише и назив дефиниције. На пример, центар описаног круга троугла ABC дефинише се као пресек медијатриса страница BC и AC на следећи начин:

```
instantiated_def(circumcenter([a,b],c),
                [inc(circumcenter([a,b],c),side_bis([b,c])),
                 inc(circumcenter([a,b],c),side_bis([a,c]))],
                'D1').
```

Генеричке дефиниције као аргументе садрже: објекат који се дефинише и који није потпуно инстанциран, листу својстава којима се објекат дефинише, листу објеката који треба да буду релевантни и назив дефиниције. Пример генеричке дефиниције којом се дефинише круг са датим центом кроз дату тачку био би:

⁵Изворни код програма доступан је на адреси: <http://argo.matf.bg.ac.rs/?content=downloads>.

```
generic_def(circle_center_through(0,A),
            [inc_k(A,circle_center_through(0,A)),
             center(0,circle_center_through(0,A))],
            [obj(circle,circle_center_through(0,A))],
            'GD02').
```

Инстанциране леме као аргументе имају тврђење леме и њен назив. На пример, лема којом се тврди да и медијатриса странице AB садржи центар описаног круга има следећи формат:

```
instantiated_lemma(inc(circumcenter([a,b],c),side_bis([a,b])),
                  'L1').
```

Генеричке леме дефинишу се следећим елементима: скупом премиса леме, њеним закључком, листом објеката чије вредности треба израчунати, скупом објеката који треба да буду релевантни и називом дате леме. На пример, лема којом се тврди да ако важи $\overrightarrow{XY}/\overrightarrow{XZ} = K/L$ и ако тачка Z припада правој p , а тачка Y јој не припада, онда тачка Z припада правој која се добија хомотетијом са коефицијентом $K/(K - L)$ праве p у односу на тачку Y , при чему права p мора бити релевантна, задаје се на следећи начин:

```
generic_lemma([sratio(X,Y,X,Z,K,L),inc(Z,P),not_true(inc(Y,P))],
              inc(X,homothety(Y,P,K1,L1)),
              [(K1,K),(L1,K-L)],
              [obj(line,P)],
              'GL02').
```

Примитивне конструкције се задају скупом познатих објеката, скупом вредности, објектом који се конструише, својствима која треба да важе да би се објекат могао конструисати, условима недегенерисаности и условима одређености, као и називом примитивне конструкције. На пример, конструкција пресечне тачке две дате праве се задаје на следећи начин:

```
basic_cons([obj(line,P), obj(line,Q)], [],
            [obj(point,A)],
            [P!=Q, inc(A,P), inc(A,Q)],
            [not_parallel(P,Q)],
            [not_true(identical_lines(P,Q))],
            'W03').
```

Предикат који чини срж механизма потраге за конструкцијом је предикат `update_known` и он као аргументе прима:

- листу познатих објеката,
- листу објеката које треба конструисати,
- листу већ изведених конструкција (парцијална конструкција),
- променљиву у коју се смешта комплетна листа конструкција када предикат успе,
- променљиве које се користе као историја услова недегенерисаности и услова одређености коришћених конструкција,
- идентификатор претходно искоришћене примитивне конструкције, као и објекат који је последњи конструисан.

Предикат користи репну рекурзију и ради по следећем принципу: позива се предикат којим се задаје нека примитивна конструкција са новим променљивим и конструкција се примењује ако су испуњени следећи услови:

- објекти који треба да буду познати у конструкцији налазе се у листи познатих објеката,
- својства која се користе у конструкцији су познате чињенице генерисане у фази препроцесирања,
- конструисани објекти су релевантни за проблем.

Ако су испуњени сви услови ажурирају се листе познатих објеката, објеката који се траже, својстава која се користе и предикат се рекурзивно позива.

Идентификатор претходно примењене примитивне конструкције, као и претходно конструисани објекат су потребни у циљу убрзања процедуре претраге (видети поглавље [4.3.3](#)).

За проверу да ли је конструктивни проблем редувант користи се предикат који је сличан претходном предикату, осим што се у случају када се утврди да је један од улазних елемената за конструктиван проблем могуће конструисати на основу преостала два неко од још неконструисаних темена троугла конструише на слободан начин и процедура наставља са претрагом.

Провера да ли је проблем зависан од положаја је нешто компликованија - ако утврдимо да је један од улазних елемената за задати проблем припада неком скупу тачака (правој или кругу) одређеном са друга два улазна елемента

за проблем, онда се тај објекат конструише на случајан начин тако да припада тој правој или кругу. У случају да се у неком моменту не може конструисати ниједан нови објекат, а неко од темена троугла и даље није конструисано, проналази се нека од конструисаних правих или кругова којима то теме припада и конструише на случајан начин тако да припада тој правој/кругу.

4.5 Пропратни резултати

Као што је већ наведено, приликом фазе препроцесирања врши се инстанцирање свих генеричких лема и при томе се добијају инстанциране леме које могу садржати различита интересантна својства. Уз сваку инстанцирану лему чува се и информација из којих је дефиниција и генеричких лема настала. Оваква интересантна својства записана су у оквиру описа конструктивног корака формалне спецификације конструкције на GCLC језику. Могуће је притом добити и нека својства која обично нису заступљена у литератури. Следи један такав пример.

Пример 4.2. *Важи да је права AO паралелна са медијатрисом дужи H_bH_c .*

Ово се може извести на следећи начин:

- тачка E_a је центар круга коме припадају тачке H_b и H_c ;
- тачка N је центар Ојлеровог круга коме припадају тачке H_b и H_c ;
- тачка E_a припада медијатриси дужи H_bH_c , јер центар круга припада медијатриси произвољне тетиве;
- тачка N припада медијатриси дужи H_bH_c , јер центар круга припада медијатриси произвољне тетиве;
- важи: $\overrightarrow{HE_a}/\overrightarrow{HA} = 1/2$;
- важи: $\overrightarrow{HN}/\overrightarrow{HO} = 1/2$.

Одавде следи да је права E_aN паралелна правој AO , а то, у ствари, даје да је медијатриса дужи H_bH_c паралелна правој AO .

За наведени пример, поменути закључак наведен је у генерисаној конструкцији на GCLC језику за проблем 582, илустрованој на слици 4.6, на следећи начин:



Слика 4.6: Аутоматски генерисана илустрација за проблем 582

```
% Constructing a line m(H_{b}H_{c}) which contains the point E_{a}
% and is parallel to the line A0
parallel m(H_{b}H_{c}) E_{a} A0
```

```
color 200 200 200
drawline m(H_{b}H_{c})
color 0 0 0
```

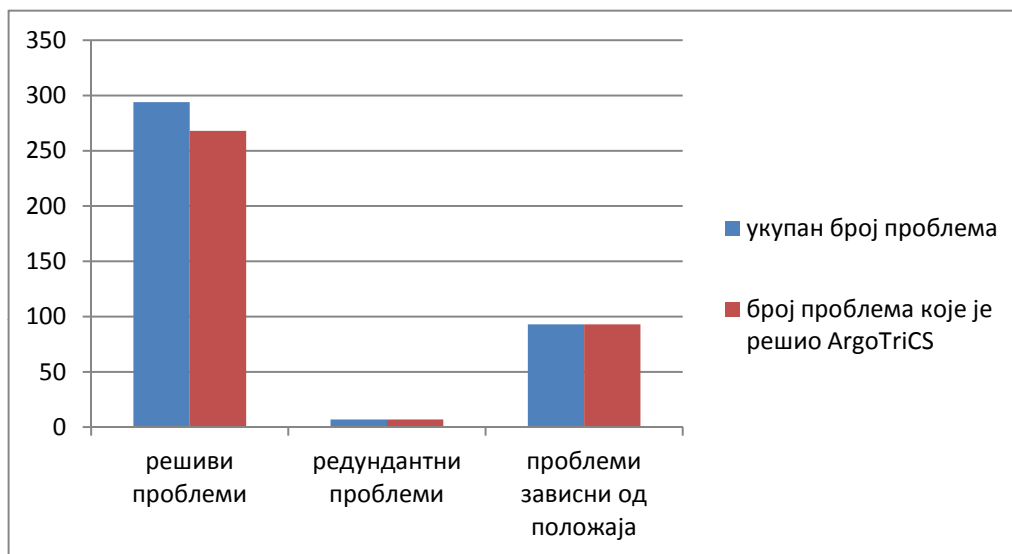
из чега се директно види да је као лема изведено својство да је медијатриса дужи $H_b H_c$ паралелна правој AO . Након тога су идентификоване дефиниције и генеричке леме које су искоришћене за извођење овог својства.

4.6 Евалуација и примери

У овом поглављу биће приказана статистика успешности решавања проблема из разматраних корпуса и биће дати примери неких решених проблема.

4.6.1 Резултати за разматране корпусе

Програм тренутно успешно решава 66 од 74 решива проблема из Верникове листе, успешно се детектују сви редундантни проблеми и проблеми зависни



Слика 4.7: Статистика решавања проблема из Верниковог корпуса у зависности од типа проблема

од положаја, као и симетрични проблеми. Према строжијем разматрању корпуса (разматраном у поглављу 4.2.1), ArgoTriCS је коришћен за решавање свих 560 тројки тачака из Верникове листе и њих 274 је означено као решиво, 93 као зависно од положаја, 7 као редундантно, док систем није успео да реши 186 проблема. Од тих 186 проблема 20 је решиво али је за њихово решавање потребно додати неко ново знање у систем (за преосталих 166 је алгебарским средствима показано да су нерешиви). Статистика успешности решавања проблема из Верниковог корпуса у зависности од типа проблема дата је на слици 4.7. Детаљна статистика решавања проблема дата је у табели 4.6.1. За сваки од проблема дат је његов статус, информација да ли је симетричан са неким проблемом, као и број тврђења (од укупно 3 – за сваки од датих објеката по једно тврђење) која су успешно доказана сваким од примењених аутоматских доказивача. Нека од ових тврђења су тривијална, у случају када је неки од задатих објеката баш теме троугла ABC .

Времена решавања проблема сежу од пар милисекунди до више од једног сата. Најтеже инстанце су оне које су зависне од положаја јер је тада потребно открити скуп тачака коме припада нека од полазних тачака.

Нисмо успели да решимо неке од проблема из Верникове листе за које је доказано да су решиви (листа статуса проблема која постоји на интернету не нуди решења свих проблема који су означени као решиви [95]). Оваквих проблема је укупно 5 од 139 битно различитих проблема датих у Верниковој листи. За ове проблеме решења нисмо успели да пронађемо у литератури нити у приватној

комуникацији са ауторима великог броја постојећих решења (не рачунамо овде решења која би се могла извести из алгебарских доказа решивости).

ArgoTriCS је успео да од проблема из Веникове листе реши 274 решивих, 7 редундантних проблема и 93 проблема зависних од положаја, укупно 374 проблема. Од укупно $3 \cdot 374 = 1122$ тврђења која су доказивана за проблеме које је ArgoTriCS успео да реши (од којих је 815 нетривијалних тврђења), Вуова метода имплементирана у оквиру доказивача OpenGeoProver успешно је доказала 679 тврђења, а доказивачи интегрисани унутар GCLC алата: Вуова метода 578 тврђења, метода површина 396 тврђења, а метода Гребнерових база 410 тврђења.⁶ Ако се за сваки од проблема утврди најбољи добијени резултат неког од разматраних доказивача – доказано је 832 тврђења.

Табела 4.1: Статус проблема из Веникове листе; S означава да је проблем решив, L да је зависан од положаја, R да је редундантан, U да није решен, $S_d(N)$ означава да је проблем симетричан у односу на дефиниције са проблемом под бројем N , а $S_{dl}(N)$ да је симетричан у односу на дефиниције и леме; затим је дат број генерисаних доказа коришћењем OpenGeoProver-а и доказивача који постоје у оквиру GCLC алата коришћењем Вуове методе, методе површина, и методе Гребнерових база, као и најбољи резултат неког доказивача за дати проблем

	Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
1.	A, B, C	S	3	3	3	3	3
2.	A, B, O	L	3	2	3	3	3
3.	A, B, M_a	S	3	3	3	3	3
4.	A, B, M_b	$S(S_d(3))$	3	3	3	3	3
5.	A, B, M_c	R	3	3	3	3	3
6.	A, B, G	S	3	3	3	2	3
7.	A, B, H_a	L	3	1	2	2	3
8.	A, B, H_b	$L(S_d(7))$	3	1	0	0	3
9.	A, B, H_c	L	3	2	3	3	3
10.	A, B, H	S	3	2	3	3	3
11.	A, B, T_a	S	2	2	3	2	3
12.	A, B, T_b	$S(S_d(11))$	2	2	3	2	3
13.	A, B, T_c	L	2	2	0	0	2
14.	A, B, I	S	1	2	3	2	3
15.	A, C, O	$L(S_{dl}(2))$	3	2	3	3	3
16.	A, C, M_a	$S(S_d(3))$	3	3	3	3	3
17.	A, C, M_b	$R(S_d(5))$	3	3	3	3	3
18.	A, C, M_c	$S(S_d(3))$	3	3	3	3	3
19.	A, C, G	$S(S_{dl}(6))$	3	3	3	3	3
20.	A, C, H_a	$L(S_d(7))$	3	1	2	2	3
21.	A, C, H_b	$L(S_d(9))$	3	2	3	3	3
22.	A, C, H_c	$L(S_d(7))$	3	1	0	0	3
23.	A, C, H	$S(S_{dl}(10))$	3	2	3	3	3
24.	A, C, T_a	$S(S_d(11))$	2	1	2	1	2
25.	A, C, T_b	$L(S_d(13))$	2	2	0	0	2
26.	A, C, T_c	$S(S_d(11))$	2	2	3	2	3
27.	A, C, I	$S(S_{dl}(14))$	2	2	3	2	3
28.	A, O, M_a	S	2	1	2	1	2
29.	A, O, M_b	L	3	0	0	0	3
30.	A, O, M_c	$L(S_{dl}(29))$	3	0	0	0	3

⁶Резултати добијени за Вуову методу имплементирану у оквиру различитих доказивача могу се разликовати јер су у питању различите имплементације, које су писали различити људи.

Табела 4.1: Статус проблема из Верникове листе - наставак

	Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
31.	A, O, G	S	2	1	1	1	2
32.	A, O, H_a	S	2	1	3	1	3
33.	A, O, H_b	S	3	1	0	0	3
34.	A, O, H_c	$S (S_{dl}(33))$	3	1	0	0	3
35.	A, O, H	S	2	1	1	1	2
36.	A, O, T_a	S	1	1	3	1	3
37.	A, O, T_b	S	1	1	0	0	1
38.	A, O, T_c	$S (S_{dl}(37))$	1	1	0	0	1
39.	A, O, I	S	1	1	0	0	1
40.	A, M_a, M_b	S	3	3	3	3	3
41.	A, M_a, M_c	$S (S_d(40))$	3	3	3	3	3
42.	A, M_a, G	R	3	3	3	3	3
43.	A, M_a, H_a	L	3	1	2	2	3
44.	A, M_a, H_b	S	3	2	0	0	3
45.	A, M_a, H_c	$S (S_d(44))$	3	2	0	0	3
46.	A, M_a, H	S	2	0	0	0	2
47.	A, M_a, T_a	S	1	1	2	1	2
48.	A, M_a, T_b	U	0	0	0	0	0
49.	A, M_a, T_c	$U (S_d(48))$	0	0	0	0	0
50.	A, M_a, I	S	2	1	3	1	3
51.	A, M_b, M_c	S	3	3	3	3	3
52.	A, M_b, G	S	3	3	3	3	3
53.	A, M_b, H_a	L	2	2	3	1	3
54.	A, M_b, H_b	L	3	2	3	3	3
55.	A, M_b, H_c	L	3	2	3	3	3
56.	A, M_b, H	S	3	1	3	3	3
57.	A, M_b, T_a	S	2	1	3	3	3
58.	A, M_b, T_b	L	2	1	0	0	2
59.	A, M_b, T_c	S	2	1	3	3	3
60.	A, M_b, I	S	2	1	3	3	3
61.	A, M_c, G	$S (S_{dl}(52))$	3	3	3	1	3
62.	A, M_c, H_a	$L (S_d(53))$	2	2	3	1	3
63.	A, M_c, H_b	$L (S_d(55))$	3	2	3	3	3
64.	A, M_c, H_c	$L (S_d(54))$	3	2	3	3	3
65.	A, M_c, H	$S (S_{dl}(56))$	3	1	3	3	3
66.	A, M_c, T_a	$S (S_d(57))$	2	1	3	3	3
67.	A, M_c, T_b	$S (S_d(59))$	2	1	3	3	3
68.	A, M_c, T_c	$L (S_d(58))$	2	1	0	0	2
69.	A, M_c, I	$S (S_{dl}(60))$	2	1	3	3	3
70.	A, G, H_a	L	1	0	0	0	1
71.	A, G, H_b	S	2	1	0	0	2
72.	A, G, H_c	$S (S_{dl}(71))$	2	2	0	0	2
73.	A, G, H	S	1	1	1	1	1
74.	A, G, T_a	S	1	1	1	1	1
75.	A, G, T_b	U	0	0	0	0	0
76.	A, G, T_c	$U (S_{dl}(75))$	0	0	0	0	0
77.	A, G, I	S	1	1	3	1	3
78.	A, H_a, H_b	S	3	1	3	3	3
79.	A, H_a, H_c	$S (S_d(78))$	3	1	3	3	3
80.	A, H_a, H	L	3	1	0	0	3
81.	A, H_a, T_a	L	2	0	0	0	2
82.	A, H_a, T_b	S	2	1	0	0	2
83.	A, H_a, T_c	$S (S_d(82))$	2	1	0	0	2
84.	A, H_a, I	S	2	1	3	2	3
85.	A, H_b, H_c	S	3	1	3	3	3
86.	A, H_b, H	L	3	0	0	0	3
87.	A, H_b, T_a	S	2	1	2	1	2
88.	A, H_b, T_b	L	2	1	3	3	3
89.	A, H_b, T_c	S	2	1	0	0	2
90.	A, H_b, I	S	2	1	3	2	3
91.	A, H_c, H	$L (S_{dl}(86))$	3	0	0	0	3
92.	A, H_c, T_a	$S (S_d(87))$	2	1	3	2	3
93.	A, H_c, T_b	$S (S_d(89))$	2	1	0	0	2
94.	A, H_c, T_c	$L (S_d(88))$	2	1	3	3	3
95.	A, H_c, I	$S (S_{dl}(90))$	2	1	3	2	3
96.	A, H, T_a	U	0	0	0	0	0
97.	A, H, T_b	U	0	0	0	0	0
98.	A, H, T_c	$U (S_{dl}(97))$	0	0	0	0	0
99.	A, H, I	U	0	0	0	0	0
100.	A, T_a, T_b	S	1	1	3	1	3
101.	A, T_a, T_c	$S (S_d(100))$	1	1	3	1	3

Табела 4.1: Статус проблема из Верникове листе - наставак

	Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
102.	A, T_a, I	L	1	1	1	1	1
103.	A, T_b, T_c	S	1	1	2	1	2
104.	A, T_b, I	S	1	1	3	1	3
105.	A, T_c, I	$S(S_{dl}(104))$	1	1	3	1	3
106.	B, C, O	$L(S_{dl}(2))$	3	2	3	3	3
107.	B, C, M_a	$R(S_d(5))$	3	3	3	3	3
108.	B, C, M_b	$S(S_d(3))$	3	3	3	3	3
109.	B, C, M_c	$S(S_d(3))$	3	3	3	3	3
110.	B, C, G	$S(S_{dl}(6))$	3	3	3	3	3
111.	B, C, H_a	$L(S_d(9))$	3	2	3	3	3
112.	B, C, H_b	$L(S_d(7))$	3	1	2	2	3
113.	B, C, H_c	$L(S_d(7))$	3	1	0	0	3
114.	B, C, H	$S(S_{dl}(10))$	3	2	3	3	3
115.	B, C, T_a	$L(S_d(13))$	2	2	0	0	2
116.	B, C, T_b	$S(S_d(11))$	2	2	3	2	3
117.	B, C, T_c	$S(S_d(11))$	2	2	3	2	3
118.	B, C, I	$S(S_{dl}(14))$	2	2	3	2	3
119.	B, O, M_a	$L(S_d(29))$	3	0	0	0	3
120.	B, O, M_b	$S(S_d(28))$	2	1	1	1	2
121.	B, O, M_c	$L(S_{dl}(29))$	3	0	0	0	3
122.	B, O, G	$S(S_d(31))$	2	1	1	1	2
123.	B, O, H_a	$S(S_d(33))$	3	1	0	0	3
124.	B, O, H_b	$S(S_d(32))$	2	1	3	1	3
125.	B, O, H_c	$S(S_{dl}(33))$	3	1	0	0	3
126.	B, O, H	$S(S_d(35))$	2	1	1	1	2
127.	B, O, T_a	$S(S_d(37))$	1	1	0	0	1
128.	B, O, T_b	$S(S_d(36))$	1	1	3	1	3
129.	B, O, T_c	$S(S_{dl}(37))$	1	1	0	0	1
130.	B, O, I	$S(S_d(39))$	1	1	0	0	1
131.	B, M_a, M_b	$S(S_d(40))$	3	3	3	3	3
132.	B, M_a, M_c	$S(S_d(51))$	3	3	3	3	3
133.	B, M_a, G	$S(S_d(52))$	3	3	3	3	3
134.	B, M_a, H_a	$L(S_d(54))$	3	2	3	3	3
135.	B, M_a, H_b	$L(S_d(53))$	2	2	3	1	3
136.	B, M_a, H_c	$L(S_d(55))$	3	2	3	3	3
137.	B, M_a, H	$S(S_d(56))$	3	1	3	3	3
138.	B, M_a, T_a	$L(S_d(58))$	2	1	0	0	2
139.	B, M_a, T_b	$S(S_d(57))$	2	1	3	3	3
140.	B, M_a, T_c	$S(S_d(59))$	2	1	3	3	3
141.	B, M_a, I	$S(S_d(60))$	2	1	3	3	3
142.	B, M_b, M_c	$S(S_d(40))$	3	3	3	3	3
143.	B, M_b, G	$R(S_d(42))$	3	3	3	3	3
144.	B, M_b, H_a	$S(S_d(44))$	3	2	0	0	3
145.	B, M_b, H_b	$L(S_d(43))$	3	1	2	2	3
146.	B, M_b, H_c	$S(S_d(44))$	3	2	0	0	3
147.	B, M_b, H	$S(S_d(46))$	2	1	1	1	2
148.	B, M_b, T_a	$U(S_d(48))$	0	0	0	0	0
149.	B, M_b, T_b	$S(S_d(47))$	1	1	2	1	2
150.	B, M_b, T_c	$U(S_d(48))$	0	0	0	0	0
151.	B, M_b, I	$S(S_d(50))$	2	1	3	1	3
152.	B, M_c, G	$S(S_{dl}(52))$	3	3	3	1	3
153.	B, M_c, H_a	$L(S_d(55))$	3	2	3	3	3
154.	B, M_c, H_b	$L(S_d(53))$	2	2	3	1	3
155.	B, M_c, H_c	$L(S_d(54))$	3	2	3	3	3
156.	B, M_c, H	$S(S_{dl}(56))$	3	1	3	3	3
157.	B, M_c, T_a	$S(S_d(59))$	2	1	3	3	3
158.	B, M_c, T_b	$S(S_d(57))$	2	1	3	3	3
159.	B, M_c, T_c	$L(S_d(58))$	2	1	0	0	2
160.	B, M_c, I	$S(S_{dl}(60))$	2	1	3	3	3
161.	B, G, H_a	$S(S_d(71))$	2	1	0	0	2
162.	B, G, H_b	$L(S_d(70))$	1	1	0	0	1
163.	B, G, H_c	$S(S_{dl}(71))$	2	2	0	0	2
164.	B, G, H	$S(S_d(73))$	1	1	1	1	1
165.	B, G, T_a	$U(S_d(75))$	0	0	0	0	0
166.	B, G, T_b	$S(S_d(74))$	1	1	1	1	1
167.	B, G, T_c	$U(S_{dl}(75))$	0	0	0	0	0
168.	B, G, I	$S(S_d(77))$	1	1	3	1	3
169.	B, H_a, H_b	$S(S_d(78))$	3	1	3	3	3
170.	B, H_a, H_c	$S(S_d(85))$	3	1	3	3	3
171.	B, H_a, H	$L(S_d(86))$	3	0	0	0	3
172.	B, H_a, T_a	$L(S_d(88))$	2	1	3	3	3

Табела 4.1: Статус проблема из Верникове листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
173. B, H_a, T_b	$S(S_d(87))$	2	1	3	2	3
174. B, H_a, T_c	$S(S_d(89))$	2	1	0	0	2
175. B, H_a, I	$S(S_d(90))$	2	1	3	2	3
176. B, H_b, H_c	$S(S_d(78))$	3	1	3	3	3
177. B, H_b, H	$L(S_d(80))$	3	1	0	0	3
178. B, H_b, T_a	$S(S_d(82))$	2	1	0	0	2
179. B, H_b, T_b	$L(S_d(81))$	2	0	0	0	2
180. B, H_b, T_c	$S(S_d(82))$	2	1	0	0	2
181. B, H_b, I	$S(S_d(84))$	2	1	3	2	3
182. B, H_c, H	$L(S_{dl}(86))$	3	0	0	0	3
183. B, H_c, T_a	$S(S_d(89))$	2	1	0	0	2
184. B, H_c, T_b	$S(S_d(87))$	2	1	3	2	3
185. B, H_c, T_c	$L(S_d(88))$	2	1	3	3	3
186. B, H_c, I	$S(S_{dl}(90))$	2	1	3	2	3
187. B, H, T_a	$U(S_d(97))$	0	0	0	0	0
188. B, H, T_b	$U(S_d(96))$	0	0	0	0	0
189. B, H, T_c	$U(S_{dl}(97))$	0	0	0	0	0
190. B, H, I	$U(S_d(99))$	0	0	0	0	0
191. B, T_a, T_b	$S(S_d(100))$	1	1	3	1	3
192. B, T_a, T_c	$U(S_d(103))$	0	0	0	0	0
193. B, T_a, I	$S(S_d(104))$	1	1	3	1	3
194. B, T_b, T_c	$S(S_d(100))$	1	1	3	1	3
195. B, T_b, I	$L(S_d(102))$	1	1	1	1	1
196. B, T_c, I	$S(S_{dl}(104))$	1	1	3	1	3
197. C, O, M_a	$L(S_{dl}(29))$	3	0	0	0	3
198. C, O, M_b	$L(S_{dl}(29))$	3	0	0	0	3
199. C, O, M_c	$S(S_{dl}(28))$	2	1	2	1	2
200. C, O, G	$S(S_{dl}(31))$	2	1	1	1	2
201. C, O, H_a	$S(S_{dl}(33))$	3	1	0	0	3
202. C, O, H_b	$S(S_{dl}(33))$	3	1	0	0	3
203. C, O, H_c	$S(S_{dl}(32))$	3	1	3	1	3
204. C, O, H	$S(S_{dl}(35))$	2	1	1	1	2
205. C, O, T_a	$S(S_{dl}(37))$	1	1	0	0	1
206. C, O, T_b	$S(S_{dl}(37))$	1	1	0	0	1
207. C, O, T_c	$S(S_{dl}(36))$	1	1	3	1	3
208. C, O, I	$S(S_{dl}(39))$	1	1	0	0	1
209. C, M_a, M_b	$S(S_d(51))$	3	3	3	3	3
210. C, M_a, M_c	$S(S_d(40))$	3	3	3	3	3
211. C, M_a, G	$S(S_{dl}(52))$	3	3	3	3	3
212. C, M_a, H_a	$L(S_d(54))$	3	2	3	3	3
213. C, M_a, H_b	$L(S_d(55))$	3	2	3	3	3
214. C, M_a, H_c	$L(S_d(53))$	2	2	3	1	3
215. C, M_a, H	$S(S_{dl}(56))$	3	1	3	3	3
216. C, M_a, T_a	$L(S_d(58))$	2	1	0	0	2
217. C, M_a, T_b	$S(S_d(59))$	2	1	3	3	3
218. C, M_a, T_c	$S(S_d(57))$	2	1	3	3	3
219. C, M_a, I	$S(S_{dl}(60))$	2	1	3	3	3
220. C, M_b, M_c	$S(S_d(40))$	3	3	3	3	3
221. C, M_b, G	$S(S_{dl}(52))$	3	3	3	3	3
222. C, M_b, H_a	$L(S_d(55))$	3	2	3	3	3
223. C, M_b, H_b	$L(S_d(54))$	3	2	3	3	3
224. C, M_b, H_c	$L(S_d(53))$	2	2	3	1	3
225. C, M_b, H	$S(S_{dl}(56))$	3	1	3	3	3
226. C, M_b, T_a	$S(S_d(59))$	2	1	3	3	3
227. C, M_b, T_b	$L(S_d(58))$	2	1	0	0	2
228. C, M_b, T_c	$S(S_d(57))$	2	1	3	3	3
229. C, M_b, I	$S(S_{dl}(60))$	2	1	3	3	3
230. C, M_c, G	$R(S_{dl}(42))$	3	3	3	3	3
231. C, M_c, H_a	$S(S_d(44))$	3	2	0	0	3
232. C, M_c, H_b	$S(S_d(44))$	3	2	0	0	3
233. C, M_c, H_c	$L(S_d(43))$	3	1	2	2	3
234. C, M_c, H	$S(S_{dl}(46))$	2	1	1	1	2
235. C, M_c, T_a	$U(S_d(48))$	0	0	0	0	0
236. C, M_c, T_b	$U(S_d(48))$	0	0	0	0	0
237. C, M_c, T_c	$S(S_d(47))$	1	1	2	1	2
238. C, M_c, I	$S(S_{dl}(50))$	2	1	2	1	2
239. C, G, H_a	$S(S_{dl}(71))$	2	1	0	0	2
240. C, G, H_b	$S(S_{dl}(71))$	2	2	0	0	2
241. C, G, H_c	$L(S_{dl}(70))$	1	1	0	0	1
242. C, G, H	$S(S_{dl}(73))$	1	1	1	1	1
243. C, G, T_a	$U(S_{dl}(75))$	0	0	0	0	0

