UNIVERZA V LJUBLJANI
FAKULTETA ZA MATEMATIKO IN FIZIKO
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Računalništvo in matematika – 2. stopnja

Žiga Putrle

NEPROTISLOVNOST KLASIČNE SINTETIČNE TEORIJE IZRAČUNLJIVOSTI

Magistrsko delo

Mentor: prof. dr. Andrej Bauer

Ljubljana, 2023

Zahvala

Zahvaljujem se mentorju prof. dr. Andreju Bauerju za predano znanje, strokovno pomoč in potrpežljivost brez katere moje delo ne bi bilo mogoče. Zahvaljujem se Sari in Streli za neomajno podporo, potrpežljivost, odrekanja in razumevanje, ko sem marsikatero popoldne preživel za knjigo. Zahvaljujem se družini za podporo tekom študija ter prijateljem za izmenjane ideje in za družabne večere, ki so študij ob delu naredili znosnejši.

Kazalo

Program dela

V magistrskem delu predstavite sintetično izračunljivost v stilu Yannicka Forsterja, vključno z osnovami teorije tipov in računa induktivnih kostrukcij. Podajte tudi semantični model sintentične izračunljvosti, ki temelji na teoriji realizabilnosti. Ali znate odgovoriti na vprašanje o neprotislovnosti pristopa Yannicka Forsterja?

prof. dr. Andrej Bauer

Osnovna literatura

[1] A. Bauer, *Notes on realizability*, [4. 12. 2023], dostopno na `https://www.andrej.com/zapiski/MGS-2022/notes-on-realizability.pdf`

[5] Y. Forster, *Computability in constructive type theory* (2021)

Podpis mentorja:

Neprotislovnost klasične sintetične teorije izračunljivosti

Povzetek

Yannick Forster v svojem doktorskem delu Computability in constructive type theory [5] predlaga nov sintetični pristop k teoriji izračunljivosti, ki omogoča razvoj teorije na klasičen način s pomočjo dokazovalnega pomočnika. Teorijo izračunljivosti razvije v računu induktivnih konstrukcij (RIK) (ang. calculus of inductive constructions), ki je osnova dokazovalnega pomočnika Coq, in predpostavi, da lahko v RIK-u uporabimo sintetično Churchevo tezo (SCT), zakon izključene sredine (ZIS) in pravila velike uporabe (PVU), ne da bi s tem uvedli protislovje. Predpostavko zagovarja s sklicevanjem na podobne že dokazane rezultate [5, poglavje 7], vendar dokaza neprotislovnosti ne poda. Naš prispevek k Forsterjevemu delu je dokaz, da je sintetična teorija izračunljivosti, ki jo razvije v RIK-u + SCT + ZIS + PVU, neprotislovna.

Osredotočimo se na del RIK-a, ki ga Forster uporablja pri razvoju teorije, in ga imenujemo fRIK. Podamo dva dokaza neprotislovnosti fRIK-a + SCT + ZIS + PVU. V prvem dokazu sodbe sistema fRIK + SCT + ZIS + PVU sintaktično preslikamo v sodbe sistema fRIK + SCT + PVU, kjer vse logične izjave nadomestimo z njihovo stabilno obliko in se tako izognemo aksiomu ZIS. Nato zgradimo model fRIK + SCT + PVU v teoriji realizabilnosti. V drugem dokazu zgradimo model fRIK + SCT + ZIS + PVU direktno. V modelih odvisne tipe interpretiramo kot družine skupkov, svet logičnih izjav $\mathbb{P}$ kot skupek $\nabla\mathrm{Per}(\mathbb{K}_1)$, kjer je $\mathrm{Per}(\mathbb{K}_1)$ kategorija delnih ekvivalenčnih relacij nad Kleenejevo prvo algebro $\mathbb{K}_1$, in svet stabilnih logičnih izjav $\mathbb{P}_{\neg\neg}$ kot skupek $\nabla 2$, kjer je 2 množica skupkov $\mathbb{0}$ in $\mathbb{1}$.

Consistency of classical synthetic computability theory

Abstract

In his doctoral thesis Computability in constructive type theory [5], Yannick Forster suggests a new synthetic approach towards computability theory that allows development of the theory in a classical way by using proof assistants. He develops the theory within calculus of inductive constructions (CIC), the underlying type system of Coq proof assistant. He assumes that in CIC we can use synthetic Church's thesis (SCT), law of excluded middle (LEM) and rules of large elimination (RLE) without making the theory inconsistent. He justifies his assumption by referencing similar proven results [5, chapter 7], but he does not provide a proof. Our contribution to Forster's work is a proof that synthetic computability theory developed in CIC + SCT + LEM + RLE is consistent.

We focus on a fragment of CIC that Forster uses to develop the theory and name it fCIC. We prove the consistency of fCIC + SCT + LEM + RLE in two ways. First, we syntactically map judgments from fCIC + SCT + LEM + RLE to fCIC + SCT + RLE, where we replace logical propositions with their stable form which allows us to avoid ZIS. Then we build a model of fCIC + SCT + RLE in the theory of realizability. In the second proof, we build the model of fRIK + SCT + ZIS + RLE directly. We model dependent types as families of assemblies, universe of propositions $\mathbb{P}$ as assembly $\nabla\mathrm{Per}(\mathbb{K}_1)$, where $\mathrm{Per}(\mathbb{K}_1)$ is a category of partial equivalence relations over Kleene's first algebra $\mathbb{K}_1$, and the universe of stable propositions $\mathbb{P}_{\neg\neg}$ as an assembly $\nabla 2$, where 2 is a set of assemblies $\mathbb{0}$ and $\mathbb{1}$.

Math. Subj. Class. (2020): 03B38, 03D75, 03D45, 03C57, 68V05

Ključne besede: teorija tipov, teorij izračunljivosti, teorija realizabilnosti, dokazovalni pomočniki

Keywords: type theory, computability theory, realizability theory, proof assistants

1 Uvod

Večina klasičnih knjig o izračunljivosti (npr. Rogers [14]) se začne s predstavitvijo računskega modela in definicijo izračunljivih funkcij kot funkcij, ki jih lahko izračunamo v predstavljenem računskem modelu. Kmalu pa se avtorji oddaljijo od konkretnih računskih modelov s predstavitvijo Church-Turingove teze in uvedbo univerzalnega stroja Φ. To omogoči abstrakten razvoj teorije izračunljivosti, kjer se le redko omeni računske modele. Namesto dokazovanja izračunljivosti funkcij v računskem modelu se avtorji sklicujejo na Church-Turingovo tezo, ki pravi, da za vsako intuitivno izračunljivo funkcijo f obstaja tako število c, da če ga podamo univerzalnemu stroju Φ, se bo ta obnašal enako kot funkcija f. Ker Church-Turingova teza ni formalno definirana, saj ideja intuitivno izračunljive funkcije nima natančne definicije, mora bralec, če želi natančno preveriti podane rezultate, vsako uporabo Church-Turingove teze nadomestiti z dokazom, da je funkcija izračunljiva v podanem računskem modelu. Izgradnja dokazov izračunljivosti funkcij je rutinsko opravilo, ki zahteva veliko časa. To pa je eden izmed razlogov, zakaj se avtorji pogosto sklicujejo na Church-Turingovo tezo.

Če želimo teorijo izračunljivosti razviti s pomočjo dokazovalnih pomočnikov, uporaba Church-Turingove teze, kot jo poznamo v klasičnem smislu, ne pride v poštev, saj ta ni natančno definirana. Dokazovalni pomočniki po svoji izgradnji zahtevajo natančno obravnavo podane teorije, kar pa je v nasprotju z namenoma nenatančno Church-Turingovo tezo. Preostane nam, da za vsak dokaz izračunljivosti funkcije podamo program, ki jo izračuna v izbranem računskem modelu. To zahteva veliko dodatnega truda, zato je razvoj teorije izračunljivosti z uporabo dokazovalnih pomočnikov zahtevnejši in se razlikuje od klasičnega pristopa, ki ga srečamo v knjigah. Posledica tega je, da se večina poskusov razvoja teorije izračunljivosti s pomočjo dokazovalnih pomočnikov konča pri dokazu Riceovega izreka.

Richman v članku Church's thesis without tears [12] predstavi sintetično obliko Churcheve teze po imenu CPF, ki se ne sklicuje na model izračunljivosti, vendar samo predpostavi obstoj funkcije Φ, ki oštevilči vse delne funkcije $\mathbb{N} \rightharpoonup \mathbb{N}$. Za razliko od Rogersa [14], ki teorijo izračunljivosti razvije v klasični teoriji množic, Richman teorijo konstruktivne analize razvije v Bishopovi konstruktivni matematiki, kjer so vse funkcije, ki jih lahko definiramo (brez uporabe predpostavk), izračunljive. Izračunljivost je zato vseprisotna v sistemu, v katerem razvije teorijo. To pa ne drži za delo Rogersa, saj v klasični teoriji množic funkcije, ki jih lahko definiramo, niso nujno izračunljive. Ključna razlika med Richmanovo obliko Churcheve teze in klasično Church-Turingovo tezo je, da je CPF univerzalna za vse funkcije v Bishopovi konstruktivni matematiki, medtem ko je klasična Church-Turingova teza univerzalna za vse intuitivno izračunljive funkcije. Posledično lahko CPF v Bishopovi konstruktivni matematiki uporabimo na kateri koli funkciji, ki jo lahko definiramo, ne da bi preverili njeno izračunljivost, saj vemo, da je ta izračunljiva. S tem se izognemo ideji intuitivne izračunljivosti, prisotne v klasični Church-Turingovi tezi, in preverjanju izračunljivosti, ki ga moramo opraviti z njeno uporabo v klasični teoriji množic. Uporaba formalnega sistema, v katerem so vse funkcije, ki jih lahko definiramo, izračunljive (npr. Bishopova konstruktivna matematika), in sintetično obliko Churcheve teze, ki se ne sklicuje na konkretni računski model, nam omogoči

razvoj teorije izračunljivosti brez dodatnega dela, na katerega so naleteli dosedanji pristopi k formalizaciji izračunljivosti z dokazovalnimi pomočniki.

Richman pri svojem delu predpostavi vsaj aksiom števne izbire (ang. axiom of countable choice), kot to velja za kasnejše delo Richmana, Bridgesa [4] in Bauerja [2]. Pri razvoju teorije izračunljivosti je aksiom števne izbire ključen, saj lahko z njegovo uporabo dokažemo mnoge rezultate, pri katerih se običajno uporabi izrek S_n^m [5, uvod v poglavje 2]. Zaradi uporabe aksioma števne izbire, sintetične oblike Churcheve teze in razvoja teorije v Bishopovi konstruktivni logiki se morajo avtorji odreči zakonu izključene sredine, saj njegova uporaba pripelje do protislovja (lema 3.5). Posledično je delo Richmana, Bridgesa in Bauerja konstruktivno in se s tem oddalji od običajnega pristopa k teoriji izračunljivosti, kjer je uporaba zakona izključene sredine dovoljena.

Yannick Forster v svojem doktorskem delu [5] predlaga nov sintetični pristop k teoriji izračunljivosti, ki se izogne dodatnemu delu in hkrati omogoči razvoj teorije na klasičen način. Forster teorijo izračunljivosti razvije v računu induktivnih konstrukcij (RIK) (ang. calculus of inductive constructions) – teorija tipov, na kateri je zasnovan dokazovalni pomočnik Coq, ter uporabi zakon izključene sredine in prilagojeno obliko sintetične Churcheve teze. Pri tem prilagojena sintetična Churcheva teza nadomesti sklicevanje na klasično Church-Turingovo tezo in uporabo izreka S_n^m. Forster se zaradi uporabe zakona izključene sredine odreče uporabi aksioma enolične izbire, zato ne velja, da vsaka funkcijska relacija določa funkcijo, kot je to običajno v klasični teoriji množic. Pomemben uvid Forsterja je, da v teoriji izračunljivosti redko definiramo funkcijo kot funkcijsko relacijo, zato se lahko brez večjih težav odrečemo aksiomu enolične izbire. Uporaba zakona izključene sredine in prilagojene oblike sintetične Churcheve teze omogoči Forsterju razvoj teorije izračunjivosti na klasičen način z uporabo dokazovalnega pomočnika Coq. V svojem delu Forster demonstrira uporabnost svojega pristopa z dokazom neodločljivosti sintetične oblike problema zaustavitve (ang. halting problem) in dokazom Riceovega izreka. Poleg tega poda sintetično rešitev za Postov problem za redukcijo več v ena (ang. many-one reduciton) in redukcijo z resničnostnimi tabelami (ang. truth-table reduction). Podani dokazi so pregledni, saj se ne zanašajo na konkretni računski model.

Forster v svojem delu predpostavi, da lahko v RIK-u uporabimo sintetično Churchevo tezo, zakon izključene sredine in pravila velike uporabe, ne da bi s tem uvedli protislovje. Predpostavko zagovarja s sklicevanjem na podobne že dokazane rezultate [5, poglavje 7], vendar dokaza neprotislovnosti ne poda. Naš prispevek k Forsterjevemu delu je dokaz neprotislovnosti teorije, ki jo Forster razvije z uporabo RIK-a, v katerem predpostavi sintetično Churchevo tezo, zakon izključene sredine in pravila velike uporabe. Podamo dva dokaza neprotislovnosti, kjer del dokaza preverimo s pomočjo dokazovanega pomočnika Agda.

V poglavju 2 podamo kratek pregled dokaza. V poglavju 3 predstavimo RIK in osnovne pojme teorije tipov. V poglavju 4 predstavimo Turingove stroje in nekatere pojme iz teorije izračunljivosti. Sledi poglavje 5.1, v katerem podamo sintetično Churchevo, tezo kot jo poda Forster v svojem delu. V poglavju 6 podamo dokaz neprotislovnosti, ki se zanaša na obstoj modela fragmenta RIK-a, ki ga uporablja Forster v svojem delu, katerega interpretacijo v teoriji ralizabilnosti podamo v poglavju 9. V poglavju 7 na kratko opišemo del dokaza, ki smo ga podali v Agdi. V

poglavju 8 pa predstavimo ideje iz teorije realizabilnosti, potrebne za interpretacijo fragmenta RIK-a in njegovih razširitev.

2 Pregled dokaza

V tem poglavju povzamemo dokaz in predpostavimo, da je bralec dobro seznanjen s teorijo tipov in teorijo realizabilnosti. Nekatere pojme uporabljene v tem poglavju postopoma uvedemo tekom sledečih poglavij. Podroben dokaz podamo v poglavju 6.

Forster v svojem delu razvije klasično teorijo izračunljivosti na sintetičen način v sistemu tipov RIK, v kateri predpostavi svojo obliko sintetične Churcheve teze (SCT), zakon izključene sredine (ZIS) in pravila velike uporabe (PVU). Pri tem predpostavi, da uporaba RIK-a z aksiomi SCT, ZIS in PVU ne pripelje do protislovja, vendar dokaza ne poda. V našem delu pokažemo, da je teorija, ki jo razvije Forster, neprotislovna.

Za namen dokaza se osredotočimo samo na del RIK-a, ki ga uporablja Forster v svojem delu, in ga poimenujemo fRIK; fRIK podamo v poglavju 6.1. Da pokažemo, da je sintetična teorija izračunljivosti, ki jo razvije Forster, neprotislovna, moramo dokazati, da uporaba fRIK-a z aksiomi SCT, ZIS in PVU ne pripelje do protislovja. Dokažemo trditev 6.1:

$$\text{fRIK} + \text{SCT} + \text{ZIS} + \text{PVU je neprotisloven.}$$

Trditev 6.1 dokažemo na dva načina:

- Prvi dokaz je sestavljen iz dveh delov:
  1. Sprva pokažemo, da če je fRIK + SCT + princip Markova (PM) + PVU neprotisloven, potem je tudi fRIK + SCT + ZIS + PVU neprotisloven.
  2. Nato pokažemo, da je fRIK + SCT + PM + PVU neprotisloven, saj zanj obstaja model.
- V drugem dokazu model fRIK + SCT + ZIS + PVU zgradimo direktno in s tem pokažemo, da je neprotisloven.

Zapis fRIK + SCT + PM + PVU in fRIK + SCT + ZIS + PVU okrajšamo z fRIK_{SMiP} in fRIK_{SZiP}.

Opomba 2.1. Sistem tipov je *neprotisloven*, če tip $\bot$ nima elementa v praznem kontekstu. Tip A ima element v kontekstu Γ, če lahko zgradimo sodbo $\Gamma \vdash e{:}A$ za nek element e. Sledi, da v neprotislovnem sistemu tipov ne moremo zgraditi sodbe $\vdash e{:}\bot$ za poljuben e.

Definicija 2.2. Sodba je *veljavna* v sistemu X, če jo lahko v sistemu X izpeljemo.

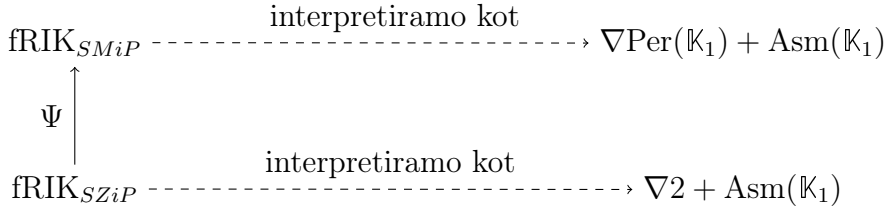
Neprotislovnost fRIK_{SZiP} sledi iz neprotislovnosti fRIK_{SMiP}, saj lahko zgradimo preslikavo Ψ, ki

- slika sodbe sistema fRIK_{SZiP} v sodbe sistema fRIK_{SMiP} in
- ohranja veljavnost sodb.

Če je fRIK_{SZiP} protisloven, obstaja sodba $\vdash e{:}\bot$, ki jo Ψ slika v sodbo $\vdash \Psi(e){:}\bot$ sistema fRIK_{SMiP}. Ker pa vemo, da fRIK_{SMiP} ni protisloven, saj zanj obstaja model, sodba $\vdash \Psi(e){:}\bot$ v fRIK_{SMiP} ne obstaja. Posledično, tudi ne obstaja sodba $\vdash e{:}\bot$ v sistemu fRIK_{SZiP}. Sledi, da je fRIK_{SZiP} neprotisloven. V sistemu fRIK_{SMiP} predpostavimo PM, saj ga potrebujemo za dokaz, da preslikava Ψ ohrani pravila PVU.

Model fRIK_{SMiP} podamo z njegovo interpretacijo v teoriji realizabilnosti, kjer za delno kombinatorno algebro vzamemo Kleenejevo prvo algebro ($\mathbb{K}_1$). Odvisne tipe interpretiramo kot družine skupkov in svet logičnih izjav $\mathbb{P}$ kot skupek $\nabla\mathrm{Per}(\mathbb{K}_1)$, kjer je $\mathrm{Per}(\mathbb{K}_1)$ kategorija delnih ekvivalenčnih relacij nad $\mathbb{K}_1$. Dokažemo, da SCT lahko realiziramo, saj zgradimo realizatorja CT_Φ in SCT_Φ, iz katerih sledi, da lahko znotraj $\mathbb{K}_1$ realiziramo tudi SCT. Pri izgradnji CT_Φ in SCT_Φ se zanašamo, da znotraj $\mathbb{K}_1$ lahko realiziramo univerzalno koračno preslikavo in da velja teorem SMN. Realizatorjev PM in PVU ne podamo eksplicitno, vendar pa je iz njihovega opisa jasno, kako jih sestaviti s pomočjo znanih konstrukcij, kot je univerzalni Turingov stroj. Podamo še interpretacijo sveta stabilnih logičnih izjav $\mathbb{P}_{\neg\neg}$ kot skupka $\nabla 2$, kjer je 2 množica skupkov $\mathbb{0}$ in $\mathbb{1}$, in s tem pridobimo interpretacijo fRIK_{SZiP}.

Povezanost posameznih delov dokaza je prikazana na sliki 1.



Slika 1: Povezanost delov dokaza

3 Račun induktivnih konstrukcij

V tem poglavju podamo kratek opis teorije tipov, računa induktivnih konstrukcij in nekaterih pojmov, ki so pomembni za naše delo. Predpostavimo, da je bralec seznanjen z osnovnimi idejami teorije tipov in njihovo uporabo. Kot osnovno literaturo za to poglavje smo uporabili uvod knjige Introduction to homotopy type theory [13], članek Introduction to the calculus of inductive constructions [10] in poglavje Aspects of CIC Forsterjevega doktorskega dela [5].

Teorija tipov je množica formalnih sistemov, s katerim lahko opišemo matematične strukture, z njimi računamo in o njih formalno sklepamo. Osnovni elementi teorije tipov so *tipi* in njihovi *elementi*. Vsak element v teoriji tipov ima tip, kar zapišemo kot *sodbo* $\Gamma \vdash a{:}A$, ki pravi, da ima v kontekstu Γ element a tip A. Tipe in njihove elemente zgradimo z uporabo *konstrukcijskih pravil*, ki so predhodno podana in predstavljajo osrednji del formalnega sistema teorije tipov. Konstrukcijsko pravilo podamo kot predpis

$$\frac{H_1, H_2, \ldots H_n}{C},$$

katerega pomen je, da če lahko zgradimo seznam hipotez $H_1, H_2, \ldots, H_n$, potem lahko zgradimo tudi zaključno sodbo C. Preprost primer konstrukcijskega pravila je

$$\frac{\Gamma \vdash a{:}A \quad \Gamma \vdash f{:}A \to B}{\Gamma \vdash f(a){:}B}.$$

Pravilo pravi, da v kontekstu Γ lahko uporabimo element $a{:}A$ in funkcijo $f{:}A \to B$, da zgradimo element $f(a){:}B$.

