Tagskiptur lambda-reikningur

Bjarki Baldursson Harksen

14. apríl 2023

Útdráttur

Lambda-reikningur er grundvöllur fallaforritunar, sem snýst um samsetningu falla til þess að fá fram niðurstöðu, frekar en notkun reiknirita og hliðarverkana eins og í hefðbundnari forritun. Við munum setja fram formlega skilgreiningu á lambda-reikningi, ásamt reikningi hans og skoða þannig fræðilega undirstöðu fallaforritunar. Við sýnum fram á jafngildi lambda-reiknings og annarra reikningslíkana. Í kjölfarið er tagskipting lambda-reiknings með einföldum tögum skoðuð og sýnt hvernig hún, með hjálp stöðlunarsetningarinnar, getur komið í veg fyrir villulaust forrit. Jafnframt er höfuðtagsreikniritið sett fram, en það gerir kleift að álykta um tögun í forriti þótt tög séu ekki beint tilgreind. Að lokum eru stuttlega ræddar tvær útvíkkanir einfaldra taga: fjölmóta tög og Hindley-Milner-tagskipting.

1 Inngangur

1.1 Fallaforritun

Frá upphafi forritunariðkunar hafa flest forritunarmál byggt á hugmyndafræði þar sem áhersla er lögð á reiknirit og framkvæmd þeirra, ásamt þeim ástandsbreytingum sem þessu fylgir. Þetta er ekki alls kostar óeðlilegt — uppbygging tölva er vel til slíkra reikninga fallin, og auk þess eru þeir almennt auðveldari en annað fyrir mennskt fólk að skilja.

Þessi hugmyndafræði verður hins vegar hættulegri því stærri sem forrit verða, til dæmis fyrir textaritilinn sem þetta skjal er skrifað með, eða PDF-lesarann sem það er (hugsanlega) lesið með. Erfiðara verður fyrir forritara að halda í huga hvernig ástand forritsins er skipulagt og hvernig það breytist á meðan á keyrslu þess stendur. Þetta gerir líklegra að forritsvillur fari fram hjá mönnum, og í allra verstu tilfellum geta þær haft mikinn skaða í för með sér.

Til eru ýmsar aðferðir sem draga úr slíkum villum, og hefur stór hluti þróunar forritunarmála farið í að innleiða margs konar lausnir á þessum vanda.

Ákveðinn flokkur forritunarmála, nefnd *fallamál*, reynir hins vegar að sneiða hjá vandanum með því að beita annarri hugmyndafræði. Á meðan hin fyrrnefndu hefðbundnu forritunarmál, oft nefnd *gildingarmál*, ýta undir hliðarverkanir til þess að útfæra reiknirit, þá forðast fallamál hliðarverkanir í lengstu lög eða banna þær jafnvel alveg.

Eins og nafngiftin bendir til nota fallamál föll sem frumeiningu í forritum sínum. Það fer eftir málinu hversu náskyld þau eru hinu stærðfræðilega hugtaki um fall. Þau mál sem lengst fara, svonefnd *hrein* fallamál, reyna að hafa algjört samræmi þar á milli: hvert inntak gefur ávallt sama úttak, óháð ástandi. Föll má líka finna í gildingarmálum, en eru þá í stórum dráttum ótengd stærðfræðilegum föllum, og þjóna frekar sem aðferð til skipulagningar og hjúpunar forrits. Stundum er meira að segja notað orðið *stefja* í stað falls, því til frekari áherslu.

Forritun með fallamálum byggir meira á því að tengja saman föll til þess að fá upp ákveðna niðurstöðu, frekar en að setja fram í skrefum hvernig hana skal finna. Tökum sem dæmi forrit sem finnur summu allra ferningstala sem eru minni en 1000. Í gildingarmálinu Python gæti þetta litið út eins og að neðan:

```
>>> sum_of_squares = 0
>>> k = 0
>>> while k*k < 1000:
...     sum_of_squares += k*k
...     k += 1
...
>>> sum_of_squares
10416
```

Hvorki forritara þessa forrits né lesanda þess er hér komið á óvart. Þetta samræmist bæði vel við þær aðgerðir sem tölvan framkvæmir þegar forritið er keyrt og við þá aðferð sem líklegast væri notuð ef þetta þyrfti að reikna í höndunum.

Ef við skoðum sama forrit skrifað í hreina fallamálinu Haskell sést greinilegur munur:

```
ghci> sumOfSquares = (sum . takeWhile (<1000) . map (^2)) [0..]
ghci> sumOfSquares
10416
```

Þetta krefst meiri umhugsunar en Python-forritið að ofan, bæði í skrift og í lestri. Hér er búið til fall sem verkar á (óendanlegan) lista náttúrlegra talna, með samskeytingu þriggja falla sem saman hafa óskuð áhrif.

Á hinn bóginn er þetta bæði styttra og stöðugra en Python hliðstæðan. Engin hætta er hér á að breytur séu misnotaðar eða ekki skilgreindar áður en þær eru notaðar — það eru engar breytur yfirhöfuð! Það er auk þess auðveldara að betrumbæta forritið ef þarfir þess breytast. Ef við til dæmis viljum geta fundið summu allra ferningstala sem eru minni en n fyrir hvaða n sem vera skal, þarf í Haskell-forritinu ekki nema að bæta við stika:

```
ghci> sumOfSquares n = (sum . takeWhile (<n) . map (^2)) [0..]
ghci> sumOfSquares 1000
10416
```

Þótt það sé ekki erfitt að gera svipaða breytingu í Python-forritinu, er þá um leið stærri snertiflötur fyrir villur að spretta upp, t.d. með því að gleyma að færa breytuskilgreiningu inn í fallið/stefjuna.

1.2 Tagskiptingar

Í forritun er stöðugt unnið með gögn af mörgum mismunandi gerðum, eins og til dæmis heiltölur, fleytitölur, strengi, satt og ósatt, lista, mengi og tré svo að eitthvað sé nefnt. Samanlagt hafa ómældar vinnustundir og gífurlegar fjárhæðir farið í að laga villur í forritum sem tengjast því að óvart sé gefið falli/stefju inntak sem er af vitlausri gerð, og að þannig hrynji forritið. Í Python getur þetta meira að segja gerst í hinum einföldustu tilvikum:

```
>>> def S(n):
...     return n + 1
...
>>> S("0")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "<stdin>", line 2, in S
TypeError: can only concatenate str (not "int") to str
```

Þessi skilaboð gefa vísbendingu um algenga lausn þessa vanda: *tagskiptingu* (e. *type system*). Með tagskiptingu forritunarmála er hverju gildi gefið ákveðið *tag* (e. *type*) sem segir til um af hvaða gerð það er, og forritara gert viðvart ef hann reynir að nota gildi af einni gerð þar sem búist er við gildi af annarri gerð. Við getum til dæmis skilgreint fallið `S` að ofan með

```
>>> def S(n: int) -> int:
...     return n + 1
```

og gert þannig grein fyrir að `S` sé einungis vel skilgreint fyrir heiltöluviðföng (táknað `int`), og skili jafnframt heiltölu. Ef reynt er að kalla á `S` með `S("0")` eins og áður, þá gefur forritunarumhverfið upp þá villu að tag gildisins `"0"` (þ.e. `str`), sé ekki í samræmi við viðfang `S` (þ.e. `int`). Þetta eykur þannig líkurnar á að tekið sé eftir slíkum villum áður en forritið kemst í hendur notenda.

Hér ætti að nefna að tagskiptingar forritunarmála eru misstrangar, og að þær leysa því mismörg vandamál. Langflest forritunarmál hafa tagskiptingu í einhverri mynd, en sum þeirra leyfa breytum að skipta um tög í miðri keyrslu, önnur ekki; sum krefjast þess að viðfangs- og skilatög falla/stefja séu gefin upp og virt, önnur nota þau einungis sem vísbendingar; sum kanna að samræmi sé í tögum við þýðingu forrits, önnur láta athugasemdir til forritara nægja. Tagskiptingar forritunarmála eru því oft flokkaðar í *veikar* og *sterkar* tagskiptingar.

Ókosturinn við sterkar tagskiptingar er að oft verða forrit lengri og erfiðari að skrifa, og krefjast meiri vinnu til þess að viðhalda. Þessi fórnarskipti eru hins vegar oft vel þess virði. Til þess að sjá það þarf ekki nema að bera saman forritunarmálin JavaScript og TypeScript. JavaScript er meðal mest notuðu forritunarmála heimsins. Það notar veika tagskiptingu með litla sem enga tagskoðun, og er alræmt fyrir að hafa margar hálfdularfullar reglur hvað varðar notkun aðgerða með gildi af mismunandi gerðum. TypeScript er hrein viðbót við JavaScript sem bætir við sterkri tagskiptingu og þýðingarskrefi, og þrátt fyrir að flækjustig á notkun þess sé hærra en hjá Javascript, þá er það mun betur þokkað: Í árlegri könnun Stack Overflow 2022 [14, Technology — Most loved, dreaded, and wanted — Programming, scripting and markup languages] naut TypeScript 4. sætis á lista yfir ástsælustu forritunarmálin, á meðan JavaScript sat í 16. sæti.

Í sumum tilvikum er hægt að draga úr ókostum strengri tagskiptinga án þess að fórna því öryggi sem þær ná fram. Algeng leið til þess er svokölluð *ályktun* tags (e. *type inference*). Þá er forritara leyft að sleppa því að setja fram tag gildis samhliða skilgreiningu þess (eins og má í veikari tagskiptingum), og almennasta tag gildisins fundið sjálfkrafa. Þetta getur þannig auðveldað forritara lífið ef hann þarf að skilgreina mikið af gildum sem hafa augljóst tag, en léttir ekki á kröfum strangra tagskiptinga að tag sé aldrei notað þar sem það ekki má. Í fallamáli eins og Haskell er þetta sérstaklega nytsamlegt, en þar er oft skilgreindur aragrúi af einföldum hjálparföllum sem svo eru tengd saman. Að setja fram tög allra þeirra myndi allt að tvöfalda línufjölda forritstextans.

Skoðum til dæmis eftirfarandi (óbestað) fall í Haskell sem finnur öll tvíburatvinntölupör minni en m, þ.e. pör frumtalna (p, q), þar sem $q = p + 2 \leq m$. Í forritinu munum við skilgreina nokkur hjálparföll og gildi, og gefa jafnframt upp tag hvers þeirra (með línum sem innihalda `::`).

```
twinPrimes :: Integer -> [(Integer, Integer)]
twinPrimes m = filter bothArePrime twinPairs
  where twinPairs :: [(Integer, Integer)]
        twinPairs = zip [1..m-2] [3..m]
        bothArePrime :: (Integer, Integer) -> Bool
        bothArePrime (p, q) = isPrime p && isPrime q
        isPrime :: Integer -> Bool
        isPrime n = null $ filter (\k -> n `mod` k == 0) [2..n-1]
```

Ef við sleppum nú hins vegar öllum tagsetningum, þá fæst talsvert styttra forrit (helmingur línanna). Haskell ályktar þá tög allra gildanna í skilgreiningunni, og fær meira að segja almennari niðurstöðu en við þegar við tilgreindum `twinPrimes :: Integer -> [(Integer, Integer)]`:

```
ghci> :type twinPrimes
twinPrimes :: Integral a => a -> [(a, a)]
```

Fallið `twinPrimes` er nú vel skilgreint fyrir öll tög sem „hegða sér“ eins og heiltölur, frekar en þau sem hafa tiltekna heiltölutagið `Integer`.

1.3 Reikningslíkön og lambda-reikningur

Hugtökin *reikningur* og *reiknanleiki* eru ekki torskilin, en erfitt er að skilgreina nákvæmlega hvað í þeim felst, að minnsta kosti í samræmi við venjulegan óformlegan skilning þeirra. Á hinn bóginn má tjá hugmyndir um reikning með *reikningslíkönum*, sem þá setja fram reglur um umbreytingu eða úrlausn frumeininga sinna, og gera þannig þessi hugtök nákvæm innan í líkaninu. Sérhvert forritunarmál felur í sér reikningslíkan, þar sem keyrsla forrita samsvarar reikningi þeirra. Slík líkön skiptast almennt í tvo flokka: líkön sem byggja á *Turing-vélum* Alan Turings og líkön sem byggja á *lambda-reikningi* Alonzo Church. Til að kanna fræðilega eiginleika forritunar er því nauðsynlegt að skoða þessi líkön sérstaklega. Hér má einnig nefna reikningslíkan *hlutrakinna falla*, en þau eru sjaldan nefnd sem grundvöllur reikningslíkans fyrir forritunarmál.

Líkön Turing-véla og lambda-reiknings eru einföldustu reikningslíkönin sem ná yfir hugmyndafræðirnar úr kafla 1.1; það fyrrnefnda er kennt við gildingarmál, en það seinna við fallamál. Líkönin komu bæði á sjónarsviðið árið 1936, og sama ár var sýnt fram á að þau væru jafngild, þ.e.a.s. að reikning í einu þeirra mætti setja fram í hinu, og öfugt. Þetta hefur í för með sér að gildingarmál og fallamál eru nákvæmlega jafn öflug, þrátt fyrir gerólíkar hugmyndir þeirra.

Við munum í þessu riti einbeita okkur að lambda-reikningi og eiginleikum hans. Á meðan stærðfræðileg framsetning Turing-véla er tiltölulega fjarri raunverulegri útfærslu gildingarmála, þá er lambda-reikningur keimlíkur fallamálum. Jafnvel má hugsa um lambda-reikning sem hið einfaldasta mögulega fallamál, og við munum sjá nokkur dæmi um þá hliðstæðu hér að neðan. Rétt eins og fyrir fallamál, þá má túlka frumeiningar lambda-reiknings, svokallaðar λ-segðir, sem föll, og að blanda λ-segðum saman og reikna upp úr þeim minnir heilmikið á forritun með fallamálum. Þess til stuðnings má skoða eitt fyrsta forritunarmál heimsins (og fyrsta fallamálið), LISP, birt árið 1960 af John McCarthy. Það notar mjög svipaðan rithátt og lambda-reikningur, og má í stórum dráttum líta á sem viðbót við hann.

Önnur ástæða fyrir að við skoðum lambda-reikning er að þægilegt er að skilgreina tagskiptingu hans. Að skoða eiginleika mismunandi tagskiptinga fyrir forritunarmál er því einfalt að gera í gegnum lambda-reikning, og niðurstöður þar má þá auðveldlega notfæra sér í almennari tilvikum. Tagskiptur lambda-reikningur er jafnvel notaður til þess að skoða tengsl tagskiptinga og rökfræði, sbr. Curry-Howard-samsvörunina [1], en við förum ekki nánar í það hér.

Í kafla 2, sem byggir í stórum dráttum á fyrstu köflum bókar Hindley og Seldin [9], munum við setja fram formlega skilgreiningu á lambda-reikningi, ásamt reikningi hans. Við munum sýna að þessi reikningur sé vel skilgreindur miðað við almennan skilning á hugtakinu „reikningur,“ og við munum sýna að lambda-reikningur sé jafngildur öðrum reikningslíkönum, sér í lagi *hlutröknum föllum*.

Í kafla 3, sem byggir mestmegnis á bókum Hindleys [8] og Barendregts [3], munum við setja fram tagskiptingu lambda-reiknings með *einföldum tögum*. Þar skoðum við þær reglur sem nota má til þess að gefa λ-segðum tög, og sýnum hvernig *stöðlunarsetningin* (3.15) felur í sér að tagskipting forrita tryggir að þau séu laus við ákveðinn algengan flokk forritunarvillna. Við sýnum einnig *höfutagsreikniritið* (3.28) sem nota má til þess að álykta tag λ-segða.

Í kafla 4 gefum við að lokum yfirlit yfir tvær tagskiptingar lambda-reiknings í viðbót, annars vegar með *fjölmóta tögum* og hins vegar með Hindley-Milner-tagskiptingu. Við sýnum *reiknirit W* (4.10) sem nota má til tagályktunar í því síðarnefnda.

2 Lambda-reikningur

2.1 Formlegt mál fyrir lambda-reikning

Í mörgum umfjöllunum um forritunarmál og eiginleika þeirra eru svokölluð *formleg mál* tekin fyrir. Hægt er að skilgreina formleg mál á nokkra vegu, en einfaldast er að líta á mál sem safn af orðum sínum, sem samanstanda af táknum úr einhverju stafrófi. Til dæmis má skilgreina formlegt mál fyrir náttúrlegar tölur skrifaðar á ensku sem mengið $\{\text{‘zero’}, \text{‘one’}, \text{‘two’}, \ldots\}$. Þetta mál hefur sem stafróf enska stafrófið $\{\text{‘a’}, \text{‘b’}, \ldots, \text{‘z’}\}$.

Við setjum þetta formlega fram í næstu skilgreiningu.

Skilgreining 2.1. Látum Σ vera mengi tákna. *Formlegt mál* $\mathcal{L}$ yfir *stafrófið* Σ er mengi endanlegra runa í Σ. Endanleg runa í Σ nefnist *stæða*, og ef hún er innihaldin í $\mathcal{L}$ nefnist hún einnig *segð* málsins $\mathcal{L}$, eða $\mathcal{L}$*-segð*.

Fyrir formlegt mál $\mathcal{L}$ táknum við stafróf þess með $\Sigma(\mathcal{L})$.

Oft er gerð sú krafa að segðir formlegs máls $\mathcal{L}$ megi mynda í endanlega mörgum skrefum með einhverjum *myndunarreglum*. Við leggjum ekki þá kröfu hér, en þó má nefna að öll mál sem við munum skoða í þessu riti uppfylla þetta.

Athugasemd 2.2. Þegar formleg mál eru tekin fyrir í stærðfræðilegum umræðum þarf oft að gæta sín á að ruglast ekki á *viðfangsmálinu* sjálfu og á *yfirmálinu*, sem notað er til þess að tala um viðfangsmálið. Ef við til dæmis ritum „$x = y$“ mætti þetta túlka á tvo vegu: Ef þetta á við um viðfangsmálið, þá táknar þetta einfaldlega táknrununa (‘x’, ‘$=$’, ‘y’), og hefur enga æðri merkingu á yfirmálinu. Á hinn bóginn má á yfirmálinu skilja þetta sem staðhæfinguna „x er segð, y er segð og x og y eru sama segðin“.

Til þess að draga úr slíkum misskilningi lánum við í framhaldinu rithátt úr rökfræðibók Reynis Axelssonar [12]. Nánari umræðu má sjá í [12, Kafli I, §2]. Í stuttu máli:

Við notum í einfaldar enskar gæsalappir til þess að afmarka tiltekin tákn og feitletraða lágstafi fyrir táknbreytur. Með því að setja $\mathbf{x}$ sem ‘a’ megum við þannig rita $\mathbf{x}$ í stað fyrsta bókstafs enska stafrófsins. Við leyfum okkur einnig að rita tákn hlið við hlið til þess að tala um tilsvarandi stæðu. Þá sleppum við að rita gæsalappir utan um stök tákn en setjum þær utan um stæðuna ef engin táknbreyta kemur þar fyrir. Þannig má bæði rita ‘abc’ og $\mathbf{x}bc$ fyrir stæðu fyrsta, annars og þriðja bókstafs enska stafrófsins. Við notum feitletraða hástafi fyrir stæðubreytur og leyfum okkur að rita slíkar breytur við hlið annara tákna til að tilgreina stæður. Ef við látum til dæmis $\mathbf{A}$ vera stæðuna ‘xyz’ þá má rita $a\mathbf{A}b$ í stað ‘$axyzb$’.

Í rökfræðitexta er meiri þörf fyrir þetta en hjá okkur; við munum aðeins skoða tiltekin, vel skilgreind mál, með þekkt stafróf. Við leyfum okkur því að nota táknið ‘=’ (og þá einnig ‘:=’), á yfirmálinu til að miðla samsemd. Þannig ritum við $\mathbf{A} = \mathbf{B}$ í merkingunni „**A** og **B** eru sama segðin“. Við gerum hið sama fyrir önnur vensl sem skilgreind verða hér á eftir.

Með formlegum málum fylgja oft *skammstafanir* sem nota má til þess að auka lesanleika segða málsins. Sem dæmi má nefna svigasetningarreglur: í máli einfaldra stærðfræðiformúla má til dæmis nota ‘$1+2\cdot 3+4$’ sem skammstöfun fyrir ‘$((1+(2\cdot 3))+4)$’. Þegar skammstafanir eru í notkun notum við enn gæsalappir þrátt fyrir að tilsvarandi táknruna sé ekki innihaldin í málinu.