Табела 4.1: Статус проблема из Верникове листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
244. C, G, T_b	U ($S_{dl}(75)$)	0	0	0	0	0
245. C, G, T_c	S ($S_{dl}(74)$)	1	1	1	1	1
246. C, G, I	S ($S_{dl}(77)$)	1	1	2	1	2
247. C, H_a, H_b	S ($S_d(85)$)	3	1	3	3	3
248. C, H_a, H_c	S ($S_d(78)$)	3	1	3	3	3
249. C, H_a, H	L ($S_{dl}(86)$)	3	0	0	0	3
250. C, H_a, T_a	L ($S_d(88)$)	2	1	3	3	3
251. C, H_a, T_b	S ($S_d(89)$)	2	1	0	0	2
252. C, H_a, T_c	S ($S_d(87)$)	2	1	3	2	3
253. C, H_a, I	S ($S_{dl}(90)$)	2	1	3	2	3
254. C, H_b, H_c	S ($S_d(78)$)	3	1	3	3	3
255. C, H_b, H	L ($S_{dl}(86)$)	3	0	0	0	3
256. C, H_b, T_a	S ($S_d(89)$)	2	0	0	0	2
257. C, H_b, T_b	L ($S_d(88)$)	2	1	3	3	3
258. C, H_b, T_c	S ($S_d(87)$)	2	1	3	2	3
259. C, H_b, I	S ($S_{dl}(90)$)	2	1	3	2	3
260. C, H_c, H	L ($S_{dl}(80)$)	3	1	0	0	3
261. C, H_c, T_a	S ($S_d(82)$)	2	1	0	0	2
262. C, H_c, T_b	S ($S_d(82)$)	2	1	0	0	2
263. C, H_c, T_c	L ($S_d(81)$)	2	0	0	0	2
264. C, H_c, I	S ($S_{dl}(84)$)	2	1	3	2	3
265. C, H, T_a	U ($S_{dl}(97)$)	0	0	0	0	0
266. C, H, T_b	U ($S_{dl}(97)$)	0	0	0	0	0
267. C, H, T_c	U ($S_{dl}(96)$)	0	0	0	0	0
268. C, H, I	U ($S_{dl}(99)$)	0	0	0	0	0
269. C, T_a, T_b	U ($S_d(103)$)	0	0	0	0	0
270. C, T_a, T_c	S ($S_d(100)$)	1	1	3	1	3
271. C, T_a, I	S ($S_{dl}(104)$)	1	1	3	1	3
272. C, T_b, T_c	S ($S_d(100)$)	1	1	3	1	3
273. C, T_b, I	S ($S_{dl}(104)$)	1	1	3	1	3
274. C, T_c, I	L ($S_{dl}(102)$)	1	1	1	1	1
275. O, M_a, M_b	S	3	2	3	3	3
276. O, M_a, M_c	S ($S_{dl}(275)$)	3	2	3	3	3
277. O, M_a, G	S	1	0	3	0	3
278. O, M_a, H_a	L	0	0	0	0	0
279. O, M_a, H_b	S	0	0	0	0	0
280. O, M_a, H_c	S ($S_{dl}(279)$)	0	0	0	0	0
281. O, M_a, H	S	0	0	2	0	2
282. O, M_a, T_a	L	1	0	0	0	1
283. O, M_a, T_b	U	0	0	0	0	0
284. O, M_a, T_c	U ($S_{dl}(283)$)	0	0	0	0	0
285. O, M_a, I	U	0	0	0	0	0
286. O, M_b, M_c	S ($S_{dl}(275)$)	3	2	3	3	3
287. O, M_b, G	S ($S_d(277)$)	1	0	3	0	3
288. O, M_b, H_a	S ($S_d(279)$)	0	0	0	0	0
289. O, M_b, H_b	L ($S_d(278)$)	2	0	0	0	2
290. O, M_b, H_c	S ($S_{dl}(279)$)	0	0	0	0	0
291. O, M_b, H	S ($S_d(281)$)	0	0	2	0	2
292. O, M_b, T_a	U ($S_d(283)$)	0	0	0	0	0
293. O, M_b, T_b	L ($S_d(282)$)	1	0	0	0	1
294. O, M_b, T_c	U ($S_{dl}(283)$)	0	0	0	0	0
295. O, M_b, I	U ($S_d(285)$)	0	0	0	0	0
296. O, M_c, G	S ($S_{dl}(277)$)	1	0	3	0	3
297. O, M_c, H_a	S ($S_{dl}(279)$)	0	0	0	0	0
298. O, M_c, H_b	S ($S_{dl}(279)$)	0	0	0	0	0
299. O, M_c, H_c	L ($S_{dl}(278)$)	2	0	1	1	2
300. O, M_c, H	S ($S_{dl}(281)$)	0	0	2	0	2
301. O, M_c, T_a	U ($S_{dl}(283)$)	0	0	0	0	0
302. O, M_c, T_b	U ($S_{dl}(283)$)	0	0	0	0	0
303. O, M_c, T_c	L ($S_{dl}(282)$)	1	0	0	0	1
304. O, M_c, I	U ($S_{dl}(285)$)	0	0	0	0	0
305. O, G, H_a	S	0	0	0	0	0
306. O, G, H_b	S ($S_d(305)$)	0	0	0	0	0
307. O, G, H_c	S ($S_{dl}(305)$)	0	0	0	0	0
308. O, G, H	R	0	0	0	0	0
309. O, G, T_a	U	0	0	0	0	0
310. O, G, T_b	U ($S_d(309)$)	0	0	0	0	0
311. O, G, T_c	U ($S_{dl}(309)$)	0	0	0	0	0
312. O, G, I	U	0	0	0	0	0
313. O, H_a, H_b	U	0	0	0	0	0
314. O, H_a, H_c	U ($S_{dl}(313)$)	0	0	0	0	0

Табела 4.1: Статус проблема из Верникове листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
315. O, H_a, H	S	0	0	0	0	0
316. O, H_a, T_a	U	0	0	0	0	0
317. O, H_a, T_b	U	0	0	0	0	0
318. O, H_a, T_c	U ($S_{dl}(317)$)	0	0	0	0	0
319. O, H_a, I	U	0	0	0	0	0
320. O, H_b, H_c	U ($S_{dl}(313)$)	0	0	0	0	0
321. O, H_b, H	S ($S_d(315)$)	1	0	0	0	1
322. O, H_b, T_a	U ($S_d(317)$)	0	0	0	0	0
323. O, H_b, T_b	U ($S_d(316)$)	0	0	0	0	0
324. O, H_b, T_c	U ($S_{dl}(317)$)	0	0	0	0	0
325. O, H_b, I	U ($S_d(319)$)	0	0	0	0	0
326. O, H_c, H	S ($S_{dl}(315)$)	1	0	0	0	1
327. O, H_c, T_a	U ($S_{dl}(317)$)	0	0	0	0	0
328. O, H_c, T_b	U ($S_{dl}(317)$)	0	0	0	0	0
329. O, H_c, T_c	U ($S_{dl}(316)$)	0	0	0	0	0
330. O, H_c, I	U ($S_{dl}(319)$)	0	0	0	0	0
331. O, H, T_a	U	0	0	0	0	0
332. O, H, T_b	U ($S_d(331)$)	0	0	0	0	0
333. O, H, T_c	U ($S_{dl}(331)$)	0	0	0	0	0
334. O, H, I	U	0	0	0	0	0
335. O, T_a, T_b	U	0	0	0	0	0
336. O, T_a, T_c	U ($S_{dl}(335)$)	0	0	0	0	0
337. O, T_a, I	U	0	0	0	0	0
338. O, T_b, T_c	U ($S_{dl}(335)$)	0	0	0	0	0
339. O, T_b, I	U ($S_d(337)$)	0	0	0	0	0
340. O, T_c, I	U ($S_{dl}(337)$)	0	0	0	0	0
341. M_a, M_b, M_c	S	3	3	3	3	3
342. M_a, M_b, G	S	3	3	3	0	3
343. M_a, M_b, H_a	S	2	2	0	0	2
344. M_a, M_b, H_b	S ($S_d(343)$)	2	2	0	0	2
345. M_a, M_b, H_c	S	3	2	0	0	3
346. M_a, M_b, H	U	0	0	0	0	0
347. M_a, M_b, T_a	U	0	0	0	0	0
348. M_a, M_b, T_b	U ($S_d(347)$)	0	0	0	0	0
349. M_a, M_b, T_c	U	0	0	0	0	0
350. M_a, M_b, I	U	0	0	0	0	0
351. M_a, M_c, G	S ($S_{dl}(342)$)	3	3	3	0	3
352. M_a, M_c, H_a	S ($S_d(343)$)	3	2	0	0	3
353. M_a, M_c, H_b	S ($S_d(345)$)	3	2	0	0	3
354. M_a, M_c, H_c	S ($S_d(343)$)	2	2	0	0	2
355. M_a, M_c, H	U ($S_{dl}(346)$)	0	0	0	0	0
356. M_a, M_c, T_a	U ($S_d(347)$)	0	0	0	0	0
357. M_a, M_c, T_b	U ($S_d(349)$)	0	0	0	0	0
358. M_a, M_c, T_c	U ($S_d(347)$)	0	0	0	0	0
359. M_a, M_c, I	U ($S_{dl}(350)$)	0	0	0	0	0
360. M_a, G, H_a	L	1	1	0	0	1
361. M_a, G, H_b	S	3	1	0	0	3
362. M_a, G, H_c	S ($S_{dl}(361)$)	3	1	0	0	3
363. M_a, G, H	S	1	0	1	0	1
364. M_a, G, T_a	S	1	0	2	0	2
365. M_a, G, T_b	U	0	0	0	0	0
366. M_a, G, T_c	U ($S_{dl}(365)$)	0	0	0	0	0
367. M_a, G, I	S	1	0	1	0	1
368. M_a, H_a, H_b	S	0	0	3	1	3
369. M_a, H_a, H_c	S ($S_d(368)$)	0	0	3	1	3
370. M_a, H_a, H	L	2	0	1	1	2
371. M_a, H_a, T_a	L	1	0	2	2	2
372. M_a, H_a, T_b	U	0	0	0	0	0
373. M_a, H_a, T_c	U ($S_d(372)$)	0	0	0	0	0
374. M_a, H_a, I	S	0	0	0	0	0
375. M_a, H_b, H_c	L	1	1	1	0	1
376. M_a, H_b, H	S	1	1	0	0	1
377. M_a, H_b, T_a	S	0	0	0	0	0
378. M_a, H_b, T_b	S	1	0	0	0	1
379. M_a, H_b, T_c	U	0	0	0	0	0
380. M_a, H_b, I	U	0	0	0	0	0
381. M_a, H_c, H	S ($S_{dl}(376)$)	1	1	0	0	1
382. M_a, H_c, T_a	S ($S_d(377)$)	0	0	0	0	0
383. M_a, H_c, T_b	U ($S_d(379)$)	0	0	0	0	0
384. M_a, H_c, T_c	S ($S_d(378)$)	1	0	0	0	1
385. M_a, H_c, I	U ($S_{dl}(380)$)	0	0	0	0	0

Табела 4.1: Статус проблема из Верникове листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
386. M_a, H, T_a	U	0	0	0	0	0
387. M_a, H, T_b	U	0	0	0	0	0
388. M_a, H, T_c	U ($S_{dl}(387)$)	0	0	0	0	0
389. M_a, H, I	U	0	0	0	0	0
390. M_a, T_a, T_b	U	0	0	0	0	0
391. M_a, T_a, T_c	U ($S_d(390)$)	0	0	0	0	0
392. M_a, T_a, I	S	0	0	3	0	3
393. M_a, T_b, T_c	U	0	0	0	0	0
394. M_a, T_b, I	U	0	0	0	0	0
395. M_a, T_c, I	U ($S_{dl}(394)$)	0	0	0	0	0
396. M_b, M_c, G	S ($S_{dl}(342)$)	3	3	3	0	3
397. M_b, M_c, H_a	S ($S_d(345)$)	3	2	0	0	3
398. M_b, M_c, H_b	S ($S_d(343)$)	3	2	0	0	3
399. M_b, M_c, H_c	S ($S_d(343)$)	3	2	0	0	3
400. M_b, M_c, H	U ($S_{dl}(346)$)	0	0	0	0	0
401. M_b, M_c, T_a	U ($S_d(349)$)	0	0	0	0	0
402. M_b, M_c, T_b	U ($S_d(347)$)	0	0	0	0	0
403. M_b, M_c, T_c	U ($S_d(347)$)	0	0	0	0	0
404. M_b, M_c, I	U ($S_{dl}(350)$)	0	0	0	0	0
405. M_b, G, H_a	S ($S_d(361)$)	3	1	0	0	3
406. M_b, G, H_b	L ($S_d(360)$)	1	1	0	0	1
407. M_b, G, H_c	S ($S_{dl}(361)$)	3	1	0	0	3
408. M_b, G, H	S ($S_d(363)$)	0	0	1	0	1
409. M_b, G, T_a	U ($S_d(365)$)	0	0	0	0	0
410. M_b, G, T_b	S ($S_d(364)$)	1	0	2	0	2
411. M_b, G, T_c	U ($S_{dl}(365)$)	0	0	0	0	0
412. M_b, G, I	S ($S_d(367)$)	0	0	1	0	1
413. M_b, H_a, H_b	S ($S_d(368)$)	0	0	3	0	3
414. M_b, H_a, H_c	L ($S_d(375)$)	2	1	1	0	2
415. M_b, H_a, H	S ($S_d(376)$)	1	1	0	0	1
416. M_b, H_a, T_a	S ($S_d(378)$)	1	0	0	0	1
417. M_b, H_a, T_b	S ($S_d(377)$)	0	0	0	0	0
418. M_b, H_a, T_c	U ($S_d(379)$)	0	0	0	0	0
419. M_b, H_a, I	U ($S_d(380)$)	0	0	0	0	0
420. M_b, H_b, H_c	S ($S_d(368)$)	0	0	3	1	3
421. M_b, H_b, H	L ($S_d(370)$)	2	0	0	0	2
422. M_b, H_b, T_a	U ($S_d(372)$)	0	0	0	0	0
423. M_b, H_b, T_b	L ($S_d(371)$)	1	0	3	3	3
424. M_b, H_b, T_c	U ($S_d(372)$)	0	0	0	0	0
425. M_b, H_b, I	S ($S_d(374)$)	0	0	0	0	0
426. M_b, H_c, H	S ($S_{dl}(376)$)	1	1	0	0	1
427. M_b, H_c, T_a	U ($S_d(379)$)	0	0	0	0	0
428. M_b, H_c, T_b	S ($S_d(377)$)	0	0	0	0	0
429. M_b, H_c, T_c	S ($S_d(378)$)	1	0	0	0	1
430. M_b, H_c, I	U ($S_{dl}(380)$)	0	0	0	0	0
431. M_b, H, T_a	U ($S_d(387)$)	0	0	0	0	0
432. M_b, H, T_b	U ($S_d(386)$)	0	0	0	0	0
433. M_b, H, T_c	U ($S_{dl}(387)$)	0	0	0	0	0
434. M_b, H, I	U ($S_d(389)$)	0	0	0	0	0
435. M_b, T_a, T_b	U ($S_d(390)$)	0	0	0	0	0
436. M_b, T_a, T_c	U ($S_d(393)$)	0	0	0	0	0
437. M_b, T_a, I	U ($S_d(394)$)	0	0	0	0	0
438. M_b, T_b, T_c	U ($S_d(390)$)	0	0	0	0	0
439. M_b, T_b, I	S ($S_d(392)$)	0	0	3	0	3
440. M_b, T_c, I	U ($S_{dl}(394)$)	0	0	0	0	0
441. M_c, G, H_a	S ($S_{dl}(361)$)	3	1	0	0	3
442. M_c, G, H_b	S ($S_{dl}(361)$)	3	1	0	0	3
443. M_c, G, H_c	L ($S_{dl}(360)$)	1	1	0	0	1
444. M_c, G, H	S ($S_{dl}(363)$)	0	0	1	0	1
445. M_c, G, T_a	U ($S_{dl}(365)$)	0	0	0	0	0
446. M_c, G, T_b	U ($S_{dl}(365)$)	0	0	0	0	0
447. M_c, G, T_c	S ($S_{dl}(364)$)	1	0	2	0	2
448. M_c, G, I	S ($S_{dl}(367)$)	0	0	1	0	1
449. M_c, H_a, H_b	L ($S_d(375)$)	2	1	1	0	2
450. M_c, H_a, H_c	S ($S_d(368)$)	1	0	3	0	3
451. M_c, H_a, H	S ($S_{dl}(376)$)	1	1	0	0	1
452. M_c, H_a, T_a	S ($S_d(378)$)	1	0	0	0	1
453. M_c, H_a, T_b	U ($S_d(379)$)	0	0	0	0	0
454. M_c, H_a, T_c	S ($S_d(377)$)	0	0	0	0	0
455. M_c, H_a, I	U ($S_{dl}(380)$)	0	0	0	0	0
456. M_c, H_b, H_c	S ($S_d(368)$)	1	0	3	0	3

Табела 4.1: Статус проблема из Верникове листе - наставак

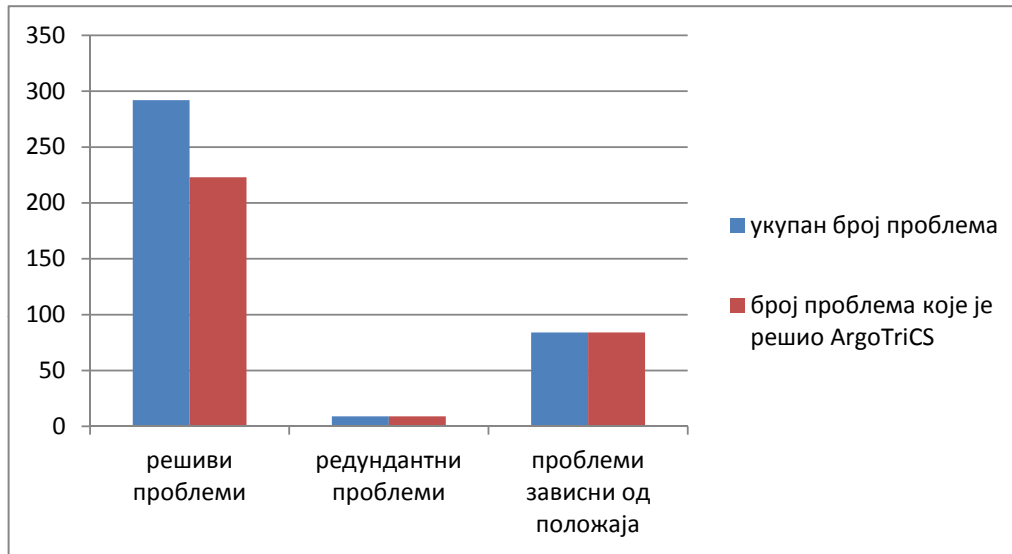
	Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
457.	M_c, H_b, H	$S (S_{dl}(376))$	1	1	0	0	1
458.	M_c, H_b, T_a	$U (S_d(379))$	0	0	0	0	0
459.	M_c, H_b, T_b	$S (S_d(378))$	1	0	0	0	1
460.	M_c, H_b, T_c	$S (S_d(377))$	0	0	0	0	0
461.	M_c, H_b, I	$U (S_{dl}(380))$	0	0	0	0	0
462.	M_c, H_c, H	$L (S_{dl}(370))$	2	0	0	0	2
463.	M_c, H_c, T_a	$U (S_d(372))$	0	0	0	0	0
464.	M_c, H_c, T_b	$U (S_d(372))$	0	0	0	0	0
465.	M_c, H_c, T_c	$L (S_d(371))$	1	0	3	3	3
466.	M_c, H_c, I	$S (S_{dl}(374))$	0	0	0	0	0
467.	M_c, H, T_a	$U (S_{dl}(387))$	0	0	0	0	0
468.	M_c, H, T_b	$U (S_{dl}(387))$	0	0	0	0	0
469.	M_c, H, T_c	$U (S_{dl}(386))$	0	0	0	0	0
470.	M_c, H, I	$U (S_{dl}(389))$	0	0	0	0	0
471.	M_c, T_a, T_b	$U (S_d(393))$	0	0	0	0	0
472.	M_c, T_a, T_c	$U (S_d(390))$	0	0	0	0	0
473.	M_c, T_a, I	$U (S_{dl}(394))$	0	0	0	0	0
474.	M_c, T_b, T_c	$U (S_d(390))$	0	0	0	0	0
475.	M_c, T_b, I	$U (S_{dl}(394))$	0	0	0	0	0
476.	M_c, T_c, I	$S (S_{dl}(392))$	0	0	3	0	3
477.	G, H_a, H_b	U	0	0	0	0	0
478.	G, H_a, H_c	$U (S_{dl}(477))$	0	0	0	0	0
479.	G, H_a, H	S	0	0	0	0	0
480.	G, H_a, T_a	S	0	0	0	0	0
481.	G, H_a, T_b	U	0	0	0	0	0
482.	G, H_a, T_c	$U (S_{dl}(481))$	0	0	0	0	0
483.	G, H_a, I	U	0	0	0	0	0
484.	G, H_b, H_c	$U (S_{dl}(477))$	0	0	0	0	0
485.	G, H_b, H	$S (S_d(479))$	0	0	0	0	0
486.	G, H_b, T_a	$U (S_d(481))$	0	0	0	0	0
487.	G, H_b, T_b	$S (S_d(480))$	0	0	0	0	0
488.	G, H_b, T_c	$U (S_{dl}(481))$	0	0	0	0	0
489.	G, H_b, I	$U (S_d(483))$	0	0	0	0	0
490.	G, H_c, H	$S (S_{dl}(479))$	0	0	0	0	0
491.	G, H_c, T_a	$U (S_{dl}(481))$	0	0	0	0	0
492.	G, H_c, T_b	$U (S_{dl}(481))$	0	0	0	0	0
493.	G, H_c, T_c	$S (S_{dl}(480))$	0	0	0	0	0
494.	G, H_c, I	$U (S_{dl}(483))$	0	0	0	0	0
495.	G, H, T_a	U	0	0	0	0	0
496.	G, H, T_b	$U (S_d(495))$	0	0	0	0	0
497.	G, H, T_c	$U (S_{dl}(495))$	0	0	0	0	0
498.	G, H, I	U	0	0	0	0	0
499.	G, T_a, T_b	U	0	0	0	0	0
500.	G, T_a, T_c	$U (S_{dl}(499))$	0	0	0	0	0
501.	G, T_a, I	U	0	0	0	0	0
502.	G, T_b, T_c	$U (S_{dl}(499))$	0	0	0	0	0
503.	G, T_b, I	$U (S_d(501))$	0	0	0	0	0
504.	G, T_c, I	$U (S_{dl}(501))$	0	0	0	0	0
505.	H_a, H_b, H_c	S	0	0	0	0	0
506.	H_a, H_b, H	S	3	2	2	0	3
507.	H_a, H_b, T_a	S	0	0	0	0	0
508.	H_a, H_b, T_b	$S (S_d(507))$	0	0	0	0	0
509.	H_a, H_b, T_c	U	0	0	0	0	0
510.	H_a, H_b, I	U	0	0	0	0	0
511.	H_a, H_c, H	$S (S_{dl}(506))$	2	1	2	0	2
512.	H_a, H_c, T_a	$S (S_d(507))$	0	0	0	0	0
513.	H_a, H_c, T_b	$U (S_d(509))$	0	0	0	0	0
514.	H_a, H_c, T_c	$S (S_d(507))$	0	0	0	0	0
515.	H_a, H_c, I	$U (S_{dl}(510))$	0	0	0	0	0
516.	H_a, H, T_a	L	1	0	0	0	1
517.	H_a, H, T_b	U	0	0	0	0	0
518.	H_a, H, T_c	$U (S_{dl}(517))$	0	0	0	0	0
519.	H_a, H, I	U	0	0	0	0	0
520.	H_a, T_a, T_b	U	0	0	0	0	0
521.	H_a, T_a, T_c	$U (S_d(520))$	0	0	0	0	0
522.	H_a, T_a, I	S	1	0	3	1	3
523.	H_a, T_b, T_c	U	0	0	0	0	0
524.	H_a, T_b, I	U	0	0	0	0	0
525.	H_a, T_c, I	$U (S_{dl}(524))$	0	0	0	0	0
526.	H_b, H_c, H	$S (S_{dl}(506))$	1	0	2	0	2
527.	H_b, H_c, T_a	$U (S_d(509))$	0	0	0	0	0

Табела 4.1: Статус проблема из Верникове листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
528. H_b, H_c, T_b	S ($S_d(507)$)	0	0	0	0	0
529. H_b, H_c, T_c	S ($S_d(507)$)	0	0	0	0	0
530. H_b, H_c, I	U ($S_{dl}(510)$)	0	0	0	0	0
531. H_b, H, T_a	U ($S_d(517)$)	0	0	0	0	0
532. H_b, H, T_b	L ($S_d(516)$)	1	0	0	0	1
533. H_b, H, T_c	U ($S_{dl}(517)$)	0	0	0	0	0
534. H_b, H, I	U ($S_d(519)$)	0	0	0	0	0
535. H_b, T_a, T_b	U ($S_d(520)$)	0	0	0	0	0
536. H_b, T_a, T_c	U ($S_d(523)$)	0	0	0	0	0
537. H_b, T_a, I	U ($S_d(524)$)	0	0	0	0	0
538. H_b, T_b, T_c	U ($S_d(520)$)	0	0	0	0	0
539. H_b, T_b, I	S ($S_d(522)$)	1	0	3	1	3
540. H_b, T_c, I	U ($S_{dl}(524)$)	0	0	0	0	0
541. H_c, H, T_a	U ($S_{dl}(517)$)	0	0	0	0	0
542. H_c, H, T_b	U ($S_{dl}(517)$)	0	0	0	0	0
543. H_c, H, T_c	L ($S_{dl}(516)$)	1	0	2	2	2
544. H_c, H, I	U ($S_{dl}(519)$)	0	0	0	0	0
545. H_c, T_a, T_b	U ($S_d(523)$)	0	0	0	0	0
546. H_c, T_a, T_c	U ($S_d(520)$)	0	0	0	0	0
547. H_c, T_a, I	U ($S_{dl}(524)$)	0	0	0	0	0
548. H_c, T_b, T_c	U ($S_d(520)$)	0	0	0	0	0
549. H_c, T_b, I	U ($S_{dl}(524)$)	0	0	0	0	0
550. H_c, T_c, I	S ($S_{dl}(522)$)	1	0	3	1	3
551. H, T_a, T_b	U	0	0	0	0	0
552. H, T_a, T_c	U ($S_{dl}(551)$)	0	0	0	0	0
553. H, T_a, I	U	0	0	0	0	0
554. H, T_b, T_c	U ($S_{dl}(551)$)	0	0	0	0	0
555. H, T_b, I	U ($S_d(553)$)	0	0	0	0	0
556. H, T_c, I	U ($S_{dl}(553)$)	0	0	0	0	0
557. T_a, T_b, T_c	U	0	0	0	0	0
558. T_a, T_b, I	S	0	0	0	0	0
559. T_a, T_c, I	S ($S_{dl}(558)$)	0	0	0	0	0
560. T_b, T_c, I	S ($S_{dl}(558)$)	0	0	0	0	0

Такође, програм успешно решава 62 од 73 решива проблема из Конелијеве листе и успешно се детектују сви редунантни проблеми, као и проблеми зависни од положаја. За 580 проблема из Конелијеве листе перформансе система ArgoTriCS су следеће: 223 је означено као решиво, 84 као зависно од положаја, 9 као редунантно, док 264 проблема систем није успео да реши. За неке од проблема је утврђено да су симетрични у односу на дефиниције, или у односу на дефиниције и леме. Подсетимо се да постоји пуно инстанци у Конелијевој листи за које је доказано да су нерешиве или чији статус још увек није утврђен. Статистика успешности решавања проблема из Конелијевог корпуса у зависности од типа проблема дата је на слици 4.8, а детаљна статистика решавања проблема дата је у табели 4.6.1.

ArgoTriCS је успео да реши 223 решива проблема, 84 проблема зависна од положаја и 9 редунантних проблема из Конелијеве листе – укупно 316 проблема. Од укупно $3 \cdot 316 = 948$ тврђења добијених из решених проблема из Конелијеве листе (од којих је 801 нетривијалних), Вуова метода имплементирана у оквиру доказивача OpenGeoProver успешно је доказала 318 тврђења, док су доказивачи интегрисани унутар GCLC алата показали следећи успех:



Слика 4.8: Статистика решавања проблема из Конелијевог корпуса у зависности од типа проблема

Вуова метода доказала је 296 тврђења, метода површина 192 тврђења, а метода Гребнерових база 129 тврђења. Узимајући у обзир најбољи резултат неког од разматраних доказивача за сваки од решених проблема добијамо да је неком од метода успешно доказано 408 од укупно 948 тврђења.

Табела 4.2: Статус проблема из Конелијеве листе; S означава да је проблем решив, L да је зависан од положаја, R да је редундантан, U да није решен, $S_d(N)$ означава да је проблем симетричан у односу на дефиниције са проблемом под бројем N , а $S_{dl}(N)$ да је симетричан у односу на дефиниције и леме; затим је дат број генерисаних доказа коришћењем OpenGeoProver-а и доказивача који постоје у оквиру GCLC алата коришћењем Вуове методе, методе површина, и методе Гребнерових база, као и најбољи резултат неког од доказивача за дати проблем

	Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
561.	A, B, E_a	S	2	2	3	2	3
562.	A, B, E_b	$S (S_d(561))$	2	2	3	2	3
563.	A, B, E_c	S	3	2	3	2	3
564.	A, B, N	S	2	2	3	2	3
565.	A, C, E_a	$S (S_d(561))$	2	2	3	3	3
566.	A, C, E_b	$S (S_d(563))$	3	2	3	3	3
567.	A, C, E_c	$S (S_d(561))$	3	2	3	3	3
568.	A, C, N	$S (S_{dl}(564))$	2	2	2	2	2
569.	A, E_a, E_b	S	1	1	3	1	3
570.	A, E_a, E_c	$S (S_d(569))$	1	1	3	1	3
571.	A, E_a, G	S	1	1	1	1	1
572.	A, E_a, H	R	1	1	3	1	3
573.	A, E_a, H_a	L	3	1	3	3	3
574.	A, E_a, H_b	L	1	1	1	1	1
575.	A, E_a, H_c	$L (S_d(574))$	2	1	2	1	2
576.	A, E_a, I	U	0	0	0	0	0
577.	A, E_a, M_a	S	1	1	1	1	1
578.	A, E_a, M_b	S	2	1	3	1	3
579.	A, E_a, M_c	$S (S_d(578))$	2	1	3	1	3

Табела 4.2: Статус проблема из Конелијеве листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
580. A, E_a, N	S	1	1	1	1	1
581. A, E_a, O	S	1	1	1	1	1
582. A, E_a, T_a	S	2	1	3	1	3
583. A, E_a, T_b	U	0	0	0	0	0
584. A, E_a, T_c	U ($S_d(583)$)	0	0	0	0	0
585. A, E_b, E_c	U	0	0	0	0	0
586. A, E_b, G	U	0	0	0	0	0
587. A, E_b, H	S	1	1	3	1	3
588. A, E_b, H_a	S	1	1	0	0	1
589. A, E_b, H_b	L	3	0	2	2	3
590. A, E_b, H_c	S	2	1	0	0	2
591. A, E_b, I	U	0	0	0	0	0
592. A, E_b, M_a	U	0	0	0	0	0
593. A, E_b, M_b	S	2	1	2	1	2
594. A, E_b, M_c	S	2	1	3	1	3
595. A, E_b, N	S	1	1	0	0	1
596. A, E_b, O	U	0	0	0	0	0
597. A, E_b, T_a	U	0	0	0	0	0
598. A, E_b, T_b	U	0	0	0	0	0
599. A, E_b, T_c	U	0	0	0	0	0
600. A, E_c, G	U ($S_{dl}(586)$)	0	0	0	0	0
601. A, E_c, H	S ($S_{dl}(587)$)	1	1	3	1	3
602. A, E_c, H_a	S ($S_d(588)$)	1	1	0	0	1
603. A, E_c, H_b	S ($S_d(590)$)	2	1	0	0	2
604. A, E_c, H_c	L ($S_d(589)$)	3	0	0	0	3
605. A, E_c, I	U ($S_{dl}(591)$)	0	0	0	0	0
606. A, E_c, M_a	U ($S_d(592)$)	0	0	0	0	0
607. A, E_c, M_b	S ($S_d(594)$)	2	1	3	1	3
608. A, E_c, M_c	S ($S_d(593)$)	2	1	2	1	2
609. A, E_c, N	S ($S_{dl}(595)$)	1	1	0	0	1
610. A, E_c, O	U ($S_{dl}(596)$)	0	0	0	0	0
611. A, E_c, T_a	U ($S_d(597)$)	0	0	0	0	0
612. A, E_c, T_b	U ($S_d(599)$)	0	0	0	0	0
613. A, E_c, T_c	U ($S_d(598)$)	0	0	0	0	0
614. A, G, N	S	1	1	1	1	1
615. A, H, N	S	1	1	1	1	1
616. A, H_a, N	S	1	1	0	0	1
617. A, H_b, N	S	2	1	0	0	2
618. A, H_c, N	S ($S_{dl}(617)$)	2	1	0	0	2
619. A, I, N	U	0	0	0	0	0
620. A, M_a, N	S	1	1	1	1	1
621. A, M_b, N	S	2	1	2	1	2
622. A, M_c, N	S ($S_{dl}(621)$)	2	1	3	1	3
623. A, N, O	S	1	1	1	1	1
624. A, N, T_a	U	0	0	0	0	0
625. A, N, T_b	U	0	0	0	0	0
626. A, N, T_c	U ($S_{dl}(625)$)	0	0	0	0	0
627. B, C, E_a	S ($S_d(563)$)	3	2	3	3	3
628. B, C, E_b	S ($S_d(561)$)	2	2	3	3	3
629. B, C, E_c	S ($S_d(561)$)	3	2	3	3	3
630. B, C, N	S ($S_{dl}(564)$)	2	2	2	2	2
631. B, E_a, E_b	S ($S_d(569)$)	1	1	2	1	2
632. B, E_a, E_c	U ($S_d(585)$)	0	0	0	0	0
633. B, E_a, G	U ($S_d(586)$)	0	0	0	0	0
634. B, E_a, H	S ($S_d(587)$)	1	1	3	1	3
635. B, E_a, H_a	L ($S_d(589)$)	3	0	2	2	3
636. B, E_a, H_b	S ($S_d(588)$)	1	1	0	0	1
637. B, E_a, H_c	S ($S_d(590)$)	2	1	0	0	2
638. B, E_a, I	U ($S_d(591)$)	0	0	0	0	0
639. B, E_a, M_a	S ($S_d(593)$)	2	1	2	1	2
640. B, E_a, M_b	U ($S_d(592)$)	0	0	0	0	0
641. B, E_a, M_c	S ($S_d(594)$)	2	1	3	1	3
642. B, E_a, N	S ($S_d(595)$)	1	1	0	0	1
643. B, E_a, O	U ($S_d(596)$)	0	0	0	0	0
644. B, E_a, T_a	U ($S_d(598)$)	0	0	0	0	0
645. B, E_a, T_b	U ($S_d(597)$)	0	0	0	0	0
646. B, E_a, T_c	U ($S_d(599)$)	0	0	0	0	0
647. B, E_b, E_c	S ($S_d(569)$)	1	1	3	1	3
648. B, E_b, G	S ($S_d(571)$)	1	1	1	1	1
649. B, E_b, H	R ($S_d(572)$)	1	1	3	1	3
650. B, E_b, H_a	L ($S_d(574)$)	1	1	1	1	1

