TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Ilja Sobolev

Asünkroonsete algebraliste efektidega programmeerimiskeelte normaliseerimisomadused

Bakalaureusetöö (9 EAP)

Juhendaja:
Danel Ahman, PhD

Tartu 2025

Asünkroonsete algebraliste efektidega programmeerimiskeelte normaliseerimisomadused

Lühikokkuvõte:

Lambda-arvutus on arvutusmudel, mis kirjeldab formaalselt programmide süntaksit ja käitumist. On olemas palju erinevaid lambda-arvutusi, millest igaüks kirjeldab mõnda programmeerimiskeelte aspekti; üks näide on $\lambda_{æ}$, mis keskendub efekte sisaldavate asünkroonsete programmide kirjeldamisele. Selles töös näitame loogiliste relatsioonide meetodit kasutades, et kõik ühest protsessist koosnevad lambda-arvutuse $\lambda_{æ}$ hästi tüübitud programmid termineeruvad. Sellist omadust kutsutakse tugevaks normaliseeruvuseks. Esitatud tõestus demonstreerib, et loogiliste relatsioonide meetod sobib hästi ka asünkroonstee programmide normaliseerimisomaduste uurimiseks. Saadud tulemus näitab, et termineerumist saab garanteerida konkreetsete struktureeritud asünkroonsete programmide jaoks, mis on tihti kasulik selliste süsteemide jaoks, mille juures usaldusväärsus ja etteennustatavus on kriitilised omadused, näiteks sardsüsteemid, reaalajakontrollerid, asjade interneti seadmed ja nutilepingud.

Võtmesõnad: Lambda-arvutus, normaliseeruvus, loogilised relatsioonid, algebralised efektid, asünkroonsus

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Normalisation properties of programming languages with asynchronous algebraic effects

Abstract:

Lambda calculus is a model of computation that formally describes the syntax and behaviour of programs. There are many lambda calculi, each describing some aspect of programming languages; one example being $\lambda_{æ}$, which focuses on describing effectful asynchronous programs. In this thesis we demonstrate how to use the method known as logical relations to show that all well-typed one-process programs in $\lambda_{æ}$ terminate. Such a property is called strong normalisation. The proof demonstrates that the method of logical relations is well-suited for reasoning about normalisation properties of asynchronous programs. The result shows that termination can be guaranteed for certain well-structured asynchronous programs, which might be useful in systems where reliability and predictability are critical, for instance embedded systems, real-time controllers, Internet of things devices and smart contracts.

Keywords: Lambda calculus, normalisation, logical relations, algebraic effects, asynchrony

CERCS: P170 Computer science, numerical analysis, systems, control

Sisukord

Sissejuhatus

Programmeerimiskeelte süntaksi ja käitumise formaalne kirjeldamine ja uurimine aitab vältida vigu nii programmeerimiskeelte disaini kui ka programmide kirjutamise tasemel ning saada parema ülevaate keelte ja arvutuste teoreetilise olemuse kohta. Üks programmeerimiskeelte formaalne mudel, mis ühendab endas nii keele süntaksit kui ka käitumist, on lambda-arvutus (λ-arvutus). Aastate vältel on seda mudelit laiendatud erinevate programmeerimiskeelte aspektide jaoks. Üks hiljutine λ-arvutus on $\lambda_{æ}$ [AP24], mis kirjeldab asünkroonseid programme. See koosneb järjestikusest (ingl *sequential*) osast, mis kirjeldab ühe protsessi käitumist, ja paralleelsest osast, mis kirjeldab protsessidevahelist kommunikatsiooni.

Töö eesmärk on tõestada, et λ-arvutuse $\lambda_{æ}$ järjestikune osa on tugevalt normaliseeruv: omadus, mis tähendab, et kõik ühest protsessist koosnevad hästi tüübitud programmid alati termineeruvad.

Üks meetod tugevalt normaliseeruvuse tõestamiseks põhineb loogilistel relatsioonidel [Tai67; Gir72]. Seda meetodit on laiendatud efekte modelleerivate λ-arvutuste jaoks [LS05]. Siin töös laiendame seda meetodit edasi λ-arvutuse $\lambda_{æ}$ jaoks.

Programmide termineeruvuse garantii on oluline selliste seadmete juures, mille käitumine peab olema etteennustatav ja mis peavad aegsasti lõpliku aja jooksul oma tulemuse tagastama, näiteks sardsüsteemid, reaalajakontrollerid, asjade interneti seadmed ja nutilepingud. Töös esitatud tulemus pakub ka teoreetilise väärtuse: see näitab, et loogiliste relatsioonide meetodit saab kasutada ka asünkroonsete keelte normaliseeruvustõestuste jaoks.

Töö on üles ehitatud järgmiselt.

- Peatükis 1 antakse töö mõistmiseks vajalikud eelteadmised: on selgitatud λ-arvutuse põhimõisteid lihtsalt tüübitud λ-arvutuse näitel; on defineeritud λ-arvutus FGCBV (*fine-grain call-by-value*, [LPT03]), millel omakorda põhineb λ-arvutus $\lambda_{æ}$, mille definitsioon samuti selles peatükis antakse; on defineeritud tugeva normaliseeruvuse mõiste ja kirjeldatud lühidalt loogiliste relatsioonide meetodit selle tõestamiseks.
- Peatükis 2 tõestatakse, et λ-arvutus FGCBV on tugevalt normaliseeruv. Siin tasub märkida, et see tulemus ei ole uus [KLO13]. Adapteeritud tõestus on esitatud eraldi peatükis sel eesmärgil, et detailsemalt tutvustada loogiliste relatsioonide meetodit ja üldiseid matemaatilisi tõestusvõtteid, mis võimaldab peatükis 3 keskenduda vaid λ-arvutuses $\lambda_{æ}$ lisanduvatele uutele asünkroonsusega seotud konstruktsioonidele.
- Peatükis 3 tõestatakse, et λ-arvutus $\lambda_{æ}$ on tugevalt normaliseeruv.
- Peatükis 4 arutletakse töö võimalike edasiarenduste üle.

Peatüki 2 vähemtähtsad tõestused on ära toodud töö lisas A.

Töös esitatud definitsioonid ja tulemused on teaduspraktika raames formaliseeritud ka interaktiivses teoreemitõestajas AGDA [Agda] olemasoleva λ-arvutuse $\lambda_{æ}$ AGDA formalisatsiooni baasil [Ahm24]. Formaliseeritud kood on avaldatud Zenodos [SA25].

1. Teoreetiline ülevaade

Lambda-arvutuse töötas 1932. aastal välja Alonzo Church [Chu32] kui süsteemi matemaatika formaliseerimise jaoks. Hiljem on λ-arvutus rakendust leidnud formaalsete keelte uurimisel kui arvutusmudel, mis võimaldab vaadelda programme induktiivselt defineeritud sümbolite jadadena, mida kutsutakse termideks, ning formaalselt uurida programmide käitumist neile termidele semantika andmise teel. Lambda-arvutus võib olla tüüpimata või tüübitud. Tüübitud λ-arvutused koosnevad induktiivselt defineeritud termidest, tüüpidest, tüüpimisreeglitest, mis seovad omavahel termid ja tüübid, ning induktiivselt defineeritud termide semantikast.

Seletame detailsemalt lahti λ-arvutuse definitsiooni lihtsalt tüübitud λ-arvutuse näitel.

1.1 Lihtsalt tüübitud lambda-arvutus

Lihtsalt tüübitud λ-arvutus on kõige põhilisem tüübitud λ-arvutus, millel teised tüübitud λ-arvutused põhinevad. Toome välja selle komponendid.

Tüübid on kas üks baastüüpidest, mida tähistame tähega b, või funktsioonitüüp:

$$X, Y ::= b \mid X \to Y.$$

Näiteks, kui Nat ja Bool on baastüübid, siis $(\mathrm{Bool} \to \mathrm{Nat}) \to \mathrm{Nat}$ on tüüp. Seda tüüpi termid vastavad funktsioonidele, mis võtavad argumendiks funktsiooni tõeväärtustest naturaalarvudesse ja tagastavad naturaalarvu. Tüüpe tähistame tähtedega X, Y, Z, U.

Term on kas muutuja, λ-abstraktsioon või funktsiooni rakendus argumendile:

$$V, W ::= x \mid \mathsf{fun}\ (x : X) \mapsto V \mid V\,W.$$

Lubame termides üheseltmõistetavuse nimel ka sulge. Näiteks $\mathsf{fun}\ (f : \mathrm{Bool} \to \mathrm{Nat}) \mapsto (f\ z)$ on term. Kui z on mingi Bool-tüüpi muutuja, siis see term tähistab funktsiooni, mis võtab argumendina funktsiooni f tõeväärtustest naturaalarvudesse ja tagastab selle väärtuse, mida tagastab funktsioon f rakendatutuna argumendile z. Allpool näitame, et vaadeldud term on tüüpi $(\mathrm{Bool} \to \mathrm{Nat}) \to \mathrm{Nat}$, nagu oodatud. Termidest võib mõelda kui programmidest või väärtustest:

- $\mathsf{fun}\ (x : X) \mapsto V$ tähistab funktsiooni, mille X-tüüpi parameetrit tähistab muutuja x ja mille keha on term V;
- $V\,W$ tähistab funktsiooni V rakendust argumendile W.

Paneme tähele, et tavaliselt tähistatakse erialases kirjanduses λ-abstraktsioone tähistusega $\lambda x : X.V$, millest tuleneb ka nimi λ-arvutus, kuid loetavuse ja järgmiste peatükkidega kokkusobivuse nimel tähistame selles töös λ-abstraktsioone tähistusega $\mathsf{fun}\ (x : X) \mapsto V$.

Termis esinev muutuja võib olla kas seotud või vaba. Muutuja on seotud, kui ta esineb mingi siduja keha sees. Lihtsalt tüübitud λ-arvutuses on üks siduja: λ-abstraktsioon $\mathsf{fun}\ (x : X) \mapsto V$.

Vastasel juhul on muutuja vaba. Näiteks termis $\mathsf{fun}\ (f : \mathrm{Bool} \to \mathrm{Nat}) \mapsto (f\ z)$ on muutuja f seotud ja muutuja z vaba. Termi V vabade muutujate hulga tähis on $fv(V)$.

Mõnedel termidel on olemas tüüp: selliseid terme nimetatakse hästi tüübitud termideks. Intuitiivselt vastab hästitüübitus sellele, et programm on mingis mõttes korrektne: näiteks funktsioonid on rakendatud ainult oma parameetri tüüpi argumentidele. Asjaolu, et term V on tüüpi X kontekstis Γ tähistatakse $\Gamma \vdash V : X$, kus Γ on muutujate ja neile omistatud tüüpide lõpliku pikkusega nimekiri, st $\Gamma = x_1 : X_1, x_2 : X_2, \ldots, x_n : X_n$. Kontekst Γ on kõikide termi vabade muutujate ja nende tüüpide nimekiri või selle mingi ülemhulk. Termi hästitüübituse mõiste on defineeritud tüüpimisreeglite kaudu. Lihtsalt tüübitud λ-arvutuse tüüpimisreeglid on:

$$\frac{x : X \in \Gamma}{\Gamma \vdash x : X} \qquad \frac{\Gamma \vdash V : X \to Y \qquad \Gamma \vdash W : X}{\Gamma \vdash V\,W : Y} \qquad \frac{\Gamma, x : X \vdash V : Y}{\Gamma \vdash \mathsf{fun}\ (x : X) \mapsto V : X \to Y}$$

Tüüpimisreegel kujul $\dfrac{P_0 \qquad P_1 \qquad \ldots \qquad P_n}{Q}$ tähendab, et eeldustest $P_0, \ldots, P_n$ järeldub Q.

Näiteks λ-abstraktsiooni tüüpimisreegel ütleb, et kui term V on tüüpi Y kontekstis $\Gamma, x : X$, st kontekstis Γ, mille lõppu on lisatud veel üks X-tüüpi muutuja x, siis λ-abstraktsioon $\mathsf{fun}\ (x : X) \mapsto V$ omab kontekstis Γ tüüpi $X \to Y$. Funktsiooni kehas V vabana esinev muutuja x seotakse λ-abstraktsiooni poolt.

Eelnevalt vaadeldud term $\mathsf{fun}\ (f : \mathrm{Bool} \to \mathrm{Nat}) \mapsto (f\ z)$ on hästi tüübitud suvalises kontekstis Γ, mis sisaldab muutujat $z : \mathrm{Bool}$. Seda tunnistab järgmine tüüpimispuu:

$$\dfrac{\dfrac{\dfrac{f : \mathrm{Bool} \to \mathrm{Nat} \in \Gamma, f : \mathrm{Bool} \to \mathrm{Nat}}{\Gamma, f : \mathrm{Bool} \to \mathrm{Nat} \vdash f : \mathrm{Bool} \to \mathrm{Nat}} \qquad \dfrac{z : \mathrm{Bool} \in \Gamma, f : \mathrm{Bool} \to \mathrm{Nat}}{\Gamma, f : \mathrm{Bool} \to \mathrm{Nat} \vdash z : \mathrm{Bool}}}{\Gamma, f : \mathrm{Bool} \to \mathrm{Nat} \vdash f\ z : \mathrm{Nat}}}{\Gamma \vdash \mathsf{fun}\ (f : \mathrm{Bool} \to \mathrm{Nat}) \mapsto (f\ z) : (\mathrm{Bool} \to \mathrm{Nat}) \to \mathrm{Nat}}$$

Kuigi termi hästitüübituse mõiste omab mõtet ainult mingis kontekstis Γ, siis sageli on mingi antud hästi tüübitud termi V korral tema kontekst tuletatav ja iseenesestmõistetav (Γ võib olla kõikide termi V vabade muutujate ja nende tüüpide loetelu), nii et lihtsuse mõttes jätame konteksti tihti implitsiitseks ja kirjutame $\Gamma \vdash V : X$ asemel lihtsalt $V : X$.

Programmide käitumine on kirjeldatud neile semantika andmise kaudu. Leidub palju erinevaid viise, kuidas λ-arvutusele semantikat anda. Selles töös vaatleme väikese sammu operatsioonilist semantikat: selle raames defineeritakse termidevaheline reduktsioonirelatsioon $V \rightsquigarrow W$, mis tähendab, et ühe reduktsioonisammu raames saame asendada termi kujul V termiga kujul W. Kui mingit konkreetset termide redutseerimise strateegiat pole täpsustatud, siis tähendab $V \rightsquigarrow W$ seda, et saame lihtsalt tüübitud λ-arvutuses asendada mingi suurema termi mistahes alamtermi kujul V alamtermiga W.

Vaadeldes terme programmidena, tähendab väide $V \rightsquigarrow W$ seda, et programm V teisendub ühe reduktsioonisammuga programmiks W. Programmi täitmine koosneb siis selliste reduktsioonisammude jadast.

Relatsiooni $\rightsquigarrow$ definitsiooni jaoks on vajalik substitutsiooni mõiste. Kui $\Gamma, x : X \vdash V : Y$ on Y-tüüpi term koos X-tüüpi vaba muutujaga x ja $\Gamma \vdash W : X$ on teine term, siis substitutsioon $V[W/x]$ on term V, kus kõik muutuja x vabad esinemised on asendatud termiga W. Siis kehtib tüüpimisrelatsioon $\Gamma \vdash V[W/x] : Y$. Substitutsiooni formaalse definitsiooni lihtsustamiseks eeldatakse tihti Barendregti konventsiooni (ingl *Barendregt's variable convention*, [Bar84]):

> Kõik vabad muutujad on valitud erinevalt kõigist seotud muutujatest.

Sellise eelduse mõistlikkus on põhjendatud asjaoluga, et seotud muutujat saab alati sobivalt ümber nimetada. Näiteks võib kõiki järgnevaid terme pidada samaväärseteks:

$$\begin{aligned}
&\mathsf{fun}\ (f : \mathrm{Bool} \to \mathrm{Nat}) \mapsto (f\ z)\\
&\mathsf{fun}\ (x : \mathrm{Bool} \to \mathrm{Nat}) \mapsto (x\ z)\\
&\mathsf{fun}\ (w : \mathrm{Bool} \to \mathrm{Nat}) \mapsto (w\ z)\\
&\vdots
\end{aligned}$$

Seega võime iga seotud muutuja jaoks valida sellise nime, et see erineks kõikidest vabadest muutujatest.

Niisiis, substitutsioon defineeritakse induktiivselt järgmiselt:

$$\begin{aligned}
x[W/x] &= W,\\
y[W/x] &= y, \text{ kui } x \neq y,\\
(\mathsf{fun}\ (y : Y) \mapsto V)[W/x] &= \mathsf{fun}\ (y : Y) \mapsto (V[W/x]),\\
(V\ V')[W/x] &= (V[W/x])\,(V'[W/x]).
\end{aligned}$$

Barendregti konventsioon tuleb siin mängu λ-abstraktsiooni substitutsiooni juures: kuna kõik vabad muutujad erinevad seotud muutujatest, siis $x \neq y$ ja väärtuses W ei saa muutuja y vabalt esineda ja seega substitutsiooni käigus ei saa ta λ-abstrakstiooni poolt seotud.

Lihtsalt tüübitud λ-arvutuse korral on relatsioon $\rightsquigarrow$ antud järgmise reduktsioonireegli abil:

$$(\mathsf{fun}\ (x : X) \mapsto V)\ W \rightsquigarrow V[W/x].$$

Näiteks ülaltoodud termi $\mathsf{fun}\ (f : \mathrm{Bool} \to \mathrm{Nat}) \mapsto (f\ z)$ rakendus $(\mathrm{Bool} \to \mathrm{Nat})$-tüüpi funktsioonile $\mathsf{fun}\ (x : \mathrm{Bool}) \mapsto n$, kus n on mingi Nat-tüüpi muutuja, redutseerub järgmiselt:

$$\begin{aligned}
&\big(\mathsf{fun}\ (f : \mathrm{Bool} \to \mathrm{Nat}) \mapsto (f\ z)\big)\ \big(\mathsf{fun}\ (x : \mathrm{Bool}) \mapsto n\big)\\
\rightsquigarrow\ &(f\ z)\big[\big(\mathsf{fun}\ (x : \mathrm{Bool}) \mapsto n\big)/f\big]\\
=\ &\big(\mathsf{fun}\ (x : \mathrm{Bool}) \mapsto n\big)\ z\\
\rightsquigarrow\ &n[z/x]\\
=\ &n.
\end{aligned}$$

Paneme tähele, et kõikide selles töös esitatud λ-arvutuste jaoks kehtivad järgmised väited, mis on tuntud ja mida me eraldi ei tõesta ning mida kasutame edaspidi viitamiseta:

(a) Tüüpide säilitamine (ingl *preservation*): kui $\Gamma \vdash V : X$ ja $V \rightsquigarrow W$, siis $\Gamma \vdash W : X$.