V teoriji tipov imamo pogosto opravka z različnimi tipi sodb. V računu induktivnih konstrukcij, ki ga uporablja Forster v svojem doktorskem delu, poznamo štiri vrste sodb:

- sodbo »$\Gamma \vdash A$ type« s pomenom »A je *dobro oblikovan tip* v kontekstu Γ«,
- sodbo »$\Gamma \vdash A \doteq B$ type« s pomenom »A in B sta *sodno enaka* (ang. judgmentally equal) *tipa* v kontekstu Γ«,
- sodbo »$\Gamma \vdash t{:}A$« s pomenom »a je *dobro formiran izraz* tipa A v kontekstu Γ« in
- sodbo »$\Gamma \vdash a \doteq b{:}A$« s pomenom »$a$ in b sta *sodno enaka izraza* tipa A v kontekstu Γ«.

Račun induktivnih konstrukcij (RIK) (ang. calculus of inductive constructions) [10] je sistem tipov, na katerem temelji dokazovalni pomočnik Coq [17]. Uporabljamo ga lahko kot programski jezik ali formalni sistem, saj podpira čisto funkcijsko programiranje brez splošne rekurzije in računskih učinkov, hkrati pa lahko v njem razvijemo logiko višjega reda. RIK združi račun konstrukcij (RK) (ang. calculus of constructions), polimorfni funkcijski jezik z močnim sistemom tipov, z idejo induktivne definicije tipov in hierarhijo svetov z ločenim impredikativnim svetom logičnih izjav $\mathbb{P}$. Induktivna definicija tipov omogoči enostavno predstavitev struktur kot so vsote (Σ), produkti (Π), naravna števila ($\mathbb{N}$) in enakost. S hierarhijo svetov pa se izognemo paradoksom Russellovega tipa. Z združitvijo RK-a, induktivne definicije tipov in hierarhije svetov dobimo jezik, primeren za razvoj matematičnih teorij s pomočjo dokazovalnih pomočnikov.

Poznamo več različic RIK-a. Forster svoje delo temelji na različici, uporabljeni za razvoj dokazovalnega pomočnika Coq verzije 8.13.2 [17].

RIK ima hierarhijo svetov (ang. hierarchy of type universes), ki jo delimo v dve skupini: *svet logičnih izjav* $\mathbb{P}$ in *računske svetove* $\mathbb{T}^1, \mathbb{T}^2, \mathbb{T}^3, \ldots$ V svetu $\mathbb{P}$ lahko po principu Curry-Howardove korespondence logične izjave predstavimo kot tipe in programe kot dokaze. Zato lahko gledamo na svet $\mathbb{P}$ kot na svet logičnih izjav, kjer je $P{:}\mathbb{P}$ izjava in $p{:}P$ njen dokaz. Podobno lahko gledamo na svetove $\mathbb{T}^i$ kot na svetove programov, kjer je $T{:}\mathbb{T}^i$ tip programa in je $t{:}T$ program tipa T. Svetovom $\mathbb{T}^i$ pravimo tudi računski svetovi. Kadar ni dvoumno, poljuben računski svet $\mathbb{T}^i$ označimo samo s $\mathbb{T}$.

V RIK-u lahko logične izjave in dokaze zgradimo le iz enostavnejših izjav in dokazov z uporabo sklepnih pravil. Pridobljeni formalni sistem je zato konstruktiven, kar pomeni, da zakon izključene sredine ne velja. Vendar ga lahko predpostavimo, ne da bi s tem uvedli protislovje.

V našem delu se zanašamo na sledeče tipe. Iz sveta $\mathbb{T}$ uporabimo

- *funkcije* $A \to B \coloneqq \lambda a.\, e$, kjer sta $a{:}A$ in $e{:}B$ za nek $A{:}\mathbb{T}$ in $B{:}\mathbb{T}$, in
- *kartezične produkte* $\Pi a{:}A.\, P(a) \coloneqq \lambda a.\, e$, kjer sta $a{:}A$ in $e{:}P(a)$ za nek $A{:}\mathbb{T}$ in družino $P{:}\, A \to \mathbb{T}$,

ter sledeče induktivno definirane tipe:

- *enojec* $\mathbb{1} \coloneqq \star$,
- *Boolove vrednosti* $\mathbb{B} \coloneqq \mathsf{true} \,|\, \mathsf{false}$,
- *naravna števila* $\mathbb{N} \coloneqq \mathsf{zero} \,|\, \mathsf{succ}\, n$, kjer $n{:}\mathbb{N}$,
- *možnosti* $\mathbb{O}\, A \coloneqq \mathsf{none} \,|\, \mathsf{some}\, x$, kjer $x{:}A$,
- *pare* $A \times B \coloneqq (x, y)$, kjer $x{:}A$ in $y{:}B$,
- *vsote* $A + B \coloneqq \mathsf{inl}\, x \,|\, \mathsf{inr}\, y$, kjer $x{:}A$ in $y{:}B$,
- *sezname* $\mathbb{L}A \coloneqq [\,] \,|\, a :: l$ kjer $a{:}A$ in $l{:}\mathbb{L}A$,
- *odvisne vsote* $\Sigma x{:}A.\, P(x) \coloneqq (x, y)$, kjer $x{:}A$ in $y{:}P(x)$ za nek $P{:}\, A \to \mathbb{T}$, in
- *enakosti* $a =_A x \;\coloneqq \mathsf{refl}\; a$, kjer $a{:}A$ in $x{:}A$.

Iz sveta $\mathbb{P}$ uporabimo

- *resnico* $\top$,
- *neresnico* $\bot$,
- *in* $A \wedge B$, kjer $A{:}\mathbb{P}$ in $B{:}\mathbb{P}$,
- *ali* $A \vee B$, kjer $A{:}\mathbb{P}$ in $B{:}\mathbb{P}$,
- *implikacijo* $A \Rightarrow B$, kjer $A{:}\mathbb{P}$ in $B{:}\mathbb{P}$,
- *obstaja* $\exists x{:}A.P(x)$, kjer $A{:}\mathbb{T}$ in $P{:}\, A \to \mathbb{P}$, in
- *za vse* $\forall x{:}A.P(x)$, kjer $A{:}\mathbb{T}$ in $P{:}\, A \to \mathbb{P}$.

Za naštete tipe veljajo klasična pravila vpeljave in uporabe. Tip $\Pi x{:}A.\, P(x)$ pogosto zapišemo kot $\Pi_{x:A} P(x)$ in okrajšamo kot $\Pi x.P(x)$, kjer tipa spremenljivke ne podamo, če je ta razviden iz konteksta. Enako velja za tipe $\Sigma, \forall$ in $\exists$.

Sistem tipov je *protisloven*, če lahko v praznem kontekstu dokažemo, da obstaja element $\bot$. Posledica neprotislovnosti sistema tipov je, da v takšnem sistemu lahko zgradimo element kateregakoli tipa. Takšen sistem postane neuporaben kot formalni sistem, saj v njem lahko dokažemo katerokoli izjavo.

Definicija 3.1. Sistem tipov je *neprotisloven*, če tip $\bot$ nima elementov v praznem kontekstu.

Posledica 3.2. *V neprotislovnem sistemu tipov ne moremo zgraditi sodbe* $\vdash e{:}A$.

3.1 Svet izjav

Svet logičnih izjav $\mathbb{P}$ je impredikativen svet tipov, ki zaobjame vse izraze, ki predstavljajo logične izjave.

Definicija 3.3. *Svet logičnih izjav* $\mathbb{P}$ definiramo rekurzivno s praviloma

$$\frac{\Gamma \vdash}{\Gamma \vdash \mathbb{P}{:}\mathbb{T}^1} \text{ in} \tag{3.1}$$

$$\frac{\Gamma, x{:}A \vdash B{:}\mathbb{P}}{\Gamma \vdash \Pi\, x{:}A.\, B{:}\mathbb{P}}, \tag{3.2}$$

kjer je Γ poljuben kontekst in $A{:}\mathbb{P}$ ali $A{:}\mathbb{T}$. S prvim pravilom (3.1) določimo, da je $\mathbb{P}$ tipa $\mathbb{T}^1$, in z drugim pravilom (3.2), da če v kontekstu $\Gamma, x{:}A$ velja B tipa $\mathbb{P}$, potem v Γ velja $\Pi x{:}A.\, B$ tipa $\mathbb{P}$. Svetu $\mathbb{P}$ pogosto rečemo tudi Prop.

Svet $\mathbb{P}$ zaobjame vse izraze, ki predstavljajo logične izjave, saj lahko z uporabo Π zgradimo vse logične operatorje $\top$, $\bot$, $\wedge$, $\vee$, $\Rightarrow$, $\exists$ in $\forall$ [15, uvod]. Iz definicije sveta $\mathbb{P}$ sledi, da je svet $\mathbb{P}$ impredikativen in da je zaprt za konstruktor Π.

Opomba 3.4. Svet $\mathbb{P}$ je *impredikativen*, saj lahko zgradimo nove tipe sveta $\mathbb{P}$ z uporabo konstruktorja Π, ki teče po vseh tipih.

Svet $\mathbb{P}$ in svetovi $\mathbb{T}$ so v RIK-u ločeni, saj znotraj RIK-a nimamo pravil, ki splošno omogočijo, da dokaz izjave $P{:}\mathbb{P}$ preslikamo v program tipa $X{:}\mathbb{T}$. Posledično v RIK-u ne moremo izpeljati aksioma izbire ali aksioma enolične izbire. Zato postaneta funkcijska relacija in funkcija ločena pojma, kar pa se razlikuje od teorije množic, kjer funkcije definiramo kot funkcijske relacije. Čeprav funkcijske relacije lahko enačimo s funkcijami na meta nivoju, ta povezava ni na voljo v sami teoriji. Povezavo lahko uvedemo v teorijo kot aksiom ter s tem združimo svet $\mathbb{P}$ in preostale svetove $\mathbb{T}$, npr. uvedemo lahko aksiom enolične izbire, prilagojen za teorijo tipov,

$$\forall R{:}\, X \to Y \to \mathbb{P}.\, (\forall x.\, \exists! y.\, R\, x\, y) \to \exists f{:}\, X \to Y.\, R\, x\, (f\, x),$$

ki za vsako funkcijsko relacijo vrne funkcijo

Ločenost svetov je ključna razlika med RIK-om in preostalimi sistemi tipov, kot sta Martin-Löfov sistem tipov in homotopska teorija tipov (ang. homotopy type theory), ki je pomembna za Forsterjevo delo, saj se z njo izognemo aksiomu enolične izbire in s tem protislovju, na katerega naletimo, če predpostavimo Churchevo tezo, aksiom enolične izbire in zakon izključene sredine (lema 3.5).

Lema 3.5. *Bishopova konstruktivna logika z aksiomom enolične izbire, zakonom izključene sredine in Churchevo tezo vodi v protislovje.*

Dokaz. Definirajmo relacijo $D \subseteq \mathbb{N} \times \{0, 1\}$ s predpisom:

$$\begin{aligned} &D(n, b) \iff \\ &\quad (n\text{-ti Turingov stroj se ustavi in } b = 1.) \vee \\ &\quad (n\text{-ti Turingov stroj se ne ustavi in } b = 0.) \end{aligned}$$

Ta relacija je enolična saj velja

$$\forall n \in \mathbb{N}.\, \forall b, b' \in \{0,1\}.\, D(n,b) \wedge D(n,b') \Rightarrow b = b'.$$

Po zakonu izključene sredine za vsak n velja $D(n,0) \vee D(n,1)$:

- če se n-ti stroj ustavi, velja $D(n,1)$ in
- če se n-ti stroj ne ustavi, velja $D(n,0)$.

Sledi, da je D funkcijska relacija. Po aksiomu enolične izbire obstaja preslikava $h\colon \mathbb{N} \to \{0,1\}$ za katero velja $\forall n \in \mathbb{N}.\, D(n,h(n))$. Preslikava h odloči ali se bo Turingov stroj ustavil in po Churchevi tezi je ta izračunljiva. To pa je v nasprotju z dobro znanim dejstvom, da ne obstaja algoritem, ki odloči ali se bo stroj ustavil. □

3.2 Odvisni pari in eksistenčni kvantifikatorji

Zaradi ločenosti svetov $\mathbb{P}$ in $\mathbb{T}$ v RIK-u ločimo med eksistenčnimi kvantifikatorji $\exists$ in odvisnimi pari Σ. Za razliko od Martin-Löfove teorije tipov, kjer $\exists$ definiramo kot Σ, sta $\exists$ in Σ v RIK-u različna koncepta. Eksistenčni kvantifikator $\exists x$ beremo kot »obstaja x«, medtem ko odvisni par Σx beremo kot »zgradimo x«. Izrazi oblike $\exists x.\, P(x)$ tako predstavljajo logične izjave sveta $\mathbb{P}$, izrazi oblike $\Sigma x.\, B(x)$ pa predstavljajo odvisne pare sveta $\mathbb{T}$.

Odvisne pare lahko slikamo v arbitrarni tip, kar pomeni, da lahko zgradimo funkcijo

$$\Pi P\colon (\Sigma x{:}X.\, A(x)) \to \mathbb{T}.\, (\Pi x{:}X.\, \Pi y{:}A(x).\, P(x,y)) \to \Pi s.\, P(s).$$

Enako pa ne velja za $\exists$, saj podobne funkcije za $P{:}(\exists x.\, A(x)) \to \mathbb{T}$ ne moremo zgraditi [5, poglavje 3.2]. Posledično v RIK-u ne moremo definirati funkcije

$$\Pi P\colon Y \to \mathbb{P}.\, (\exists y.\, P(y)) \to \Sigma y.\, P(y),$$

ki za poljuben predikat P iz eksistenčnega dokaza $\exists y.\, P(y)$ izlušči pričo y in dokaz, da zanjo velja $P(y)$.

3.3 Pravila velike uporabe

Pravila velike uporabe (ang. large elimination rules) omogočajo uporabo elementov tipa $\mathbb{P}$ za izgradnjo elementov tipa $\mathbb{T}$. RIK večinoma ne dovoli pravil velike uporabe, saj lahko z njimi združimo svet izjav $\mathbb{P}$ in računske svetove $\mathbb{T}$. Izjemoma pa lahko uporabimo sledeča pravila, ki so neškodljiva [8, poglavje 2.3]:

- *pravilo velike uporabe* $\bot$

$$\Pi A{:}\mathbb{T}.\, \bot \to A, \qquad (\mathrm{VU}\bot)$$

- *pravilo velike uporabe enakosti*

$$\Pi X{:}\mathbb{T}.\, \Pi A\colon X \to \mathbb{T}.\, \Pi x_1 x_2{:}X.\, x_1 =_X x_2 \to A(x_1) \to A(x_2) \text{ in} \qquad (\mathrm{VU}{=})$$

- *pravilo velike uporabe eksistenčnih kvantifikatorjev odločljivih predikatov na naravnih številih*,

$$\Pi f\colon \mathbb{N} \to \mathbb{B}.\, (\exists m{:}\mathbb{N}.\, f(m) =_{\mathbb{B}} \mathsf{true}) \to \Sigma m{:}\mathbb{N}.\, f(m) =_{\mathbb{B}} \mathsf{true}. \qquad (\mathrm{VU}\exists\mathrm{P}\mathbb{N})$$

Pravila VU$\bot$, VU$=$ in VU$\exists$P$\mathbb{N}$ skupaj imenujemo *pravila velike uporabe* (PVU).

3.4 Enakost in tip identifikacij

V teoriji tipov enakost elementov $a{:}A$ in $b{:}A$ predstavimo kot tip $a =_A b$, katerega elementi so dokazi, da sta a in b enaka elementa. Definiramo ga kot družino tipov $a =_A x$, indeksirano z elementom $x{:}A$, in ga imenujemo *tip identifikacij* (ang. identity type), saj elementi $a =_A b$ identificirajo enakost elementov a in b. Tip identifikacij podamo s sledečimi pravili: tip zgradimo z uporabo pravila

$$\frac{\Gamma \vdash a{:}A}{\Gamma, x{:}A \vdash a =_A x},$$

njegov element zgradimo s pravilom vpeljave

$$\frac{\Gamma \vdash a{:}A}{\Gamma \vdash \mathsf{refl}_a \colon a =_A a},$$

uporabimo pa ga s pravilom uporabe

$$\frac{\Gamma \vdash a{:}A \quad \Gamma, x{:}A, p\colon a =_A x \vdash P(x,p)\ \text{type}}{\Gamma \vdash \mathsf{ind\text{-}eq}_a \colon P(a, \mathsf{refl}_a) \to \Pi_{(x:A)}\Pi_{(p:a=_A x)}P(x,p)},$$

ki mu pripada računsko pravilo

$$\frac{\Gamma \vdash a{:}A \quad \Gamma, x{:}A, p\colon a =_A x \vdash P(x,p)\ \text{type}}{\Gamma, u{:}P(a,\mathsf{refl}_a) \vdash \mathsf{ind\text{-}eq}_a(u,a,\mathsf{refl}_a) \doteq u \colon P(a,\mathsf{refl}_a)}.$$

Opazimo, da je tip $=_A$ definiran induktivno in je generiran s konstruktorjem refl_a, s katerim identificiramo, da je element $a{:}A$ enak samemu sebi. Pravilo uporabe pa nam omogoči, da iz elementa tipa $P(a, \mathsf{refl}_a)$ pridobimo element tipa $P(x,p)$ za poljuben $x{:}A$ in identifikacijo $p\colon a =_A x$. Z uporabo pravila vpeljave in uporabe ter preostalih pravil teorije izračunljivosti lahko pokažemo, da tip identifikacij predstavlja ekvivalenčno relacijo. Podrobnejša predstavitev tipa identifikacij je podana v [13, poglavje 5.1].

Ali sta elementa $x{:}A$ in $y{:}A$ enaka, lahko v nekaterih primerih odločimo z uporabo preslikave

$$D_A \colon \Pi(x,y{:}A).\, x =_A y + \neg(x =_A y).$$

Če takšna preslikava obstaja, pravimo, da ima A *odločljivo enakost*. Primer tipov z odločljivo enakostjo sta $\mathbb{B}$ in $\mathbb{N}$, primer tipa z neodločljivo enakostjo pa realna števila $\mathbb{R}$.

Definicija 3.6. Za odločljivo enakost $=_A$ lahko zgradimo preslikavo

$$x =^d_A y \coloneqq (T \circ B_A \circ D_A)(x,y),$$

kjer je preslikava $B_A \colon x =_A y + \neg(x =_A y) \to \mathbb{B}$ definirana kot

$$B_A(\mathsf{inl}(e)) = \mathsf{true},\ \text{kjer } e\colon x =_A y,$$
$$B_A(\mathsf{inr}(e)) = \mathsf{false},\ \text{kjer } e\colon \neg(x =_A y),$$

preslikava $T\colon \mathbb{B} \to \mathbb{P}$ pa kot

$$T(\top) = \mathsf{true},$$
$$T(\bot) = \mathsf{false}.$$

Lema 3.7. *Za vse $x, y{:}A$ velja $x =_A y \iff x =^d_A y$.*

Dokaz. Sledi iz definicije preslikave $=^d_A$. □

Pravimo, da D_A in $=^d_A$ *odločita*, ali sta elementa tipa A enaka. Če ima A odločljivo enakost, pogosto uporabimo $=^d_A$ namesto $=_A$.

3.5 Zakon izključene sredine in princip Markova

Zakon izključene sredine in princip Markova v RIK-u ne moremo dokazati lahko pa ju predpostavimo.

Definicija 3.8. *Zakon izključene sredine* (ZIS) se glasi:

$$\forall P{:}\,\mathbb{P}.\, P \vee \neg P.$$

Definicija 3.9. *Princip Markova* (PM) se glasi: za odločljiv predikat $P{:}\,\mathbb{N} \to \mathbb{P}$ velja

$$\neg\neg\exists a{:}A.\, P(a) \to \exists a{:}A.\, P(a).$$

3.6 Svet Tarskega in Russellov svet

V teoriji tipov svetove običajno podamo na dva načina: v slogu Tarskega ali v slogu Russla.

Svet Tarskega je tip U s preslikavo El, tako da velja

$$\frac{\Gamma \vdash a : U}{\Gamma \vdash El(a)\ \mathrm{type}}.$$

Elemente U lahko vidimo kot kode tipov, preslikavo El pa kot dekodirnik, ki vsaki kodi $a{:}U$ priredi tip $El(a)$.

Russellov svet je tip U, za katerega velja

$$\frac{\Gamma \vdash a : U}{\Gamma \vdash a\ \mathrm{type}},$$

kjer so elementi $a : U$ tipi. Russellov svet je ekvivalenten svetu Tarskega, kjer je preslikava El identiteta.

Definicija 3.10. Svet Tarskega (U, El) je *zaprt za kartezični produkt*, ko lahko za vsak $a{:}U$ in $P{:}\,El(a) \to U$ konstruiramo $\Pi(a, P)$, za katerega velja $El(\Pi(a, P)) = \Pi x{:}\,El(a).\, P(x)$.

Zgornja definicija prav tako velja za Russellove svetove, saj so ti posebni primer svetov Tarskega.

4 Računski model

Računski model je matematični model, ki opiše, kaj je računanje in kako računamo. Poznamo več računskih modelov, med katerimi je najbolj znan Turingov model [18]: računanje predstavi kot delovanje abstraktnega stroja, ki upravlja s simboli zapisanimi na traku, v skladu z listo pravil. V našem delu uporabimo dva računska modela: Turingove stroje in delno kombinatorno algebro. V tem poglavju podamo neformalen in kratek opis Turingovih strojev, ko to naredi Bauer v [1, poglavje 2.1], ter predstavimo nekatere ideje iz teorije izračunljivosti. V poglavju 8.1 pa se posvetimo delni kombinatorni algebri. Podrobna definicija Turingovega modela je podana v [14, poglavje 1.5].