Табела 4.2: Статус проблема из Конелијеве листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
651. B, E_b, H_b	L ($S_d(573)$)	3	1	3	3	3
652. B, E_b, H_c	L ($S_d(574)$)	2	1	2	1	2
653. B, E_b, I	U ($S_d(576)$)	0	0	0	0	0
654. B, E_b, M_a	S ($S_d(578)$)	3	1	3	1	3
655. B, E_b, M_b	S ($S_d(577)$)	1	1	1	1	1
656. B, E_b, M_c	S ($S_d(578)$)	2	1	3	1	3
657. B, E_b, N	S ($S_d(580)$)	1	1	1	1	1
658. B, E_b, O	S ($S_d(581)$)	2	1	1	1	2
659. B, E_b, T_a	U ($S_d(583)$)	0	0	0	0	0
660. B, E_b, T_b	S ($S_d(582)$)	2	1	3	1	3
661. B, E_b, T_c	U ($S_d(583)$)	0	0	0	0	0
662. B, E_c, G	U ($S_{dl}(586)$)	0	0	0	0	0
663. B, E_c, H	S ($S_{dl}(587)$)	1	1	3	1	3
664. B, E_c, H_a	S ($S_d(590)$)	2	1	0	0	2
665. B, E_c, H_b	S ($S_d(588)$)	1	1	0	0	1
666. B, E_c, H_c	L ($S_d(589)$)	3	0	0	0	3
667. B, E_c, I	U ($S_{dl}(591)$)	0	0	0	0	0
668. B, E_c, M_a	S ($S_d(594)$)	2	1	3	1	3
669. B, E_c, M_b	U ($S_d(592)$)	0	0	0	0	0
670. B, E_c, M_c	S ($S_d(593)$)	2	1	2	1	2
671. B, E_c, N	S ($S_{dl}(595)$)	1	1	0	0	1
672. B, E_c, O	U ($S_{dl}(596)$)	0	0	0	0	0
673. B, E_c, T_a	U ($S_d(599)$)	0	0	0	0	0
674. B, E_c, T_b	U ($S_d(597)$)	0	0	0	0	0
675. B, E_c, T_c	U ($S_d(598)$)	0	0	0	0	0
676. B, G, N	S ($S_d(614)$)	1	1	1	1	1
677. B, H, N	S ($S_d(615)$)	1	1	1	1	1
678. B, H_a, N	S ($S_d(617)$)	2	1	0	0	2
679. B, H_b, N	S ($S_d(616)$)	1	1	0	0	1
680. B, H_c, N	S ($S_{dl}(617)$)	2	1	0	0	2
681. B, I, N	U ($S_d(619)$)	0	0	0	0	0
682. B, M_a, N	S ($S_d(621)$)	2	1	2	1	2
683. B, M_b, N	S ($S_d(620)$)	1	1	1	1	1
684. B, M_c, N	S ($S_{dl}(621)$)	2	1	2	1	2
685. B, N, O	S ($S_d(623)$)	1	1	1	1	1
686. B, N, T_a	U ($S_d(625)$)	0	0	0	0	0
687. B, N, T_b	U ($S_d(624)$)	0	0	0	0	0
688. B, N, T_c	U ($S_{dl}(625)$)	0	0	0	0	0
689. C, E_a, E_b	U ($S_d(585)$)	0	0	0	0	0
690. C, E_a, E_c	S ($S_d(569)$)	1	1	3	1	3
691. C, E_a, G	U ($S_{dl}(586)$)	0	0	0	0	0
692. C, E_a, H	S ($S_{dl}(587)$)	1	1	3	1	3
693. C, E_a, H_a	L ($S_d(589)$)	3	0	2	2	3
694. C, E_a, H_b	S ($S_d(590)$)	2	1	0	0	2
695. C, E_a, H_c	S ($S_d(588)$)	1	1	0	0	1
696. C, E_a, I	U ($S_{dl}(591)$)	0	0	0	0	0
697. C, E_a, M_a	S ($S_d(593)$)	2	1	2	1	2
698. C, E_a, M_b	S ($S_d(594)$)	2	1	3	1	3
699. C, E_a, M_c	U ($S_d(592)$)	0	0	0	0	0
700. C, E_a, N	S ($S_{dl}(595)$)	1	1	0	0	1
701. C, E_a, O	U ($S_{dl}(596)$)	0	0	0	0	0
702. C, E_a, T_a	U ($S_d(598)$)	0	0	0	0	0
703. C, E_a, T_b	U ($S_d(599)$)	0	0	0	0	0
704. C, E_a, T_c	U ($S_d(597)$)	0	0	0	0	0
705. C, E_b, E_c	S ($S_d(569)$)	1	1	3	1	3
706. C, E_b, G	U ($S_{dl}(586)$)	0	0	0	0	0
707. C, E_b, H	S ($S_{dl}(587)$)	1	1	3	1	3
708. C, E_b, H_a	S ($S_d(590)$)	2	1	0	0	2
709. C, E_b, H_b	L ($S_d(589)$)	3	0	0	0	3
710. C, E_b, H_c	S ($S_d(588)$)	1	1	0	0	1
711. C, E_b, I	U ($S_{dl}(591)$)	0	0	0	0	0
712. C, E_b, M_a	S ($S_d(594)$)	2	1	3	1	3
713. C, E_b, M_b	S ($S_d(593)$)	2	1	2	1	2
714. C, E_b, M_c	U ($S_d(592)$)	0	0	0	0	0
715. C, E_b, N	S ($S_{dl}(595)$)	1	1	0	0	1
716. C, E_b, O	U ($S_{dl}(596)$)	0	0	0	0	0
717. C, E_b, T_a	U ($S_d(599)$)	0	0	0	0	0
718. C, E_b, T_b	U ($S_d(598)$)	0	0	0	0	0
719. C, E_b, T_c	U ($S_d(597)$)	0	0	0	0	0
720. C, E_c, G	S ($S_{dl}(571)$)	1	1	1	1	1
721. C, E_c, H	R ($S_{dl}(572)$)	1	1	3	1	3

Табела 4.2: Статус проблема из Конелијеве листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
722. C, E_c, H_a	L ($S_d(574)$)	1	1	1	1	1
723. C, E_c, H_b	L ($S_d(574)$)	2	1	1	1	2
724. C, E_c, H_c	L ($S_d(573)$)	3	1	3	3	3
725. C, E_c, I	U ($S_{dl}(576)$)	0	0	0	0	0
726. C, E_c, M_a	S ($S_d(578)$)	3	1	3	1	3
727. C, E_c, M_b	S ($S_d(578)$)	2	1	3	1	3
728. C, E_c, M_c	S ($S_d(577)$)	1	1	1	1	1
729. C, E_c, N	S ($S_{dl}(580)$)	1	1	1	1	1
730. C, E_c, O	S ($S_{dl}(581)$)	2	1	1	1	2
731. C, E_c, T_a	U ($S_d(583)$)	0	0	0	0	0
732. C, E_c, T_b	U ($S_d(583)$)	0	0	0	0	0
733. C, E_c, T_c	S ($S_d(582)$)	2	1	2	1	2
734. C, G, N	S ($S_{dl}(614)$)	1	1	1	1	1
735. C, H, N	S ($S_{dl}(615)$)	1	1	1	1	1
736. C, H_a, N	S ($S_{dl}(617)$)	2	1	0	0	2
737. C, H_b, N	S ($S_{dl}(617)$)	2	1	0	0	2
738. C, H_c, N	S ($S_{dl}(616)$)	1	1	0	0	1
739. C, I, N	U ($S_{dl}(619)$)	0	0	0	0	0
740. C, M_a, N	S ($S_{dl}(621)$)	3	1	3	1	3
741. C, M_b, N	S ($S_{dl}(621)$)	2	1	3	1	3
742. C, M_c, N	S ($S_{dl}(620)$)	1	1	1	1	1
743. C, N, O	S ($S_{dl}(623)$)	1	1	1	1	1
744. C, N, T_a	U ($S_{dl}(625)$)	0	0	0	0	0
745. C, N, T_b	U ($S_{dl}(625)$)	0	0	0	0	0
746. C, N, T_c	U ($S_{dl}(624)$)	0	0	0	0	0
747. E_a, E_b, E_c	S	0	0	0	0	0
748. E_a, E_b, G	U	0	0	0	0	0
749. E_a, E_b, H	S	0	0	1	0	1
750. E_a, E_b, H_a	S	0	0	0	0	0
751. E_a, E_b, H_b	S ($S_d(750)$)	0	0	0	0	0
752. E_a, E_b, H_c	S	0	0	0	0	0
753. E_a, E_b, I	U	0	0	0	0	0
754. E_a, E_b, M_a	L	1	0	0	0	1
755. E_a, E_b, M_b	L ($S_d(754)$)	1	1	1	0	1
756. E_a, E_b, M_c	S	3	0	0	0	3
757. E_a, E_b, N	L	0	0	0	0	0
758. E_a, E_b, O	U	0	0	0	0	0
759. E_a, E_b, T_a	U	0	0	0	0	0
760. E_a, E_b, T_b	U ($S_d(759)$)	0	0	0	0	0
761. E_a, E_b, T_c	U	0	0	0	0	0
762. E_a, E_c, G	U ($S_{dl}(748)$)	0	0	0	0	0
763. E_a, E_c, H	S ($S_{dl}(749)$)	0	0	1	0	1
764. E_a, E_c, H_a	S ($S_d(750)$)	0	0	0	0	0
765. E_a, E_c, H_b	S ($S_d(752)$)	0	0	0	0	0
766. E_a, E_c, H_c	S ($S_d(750)$)	0	0	0	0	0
767. E_a, E_c, I	U ($S_{dl}(753)$)	0	0	0	0	0
768. E_a, E_c, M_a	L ($S_d(754)$)	1	0	0	0	1
769. E_a, E_c, M_b	S ($S_d(756)$)	3	0	0	0	3
770. E_a, E_c, M_c	L ($S_d(754)$)	1	1	1	0	1
771. E_a, E_c, N	L ($S_{dl}(757)$)	0	0	0	0	0
772. E_a, E_c, O	U ($S_{dl}(758)$)	0	0	0	0	0
773. E_a, E_c, T_a	U ($S_d(759)$)	0	0	0	0	0
774. E_a, E_c, T_b	U ($S_d(761)$)	0	0	0	0	0
775. E_a, E_c, T_c	U ($S_d(759)$)	0	0	0	0	0
776. E_a, G, H	S	0	0	0	0	0
777. E_a, G, H_a	S	0	0	0	0	0
778. E_a, G, H_b	U	0	0	0	0	0
779. E_a, G, H_c	U ($S_{dl}(778)$)	0	0	0	0	0
780. E_a, G, I	U	0	0	0	0	0
781. E_a, G, M_a	S	0	0	0	0	0
782. E_a, G, M_b	U	0	0	0	0	0
783. E_a, G, M_c	U ($S_{dl}(782)$)	0	0	0	0	0
784. E_a, G, N	S	0	0	0	0	0
785. E_a, G, O	S	0	0	0	0	0
786. E_a, G, T_a	U	0	0	0	0	0
787. E_a, G, T_b	U	0	0	0	0	0
788. E_a, G, T_c	U ($S_{dl}(787)$)	0	0	0	0	0
789. E_a, H, H_a	L	2	0	3	3	3
790. E_a, H, H_b	L	0	0	0	0	0
791. E_a, H, H_c	L ($S_{dl}(790)$)	0	0	0	0	0
792. E_a, H, I	U	0	0	0	0	0

Табела 4.2: Статус проблема из Конелијеве листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
793. E_a, H, M_a	S	0	0	0	0	0
794. E_a, H, M_b	S	1	0	3	0	3
795. E_a, H, M_c	S ($S_{dl}(794)$)	1	0	3	0	3
796. E_a, H, N	S	0	0	0	0	0
797. E_a, H, O	S	0	0	0	0	0
798. E_a, H, T_a	S	2	0	0	0	2
799. E_a, H, T_b	U	0	0	0	0	0
800. E_a, H, T_c	U ($S_{dl}(799)$)	0	0	0	0	0
801. E_a, H_a, H_b	S	2	0	3	1	3
802. E_a, H_a, H_c	S ($S_d(801)$)	2	0	3	1	3
803. E_a, H_a, I	U	0	0	0	0	0
804. E_a, H_a, M_a	L	1	0	0	0	1
805. E_a, H_a, M_b	S	1	1	0	0	1
806. E_a, H_a, M_c	S ($S_d(805)$)	1	1	0	0	1
807. E_a, H_a, N	L	0	0	0	0	0
808. E_a, H_a, O	S	1	0	3	0	3
809. E_a, H_a, T_a	L	1	0	0	0	1
810. E_a, H_a, T_b	U	0	0	0	0	0
811. E_a, H_a, T_c	U ($S_d(810)$)	0	0	0	0	0
812. E_a, H_b, H_c	L	0	0	0	0	0
813. E_a, H_b, I	U	0	0	0	0	0
814. E_a, H_b, M_a	L	1	1	1	0	1
815. E_a, H_b, M_b	S	1	1	0	0	1
816. E_a, H_b, M_c	S	1	1	0	0	1
817. E_a, H_b, N	L	0	0	0	0	0
818. E_a, H_b, O	U	0	0	0	0	0
819. E_a, H_b, T_a	U	0	0	0	0	0
820. E_a, H_b, T_b	U	0	0	0	0	0
821. E_a, H_b, T_c	U	0	0	0	0	0
822. E_a, H_c, I	U ($S_{dl}(813)$)	0	0	0	0	0
823. E_a, H_c, M_a	L ($S_d(814)$)	1	1	2	0	2
824. E_a, H_c, M_b	S ($S_d(816)$)	1	1	0	0	1
825. E_a, H_c, M_c	S ($S_d(815)$)	1	1	0	0	1
826. E_a, H_c, N	L ($S_{dl}(817)$)	0	0	0	0	0
827. E_a, H_c, O	U ($S_{dl}(818)$)	0	0	0	0	0
828. E_a, H_c, T_a	U ($S_d(819)$)	0	0	0	0	0
829. E_a, H_c, T_b	U ($S_d(821)$)	0	0	0	0	0
830. E_a, H_c, T_c	U ($S_d(820)$)	0	0	0	0	0
831. E_a, I, M_a	S	0	0	0	0	0
832. E_a, I, M_b	U	0	0	0	0	0
833. E_a, I, M_c	U ($S_{dl}(832)$)	0	0	0	0	0
834. E_a, I, N	S	0	0	0	0	0
835. E_a, I, O	U	0	0	0	0	0
836. E_a, I, T_a	U	0	0	0	0	0
837. E_a, I, T_b	U	0	0	0	0	0
838. E_a, I, T_c	U ($S_{dl}(837)$)	0	0	0	0	0
839. E_a, M_a, M_b	L	1	1	1	0	1
840. E_a, M_a, M_c	L ($S_d(839)$)	1	1	1	0	1
841. E_a, M_a, N	R	0	0	0	0	0
842. E_a, M_a, O	S	0	0	1	0	1
843. E_a, M_a, T_a	U	0	0	0	0	0
844. E_a, M_a, T_b	U	0	0	0	0	0
845. E_a, M_a, T_c	U ($S_d(844)$)	0	0	0	0	0
846. E_a, M_b, M_c	S	2	2	0	0	2
847. E_a, M_b, N	L	1	1	1	0	1
848. E_a, M_b, O	S	1	1	1	0	1
849. E_a, M_b, T_a	U	0	0	0	0	0
850. E_a, M_b, T_b	U	0	0	0	0	0
851. E_a, M_b, T_c	U	0	0	0	0	0
852. E_a, M_c, N	L ($S_{dl}(847)$)	1	1	1	0	1
853. E_a, M_c, O	S ($S_{dl}(848)$)	1	1	1	0	1
854. E_a, M_c, T_a	U ($S_d(849)$)	0	0	0	0	0
855. E_a, M_c, T_b	U ($S_d(851)$)	0	0	0	0	0
856. E_a, M_c, T_c	U ($S_d(850)$)	0	0	0	0	0
857. E_a, N, O	S	0	0	0	0	0
858. E_a, N, T_a	U	0	0	0	0	0
859. E_a, N, T_b	U	0	0	0	0	0
860. E_a, N, T_c	U ($S_{dl}(859)$)	0	0	0	0	0
861. E_a, O, T_a	U	0	0	0	0	0
862. E_a, O, T_b	U	0	0	0	0	0
863. E_a, O, T_c	U ($S_{dl}(862)$)	0	0	0	0	0

Табела 4.2: Статус проблема из Конелијеве листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
864. E_a, T_a, T_b	U	0	0	0	0	0
865. E_a, T_a, T_c	U ($S_d(864)$)	0	0	0	0	0
866. E_a, T_b, T_c	U	0	0	0	0	0
867. E_b, E_c, G	U ($S_{dl}(748)$)	0	0	0	0	0
868. E_b, E_c, H	S ($S_{dl}(749)$)	0	0	1	0	1
869. E_b, E_c, H_a	S ($S_d(752)$)	0	0	0	0	0
870. E_b, E_c, H_b	S ($S_d(750)$)	0	0	0	0	0
871. E_b, E_c, H_c	S ($S_d(750)$)	0	0	0	0	0
872. E_b, E_c, I	U ($S_{dl}(753)$)	0	0	0	0	0
873. E_b, E_c, M_a	S ($S_d(756)$)	3	0	0	0	3
874. E_b, E_c, M_b	L ($S_d(754)$)	1	1	0	0	1
875. E_b, E_c, M_c	L ($S_d(754)$)	1	1	0	0	1
876. E_b, E_c, N	L ($S_{dl}(757)$)	0	0	0	0	0
877. E_b, E_c, O	U ($S_{dl}(758)$)	0	0	0	0	0
878. E_b, E_c, T_a	U ($S_d(761)$)	0	0	0	0	0
879. E_b, E_c, T_b	U ($S_d(759)$)	0	0	0	0	0
880. E_b, E_c, T_c	U ($S_d(759)$)	0	0	0	0	0
881. E_b, G, H	S ($S_d(776)$)	0	0	0	0	0
882. E_b, G, H_a	U ($S_d(778)$)	0	0	0	0	0
883. E_b, G, H_b	S ($S_d(777)$)	0	0	0	0	0
884. E_b, G, H_c	U ($S_{dl}(778)$)	0	0	0	0	0
885. E_b, G, I	U ($S_d(780)$)	0	0	0	0	0
886. E_b, G, M_a	U ($S_d(782)$)	0	0	0	0	0
887. E_b, G, M_b	S ($S_d(781)$)	0	0	0	0	0
888. E_b, G, M_c	U ($S_{dl}(782)$)	0	0	0	0	0
889. E_b, G, N	S ($S_d(784)$)	0	0	0	0	0
890. E_b, G, O	S ($S_d(785)$)	0	0	0	0	0
891. E_b, G, T_a	U ($S_d(787)$)	0	0	0	0	0
892. E_b, G, T_b	U ($S_d(786)$)	0	0	0	0	0
893. E_b, G, T_c	U ($S_{dl}(787)$)	0	0	0	0	0
894. E_b, H, H_a	L ($S_d(790)$)	0	0	0	0	0
895. E_b, H, H_b	L ($S_d(789)$)	2	0	3	3	3
896. E_b, H, H_c	L ($S_{dl}(790)$)	0	0	0	0	0
897. E_b, H, I	U ($S_d(792)$)	0	0	0	0	0
898. E_b, H, M_a	S ($S_d(794)$)	1	0	3	0	3
899. E_b, H, M_b	S ($S_d(793)$)	0	0	0	0	0
900. E_b, H, M_c	S ($S_{dl}(794)$)	1	0	3	0	3
901. E_b, H, N	S ($S_d(796)$)	0	0	0	0	0
902. E_b, H, O	S ($S_d(797)$)	0	0	0	0	0
903. E_b, H, T_a	U ($S_d(799)$)	0	0	0	0	0
904. E_b, H, T_b	S ($S_d(798)$)	0	0	1	0	1
905. E_b, H, T_c	U ($S_{dl}(799)$)	0	0	0	0	0
906. E_b, H_a, H_b	S ($S_d(801)$)	0	0	0	0	0
907. E_b, H_a, H_c	L ($S_d(812)$)	0	0	0	0	0
908. E_b, H_a, I	U ($S_d(813)$)	0	0	0	0	0
909. E_b, H_a, M_a	S ($S_d(815)$)	1	1	0	0	1
910. E_b, H_a, M_b	L ($S_d(814)$)	1	1	1	0	1
911. E_b, H_a, M_c	S ($S_d(816)$)	1	1	0	0	1
912. E_b, H_a, N	L ($S_d(817)$)	0	0	0	0	0
913. E_b, H_a, O	U ($S_d(818)$)	0	0	0	0	0
914. E_b, H_a, T_a	U ($S_d(820)$)	0	0	0	0	0
915. E_b, H_a, T_b	U ($S_d(819)$)	0	0	0	0	0
916. E_b, H_a, T_c	U ($S_d(821)$)	0	0	0	0	0
917. E_b, H_b, H_c	S ($S_d(801)$)	2	0	3	1	3
918. E_b, H_b, I	U ($S_d(803)$)	0	0	0	0	0
919. E_b, H_b, M_a	S ($S_d(805)$)	1	1	0	0	1
920. E_b, H_b, M_b	L ($S_d(804)$)	2	0	0	0	2
921. E_b, H_b, M_c	S ($S_d(805)$)	1	1	0	0	1
922. E_b, H_b, N	L ($S_d(807)$)	0	0	0	0	0
923. E_b, H_b, O	S ($S_d(808)$)	1	0	3	0	3
924. E_b, H_b, T_a	U ($S_d(810)$)	0	0	0	0	0
925. E_b, H_b, T_b	L ($S_d(809)$)	1	0	0	0	1
926. E_b, H_b, T_c	U ($S_d(810)$)	0	0	0	0	0
927. E_b, H_c, I	U ($S_{dl}(813)$)	0	0	0	0	0
928. E_b, H_c, M_a	S ($S_d(816)$)	1	1	0	0	1
929. E_b, H_c, M_b	L ($S_d(814)$)	1	1	2	0	2
930. E_b, H_c, M_c	S ($S_d(815)$)	1	1	0	0	1
931. E_b, H_c, N	L ($S_{dl}(817)$)	1	0	0	0	1
932. E_b, H_c, O	U ($S_{dl}(818)$)	0	0	0	0	0
933. E_b, H_c, T_a	U ($S_d(821)$)	0	0	0	0	0
934. E_b, H_c, T_b	U ($S_d(819)$)	0	0	0	0	0

Табела 4.2: Статус проблема из Конелијеве листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
935. E_b, H_c, T_c	U ($S_d(820)$)	0	0	0	0	0
936. E_b, I, M_a	U ($S_d(832)$)	0	0	0	0	0
937. E_b, I, M_b	S ($S_d(831)$)	0	0	0	0	0
938. E_b, I, M_c	U ($S_{dl}(832)$)	0	0	0	0	0
939. E_b, I, N	S ($S_d(834)$)	0	0	0	0	0
940. E_b, I, O	U ($S_d(835)$)	0	0	0	0	0
941. E_b, I, T_a	U ($S_d(837)$)	0	0	0	0	0
942. E_b, I, T_b	U ($S_d(836)$)	0	0	0	0	0
943. E_b, I, T_c	U ($S_{dl}(837)$)	0	0	0	0	0
944. E_b, M_a, M_b	L ($S_d(839)$)	1	2	2	0	2
945. E_b, M_a, M_c	S ($S_d(846)$)	2	2	0	0	2
946. E_b, M_a, N	L ($S_d(847)$)	1	0	0	0	1
947. E_b, M_a, O	S ($S_d(848)$)	1	1	1	0	1
948. E_b, M_a, T_a	U ($S_d(850)$)	0	0	0	0	0
949. E_b, M_a, T_b	U ($S_d(849)$)	0	0	0	0	0
950. E_b, M_a, T_c	U ($S_d(851)$)	0	0	0	0	0
951. E_b, M_b, M_c	L ($S_d(839)$)	1	2	3	0	3
952. E_b, M_b, N	R ($S_d(841)$)	1	0	1	0	1
953. E_b, M_b, O	S ($S_d(842)$)	0	0	1	0	1
954. E_b, M_b, T_a	U ($S_d(844)$)	0	0	0	0	0
955. E_b, M_b, T_b	U ($S_d(843)$)	0	0	0	0	0
956. E_b, M_b, T_c	U ($S_d(844)$)	0	0	0	0	0
957. E_b, M_c, N	L ($S_{dl}(847)$)	1	1	0	0	1
958. E_b, M_c, O	S ($S_{dl}(848)$)	1	1	1	0	1
959. E_b, M_c, T_a	U ($S_d(851)$)	0	0	0	0	0
960. E_b, M_c, T_b	U ($S_d(849)$)	0	0	0	0	0
961. E_b, M_c, T_c	U ($S_d(850)$)	0	0	0	0	0
962. E_b, N, O	S ($S_d(857)$)	0	0	0	0	0
963. E_b, N, T_a	U ($S_d(859)$)	0	0	0	0	0
964. E_b, N, T_b	U ($S_d(858)$)	0	0	0	0	0
965. E_b, N, T_c	U ($S_{dl}(859)$)	0	0	0	0	0
966. E_b, O, T_a	U ($S_d(862)$)	0	0	0	0	0
967. E_b, O, T_b	U ($S_d(861)$)	0	0	0	0	0
968. E_b, O, T_c	U ($S_{dl}(862)$)	0	0	0	0	0
969. E_b, T_a, T_b	U ($S_d(864)$)	0	0	0	0	0
970. E_b, T_a, T_c	U ($S_d(866)$)	0	0	0	0	0
971. E_b, T_b, T_c	U ($S_d(864)$)	0	0	0	0	0
972. E_c, G, H	S ($S_{dl}(776)$)	0	0	0	0	0
973. E_c, G, H_a	U ($S_{dl}(778)$)	0	0	0	0	0
974. E_c, G, H_b	U ($S_{dl}(778)$)	0	0	0	0	0
975. E_c, G, H_c	S ($S_{dl}(777)$)	0	0	0	0	0
976. E_c, G, I	U ($S_{dl}(780)$)	0	0	0	0	0
977. E_c, G, M_a	U ($S_{dl}(782)$)	0	0	0	0	0
978. E_c, G, M_b	U ($S_{dl}(782)$)	0	0	0	0	0
979. E_c, G, M_c	S ($S_{dl}(781)$)	0	0	0	0	0
980. E_c, G, N	S ($S_{dl}(784)$)	0	0	0	0	0
981. E_c, G, O	S ($S_{dl}(785)$)	0	0	0	0	0
982. E_c, G, T_a	U ($S_{dl}(787)$)	0	0	0	0	0
983. E_c, G, T_b	U ($S_{dl}(787)$)	0	0	0	0	0
984. E_c, G, T_c	U ($S_{dl}(786)$)	0	0	0	0	0
985. E_c, H, H_a	L ($S_{dl}(790)$)	0	0	0	0	0
986. E_c, H, H_b	L ($S_{dl}(790)$)	0	0	0	0	0
987. E_c, H, H_c	L ($S_{dl}(789)$)	2	0	3	3	3
988. E_c, H, I	U ($S_{dl}(792)$)	0	0	0	0	0
989. E_c, H, M_a	S ($S_{dl}(794)$)	1	0	3	0	3
990. E_c, H, M_b	S ($S_{dl}(794)$)	1	0	3	0	3
991. E_c, H, M_c	S ($S_{dl}(793)$)	0	0	0	0	0
992. E_c, H, N	S ($S_{dl}(796)$)	0	0	0	0	0
993. E_c, H, O	S ($S_{dl}(797)$)	0	0	0	0	0
994. E_c, H, T_a	U ($S_{dl}(799)$)	0	0	0	0	0
995. E_c, H, T_b	U ($S_{dl}(799)$)	0	0	0	0	0
996. E_c, H, T_c	S ($S_{dl}(798)$)	0	0	0	0	0
997. E_c, H_a, H_b	L ($S_d(812)$)	0	0	0	0	0
998. E_c, H_a, H_c	S ($S_d(801)$)	0	0	0	0	0
999. E_c, H_a, I	U ($S_{dl}(813)$)	0	0	0	0	0
1000. E_c, H_a, M_a	S ($S_d(815)$)	1	1	0	0	1
1001. E_c, H_a, M_b	S ($S_d(816)$)	1	1	0	0	1
1002. E_c, H_a, M_c	L ($S_d(814)$)	1	1	1	0	1
1003. E_c, H_a, N	L ($S_{dl}(817)$)	0	0	0	0	0
1004. E_c, H_a, O	U ($S_{dl}(818)$)	0	0	0	0	0
1005. E_c, H_a, T_a	U ($S_d(820)$)	0	0	0	0	0

Табела 4.2: Статус проблема из Конелијеве листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
1006. E_c, H_a, T_b	U ($S_d(821)$)	0	0	0	0	0
1007. E_c, H_a, T_c	U ($S_d(819)$)	0	0	0	0	0
1008. E_c, H_b, H_c	S ($S_d(801)$)	0	0	0	0	0
1009. E_c, H_b, I	U ($S_{dl}(813)$)	0	0	0	0	0
1010. E_c, H_b, M_a	S ($S_d(816)$)	1	1	0	0	1
1011. E_c, H_b, M_b	S ($S_d(815)$)	1	1	0	0	1
1012. E_c, H_b, M_c	L ($S_d(814)$)	1	1	1	0	1
1013. E_c, H_b, N	L ($S_{dl}(817)$)	1	0	0	0	1
1014. E_c, H_b, O	U ($S_{dl}(818)$)	0	0	0	0	0
1015. E_c, H_b, T_a	U ($S_d(821)$)	0	0	0	0	0
1016. E_c, H_b, T_b	U ($S_d(820)$)	0	0	0	0	0
1017. E_c, H_b, T_c	U ($S_d(819)$)	0	0	0	0	0
1018. E_c, H_c, I	U ($S_{dl}(803)$)	0	0	0	0	0
1019. E_c, H_c, M_a	S ($S_d(805)$)	1	1	0	0	1
1020. E_c, H_c, M_b	S ($S_d(805)$)	1	1	0	0	1
1021. E_c, H_c, M_c	L ($S_d(804)$)	2	0	0	0	2
1022. E_c, H_c, N	L ($S_{dl}(807)$)	0	0	0	0	0
1023. E_c, H_c, O	S ($S_{dl}(808)$)	1	0	3	0	3
1024. E_c, H_c, T_a	U ($S_d(810)$)	0	0	0	0	0
1025. E_c, H_c, T_b	U ($S_d(810)$)	0	0	0	0	0
1026. E_c, H_c, T_c	L ($S_d(809)$)	1	0	1	0	1
1027. E_c, I, M_a	U ($S_{dl}(832)$)	0	0	0	0	0
1028. E_c, I, M_b	U ($S_{dl}(832)$)	0	0	0	0	0
1029. E_c, I, M_c	S ($S_{dl}(831)$)	0	0	0	0	0
1030. E_c, I, N	S ($S_{dl}(834)$)	0	0	0	0	0
1031. E_c, I, O	U ($S_{dl}(835)$)	0	0	0	0	0
1032. E_c, I, T_a	U ($S_{dl}(837)$)	0	0	0	0	0
1033. E_c, I, T_b	U ($S_{dl}(837)$)	0	0	0	0	0
1034. E_c, I, T_c	U ($S_{dl}(836)$)	0	0	0	0	0
1035. E_c, M_a, M_b	S ($S_d(846)$)	2	2	0	0	2
1036. E_c, M_a, M_c	L ($S_d(839)$)	1	2	2	0	2
1037. E_c, M_a, N	L ($S_{dl}(847)$)	1	0	0	0	1
1038. E_c, M_a, O	S ($S_{dl}(848)$)	1	1	1	0	1
1039. E_c, M_a, T_a	U ($S_d(850)$)	0	0	0	0	0
1040. E_c, M_a, T_b	U ($S_d(851)$)	0	0	0	0	0
1041. E_c, M_a, T_c	U ($S_d(849)$)	0	0	0	0	0
1042. E_c, M_b, M_c	L ($S_d(839)$)	1	2	2	0	2
1043. E_c, M_b, N	L ($S_{dl}(847)$)	1	1	0	0	1
1044. E_c, M_b, O	S ($S_{dl}(848)$)	1	1	1	0	1
1045. E_c, M_b, T_a	U ($S_d(851)$)	0	0	0	0	0
1046. E_c, M_b, T_b	U ($S_d(850)$)	0	0	0	0	0
1047. E_c, M_b, T_c	U ($S_d(849)$)	0	0	0	0	0
1048. E_c, M_c, N	R ($S_{dl}(841)$)	1	0	1	0	1
1049. E_c, M_c, O	S ($S_{dl}(842)$)	0	0	1	0	1
1050. E_c, M_c, T_a	U ($S_d(844)$)	0	0	0	0	0
1051. E_c, M_c, T_b	U ($S_d(844)$)	0	0	0	0	0
1052. E_c, M_c, T_c	U ($S_d(843)$)	0	0	0	0	0
1053. E_c, N, O	S ($S_{dl}(857)$)	0	0	0	0	0
1054. E_c, N, T_a	U ($S_{dl}(859)$)	0	0	0	0	0
1055. E_c, N, T_b	U ($S_{dl}(859)$)	0	0	0	0	0
1056. E_c, N, T_c	U ($S_{dl}(858)$)	0	0	0	0	0
1057. E_c, O, T_a	U ($S_{dl}(862)$)	0	0	0	0	0
1058. E_c, O, T_b	U ($S_{dl}(862)$)	0	0	0	0	0
1059. E_c, O, T_c	U ($S_{dl}(861)$)	0	0	0	0	0
1060. E_c, T_a, T_b	U ($S_d(866)$)	0	0	0	0	0
1061. E_c, T_a, T_c	U ($S_d(864)$)	0	0	0	0	0
1062. E_c, T_b, T_c	U ($S_d(864)$)	0	0	0	0	0
1063. G, H, N	R	0	0	0	0	0
1064. G, H_a, N	S	0	0	0	0	0
1065. G, H_b, N	S ($S_d(1064)$)	0	0	0	0	0
1066. G, H_c, N	S ($S_{dl}(1064)$)	0	0	0	0	0
1067. G, I, N	U	0	0	0	0	0
1068. G, M_a, N	S	0	0	1	0	1
1069. G, M_b, N	S ($S_d(1068)$)	0	0	1	0	1
1070. G, M_c, N	S ($S_{dl}(1068)$)	0	0	1	0	1
1071. G, N, O	R	0	0	0	0	0
1072. G, N, T_a	U	0	0	0	0	0
1073. G, N, T_b	U ($S_d(1072)$)	0	0	0	0	0
1074. G, N, T_c	U ($S_{dl}(1072)$)	0	0	0	0	0
1075. H, H_a, N	S	0	0	0	0	0
1076. H, H_b, N	S ($S_d(1075)$)	0	0	0	0	0