(b) Reduktsiooni stabiilsus substitutsiooni suhtes: olgu $\Gamma, x : X \vdash V : Y$, $\Gamma, x : X \vdash V' : Y$ ja $\Gamma \vdash W : X$. Kui $V \rightsquigarrow V'$, siis $V[W/x] \rightsquigarrow V'[W/x]$.

1.2 Efektidega lambda-arvutus

Lihtsalt tüübitud λ-arvutus võimaldab väljendada ainult puhtaid, efektivabu funktsioone, st selliseid funktsioone, mis sõltuvad ainult oma sisendargumentide väärtustest. Seetõttu erinevad nad reaalsetest arvutiprogrammidest, mis on kasulikud just sel põhjusel, et nad saavad reageerida kasutaja käskudele (näiteks klaviatuuri ja arvutihiire abil) ja väljastada oma töö tulemusi (näiteks ekraanile või faili). Sellist programmidepoolset sisendite ja väljundite kasutamist nimetatakse arvutuslikeks efektideks (edaspidi lihtsalt efektideks).

Ühe matemaatilises mõttes loomuliku viisi efektide λ-arvutuses modelleerimiseks pakkus 1991. aastal välja Eugenio Moggi [Mog91]. Neid ideid kasutab ka λ-arvutus FGCBV (*fine-grain call-by-value*) [LPT03], mida kirjeldame selles alampeatükis. Lambda-arvutus FGCBV on omakorda aluseks λ-arvutusele $\lambda_{æ}$, mille normaliseerimisomadusi uurime peatükis 3. Märgime, et λ-arvutusi FGCBV ja $\lambda_{æ}$ nimetame parema loetavuse nimel edaspidi lihtsalt keelteks FGCBV ja $\lambda_{æ}$.

Arvutuslike efektide paremaks modelleerimiseks eristab keel FGCBV kahte termide klassi: väärtused ja arvutused. Seega on keeles FGCBV ka kaks tüüpimisrelatsiooni: $\Gamma \vdash_{\mathsf{v}} V : X$ ja $\Gamma \vdash_{\mathsf{c}} M : X$. Esimene neist tähendab seda, et V on (kontekstis Γ) X-tüüpi väärtus ja teine tähendab, et M on (kontekstis Γ) arvutus, mis tagastab X-tüüpi väärtusi. Selguse mõttes kasutame tähti M, N, L arvutuste tähistamiseks ja tähti V, W väärtuste tähistamiseks.

Toome välja keele FGCBV tüübid, väärtused ning arvutused. Tüübid:

$$X, Y = b \mid 1 \mid 0 \mid X \times Y \mid X + Y \mid X \to Y.$$

Väärtused ja arvutused:

$$\begin{aligned} V, W ::= {} & x \mid () \mid (V, W) \mid \mathsf{fun}\ (x : X) \mapsto M \mid \mathsf{inl}_Y\, V \mid \mathsf{inr}_X\, W, \\ M, N ::= {} & V\, W \mid \mathsf{return}\, V \mid \mathsf{let}\ x = M\ \mathsf{in}\ N \\ \mid {} & \mathsf{match}\ V\ \mathsf{with}\ \{(x, y) \mapsto M\} \\ \mid {} & \mathsf{match}\ V\ \mathsf{with}\ \{\}_Z \\ \mid {} & \mathsf{match}\ V\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\}. \end{aligned}$$

Järgnevalt selgitame uute tüüpide ja termide tähendust:

- $X \times Y$ on korrutistüüp. Seda tüüpi väärtused on X- ja Y-tüüpi väärtuste järjestatud paarid: kui V ja W on vastavalt X- ja Y- tüüpi väärtused, siis väärtus (V, W) on $(X \times Y)$-tüüpi paar.

- $\mathsf{match}\ V\ \mathsf{with}\ \{(x, y) \mapsto M\}$ on arvutus, mis lõhub $(X \times Y)$-tüüpi paari kaheks tema koostisosaks, mida tähistatakse muutujatega x ja y, ja käivitab järgmisena arvutuse M, mis võib kasutada mõlemat koostisosa läbi muutujate x ja y.
- $X + Y$ on summatüüp. Seda tüüpi väärtus on kas X-tüüpi väärtus või Y-tüüpi väärtus: $\mathsf{inl}_Y\ V$ loob $(X + Y)$-tüüpi termi X-tüüpi termist V ja $\mathsf{inr}_X\ W$ loob $(X + Y)$-tüüpi termi Y-tüüpi termist W.
- $\mathsf{match}\ V\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\}$ on arvutus, mis vaatab, mis tüüpi väärtus sisaldub $(X + Y)$-tüüpi termis V: kui X-tüüpi (vastavalt Y-tüüpi), siis omistatakse see väärtus X-tüüpi (vastavalt Y-tüüpi) muutujale x (vastavalt y), ning seejärel käivitatakse arvutus M (vastavalt N).
- 1 on ühiktüüp. On ainult üks väärtus tüübiga 1: see on ühikväärtus $()$. Ühiktüübi abil võib esitada näiteks tõeväärtustüübi: see on antud ühiktüüpide summana $1 + 1$; sellisel juhul esitatakse tõeväärtused true ja false kui $\mathsf{true} = \mathsf{inl}_1\ ()$ ja $\mathsf{false} = \mathsf{inr}_1\ ()$ (või vastupidi).
- 0 on tühi tüüp. Seda tüüpi väärtusi ei saa kuidagi konstrueerida; tühjast tüübist võib mõelda kui vastuolust. On olemas match-term tühja tüübi jaoks: kui $V : 0$, siis $\mathsf{match}\ V\ \mathsf{with}\ \{\}_Z : Z$. See term vastab loogikareeglile, et vastuolust järeldub kõik. Tühja tüüpi võib kasutada näiteks erindite viskamise väljendamiseks: selleks tuuakse sisse konstruktsioon $\mathsf{throw} : E \to 0$, kus E on mingi erindite hulk.
- $\mathsf{let}\ x = M\ \mathsf{in}\ N$ on arvutus, mis käivitab esimesena arvutuse M, omistab tagastatud väärtuse muutujale x ja käivitab seejärel arvutuse N.
- $\mathsf{return}\ V$ on arvutus, mis tagastab väärtuse V ja ei tee midagi muud.

Juhime tähelepanu, et λ-abstraktsioonid (fun-termid) on väärtused ja funktsioonide rakendused $V\ W$ on arvutused, mis vastab sellele, et funktsiooni hakatakse arvutusena täitma alles pärast tema rakendamist mõnele argumendile. Keele FGCBV tüüpimisreeglid on järgmised:

$$\frac{x : X \in \Gamma}{\Gamma \vdash_{\mathsf{v}} x : X} \qquad \frac{}{\Gamma \vdash_{\mathsf{v}} () : 1} \qquad \frac{\Gamma, x : X \vdash_{\mathsf{c}} M : Y}{\Gamma \vdash_{\mathsf{v}} \mathsf{fun}\ (x : X) \mapsto M : X \to Y}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} V : X \qquad \Gamma \vdash_{\mathsf{v}} W : Y}{\Gamma \vdash_{\mathsf{v}} (V, W) : X \times Y} \qquad \frac{\Gamma \vdash_{\mathsf{v}} V : X}{\Gamma \vdash_{\mathsf{v}} \mathsf{inl}_Y\ V : X + Y} \qquad \frac{\Gamma \vdash_{\mathsf{v}} W : Y}{\Gamma \vdash_{\mathsf{v}} \mathsf{inr}_X\ W : X + Y}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} V : X \to Y \qquad \Gamma \vdash_{\mathsf{v}} W : X}{\Gamma \vdash_{\mathsf{c}} V\ W : Y}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} V : X}{\Gamma \vdash_{\mathsf{c}} \mathsf{return}\ V : X} \qquad \frac{\Gamma \vdash_{\mathsf{c}} M : X \qquad \Gamma, x : X \vdash_{\mathsf{c}} N : Y}{\Gamma \vdash_{\mathsf{c}} \mathsf{let}\ x = M\ \mathsf{in}\ N : Y}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} V : X \times Y \qquad \Gamma, x : X, y : Y \vdash_{\mathsf{c}} M : Z}{\Gamma \vdash_{\mathsf{c}} \mathsf{match}\ V\ \mathsf{with}\ \{(x, y) \mapsto M\} : Z} \qquad \frac{\Gamma \vdash_{\mathsf{v}} V : 0}{\Gamma \vdash_{\mathsf{c}} \mathsf{match}\ V\ \mathsf{with}\ \{\}_Z : Z}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} V : X + Y \qquad \Gamma, x : X \vdash_{\mathsf{c}} M : Z \qquad \Gamma, y : Y \vdash_{\mathsf{c}} N : Z}{\Gamma \vdash_{\mathsf{c}} \mathsf{match}\ V\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\} : Z}$$

Näiteks let-termi tüüpimisreegel ütleb, et kui M on (kontekstis Γ) X-tüüpi arvutus ja N on (kontekstis $\Gamma, x : X$) Y-tüüpi arvutus, siis $\mathsf{let}\ x = M\ \mathsf{in}\ N$ on (kontekstis Γ) Y-tüüpi arvutus. See on kooskõlas intuitsiooniga: arvutuse $\mathsf{let}\ x = M\ \mathsf{in}\ N$ tagastatav väärtus on sama tüüpi, mis viimasena täidetava arvutuse N poolt tagastatav väärtus.

Keele FGCBV semantika erineb lihtsalt tüübitud λ-arvutuse semantikast selle poolest, et reduktsioonireegleid võib rakendada vaid teatud alamtermidele. See modelleerib CBV (*call-by-value*) reduktsioonistrateegiat. Reduktsioonireeglid on defineeritud järgmiselt:

$$(\mathsf{fun}\ (x : X) \mapsto M)\, V \rightsquigarrow M[V/x], \tag{1}$$

$$\mathsf{let}\ x = (\mathsf{return}\ V)\ \mathsf{in}\ M \rightsquigarrow M[V/x], \tag{2}$$

$$\mathsf{match}\ (V, W)\ \mathsf{with}\ \{(x, y) \mapsto M\} \rightsquigarrow M[V/x, W/y], \tag{3}$$

$$\mathsf{match}\ (\mathsf{inl}_Y\, V)\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\} \rightsquigarrow M[V/x], \tag{4}$$

$$\mathsf{match}\ (\mathsf{inr}_X\, W)\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\} \rightsquigarrow N[W/y], \tag{5}$$

$$\mathsf{let}\ y = (\mathsf{let}\ x = M\ \mathsf{in}\ N)\ \mathsf{in}\ L \rightsquigarrow \mathsf{let}\ x = M\ \mathsf{in}\ (\mathsf{let}\ y = N\ \mathsf{in}\ L), \qquad \text{kui } x \notin fv(L), \tag{a}$$

$$\frac{M \rightsquigarrow N}{\mathcal{E}[M] \rightsquigarrow \mathcal{E}[N]},\ \text{kus}$$

$$\mathcal{E} ::= [\,] \mid \mathsf{let}\ x = \mathcal{E}\ \mathsf{in}\ N.$$

$\mathcal{E}$ on siin evalueerimiskontekst (ingl *evaluation context*), mis kirjeldab, millistele alamtermidele võib rakendada reduktsioonireegleid. Näiteks kui kehtib reduktsioon $M \rightsquigarrow N$, siis on evalueerimiskonteksti reegliga võimalik sooritada järgmised reduktsioonisammud:

$$\mathsf{let}\ x = M\ \mathsf{in}\ L \rightsquigarrow \mathsf{let}\ x = N\ \mathsf{in}\ L,$$
$$\mathsf{let}\ y = (\mathsf{let}\ x = M\ \mathsf{in}\ L)\ \mathsf{in}\ L' \rightsquigarrow \mathsf{let}\ y = (\mathsf{let}\ x = N\ \mathsf{in}\ L)\ \mathsf{in}\ L',$$
$$\vdots$$

Selline reduktsioonireeglite esitus tagab näiteks selle, et väärtused kunagi ei redutseeru ja et termis $\mathsf{let}\ x = M\ \mathsf{in}\ N$ võib esimesena redutseeruda ainult term M (ja mitte N), sest selle termi intuitiivse tähenduse kohaselt käivitatakse esimesena M, ja alles pärast seda, kui M tagastab väärtuse V, käivitatakse term $N[V/x]$, kus muutuja x on asendatud termi M poolt tagastatud väärtusega V.

Märgime, et kirjeldatud λ-arvutus ei sisalda konstruktsioone konkreetsete efektide modelleerimiseks, vaid on mõeldud kui baas, mille peale võib lisada erinevaid arvutuslikke efekte ja teisi uusi konstruktsioone. Järgmises peatükis kirjeldatav keel $\lambda_{\text{æ}}$ on üks sellistest keele FGCBV edasiarendustest.

1.3 Asünkroonsete algebraliste efektidega lambda-arvutus

Asünkroonsete algebraliste efektidega keel $\lambda_{æ}$ [AP21; AP24] on keelel FGCBV põhinev keel, mis kombineerib algebralisi efekte [PP02] ja asünkroonsust. See koosneb järjestikusest ja paralleelsest osast. Selles töös käsitleme ainult järjestikust osa, mis kirjeldab ühe eraldiseisva protsessi käitumist.

Keele $\lambda_{æ}$ konstruktsioonid põhinevad olemasoleval ja laialdaselt kasutataval asünkroonsete programmide struktuureerimise viisil, mille nimi on tulevikuväärtused ja lubadused (ingl *futures and promises*). Nende abil võib keeles $\lambda_{æ}$ kirjutada näiteks klient-serveri programme ja modelleerida kaugprotseduurikutseid [AP24:ptk 2.7, ptk 6.3]. Tulevikuväärtused ja lubadused on kasutatud kaasaegsetes programmeerimiskeeltes, nagu näiteks keeles SCALA [AP24:ptk 2.5].

Keele $\lambda_{æ}$ disain on inspireeritud algebraliste efektide poolt, mis on üheks viisiks arvutuslike efektide teoreetiliseks kirjelduseks, ja millel põhineb näiteks arvutuslike efektide käsitlus programmeerimiskeeles OCAML [AP24:ptk 6.2]. Erinevalt tavalisest, sünkroonsest algebraliste efektide käsitlusest, modelleerib keel $\lambda_{æ}$ tõrjuvat multitegumtöötlust (ingl *preemptive multitasking*), mis on lähedasem reaalsetele asünkroonsetele programmidele: see tähendab, et programmi täitmine saab olla katkestatud välise mõju (näiteks teisest protsessist tuleva katkestuse) tagajärjel, ilma katkestatava protsessi enda kooperatsioonita [AP24:ptk 6.2].

Selgitame kõigepealt keele $\lambda_{æ}$ teoreetilist tausta. On antud signaalide ja katkestuste nimede hulk Σ. Tema elemente tähistame tavaliselt $\mathsf{op}, \mathsf{op}', \ldots$. Igale nimele $\mathsf{op} \in \Sigma$ on vastavusse seotud tema kantava väärtuse (ingl *payload*) tüüp A_{op}. Iga programm on modelleeritud lõpliku protsesside hulgana. Iga protsess saab saata signaali $\mathsf{op} \in \Sigma$ mingi argumendiga $V : A_{\mathsf{op}}$. See signaal edastatakse igale teisele protsessile, mis saavad vastava katkestuse. Katkestustele reageerimiseks võib iga protsess paigaldada katkestuste töötlejaid, mis vastava katkestuse saamisel käivitavad mingi määratud arvutuse M ja arvutuse M poolt töö lõpetamisel taaskäivitavad katkestatud programmi arvutuse M poolt tagastatud väärtusega. Samuti on keeles $\lambda_{æ}$ võimalik protsesside täitmist blokeerida selleks, et oodata, kuni protsessile edastatakse mingi teatud katkestus.

Järgnevalt toome välja keele $\lambda_{æ}$ konstruktsioonid. Meenutame, et $\lambda_{æ}$ põhineb keelel FGCBV, st kõik keele FGCBV termid, tüübid ja reduktsioonireeglid (välja arvatud a) sisalduvad keeles $\lambda_{æ}$; nende asemel kirjutame "...". Reduktsioonireegel (a) on keelest $\lambda_{æ}$ ära jäetud põhjusel, et see järeldub teistest reduktsioonireeglitest. Keele $\lambda_{æ}$ väärtused ja arvutused on järgmised:

$$
\begin{aligned}
V, W ::= {} & \ldots \\
\mid {} & \langle V \rangle, \\
M, N ::= {} & \ldots \\
\mid {} & \uparrow \mathsf{op}\,(V, M) \mid \downarrow \mathsf{op}\,(V, M) \\
\mid {} & \mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N \\
\mid {} & \mathsf{await}\ V\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ M.
\end{aligned}
$$

Keele $\lambda_{æ}$ tüübid on jaotatud põhitüüpideks, väärtuste tüüpideks ja arvutuste tüüpideks. Põhitüübid kirjeldavad neid väärtusi, mis võivad olla signaalide ja katkestuste argumentideks. Arvutuste tüübid on kujul $X\ !\ (o, \iota)$, kus o ja ι on efektiannotatsioonid, mis kirjeldavad seda, milliseid signaale ja milliste katkestuste töötlejaid saab antud tüüpi term sisaldada. Selle lõputöö käigus sai tähele pandud, et efektiannotatsioonid ei mõjuta keele normaliseerimisomadusi ja seega vaatame selguse mõttes selles töös keele üldisemat versiooni, milles efektiannotatsioonid on arvutuste tüüpidest välja jäetud. See tähendab muuhulgas seda, et väärtuste tüüpe ja arvutuste tüüpe ei eristata.