Definicija 4.1. *Turingov stroj* je naprava, ki izvaja operacije na trakovih in njihovih glavah. *Trakovi* so sestavljeni iz neskončnega zaporedja *celic*. V celice lahko pišemo in z njih beremo s pomočjo glave. Vsakemu traku pripada ena *glava*, ki jo lahko premikamo levo in desno po traku med celicami. Glava vedno kaže na eno celico na traku, ki je bodisi prazna bodisi vsebuje simbol iz končne abecede. Abeceda, ki jo običajno izberemo, vsebuje le simbola 0 in 1. Turingov stroj ima tri trakove: *vhodni trak*, *delovni trak* in *izhodni trak*. Z vhodnega traku lahko samo beremo, z delovnega traku lahko beremo in nanj pišemo, na izhodni trak pa lahko pišemo v vsako celico največ enkrat. Turingov stroj opravlja operacije nad trakovi in glavami glede na končno listo preprostih pravil, ki ji pravimo *program*. Turingov stroj lahko izvede sledeče operacije nad posameznim trakom, če je to dovoljeno glede na trak: zapiše simbol 0 ali 1 v celico, na katero kaže glava, izbriše simbol v celici, na katero kaže glava, ter glavo traku premakne levo ali desno. Katero operacijo stroj izvede, je odvisno od programa, vrednosti celic, na katere kažejo glave trakov, in *internega stanja*, ki ga predstavimo z naravnim številom. Turingov stroj izvaja operacije zaporedoma, eno operacijo za drugo, in lahko konča svoje izvajanje v končno mnogo korakih ali teče v neskončnost.

Poznamo različne vrste Turingovih strojev, ki so med seboj ekvivalentne v smislu, da lahko z eno vrsto stroja simuliramo drugo. Rogers v [14, poglavje 1.5] predstavi vrsto Turingovih strojev z enim trakom, s katero lahko simuliramo obnašanje našega stroja, tako da zapišemo več trakov na enega s prepletanjem.

Definicija 4.2. Turingov stroj T *izračuna delno preslikavo* $f\colon \mathbb{N} \rightharpoonup \mathbb{N}$, če za vsak $n \in \mathbb{N}$,

- kjer je $f(n)$ definiran velja: ko število n zapišemo na vhodni trak stroja T in stroj zaženemo, se stroj T ustavi v končno mnogo korakih in je na izhodnem traku zapisano število $f(n)$, in
- kjer $f(n)$ ni definiran velja: ko število n zapišemo na vhodni trak stroja T in stroj zaženemo, se stroj T ne ustavi.

Definicija 4.3. Delna preslikava $f\colon \mathbb{N} \rightharpoonup \mathbb{N}$ je *izračunljiva*, če zanjo obstaja Turingov stroj, ki jo izračuna.

Izrek 4.4 (Kleenejev izrek o normalni obliki [6]). *Vsako izračunljivo delno preslikavo* $f\colon \mathbb{N} \rightharpoonup \mathbb{N}$ *lahko zapišemo kot*

$$y \mapsto U(min\{z \in \mathbb{N} \mid T(x, y, z)\}),$$

kjer x *predstavlja kodo Turingovega stroja, ki izračuna* f*,* T *Kleenejev* T*-predikat in* U *pripadajočo funkcijo za pridobitev rezultata iz izvajalne sledi* z*. Kleene v svojem delu [6] natančno opiše preslikavi* T *in* U*, katerih pomen podamo neformalno kot* $T(x, y, z) :=$ *»Stroj, oštevilčen s kodo* x*,* z *vhodnim trakom s kodo* y *izvede zaporedje korakov izvajalne sledi, oštevilčene s kodo* z*.« in* $U(z) :=$ *»Število, zapisano na traku ob koncu zadnjega koraka v zaporedju korakov izvajalne sledi, oštevilčene s kodo* z*.«.*

Kleenejev izrek lahko uporabimo za definicijo *standardnega številčenja* (*kodiranja*) izračunljivih delnih funkcij. Definiramo

$$\varphi_x(y) = U(min\{z \in \mathbb{N} \mid T(x, y, z)\}),$$

kjer je $\varphi_0, \varphi_1, \varphi_2, ...$ zaporedje vseh izračunljivih delnih funkcij. Funkcije se znotraj zaporedja lahko večkrat ponovijo. Pravimo, da število $i \in \mathbb{N}$ predstavlja kodo Turingovega stroja, ki realizira φ_i.

Kleenejev izrek o normalni obliki lahko posplošimo na preslikave več spremenljivk. Za vsak k obstajata Kleenejev predikat $T^{(k)}(x, y_1, y_2, \ldots, y_k, z)$ in $U^{(k)}(z)$, ki vrne rezultat iz izvajalne sledi oštevilčene s kodo z. Podobno obstaja standardno številčenje izračunljivih delnih funkcij s k spremenljivkami:

$$\varphi_x^{(k)}(y_1, y_2, \ldots, y_n) = U^{(k)}(min\{z \in \mathbb{N} \mid T^{(k)}(x, y_1, y_2, \ldots, y_k, z)\}).$$

Standardno številčenje ima sledeče lastnosti.

Izrek 4.5 (UKP). *Obstaja izračunljiva preslikava, ki jo imenujemo univerzalna koračna preslikava,*

$$o\colon \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{N},$$

ki sprejme kodo Turingovega stroja x*, vhodno število* y *in število korakov* k*. Preslikava* o *poskuša izvesti* k *korakov stroja* x *in vrne* $\varphi_x(y) + 1$*, če je v* k *korakih stroj* x *prišel v končno stanje in je na izhodnem traku zapisano število* $\varphi_x(y)$*. Če stroj ni v končnem stanju, preslikava* o *vrne vrednost* 0*. Preslikava* o *je ustaljena (ang. stationary), v smislu, da velja*

$$\forall x, y, k_1, z \in \mathbb{N}.\, z > 0 \to o(x, y, k_1) = z \to \forall k_2 \in \mathbb{N}.\, k_2 \geq k_1 \to o(x, y, k_2) = z.$$

Dokaz. Preslikava o je izračunljiva, saj vemo, da obstaja univerzalni Turingov stroj (UTS) [1, Izrek 2.1.2], ki za poljubno kodo stroja x izvede $\varphi_x(y)$ in vrne rezultat. UTS lahko predelamo tako, da izvede samo k korakov in vrne primeren rezultat glede na to, ali je stroj, ki ga izvaja, dosegel končno stanje. □

Podobno kot Kleenejev izrek o normalni oblki lahko univerzalno koračno preslikavo o posplošimo na preslikave več spremenljivk z uporabo združitvenih preslikav (ang. paring functions) $\langle \_, \_, \ldots, \_ \rangle\colon \mathbb{N}^k \to \mathbb{N}$ [14, poglavje 5.3], tako da velja

$$\begin{aligned}
&\forall x, y_0, y_1, \ldots, y_k, n \in \mathbb{N}.\\
&\quad o(x, \langle y_0, y_1, \ldots, y_k \rangle, n) > 0 \to\\
&\quad o(x, \langle y_0, y_1, \ldots, y_k \rangle, n) = \varphi_x^{(k)}(y_0, y_1, \ldots, y_k) + 1.
\end{aligned}$$

Opomba 4.6. Pravimo, da sta delni preslikavi $f, g\colon \mathbb{N} \rightharpoonup \mathbb{N}$ *enaki*, kadar za vsako naravno število vrneta enak rezultat ali pa sta obe nedefinirani.

Opomba 4.7. Z enačbo $a \simeq b$ označimo, da če je definirana ena stran enačbe, potem je definirana tudi druga in med njima velja enakost.

Izrek 4.8 (SMN)**.** *Obstaja izračunljiva preslikava $q\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, za katero velja, da za vse $x, y, z \in \mathbb{N}$ velja*

$$\varphi_{q(x,y)}(z) \simeq \varphi_x^{(2)}(y, z).$$

Dokaz. Preslikava q je izračunljiva, saj obstaja Turingov stroj, ki naredi sledeče: z vhodnega traku prebere kodo stroja x in naravno število y ter vrne kodo Turingovega stroja, ki prebere število z in nato izvede $\varphi_x^{(2)}(y, z)$. □

Izrek SMN lahko predstavimo tudi na sledeči način.

Lema 4.9 (SMN_o)**.** *Obstaja izračunljiva preslikava $q\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, za katero velja, da za vse $x, y, z \in \mathbb{N}$ velja*

$$\exists k \in \mathbb{N}.\, o(q(x, y), z, k) = z \leftrightarrow \exists k \in \mathbb{N}. o(x, \langle y, z\rangle, k) = z.$$

Dokaz. Uporabimo enak q kot pri 4.8. □

Lema 4.10. *Obstaja koda n_o, za katero velja, za vse $x, y, k \in \mathbb{N}$*

$$\varphi_{n_o}(x, y, k) = o(x, y, k).$$

Lema 4.11. *Obstaja koda n_q, za katero velja, za vse $x, y \in \mathbb{N}$*

$$\varphi_{n_q}(x, y) = q(x, y).$$

Lema 4.12. *Obstaja koda n_p, za katero velja, za vse $x, y \in \mathbb{N}$*

$$\varphi_{n_p}(x, y) = \langle x, y\rangle.$$

Dokaz. Obstoj kod n_o, n_q in n_p sledi iz Kleenejevega izreka o normalni obliki in dejstva, da so preslikave o, q in $\langle \_, \_\rangle$ izračunljive. □

5 Sintetična teorija izračunljivosti

Poznamo dva različna pristopa k matematiki: *sintetični* in *analitični pristop*. Kot opiše Bauer v [3, uvod], pri sintetičnem pristopu vzamemo osnovne objekte teorije kot primitivne ideje, njihove lastnosti opišemo z aksiomi in teorijo nato razvijemo samo z uporabo primitivnih idej in aksiomov. Pri analitičnem pristopu pa osnovne objekte teorije zgradimo iz drugih matematičnih objektov, njihove lastnosti izpeljemo in teorijo razvijemo v širšem matematičnem okolju z uporabo drugih teorij.

Pogosto rečemo, da je Evklidova geometrija sintetična in da je Descartova kartezična geometrija analitična. V sintetični geometriji sta točka in premica ločena primitivna objekta, katerih razmerje opišemo z aksiomi, npr. za vsaki dve točki obstaja samo ena premica, ki poteka skozi obe točki. V kartezični geometriji pa definiramo

prostor kot kartezični produkt naravnih števil. Točko nato zgradimo kot par naravnih števil in premico predstavimo kot podmnožico točk ravnine. Lastnosti točk in premic nato sledijo iz definicij, npr. točka a leži na premici l, če velja $a \in l$, kjer $\in$ predstavlja relacijo pripadnosti iz teorije množic.

Oba pristopa k matematiki imata svoje prednosti in slabosti: sintetični pristop nam omogoči konstrukcijo dokazov, iz katerih lažje razberemo bistvo, medtem ko imamo z analitičnim pristopom večji nabor orodij, s katerimi lahko rešimo dani problem. Ker pri sintetičnem pristopu osnovnih objektov ne gradimo, pač pa jih vzamemo kot primitivne, imamo večjo svobodo izbire, katere objekte vzamemo kot osnovne; to nam omogoči, da za osnovne objekte vzamemo samo objekte, ki nas zanimajo. Na primer: v sintetični diferencialni geometriji predpostavimo, da so vse preslikave gladke; v sintetični topologiji predpostavimo, da so vse preslikave neprekinjene; v sintetični teoriji izračunljivosti predpostavimo, da so vse preslikave izračunljive. To nam omogoči, da imajo osnovni objekti po definiciji lastnosti, ki so pomembne za razvoj teorije. Lastnosti so dane in jih ni potrebno izpeljati.

V teoriji izračunljivosti nas zanimajo izračunljive preslikave, zato jih v sintetičnem pristopu vzamemo kot osnovne objekte naše teorije. Sintetični pristop k teoriji izračunljivosti ima tako idejo izračunljivosti vgrajeno v osnovne objekte same teorije, medtem ko za klasični pristop to ne drži. Posledično RIK postane naravna izbira za razvoj sintetične teorije izračunljivosti, saj so vse preslikave, ki jih lahko definiramo brez uporabe spremenljivk, izračunljive.

5.1 Sintetična Churcheva teza

Večina klasičnih knjig o izračunljivosti se začne s predstavitvijo računskega modela in definicijo izračunljivih funkcij kot funkcij, ki jih lahko izračunamo v predstavljenem računskem modelu. Rogers v svoji knjigi [14] uporabi μ-rekurzivne funkcije (ang. μ-recursive functions), za katere uvede univerzalno koračno preslikavo Φ, ki je ključna za nadaljnji razvoj teorije. Razvoj teorije nadaljuje (kot avtorji podobnih knjig) s predstavitvijo Church-Turingove teze [14, poglavje 1.7], iz katere sledi, da so vse intuitivno izračunljive funkcije izračunljive kot μ-rekurzivne funkcije. S tem Φ pridobi sledečo neformalno lastnost:

$$\forall f\colon \mathbb{N} \to \mathbb{N}.\ f \text{ je intuitivno izračunljiva} \to \exists x{:}\mathbb{N}.\, \forall y{:}\mathbb{N}.\, \exists k{:}\mathbb{N}.\, \Phi_x^k y =_{\mathbb{N}} \mathsf{some}(fx).$$

Lastnost je neformalna, ker se zanaša na idejo intuitivne izračunljivosti, ki namenoma nima natančne definicije. To omogoči Rogersu, da razvije teorijo izračunljivosti na abstrakten način, saj se z uporabo intuitivne izračunljivosti in Church-Turingove teze izogne gradnji funkcij v konkretnem računskem modelu. Posledično lahko univerzalno koračno preslikavo Φ zgradimo v katerem koli računskem modelu.

Rogers razvije teorijo izračunljivosti v klasični teoriji množic, kjer vsaka funkcija, ki jo lahko definiramo, ni nujno izračunljiva. Posledično moramo, če želimo biti natančni, preveriti vsako uporabo Church-Turingove teze, saj je ideja intuitivne izračunljivosti nenatančna. Forster v svojem doktorskem delu ne razvije teorije izračunljivosti v klasični teoriji množic, vendar v RIK-u, kjer so vse funkcije, ki jih lahko definiramo (brez uporabe predpostavk), izračunljive. Posledično je naravno predpostaviti, da je Φ univerzalna za vse funkcije $\mathbb{N} \to \mathbb{N}$. Zaradi zgodovinskih razlogov takšnemu aksiomu pravimo (formalna) Churcheva teza (CT). Forster definira

CT_Φ [5, poglavji 6.1 in 6.2] , ki je parameteriziran z ustaljeno univerzalno koračno preslikavo $\Phi\colon \mathbb{N} \to \mathbb{N} \to \mathbb{N} \to \mathbb{ON}$. Uporabo koračne preslikave na kodi x z vhodnim argumentom y in s k koraki pa zapiše kot $\Phi_x^k y$ namesto kot $\Phi\, x\, y\, k$.

Definicija 5.1 (CT_Φ)**.** *Churcheva teza* (CT_Φ) [5, poglavje 6.1] se glasi: obstaja ustaljena koračna univerzalna preslikava $\Phi\colon \mathbb{N} \to \mathbb{N} \to \mathbb{N} \to \mathbb{ON}$, za katero velja

$$\Pi f\colon \mathbb{N} \to \mathbb{N}.\, \exists x{:}\mathbb{N}.\, \forall y{:}\mathbb{N}.\, \exists k{:}\mathbb{N}.\, \Phi_x^k y =_{\mathbb{N}} \mathsf{some}(fy).$$

Opomba 5.2. Univerzalna koračna preslikava Φ je ustaljena (ang. stationary), ker zanjo velja

$$\forall xyk_1k_2z{:}\mathbb{N}.\, \Phi_x^{k_1} y = \mathsf{some}\, z \to k_2 \geq k_1 \to \Phi_x^{k_2} y = \mathsf{some}\, z.$$

CT_Φ v RIK-u ne moremo dokazati, lahko pa ga predpostavimo. Konsistentnost CT_Φ in dela RIK-a, ki ga uporablja Forster v svojem doktorskem delu, dokažemo v poglavju 9. Ker je CT_Φ univerzalen za vse preslikave $\mathbb{N} \to \mathbb{N}$ v RIK-u, kjer so vse funkcije, ki jih lahko definiramo, izračunljive, uporabe CT_Φ ni potrebno preveriti, kot je to potrebno pri uporabi Church-Turingove teze v klasični teoriji množic.

Definicijo CT_Φ lahko razširimo na preslikave več spremenljivk z uporabo združitvenih preslikav $\langle \_, \_, \ldots, \_ \rangle\colon \mathbb{N}^k \to \mathbb{N}$ [14, poglavje 5.3]. Za delno aplikacijo preslikav več spremenljivk uporabimo izrek SMN_Φ.

Definicija 5.3 (SMN_Φ)**.** Izrek *SMN*$_\Phi$ [5, poglavje 6.1] se glasi: za ustaljeno koračno univerzalno preslikavo $\Phi\colon \mathbb{N} \to \mathbb{N} \to \mathbb{N} \to \mathbb{ON}$ velja

$$\begin{aligned}&\Sigma q\colon \mathbb{N} \to \mathbb{N} \to \mathbb{N}.\, \Pi x y_0 y_1 z{:}\mathbb{N}.\\ &\quad (\exists k{:}\mathbb{N}.\Phi_{q\, x\, y_0}^k y_1 = \mathsf{some}\, z) \leftrightarrow (\exists k.\Phi_x^k \langle y_0, y_1\rangle = \mathsf{some}\, z).\end{aligned}$$

Če uporabimo abstrakten Φ in predpostavimo CT_Φ, se lahko izognemo definiciji konkretnega računskega modela. Izkaže pa se, da je razvoj teorije z uporabo SMN_Φ naporen. Forster zato uvede sintetično Churchevo tezo, ki predpostavi koračno preslikavo Φ in je priročnejša oblika aksiomov CT_Φ in SMN_Φ.

Definicija 5.4 (SCT)**.** *Sintetična Churcheva teza* (SCT) [5, poglavje 6.2] se glasi:

$$\begin{aligned}&\Sigma\phi\colon \mathbb{N} \to \mathbb{N} \to \mathbb{N} \to \mathbb{ON}.\\ &\quad (\forall xyk_0k_1z.\, \phi_x^{k_1} y = \mathsf{some}\, z \Rightarrow k_2 \geq k_1 \Rightarrow \phi_x^{k_2} y = \mathsf{some}\, z) \wedge\\ &\quad \forall (f\colon \mathbb{N} \to \mathbb{N} \to \mathbb{N}).\, \exists \gamma\colon \mathbb{N} \to \mathbb{N}.\, \forall iy.\, \exists k.\, \phi_{\gamma i}^k y = \mathsf{some}\, (f_i\, y),\end{aligned}$$

kjer je Φ ustaljena koračna preslikava, univerzalno parameterizirana za preslikave $\mathbb{N} \to \mathbb{N}$. Preslikava Φ je univerzalno parameterizirana, saj lahko za vsako družino funkcij $f_i\colon \mathbb{N} \to \mathbb{N}$, parameterizirano z $i{:}\mathbb{N}$, dobimo kodirno funkcijo γ, tako da je γi koda za f_i.

6 Dokaz neprotislovnosti

V tem poglavju pokažemo, da je teorija, ki jo razvije Forster v svojem doktorskem delu [5], neprotislovna. Poglavje začnemo s kratkim pregledom ideje dokaza in nato nadaljujemo z natančno obravnavo posameznih delov. Tekom poglavja se sklicujemo na dele dokaza, ki smo jih dokazali s pomočjo dokazovalnega pomočnika Agda. Koda dokaza pa je dostopna na [11].

Forster razvije sintetično teorijo izračunljivosti v RIK-u, na katerem temelji dokazovalni pomočnik Coq, vendar pri razvoju teorije uporabi le omejen del RIK-a. Za namen dokaza se omejimo na del RIK-a, ki ga potrebujemo za razvoj teorije, in ga poimenujemo fRIK. Sistem fRIK podamo kasneje v poglavju 6.1. Poleg fRIK-a za razvoj teorije potrebujemo še sintetično Churchevo tezo (SCT) (def. 5.4), zakon izključene sredine (ZIS) (def. 3.8) in pravila velike uporabe (PVU) (pog. 3.3), ki jih privzame kot aksiome. Da pokažemo, da je sintetična teorija izračunljivosti neprotislovna, moramo dokazati, da uporaba fRIK z aksiomi SCT, ZIS in PVU ne pripelje do protislovja.

Trditev 6.1. *fRIK s SCT, ZIS in PVU je neprotisloven.*

Najprej dokažemo, da je fRIK s SCT in ZIS neprotisloven (poglavje 6.2), nato pa dokažemo, da lahko predpostavimo tudi PVU, ne da bi s tem uvedli protislovje (poglavje 6.3).

Opomba 6.2. Besedne zveze s podobno strukturo, kot jo ima »fRIK s SCT, ZIS in PVU«, pogosto okrajšamo kot »fRIK + SCT + ZIS + PVU«, kjer s simbolom »+« označimo, da fRIK uporabimo skupaj z aksiomi SCT, ZIS in PVU.