Табела 4.2: Статус проблема из Конелијеве листе - наставак

Проблем	Статус	OGP (Wu)	GCLC (Area)	GCLC (Wu)	GCLC (GB)	MAX
1077. H, H_c, N	$S (S_{dl}(1075))$	0	0	0	0	0
1078. H, I, N	U	0	0	0	0	0
1079. H, M_a, N	S	0	0	0	0	0
1080. H, M_b, N	$S (S_d(1079))$	0	0	0	0	0
1081. H, M_c, N	$S (S_{dl}(1079))$	0	0	0	0	0
1082. H, N, O	R	0	0	0	0	0
1083. H, N, T_a	U	0	0	0	0	0
1084. H, N, T_b	$U (S_d(1083))$	0	0	0	0	0
1085. H, N, T_c	$U (S_{dl}(1083))$	0	0	0	0	0
1086. H_a, H_b, N	L	0	0	0	0	0
1087. H_a, H_c, N	$L (S_{dl}(1086))$	0	0	0	0	0
1088. H_a, I, N	S	0	0	0	0	0
1089. H_a, M_a, N	L	0	0	0	0	0
1090. H_a, M_b, N	L	1	1	1	0	1
1091. H_a, M_c, N	$L (S_{dl}(1090))$	1	1	1	0	1
1092. H_a, N, O	S	0	0	0	0	0
1093. H_a, N, T_a	U	0	0	0	0	0
1094. H_a, N, T_b	U	0	0	0	0	0
1095. H_a, N, T_c	$U (S_{dl}(1094))$	0	0	0	0	0
1096. H_b, H_c, N	$L (S_{dl}(1086))$	0	0	0	0	0
1097. H_b, I, N	$S (S_d(1088))$	0	0	0	0	0
1098. H_b, M_a, N	$L (S_d(1090))$	1	1	0	0	1
1099. H_b, M_b, N	$L (S_d(1089))$	1	0	1	0	1
1100. H_b, M_c, N	$L (S_{dl}(1090))$	1	1	1	0	1
1101. H_b, N, O	$S (S_d(1092))$	0	0	0	0	0
1102. H_b, N, T_a	$U (S_d(1094))$	0	0	0	0	0
1103. H_b, N, T_b	$U (S_d(1093))$	0	0	0	0	0
1104. H_b, N, T_c	$U (S_{dl}(1094))$	0	0	0	0	0
1105. H_c, I, N	$S (S_{dl}(1088))$	0	0	0	0	0
1106. H_c, M_a, N	$L (S_{dl}(1090))$	1	1	0	0	1
1107. H_c, M_b, N	$L (S_{dl}(1090))$	1	1	2	0	2
1108. H_c, M_c, N	$L (S_{dl}(1089))$	1	0	1	0	1
1109. H_c, N, O	$S (S_{dl}(1092))$	1	0	0	0	1
1110. H_c, N, T_a	$U (S_{dl}(1094))$	0	0	0	0	0
1111. H_c, N, T_b	$U (S_{dl}(1094))$	0	0	0	0	0
1112. H_c, N, T_c	$U (S_{dl}(1093))$	0	0	0	0	0
1113. I, M_a, N	S	0	0	0	0	0
1114. I, M_b, N	$S (S_d(1113))$	0	0	0	0	0
1115. I, M_c, N	$S (S_{dl}(1113))$	0	0	0	0	0
1116. I, N, O	U	0	0	0	0	0
1117. I, N, T_a	U	0	0	0	0	0
1118. I, N, T_b	$U (S_d(1117))$	0	0	0	0	0
1119. I, N, T_c	$U (S_{dl}(1117))$	0	0	0	0	0
1120. M_a, M_b, N	L	2	2	2	0	2
1121. M_a, M_c, N	$L (S_{dl}(1120))$	2	2	2	0	2
1122. M_a, N, O	S	1	0	0	0	1
1123. M_a, N, T_a	U	0	0	0	0	0
1124. M_a, N, T_b	U	0	0	0	0	0
1125. M_a, N, T_c	$U (S_{dl}(1124))$	0	0	0	0	0
1126. M_b, M_c, N	$L (S_{dl}(1120))$	2	2	2	0	2
1127. M_b, N, O	$S (S_d(1122))$	1	0	0	0	1
1128. M_b, N, T_a	$U (S_d(1124))$	0	0	0	0	0
1129. M_b, N, T_b	$U (S_d(1123))$	0	0	0	0	0
1130. M_b, N, T_c	$U (S_{dl}(1124))$	0	0	0	0	0
1131. M_c, N, O	$S (S_{dl}(1122))$	1	0	0	0	1
1132. M_c, N, T_a	$U (S_{dl}(1124))$	0	0	0	0	0
1133. M_c, N, T_b	$U (S_{dl}(1124))$	0	0	0	0	0
1134. M_c, N, T_c	$U (S_{dl}(1123))$	0	0	0	0	0
1135. N, O, T_a	U	0	0	0	0	0
1136. N, O, T_b	$U (S_d(1135))$	0	0	0	0	0
1137. N, O, T_c	$U (S_{dl}(1135))$	0	0	0	0	0
1138. N, T_a, T_b	U	0	0	0	0	0
1139. N, T_a, T_c	$U (S_{dl}(1138))$	0	0	0	0	0
1140. N, T_b, T_c	U	0	0	0	0	0

4.6.2 Анализа коришћеног знања

Током решавања проблема из датих корпуса направљена је анализа броја потребних дефиниција/лема/примитивних конструкција. На слици 4.9 приказан је број коришћених дефиниција за решавање првих N проблема из Верниковог и Конелијевог корпуса, при чему $1 \leq N \leq 560$ одговара проблемима из Верникове листе, а $561 \leq N \leq 1140$ проблемима из Конелијеве листе (корпуси су на слици међусобно раздвојени хоризонталном линијом). Као што се и претпостављало укупан број дефиниција на почетку брзо расте, а затим све спорије, да би на крају постао скоро константна функција, тј. број нових дефиниција које су потребне за решавање наредних проблема из листе постаје јако мали. Слично важи и за број потребних лема и примитивних конструкција и одговарајуће статистике приказане су на сликама 4.10 и 4.11. Посебно је важно истакнути да је за проблеме из новог корпуса потребан јако мали број нових дефиниција, лема и примитивних конструкција у односу на знање потребно за решавање проблема из већ решаваног корпуса.

4.6.3 Примери

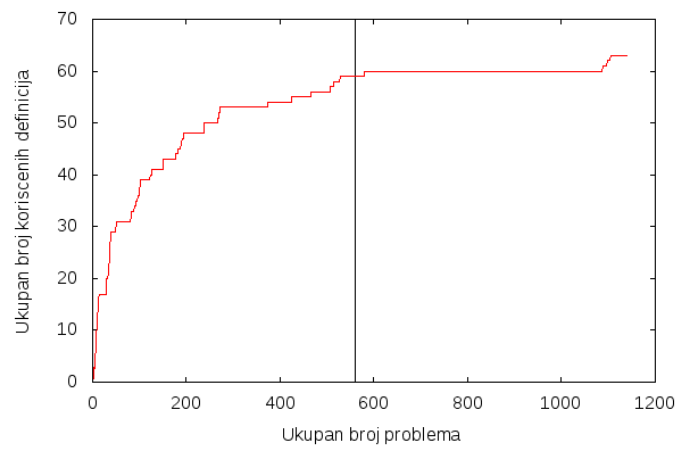
Након што се конструкција нађе и поједностави, она се може извести у различите формате. Тренутно је подржан излаз у природни језик (на енглеском језику, форматиран у L^AT_EX формату), као и излаз у GCLC језик [54]. У доказу на природном језику штампа се поставка проблема, низ конструктивних корака, услови недегенерисаности и услови неодређености, као и које се примитивне конструкције, дефиниције и леме користе у доказу.

На пример, генерисана конструкција на природном језику за проблем 32: A, O, H_a има следећу форму:

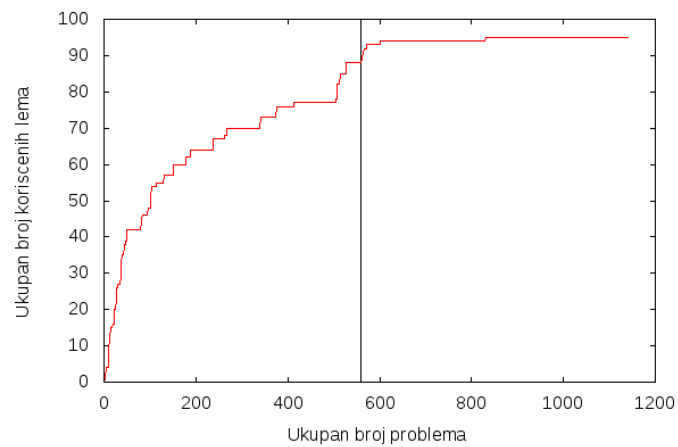
Problem 32: Given a point A , a point O , and a point H_a construct the triangle ABC .

Construction:

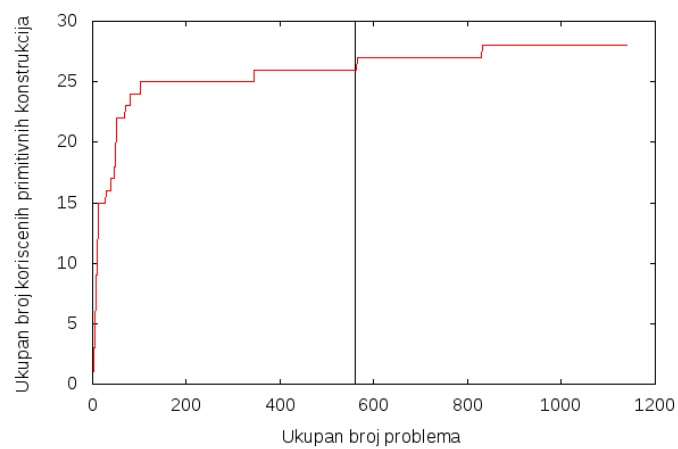
1. Using the point A and the point H_a construct a line h_a (rule W02);
% DET: points A and H_a are not the same;
2. Using the point A and the point O construct a circle $k(O, C)$ (rule W06);
% NDG: points A and O are not the same;
3. Using the point H_a and the line h_a construct a line a (rule W10);



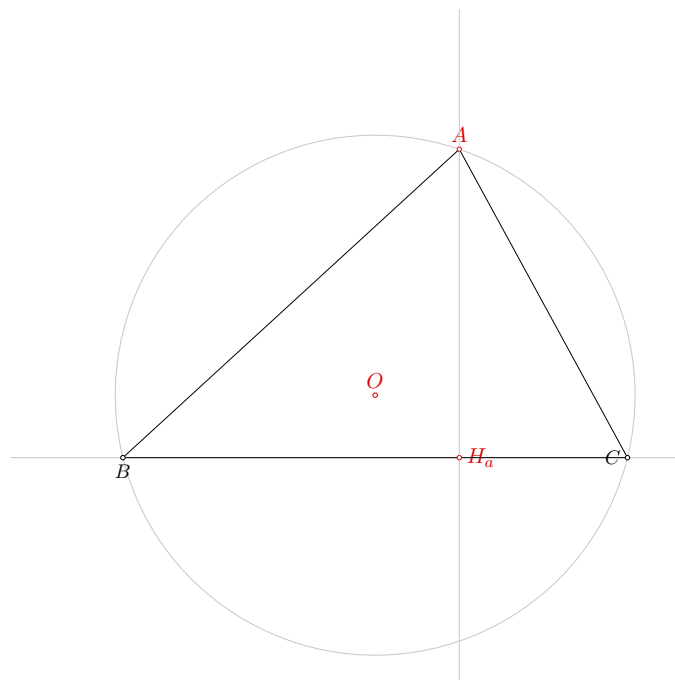
Слика 4.9: Статистика броја коришћених дефиниција у проблемима из Верниковог и Конелијевог корпуса



Слика 4.10: Статистика броја коришћених лема у проблемима из Верниковог и Конелијевог корпуса



Слика 4.11: Статистика броја коришћених примитивних конструкција у проблемима из Верниковог и Конелијевог корпуса



Слика 4.12: Аутоматски генерисана илустрација за проблем 32

4. Using the circle $k(O, C)$ and the line a construct a point C and a point B (rule W04);

% NDG: line a and circle $k(O, C)$ intersect.

Non-degenerate conditions: line a and circle $k(O, C)$ intersect; points A and O are not the same.

Determination conditions: points A and H_a are not the same.

Rules used: [W02, W04, W06, W10a]

Definitions and lemmas used: [D5, D8, D26, GD01, L11, L12]

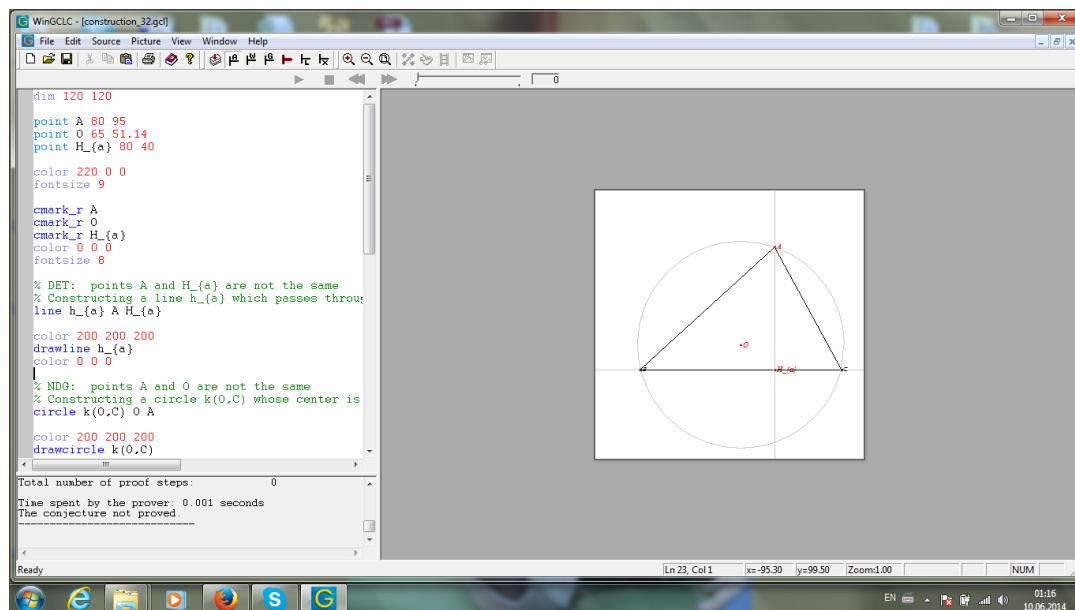
Поред излаза на природном језику, аутоматски се генерише и излаз формулисан коришћењем геометријског језика GCLC [54], којим се конструкција задаје на формалан начин, а такође се може и визуелизовати. Поред конструкције, генерише се и спецификација улаза у доказиваче који су интегрисани у алат GCLC (постоје доказивачи засновани на Вуовој методи, методи Гребневих база и методи површина) којим се доказује да генерисана конструкција задовољава дату спецификацију, тј. добија се доказ коректности конструкције. Као излаз из ових доказивача добија се доказни објекат и услови недегенерисаности.

Аутоматски генерисан излаз у GCLC-у за претходни пример има следећи облик:

```

% free points
point A 80 95
point O 65 51.14
point H_a 80 40
color 220 0 0
fontsize 9
cmark_r A
cmark_r O
cmark_r H_a
color 0 0 0
fontsize 8
% synthesized construction
% DET: points A and H_a are not the same
% Constructing a line h_a which passes through point A and point H_a
line h_a A H_a
color 200 200 200
drawline h_a
color 0 0 0
% NDG: points A and O are not the same
% Constructing a circle k(O,C) whose center is at point O and which
passes through point A
circle k(O,C) O A
color 200 200 200
drawcircle k(O,C)
color 0 0 0
% Constructing a line a which is perpendicular to line h_a and which
passes through point H_a
perp a H_a h_a
color 200 200 200
drawline a
color 0 0 0
% NDG: line a and circle k(O,C) intersect
% Constructing points C and B which are in intersection of k(O,C) and a
intersec2 C B k(O,C) a
drawsegment A B
drawsegment A C
drawsegment B C
% Non-degenerate conditions: line a and circle k(O,C) intersect; points
A and O are not the same
% Determination conditions: points A and H_a are not the same

```

Слика 4.13: Илустрација генерисане конструкције за пример A, O, H_a

```
% verification
line _b A C
towards _M_b C A 0.5
line _a B C
towards _M_a B C 0.5
perp _m_a _M_a _a
perp _m_b _M_b _b
intersec _O _m_a _m_b
perp _h_a A _a
intersec _H_a _a _h_a
prove identical O _O
prove identical H_a _H_a
```

Визуелни приказ генерисане конструкције добијен коришћењем алата GCLC дат је на слици 4.13 и слици 4.12.

Као што је већ напоменуто, за проблеме који су зависни од положаја нека од тачака у конструкцији конструише се на произвољан начин на некој правој или кругу. Код неких од тих проблема у зависности од тога где се тачка изабере нека два круга (или круг и права) која се користе у конструкцији имају или немају пресек, и као последица тога полазни проблем има или нема решење. Такав је случај на пример са проблемима 289 (O, M_b, H_b), 371 (M_a, H_a, T_a).

Аутоматски генерисана спецификација улаза у OpenGeoProver у XML формату за доказ коректности конструкције за проблем A, O, H_a има следећу форму:

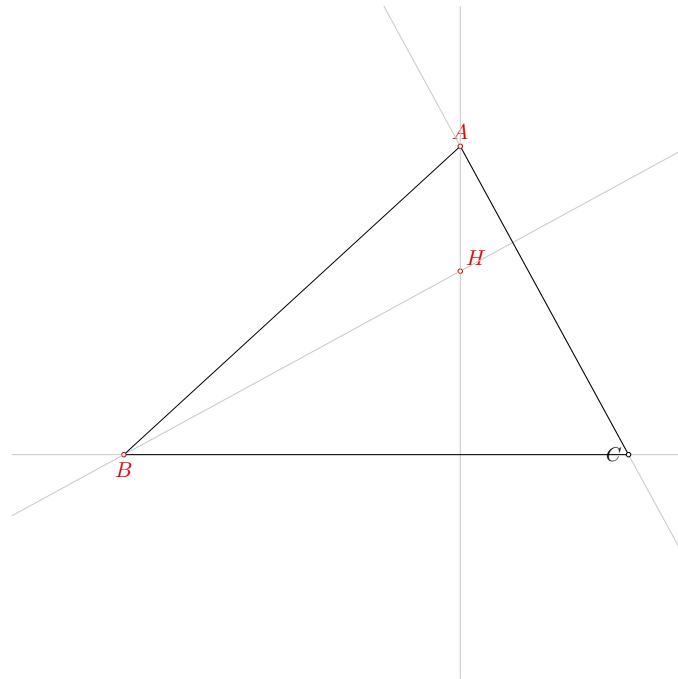
```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE geothm SYSTEM "geothm.dtd">
<geothm name="Construction no. 32">
  <constructions>
    <pfree label="$A$" />
    <pfree label="$O$" />
    <pfree label="$H_{a}$" />
    <ltwopts label="$h_{a}$" point1="$A$" point2="$H_{a}$" />
    <ccenterpt label="$k(O,C)$" center="$O$" point="$A$" />
    <lperp label="$a$" point="$H_{a}$" baseline="$h_{a}$" />
    <pintersect label="$C$" set1="$k(O,C)$" set2="$a$" />
    <pintersect label="$B$" set1="$k(O,C)$" set2="$a$" />
    <ltwopts label="$\_b$" point1="$A$" point2="$C$" />
    <psegdiv label="$\_M_{b}$" point1="$C$" point2="$A$" coeff="1" />
    <ltwopts label="$\_a$" point1="$B$" point2="$C$" />
    <psegdiv label="$\_M_{a}$" point1="$B$" point2="$C$" coeff="1" />
    <lperp label="$\_m_{a}$" point="$\_M_{a}$" baseline="$\_a$" />
    <lperp label="$\_m_{b}$" point="$\_M_{b}$" baseline="$\_b$" />
    <pintersect label="$\_O$" set1="$\_m_{a}$" set2="$\_m_{b}$" />
    <lperp label="$\_h_{a}$" point="$A$" baseline="$\_a$" />
    <pintersect label="$\_H_{a}$" set1="$\_a$" set2="$\_h_{a}$" />
  </constructions>
  <statement>
    <identicpts point1="$O$" point2="$\_O$" />
    <identicpts point1="$H_{a}$" point2="$\_H_{a}$" />
  </statement>
</geothm>

```

Најдужа генерисана “чиста” конструкција састоји се од 19 примитивних конструктивних корака (за проблем 831: E_a, I, M_a). Многе од генерисаних конструкција поклапају се са онима које се могу наћи у литератури. Ипак, код проблема за које постоји више различитих решења, оно које ће систем пронаћи зависи од редоследа примитивних конструкција/дефиниција/лема. У наставку је дат пример две различите генерисане конструкције за проблем 10: A, B, H , које су илустроване на сликама 4.14 и 4.15.

Problem 10: Given a point A , a point B , and a point H construct the triangle ABC .



Слика 4.14: Аутоматски генерисана илустрација првог решења проблема 10

Construction:

1. Using the point A and the point H , construct a line h_a (rule W02);
 % DET: points A and H are not the same;
2. Using the point B and the point H , construct a line h_b (rule W02);
 % DET: points B and H are not the same;
3. Using the point A and the line h_b , construct a line b (rule W10); ;
4. Using the point B and the line h_a , construct a line a (rule W10); ;
5. Using the line b and the line a , construct a point C (rule W03);
 % NDG: lines b and a are not parallel
 % DET: lines b and a are not the same.

Non-degenerate conditions: lines b and a are not parallel.

Determination conditions: lines b and a are not the same; points B and H are not the same; points A and H are not the same.

Rules used: [W02, W03, W10]

Definitions and lemmas used: [D3, D8, D9, GD01]

Problem 10: Given a point A , a point B , and a point H construct the triangle ABC .

Construction:

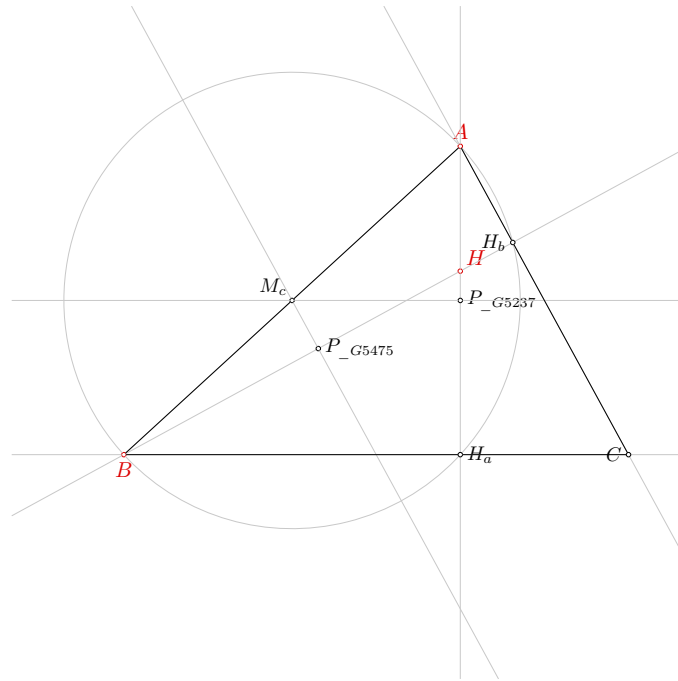
1. Using the point A and the point B , construct a point M_c (rule W01);
2. Using the point A and the point H , construct a line h_a (rule W02);
% DET: points A and H are not the same;
3. Using the point B and the point H , construct a line h_b (rule W02);
% DET: points B and H are not the same;
4. Using the point A and the point M_c , construct a circle $k(M_c, A)$ (rule W06);
% NDG: points A and M_c are not the same;
5. Using the circle $k(M_c, A)$, the line h_a , the point M_c and the point A , construct a point H_a (rule W05);
% NDG: line h_a and circle $k(M_c, A)$ intersect;
6. Using the point H_a and the point B , construct a line a (rule W02);
% DET: points H_a and B are not the same;
7. Using the circle $k(M_c, A)$, the line h_b , the point M_c and the point B , construct a point H_b (rule W05);
% NDG: line h_b and circle $k(M_c, A)$ intersect;
8. Using the point H_b and the point A , construct a line b (rule W02);
% DET: points H_b and A are not the same;
9. Using the line a and the line b , construct a point C (rule W03);
% NDG: lines a and b are not parallel
% DET: lines a and b are not the same.

Non-degenerate conditions: lines a and b are not parallel; line h_b and circle $k(M_c, A)$ intersect; line h_a and circle $k(M_c, A)$ intersect; points A and M_c are not the same.

Determination conditions: lines a and b are not the same; points H_b and A are not the same; points H_a and B are not the same; points B and H are not the same; points A and H are not the same.

Rules used: [W01, W02, W03, W05, W06]

Definitions and lemmas used: [D3, D5, D6, D8, D9, D20, GD01, GD02, L40, L41, L42]



Слика 4.15: Аутоматски генерисана илустрација другог решења проблема 10

Као што се може приметити први генерисани доказ је доста једноставнији од другог, чак је једноставнији и од оног који се може наћи у литератури [95].

Приликом решавања симетричних инстанци проблема, може се десити да се генеришу суштински другачија, несиметрична решења. На пример, решивши проблеми 40: A, M_a, M_b и 210: C, M_a, M_c су симетрични у односу на дефиниције, а генерисана решења нису изоморфна. У наставку су дата решења ових проблема, илустрована на сликама 4.16 и 4.17.

Аутоматски генерисана конструкција за проблем 40: A, M_a, M_b има следећу форму:

Problem 40: Given a point A , a point M_a and a point M_b , construct the triangle ABC .

Construction:

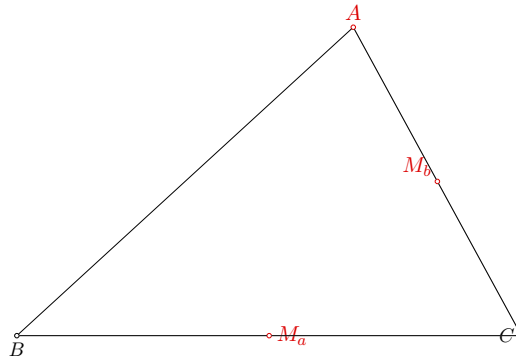
1. Using the point A and the point M_b , construct a point C (rule W01);
2. Using the point M_a and the point C , construct a point B (rule W01);
 % NDG: points M_a and C are not the same.

Non-degenerate conditions: points M_a and C are not the same.

Determination conditions: .

Rules used: [W01]

Definitions and lemmas used: [D21, D22, GL03, GL04]



Слика 4.16: Аутоматски генерисана илустрација за проблем 40

За проблем 210: C, M_a, M_c генерише се наредна конструкција:

Problem 210: Given a point C , a point M_a and a point M_c , construct the triangle ABC .

Construction:

1. Using the point C and the point M_a , construct a point B (rule W01);
2. Using the point C and the point M_c , construct a point G (rule W01);
3. Using the point M_a and the point G , construct a point A (rule W01).

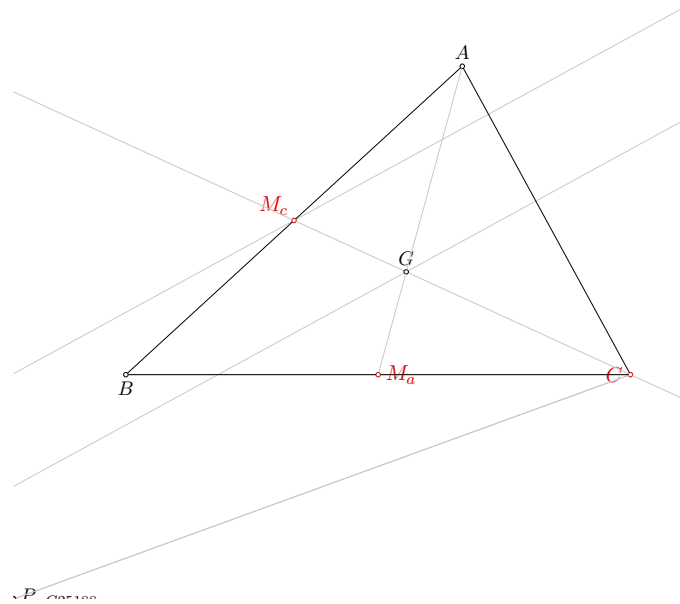
Non-degenerate conditions: .

Determination conditions: .

Rules used: [W01]

Definitions and lemmas used: [D21, GL03, GL04, L55, L57]

Такође, проблеми 88: A, H_b, T_b и 185: B, H_c, T_c зависни су од положаја и међусобно симетрични у односу на дефиниције, а добијена решења нису симетрична. Подсетимо се да решавање проблема за које се утврди да су зависни од положаја креће тако што се један од датих објеката конструише на произвољан начин тако да припада скупу тачака одређеним са друга два објекта, а затим се конструкција наставља на уобичајен начин. Јасно је да је у случају ових



Слика 4.17: Аутоматски генерисана илустрација за проблем 210

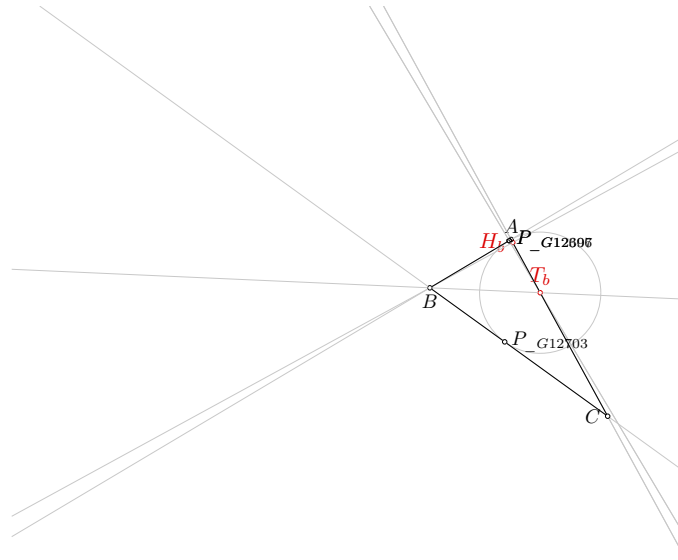
проблема дато мање информација (јер дати објекти нису међусобно независни) те ће у неком моменту конструкције бити неопходно да се неки од објеката конструише на произвољан начин (тако да припада неком скупу тачака или не).