Põhitüübid on

$$A, B ::= b \mid 1 \mid 0 \mid A \times B \mid A + B.$$

Väärtuste ja arvutuste tüübid on

$$X, Y ::= A \mid X \times Y \mid X + Y \mid X \to Y \mid \langle X \rangle.$$

Võrreldes keelega FGCBV, lisanduvad järgmised tüüpimisreeglid:

$$\frac{\mathsf{op} \in \Sigma \qquad \Gamma \vdash_{\mathsf{v}} V : A_{\mathsf{op}} \qquad \Gamma \vdash_{\mathsf{c}} M : X}{\Gamma \vdash_{\mathsf{c}} \uparrow \mathsf{op}\,(V, M) : X} \qquad \frac{\mathsf{op} \in \Sigma \qquad \Gamma \vdash_{\mathsf{v}} V : A_{\mathsf{op}} \qquad \Gamma \vdash_{\mathsf{c}} M : X}{\Gamma \vdash_{\mathsf{c}} \downarrow \mathsf{op}\,(V, M) : X}$$

$$\frac{\Gamma \vdash_{\mathsf{v}} V : \langle X \rangle \qquad \Gamma, x : X \vdash_{\mathsf{c}} M : Y}{\Gamma \vdash_{\mathsf{c}} \mathsf{await}\ V\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ M : Y} \qquad \frac{\Gamma \vdash_{\mathsf{v}} V : X}{\Gamma \vdash_{\mathsf{v}} \langle V \rangle : \langle X \rangle}$$

$$\frac{\mathsf{op} \in \Sigma \qquad \Gamma, x : A_{\mathsf{op}} \vdash_{\mathsf{c}} M : \langle X \rangle \qquad \Gamma, p : \langle X \rangle \vdash_{\mathsf{c}} N : Y}{\Gamma \vdash_{\mathsf{c}} \mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N : Y}$$

Selgitame uute termide ja tüüpide tähendust:

- $\uparrow \mathsf{op}\,(V, M)$ on arvutus, mis saadab signaali op argumendiga V, ja jätkab seejärel arvutuse M täitmist.
- $\downarrow \mathsf{op}\,(V, M)$ on arvutus M, millesse on saabunud katkestus op argumendiga V.
- $\mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N$ on arvutus, mis täidab arvutust N kuni saab katkestuse op, mille järel käivitatakse katkestuse töötleja M katkestusega kaasas oleva argumendiga, omistatakse arvutuse M poolt tagastatud väärtus muutujale p ja jätkatakse N täitmist.
- $\langle X \rangle$ on lubaduse tüüp (ingl *promise type*), mis tähistab väärtusi, mis on hetkel tundmatud, aga mis võivad saada tulevikus kättesaadavaks. Seda tüüpi väärtusi tagastavad katkestuste töötlejad. Seda tüüpi väärtusi saab konstrueerida kujul $\langle V \rangle$, kus V on X-tüüpi väärtus. Lubadus on täidetud, kui protsess on katkestatud ja katkestuste töötleja tagastab väärtuse, mis seejärel antakse katkestatud arvutusele. Lubaduse täitmist võib oodata mõne lubaduse tüüpi väärtuse peal await-konstruktsiooni kasutades.
- $\mathsf{await}\ V\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ M$ tähistab programmi eksplitsiitset peatamist (ehk blokeerimist) kuni $\langle X \rangle$-tüüpi väärtus V saab täidetud väärtusega $W : X$, mille järel omistatakse muutujale x väärtus W ja käivitatakse arvutus $M[W/x]$.

Nende termide arvutuslik käitumine on võetud kokku reduktsioonireeglitega [AP24:lk 13], mis on esitatud tabelis 1. Reduktsioonireeglid on jaotatud järgmistesse rühmadesse:

- Esimesed viis reeglit on samad, mis keele FGCBV operatsioonilises semantikas.
- Järgmised kolm reeglit (6)-(8) väljendavad signaalide, katkestuste töötlejate ja await-konstruktsioonide algebralisust: see tähendab, et need kolm termi propageeruvad let-termidest välja.
- Järgmine reegel (9) ütleb, et signaalid propageeruvad väljapoole mööda katkestuste töötlejatest.
- Järgmised viis reeglit (10)-(14) kirjeldavad katkestuste sissepoole levimist: katkestused ei tee midagi return-termiga, propageeruvad signaalide ja await-termide sisse, vastava katkestuse jaoks katkestuste töötleja kohtamisel käivitavad selle katkestuste töötleja ja mingi teise katkestuse jaoks katkestuste töötleja kohtamisel propageeruvad selle sisse.
- Viimane reegel (15) väljendab blokeeritud protsessi jätkamist lubaduse täitumise tulemusena.
- Evalueerimiskonteksti reegel ütleb, et reduktsioonid saavad toimuda let-termide, signaalide, katkestuste ja katkestuste töötlejate sees, kuid mitte await-termi sees, mis vastab intuitsioonile, et await-termiga blokeeritud protsess on peatunud olekus.

Tasub märkida, et reduktsioonireeglite (9), (12) ja (13) juures on hästi tüübitud termide puhul tingimus $p \notin fv(V)$ ülearune: kuna hästi tüübitud väärtuse V tüüp on nendes reeglites põhitüüp, siis V ei saa sisaldada lubaduse tüüpi vaba muutujat p. See tingimus on välja toodud, sest reduktsioonireeglid on defineeritud kõigi termide hulga peal, mitte ainult hästi tüübitud termide peal.

Märgime ka seda, et reduktsioonireeglite (7), (8) ja (14) juures olevad tingimused, et mingi muutuja ei ole vaba mingis termis, on samuti ülearused: nad on eeldatud implitsiitselt, sest kasutame Barendregti konventsiooni.

Toome kahest protsessist koosneva keele $\lambda_{æ}$ programmi näide. Esimene protsess saadab signaali op argumendiga $V : A_{\mathsf{op}}$ ja ei tee midagi muud:

$$\uparrow \mathsf{op}\,(V, \mathsf{return}\,()).$$

Teine protsess koosneb suvalisest arvutusest N, mis peab ootama kuni protsess saab katkestuse op ja arvutusele N on edastatud katkestuse argument V:

$$\mathsf{promise}\,(\mathsf{op}\;x \mapsto \mathsf{return}\,\langle x\rangle)\;\mathsf{as}\;p\;\mathsf{in}\;(\mathsf{await}\;p\;\mathsf{until}\;\langle x\rangle\;\mathsf{in}\;N).$$

Kuna senini oleme defineerinud ainult keele $\lambda_{æ}$ järjestikuse osa, siis kirjeldame lühidalt ka tema paralleelset osa. Keele $\lambda_{æ}$ paralleelne osa ühendab eraldiseisvad järjestikused arvutused

Tabel 1. Keele $\lambda_{æ}$ reduktsioonireeglid ja evalueerimiskontekst

. . . FGCBV reduktsioonireeglid (1)-(5) (ilma reeglita a) . . .

Signaalide, katkestuste töötlejate ja *await*-konstruktsiooni algebralisus

$$\mathsf{let}\ x = (\uparrow \mathsf{op}\,(V, M))\ \mathsf{in}\ N \rightsquigarrow \uparrow \mathsf{op}\,(V, \mathsf{let}\ x = M\ \mathsf{in}\ N) \tag{6}$$

$$\mathsf{let}\ x = (\mathsf{promise}\ (\mathsf{op}\ y \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N_1)\ \mathsf{in}\ N_2 \rightsquigarrow \mathsf{promise}\ (\mathsf{op}\ y \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ (\mathsf{let}\ x = N_1\ \mathsf{in}\ N_2), \quad \text{kui } p \notin fv(N_2) \tag{7}$$

$$\mathsf{let}\ x = (\mathsf{await}\ V\ \mathsf{until}\ \langle y \rangle\ \mathsf{in}\ M)\ \mathsf{in}\ N \rightsquigarrow \mathsf{await}\ V\ \mathsf{until}\ \langle y \rangle\ \mathsf{in}\ (\mathsf{let}\ x = M\ \mathsf{in}\ N), \quad \text{kui } y \notin fv(N) \tag{8}$$

Signaalide ja katkestuste töötlejate omavaheline kommutatiivsus

$$\mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in} \uparrow \mathsf{op}'\,(V, N) \rightsquigarrow \uparrow \mathsf{op}'\,(V, \mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N), \quad \text{kui } p \notin fv(V) \tag{9}$$

Katkestuste levimine

$$\downarrow \mathsf{op}\,(V, \mathsf{return}\ W) \rightsquigarrow \mathsf{return}\ W \tag{10}$$

$$\downarrow \mathsf{op}\,(V, \uparrow \mathsf{op}'\,(W, M)) \rightsquigarrow \uparrow \mathsf{op}'\,(W, \downarrow \mathsf{op}\,(V, M)) \tag{11}$$

$$\downarrow \mathsf{op}\,(V, \mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N) \rightsquigarrow \mathsf{let}\ p = M[V/x]\ \mathsf{in} \downarrow \mathsf{op}\,(V, N), \quad \text{kui } p \notin fv(V) \tag{12}$$

$$\downarrow \mathsf{op}'\,(V, \mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N) \rightsquigarrow \mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in} \downarrow \mathsf{op}'\,(V, N), \quad \text{kui } \mathsf{op} \neq \mathsf{op}' \text{ ja } p \notin fv(V) \tag{13}$$

$$\downarrow \mathsf{op}\,(V, \mathsf{await}\ W\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ M) \rightsquigarrow \mathsf{await}\ W\ \mathsf{until}\ \langle x \rangle\ \mathsf{in} \downarrow \mathsf{op}\,(V, M), \quad \text{kui } x \notin fv(V) \tag{14}$$

Lubaduse täitmise ootamine

$$\mathsf{await}\ \langle V \rangle\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ M \rightsquigarrow M[V/x] \tag{15}$$

Evalueerimiskonteksti reegel

$$\frac{M \rightsquigarrow N}{\mathcal{E}[M] \rightsquigarrow \mathcal{E}[N]},\ \text{kus}$$

$$\mathcal{E} ::= [\,] \mid \mathsf{let}\ x = \mathcal{E}\ \mathsf{in}\ N \mid \uparrow \mathsf{op}\,(V, \mathcal{E}) \mid \downarrow \mathsf{op}\,(V, \mathcal{E}) \mid \mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ \mathcal{E}$$

paralleelseteks protsessideks paralleelse kompositsiooni operaatori $||$ abil. Seega mõlemast protsessist koosnev programm on

$$\uparrow \mathsf{op}\,(V, \mathsf{return}\,()) \,||\, \mathsf{promise}\,(\mathsf{op}\; x \mapsto \mathsf{return}\,\langle x \rangle)\;\mathsf{as}\; p\;\mathsf{in}\;(\mathsf{await}\; p\;\mathsf{until}\;\langle x \rangle\;\mathsf{in}\; N)$$

Demonstreerime nüüd keele $\lambda_{\text{æ}}$ operatsioonilise semantika abil, et see programm käitub nagu eeldatud. Esiteks, esimese protsessi signaal teisendatakse katkestuseks teise programmi jaoks:

$$\rightsquigarrow \mathsf{return}\,() \,||\, \downarrow \mathsf{op}\,(V, \mathsf{promise}\,(\mathsf{op}\; x \mapsto \mathsf{return}\,\langle x \rangle)\;\mathsf{as}\; p\;\mathsf{in}\;(\mathsf{await}\; p\;\mathsf{until}\;\langle x \rangle\;\mathsf{in}\; N))$$

Esimene protsess enam ei muutu, seega vaatame edasi ainult teise protsessi reduktsioone. Järgmisena käivitab saabunud katkestus töötleja koodi (reduktsioonireegel (12)), kus x asendatakse väärtusega V:

$$\begin{aligned} &\rightsquigarrow \mathsf{let}\; p = (\mathsf{return}\,\langle x \rangle)[V/x]\;\mathsf{in}\;(\mathsf{await}\; p\;\mathsf{until}\;\langle x \rangle\;\mathsf{in}\; N) \\ &= \mathsf{let}\; p = \mathsf{return}\,\langle V \rangle\;\mathsf{in}\;(\mathsf{await}\; p\;\mathsf{until}\;\langle x \rangle\;\mathsf{in}\; N) \end{aligned}$$

Kuna katkestuse töötleja tagastab koheselt väärtuse, st on kujul $\mathsf{return}\,\langle V \rangle$, siis on järgmiseks reduktsioonisammuks (2):

$$\begin{aligned} &\rightsquigarrow (\mathsf{await}\; p\;\mathsf{until}\;\langle x \rangle\;\mathsf{in}\; N)[\langle V \rangle / p] \\ &= \mathsf{await}\;\langle V \rangle\;\mathsf{until}\;\langle x \rangle\;\mathsf{in}\; N \end{aligned}$$

Täidetud lubadus võimaldab nüüd jätkata seni blokeeritud arvutuse N täitmist (reduktsioonireegel (15)), kus muutuja x on asendatud väärtusega V:

$$\rightsquigarrow N[V/x].$$

1.4 Normaliseeruvus

Üks sagedasti soovitav tüübitud λ-arvutuste omadus on tugev normaliseeruvus.

Definitsioon 1.1. *Term M on normaalkujul, kui ei leidu teist sellist termi N nii, et $M \rightsquigarrow N$.*

Definitsioon 1.2. *Term M on tugevalt normaliseeruv, kui ei leidu lõpmatut reduktsioonide jada, mis algab termiga M. Kõikide tugevalt normaliseeruvate termide hulka tähistatakse $\mathcal{SN}$.*

Definitsioon 1.3. *Lambda-arvutus on tugevalt normaliseeruv, kui kõik tema hästi tüübitud termid on tugevalt normaliseeruvad.*

Vahetult definitsioonidest järeldub, et term M on tugevalt normaliseeruv parajasti siis, kui iga reduktsioonide jada, mis algab termiga M, jõuab lõpuks normaalkujul termini. Intuitiivselt tähendab tugev normaliseeruvus seda, et igast termist on alati võimalik jõuda tema normaalkujuni, rakendades igal sammul mingit sobivat reduktsioonireeglit. Vaadeldes terme programmidena, tähendab reduktsioonireeglite rakendamine programmi täitmist, seega tugev normaliseeruvus vastab sellele, et iga programm alati termineerub.

Samuti järeldub vahetult definitsioonidest järgmine väide:

Teoreem 1.1. *Term M on tugevalt normaliseeruv parajasti siis, kui iga termi N korral, kui $M \rightsquigarrow N$, siis N on tugevalt normaliseeruv.*

Kui term on tugevalt normaliseeruv, siis kõikide reduktsioonijadade hulgas, mis algavad selle termiga, leidub pikim.

Definitsioon 1.4. *Olgu $M \in \mathcal{SN}$. Siis $max(M)$ on pikima termist M algava reduktsioonijada pikkus.*

On teada, et kehtib järgmine väide.

Teoreem 1.2. *Lihtsalt tüübitud λ-arvutus on tugevalt normaliseeruv.*

Kirjeldame lühidalt ühte meetodit selle väite tõestamiseks, mis põhineb loogiliste relatsioonide mõistel ja mille nimi on Tait-Girardi meetod [How91:ptk 6]. Loogiliste relatsioonide meetodi töötas 1967. aastal välja William Walker Tait [Tai67] ja seda täiendas 1972. aastal Jean-Yves Girard [Gir72]. Tõestus koosneb järgmistest sammudest:

1. Iga tüübi X jaoks defineeritakse induktiivselt teatud X-tüüpi termide hulk $\mathbf{Red}_X$:
   (a) Baastüübi b korral: $V \in \mathbf{Red}_b$, kui V on tugevalt normaliseeruv.
   (b) Funktsioonitüübi $X \to Y$ korral: $V \in \mathbf{Red}_{X\to Y}$, kui iga $W \in \mathbf{Red}_X$ korral $V\,W \in \mathbf{Red}_Y$.

   Hulga $\mathbf{Red}_X$ elemente nimetatakse redutseeritavateks termideks. Kuigi see mõiste kõlab sarnaselt mõistele reduktsioon, tuleb neid kahte mõistet järgnevas eristada.
2. Induktsiooniga tüübi X struktuuri järgi tõestatakse, et kui $V \in \mathbf{Red}_X$, siis $V \in \mathcal{SN}$.
3. Induktsiooniga termi struktuuri järgi tõestatakse järgmine väide, mida nimetatakse loogiliste relatsioonide fundamentaalteoreemiks:

   > Olgu $V : X$ term, mille kõik vabad muutujad on $x_1 : X_1, \ldots, x_n : X_n$. Olgu $W_1 \in \mathbf{Red}_{X_1}, \ldots, W_n \in \mathbf{Red}_{X_n}$, siis $V[W_1/x_1, \ldots, W_n/x_n] \in \mathbf{Red}_X$.

Võttes sammu 3 väites $W_1 = x_1, \ldots, W_n = x_n$ (loomulikult tuleb enne näidata, et muutujad on redutseeritavad) ja rakendades sammu 2 saame, et iga hästi tüübitud term on tugevalt normaliseeruv, mis tähendab, et lihtsalt tüübitud λ-arvutus on tugevalt normaliseeruv.

Märgime, et teoreemi 1.2 ei saa tõestada lihtsa induktsiooniga termi struktuuri järgi: raske juht on funktsiooni rakendus $V\,W$. Raskus seisneb selles, et kui $V = \mathsf{fun}\ (x : X) \mapsto V'$, siis kehtib reduktsioon $V\,W \rightsquigarrow V'[W/x]$, mille käigus võib term W saada suvaliselt palju kordi kopeeritud, mille tõttu ei saa me järeldada termi $V'[W/x]$ tugevalt normaliseeruvust termide V ja W tugevalt normaliseeruvusest.

Loogiliste relatsioonide meetodit on laiendatud 2005. aastal Sam Lindley ja Ian Starki poolt ka efektidega λ-arvutustele [LS05]. Selles lõputöös vaatame esmalt, kuidas seda laiendust rakendada keelele FGCBV, ning seejärel arendame seda edasi keele $\lambda_{\text{æ}}$ jaoks.

2. Keele FGCBV tugevalt normaliseeruvus

Selles peatükis on välja toodud keele FGCBV tugeva normaliseeruvuse tõestus, kasutades Lindley ja Starki artiklis [LS05] esitatud meetodi adaptatsiooni. Nagu mainitud sissejuhatuses, siis see tulemus ei ole uus: näiteks sisalduvad kõik keele FGCBV termid ja reduktsioonireeglid keeles λ_{eff}, mis defineeriti ja mille jaoks tõestati tugev normaliseeruvus Kammari jt poolt 2013. aastal [KLO13].