Opomba 6.3. Prav tako pogosto uporabimo sledeče okrajšave:

$$
\begin{aligned}
\mathit{fRIK}_{SiZ} &:= \text{fRIK} + \text{SCT} + \text{ZIS},\\
\mathit{fRIK}_{S} &:= \text{fRIK} + \text{SCT},\\
\mathit{fRIK}_{SZiP} &:= \text{fRIK} + \text{SCT} + \text{ZIS} + \text{PVU in}\\
\mathit{fRIK}_{SMiP} &:= \text{fRIK} + \text{SCT} + \text{PM} + \text{PVU}.
\end{aligned}
$$

6.1 Fragment RIK-a (fRIK)

Forster v svojem delu uporabi omejen nabor tipov, ki jih navede v prilogi A svojega doktorskega dela [5].

Definicija 6.4. *Fragment RIK-a (fRIK)* je podsistem tipov RIK-a, v katerem lahko zgradimo vse tipe, ki jih uporabi Forster pri razvoju sintetične teorije izračunljivosti. Ta vsebuje:

- tipe $\mathbb{1}$ (enojec), $\mathbb{B}$ (Boolove vrednosti), $\mathbb{N}$ (naravna števila), $\mathbb{0}$ (mogoče), X^n (vektor), $\mathbb{L}$ (lista), $\to$ (preslikava), Π (odvisni produkti), Σ (odvisne vsote) in $=$ (enakost) ter
- ločen svet izjav $\mathbb{P}$ s tipi $\bot$ (neresnica), $\top$ (resnica), $\wedge$ (in), $\vee$ (ali), $\Rightarrow$ (sledi), $\exists$ (obstaja) in $\forall$ (za vse).

fRIK ne podpira splošnih induktivnih tipov, zato tipov, kot so $\mathbb{N}$, Σ, $\top$, in $\bot$, ne moremo definirati na običajen način, kot to naredimo v RIK-u. Kot osnovne tipe predpostavimo $\mathbb{1}$, $\mathbb{B}$, $\mathbb{N}$, $\mathbb{L}$, Π, Σ, $=$ in $\forall$, preostale tipe pa zgradimo iz osnovnih tipov na običajen način, kot na primer:

- $\mathbb{O}A \coloneqq \Pi x{:}\mathbb{B}.\,(\mathsf{if}\ x\ \mathsf{then}\ \mathbb{1}\ \mathsf{else}\ A)$, kjer $\mathsf{none} \coloneqq (\mathsf{false}, \star)$ in $\mathsf{some}\ a \coloneqq (\mathsf{true}, a)$,
- $A \to B \coloneqq \Pi x{:}A.\,B$, kjer $A, B{:}\,\mathbb{T}$ in B ni odvisen od x,
- $P \Rightarrow R \coloneqq \forall \_{:}P.\,R$, kjer z _ nadomestimo ime spremenljivke, ki je v izrazu ne uporabimo,
- $\bot \coloneqq \forall P{:}\mathbb{P}.\,P$,
- $\top \coloneqq \forall P{:}\mathbb{P}.\,P \Rightarrow P$,
- $P \wedge R \coloneqq \forall Q{:}\mathbb{P}.\,(P \Rightarrow (R \Rightarrow Q)) \Rightarrow Q$,
- $P \vee R \coloneqq \forall Q{:}\mathbb{P}.\,(P \Rightarrow Q) \Rightarrow ((R \Rightarrow Q) \Rightarrow Q)$, in
- $\exists x{:}A.\,P(x) \coloneqq \forall Q{:}\mathbb{P}.\,(\forall x{:}A.\,P(x) \Rightarrow Q) \Rightarrow Q$ kjer $P{:}\,A \to \mathbb{P}$.

6.2 fRIK s SCT in ZIS je neprotisloven

V poglavju dokažemo sledečo trditev.

Trditev 6.5. *fRIK + SCT + ZIS je neprotisloven.*

Dokaz je sestavljen iz dveh korakov. V prvem koraku dokažemo, da je fRIK + SCT neprotisloven, saj zanj obstaja model. V drugem koraku pa dokažemo, da če je fRIK + SCT neprotisloven, potem lahko predpostavimo tudi ZIS, ne da bi s tem uvedli protislovje.

Lema 6.6. *fRIK + SCT je neprotisloven.*

Dokaz. fRIK + CT_Φ + SMN_Φ je neprotisloven, saj zanj obstaja model, ki ga podamo v poglavju 9.2. Iz predpostavk CT_Φ (def. 5.1) in SMN_Φ (def. 5.3) sledi, da velja tudi SCT, kot to pokaže Forster v izreku 6.2 v [5]. □

Lema 6.7. *Če je fRIK + SCT neprotisloven, potem je fRIK + SCT + ZIS neprotisloven.*

Da dokažemo lemo 6.7, zgradimo sintaktično preslikavo, ki slika sodbe iz formalnega sistema v formalni sistem, in uporabimo sledečo idejo.

Lema 6.8. *Naj bosta X in Y sistema tipov ter $\psi{:}\,X \to Y$ preslikava, za katero velja, da slika*

- *veljavne sodbe sistema X v veljavne sodbe sistema Y in*
- *sodbo neresnice $\bot_X$ sistema X v sodbo neresnice $\bot_Y$ sistema Y.*

Če je Y neprotisloven, je tudi X neprotisloven.

Dokaz. Predpostavimo, da v sistemu X lahko izpeljemo sodbo $\vdash e{:}\bot_x$. Po definiciji preslikave ψ jo ta preslika v veljavno sodbo $\vdash e'{:}\bot_y$ sistema Y. Ker je sistem Y neprotisloven, v njem ne moremo izpeljati sodbe $\vdash e'{:}\bot_y$, zato sodba ni veljavna. Prišli smo do protislovja. Sledi, da v sistemu X ne moremo izpeljati sodbe $\vdash e{:}\bot_x$, zato je X neprotisloven. □

Opomba 6.9. Na preslikavo ψ lahko gledamo kot na preslikavo, ki vsaki sodbi iz X pripiše/dodeli njen pomen v sistemu Y.

Zgradili bomo preslikavo Ψ, ki bo vsako sodbo iz formalnega sistema fRIK + SCT + ZIS slikala v formalni sistem fRIK + SCT, tako da

- preslikava Ψ slika veljavne sodbe v veljavne sodbe in
- $\Psi(\Gamma \vdash a{:}\bot) = \Gamma \vdash a'{:}\bot$,

kot to velja za ψ v lemi 6.8, ter

- preslikava Ψ ohrani veljavnost SCT-ja in ZIS-a.

Opomba 6.10. V dokazu leme 6.6 smo dokazali, da je fRIK s SCT neprotisloven. Če zgradimo preslikavo Ψ, bo iz leme 6.8 sledilo, da je fRIK s SCT in ZIS neprotisloven, kar želimo dokazati.

Preden podamo preslikavo Ψ, uvedemo potrebne okrajšave, definiramo svet stabilnih izjav in za vsak logični operator podamo izraz, ki predstavlja njegovo stabilno obliko.

Definicija 6.11. fRIK + SCT + ZIS okrajšamo s $fRIK_{SiZ}$.

Definicija 6.12. fRIK + SCT okrajšamo s $fRIK_S$.

Uvedemo svet stabilnih izjav.

Definicija 6.13. Izjava $p{:}\mathbb{P}$ je *stabilna*, če zanjo velja $\neg\neg p \Rightarrow p$.

Vse stabilne izjave lahko združimo v svet stabilnih izjav $\mathbb{P}_{\neg\neg}$, ki mu pravimo »Prop ne ne«, in ga definiramo s sledečo definicijo.

Definicija 6.14. *Svet stabilnih logičnih izjav* $\mathbb{P}_{\neg\neg}$ definiramo v slogu Tarskega kot par

$$\mathbb{P}_{\neg\neg} := (\Sigma p{:}\mathbb{P}.\, \neg\neg p \Rightarrow p, El_{\mathbb{P}_{\neg\neg}}),$$

kjer je $El_{\mathbb{P}_{\neg\neg}}{:}\, (\Sigma p{:}\mathbb{P}.\, \neg\neg p \Rightarrow p) \to \mathbb{T}^1$ preslikava s predpisom $El_{\mathbb{P}_{\neg\neg}}(p) = El_{\mathbb{P}}(\pi_1 p)$.

Elementi sveta $\mathbb{P}_{\neg\neg}$ so pari tipa $\Sigma p{:}\,\mathbb{P}.\, \neg\neg p \Rightarrow p$, kjer za vsak $P{:}\mathbb{P}_{\neg\neg}$ velja, da je $\pi_1 P{:}\mathbb{P}$ stabilna izjava in $\pi_2 P$ dokaz njene stabilnosti.

Opomba 6.15. Vsako stabilno izjavo sveta $\mathbb{P}_{\neg\neg}$ lahko pretvorimo v izjavo sveta $\mathbb{P}$ s prvo projekcijo.

Za vse običajne logične operatorje lahko najdemo njihovo stabilno obliko, za katero veljajo enaka pravila sklepanja.

Opomba 6.16. Prvo projekcijo stabilne izjave $P\colon \mathbb{P}_{\neg\neg}$ pogosto namesto $\pi_1 P$ zapišemo samo kot P.

Definicija 6.17. *Stabilni logični operatorje* označimo s podpisom $_n$ in ji definiramo kot:

- $\bot_n \coloneqq \bot$,
- $\top_n \coloneqq \top$,
- $\neg_n A \coloneqq \neg A$,
- $A \vee_n B \coloneqq \neg\neg(A \vee B)$,
- $A \wedge_n B \coloneqq \neg\neg(A \wedge B)$,
- $A \to_n B \coloneqq A \to B$,
- $\exists_n x{:}A.\, P(x) \coloneqq \neg\neg\exists x{:}A.\, P(x)$, in
- $\forall_n x{:}A.\, P(x) \coloneqq \forall x{:}A.\, P(x)$.

Dokaze stabilnosti operatorjev podamo v priloženem dokazu [11, modul Logic_n].

Da bi se prepričali, da stabilna oblika operatorjev res predstavlja želene logične operatorje, moramo pokazati, da zanje veljajo običajna pravila sklepanja. Na primer, pravila vpeljave in uporabe disjunkcije morajo veljati tako za $\vee$ kot tudi za njegovo stabilno obliko $\vee_n$.

Lema 6.18. *Za stabilne logične operatorje veljajo običajna pravila sklepanja.*

Dokaz. Pravila sklepanja za stabilne logične operatorje, npr. $\vee_n$-intro-left, $\vee_n$-intro-right in $\vee_n$-elim, podamo v priloženem dokazu [11, modul Logic_n]. □

Opomba 6.19. Vsaki izjavi sveta $\mathbb{P}$ lahko priredimo njeno stabilno obliko z rekurzivnim sprehodom po delih izjave, kjer vsak logični operator nadomestimo z njegovo stabilno obliko.

Lema 6.20. *V svetu $\mathbb{P}_{\neg\neg}$ velja*

$$\forall_n P{:}\mathbb{P}_{\neg\neg}.\, P \vee_n \neg_n P. \tag{ZIS_n}$$

Dokaz. V svetu, kjer so vse izjave stabilne, velja ZIS, saj je dobro znano dejstvo, da

$$(\forall p{:}A.\, \neg\neg p \Rightarrow p) \Leftrightarrow (\forall p{:}A.\, p \vee \neg p),$$

kjer je A poljuben svet izjav. □

Opomba 6.21. Ker v fRIK_{SiZ} veljata aksioma SCT in ZIS, imamo konstanti $a_{\mathrm{SCT}}\colon \mathrm{SCT}$ in $a_{\mathrm{ZIS}}\colon \mathrm{ZIS}$.

Sedaj podamo preslikavo Ψ. Osnovna ideja preslikave Ψ je, da slika vse sodbe iz sistema fRIK_{SiZ} v sistem fRIK_S in pri tem ohrani vse sodbe enake, razen sodb izjav sveta $\mathbb{P}$, ki jih slika v sodbe stabilne oblike sveta $\mathbb{P}_{\neg\neg}$ z enakim pomenom.

Definicija 6.22 (Ψ)**.** Preslikavo $\Psi\colon \mathrm{fRIK}_{SiZ} \to \mathrm{fRIK}_S$ definiramo rekurzivno na sodbah sistema fRIK_{SiZ}:

$$\begin{aligned}
&\Psi(\Gamma \vdash A \text{ type}) \coloneqq \Psi(\Gamma) \vdash \Psi(A) \text{ type},\\
&\Psi(\Gamma \vdash A \doteq B \text{ type}) \coloneqq \Psi(\Gamma) \vdash \Psi(A) \doteq \Psi(B) \text{ type},\\
&\Psi(\Gamma \vdash a{:}A) \coloneqq \Psi(\Gamma) \vdash \Psi(a){:}\,\Psi(A) \text{ in}\\
&\Psi(\Gamma \vdash a \doteq b{:}A) \coloneqq \Psi(\Gamma) \vdash \Psi(a) \doteq \Psi(b){:}\,\Psi(A),
\end{aligned}$$

tako da preslikava ohranja kontekst, tipe in izraze z izjemo:

- tipa $\mathbb{P}$, ki ga slika v tip $\mathbb{P}_{\neg\neg}$,
- logičnih izjav sveta $\mathbb{P}$, ki jih slika v njihovo stabilno obliko sveta $\mathbb{P}_{\neg\neg}$, kot to opišemo v opombi 6.19,
- izpeljav logičnih izjav sveta $\mathbb{P}$, kjer pravila vpeljave in uporabe logičnih operatorjev nadomesti z enakimi pravili stabilne oblike, in
- konstant $a_{\mathrm{SCT}}\colon \mathrm{SCT}$ in $a_{\mathrm{ZIS}}\colon \mathrm{ZIS}$, ki ju slika v izraza $a_{\mathrm{SCT}_n}\colon \Psi(\mathrm{SCT})$ in $a_{\mathrm{ZIS}_n}\colon \Psi(\mathrm{ZIS})$.

Izraza a_{SCT_n} in a_{ZIS_n} podamo v priloženem dokazu [11, definicija SCT_n in ZIS_n].

Opazimo, da ima preslikava sledečo lastnost.

Opomba 6.23. Vse izjave, ki jih preslikamo s Ψ, so stabilne.

Sedaj dokažemo, da ima preslikava Ψ želene lastnosti. Najprej dokažemo, da Ψ ohrani SCT in ZIS. Nato dokažemo, da Ψ ohranja veljavnost sodb.

Lema 6.24. *Preslikava Ψ ohrani SCT.*

Dokaz. Preslikava Ψ slika $\vdash a_{\mathrm{SCT}}\colon \mathrm{SCT}$ (def. 5.4) v $\vdash a_{\mathrm{SCT}_n}\colon \Psi(\mathrm{SCT})$, kjer je

$$\begin{aligned}
&\Psi(\mathrm{SCT}) = \Sigma\phi\colon \mathbb{N} \to \mathbb{N} \to \mathbb{N} \to \mathbb{O}\mathbb{N}.\\
&\quad (\forall_n xyk_1k_2z.\, \phi_x^{k_1} y =^d_{\mathbb{B}_n} \mathsf{some}\, z \Rightarrow_n k_2 \geq k_1 \Rightarrow_n \phi_x^{k_2} y =^d_{\mathbb{B}_n} \mathsf{some}\, z) \wedge \qquad (\mathrm{SCT}_n)\\
&\quad \forall_n(f\colon \mathbb{N} \to \mathbb{N} \to \mathbb{N}).\, \exists_n\gamma\colon \mathbb{N} \to \mathbb{N}.\, \forall_n iy.\, \exists_n k.\, \phi_{\gamma i}^k y =^d_{\mathbb{B}_n} \mathsf{some}\,(f_i\, y),
\end{aligned}$$

ki ga označimo s SCT_n. Da bi dokazali, da Ψ ohranja SCT, moramo pokazati da v fRIK_S velja $\vdash a_{\mathrm{SCT}_n}\colon \mathrm{SCT}_n$. Izraz $a_{\mathrm{SCT}_n}\colon \mathrm{SCT}_n$ zgradimo v priloženem dokazu [11, definicija SCT_n] s pomočjo SCT. □

Lema 6.25. *Preslikava Ψ ohrani ZIS.*

Dokaz. Preslikava Ψ slika $\vdash a_{\mathrm{ZIS}}$:ZIS v $\vdash a_{\mathrm{ZIS}_n}$:$\Psi$(ZIS), kjer

$$\Psi(\mathrm{ZIS}) = \forall_n P{:}\mathbb{P}_{\neg\neg}.\, P \vee_n \neg_n P, \qquad (\mathrm{ZIS}_n)$$

ki ga označimo z ZIS_n. Ker je $\mathbb{P}_{\neg\neg}$ svet stabilnih izjav, v njem velja ZIS_n (lema 6.20). Izraz $a_{\mathrm{ZIS}_n}\colon \mathrm{ZIS}_n$ zgradimo v priloženem dokazu [11, definicija ZIS_n]. □

Lema 6.26. *Preslikava Ψ ohranja veljavnost sodb.*

Dokaz. Lemo dokažemo z indukcijo po drevesu izpeljave poljubne sodbe.

Preslikava Ψ ohranja veljavnost sodb, če velja, da ko lahko izpeljemo sodbo J v fRIK_{SiZ}, lahko izpeljemo tudi sodbo $\Psi(J)$ v fRIK_S. Ali preslikava Ψ ohranja veljavnost izpeljav, je odvisno od sprememb, ki jih Ψ naredi na posamezni sodbi. Opazimo, da Ψ ohrani vse tipe sveta $\mathbb{T}$ in njihove izraze. Spremeni pa vse izjave sveta $\mathbb{P}$ in njihova pravila sklepanja ter slika konstanti a_{SCT} in a_{ZIS} v izraza a_{SCT_n} in a_{ZIS_n}. Za dele izjav, kjer se tipi in izrazi ne spremenijo, izpeljave ostanejo enake. Prepričati pa se moramo, da se veljavnost sodb ohrani v primeru, ko sodbe vsebujejo izjave sveta $\mathbb{P}$, konstanto a_{SCT} ali konstanto a_{ZIS}.

Iz definicije vidimo, da preslikava Ψ slika vse izjave iz sveta $\mathbb{P}$ v svet $\mathbb{P}_{\neg\neg}$, pri čemer le spremeni logične operatorje v njihovo stabilno obliko. Ker imajo stabilni logični operatorji enaka pravila sklepanja (lema 6.17), preslikava Ψ ohrani izpeljavo izjav, saj vedno slika pravila sklepanja v pravila sklepanja stabilnih operatorjev, npr. $\wedge$-intro v $\wedge_n$-intro in $\vee$-elim v $\vee_n$-elim. Posledično, če lahko izjavo $P{:}\mathbb{P}$ dokažemo pred preslikavo z dokazom p, lahko njeno sliko $\Psi(P){:}\mathbb{P}_{\neg\neg}$ dokažemo tudi po preslikavi z dokazom $\Psi(p)$.

Če sodba vsebuje konstanto a_{SCT} oziroma a_{ZIS}, jo preslikava Ψ preslika v izraz a_{SCT_n} oziroma a_{ZIS_n}. Ker smo v lemah 6.24 in 6.25 pokazali, da preslikava Ψ ohrani SCT in ZIS, vemo, da lahko izraza a_{SCT_n} in a_{ZIS_n} zgradimo tudi v SCT_S. □

Podamo dokaz leme 6.7: če je fRIK_S neprotisloven, potem je fRIK_{SiZ} neprotisloven.

Dokaz (Lema 6.7). Predpostavimo, da je fRIK_S neprotisloven. Za dokaz uporabimo pristop, predstavljen v lemi 6.8. Zgradili smo preslikavo Ψ, ki slika sodbe iz sistema fRIK_{SiZ} v sistem fRIK_S, pri čemer slika $\bot$ v $\bot$ in ohranja veljavnost sodi (lema 6.26). Ker je fRIK_S neprotisloven, znotraj fRIK_S ne moremo zgraditi sodbe $\vdash p{:}\bot$ za nek izraz p. Od tod sledi, da prav tako ne moremo zgraditi sodbe $\vdash p'{:}\bot$ v fRIK_{SiZ}: če bi jo lahko, bi jo Ψ slikal v sodbo $\vdash \Psi(p'){:}\bot$ sveta fRIK_S. Sledi, da je fRIK_{SiZ} neprotisloven. □

Dele dokaza združimo in dokažemo končno trditev 6.5: fRIK + SCT + ZIS je neprotisloven.

Dokaz. Trditev je posledica lem 6.6 in 6.7. □

6.3 Predpostavimo lahko pravila velike uporabe

V poglavju 6.2 smo pokazali, da je fRIK + SCT + ZIS neprotisloven. V nadaljevanju pokažemo, da lahko znotraj fRIK-a, v katerem predpostavimo SCT in ZIS, predpostavimo tudi pravila uporabe PVU, ne da bi s tem uvedli protislovje.

Forster v svojem delu uporabi tri velika pravila uporabe PVU: pravilo velike uporabe neresnice (VU$\bot$), pravilo velike uporabe enakosti (VU$=$), in pravilo velike eliminacije eksistenčnih kvantifikatorjev odločljivih predikatov na naravnih številih (VU$\exists$P$\mathbb{N}$).

Dokaz ima enako strukturo kot dokaz trditve 6.5 in ga podamo kot njegovo nadgradnjo. Najprej dokažemo, da je fRIK + SCT + PM + PVU neprotisloven.

Nato dokažemo, da iz neprotislovnosti fRIK + SCT + PM + PVU sledi, da je fRIK + SCT + ZIS + PVU neprotisloven.