Генерисана конструкција на природном језику за проблем 88: A, H_b, T_b (илустрована на слици 4.18) има следећу форму:

Problem 88: Given a point A , a point H_b and a point T_b , construct the triangle ABC .

Construction:

1. Using the point H_b and the point T_b , construct a line b (rule W02);
% DET: points H_b and T_b are not the same;
2. Choose freely a point A on the line b (rule WOnline1) ;
3. Using the point H_b and the line b , construct a line h_b (rule W10);
4. Choose freely a point B on the line h_b (rule WOnline2);
5. Using the point B and the point A , construct a line c (rule W02);
% DET: points B and A are not the same;



Слика 4.18: Аутоматски генерисана илустрација за проблем 88

6. Using the point T_b and the line c , construct a circle $k(T_b, foot[T_b, c])$ (rule W11);
 % NDG: point T_b is not incident to the line c ;
7. Using the circle $k(T_b, foot[T_b, c])$, the point B , the point T_b and the line c , construct a line a (rule W13);
 % NDG: point B is outside the circle $k(T_b, foot[T_b, c])$;
8. Using the line a and the line b , construct a point C (rule W03);
 % NDG: lines a and b are not parallel
 % DET: lines a and b are not the same.

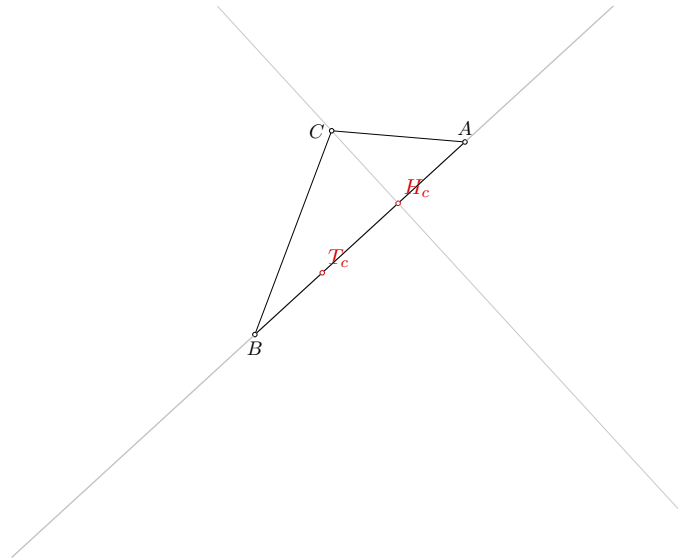
Non-degenerate conditions: lines a and b are not parallel; point B is outside the circle $k(T_b, foot[T_b, c])$; point T_b is not incident to the line c .

Determination conditions: lines a and b are not the same; points B and A are not the same; points H_b and T_b are not the same.

Rules used: [W02, W03, W10, W11, W13, WOnline1, WOnline2]

Definitions and lemmas used: [D24, D6, D9, GD01, GD02, GL10, GL11, L9]

У наставку је дата аутоматски генерисана конструкција на природном језику за проблем 185: B, H_c, T_c (илустрована на слици 4.19):



Слика 4.19: Аутоматски генерисана илустрација за проблем 185

Problem 185: Given a point B , a point H_c and a point T_c , construct the triangle ABC .

Construction:

1. Using the point H_c and the point T_c , construct a line c (rule W02);
 % DET: points H_c and T_c are not the same;
2. Choose freely a point B on the line c (rule WOnline1) ;
3. Using the point H_c and the line c , construct a line h_c (rule W10); ;
4. Choose freely a point A on the line c (rule WOnline1) ;
5. Choose freely a point C on the line h_c (rule WOnline2).

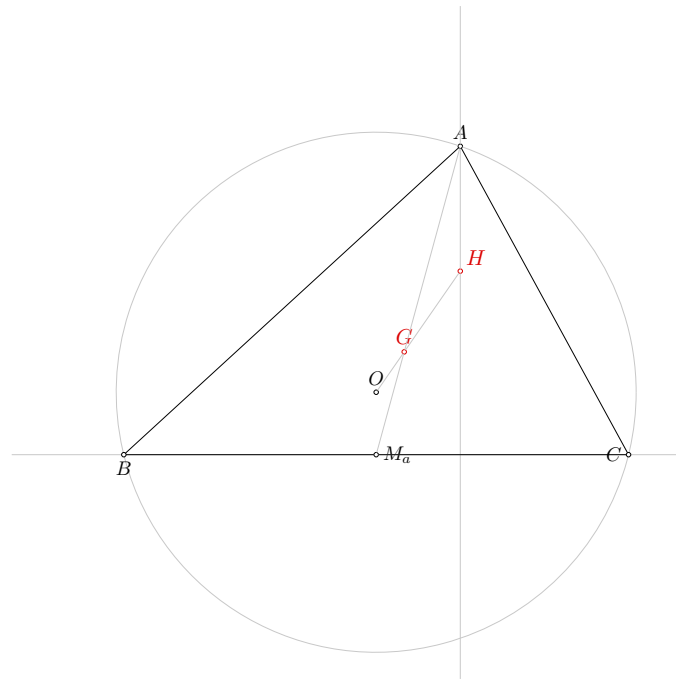
Non-degenerate conditions: .

Determination conditions: points H_c and T_c are not the same.

Rules used: [W02, W10, WOnline1, WOnline2]

Definitions and lemmas used: [D10, D25, D7]

Следи и пример генерисане конструкције за редундантан проблем. У питању је проблем 308: G, H, O за чије задате тачке важи да припадају Ојлеровој правој и важи однос: $\overrightarrow{HG}/\overrightarrow{GO} = 2$. Према томе, једна од ових тачака може



Слика 4.20: Аутоматски генерисана илустрација за проблем 308

се конструисати на основу остале две. У овом решењу се једно од темена конструира на произвољан начин, а затим се конструкција наставља на уобичајен начин. Илустрација генерисаног решења дата је на слици 4.20.

Problem 308: Given a point G , a point H , and a point O , construct the triangle ABC .

Construction:

1. Using the point G and the point H , construct a point N (rule W01);
2. Using the point G and the point H , construct a point O (rule W01);
3. Choose freely a point A (rule free);
4. Using the point A and the point G , construct a point M_a (rule W01);
5. Using the point A and the point H , construct a point E_a (rule W01);
6. Using the point A and the point H , construct a line h_a (rule W02);
 % DET: points A and H are not the same;
7. Using the point A and the point O , construct a circle $k(O, C)$ (rule W06);
 % NDG: points A and O are not the same;

8. Using the point M_a and the point N , construct a circle $k(N, M_a)$ (rule W06);
% NDG: points M_a and N are not the same;
9. Using the circle $k(N, M_a)$, the line h_a , the point N and the point E_a , construct a point H_a (rule W05);
% NDG: line h_a and circle $k(N, M_a)$ intersect
% DET: points E_a and H_a must be different;
10. Using the point H_a and the point M_a , construct a line a (rule W02);
% DET: points H_a and M_a are not the same;
11. Using the circle $k(O, C)$ and the line a , construct a point C and a point B (rule W04);
% NDG: line a and circle $k(O, C)$ intersect.

Non-degenerate conditions: line a and circle $k(O, C)$ intersect; line h_a and circle $k(N, M_a)$ intersect; points M_a and N are not the same; points A and O are not the same.

Determination conditions: points H_a and M_a are not the same; points E_a and H_a must be different; points A and H are not the same.

Rules used: [W01, W02, W04, W05, W06, free]

Definitions and lemmas used: [D21, D26, D28, D3, D32, D5, D8, GD01, GL03, GL04, GL09, L11, L12, L16, L19, L22, L55, L58]

4.7 Поређење са другим системима

Упркос дугој традицији решавања конструктивних задатака уз помоћ лењира и шестара, њихово аутоматско решавање једва да се помиње у рачунарству. Постоји тек неколико радова на тему аутоматизације решавања конструктивних проблема у геометрији коришћењем геометријског приступа [59, 16, 90, 68, 47] и неколико радова на тему алгебарског решавања истог проблема [17, 42]. Ниједна од ових метода не разматра доказивање исправности добијене конструкције.

Шрек је 1994. године развио у PROLOG-у систем Progé [90]. За разлику од приказаног приступа конструктивним проблемима у коме је циљ разматрати добро дефинисани корпус са коначним бројем конструктивних проблема, систем Progé представља општи оквир у коме експерт може да пројектује геометријски универзум којем посматрани проблем припада. У оквиру овог система дефинисани су генерички механизми за руковање знањем на више различитих нивоа,

независно од инстанце геометријског универзума која се користи. У оквиру система се не разматрају ни дефиниције нити леме, већ се користе типови за типизацију објеката, функцијски и предикатски симболи, као и скуп аксиома. Аксиоме нису нужно независне нити ниског нивоа као су у аксиоматизацијама геометрије. Функцијски симболи означавају и конструктибилност и конструкцију. У случају да функцијски симболи треба да буду интерпретирани као мултифункције, као резултат се добија листа. Progé се бави искључиво конструктибилношћу, те се питања неконструктибилности не могу решити помоћу њега. Progé је развијен са идејом да и наставницима и студентима обезбеди појмове и језик који је близак њиховом искуству везаном за геометрију, а онда се база знања дели у природне компоненте у складу са тим поимањем геометрије.

Приступ који су развили Гао и Чу [42] свим тачкама додељује координате и након тога се геометријска ограничења трансформишу у скуп једначина. Након тога користи се или Вуова метода или метода Гребнерових база да би се извршила триангулација добијеног система јер се овакви системи много једноставније решавају. На овај начин одређују се позиције непознатих објеката на основу позиција познатих геометријских објеката и на основу скупа задатих ограничења. У обзир се узимају и имплицитне информације изведене из осталих ограничења. Трансформацијом скупа ограничења у алгебарску форму може се добити процедура одлучивања да ли се неки дијаграм може конструирати коришћењем лењира и шестара.

Постоји и синтетичка метода за аутоматско генерисање конструкцијских корака у геометријским проблемима који су развили Чу, Гао и Занг и који као улаз очекује декларативни опис геометријске фигуре, а као излаз даје процедуру како нацртати ту фигуру коришћењем лењира и шестара [20]. У овој методи се такође гради база знања, али која је локална за дати конструктивни проблем – она се састоји из својстава која се могу извести из скупа ограничења којим је проблем задат коришћењем фиксног скупа геометријских аксиома. Они не разматрају конструктивне примитиве, већ се за сваки од објеката (права, круг, ...) задаје са колико је својстава задат и каква треба да буду та својства.

Резултати који би се сматрали најближим овој тематици тичу се решавања геометријских ограничења у дизајну уз помоћ рачунара [1, 79, 12, 34], али су тамо циљеви доста другачији [90, 91].

Ниједна од поменутих метода за решавање конструктивних проблема не разматра доказивање коректности решења. На пример, метода коју је развио Галвани [47] изведен је из опште методе за тестирање и синтезу делова софтвера. Метода долази до формалне конструкције коришћењем вероватносног приступа

за налажење конкретног решења које служи за навођење претраге у великом простору формалних функционалних термова. За овакву методу доказ коректности је заиста неопходан.

Сви поменути системи су засновани на *универзалном приступу*, тј. са идејом да се дефинишу општи механизми и општи оквир у коме је могуће креирати свој геометријски универзум коме разматрани проблем припада. За разлику од њих, систем ArgoTriCS је заснован на *корпусном приступу* у коме је циљ размотрити добро дефинисани корпус који се састоји од коначног скупа конструктивних проблема са намером да се реши комплетно и да може да се користи у оквиру неког образовног софтвера.

Од конструктивних проблема разматраних у раду који описује Галванијев метод [47] систем ArgoTriCS успешно решава све проблеме у којима је задатак да се конструише троугао уколико су познате позиције три значајне тачке (таквих је 4 од укупно 25 проблема); систем ArgoTriCS тренутно може да реши само мали проценат проблема које су разматрали Гао и Чу, јер многи од њих нису проблеми у којима је потребно конструисати троугао, а чак и када то јесте циљ, често нису задате позиције значајних тачака у троуглу, већ нека ограничења која још увек нису уграђена у наш систем (растојања тачака, величине углова).

Нека истраживања основа геометрије такође разматрају геометријске конструкције са циљем дефинисања конструктивне геометрије кроз елиминацију квантификатора и коришћење функционалних симбола [80, 7].

Постоје, такође, и истраживања која се баве дефинисањем језика погодног за описивање геометријских конструкција, који би омогућио и анализу конструкција датих на том језику [44]. Поред овога, циљ ових истраживања је и омогућити да се провере нека својства конструкција, као на пример еквиваленција конструкција или једнакост углова или растојања у конструкцијама.

4.8 Формализација

Са логичке тачке гледишта, решавање конструктивног проблема подразумева доказивање теореме облика:

$$\forall X \exists Y. \Psi(X, Y) \quad (4.1)$$

на интуиционистички начин. Пронађени сведок за Y представља *конструкцију* и може да укључује само тачке које је могуће конструисати коришћењем лењира и шетара почев од скупа X .

Ипак, немају сви конструктивни проблеми решење: неки проблеми немају решење, а неки га имају само под неким додатних скупом услова, који нису унапред познати. Стога за дате објекте X могу бити дата нека ограничења, те је уместо доказивања једначине (4.1), често задатак открити $\Phi(X)$ и доказати тврђење:

$$\forall X.(\Phi(X) \Rightarrow \exists Y.\Psi(X, Y)) \quad (4.2)$$

Горњим тврђењем се доказује да решење постоји под неким скупом услова. Али могуће је тврдити и више:

$$\forall X.(\Phi(X) \Rightarrow \exists Y.\Psi(X, Y) \wedge \neg\Phi(X) \Rightarrow \neg\exists Y.\Psi(X, Y)) \quad (4.3)$$

Овим тврђењем се добија потпуна карактеризација решивости проблема: тврди се да решење постоји под неким скупом услова Φ а да иначе не постоји решење. Често се услови решивости не могу изразити само у терминима датих објеката X , већ морају да укључе и неке помоћне објекте (који се користе током конструкције).

Приликом решавања конкретних класа конструктивних проблема, неки циљни услови се могу подразумевати. На пример, приликом решавања проблема конструкције троугла, имплицитни циљни услов је да конструисане тачке A , B и C не буду колинеарне.

Као што је већ напоменуто у поглављу 2.2, решење конструктивног проблема, традиционално, укључује наредне четири фазе/компоненте [33]:

Анализа: У анализи се креће од претпоставке да одређени геометријски објекти задовољавају услове проблема $\Psi(X, Y)$ и доказује се да важе особине $Plans(X, Y)$ које омогућавају конструкцију;

Конструкција: Конструкција је заснована на анализи, то јест, на процедуралном пандану спецификације $\Psi(X, Y)$ коришћењем лењира и шестара;

Доказ: У фази доказивања треба доказати да конструисане фигуре задовољавају услове $\Psi(X, Y)$ (могуће под неким додатним условима);

Дискусија: Дискусија треба да одреди потребне и довољне услове да би решење постојало и, такође, требало би да се размотри колико различитих решења проблема постоји. У идеалном случају број решења би требало да буде изражен у функцији међусобног односа датих објеката, али понекад је

довољно изразити га у функцији међусобног односа објеката добијених током конструкције.

У претходним радовима на тему геометријских конструкција прве две фазе су обично стављане у први план, док се друге две ређе помињу (а врло ретко их је лако добити). У тексту који следи користићемо интерактивни доказивач теорема Isabelle и аутоматски доказивач теорема за кохерентну логику ArgoCLP у циљу генерисања машински проверивих решења проблема конструкције троугла [69]. Као мотивациони пример користићемо проблем 4: A, B, G из Верникове листе:

Ако су дате тачке A, B и G , конструисати троугао ABC , тако да је тачка G тежиште троугла ABC .

У овом проблему, $\Psi(X, Y)$ означава својство $\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)$, односно задатак је доказати:

$$\forall A, B, G. (? \Leftrightarrow \exists C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C))),$$

при чему предикат $\text{centroid}(G, A, B, C)$ означава да је G тежиште троугла ABC , предикат $\text{collinear}(A, B, C)$ да су тачке A, B и C колинеарне, а $?$ је услов, који унапред није познат, а који карактерише када је проблем решив.

4.8.1 Анализа

Сврха анализе је да се идентификује знање које је довољно за процедуралну спецификацију $\text{Plans}(X, Y')$ конструкције за дату декларативну спецификацију $\Psi(X, Y)$. Прецизније, анализа се састоји из низа тврђења следеће форме, за $k = 1, \dots, n$:⁷

$$\forall X, Y'. (\Phi_a(X) \wedge \Psi(X, Y) \wedge \text{Def}(X, Y') \wedge \bigwedge_{i=1}^{k-1} \text{Rel}_i(X, Y'_i) \Rightarrow \text{Rel}_k(X, Y'_k)) \quad (4.4)$$

при чему важи:

- Y' је низ променљивих $y_1, \dots, y_n$ тако да важи $Y \subseteq Y'$ (неформално, $Y' \setminus Y$ су помоћне тачке које су конструисане и користе се у конструкцији, поред објеката из Y);
- Y'_k је низ променљивих $y_1, \dots, y_k$;

⁷У каснијим стадијумима решења, дати услов $\Phi_a(X)$ се може проширити у неки услов Φ за који важи једначина (4.3).

- y_n припада Y ;
- $\Phi_a(X)$ представља неки скуп ограничења на задатим објектима (потенцијално $\top$, ако нема ограничења);
- $Def(X, Y')$ уводи својства објеката из скупа $Y' \setminus Y$;
- $Rel_k(X, Y'_k)$ је формула која одговара ефективном начину конструисања y_k , коришћењем лењира и шестара из X и Y'_{k-1} .⁸

Означимо $\bigwedge_{i=1}^n Rel_i(X, Y'_i)$ са $Cons(X; Y')$. Из горњег низа теорема, следи наредно тврђење:

$$\forall X, Y'. (\Phi_a(X) \wedge \Psi(X, Y) \wedge Def(X, Y') \Rightarrow Cons(X; Y')) \quad (4.5)$$

У циљу омогућавања конструкције као ефективне процедуре, потребно је имплицитне односе $Rel_i(X, Y'_i)$ (за $i = 1$ до n) претворити у форму $\bigvee_{k=1}^{K_i} y_i = RC_{i,k}(X, Y'_i)$ [91, 36], која изражава начин(е) на који се y_i може добити из X и Y'_i коришћењем лењира и шестара.⁹ У горњој формули K_i означава број различитих начина на које се може конструисати y_i . Овај број мора бити коначан, иако неки начини могу да укључују бесконачне изборе. На пример, може се десити да је y_i или пресечна тачка правих p и q или произвољна тачка на правој r . Мора да важи:

$$\forall X, Y'. (Rel_i(X, Y'_i) \Leftrightarrow \bigvee_{k=1}^{K_i} y_i = RC_{i,k}(X, Y'_i)) \quad (4.6)$$

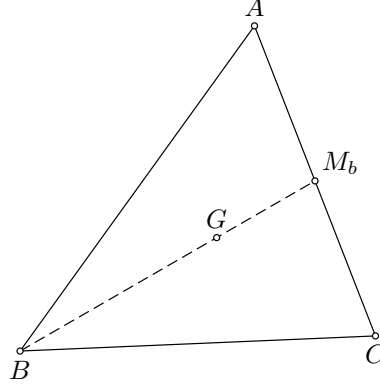
С обзиром на то да $Cons(X; Y')$ означава $\bigwedge_{i=1}^n Rel_i(X, Y'_i)$, такође важи:

$$\forall X, Y'. (Cons(X; Y') \Leftrightarrow \bigwedge_{i=1}^n \bigvee_{k=1}^{K_i} y_i = RC_{i,k}(X, Y'_i)) \quad (4.7)$$

Ако означимо са $Plan_j(X, Y')$ конјункцију $\bigwedge_{i=1}^n y_i = RC_{i,k_i}(X, Y'_i)$, за неко $k \in \{1, \dots, K_i\}$, за свако $i = 1, \dots, n$, онда се, коришћењем дистрибутивности,

⁸Ова формула може да укључује дисјункције које одговарају различитим случајевима за X и Y . На пример, $(A \neq B \wedge midpoint(C, A, B)) \vee (A = B \wedge C = A)$

⁹Строго говорећи, функције $RC_{i,k}$ могу да укључују и више од лењира и шестара. На пример, може бити случај да се може изабрати само једна од пресечних тачака два круга (на пример, „која се разликује од тачке...“, „која није са исте стране...“, итд.). Такође, неки од $RC_{i,k}$ могу бити недетерминистички, на пример „изабери на случајан начин тачку на правој...“.



Слика 4.21: Илустрација решења за мотивациони пример

добија неких J дисјунката као појединачни конструкцијски планови:

$$\forall X, Y'. (Cons(X; Y') \Leftrightarrow \bigvee_{j=1}^J Plan_j(X, Y')) \quad (4.8)$$

Ако означимо $\bigvee_{j=1}^J Plan_j(X, Y')$ са $Plans(X, Y')$, из горње формуле и формуле (4.5), важи:

$$\forall X, Y'. (\Phi_a(X) \wedge \Psi(X, Y) \wedge Def(X, Y') \Rightarrow Plans(X, Y')) \quad (4.9)$$

С обзиром на то да нас интересују ефективне конструкције изражене кроз $Plans(X, Y')$, и ако се уведе у $Y' \setminus Y$ помоћни објекти који су потребни у конструкцији, онда је неопходно да се они могу дефинисати за свако решење. Овај захтев изражен за Def има следећу форму:

$$\forall X, Y. (\Phi_a(X) \wedge \Psi(X, Y) \Rightarrow \exists Y' \setminus Y. Def(X, Y')) \quad (4.10)$$

Постоји суптилан услов за $Plans(X, Y')$ — он треба да буде довољно прецизан да омогући конструкцију, али такође и довољно јак да би се доказало да конструисани објекти задовољавају спецификацију.

Због специфичног циља, анализа је више процедура претраге, него процедура доказивања. Она се може имплементирати као процес претраге, док на крају, изводи захтевану формулу.

Пример 4.3. Нека предикат $sratio(P, Q, R, S, m, n)$ означава да важи однос $\overrightarrow{PQ} \cdot n = \overrightarrow{RS} \cdot m$, а нека $midpoint(P, Q, R)$ означава да је тачка P средиште

дужи QR . Први корак извођења мотивационог примера гласи (слика 4.21):

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C) \wedge \text{midpoint}(M_b, A, C) \\ & \Rightarrow \text{sratio}(B, M_b, B, G, 3, 2)) \end{aligned}$$

а други корак извођења:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C) \wedge \text{midpoint}(M_b, A, C) \\ & \wedge \text{sratio}(B, M_b, B, G, 3, 2) \\ & \Rightarrow \text{sratio}(A, C, A, M_b, 2, 1)) \end{aligned}$$

Када се ова два корака искомбинују добија се:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C) \wedge \text{midpoint}(M_b, A, C) \\ & \Rightarrow \text{sratio}(B, M_b, B, G, 3, 2) \wedge \text{sratio}(A, C, A, M_b, 2, 1)) \end{aligned}$$

Овде је $\Phi_a(A, B, G)$ једнако $\top$ (што значи да не постоје нека ограничења за A, B, G , али може бити неких предуслова који ће бити додати током фазе доказивања), $\text{Def}(A, B, C, G, M_b)$ је $\text{midpoint}(M_b, A, C)$, а $\text{Cons}(A, B, G; M_b, C)$ је $\text{sratio}(B, M_b, B, G, 3, 2) \wedge \text{sratio}(A, C, A, M_b, 2, 1)$.

Нека је sratioF парцијална функција тако да важи $Q = \text{sratioF}(P, R, S, m, n)$ ако је $\overrightarrow{PQ} \cdot n = \overrightarrow{RS} \cdot m$. У мотивационом примеру не постоје другачији начини конструкције, па је стога K_i једнако 1. Тада важи следеће:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (\text{sratio}(B, M_b, B, G, 3, 2) \wedge \text{sratio}(A, C, A, M_b, 2, 1)) \\ & \Rightarrow (M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1)) \end{aligned}$$

и директна последица претходне две формуле је:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C) \wedge \text{midpoint}(M_b, A, C) \quad (4.11) \\ & \Rightarrow M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1)) \end{aligned}$$

У складу са формулом (4.10), добијамо наредно тврђење које се једноставно доказује:

$$\begin{aligned} & \forall A, B, C, G. \\ & (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)) \\ & \Rightarrow \exists M_b. (\text{midpoint}(M_b, A, C)) \end{aligned} \quad (4.12)$$

4.8.2 Конструкција

Фаза анализе даје формулу која омогућава ефективне конструкције. За свако $j \in \{1, \dots, J\}$, $\text{Plan}_j(X, Y')$ даје један *конструкцијски план* у форми:

- дати су објекти X (као *слободни* објекти);
- за $i = 1$ до n
конструиши y_i као $y_i = RC_{i,k}(X, Y'_i)$ (за неко $k \in \{1, \dots, K_i\}$)

При томе се могу користити и сложени конструкцијски кораци (на пример, конструкција средишта дужи), те стога треба доказати да се сваки од $RC_{i,k}$ може изразити коришћењем лењира и шестара.

Пример 4.4. Конструкција, изведена из формуле $\text{sratio}(B, M_b, B, G, 3, 2) \wedge \text{sratio}(A, C, A, M_b, 2, 1)$ има следећу форму:

1. дате су тачке A, B и G (као слободне тачке);
2. $M_b = \text{sratioF}(B, B, G, 3, 2)$;
3. $C = \text{sratioF}(A, A, M_b, 2, 1)$.

4.8.3 Доказ

Током фазе доказивања, треба доказати коректност сваког конструкцијског плана $\text{Plan}_j(X, Y')$. Треба доказати:

$$\forall X.Y'. (\Phi_a(X) \wedge ? \wedge \text{Plan}_j(X, Y') \Rightarrow \Psi(X, Y)) \quad (4.13)$$

при чему је са $?$ означен неки услов који тек треба открити.

Аутоматски доказивачи теорема за геометрију обично подржавају процедуралну репрезентацију геометријске конструкције и могу да раде над тврђењима у горњем формату.

Прво се покушава доказати тврђење:

$$\forall X.Y'.(\Phi_a(X) \wedge Plan_j(X, Y') \Rightarrow \Psi(X, Y)) \quad (4.14)$$

Ако је тврђење доказано коришћењем аутоматског доказивача теорема, он, такође, може да врати и неке услове недегенерисаности (треба обратити пажњу да ови услови недегенерисаности нису нужно најслабији услови под којима важи тврђење), те је доказано слабије тврђење:

$$\forall X, Y'.(\Phi_a(X) \wedge NDG(X, Y') \wedge Plan_j(X, Y') \Rightarrow \Psi(X, Y)) \quad (4.15)$$

Пример 4.5. *Треба доказати формулу:*

$$\begin{aligned} &\forall A, B, G, M_b, C. \\ &(? \wedge M_b = sratioF(B, B, G, 3, 2) \wedge C = sratioF(A, A, M_b, 2, 1) \\ &\Rightarrow centroid(G, A, B, C) \wedge \neg collinear(A, B, C)) \end{aligned}$$

при чему је са ? означен неки скуп предуслова изражен у терминима тачака A , B и G који тек треба открити.

Кренимо од дела који се односи на $centroid(G, A, B, C)$. Претпоставимо да доказивач теорема подржава функцију $sratioF$, али не и предикат $centroid$ и да имамо наредну дефиницију тежишта:

$$\begin{aligned} &\forall A, B, C, M_a, M_b. \\ &(M_a = sratioF(B, B, C, 1, 2) \wedge M_b = sratioF(C, C, A, 1, 2) \wedge \\ &G = intersec(AM_a, BM_b) \Rightarrow centroid(G, A, B, C)) \end{aligned}$$

Аутоматском (на пример, алгебарском) доказивачу се може проследити

наредно тврђење:

$$\begin{aligned} & \forall A, B, G, M_b, C, M'_a, M'_b, G'. \\ & (M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1) \wedge \\ & M'_a = \text{sratioF}(B, B, C, 1, 2) \wedge M'_b = \text{sratioF}(C, C, A, 1, 2) \wedge \\ & G' = \text{intersec}(AM'_a, BM'_b) \Rightarrow G = G') \end{aligned}$$

На пример, горња теорема је доказана доказивачем заснованим на Вуовој методи, која је имплементирана у оквиру доказивача *OpenGeoProver* [66]. Доказивач успешно доказује ово тврђење и враћа наредне услове недегенерисаности: “праве AM'_a и BM'_b нису паралелне и тачке A и M'_a нису идентичне” ($\neg \text{parallel}(AM'_a, BM'_b) \wedge A \neq M'_a$), стога смо доказали наредно тврђење:

$$\begin{aligned} & \forall A, B, G, M_b, M'_a, M'_b, C. \\ & \neg \text{parallel}(AM'_a, BM'_b) \wedge A \neq M'_a \wedge \\ & M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1) \wedge \\ & M'_a = \text{sratioF}(B, B, C, 1, 2) \wedge M'_b = \text{sratioF}(C, C, A, 1, 2) \\ & \Rightarrow \text{centroid}(G, A, B, C) \end{aligned}$$

Треба, такође, доказати да тачке A , B и C нису колинеарне (под неким условом). Лако се доказује (коришћењем доказивача за кохерентну логику *ArgoCLP* [98]) да важи тврђење:

$$\begin{aligned} & \forall A, B, C, G, M_b. \\ & (\text{collinear}(A, B, C) \wedge M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1) \\ & \Rightarrow \text{collinear}(A, B, G)) \end{aligned}$$

а његова контрапозиција гласи:

$$\begin{aligned} & \forall A, B, C, G, M_b. \\ & (\neg \text{collinear}(A, B, G) \wedge M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1) \\ & \Rightarrow \neg \text{collinear}(A, B, C)) \end{aligned}$$

Формула $\text{collinear}(A, B, G)$ је откривена испробавањем коначног броја предикатских симбола над датим тачкама A, B и G .

Из претходних теорема следи:

$$\begin{aligned}
& \forall A, B, G, M_b, M'_a, M'_b, C. \\
& (\neg \text{parallel}(AM'_a, BM'_b) \wedge A \neq M'_a \wedge \neg \text{collinear}(A, B, G) \wedge \\
& M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1) \wedge \\
& M'_a = \text{sratioF}(B, B, C, 1, 2) \wedge M'_b = \text{sratioF}(C, C, A, 1, 2) \\
& \Rightarrow \text{centroid}(G, A, B, C) \wedge \neg \text{collinear}(A, B, C))
\end{aligned} \tag{4.16}$$

4.8.4 Дискусија

Подсетимо се, да у општем случају, желимо да одредимо потребне и довољне услове да би постојало решење конструктивног проблема (и да одредимо број решења проблема). У идеалном случају, током фазе дискусије би требало да докажемо тврђење облика (4.3). Да бисмо поједноставили причу, претпоставимо да $\text{Plans}(X, Y')$ има само један дисјункт (то јест један конструкцијски план), али се сва наредна разматрања могу једноставно уопштити на случај када има више дисјункта.

Из фазе анализе следи:

$$\forall X, Y'. (\Phi_a(X) \wedge \Psi(X, Y) \wedge \text{Def}(X, Y') \Rightarrow \text{Plans}(X, Y')) \tag{4.17}$$

као и:

$$\forall X. (\exists Y'. (\Phi_a(X) \wedge \Psi(X, Y) \wedge \text{Def}(X, Y')) \Rightarrow \exists Y'. \text{Plans}(X, Y')) \tag{4.18}$$

Из горње формуле и једначине (4.10) добија се:

$$\forall X. (\exists Y. (\Phi_a(X) \wedge \Psi(X, Y)) \Rightarrow \exists Y'. \text{Plans}(X, Y')) \tag{4.19}$$

и

$$\forall X. (\Phi_a(X) \Rightarrow (\exists Y. \Psi(X, Y) \Rightarrow \exists Y'. \text{Plans}(X, Y'))) \tag{4.20}$$

С друге стране, из фазе доказивања следи:

$$\forall X, Y'. (\Phi_a(X) \wedge \text{NDG}(X, Y') \wedge \text{Plans}(X, Y') \Rightarrow \Psi(X, Y)) \tag{4.21}$$

те стога следи и:

$$\forall X.(\exists Y'.(\Phi_a(X) \wedge NDG(X, Y') \wedge Plans(X, Y')) \Rightarrow \exists Y.\Psi(X, Y)) \quad (4.22)$$

као и:

$$\forall X.(\Phi_a(X) \Rightarrow (\exists Y'.(NDG(X, Y') \wedge Plans(X, Y')) \Rightarrow \exists Y.\Psi(X, Y))) \quad (4.23)$$

Стога, добијени су потребни (4.20) и довољни (4.23) услови за $\exists Y.\Psi(X, Y)$ (под претпоставкама $\Phi_a(X)$). Ипак, ови услови нису једнаки, те стога још увек није дата комплетна карактеризација решивости проблема. Може се покушати да се открије¹⁰ формула $\Phi_d(X)$ тако да важи:

$$\forall X.(\Phi_a(X) \Rightarrow (\Phi_d(X) \Rightarrow \exists Y'(NDG(X, Y') \wedge Plans(X, Y')))) \quad (4.24)$$

Ако важи:

$$\forall X.(\Phi_a(X) \Rightarrow (\exists Y.\Psi(X, Y) \Rightarrow \Phi_d(X))) \quad (4.25)$$

онда је $\Phi_a(X) \wedge \Phi_d(X)$ тражена формула $\Phi(X)$, и коначно смо доказали теорему (4.3).