2.1 Redutseeritavuse definitsioon

Et laiendada eelmises peatükis kirjeldatud tõestusmeetodit keelele FGCBV, tuleb defineerida redutseeritavus ka arvutuste jaoks. Üks viis selle tegemiseks on $\top\top$-*lifting* [LS05]. See meetod põhineb jätkudel, millest võib mõelda kui let-termidest koosnevatest programmi ümbritsevatest kontekstidest, nagu näiteks

$$\mathsf{let}\ x = (\mathsf{let}\ y = (-)\ \mathsf{in}\ M)\ \mathsf{in}\ L,$$

kus $(-)$ on "auk", millesse võib asendada sobivat tüüpi arvutusi. Järgnevalt defineerime jätkud ja nendega seotud mõisted formaalselt, adapteerides Lindley ja Starki artikli definitsiooni [LS05:lk 6] keele FGCBV jaoks ja pöörates rohkem tähelepanu jätkude tüüpimisele.

Term-abstraktsioon $(x)N$ on arvutus N koos eristatud (termis N) vaba muutujaga x. Term-abstraktsioonide tüüpi määrab tüüpimisrelatsioon $\Gamma \vdash (x)N : X \multimap Y$, mis on samaväärne väitega $\Gamma, x : X \vdash_{\mathsf{c}} N : Y$. Analoogiliselt defineeritakse ka term-abstraktsioone kahe eristatud (termis N) vaba muutujaga: $\Gamma \vdash (x, y)N : [X, Y] \multimap Z$, mis on samaväärne väitega $\Gamma, x : X, y : Y \vdash_{\mathsf{c}} N : Z$.

Jätk K on lõplik term-abstraktsioonide jada:

$$K ::= \mathit{Id} \mid K \circ (x)N.$$

Jätku K pikkus $|K|$ on selles jätkus esinevate term-abstraktsioonide arv:

$$|\mathit{Id}| = 0 \qquad |K \circ (x)N| = |K| + 1.$$

Jätku tüübi määravad järgmised tüüpimisreeglid:

$$\frac{}{\Gamma \vdash \mathit{Id} : X \multimap X} \qquad \frac{\Gamma \vdash (x)N : X \multimap Y \qquad \Gamma \vdash K : Y \multimap Z}{\Gamma \vdash K \circ (x)N : X \multimap Z}$$

Jätku $\Gamma \vdash K : X \multimap Y$ rakendus termile $\Gamma \vdash_{\mathsf{c}} M : X$ on term, mis on saadud termi M asetamisel auku $(-)$. Saadavat termi tähistatakse $K \mathbin{@} M$ ja ta on defineeritud induktiivselt kui

$$\mathit{Id} \mathbin{@} M = M,$$
$$(K \circ (x)N) \mathbin{@} M = K \mathbin{@} (\mathsf{let}\ x = M\ \mathsf{in}\ N).$$

Märgime, et jätku rakendamine termile omab mõtet ainult siis, kui jätku ja termi tüüpimiskontekstid Γ on samad.

Jätkude jaoks saab defineerida ka reduktsioonirelatsiooni: kui K, K' on jätkud nii, et $\Gamma \vdash K : X \multimap Y$ ja $\Gamma \vdash K' : X \multimap Y$, siis

$$K \rightsquigarrow K' \Leftrightarrow \text{iga } \Gamma \vdash_{\mathsf{c}} M : X \text{ korral } K \mathbin{@} M \rightsquigarrow K' \mathbin{@} M.$$

Kuna väärtused ja arvutused on keeles FGCBV selgelt eristatud, siis tähistame X-tüüpi redutseeritavate väärtuste hulka $\mathbf{VRed}_X$ ja X-tüüpi redutseeritavate arvutuste hulka $\mathbf{CRed}_X$. Lisaks sellele defineerime redutseeritavuse ka jätkudel: $(X \multimap Y)$-tüüpi redutseeritavate jätkude hulka tähistame $\mathbf{KRed}_X$. Arvutuste ja jätkude redutseeritavuse defineerime järgmiselt:

$$M \in \mathbf{CRed}_X \Leftrightarrow \text{iga } K \in \mathbf{KRed}_X \text{ korral } K \mathbin{@} M \in \mathcal{SN},$$
$$K \in \mathbf{KRed}_X \Leftrightarrow \text{iga } V \in \mathbf{VRed}_X \text{ korral } K \mathbin{@} (\mathsf{return}\, V) \in \mathcal{SN}.$$

Intuitiivselt tähendab hulk $\mathbf{KRed}_X$ programmi ümbritsevaid kontekste, mis on mõnes mõttes tugevalt normaliseeruvad. Selle kohaselt on programm $M : X$ redutseeritav, kui ta on tugevalt normaliseeruv suvalises ümbritsevas tugevalt normaliseeruvas kontekstis.

Järgnevalt kirjeldame Lindley ja Starki tõestuse põhjal [LS05] summatüüpi väärtuste redutseeritavust. Selleks kasutatakse summajätke S, mis on jätkud, mille lõppu on lisatud match-term:

$$S ::= K \circ ((x)M, (y)N)_+.$$

Summajätku tüübi tähistame $[X+Y] \multimap Z$. Nurksulud on lisatud selleks, et eristada summajätku tüüpi ja tavalise jätku tüüpi $X+Y \multimap Z$, mille auk on tüüpi $X+Y$. Summajätku tüüpimisreegel on defineeritud kui

$$\frac{\Gamma \vdash K : Z \multimap U \qquad \Gamma \vdash (x)M : X \multimap Z \qquad \Gamma \vdash (y)N : Y \multimap Z}{\Gamma \vdash K \circ ((x)M, (y)N)_+ : [X+Y] \multimap U}$$

ja summajätku rakendus väärtusele $V : X + Y$ on defineeritud kui

$$(K \circ ((x)M, (y)N)_+) \mathbin{@} V = K \mathbin{@} (\mathsf{match}\ V\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\}).$$

Rõhutame, et summajätke rakendatakse väärtustele, mitte arvutustele, nagu tavaliste jätkude juures. Intuitiivselt tähistavad summajätkud programme ümbritsevaid kontekste kujul

$$\mathsf{let}\ x = (\mathsf{let}\ y = (\mathsf{match}\ (-)\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\})\ \mathsf{in}\ M)\ \mathsf{in}\ L,$$

mille rakendus väärtusele V on selle väärtuse asetamine auku $(-)$. Redutseeritavate $([X+Y] \multimap Z)$-tüüpi summajätkude hulka tähistame $\mathbf{SRed}_{X,Y}$. Summatüüpi väärtuste redutseeritavus defineeritakse summajätkude abil:

$$V \in \mathbf{VRed}_{X+Y} \Leftrightarrow \text{iga } S \in \mathbf{SRed}_{X,Y} \text{ korral } S \mathbin{@} V \in \mathcal{SN},$$
$$S \in \mathbf{SRed}_{X,Y} \quad \Leftrightarrow \begin{cases} \text{iga } V \in \mathbf{VRed}_X \text{ korral } S \mathbin{@} (\mathsf{inl}_Y\, V) \in \mathcal{SN}, \text{ ja} \\ \text{iga } W \in \mathbf{VRed}_Y \text{ korral } S \mathbin{@} (\mathsf{inr}_X\, W) \in \mathcal{SN}. \end{cases}$$

Analoogiliselt summatüüpi väärtustega, defineerime redutseeritavuse ka korrutistüüpi väärtuste jaoks läbi korrutisjätku P mõiste:

$$P ::= K \circ ((x,y)N)_\times$$

$$\frac{\Gamma \vdash K : Z \multimap U \qquad \Gamma \vdash (x,y)N : [X,Y] \multimap Z}{\Gamma \vdash K \circ ((x,y)N)_\times : [X \times Y] \multimap U}$$

$$(K \circ ((x,y)N)_\times) \mathbin{@} V = K \mathbin{@} (\mathsf{match}\ V\ \mathsf{with}\ \{(x,y) \mapsto N\}).$$

Redutseeritavate $([X \times Y] \multimap Z)$-tüüpi korrutisjätkude hulka tähistame $\mathbf{PRed}_{X,Y}$. Korrutistüüpi väärtuste redutseeritavus on siis defineeritud kui

$$V \in \mathbf{VRed}_{X \times Y} \Leftrightarrow \text{iga } P \in \mathbf{PRed}_{X,Y} \text{ korral } P \mathbin{@} V \in \mathcal{SN},$$
$$P \in \mathbf{PRed}_{X,Y} \Leftrightarrow \text{iga } V \in \mathbf{VRed}_X \text{ ja } W \in \mathbf{VRed}_Y \text{ korral } P \mathbin{@} (V,W) \in \mathcal{SN}.$$

Kuna keeles FGCBV väärtused ei redutseeru, siis on need triviaalselt tugevalt normaliseeruvad, mille tõttu võime võtta kõik baastüüpi väärtused redutseeritavateks.

Lõpuks, funktsioonitüüpi väärtuse redutseeritavuse definitsioon jääb põhimõtteliselt samaks nagu lihtsalt tüübitud λ-arvutuse juures, arvestades nüüd ainult sellega, et funktsiooni rakendus on arvutus:

$$V \in \mathbf{VRed}_{X \to Y} \Leftrightarrow \text{iga } W \in \mathbf{VRed}_X \text{ korral } V\,W \in \mathbf{CRed}_Y.$$

Redutseeritavuse definitsioon keelele FGCBV on kokku võetud järgmiselt:

$$V \in \mathbf{VRed}_b \qquad \text{kehtib alati,}$$
$$V \in \mathbf{VRed}_1 \qquad \text{kehtib alati,}$$
$$V \in \mathbf{VRed}_0 \qquad \text{kehtib alati,}$$
$$V \in \mathbf{VRed}_{X \times Y} \Leftrightarrow \text{iga } P \in \mathbf{PRed}_{X,Y} \text{ korral } P \mathbin{@} V \in \mathcal{SN},$$
$$V \in \mathbf{VRed}_{X+Y} \Leftrightarrow \text{iga } S \in \mathbf{SRed}_{X,Y} \text{ korral } S \mathbin{@} V \in \mathcal{SN},$$
$$V \in \mathbf{VRed}_{X \to Y} \Leftrightarrow \text{iga } W \in \mathbf{VRed}_X \text{ korral } V\,W \in \mathbf{CRed}_Y,$$
$$M \in \mathbf{CRed}_X \Leftrightarrow \text{iga } K \in \mathbf{KRed}_X \text{ korral } K \mathbin{@} M \in \mathcal{SN},$$
$$K \in \mathbf{KRed}_X \Leftrightarrow \text{iga } V \in \mathbf{VRed}_X \text{ korral } K \mathbin{@} (\mathsf{return}\ V) \in \mathcal{SN},$$
$$P \in \mathbf{PRed}_{X,Y} \Leftrightarrow \text{iga } V \in \mathbf{VRed}_X \text{ ja } W \in \mathbf{VRed}_Y \text{ korral } P \mathbin{@} (V,W) \in \mathcal{SN},$$
$$S \in \mathbf{SRed}_{X,Y} \Leftrightarrow \begin{cases} \text{iga } V \in \mathbf{VRed}_X \text{ korral } S \mathbin{@} (\mathsf{inl}_Y\ V) \in \mathcal{SN}, \text{ ja} \\ \text{iga } W \in \mathbf{VRed}_Y \text{ korral } S \mathbin{@} (\mathsf{inr}_X\ W) \in \mathcal{SN}. \end{cases}$$

2.2 Redutseeritavusest järeldub tugev normaliseeruvus

Tait-Girardi meetodi kohaselt peame järgmise sammuna näitama, et kõik redutseeritavad termid on tugevalt normaliseeruvad. Väärtuste korral on see väide ilmne, sest väärtused keeles FGCBV ei redutseeru. Arvutuste korral näitab seda väidet järgmine teoreem.

Teoreem 2.1. *Iga* $M \in \mathbf{CRed}_X$ *korral* $M \in \mathcal{SN}$.

Tõestus. Olgu $M \in \mathbf{CRed}_X$. Paneme tähele, et iga $V \in \mathbf{VRed}_X$ korral $\mathit{Id} \mathbin{@} (\mathsf{return}\, V) = \mathsf{return}\, V \in \mathcal{SN}$. Seega $\mathit{Id} \in \mathbf{KRed}_X$. Kuna $M \in \mathbf{CRed}_X$ ja $\mathit{Id} \in \mathbf{KRed}_X$, siis $M = \mathit{Id} \mathbin{@} M \in \mathcal{SN}$. □

2.3 Iga hästi tüübitud term on redutseeritav

Ülejäänud tõestus on pühendatud sellele, et näidata, et iga hästi tüübitud term on redutseeritav. Selleks tõestame, et redutseeritavatest termidest konstrueeritud termid on redutseeritavad. Esmalt käsitleme kõiki väärtuste kujusid, ja seejärel arvutuste kujusid. Paljudes lemmades tuleb kasuks järgmine tulemus, mis ütleb, et jätku redutseerimine vähendab selle pikkust. Seda tulemust kasutatakse peaaegu igas lemmas, nii et kasutame seda edaspidi viitamiseta.

Lemma 2.2. *Olgu* $K, K' : X \multimap Y$ *jätkud nii, et* $K \rightsquigarrow K'$. *Siis* $|K'| < |K|$.

Tõestus. Ainus võimalik reduktsioon on (a), mille tulemusena kaks järjestikust term-abstraktsiooni sulavad ühte:

$$K = \mathit{Id} \circ \cdots \circ (x)M \circ (y)N \circ \cdots,$$
$$K' = \mathit{Id} \circ \cdots \circ (y)(\mathsf{let}\ x = N\ \mathsf{in}\ M) \circ \cdots,$$

nii et $|K'| = |K| - 1$. □

Väärtuste kujud on muutuja, λ-abstrakstioon, paar, injektsioonid ja ühikväärtus. Nendest viimase kolme redutseeritavus järeldub vahetult redutseeritavuse definitsioonist:

Lemma 2.3. *Olgu* $V \in \mathbf{VRed}_X$ *ja* $W \in \mathbf{VRed}_Y$. *Siis* $(V, W) \in \mathbf{VRed}_{X \times Y}$.

Lemma 2.4. *Olgu* $V \in \mathbf{VRed}_X$ *ja* $W \in \mathbf{VRed}_Y$. *Siis* $\mathsf{inl}_Y\, V \in \mathbf{VRed}_{X+Y}$ *ja* $\mathsf{inr}_X\, W \in \mathbf{VRed}_{X+Y}$.

Lemma 2.5. *Ühikväärtus on redutseeritav:* $() \in \mathbf{VRed}_1$.

Funktsioonide redutseeritavust näitab järgmine lemma.

Lemma 2.6. *Olgu* $(x)N : X \multimap Y$ *term-abstraktsioon nii, et* $N[W/x] \in \mathbf{CRed}_Y$ *iga* $W \in \mathbf{VRed}_X$ *korral. Siis* $\mathsf{fun}\ (x : X) \mapsto N \in \mathbf{VRed}_{X \to Y}$.

Tõestus. Olgu $W \in \mathbf{VRed}_X$. Peame näitama, et $(\mathsf{fun}\ (x : X) \mapsto N)\, W \in \mathbf{CRed}_Y$, ehk et $K \mathbin{@} ((\mathsf{fun}\ (x : X) \mapsto N)\, W) \in \mathcal{SN}$, iga $K \in \mathbf{KRed}_Y$ korral. Näitame induktiooniga $|K|$ järgi, et term $K \mathbin{@} ((\mathsf{fun}\ (x : X) \mapsto N)\, W)$ redutseerub ainult tugevalt normaliseeruvateks termideks. Võimalikud reduktsioonid on järgmised (siin ja edaspidises lisame iga juhu ette vastava reduktsioonireegli numbri, kui see juht vastab mingile konkreetsele reduktsioonireeglile):

(a) $\rightsquigarrow K' \mathbin{@} ((\mathsf{fun}\ (x : X) \mapsto N)\, W)$, kui $K \rightsquigarrow K'$: siis $|K'| < |K|$ ja seega induktsiooni hüpoteesi põhjal $K' \mathbin{@} ((\mathsf{fun}\ (x : X) \mapsto N)\, W) \in \mathcal{SN}$;

(1) $\rightsquigarrow K \mathbin{@} (N[W/x])$. Kuna $W \in \mathbf{VRed}_X$, siis $N[W/x] \in \mathbf{CRed}_Y$ lemma eelduse põhjal. Järelikult, hulga $\mathbf{CRed}_Y$ definitsiooni põhjal, saame $K \mathbin{@} (N[W/x]) \in \mathcal{SN}$. □

Jääb näidata, et muutujad on redutseeritavad.

Lemma 2.7. *Olgu $z : Z$. Siis $z \in \mathbf{VRed}_Z$.*

Tõestus. Vaatleme juhte sõltuvalt tüübi Z kujust:

- $Z = b$: otse $\mathbf{VRed}_b$ definitsioonist järeldub, et $z \in \mathbf{VRed}_b$.
- $Z = X \to Y$: olgu $V \in \mathbf{VRed}_X$. Peame näitama, et $z\,V \in \mathbf{CRed}_Y$. Olgu $K \in \mathbf{KRed}_Y$. Induktsiooniga $|K|$ järgi näitame, et termi $K @ (z\,V) \in \mathcal{SN}$ kõik reduktsioonid on tugevalt normaliseeruvad. On võimalik ainult üks reduktsioon:
  (a) $\rightsquigarrow K' \mathbin{@} (z\,V)$, kui $K \rightsquigarrow K'$: siis $|K'| < |K|$ ja seega induktsiooni hüpoteesi põhjal $K' \mathbin{@} (z\,V) \in \mathcal{SN}$.
- $Z = X \times Y$: olgu $P \in \mathbf{PRed}_{X,Y}$. Siis $P = K \circ ((x, y)N)_\times$ mingi jätku K korral. See, et $P \mathbin{@} z \in \mathcal{SN}$ järeldub induktsioonist $|K|$ järgi nagu eelmises juhus.
- $Z = X + Y$: olgu $S \in \mathbf{SRed}_{X,Y}$. Siis $S = K \circ ((x)M, (y)N)_+$ mingi jätku K korral. See, et $S \mathbin{@} z \in \mathcal{SN}$, järeldub induktsioonist $|K|$ järgi nagu eelmises kahes juhus. □

Nüüd liigume arvutuste juurde. Arvutused on return- ja let-termid, match-termid summa- ja korrutistüüpide jaoks, ja funktsioonide rakendused.

Lemma 2.8. *Olgu $V \in \mathbf{VRed}_X$. Siis* return $V \in \mathbf{CRed}_X$.

Tõestus. Olgu $K \in \mathbf{KRed}_X$. Peame näitama, et $K \mathbin{@} (\mathsf{return}\ V) \in \mathcal{SN}$. See aga järeldub otse hulga $\mathbf{KRed}_X$ definitsioonist. □

Järgmine lemma ja selle tõestus on peaaegu samad, mis Lemma 7 tõestus Lindley ja Starki artiklis [LS05]. See lemma väidab, et teatud tingimusi rahuldav term-abstraktsioon võib olla lükatud jätku sisse. Seda tulemust kasutatakse siin lemma 2.10 tõestuse juures.

Lemma 2.9. *Olgu $(x)N : X \multimap Y$ term-abstraktsioon ja $K \in \mathbf{KRed}_Y$ jätk nii, et $K \mathbin{@} (N[V/x]) \in \mathcal{SN}$ iga $V \in \mathbf{VRed}_X$ jaoks. Siis $K \circ (x)N \in \mathbf{KRed}_X$.*

Tõestus. Lisas A.1. □

Nüüd on võimalik tõestada, et redutseeritavatest termidest koosnev let-term on redutseeritav. Seda teeme let-termi jätku sisse lükkamise teel:

Lemma 2.10. *Olgu $M \in \mathbf{CRed}_X$ ja $(x)N : X \multimap Y$ term-abstraktsioon nii, et $N[V/x] \in \mathbf{CRed}_Y$ iga $V \in \mathbf{VRed}_X$ korral. Siis* let $x = M$ in $N \in \mathbf{CRed}_Y$.

Tõestus. Olgu $K \in \mathbf{KRed}_Y$. Peame näitama, et $K \mathbin{@} (\mathsf{let}\ x = M\ \mathsf{in}\ N) \in \mathcal{SN}$. Kuna $K \mathbin{@} (\mathsf{let}\ x = M\ \mathsf{in}\ N) = (K \circ (x)N) \mathbin{@} M$ ja $M \in \mathbf{CRed}_X$, siis selleks piisab, kui näidata, et $K \circ (x)N \in \mathbf{KRed}_X$. Nüüd $K \mathbin{@} (N[V/x]) \in \mathcal{SN}$ iga $V \in \mathbf{VRed}_X$ jaoks, sest $K \in \mathbf{KRed}_Y$ ja lemma eelduse põhjal $N[V/x] \in \mathbf{CRed}_Y$. Järelikult lemma 2.9 põhjal $K \circ (x)N \in \mathbf{KRed}_X$, nagu soovitud. □

Järgnevalt näitame, et redutseeritavatest termidest koosnev match-term summatüübi jaoks on redutseeritav. Selleks näitame esmalt, et match-termi lükkamine redutseeritavasse jätku säilitab redutseeritavust, st moodustatud summajätk on redutseeritav:

Lemma 2.11. *Olgu $V : X$ väärtus, $(x)M : X \multimap Z$, $(y)N : Y \multimap Z$ term-abstraktsioonid ja $K : Z \multimap U$ jätk nii, et $K \mathbin{@} (M[V/x]) \in \mathcal{SN}$. Siis $K \mathbin{@} (\mathsf{match}\ (\mathsf{inl}_Y\, V)\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\}) \in \mathcal{SN}$.*

Tõestus. Näitame induktsiooniga $|K|$ järgi, et kõik termi $K \mathbin{@} (\mathsf{match}\ (\mathsf{inl}_Y\, V)\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\})$ reduktsioonid on tugevalt normaliseeruvad. Võimalikud reduktsioonid on järgmised:

(4) $\rightsquigarrow K \mathbin{@} (M[V/x])$: see on tugevalt normaliseeruv lemma eelduse põhjal.

(a) $\rightsquigarrow K' \mathbin{@} (\mathsf{match}\ (\mathsf{inl}_Y\, V)\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\})$, kui $K \rightsquigarrow K'$: siis $|K'| < |K|$ ja $K' \mathbin{@} (M[V/x]) \in \mathcal{SN}$. Induktsiooni hüpoteesi põhjal saame, et $K' \mathbin{@} (\mathsf{match}\ (\mathsf{inl}_Y\, V)\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\}) \in \mathcal{SN}$. □

Sama lemma $\mathsf{inr}_X\, W$ jaoks tõestatakse analoogselt:

Lemma 2.12. *Olgu $W : Y$ väärtus, $(x)M : X \multimap Z$, $(y)N : Y \multimap Z$ term-abstraktsioonid ja $K : Z \multimap U$ jätk nii, et $K \mathbin{@} (N[W/y]) \in \mathcal{SN}$. Siis $K \mathbin{@} (\mathsf{match}\ (\mathsf{inr}_X\, W)\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\}) \in \mathcal{SN}$.*

Lemma summatüübi jaoks tõestame match-termi jätku lükkamise teel:

Lemma 2.13. *Olgu $V \in \mathbf{VRed}_{X+Y}$, $(x)M : X \multimap Z$ ja $(y)N : Y \multimap Z$ nii, et $M[V/x] \in \mathbf{CRed}_Z$ iga $V \in \mathbf{VRed}_X$ korral ja $N[W/y] \in \mathbf{CRed}_Z$ iga $W \in \mathbf{VRed}_Y$ korral. Siis $(\mathsf{match}\ V\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\}) \in \mathbf{CRed}_Z$.*

Tõestus. Olgu $K \in \mathbf{KRed}_Z$. Peame näitama, et $K \mathbin{@} (\mathsf{match}\ V\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\}) \in \mathcal{SN}$. Kuna $(\mathsf{match}\ V\ \mathsf{with}\ \{\mathsf{inl}\, x \mapsto M, \mathsf{inr}\, y \mapsto N\}) = (K \circ ((x)N, (y)M)_+) \mathbin{@} V$ ja $V \in \mathbf{VRed}_{X+Y}$, siis selleks piisab, kui $K \circ ((x)N, (y)M)_+ \in \mathbf{SRed}_{X,Y}$, ehk et kui $(K \circ ((x)N, (y)M)_+) \mathbin{@} (\mathsf{inl}_Y\, V) \in \mathcal{SN}$ iga $V \in \mathbf{VRed}_X$ korral ja $(K \circ ((x)N, (y)M)_+) \mathbin{@} (\mathsf{inr}_X\, W) \in \mathcal{SN}$ iga $W \in \mathbf{VRed}_Y$ korral. Mõlemad väited kehtivad vastavalt lemma 2.11 ja 2.12 põhjal, sest $K \mathbin{@} (M[V/x]) \in \mathcal{SN}$, kuna $K \in \mathbf{KRed}_Z$ ja lemma eelduse põhjal $M[V/x] \in \mathbf{CRed}_Z$, ja sarnaselt $K \mathbin{@} (N[W/y]) \in \mathcal{SN}$. □

Analoogiliselt tõestame ka lemma korrutistüübi jaoks:

Lemma 2.14. *Olgu* $V \in \mathbf{VRed}_{X \times Y}$ *ja* $(x, y)N : [X, Y] \multimap Z$ *nii, et* $N[V/x, W/y] \in \mathbf{CRed}_Z$ *iga* $V \in \mathbf{VRed}_X$ *ja* $W \in \mathbf{VRed}_Y$ *korral. Siis* $(\mathsf{match}\ V\ \mathsf{with}\ \{(x, y) \mapsto N\}) \in \mathbf{CRed}_Z$.

Tõestus. Lisas A.2. □

Lemma tühja tüübi jaoks järeldub induktsioonist jätku pikkuse järgi nagu lemmas 2.7:

Lemma 2.15. *Olgu* $V \in \mathbf{VRed}_0$. *Siis* $\mathsf{match}\ V\ \mathsf{with}\ \{\}_Z \in \mathbf{CRed}_Z$.

Lemma funktsiooni rakenduste jaoks järeldub otse hulga $\mathbf{VRed}_{X \to Y}$ definitsioonist:

Lemma 2.16. *Olgu* $V \in \mathbf{VRed}_{X \to Y}$ *ja* $W \in \mathbf{VRed}_X$. *Siis* $V\ W \in \mathbf{CRed}_Y$.

Nüüd on võimalik tõestatud lemmasid kasutades näidata, et iga hästi tüübitud term, milles kõik vabad muutujad on asendatud redutseeritavate termidega, on omakorda redutseeritav. Seda tehakse vaadeldes kõiki võimalikke antud termi kujusid. Kuna kõik juhud on sarnased (iga juht kasutab ühte lemmadest 2.3, 2.4, 2.5, 2.6, 2.8, 2.10, 2.13, 2.14, 2.15, 2.16), siis toome siin välja ainult mõned juhud. Täielik tõestus on ära toodud lisas A.3.

Teoreem 2.17. *Olgu* $T : Z$ *hästi tüübitud väärtus või arvutus, mille vabad muutujad on* $x_1 : X_1, \ldots, x_n : X_n$. *Siis iga* $V_1 \in \mathbf{VRed}_{X_1}, \ldots, V_n \in \mathbf{VRed}_{X_n}$ *korral, kas* $T[V_1/x_1, \ldots, V_n/x_n] \in \mathbf{VRed}_Z$, *kui* T *on väärtus, või* $T[V_1/x_1, \ldots, V_n/x_n] \in \mathbf{CRed}_Z$, *kui* T *on arvutus.*

Tõestus. Ruumi kokkuhoiu nimel kirjutame $V_1/x_1, \ldots, V_n/x_n$ asemel $\overline{V}/\overline{x}$. Tõestame teoreemi induktsiooniga termi T struktuuri järgi:

- Kui $T = z$, siis leidub indeks i nii, et $z = x_i$. Eelduse põhjal $V_i \in \mathbf{VRed}_Z$ ja seega saame, et $T[\overline{V}/\overline{x}] = V_i \in \mathbf{VRed}_Z$.
- Kui $T = \mathsf{fun}\ (x : X) \mapsto N$, kus $N : Y$, siis $T[\overline{V}/\overline{x}] = \mathsf{fun}\ (x : X) \mapsto N[\overline{V}/\overline{x}]$. Induktsiooni hüpoteesi põhjal $N[\overline{V}/\overline{x}, W/x] \in \mathbf{CRed}_Y$ iga $W \in \mathbf{VRed}_X$ jaoks. Järelikult lemma 2.6 põhjal saame, et $T[\overline{V}/\overline{x}] \in \mathbf{VRed}_{X \to Y}$.
- Kui $T = \mathsf{match}\ V\ \mathsf{with}\ \{(x, y) \mapsto N\}$, kus $V : X \times Y$, siis $T[\overline{V}/\overline{x}] = \mathsf{match}\ V[\overline{V}/\overline{x}]\ \mathsf{with}\ \{(x, y) \mapsto N[\overline{V}/\overline{x}]\}$. Induktsiooni hüpoteesi põhjal $V[\overline{V}/\overline{x}] \in \mathbf{VRed}_{X \times Y}$ ja $N[\overline{V}/\overline{x}, V/x, W/y] \in \mathbf{CRed}_Z$ iga $V \in \mathbf{VRed}_X$ ja $W \in \mathbf{VRed}_Y$ jaoks. Järelikult lemma 2.14 põhjal saame, et $T[\overline{V}/\overline{x}] \in \mathbf{CRed}_Z$. □

Lõpuks jõuame soovitud tulemuseni.

Teoreem 2.18. *Kõik keele FGCBV termid on tugevalt normaliseeruvad.*

Tõestus. Olgu $T : Z$ hästi tüübitud term. Kui T on väärtus, siis see on triviaalselt tugevalt normaliseeruv. Kui T on arvutus, siis võttes teoreemis 2.17 $V_1 = x_1, \ldots, V_n = x_n$ (teoreemi 2.7 põhjal iga $x_i \in \mathbf{VRed}_{X_i}$) saame, et $T \in \mathbf{CRed}_Z$. Teoreemi 2.1 põhjal saame siis, et $T \in \mathcal{SN}$. □

3. Asünkroonsete efektidega lambda-arvutuse tugevalt normaliseeruvus

Selles peatükis anname tugeva normaliseeruvuse tõestuse keele $\lambda_{æ}$ jaoks. Nagu varem öeldud, siis see tõestus seisneb eelmises peatükis keele FGCBV jaoks toodud tõestuse laiendamises keele $\lambda_{æ}$ asünkroonsetele konstruktsioonidele.

Esiteks, lisame tõestuse (konkreetselt lemma 3.10 tõestuse) tehniliseks lihtsustamiseks keelde erilise lubaduse tüüpi konstandi $\star$:

$$V, W ::= \ldots \mid \star \qquad\qquad \frac{}{\Gamma \vdash_{\mathsf{v}} \star : \langle X \rangle}$$

Intuitiivselt on $\star$ igas tüüpimiskontekstis kättesaadav konstant, mis kirjeldab suvalist määramata lubadust. Täpsemalt, vaadeldes hästi tüübitud arvutust $\Gamma \vdash \mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N : Y$, kus $\Gamma, p : \langle X \rangle \vdash N : Y$, soovime uurida arvutuse N võimalikku käitumist, mis ei sõltu (termis N) vabast muutujast p. Selleks uurime termi $N[\star/p]$, mille tüüpimiskontekst on Γ. Siis term $N[\star/p]$ käitub nagu term N, kus lubadus p ei ole veel täidetud.

Nimetame saadud keelt selguse mõttes $\lambda^{\star}_{æ}$. Märgime, et töös tõestatud tulemus kehtib ka ilma $\star$-ta keele $\lambda_{æ}$ jaoks, sest kõik keele $\lambda_{æ}$ termid sisalduvad ka keeles $\lambda^{\star}_{æ}$, ja kõik keele $\lambda^{\star}_{æ}$ reduktsioonireeglid langevad üks-ühele kokku keele $\lambda_{æ}$ reduktsioonireeglitega.

3.1 Redutseeritavuse definitsioon

Et defineerida jätkud keele $\lambda^{\star}_{æ}$ jaoks, teeme järgmised tähelepanekud keele $\lambda^{\star}_{æ}$ reduktsioonireeglite (tabel 1) kohta:

- await-termid, katkestuste töötlejad ja signaalid liiguvad termis seestpoolt väljapoole, välja let-termidest ja mööda katkestustest;
- await-termid käituvad nagu match-termid $\langle X \rangle$-tüüpi väärtuste jaoks;
- signaalid liiguvad väljapoole mööda katkestuste töötlejatest;
- let-termid ja katkestused omavahel ei reageeri.

Need tähelepanekud osutavad sellele, et lisaks let-termidele, on jätkudes mõistlik lubada ka katkestusi. Niisiis, laiendame jätkude definitsiooni kujule

$$K ::= \mathit{Id} \mid K \circ (x)N \mid K \circ \downarrow \mathsf{op}\,(V)$$

ja jätkude rakenduse termidele kujule

$$\begin{aligned} \mathit{Id} \mathbin{@} M &= M, \\ (K \circ (x)N) \mathbin{@} M &= K \mathbin{@} (\mathsf{let}\ x = M\ \mathsf{in}\ N), \\ (K \circ \downarrow \mathsf{op}\,(V)) \mathbin{@} M &= K \mathbin{@} (\downarrow \mathsf{op}\,(V, M)). \end{aligned}$$

Jätkude pikkus defineeritakse tavapärasel viisil:

$$|\mathit{Id}| = 0 \qquad |K \circ (x)N| = |K \circ \downarrow \mathsf{op}\,(V)| = |K| + 1.$$

Jätkude tüüpimisreeglid on järgmised:

$$\frac{}{\Gamma \vdash \mathit{Id} : X \multimap X} \qquad \frac{\Gamma \vdash (x)N : X \multimap Y \qquad \Gamma \vdash K : Y \multimap Z}{\Gamma \vdash K \circ (x)N : X \multimap Z}$$

$$\frac{\mathsf{op} \in \Sigma \qquad \Gamma \vdash_{\mathsf{v}} V : A_{\mathsf{op}} \qquad \Gamma \vdash K : X \multimap Y}{\Gamma \vdash K \circ \downarrow \mathsf{op}\,(V) : X \multimap Y}$$

Tänu redukstioonireegli (a) puudumisele keelest $\lambda_{\text{æ}}$ (ja seega ka keelest $\lambda^{\star}_{\text{æ}}$), jätkud siin ei redutseeru, st ei leidu jätke K, K' nii, et iga M korral $K \mathbin{@} M \rightsquigarrow K' \mathbin{@} M$. Selles võib veenduda vaadates läbi kõik reduktsioonireeglid.

Redutseeritavate arvutuste definitsioon jääb samaks:

$$M \in \mathbf{CRed}_X \Leftrightarrow \text{ iga } K \in \mathbf{KRed}_X \text{ korral } K \mathbin{@} M \in \mathcal{SN}.$$

Hulga $\mathbf{KRed}_X$ definitsioonile lisandub veel üks tingimus, mida läheb vaja reduktsioonireegli (12) käsitlemisel lemma 3.10 juures:

$$K \in \mathbf{KRed}_X \Leftrightarrow \begin{cases} \text{iga } V \in \mathbf{VRed}_X \text{ korral } K \mathbin{@} (\mathsf{return}\, V) \in \mathcal{SN}, \text{ ja} & \textbf{(KRed1)} \\ \text{iga } \downarrow \mathsf{op}\,(V) \in K \text{ korral } V \in \mathbf{VRed}_{A_{\mathsf{op}}}. & \textbf{(KRed2)} \end{cases}$$

Kuna võrreldes keelega FGCBV lisandub keeles $\lambda^{\star}_{\text{æ}}$ tüüpide hulka lubaduse tüüp, siis peame ka selle tüübi jaoks defineerima redutseeritavuse: see on defineeritud sarnaselt korrutis- ja summatüüpi väärtuste redutseeritavusele:

$$V \in \mathbf{VRed}_{\langle X \rangle} \Leftrightarrow \text{ iga } A \in \mathbf{ARed}_X \text{ korral } A \mathbin{@} V \in \mathcal{SN},$$
$$A \in \mathbf{ARed}_X \Leftrightarrow \text{ iga } V \in \mathbf{VRed}_X \text{ korral } A \mathbin{@} \langle V \rangle \in \mathcal{SN},$$

kus A tähistab await-jätkusid:

$$A ::= K \circ ((x)N)_{\langle\rangle}$$

$$(K \circ ((x)N)_{\langle\rangle}) \mathbin{@} V = K \mathbin{@} (\mathsf{await}\ V\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ N)$$

$$\frac{\Gamma \vdash K : Y \multimap Z \qquad \Gamma \vdash (x)N : X \multimap Y}{\Gamma \vdash K \circ ((x)N)_{\langle\rangle} : [\langle X \rangle] \multimap Z}$$

3.2 Redutseeritavusest järeldub tugev normaliseeruvus

Seda, et redutseeritavad termid on tugevalt normaliseeruvad, tõestame täpselt samamoodi nagu tegime keele FGCBV jaoks teoreemis 2.1:

Teoreem 3.1. *Iga $M \in \mathbf{CRed}_X$ korral $M \in \mathcal{SN}$.*

3.3 Iga hästi tüübitud term on redutseeritav

Nagu keele FGCBV juures, alustame väärtustest. On kaks uut väärtust: $\star$ ja $\langle V \rangle$.

Lemma 3.2. *Konstant $\star$ on redutseeritav:* $\star \in \mathbf{VRed}_{\langle X \rangle}$.

Tõestus. Olgu $A \in \mathbf{ARed}_X$. Peame näitama, et $A \mathbin{@} \star \in \mathcal{SN}$. See kehtib triviaalselt, sest term $A \mathbin{@} \star$ on normaalkujul. □

Lemma 3.3. *Olgu $V \in \mathbf{VRed}_X$. Siis $\langle V \rangle \in \mathbf{VRed}_{\langle X \rangle}$.*

Tõestus. See väide järeldub vahetult hulkade $\mathbf{ARed}_X$ ja $\mathbf{VRed}_{\langle X \rangle}$ definitsioonidest. □

Edasi liigume arvutuste juurde. Uued arvutused on katkestuste töötlejad, await-termid, signaalid ja katkestused. Lisaks nõuab let-term oma käsitluses muudatusi. Ülejäänud arvutuste jaoks tõestatakse redutseeritavus nagu keele FGCBV juures.

Katkestuse ja let-termi juhud tõestame lükates vastava konstruktsiooni jätku sisse. Näitame, et redutseeritavasse jätku let-termi või katkestuse lükkamine säilitab jätku redutseeritavust. Lemmas 3.4 eeldame mõnevõrra nõrgemat tingimust kui $K \in \mathbf{KRed}_X$, mis tuleb kasuks lemma 3.10 esimese juhu tõestamisel.

Lemma 3.4. *Olgu $K : Y \multimap Z$ jätk ja $(x)N : X \multimap Y$ term-abstraktsioon sellised, et K vastab tingimusele* **KRed2** *ja $K \mathbin{@} (N[W/x]) \in \mathcal{SN}$ iga $W \in \mathbf{VRed}_X$ korral. Siis $K \circ (x)N \in \mathbf{KRed}_X$.*

Tõestus. On ilmne, et kui K vastab tingimusele **KRed2**, siis ka $K \circ (x)N$ vastab tingimusele **KRed2**. Jääb kontrollida tingimus **KRed1**. Olgu $W \in \mathbf{VRed}_X$. Term $(K \circ (x)N) \mathbin{@} (\mathsf{return}\, W)$ redutseerub ainult termiks $K \mathbin{@} (N[W/x])$ reduktsioonireegliga (2). Term $K \mathbin{@} (N[W/x])$ on tugevalt normaliseeruv lemma eelduse põhjal. Seega $(K \circ (x)N) \mathbin{@} (\mathsf{return}\, W) \in \mathcal{SN}$ ja **KRed1** kehtib. Järelikult $K \circ (x)N \in \mathbf{KRed}_X$. □

Lemma 3.5. *Olgu $K \in \mathbf{KRed}_X$, $\mathsf{op} \in \Sigma$ ja $V \in \mathbf{VRed}_{A_{\mathsf{op}}}$. Siis $K \circ \downarrow \mathsf{op}\,(V) \in \mathbf{KRed}_X$.*

Tõestus. Kuna K vastab tingimusele **KRed2** ja $V \in \mathbf{VRed}_{A_{\mathsf{op}}}$ siis ka $K \circ \downarrow \mathsf{op}\,(V)$ vastab tingimusele **KRed2**. Jääb kontrollida tingimus **KRed1**. Olgu $W \in \mathbf{VRed}_X$. Term $(K \circ \downarrow \mathsf{op}\,(V)) \mathbin{@} (\mathsf{return}\, W)$ redutseerub ainult termiks $K \mathbin{@} (\mathsf{return}\, W)$ reduktsioonireegliga (10). Term $K \mathbin{@} (\mathsf{return}\, W)$ on tugevalt normaliseeruv, sest $K \in \mathbf{KRed}_X$ ja $W \in \mathbf{VRed}_X$. Seega $(K \circ \downarrow \mathsf{op}\,(V)) \mathbin{@} (\mathsf{return}\, W) \in \mathcal{SN}$ ja **KRed1** kehtib. Järelikult $K \circ \downarrow \mathsf{op}\,(V) \in \mathbf{KRed}_X$. □

Nüüd tõestame let-termi ja katkestuse redutseeruvuse vastava termi jätku sisse lükkamise teel:

Lemma 3.6. *Olgu $M \in \mathbf{CRed}_X$ ja arvutus $N : Y$ selline, et $N[W/x] \in \mathbf{CRed}_Y$ iga $W \in \mathbf{VRed}_X$ korral. Siis* $\mathsf{let}\ x = M\ \mathsf{in}\ N \in \mathbf{CRed}_Y$.

Tõestus. Olgu $K \in \mathbf{KRed}_Y$. Peame näitama, et $K \mathbin{@} (\mathsf{let}\ x = M\ \mathsf{in}\ N) \in \mathcal{SN}$. Kuna iga $W \in \mathbf{VRed}_X$ korral $N[W/x] \in \mathbf{CRed}_Y$, siis $K \mathbin{@} (N[W/x]) \in \mathcal{SN}$ ja seega lemma 3.4 põhjal $K \circ (x)N \in \mathbf{KRed}_X$. Kuna veel $M \in \mathbf{CRed}_X$, siis $K \mathbin{@} (\mathsf{let}\ x = M\ \mathsf{in}\ N) = (K \circ (x)N) \mathbin{@} M \in \mathcal{SN}$ nagu soovitud. □

Lemma 3.7. *Olgu* $\mathsf{op} \in \Sigma, V \in \mathbf{VRed}_{A_{\mathsf{op}}}$ *ja* $M \in \mathbf{CRed}_X$. *Siis* $\downarrow \mathsf{op}\,(V, M) \in \mathbf{CRed}_X$.

Tõestus. Olgu $K \in \mathbf{KRed}_X$. Peame näitama, et $K \mathbin{@} \downarrow \mathsf{op}\,(V, M) \in \mathcal{SN}$. Kuna $V \in \mathbf{VRed}_{A_{\mathsf{op}}}$, siis lemma 3.5 põhjal $K \circ \downarrow \mathsf{op}\,(V) \in \mathbf{KRed}_X$. Kuna veel $M \in \mathbf{CRed}_X$, siis $K \mathbin{@} \downarrow \mathsf{op}\,(V, M) = (K \circ \downarrow \mathsf{op}\,(V)) \mathbin{@} M \in \mathcal{SN}$ nagu soovitud. □

Järgmisena tõestame lemmad signaali, katkestuste töötleja ja await-termi jaoks. Selleks läheb vaja kolme abitulemust, mis ütlevad, et kui lisada tugevalt normaliseeruva jätku termile rakendamisse juurde signaal, katkestuste töötleja või await-term, siis saadud term on ka tugevalt normaliseeruv.

Lemma 3.8. *Olgu* $M : X$ *arvutus ja* $K : X \multimap Y$ *jätk, nii et* $K \mathbin{@} M \in \mathcal{SN}$, *ja olgu* $\mathsf{op} \in \Sigma$ *ja* $V : A_{\mathsf{op}}$. *Siis* $K \mathbin{@} \uparrow \mathsf{op}\,(V, M) \in \mathcal{SN}$.

Tõestus. Tõestame induktsiooniga $|K| + max(K \mathbin{@} M)$ järgi, et termi $K \mathbin{@} \uparrow \mathsf{op}\,(V, M)$ kõik reduktsioonid on tugevalt normaliseeruvad. Võimalikud reduktsioonid on järgmised:

(6) $\rightsquigarrow K' \mathbin{@} \uparrow \mathsf{op}\,(V, \mathsf{let}\ x = M\ \mathsf{in}\ N)$, kui $K = K' \circ (x)N$: siis $|K'| < |K|$ ja $max(K' \mathbin{@} (\mathsf{let}\ x = M\ \mathsf{in}\ N)) = max(K \mathbin{@} M)$, sest $K' \mathbin{@} (\mathsf{let}\ x = M\ \mathsf{in}\ N) = K \mathbin{@} M \in \mathcal{SN}$. Seega induktsiooni hüpoteesi põhjal $K' \mathbin{@} \uparrow \mathsf{op}\,(V, \mathsf{let}\ x = M\ \mathsf{in}\ N) \in \mathcal{SN}$.

(11) $\rightsquigarrow K' \mathbin{@} \uparrow \mathsf{op}\,(V, \downarrow \mathsf{op}'\,(V', M))$, kui $K = K' \circ \downarrow \mathsf{op}'\,(V')$: siis $|K'| < |K|$ ja $max(K' \mathbin{@} \downarrow \mathsf{op}'\,(V', M)) = max(K \mathbin{@} M)$. Seega induktsiooni hüpoteesi põhjal $K' \circ \uparrow \mathsf{op}\,(V, \downarrow \mathsf{op}'\,(V', M)) \in \mathcal{SN}$.

- $\rightsquigarrow K \mathbin{@} \uparrow \mathsf{op}\,(V, M')$, kui $M \rightsquigarrow M'$: siis $K \mathbin{@} M \rightsquigarrow K \mathbin{@} M'$. Seega $max(K \mathbin{@} M') < max(K \mathbin{@} M)$ ja induktsiooni hüpoteesi põhjal $K \mathbin{@} \uparrow \mathsf{op}\,(V, M') \in \mathcal{SN}$. □

Katkestuste töötleja jaoks lemma tõestamine nõuab esmalt üht abitulemust.

Lemma 3.9. *Olgu* $\mathsf{op} \in \Sigma$, $K : X \multimap Y$ *jätk ja* $N : X$ *arvutus nii, et* $K \mathbin{@} \uparrow \mathsf{op}\,(V, N) \in \mathcal{SN}$. *Siis* $max(K \mathbin{@} N) \leq max(K \mathbin{@} \uparrow \mathsf{op}\,(V, N))$.

Tõestus. On selge, et kuna $K \mathbin{@} \uparrow \mathsf{op}\,(V, N) \in \mathcal{SN}$, siis ka $K \mathbin{@} N \in \mathcal{SN}$. Väite tõestuseks piisab, kui näidata, et igale reduktsioonide jadale $K \mathbin{@} N \rightsquigarrow N_1 \rightsquigarrow N_2 \rightsquigarrow \ldots \rightsquigarrow N_n$ pikkusega n vastab reduktsioonide jada, mis algab termist $K \mathbin{@} \uparrow \mathsf{op}\,(V, N)$ ja mille pikkus ei ole väiksem kui n. Liigutades signaali mööda jätku K liikmetest saame termist $K \mathbin{@} \uparrow \mathsf{op}\,(V, N)$ termi $\uparrow \mathsf{op}\,(V, K \mathbin{@} N)$. Nüüd sellest termist on olemas reduktsioonide jada $\uparrow \mathsf{op}\,(V, K \mathbin{@} N) \rightsquigarrow$

$\uparrow \mathsf{op}\,(V, N_1) \rightsquigarrow \uparrow \mathsf{op}\,(V, N_2) \rightsquigarrow \ldots \rightsquigarrow \uparrow \mathsf{op}\,(V, N_n)$ pikkusega n. Niisiis leidsime soovitud reduktsioonide jada. □

Järgnev lemma katkestuste töötlejate jaoks on töö kõige tähtsam tulemus. See on ka koht, kus kasutatakse lubaduse tüüpi konstanti $\star$. Nimelt soovime teha induktsiooni "$K \mathbin{@} N$" järgi, mis ei ole aga korrektne rakendus, sest kontekstide erinevuse tõttu ei saa jätku K rakendada termile N: termi N kontekst on $\Gamma, p : \langle X \rangle$, aga jätku K kontekst on Γ. Selle asemel teeme induktsiooni $max(K \mathbin{@} (N[\star/p]))$ järgi: see rakendus on korrektne, sest termi $N[\star/p]$ kontekst on Γ. Täpsuse mõttes toome järgmise lemma väites välja kõik termide kontekstid.

Lemma 3.10. *Olgu* $\Gamma \vdash K : Y \multimap Z$ *jätk ja* $\Gamma, p : \langle X \rangle \vdash_{\mathsf{c}} N : Y$ *arvutus nii, et*

- $K \mathbin{@} (N[W/p]) \in \mathcal{SN}$ *iga* $\Gamma \vdash_{\mathsf{v}} W : \langle X \rangle$ *korral, mille jaoks kehtib* $W \in \mathbf{VRed}_{\langle X \rangle}$*;*
- K *vastab tingimusele* $\mathbf{KRed2}$*;*

ja olgu $\mathsf{op} \in \Sigma$ *ja* $\Gamma, x : A_{\mathsf{op}} \vdash_{\mathsf{c}} M : \langle X \rangle$ *selline, et* $M[V/x] \in \mathbf{CRed}_{\langle X \rangle}$ *iga* $\Gamma \vdash_{\mathsf{v}} V : A_{\mathsf{op}}$ *korral, mille jaoks kehtib* $V \in \mathbf{VRed}_{A_{\mathsf{op}}}$*. Siis* $K \mathbin{@} (\mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N) \in \mathcal{SN}$*.*

Tõestus. Lemma 3.2 põhjal $\star \in \mathbf{VRed}_{\langle X \rangle}$. Seega eelduse põhjal $K \mathbin{@} (N[\star/p]) \in \mathcal{SN}$. Tõestame induktsiooniga $|K| + max(K \mathbin{@} (N[\star/p]))$ ja termi N struktuuri järgi, et termi $K \mathbin{@} (\mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N)$ kõik reduktsioonid on tugevalt normaliseeruvad. Võimalikud reduktsioonid on järgmised:

(12) $\rightsquigarrow K' \mathbin{@} (\mathsf{let}\ p = M[V/x]\ \mathsf{in} \downarrow \mathsf{op}\,(V, N))$, kui $K = K' \circ \downarrow \mathsf{op}\,(V)$ ja $p \notin fv(V)$: kuna

$$K' \mathbin{@} (\mathsf{let}\ p = M[V/x]\ \mathsf{in} \downarrow \mathsf{op}\,(V, N)) = (K' \circ (p) \downarrow \mathsf{op}\,(V, N)) \mathbin{@} (M[V/x])$$

ja lemma eelduse põhjal $M[V/x] \in \mathbf{CRed}_{\langle X \rangle}$ (sest $V \in \mathbf{VRed}_{A_{\mathsf{op}}}$, mis järeldub sellest, et K vastab tingimusele $\mathbf{KRed2}$), siis termi $(K' \circ (p) \downarrow \mathsf{op}\,(V, N)) \mathbin{@} (M[V/x])$ tugevaks normaliseeruvuseks piisab näidata, et $K' \circ (p) \downarrow \mathsf{op}\,(V, N) \in \mathbf{KRed}_{\langle X \rangle}$. See kehtib lemma 3.4 põhjal, sest iga $W \in \mathbf{VRed}_{\langle X \rangle}$ korral

$$K' \mathbin{@} (\downarrow \mathsf{op}\,(V, N)[W/p]) = K' \mathbin{@} \downarrow \mathsf{op}\,(V, N[W/p]) = K \mathbin{@} (N[W/p]) \in \mathcal{SN},$$

kus esimene võrdus kehtib, sest $p \notin fv(V)$.

(13) $\rightsquigarrow K' \mathbin{@} (\mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in} \downarrow \mathsf{op}'\,(V, N))$, kui $K = K' \circ \downarrow \mathsf{op}'\,(V)$, $\mathsf{op} \neq \mathsf{op}'$ ja $p \notin fv(V)$: siis $|K'| < |K|$, K' vastab tingimusele $\mathbf{KRed2}$ ja iga $W \in \mathbf{VRed}_{\langle X \rangle}$ (sh $\star$) korral $K' \mathbin{@} ((\downarrow \mathsf{op}'\,(V, N))[W/p]) = K' \mathbin{@} \downarrow \mathsf{op}'\,(V, N[W/p]) = K \mathbin{@} (N[W/p]) \in \mathcal{SN}$, seega induktsiooni hüpoteesi põhjal

$$K' \mathbin{@} (\mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in} \downarrow \mathsf{op}'\,(V, N)) \in \mathcal{SN}.$$

(7) $\rightsquigarrow K' \mathbin{@} (\mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ (\mathsf{let}\ x = N\ \mathsf{in}\ L))$, kui $K = K' \circ (x)L$ ja $p \notin fv(L)$: see juht on analoogne eelmise juhuga.

(9) $\rightsquigarrow K \mathbin{@} \uparrow \mathsf{op}'\,(V, \mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N')$, kui $N = \uparrow \mathsf{op}'\,(V, N')$ ja $p \notin fv(V)$: kuna N' on termi N alamterm ja iga $W \in \mathbf{VRed}_{\langle X \rangle}$ (sh $\star$) korral kehtib lemma 3.9 põhjal

$$max(K \mathbin{@} (N'[W/p])) \leq max(K \mathbin{@} (\uparrow \mathsf{op}\,(V, N'[W/p]))) = max(K \mathbin{@} (N[W/p])),$$

siis induktsiooni hüpoteesi põhjal $K \mathbin{@} (\mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N') \in \mathcal{SN}$. Lemma 3.8 põhjal siis $K \mathbin{@} \uparrow \mathsf{op}'\,(V, \mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N') \in \mathcal{SN}$.

- $\rightsquigarrow K \mathbin{@} (\mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N')$, kui $N \rightsquigarrow N'$: siis iga $W \in \mathbf{VRed}_{\langle X \rangle}$ (sh $\star$) korral $K \mathbin{@} (N'[W/p]) \in \mathcal{SN}$, sest $K \mathbin{@} (N[W/p]) \in \mathcal{SN}$ ja $K \mathbin{@} (N[W/p]) \rightsquigarrow K \mathbin{@} (N'[W/p])$, seega induktsiooni hüpoteesi põhjal $K \mathbin{@} (\mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N') \in \mathcal{SN}$. □

Järgmine lemma on await-termi jaoks, mille käsitlus sarnaneb match-termidele.

Lemma 3.11. *Olgu $K : Y \multimap Z$ jätk, $(x)N : X \multimap Y$ term-abstraktsioon ja $W : X$ väärtus nii, et $K \mathbin{@} (N[W/x]) \in \mathcal{SN}$. Siis $K \mathbin{@} (\mathsf{await}\ \langle W \rangle\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ N) \in \mathcal{SN}$.*

Tõestus. Tõestame induktsiooniga $|K|$ järgi, et termi $K \mathbin{@} (\mathsf{await}\ \langle W \rangle\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ N)$ kõik reduktsioonid on tugevalt normaliseeruvad. Võimalikud reduktsioonid on järgmised:

(15) $\rightsquigarrow K \mathbin{@} (N[W/x])$: see on tugevalt normaliseeruv lemma eelduse põhjal.

(14) $\rightsquigarrow K' \mathbin{@} (\mathsf{await}\ \langle W \rangle\ \mathsf{until}\ \langle x \rangle\ \mathsf{in} \downarrow \mathsf{op}\,(V, N))$, kui $K = K' \circ \downarrow \mathsf{op}\,(V)$ ja $x \notin fv(V)$: siis $|K'| < |K|$ ja

$$K' \mathbin{@} (\downarrow \mathsf{op}\,(V, N)[W/x]) = K' \mathbin{@} \downarrow \mathsf{op}\,(V, N[W/x]) = K \mathbin{@} (N[W/x]) \in \mathcal{SN},$$

seega induktsiooni hüpoteesi põhjal $K' \mathbin{@} (\mathsf{await}\ \langle W \rangle\ \mathsf{until}\ \langle x \rangle\ \mathsf{in} \downarrow \mathsf{op}\,(V, N)) \in \mathcal{SN}$.

(8) $\rightsquigarrow K' \mathbin{@} (\mathsf{await}\ \langle W \rangle\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ (\mathsf{let}\ y = N\ \mathsf{in}\ L))$, kui $K = K' \circ (y)L$ ja $x \notin fv(L)$: see juht on analoogne eelmise juhuga. □

Nüüd on võimalik näidata, et redutseeritavatest termidest koosnevad signaalid, katkestuste töötlejad ja await-termid on redutseeritavad.

Lemma 3.12. *Olgu $\mathsf{op} \in \Sigma, V : A_{\mathsf{op}}$ ja $M \in \mathbf{CRed}_X$. Siis $\uparrow \mathsf{op}\,(V, M) \in \mathbf{CRed}_X$.*

Tõestus. Peame näitama, et iga $K \in \mathbf{KRed}_X$ korral $K \mathbin{@} \uparrow \mathsf{op}\,(V, M) \in \mathcal{SN}$. Kuna $M \in \mathbf{CRed}_X$, siis $K \mathbin{@} M \in \mathcal{SN}$. Lemma 3.8 põhjal $K \mathbin{@} \uparrow \mathsf{op}\,(V, M) \in \mathcal{SN}$. □

Lemma 3.13. *Olgu $\mathsf{op} \in \Sigma$ ja $M : \langle X \rangle, N : Y$ sellised, et $M[V/x] \in \mathbf{CRed}_{\langle X \rangle}$ iga $V \in \mathbf{VRed}_{A_{\mathsf{op}}}$ korral ja $N[W/p] \in \mathbf{CRed}_Y$ iga $W \in \mathbf{VRed}_{\langle X \rangle}$ korral. Siis $\mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N \in \mathbf{CRed}_Y$.*

Tõestus. Olgu $K \in \mathbf{KRed}_Y$. Peame näitama, et $K \mathbin{@} (\mathsf{promise}\ (\mathsf{op}\ x \mapsto M_{\mathsf{op}})\ \mathsf{as}\ p\ \mathsf{in}\ N) \in \mathcal{SN}$. See kehtib lemma 3.10 põhjal, sest K vastab tingimusele **KRed2** ja $K \mathbin{@} (N[W/p]) \in \mathcal{SN}$ iga $W \in \mathbf{VRed}_{\langle X \rangle}$ korral, sest $K \in \mathbf{KRed}_Y$ ja lemma eelduse põhjal $N[W/p] \in \mathbf{CRed}_Y$. □

Lemma 3.14. *Olgu $V \in \mathbf{VRed}_{\langle X \rangle}$ ja $N : Y$ selline, et $N[W/x] \in \mathbf{CRed}_Y$ iga $W \in \mathbf{VRed}_X$ korral. Siis* $\mathsf{await}\ V\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ N \in \mathbf{CRed}_Y$.

Tõestus. Olgu $K \in \mathbf{KRed}_Y$. Peame näitama, et $K \mathbin{@} (\mathsf{await}\ V\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ N) \in \mathcal{SN}$. Kuna $K \mathbin{@} (\mathsf{await}\ V\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ N) = (K \circ ((x)N)_{\langle\rangle}) \mathbin{@} V$ ja $V \in \mathbf{VRed}_{\langle X \rangle}$, siis selleks piisab, kui $K \circ ((x)N)_{\langle\rangle} \in \mathbf{ARed}_X$ ehk $(K \circ ((x)N)_{\langle\rangle}) \mathbin{@} \langle W \rangle \in \mathcal{SN}$ iga $W \in \mathbf{VRed}_X$ korral. Olgu $W \in \mathbf{VRed}_X$. Siis lemma eelduse põhjal $N[W/x] \in \mathbf{CRed}_Y$ ja seega $K \mathbin{@} (N[W/x]) \in \mathcal{SN}$. Järelikult lemma 3.11 põhjal $K \mathbin{@} (\mathsf{await}\ \langle W \rangle\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ N) \in \mathcal{SN}$ ja seega $(K \circ ((x)N)_{\langle\rangle}) \mathbin{@} \langle W \rangle = K \mathbin{@} (\mathsf{await}\ \langle W \rangle\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ N) \in \mathcal{SN}$ nagu soovitud. □

Lõpuks jõuame loogiliste relatsioonide fundamentaalteoreemini.

Teoreem 3.15. *Olgu $T : Z$ hästi tüübitud väärtus või arvutus, mille vabad muutujad on $x_1 : X_1, \ldots, x_n : X_n$. Siis iga $V_1 \in \mathbf{VRed}_{X_1}, \ldots, V_n \in \mathbf{VRed}_{X_n}$ korral $T[\overline{V}/\overline{x}] \in \mathbf{VRed}_Z$, kui T on väärtus, või $T[\overline{V}/\overline{x}] \in \mathbf{CRed}_Z$, kui T on arvutus.*

Tõestus. Kasutame induktsiooni termi T struktuuri järgi (täpsemini, termi T hästitüübituse tuletuspuu struktuuri järgi). Võrreldes vastava tõestusega keele FGCBV jaoks, vaatame siin ainult keeles $\lambda^{\star}_{\text{æ}}$ lisanduvaid uusi juhtusid. Ülejäänud juhtude käsitlus jääb samasuguseks nagu keele FGCBV puhul.

- Kui $T = \star$, siis $T[\overline{V}/\overline{x}] = \star$. Lemma 3.2 põhjal $T[\overline{V}/\overline{x}] \in \mathbf{VRed}_{\langle X \rangle}$.
- Kui $T = \langle V \rangle$, kus $V : X$, siis $T[\overline{V}/\overline{x}] = \langle V[\overline{V}/\overline{x}] \rangle$. Induktsiooni hüpoteesi põhjal $V[\overline{V}/\overline{x}] \in \mathbf{VRed}_{\langle X \rangle}$. Järelikult lemma 3.3 põhjal $T[\overline{V}/\overline{x}] \in \mathbf{VRed}_{\langle X \rangle}$.
- Kui $T = {\downarrow}\mathsf{op}\,(V, M)$, siis $T[\overline{V}/\overline{x}] = {\downarrow}\mathsf{op}\,(V[\overline{V}/\overline{x}], M[\overline{V}/\overline{x}])$. Induktsiooni hüpoteesi põhjal $V[\overline{V}/\overline{x}] \in \mathbf{VRed}_{A_{\mathsf{op}}}$ ja $M[\overline{V}/\overline{x}] \in \mathbf{CRed}_Z$. Järelikult lemma 3.7 põhjal ${\downarrow}\mathsf{op}\,(V[\overline{V}/\overline{x}], M[\overline{V}/\overline{x}]) \in \mathbf{CRed}_Z$.
- Kui $T = {\uparrow}\mathsf{op}\,(V, M)$, siis $T[\overline{V}/\overline{x}] = {\uparrow}\mathsf{op}\,(V[\overline{V}/\overline{x}], M[\overline{V}/\overline{x}])$. Induktsiooni hüpoteesi põhjal $M[\overline{V}/\overline{x}] \in \mathbf{CRed}_Z$ ja lemma 3.12 põhjal ${\uparrow}\mathsf{op}\,(V[\overline{V}/\overline{x}], M[\overline{V}/\overline{x}]) \in \mathbf{CRed}_Z$.
- Kui $T = \mathsf{promise}\ (\mathsf{op}\ x \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N$, siis

  $$T[\overline{V}/\overline{x}] = \mathsf{promise}\ (\mathsf{op}\ x \mapsto M[\overline{V}/\overline{x}])\ \mathsf{as}\ p\ \mathsf{in}\ N[\overline{V}/\overline{x}].$$

  Induktsiooni hüpoteesi põhjal $M[\overline{V}/\overline{x}, V/x] \in \mathbf{CRed}_{\langle X \rangle}$ iga $V \in \mathbf{VRed}_{A_{\mathsf{op}}}$ jaoks ja $N[\overline{V}/\overline{x}, W/p] \in \mathbf{CRed}_Z$ iga $W \in \mathbf{VRed}_{\langle X \rangle}$ jaoks. Järelikult lemma 3.13 põhjal $\mathsf{promise}\ (\mathsf{op}\ x \mapsto M[\overline{V}/\overline{x}])\ \mathsf{as}\ p\ \mathsf{in}\ N[\overline{V}/\overline{x}] \in \mathbf{CRed}_Z$.

- Kui $T = \mathsf{await}\ V\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ N$, siis $T[\overline{V}/\overline{x}] = \mathsf{await}\ V[\overline{V}/\overline{x}]\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ N[\overline{V}/\overline{x}]$. Induktsiooni hüpoteesi põhjal $V[\overline{V}/\overline{x}] \in \mathbf{VRed}_{\langle X \rangle}$ ja $N[\overline{V}/\overline{x}, W/x] \in \mathbf{CRed}_Z$ iga $W \in \mathbf{VRed}_X$ jaoks. Seega lemma 3.14 põhjal $\mathsf{await}\ V[\overline{V}/\overline{x}]\ \mathsf{until}\ \langle x \rangle\ \mathsf{in}\ N[\overline{V}/\overline{x}] \in \mathbf{CRed}_Z$. □

Soovitud tulemuse saamiseks jääb vaid näidata, et muutujad on redutseeritavad. Seda tehakse sarnaselt keele FGCBV sama omaduse (lemma 2.7) tõestusele:

Lemma 3.16. *Olgu $z : Z$. Siis $z \in \mathbf{VRed}_Z$.*

Seda, et iga hästi tüübitud term on tugevalt normaliseeruv, tõestatakse täpselt samamoodi nagu keele FGCBV juures (teoreem 2.18):

Teoreem 3.17. *Kõik keele $\lambda^\star_{æ}$ hästi tüübitud termid on tugevalt normaliseeruvad.*

Lõpuks, keele $\lambda_{æ}$ tugev normaliseeruvus järeldub otse keele $\lambda^\star_{æ}$ tugevalt normaliseeruvusest:

Järeldus 3.18. *Kõik keele $\lambda_{æ}$ hästi tüübitud termid on tugevalt normaliseeruvad.*

Tõestus. Olgu $T : Y$ keele $\lambda_{æ}$ hästi tüübitud term. Siis T on ka keele $\lambda^\star_{æ}$ hästi tüübitud term. On ilmne, et iga termist T algav reduktsioonide jada keeles $\lambda_{æ}$ sisaldub täpselt samal kujul ka keeles $\lambda^\star_{æ}$. Teoreem 3.17 ütleb, et kõik sellised jadad on keeles $\lambda^\star_{æ}$ lõplikud, mis tähendab, et term T on tugevalt normaliseeruv nii keeles $\lambda^\star_{æ}$ kui ka keeles $\lambda_{æ}$. □

Peatüki lõpetuseks kirjeldame, kuidas tugev normaliseeruvus seostub olemasolevate teoreemidega keele $\lambda_{æ}$ hästi tüübitud termide kohta [AP24:ptk 3.4]. Nimetame arvutust M piisavalt suletuks, kui kõik tema vabad muutujad on lubaduse tüüpi, st iga $p \in fv(M)$ korral $p : \langle X \rangle$ mingi tüüpi X jaoks. Ahman ja Pretnar defineerivad oma artiklis kõik võimalikud piisavalt suletud termide normaalkujud [AP24:ptk 3.4]. Seda, et piisavalt suletud arvutus M, mille kõigi vabade muutujate hulk on Ψ, on normaalkujul, tähistatakse $\mathsf{CompRes}\langle \Psi \mid M \rangle$. Samas artiklis on tõestatud ka progressiteoreem, mis väidab, et iga piisavalt suletud arvutus M kas saab teha reduktsioonisammu, või ta on juba normaalkujul [AP24:teoreem 3.2]:

Teoreem 3.19 (Progress)**.** *Olgu M piisavalt suletud arvutus, st olgu ta hästi tüübitud kujul*

$$p_1 : \langle X_1 \rangle, \ldots, p_n : \langle X_n \rangle \vdash_{\mathsf{c}} M : Y\ !\ (o, \iota).$$

Siis kas

(a) leidub arvutus N nii, et $M \rightsquigarrow N$, või

(b) arvutus M on normaalkujul, st kehtib $\mathsf{CompRes}\langle \{p_1, \ldots, p_n\} \mid M \rangle$.

Kombineerides progressiteoreemi ja tugevalt normaliseeruvust saame järeldada, et keele $\lambda_{æ}$ kõik hästi tüübitud arvutused redutseeruvad lõpuks normaalkujul olevaks termiks:

Järeldus 3.20. *Olgu M piisavalt suletud arvutus, mille kõikide vabade muutujate hulk on Ψ. Siis iga arvutusest M algav reduktsioonijada viib lõpuks mingi normaalkujul termini N, mille jaoks kehtib $\mathsf{CompRes}\langle \Psi \mid N \rangle$.*

4. Võimalikud edasiarendused

Keelt $\lambda_{æ}$ tutvustavas artiklis [AP24] on kirjeldatud ka selle keele kõrgemat järku laiendusi. Üheks selle lõputöö edasiarenduseks oleks uurida nende laienduste normaliseerimisomadusi.

Esimene laiendus on taaspaigaldatavad katkestuste töötlejad (ingl *reinstallable interrupt handlers*). Nad võimaldavad käivitada sama katkestuse töötlejat mitu korda, mis oleks kasulik näiteks server-programmide modelleerimise jaoks. Töö käigus leidsime, et sellise laiendusega keel kaotab tugeva normaliseeruvuse: vaatleme näiteks järgmist hästi tüübitud arvutust L:

$$\begin{aligned} &L = \downarrow \mathsf{op}\,(V, \mathsf{promise}\ (\mathsf{op}\ x\ r \mapsto \downarrow \mathsf{op}\,(V, r\,()))\ \mathsf{as}\ p\ \mathsf{in}\ \mathsf{return}\ p), \\ &\Gamma \vdash L : \langle X \rangle\,!\,(\varnothing, \{\mathsf{op} \mapsto (\varnothing, \{\mathsf{op} \mapsto (\varnothing, \cdots)\})\}), \end{aligned}$$

kus $\Gamma \vdash V : A_{\mathsf{op}}$ on suvaline vastavalt katkestuse op tüübile tüübitud väärtus. Arvutus L ei ole tugevalt normaliseeruv, sest temast algab lõpmatu reduktsioonide jada kujul

$$\begin{aligned} L \rightsquigarrow \cdots \rightsquigarrow\ &\mathsf{let}\ p = L\ \mathsf{in} \downarrow \mathsf{op}\,(V, \mathsf{return}\ p) \rightsquigarrow \cdots \rightsquigarrow \\ &\mathsf{let}\ p = (\mathsf{let}\ p = L\ \mathsf{in} \downarrow \mathsf{op}\,(V, \mathsf{return}\ p))\ \mathsf{in} \downarrow \mathsf{op}\,(V, \mathsf{return}\ p) \rightsquigarrow \cdots \end{aligned}$$

Üks viis kuidas võiks selle laienduse jaoks tugeva normaliseeruvust taastada oleks keelata katkestuste väljakutsumist katkestuste töötleja koodi sees, sest praktikas peaksid katkestused tulema teistest protsessidest, mitte olema kirjutatud protsessi koodi programmeerija poolt.

Selline keeld võib olla realiseeritud näiteks tüüpimisrelatsiooni lipu f lisamisega, mis ütleks, et kas antud termis on katkestused süntaktiliselt lubatud või mitte. Konkreetsemalt oleksid nüüd efektiannotatsioonid kujul (o, ι, f), kus $f \in F = \{0, 1\}$. Kui $f = 0$, siis katkestused ei ole vastava annotatsiooniga tüübitud arvutuses lubatud; vastasel juhul on nad lubatud. See keeld oleks realiseeritud katkestuse tüüpimisreegli poolt, mis paneks katkestuse tüübi efektiannotatsioonis lipu f väärtuseks 1:

$$\frac{\Gamma \vdash V : A_{\mathsf{op}} \qquad \Gamma \vdash M : X\,!\,(o, i, f)}{\Gamma \vdash \downarrow \mathsf{op}\,(V, M) : X\,!\,(\mathsf{op} \downarrow (o, i), 1)}$$

Teisalt, katkestuste töötlejate tüüpimisreeglis kasutaksime lipu väärtust $f = 0$ selleks, et mitte lubada katkestusi katkestuste töötleja koodi sees:

$$\frac{\begin{array}{c}(o', \iota') \sqsubseteq_{O \times I} \iota(\mathsf{op}) \\ \Gamma, x : A_{\mathsf{op}}, r : 1 \to \langle X \rangle\,!\,(\emptyset, \{\mathsf{op} \mapsto (o', \iota', 0)\}) \vdash_{\mathsf{c}} M : \langle X \rangle\,!\,(o', \iota', 0) \\ \Gamma, p : \langle X \rangle \vdash_{\mathsf{c}} N : Y\,!\,(o, \iota, f)\end{array}}{\Gamma \vdash_{\mathsf{c}} \mathsf{promise}\ (\mathsf{op}\ x\ r \mapsto M)\ \mathsf{as}\ p\ \mathsf{in}\ N : Y\,!\,(o, \iota, f)}$$

Lisaks defineerime ka osalise järjestuse hulgal F: $f \sqsubseteq_F f' \Leftrightarrow f \leq f'$ ja muudame vastavalt ka alamtüüpimise reeglit arvutuste jaoks.

Nii seda, et kas lipu lisamine tõepoolest muudab keele laienduse tugevalt normaliseeruvaks, kui ka teiste laienduste normaliseerimisomadusi, ei ole lõputöö käigus sügavamalt uuritud.

Kokkuvõte

Töö käigus uuriti asünkroonsete algebraliste efektidega programmeerimiskeelte normaliseerimisomadusi. Selleks uuriti ja kirjeldati loogiliste relatsioonide meetodit tugeva normaliseeruvuse tõestuseks. Selle meetodi abil tõestati tugev normaliseeruvus λ-arvutuste FGCBV ja $\lambda_{æ}$ jaoks, millega saavutati edukalt töö eesmärk. Lisaks sellele arutleti töö võimalike edasiarenduste üle: leiti, et keele $\lambda_{æ}$ üks kõrgemat järku laiendus ei ole tugevalt normaliseeruv ja pakuti välja muudatus, mis võiks taastada keele tugevalt normaliseeruvuse.

Viidatud kirjandus

[Agda] The Agda Team. The Agda Wiki. https://wiki.portal.chalmers.se/agda/pmwiki.php (15.05.2025).

[Ahm24] Ahman D. Agda formalisation of the $\lambda_{æ}$-calculus. https://github.com/danelahman/higher-order-aeff-agda. 2024. (15.05.2025).

[AP21] Ahman D. and Pretnar M. Asynchronous effects. *Proc. ACM Program. Lang.* 5.POPL (Jan. 2021). DOI: 10.1145/3434305.

[AP24] Ahman D. and Pretnar M. Higher-Order Asynchronous Effects. *Logical Methods in Computer Science* Volume 20, Issue 3, 26 (Sept. 2024). DOI: 10.46298/lmcs-20(3:26)2024.

[Bar84] Barendregt H. P. The Lambda Calculus: Its Syntax and Semantics. Elsevier, 1984. DOI: 10.1016/B978-0-444-87508-2.50010-1.

[Chu32] Church A. A Set of Postulates for the Foundation of Logic. *Annals of Mathematics* 33.2 (1932), pp. 346–366. DOI: 10.2307/1968337.

[Gir72] GIRARD J.-Y. Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur. Thèse de doct. Université de Paris 7, 1972.

[How91] Howard W. A. Jean-Yves Girard, Paul Taylor, and Yves LaFont. Proofs and types. Cambridge tracts in theoretical computer science, no. 7. Cambridge University Press, Cambridge etc. 1989, xi + 176 pp. *Journal of Symbolic Logic* 56.2 (1991). DOI: 10.2307/2274726.

[KLO13] Kammar O., Lindley S., and Oury N. Handlers in action. *Proceedings of the 18th ACM SIGPLAN international conference on Functional programming* (2013). DOI: 10.1145/2500365.2500590.

[LPT03] Levy P., Power J., and Thielecke H. Modelling environments in call-by-value programming languages. *Information and Computation* 185.2 (2003), pp. 182–210. DOI: 10.1016/S0890-5401(03)00088-9.

[LS05] Lindley S. and Stark I. Reducibility and $\top\top$-Lifting for Computation Types. *Lecture Notes in Computer Science 3461*. Proceedings of Typed Lambda Calculi and Applications 2005. Apr. 2005, pp. 262–277. DOI: 10.1007/11417170_20.

[Mog91] Moggi E. Notions of computation and monads. *Information and Computation* 93.1 (1991). Selections from 1989 IEEE Symposium on Logic in Computer Science, pp. 55–92. DOI: 10.1016/0890-5401(91)90052-4.

[PP02] Plotkin G. D. and Power J. Notions of Computation Determine Monads. *Proceedings of the 5th International Conference on Foundations of Software Science and Computation Structures*. FoSSaCS '02. Berlin, Heidelberg: Springer-Verlag, 2002, pp. 342–356. DOI: 10.1007/3-540-45931-6_24.

[SA25] Sobolev I. and Ahman D. Formalisation of the strong normalisation proof for the AEff language. May 2025. DOI: 10.5281/zenodo.15419233.

[Tai67] Tait W. W. Intensional interpretations of functionals of finite type I. *Journal of Symbolic Logic* 32 (1967), pp. 198–212. DOI: 10.2307/2271658.

A. Lisa

A.1 Lemma 2.9 tõestus

Tõestus on peaaegu sama, mis Lindley ja Starki artiklis ([LS05:lk 9, Lemma 7]), välja arvatud see, et induktsiooni rakendatakse $max(K \mathbin{@} (N[V/x]))$ järgi, mitte $max(K \mathbin{@} N)$ järgi, sest $K \mathbin{@} N$ ei ole korrektne rakendus.

Lemma 2.9. *Olgu $K \in \mathbf{KRed}_Y$ jätk ja $(x)N : X \multimap Y$ term-abstraktsioon nii, et $N[V/x] \in \mathbf{CRed}_Y$ iga $V \in \mathbf{VRed}_X$ jaoks. Siis $K \circ (x)N \in \mathbf{KRed}_X$.*

Tõestus. Olgu $V \in \mathbf{VRed}_X$. Peame näitama, et

$$(K \circ (x)N) \mathbin{@} (\mathsf{return}\ V) = K \mathbin{@} (\mathsf{let}\ x = (\mathsf{return}\ V\ \mathsf{in}\ N)) \in \mathcal{SN}.$$

Kuna lemma eelduse põhjal $N[V/x] \in \mathbf{CRed}_Y$ ja $K \in \mathbf{KRed}_Y$, siis $K \mathbin{@} (N[V/x]) \in \mathcal{SN}$. Induktsiooniga $|K| + max(K \mathbin{@} (N[V/x]))$ järgi näitame, et kõik termi $K \mathbin{@} (\mathsf{let}\ x = (\mathsf{return}\ V)\ \mathsf{in}\ N)$ reduktsioonid on tugevalt normaliseeruvad. Võimalikud reduktsioonid on järgmised:

(2) $\rightsquigarrow K \mathbin{@} (N[V/x])$: see on tugevalt normaliseeruv eelduse põhjal.

(a) $\rightsquigarrow K' \mathbin{@} (\mathsf{let}\ x = (\mathsf{return}\ V)\ \mathsf{in}\ N)$, kui $K \rightsquigarrow K'$: siis $|K'| < |K|$ ja seega $max(K' \mathbin{@} (N[V/x])) < max(K \mathbin{@} (N[V/x]))$. Järelikult induktsiooni hüpoteesi põhjal $K' \mathbin{@} (\mathsf{let}\ x = (\mathsf{return}\ V)\ \mathsf{in}\ N) \in \mathcal{SN}$.

(a) $\rightsquigarrow K' \mathbin{@} (\mathsf{let}\ x = (\mathsf{return}\ V)\ \mathsf{in}\ (\mathsf{let}\ y = N\ \mathsf{in}\ M))$, kui $K = K' \circ (y)M$ ja $x \notin fv(M)$: siis $|K'| < |K|$ ja

$$K' \mathbin{@} ((\mathsf{let}\ y = N\ \mathsf{in}\ M)[V/x]) = K' \mathbin{@} (\mathsf{let}\ y = N[V/x]\ \mathsf{in}\ M) = K \mathbin{@} (N[V/x]),$$

mis on tugevalt normaliseeruv eelduse põhjal, ja seega

$$|K'| + max(K' \mathbin{@} ((\mathsf{let}\ y = N\ \mathsf{in}\ M)[V/x])) < |K| + max(K \mathbin{@} (N[V/x])).$$

Järelikult induktsiooni hüpoteesi põhjal $K' \mathbin{@} (\mathsf{let}\ x = (\mathsf{return}\ V)\ \mathsf{in}\ (\mathsf{let}\ y = N\ \mathsf{in}\ M)) \in \mathcal{SN}$. □

A.2 Lemma 2.14 tõestus

Tõestame kõigepealt järgmise väite:

Lemma A.2. *Olgu $V : X, W : Y$ väärtused, $(x, y)N : [X, Y] \multimap Z$ term-abstraktsioon ja $K : Z \multimap W$ jätk nii, et $K \mathbin{@} (N[V/x, W/y]) \in \mathcal{SN}$. Siis $K \mathbin{@} (\mathsf{match}\ (V, W)\ \mathsf{with}\ \{(x, y) \mapsto N\}) \in \mathcal{SN}$.*

Tõestus. Näitame induktsiooniga $|K|$ järgi, et kõik termi

$$K \mathbin{@} (\mathsf{match}\ (V, W)\ \mathsf{with}\ \{(x, y) \mapsto N\})$$

reduktsioonid on tugevalt normaliseeruvad. Võimalikud reduktsioonid on järgmised:

(3) $\rightsquigarrow K \mathbin{@} N[V/x, W/y]$: see on tugevalt normaliseeruv lemma eelduse põhjal.

(a) $\rightsquigarrow K' \mathbin{@} (\mathsf{match}\ (V,W)\ \mathsf{with}\ \{(x,y) \mapsto N\})$, kui $K \rightsquigarrow K'$: siis $|K'| < |K|$ ja $K' \mathbin{@} (N[V/x, W/y]) \in \mathcal{SN}$. Järelikult induktsiooni hüpoteesi põhjal $K' \mathbin{@} (\mathsf{match}\ (V,W)\ \mathsf{with}\ \{(x,y) \mapsto N\}) \in \mathcal{SN}$. □

Nüüd saame tõestada lemma 2.14.

Lemma 2.14. *Olgu $V \in \mathbf{VRed}_{X \times Y}$ ja $(x,y)N : [X,Y] \multimap Z$ nii, et $N[V/x, W/y] \in \mathbf{CRed}_Z$ iga $V \in \mathbf{VRed}_X$ ja $W \in \mathbf{VRed}_Y$ korral. Siis $(\mathsf{match}\ V\ \mathsf{with}\ \{(x,y) \mapsto N\}) \in \mathbf{CRed}_Z$.*

Tõestus. Olgu $K \in \mathbf{KRed}_Z$. Peame näitama, et $K \mathbin{@} (\mathsf{match}\ V\ \mathsf{with}\ \{(x,y) \mapsto N\}) \in \mathcal{SN}$. Kuna $K \mathbin{@} (\mathsf{match}\ V\ \mathsf{with}\ \{(x,y) \mapsto N\}) = (K \circ ((x,y)N)_\times) \mathbin{@} V$ ja $V \in \mathbf{VRed}_{X \times Y}$, siis selleks piisab, kui $K \circ ((x,y)N)_\times \in \mathbf{PRed}_{X,Y}$, ehk et $(K \circ ((x,y)N)_\times) \mathbin{@} (V,W) \in \mathcal{SN}$ iga $V \in \mathbf{VRed}_X$ ja $W \in \mathbf{VRed}_Y$ korral. See kehtib lemma A.2 põhjal, sest $K \mathbin{@} (N[V/x, W/y]) \in \mathcal{SN}$ (kuna $K \in \mathbf{KRed}_Z$ ja lemma eelduse põhjal $N[V/x, W/y] \in \mathbf{CRed}_Z$). □

A.3 Teoreemi 2.17 detailne tõestus

Teoreem 2.17. *Olgu $T : Z$ hästi tüübitud väärtus või arvutus, mille vabad muutujad on $x_1 : X_1, \ldots, x_n : X_n$. Siis iga $V_1 \in \mathbf{VRed}_{X_1}, \ldots, V_n \in \mathbf{VRed}_{X_n}$ korral kas $T[V_1/x_1, \ldots, V_n/x_n] \in \mathbf{VRed}_Z$, kui T on väärtus, või $T[V_1/x_1, \ldots, V_n/x_n] \in \mathbf{CRed}_Z$, kui T on arvutus.*

Tõestus. Ruumi kokkuhoiu nimel kirjutame $V_1/x_1, \ldots, V_n/x_n$ asemel $\overline{V}/\overline{x}$. Tõestame teoreemi induktsiooniga termi T struktuuri järgi:

- Kui $T = z$, siis leidub indeks i nii, et $z = x_i$. Eelduse põhjal $V_i \in \mathbf{VRed}_Z$ ja seega $T[\overline{V}/\overline{x}] = V_i \in \mathbf{VRed}_Z$.
- Kui $T = ()$, siis $T[\overline{V}/\overline{x}] = ()$. Lemma 2.5 põhjal $T[\overline{V}/\overline{x}] \in \mathbf{VRed}_1$.
- Kui $T = (V,W)$, kus $V : X$ ja $W : Y$, siis $T[\overline{V}/\overline{x}] = (V[\overline{V}/\overline{x}], W[\overline{V}/\overline{x}])$. Induktsiooni hüpoteesi põhjal $V[\overline{V}/\overline{x}] \in \mathbf{VRed}_X$ ja $W[\overline{V}/\overline{x}] \in \mathbf{VRed}_Y$. Järelikult lemma 2.3 põhjal $T[\overline{V}/\overline{x}] \in \mathbf{VRed}_{X \times Y}$.
- Kui $T = \mathsf{inl}_Y\, V$, kus $V : X$, siis $T[\overline{V}/\overline{x}] = \mathsf{inl}_Y\, (V[\overline{V}/\overline{x}])$. Induktsiooni hüpoteesi põhjal $V[\overline{V}/\overline{x}] \in \mathbf{VRed}_X$. Järelikult lemma 2.4 põhjal $T[\overline{V}/\overline{x}] \in \mathbf{VRed}_{X+Y}$.
- Juht $T = \mathsf{inr}_X\, W$ on sümmeetriline eelmise juhuga.
- Kui $T = \mathsf{fun}\ (x : X) \mapsto N$, kus $N : Y$, siis $T[\overline{V}/\overline{x}] = \mathsf{fun}\ (x : X) \mapsto (N[\overline{V}/\overline{x}])$. Induktsiooni hüpoteesi põhjal $N[\overline{V}/\overline{x}, W/x] \in \mathbf{CRed}_Y$ iga $W \in \mathbf{VRed}_X$ jaoks. Järelikult lemma 2.6 põhjal $T[\overline{V}/\overline{x}] \in \mathbf{VRed}_{X \to Y}$.
- Kui $T = \mathsf{return}\, V$, siis $T[\overline{V}/\overline{x}] = \mathsf{return}\, (V[\overline{V}/\overline{x}])$. Induktsiooni hüpoteesi põhjal $V[\overline{V}/\overline{x}] \in \mathbf{VRed}_Z$. Järelikult lemma 2.8 põhjal $\mathsf{return}\, (V[\overline{V}/\overline{x}]) \in \mathbf{CRed}_Z$.

- Kui $T = \mathsf{let}\ x = N\ \mathsf{in}\ L$, kus $N : X$ ja $L : Z$, siis induktsiooni hüpoteesi põhjal $N[\overline{V}/\overline{x}] \in \mathbf{CRed}_X$ ja $L[\overline{V}/\overline{x}, V/x] \in \mathbf{CRed}_Z$ iga $V \in \mathbf{VRed}_X$ jaoks. Järelikult lemma 2.10 põhjal $T[\overline{V}/\overline{x}] = (\mathsf{let}\ x = N[\overline{V}/\overline{x}]\ \mathsf{in}\ L[\overline{V}/\overline{x}]) \in \mathbf{CRed}_Z$.
- Kui $T = \mathsf{match}\ V\ \mathsf{with}\ \{(x, y) \mapsto N\}$, kus $V : X \times Y$, siis

  $$T[\overline{V}/\overline{x}] = \mathsf{match}\ V[\overline{V}/\overline{x}]\ \mathsf{with}\ \{(x, y) \mapsto N[\overline{V}/\overline{x}]\}.$$

  Induktsiooni hüpoteesi põhjal $V[\overline{V}/\overline{x}] \in \mathbf{VRed}_{X\times Y}$ ja $N[\overline{V}/\overline{x}, V/x, W/y] \in \mathbf{CRed}_Y$ iga $V \in \mathbf{VRed}_X$ ja $W \in \mathbf{VRed}_Y$ korral. Järelikult lemma 2.14 põhjal $T[\overline{V}/\overline{x}] \in \mathbf{CRed}_Z$.
- Juht $T = \mathsf{match}\ V\ \mathsf{with}\ \{\mathsf{inl}\ x \mapsto M, \mathsf{inr}\ y \mapsto N\}$ on sarnane eelmise juhuga, kasutades lemma 2.14 asemel nüüd lemmat 2.13.
- Kui $T = \mathsf{match}\ V\ \mathsf{with}\ \{\}_Z$, siis $T[\overline{V}/\overline{x}] = \mathsf{match}\ V[\overline{V}/\overline{x}]\ \mathsf{with}\ \{\}_Z$. Indukstiooni hüpoteesi põhjal $V[\overline{V}/\overline{x}] \in \mathbf{VRed}_0$. Järelikult lemma 2.15 põhjal $T[\overline{V}/\overline{x}] \in \mathbf{CRed}_Z$.
- Kui $T = V\,W$, kus $V : X \to Z$ ja $W : X$, siis $T[\overline{V}/\overline{x}] = V[\overline{V}/\overline{x}]\,W[\overline{V}/\overline{x}]$ ja induktsiooni hüpoteesi põhjal $V[\overline{V}/\overline{x}] \in \mathbf{VRed}_{X\to Z}$ ja $W_2[\overline{V}/\overline{x}] \in \mathbf{VRed}_X$. Järelikult lemma 2.16 põhjal $T[\overline{V}/\overline{x}] \in \mathbf{CRed}_Z$. □

Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Ilja Sobolev,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose **Asünkroonsete algebraliste efektidega programmeerimiskeelte normaliseerimisomadused**, mille juhendaja on **Danel Ahman**, reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada Tartu Ülikooli digitaalarhiivi kuni autoriõiguse kehtivuse lõppemiseni;
2. annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi kaudu Creative Commonsi litsentsiga CC BY NC ND 4.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni;
3. olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile;
4. kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Ilja Sobolev

15.05.2025