Lema 6.27. *fRIK + SCT + PM + PVU je neprotisloven.*

Dokaz. fRIK + SCT + PM + PVU je neprotisloven, saj zanj obstaja model. Za model vzamemo enak model kot za fRIK + SCT v lemi 6.6, ki ga razširimo z interpretacijo pravila PM in pravil PVU. Interpretacij pravila PM in pravil PVU ne podamo, vendar menimo, da jih lahko podamo na podoben način, kot smo podali interpretaciji CT_Φ in SMN_Φ (poglavje 9). □

Lema 6.28. *Če je fRIK + SCT + PM + PVU neprotisloven, potem je fRIK + SCT + ZIS + PVU neprotisloven.*

Za enostavnejši zapis uvedemo sledeči okrajšavi.

Definicija 6.29. fRIK + SCT + ZIS + PVU okrajšamo s $fRIK_{SZiP}$.

Definicija 6.30. fRIK + SCT + PM + PVU okrajšamo s $fRIK_{SMiP}$.

Za dokaz uporabimo enako idejo, kot smo jo uporabili pri dokazu leme 6.7, kjer preslikavo Ψ razširimo, tako da poleg pravil SCT in ZIS ohranja tudi pravila PVU.

Definicija 6.31 ($\boldsymbol{\Psi'}$). Preslikavo $\Psi'\colon \mathrm{fRIK}_{SZiP} \to \mathrm{fRIK}_{SMiP}$ definiramo podobno kot preslikavo 6.22 z dvema spremembama:

- preslikava Ψ' slika iz fRIK_{SZiP} v fRIK_{SMiP} in
- dodatno slika konstante:
  - $a_{\mathrm{VU}\bot}$:VU$\bot$ v izraz $a_{\mathrm{VU}\bot_n}$:Ψ(VU$\bot$),
  - $a_{\mathrm{VU}=}$:VU$=$ v izraz $a_{\mathrm{VU}=_n}$:Ψ(VU$=$) in
  - $a_{\mathrm{VU}\exists\mathrm{P}\mathbb{N}}$:VU$\exists$P$\mathbb{N}$ v izraz $a_{\mathrm{VU}\exists\mathrm{P}\mathbb{N}_n}$:$\Psi$(VU$\exists$P$\mathbb{N}$).

Izraze $a_{\mathrm{VU}\bot_n}$, $a_{\mathrm{VU}=_n}$ in $a_{\mathrm{VU}\exists\mathrm{P}\mathbb{N}_n}$ podamo v priloženem dokazu [11, definicija LE$\bot_n$, LE$=_n$ in LE$\exists$P$\mathbb{N}_n$]

Pokažemo, da ima Ψ' želene lastnosti.

Lema 6.32. *Preslikava Ψ' ohrani pravilo* VU$\bot$.

Dokaz. Pravilo VU$\bot$ je definirano kot $\Pi A{:}\mathbb{T}.\, \bot \to A$ (def. 3.3). Preslikava Ψ slika pravilo VU$\bot$ v

$$\Pi A{:}\mathbb{T}.\, \bot_n \to A. \qquad (\mathrm{VU}\bot_n)$$

Opazimo, da preslikava Ψ' spremeni le $\bot$ v $\bot_n$. Ker pa smo $\bot_n$ definirali kot $\bot$, pravilo ostane enako in zato velja tudi po preslikavi. Izgradnjo izraza $a_{\mathrm{VU}\bot_n}$ podamo v priloženem dokazu [11, definicija LE$\bot_n$]. □

Lema 6.33. *Preslikava Ψ' ohrani pravilo* VU$=$.

Dokaz. Preslikava Ψ' pravila ne spremeni. Zato pravilo velja tudi po preslikavi. Izraz $a_{\mathrm{VU}=_n}$ definiramo kot konstanto $a_{\mathrm{VU}=}$. □

Lema 6.34. *Preslikava Ψ' ohrani pravilo* VU∃PN*, če predpostavimo* PM*.*

Dokaz. Pravilo VU∃PN (def. 3.3) je definirano kot

$$\Pi f{:}\,\mathbb{N} \to \mathbb{B}.\, (\exists m{:}\mathbb{N}.\, fm =^d_{\mathbb{B}} \mathsf{true}) \to \Sigma m{:}\mathbb{N}.\, fm =^d_{\mathbb{B}} \mathsf{true},$$

kjer ima $=^d_{\mathbb{B}} : \mathbb{B} \to \mathbb{B} \to \mathbb{P}$ predpis

$$\begin{aligned}(\mathsf{true} =^d_{\mathbb{B}} \mathsf{true}) &= \top\\ (\mathsf{false} =^d_{\mathbb{B}} \mathsf{false}) &= \top\\ (\mathsf{true} =^d_{\mathbb{B}} \mathsf{false}) &= \bot\\ (\mathsf{false} =^d_{\mathbb{B}} \mathsf{true}) &= \bot.\end{aligned}$$

Preslikava Ψ' slika $\vdash a_{\mathrm{VU\exists PN}}{:}\,\mathrm{VU\exists PN}$ v $\vdash a_{\mathrm{VU\exists PN}_n}{:}\,\Psi(\mathrm{VU\exists PN})$, kjer

$$\begin{aligned}&\Psi(\mathrm{VU\exists PN}) = && (\mathrm{VU\exists PN}_n)\\ &\quad \Pi f{:}\,\mathbb{N} \to \mathbb{B}.\, (\exists_n m{:}\mathbb{N}.\, fm =^d_{\mathbb{B}n} \mathsf{true}) \to \Sigma m{:}\mathbb{N}.\, fm =^d_{\mathbb{B}n} \mathsf{true}\end{aligned}$$

in $=^d_{\mathbb{B}n} : \mathbb{B} \to \mathbb{B} \to \mathbb{P}_{\neg\neg}$ slika enako kot $=^d_{\mathbb{B}}$, vendar namesto da slika v $\bot$ in $\top$, slika v $\bot_n$ in $\top_n$.

Opazimo, da preslikava Ψ' spremeni VU∃PN na dveh mestih: logični operator $\exists$ slika v $\exists_n$, preslikavo $=^d_{\mathbb{B}}$ pa v $=^d_{\mathbb{B}n}$. Preslikavi $=^d_{\mathbb{B}}$ in $=^d_{\mathbb{B}n}$ predstavljata isto preslikavo, saj smo definirali $\bot_n$ kot $\bot$ in $\top_n$ kot $\top$. Od tod sledi, da preslikava spremeni le $\exists$ v $\exists_n$, kjer smo $\exists_n$ definirali kot $\neg\neg\exists$ (def. 6.17).

Da pokažemo, da preslikava Ψ' ohrani pravilo $\mathrm{VU\exists PN}_n$, moramo pokazati, da v fRIK_{SMiP} lahko izpeljemo $\vdash a_{\mathrm{VU\exists PN}_n}{:}\mathrm{VU\exists PN}_n$. Sledi skica dokaza. Natančno izgradnjo izraza $a_{\mathrm{VU\exists PN}_n}$ podamo v priloženem dokazu [11, definicija LE∀P$\mathbb{N}_n$].

Za poljuben $f{:}\,\mathbb{N} \to B$ predpostavimo $\exists_n m.\, fm =^d_{\mathbb{B}}$ true in pokažemo, da velja $\Sigma m.fm =^d_{\mathbb{B}}$ true. Vemo, da v fRIK_{SiZ} velja VU∃PN. Da pridobimo $\Sigma m.\, fm =^d_{\mathbb{B}}$ true z uporabo VU∃PN, moramo pokazati, da obstaja priča m, za katero velja $fm =^d_{\mathbb{B}}$ true. Uporabimo PM (def. 3.9) in iz predpostavke $\exists_n m.\, fm =^d_{\mathbb{B}}$ true, ki jo po definiciji $\exists_n$ lahko zapišemo kot $\neg\neg\exists m.\, fm =^d_{\mathbb{B}}$ true, pridobimo pričo m, ki jo potrebujemo. □

Opomba 6.35. Da lahko dokažemo lemo 6.34, uporabimo PM (def. 3.9) in ga zato predpostavimo kot aksiom formalnega sistema, v katerega slika preslikava Ψ'.

Lema 6.36. *Preslikava Ψ' ohrani pravila PVU.*

Dokaz. Posledica lem 6.32, 6.33 in 6.34. □

Lema 6.37. *Preslikava Ψ' ohranja SCT in ZIS.*

Dokaz. Preslikava Ψ' slika SCT in ZIS enako kot Ψ. Ker Ψ ohranja SCT in ZIS (lemi 6.24 in 6.25), ju ohranja tudi Ψ'. □

Lema 6.38. *Preslikava Ψ' ohranja veljavnost sodb.*

Dokaz. Preslikava Ψ' slika enako kot Ψ, z izjemo izrazov $a_{\mathrm{VU}\bot}$, $a_{\mathrm{VU}=}$ in $a_{\mathrm{VU}\exists\mathrm{P}\mathbb{N}}$, ki jih slika v izraze $a_{\mathrm{VU}\bot_n}$, $a_{\mathrm{VU}=_n}$ in $a_{\mathrm{VU}\exists\mathrm{P}\mathbb{N}_n}$. Vemo, da preslikava Ψ ohranja veljavnost sodb, kot smo to pokazali v dokazu leme 6.26. Prav tako vemo, da preslikava Ψ' ohranja pravila PVU (lema 6.36). Sledi, da preslikava Ψ' ohranja veljavnost vseh sodb. □

Podamo dokaz leme 6.28: če je fRIK_{SMiP} neprotisloven, potem je fRIK_{SZiP} neprotisloven.

Dokaz (Lema 6.28). Predpostavimo, da je fRIK_{SMiP} neprotisloven. Za dokaz uporabimo pristop, predstavljen v lemi 6.8. Zgradili smo preslikavo Ψ', ki slika sodbe iz sistema fRIK_{SZiP} v sistem fRIK_{SMiP}, pri čemer slika $\bot$ v $\bot$ in ohranja veljavnost sodb (lema 6.38). Ker je fRIK_{SMiP} neprotisloven, znotraj fRIK_{SMiP} ne moremo zgraditi sodbe $\vdash p{:}\bot$ za nek izraz p. Od tod sledi, da prav tako ne moremo zgraditi sodbe $\vdash p'{:}\bot$ v fRIK_{SZiP}: če bi jo lahko, bi jo Ψ' slikal v sodbo $\vdash \Psi'_i(p'){:}\bot$ sveta fRIK_{SMiP}. Sledi, da je fRIK_{ZiP} neprotisloven. □

Dele dokaza združimo in dokažemo končno trditev 6.1: fRIK + SCT + ZIS + PVU je neprotisloven.

Dokaz (Trditev 6.1). Trditev je posledica lem 6.27 in 6.28. □

S tem smo zaključili dokaz neprotislovnosti formalnega sistema, ki ga uporablja Forster za razvoj svoje teorije.

7 Del dokaza neprotislovnosti v Agdi

Agda [16] je funkcijski programski jezik z odvisnimi tipi zasnovan na Martin-Löfovem sistemu tipov in opremljen z lastnostmi uporabnimi za praktično programiranje kot so ujemanje vzorcev (ang. pattern matching), inferenca tipov (ang. type inference) in sistem modulov (ang. module system). Agdo lahko uporabljamo kot programski jezik in dokazovalni pomočnik, saj močan sistem tipov omogoča, da po Curry-Howardovi korespondenci predstavimo logične izjave kot tipe in dokaze kot programe. Trenutno verzijo Agde (Agda 2) je načrtoval in implementiral Ulf Norell [9].

Z Agdo v našem delu podamo del dokaza neprotislovnosti, ki ga opišemo v preostanku poglavja. Za dokaz uporabimo verzijo Agde 2.6.3 [16]. Koda dokaza je dostopna na [11].

V Agdi svet logičnih izjav $\mathbb{P}$ definiramo v slogu Tarskega kot strukturo CICProp s tipom prop in preslikavo $\vdash\colon \mathsf{prop} \to \mathbb{T}$. CICProp opremimo s konstruktorjem all za izgradnjo kartezičnega produkta družine izjav $A \to \mathsf{prop}$ ter s pravilom vpeljave all-intro in uporabe all-elim. Pokažemo, da v svetu CICProp veljajo običajni logični operatorji in pravila sklepanja, katera zgradimo z uporabo konstruktorja all ter pravil all-intro in all-elim.

Nadaljujemo z izgradnjo sveta stabilnih logičnih izjav $\mathbb{P}_{\neg\neg}$ v slogu Tarskega in ga podamo kot strukturo $\mathsf{CICProp}{\neg\neg}$:

$$\begin{aligned}\mathsf{CICProp}{\neg\neg}.\mathsf{prop} &:= \{`p\colon \mathsf{prop},\ \mathsf{stable}\colon (\vdash \neg\neg `p) \to \vdash `p\}\\ \mathsf{CICProp}{\neg\neg}.\vdash \quad &:= \{`p, \mathsf{prop}\} \to \vdash `p.\end{aligned}$$

Struktura $\mathsf{CICProp}{\neg\neg}$ vsebuje le stabilne izjave $`p$ sveta $\mathsf{CICProp}$. Za vsako izjavo $`p$ je stable dokaz njene stabilnosti. Pokažem, da v $\mathsf{CICProp}{\neg\neg}$ lahko zgradimo običajne logične operatorje in pravila sklepanja.

Nato dokažemo še sledeče izjave:

- V $\mathsf{CICProp}{\neg\neg}$ velja ZIS [11, definicija ZIS_n].
- Če SCT velja v $\mathsf{CICProp}$, velja tudi v $\mathsf{CICProp}{\neg\neg}$. [11, definicija SCT_n]
- Če pravila velike uporabe VU$\bot$, VU$=$ in VU$\exists$P$\mathbb{N}$ veljajo v $\mathsf{CICProp}$, veljajo tudi v $\mathsf{CICProp}{\neg\neg}$. [11, definicije $\mathsf{LE}\bot_n$, $\mathsf{LE}\equiv_n$in $\mathsf{LE}\forall\mathsf{P}\mathbb{N}_n$]

8 Teorija realizabilnosti

V tem poglavju opišemo osnovne pojme teorije realizabilnosti, ki jih potrebujemo za razumevanje izgradnje modelov fRIK + SCT in fRIK + SCT + ZIS. Kot osnovno literaturo za to poglavje smo uporabili zapiske prof. Andreja Bauerja [1], ki jih je pripravil kot spremno besedilo za predmet Izračunljiva topologija (Univerza v Ljubljani, 2009) in gostujoča predavanja na Midlands Graduate School (Univerza v Nottinghamu, 2022).

Začetnik teorije realizabilnosti je Stephen Kleene, ki je uporabil idejo realizacije za izgradnjo modela konstruktivne teorije naravnih števil [7]. Osnovno vprašanje, ki motivira teorijo realizabilnosti, se glasi:

Kako lahko implementiramo dano matematično strukturo?

Za nekatere strukture je odgovor enostaven. Grupo implementiramo kot:

- tip, ki predstavlja elemente grupe,
- vrednost, ki predstavlja nevtralni element,
- preslikavo, ki implementira operacijo na grupah, in
- primeren inverz.

Odgovor na vprašanje pa ni enostaven, če si ogledamo zanimivejše strukture: na primer, kako implementiramo realna števila in prostor gladkih preslikav?

V teoriji realizabilnosti implementacijo strukture opišemo z uporabo *realizacijske relacije*

$$r \Vdash x,$$

ki vsakemu elementu x znotraj strukture pripiše element r, ki ga implementira. Pravimo da r *realizira (implementira, predstavlja, je priča)* x. Poglavje začnemo z opisom delnih kombinatornih algeber, katerih elemente uporabimo kot osnovne gradnike za implementacijo struktur, in nadaljujemo z definicijo skupkov s katerimi opišemo, kako so strukture implementirane.

8.1 Delna kombinatorna algebra

Definicija 8.1. *Delna kombinatorna algebra* (ang. partial combinatory algebra) $(\mathbb{A}, \cdot)$ je množica $\mathbb{A}$ z delno preslikavo $\cdot : \mathbb{A} \times \mathbb{A} \rightharpoonup \mathbb{A}$, za katero obstajata elementa $\mathtt{K}, \mathtt{S} \in \mathbb{A}$, da za vse $x, y, z \in \mathbb{A}$ velja

$$K \cdot x \cdot y = x, \quad \mathtt{S} \cdot x \cdot y \cdot z \simeq (x \cdot z) \cdot (y \cdot z) \quad \text{in} \quad \mathtt{S} \cdot x \cdot y \downarrow .$$

Na preslikavo $\cdot$ gledamo kot aplikacijo in njen zapis $x \cdot y$ običajno okrajšamo kot $x\,y$. Aplikacija $\cdot$ je levo asociirana.

Za delno kombinatorno algebro $\mathbb{A}$ definiramo *izraze* induktivno glede na strukturo izraza:

- vsak element $a \in \mathbb{A}$ je izraz,
- spremenljivka je izraz in
- če sta e_1 in e_2 izraza, potem je $e_1 \cdot e_2$ izraz.

Izraz je *zaprt*, če ne vsebuje spremenljivk. Pravimo, da je zaprt izraz *definiran*, ko so vse aplikacije v izrazu e definirane, kar zapišemo kot $e\downarrow$. Izraza e in e' sta $e \simeq e'$, če in samo če $e\downarrow$, $e'\downarrow$ in $e = e'$, ali $e\uparrow$ in $e'\uparrow$. Če sta e in e' izraza, katerih spremenljivke so v $x_1, x_2, \ldots, x_n$, zapišemo $e \simeq e'$, ko za vse $a_1, a_2, \ldots, a_n \in \mathbb{A}$ velja

$$e[a_1/x_1, a_2/x_2, \ldots, a_n/x_n] \simeq e'[a_1/x_1, a_2/x_2, \ldots, a_n/x_n].$$

Ko je $e\downarrow$, je $e \in \mathbb{A}$.

Trditev 8.2 (Kombinatorna polnost)**.** *Naj bo $(\mathbb{A}, \cdot)$ delna kombinatorna algebra. Za vsako spremenljivko x in izraz e obstaja izraz e', katerega spremenljivke so enake spremenljivkam izraza e z izjemo spremenljivke x, tako da $e'\downarrow$ in $e' \cdot a \simeq e[a/x]$.*

Izraz e' pogosto zapišemo kot $\langle x \rangle e$. Meta-notacija $\langle x \rangle e$ igra podobno vlogo kot λ-abstrakcija v λ-računu brez tipov. Zapis $\langle x \rangle \langle y \rangle e$ okrajšamo kot $\langle x, y \rangle e$.

Znotraj poljubne delne kombinatorne algebre $\mathbb{A}$ lahko zgradimo pare, Boolove vrednosti, sezname, naravna števila in možnosti.

Definicija 8.3. Delne kombinatorne algebre imajo *pare*, saj lahko zgradimo elemente

$$\begin{aligned} \mathtt{pair} &\coloneqq \langle x, y, z \rangle\, z\, x\, y, \\ \mathtt{fst} &\coloneqq \langle p \rangle\, p\, (\langle x, y \rangle\, x) \text{ in} \\ \mathtt{snd} &\coloneqq \langle p \rangle\, p\, (\langle x, y \rangle\, y), \end{aligned}$$

za katere velja: za vse $a, b \in \mathbb{A}$,

$$\mathtt{fst}\ (\mathtt{pair}\ a\ b) = a \qquad \text{in} \qquad \mathtt{snd}\ (\mathtt{pair}\ a\ b) = b.$$

Definicija 8.4. Delne kombinatorne algebre imajo *Boolove vrednosti*, saj lahko zgradimo elemente

$$\begin{aligned} \texttt{true} &:= \langle x, y\rangle\, x, \\ \texttt{false} &:= \langle x, y\rangle\, y \text{ in} \\ \texttt{if} &:= \langle x\rangle\, x, \end{aligned}$$

za katere velja: za vse $x, y \in \mathbb{N}$,

$$\texttt{if true}\ x\ y = x \quad \text{in} \quad \texttt{if false}\ x\ y = y.$$

Definicija 8.5. Delne kombinatorne algebre imajo *sezname*, saj lahko zgradimo elemente

$$\begin{aligned} \texttt{nil} &:= \texttt{pair true true}, \\ \texttt{cons}\ x\ l &:= \texttt{pair false (pair}\ x\ l), \\ \texttt{isempty} &:= \texttt{fst}, \\ \texttt{hd}\ l &:= \texttt{snd (fst}\ l) \text{ in} \\ \texttt{tl}\ l &:= \texttt{snd (snd}\ l), \end{aligned}$$

za katere velja: za vse $x, l \in \mathbb{A}$,

$$\begin{aligned} &\texttt{isempty nil} = \texttt{true}, \\ &\texttt{isempty (cons}\ x\ l) = \texttt{false}, \\ &\texttt{hd (cons}\ x\ l) = x \text{ in} \\ &\texttt{tl (cons}\ x\ l) = l. \end{aligned}$$

Funkcije lahko definiramo rekurzivno z uporabo kombinatorjev fiksnih točk (ang. fixed point combinators) `X` in `Y`, ki jih zgradimo kot

$$\begin{aligned} \texttt{W} &:= \langle x, y\rangle\, y(x\,x\,y), \\ \texttt{Z} &:= \texttt{W}\,\texttt{W}, \\ \texttt{X} &:= \langle x, y, z\rangle\, y(x\,x\,y)z, \text{ in} \\ \texttt{Y} &:= \texttt{X}\,\texttt{X}. \end{aligned}$$

Za kombinatorja `X` in `Y` velja: za vse $f, x \in \mathbb{A}$,

$$\texttt{Z}\,f \simeq f(\texttt{Z}\,f), \quad \texttt{Y}f\!\downarrow \quad \text{in} \quad (\texttt{Y}\,f)\,x \simeq f(\texttt{Y}\,f)\,x.$$

Definicija 8.6. Delne kombinatorne algebre imajo *naravna števila* saj lahko zgradimo

$$\begin{aligned} \texttt{zero} &:= \texttt{S K K}, \\ \texttt{succ} &:= \langle x\rangle\ \texttt{pair false}\ x, \\ \texttt{iszero} &:= \texttt{fst}, \\ \texttt{pred} &:= \langle x\rangle\ \texttt{if (iszero}\ x)\ \texttt{zero (snd}\ x) \text{ in} \\ \texttt{rec} &:= \langle x, f, m\rangle\ ((\texttt{Z R})\,x\,f\,m\ \texttt{zero}), \end{aligned}$$

kjer je `rec` primitivna rekurzija, za katero velja: za vse $a, f \in \mathbb{A}$ in $n \in \mathbb{N}$,

$$\texttt{rec}\ a\, f\, \overline{0} = a \qquad \text{in} \qquad \texttt{rec}\ a\, f\, \overline{n+1} \simeq f\, \overline{n}\, (\texttt{rec}\ a\, f\, \overline{n}),$$

kjer

$$\overline{0} = \texttt{zero} \qquad \text{in} \qquad \overline{n+1} = \texttt{succ}\ \overline{n}$$

ter

$$\texttt{R} \coloneqq \langle r, x, f, m\rangle\ \texttt{if}\ (\texttt{iszero}\ m)(\texttt{K}\, x)(\langle y\rangle f(\texttt{pred}\ m)(r\, x\, f\, (\texttt{pred}\ m)\, \texttt{I})).$$

Znotraj delne kombinatorne algebre lahko zgradimo tudi pomanjševalni kombinator `min` (ang. minimization combinator), za katerega velja: za vse $f \in \mathbb{A}$ in $n \in \mathbb{N}$,

$$\texttt{min}\, f = \overline{n} \iff \exists k > 0.\, f\, \overline{n} = \overline{k} \text{ in } \forall m < n.\, f\, \overline{m} = \overline{0}.$$

Izgradnje kombinatorja `min` ne podamo.