Við setjum nú fram lambda-reikning sem formlegt mál. Þetta er ólíkt venju. Flest rit um lambda-reikning sleppa því að skilgreina lambda-reikning á þennan hátt, og hugsa um segðir lambda-reiknings sem sjálfstæð stærðfræðileg fyrirbæri. Rithátturinn sem settur er fram í athugasemd 2.2 er því þar sjaldséður. Hins vegar lítum við hér á að lambda-reikningur sé eins og einfalt forritunarmál, og í því samhengi er sjálfsagt að skoða formleg mál.

Í skilgreiningum formlegra mála (eins og hér að neðan) gerum við ráð fyrir að sé stafróf þess ritað $A \cup B$, þá séu táknmengin A og B sundurlæg.

Skilgreining 2.3. Formlegt mál λ fyrir *lambda-reikning* er skilgreint yfir stafrófið $\mathcal{V} \cup \{‘\lambda’, ‘.’, ‘(’, ‘)’\}$, þar sem $\mathcal{V}$ er teljanlega óendanlegt mengi *breyta*. Segðir málsins eru gefnar með eftirfarandi myndunarreglum:

(i) Stæða af gerðinni $\mathbf{x}$ þar sem $\mathbf{x}$ er breyta er λ-segð.

(ii) Ef $\mathbf{M}$ er λ-segð og $\mathbf{x}$ er breyta, þá er $(\lambda\mathbf{x}.\,\mathbf{M})$ λ-segð.

(iii) Ef $\mathbf{M}$ og $\mathbf{N}$ eru λ-segðir, þá er $(\mathbf{M}\,\mathbf{N})$ λ-segð.

Segð sem fæst með beitingu myndunarreglu (ii) nefnist *alhæfingarsegð* og segð sem fæst með beitingu myndunarreglu (iii) nefnist *beitingarsegð*.

Við sleppum því að tilgreina nákvæmlega breytutáknin í $\mathcal{V}$, en við gerum hins vegar ráð fyrir því að allir lágstafir enska stafrófsins komi þar fyrir og leyfum táknum að vera tölusett með náttúrlegri tölu. Til dæmis notum við táknin ‘x’, ‘x_0’, ‘x_1’, … sem breytur. Lesandi má sjálfur ráða hvaða tákn eru viðeigandi umfram þessi.

Munum úr kafla 1.3 að líta má á lambda-reikning sem hið einfaldasta fallaforritunarmál, en lambda-reikningur inniheldur í rauninni ekkert nema föll. Til þess að samræmast almennum skilningi á falli þurfum við bæði að hafa leið til þess að skilgreina fall og leið til þess að beita falli á viðfang. Alhæfingasegð $(\lambda\mathbf{x}.\,\mathbf{M})$ er ætluð til þess að tákna skilgreiningu falls af breytistærð $\mathbf{x}$ sem gefið er með formúlunni $\mathbf{M}$, og beitingarsegð $(\mathbf{M}\,\mathbf{N})$ táknar þá beitingu fallsins sem gefið er með $\mathbf{M}$ á viðfangið $\mathbf{N}$.

Dæmi 2.4. Skoðum eftirfarandi λ-segðir:

- $\mathbf{I} :=$ '$(\lambda x.\, x)$'.
- $\mathbf{K} :=$ '$(\lambda x.\, (\lambda y.\, x))$'.
- $\mathbf{C} :=$ '$(\lambda f.\, (\lambda g.\, (\lambda x.\, (f\, (g\, x)))))$'.
- $\mathbf{W} :=$ '$(\lambda f.\, (\lambda x.\, ((f\, x)\, x)))$'.

Við hugsum okkur að segðin $\mathbf{I}$ standi fyrir samsemdarvörpunina $I(x) = x$. Á sama hátt hugsum við að segðin $(\mathbf{K}\, x)$ tákni fastafallið $K_x(y) = x$, að segðin $((\mathbf{C}\, f)\, g)$ tákni samskeytinguna $f \circ g$, og að segðin $(\mathbf{W}\, f)$ tákni fallið $W_f(x) = f(x, x)$.

Athugasemd 2.5. Tökum eftir að skilgreining lambda-reiknings leyfir ekki að föll séu skilgreind með fleiri en eitt viðfang. Þetta er hins vegar mjög algengt bæði víðar í stærðfræði og í forritun. Við komumst hjá þessum takmörkunum með því að leyfa falli að skila öðru falli.

Skoðum til dæmis segðina $\mathbf{K}$ úr dæmi 2.4. Þar hugsuðum við um segðina $(\mathbf{K}\, x)$ að standa fyrir fallið $K_x : A \to A$, $K_x(y) = x$, en einnig má hugsa sér að $\mathbf{K}$ standi fyrir fallið $K : A \times A \to A$, $K(x, y) = x$. Þar sem þetta fall er tvístætt er ekki hægt að setja það fram beint með λ-segð, en við getum búið til jafngilt fall $K' : A \to (A \to A)$, $K'(x)(y) = x$, þar sem $A \to A$ táknar mengi allra falla af gerðinni $A \to A$. Svipað má gera fyrir segðirnar $\mathbf{C}$ og $\mathbf{W}$.

Það reynist vera mjög algengt að þurfa að tákna fall af mörgum breytistærðum í lambda-reikningi — nógu algengt til þess að við tökum upp sérstakar skammstafanir til þess að auka skilning á slíkum segðum:

(i) Í stað $(\lambda\mathbf{x}.\, (\lambda\mathbf{y}.\, \mathbf{M}))$ ritum við $(\lambda\mathbf{x}\mathbf{y}.\, \mathbf{M})$.

(ii) Í stað $((\mathbf{M}\, \mathbf{N}_1)\, \mathbf{N}_2)$ ritum við $(\mathbf{M}\, \mathbf{N}_1\, \mathbf{N}_2)$.

Við leyfum okkur einnig að nota þrípunkt þegar fjöldi viðfanga er mikill eða óþekktur, t.d. með $(\lambda\mathbf{x}_1 \ldots \mathbf{x}_k.\, \mathbf{M})$ og $(\mathbf{M}\, \mathbf{N}_1\, \cdots\, \mathbf{N}_k)$.

Með þessum skammstöfunum má þannig rita

- $\mathbf{K} =$ '$(\lambda xy.\, x)$'.
- $\mathbf{C} =$ '$(\lambda fgx.\, (f\, (g\, x)))$'.
- $\mathbf{W} =$ '$(\lambda fx.\, (f\, x\, x))$'.

Oft er einnig æskilegt að beita sama falli mörgum sinnum á eitthvert viðfang. Í mörgum stærðfræðiritum er tekinn upp rithátturinn $f^n = \underbrace{f \circ \cdots \circ f}_{n \text{ sinnum}}$ Við tökum því upp eina skammstöfun enn:

(iii) Í stað $(\underbrace{\mathbf{M} \cdots (\mathbf{M}\,(\mathbf{M}}_{n \text{ sinnum}}\, \mathbf{N})))$ ritum við $(\mathbf{M}^n\, \mathbf{N})$.

Margir höfundar leyfa einnig að sleppa svigasetningum þegar ljóst er hver segðin er í raun. Við tökum ekki upp þá venju hér.

2.2 Einföldun og jafngildi λ-segða

Þar sem ætlunin er að lambda-reikningur lýsi föllum, þá er nauðsynlegt að skilgreina jafngildi segða þannig að tvær segðir séu jafngildar ef þær á einhvern hátt „þýða“ það sama. Til dæmis getum við skilgreint föllin f og g með $f(x) = x^2$ og $g(y) = y^2$, en þótt f og g séu skilgreind sem föll af mismunandi breytistærðum, þá eru föllin þau sömu. Á sama hátt viljum við til dæmis að '$(\lambda x.\, x)$' og '$(\lambda y.\, y)$' séu jafngildar segðir, þótt þær séu ekki sama segðin, enda tákna þær báðar samsemdarvörpunina. Tvær λ-segðir sem eru jafngildar á þennan máta eru sagðar vera *α-jafngildar* (sjá skilgreiningu 2.10).

Við viljum auk þess að sé fall kallað með ákveðnu viðfangi, að þá fáist útkoman úr fallinu með því að setja viðfangið inn fyrir formúlu fallsins. Ef til dæmis fall f er skilgreint með $f(x) = 2x^2 + 3$, þá ætti að vera jafngilt að rita $f(1)$ og $2 \cdot 1^2 + 3$, eða jafnvel 5. Að sama skapi ætti segðin '$((\lambda x.\, x)\, y)$' að vera jafngild 'y'. Meira að segja má segja að seinni segðin sé einfölduð útgáfa af þeirri fyrri. Þessu er miðlað með *β-einföldun* (sjá skilgreiningu 2.13).

Bæði α-jafngildi og β-einföldun krefjast þess að skilgreina nákvæmlega hvað felst í að skipta út breytum fyrir aðrar breytur eða fyrir λ-segðir, og hvenær þetta má gera án þess að breyta merkingu segðarinnar.

Skilgreining 2.6. Látum $\mathbf{M}, \mathbf{N}$ vera λ-segðir. Ef til eru stæður $\mathbf{A}, \mathbf{B}$ (hugsanlega tómar) þannig að rita megi $\mathbf{M} = \mathbf{ANB}$, þá nefnist $\mathbf{N}$ *hlutsegð* í $\mathbf{M}$.

Ef breyta $\mathbf{x}$ kemur fyrir í hlutsegð í $\mathbf{M}$ af gerðinni $(\lambda \mathbf{x}.\, \mathbf{P})$, þá segjum við að breytan $\mathbf{x}$ sé *bundin* í $\mathbf{M}$ á þeim stað. Ef $\mathbf{x}$ kemur einhvers staðar fyrir á stað sem er ekki hluti af slíkri hlutyrðingu í $\mathbf{M}$, þá segjum við að $\mathbf{x}$ sé *frjáls* í $\mathbf{M}$ á þeim stað.

Látum nú $\mathbf{Q}$ vera λ-segð og $\mathbf{x}$ vera breytu. Við segjum að $\mathbf{Q}$ sé *innsetjanleg* fyrir $\mathbf{x}$ í $\mathbf{M}$ ef fyrir hverja frjálsa breytu $\mathbf{y}$ í $\mathbf{Q}$ innihaldi $\mathbf{M}$ enga hlutsegð af gerðinni $(\lambda \mathbf{y}.\, \mathbf{P})$, þannig að $\mathbf{x}$ komi fyrir frjáls í $\mathbf{P}$.

Dæmi 2.7. Skoðum λ-segðina $\mathbf{M} :=$ '$(x\,(\lambda x.(y\,x)))$'. Breytan 'x' kemur hér þrisvar fyrir. Á fyrsta stað hennar er hún fráls, en á hinum tveimur er hún bundin. Breytan 'y' kemur einu sinni fyrir, og hún kemur fyrir frjáls.

Segðin 'x' er ekki innsetjanleg fyrir 'y' í $\mathbf{M}$, enda inniheldur $\mathbf{M}$ hlutsegðina '$(\lambda x.\,(y\,x))$', og 'y' er þar frjáls. Hins vegar er 'z' innsetjanleg fyrir 'y' í $\mathbf{M}$.

Lauslega skilið er segð innsetjanleg fyrir breytu í annarri segð ef engin breyta sem áður var frjáls verður bundin við það að setja segðina í staðinn fyrir breytuna. Ef við myndum til dæmis setja 'x' inn fyrir 'y' í $\mathbf{M}$ fæst segðin '$(x\,(\lambda x.\,(x\,x)))$', en hér verður breytan sem áður var 'y' bundin. Þetta breytir því merkingu segðarinnar. Ef við setjum hins vegar 'z' inn fyrir 'y' í $\mathbf{M}$ fæst segðin '$(x\,(\lambda x.\,(z\,x)))$', en hér er viðhaldið bundnum og frjálsum breytum. Nánari umræðu um þetta má sjá í [12, Kafli II, §1, (1.5)].

Skilgreining 2.8. Látum $\mathbf{M}, \mathbf{N}_1, \ldots, \mathbf{N}_m$ vera λ-segðir og $\mathbf{x}_1, \ldots, \mathbf{x}_m$ vera ólíkar breytur þannig að $\mathbf{N}_i$ sé innsetjanleg fyrir $\mathbf{x}_i$ í $\mathbf{M}$ fyrir öll $i = 1, \ldots, m$. Við táknum með $\mathbf{M}_{\mathbf{x}_1,\ldots,\mathbf{x}_m}[\mathbf{N}_1, \ldots, \mathbf{N}_m]$ þá λ-segð sem fæst frá $\mathbf{M}$ með því að setja $\mathbf{N}_i$ í stað $\mathbf{x}_i$ alls staðar sem $\mathbf{x}_i$ kemur fyrir frjáls í $\mathbf{M}$, samtímis fyrir $i = 1, \ldots, m$.

Ef sett er $\mathfrak{S} := \{(\mathbf{x}_1, \mathbf{N}_1), \ldots, (\mathbf{x}_m, \mathbf{N}_m)\}$ leyfum við okkur einnig að rita $\mathbf{M}[\mathfrak{S}]$ í stað $\mathbf{M}_{\mathbf{x}_1,\ldots,\mathbf{x}_m}[\mathbf{N}_1, \ldots, \mathbf{N}_m]$, og köllum þá $\mathfrak{S}$ *innsetningasafn*. Tvennd $(\mathbf{x}, \mathbf{N})$ úr $\mathfrak{S}$ nefnist þá jafnframt *innsetning* segðarinnar $\mathbf{N}$ *fyrir* $\mathbf{x}$.

Dæmi 2.9. Látum $\mathbf{M}$ vera eins og í dæmi 2.7. Þá er

$$\begin{aligned} \mathbf{M}_{`y'}[`z'] &= `(x\,(\lambda x.\,(z\,x)))', \\ \mathbf{M}_{`x'}[`z'] &= `(z\,(\lambda x.\,(y\,x)))', \\ \text{og} \quad \mathbf{M}_{`x',`y'}[`y', `z'] &= `(y\,(\lambda x.\,(z\,x)))'. \end{aligned}$$

Athugum að það skilyrði úr skilgreiningu 2.8 að segðir séu settar inn fyrir breytur samtímis er nauðsynlegt. Ef, í síðusta dæminu að ofan, væri fyrst sett 'y' inn fyrir 'x' og síðan 'z' inn fyrir 'y', þá fengist segðin

$$\mathbf{M}_{`x'}[`y']_{`y'}[`z'] = `(z\,(\lambda x.\,(z\,x)))' \neq \mathbf{M}_{`x',`y'}[`y', `z'].$$

Með þessu getum við snúið okkur að því að skilgreina α-jafngildi og β-einföldun.

Skilgreining 2.10. Látum $\mathbf{A}(\lambda\mathbf{x}.\,\mathbf{M})\mathbf{B}$ vera λ-segð og gerum ráð fyrir að breyta $\mathbf{y}$ sé innsetjanleg fyrir $\mathbf{x}$ í $\mathbf{M}$. Við ritum þá

$$\mathbf{A}(\lambda\mathbf{x}.\,\mathbf{M})\mathbf{B} \twoheadrightarrow_\alpha \mathbf{A}(\lambda\mathbf{y}.\,\mathbf{M}_{\mathbf{x}}[\mathbf{y}])\mathbf{B}$$

og skilgreinum þannig vensl $\twoheadrightarrow_\alpha$ yfir λ. Við segjum að $\mathbf{A}(\lambda\mathbf{y}.\,\mathbf{M}_\mathbf{x}[\mathbf{y}])\mathbf{B}$ fáist með *α-umskipti* frá $\mathbf{A}(\lambda\mathbf{x}.\,\mathbf{M})\mathbf{B}$.

Við skilgreinum auk þess vensl $\sim_\alpha$ yfir λ þannig að $\mathbf{P} \sim_\alpha \mathbf{Q}$ ef og aðeins ef til er runa $\mathbf{A}_0, \ldots, \mathbf{A}_n$ af λ-segðum, þannig að

$$\mathbf{P} = \mathbf{A}_0 \twoheadrightarrow_\alpha \cdots \twoheadrightarrow_\alpha \mathbf{A}_n = \mathbf{Q}.$$

Við segjum þá að $\mathbf{P}$ og $\mathbf{Q}$ séu *α-jafngildar*. Við köllum rununa $\mathbf{A}_0, \ldots, \mathbf{A}_n$ *útleiðslu* venslanna.

Dæmi 2.11. Skoðum segðina '$(\lambda xy.\,(y\,x\,y))$'. Þegar leyst er úr skammstöfunum fæst segðin '$(\lambda x.\,(\lambda y.\,((y\,x)\,y)))$'. Við höfum

$$\text{'}(\lambda x.\,(\lambda y.\,((y\,x)\,y)))\text{'} \twoheadrightarrow_\alpha \text{'}(\lambda a.\,(\lambda y.\,((y\,a)\,y)))\text{'} \twoheadrightarrow_\alpha \text{'}(\lambda a.\,(\lambda b.\,((b\,a)\,b)))\text{'},$$

svo '$(\lambda xy.\,(y\,x\,y))$' $\sim_\alpha$ '$(\lambda ab.\,(b\,a\,b))$'.

Eftirfarandi setning réttlætir nafngiftina „α-jafngildi“:

Setning 2.12. *Venslin $\sim_\alpha$ eru jafngildisvensl.*

Sönnun: Látum $\mathbf{P}, \mathbf{Q}, \mathbf{R}$ vera λ-segðir.

(i) *Sjálfhverfni.* Runan sem inniheldur einungis $\mathbf{P}$ er útleiðsla á $\mathbf{P} \sim_\alpha \mathbf{P}$.

(ii) *Samhverfni.* Tökum fyrst eftir að ef $\mathbf{y}$ er innsetjanleg fyrir $\mathbf{x}$ í $\mathbf{M}$, þá er $\mathbf{x}$ innsetjanleg fyrir $\mathbf{y}$ í $\mathbf{M}_\mathbf{x}[\mathbf{y}]$ og $\mathbf{M}_\mathbf{x}[\mathbf{y}]_\mathbf{y}[\mathbf{x}] = \mathbf{M}$, svo venslin $\twoheadrightarrow_\alpha$ eru samhverf (ítarlegri sönnun á þessu má finna í [9, Lemma A1.5]).

Látum $\mathbf{A}_0, \ldots, \mathbf{A}_n$ vera útleiðslu venslanna $\mathbf{P} \sim_\alpha \mathbf{Q}$. Þá er $\mathbf{A}_n, \ldots, \mathbf{A}_0$ útleiðsla venslanna $\mathbf{Q} \sim_\alpha \mathbf{P}$.

(iii) *Gegnvirkni.* Látum $\mathbf{A}_0, \ldots, \mathbf{A}_n$ vera útleiðslu $\mathbf{P} \sim_\alpha \mathbf{Q}$ og $\mathbf{B}_0, \ldots, \mathbf{B}_m$ vera útleiðslu $\mathbf{Q} \sim_\alpha \mathbf{R}$. Þá er $\mathbf{A}_0, \ldots, \mathbf{A}_n, \mathbf{B}_1, \ldots, \mathbf{B}_m$ útleiðsla venslanna $\mathbf{P} \sim_\alpha \mathbf{R}$. □

Við skilgreinum nú β-einföldun á svipaðan hátt og α-umskipti.

Skilgreining 2.13. Látum $\mathbf{A}((\lambda\mathbf{x}.\,\mathbf{M})\,\mathbf{N})\mathbf{B}$ vera λ-segð og gerum ráð fyrir að $\mathbf{N}$ sé innsetjanleg fyrir $\mathbf{x}$ í $\mathbf{M}$. Við ritum þá

$$\mathbf{A}((\lambda\mathbf{x}.\,\mathbf{M})\,\mathbf{N})\mathbf{B} \twoheadrightarrow_\beta \mathbf{A}\,\mathbf{M}_\mathbf{x}[\mathbf{N}]\,\mathbf{B}$$

og skilgreinum þannig vensl $\twoheadrightarrow_\beta$ yfir λ. Við segjum að $\mathbf{A}\,\mathbf{M}_\mathbf{x}[\mathbf{N}]\,\mathbf{B}$ fáist með *β-einföldun* frá $\mathbf{A}((\lambda\mathbf{x}.\,\mathbf{M})\,\mathbf{N})\mathbf{B}$.

Dæmi 2.14. Við höfum eftirfarandi:

(a) ‘$((\lambda x.\, x)\, y)$’ $\twoheadrightarrow_\beta$ ‘y’.

(b) ‘$((\lambda x.\, z)\, y)$’ $\twoheadrightarrow_\beta$ ‘z’.

(c) ‘$((\lambda x.\, ((\lambda y.\, (y\, x))\, z))\, a)$’ $\twoheadrightarrow_\beta$ ‘$((\lambda x.\, (z\, x))\, a)$’ $\twoheadrightarrow_\beta$ ‘$(z\, a)$’.

(d) ‘$((\lambda x.\, ((\lambda y.\, (y\, x))\, z))\, a)$’ $\twoheadrightarrow_\beta$ ‘$((\lambda y.\, (y\, a))\, z)$’ $\twoheadrightarrow_\beta$ ‘$(z\, a)$’.

(e) Setjum $\mathbf{\Omega} \coloneqq$ ‘$((\lambda x.\, (x\, x))\, (\lambda x.\, (x\, x)))$’. Þá höfum við

$$\mathbf{\Omega} \twoheadrightarrow_\beta \mathbf{\Omega} \twoheadrightarrow_\beta \mathbf{\Omega} \twoheadrightarrow_\beta \cdots .$$

(f) Setjum $\mathbf{\Theta} \coloneqq$ ‘$((\lambda x.\, (x\, x\, y))\, (\lambda x.\, (x\, x\, y)))$’. Þá höfum við

$$\mathbf{\Theta} \twoheadrightarrow_\beta (\mathbf{\Theta}\, y) \twoheadrightarrow_\beta (\mathbf{\Theta}\, y\, y) \twoheadrightarrow_\beta \cdots .$$

Með (c) og (d) sést að segðina ‘$((\lambda x.\, ((\lambda y.\, (y\, x))\, z))\, a)$’ má einfalda á tvo mismunandi vegu. Báðar einfaldanir leiða hins vegar til sömu óeinfaldanlegrar segðar. Þetta kemur aftur við sögu í setningu 2.21, en á meðan má lesandi velta sér fyrir hvort þetta gildi almennt.

Við sjáum einnig í (e) og (f) að segð sem fæst með β-einföldun þarf ekki endilega að vera einfaldari en upphaflega segðin.

Venslin $\twoheadrightarrow_\beta$ eru takmörkuð að því leyti að ef $\mathbf{P} \twoheadrightarrow_\beta \mathbf{Q}$, þá má mynda $\mathbf{Q}$ með nákvæmlega einni einföldun frá $\mathbf{P}$. Í almennari skilningi er þægilegt að tala um að segð megi mynda með núll eða fleiri einföldunum, og um leið taka tillit til þess að oft þurfi að framkvæma breytuskipti áður en hægt sé að einfalda. Við getum skilgreint önnur vensl í þessum tilgangi.

Auk þess viljum við skilgreina vensl sem ná utan um hvenær tvær λ-segðir „þýða“ það sama (sbr. umræðu í byrjun þessa undirkafla).

Skilgreining 2.15. Setjum

$$\begin{aligned} \twoheadrightarrow_{\alpha,\beta} &\coloneqq \twoheadrightarrow_\alpha \cup \twoheadrightarrow_\beta \\ \text{og} \quad \twoheadleftrightarrow_{\alpha,\beta} &\coloneqq \twoheadrightarrow_\alpha \cup \twoheadrightarrow_\beta \cup \twoheadleftarrow_\beta, \end{aligned}$$

þar sem $\twoheadleftarrow_\beta$ eru viðsnúin venslin $\twoheadrightarrow_\beta$ (munum að vensl yfir λ eru hlutmengi í $\lambda \times \lambda$). Við skilgreinum vensl $\twoheadrightarrow$ og $\sim_\beta$ yfir λ þannig að

$$\begin{aligned} \mathbf{P} \twoheadrightarrow \mathbf{Q} &\text{ ef til er runa } \mathbf{A}_0, \dots, \mathbf{A}_n \text{ þ.a. } \mathbf{P} = \mathbf{A}_0 \twoheadrightarrow_{\alpha,\beta} \cdots \twoheadrightarrow_{\alpha,\beta} \mathbf{A}_n = \mathbf{Q} \\ \text{og } \mathbf{P} \sim_\beta \mathbf{Q} &\text{ ef til er runa } \mathbf{A}_0, \dots, \mathbf{A}_n \text{ þ.a. } \mathbf{P} = \mathbf{A}_0 \twoheadleftrightarrow_{\alpha,\beta} \cdots \twoheadleftrightarrow_{\alpha,\beta} \mathbf{A}_n = \mathbf{Q}. \end{aligned}$$

Við köllum þá rununa $\mathbf{A}_0, \dots, \mathbf{A}_n$ *útleiðslu* tilsvarandi vensla.

Ef $\mathbf{P} \twoheadrightarrow \mathbf{Q}$ segjum við að $\mathbf{P}$ *einfaldist* í $\mathbf{Q}$, og ef $\mathbf{P} \sim_\beta \mathbf{Q}$ segjum við að $\mathbf{P}$ og $\mathbf{Q}$ séu *β-jafngild*.

Athugasemd 2.16. Lauslega skilið þá gildir $\mathbf{P} \sim_\beta \mathbf{Q}$ ef beita má núll eða fleiri breytuskiptum, β-einföldunum, eða viðsnúnum β-einföldunum frá $\mathbf{P}$ til þess að mynda $\mathbf{Q}$. Þetta er nægilegt til þess að $\mathbf{P}$ og $\mathbf{Q}$ „þýða" hið sama, með þeim tiltölulega óformlega skilningi á því sem við höfum talað um hingað til.

Athuga ber að ekki væri nægilegt að skilgreina $\sim_\beta$ með $\twoheadrightarrow \cup \twoheadleftarrow$, enda er '$((\lambda x.\,(z\,x))\,a)$' $\sim_\beta$ '$((\lambda y.\,(y\,a))\,z)$' samkvæmt dæmi 2.14(c) og (d), en hvorki gildir '$((\lambda x.\,(z\,x))\,a)$' $\twoheadrightarrow$ '$((\lambda y.\,(y\,a))\,z)$' né '$((\lambda y.\,(y\,a))\,z)$' $\twoheadrightarrow$ '$((\lambda x.\,(z\,x))\,a)$'.

Eftirfarandi setning réttlætir eins og áður nafngiftina „β-jafngildi":

Setning 2.17. *Venslin $\sim_\beta$ eru jafngildisvensl.*

Sönnun: Látum $\mathbf{P}, \mathbf{Q}, \mathbf{R}$ vera λ-segðir.

(i) *Sjálfhverfni.* Runan sem inniheldur einungis $\mathbf{P}$ er útleiðsla á $\mathbf{P} \sim_\beta \mathbf{P}$.

(ii) *Samhverfni.* Látum $\mathbf{A}_0, \ldots, \mathbf{A}_n$ vera útleiðslu venslanna $\mathbf{P} \sim_\beta \mathbf{Q}$. Þá er $\mathbf{A}_n, \ldots, \mathbf{A}_0$ útleiðsla venslanna $\mathbf{Q} \sim_\beta \mathbf{P}$.

(iii) *Gegnvirkni.* Látum $\mathbf{A}_0, \ldots, \mathbf{A}_n$ vera útleiðslu $\mathbf{P} \sim_\beta \mathbf{Q}$ og $\mathbf{B}_0, \ldots, \mathbf{B}_m$ vera útleiðslu $\mathbf{Q} \sim_\beta \mathbf{R}$. Þá er $\mathbf{A}_0, \ldots, \mathbf{A}_n, \mathbf{B}_1, \ldots, \mathbf{B}_m$ útleiðsla venslanna $\mathbf{P} \sim_\beta \mathbf{R}$. □

2.3 Church-Rosser-setningin

Við höfum hingað til skilgreint lambda-reikning og umritun λ-segða í jafngildar segðir, en ekki er ljóst í fyrstu hvernig þetta skilgreinir líkan fyrir reikning. Hugmyndin er að hliðstæða sé á milli skrefa í einföldun λ-segðar og skrefa í reikningi, en áður en við megum líkja þessu saman þarf að ganga úr skugga um að einföldunin uppfylli nauðsynlegar kröfur. Sem dæmi má nefna að ef halda megi áfram á tvo mismunandi vegu á tilteknu stigi í framkvæmd reiknings þá ætti að búast við að báðir möguleikar gefi sömu lokaniðurstöðu. Reikning talnareiknisegðarinnar $1 + 2 + 3$ má til dæmis framkvæma á tvo vegu:

$$1 + 2 + 3 = 3 + 3 = 6 \quad \text{og} \quad 1 + 2 + 3 = 1 + 5 = 6,$$

en báðir reikningar gefa sama lokasvar.

Við höfum auk þess ekki fært mikil rök fyrir því að β-jafngildi sé nægilegt til þess að ná yfir að segðir „þýði" hið sama.

Hér á eftir verður sýnt fram á báða þessa eiginleika, en þeir eru afleiðing af geysimikilvægri setningu sem kennd er við Alonzo Church og J. Barkley Rosser. Áður en við setjum fram þessa setningu skoðum við hins vegar ákveðinn flokk λ-segða sem liggur til grundvallar í fræðinni um lambda-reikning. Hugsa má að þessar segðir séu eins einfaldar og þær geta orðið, og við notum þær hliðstætt við „lokaniðurstöðu" í umræðum um almennan reikning.

Skilgreining 2.18. Við segjum að λ-segð $\mathbf{P}$ sé á *β-staðalformi* ef ekki eru til λ-segðir $\mathbf{P}', \mathbf{Q}$ þannig að $\mathbf{P} \sim_\alpha \mathbf{P}' \twoheadrightarrow_\beta \mathbf{Q}$.

Ef $\mathbf{P}$ er á β-staðalformi, og $\mathbf{M} \sim_\beta \mathbf{P}$, þá er $\mathbf{M}$ sögð hafa $\mathbf{P}$ sem β-staðalform.

Athugasemd 2.19. Ástæða þess að skilgreining 2.18 notar α-jafngilda segð $\mathbf{P}'$ er af því að $\mathbf{P}$ gæti innihaldið hlutsegð af gerðinni $((\lambda\mathbf{x}.\,\mathbf{M})\,\mathbf{N})$, án þess að $\mathbf{N}$ sé innsetjanleg fyrir $\mathbf{x}$ í $\mathbf{M}$. Hins vegar má þetta ávallt laga með breytuskiptum. Tökum til dæmis segðina '$\mathbf{P} := ((\lambda xy.\,(x\,y))\,y\,x)$'. Ekki er til segð $\mathbf{Q}$ þannig að $\mathbf{P} \twoheadrightarrow_\beta \mathbf{Q}$, en $\mathbf{P}$ er samt ekki á β-staðalformi, enda fæst

$$\mathbf{P} \sim_\alpha \text{'}((\lambda ab.\,(a\,b))\,y\,x)\text{'} \twoheadrightarrow_\beta \text{'}((\lambda b.\,(y\,b))\,x)\text{'} \twoheadrightarrow_\beta \text{'}(y\,x)\text{'}.$$

Við sjáum hins vegar að $\mathbf{P}$ hefur segðina '$(y\,x)$' sem β-staðalform.

Með þetta í huga má sjá að segð er á β-staðalformi ef og aðeins ef hún inniheldur enga hlutsegð á forminu $((\lambda\mathbf{x}.\,\mathbf{M})\,\mathbf{N})$.

Athugasemd 2.20. Mögulegt er fyrir λ-segð að hafa ekkert β-staðalform. Tökum til dæmis segðina $\mathbf{\Omega}$ úr dæmi 2.14(e). Ljóst er að eina leiðin til þess að β-einfalda $\mathbf{\Omega}$ er með $\mathbf{\Omega} \twoheadrightarrow_\beta \mathbf{\Omega}$, og breytuskipti hjálpa hér ekki. Þar með hefur $\mathbf{\Omega}$ ekkert β-staðalform. Á sama hátt sést að $\mathbf{\Theta}$ úr 2.14(f) hefur ekkert β-staðalform.

Að segð hafi einhverja óendanlega einföldun er hins vegar ekki nóg til þess að segja að hún hafi ekkert β-staðalform. Skoðum segðina $((\lambda x.\,y)\,\mathbf{\Omega})$. Annars vegar fæst einföldunin $((\lambda x.\,y)\,\mathbf{\Omega}) \twoheadrightarrow_\beta$ 'y', og hins vegar fæst óendanlega einföldunin $((\lambda x.\,y)\,\mathbf{\Omega}) \twoheadrightarrow_\beta ((\lambda x.\,y)\,\mathbf{\Omega}) \twoheadrightarrow_\beta ((\lambda x.\,y)\,\mathbf{\Omega}) \twoheadrightarrow_\beta \cdots$. Segðin hefur því β-staðalformið 'y', þótt hún eigi sér óendanlega einföldun. Við ræðum þetta nánar í kafla 3.2.

Við erum nú tilbúin til þess að skoða Church-Rosser-setninguna. Sönnun hennar er of löng til þess að setja fram hér. Áhugasömum lesanda er bent á [9, Appendix A2], [2, Chapter 3, §2], og [13, 4.4]. Þessar heimildir nota sönnun sem kennd er við William Tait og Per Martin-Löf. Aðra sönnun sem notar svokallaða *Takahashi-þýðingu* má finna í [11].

Setning 2.21 (*Church-Rosser*)**.** *Ef* $\mathbf{P} \twoheadrightarrow \mathbf{M}$ *og* $\mathbf{P} \twoheadrightarrow \mathbf{N}$*, þá er til λ-segð* $\mathbf{Q}$ *þannig að* $\mathbf{M} \twoheadrightarrow \mathbf{Q}$ *og* $\mathbf{N} \twoheadrightarrow \mathbf{Q}$.

Óformlega segir Church-Rosser-setningin að ef einföldun segðar getur farið í tvær mismunandi áttir, þá er alltaf hægt að samræma tilsvarandi einfaldanir þannig að þæri gefi sömu segð.

Munum eftir einföldununum úr dæmi 2.14(c) og (d). Þar sáum við að segðin '$((\lambda x.\,((\lambda y.\,(y\,x))\,z))\,a)$' hefur tvær einfaldanir en aðeins eitt β-staðalform. Í þessu einfalda tilviki er það bein afleiðing af Church-Rosser-setningu, en velta má fyrir sér hvort þetta sé líka rétt fyrir flóknari segðir.

Sjá má af skilgreiningu β-staðalforms að ef segð hefur β-staðalform $\mathbf{P}$ og $\mathbf{P} \sim_\alpha \mathbf{P}'$, þá hefur segðin einnig β-staðalform $\mathbf{P}'$. Ef $\mathbf{P} \neq \mathbf{P}'$ sést því að segðin hefur tvö mismunandi β-staðalform. Til dæmis hefur segðin '$(x\,(\lambda y.\, y))$' bæði sig sjálfa og segðina '$(x\,(\lambda z.\, z))$' sem β-staðalform. Eftirfarandi einföld afleiðing Church-Rosser-setningar gefur hins vegar að þetta er eina leiðin sem β-staðalform segðar geta verið ólík:

Fylgisetning 2.22. *Ef $\mathbf{P} \twoheadrightarrow \mathbf{M}$ og $\mathbf{P} \twoheadrightarrow \mathbf{N}$ og $\mathbf{M}$ og $\mathbf{N}$ eru á β-staðalformi, þá gildir $\mathbf{N} \sim_\alpha \mathbf{M}$.*

Sönnun: Samkvæmt Church-Rosser-setningu er til λ-segð $\mathbf{Q}$ þannig að $\mathbf{M} \twoheadrightarrow \mathbf{Q}$ og $\mathbf{N} \twoheadrightarrow \mathbf{Q}$. Nú er $\mathbf{M}$ á β-staðalformi, svo ekki er til $\mathbf{A}$ þ.a. $\mathbf{M} \twoheadrightarrow_\beta \mathbf{A}$. Því höfum við $\mathbf{M} \sim_\alpha \mathbf{Q}$. Eins fæst $\mathbf{N} \sim_\alpha \mathbf{Q}$. Þar með gildir $\mathbf{M} \sim_\alpha \mathbf{N}$. □

Þannig er tryggt að réttmætt sé að tala um að segð *hafi* β-staðalform, því að í þessu felst að ef slíkt form er til, þá er það ótvírætt upp að bundnum breytuskiptum.

Önnur einföld afleiðing Church-Rosser-setningar staðfestir að β-jafngildi þýði það sem óformleg túlkun þess segir:

Fylgisetning 2.23. *Ef $\mathbf{M} \sim_\beta \mathbf{N}$, þá er til λ-segð $\mathbf{Q}$ þannig að $\mathbf{M} \twoheadrightarrow \mathbf{Q}$ og $\mathbf{N} \twoheadrightarrow \mathbf{Q}$.*

Sönnun: Gerum ráð fyrir að $\mathbf{M} \sim_\beta \mathbf{N}$ og látum $\mathbf{A}_0, \ldots, \mathbf{A}_n$ vera útleiðslu þess. Við sönnum niðurstöðuna með þrepun yfir n. Ef $n = 0$, þá er $\mathbf{M} = \mathbf{N}$ og niðurstaðan er ljós. Gerum því ráð fyrir að $n > 0$ og að niðurstaðan sé sönn fyrir $n - 1$. Þá er til λ-segð $\mathbf{Q}'$ þannig að $\mathbf{M} \twoheadrightarrow \mathbf{Q}'$ og $\mathbf{A}_{n-1} \twoheadrightarrow \mathbf{Q}'$.

Nú gildir eitt af $\mathbf{A}_{n-1} \sim_\alpha \mathbf{N}$, $\mathbf{A}_{n-1} \twoheadrightarrow_\beta \mathbf{N}$ eða $\mathbf{N} \twoheadrightarrow_\beta \mathbf{A}_{n-1}$. Ef $\mathbf{A}_{n-1} \sim_\alpha \mathbf{N}$ eða $\mathbf{N} \twoheadrightarrow \mathbf{A}_{n-1}$, þá höfum við $\mathbf{N} \twoheadrightarrow \mathbf{A}_{n-1}$, og því $\mathbf{N} \twoheadrightarrow \mathbf{Q}'$ Niðurstaðan er þá sönn með $\mathbf{Q} = \mathbf{Q}'$.

Ef hins vegar $\mathbf{A}_{n-1} \twoheadrightarrow \mathbf{N}$, þá gefur Church-Rosser-setning að til er λ-segð $\mathbf{Q}$ þannig að $\mathbf{Q}' \twoheadrightarrow \mathbf{Q}$ og $\mathbf{N} \twoheadrightarrow \mathbf{Q}$. Fyrri venslin gefa að $\mathbf{M} \twoheadrightarrow \mathbf{Q}$, sem ásamt þeim seinni sannar niðurstöðuna. □

Hugsa má að þessi setning segi að tvær jafngildar segðir tákni sama hlutinn, enda má einfalda þær niður í sömu segðina.

2.4 Fastapunktsfléttir

Við tökum nú dálitla hjáleið frá viðfangsefni kaflans og skoðum svokallaðan *fastapunktsfléttti* (e. *fixed-point combinator*). Meginniðurstöðu undirkaflans má finna í setningu 2.29, en hana er vert að skoða bæði í samhengi lambda-reiknings og almennrar fallaforritunar.

Munum að fastapunktur falls f er gildi c þannig að $f(c) = c$. Til dæmis hefur fallið $f(x) = 2x - 1$ fastapunktinn 1. Einnig er eðlilegt að spyrja hvort λ-segð hafi fastapunkt, enda er ætlunin að λ-segðir séu túlkaðar sem föll. Ef $\mathbf{M}$ er λ-segð er þá spurningin hvort til sé λ-segð $\mathbf{N}$ þannig að $(\mathbf{M}\,\mathbf{N}) \sim_\beta \mathbf{N}$.

Við getum ímyndað okkur að smíða megi λ-segð sem finnur slíka fastapunkta.

Skilgreining 2.24. Látum $\mathbf{Y}$ vera λ-segð með enga frjálsa breytu. Við köllum $\mathbf{Y}$ *fastapunktsflétti* ef $(\mathbf{Y}\,f) \sim_\beta (f\,(\mathbf{Y}\,f))$. Fyrir λ-segð $\mathbf{M}$ köllum við þá segðina $(\mathbf{Y}\,\mathbf{M})$ *fastapunkt* $\mathbf{M}$.

Athugasemd 2.25. Við notum orðið *fléttir* sem þýðingu enska orðsins *combinator*, en í samhengi lambda-reiknings er þetta einfaldlega λ-segð sem hefur engar frjálsar breytur. Það er því óhætt að blanda saman fléttum án breytuskörunar.

Til er líkan sem er jafngilt lambda-reikningi og er oft talað um samhliða honum, nefnilega *fléttureikningur* (e. *combinatory calculus*), sem notar flétti sem sjálfstætt fyrirbæri og gerir þannig ekki greinarmun á bundnum og frjálsum breytum. Þetta einfaldar margar sannanir sem annars væru flóknar.

Við munum ekki tala nánar um fléttureikning hér, en ef lesandi hefur áhuga má finna inngang að honum í [9, Chapter 2]. Margar niðurstöður í þeirri bók gilda bæði fyrir lambda-reikning og fléttureikning.

Athugasemd 2.26. Tökum eftir að ef fastapunktsfléttir $\mathbf{Y}$ er til, þá finnur hann fastapunkt fyrir sérhverja λ-segð. Þetta er bein afleiðing af þeirri niðurstöðu að ef $\mathbf{M} \sim_\beta \mathbf{M}'$, þá er $((\lambda\mathbf{x}.\ \mathbf{M})\,\mathbf{N}) \sim_\beta ((\lambda\mathbf{x}.\ \mathbf{M}')\,\mathbf{N})$ og því $\mathbf{M}_\mathbf{x}[\mathbf{N}] \sim_\beta \mathbf{M}'_\mathbf{x}[\mathbf{N}]$.

Þetta gæti komið á óvart — fráleitt væri að halda að sérhvert fall hefði fastapunkt, svo af hverju ætti sérhver λ-segð að hafa fastapunkt? Hins vegar er ekki víst að fastapunktur sem fastapunktsfléttir finnur hafi β-staðalform, og í því tilfelli má það á einhvern hátt túlka sem að fastapunkturinn sé óreiknanlegur.

Enn er óljóst hvort eiginleika fastapunktsflétta sé vert að skoða, enda höfum við ekki sýnt fram á tilvist fastapunktsfléttis. Þetta gerum við nú.

Setning 2.27. *Setjum* $\mathbf{U} :=$ ‘$(\lambda ux.\,(x\,(u\,u\,x)))$’. *Segðin* $\mathbf{Y} := (\mathbf{U}\,\mathbf{U})$ *er fastapunktsfléttir.*

Sönnun: Við fáum

$$(\mathbf{Y}\,f) = (\mathbf{U}\,\mathbf{U}\,f) \twoheadrightarrow_\beta ((\lambda x.(x\,(\mathbf{U}\,\mathbf{U}\,x)))\,f) \twoheadrightarrow_\beta (f\,(\mathbf{U}\,\mathbf{U}\,f)) = (f\,(\mathbf{Y}\,f)). \quad \square$$

Athugasemd 2.28. Fastapunktsfléttirinn að ofan er kenndur við Alan Turing, en hann er ekki eini fastapunktsfléttirinn sem til er. Annar slíkur fléttir er kenndur við Haskell Curry og Paul Rosenbloom:

$$\mathbf{Y}' := (\lambda x.\, (\mathbf{V}\,\mathbf{V})), \quad \text{þar sem} \quad \mathbf{V} := \text{‘}(\lambda y.\, (x\,(y\,y)))\text{’}.$$

Lesandi má staðfesta að þetta sé í raun fastapunktsfléttir.

Næsta setning útskýrir áhuga okkar á fastapunktsfléttum bæði frá sjónarhóli lambda-reiknings og fallaforritunar.

Setning 2.29. *Látum* $\mathbf{Z}$ *vera* λ*-segð og* $\mathbf{x}, \mathbf{y}_1, \ldots, \mathbf{y}_n$ *vera breytur. Til er* λ*-segð* $\mathbf{X}$ *þannig að*

$$(\mathbf{X}\,\mathbf{y}_1\,\cdots\,\mathbf{y}_n) \sim_\beta \mathbf{Z}_{\mathbf{x}}[\mathbf{X}].$$

Sönnun: Látum $\mathbf{Y}$ vera fastapunktsfléttí (sem er til skv. setningu 2.27) og setjum $\mathbf{X} := (\mathbf{Y}\,(\lambda \mathbf{x}\,\mathbf{y}_1\,\ldots\,\mathbf{y}_n.\,\mathbf{Z}))$. Þá inniheldur $\mathbf{X}$ engar frjálsar breytur og er því greinilega innsetjanleg fyrir $\mathbf{x}$ í $\mathbf{Z}$. Við fáum þá

$$\mathbf{X} \sim_\beta ((\lambda \mathbf{x}\,\mathbf{y}_1\,\ldots\,\mathbf{y}_n.\,\mathbf{Z})\,\mathbf{X}) \twoheadrightarrow_\beta (\lambda \mathbf{y}_1\,\ldots\,\mathbf{y}_n.\,\mathbf{Z}_{\mathbf{x}}[\mathbf{X}])$$

og því

$$(\mathbf{X}\,\mathbf{y}_1\,\cdots\,\mathbf{y}_n) \sim_\beta \mathbf{Z}_{\mathbf{x}}[\mathbf{X}]_{\mathbf{y}_1,\ldots,\mathbf{y}_n}[\mathbf{y}_1, \ldots, \mathbf{y}_n] = \mathbf{Z}_{\mathbf{x}}[\mathbf{X}]. \qquad \square$$

Mikilvægi þessarar setningar felst í því að hún leyfir okkur að leysa jöfnur af λ-segðum. Skoðum til dæmis venslin

$$(\mathbf{X}\,a\,b) \sim_\beta (a\,(\mathbf{X}\,(a\,b)\,a)) = \text{‘}(a\,(x\,(a\,b)\,a))\text{’}_{\text{‘}x\text{’}}[\mathbf{X}].$$

Þetta má líta á sem jöfnu með óþekktri breytistærð $\mathbf{X}$. Setning 2.29 tryggir að þessi jafna hafi lausn, nefnilega segðina $(\mathbf{Y}\,(\lambda xab.\,(a\,(x\,(a\,b)\,a))))$.

Þessi niðurstaða er einstaklega mikilvæg í fallaforritun, en hún leyfir að skilgreina *endurkvæm föll*. Tökum fallið $F : \mathbb{N} \to \mathbb{N}$, $F(n) = n!$. Það má einnig skilgreina endurkvæmt:

$$F(n) = \begin{cases} 1 & \text{ef } n = 0 \\ n \cdot F(n-1) & \text{annars} \end{cases}.$$

Í fallamálinu Haskell má þetta skilgreina á svipaðan hátt:

```
ghci> fact n = if n == 0 then 1 else n * fact (n-1)
ghci> fact 5
120
```

Þetta krefst hins vegar að málið geti leyst úr nafninu `fact` í miðri skilgreiningu þess. Í lambda-reikningi er engin leið til þess að gefa λ-segðum nöfn (nema á yfirmálinu), svo ekki má skilgreina endurkvæmar λ-segðir. Hins vegar má líta á skilgreininguna á `fact` að ofan sem jöfnu með óþekkta stærð `fact`, og setning 2.29 gefur þá í skyn að `fact` megi einnig setja fram með fastapunktsflétti. Haskell hefur útfærslu á fastapunktsflétti með fallinu `fix`, og við fáum:

```
ghci> fact' = fix (\f n -> if n == 0 then 1 else n * f (n-1))
ghci> fact' 5
120
```

Meira um notkun fallsins `fix` í Haskell má finna í [16].

Við getum einnig smíðað λ-segð sem er framsetning fallsins F. Hana má sjá í dæmi 2.37.

2.5 Hlutrakin föll

Við snúum okkur nú aftur að því að skoða hvernig lambda-reikningur skilgreinir líkan fyrir reikning. Á 4. áratug síðustu aldar komu fram þrjár mismunandi hugmyndir fyrir reiknanleika með stuttu millibili. Árið 1936 setti Alonzo Church fram lambda-reikning, sem við höfum þegar kynnst. Sama ár setti Alan Turing fram hugmyndina um *Turing-vél* ásamt *Turing-reiknanlegum* föllum. Nokkru fyrr, árið 1933, kynnti Kurt Gödel svokölluð *hlutrakin* föll í tengslum við rökfræði. Saman sýndu Church og Turing fram á, ásamt Kleene og Rosser, að öll þrjú fyrirbærin væru jafngild. Við munum hér á eftir skoða sérstaklega jafngildi lambda-reiknings og hlutrakinna falla.

Kynnum okkur fyrst hvernig hlutrakin föll eru skilgreind. Skilgreiningin hér fyrir neðan byggir á [12, Kafli IV, §1, (1.2)] og [9, Chapter 4].

Skilgreining 2.30. Við skilgreinum *frumstætt rakin föll* af gerðinni $\mathbb{N}^k \to \mathbb{N}$ endurkvæmt með eftirfarandi reglum:

I. Eftirfarandi föll eru frumstætt rakin:

- *Núllfallið* $Z : \mathbb{N} \to \mathbb{N},\ Z(n) = 0$.
- *Eftirfarafallið* $S : \mathbb{N} \to \mathbb{N},\ S(n) = n + 1$.
- *Ofanvarpsföllin* $\pi_i^k : \mathbb{N}^k \to \mathbb{N},\ \pi_i^k(n_1, \ldots, n_k) = n_i$.

II. Ef $G : \mathbb{N}^m \to \mathbb{N}$ og $H_1, \ldots, H_m : \mathbb{N}^k \to \mathbb{N}$ eru frumstætt rakin föll, þá er fallið $F : \mathbb{N}^k \to \mathbb{N}$ skilgreint með

$$F(n_1, \ldots, n_k) = G(H_1(n_1, \ldots, n_k), \ldots, H_m(n_1, \ldots, n_k))$$

frumstætt rakið fall.

III. Ef $G : \mathbb{N}^k \to \mathbb{N}$ og $H : \mathbb{N}^{k+2} \to \mathbb{N}$ eru frumstætt rakin föll, þá er fallið $F : \mathbb{N}^{k+1} \to \mathbb{N}$ skilgreint með

$$\begin{aligned} F(n_1, \ldots, n_k, 0) &= G(n_1, \ldots, n_k), \\ F(n_1, \ldots, n_k, m+1) &= H(n_1, \ldots, n_k, m, F(n_1, \ldots, n_k, m)) \end{aligned}$$

frumstætt rakið fall.

Við skilgreinum auk þess *hlutrakin föll* sem föll af gerðinni $A \to \mathbb{N}$, þar sem $A \subseteq \mathbb{N}^k$, með reglu IV að neðan:

IV. Ef $G : \mathbb{N}^{k+1} \to \mathbb{N}$ og $H : \mathbb{N} \to \mathbb{N}$ eru frumstætt rakin föll og A er mengi þeirra $x \in \mathbb{N}^k$ sem uppfylla að til sé $m \in \mathbb{N}$ þannig að $G(x, m) = 0$, þá er fallið $F : A \to \mathbb{N}$ skilgreint með

$$F(n_1, \ldots, n_k) = H(\min\{m \mid G(n_1, \ldots, n_k, m) = 0\})$$

hlutrakið fall.

Hlutrakið fall sem skilgreint er á öllu $\mathbb{N}^k$ nefnist *rakið fall.*

Hugsunin á bak við hlutrakin föll er sú að ávallt megi reikna út gildi hlutrakins falls á öllu skilgreiningarmengi sínu (þar sem reikningur er túlkaður á óformlegan, hversdagslegan hátt) með því að rekja sig niður skilgreiningu þess.

Ef föllin $G, H_1, \ldots, H_m$ eru reiknanleg, þá ætti samskeytingin F úr reglu II einnig að vera reiknanleg, því að einfaldlega má reikna gildi $H_1, \ldots, H_m$ og svo tilsvarandi gildi G.

Einnig ætti þrepunin sem fallið F úr reglu III skilgreinir að vera reiknanlegt, því að reikna má hvert gildi fallsins niður þar til síðasta viðfangið er 0, og reikna svo út niðurstöðuna með G og H.

Fallið F sem fæst úr reglu IV er auk þess reiknanlegt, enda má finna viðeigandi m með því að leita upp frá 0 með því að reikna hvort G skili 0 hverju sinni, og svo reikna tilsvarandi gildi H.

Svokölluð *Church-Turing-tilgátan* gengur lengra og segir að þau föll sem við myndum kalla reiknanleg með okkar venjulega skilningi séu nákvæmlega þau sem hlutrakin föll (og þá einnig Turing-vélar og lambda-reikningur) ná yfir. Þetta er í rauninni ósannanleg staðhæfing enda höfum við enga sjálfstæða, almenna skilgreiningu á reiknanleika. Tilgátuna má jafnvel líta á sem skilgreiningu þess hugtaks.

Athugasemd 2.31. Við hugsum okkur að stak í $\mathbb{N}$ megi líka líta á sem fall af gerðinni $\varnothing \to \mathbb{N}$. Við leyfum því að hlutrakið fall sé af gerðinni $\mathbb{N}^k \to \mathbb{N}$ með $k = 0$, en það er þá einfaldlega náttúrleg tala. Þetta kemur mest við sögu í reglu III í skilgreiningu 2.30, en þá má setja $G \in \mathbb{N}$. Lesanda í vafa er bent á [12, Kafli IV, §1, (1.4), (4)].

Dæmi 2.32. Skoðum nokkur hlutrakin föll:

(a) Fallið $O_+ : \mathbb{N}^2 \to \mathbb{N},\ O_+(n,m) = n+m$ er frumstætt rakið, enda gildir

$$\begin{aligned} & O_+(n,0) = n = \pi_1^1(n) \\ \text{og} \quad & O_+(n,m+1) = S(n+m) = (S \circ \pi_3^3)(n,m,O_+(n,m)). \end{aligned}$$

Fallið π_1^1 er frumstætt rakið skv. reglu I, fallið $S \circ \pi_1^3$ er frumstætt rakið skv. reglu II og því er O_+ frumstætt rakið skv. reglu III.

(b) Fallið $O_\times : \mathbb{N}^2 \to \mathbb{N},\ O_\times(n,m) = n \cdot m$ er frumstætt rakið enda gildir

$$O_\times(n,0) = 0 = Z(n) \quad \text{og} \quad O_\times(n,m+1) = O_+(n,O_\times(n,m)).$$

(c) Fallið $F : \mathbb{N} \to \mathbb{N},\ F(n) = n!$ er frumstætt rakið, enda gildir

$$F(0) = 1 \quad \text{og} \quad F(n+1) = O_\times(n+1, F(n+1)).$$

(d) Fallið $P : \mathbb{N} \to \mathbb{N}$,

$$P(n) = \begin{cases} 0 & \text{ef } n = 0 \\ n-1 & \text{annars} \end{cases}.$$

er frumstætt rakið, enda fæst það frá reglu III með $G = 0$ og $H = \pi_1^2$.

(e) Fallið $P^* : \mathbb{N}^* \to \mathbb{N},\ P^*(n) = n-1$ er hlutrakið en ekki rakið. Það fæst frá reglu IV með $H = P$ og

$$G(n,m) = \begin{cases} 0 & \text{ef } n = m \neq 0 \\ 1 & \text{annars} \end{cases}.$$

Við eftirlátum lesanda að sýna að G sé frumstætt rakið.

(f) *Ackermann-fallið* $A : \mathbb{N} \to \mathbb{N}$,

$$\begin{aligned} A(0,n) &= n+1, \\ A(m+1,0) &= A(m,1), \\ A(m+1,n+1) &= A(m, A(m+1,n)) \end{aligned}$$

er rakið en ekki frumstætt rakið. Sönnun þess að A sé rakið má finna í [6], og sönnun þess að A sé ekki frumstætt rakið má finna í [5].

Markmið þessa kafla er að sýna fram á jafngildi hlutrakinna falla og lambda-reiknings. Þar sem hlutrakin föll hafa náttúrlegar tölur sem grundvöll þurfum við þá að hafa einhverja leið til þess að tákna náttúrlegar tölur í lambda-reikningi. Helsta leiðin til þess er að nota svokallaðar *Church-tölur*:

Skilgreining 2.33. Látum n vera náttúrlega tölu. *Church-tala* fyrir n er λ-segðin $\bar{n} :=$ '$(\lambda xy.\,(x^n\, y))$'.

Lesanda sem kannast við Peano-reikning ætti ekki að vera komið hér á óvart, enda er þessi skilgreining hrein hliðstæða við skilgreiningu náttúrlegra talna þar. Þetta sést enn skýrar ef við notum táknin 'S' og '0' sem breytur:

$$\begin{array}{lll}
\bar{0} := \text{`}(\lambda xy.\,y)\text{'} & (\bar{0}\,S\,0) \twoheadrightarrow \text{`}0\text{'} & 0 := 0 \\
\bar{1} := \text{`}(\lambda xy.\,(x\,y))\text{'} & (\bar{1}\,S\,0) \twoheadrightarrow \text{`}(S\,0)\text{'} & 1 := S(0) \\
\bar{2} := \text{`}(\lambda xy.\,(x\,(x\,y)))\text{'} & (\bar{2}\,S\,0) \twoheadrightarrow \text{`}(S\,(S\,0))\text{'} & 2 := S(S(0)) \\
\bar{3} := \text{`}(\lambda xy.\,(x\,(x\,(x\,y))))\text{'} & (\bar{3}\,S\,0) \twoheadrightarrow \text{`}(S\,(S\,(S\,0)))\text{'} & 3 := S(S(S(0)))
\end{array}$$

o.s.frv.

Við viljum nú geta sagt til um hvort hlutrakið fall megi setja fram með λ-segð. Fyrst þurfum við að skilgreina nákvæmlega við hvað er átt þegar það er sagt:

Skilgreining 2.34. Látum A vera hlutmengi í $\mathbb{N}^k$ og $f : A \to \mathbb{N}$. Við segjum að λ-segð $\mathbf{F}$ sé *framsetning* fallsins f (*sem λ-segð*) ef

- fyrir $(n_1, \ldots, n_k) \in A$ gildir $(\mathbf{F}\,\bar{n}_1\,\cdots\,\bar{n}_k) \sim_\beta \overline{f(n_1, \ldots, n_k)}$, og
- fyrir $(n_1, \ldots, n_k) \notin A$ gildir að $(\mathbf{F}\,\bar{n}_1\,\cdots\,\bar{n}_k)$ hefur ekkert β-staðalform.

Við segjum þá jafnframt að f sé *framsetjanlegt* (*með λ-segð*) og ritum $[\![\mathbf{F}]\!] = f$.

Dæmi 2.35. Skoðum λ-segðina $\mathbf{D} :=$ '$(\lambda xyz.\,(z\,(\lambda a.\,y)\,x))$'. Gefum okkur λ-segðir $\mathbf{M}, \mathbf{N}$ ásamt breytu $\mathbf{a}$ sem er hvergi frjáls í $\mathbf{N}$, og athugum að

$$(\mathbf{D}\,\mathbf{M}\,\mathbf{N}\,\bar{m}) \twoheadrightarrow (\bar{m}\,(\lambda\mathbf{a}.\,\mathbf{N})\,\mathbf{M}) \twoheadrightarrow ((\lambda\mathbf{a}.\,\mathbf{N})^m\,\mathbf{M}) \twoheadrightarrow \begin{cases} \mathbf{M} & \text{ef } m = 0 \\ \mathbf{N} & \text{annars} \end{cases},$$

Sér í lagi höfum við

$$(\mathbf{D}\,\bar{p}\,\bar{q}\,\bar{m}) \twoheadrightarrow \begin{cases} \bar{p} & \text{ef } m = 0 \\ \bar{q} & \text{annars} \end{cases},$$

Segðin $\mathbf{D}$ er þá framsetning fallsins $D(p, q, 0) = p$, $D(p, q, m) = q$.

Þessi λ-segð er afar nytsamleg og við munum nota hana nokkrum sinnum hér á eftir. Hún leyfir okkur að velja á milli tveggja viðfanga eftir því þriðja. Við köllum hana því *skilyrðingaflétti*.

Dæmi 2.36. Skoðum föllin úr dæmi 2.32.

(a) Fallið O_+ er framsetjanlegt með λ-segðinni

$$\mathbf{O}_+ \coloneqq \text{‘}(\lambda xypq.\,(x\,p\,(y\,p\,q)))\text{’}$$

(sjá [3, Proposition 2.2.2]).

(b) Fallið $O_\times$ er framsetjanlegt með λ-segðinni

$$\mathbf{O}_\times \coloneqq \text{‘}(\lambda xyz.\,(x\,(y\,z)))\text{’}$$

(sjá [3, Proposition 2.2.2]).

(d) Fallið P er framsetjanlegt með λ-segðinni

$$\mathbf{P} \coloneqq \text{‘}(\lambda xyz.\,(x\,(\lambda ab.\,(b\,(a\,y)))\,(\lambda b.\,z)\,(\lambda a.\,a)))\text{’},$$

enda fæst

$$\begin{aligned}
&(\mathbf{P}\,\bar{0}) && \twoheadrightarrow \text{‘}(\lambda yz.\,((\lambda b.\,z)\,(\lambda a.\,a)))\text{’} \twoheadrightarrow \text{‘}(\lambda yz.\,z)\text{’} \sim_\alpha \bar{0}\\
\text{og}\quad &(\mathbf{P}\,\overline{m+1}) && \twoheadrightarrow \text{‘}(\lambda yz.\,((\lambda ab.\,(b\,(a\,y)))^{m+1}\,(\lambda b.\,z)\,(\lambda a.\,a)))\text{’}\\
& && \twoheadrightarrow \text{‘}(\lambda yz.\,((\lambda ab.\,(b\,(a\,y)))^{m}\,(\lambda b.\,(b\,z))\,(\lambda a.\,a)))\text{’}\\
& && \vdots\\
& && \twoheadrightarrow \text{‘}(\lambda yz.\,((\lambda ab.\,(b\,(a\,y)))^{m-k}\,(\lambda b.\,(b\,(y^k z)))\,(\lambda a.\,a)))\text{’}\\
& && \vdots\\
& && \twoheadrightarrow \text{‘}(\lambda yz.\,((\lambda b.\,(b\,(y^m\,z)))\,(\lambda a.\,a)))\text{’}\\
& && \twoheadrightarrow \text{‘}(\lambda yz.\,(y^m\,z))\text{’}\\
& && \sim_\alpha \bar{m}.
\end{aligned}$$

Dæmi 2.37. Skoðum fallið $F(n) = n!$ úr dæmi 2.32(c). Umræður í lok kafla 2.4 benda til þess að F sé framsetjanlegt með λ-segðinni

$$\mathbf{F} \coloneqq (\mathbf{Y}\,(\lambda fx.\,(\mathbf{D}\,\bar{1}\,(\mathbf{O}_\times\,x\,(f\,(\mathbf{P}\,x)))\,x))),$$

þar sem $\mathbf{D}$ er skilyrðingaflèttirinn úr dæmi 2.35. Í atriði III í sönnun hjálparsetningar 2.38 verður þetta staðfest.

Nú má byrja að sýna fram á jafngildi hlutrakinna falla og lambda-reiknings. Við sýnum fyrst að lambda-reikningur nái yfir frumstætt rakin föll.

Hjálparsetning 2.38. *Sérhvert frumstætt rakið fall er framsetjanlegt.*

Sönnun: Við notum þrepun yfir reglur I–III í skilgreiningu 2.30.

I. Einfalt er að staðfesta eftirfarandi:

- Núllfallið Z er framsetjanlegt með $\mathbf{Z} := (\lambda x.\, \bar{0})$.
- Eftirfarafallið S er framsetjanlegt með $\mathbf{S} :=$ '$(\lambda xab.\, (a\, (x\, a\, b)))$'.
- Ofanvarpsfallið π_i^k er framsetjanlegt með $\boldsymbol{\pi}_i^k :=$ '$(\lambda x_1 \ldots x_k.\, x_i)$'.

II. Gerum ráð fyrir að $\mathbf{G}, \mathbf{H}_1, \ldots, \mathbf{H}_m$ séu framsetningar $G, H_1, \ldots, H_m$. Þá er F framsetjanlegt með λ-segðinni

$$\mathbf{F} := (\lambda x_1 \ldots x_k.\, (\mathbf{G}\, (\mathbf{H}_1\, x_1\, \cdots\, x_k)\, \cdots\, (\mathbf{H}_m\, x_1\, \cdots\, x_k)).$$

enda fæst

$$\begin{aligned}
(\mathbf{F}\, \bar{n}_1\, \cdots\, \bar{n}_k) &\twoheadrightarrow (\mathbf{G}\, (\mathbf{H}_1\, \bar{n}_1\, \cdots\, \bar{n}_k)\, \cdots\, (\mathbf{H}_m\, \bar{n}_1\, \cdots\, \bar{n}_k)) \\
&\sim_\beta (\mathbf{G}\, \overline{H_1(n_1, \ldots, n_k)}\, \cdots\, \overline{H_m(n_1, \ldots, n_k)}) \\
&\sim_\beta \overline{G(H_1(n_1, \ldots, n_k), \ldots, H_m(n_1, \ldots, n_k))} \\
&= \overline{F(n_1, \ldots, n_k)}.
\end{aligned}$$

III. Okkur nægir að finna λ-segð $\mathbf{R}$ sem uppfyllir, fyrir sérhverjar λ-segðir $\mathbf{M}$ og $\mathbf{N}$, að

$$(\mathbf{R}\, \mathbf{M}\, \mathbf{N}\, \bar{0}) \sim_\beta \mathbf{M} \quad \text{og} \quad (\mathbf{R}\, \mathbf{M}\, \mathbf{N}\, \overline{m+1}) \sim_\beta (\mathbf{N}\, \bar{m}\, (\mathbf{R}\, \mathbf{M}\, \mathbf{N}\, \bar{m})).$$

Ef slík segð $\mathbf{R}$ er til, og $\mathbf{G}, \mathbf{H}$ eru framsetningar G, H, þá er F framsetjanlegt með λ-segðinni

$$\mathbf{F} := (\lambda x_1 \ldots x_k y.\, (\mathbf{R}\, (\mathbf{G}\, x_1\, \cdots\, x_k)\, (\mathbf{H}\, x_1\, \cdots\, x_k)\, y))$$

enda fæst

$$\begin{aligned}
(\mathbf{F}\, \bar{n}_1\, \cdots\, \bar{n}_k\, \bar{0}) &\twoheadrightarrow (\mathbf{R}\, (\mathbf{G}\, \bar{n}_1\, \cdots\, \bar{n}_k)\, (\mathbf{H}\, \bar{n}_1\, \cdots\, \bar{n}_k)\, \bar{0}) \\
&\sim_\beta (\mathbf{G}\, \bar{n}_1\, \cdots\, \bar{n}_k) \\
&\sim_\beta \overline{G(n_1, \ldots, n_k)} \\
&= \overline{F(n_1, \ldots, n_k, 0)},
\end{aligned}$$

og ef gert er ráð fyrir að $(\mathbf{F}\,\bar{n}_1\,\cdots\,\bar{n}_k\,\bar{m}) \sim_\beta \overline{F(n_1,\ldots,n_k,m)}$ fæst

$$\begin{aligned}(\mathbf{F}\,\bar{n}_1\,\cdots\,\bar{n}_k\,\overline{m+1}) \twoheadrightarrow\ & (\mathbf{R}\,(\mathbf{G}\,\bar{n}_1\,\cdots\,\bar{n}_k)\,(\mathbf{H}\,\bar{n}_1\,\cdots\,\bar{n}_k)\,\overline{m+1})\\ \sim_\beta\ & (\mathbf{H}\,\bar{n}_1\cdots\bar{n}_k\,\bar{m}(\mathbf{R}\,(\mathbf{G}\,\bar{n}_1\cdots\bar{n}_k)\,(\mathbf{H}\,\bar{n}_1\cdots\bar{n}_k)\,\bar{m}))\\ \sim_\beta\ & (\mathbf{H}\,\bar{n}_1\,\cdots\,\bar{n}_k\,\bar{m}\,\overline{F(n_1,\ldots,n_k,m)})\\ \sim_\beta\ & \overline{H(n_1,\ldots,n_k,m,F(n_1,\ldots,n_k,m))}\\ =\ & \overline{F(n_1,\ldots,n_k,m+1)}.\end{aligned}$$

Við snúum okkur nú að því að finna viðeigandi λ-segð $\mathbf{R}$. Við viljum að $\mathbf{R}$ uppfylli

$$(\mathbf{R}\,x\,y\,z) \sim_\beta (\mathbf{D}\,x\,(y\,(\mathbf{P}\,z)\,(\mathbf{R}\,x\,y\,(\mathbf{P}\,z)))\,z),$$

þar sem $\mathbf{D}$ er skilyrðingafléttirinn úr dæmi 2.35 og $\mathbf{P}$ er framsetning fallsins P úr dæmi 2.36(d), en slíkt $\mathbf{R}$ er til samkvæmt setningu 2.29, og fæst með

$$\mathbf{R} \coloneqq (\mathbf{Y}\,(\lambda rxyz.\,(\mathbf{D}\,x\,(y\,(\mathbf{P}\,z)\,(r\,x\,y\,(\mathbf{P}\,z)))\,z))). \qquad \square$$

Nú má snúa sér að reglu IV úr skilgreiningu 2.30. Við sýnum að rakin föll séu framsetjanleg, en að sanna niðurstöðuna fyrir hlutrakin föll er talsvert flóknara.

Hjálparsetning 2.39. *Sérhvert rakið fall er framsetjanlegt.*

Sönnun: Við höfum þegar sýnt að frumstætt rakið fall er framsetjanlegt, svo það nægir að sýna að rakið fall sem fæst með reglu IV úr skilgreiningu 2.30 sé framsetjanlegt.

Látum $\mathbf{G}, \mathbf{H}$ vera framsetningar G, H úr IV og látum $F : \mathbb{N}^k \to \mathbb{N}$ vera skilgreint eins og í IV (við gerum hér ráð fyrir að $A = \mathbb{N}^k$). Okkur nægir að finna λ-segð $\mathbf{L}$ sem uppfyllir

$$(\mathbf{L}\,\bar{n}_1\,\cdots\,\bar{n}_k\,\bar{m}) \sim_\beta \begin{cases}\bar{m} & \text{ef } G(n_1,\ldots,n_k,m)=0\\ (\mathbf{L}\,\bar{n}_1\,\cdots\,\bar{n}_k\,\overline{m+1}) & \text{annars}\end{cases}.$$

Ef slíkt $\mathbf{L}$ er til má auðveldlega sannfæra sig um að F sé framsetjanlegt með

$$\mathbf{F} \coloneqq (\lambda x_1\ldots x_k.\,(\mathbf{H}\,(\mathbf{L}\,x_1\,\cdots\,x_k\,\bar{0}))).$$

Skoðum aftur $\mathbf{D}$ frá dæmi 2.35. Við viljum að $\mathbf{L}$ uppfylli

$$(\mathbf{L}\,x_1\,\cdots\,x_k\,y) \sim_\beta (\mathbf{D}\,y\,(\mathbf{L}\,x_1\,\cdots\,x_k\,(\mathbf{S}\,y))\,(\mathbf{G}\,x_1\,\cdots\,x_k\,y)),$$

en slíkt $\mathbf{L}$ er til samkvæmt setningu 2.29, og fæst með

$$\mathbf{L} \coloneqq (\mathbf{Y}\,(\lambda \ell x_1\ldots x_k y.\,(\mathbf{D}\,y\,(\ell\,x_1\,\cdots\,x_k\,(\mathbf{S}\,y))\,(\mathbf{G}\,x_1\,\cdots\,x_k\,y)))). \qquad \square$$

Hér er freistandi að halda að þessi sönnun gildi fyrir öll hlutrakin föll, auk rakinna falla. Hins vegar er auðvelt að finna dæmi um hlutrakið fall F þannig að $\mathbf{F}$ eins og gefið er að ofan sé ekki framsetning F; ef til dæmis $H = Z$ og $G = \pi^2_1$, og F er gefið með reglu IV, þá er $F(n)$ ekki skilgreint nema ef $n = 0$, en

$$(\mathbf{F}\,\bar{n}) \twoheadrightarrow_\beta (\mathbf{Z}\,(\mathbf{L}\,\bar{n}\,\bar{0})) \twoheadrightarrow_\beta \bar{0}.$$

Með vandvirkni má breyta segðinni $\mathbf{F}$ til þess að leysa úr þessum vanda, og sanna þannig niðurstöðuna fyrir öll hlutrakin föll. Til fróðleiks er ein slík segð $\mathbf{F}'$ skilgreind með

$$\begin{aligned}
\mathbf{T} &:= (\lambda x.\,(\mathbf{D}\,\bar{0}\,(\lambda ab.\,(a\,(x\,(\mathbf{S}\,b))\,x\,(\mathbf{S}\,b))))),\\
\mathbf{P} &:= (\lambda xy.\,(\mathbf{T}\,x\,(x\,y)\,(\mathbf{T}\,x)\,y)),\\
\mathbf{F}' &:= (\lambda x_1 \ldots x_k.\,(\mathbf{P}\,(\mathbf{G}\,x_1\,\cdots\,x_k)\,\bar{0}\,(\lambda a.\,a)\,(\mathbf{F}\,x_1\,\cdots\,x_k)))
\end{aligned}$$

(sjá [9, Theorem 4.23]). Að sanna það að $\mathbf{F}'$ uppfyllir nauðsynleg skilyrði myndi flækja þessa framsetningu um of. Til dæmis þyrfti svokallaða vinstrislagsíðaeinföldunarsetningu (e. *quasi-leftmost reduction theorem*) til að ganga úr skugga um að $\mathbf{F}'$ hafi þann eiginleika sem $\mathbf{F}$ vantar. Þessi setning er tekin fyrir í [9, Chapter 3D].

Með þetta í huga látum við nægja að setja fram þessa niðurstöðu án sönnunar. Áhugasömum lesanda er bent á [9, Theorem 4.23].

Setning 2.40. *Sérhvert hlutrakið fall er framsetjanlegt.*

Glöggur lesandi hefur eflaust tekið eftir að sannanir hjálparsetninga 2.38 og 2.39 nota segðir sem fást með beitingu setningar 2.29. Þessi setning tryggir í sjálfu sé ekki að setningar sem fást með beitingu hennar séu á β-staðalformi, enda er þar notaður fastapunktsfléttir. Við getum þá ekki sagt til um það hvort hlutrakið fall sé ávallt framsetjanlegt með λ-segð á β-staðalformi. Lesanda leikur eflaust forvitni á að vita þetta, og er raunin sú að það er ávallt hægt. Að sýna þetta krefst hins vegar talsvert flóknari λ-segða og lengri sannana en við höfum færi á hér. Við látum því einnig nægja að setja fram eftirfarandi setningu án sönnunar. Áhugasömum lesanda er aftur bent á [9, Theroem 4.24].

Setning 2.41. *Sérhvert hlutrakið fall er framsetjanlegt með λ-segð á β-staðalformi.*

Við höfum hingað til sýnt að lambda-reikningur sé að minnsta kosti eins öflugur og hlutröknu föllin, en markmið þessa kafla var að sýna að þetta séu jafngild líkön. Við eigum því eftir að sýna að lambda-reikningur sé ekki öflugri en hlutröknu föllin, þ.e.a.s. að sérhvert fall sem er framsetjanlegt með λ-segð sé hlutrakið.

Því miður er þetta enn önnur setning sem hefur of langa sönnun til þess að sýna hér. Enn önnur setning — í þetta sinn meginniðurstaða undirkaflans — kemur því hér fram án sönnunar. Fyrsta sönnun hennar kom fram í [10].

Setning 2.42. *Fall er framsetjanlegt þá og því aðeins að það sé hlutrakið.*

3 Einföld tagskipting

3.1 Einföld tög og úthlutanir

Í þeim lambda-reikningi sem við höfum skoðað fram að þessu hefur engin leið verið til þess að aðgreina mismunandi gerðir af viðföngum innan kerfisins. Að túlka λ-segðir sem föll hefur þá þann ókost að setja þarf upp samkomulag um hvernig viðföng λ-segða skuli líta út til þess að segja megi að segð sé framsetning stærðfræðilegs falls.

Við höfum þegar lent í þessari flækju þegar við skilgreindum Church-tölur og framsetjanleg föll af gerðinni $\mathbb{N}^k \to \mathbb{N}$. Þar sögðum við (í skilgreiningu 2.34) að λ-segð væri framsetning slíks falls ef beiting þess með Church-tölurnar $\bar{n}_1, \cdots, \bar{n}_k$ sem viðföng gæfi Church-töluna fyrir gildi fallsins í $(n_1, \cdots, n_k)$. Hins vegar er á þennan hátt ekkert sagt til um hvernig framsetningin hegðar sér þegar henni er beitt á viðfang sem ekki er Church-tala. Til dæmis má skoða segðina $\mathbf{\Omega}$ úr dæmi 2.14(e). Þetta er greinilega ekki Church-tala, og þá er ekkert vit í að beita slíkri framsetningu á viðfang $\mathbf{\Omega}$. Á hinn bóginn er ekkert sem stoppar það. Við fáum til dæmis

$$\begin{aligned} &(\mathbf{Z}\,\mathbf{\Omega}) \sim_\beta \bar{0}, \\ &(\mathbf{O}_\times\,\mathbf{\Omega}\,\bar{0}) \sim_\beta (\lambda z.\,(\mathbf{\Omega}\,(\bar{0}\,z))) \sim_\beta (\mathbf{K}\,(\mathbf{\Omega}\,\mathbf{I})) \\ \text{og}\quad &(\mathbf{O}_\times\,\bar{0}\,\mathbf{\Omega}) \sim_\beta (\lambda z.\,(\bar{0}\,(\mathbf{\Omega}\,z))) \sim_\beta (\mathbf{K}\,\mathbf{I}). \end{aligned}$$

Stundum fæst niðurstaða sem er í einhverju samræmi við það sem við myndum búast við, eins og fyrir $(\mathbf{Z}\ \mathbf{\Omega})$, en í öðrum tilvikum fæst niðurstaða sem virðist vera hrein þvæla í samhengi framsetjanlegra falla. Meira að segja sést að segðin $\mathbf{O}_\times$, sem er framsetning margföldunar, er ekki endilega víxlin fyrir viðföng sem ekki eru Church-tölur, og gefur ekki endilega Church-töluna $\bar{0}$ ef eitt viðfanganna er $\bar{0}$.

Ekki er erfitt að ímynda sér að stundum sé æskilegt að geta talað um framsetningar falla af annarri gerð en $\mathbb{N}^k \to \mathbb{N}$, og þá þarf að leika þennan leik í hvert sinn. Í almennri forritun er þetta einnig mikilvægt, sbr. kafla 1.2. Í mörgum tilvikum má hins vegar í staðinn nota *tagskiptingu.* Tagskipting er hagnýting á *gerðarfræði* (e. *type theory*) og er notuð í einhverri mynd í langflestum forritunarmálum nútímans. Með tagskiptingu lambda-reiknings gefum við λ-segðum *tög* (í gerðarfræði er oft notað orðið *gerð* í staðinn fyrir tag, en orðið tag er algengara þegar talað er um forritunarmál). Tag segðar segir svo til um hvernig hana megi nota með öðrum segðum.

Við munum í þessum kafla skoða svokallaða *einfalda tagskiptingu*, sem er, eins og nafnið bendir til, einfaldasta (áhugaverða) tagskipting lambda-reiknings. Til eru tvær ráðandi útgáfur af einfaldri tagskiptingu í notkun, sem kenndar eru við Church annars vegar og við Curry hins vegar. Með Church-útgáfunni eru tög bindandi breyta skrifuð inn í λ-segðir, og litið á að tilsvarandi fall sé aðeins vel skilgreint fyrir viðföng af því tagi, þ.e. ritað er $(\lambda\mathbf{x}{:}\sigma.\,\mathbf{M})$ til þess að tákna að fallið hafi viðfangstag σ. Þetta hefur í för með sér að til dæmis tákni $(\lambda x{:}\sigma.\,x)$ og $(\lambda x{:}\tau.\,x)$ mismunandi föll ef σ og τ eru ólík, þótt við hugsum um báðar segðirnar sem samsemdarvörpunina. Segðunum er þá gefið mismunandi tög, annars vegar $(\sigma \to \sigma)$ og hins vegar $(\tau \to \tau)$. Stundum er þetta sjálfsagt, en í okkar tilviki viljum við frekar skoða Curry-útgáfuna. Ef lesandi hefur áhuga má framsetningu á Church-útgáfunni finna í ritgerð Arnars Ágústs í þessu sama bindi [1].

Curry-útgáfa einfaldrar tagskiptingar snertir ekki segðir lambda-reiknings, en reiðir sig einungis á útleiðslureglur sem nota má til þess að gefa λ-segðum tög. Þá má gefa λ-segðinni '$(\lambda x.\,x)$' bæði tagið $(\sigma \to \sigma)$ og tagið $(\tau \to \tau)$.

Við byrjum á því að skilgreina einföld tög sem formlegt mál.

Skilgreining 3.1. Formlegt mál $\mathcal{S}$ fyrir *einföld tög* er skilgreint yfir stafrófið $\mathcal{V} \cup \{`\to', `(', `)'\}$, þar sem $\mathcal{V}$ er teljanlega óendanlegt mengi *tagbreyta*. Við köllum segðir þess (*einföld*) *tög*, og eru þær gefnar með eftirfarandi reglum:

(i) Stæða af gerðinni $\mathbf{a}$ þar sem $\mathbf{a}$ er tagbreyta er einfalt tag.

(ii) Ef σ og τ eru einföld tög, þá er $(\sigma \to \tau)$ einfalt tag

Segðir sem fást frá myndunarreglu (ii) nefnast *vörpunartög*.

Athugasemd 3.2. Við tökum upp þá skammmstöfun að '$\to$' sé tengin til hægri, þ.e.a.s. við ritum $(\alpha \to \beta \to \gamma)$ í stað $(\alpha \to (\beta \to \gamma))$. Við leyfum okkur jafnframt að nota þrípunkt þegar mörg tög liggja saman, eins og til dæmis með $(\sigma_1 \to \cdots \to \sigma_n)$.

Athugasemd 3.3. Þegar við tölum um einföld tög notum við gríska lágstafi á sama hátt og við höfum notað feitletraða hástafi í λ-segðum. Þannig ætti til dæmis ekki að hugsa um σ sem táknið 'σ', heldur sem óþekkta segð í $\mathcal{S}$.

Þegar við tölum um tagbreytur notum við feitletraða lágstafi enska stafrófsins, líkt og þegar við tölum um breytur í λ-segðum. Við grípum hins vegar frekar til stafa sem liggja framar í stafrófinu en aftar. Til dæmis notum við $\mathbf{a}$ oftar en ekki til þess að tákna tagbreytu.

Nú skilgreinum við þær útleiðslureglur sem nota má til þess að gefa λ-segð tag. Rithátturinn hér á eftir kann að koma á óvart, en hann er sá sami og er notaður í formlegum sönnunum í stærðfræðilegri rökfræði. Ástæða þess hefur að gera með Curry-Howard-samsvörunina, sem lesa má nánar um í [1]. Útleiðslureglurnar að neðan mynda svokallað *náttúrlegt afleiðslukerfi.* Við tölum ekki nánar um það hér, en þetta er skilgreint í [1].

Skilgreining 3.4. Látum $\mathbf{P}$ vera λ-segð og τ vera einfalt tag. Segð af gerðinni $\mathbf{P} : \tau$ er kölluð *úthlutun.* Þetta skilgreinir formlegt mál yfir stafrófið $\Sigma(\lambda) \cup \Sigma(\mathcal{S}) \cup \{\text{‘:’}\}$. Sagt er að úthlutunin *gefi* $\mathbf{P}$ tagið τ.

Látum Γ vera mengi úthlutana sem gefa aðeins breytum tög, og gerum ráð fyrir að ef $\mathbf{x}:\sigma, \mathbf{x}:\tau \in \Gamma$, þá sé $\sigma = \tau$ (þ.e.a.s. engri breytu er gefið fleira en eitt tag). Við köllum þá Γ *samhengi.*

Sagt er að Γ *leiði af sér úthlutunina* A og ritað $\Gamma \vdash A$ ef þetta má leiða út með því að beita eftirfarandi útleiðslureglum endanlega oft:

(i) Ef $\mathbf{x}:\sigma \in \Gamma$, þá $\Gamma \vdash \mathbf{x}:\sigma$.

(ii) Ef $\Gamma, \mathbf{x}:\sigma \vdash \mathbf{M}:\tau$, þá $\Gamma \vdash (\lambda\mathbf{x}.\,\mathbf{M}):(\sigma \to \tau)$.

(iii) Ef $\Gamma \vdash \mathbf{M}:(\sigma \to \tau)$ og $\Gamma \vdash \mathbf{N}:\sigma$, þá $\Gamma \vdash (\mathbf{M}\,\mathbf{N}):\tau$.

Athugasemd 3.5. Við leyfum okkur að rita $\mathbf{x}_1 : \sigma_1, \ldots, \mathbf{x}_n : \sigma_n \vdash A$ í stað $\{\mathbf{x}_1 : \sigma_1, \ldots, \mathbf{x}_n : \sigma_n\} \vdash A$. Við ritum einnig $\Gamma, \mathbf{x}:\sigma \vdash A$ í stað $\Gamma \cup \{\mathbf{x}:\sigma\} \vdash A$ (eins og gert var í (ii)), og gerum þá jafnframt ráð fyrir að breytan $\mathbf{x}$ komi hvergi fyrir í úthlutunum Γ. Auk þess ritum við $\vdash A$ í stað $\varnothing \vdash A$.

Dæmi 3.6. Skoðum nokkrar úthlutanir:

(a) Fyrir breytu $\mathbf{x}$ og einfalt tag σ gildir $\vdash (\lambda\mathbf{x}.\,\mathbf{x}):(\sigma \to \sigma)$:

Útleiðsluregla (i) gefur $\mathbf{x} : \sigma \vdash \mathbf{x} : \sigma$, og þar með fæst niðurstaðan frá útleiðslureglu (ii).

(b) Fyrir breytur $\mathbf{x}$ og $\mathbf{y}$ og einföld tög σ og τ gildir $\vdash (\lambda\mathbf{x}\mathbf{y}.\,\mathbf{x}):(\sigma \to \tau \to \sigma)$:

Við höfum $\mathbf{x} : \sigma, \mathbf{y} : \tau \vdash \mathbf{x} : \sigma$ skv. útleiðlsureglu (i), svo $\mathbf{x}:\sigma \vdash (\lambda\mathbf{y}.\,\mathbf{x}) : (\tau \to \sigma)$ skv. útleiðslureglu (ii). Niðurstaðan fæst nú með annarri beitingu á útleiðslureglu (ii).

(c) Fyrir $\mathbf{I} = \text{‘}(\lambda x.\, x)\text{’}$ og einfalt tag σ gildir $\vdash (\mathbf{I}\,\mathbf{I}):(\sigma \to \sigma)$:

Skv. (a) höfum við bæði $\vdash \mathbf{I}:(\sigma \to \sigma)$ og $\vdash \mathbf{I}:((\sigma \to \sigma) \to (\sigma \to \sigma))$, svo niðurstaðan fæst með útleiðslureglu (iii).

(d) Ekki er til samhengi Γ sem gefur segðinni '$(x\,x)$' tag:

Útleiðsla þess þyrfti að ljúka á beitingu á útleiðslureglu (iii), en þá þyrfti að gilda $\Gamma \vdash$ 'x'$:\sigma$ og $\Gamma \vdash$ 'x'$:(\sigma \to \tau)$ fyrir einhver σ, τ. Þessar úthlutanir má einungis fá með útleiðslureglu (i), en þá höfum við 'x' $: \sigma \in \Gamma$ og 'x'$:(\sigma \to \tau)$. Þetta er í mótsögn við að Γ er samhengi.

Liður (d) að ofan sýnir að ekki megi gefa öllum λ-segðum tag. Þetta er ekki ókostur — einn tilgangur tagskiptinga er að skera úr um hvaða λ-segðir koma heim og saman við vel skilgreindar aðgerðir. Segðin '$(x\,x)$' táknar beitingu einhvers falls á sjálft sig, en þetta má líkja við þversögn Russells úr mengjafræði um hvort mengi allra mengja innihaldi sig sjálft. Meira að segja var það Russell sjálfur sem kynnti gerðarfræði fyrst til sögunnar, einmitt til þess að leysa þennan vanda.

Við höfum þá sérstakan áhuga á λ-segðum sem gefa má eitthvert tag, eða svokölluðum *taganlegum* segðum:

Skilgreining 3.7. Látum $\mathbf{P}$ vera λ-segð og látum $\mathbf{x}_1, \ldots, \mathbf{x}_n$ vera upptalningu frjálsa breyta sem koma fyrir í $\mathbf{P}$. Sagt er að $\mathbf{P}$ sé *taganleg* ef til eru tög $\sigma_1, \ldots, \sigma_n, \tau$ þannig að

$$\mathbf{x}_1 : \sigma_1, \ldots, \mathbf{x}_n : \sigma_n \vdash \mathbf{P} : \tau.$$

3.2 Stöðlunarsetningin

Skoðum segðina $(\mathbf{K}\,x\,\boldsymbol{\Omega})$, með $\mathbf{K}$ úr dæmi 2.4. og $\boldsymbol{\Omega}$ úr dæmi 2.14(e). Tökum eftir að þessi segð hefur nákvæmlega tvær β-einfaldanir:

$$(\mathbf{K}\,x\,\boldsymbol{\Omega}) \twoheadrightarrow_\beta \begin{cases} \text{`}x\text{'} \\ (\mathbf{K}\,x\,\boldsymbol{\Omega}) \end{cases}.$$

Greinilegt er að segðin hafi 'x' sem β-staðalform, en ef farið er rangt að í einföldun segðarinnar má auðveldlega lenda í óendanlegu einfölduninni

$$(\mathbf{K}\,x\,\boldsymbol{\Omega}) \twoheadrightarrow_\beta (\mathbf{K}\,x\,\boldsymbol{\Omega}) \twoheadrightarrow_\beta (\mathbf{K}\,x\,\boldsymbol{\Omega}) \twoheadrightarrow_\beta \cdots.$$

Þessu má líkja við að forrit lendi í óendanlegri lykkju þrátt fyrir að geta sneitt hjá því.

Ein ástæða þess að við skoðum tagskiptingar og taganlegar segðir er til að geta sett fram kerfi sem eykur traust á því að forrit skili réttri niðurstöðu með því að útiloka alveg tilteknar gerðir af villum. Sjaldnast er forrit vel skrifað ef það getur lent í þessari klípu, en við munum sjá með svokallaðri *stöðlunarsetningu* að taganlegar λ-segðir eru óhultar gagnvart því.

Skilgreinum fyrst nákvæmlega hvað ofangreint dæmi skýrir frá.

Skilgreining 3.8. Sagt er að λ-segð $\mathbf{P}$ sé *veikt staðlandi* ef hún á sér β-staðalform. Enn fremur er sagt að $\mathbf{P}$ sé *strangt staðlandi* ef sérhver runa $\mathbf{P}_1, \mathbf{P}_2, \ldots$ sem uppfyllir $\mathbf{P} \twoheadrightarrow \mathbf{P}_1 \twoheadrightarrow \mathbf{P}_2 \twoheadrightarrow \cdots$ inniheldur segð á β-staðalformi (þ.e. engin einföldun $\mathbf{P}$ hefur óendanlega margar β-einfaldanir).

Mengi allra veikt staðlandi λ-segða er táknað WN, og mengi allra strangt staðlandi λ-segða er táknað SN.

Dæmi 3.9. Við höfum þegar séð:

- Segðin $(\mathbf{K}\, x\, y)$ er strangt staðlandi.
- Segðin $(\mathbf{K}\, x\, \boldsymbol{\Omega})$ er veikt staðlandi en ekki strangt staðlandi.
- Segðin $\boldsymbol{\Omega}$ er hvorki veikt né strangt staðlandi.

Ljóst er að ströng stöðlun er sterkara skilyrði en veik stöðlun, þ.e.a.s. sérhver strangt staðlandi λ-segð er jafnframt veikt staðlandi. Meginniðurstaða þessa kafla, stöðlunarsetningin (3.15), segir að λ-segð sé ekki taganleg nema hún sé strangt staðlandi. Þannig gildir að ef hægt er að gefa λ-segð tag, þá á hún ekki bara β-staðalform, heldur er tryggt að sérhver einföldun hennar leiðir til þess.

Restin af þessum undirkafla mun fara í að sanna stöðlunarsetninguna. Við förum eins að og í [3, 4.3.1–4.3.6], en þar er miðað við sönnun sem kennd er við Tait og Girard. Stöðlunarsetningin er þá hér í rauninni fylgisetning tveggja hjálparsetninga (3.12 og 3.14). Fyrst þurfum að skilgreina nokkur hugtök sem þessar hjálparsetningar byggja á.

Skilgreining 3.10. Við úthlutum sérhverju einföldu tagi σ mengi $\chi(\sigma) \subseteq \lambda$, þannig að $\chi(\mathbf{a}) = \mathrm{SN}$ fyrir sérhverja tagbreytu $\mathbf{a}$, og $\chi((\sigma \to \tau))$ er mengi þeirra λ-segða $\mathbf{P}$ sem uppfylla að $(\mathbf{P}\, \mathbf{A})$ sé í $\chi(\tau)$ fyrir sérhvert $\mathbf{A}$ úr $\chi(\sigma)$.

Skilgreining 3.11. Látum $\mathcal{M}$ vera mengi af λ-segðum. Við segjum að $\mathcal{M}$ sé *mettað* ef eftirfarandi skilyrði eru uppfyllt:

(i) Sérhver segð í $\mathcal{M}$ er strangt staðlandi.

(ii) Sérhver λ-segð sem rita má $(\mathbf{x}\, \mathbf{N}_1 \cdots \mathbf{N}_m)$, þar sem $\mathbf{x}$ er breyta og $\mathbf{N}_1, \ldots, \mathbf{N}_m$ eru strangt staðlandi λ-segðir, er innihaldin í $\mathcal{M}$.

(iii) Ef $\mathcal{M}$ inniheldur λ-segð $(\mathbf{P}_{\mathbf{x}}[\mathbf{Q}]\, \mathbf{N}_1 \cdots \mathbf{N}_m)$, þar sem $\mathbf{Q}$ er strangt staðlandi, þá inniheldur $\mathcal{M}$ einnig segðina $((\lambda \mathbf{x}.\, \mathbf{P})\, \mathbf{Q}\, \mathbf{N}_1 \cdots \mathbf{N}_m)$.

Við getum nú sannað fyrri hjálparsetninguna:

Hjálparsetning 3.12. *Mengið $\chi(\sigma)$ er mettað fyrir sérhvert einfalt tag σ.*

Sönnun: Okkur nægir að sýna að SN sé mettað og að ef $\chi(\sigma)$ og $\chi(\tau)$ séu mettuð, þá sé $\chi((\sigma \to \tau))$ mettað. Ef þetta tvennt gildir fæst niðurstaðan með þrepun yfir myndunarreglur einfaldra taga.

Við sýnum fyrst að SN sé mettað:

(i) Sérhver segð í SN er strangt staðlandi samkvæmt skilgreiningu.

(ii) Látum $\mathbf{x}$ vera breytu og $\mathbf{N}_1, \ldots, \mathbf{N}_m$ vera strangt staðlandi λ-segðir. Nú má ekki einfalda λ-segðina $\mathbf{M} := (\mathbf{x}\,\mathbf{N}_1 \cdots \mathbf{N}_m)$ nema með því að einfalda eina af hlutsegðunum $\mathbf{N}_1, \ldots, \mathbf{N}_m$, en þar sem sérhver þeirra er strangt staðlandi er ljóst að $\mathbf{M}$ er þá einnig strangt staðlandi.

(iii) Gerum ráð fyrir að $\mathbf{Q}$ og $\mathbf{M} := (\mathbf{P}_{\mathbf{x}}[\mathbf{Q}]\,\mathbf{N}_1 \cdots \mathbf{N}_m)$ séu strangt staðlandi. Þá er hver hlutsegðanna $\mathbf{N}_1 \ldots, \mathbf{N}_m$ strangt staðlandi, auk hlutsegðarinnar $\mathbf{P}_{\mathbf{x}}[\mathbf{Q}]$. Nú hefur $\mathbf{P}_{\mathbf{x}}[\mathbf{Q}]$ sömu einfaldanir og $\mathbf{P}$, ásamt hugsanlega einföldunum $\mathbf{Q}$, svo ljóst er að $\mathbf{P}$ er líka strangt staðlandi.

Eina einföldun segðarinnar $\mathbf{M}' := ((\lambda\mathbf{x}.\,\mathbf{P})\,\mathbf{Q}\,\mathbf{N}_1 \cdots \mathbf{N}_m)$ sem er ekki einföldun strangt staðlandi hlutsegðanna $\mathbf{P}, \mathbf{Q}, \mathbf{N}_1, \ldots, \mathbf{N}_m$ er nú einföldunin $\mathbf{M}' \twoheadrightarrow_\beta \mathbf{M}$ en þar með er ljóst að $\mathbf{M}'$ er strangt staðlandi.

Við gerum nú ráð fyrir að $\chi(\sigma)$ og $\chi(\tau)$ séu mettuð og sýnum að þá sé $\chi((\sigma \to \tau))$ mettað:

(i) Látum $\mathbf{P}$ vera λ-segð í $\chi((\sigma \to \tau))$, og $\mathbf{A}$ vera λ-segð í $\chi(\sigma)$. Þá er $(\mathbf{P}\,\mathbf{A})$ í $\chi(\tau)$, og þar með strangt staðlandi. Þar sem $\mathbf{P}$ er hlutsegð í $(\mathbf{P}\,\mathbf{A})$ gefur þetta einnig að $\mathbf{P}$ er strangt staðlandi.

(ii) Látum $\mathbf{x}$ vera breytu og $\mathbf{N}_1, \ldots, \mathbf{N}_m$ vera strangt staðlandi λ-segðir. Við viljum sýna að fyrir $\mathbf{A}$ úr $\chi(\sigma)$, þá sé $(\mathbf{x}\,\mathbf{N}_1 \cdots \mathbf{N}_m\,\mathbf{A})$ í $\chi(\tau)$, en þetta er ljóst, enda er $\mathbf{A}$ þá strangt staðlandi og $\chi(\tau)$ mettað.

(iii) Gerum ráð fyrir að $\mathbf{Q}$ sé strangt staðlandi og að $(\mathbf{P}_{\mathbf{x}}[\mathbf{Q}]\,\mathbf{N}_1 \cdots \mathbf{N}_m)$ sé innihaldið í $\chi((\sigma \to \tau))$. Látum $\mathbf{A}$ vera úr $\chi(\sigma)$. Þá er segðin $(\mathbf{P}_{\mathbf{x}}[\mathbf{Q}]\,\mathbf{N}_1 \cdots \mathbf{N}_m\,\mathbf{A})$ í $\chi(\tau)$, og þá líka segðin $((\lambda\mathbf{x}.\,\mathbf{P})\,\mathbf{Q}\,\mathbf{N}_1 \cdots \mathbf{N}_m\,\mathbf{A})$. Þetta sýnir að $((\lambda\mathbf{x}.\,\mathbf{P})\,\mathbf{Q}\,\mathbf{N}_1 \cdots \mathbf{N}_m)$ er í $\chi((\sigma \to \tau))$. □

Við snúum okkur nú að seinni hjálparsetningunni. Áður en hún er sett fram þurfum við að eina skilgreiningu í viðbót. Við notum hér á eftir innsetningasöfn, sem skilgreind voru í skilgreiningu 2.8.

Skilgreining 3.13. Við segjum að innsetningasafn $\mathfrak{S}$ *svali úthlutuninni* $\mathbf{M}:\sigma$ ef segðin $\mathbf{M}[\mathfrak{S}]$ er innihaldin í $\chi(\sigma)$. Við segjum jafnframt að $\mathfrak{S}$ *svali samhenginu* Γ ef $\mathfrak{S}$ svalar sérhverri úthlutun $\mathbf{x}:\sigma$ í Γ.

Enn fremur er sagt að samhengi Γ *svali úthlutuninni* $\mathbf{M} : \sigma$ ef sérhvert innsetningasafn $\mathfrak{S}$ sem svalar samhenginu Γ svalar úthlutuninni $\mathbf{M}:\sigma$.

Þetta leyfir okkur að tengja saman $\chi(\sigma)$ og úthlutanir með σ:

Hjálparsetning 3.14. *Ef* $\Gamma \vdash \mathbf{M}:\sigma$*, þá svalar* Γ *úthlutuninni* $\mathbf{M}:\sigma$*.*

Sönnun: Við sönnum niðurstöðuna með þrepun yfir myndunarreglur λ-segða (skilgreining 2.3) fyrir hverja af útleiðslureglum (i)–(iii) í skilgreiningu 3.4.

(i) Ef $\Gamma \vdash \mathbf{x}:\sigma$, þá höfum við $\mathbf{x}:\sigma \in \Gamma$.

Ef nú innsetningasafn $\mathfrak{S}$ svalar Γ, þá gildir sér í lagi að $\mathfrak{S}$ svalar úthlutuninni $\mathbf{x}:\sigma$, en þar með svalar Γ úthlutuninni $\mathbf{x}:\sigma$.

(ii) Ef $\Gamma \vdash (\lambda \mathbf{x}.\, \mathbf{M}):(\sigma \to \tau)$, þá gildir $\Gamma, \mathbf{x}:\sigma \vdash \mathbf{M}:\tau$.

Gerum ráð fyrir að innsetningasafn $\mathfrak{S}$ svali Γ, og látum $\mathfrak{S}'$ vera það innsetningasafn sem fæst frá $\mathfrak{S}$ með því að fjarlægja úr því allar innsetningar fyrir $\mathbf{x}$. Við viljum sýna að

$$(\lambda \mathbf{x}.\, \mathbf{M})[\mathfrak{S}] = (\lambda \mathbf{x}.\, \mathbf{M}[\mathfrak{S}']) \in \chi((\sigma \to \tau)),$$

en til þess nægir að sýna að um sérhverja λ-segð $\mathbf{N}$ úr $\chi(\sigma)$ gildi

$$((\lambda \mathbf{x}.\, \mathbf{M}[\mathfrak{S}'])\ \mathbf{N}) \in \chi(\tau),$$

Gefum okkur slíka segð $\mathbf{N}$. Tökum eftir að gert er ráð fyrir að $(\lambda \mathbf{x}.\, \mathbf{M})[\mathfrak{S}]$ sé vel skilgreint, en þá gildir um sérhvert $(\mathbf{y}, \mathbf{P})$ í $\mathfrak{S}$ að $\mathbf{P}$ er innsetjanlegt fyrir $\mathbf{y}$ í $(\lambda \mathbf{x}.\, \mathbf{M})$, og því kemur $\mathbf{x}$ ekki fyrir frjáls í $\mathbf{P}$. Þar með höfum við

$$\mathbf{M}[\mathfrak{S}']_{\mathbf{x}}[\mathbf{N}] = \mathbf{M}[\mathfrak{S}' \cup \{(\mathbf{x}, \mathbf{N})\}]$$

Samkvæmt þrepunarforsendu höfum við nú að $\Gamma \cup \{\mathbf{x}:\sigma\}$ svalar $\mathbf{M}:\tau$, og ljóst er að $\mathfrak{S}' \cup \{(\mathbf{x}, \mathbf{N})\}$ svalar $\Gamma \cup \{\mathbf{x}:\sigma\}$. Við höfum því að $\mathfrak{S}' \cup \{(\mathbf{x}, \mathbf{N})\}$ svalar $\mathbf{M}:\tau$, eða með öðrum orðum

$$\mathbf{M}[\mathfrak{S}' \cup \{(\mathbf{x}, \mathbf{N})\}] \in \chi(\tau).$$

Nú er $\chi(\tau)$ mettað, svo af ofangreindu leiðir $((\lambda \mathbf{x}.\, \mathbf{M}[\mathfrak{S}'])\ \mathbf{N}) \in \chi(\tau)$.

(iii) Ef $\Gamma \vdash (\mathbf{M}\,\mathbf{N})\!:\!\tau$, þá er til tag σ þannig að $\Gamma \vdash \mathbf{M}\!:\!(\sigma \to \tau)$ og $\Gamma \vdash \mathbf{N}\!:\!\sigma$.

Gerum ráð fyrir að innsetningasafn $\mathfrak{S}$ svali Γ. Samkvæmt þrepunarforsendu höfum við þá að $\mathfrak{S}$ svalar báðum úthlutununum $\mathbf{M} : (\sigma \to \tau)$ og $\mathbf{N}\!:\!\sigma$. Við höfum þá

$$\mathbf{M}[\mathfrak{S}] \in \chi((\sigma \to \tau)) \quad \text{og} \quad \mathbf{N}[\mathfrak{S}] \in \chi(\sigma).$$

Þar með fæst

$$(\mathbf{M}\,\mathbf{N})[\mathfrak{S}] = (\mathbf{M}[\mathfrak{S}]\ \mathbf{N}[\mathfrak{S}]) \in \chi(\tau),$$

þ.e. $\mathfrak{S}$ svalar úthlutuninni $(\mathbf{M}\,\mathbf{N})\!:\!\tau$. □

Nú fæst stöðlunarsetningin sem fylgiseting hjálparsetninganna að ofan:

Setning 3.15 (*Stöðlunarsetningin*)**.** *Sérhver taganleg λ-segð er strangt staðlandi.*

Sönnun: Látum $\mathbf{M}$ vera λ-segð og $\mathbf{x}_1, \ldots, \mathbf{x}_n$ vera upptalningu frjálsra breyta í $\mathbf{M}$. Gerum ráð fyrir að

$$\mathbf{x}_1\!:\!\sigma_1, \ldots, \mathbf{x}_n\!:\!\sigma_n \vdash \mathbf{M}\!:\!\tau.$$

Þá gefur hjálparsetning 3.14 að samhengið $\Gamma \coloneqq \{\mathbf{x}_1 : \sigma_1, \ldots, \mathbf{x}_n : \sigma_n\}$ svalar úthlutuninni $\mathbf{M}\!:\!\tau$.

Við höfum nú að $\mathbf{x}_i \in \chi(\sigma_i)$ fyrir $i = 1, \ldots, n$, enda er $\chi(\sigma_i)$ mettað. Þar með er ljóst að tóma innsetningasafnið $\varnothing$ svalar samhenginu Γ, og þar með úthlutuninni $\mathbf{M} : \sigma$ Þetta gefur að $\mathbf{M} = \mathbf{M}[\varnothing] \in \chi(\tau)$, en þar sem $\chi(\tau)$ er mettað sýnir þetta jafnframt að $\mathbf{M}$ er strangt staðlandi. □

3.3 Höfuðtagsreikniritið

Við höfum hingað til séð hvernig nota má tög til þess að styrkja traust forritara til forrits síns. Hins vegar má ímynda sér að forrit sem nota tagskipanir verði lengri og flóknari, og að þau verði þess vegna bæði erfiðari að skrifa og erfiðari að lesa og skilja.

Hins vegar eru til aðferðir sem leyfa forritara að sleppa því að gera grein fyrir tagúthlutunum, en fá samt þá kosti sem tagskipting gefur. Þetta má til dæmis gera með ályktun tags, sbr. kafla 1.2. Einföld tagskipting hefur þann eiginleika að hægt er að álykta tag sérhverrar taganlegrar λ-segðar á endanlegum tíma. Þetta má gera með *höfuðtagsreikniritinu* svokallaða. Höfuðtagsreikniritið getur auk þess ákvarðað hvort λ-segð sé taganleg eða ekki.

Samkvæmt stöðlunarsetningunni gefur þetta okkur þar með einnig aðferð til þess að segja til um að λ-segð hafi pottþétt β-staðalform. Þetta er merkilegt af þeirri ástæðu að almenna verkefnið að ákvarða hvort λ-segð hafi β-staðalform, nefnist *stöðvunarverkefnið* (e. *halting problem*), og er vel þekkt að það sé óleysanlegt. Með höfuðtagsreikniritinu getum við hins vegar skoðað forrit áður en það er keyrt og gefið vottorð um að það muni stöðvast ef það er taganlegt.

Við setjum hér fram höfuðtagsreikniritið samkvæmt [8, Chapter 3]. Reikniritið finnur svokallað *höfuðtag* (e. *principal type*) segðar, en það er, lauslega skilið, alemennasta tag sem henni má gefa. Sérhvert tag sem gefa má segðinni má þannig hugsa um sem sértilvik af höfuðtagi þess, en í því felst að það megi mynda með innsetningu fyrir breytur höfuðtagsins. Áður en við skilgreinum nákvæmlega höfuðtag þurfum við því að skilgreina slíka innsetningu. Það má gera á svipaðan hátt og innsetningu í λ-segðir, en fyrir einföld tög er það þó nokkuð einfaldara, enda eru þar engar bundnar breytur.

Skilgreining 3.16. Látum $\mathbf{a}_1, \ldots, \mathbf{a}_n$ vera ólíkar tagbreytur og $\tau, \sigma_1, \ldots, \sigma_n$ vera einföld tög. Við táknum með $\tau_{\mathbf{a}_1,\ldots,\mathbf{a}_n}[\sigma_1, \ldots, \sigma_n]$ það einfalt tag sem fæst með því að setja tagið σ_i inn fyrir tagbreytuna $\mathbf{a}_i$ fyrir $i = 1, \ldots, n$ samtímis.

Fyrir samhengi $\Gamma = \{\mathbf{x}_1 : \tau_1, \ldots, \mathbf{x}_m : \tau_n\}$ ritum við einnig

$$\Gamma_{\mathbf{a}_1,\ldots,\mathbf{a}_n}[\sigma_1, \ldots, \sigma_n] \coloneqq \{\mathbf{x}_i : \tau_{i\mathbf{a}_1,\ldots,\mathbf{a}_n}[\sigma_1, \ldots, \sigma_n] \mid i = 1, \ldots, m\}.$$

Ef sett er $\mathfrak{S} \coloneqq \{(\mathbf{a}_1, \sigma_1), \ldots, (\mathbf{a}_n, \sigma_n)\}$ leyfum við okkur einnig að rita $\tau[\mathfrak{S}]$ og $\Gamma[\mathfrak{S}]$ í stað $\tau_{\mathbf{a}_1,\ldots,\mathbf{a}_n}[\sigma_1, \ldots, \sigma_n]$ og $\Gamma_{\mathbf{a}_1,\ldots,\mathbf{a}_n}[\sigma_1, \ldots, \sigma_n]$, og köllum þá $\mathfrak{S}$ *innsetningasafn*. Tvennd $(\mathbf{a}, \sigma)$ úr $\mathfrak{S}$ nefnist þá jafnframt *innsetning* tagsins σ *fyrir* $\mathbf{a}$.

Við munum auk þess seinna þurfa eftirfarandi ritháttt fyrir eitt innsetningasafn sem hefur sömu áhrif og tvö, beitt eitt á eftir öðru

Skilgreining 3.17. Látum $\mathfrak{S}_1, \mathfrak{S}_2$ vera innsetningasöfn, og látum $\mathfrak{S}_2'$ fást frá $\mathfrak{S}_2$ með því að fjarlægja úr því innsetningar fyrir breytur sem $\mathfrak{S}_1$ inniheldur einnig innsetningar fyrir. Við skilgreinum

$$\mathfrak{S}_1 \triangleright \mathfrak{S}_2 \coloneqq \{(\mathbf{a}, \sigma[\mathfrak{S}_2]) \mid (\mathbf{a}, \sigma) \in \mathfrak{S}_1\} \cup \mathfrak{S}_2'.$$

Setning 3.18. *Fyrir $\mathfrak{S}_1$, $\mathfrak{S}_2$ og τ gildir $\tau[\mathfrak{S}_1 \triangleright \mathfrak{S}_2] = \tau[\mathfrak{S}_1][\mathfrak{S}_2]$.*

Sönnun: Eftirlátin lesanda. $\square$

Nú skilgreinum við höfuðtag í samræmi við skilning þess að ofan:

Skilgreining 3.19. Látum $\mathbf{x}_1, \ldots, \mathbf{x}_n$ vera upptalningu á frjálsum breytum λ-segðar $\mathbf{M}$, og gerum ráð fyrir að eftirfarandi skilyrði séu uppfyllt:

(i) Við höfum $\mathbf{x}_1 : \sigma_1, \ldots, \mathbf{x}_n : \sigma_n \vdash \mathbf{M} : \tau$.

(ii) Ef $\mathbf{x}_1 : \sigma'_1, \ldots, \mathbf{x}_n : \sigma'_n \vdash \mathbf{M} : \tau'$, þá er til innsetningasafn $\mathfrak{S}$ þannig að $\sigma'_i = \sigma_i[\mathfrak{S}]$ fyrir öll $i = 1, \ldots, n$, og $\tau' = \tau[\mathfrak{S}]$.

Við segjum þá að tagið τ sé *höfuðtag* segðarinnar $\mathbf{M}$, og að parið (Γ, τ), með $\Gamma \coloneqq \{\mathbf{x}_1 : \sigma_1, \ldots, \mathbf{x}_n : \sigma_n\}$, sé *höfuðpar* hennar.

Dæmi 3.20. Í dæmi 3.6(a) sáum við að fyrir sérhvert einfalt tag σ má gefa segðinni $\mathbf{I} \coloneqq$ '$(\lambda x.\, x)$' tagið $(\sigma \to \sigma)$. Auk þess er ekki erfitt að sjá að tög á þessu formi eru einu tögin sem gefa má $\mathbf{I}$. Þar sem við höfum að $(\sigma \to \sigma) =$ '$(a \to a)$'$_{\text{'}a\text{'}}[\sigma]$ sýnir þetta að $\mathbf{I}$ hefur höfuðtag '$(a \to a)$' (og höfuðpar $(\varnothing,$ '$(a \to a)$'$)$).

Enn fremur sést að $\mathbf{I}$ hefur höfuðtag $(\mathbf{a} \to \mathbf{a})$, fyrir sérhverja tagbreytu $\mathbf{a}$. Þetta sýnir að höfuðtög eru ekki ótvírætt ákvörðuð. Hins vegar eru höfuðtög ótvíræð fram að breytuskiptum. Við sýnum ekki þá niðurstöðu í smáatriðum hér.

Áður en við setjum fram höfuðtagsreikniritið, þurfum við fyrst að skoða svokallaða *jafnara*, ásamt *jöfnunarreikniriti.* Höfuðtagsreikniritið mun nota jöfnunarreikniritið í framkvæmd þess.

Skilgreining 3.21. Sagt er að innsetningasafn $\mathfrak{S}$ sé *jafnari* taganna α og β ef $\alpha[\mathfrak{S}] = \beta[\mathfrak{S}]$. Tögin α og β eru þá sögð vera jafnanleg.

Enn fremur segjum við að $\mathfrak{S}$ sé *jafnari* runanna $\alpha_1, \ldots, \alpha_n$ og $\beta_1, \ldots, \beta_n$, ef $\mathfrak{S}$ er jafnari taganna α_i og β_i fyrir $i = 1, \ldots, n$.

Dæmi 3.22. Látum $\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}$ vera einhverjar tagbreytur.

(a) Tögin $\alpha \coloneqq (\mathbf{a} \to \mathbf{b})$ og $\beta \coloneqq (\mathbf{c} \to \mathbf{d})$ eru jafnanleg. Eftirfarandi eru dæmi um jafnara þeirra:

   (i) Ef $\mathfrak{S} = \{(\mathbf{c}, \mathbf{a}), (\mathbf{d}, \mathbf{b})\}$, þá er $\alpha[\mathfrak{S}] = \beta[\mathfrak{S}] = (\mathbf{a} \to \mathbf{b})$.

   (ii) Ef $\mathfrak{S} = \{(\mathbf{a}, \mathbf{c}), (\mathbf{b}, \mathbf{d})\}$, þá er $\alpha[\mathfrak{S}] = \beta[\mathfrak{S}] = (\mathbf{c} \to \mathbf{d})$.

   (iii) Ef $\mathfrak{S} = \{(\mathbf{b}, \mathbf{a}), (\mathbf{c}, \mathbf{a}), (\mathbf{d}, \mathbf{a})\}$, þá er $\alpha[\mathfrak{S}] = \beta[\mathfrak{S}] = (\mathbf{a} \to \mathbf{a})$.

   (iv) Ef $\mathfrak{S} = \{(\mathbf{a}, \sigma), (\mathbf{b}, \tau), (\mathbf{c}, \sigma), (\mathbf{d}, \tau)\}$, þá er $\alpha[\mathfrak{S}] = \beta[\mathfrak{S}] = (\sigma \to \tau)$.

(b) Tögin $\alpha \coloneqq (\mathbf{a} \to \mathbf{b} \to \mathbf{b})$ og $\beta \coloneqq ((\mathbf{c} \to \mathbf{c}) \to \mathbf{a})$ eru jafnanleg. Dæmi um jafnara er $\mathfrak{S} = \{(\mathbf{a}, (\mathbf{b} \to \mathbf{b})), (\mathbf{c}, \mathbf{b})\}$, en þá er

$$\alpha[\mathfrak{S}] = \beta[\mathfrak{S}] = ((\mathbf{b} \to \mathbf{b}) \to \mathbf{b} \to \mathbf{b}).$$

(c) Tögin $\alpha \coloneqq (\mathbf{a} \to \mathbf{b} \to \mathbf{c})$ og $\beta \coloneqq ((\mathbf{a} \to \mathbf{b}) \to \mathbf{c})$ eru ekki jafnanleg. Ef þau ættu sér jafnara $\mathfrak{S}$, þá hlyti að gilda

$$\mathbf{a}[\mathfrak{S}] = (\mathbf{a} \to \mathbf{b})[\mathfrak{S}] = (\mathbf{a}[\mathfrak{S}] \to \mathbf{b}[\mathfrak{S}]),$$

en það er fráleitt.

Frá lið (a) að ofan má álykta að sumir jafnarar eru, í vissum skilningi, almennari en aðrir. Til dæmis má segja að tagið $(\mathbf{a} \to \mathbf{b})$ sé almennara en $(\mathbf{a} \to \mathbf{a})$, enda fæst hið síðara frá því fyrra með innsetningi fyrir breytuna $\mathbf{b}$. Þetta er náskylt hugmyndinni um höfuðtög, og má setja þetta formlega fram í næstu skilgreiningu:

Skilgreining 3.23. Látum $\mathfrak{U}, \mathfrak{S}$ vera jafnara runanna $\alpha_1, \ldots, \alpha_n$ og $\beta_1, \ldots, \beta_n$. Við segjum að $\mathfrak{U}$ sé *almennari* en $\mathfrak{S}$ ef til er innsetningasafn $\mathfrak{S}'$ þannig að

$$\alpha_i[\mathfrak{S}] = \alpha_i[\mathfrak{U}][\mathfrak{S}'], \quad i = 1, \ldots, n.$$

Við segjum jafnframt að $\mathfrak{U}$ sé *almennur* jafnari runanna ef hann er almennari en sérhver annar jafnari runanna.

Nú má setja fram jöfnunarreikniritið. Eins og nafn þess bendir til, finnur það jafnara tveggja runa. Gott er að hafa eiginleika reikniritsins í huga á meðan á lestri þess stendur, svo við setjum fram eftirfarandi setningu um reikniritið áður en við skilgreinum það. Við munum ekki sanna þessa setningu hér.

Setning 3.24. *Að gefnum runum $\alpha_1, \ldots, \alpha_n$ og $\beta_1, \ldots, \beta_n$ í $\mathcal{S}$ skilar jöfnunarreikniritið almennan jafnara runanna ef jafnari þeirra er til, og $\perp$ ef ekki.*

Stakið $\perp$ (nefnt *botn*) er stundum notað í tölvunarfræði til þess að tákna óskilgreint gildi, og í rökfræði til þess að tákna mótsagnakennt fyrirbæri (sjá [1]). Við notum það í jöfnunar- og höfuðtagsreiknitinu til þess að tákna tilvistarleysi þess sem reynt er að finna.

Jöfnunarreikniritið hér á eftir er kennt við Robinson. Til eru hraðvirkari reiknirit til jöfnunar, en reiknirit Robinson hefur þann kost að vera mjög einfalt að setja fram. Útfærslu höfundar á jöfnunarreikniritinu í Haskell má finna í [4, `src/Language/SimpleTypes/Substitution.hs`, `unifier`]

Skilgreining 3.25 (*Jöfnunarreikniritið*)**.** Eftirfarandi reiknirit, nefnt *jöfnunarreikniritið*, fær sem inntak runur $\alpha_1, \ldots, \alpha_n$ og $\beta_1, \ldots, \beta_n$ af einföldum tögum og skilar annað hvort innsetningarsafni $\mathfrak{S}$ eða $\perp$. Við táknum útkomu reikniritsins með $\mathrm{U}(\alpha_1, \ldots, \alpha_n; \beta_1, \ldots, \beta_n)$.

I. Sett er $\mathfrak{S} \coloneqq \varnothing$.

II. Ef $\alpha_i[\mathfrak{S}] = \beta_i[\mathfrak{S}]$ fyrir öll $i = 1, \ldots, n$, þá er skilað $\mathfrak{S}$.

III. Annars er fundið minnsta i þ.a. $\alpha_i[\mathfrak{S}] \neq \beta_i[\mathfrak{S}]$, sett $\gamma_1 \coloneqq \alpha_i[\mathfrak{S}]$ og $\gamma_2 \coloneqq \beta_i[\mathfrak{S}]$, og framkæmt eftirfarandi skref:

   a. Ef γ_2 er tagbreyta, þá er sett $\mathbf{a} \coloneqq \gamma_2$ og $\sigma \coloneqq \gamma_1$ og farið áfram í IV.

   b. Ef γ_1 er tagbreyta, þá er sett $\mathbf{a} \coloneqq \gamma_1$ og $\sigma \coloneqq \gamma_2$ og farið áfram í IV.

   c. Annars höfum við $\gamma_1 = (\tau_1 \to \tau_1')$ og $\gamma_2 = (\tau_2 \to \tau_2')$.

      i. Ef $\tau_1 \neq \tau_2$, þá er sett $\gamma_1 \coloneqq \tau_1$ og $\gamma_2 \coloneqq \tau_2$ og farið aftur í IIIa.

      ii. Ef $\tau_1' \neq \tau_2'$, þá er sett $\gamma_1 \coloneqq \tau_1'$ og $\gamma_2 \coloneqq \tau_2'$ og farið aftur í IIIa.

IV. Ef tagbreytan $\mathbf{a}$ er kemur fyrir í taginu σ, þá er skilað $\perp$.

V. Annars er sett $\mathfrak{S} \coloneqq \mathfrak{S} \rhd \{(\mathbf{a}, \sigma)\}$, og farið í II.

Lauslega skilið, þá byggir jöfnunarreikniritið upp jafnara $\mathfrak{S}$ smátt og smátt með því að finna fyrstu misræmi í $\alpha_1[\mathfrak{S}], \ldots, \alpha_n[\mathfrak{S}]$ og $\beta_1[\mathfrak{S}], \ldots, \beta_n[\mathfrak{S}]$ og bæta við $\mathfrak{S}$ minnstu innsetningu sem jafnar þessi misræmi.

Hugsa má einnig að reikniritið setji inn fyrir tagbreytur $\alpha_1, \ldots, \alpha_n$ og $\beta_1, \ldots, \beta_n$ þangað til runurnar eru jafnar, enda hefur uppfærða innsetningasafnið í skrefi V sömu áhrif og að beita fyrst $\mathfrak{S}$, og setja svo σ inn fyrir $\mathbf{a}$.

Dæmi 3.26. Skoðum framkvæmd jöfnunarreikniritsins, með α og β úr 3.22.

(a) Ef $\alpha = (\mathbf{a} \to \mathbf{b})$ og $\beta = (\mathbf{c} \to \mathbf{d})$, þá setur jöfnunarreikniritið fyrst $\mathbf{a}' \coloneqq \mathbf{c}$ og $\sigma \coloneqq \mathbf{a}$, og setur $\mathfrak{S} = \{(\mathbf{a}', \sigma)\} = \{(\mathbf{c}, \mathbf{a})\}$.

   Því næst setur það $\mathbf{a}' \coloneqq \mathbf{d}$ og $\sigma \coloneqq \mathbf{b}$, og uppfærir $\mathfrak{S} \coloneqq \{(\mathbf{c}, \mathbf{a}), (\mathbf{d}, \mathbf{b})\}$.

   Að þessu loknu er $\alpha[\mathfrak{S}] = \beta[\mathfrak{S}]$, og reikiniritið skilar $\mathfrak{S}$.

(b) Ef $\alpha = (\mathbf{a} \to \mathbf{b} \to \mathbf{b})$ og $\beta = ((\mathbf{c} \to \mathbf{c}) \to \mathbf{a})$, þá setur jöfnunarreikniritið fyrst $\mathbf{a}' \coloneqq \mathbf{a}$ og $\sigma \coloneqq (\mathbf{c} \to \mathbf{c})$, og setur $\mathfrak{S} \coloneqq \{(\mathbf{a}, (\mathbf{c} \to \mathbf{c}))\}$.

   Því næst setur það $\mathbf{a}' \coloneqq \mathbf{c}$ og $\sigma \coloneqq \mathbf{b}$, og $\mathfrak{S} \coloneqq \{(\mathbf{a}, (\mathbf{b} \to \mathbf{b})), (\mathbf{c}, \mathbf{b})\}$.

   Að þessu loknu er $\alpha[\mathfrak{S}] = \beta[\mathfrak{S}]$, og reikiniritið skilar $\mathfrak{S}$.

(c) Ef $\alpha = (\mathbf{a} \to \mathbf{b} \to \mathbf{c})$ og $\beta = ((\mathbf{a} \to \mathbf{b}) \to \mathbf{c})$, þá setur jöfnunarreikniritið $\mathbf{a}' \coloneqq \mathbf{a}$ og $\sigma \coloneqq (\mathbf{a} \to \mathbf{b})$, og tekur eftir að þá er $\mathbf{a}'$ innihaldið í σ.

Reikniritið skilar því $\perp$.

Nú erum við tilbúin að setja fram höfuðtagsreikniritið. Aftur er gott að skoða eiginleika þess áður en reikniritið er sett fram. Nafngiftin bendir til að reikniritið gefi höfuðtag, en það skilar höfuðpari.

Setning 3.27. *Að gefinni λ-segð $\mathbf{M}$ skilar höfuðtagsreiknitið höfuðpari (Γ, τ) fyrir $\mathbf{M}$ ef $\mathbf{M}$ er taganleg, og $\perp$ ef hún er það ekki.*

Sönnun þessarar niðurstöðu, meðfram framsetningu reikniritsins, má finna í [8, Chapter 3E].

Útfærslu höfundar á reikniritinu í Haskell í samræmi við framsetningu þess að neðan er að finna í [4, `src/Language/SimpleTypes/Inference.hs`, `principalPair`].

Skilgreining 3.28 (*Höfuðtagsreikniritið*)**.** Eftirfarandi endurkvæma reiknirit, nefnt *höfuðtagsreikniritið*, fær sem inntak λ-segð $\mathbf{M}$ og skilar annað hvort pari (Γ, τ) af samhengi Γ og einföldu tagi τ, eða $\perp$. Við tákum útkomu reikniritisns með $\mathrm{PT}(\mathbf{M})$.

I. Ef $\mathbf{M}$ er breytusegð $\mathbf{x}$, þá er skilað $(\{\mathbf{x}\!:\!\mathbf{a}\}, \mathbf{a})$, þar sem $\mathbf{a}$ er tagbreyta.

II. Ef $\mathbf{M}$ er alhæfingarsegð $(\lambda \mathbf{x}.\, \mathbf{N})$, og $\mathrm{PT}(\mathbf{N}) = (\Gamma, \tau')$:

   a. Ef $\mathbf{x}$ er frjáls í $\mathbf{N}$ og $\mathbf{x}\!:\!\sigma \in \Gamma$, þá er skilað $(\Gamma \setminus \{\mathbf{x}\!:\!\sigma\}, (\sigma \to \tau'))$.

   b. Ef $\mathbf{x}$ er ekki frjáls í $\mathbf{N}$, þá er skilað $(\Gamma, (\mathbf{a} \to \tau'))$, þar sem $\mathbf{a}$ er einhver tagbreyta sem kemur hvergi fyrir í τ', né í úthlutunum Γ.

III. Ef $\mathbf{M}$ er beitingarsegð $(\mathbf{P}\,\mathbf{Q})$, og $\mathrm{PT}(\mathbf{P}) = (\Gamma_1, \tau_1)$ og $\mathrm{PT}(\mathbf{Q}) = (\Gamma_2, \tau_2)$:

   a. Fyrst er skipt út tagbreytum í Γ_2 og τ_2 þannig að engin tagbreyta komi fyrir bæði í Γ_1 eða τ_1, og í Γ_2 eða τ_2.

   b. Ef $\mathbf{x}_1 : \alpha_1, \ldots, \mathbf{x}_n : \alpha_n \in \Gamma_1$ og $\mathbf{x}_1 : \beta_1, \ldots, \mathbf{x}_n : \beta_n \in \Gamma_2$, þar sem $\mathbf{x}_1, \ldots, \mathbf{x}_n$ er upptalning allra breyta sem koma fyrir frjálst í báðum segðunum $\mathbf{P}$ og $\mathbf{Q}$, og

$$\mathrm{U}(\alpha_1, \ldots, \alpha_n, \tau_1; \beta_1, \ldots, \beta_n, (\tau_2 \to \mathbf{a})) = \mathfrak{S},$$

   með $\mathfrak{S} = \{(\mathbf{a}_1, \sigma_1), \ldots, (\mathbf{a}_n, \sigma_n)\}$:

i. Fyrst eru fjarlægt innsetningar $(\mathbf{a}_j, \sigma_j)$ úr $\mathfrak{S}$ sem uppfylla að $\mathbf{a}_j$ sé ekki innihaldin í neinu af $\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n, \tau_1, (\tau_2 \to \mathbf{a})$. Einnig er bætt við $\mathfrak{S}$ innsetningum $(\mathbf{b}, \mathbf{b})$ sem uppfylla að $\mathbf{b}$ er tagbreyta innihaldin í $\alpha_1, \ldots, \alpha_n, \beta_1, \ldots, \beta_n, \tau_1$ eða $(\tau_2 \to \mathbf{a})$, en er ekki eitt af $\mathbf{a}_1, \ldots, \mathbf{a}_m$.

ii. Skipt er út tagbreytum í $\sigma_1, \ldots, \sigma_m$ þannig að engin tagbreyta úr Γ_1, τ_1, Γ_2 eða τ_2 komi þar fyrir, nema ef til sé innsetning fyrir hana í $\mathfrak{S}$.

iii. Skilað er $(\Gamma_1[\mathfrak{S}] \cup \Gamma_2[\mathfrak{S}], \mathbf{a}[\mathfrak{S}])$.

IV. Annars er skilað $\perp$.

Hér ber að nefna að ofangreind lýsing á höfuðtagsreikniritinu er strangt til tekið ekki reiknirit, enda eru þau skref þar sem valin er ný tagbreyta ekki löggeng (því að ótakmarkaður fjöldi tagbreyta getur komið þar til greina). Hins vegar má laga þetta með því að gefa sér upptalningu á hinu teljanlega mengi tagbreyta og velja alltaf þá fyrstu sem uppfyllir nauðsynleg skilyrði.

4 Útvíkkun einfaldrar tagskiptingar

Við ljúkum þessu riti með því að ræða stuttlega um víðara efni hvað varðar tagskiptingu lambda-reiknings. Annars vegar ræðum við tagskiptingu með *fjölmóta tögum*, og hins vegar skoðum við *Hindley-Milner-tagskiptingu*. Ekki verður farið sérlega djúpt í smáatriði, en lesandi ætti að fá hygmynd um þau hugtök og niðurstöður sem liggja þar að grunni, ásamt kostum og ókostum þessara tagskiptinga yfir þá með einföldum tögum.

4.1 Fjölmóta tög

Þegar tagskiptingar eru notaðar í raun, þá er nauðsynlegt að gefa sér svokallaða *tagfasta*, en það eru grunntög sem hafa einhverja ákveðna merkingu. Í Haskell höfum við til dæmis tagfastana `Int` fyrir 32-bita heiltölur, `Float` fyrir 32-bita fleytitölur, og `String` fyrir strengi. Þessa tagfasta má nota á sama hátt og við höfum notað tagbreytur í einfaldri tagskiptingu. Þannig má í Haskell gefa fallinu `id`, sem er útfærsla á λ-segðinni $\mathbf{I}$ = '$(\lambda x.\, x)$', til dæmis tögin $(\texttt{Int} \to \texttt{Int})$ eða $((\texttt{String} \to \texttt{Float}) \to (\texttt{String} \to \texttt{Float}))$. Hins vegar myndi lítið þýða að gefa `id` tagið '$(a \to a)$', enda stendur tagbreytan 'a' ekki fyrir neitt tiltekið tag. Það sem við meinum þegar við gefum `id` tagið '$(a \to a)$' er að sérhvern tagfasta (eða tag sem inniheldur einungis tagfasta) má stinga inn fyrir 'a' og fá þannig gilt tag fyrir `id`.

Þessa hugmynd höfum við enga leið (nema á yfirmálinu) til þess að miðla fyrir einföld tög, en ekki er erfitt að útvíkka skilgreiningu þeirra svo að það sé hægt. Spyrjum Haskell hvaða tag það gefur `id`:

```
ghci> :type id
id :: forall a. a -> a
```

Hér höfum við táknið `forall` sem hluta af tagkerfi Haskell, sem þá þýðir einmitt það sem við viljum. Þetta er það sem er gert fyrir *fjölmóta tög* (e. *polymorphic types*), en þá er bætt magnaranum '$\forall$' við mál einfaldra taga:

Skilgreining 4.1. Formlegt mál $\mathcal{P}$ fyrir *fjölmóta tög* er skilgreint yfir stafrófið $\mathcal{V} \cup \{`\forall', `.', `\to', `(', `)'\}$, þar sem $\mathcal{V}$ er teljanlega óendanlegt mengi *tagbreyta*. Við köllum segðir þess (*fjölmóta*) *tög*, og eru þær gefnar með eftirfarandi myndunarreglum:

(i) Stæða af gerðinni $\mathbf{a}$ þar sem $\mathbf{a}$ er tagbreyta er fjölmóta tag.

(ii) Ef σ og τ eru fjölmóta tög, þá er $(\sigma \to \tau)$ fjölmóta tag

(iii) Ef σ er fjölmóta tag og $\mathbf{a}$ er tagbreyta, þá er $(\forall \mathbf{a}.\, \sigma)$ fjölmóta tag.

Segðir sem fást með myndunarreglu (ii) nefnast *vörpunartög* og segðir sem fást með myndunarreglu (iii) nefnast *allsherjartög*.

Fjölmóta tög eru nú einu skrefi líkari λ-segðum en einföld tög, en allsherjartög $(\forall \mathbf{a}.\, \sigma)$ líkjast að einhverju leiti alhæfingasegðum $(\lambda \mathbf{x}.\, \mathbf{M})$. Þessar λ-segðir voru ástæðan fyrir því að skilja þurfti á milli bundinna og frjálsra breyta, og svipað þarf þá nú að gera fyrir tagbreytur, enda er tagbreytunni $\mathbf{a}$ gefin einhver merking í taginu $(\forall \mathbf{a}.\, \sigma)$.

Við eftirlátum lesanda smáatriði þess að skilgreina bundnar og frjálsar tagbreytur, ásamt innsetjanlegum fjölmóta tögum, en þetta má gera á sama hátt og í skilgreiningu 2.6. Við gefum okkur einnig sama ritthátt fyrir innsetningu taga fyrir tagbreytur úr skilgreiningu 3.16, og gerum ráð fyrir að tag sé þá innsetjanlegt fyrir tilsvarandi tagbreytu.

Við getum einnig skilgreint hliðstæðu við α-umskipti og -jafngildi, enda hugsum við að tögin '$(\forall a.\, a)$' og '$(\forall b.\, b)$' þýði það sama.

Skilgreining 4.2. Við skilgreinum *úthlutanir* og *samhengi* fyrir fjölmóta tög á nákvæmlega sama hátt og við gerðum fyrir einföld tög í skilgreiningu 3.4.

Ef Γ er samhengi fyrir fjölmóta tög þá segjum við að Γ *leiði af sér* úthlutunina A og ritum $\Gamma \vdash_2 A$ ef þetta má leiða út með því að beita útleiðslureglum (i)–(iii) í skilgreiningu 3.4 (með $\vdash_2$ í stað $\vdash$), ásamt útleiðslureglum (iv) og (v) hér að neðan, endanlega oft.

(iv) Ef $\Gamma \vdash_2 \mathbf{M}:\sigma$ og $\mathbf{a}$ er tagbreyta sem kemur hvergi fyrir frjáls í úthlutunum Γ, þá $\Gamma \vdash_2 \mathbf{M}:(\forall \mathbf{a}.\, \sigma)$.

(v) Ef $\Gamma \vdash_2 \mathbf{M}:(\forall \mathbf{a}.\, \sigma)$ og τ er fjölmóta tag, þá $\Gamma \vdash_2 \mathbf{M}:\sigma_{\mathbf{a}}[\tau]$.

Ljóst er að sérhver λ-segð sem er taganleg með einföldu tagi er einnig taganleg með fjölmóta tagi, enda eru fjölmóta tög hrein útvíkkun á einföldum tögum. Fjölmóta tög eru einnig öflugri en einföld tög:

Dæmi 4.3. Gefum okkur breytu $\mathbf{x}$ og tagbreytu $\mathbf{a}$.

(a) Við höfum $\vdash_2 (\lambda \mathbf{x}.\, \mathbf{x}):(\mathbf{a} \to \mathbf{a})$ skv. dæmi 3.6(a), svo útleiðsluregla (iv) gefur $\vdash_2 (\lambda \mathbf{x}.\, \mathbf{x}):(\forall \mathbf{a}.\, (\mathbf{a} \to \mathbf{a}))$.

(b) Segðin $(\mathbf{x}\,\mathbf{x})$ er ekki taganleg með einföldu tagi, skv. dæmi 3.6(d).

Hins vegar höfum við $\mathbf{x}:(\forall \mathbf{a}.\, \mathbf{a}) \vdash_2 (\mathbf{x}\,\mathbf{x}):(\forall \mathbf{a}.\, \mathbf{a})$, enda gildir $\mathbf{x}:(\forall \mathbf{a}.\, \mathbf{a}) \vdash_2 \mathbf{x}:(\forall \mathbf{a}.\, \mathbf{a})$ skv. (i) og þá $\mathbf{x}:(\forall \mathbf{a}.\, \mathbf{a}) \vdash_2 \mathbf{x}:((\forall \mathbf{a}.\, \mathbf{a}) \to (\forall \mathbf{a}.\, \mathbf{a}))$ skv. (v), svo niðurstaðan fæst með (iii).

Enn fremur sést frá þessu að $\vdash_2 (\lambda \mathbf{x}.\, (\mathbf{x}\,\mathbf{x})):((\forall \mathbf{a}.\, \mathbf{a}) \to (\forall \mathbf{a}.\, \mathbf{a}))$ skv. (ii).

Þótt að fjölmóta tagskipting sé öflugri en einföld tagskipting, þá uppfyllir hún ennþá stöðlunarsetninguna, þ.e.a.s. λ-segð sem er taganleg með fjölmóta tagi hefur enga óendanlega einföldun (sjá [3, 4.3.7–4.3.11]). Hins vegar sakna fjölmóta tög ályktunareiginleika einfaldra taga, en ályktun fjölmóta tags er óákvarðanlegt verkefni. Sönnun þessarar niðurstöðu má finna í [15].

Setning 4.4. *Ekki er til reiknirit sem finnur ávallt fjölmóta tag fyrir óþekkta λ-segð í endanlega mörgum skrefum.*

Oft er því óhagkvæmt að nota fjölmóta tög í þessari mynd. Hins vegar eru ýmsar leiðir til að halda í helstu kosti fjölmóta taga án þess að lenda í þeirri gryfju að ályktun tags verði óákvarðanleg. Einni þeirri þekktustu þeirra er lýst hér á eftir.

4.2 Hindley-Milner-tagskipting

Hindley-Milner-tagskipting og tilsvarandi ályktunarreiknirit var þróuð í tengslum við forritunarmálið ML, sem er líkt Haskell á marga vegu. Hún er kennd við J. Roger Hindley og Robin Milner, ásamt Luis Damas, sem sannaði niðurstöður þeirra í doktorsritgerð sinni [7].

Tagskiptingin notfærir sér magnarann '$\forall$' úr fjölmóta tögum, en setur skorður á hvar hann má fara, nefnilega einungis yst í tagi. Til þess að fá aftur eitthvað af því afli sem við þetta tapast er lambda-reikningurinn sjálfur útvíkkaður með nýjum segðum: *látsegðum.* Þessar segðir eru oftar en ekki ritaðar ($\texttt{let}\ \mathbf{x} = \mathbf{N}\ \texttt{in}\ \mathbf{M}$), en við munum í staðinn nota rith//áttinn $(\mathbf{x} \leftarrow \mathbf{N};\, \mathbf{M})$.

Skilgreining 4.5. Formlegt mál λ_{HM} fyrir *Hindley-Milner-lambda-reikning* er skilgreint yfir stafrófið $\mathcal{V} \cup \{`\lambda', `.', `(', `)', `\leftarrow', `;'\}$, þar sem $\mathcal{V}$ er teljanlega óendanlegt mengi *breyta*. