Komputlingvoscienca kaj logika analizado de matematikaj tekstoj

Marcos Cramer
Universitato de Bonn
cramer@math.uni-bonn.de
http://www.naproche.net

Resumo

Ni prezentas la projekton Naproche, kiu studas la matematikan faklingvon kaj celas krei komputilan programon kiu povu analizi matematikajn tekstojn per metodoj de la komputa lingvoscienco kaj kontroli ilian logikan senerarecon. La matematika faklingvo havas apartajn interesajn ecojn: La lingvaĵo konsistas el miksaĵo de formuloj kaj naturlingvaj elementoj. La uzata lingvo estas daŭre riĉigata pere de difinoj: Aldoniĝas novaj vortoj kaj novaj simboloj kun precize fiksita signifo. La ĉefa parto de matematikaj faktekstoj estas matematikaj pruvoj, en kiuj ĉiu aserto devas logike sekvi el antaŭe konataj asertoj. Ni komparas ĉi tiajn ecojn kun la ecoj de la ĝenerala lingvo, kaj montras kiel ili influas la kreadon de la celata programo.

1 La matematika faklingvo

Per la esprimo *matematika faklingvo* ni celas la lingvaĵon de universitat-nivelaj matematikaj lernolibroj kaj de publikigaĵoj en matematikaj fakrevuoj, kaj per la esprimo *matematikaj tekstoj* ni celas ajnajn tekstojn skribitajn en la matematika faklingvo. Unuavide matematikaj tekstoj impresas pro la multeco de formuloj aperantaj en ili. Tamen estas ankaŭ multaj naturlingvaj elementoj en matematikaj tekstoj, kiuj obeas la kutimajn gramatikajn regulojn de la lingvo de la teksto. Supre de la sekva paĝo estas ekzemplo de matematika teksto el Matthias (1995) (paĝo 19).

Oni povas distingi du specojn de enhavo en matematika teksto:

- La *matematika enhavo*, kiu temas pri la matematikaj objektoj (nombroj, funkcioj, vektoroj, aroj, korpoj, grupoj, topologiaj spacoj ktp) kaj iliaj matematikaj ecoj kaj rilatoj (esti para/nepara, esti derivaĵo de, esti subaro de, ktp).
- La *metamatematika enhavo*, ekzemple informoj pri tio, por kio utilas iu difino aŭ teoremo, informoj pri tio, kiu unuafoje pruvis iun teoremon, kaj klarigoj pri tio, kial iu pruvmetodo estas uzata aŭ ne uzata en certa situacio.

Por la resto de ĉi tiu artikolo nin interesas nur la matematika enhavo de matematikaj tekstoj.

4.4 Interŝanĝo de bazaj vektoroj

Teoremo 4.4.1 *Estu $w_1, \ldots, w_s$ lineare sendependaj vektoroj el vektorspaco V kun bazo $\{v_1, \ldots, v_n\}$. Tiam ni povas elekti el la vektoroj $v_1, \ldots, v_n$ vektorojn $v'_1, \ldots, v'_m$ tiel, ke ankaŭ $\{w_1, \ldots, w_s, v'_1, \ldots, v'_m\}$ estas bazo de V.*

Pruvo. Konsideru ĉiujn subarojn de $M = \{w_1, \ldots, w_s, v_1, \ldots, v_n\}$, kiuj entenas $M_1 = \{w_1, \ldots, w_s\}$ kaj estas generantaroj de V. Almenaŭ M mem havas tiujn ĉi ecojn. Elektu tian subaron kun minimuma nombro de vektoroj: $M_2 = \{w_1, \ldots, w_s, v'_1, \ldots, v'_m\}$. Ni nun montras per nerekta pruvo, ke M_2 estas lineare sendependa, kio pruvas la teoremon. Se M_2 estus lineare dependa, ni havus $\lambda_1, \ldots, \lambda_s, \mu_1, \ldots, \mu_m \in K$, kiuj ne ĉiuj estas nuloj, tiel ke

$$\lambda_1 w_1 + \ldots + \lambda_s w_s + \mu_1 v'_1 + \ldots + \mu_m v'_m = 0$$

Nun ekzistas iu $i \in \{1, \ldots, m\}$ kun $\mu_i \neq 0$, ĉar alikaze $\{w_1, \ldots, w_s\}$ estus lineare dependa. Tio signifas, ke ne nur $\mu_i v'_i \in \mathrm{Lin}\{w_1, \ldots, w_s, v'_1, \ldots, v'_{i-1}, v'_{i+1}, \ldots, v'_m\}$, sed eĉ $v'_i \in \mathrm{Lin}\{w_1, \ldots, w_s, v'_1, \ldots, v'_{i-1}, v'_{i+1}, \ldots, v'_m\}$. Tio estas kontraŭdiro al la minimumeco de M_2, ĉar ankaŭ $M_2 \backslash \{v'_i\}$ havas la deziratajn ecojn. □

Matematikaj tekstoj kutime estas dividitaj en certajn partojn, kiuj strukturas la tekston, ekzemple en aksiomojn, difinojn, teoremojn, helpasertojn (ankaŭ nomatajn *lemoj*) kaj pruvojn. Ofte – precipe en pli altnivelaj tekstoj – la aksiomoj kaj difinoj bezonataj por la pruvoj estas parte ellasataj, ĉar ili apartenas al la fona scio de la koncerna subfako de la matematiko. Por la logika analizado de matematikaj tekstoj nin tamen interesas ĉefe tekstoj, kiuj estas logike fermitaj en si mem, do kies pruvoj dependas nur de aksiomoj kaj difinoj prezentitaj en la teksto mem.

Matematikaj tekstoj apartiĝas de aliaj tekstospecoj per apartaj specialaj ecoj: Kiel jam menciite, matematikaj tekstoj kombinas naturlingvajn esprimojn kun matematikaj simboloj kaj formuloj. Aldone, oni evitas malfacile interpreteblajn plursencajn frazojn. Unu maniero por eviti plursencecon estas la uzo de matematikaj simboloj, ekzemple pere de variantoj (x, y, z kaj simile) anstataŭ anaforaj pronomoj ("ĝi", "tio" kaj simile).

Oni povas enkonduki iun supozon, kaj pli poste en la teksto retiri tiun supozon. Jen ekzemplo de tio el Matthias (1995) (la graslitere markita supozo estas retirata fine de la pruvo, do ne plu uzebla poste):

> $\{u_1, \ldots, u_m, w_{r+1}, \ldots, w_n\}$ estas lineare sendependa:
> **Ni supozu, ke ekzistas $\boldsymbol{\lambda_1, \ldots, \lambda_m, \mu_{r+1}, \ldots, \mu_n \in K}$ tiel ke**
>
> $$\boldsymbol{\lambda_1 u_1 + \cdots + \lambda_m u_m + \mu_{r+1} w_{r+1} + \cdots + \mu_n w_n = 0}$$
>
> Tiam $v := \lambda_1 u_1 + \cdots + \lambda_m u_m = -\mu_{r+1} w_{r+1} - \cdots - \mu_n w_n$. Sekve $v \in U \cap W$. Nun $v \in Lin\{v_1, \ldots, v_r\}$ kaj $v \in Lin\{w_{r+1}, \ldots, w_n\}$. Ĉar $\{v_1, \ldots, v_r, w_{r+1}, \ldots, w_n\}$ estas lineare sendependa, ni konkludas ke $v = 0$. Ĉar $\{u_1, \ldots, u_m\}$ kaj $\{w_{r+1}, \ldots, w_n\}$ estas lineare sendependaj aroj (la lasta pro Rimarko 4.3.3), ni ricevas $\lambda_1 = \cdots = \lambda_m = \mu_r + 1 = \cdots = \mu_n = 0$. □

Matematikaj tekstoj estas tre strukturitaj: Je malloka nivelo, ili estas dividitaj en tekstoblokojn, ekzemple en aksiomojn, difinojn, teoremojn kaj pruvojn. Ene de pruvo, supozoj povas esti ingitaj en aliaj supozoj, tiel ke la validecregionoj de la supozoj difinas hierarĥian pruvo-strukturon.

Jen kelkaj pluaj specialaj ecoj de matematikaj tekstoj:

- Difinoj aldonas novajn simbolojn kaj esprimojn al la vortaro kaj fiksas iliajn signifojn.
- En pruvo, ĉiu aserto devas logike sekvi el aksiomoj, difinoj kaj antaŭe pruvitaj asertoj.
- La paŝoj de pruvo estas ofte pravigitaj per resendo al rezultoj el aliaj tekstoj aŭ el pli fruaj partoj de la sama teksto.

1.1 Formala kaj duonformala matematiko

Ene de la matematiko, ekzistas la subfako *formala matematiko*, kiu okupiĝas pri formalaj pruvoj skribitaj en formalaj lingvoj. *Formalaj lingvoj* kutime konsistas nur el formuloj, kaj tute evitas plursencecon. En *formala pruvo*, ĉiu paŝo de la pruvo devas sekvi el la antaŭaj paŝoj per klare difinitaj sintaksaj reguloj, tiel ke oni povas aŭtomate kontroli, ĉu iu formala pruvo estas senerara aŭ ne. Kutime matematikistoj ne prezentas siajn pruvojn en ĉi tia formala maniero, sed *duonformale* per la matematika faklingvo prezentita antaŭe. Por tio ekzistas diversaj kialoj, interalie la jenaj:

- Teksto en formala lingvo kutime estas malfacile legebla.
- En formala lingvo oni ne povas sin esprimi libere, sed nur laŭ malmultaj antaŭdifinitaj modeloj.
- Formalaj pruvoj kutime estas tede detalaj kaj longegaj, tiel ke oni pro la multaj detaloj pretervidas la esencon.

Jen ekzemple de formala pruvo:

Γ	$\neg(\varphi \to \neg\psi)$					(1)
Γ	$\neg\varphi$	$\neg\varphi$			supozregulo	(2)
Γ	$\neg\varphi$	φ	φ		supozregulo	(3)
Γ	$\neg\varphi$	φ	$\neg\varphi$		(2) + monotoneco	(4)
Γ	$\neg\varphi$	φ	$\perp$		(3,4) + $\perp$-enigo	(5)
Γ	$\neg\varphi$	φ	$\neg\neg\psi$	$\perp$	(5) + monotoneco	(6)
Γ	$\neg\varphi$	φ	$\neg\psi$		(6) + $\perp$-eligo	(7)
Γ	$\neg\varphi$	$\varphi \to \neg\psi$			(7) + $\to$-enigo	(8)
Γ	$\neg\varphi$	$\neg(\varphi \to \neg\psi)$			(1) + monotoneco	(9)
Γ	$\neg\varphi$	$\perp$			(8,9) + $\perp$-enigo	(10)
Γ	φ				(10) + $\perp$-eligo	(11)

2 La projekto Naproche

La projekto *Naproche* estas komuna projekto de la universitatoj de Bonn kaj Duisburg-Essen en Germanujo, kiu studas la matematikan faklingvon. Ĝia ĉefa celo estas la evoluigo de *reguligita lingvo* por matematikaj tekstoj (Cramer et al., 2010) kaj realigo de komputila programo, kiu povas aŭtomate transformi la tekstojn en tiu reguligita lingvo en samvalorajn formulojn de kutima predikata logiko per metodoj de la komputa lingvoscienco, kaj per tiuj formuloj kaj helpe de aŭtomataj teorem-pruviloj kontroli la logikan senerarecon de la teksto (Kühlwein et al., 2009).

Reguligita lingvo estas subaro de natura lingvo, kies vortprovizo kaj gramatiko estas limigitaj por eviti plursencecon kaj tiel ebligi ĝian logikan traktadon fare de komputilo. Momente la projekto koncentriĝas pri la anglalingva matematika faklingvo.

Ni celas ĉefe al du eblaj uz-kampoj de la Naproche-programo:

- Ĝia uzo en universitat-nivela instruado de matematiko: Studentoj povus ekzerci sin per skribado de pruvoj en la reguligita lingvo de Naproche, kaj la programo tiam povus aŭtomate kontroli la ĝustecon de la pruvo.

- Ĝia uzo por alproksimigi la formalan matematikon al la neformala matematiko: Pruvoj en la reguligita lingvo de Naproche povas esti konsiderataj formalaj pruvoj, ĉar ili estas aŭtomate kontroleblaj, sed samtempe ili estas facile legeblaj kaj similaj al kutimaj matematikaj tekstoj.

Por komparo de kutima matematika pruvo, formala pruvo kaj pruvo en Naproche, ni montras tri versiojn de la pruvo ke $\sqrt{2}$ estas neracia nombro.

Kiel ekzemplon de kutima matematika pruvo, mi citas el Hardy kaj Wright (1938):

> If $\sqrt{2}$ is rational, then the equation $a^2 = 2b^2$ is soluble in integers a, b with $(a, b) = 1$. Hence a^2 is even, and therefore a is even. If $a = 2c$, then $4c^2 = 2b^2$, $2c^2 = b^2$, and b is also even, contrary to the hypothesis that $(a, b) = 1$.

Jen parto de la ĉi-supra pruvo en la programlingveca formala lingvo Mizar el la Universitato de Bjalistoko (Wiedijk, 2006):

```
theorem
  sqrt 2 is irrational
proof
 assume sqrt 2 is rational;
 then consider i being Integer, n being Nat such that
W1: n<>0 and
W2: sqrt 2=i/n and
W3: for i1 being Integer, n1 being Nat st n1<>0
 & sqrt 2=i1/n1 holds n<=n1 by RAT_1:25;
A5: i=sqrt 2*n by W1,XCMPLX_1:88,W2;
C: sqrt 2>=0 & n>0 by W1,NAT_1:19,SQUARE_1:93;
 then i>=0 by A5,REAL_2:121;
 then reconsider m = i as Nat by INT_1:16;
```

Kaj fine, la sama pruvo en la reguligita lingvo de Naproche:

> Theorem. $\sqrt{2}$ is irrational.
> Proof.

> Assume that $\sqrt{2}$ is rational. Then there are integers a, b such that $a^2 = 2 \cdot b^2$ and $gcd(a, b) = 1$. Hence a^2 is even, and therefore a is even. So there is an integer c such that $a = 2 \cdot c$. Then $4 \cdot c^2 = 2 \cdot b^2$, $2 \cdot c^2 = b^2$, and b is even. Contradiction. Qed.

2.1 Malfaciloj ĉe logikigo de natura lingvo

La tradukado de natura lingvo al logikaj formuloj alportas diversajn malfacilaĵojn. Unu en la fakliteraturo de la ligvoscienca logiko tre fama ekzemplo de tia malfacilaĵo estas la tiel nomataj *azeno-frazoj*:

Iu kamparano posedas azenon.	$\rightsquigarrow$	$\exists x\ \exists y\ (kamparano(x) \wedge azeno(y) \wedge posedas(x, y))$
Se iu kamparano posedas azenon, li batas ĝin.	$\rightsquigarrow$	$\forall x\ \ \forall y\ \ ((kamparano(x) \wedge azeno(y) \wedge posedas(x, y)) \rightarrow batas(x, y))$

Nedifinitaj substantivoj (en Esperanto esprimitaj per antaŭmeto de "iu" aŭ simple per forlaso de la difina artikolo) kutime estas tradukataj helpe de ekzisto-kvantigilo ($\exists$), kiel en la unua ekzemplo. Tamen en la dua ekzemplo ili devas esti tradukataj per universalaj kvantigiloj ($\forall$). Do kvankam la unua frazo estas subfrazo de la dua frazo, en la logika formulo kiu reprezentas la enhavon de la dua frazo, oni ne trovas subformulon kiu reprezentas la enhavon de la unua formulo. Tio alportas malfacilaĵojn al la aŭtomata tradukado de natura lingvo al logikaj formuloj.

Por solvi ĉi tiajn problemojn, la nederlanda lingvisto Hans Kamp disvolvis la *Diskurs-Reprezentan Teorion*, en kiu oni tradukas tekstojn unue en *Diskurs-Reprezentajn Strukturojn*, kiujn oni poste povas plutraduki en kutimajn logikajn formulojn (Kamp kaj Reyle, 1993). Diskurs-reprezentaj strukturoj estas logikaj strukturoj kiuj havas precize la saman esprimpovon kiel kutimaj predikat-logikaj formuloj kaj estas facile intertradukeblaj kun logikaj formuloj, sed reprezentas la logikan enhavon en maniero pli simila al la natura lingvo, tiel ke ne estiĝas tiaj problemoj kiel kun la azeno-frazoj.

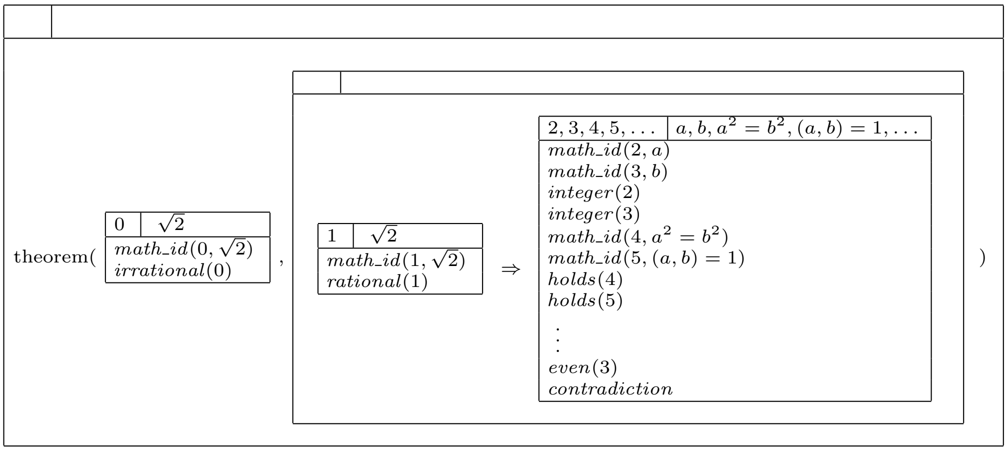
Oni kutime bildigas diskurs-reprezentajn strukturojn per kesta strukturo. Jen la diskurs-reprezenta strukturo de la dua azeno-frazo:

$$\boxed{\begin{array}{c} \\ \hline \boxed{\begin{array}{l} x, y \\ \hline \text{kamparano}(x) \\ \text{azeno}(y) \\ \text{posedas}(x, y) \end{array}} \Rightarrow \boxed{\begin{array}{l} \\ \hline \text{batas}(x, y) \end{array}} \end{array}}$$

La kesto maldekstre de la signo $\Rightarrow$ estas la diskurs-reprezenta strukturo de la unua azeno-frazo. Do ĉe la diskurs-reprezentaj strukturoj ni ja havas la dezirindan econ, ke logika traduko de subfrazo estas sama kiel se tiu subfrazo estas memstara frazo.

2.2 Pruv-reprezentaj strukturoj

Por bone redoni la specialaĵojn de la matematika faklingvo, ni kreis apartan variaĵon de la diskurs-reprezentaj strukturoj, kiujn ni nomas *Pruv-Reprezentaj Strukturoj*. Jen kelkaj el la specialaĵoj de pruv-reprezentaj strukturoj:



- Ili bone reprezentas la strukturajn ecojn de matematika teksto, do ĝian dividon en aksiomoj, difinoj, teoremoj, pruvoj ktp, kaj la hierarĥian strukturiĝon de la pruvoj per enkonduko kaj retiro de supozoj.

- Ili ebligas distingon inter variantoj enkondukitaj malimplicite en la matematika teksto (kiel a, b kaj c en la pruvo de la neracieco de $\sqrt{2}$), kaj variantoj enkondukitaj implicite en la traduko de natura lingvo al logiko (kiel x kaj y en la logika traduko de la azeno-frazoj).

- Ili havas apartajn markilojn por resendoj al antaŭaj rezultoj uzataj kiel pravigoj por pruvpaŝoj.

Supre de la paĝo troveblas ekzemplo de pruv-reprezenta strukturo, nome mallongigita versio de la pruv-reprezenta strukturo de la pruvo ke $\sqrt{2}$ estas neracia nombro.

2.3 La logika kontrolado

Por kontroli la logikan senerarecon de pruvoj, ni uzas *aŭtomatajn teorem-pruvilojn*. Aŭtomata teorem-pruvilo estas komputila programo, al kiu oni povas doni logikan problemon en la formo de listo de formuloj uzotaj kiel aksiomoj kaj unu formulo nomata *konjekto*, kiun la sistemo provu pruvi surbaze de tiuj aksiomoj. Estas tri eblaj eligoj ĉe aŭtomata teorem-pruvilo:

- *Pruvita*: Tio signifas ke la programo sukcesis pruvi ke la konjekto logike sekvas el la aksiomo.

- *Kontraŭekzemplo trovita*: Tio signifas, ke la programo sukcesis trovi kontraŭ-ekzemplon, kiu montras ke la konjekto ne sekvas logike el la aksiomoj.

- *Eltempiĝo*: Tio signifas, ke la teorem-pruvilo trovis nek pruvon nek kontraŭekzemplon en la tempo, kiun oni donis al ĝi por la serĉado.

Post la kreo de pruv-reprezenta strukuro surbaze de matematika teksto en nia reguligita lingvo, tiu strukturo estas transdonita al la logik-kontrolilo por kontrolado de ĝia logika senerareco. La logik-kontrolilo trairas la pruv-reprezentan strukturon paŝon post paŝo, kaj ĉe ĉiu paŝo tenas en la memoro liston de logikaj formuloj, nomataj *antaŭkondiĉoj*, kiuj reprezentas la ĝis tiam kolektitan matematikan scion. Ĉe ĉiu parto de pruv-reprezenta strukturo, kiu reprezentas novan aserton en pruvo, la logik-kontrolilo sendas al aŭtomata teorem-pruvilo logikan problemon, kies konjekto estas tiu aserto kaj kies aksimoj estas la ĝis tiam kolektitaj antaŭkondiĉoj. La tuta pruvo estas rekonata kiel senerara se ĉe ĉiuj problemoj tiel senditaj al teorem-pruvilo, ĝia eligo estis "pruvita".

3 Apartaj problemoj

3.1 Analizado de matematikaj formuloj

La lingvo de matematikaj formuloj mem estas nekredeble riĉa lingvo, daŭre riĉigata pere de difinoj. Oni devas klare distingi la kutiman lingvon de matematikaj formuloj, kiu estas neformala lingvo, kaj la formulo-lingvon de difinita logika sistemo, kiu ĉiam estas formala lingvo. La sintaksa analizado de matematikaj formuloj estas tre malfacila tasko.

Konsideru ekzemple la simplan formulan esprimon $a(x+y)$. Se a estas nomo de iu funkcio, tiam oni devas analizi tiun formulon kiel la valoron de tiu funkcio ĉe $x + y$. Se a same kiel x kaj y estas nombro, tiam oni devas analizi tiun formulon kiel produton de a kaj $x+y$. Do la sintaksa analizado de matematikaj formuloj dependas de informoj, kiuj estis donitaj antaŭe en la teksto, eble per naturlingva esprimo. Kontentiga solvo al ĉi tiaj problemoj estas la plej nova rezulto de mia laboro ĉe Naproche, kies realigo en la programo ankoraŭ ne estas finita.

3.2 Kunigaj kaj disigaj interpretoj de multenombro

Substantivoj en multenombro povas havi kunigan aŭ disigan interpreton: Ekzemple, la frazo "Tri viroj portis pianon" povas aŭ signifi ke tri viroj kune portis pianon (en unusola porto-ago), aŭ ke estis tri apartaj porto-agoj, ĉiu kun alia viro portanta pianon. La unua nomiĝas *kuniga interpreto* kaj la dua *disiga interpreto* de la multenombro.

En matematiko ekzistas kaj kunige kaj disige interpretendaj uzoj de la multenombro: Ekzemple, la frazo "La linioj L kaj M estas paralelaj" estas interpretenda kunige, dum la frazo "2 kaj 3 estas primoj" estas interpretenda disige. Tamen en matematiko apenaŭ aperas uzoj de multenombro kiuj estas kaj kunige kaj disige interpreteblaj kaj tial dusencaj. Mi disvolvis algoritmon por Naproche, kiu decidas, ĉu iu uzo de multenombro estas interpretenda kunige aŭ disige (Cramer kaj Schröder, 2011). La rezulto de ĉi tiu algoritmo preskaŭ ĉiam koincidas kun la natura interpreto de la multenombro.

4 Rezultoj

Ni tradukis diversajn matematikajn tekstojn al nia reguligita lingvo, kaj kontrolis ilin per nia programo: Nia ĉefa testo-teksto estas la verko "Grundlagen der Analysis" ("Fundamentoj de la analizo"), de kiu ni tradukis kaj sukcese kontroligis ĝis nun unu kaj duonan ĉapitrojn.

Ni ankaŭ elprovis nian sistemon ĉe la klasika verko "Elementoj" de Eŭklido. Liaj pruvoj ofte dependas de geometria intuicio. Por havi logike senmankajn pruvojn, oni devas aldoni multajn aksiomojn. Sed tiam, pro la multeco de la aksiomoj, la aŭtomataj teorem-pruviloj en multaj okazoj havas malfacilaĵon trovi solvojn al la problemoj senditaj al ili. Tial la logika kontrolado de la Elementoj per nia programo ankoraŭ funkcias nur por etaj fragmentoj de ĝi.

Aldone ni havas diversajn mallongajn testo-tekstojn, ekzemple pri aroteorio kaj grupteorio.

5 Konkludo

La matematika faklingvo havas interesajn ecojn, kiuj apartigas la komputlingvosciencan analizadon de matematikaj tekstoj disde aliaj uzoj de la komputa lingvoscienco. Kadre de la projekto Naproche ni evoluigis (kaj pluevoluigas) reguligitan version de la matematika faklingvo (angla). Nia programo Naproche povas analizi tekstojn en tiu reguligita lingvo per komputlingvosciencaj metodoj kaj kontroli ilian logikan senerarecon helpe de aŭtomataj teorem-pruviloj.

Citaĵoj

Marcos Cramer, Bernhard Fisseni, Peter Koepke, Daniel Kühlwein, Bernhard Schröder kaj Jip Veldman. *The Naproche Project – Controlled Natural Language Proof Checking of Mathematical Texts.* 2010. LNAI 5972, prelegkolekto de la seminario CNL 2009.

Marcos Cramer kaj Bernhard Schröder. *Interpreting Plurals in the Naproche CNL.* 2011. Publikigota en prelegkolekto de la seminario CNL 2010.

Godrey Hardy kaj Edward Wright. *An Introduction to the Theory of Numbers.* Oxford University Press, 1938.

Hans Kamp kaj Uwe Reyle. *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language.* Kluwer Academic Publisher, 1993.

Daniel Kühlwein, Marcos Cramer, Peter Koepke kaj Bernhard Schröder. *The Naproche System.* 2009. Nur rete publikigita kontribuo al la konferenco Calculemus 2009.

Ulrich Matthias. *Fundamentoj de lineara algebro.* 1995.

Freek Wiedijk. *The Seventeen Provers of the World.* Springer, 2006.