Opomba 8.7. S predstavljenimi kombinatorji lahko znotraj delne kombinatorne algebre zgradimo katerokoli splošno rekurzivno funkcijo.

Definicija 8.8. Delne kombinatorne algebre imajo elemente *možnosti* (ang. option type) saj lahko zgradimo

$$\begin{aligned} \texttt{none} &\coloneqq \texttt{pair false false}, \\ \texttt{some}\ x &\coloneqq \texttt{pair true}\ x, \\ \texttt{anything} &\coloneqq \texttt{fst} \text{ in} \\ \texttt{get} &\coloneqq \texttt{snd}, \end{aligned}$$

za katere velja: za vse $a \in \mathbb{A}$,

$$\begin{aligned} &\texttt{anything}\ (\texttt{some}\ a) = \texttt{true}, \\ &\texttt{anything}\ \texttt{none} = \texttt{false} \text{ in} \\ &\texttt{get}\ (\texttt{some}\ a) = a. \end{aligned}$$

8.2 Kleenejeva prva algebra

Kleenejeva prva algebra $\mathbb{K}_1$ je delna kombinatorna algebra $(\mathbb{N}, \cdot)$, katere elementi so naravna števila, ki predstavljajo kode izračunljivih delnih preslikav $\mathbb{N} \rightharpoonup \mathbb{N}$. Za potrebe našega dela predpostavimo, da so delne preslikave oštevilčene na standarden način, na primer z uporabo Kleenejevega izreka o normalni obliki (izrek 4.4). Vsako število x tako predstavlja Turingov stroj, ki izračuna delno preslikavo $\varphi_n \colon \mathbb{N} \rightharpoonup \mathbb{N}$. Aplikacijo $m \cdot n$ definiramo kot $\varphi_m(n)$ in jo pogosto zapišemo kot $\{m\}(n)$, da se izognemo zamenjavi za operacijo množenja z enako oznako. Večkratno aplikacijo $\{m\}(n)(p)$ okrajšamo kot $\{m\}(n, p)$.

Kombinator `K` lahko enostavno pridobimo, saj je razvidno, da je preslikava

$$p(x, y) = x$$

izračunljiva. Po izreku SMN zato obstaja izračunljiva preslikava $l\colon \mathbb{N} \to \mathbb{N}$, za katero velja $\varphi_{l(x)}(y) = x$. Kombinator $\mathtt{K}$ je katerokoli število, za katero velja $l = \varphi_{\mathtt{K}}$.

Izgradnja kombinatorja $\mathtt{S}$ pa je nekoliko težja. Delna preslikava

$$g(x, y, z) = \varphi_{\varphi_x(z)}(\varphi_y(z))$$

je izračunljiva z večkratno uporabo izreka UTS. Po izreku SMN obstaja izračunljiva preslikava $r\colon \mathbb{N} \to \mathbb{N}$, za katero velja $\varphi_{r(x,y)}(z) = g(x, y, z)$. S ponovno uporabo izreka SMN dobimo izračunljivo preslikavo $q\colon \mathbb{N} \to \mathbb{N}$, kjer $\varphi_{q(x)}(y) = r(x, y)$. Kombinator $\mathtt{S}$ je katerokoli število, za katero velja $q = \varphi_{\mathtt{S}}$.

8.3 Skupki

Definicija 8.9. *Skupek* (ang. assembly) $S = (|S|, \Vdash_S)$ nad delno kombinatorno algebro $\mathbb{A}$ je par množice elementov $|S|$ in relacije $\Vdash_S$ med $\mathbb{A}$ in $|S|$, za katerega velja, da za vsak $x \in |S|$ obstaja $\mathtt{x} \in \mathbb{A}$, tako da $\mathtt{x} \Vdash_S x$. Pravimo, da $\mathtt{x}$ *realizira* x.

Preslikava skupkov (ang. assembly map) $f\colon S \to T$ med skupkoma S in T je preslikava $f\colon |S| \to |T|$ med množicama $|S|$ in $|T|$, za katero obstaja takšna preslikava $\mathtt{f} \in \mathbb{A}$, da velja: za vse $x \in |S|$ in $\mathtt{x} \in \mathbb{A}$, če $\mathtt{x} \Vdash_S x$, potem $\mathtt{fx}\downarrow$ in $\mathtt{fx} \Vdash_T f(x)$.

Asm(𝔸) je množica vseh skupkov, definiranih nad delno kombinatorno algebro $\mathbb{A}$.

Skupki in preslikave skupkov nad $\mathbb{A}$ tvorijo kategorijo Asm($\mathbb{A}$). Če preslikavi $f\colon S \to T$ in $g\colon T \to U$ realizirata $\mathtt{f}, \mathtt{g} \in \mathbb{A}$, njuno kompozicijo $g \circ f$ realizira $\langle x\rangle\, \mathtt{g}\,(\mathtt{f}\,x) = \mathtt{S}\,(\mathtt{K}\,\mathtt{g})(\mathtt{S}\,(\mathtt{K}\,\mathtt{f})(\mathtt{S}\,\mathtt{K}\,\mathtt{K}))$. Identiteto $id_S\colon |S| \to |S|$ realiziramo kot $\langle x\rangle x = \mathtt{S}\,\mathtt{K}\,\mathtt{K}$. Kompozicija je asociativna, saj je kompozicija preslikav.

Pogosto z enako črko označimo element in njegov realizator, vendar uporabimo drugačno pisavo: na primer, elementi x, y, f in g bi imeli realizatorje $\mathtt{x}, \mathtt{y}, \mathtt{f}$ in $\mathtt{g}$.

Definicija 8.10. *Skromen skupek* (ang. modest assembly) S je skupek, katerega elementi si ne delijo realizatorjev:

$$\forall r \in \mathbb{A}.\, \forall x, y \in |S|.\, (r \Vdash_S x \wedge r \Vdash_S y \Rightarrow x = y).$$

Mod($\mathbb{A}$) je množica vseh skromnih skupkov, definiranih nad delno kombinatorno algebro $\mathbb{A}$. Mod($\mathbb{A}$) je polna podkatetegorija Asm($\mathbb{A}$), katere objekti so skromni skupki.

Izkaže se, da lahko večino struktur v izračunljivi matematiki predstavimo kot skromne skupke.

Definicija 8.11. Relacija $\Vdash_S$ skupka S je *trivialna*, če za vse $x \in S$ velja

$$\forall \mathtt{r} \in \mathbb{A}.\, \mathtt{r} \Vdash_S x.$$

Definicija 8.12. Vsako množico lahko spremenimo v skupek z uporabo preslikave $\nabla\colon \mathrm{Set} \to \mathrm{Asm}(\mathbb{A})$ s predpisom

$$\nabla X = (X, \Vdash_{\nabla X}),$$

ki slika množice v skupke s trivialno relacijo.

Da pridobimo intuicijo o skupkih, si oglejmo nekatere konkretne skupke.

Definicija 8.13. *Prazen skupek* $\mathbb{0}$ je skupek $(\emptyset, \Vdash_\emptyset)$, kjer je $\Vdash_\emptyset$ prazna množica.

Definicija 8.14. *Enojec* $\mathbb{1}$ je skupek $(\{\star\}, \Vdash_\mathbb{1})$ s trivialno relacijo.

Definicija 8.15. *Skupek Boolovih vrednosti* $\mathbb{B}$ je skupek $(\mathbb{B}, \Vdash_\mathbb{B})$, kjer Boolove vrednosti tipa $\mathbb{B}$ in pravilo if realiziramo kot

$$\begin{aligned} &\texttt{true} \;\Vdash_\mathbb{B} \mathsf{true},\\ &\texttt{false} \Vdash_\mathbb{B} \mathsf{false} \text{ in}\\ &\texttt{if} \;\;\Vdash_\mathbb{B} \mathsf{if} \end{aligned}$$

Elementi `true`, `false` in `if` so elementi, kot smo jih definirali v 8.4.

Definicija 8.16. *Skupek naravnih števil* $\mathbb{N}$ je skupek $(\mathbb{N}, \Vdash_\mathbb{N})$, kjer naravna števila tipa $\mathbb{N}$ realiziramo s številkami $\overline{n}$, kot smo jih definirali v definiciji 8.6, tako da velja: za vse $\texttt{r} \in \mathbb{A}$ in $n \in \mathbb{N}$,

$$\texttt{r} \Vdash_\mathbb{N} n \iff \texttt{r} = \overline{n}.$$

Operator succ realiziramo kot `succ` in 0 kot $\overline{0}$. Za poljuben skupek S z elementom $z \in S$ in preslikavo $f\colon |S| \to |S|$, ki ju realizirata $\texttt{z}, \texttt{f} \in \mathbb{A}$, je edinstvena preslikava $\overline{f}\colon \mathbb{N} \to |S|$, za katero velja: za vse $n \in \mathbb{N}$,

$$\overline{f}(0) = z \qquad \text{in} \qquad \overline{f}(n+1) = f(\overline{f}(n)),$$

realizirana z `rec z f`.

Definicija 8.17. *Skupek seznamov* $\mathbb{L}A$ za poljuben skupek A je skupek $(\mathbb{L}A, \Vdash_{\mathbb{L}A})$, kjer sezname tipa $\mathbb{L}A$ in pravila $\mathsf{isempty}$, hd in tl realiziramo kot

$$\begin{aligned} &\texttt{nil} \qquad \Vdash_{\mathbb{L}A} [\,],\\ &\texttt{const a l} \Vdash_{\mathbb{L}A} a{::}l, \text{ kjer } \texttt{a} \Vdash_A a \text{ in } \;\texttt{l} \Vdash_{\mathbb{L}A} l,\\ &\texttt{isempty} \;\Vdash_{\mathbb{L}A} \mathsf{isempty},\\ &\texttt{hd} \qquad \Vdash_{\mathbb{L}A} \mathsf{hd}, \text{ in}\\ &\texttt{tl} \qquad \Vdash_{\mathbb{L}A} \mathsf{tl}, \end{aligned}$$

Elementi `nil`, `cons`, `isempty`, `hd` in `tl` so elementi, kot smo jih definirali v definiciji 8.5.

Opomba 8.18. Kasneje bomo elemente Kleenejeve prve algebre uporabili kot osnovne gradnike za realizacijo fRIK-a, kjer bomo s skupki opisali, kateri elementi Kleenejeve prve algebre realizirajo katere elemente fRIK-a. Intuitivno lahko na elemente Kleenejeve prve algebre gledamo kot na »kode« programov, zapisanih v nizkonivojskem programskem jeziku, ki ga lahko izvedemo, in realizacijo fRIK-a kot njegovo implementacijo v takšnem jeziku.

8.4 Kategorična struktura skupkov

Vsakodnevno matematiki pri svojem delu uporabljajo omejeno število konstrukcij na množicah: produkte, vsote, podmnožice, kvocientne množice, slike, prostore preslikav, induktivne in ko-induktivne definicije, potenčne množice, unije, preseke in komplemente množic. Za vse naštete konstrukcije, razen potenčne množice, obstajo analogne konstrukcije na skupkih, kar nam omogoča, da mnoge znane konstrukcije na množicah prenesemo na skupke. V nadaljevanju podamo definicijo produktov in exponentov na skupkih.

Definicija 8.19. *Dvojiški produkt skupkov* S in T je skupek $(|S| \times |T|, \Vdash_{S\times T})$ z relacijo

$$q \Vdash_{S\times T} (x,y) \iff \mathtt{fst}\, q \Vdash_S x \wedge \mathtt{snd}\, q \Vdash_T y.$$

Za vsak $\mathtt{x}, \mathtt{y} \in A$, če $\mathtt{x} \Vdash_S x$ in $\mathtt{y} \Vdash_T y$, potem $\mathtt{pair\ x\ y} \Vdash_{S\times T} (x,y)$. Projekciji π_1 in π_2, realiziramo kot `fst` and `snd`. Elementi `pair`, `fst` in `snd` so elementi, kot smo jih definirali v definiciji 8.3

Definicija 8.20. *Eksponent skupkov* S in T je skupek

$$T^S = (\{f{:}S \to T \mid f \text{ je realiziran}\}, \Vdash_{T^S})$$

z relacijo

$$\mathtt{f} \Vdash_{T^S} f \iff \forall x \in S.\, \forall \mathtt{x} \in \mathbb{A}.\, \mathtt{x} \Vdash_S x \Rightarrow \mathtt{fx}{\downarrow} \wedge \mathtt{fx} \Vdash_T f(x).$$

Evaluacijo $e{:}\, T^S \times S \to T$, kjer $e(f,x) = f(x)$, realizira

$$\mathtt{e} = \langle p\rangle(\mathtt{fst}\, p)(\mathtt{snd}\, p).$$

Za vsako preslikavo $f{:}\, U \times S \to T$, ki jo realizira $\mathtt{f} \in \mathbb{A}$, obstaja njena *transpozicija* $\hat{f}{:}\, U \to S \to T$, kjer $\hat{f}(x)(y) = f(x,y)$, ki jo realizira

$$\mathtt{trans\ f} = \langle x\rangle\langle y\rangle\, \mathtt{f}(\mathtt{pair}\ x\, y).$$

Eksponent skupkov S in T zapišemo tudi $S \to T$.

Lema 8.21. *Kategoriji Asm(𝔸) in Mod(𝔸) sta zaprti za kartezične produkte.*

Dokaz. Za vsak skupek $S, T \in Asm(\mathbb{K}_1)$ lahko zgradimo eksponent $T^S \in Asm(\mathbb{K}_1)$, kot smo to opisali v definiciji 8.20. Prav tako lahko za vsak skromni skupek $S, T \in Mod(\mathbb{K}_1)$ zgradimo eksponent $T^S \in Mod(\mathbb{K}_1)$, kjer je T^S skromni skupek [1, izjava 3.6.12]. □

8.5 Odvisni tipi

V matematiki se pogosto srečamo s parametriziranimi konstrukcijami, kot so na primer zvezne preslikave $f{:}[a,b] \to \mathbb{R}$, katerih domena je odvisna od vrednosti spremenljivk a in b, ter ciklične grupe $\mathbb{Z}_n$, kjer je dolžina cikla odvisna od števila n. Takšne konstrukcije predstavljajo *družine* objektov, ki so oštevilčeni z vrednostjo vhodnih spremenljivk, ki jim pravimo *parametri*. Da lahko predstavimo takšne konstrukcije v teoriji realizabilnosti, uvedemo pojem *družine skupkov*.

Definicija 8.22. *Družina skupkov* $S\colon I \to \mathrm{Asm}(\mathbb{A})$ je preslikava $S\colon |I| \to \mathrm{Asm}(\mathbb{A})$, ki slika elemente množice $|I|$ v skupke.

Pravimo, da je I *indeksni skupek* in da je S *indeksiran* z I. S $\mathrm{Fam}(I)$ označimo zbirko vseh družin skupkov, ki so indeksirana z I. Aplikacijo indeksa $i \in I$ na družino skupkov $S \in \mathrm{Fam}(I)$ pogosto označimo z A_i namesto z $A(i)$.

Definicija 8.23. *Preslikava družin skupkov* $f\colon A \to B$ med družinama skupkov $A, B \in \mathrm{Fam}(I)$ je taka družina preslikav $(f_i\colon |A_i| \to |B_i|)_{i\in|I|}$, za katero obstaja $\mathtt{f} \in \mathbb{A}$, da za vse $i \in I$, $x \in A_i$ in $\mathtt{i}, \mathtt{x} \in \mathbb{A}$ velja

$$\mathtt{i} \Vdash_I i \wedge \mathtt{x} \Vdash_{A_i} x \Rightarrow \mathtt{f\ \ i\ \ x} \Vdash_{B_i} fx.$$

Zbirka $\mathrm{Fam}(I)$ s preslikavami družin skupkov tvori kategorijo.

Družino $A \in \mathrm{Fam}(I)$ lahko *reindeksiramo* s preslikavo $r\colon J \to I$, tako da komponiramo A z r. Reindeksiranje družin nam omogoča definicijo kartezičnih produktov in vsot na družinah, ki so odvisne od več spremenljivk, kjer se ob konstrukciji produkta ali vsote želimo osredotočiti na samo eno od spremenljivk. Na primer, če imamo družino $A \in \mathrm{Fam}(I \times J)$ in se želimo pri izgradnji produkta osredotočiti samo na spremenljivko J, potem lahko za poljuben $i \in I$ z uporabo preslikave $r\colon J \to I \times J$, kjer $r(j) \coloneqq (i, j)$, zgradimo kartezični produkt $\Pi_J(A \circ r)$, ki teče le po elementih J.

8.6 Produkti in vsote družin skupkov

Dve osnovni operaciji na družinah množic, ki ju pogosto srečamo, sta kartezični produkt in vsota. Sprva ju definiramo na družinah množic in nato razširimo njuno definicijo še na družine skupkov.

Družina množic je preslikava $A\colon I \to \mathrm{Set}$, ki slika iz množice I v množico vseh množic Set. Z oznako $\mathrm{Fam}'(\mathrm{I})$ označimo zbirko vseh družin množic, ki so indeksirana z I. Za dano družino množic $A \in \mathrm{Fam}'(I)$ lahko definiramo kartezični produkt $\Pi_I A$ in vsoto $\Sigma_I A$ kot množici

$$\Pi_I A = \{u\colon I \to \bigcup_{i\in I} A_i \mid \forall i \in I.\, ui \in A_i\} \text{ in}$$
$$\Sigma_I A = \{(i, x) \mid i \in I \wedge x \in A_i\}.$$

Elementom kartezičnega produkta $\Pi_I A$ pravimo *izbirne preslikave*. Intuicijo za to poimenovanje podamo skozi primer 8.24.

Primer 8.24. Da pridobimo intuicijo, zakaj elementom $\Pi_I A$ pravimo izbirne preslikave, si oglejmo natančen pomen trditve

$$\text{»}(a + b)/2 \text{ je element zaprtega intervala } [a, b]\text{«}.$$

Sprva moramo iz trditve uganiti, da sta a in b realni števili in da velja $a < b$. To predstavlja naš kontekst, ki ga podamo v slogu teorije tipov kot

$$a{:}\mathbb{R},\ b{:}\mathbb{R},\ p{:}(a < b),$$

kjer je p dokaz trditve $a < b$. V danem kontekstu zgradimo družino množic C zaprtih intervalov

$$C_{(a,b,p)} = [a, b] = \{x \in \mathbb{R} \mid a \leq x \leq b\}.$$

Izraz $(a + b)/b$ tako predstavlja predpis, ki vsakemu intervalu $[a, b]$ določi eno točko glede na podani kontekst. Pravimo, da je $(a + b)/2$ izbirna preslikava za C, saj vsakemu zaprtemu intervalu $[a, b]$ priredi eno točko znotraj intervala. Enako vlogo kot $(a + b)/2$ imajo elementi $u \in \Pi_I A$, saj u vsaki množici A_i priredi en element, ki je del te množice.

Da lahko izrazimo odvisnost vsot in produktov od konteksta, potrebujemo posplošeno obliko produktov in vsot, ki jim pravimo *parameterizirani produkti in vsote* in jih definiramo z uporabo reindeksiranja. Predpostavimo $r\colon J \to I$ in družino $A \in \mathrm{Fam}'(J)$. Za poljubno množico $K \subseteq I$ zapišemo $r^*K = \{j \in J \mid rj \in K\}$. Kartezični produkt $\Pi_r A \in \mathrm{Fam}'(I)$ in vsoto $\Sigma_r A \in \mathrm{Fam}'(I)$ definiramo kot

$$(\Pi_r A)_i = \{u\colon r^*\{i\} \to \bigcup_{j \in r^*\{i\}} A_j \mid \forall j \in r^*\{i\}.\, uj \in A_j\} \text{ in}$$
$$(\Sigma_r A)_i = \{(j, x) \mid j \in r^*\{i\} \wedge x \in A_j\}.$$

Sedaj lahko podamo parameterizirano obliko kartezičnih produktov in vsot družin skupkov.

Definicija 8.25. Za poljubne $A \in \mathrm{Fam}(J)$, $r\colon J \to I$ in $i \in I$ je *kartezični produkt družine skupkov* A skupek $((\Pi_r A)_i, \Vdash_{(\Pi_r A)_i})$ kjer

$$|(\Pi_r A)_i| = \{u \in (\Pi_r |A|)_i \mid \exists \mathtt{u} \in \mathbb{A}.\, \mathtt{u} \Vdash_{(\Pi_r A)_i} u\} \text{ in}$$

$$\mathtt{u} \Vdash_{(\Pi_r A)_i} u \iff \forall j \in r^*\{i\}.\, \forall \mathtt{j} \in \mathbb{A}.\, \mathtt{j} \Vdash_J j \Rightarrow \mathtt{uj} \Vdash_{A_j} uj.$$

Definicija 8.26. Za poljubne $A \in \mathrm{Fam}(J)$, $r\colon J \to I$ in $i \in I$ je *vsota družine skupkov* A skupek $((\Sigma_r A)_i, \Vdash_{(\Sigma_r A)_i})$ kjer

$$|(\Sigma_r A)_i| = \{(j, x) \in (\Sigma_r |A|)_i \mid \exists \mathtt{p} \in \mathbb{A}.\, \mathtt{p} \Vdash_{(\Sigma_r A)_i} (j, x)\} \text{ in}$$

$$\mathtt{p} \Vdash_{(\Sigma_r A)_i} (j, x) \iff \mathtt{fst\ p} \Vdash_J j \wedge \mathtt{snd\ p} \Vdash_{A_i} x.$$

Za boljše razumevanje podamo tudi ne-parameterizirano obliko produktov in vsot družin skupkov.

Definicija 8.27. *Ne-parameteriziran kartezični produkt družine skupkov* $A \in \mathrm{Fam}(I)$ je skupek $(\Pi_I A, \Vdash_{\Pi_I A})$ kjer

$$|\Pi_I A| = \{u \in \Pi_{|I|} |A| \mid \exists \mathtt{u} \in \mathbb{A}.\, \mathtt{u} \Vdash_{\Pi_I A} u\} \text{ in}$$

$$\mathtt{u} \Vdash_{\Pi_I A} u \iff \forall i \in I.\, \forall \mathtt{i} \in \mathbb{A}.\, \mathtt{i} \Vdash_I i \Rightarrow \mathtt{ui} \Vdash_A ui.$$

Definicija 8.28. *Ne-parameterizirana vsota družine skupkov* $A \in \mathrm{Fam}(I)$ je skupek $(\Sigma_I A, \Vdash_{\Sigma_I A})$ kjer

$$|\Sigma_I A| = \{(i, x) \in \Sigma_{|I|} |A| \mid \exists \mathtt{p} \in \mathbb{A}.\, \mathtt{p} \Vdash_{\Sigma_I A} (i, x)\} \text{ in}$$

$$\mathtt{p} \Vdash_{\Sigma_I A} p \iff \mathtt{fst\ p} \Vdash_I i \wedge \mathtt{snd\ p} \Vdash_{A_i} x.$$

8.7 Delne ekvivalenčne relacije

Relaciji, ki je simetrična in tranzitivna, pravimo *delna ekvivalenčna relacija (der). Vsakemu skromnemu skupku* S lahko priredimo der $\approx_S$.

Definicija 8.29. *Der* $\approx_S$ skupka S nad $\mathbb{A}$ je der, definirana nad elementi $\mathbb{A}$, za katero velja: za vse $q, r \in \mathbb{A}$,

$$q \approx_S r \iff \exists x \in |S|.\, q \Vdash_S x \wedge r \Vdash_S x.$$

Realizator $r \in \mathbb{A}$ je *totalen* (ang. total), če $r \approx_S r$. Množico vseh totalnih realizatorjev označimo kot $\mathrm{Supp}(S) = \{r \in A \mid r \approx_S r\}$. Vsak $r \in \mathrm{Supp}(S)$ določa ekvivalenčni razred $[r]_S = \{q \in \mathbb{A} \mid r \approx_S q\}$.

Definicija 8.30. *Ekstenzionalni realizator* (ang. extensional realizer) derov $\approx_S$ in $\approx_T$ je $p \in \mathbb{A}$ za katerega velja: za vse $q, r \in \mathbb{A}$, če velja $q \approx_S r$, potem $p\,q{\downarrow}$ in $p\,r{\downarrow}$ in $p\,q \approx_T p\,r$.

Ekstenzionalna realizatorja p in p' sta *ekvivalentna*, ko iz $q \approx_S r$ sledi $p\,q \approx_T p'\,r$. Ekvivalenčni razred ekstenzionalnih realizatorja p označimo kot $[p]$. Deri in ekvivalenčni razredi ekstenzionalnih realizatorjev tvorijo kategorijo $\mathrm{Per}(\mathbb{A})$, katere objekti so deri nad $\mathbb{A}$ in morfizmi so ekvivalenčni razredi ekstenzionalnih realizatorjev. Kompozicija ekvivalenčnih razredov ekstenzionalnih realizatorjev $[p]\colon S \to T$ in $[q]\colon T \to U$ je $[q \circ p]\colon S \to U$, kjer $q \circ p = \langle x \rangle q(p\,x)$. Identiteto $\mathrm{id}_S\colon S \to S$ realiziramo kot $\langle x \rangle x$.

Lema 8.31. *Kategoriji* $\mathrm{Mod}(\mathbb{A})$ *in* $\mathrm{Per}(\mathbb{A})$ *sta ekvivalentni.*

Dokaz. Vemo, da vsakemu skromnemu skupku S lahko pripišemo der $\approx_S$, kot smo to opisali v definiciji 8.29. Definiramo preslikavo $o\colon \mathrm{Mod}(\mathbb{A}) \to \mathrm{Per}(\mathbb{A})$, kjer

$$o(S) \coloneqq \approx_S.$$

Opazimo, da preslikava $f\colon S \to T$ skupkov S in T, ki ga realizira $p \in \mathbb{A}$, določi morfizem derov $[p]\colon \approx_S \to \approx_T$.

V obratno smer definiramo preslikavo $\iota\colon \mathrm{Per}(\mathbb{A}) \to \mathrm{Mod}(\mathbb{A})$, kjer

$$\iota(\approx_S) \coloneqq (\mathrm{Supp}(\approx_S)/\approx_S, \Vdash_S) \text{ in}$$

$$r \Vdash_S [q]_S \iff r \in q.$$

Z morfizmom $[p]\colon \approx_S \to \approx_T$ lahko zgradimo $\iota[p]\colon \mathrm{Supp}(\approx_S)/\approx_S \to \mathrm{Supp}(\approx_T)/\approx_T$, kjer $\iota[p][r]_S \coloneqq [p\,r]_T$, ki ga realizira p.

Preslikavi o in ι tvorita ekvivalenco kategorij $\mathrm{Mod}(\mathbb{A})$ in $\mathrm{Per}(\mathbb{A})$, kot to pokaže Bauer v [1, lema 3.3.2]. □

Preslikavi o in ι nista le ekvivalenca, saj poleg $\iota(o(A)) \cong A$ velja enakost $o(\iota(R)) = R$.

9 Interpretacija fRIK-a in njegovih razširitev

V tem poglavju predstavimo interpretacijo fRIK-a in njegovih razširitev fRIK + SCT (fRIK$_S$) in fRIK + SCT + ZIS (fRIK$_{SiZ}$) v teoriji realizabilnosti.

Poglavje začnemo s interpretacijo fRIK-a v teoriji realizabilnosti, kjer za delno kombinatorno algebro vzamemo $\mathbb{K}_1$. Pokažemo, da lahko znotraj $\mathbb{K}_1$, realiziramo tudi CT$_\Phi$ in SMN$_\Phi$. Iz česar sledi, da lahko realiziramo tudi SCT in s tem pridobimo interpretacijo fRIK$_S$. Podamo še interpretacijo sveta $\mathbb{P}_{\neg\neg}$, znotraj katere velja ZIS, kar omogoči interpretacijo fRIK$_{SiZ}$. Interpretacije pravil PM in PVU ne podamo, vendar menimo, da jih znotraj $\mathbb{K}_1$ lahko realiziramo. S tem pridobimo še interpretacijo fRIK + SCT + PM + PVU (fRIK$_{SMiP}$) in fRIK + SCT + ZIS + PVU (fRIK$_{SZiP}$).

9.1 Interpretacija fRIK-a

fRIK interpretiramo znotraj teorije realizabilnosti, kjer za delno kombinatorno algebro vzamemo Kleenejevo prvo algebro ($\mathbb{K}_1$). V teoriji realizabilnosti so neodvisni tipi skupki, odvisni tipi so družine skupkov, logične izjave sveta $\mathbb{P}$ so delne ekvivalenčne relacije skupka $\nabla\mathrm{Per}(\mathbb{K}_1)$, njihovi dokazi pa so elementi ekvivalenčnih razredov.

Potrebne ideje za interpretacijo fRIK-a v teoriji realizabilnosti smo predstavili v poglavju 8. Sedaj opišemo, kako z njimi interpretiramo fRIK.

Kontekst

Kontekst je v teoriji realizabilnosti skupek, pri čemer:

- je prazen kontekst enojec $\mathbb{1}$ in
- če je Γ kontekst in $A \in \mathrm{Fam}(\Gamma)$, potem je razširjeni kontekst vsota $\Sigma_\Gamma A$.

Z večkratno razširitvijo konteksta dobimo skupek

$$\Sigma_{\Sigma_{\dots\Sigma_{\Sigma_{\mathbf{1}} S_1} S_2 \dots} S_{n-1}} S_n,$$

ki mu pravimo *teleskop* in ga pogosto zapišemo kot

$$x_1{:}S_1, x_2{:}S_2, \dots, x_{n-1}{:}S_{n-1}, x_n{:}S_n,$$

kjer so $x_1, x_2, \dots, x_{n-1}, x_n$ različne spremenljivke. Sklicevanje na spremenljivke nam omogoča enostaven dostop do predpostavk in nadomesti gnezdenje projekcij. Teleskop skupkov

$$\Gamma = (x_1{:}S_1, x_2{:}S_2, \dots, x_n{:}S_n)$$

je tako skupek $\Gamma = (|\Gamma|, \Vdash_\Gamma)$, kjer

$$|\Gamma| = \{(a_1, a_2, \dots, a_n) \mid \forall i \leq n.\, a_i \in (S_i)_{(a_1, a_2, \dots, a_{i-1})}|\},$$

$$\mathtt{r} \Vdash_\Gamma (a_1, a_2, \dots, a_n) \iff \forall i \leq n.\, \mathtt{proj}_{n,i}\, \mathtt{r} \Vdash_{(S_i)_{(a_1, a_2, \dots, a_{i-1})}} a_i.$$

Pri tem je $\mathtt{proj}_{n,i}$ i-ta projekcija n-terke, ki jo dobimo z gnezdenjem projekcij `fst` in `snd`.

Sodbe

Sodbi $\Gamma \vdash A$ type in $\Gamma \vdash t{:}A$ imata v teoriji realizabilnosti sledeči pomen:

- $\Gamma \vdash A$ type je izjava $A \in \mathrm{Fam}(\Gamma)$ in
- $\Gamma \vdash t : A$ je izjava $t \in \Pi_\Gamma A$.

Zapis $\Gamma \vdash A$ type pogosto nadomestimo z zapisom $A \in \mathrm{Fam}(\Gamma)$, $\Gamma \vdash t : A$ pa z $t \in \Pi_\Gamma A$.

Preprosti tipi

Preprosti tipi $\mathbb{1}, \mathbb{B}$, $\mathbb{N}$ in $\mathbb{L}A$ za poljuben tip A so v teoriji realizabilnosti skupki $\mathbb{1}, \mathbb{B}$, $\mathbb{N}$ in $\mathbb{L}A$, kot smo jih predstavili v poglavju 8.3.

Preslikave $A{\to}B$ so v teoriji realizabilnosti realizirane preslikave med skupkoma A in B. Tip $A{\to}B$ je tako eksponent skupkov B^A, kot ga definiramo v definiciji 8.20. Skupek $A{\to}B$ je poenostavljena oblika kartezičnega produkta skupkov $\Pi_{x:A} B$, kjer je B neodvisen od $x{:}A$, zato zanj veljajo enaka pravila vpeljave in uporabe, kot jih podamo v poglavju 8.6.

Odvisni tipi

Odvisni tipi so v teoriji realizabilnosti družine skupkov. Če je B odvisni tip, ki je odvisen od konteksta Γ, in je e njegova izbirna preslikava, potem je $B \in \mathrm{Fam}(\Gamma)$, kar zapišemo kot $\Gamma \vdash B$ type, in je $e \in \Pi_\Gamma B$, kar zapišemo kot $\Gamma \vdash e{:}B$.

Če imamo družino skupkov $\Gamma, x{:}A \vdash B_x$ type, ki je indeksirana s kontekstom $\Gamma, x{:}A$, lahko zgradimo kartezični produkt družine skupkov po spremenljivki x z uporabo primernega reindeksiranja:

$$\Gamma \vdash \Pi_r B \text{ type},$$

kot smo to opisali v definiciji 8.25, kjer je $r{:}\,(\Gamma, x{:}A) \to \Gamma$ projekcija $r(\gamma, x) = \gamma$ in $\Pi_r B \in \mathrm{Fam}(\Gamma)$.

Opomba 9.1. Če razvijemo definicijo množice $|(\Pi_r B)_\gamma|$ za nek $\gamma \in \Gamma$, opazimo, da

$$\begin{aligned} |(\Pi_r A)_\gamma| &= \{u{:}\, r^*\{\gamma\} \to \cup_{\delta \in r^*\{\gamma\}} A_\delta \,|\, \forall_{\delta \in r^*\{\gamma\}}\, u\,\delta \in A_\delta\} \\ &\cong \{u{:}\, A_\gamma \to \cup_{a \in A_\gamma} B_{(\gamma,a)} \,|\, \forall_{a \in A_\gamma}\, u\,a \in B_{(\gamma,a)}\}. \end{aligned}$$

Za lažjo berljivost namesto zapisa $\Gamma \vdash \Pi_r B$ type uporabljamo zapis $\Gamma \vdash \Pi_{x:A} B_x$ type ali $\Gamma \vdash \Pi_A B$ type, kjer predpostavimo, da smo za izgradnjo produkta uporabili primerno reindeksiranje.

Izbirno preslikavo $\Gamma, x{:}A \vdash e{:}B_x$ lahko λ-abstrahiramo v $\Gamma \vdash f{:}\,\Pi_A B$ z uporabo transpozicije `trans`, tako da

$$\texttt{trans}\,\texttt{e} \Vdash_{\Pi_\Gamma \Pi_A B} f,$$

kjer $\texttt{e} \Vdash_{\Pi_{(\Gamma, x:A)} B_x} e$ in je `trans` operator, kot smo ga definirali v definiciji 8.20. Preslikavo $\Gamma \vdash f{:}\,\Pi_A B$ lahko apliciramo na $\Gamma \vdash x{:}A$, tako da

$$\texttt{S}\,\texttt{f}\,\texttt{x} \Vdash_{\Pi_\Gamma B_x} fx,$$

kjer je $\mathtt{f} \Vdash_{\Pi_\Gamma \Pi_A B} f$ in $\mathtt{x} \Vdash_{\Pi_\Gamma A} x$.

Podobno lahko, če imamo družino skupkov $\Gamma, x{:}A \vdash B_x$ type, zgradimo vsoto družine skupkov po spremenljivki x z uporabo primernega reindeksiranja:

$$\Gamma \vdash \Sigma_r B \text{ type},$$

kjer je $r{:}\,(\Gamma, x{:}A) \to \Gamma$ projekcija $r(\gamma, x) = \gamma$ in $\Sigma_r B \in \mathrm{Fam}(\Gamma)$, kot to opišemo v definiciji 8.26. Za lažjo berljivost namesto zapisa $\Gamma \vdash \Sigma_r B$ type uporabljamo zapis $\Gamma \vdash \Sigma_{x:A} B_x$ type ali $\Gamma \vdash \Sigma_A B$ type, kjer predpostavimo, da smo za izgradnjo produkta uporabili primerno reindeksiranje.

Iz elementov $\Gamma \vdash a{:}A$ in $\Gamma \vdash b{:}B_x$ lahko zgradimo par $\Gamma \vdash (a, b){:}\,\Sigma_{x:A} B_x$ z uporabo konstruktorja `pair`, tako da

$$\langle \gamma \rangle\, \mathtt{pair}\, (\mathtt{a}\, \gamma)\, (\mathtt{b}\, \gamma) \Vdash_{\Pi_\Gamma \Sigma_{x:A} B_x} (a, b),$$

kjer $\mathtt{a} \Vdash_{\Pi_\Gamma A} a$ in $\mathtt{b} \Vdash_{\Pi_\Gamma B_x} b$. Par $\Gamma \vdash (a, b){:}\,\Sigma_{x:A} B_x$ lahko uporabimo s projekcijama `fst` in `snd`, tako da

$$\langle \gamma \rangle\, \mathtt{fst}\, (\mathtt{p}\, \gamma) \Vdash_{\Pi_\Gamma A} a \text{ in } \langle \gamma \rangle\, \mathtt{snd}\, (\mathtt{p}\, \gamma) \Vdash_{\Pi_\Gamma B_x} b,$$

kjer $\mathtt{p} \Vdash_{\Pi_\Gamma \Sigma_{x:A} B_x} (a, b)$ in so `pair`, `fst` in `snd` kombinatorji, kot smo jih definirali v definiciji 8.3.

Tipa preslikav $A \to B$ in produktov $A \times B$ sta poenostavljeni obliki kartezičnega produkta $\Pi_A B$ in vsote $\Sigma_A B$, kjer je $B \in \mathrm{Fam}(A)$ konstantna družina skupkov.

Tip identifikacij

V teoriji realizabilnosti je tip identifikacij družina skupkov $\mathrm{Id}_A \in \mathrm{Fam}(A \times A)$, kjer

$$\mathrm{Id}_A(a, b) = \begin{cases} \mathbb{0} & \text{če } a \neq b, \\ \mathbb{1} & \text{če } a = b. \end{cases} \tag{9.1}$$

Enakost elementa $\Gamma \vdash a{:}A$ samemu sebi dokažemo s pravilom vpeljave

$$\Gamma \vdash \mathrm{refl}_a{:}\, a =_A a,$$

tako da

$$\mathtt{refl}_a \Vdash_{\mathrm{Id}_A(a,a)} \mathrm{refl}_a,$$

kjer

$$\mathtt{refl}_a \coloneqq \mathtt{*}.$$

Za vsako družino $P(x, p)$, indeksirano z $x{:}A$ in $p{:}\, a =_A x$, obstaja pravilo

$$\Gamma \vdash \text{ind-eq}_a{:}\, P(a, \mathrm{refl}_a) \to \Pi_{(x:A)} \Pi_{(p:\, a =_A x)} P(x, p),$$

s katerim lahko element tipa $P(a, \mathrm{refl}_a)$ uporabimo za izgradnjo elementa tipa $P(x, p)$, kjer $\Gamma, x{:}A, p{:}\, a =_A x \vdash P(x, p)$ type in $\Gamma \vdash a{:}A$. Velja

$$\langle \gamma \rangle\, \mathtt{ind\text{-}eq}_a\, (\mathtt{k}\, \gamma)(\mathtt{b}\, \gamma)(\mathtt{p}\, \gamma) \Vdash_{\Pi_\Gamma P(b,p)} \text{ind-eq}_a\, k\, b\, p,$$

kjer

$$\texttt{ind-eq}_a \coloneqq \langle k \rangle \langle \_ \rangle \langle \_ \rangle\, k$$

ter $\texttt{k} \Vdash_{\Pi_\Gamma P(a,\mathrm{refl}_a)} k$, $\texttt{b} \Vdash_{\Pi_\Gamma A} b$ in $\texttt{p} \Vdash_{\Pi_\Gamma a =_A b} p$.

Svet logičnih izjav $\mathbb{P}$

Svet logičnih izjav $\mathbb{P}$ je v teoriji realizabilnosti svet Tarskega $\nabla\mathrm{Per}(\mathbb{K}_1)$ s preslikavo $El\colon \nabla\mathrm{Per}(\mathbb{K}_1) \to \mathrm{Asm}(\mathbb{K}_1)$, kjer $El(P) \coloneqq \iota(P)$ in je ι preslikava, ki smo jo definirali v dokazu leme 8.31. Izjave P sveta $\mathbb{P}$ so tako deri, ki jih lahko dokažemo tako, da zgradimo element $r \in \mathbb{K}_1$ ekvivalenčnega razreda $[r]_P \in El(P)$.

Opomba 9.2. Za vsak skromni skupek M velja $El(o(M)) \cong M$, saj sta kategoriji $\mathrm{Per}(\mathbb{A})$ in $\mathrm{Mod}(\mathbb{A})$ ekvivalentni, kot smo dokazali v lemi 8.31.

Opomba 9.3. Objekti kategorije $\nabla\mathrm{Mod}(\mathbb{K}_1)$ tvorijo pravi razred. Posledično sveta logičnih izjav $\mathbb{P}$ ne moremo interpretirati direktno kot Russellov svet $\nabla\mathrm{Mod}(\mathbb{K}_1)$, saj izraza $\nabla\mathrm{Mod}(\mathbb{K}_1)$ ne moremo zgraditi, ker ∇ sprejme le množice.

Lema 9.4. *Svet $\nabla Per(\mathbb{K}_1)$ je zaprt za kartezične produkte.*

Dokaz. Za vsako družino derov $P\colon I \to \nabla\mathrm{Per}(\mathbb{K}_1)$ lahko tvorimo družino skromnih skupkov $El \circ P$, katere kartezični produkt je $\Pi_I(El \circ P)$, ki je prav tako skromni skupek, saj je $\mathrm{Mod}(\mathbb{K}_1)$ zaprt za kartezične produkte, kot smo pokazali v lemi 8.21. Ker je $\Pi_I(El \circ P)$ skromen, ga lahko preslikamo v der s preslikavo $o\colon \mathrm{Mod}(\mathbb{K}_1) \to \mathrm{Per}(\mathbb{K}_1)$, kot smo jo definirali v 8.31. Dobimo $o(\Pi_r(El \circ P)) \in \nabla\mathrm{Per}(\mathbb{K}_1)$, kjer

$$El(o(\Pi_I(El \circ P)) \cong \Pi_I(El \circ P),$$

saj sta kategoriji $\mathrm{Per}(\mathbb{A})$ in $\mathrm{Mod}(\mathbb{A})$ ekvivalentni. □

Izjave pogosto dokažemo s pomočjo (nedokazanih) predpostavk. Pravimo, da so izjave odvisne od predpostavk, s katerimi jih dokažemo, zato jih po Curry-Howardovi korespondenci predstavimo kot odvisne tipe. Odvisne izjave tako postanejo družine derov, indeksiranih po kontekstu Γ, kjer je kontekst Γ skupek, ki predstavlja predpostavke. Sodbo $\Gamma \vdash P$ type lahko tako razumemo kot $El \circ P \in \mathrm{Fam}(\Gamma)$, kjer $P\colon \Gamma \to \nabla\mathrm{Per}(\mathbb{K}_1)$, in sodbo $\Gamma \vdash p{:}P$ kot $p \in \Pi_\Gamma(El \circ P)$. Izjavo P dokažemo tako, da zgradimo izbirno preslikavo $p \in \Pi_\Gamma(El \circ P)$, ki za dokaze predpostavk $\gamma \in \Gamma$, vrne dokaz $p\,\gamma \in (El \circ P)_\gamma$ izjave P_γ.

Če imamo izjavo $\Gamma, x{:}A \vdash P$ type, kjer $P\colon \Gamma, x{:}A \to \nabla\mathrm{Per}(\mathbb{K}_1)$, lahko zgradimo izjavo

$$\Gamma \vdash \forall_{x:A} P(x) \text{ type},$$

kjer

$$\forall_{x:A} P(x) \coloneqq o(\Pi_{x:A}(El \circ P)_x).$$

Iz dokaza $\Gamma, x{:}A \vdash p\colon P$ lahko dokažemo $\Gamma \vdash p'\colon \forall_{x:A} P(x)$ z uporabo pravila vpeljave

$$\texttt{all-intro} \coloneqq \texttt{trans},$$

tako da

$$\texttt{all-intro}\,\texttt{p} \Vdash_{\Pi_\Gamma \forall_{x:A} P(x)} p',$$

kjer $\mathtt{p} \Vdash_{\Pi_{(\Gamma,x:A)P}} p$ in je `trans` operator, kot smo ga definirali v definiciji 8.20. Iz dokaza $\Gamma \vdash p' : \forall_{x:A} P(x)$ lahko pokažemo, da za vsak $\Gamma \vdash a : A$ velja $\Gamma \vdash p'a : P(a)$, s pravilom uporabe

$$\texttt{all-elim p' a} \coloneqq \texttt{S p' a},$$

tako da

$$\texttt{all-elim p' a} \Vdash_{\Pi_\Gamma P(a)} p'\, a,$$

kjer $\mathtt{p'} \Vdash_{\Pi_\Gamma \forall_{x:A} P(x)} p'$ in $\mathtt{a} \Vdash_{\Pi_\Gamma A} a$.

Z uporabo $\forall$ lahko zgradimo kvantifikator $\exists$, operatorje $\Rightarrow$, $\wedge$, $\vee$ in vrednosti $\top$ ter $\bot$, kot je to opisano v [15, uvod]. Zgradimo le eksistencialni kvantifikator $\exists$. Če imamo izjavo $\Gamma, x{:}A \vdash P$ type, kjer $P{:}\,\Gamma, x{:}A \to \nabla\mathrm{Per}(\mathbb{K}_1)$, lahko zgradimo izjavo:

$$\Gamma \vdash \exists_{x:A} P_x \text{ type},$$

kjer

$$\exists_{x:A} P(x) \coloneqq \forall_{Q:\mathbb{P}} (\forall_{x:A} P(x) \Rightarrow Q) \Rightarrow Q$$

in $Q{:}\,\Gamma \to \nabla\mathrm{Per}(\mathbb{K}_1)$. Iz dokaza $\Gamma \vdash p{:}P(a)$ za nek $\Gamma \vdash a{:}A$ lahko dokažemo $\Gamma \vdash p'{:}\,\exists_{x:A} P(x)$ z uporabo pravila vpeljave

$$\texttt{exists-intro a p} \coloneqq \langle\gamma\rangle\langle Q\rangle\langle l\rangle\, (l\,\gamma)\, (\mathtt{a}\,\gamma)\, (\mathtt{p}\,\gamma),$$

tako da

$$\texttt{exists-intro a p} \Vdash_{\Pi_\Gamma \exists_{x:A} P(x)} p',$$

kjer $\mathtt{a} \Vdash_{\Pi_\Gamma A} a$ in $\mathtt{p} \Vdash_{\Pi_\Gamma P(a)} p$. Iz dokazov $\Gamma \vdash p' : \exists_{x:A} P(x)$ in $\Gamma \vdash l{:}\,\Pi_{x:A} P(x) \to Q$ lahko dokažemo $\Gamma \vdash q{:}\,Q$ z uporabo pravila uporabe

$$\texttt{exists-elim p' l} \coloneqq \langle\gamma\rangle\, \mathtt{p'}\ \gamma\ (\langle\_\rangle\texttt{*})\ \mathtt{l},$$

tako da

$$\texttt{exists-elim p' l} \Vdash_{\Pi_\Gamma Q} q,$$

kjer $\mathtt{p'} \Vdash_{\Pi_\Gamma \exists_{x:A} P(x)} p'$, $\mathtt{l} \Vdash_{\Pi_\Gamma \Pi_{x:A} P(x) \to Q} l$ in $\langle\_\rangle\texttt{*} \Vdash_{\Pi_\Gamma \nabla\mathrm{Per}(\mathbb{K}_1)} Q$.

S tem zaključimo interpretacijo fRIK-a.

9.2 Realizacija CT_Φ in SMN_Φ

V poglavju 9.1 smo podali interpretacijo fRIK-a v teoriji realizabilnosti, kjer smo za delno kombinatorno algebro uporabili $\mathbb{K}_1$. V tem poglavju pokažemo, da znotraj $\mathbb{K}_1$ lahko realiziramo tudi CT_Φ in SMN_Φ, iz česar sledi, da lahko realiziramo tudi SCT in s tem pridobimo interpretacijo fRIK_S v teoriji realizabilnosti.

Elementi $\mathbb{K}_1$ so naravna števila n_f, ki predstavljajo kode izračunljivih delnih funkcij $f{:}\,\mathbb{N} \rightharpoonup \mathbb{N}$, med katerimi se nahajata tudi koda n_o (lema 4.10) univerzalne koračne preslikave in koda n_q (lema 4.11) preslikave q izreka SMN (izrek 4.8). Ker vemo, da koračno preslikavo lahko zgradimo kot Turingov stroj in imamo zanjo kodo n_o, lahko realiziramo CT_Φ (def. 5.1):

$$\Pi f{:}\,\mathbb{N} \to \mathbb{N}.\, \exists n_f{:}\mathbb{N}.\, \forall x{:}\mathbb{N}.\, \exists k{:}\mathbb{N}.\, \Phi^k_{n_f} x = \mathsf{some}(fx),$$

kjer je Φ univerzalna koračna preslikava, kot

$$\langle n_f \rangle \texttt{ exists-intro } n_f \, (\langle x \rangle \texttt{ exists-intro } K \, *),$$

kjer

$$K \coloneqq \texttt{min } (\langle k \rangle \, n_o(n_f, x, k)).$$

Z $n_o(n_f, x, k)$ okrajšamo $n_o(\texttt{pair } n_f \, (\texttt{pair } x \, k))$.

Opomba 9.5. Ko gradimo elemente $\mathbb{K}_1$, pogosto okrajšamo aplikacijo

$$n_g \, (\texttt{pair} \, (\texttt{pair} \, \ldots \, (\texttt{pair } a_1 \, a_2) \, a_3) \, \ldots) \, a_{n-1}) \, a_n)),$$

kot

$$n_g(a_1, a_2, a_3, \ldots, a_{n-1}, a_n),$$

kjer preslikava $g \colon \mathbb{N} \times \cdots \times \mathbb{N} \to \mathbb{N}$ sprejme n argumentov.

Pri izgradnji realizatorja uporabimo kodo koračne preslikave n_o, za katero vemo, da obstaja (lema 4.10). Ker so preslikave $\mathbb{N} \to \mathbb{N}$ v $\mathbb{K}_1$ predstavljene s kodami, nam kode preslikave f ni potrebno iskati, saj je ta že podana kot koda n_f. Preostane nam, da najdemo število korakov k, pri katerem je $\Phi^k_{n_f} x = \mathsf{some}(fx)$: število k pridobimo z izrazom K. Ker je f povsod definirana, vemo, da bo `min` v K zaključil svoje izvajanje v končno mnogo korakih, iz definicije n_o pa vemo, da bo vrednost, ki jo vrne $n_o \, (n_f, x, K)$, enaka $\texttt{succ}(n_f \, x)$. Opazimo, da lahko po definiciji koračne preslikave n_o preslikavo Φ realiziramo kot

$$\texttt{phi} \coloneqq \langle x, y, k \rangle \texttt{ if iszero}(n_o(x, y, k)) \texttt{ none some}(\texttt{pred}(n_o(x, y, k))).$$

Od tod sledi, da je $\texttt{phi}(n_f, x, K)$ vedno enak $\texttt{some}(n_f \, x)$, zato velja

$$\texttt{Id}_{\mathbb{N}} \texttt{ phi}(n_f, x, K) \texttt{ some}(n_f \, x) = \mathbb{1},$$

katerega element lahko vedno realiziramo z `*`.

Podobno lahko realiziramo tudi SMN_ϕ (def. 5.3):

$$\begin{aligned} &\Sigma q \colon \mathbb{N} \to \mathbb{N} \to \mathbb{N}. \, \Pi n x y z \colon \mathbb{N}. \\ &\quad (\exists k \colon \mathbb{N}. \, \Phi^k_{q \, n \, x}(y) = \mathsf{some} \, z) \leftrightarrow (\exists k \colon \mathbb{N}. \, \Phi^k_n \langle x, y \rangle = \mathsf{some} \, z) \end{aligned}$$

kot

$$\texttt{pair } n_q \, (\langle n \rangle \langle x \rangle \langle y \rangle \langle z \rangle \texttt{ pair } I_r \; I_l).$$

Pri tem je n_q koda preslikave q, kot smo jo definirali v 4.8, ter

$$\begin{aligned} I_r &\coloneqq \langle p \rangle \texttt{ exists-intro } K_r \texttt{ *} \text{ in} \\ I_l &\coloneqq \langle p' \rangle \texttt{ exists-intro } K_l \texttt{ *}, \end{aligned}$$

kjer

$$\begin{aligned} K_r &\coloneqq \texttt{min } (\langle k \rangle \; n_o(n, n_p(x, y), k)) \text{ in} \\ K_l &\coloneqq \texttt{min } (\langle k \rangle \; n_o(n_q(n, x), y, k)), \end{aligned}$$

pri čemer je n_p koda združitvene preslikave $\langle \_, \_ \rangle$ (lema 4.12). Ker je $\leftrightarrow$ okrajšava za desno in levo preslikavo, moramo realizirati

$$(\exists k{:}\mathbb{N}.\, \Phi^k_{q\,n\,x}(y) = \mathsf{some}\, z) \to (\exists k{:}\mathbb{N}.\, \Phi^k_n\langle x, y\rangle = \mathsf{some}\, z) \quad \text{in}$$
$$(\exists k{:}\mathbb{N}.\, \Phi^k_{q\,n\,x}(y) = \mathsf{some}\, z) \leftarrow (\exists k{:}\mathbb{N}.\, \Phi^k_n\langle x, y\rangle = \mathsf{some}\, z).$$

Oba izraza realiziramo na podoben način z I_r in I_l: kot pri realizaciji CT_ϕ najdemo števili korakov K_r in K_l, za kateri velja $\Phi^{K_r}_n\langle x, y\rangle = \mathsf{some}\, z$ in $\Phi^{K_l}_{q\,n\,x}(y) = \mathsf{some}\, z$. Ker vemo, da enakost drži, ju lahko realiziramo z $\mathbb{1}$, katerega element je $*$. Ker prejmemo dokaza p in p' izjav $\exists k{:}\mathbb{N}.\, \Phi^k_{q\,n\,x}(y)$ in $\exists k{:}\mathbb{N}.\, \Phi^k_n\langle x, y\rangle$, vemo, da obstajata takšna $k, k'{:}\,\mathbb{N}$, da $\Phi^k_n\langle x, y\rangle \neq \mathsf{none}$ in $\Phi^{k'}_{q\,n\,x}(y) \neq \mathsf{none}$. Zato `min` v K_r in K_l zaključi svoje izvajanje v končno mnogo korakih.

9.3 Interpretacija sveta $\mathbb{P}_{\neg\neg}$

V poglavju 9.1 smo podali interpretacijo fRIK-a. Sedaj pa podamo še interpretacijo sveta $\mathbb{P}_{\neg\neg}$, v katerem velja ZIS. Opazimo, da CT_Φ in SMN_Φ lahko realiziramo tudi znotraj sveta $\mathbb{P}_{\neg\neg}$ z zamenjavo pravil vpeljave in uporabe logičnih operatorjev sveta $\mathbb{P}$ z operatorji sveta $\mathbb{P}_{\neg\neg}$. Od tod sledi, da znotraj $\mathbb{P}_{\neg\neg}$ velja tudi SCT. Tako pridobimo interpretacijo fRIK_{SiZ}.

Svet logičnih izjav $\mathbb{P}_{\neg\neg}$ je v teoriji realizabilnosti Russellov svet $\nabla 2$, kjer je 2 množica skupkov $\{\mathbb{0}, \mathbb{1}\}$. Vsaka izjava je tako skupek $\mathbb{0}$ ali $\mathbb{1}$, odvisno od tega ali izjavo lahko dokažemo. Svet $\mathbb{P}_{\neg\neg}$ interpretiramo kot $\nabla 2$, saj sta $\mathbb{P}_{\neg\neg}$ in $\nabla 2$ izomorfna.

Lema 9.6. *$\mathbb{P}_{\neg\neg}$ in $\nabla 2$ sta izomorfna.*

Dokaz. Za vsako izjavo $P \in \mathbb{P}_{\neg\neg}$ velja $P \Leftrightarrow \neg\neg P$, saj je ta stabilna in lahko dokažemo $P \Rightarrow \neg\neg P$. Sledi, da sta izjavi P in $\neg\neg P$ enako informativni, saj lahko vsako pojavitev izjave P nadomestimo z $\neg\neg P$ in obratno. Opazimo, da za vse $M \in \mathrm{Mod}(\mathbb{K}_1)$, velja

$$\approx_{\neg\neg M} = \begin{cases} \approx_\bot & \text{če } M \cong \bot \text{ in} \\ \approx_\top & \text{če } M \ncong \bot, \end{cases}$$

kar sledi iz definicije negacije, kjer $\neg M \coloneqq M \to \bot$. V primeru, da je $M \cong \bot$, bo $\iota(\approx_M) \to \bot \cong \top$ in zato

$$o((\iota(\approx_M) \to \bot) \to \bot) \cong \approx_\bot,$$

v primeru, da $M \ncong \bot$, pa bo $\iota(\approx_M) \to \bot \cong \bot$ in zato

$$o((\iota(\approx_M) \to \bot) \to \bot) \cong \approx_\top .$$

Ker vemo, da $\forall_{P:\mathbb{P}_{\neg\neg}} P \Leftrightarrow \neg\neg P$, sledi, da za vsak $P{:}\mathbb{P}$:

$$P \cong \approx_\bot \text{ ali } P \cong \approx_\top .$$

Zgradimo izomorfizem $f{:}\,\mathbb{P}_{\neg\neg} \to \nabla 2$, kjer

$$f(P) = \begin{cases} \mathbb{0} & \text{če } P \cong \approx_\bot \text{ in} \\ \mathbb{1} & \text{če } P \cong \approx_\top, \end{cases}$$

ki je trivialno realizabilna. □

Opomba 9.7. Svet $\mathbb{P}_{\neg\neg}$ lahko prav tako interpretiramo kot svet Tarskega $\nabla 2$, kjer $2 \coloneqq \{0, 1\}$, s preslikavo $El(0) \coloneqq \mathbb{0}$ in $El(1) \coloneqq \mathbb{1}$. Vendar pa je priročnejše, če ga interpretiramo kot Russellov svet $\nabla 2$, kjer $2 \coloneqq \{\mathbb{0}, \mathbb{1}\}$, saj se s tem izognemo uporabi preslikave El.

Lema 9.8. *Svet $\nabla 2$ je zaprt za kartezične produkte.*

Dokaz. Za poljuben $A \in Asm(\mathbb{K}_1)$ in družino skupkov $P\colon A \to \nabla 2$ lahko zgradimo kartezični produkt $\Pi_{a:A} P(a)$. Opazimo, da če za vse $a{:}A$ je $P(a) = \mathbb{1}$, potem je $(\Pi_{a:A} P(a)) \cong \mathbb{1}$. Če pa za nek $a{:}A$ velja $P(a) = \mathbb{0}$, potem je $(\Pi_{a:A} P(a)) \cong \mathbb{0}$. V obeh primerih je $\Pi_{a:A} P(a)$ izomorfen elementu $\nabla 2$. □

Eksistenčni kvantifikator definiramo kot

$$\forall_{x:A} P(x) \coloneqq \Pi_{x:A} P(x).$$

Zanj veljajo enaka pravila sklepanja kot za Π.

Lema 9.9. *V $\nabla 2$ velja ZIS.*

Dokaz. ZIS velja v $\nabla 2$, saj lahko za ekvivalentno izjavo $\forall_{(P:\nabla 2)} \neg\neg P \Rightarrow P$ zgradimo realizator njenega elementa: $\langle p \rangle \langle q \rangle *$, kjer $p \Vdash_{\nabla 2} P$ in $q \Vdash_{\neg\neg P} q'$ za nek $q'\colon \neg\neg P$. Ker imamo realizator elementa $\neg\neg P$, vemo, da P ni prazen. Zato je $P \cong \mathbb{1}$ in $*$ realizira element skupka P. □

Literatura

[1] A. Bauer, *Notes on realizability*, [4. 12. 2023], dostopno na `https://www.andrej.com/zapiski/MGS-2022/notes-on-realizability.pdf`.

[2] A. Bauer, *First steps in synthetic computability theory*, Electronic Notes in Theoretical Computer Science **155** (2006) 5–31.

[3] A. Bauer, *Synthetic mathematics with an excursion into computability theory*, 2021, [ogled 3. 9. 2022], dostopno na `https://vimeo.com/510188470`.

[4] D. Bridges, F. Richman in dr., *Varieties of constructive mathematics*, **97**, Cambridge University Press, 1987.

[5] Y. Forster, *Computability in constructive type theory* (2021).

[6] S. C. Kleene, *Recursive predicates and quantifiers*, Transactions of the American Mathematical Society **53**(1) (1943) 41–73.

[7] S. C. Kleene, *On the interpretation of intuitionistic number theory*, The journal of symbolic logic **10**(4) (1945) 109–124.

[8] D. Larchey-Wendling in J.-F. Monin, *The braga method: Extracting certified algorithms from complex recursive schemes in coq*, v: PROOF AND COMPUTATION II: From Proof Theory and Univalent Mathematics to Program Extraction and Verification, World Scientific, 2022, str. 305–386.

[9] U. Norell, *Towards a practical programming language based on dependent type theory*, **32**, Chalmers University of Technology, 2007.

[10] C. Paulin-Mohring, *Introduction to the calculus of inductive constructions*, 2015.

[11] Ž. Putrle, *Consistency of classical synthetic computability theory*, [ogled 23. 11. 2023], dostopno na `https://github.com/zputrle/master-thesis`.

[12] F. Richman, *Church's thesis without tears*, The Journal of symbolic logic **48**(3) (1983) 797–803.

[13] E. Rijke, *Introduction to homotopy type theory*, arXiv preprint arXiv:2212.11082 (2022).

[14] H. Rogers Jr, *Theory of recursive functions and effective computability*, 1987.

[15] T. Streicher, *Semantics of type theory: correctness, completeness and independence results*, Springer Science & Business Media, 1991.

[16] The Agda Team (2023), *The Agda Proof Assistant version 2.6.3*, 2023, [ogled 13. 9. 2023], dostopno na `https://agda.readthedocs.io/en/v2.6.3/index.html`.

[17] The Coq Development Team (2021), *The Coq Proof Assistant version 8.13.2*, 2020, [ogled 24. 8. 2022], dostopno na `https://zenodo.org/record/4501022`.

[18] A. M. Turing in dr., *On computable numbers, with an application to the entscheidungsproblem*, J. of Math **58**(345-363) (1936) 5.