У неким случајевима могуће је елиминисати неке од конјункта $ndg(X, Y')$ формуле $NDG(X, Y')$. На пример, неки конјункт може да следи из неког другог конјункта, па у том случају први конјункт се може изоставити. Такође, ако важи:

$$\forall X, Y'.(Plans(X, Y') \Rightarrow ndg(X, Y')) \quad (4.26)$$

може се елиминисати $ndg(X, Y')$ из формуле $NDG(X, Y')$ у једначини (4.21). У неким случајевима, на овај начин се могу елиминисати сви конјункти из $NDG(X)$ и онда ће формула $\Phi(X)$ постати $\top$.

Постоје случајеви у којима Φ_d укључује, такође, и неки Y' , али овде ће бити разматран само једноставан случај.

На овај начин добијена је карактеризација решивости конструктивног проблема. Што се броја решења проблема тиче, у случају када је проблем решив број решења једнак је производу n бројева могућих избора за свако од y_i (погледати поглавље 4.8.1).

¹⁰тестирањем коначног броја предиката над објектима из X

Пример 4.6. На основу формуле (4.11) из анализе проблема важи:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C) \wedge \text{midpoint}(M_b, A, C)) \\ & \Rightarrow M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1)) \end{aligned}$$

и стога следи:

$$\begin{aligned} & \forall A, B, G. \\ & \exists M_b, C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C) \wedge \text{midpoint}(M_b, A, C)) \\ & \Rightarrow \exists M_b, C. (M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1) \\ & \wedge \neg \text{collinear}(A, B, C)) \end{aligned}$$

а, захваљујући формули (4.12), важи и:

$$\begin{aligned} & \forall A, B, G. \\ & \exists C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)) \\ & \Rightarrow \exists M_b, C. (M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1) \quad (4.27) \\ & \wedge \neg \text{collinear}(A, B, C)) \end{aligned}$$

Није тешко доказати лему:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1) \quad (4.28) \\ & \wedge \neg \text{collinear}(A, B, C)) \\ & \Rightarrow \neg \text{collinear}(A, B, G) \end{aligned}$$

За избор: $\Phi_d(A, B, G) = \neg \text{collinear}(A, B, G)$, коришћењем формула (4.27) и (4.28), може се доказати наредна формула:

$$\begin{aligned} & \forall A, B, G. \\ & \exists C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)) \Rightarrow \neg \text{collinear}(A, B, G) \quad (4.29) \end{aligned}$$

С друге стране, из формула (4.16) из фазе доказивања, коришћењем леме:

$$\begin{aligned} & \forall A, B, C, G, M'_a, M'_b. \\ & (\neg \text{collinear}(A, B, C) \wedge M'_a = \text{sratioF}(B, B, C, 1, 2) \wedge M'_b = \text{sratioF}(C, C, A, 1, 2) \\ & \Rightarrow \neg \text{parallel}(AM'_a, BM'_b) \wedge A \neq M'_a) \end{aligned}$$

и леме:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (\neg \text{collinear}(A, B, G) \wedge M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1) \\ & \Rightarrow \neg \text{collinear}(A, B, C)) \end{aligned}$$

добуја се:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & (M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1) \wedge \neg \text{collinear}(A, B, G) \\ & \Rightarrow \neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)) \end{aligned}$$

и такође:

$$\begin{aligned} & \forall A, B, G. \\ & \exists M_b, C. (M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1) \wedge \\ & \neg \text{collinear}(A, B, G)) \\ & \Rightarrow \exists C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)) \end{aligned}$$

Из претходне теореме и теореме:

$$\begin{aligned} & \forall A, B, G, M_b, C. \\ & \neg \text{collinear}(A, B, G) \\ & \Rightarrow \exists M_b, C. (M_b = \text{sratioF}(B, B, G, 3, 2) \wedge C = \text{sratioF}(A, A, M_b, 2, 1)) \end{aligned}$$

следи:

$$\begin{aligned} & \forall A, B, G. \\ & \neg \text{collinear}(A, B, G) \Rightarrow \exists C. (\neg \text{collinear}(A, B, C) \wedge \text{centroid}(G, A, B, C)) \quad (4.30) \end{aligned}$$

чиме се добија други смер тврђења које је требало доказати.

Према томе, из формула (4.29) и (4.30) доказана је формула:

$$\forall A, B, G.$$

$$\neg collinear(A, B, G) \Leftrightarrow \exists C. (\neg collinear(A, B, C) \wedge centroid(G, A, B, C))$$

Сви докази из поменутог примера су у потпуности формализовани коришћењем интерактивног доказивача теорема Isabelle¹¹. Комплетан доказни документ, који садржи сва потребна геометријска тврђења која се користе као леме (које одговарају малим теоремама у еуклидској геометрији) и сви докази имају око 1200 линија кода. Може се очекивати да за многе конструктивне проблеме форма доказа буде врло слична.

4.8.5 Аутоматизација процеса формализације

Крајњи циљ система ArgoTriCS јесте аутоматско генерисање машински проверивих решења конструктивних проблема из Верникове листе [69]. Овај комплексан задатак захтева синергију алата за решавање конструктивних проблема, алгебарских аутоматских доказивача теорема, синтетичких аутоматских доказивача теорема и интерактивних доказивача теорема, потпомогнуту човековим навођењем и динамичким геометријским алатима за визуелизацију.

Тренутна аутоматизација је сконцентрисана на решавање проблема из поткорпуса Верниковог корпуса код којих су на улазу дате три тачке из скупа тачака: $A, B, C, M_a, M_b, M_c, G$, при чему се исти приступ може користити за комплетан Верников корпус, као и за многе друге корпуре. Овај мини-корпус састоји се из 35 проблема, од којих је 7 решивих, битно различитих.¹² Нећемо се бавити формализацијом решења редундантних проблема и проблема зависних од положаја. За проверу исправности решења биће коришћен интерактивни доказивач Isabelle/Isar.

Анализа.

Траг решења добијен из система ArgoTriCS (који у себи укључује и подскупове дефиниција, лема и конструкцијских примитива који су потребни за решавање тог проблема) преводи се у низ теорема (4.4) и теорему (4.5). Ова тврђења, заједно са релевантним аксиомама и лемама, прослеђују се као улаз

¹¹Сви докази доступни су на:

http://poincare.matf.bg.ac.rs/~vesnap/PhD_repository.zip.

¹²Формални докази проблема из овог поткорпуса доступни су на: http://poincare.matf.bg.ac.rs/~vesnap/PhD_repository.zip.

доказивачу за кохерентну логику ArgoCLP који производи машински провериве доказе [98]. ArgoTriCS тренутно може да аутоматски генерише улазне датотеке за алат ArgoCLP (који се састоји из скупа релевантних аксиома, скупа релевантних лема и теореме коју треба доказати) само за ограничен број конструктивних проблема. С обзиром на то да ArgoCLP не подржава функцијске симболе, уместо формуле (4.6) и њених последица биће доказане одговарајуће формуле које уместо функцијских симбола користе одговарајуће предикатске симболе.

Конструкција.

Формални опис конструкције коришћењем језика GCLC се аутоматски изводи из система ArgoTriCS у динамички геометријски алат GCLC где се може визуелизовати и сачувати у великом броју различитих формата, укључујући и $\text{\LaTeX}$, EPS и SVG.

Доказ.

ArgoTriCS аутоматски генерише спецификацију улаза у аутоматски доказивач теорема OpenGeoProver [66] и у доказиваче интегрисане у алат GCLC [54]. Ови алати као излаз враћају доказни објекат, као и скуп услова недегенерисаности. За велики број конструктивних проблема из Верникове и Конелијеве листе централна теорема (4.14) је успешно доказана коришћењем неког од ових доказивача.

Дискусија.

Услови недегенерисаности добијени од стране доказивача могу да укључују неке помоћне објекте. Тврђења која су потребна за превођење ових услова у оне који укључују само задате објекте могу се доказати коришћењем доказивача ArgoCLP. У циљу откривања потребног и довољног услова да би постојало решење проблема ($\neg \textit{collinear}(A, B, G)$ у мотивационом примеру) тестира се коначан број предиката над скупом датих објеката.¹³ Такође, врши се провера да ли из неког конјункта формуле NDG следи неки други. Коначно, финални доказ се добија спајањем Isabelle фрагмената доказа једноставним корацима.

¹³За све проблеме из разматраног мини корпуса показано је да услов да услов да су задате тачке неколинеарне представља и потребан и довољан услов.

4.9 Summary of Chapter Automated Solving of Construction Problems

In this chapter a tool for solving geometric construction problems from three located points is presented [68]. The key features of this system are:

1. identification of relevant geometric knowledge,
2. suitable representation of knowledge which enables efficient search,
3. relatively simple search system based on forward chaining mechanism,
4. guiding that restricts construction of new objects only to those objects that could be relevant for construction,
5. mechanisms for identification of symmetric problems, redundant problems and problems that are locus dependent,
6. mechanisms which enable connection with automated theorem provers and interactive theorem provers.

ArgoTriCS as its output generate:

1. informal description of the construction in natural language form,
2. formal specification of the construction using GCLC language, with corresponding illustration,
3. proofs that generated constructions are correct (using automated theorem provers available within OpenGeoProver and GCLC tool),
4. discussion which determines when the solution exists and how many solutions there are,
5. input files for formalization of complete solutions to construction problems.

The prover has been implemented in PROLOG. It has been tested on problems from Wernick's and Connelly's corpora and it succeeded to identify all symmetric, redundant and locus dependent problems, and to solve most solvable problems (altogether 690 problems out of 1140, while from the remaining 450 problems 202 problems are proved unsolvable and the status of 157 problems is still unknown). Solving times span from a couple of milliseconds to more than an hour. The longest proof generated consists of 19 construction steps. Most of the solutions

generated coincide with the ones found in literature. Still, sometimes some new solutions are found. Example proofs are given and a comparison with other systems for solving construction problems [90, 42, 20, 47] is given. Many of the solutions generated automatically has been proved correct using automated theorem provers (for Wernick's corpus the statistics is the following: out of 1122 statements, Wu's method integrated into OpenGeoProver [66] has proved 679, Wu's method integrated into GCLC prover [54] has proved 578, Area method 396 statements, while Gröbner basis method 410 statements; for Connelly's corpus the statistics is the following: out of 948 statements, Wu's method integrated into OpenGeoProver has proved 318, Wu's method integrated into GCLC prover has proved 296, Area method 192 statements, while Gröbner basis method 129 statements).

At the end of this chapter, a process of formalization of generated solutions is described and one example of formalized solution is given. Automation of this process is briefly described at the very end [69].

Закључци и даљи рад

У оквиру рада на овој тези основна мотивација била је аутоматизација решавања конструктивних проблема у геометрији и формализација генерисаних решења. У те сврхе развијен је систем за аутоматско решавање једне класе конструктивних проблема у геометрији. То су проблеми у којима је потребно конструисати троугао уколико су задате позиције неке три значајне тачке у троуглу. За потребе генерисања формалних доказа конструктивних проблема коришћен је доказивач за кохерентну логику, развијен током рада на тези у сарадњи са још два аутора.

Развијени систем је у стању да реши више стотина проблема што је више него што је разматрано у оквиру било ког другог сличног система. Ово је једини систем у којем се дискутује о доказима исправности генерисаних конструкција. Такође, по први пут, за неке од проблема генеришу се формални докази.

Правци за даљи рад везано за доказивање теорема у кохерентној логици, као и за решавање конструктивних проблема су многобројни.

Доказивање теорема у кохерентној логици

У оквиру ове тезе приказана је метода за доказивање теорема у кохерентној логици који се може применити на произвољну теорију у којој су аксиоме, као и тврђење које се доказује, дате у кохерентној форми. Предложена метода имплементиран је у оквиру доказивача ArgoCLP. Доказивач генерише формалне, машински провериве доказе, као и читљиве доказе у форми природног језика, који се састоје од корака уобичајених за традиционалне доказе у геометрији. Систем је успешно примењен на различите аксиоматске системе и разне теореме из стандарних уџбеника из области геометрије. Обзиром да су генерисани докази и формални и читљиви, они се могу користити како у образовне сврхе, тако и као допринос изградњи корпуса формализованог математичког знања.

Као један модул овог система развијен је систем за поједностављивање доказа у кохерентној логици и он је имплементиран у оквиру истог доказивача.

У плану је да се развијени доказивач за кохерентну логику ArgoCLP примени на што веће фрагменте различитих теорија и да се тиме на аутоматски начин генеришу и формално верификовани докази, као и читљиви докази тврђења из те теорије. Један од циљева био би аутоматско генерисање уџбеника чија је исправност формално верификована.

Што се модула за поједностављивање доказа тиче, било би интересно искористити информацију о релевантним и ирелевантним чињеницама не само у контексту добијања “чистих” и краћих доказа, већ и приликом навођења преосталог процеса доказивања (на пример, у наредним гранама доказа). На овај начин, докази се не би поједностављивали само *post-festum*, већ и током процеса доказивања. Такође, било би интересно проширити процедуру поједностављивања модулом за екстракцију лема, којим би се издвајали фрагменти доказа који се могу упарити. Ово би значајно допринело краткоћи доказа, ипак, захтевало би другачије технике од оних које су до сада коришћене. У плану је и да се процедура поједностављивања уопшти на неке богатије и изражајније логичке фрагменте.

Решавање конструктивних задатака у геометрији

Као други део ове тезе развијен је систем за аутоматско решавање класе конструктивних проблема у којима је потребно конструисати троугао уколико су задате локације три значајне тачке у троуглу. Систем је имплементиран у оквиру алата ArgoTriCS и примењен на Верников и Конелијев корпус. Према нашем сазнању, ово је први систематски приступ решавању скупа проблема из неког корпуса конструктивних проблема. Предложени систем користи систематизовано геометријско знање које се састоји из релативно малог скупа дефиниција, лема и примитивних конструкција. Систем ArgoTriCS генерише решења која укључују опис конструкције на природном језику, формални опис конструкције (коришћењем језика GCLC), доказ да конструкција одговара да тој спецификацији (коришћењем спољашњих аутоматских доказивача теорема, заснованих на методи површина, Вуовој методи и методи Гребнерових база) и услове недегенерисаности при којима решење постоји.

Систем ArgoTriCS подразумева да су све конструисане тачке различите. То значи да су генерисане конструкције увек линеарне и да никада не долази до гранања. Анализом решења проблема из Верниковог корпуса дошло се до тога да у највећем броју случајева тачке чија се различитост

претпоставља, а које су добијене као пресечне тачке два круга или праве и круга, и морају бити различите – такав је случај са паровима тачкама $(A, B), (B, C), (A, C), (A, H_a), (B, H_b), (C, H_c), (A, N_a), (B, N_b), (C, N_c)$. Уз помоћ доказивача ArgoCLP, лако би се показала различитост ових тачака, ако претпоставимо да су темена троугла ABC увек различита. У овом случају би у анализи требало имати гранање, али би једна од грана (којом се претпоставља да су неке од ових тачака различите) водила у контрадикцију. Приликом решавања проблема из Верниковог корпуса 82 пута је претпостављена различитост ових тачака.

Ипак, у неким ситуацијама приликом решавања проблема претпостављена је различитост тачака које могу бити идентичне за неке троуглове – то су парови тачака $(A, H_b), (A, H_c), (B, H_a), (B, H_c), (C, H_a), (C, H_b)$ и у овим решењима је изгубљена једна грана решења (у случају када је троугао правоугли). Оваква врста претпоставке направљена је 30 пута.

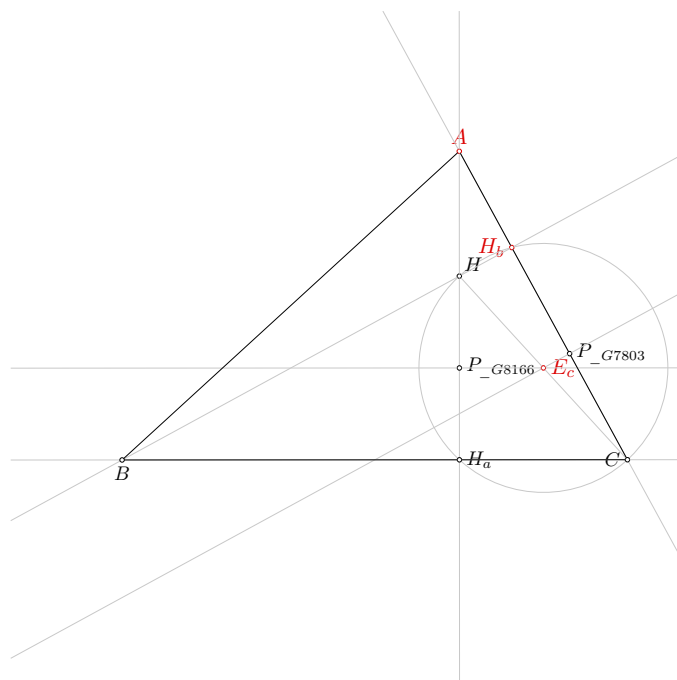
За проблеме из Конелијевог корпуса добијају се још неки парови тачака чија се различитост претпоставља, а које морају бити различите, као што су $(M_a, E_a), (M_b, E_b), (M_c, E_c), (E_a, H_a), (E_b, H_b), (E_c, H_c)$, и парова тачака које могу бити идентичне, као што су $(H, H_a), (H, H_b), (H, H_c), (M_a, H_a), (M_b, H_b), (M_c, H_c), (H_a, H_b), (H_a, H_c), (H_b, H_c)$.

Пример овакве конструкције добија се за проблем 603: A, E_c, H_b у чијем се трећем кораку претпоставља да су тачке H_b и C нужно различите, а у седмом да су тачке H и H_a различите, што не мора бити случај код правоуглог троугла са правим углом у темену C . Генерисана конструкција одговара илустрацији датој на слици 5.1.

Problem 603: Given a point A , a point E_c and a point H_b , construct the triangle ABC .

Construction:

1. Using the point A and the point H_b , construct a line b (rule W02);
% DET: points A and H_b are not the same;
2. Using the point H_b and the point E_c , construct a circle $k(E_c, C)$ (rule W06);
% NDG: points H_b and E_c are not the same;
3. Using the circle $k(E_c, C)$, the line b , the point E_c and the point H_b , construct a point C (rule W05);
% NDG: line b and circle $k(E_c, C)$ intersect;
4. Using the point E_c and the point C , construct a point H (rule W01);



Слика 5.1: Аутоматски генерисана илустрација за проблем 603

5. Using the point A and the point H , construct a line h_a (rule W02);
 % DET: points A and H are not the same;
6. Using the point H_b and the point H , construct a line h_b (rule W02);
 % DET: points H_b and H are not the same;
7. Using the circle $k(E_c, C)$, the line h_a , the point E_c and the point H , construct a point H_a (rule W05);
 % NDG: line h_a and circle $k(E_c, C)$ intersect;
8. Using the point C and the point H_a , construct a line a (rule W02);
 % DET: points C and H_a are not the same;
9. Using the line h_b and the line a , construct a point B (rule W03);
 % NDG: lines h_b and a are not parallel % DET: lines h_b and a are not the same.

Non-degenerate conditions: lines h_b and a are not parallel; line h_a and circle $k(E_c, C)$ intersect; line b and circle $k(E_c, C)$ intersect; points H_b and E_c are not the same.

Determination conditions: lines h_b and a are not the same; points C and H_a are not the same; points H_b and H are not the same; points A and H are not the same; points A and H_b are not the same.

Rules used: [W01, W02, W03, W05, W06]

Definitions and lemmas used: [D3, D30, D5, D6, D8, D9, GD01, GD02, GL03, GL04, L52, L53, L54]

У случају да је троугао ABC правоугли са правим углом код темена C , праве a и h_b би биле идентичне те теме B не бисмо могли да конструишемо као пресечну тачку ове две праве. У овом случају су заправо на улазу дате тачке A и $H_b = E_c = C$ (јер је тачка E_c дефинисана као $\overrightarrow{CE_c}/\overrightarrow{CH} = 1/2$, а обзиром да је $C = H_b = H_a = H$ важи и да је $E_c = C$), те је проблем редундантан, а теме B може да буде произвољна тачка праве $a = h_b$.

Као један од праваца даљег рада планира се да се систем ArgoTriCS прошири тако да укључи обраду гранања у случајевима вишеструких пресечних тачака. Као излаз требало би добити конструкцију у којој је могуће гранање и у којој се у терминима положаја задатих тачака дискутује када која грана важи.

У плану је, такође, разматрање других корпуса проблема конструкције троуглова, као што су они у којима се разматрају и углови троугла, полупречници уписаног и описаног круга, као и полуобим троугла, или чак и полупречници споља описаних кругова и површина троугла, приказани у поглављу 2.2.4. На основу искуства добијеног решавањем проблема из два корпуса разматрана у тези, претпоставља се да не би било потребно додати много новог знања да би се решили проблеми из неког новог корпуса. Такође, у циљу смањења простора претраге, решавач би прво требало да одреди фамилију проблема којој проблем припада и да користи само одговарајућа правила.

Треба, такође, искористити услове одређености примитивних конструкција, тј. размотрити ситуације када објекат који се конструише није јединствено одређен већ укључује бесконачно много решења (као на пример конструкција праве кроз две тачке које су једнаке).

Са идејом примене у образовању, било би интересантно интегрисати овај аутоматизовани процес у неки динамички систем за геометрију, слично као што су аутоматски доказивачи теорема интегрисани у алате као GCLC и GeoGebra. Било би посебно интересантно направити да се генерисана конструкција приказује корак по корак током, на пример, 30 секунди. Ово је тренутно могуће урадити у WinGCLC-у. Овим би се ученицима могла понудити парцијална решења конструктивних проблема која би онда они могли да надограђују. У образовном контексту ово би било од великог значаја.

Аутоматско генерисање формално верификованих решења конструктивних проблема

У плану је наставак рада на аутоматизацији формализације решења добијених системом ArgoTriCS. С обзиром на то да су формални докази генерисани само за један подскуп Верниковог корпуса, планира се наставак рада на овом питању – треба аутоматизовати формализацију преосталих решивих проблема из Верниковог и Конелијевог корпуса и уопштити приступ и на редундантне проблеме и проблеме зависне од положаја.

У формално верификованим решењима конструктивних проблема постоје две празнине. Једна од њих је веза са екстерним алгебарским доказивачима теорема. Тврђења која се доказују током фазе доказивања се доказују алгебарским доказивачима теорема, али не постоји поуздана веза између њих и интерактивног доказивача теорема Isabelle. Стога се тврђења доказана екстерним доказивачима теорема користе као аксиоме. Тренутно, постоје неке ограничене формализације алгебарских доказивача за геометрију коришћењем интерактивних доказивача теорема [45, 48], али не и за Isabelle. За теореме доказане доказивачем ArgoCLP добијају се формални докази у Isabelle-у. Друга празнина постоји између генерисаних доказа и типичних геометријских аксиома (на пример, Хилбертових аксиома или аксиома Тарског). У добијеним доказима користе се геометријске леме високог нивоа као аксиоме. Планира се да се све коришћене леме и конструкцијске примитиве изведу из одговарајућег скупа аксиома, као и елементарних конструкцијских корака за лењир и шестар, и то је веома сложен задатак. Недавно је на овом пољу направљен помак, тако што су неке леме високог нивоа формално доказане у Coq-у [14, 13]. Ово су места на којима је могуће унапредити везу система ArgoTriCS са интерактивним доказивачем теорема Isabelle у циљу потпуне формализације решења конструктивних проблема.

5.1 Summary of Chapter Conclusions and Future Work

The system ArgoTriCS is capable of solving few hundreds of construction problems, which is more than it is considered within any similar system. This is the only system that deals with the problem of proving correctness of generated constructions. Also, for the first time, for some of construction problems formal (machine verifiable) proofs are generated. Still, there are many possible directions for the future work.

We plan to apply ArgoCLP to fragments of different theories and to, in that way,

obtain formal and readable proofs for a range of statements. One of the goals would be to automatically generate mathematical textbooks whose correctness would be formally verified.

Considering the proof simplification module, it would be interesting to use information on relevant and irrelevant facts not only in the context of obtaining shorter proofs, but also to guide the remaining search process (e.g., in future branches). In this way, proofs would not be simplified only *post-festum*, but also during the proving process. We are planning to apply this simplification procedure to some more expressive fragments of logic.

The system ArgoTriCS assumes that all constructed points are different. Therefore, the obtained constructions are always linear and they have no branchings. The analysis of the problems from Wernick’s list led to the observation that most of the times when points are assumed to be different it holds that points indeed must be different (like it is the case with the points A and H_a). However, there are situations when it is assumed that two points are different, but the points need not be different (like it is the case with the points A and H_b which could be identical in case of the triangle with right angle at vertex A). We plan to extend ArgoTriCS so that it can deal with multiple intersection points.

We also plan to consider other corpora of triangle construction points. As experience suggests, not much knowledge should be added for solving the problems from additional corpus. It should be, also, interesting to integrate this automated process into a dynamic geometric system, like automated provers are integrated in GCLC or in GeoGebra.

Considering formal proofs, we plan to continue working on automation of formalization process, by considering the remaining problems from Wernick’s and Connelly’s corpora. In formally verified solutions which are automatically generated, there are still two gaps:

1. there is no trusted link between Isabelle and external automated theorem provers for geometry;
2. all generated proofs use high-level geometry lemmas as axioms – these lemmas should be derived from the corresponding set of axioms and elementary construction primitives for ruler and compass.

Литература

- [1] B. Aldefeld. Variations of geometries based on a geometric-reasoning method. *Computer-Aided Design*, 20(3):117–126, 1988.
- [2] R.C. Alperin. A mathematical theory of origami constructions and numbers. *New York Journal of Mathematics*, 6:119–133, 2000.
- [3] J. Anglesio and V. Schindler. Solution to problem 10719. *American Mathematical Monthly*, 107:952–954, 2000.
- [4] K. Arkoudas. Simplifying proofs in Fitch-style natural deduction systems. *Journal of Automated Reasoning*, 34(3):239–294, 2005.
- [5] J. Avigad, E. Dean, and J. Mumma. A formal system for Euclid’s Elements. *The Review of Symbolic Logic*, 2(4):700–768, 2009.
- [6] V.J. Baston and F.A. Bostock. On the impossibility of ruler-only constructions. In *Proceedings of the American Mathematical Society*, volume 110, pages 1017–1025, 1990.
- [7] M. Beeson. Constructive geometry. In *Tenth Asian Logic Colloquium*. World Scientific, 2010.
- [8] G. Berzsenyi. Constructing triangles from three given parts. *Quantum*, page 30, 1994.
- [9] M. Bezem and T. Coquand. Automating coherent logic. In *Logic for Programming, Artificial Intelligence, and Reasoning*, volume 3835 of *LNCS*. Springer, 2005.
- [10] M. Bezem and D. Hendriks. On the mechanization of the proof of Hessenberg’s theorem in coherent logic. *Journal of Automated Reasoning*, 40(1), 2008.
- [11] K. Borsuk and W. Szmielew. *Foundations of Geometry*. North-Holland Publishing Company, Amsterdam, 1960.

- [12] W. Bouma, I. Fudos, C. Hoffmann, J. Cai, and R. Paige. A geometric constraint solver. *Computer-Aided Design*, 27(6):487–501, 1995.
- [13] P. Boutry, J. Narboux, P. Schreck, and G. Braun. Using small scale automation to improve both accessibility and readability of formal proofs in geometry. In Francisco Botana and Pedro Quaresma, editors, *Automated Deduction in Geometry 2014*, Proceedings of ADG 2014, pages 1–19, 2014.
- [14] G. Braun and J. Narboux. From Tarski to Hilbert. In *Automated Deduction in Geometry*, volume 7993 of *LNCS*, pages 89–109. Springer, 2013.
- [15] B. Buchberger. *An Algorithm for finding a basis for the residue class ring of a zero-dimensional polynomial ideal*. PhD thesis, University of Innsbruck, 1965.
- [16] M. Buthion. Un programme qui résout formellement des problèmes de constructions géométriques. *RAIRO Informatique*, 13(1):73–106, 1979.
- [17] G. Chen. Les constructions à la règle et au compas par une méthode algébrique. Technical Report Rapport de DEA, Université Louis Pasteur, 1992.
- [18] G. Chen. Les constructions géométriques à la règle et au compas par une méthode algébrique. Master’s thesis, University of Strasbourg, 1992.
- [19] L. Chiarabini. *Program Development by Proof Transformation*. PhD thesis, LMU München: Faculty of Mathematics, Computer Science and Statistics, 2009.
- [20] S.-C. Chou, X.-S. Gao, and J.-Z. Zhang. Automated generation of construction steps for geometric constraint problems. In Robert Veroff and Gail W. Pieper, editors, *Automated Reasoning and Its Applications*, pages 49–69. MIT Press, 1997.
- [21] S.C. Chou and X.S. Gao. Automated reasoning in geometry. *Handbook of Automated Reasoning*, 5(1), 2001.
- [22] S.C. Chou, X.S. Gao, and J.Z. Zhang. Automated production of traditional proofs for constructive geometry theorems. In *IEEE Symposium on Logic in Computer Science LICS*. IEEE Computer Society Press, 1993.
- [23] S.C. Chou, X.S. Gao, and J.Z. Zhang. Automated generation of readable proofs with geometric invariants, ii. theorem proving with full-angles. *Journal of Automated Reasoning*, 17, 1996.

- [24] B. Coburn and D. Miller. Two comments on lemmon’s beginning logic. *Notre Dame Journal of Formal Logic*, 18(4), 1977.
- [25] H. Connelly. An extension of triangle constructions from located points. *Forum Geometricorum*, 9:109–112, 2009.
- [26] H. Connelly, N. Dergiades, and J.P. Ehrmann. Construction of triangle from a vertex and the feet of two angle bisectors. *Forum Geometricorum*, 5:53–56, 2007.
- [27] H. Connelly and B. Randrianantoanina. An angle bisector parallel applied to triangle construction. *Forum Geometricorum*, 9:161–163, 2009.
- [28] E. Danneels. A simple construction of a triangle from its centroid, incenter, and a vertex. *Forum Geometricorum*, 5:53–56, 2005.
- [29] P.J. Davis. The rise, fall, and possible transfiguration of triangle geometry: A mini-history. *The American Mathematical Monthly*, 102(3):204–214, 1995.
- [30] H. De Nivelles and J. Meng. Geometric resolution: A proof procedure based on finite model search. In *IJCAR 2006*, volume 4130 of *LNCS*. Springer, 2006.
- [31] C. Dehlinger, J. F. Dufourd, and P. Schreck. Higher-order intuitionistic formalization and proofs in hilbert’s elementary geometry. In *Automated Deduction in Geometry*, Lectures Notes in Artificial Intelligence, pages 303–323. Springer, 2000.
- [32] The Coq development team. *The Coq proof assistant reference manual, version 8.2*. TypiCal Project, 2009.
- [33] M. Djorić and P. Janičić. Constructions, instructions, interactions. *Teaching Mathematics and its Applications*, 23(2):69–88, 2004.
- [34] J.-F. Dufourd, P. Mathis, and P. Schreck. Geometric construction by assembling solved subfigures. *Artificial Intelligence Journal*, 99(1):73–119, 1998.
- [35] J. Duprat. Une axiomatique de la géométrie plane en Coq. In *Actes des JFLA*. INRIA, 2008.
- [36] C. Essert-Villard, P. Schreck, and J. F. Dufourd. Sketch-based pruning of a solution space within a formal geometric constraint solver. *Artificial Intelligence*, 124:139–159, 2000.

-
- [37] J. Fisher and M. Bezem. Skolem machines and geometric logic. In *4th International Colloquium on Theoretical Aspects of Computing — ICTAC 2007*, volume 4711 of *LNCS*. Springer, 2007.
- [38] V.B. Fursenko. Lexicographic account of triangle construction problems (part i). *Mathematics in schools*, 5:4–30, 1937.
- [39] V.B. Fursenko. Lexicographic account of triangle construction problems (part ii). *Mathematics in schools*, 6:21–45, 1937.
- [40] M. Ganesalingam and T. Gowers. A fully automatic problem solver with human-style output. *CoRR*, abs/1309.4501, 2013.
- [41] X.S. Gao and S.C. Chou. Solving geometric constraint systems. i. a global propagation approach. *Computer-Aided Design*, 30(1):47–54, 1998.
- [42] X.S. Gao and S.C. Chou. Solving geometric constraint systems ii. a symbolic approach and decision of rc-constructibility. *Computer-Aided Design*, 30:115–122, 1998.
- [43] H. Gelernter. Realization of a geometry-theorem proving machine. In *Computer and Thought*, pages 134–152. Mac Graw-Hill, 1963.
- [44] M. Grima and G.J. Pace. An embedded geometrical language in Haskell: Construction, visualisation, proof. In *Computer Science Annual Workshop*, 2007.
- [45] B. Grégoire, L. Pottier, and L. Théry. Proof certificates for algebra and their application to automatic geometry theorem proving. In Thomas Sturm and Christoph Zengler, editors, *Automated Deduction in Geometry*, volume 6301 of *Lecture Notes in Computer Science*, pages 42–59. Springer Berlin Heidelberg, 2011.
- [46] F. Guilhot. Formalisation en Coq d’un cours de géométrie pour le lycée. *Journées Francophones des Langages Applicatifs*, 2004.
- [47] S. Gulwani, V. Korthikanti, and A. Tiwari. Synthesizing geometry constructions. In *Programming Language Design and Implementation*, pages 50–61. ACM, 2011.
- [48] J.-D. Génevaux, J. Narboux, and P. Schreck. Formalization of Wu’s simple method in Coq. In Jean-Pierre Jouannaud and Zhong Shao, editors, *Certified*

- Programs and Proofs*, volume 7086 of *Lecture Notes in Computer Science*, pages 71–86. Springer Berlin Heidelberg, 2011.
- [49] J. Harrison. Hol light: A tutorial introduction. In *Formal Methods in Computer-Aided Design*, LNCS 1166. Springer, 1996.
- [50] T. Heath. *The Thirteen Books of Euclid's Elements*. Dover Publications, New York, 1956.
- [51] D. Hilbert. *Grundlagen der Geometrie*. Leipzig, 1899.
- [52] A. Hungerbühler. A short elementary proof of the Mohr–Mascheroni theorem. *The American Mathematical Monthly*, 101(8):784, 1994.
- [53] P. Janičić. Jedan metod za automatsko dokazivanje teorema geometrije, magistarska teza. Master's thesis, Matematički fakultet, 1996.
- [54] P. Janičić. Geometry constructions language. *Journal of Automated Reasoning*, 44(1-2):3–24, 2010.
- [55] P. Janičić and S. Kordić. Euclid - the geometry theorem prover. *FILOMAT*, 9(3), 1995.
- [56] G. Kahn. Constructive geometry according to Jan von Plato, 1995. Coq contribution, V5.10.
- [57] D. Kapur. Using Gröbner bases to reason about geometry problems. *Journal of Symbolic Computation*, 2(4), 1986.
- [58] M. Koshimura and R. Hasegawa. Proof simplification for model generation and its applications. In *LPAR'00 Proceedings of the 7th international conference on Logic for programming and automated reasoning*, volume 1955, pages 96–113. Springer-Verlag Berlin, Heidelberg, 2000.
- [59] R. S. Latham and A. E. Middleditch. Connectivity analysis: a tool for processing geometric constraints. *Computer-Aided Design*, 28(11):917–928, 1996.
- [60] H. Lebesgue. *Leçons sur les constructions géométriques*. Gauthier-Villars, Paris, 1950.
- [61] L. Lopes. *Manuel de construction de triangles*. QED Texte, 1996.
- [62] Z. Lučić. *Euklidska i hiperbolička geometrija*. Grafitti i Matematički fakultet, Beograd, 1994.

- [63] Z. Lučić. *Ogledi iz istorije antičke geometrije*. Službeni glasnik, 2009.
- [64] N. Magaud, J. Narboux, and P. Schreck. Formalizing projective plane geometry in Coq. In *Automated Deduction in Geometry*, 2008.
- [65] N. Magaud, J. Narboux, and P. Schreck. Formalizing Desargues’ theorem in Coq using ranks. In *ACM symposium on Applied Computing*. ACM, 2009.
- [66] F. Marić, I. Petrović, D. Petrović, and P. Janičić. Formalization and implementation of algebraic methods in geometry. In *THedu*, volume 79, pages 63–81. EPTCS, 2011.
- [67] V. Marinković. Proof simplification in the framework of coherent logic. *Computing and Informatics*, page to appear, 2015.
- [68] V. Marinković and P. Janičić. Towards understanding triangle construction problems. In J. Jeuring et al., editor, *Intelligent Computer Mathematics - CICM 2012*, volume 7362 of *Lecture Notes in Computer Science*. Springer, 2012.
- [69] V. Marinković, P. Janičić, and P. Schreck. Solving geometric construction problems supported by theorem proving. In F. Botana and P. Quaresma, editors, *Automated Deduction in Geometry – ADG 2014*, 2014.
- [70] G. Martin. *Geometric Constructions*. Springer, 1998.
- [71] W. McCune and L. Wos. Otter: The cade-13 competition incarnations. *Journal of Automated Reasoning*, 18(2):211–220, 1997.
- [72] L. Meikle and J. Fleuriot. Formalizing Hilbert’s Grundlagen in Isabelle/Isar. In *Theorem Proving in Higher Order Logics*. LNCS 2758, Springer, 2003.
- [73] J.K. Merikoski and T. Tossavainen. Two approaches to geometrography. *Journal for Geometry and Graphics*, 14(1):15–28, 2010.
- [74] L.F. Meyers. Update on William Wernick’s “triangle constructions with three located points”. *Mathematics Magazine*, 69(1):46–49, 1996.
- [75] A. Montesdeoca. Resoulcion de triangulos. <http://amontes.webs.ull.es/pdf/trresolu.pdf>.
- [76] J. Narboux. Mechanical theorem proving in Tarski’s geometry. In *Automated Deduction in Geometry*. LNCS 4869, Springer, 2007.

-
- [77] M. Nikolić and P. Janičić. CDCL-based abstract state transition system for coherent logic. In *Intelligent Computer Mathematics*, volume 7362 of *Lecture Notes in Computer Science*, pages 264–279. Springer, 2012.
- [78] T. Nipkow, L.C. Paulson, and M. Wenzel. Isabelle/HOL - a proof assistant for higher-order logic. *LNCS Springer*, 2283, 2002.
- [79] J. Owen. Algebraic solution for geometry from dimensional constraints. In *Proceedings of the 1th ACM Symposium of Solid Modeling and CAD/CAM Applications*, pages 397–407. ACM Press, 1991.
- [80] V. Pambuccian. Axiomatizing geometric constructions. *Journal of Applied Logic*, 6(1):24–46, 2008.
- [81] F.J. Pelletier and A.P. Hazen. Natural deduction. *Handbook of the History of Logic*, 11:341–414, 2012.
- [82] F. Piroi. *Tools for Using Automated Provers in Mathematical Theory Exploration*. PhD thesis, Johannes Kepler University, RISC, Linz, Austria, 2004.
- [83] A. Polonsky. *Proofs, Types, and Lambda Calculus*. PhD thesis, University of Bergen, 2011.
- [84] D. Prawitz. Ideas and results in proof theory. In *Proceedings of the 2. Scandinavian Logic Symposium*, pages 237–309, 1971.
- [85] A. Quaife. Automated development of Tarski’s geometry. *Journal of Automated Reasoning*, 5(1), 1989.
- [86] S. Rahul and G. Necula. Proof optimization using lemma extraction. Technical report, University of California, Berkeley, 2001.
- [87] A. Riazanov and A. Voronkov. The design and implementation of Vampire. *AI Communications*, 15:91–110, 2002.
- [88] J.A. Robinson. A machine oriented logic based on the resolution principle. *Journal of ACM*, 12:23–41, 1965.
- [89] P. Schreck. *Automatisation des constructions géométriques à la règle et au compas*. PhD thesis, Université de Strasbourg, 1993.
- [90] P. Schreck. Implantation d’un système à base de connaissances pour les constructions géométriques. *Revue d’Intelligence Artificielle*, 8(3):223–247, 1994.

- [91] P. Schreck. Robustness in cad geometric construction. In *Proceedings of the fifth International Conference on Information Visualisation, IV2001 (London)*, pages 111–116. IEEE, 2001.
- [92] P. Schreck and P. Mathis. Rc-constructibility of problems in Wernick’s list. In *International Workshop on Automated Deduction in Geometry*, pages 85–104, 2014.
- [93] S. Schulz. E - a brainiac theorem prover. *AI Communications*, 15:111–126, 2002.
- [94] W. Schwabhäuser, W. Szmielew, and A. Tarski. *Metamathematische Methoden in der Geometrie*. Springer, 1983.
- [95] E. Specht. Wernicks liste. <http://hydra.nat.uni-magdeburg.de/wernick/>.
- [96] I. Stewart. *Galois Theory*. Chapman and Hall Ltd., 1973.
- [97] S. Stojanović, J. Narboux, M. Bezem, and P. Janičić. A vernacular for coherent logic. In *Intelligent Computer Mathematics, CICM*, pages 388–403, 2014.
- [98] S. Stojanović, V. Pavlović, and P. Janičić. A coherent logic based geometry theorem prover capable of producing formal and readable proofs. In *Automated Deduction in Geometry*, volume 6877 of *LNAI*, pages 201–220. Springer, 2011.
- [99] G. Sutcliffe. The TPTP problem library and associated infrastructure: The fof and cnf parts, v3.5.0. *Journal of Automated Reasoning*, 43(4):337–362, 2009.
- [100] R.E. Tarjan. Efficiency of a good but not linear set union algorithm. *Journal of ACM*, 22(2), 1975.
- [101] A. Tarski. *What is elementary geometry? The Axiomatic Method, with Special Reference to Geometry and Physics*. North-Holland, 1959.
- [102] R. Thiele and L. Wos. Hilbert’s twenty-fourth problem. *Journal of Automated Reasoning*, 29:67–89, 2002.
- [103] M. Trost. Konstrukcije trikotnikov s programom RIŠ. Master’s thesis, Univerza v Mariboru, Fakulteta za naravoslovje in matematiko, 2009.
- [104] A. Trybulec. Mizar. In *The Seventeen Provers of the World*, LNCS 3600. Springer, 2006.

- [105] A. V. Ustinov. On the construction of a triangle from the feet of its angle bisectors. *Forum Geometricorum*, 9:279–280, 2009.
- [106] C. Weidenbach, D. Dimova, A. Fietzke, R. Kumar, M. Suda, and P. Wischniewski. Spass version 3.5. In *Automated Deduction - CADE-22 Proceedings*, volume 5663 of *LNCS*, pages 140–145. Springer, 2009.
- [107] M. Wenzel. Isar - a generic interpretative approach to readable formal proof documents. In Bertot Y., Dowek G., Hirschowitz A., Paulin C., and Thierry L., editors, *Theorem Proving in Higher Order Logics*, volume 1690 of *LNCS*, pages 167–184. Springer, 1999.
- [108] W. Wernick. Triangle constructions with three located points. *Mathematics Magazine*, 55(4):227–230, 1982.
- [109] W.T. Wu. On the decision problem and the mechanization of theorem proving in elementary geometry. *Scientia Sinica*, 21, 1978.

Додатак А

Хилбертов аксиоматски систем

А.1 Дефиниције

Дефиниција D1: $\forall A : point \ \forall B : point \ \forall C : point \ \forall p : line \ (inc(A, p) \wedge inc(B, p) \wedge inc(C, p) \Rightarrow col(A, B, C))$

Дефиниција D2: $\forall A : point \ \forall B : point \ \forall C : point \ (col(A, B, C) \Rightarrow \exists p : line \ (inc(A, p) \wedge inc(B, p) \wedge inc(C, p)))$

Дефиниција D3: $\forall A : point \ \forall B : point \ \forall C : point \ \forall D : point \ \forall \alpha : plane \ (inc(A, \alpha) \wedge inc(B, \alpha) \wedge inc(C, \alpha) \wedge inc(D, \alpha) \Rightarrow comp(A, B, C, D))$

Дефиниција D4: $\forall A : point \ \forall B : point \ \forall C : point \ \forall D : point \ (comp(A, B, C, D) \Rightarrow \exists \alpha : plane \ (inc(A, \alpha) \wedge inc(B, \alpha) \wedge inc(C, \alpha) \wedge inc(D, \alpha)))$

Дефиниција D5: $\forall p : line \ \forall q : line \ \forall A : point \ (p \neq q \wedge inc(A, p) \wedge inc(A, q) \Rightarrow int(p, q))$

Дефиниција D6: $\forall p : line \ \forall q : line \ (int(p, q) \Rightarrow \exists A : point \ (inc(A, p) \wedge inc(A, q) \wedge p \neq q))$

Дефиниција D7: $\forall \alpha : plane \ \forall \beta : plane \ \forall A : point \ (\alpha \neq \beta \wedge inc(A, \alpha) \wedge inc(A, \beta) \Rightarrow int(\alpha, \beta))$

Дефиниција D8: $\forall \alpha : plane \ \forall \beta : plane \ (int(\alpha, \beta) \Rightarrow \exists A : point \ (inc(A, \alpha) \wedge inc(A, \beta) \wedge \alpha \neq \beta))$

Дефиниција D9: $\forall A : point \ \forall p : line \ \forall \alpha : plane \ (inc(A, p) \wedge inc(A, \alpha) \wedge \neg inc(p, \alpha) \Rightarrow int(p, \alpha))$

Дефиниција D10: $\forall p : line \ \forall \alpha : plane \ (int(p, \alpha) \Rightarrow \exists A : point \ (inc(A, p) \wedge inc(A, \alpha) \wedge \neg inc(p, \alpha)))$

Дефиниција D11: $\forall A : point \ \forall B : point \ \forall C : point \ \forall X : point \ \forall Y : point \ \forall Z : point \ \forall U : point \ \forall V : point \ (A \neq B \wedge A \neq C \wedge X \neq Y \wedge X \neq Z \wedge col(X, Y, U) \wedge \neg bet(U, X, Y) \wedge col(X, Z, V) \wedge$

$\neg bet(V, X, Z) \wedge cong(A, B, X, U) \wedge cong(A, C, X, V) \wedge cong(B, C, U, V) \Rightarrow cong_angle(A, B, C, X, Y, Z))$

Дефиниција D12: $\forall A : point \ \forall B : point \ \forall C : point \ \forall X : point \ \forall Y : point \ \forall Z : point (cong_angle(A, B, C, X, Y, Z) \Rightarrow \exists U : point \ \exists V : point (A \neq B \wedge A \neq C \wedge X \neq Y \wedge X \neq Z \wedge col(X, Y, U) \wedge \neg bet(U, X, Y) \wedge col(X, Z, V) \wedge \neg bet(V, X, Z) \wedge cong(A, B, X, U) \wedge cong(A, C, X, V) \wedge cong(B, C, U, V)))$

A.2 Аксиоме

Аксиома I1: $\forall A : point \ \forall B : point (A \neq B \Rightarrow \exists p : line (inc(A, p) \wedge inc(B, p)))$

Аксиома I2: $\forall A : point \ \forall B : point \ \forall p : line \ \forall q : line (inc(A, p) \wedge inc(A, q) \wedge inc(B, p) \wedge inc(B, q) \wedge A \neq B \Rightarrow p = q)$

Аксиома I3a: $\forall p : line \ \exists A : point \ \exists B : point (inc(A, p) \wedge inc(B, p) \wedge A \neq B)$

Аксиома I3b: $\exists A : point \ \exists B : point \ \exists C : point (\neg col(A, B, C))$

Аксиома I4a: $\forall A : point \ \forall B : point \ \forall C : point (A \neq B \wedge A \neq C \wedge B \neq C \wedge \neg col(A, B, C) \Rightarrow \exists \alpha : plane (inc(A, \alpha) \wedge inc(B, \alpha) \wedge inc(C, \alpha)))$

Аксиома I4b: $\forall \alpha : plane \ \exists A : point (inc(A, \alpha))$

Аксиома I5: $\forall \alpha : plane \ \forall \beta : plane \ \forall A : point \ \forall B : point \ \forall C : point (A \neq B \wedge A \neq C \wedge B \neq C \wedge inc(A, \alpha) \wedge inc(A, \beta) \wedge inc(B, \alpha) \wedge inc(B, \beta) \wedge inc(C, \alpha) \wedge inc(C, \beta) \wedge \neg col(A, B, C) \Rightarrow \alpha = \beta)$

Аксиома I6: $\forall \alpha : plane \ \forall p : line \ \forall A : point \ \forall B : point (inc(A, \alpha) \wedge inc(A, p) \wedge inc(B, \alpha) \wedge inc(B, p) \wedge A \neq B \Rightarrow inc(p, \alpha))$

Аксиома I7: $\forall \alpha : plane \ \forall \beta : plane \ \forall A : point (\alpha \neq \beta \wedge inc(A, \alpha) \wedge inc(A, \beta) \Rightarrow \exists B : point (A \neq B \wedge inc(B, \alpha) \wedge inc(B, \beta)))$

Аксиома I8: $\exists A : point \ \exists B : point \ \exists C : point \ \exists D : point (\neg comp(A, B, C, D))$

Аксиома II1: $\forall A : point \ \forall B : point \ \forall C : point (bet(A, B, C) \Rightarrow col(A, B, C) \wedge A \neq B \wedge A \neq C \wedge B \neq C \wedge bet(C, B, A))$

Аксиома II2: $\forall A : point \ \forall B : point (A \neq B \Rightarrow \exists C : point (bet(A, B, C)))$

Аксиома II3: $\forall A : point \ \forall B : point \ \forall C : point (A \neq B \wedge A \neq C \wedge B \neq C \wedge col(A, B, C) \Rightarrow bet(A, B, C) \vee bet(B, C, A) \vee bet(C, A, B))$

Аксиома II4: $\forall A : point \ \forall B : point \ \forall C : point \ \forall D : point \ \forall \alpha : plane \ \forall p : line (A \neq B \wedge A \neq C \wedge B \neq C \wedge \neg col(A, B, C) \wedge inc(A, \alpha) \wedge inc(B, \alpha) \wedge inc(C, \alpha) \wedge inc(p, \alpha) \wedge \neg inc(A, p) \wedge inc(D, p) \wedge bet(B, D, C) \Rightarrow (\exists E : point ((inc(E, p) \wedge bet(C, E, A))) \vee (\exists F : point (inc(F, p) \wedge bet(A, F, B))))$

Аксиома III1: $\forall A : \text{point} \ \forall B : \text{point} \ \forall C : \text{point} \ \forall D : \text{point} \ (A \neq B \ \wedge \ C \neq D \Rightarrow \exists E : \text{point} \ (\text{bet}(B, A, E) \ \wedge \ \text{cong}(A, E, C, D)))$

Аксиома III2: $\forall A : \text{point} \ \forall B : \text{point} \ \forall C : \text{point} \ \forall D : \text{point} \ \forall E : \text{point} \ \forall F : \text{point} \ (\text{cong}(A, B, C, D) \ \wedge \ \text{cong}(A, B, E, F) \Rightarrow \text{cong}(C, D, E, F))$

Аксиома III3: $\forall A : \text{point} \ \forall B : \text{point} \ \forall C : \text{point} \ \forall A_1 : \text{point} \ \forall B_1 : \text{point} \ \forall C_1 : \text{point} \ (\text{bet}(A, B, C) \ \wedge \ \text{bet}(A_1, B_1, C_1) \ \wedge \ \text{cong}(A, B, A_1, B_1) \ \wedge \ \text{cong}(B, C, B_1, C_1) \Rightarrow \text{cong}(A, C, A_1, C_1))$

Аксиома III4_a: $\forall A : \text{point} \ \forall B : \text{point} \ \forall C : \text{point} \ (A \neq B \ \wedge \ A \neq C \Rightarrow \text{cong_angle}(A, B, C, A, B, C))$

Аксиома III4_b1: $\forall A : \text{point} \ \forall B : \text{point} \ \forall C : \text{point} \ \forall P : \text{point} \ \forall Q : \text{point} \ \forall M : \text{point} \ (A \neq B \ \wedge \ A \neq C \ \wedge \ P \neq Q \ \wedge \ \neg \text{col}(P, Q, M) \Rightarrow \exists l : \text{line} \ \exists R_1 : \text{point} \ \exists R_2 : \text{point} \ \exists N_1 : \text{point} \ \exists N_2 : \text{point} \ (\text{cong_angle}(A, B, C, P, Q, R_1) \ \wedge \ \text{cong_angle}(A, B, C, P, Q, R_2) \ \wedge \ \text{inc}(P, l) \ \wedge \ \text{inc}(Q, l) \ \wedge \ \text{bet}(M, N_1, R_1) \ \wedge \ \text{inc}(N_1, l) \ \wedge \ \text{bet}(R_2, N_2, R_1) \ \wedge \ \text{inc}(N_2, l)))$

Аксиома III4_b2: $\forall A : \text{point} \ \forall B : \text{point} \ \forall C : \text{point} \ \forall P : \text{point} \ \forall M : \text{point} \ \forall R_1 : \text{point} \ \forall R_2 : \text{point} \ \forall N_1 : \text{point} \ \forall N_2 : \text{point} \ \forall l : \text{line} \ (A \neq B \ \wedge \ A \neq C \ \wedge \ P \neq Q \ \wedge \ \neg \text{col}(P, Q, M) \ \wedge \ \text{cong_angle}(A, B, C, P, Q, R_1) \ \wedge \ \text{cong_angle}(A, B, C, P, Q, R_2) \ \wedge \ \text{inc}(P, l) \ \wedge \ \text{inc}(Q, l) \ \wedge \ \text{bet}(M, N_1, R_1) \ \wedge \ \text{inc}(N_1, l) \ \wedge \ \text{bet}(M, N_2, R_2) \ \wedge \ \text{inc}(N_2, l) \Rightarrow \text{bet}(P, R_1, R_2) \ \vee \ \text{bet}(P, R_2, R_1) \ \vee \ R_1 = R_2)$

Аксиома III5: $\forall A : \text{point} \ \forall B : \text{point} \ \forall C : \text{point} \ \forall X : \text{point} \ \forall Y : \text{point} \ \forall Z : \text{point} \ (\text{cong}(A, B, X, Y) \ \wedge \ \text{cong}(A, C, X, Z) \ \wedge \ \text{cong_angle}(A, B, C, X, Y, Z) \Rightarrow \text{cong_angle}(B, A, C, Y, X, Z))$

Аксиома IV_a: $\forall A : \text{point} \ \forall p : \text{line} \ \forall \alpha : \text{plane} \ (\neg \text{inc}(A, p) \ \wedge \ \text{inc}(A, \alpha) \ \wedge \ \text{inc}(p, \alpha) \Rightarrow \exists q : \text{line} \ (\text{inc}(A, q) \ \wedge \ \text{inc}(q, \alpha) \ \wedge \ \neg \text{int}(q, p)))$

Аксиома IV_b: $\forall A : \text{point} \ \forall p : \text{line} \ \forall \alpha : \text{plane} \ \forall q : \text{line} \ \forall r : \text{line} \ (\neg \text{inc}(A, p) \ \wedge \ \text{inc}(A, \alpha) \ \wedge \ \text{inc}(p, \alpha) \ \wedge \ \text{inc}(A, q) \ \wedge \ \text{inc}(q, \alpha) \ \wedge \ \neg \text{int}(p, q) \ \wedge \ \text{inc}(A, r) \ \wedge \ \text{inc}(r, \alpha) \ \wedge \ \neg \text{int}(p, r) \Rightarrow q = r)$

Додатак В

Списак лема и конструкцијских примитива

В.1 Леме

ArgoTriCS за решавање проблема из Верникове листе користи следеће инстанциране леме:

L1 тачка O припада медијатриси дужи AB ;

L2 тачка I припада дужи CT_c ;

L3 тачка H припада дужи CH_c ;

L4 тачка G припада дужи CM_c ;

L5–L7 тачке A , B и C припадају редом бисектрисама углова код темена A , B и C ¹;

L8–L10 тачка T_a је на једнакој удаљености од правих AB и AC ; тачка T_b је на једнакој удаљености од правих AB и BC ; тачка T_c је на једнакој удаљености од правих AC и BC ;

L11–L12 тачке A и B припадају кругу $k(O, C)$;

L13–L14 тачке P_b и P_c припадају кругу $k(I, P_a)$;

L25–L27 тачке N_a , N_b и N_c припадају редом правама OM_a , OM_b и OM_c ;

L28–L30 праве AB , BC и CA додирују круг $k(I, P_a)$;

¹Бисектриса угла се у систему ArgoTriCS дефинише као права која дели угао на два једнака дела.

- L31–L36** тачке B и I припадају кругу $k(N_a, C)$; тачке C и I припадају кругу $k(N_b, A)$; тачке A и I припадају кругу $k(N_c, B)$;
- L37–L45** тачке C , H_b и H_c припадају кругу $k(M_a, B)$; тачке A , H_a и H_c припадају кругу $k(M_b, C)$; тачке B , H_a и H_b припадају кругу $k(M_c, A)$;
- L55–L57** $\overrightarrow{AG}/\overrightarrow{AM_a} = 2/3$, $\overrightarrow{BG}/\overrightarrow{BM_b} = 2/3$, $\overrightarrow{CG}/\overrightarrow{CM_c} = 2/3$;
- L58** $\overrightarrow{HG}/\overrightarrow{HO} = 2/3$;
- L59–L61** стране троугла су управне на круг $k(I, P_a)$;
- L62–L64** $\overrightarrow{P_aM_a}/\overrightarrow{P_aP'_a} = 1/2$, $\overrightarrow{P_bM_b}/\overrightarrow{P_bP'_b} = 1/2$, $\overrightarrow{P_cM_c}/\overrightarrow{P_cP'_c} = 1/2$;
- L71–L73** тачка A припада кругу над пречником $T_aT'_a$; тачка B припада кругу над пречником $T_bT'_b$; тачка C припада кругу над пречником $T_cT'_c$;
- L74–L76** тачка P_a припада кругу над пречником IM_a ; тачка P_b припада кругу над пречником IM_b ; тачка P_c припада кругу над пречником IM_c ;
- L77–L79** $\mathcal{H}(B, C; T_a, T'_a)$, $\mathcal{H}(A, C; T_b, T'_b)$, $\mathcal{H}(A, B; T_c, T'_c)$, при чему је са $\mathcal{H}(X, Y, Z, W)$ означено да је пар тачака X, Y хармонијски спрегнут са паром тачака Z, W ;
- L80–L85** тачка A припада скупу тачака из кога се дуж T_aH_b види под углом $\angle BAC/2$, као и скупу тачака из кога се дуж T_aH_c види под углом $\angle BAC/2$; тачка B припада скупу тачака из кога се дуж T_bH_c види под углом $\angle CBA/2$, као и скупу тачака из кога се дуж T_bH_a види под углом $\angle CBA/2$; тачка C припада скупу тачака из кога се дуж T_cH_a види под углом $\angle ACB/2$, као и скупу тачака из кога се дуж T_cH_b види под углом $\angle ACB/2$;
- L86–L88** тачка I припада скупу тачака из којих се дуж T_bT_c види под углом $\angle T_bIT_c$, скупу тачака из којих се дуж T_bT_a види под углом $\angle T_aIT_b$, као и скупу тачака из којих се дуж T_aT_c види под углом $\angle T_cIT_a$;
- L89–L91** тачка A припада скупу тачака из којих се дуж T_bT_c види под углом $\angle BAC$, скупу тачака из којих се дуж T_cT_a види под углом $\angle CBA$, као и скупу тачака из којих се дуж T_aT_b види под углом $\angle ACB$;
- L92–L97** тачка A припада скупу тачака из којих се дуж T_bI види под углом $\angle BAC/2$, као и скупу тачака из којих се дуж T_cI види под углом $\angle BAC/2$; тачка B припада скупу тачака из којих се дуж T_aI види под углом $\angle CBA/2$, као и скупу тачака из којих се дуж T_cI види под углом $\angle CBA/2$;

тачка C припада скупу тачака из којих се дуж T_bI види под углом $\angle ACB/2$, као и скупу тачака из којих се дуж T_aI види под углом $\angle ACB/2$;

L98–L103 $\angle BAI = \angle BAC/2$, $\angle IAC = \angle BAC/2$, $\angle CBI = \angle CBA/2$, $\angle IBA = \angle CBA/2$, $\angle ACI = \angle ACB/2$, $\angle ICB = \angle ACB/2$;

L104–L106 $\angle HAI = \angle IAO$, $\angle HBI = \angle IBO$, $\angle HCI = \angle ICO$;

L107–L109 $\angle H_bH_aC = \angle BAC$, $\angle H_aH_bC = \angle CBA$, $\angle H_bH_cA = \angle ACB$, $\angle H_cH_bA = \angle CBA$, $\angle H_cH_aB = \angle CBA$, $\angle H_aH_cB = \angle ACB$;

L110–L112 $\angle T_cIT_b = \angle BAC/2 + \pi/2$, $\angle T_bIT_a = \angle ACB/2 + \pi/2$, $\angle T_aIT_c = \angle CBA/2 + \pi/2$;

L113–L115 праве AH , BH и CH су бисектрисе унутрашњих углова троугла $H_aH_bH_c$.

Такође, потребне су и следеће опште леме:

GL01 Центар круга припада медијатриси произвољне тетиве тог круга;

GL02 Ако важи $\overrightarrow{XY}/\overrightarrow{XZ} = r$, тачка Z припада правој p , а тачка Y јој не припада, онда тачка X припада правој која је слика праве p при хомотетији са центром у тачки Y са коефицијентом $r/(r-1)$;

GL03 Ако важи $\overrightarrow{XY}/\overrightarrow{ZW} = r$ онда важи и $\overrightarrow{ZW}/\overrightarrow{XY} = 1/r$;

GL04 Ако важи $\overrightarrow{XY}/\overrightarrow{XW} = r$ онда важи и $\overrightarrow{WY}/\overrightarrow{WX} = 1-r$;

GL05 Ако важи $\mathcal{H}(X, Y; Z, W)$ онда важи и $\mathcal{H}(Y, X; W, Z)$;

GL06 Ако важи $\mathcal{H}(X, Y; Z, W)$ онда важи и $\mathcal{H}(Z, W; X, Y)$;

GL07 Ако важи $\mathcal{H}(X, Y; Z, W)$ онда важи и $\mathcal{H}(W, Z; Y, X)$;

GL08 Ако важи $\overrightarrow{XY}/\overrightarrow{XZ} = r$, $\overrightarrow{XU}/\overrightarrow{XV} = r$, при чему су тачке Y и U различите, тачке Y и U припадају правој p , а тачке Z и V правој q , онда су праве p и q паралелне;

GL09 Ако важи $\overrightarrow{XY}/\overrightarrow{XZ} = r$ и тачке X и Z припадају правој p , онда и тачка Y припада правој p ;

GL10–GL11 Ако је тачка X на једнакој удаљености од правих p и q онда су праве p и q тангенте на круг са центром у тачки X чији је полупречник једнак овој удаљености;

- GL12** Ако важи $\alpha = \beta/2^{L_1}$ и $\gamma = K_1 \cdot \beta/2^{K_2} + K_3 \cdot \pi/2^{K_4}$ онда важи и $\gamma = K_1 \cdot \alpha/2^{K_2-L_1} + K_3 \cdot \pi/2^{K_4}$;
- GL13** Ако важи $\alpha = \beta/2^{L_1}$ и $\beta = K_1 \cdot \gamma/2^{K_2} + K_3 \cdot \pi/2^{K_4}$ онда важи и $\alpha = K_1 \cdot \gamma/2^{K_2+L_1} + K_3 \cdot \pi/2^{K_4+L_1}$;
- GL14** Ако важи $\alpha = \beta/2^{L_1}$ и $\gamma = K_1 \cdot \alpha/2^{K_2} + K_3 \cdot \pi/2^{K_4}$ онда важи и $\gamma = K_1 \cdot \beta/2^{K_2+L_1} + K_3 \cdot \pi/2^{K_4}$;
- GL15** Ако важи $\alpha = \beta/2^{L_1}$ и $\alpha = K_1 \cdot \gamma/2^{K_2} + K_3 \cdot \pi/2^{K_4}$ онда важи и $\beta = K_1 \cdot \gamma/2^{K_2-L_1} + K_3 \cdot \pi/2^{K_4-L_1}$;
- GL16** Ако важи $\overrightarrow{XY}/\overrightarrow{XZ} = r$, тачка X припада кругу C , а тачка Z му не припада, онда тачка Y припада слици круга C при хомотетији са центром у тачки Z са коефицијентом $r/(r-1)$;
- GL17** Ако је $\alpha = \beta$ онда важи да је $\beta = \alpha$;
- GL18** Ако је $\alpha = \beta/2$ онда важи да је $\beta = 2 \cdot \alpha$;
- GL19** Ако је $\alpha = \beta/2 + \pi$ онда важи да је $\beta = 2 \cdot \alpha - 2 \cdot \pi$.
- GL20** Ако је $\alpha = \beta/2 + \pi/2$ онда важи да је $\beta = 2 \cdot \alpha - \pi$.

За потребе решавања проблема из Конелијевог корпуса потребне су и следеће леме:

- L15–L16** $\overrightarrow{NG}/\overrightarrow{NO} = 1/3$; $\overrightarrow{HN}/\overrightarrow{HG} = 3/4$;
- L17–L24** тачке $M_b, M_c, H_a, H_b, H_c, E_a, E_b$ и E_c припадају кругу $k(N, M_a)$;
- L46–L54** тачке H, H_b и H_c припадају кругу $k(E_a, A)$; тачке H, H_a и H_c припадају кругу $k(E_b, B)$; тачке H, H_b и H_a припадају кругу $k(E_c, C)$;
- L116–L118** бисектриса угла $\angle BAC$ паралелна је правој $M_a P_a^1$; бисектриса угла $\angle ACB$ паралелна је правој $M_c P_c^1$; бисектриса угла $\angle CBA$ паралелна је правој $M_b P_b^1$;
- L119** Ојлеров круг и уписани круг се додирују изнутра.

B.2 Примитивне конструкције

Примитивне конструкције које се користе за решавање проблема из Верниковог корпуса су:

- W01** Ако су дате тачке X , Z и W , и рационалан број r могуће је конструисати тачку Y тако да важи: $\overrightarrow{XY}/\overrightarrow{ZW} = r$;
- W02** Ако су дате две тачке X и Y могуће је конструисати праву XY ; услов одређености је да су тачке X и Y различите;
- W03** Ако су дате две праве, могуће је конструисати њихову заједничку тачку; услов недегенерисаности је да праве нису паралелне, а услов одређености да нису једнаке;
- W04** Ако су дати права и круг, могуће је конструисати њихове заједничке тачке; услов недегенерисаности је да се права и круг секу;
- W05** Ако су дати права и круг и једна њихова заједничка тачка, могуће је конструисати њихову другу заједничку тачку; услов недегенерисаности је да се права и круг секу;
- W06** Ако су дате две различите тачке X и Y могуће је конструисати круг $k(X, Y)$ са центром у тачки X који садржи тачку Y ; услов недегенерисаности је да су тачке X и Y различите;
- W07** Ако су дата два круга, могуће је конструисати њихове две заједничке тачке; услов недегенерисаности је да се кругови секу, а услов одређености да су кругови различити;
- W08** Ако су дата два круга и једна њихова заједничка тачка, могуће је конструисати њихову другу заједничку тачку; услов недегенерисаности је да се кругови секу, а услов одређености да су кругови различити;
- W09** Ако су дате тачке X и Y могуће је конструисати круг са пречником $\overline{XY}$; услов недегенерисаности је да су тачке различите;
- W10** Ако су дати тачка X и права p могуће је конструисати праву q која садржи тачку X и управна је на праву p ;
- W11** Ако су дати права p и тачка X која не припада правој p могуће је конструисати круг k са центром X који додирује праву p ; услов недегенерисаности је да тачка X не припада правој p ;

- W12** Ако су дати круг k и тачка X ван круга k , могуће је конструисати две тангенте из тачке X на круг k ; услов недегенерисаности је да је тачка X ван круга k ;
- W13** Ако су дати круг k , тачка X ван круга k и једна тангента из тачке X на круг k , могуће је конструисати другу тангенту из тачке X на круг k ; услов недегенерисаности је да је тачка X ван круга k ;
- W14** Ако су дате тачке X и Y могуће је конструисати медијатрису дужи $\overline{XY}$; услов одређености је да су тачке X и Y различите;
- W15** Ако су дати тачка X , права p и рационалан број r , могуће је конструисати праву која је слика праве p при хомотетији у односу на тачку X са коефицијентом r ;
- W16** Ако су дати тачка X и права p могуће је конструисати праву која садржи тачку X и паралелна је правој p ;
- W17** Ако су дате тачке X и Y и угао α могуће је конструисати праву q тако да је угао $\angle(\overline{XY}, q) = A \cdot \alpha/2^B + C \cdot \pi/2^D$; ²
- W19** Ако су дате тачке X , Y и Z могуће је конструисати тачку W која је хармонијски спрегнута са осталим; услов недегенерисаности је да су тачке X и Y различите, тачке Y и Z различите и да тачка Y није средиште дужи XZ ;
- W20** Ако су дате тачке X и Y и угао α могуће је конструисати скуп тачака из којих се дуж $\overline{XY}$ види под углом $A \cdot \alpha/2^B + C \cdot \pi/2^D$.

За потребе решавања проблема из Конелијевог корпуса потребне је и наредна примитивна конструкција:

- W22** Ако су дати тачка X и круг k_1 могуће је конструисати круг k_2 са центром у тачки X који изнутра додирује круг k_1 ; услов недегенерисаности је да је тачка X унутар круга k_1 и да тачка X није центар круга k_1 ;

²У овом и наредним правилима користи се својство да ако је угао α конструктибилан, онда је конструктибилан и угао $K_1 \cdot \alpha/2^{K_2} + K_3 \cdot \pi/2^{K_4}$, где су K_1, K_2, K_3, K_4 природни бројеви

Додатак С

Примери решених задатака

C.1 Problem 582

C.1.1 Problem

Problem 582: Given a point A , a point E_a and a point T_a , construct the triangle ABC .

C.1.2 Status

The problem is solvable.

C.1.3 Solution

Construction in natural language form

1. Using the point A and the point E_a , construct a point H (rule W01); ;
2. Using the point A and the point E_a , construct a line h_a (rule W02); % DET: points A and E_a are not the same;
3. Using the point A and the point T_a , construct a line s_a (rule W02); % DET: points A and T_a are not the same;
4. Using the point T_a and the line h_a , construct a line a (rule W10); ;
5. Using the point E_a , the point A , the point T_a , the line s_a and the line h_a , construct a line AO (rule W17); % NDG: points A and T_a are not the same; points E_a and A are not the same % DET: points A and T_a are not the same;
6. Using the point E_a and the line AO , construct a line $m(H_bH_c)$ (rule W16); ;

7. Using the line $m(H_bH_c)$ and the line a , construct a point M_a (rule W03); %
NDG: lines $m(H_bH_c)$ and a are not parallel % DET: lines $m(H_bH_c)$ and a are
not the same;
8. Using the point M_a and the point A , construct a point G (rule W01); ;
9. Using the point H and the point G , construct a point O (rule W01); ;
10. Using the point A and the point O , construct a circle $k(O, C)$ (rule W06); %
NDG: points A and O are not the same;
11. Using the circle $k(O, C)$ and the line a , construct a point C and a point B
(rule W04); % NDG: line a and circle $k(O, C)$ intersect.

Non-degenerate conditions: line a and circle $k(O, C)$ intersect; points A and O are not the same; lines $m(H_bH_c)$ and a are not parallel; points A and T_a are not the same; points E_a and A are not the same.

Determination conditions: lines $m(H_bH_c)$ and a are not the same; points A and T_a are not the same; points A and T_a are not the same; points A and E_a are not the same.

Rules used: [W01, W02, W03, W04, W06, W10a, W16, W17]

Definitions and lemmas used: [D21, D23, D26, D28, D3, D8, GD01, GD02, GL01, GL03, GL04, GL09, GL17, L104, L11, L12, L38, L39, L47, L48, L5, L55, L58]

Solving time: 27.8 seconds.

Construction in GCLC language

```
dim 120 120
```

```
point A 80 95
```

```
point E_{a} 80 83.86
```

```
point T_{a} 70.86 40
```

```
color 220 0 0
```

```
fontsize 9
```

```
cmark_t A
```

```
cmark_r E_{a}
```

```
cmark_rb T_{a}
```

```
color 0 0 0
```

```
fontsize 8
```

```

% Constructing a point H such that  $AH/AE_{\{a\}}=2$ 
towards H A E_{a} 2
cmark_rt H
color 200 200 200
drawsegment A H
color 0 0 0

% DET: points A and E_{a} are not the same
% Constructing a line h_{a} which passes through point A and point E_{a}
line h_{a} A E_{a}

color 200 200 200
drawline h_{a}
color 0 0 0

% DET: points A and T_{a} are not the same
% Constructing a line s_{a} which passes through point A and point T_{a}
line s_{a} A T_{a}

color 200 200 200
drawline s_{a}
color 0 0 0

% Constructing a line a which is perpendicular to line h_{a} and which passes
    through point T_{a}
perp a T_{a} h_{a}

color 200 200 200
drawline a
color 0 0 0

% NDG: points A and T_{a} are not the same; points E_{a} and A are not the same%
    DET: points A and T_{a} are not the same
% Constructing an angle V[_G6421] which is equal to the angle E_{a}AT_{a}
angle_o V[_G6421] E_{a} A T_{a}

% Calculating value angle[_G6500] using formula  $1/\text{pow}(2,0)*V[_G6421]+0/\text{pow}(2,0)$ 
    *180
expression angle[_G6500] { 1/pow(2,0)*V[_G6421]+0/pow(2,0)*180 }

% Constructing a point P_{\_G6497} which is an image of the point T_{a} in a
    rotation around the point A for the angle  $1/\text{pow}(2,0)*V[_G6421]+0/\text{pow}(2,0)*180$ 
rotate P_{\_G6497} A angle[_G6500] T_{a}
cmark_r P_{\_G6497}
color 200 200 200

```

```

drawarc_p A T_{a} angle[_G6500]
color 0 0 0

% Constructing a line AO which passes through point A and point P_{\_G6497}
line AO A P_{\_G6497}

color 200 200 200
drawline AO
color 0 0 0

% Constructing a line m(H_{b}H_{c}) which contains the point E_{a} and is
parallel to the line AO
parallel m(H_{b}H_{c}) E_{a} AO

color 200 200 200
drawline m(H_{b}H_{c})
color 0 0 0

% NDG: lines m(H_{b}H_{c}) and a are not parallel% DET: lines m(H_{b}H_{c}) and a
are not the same
% Constructing a point M_{a} which belongs to line m(H_{b}H_{c}) and line a
intersec M_{a} m(H_{b}H_{c}) a
cmark_r M_{a}

% Constructing a line L_{\_G7014} which passes through point M_{a} and point A
line L_{\_G7014} M_{a} A

color 200 200 200
drawline L_{\_G7014}
color 0 0 0

% Constructing a point P_{\_G7115} with coordinates (0,0)
point P_{\_G7115} 0 0
cmark_r P_{\_G7115}

% Constructing a point P_{\_G7039} such that M_{a}P_{\_G7039}/M_{a}P_{\_G7115}=1
towards P_{\_G7039} M_{a} P_{\_G7115} 1
cmark_r P_{\_G7039}
color 200 200 200
drawsegment M_{a} P_{\_G7039}
color 0 0 0

% Constructing a point P_{\_G7084} such that M_{a}P_{\_G7084}/M_{a}P_{\_G7115}=3
towards P_{\_G7084} M_{a} P_{\_G7115} 3
cmark_r P_{\_G7084}

```

```

color 200 200 200
drawsegment M_{a} P_{\_G7084}
color 0 0 0

% Constructing a line L_{\_G7045} which passes through point A and point P_{\_G
7084}
line L_{\_G7045} A P_{\_G7084}

color 200 200 200
drawline L_{\_G7045}
color 0 0 0

% Constructing a line L_{\_G7008} which contains the point P_{\_G7039} and is
parallel to the line L_{\_G7045}
parallel L_{\_G7008} P_{\_G7039} L_{\_G7045}

color 200 200 200
drawline L_{\_G7008}
color 0 0 0

% Constructing a point G which belongs to line L_{\_G7008} and line L_{\_G7014}
intersec G L_{\_G7008} L_{\_G7014}
cmark_t G

% Constructing a point O such that HO/HG=1.5
towards O H G 1.5
cmark_t O
color 200 200 200
drawsegment H O
color 0 0 0

% NDG: points A and O are not the same
% Constructing a circle k(O,C) whose center is at point O and which passes
through point A
circle k(O,C) O A

color 200 200 200
drawcircle k(O,C)
color 0 0 0

% NDG: line a and circle k(O,C) intersect
% Constructing points C and B which are in intersection of k(O,C) and a
intersec2 C B k(O,C) a
cmark_l C
cmark_b B

```


Space Complexity: The biggest polynomial obtained during prover execution contains 0 terms.

Time Complexity: Time spent by the prover is 0.029 seconds.

NDG conditions There are no NDG conditions for this theorem

Proving $E_a = E_a$

Status: Theorem has been proved.

Space Complexity: The biggest polynomial obtained during prover execution contains 103 terms.

Time Complexity: Time spent by the prover is 4.005 seconds.

NDG conditions Points A and B are not identical

Points T_a , A and H are not collinear

Line through points $P_{G256897}$ and A is not perpendicular to line through points A and E_a

Points A , $P_{G257295}$ and M_a are not collinear

Points A and $P_{G257295}$ are not identical

Line through points A and E_a is not parallel with line through points B and C

Proving $T_a = T_a$ Proving failed

GCLC - Area method

Proving $A = A$ NDG conditions are:

$S_{T_a A E_a} \neq 0$ i.e., points T_a , A and E_a are not collinear (foot is not the point itself; construction based assumption)

$S_{E_a T_a F_a^0} \neq S_{P_{m(H_b H_c)}^1 T_a F_a^0}$ i.e., lines $E_a P_{m(H_b H_c)}^1$ and $T_a F_a^0$ are not parallel (construction based assumption)

$S_{P_{G246088} M_a A} \neq S_{P_{L_{G246057}}^2 M_a A}$ i.e., lines $P_{G246088} P_{L_{G246057}}^2$ and $M_a A$ are not parallel (construction based assumption)

$S_{ABC} \neq 0$ i.e., points A , B and C are not collinear (foot is not the point itself; construction based assumption)

$S_{BAC} \neq 0$ i.e., points B , A and C are not collinear (foot is not the point itself; construction based assumption)

$S_{ABF^4_{h_b}} \neq S_{F^3_{h_a}BF^4_{h_b}}$ i.e., lines $AF^3_{h_a}$ and $BF^4_{h_b}$ are not parallel (construction based assumption)

Total number of proof steps: 1

Time spent by the prover: 0.000 seconds

Proving $E_a = _E_a$ Proving failed

Proving $T_a = _T_a$ Proving failed

GCLC - Wu method

Proving $A = A$

Status: The conjecture has been proved.

Space Complexity: The biggest polynomial obtained during proof process contained 0 terms.

Time Complexity: Time spent by the prover is 0.010 seconds. There are no ndg conditions.

Proving $E_a = _E_a$

Status: The conjecture has not been proved nor disproved.

Space Complexity: The biggest polynomial obtained during proof process contained 3720 terms.

Time Complexity: Time spent by the prover is 8.090 seconds. There are no ndg conditions.

Proving $T_a = _T_a$

Status: The conjecture has not been proved nor disproved.

Space Complexity: The biggest polynomial obtained during proof process contained 8 terms.

Time Complexity: Time spent by the prover is 0.020 seconds. There are no ndg conditions.

GCLC - Grobner basis method**Proving** $A=A$ **Status:** The conjecture has been proved.**Space Complexity:** The biggest polynomial obtained during proof process contained 0 terms.**Time Complexity:** Time spent by the prover is 0.010 seconds. There are no ndg conditions.**Proving** $E_a=_E_a$ Proving failed**Proving** $T_a=_T_a$ Proving failed

C.2 Problem 589

C.2.1 Problem

Problem 589: Given a point E_b , a point H_b and a point A , construct the triangle ABC .

C.2.2 Status

The problem is locus dependent.

C.2.3 Solution

Construction in natural language form

1. Using the point E_b and the point H_b , construct a line h_b (rule W02); % DET: points E_b and H_b are not the same;
2. Using the point H_b and the line h_b , construct a line b (rule W10); ;
3. Choose freely a point A on the line b (rule WOnline2);
4. Choose freely a point B on the line h_b (rule WOnline1) ;
5. Using the point B and the point E_b , construct a point H (rule W01); ;
6. Using the point B and the point A , construct a line c (rule W02); % DET: points B and A are not the same;

7. Using the point B and the point E_b , construct a circle $k(E_b, B)$ (rule W06);
% NDG: points B and E_b are not the same;
8. Using the circle $k(E_b, B)$, the line c , the point E_b and the point B , construct
a point H_c (rule W05); % NDG: line c and circle $k(E_b, B)$ intersect % DET:
points B and H_c must be different;
9. Using the point H and the point H_c , construct a line h_c (rule W02); % DET:
points H and H_c are not the same;
10. Using the line b and the line h_c , construct a point C (rule W03); % NDG: lines
 b and h_c are not parallel % DET: lines b and h_c are not the same.

Non-degenerate conditions: lines b and h_c are not parallel; line c and circle $k(E_b, B)$ intersect; points B and E_b are not the same.

Determination conditions: lines b and h_c are not the same; points H and H_c are not the same; points B and H_c must be different; points B and A are not the same; points E_b and H_b are not the same.

Rules used: [W01, W02, W03, W05, W06, W10, WOnline1, WOnline2]

Definitions and lemmas used: [D10, D29, D3, D6, D7, D9, GD01, GD02, GL03, GL09, L3, L51]

Solving time: 63.2 seconds.

Construction in GCLC language

```
dim 120 120
```

```
point E_{b} 50 56.36
```

```
point H_{b} 89.36 77.83
```

```
point A 80 95
```

```
color 220 0 0
```

```
fontsize 9
```

```
cmark_r E_{b}
```

```
cmark_l H_{b}
```

```
cmark_t A
```

```
color 0 0 0
```

```
fontsize 8
```

```
% DET: points E_{b} and H_{b} are not the same
```

```
% Constructing a line h_{b} which passes through point E_{b} and point H_{b}
```

```
line h_{b} E_{b} H_{b}
```

```

color 200 200 200
drawline h_{b}
color 0 0 0

% Constructing a line b which is perpendicular to line h_{b} and which passes
    through point H_{b}
perp b H_{b} h_{b}

color 200 200 200
drawline b
color 0 0 0

% Generating random value V[_G85528]
random V[_G85528]

% Calculating value V[_G85549] using formula V[_G85528]*20
expression V[_G85549] { V[_G85528]*20 }

% Constructing a point A which is a point for which holds H_{b}A = V[_G85549] and
    angle E_{b}H_{b}A = 90
turtle A E_{b} H_{b} 90 V[_G85549]
cmark_t A

% Choosing randomly a point B on the line E_{b}H_{b}
online B E_{b} H_{b}
cmark_b B
color 200 200 200
drawline E_{b} H_{b}
color 0 0 0

% Constructing a point H such that BH/BE_{b}=2
towards H B E_{b} 2
cmark_rt H
color 200 200 200
drawsegment B H
color 0 0 0

% DET: points B and A are not the same
% Constructing a line c which passes through point B and point A
line c B A

color 200 200 200
drawline c
color 0 0 0

```

```

% NDG: points B and E_{b} are not the same
% Constructing a circle k(E_{b},B) whose center is at point E_{b} and which
    passes through point B
circle k(E_{b},B) E_{b} B

color 200 200 200
drawcircle k(E_{b},B)
color 0 0 0

% NDG: line c and circle k(E_{b},B) intersect% DET: points B and H_{c} must be
    different
% Constructing a point P_{\_G86069} which is a foot of the point E_{b} on the
    line c
foot P_{\_G86069} E_{b} c
cmark_r P_{\_G86069}
color 200 200 200
drawline E_{b} P_{\_G86069}
color 0 0 0

% Constructing a point H_{c} which is an image of the point B in the symmetry to
    point/line P_{\_G86069}
sim H_{c} P_{\_G86069} B
cmark_rt H_{c}

% DET: points H and H_{c} are not the same
% Constructing a line h_{c} which passes through point H and point H_{c}
line h_{c} H H_{c}

color 200 200 200
drawline h_{c}
color 0 0 0

% NDG: lines b and h_{c} are not parallel% DET: lines b and h_{c} are not the
    same
% Constructing a point C which belongs to line b and line h_{c}
intersec C b h_{c}
cmark_l C

drawsegment A B
drawsegment A C
drawsegment B C

% Non-degenerate conditions: lines b and h_{c} are not parallel; line c and
    circle k(E_{b},B) intersect; points B and E_{b} are not the same

```


Line through points B and H_c is not perpendicular to line through points H_c and E_b

Points A , C and E_b are not collinear

Proving $H_b = H_b$

Status: Theorem has been proved.

Space Complexity: The biggest polynomial obtained during prover execution contains 6 terms.

Time Complexity: Time spent by the prover is 0.152 seconds.

NDG conditions Points A and B are not identical

Point E_b is not on circle with center H_b and point from it B

Line through points B and H_c is not perpendicular to line through points H_c and E_b

Points A , C and E_b are not collinear

Proving $A = A$

Status: Theorem has been proved.

Space Complexity: The biggest polynomial obtained during prover execution contains 0 terms.

Time Complexity: Time spent by the prover is 0.013 seconds.

NDG conditions There are no NDG conditions for this theorem

GCLC - Area method

Proving $E_b = E_b$ Proving failed

Proving $H_b = H_b$ Proving failed

Proving $A = A$ Proving failed

GCLC - Wu method**Proving** $E_b = _ E_b$ **Status:** The conjecture has not been proved nor disproved.**Space Complexity:** The biggest polynomial obtained during proof process contained 9 terms.**Time Complexity:** Time spent by the prover is 0.000 seconds. There are no ndg conditions.**Proving** $H_b = _ H_b$ **Status:** The conjecture has not been proved nor disproved.**Space Complexity:** The biggest polynomial obtained during proof process contained 16 terms.**Time Complexity:** Time spent by the prover is 0.020 seconds. There are no ndg conditions.**Proving** $A = A$ Proving failed**GCLC - Grobner basis method****Proving** $E_b = _ E_b$ **Status:** The conjecture has not been proved nor disproved.**Space Complexity:** The biggest polynomial obtained during proof process contained 20 terms.**Time Complexity:** Time spent by the prover is 0.020 seconds. There are no ndg conditions.**Proving** $H_b = _ H_b$ **Status:** The conjecture has not been proved nor disproved.**Space Complexity:** The biggest polynomial obtained during proof process contained 20 terms.**Time Complexity:** Time spent by the prover is 0.050 seconds. There are no ndg conditions.

Proving $A=A$ Proving failed

C.3 Problem 42

C.3.1 Problem

Problem 42: Given a point M_a , a point G and a point A , construct the triangle ABC .

C.3.2 Status

The problem is redundant.

C.3.3 Solution

Construction in natural language form

1. Using the point M_a and the point G , construct a point A (rule W01); ;
2. Choose freely a point B (rule free);
3. Using the point B and the point M_a , construct a point C (rule W01); .

Non-degenerate conditions: .

Determination conditions: .

Rules used: [W01, free]

Definitions and lemmas used: [D21, GL03, GL04, L55]

Solving time: 0.0 seconds.

Construction in GCLC language

```
dim 120 120
```

```
point M_{a} 65 40
```

```
point G 70 58.33
```

```
point A 80 95
```

```
color 220 0 0
```

```
fontsize 9
```

```
cmark_r M_{a}
```

```
cmark_t G
```

```
cmark_t A
```

```

color 0 0 0
fontsize 8

% Constructing a point A such that  $M_{\{a\}}A/M_{\{a\}}G=3$ 
towards A M_{a} G 3
cmark_t A
color 200 200 200
drawsegment M_{a} A
color 0 0 0

% Constructing a free point B
point B 20 40

cmark_b B

% Constructing a point C such that  $BC/BM_{\{a\}}=2$ 
towards C B M_{a} 2
cmark_l C
color 200 200 200
drawsegment B C
color 0 0 0

drawsegment A B
drawsegment A C
drawsegment B C

% Non-degenerate conditions:
% Determination conditions:

```

Illustration

Illustration of the constructed figure is given in [Figure C.3](#)

C.3.4 Correctness proof

OGP - Wu method

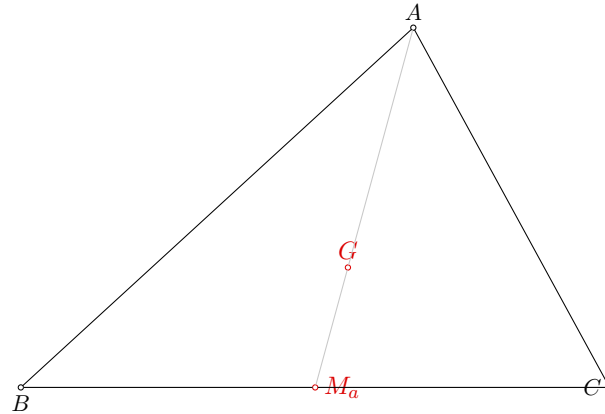
Proving $M_a = _M_a$

Status: Theorem has been proved.

Space Complexity: The biggest polynomial obtained during prover execution contains 3 terms.

Time Complexity: Time spent by the prover is 0.054 seconds.

NDG conditions There are no NDG conditions for this theorem

Слика C.3: *Illustration of the problem 0042*

Proving $G = _G$

Status: Theorem has been proved.

Space Complexity: The biggest polynomial obtained during prover execution contains 77 terms.

Time Complexity: Time spent by the prover is 0.784 seconds.

NDG conditions Line through points A and $_M_a$ is not parallel with line through points B and $_M_b$

Points $_M_a$, G and M_a are not collinear

Proving $A = A$

Status: Theorem has been proved.

Space Complexity: The biggest polynomial obtained during prover execution contains 2 terms.

Time Complexity: Time spent by the prover is 0.025 seconds.

NDG conditions There are no NDG conditions for this theorem

GCLC - Area method

Proving $M_a = M_a$ NDG conditions are:

$S_{AB_{M_b}} \neq S_{M_a B_{M_b}}$ i.e., lines A_{M_a} and B_{M_b} are not parallel (construction based assumption)

Total number of proof steps: 41

Time spent by the prover: 0.000 seconds

Proving $G = G$ NDG conditions are:

$S_{AB_{M_b}} \neq S_{M_a B_{M_b}}$ i.e., lines A_{M_a} and B_{M_b} are not parallel (construction based assumption)

$S_{ABC} \neq 0$ i.e., points A , B and C are not collinear (cancellation assumption)

Total number of proof steps: 588

Time spent by the prover: 0.120 seconds

Proving $A = A$ NDG conditions are:

$S_{AB_{M_b}} \neq S_{M_a B_{M_b}}$ i.e., lines A_{M_a} and B_{M_b} are not parallel (construction based assumption)

Total number of proof steps: 1

Time spent by the prover: 0.000 seconds

GCLC - Wu method

Proving $M_a = M_a$

Status: The conjecture has been proved.

Space Complexity: The biggest polynomial obtained during proof process contained 12 terms.

Time Complexity: Time spent by the prover is 0.010 seconds. There are no ndg conditions.

Proving $G = G$

Status: The conjecture has been proved.

Space Complexity: The biggest polynomial obtained during proof process contained 939 terms.

Time Complexity: Time spent by the prover is 1.420 seconds. There are no ndg conditions.

Proving $A=A$

Status: The conjecture has been proved.

Space Complexity: The biggest polynomial obtained during proof process contained 0 terms.

Time Complexity: Time spent by the prover is 0.000 seconds. There are no ndg conditions.

GCLC - Grobner basis method

Proving $M_a = _M_a$

Status: The conjecture has been proved.

Space Complexity: The biggest polynomial obtained during proof process contained 319 terms.

Time Complexity: Time spent by the prover is 15.460 seconds. There are no ndg conditions.

Proving $G = _G$

Status: The conjecture has been proved.

Space Complexity: The biggest polynomial obtained during proof process contained 319 terms.

Time Complexity: Time spent by the prover is 15.470 seconds. There are no ndg conditions.

Proving $A=A$

Status: The conjecture has been proved.

Space Complexity: The biggest polynomial obtained during proof process contained 0 terms.

Time Complexity: Time spent by the prover is 0.000 seconds. There are no ndg conditions.

Биографија кандидата

Весна Н. Маринковић (рођена Павловић) рођена је у Нишу 24. марта 1982. године. Основну школу “Вожд Карађорђе” и Гимназију “Бора Станковић” завршила је у Нишу са просечном оценом 5,00 као ђак генерације и носилац Вукових диплома. Изабрана је за најбољег математичара у генерацији града Ниша 1994. године. Током основне и средње школе била је добитник великог броја награда на такмичењима, од којих су најважније прва награда на републичком такмичењу из математике и из информатике у осмом разреду основне школе, као и прво место у Б категорији на републичком такмичењу из математике у трећем и четвртом разреду средње школе. Током средње школе учествовала је на међународном математичком такмичењу “Турнир градова” у Хамбургу (Немачка) и освојила трећу награду. У Нишу је завршила и Основну музичку школу “Др Војислав Вучковић” са одличним успехом.

Студије на смеру Рачунарство и информатика на Математичком факултету Универзитета у Београду уписала је школске 2000/01. године, а дипломирала је јула 2006. године са просечном оценом 9,45. Током основних студија два пута је боравила по три месеца на Електротехничком факултету у Љубљани (Словенија) у оквиру програма студентске размене. Била је стипендиста Министарства за науку и технологију. Докторске академске студије на смеру Рачунарство и информатика уписала је јануара 2007. године. Током докторских студија радила је током три месеца као гостујући истраживач на Политехничком универзитету у Валенсији (Шпанија), у оквиру пројекта DEUKS, а такође учествовала у раду две летње школе у области теоријског рачунарства: у Минхену (Немачка) и Дагштулу (Немачка) и на неколико међународних конференција у земљи и иностранству.

Од октобра 2006. године запослена је као сарадник у настави, а од октобра 2009. године као асистент на Катедри за рачунарство Математичког факултета Универзитета у Београду. Током досадашњег рада на Математичком факултету држала је вежбе из десет предмета. Члан је групе за аутоматско резоновање Математичког факултета. Од 2006. до 2010. године била је укључена

чена у пројекат 144030 Министарства науке Републике Србије, а од 2011. до данас укључена је у пројекат 174021 Министарства просвете и науке Републике Србије.

Прилог 1.

Изјава о ауторству

Потписани-а _____

број индекса _____

Изјављујем

да је докторска дисертација под насловом

- резултат сопственог истраживачког рада,
- да предложена дисертација у целини ни у деловима није била предложена за добијање било које дипломе према студијским програмима других високошколских установа,
- да су резултати коректно наведени и
- да нисам кршио/ла ауторска права и користио интелектуалну својину других лица.

Потпис докторанда

У Београду, _____

Прилог 2.

**Изјава о истоветности штампане и електронске
верзије докторског рада**

Име и презиме аутора _____

Број индекса _____

Студијски програм _____

Наслов рада _____

Ментор _____

Потписани/а _____

Изјављујем да је штампана верзија мог докторског рада истоветна електронској верзији коју сам предао/ла за објављивање на порталу **Дигиталног репозиторијума Универзитета у Београду**.

Дозвољавам да се објаве моји лични подаци везани за добијање академског звања доктора наука, као што су име и презиме, година и место рођења и датум одбране рада.

Ови лични подаци могу се објавити на мрежним страницама дигиталне библиотеке, у електронском каталогу и у публикацијама Универзитета у Београду.

Потпис докторанда

У Београду, _____

Прилог 3.

Изјава о коришћењу

Овлашћујем Универзитетску библиотеку „Светозар Марковић“ да у Дигитални репозиторијум Универзитета у Београду унесе моју докторску дисертацију под насловом:

која је моје ауторско дело.

Дисертацију са свим прилозима предао/ла сам у електронском формату погодном за трајно архивирање.

Моју докторску дисертацију похрањену у Дигитални репозиторијум Универзитета у Београду могу да користе сви који поштују одредбе садржане у одабраном типу лиценце Креативне заједнице (Creative Commons) за коју сам се одлучио/ла.

1. Ауторство
2. Ауторство - некомерцијално
3. Ауторство – некомерцијално – без прераде
4. Ауторство – некомерцијално – делити под истим условима
5. Ауторство – без прераде
6. Ауторство – делити под истим условима

(Молимо да заокружите само једну од шест понуђених лиценци, кратак опис лиценци дат је на полеђини листа).

Потпис докторанда

У Београду, _____

1. Ауторство - Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце, чак и у комерцијалне сврхе. Ово је најслободнија од свих лиценци.

2. Ауторство – некомерцијално. Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела.

3. Ауторство - некомерцијално – без прераде. Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца не дозвољава комерцијалну употребу дела. У односу на све остале лиценце, овом лиценцом се ограничава највећи обим права коришћења дела.

4. Ауторство - некомерцијално – делити под истим условима. Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца не дозвољава комерцијалну употребу дела и прерада.

5. Ауторство – без прераде. Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, без промена, преобликовања или употребе дела у свом делу, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце. Ова лиценца дозвољава комерцијалну употребу дела.

6. Ауторство - делити под истим условима. Дозвољавање умножавање, дистрибуцију и јавно саопштавање дела, и прераде, ако се наведе име аутора на начин одређен од стране аутора или даваоца лиценце и ако се прерада дистрибуира под истом или сличном лиценцом. Ова лиценца дозвољава комерцијалну употребу дела и прерада. Слична је софтверским лиценцама, односно лиценцама отвореног кода.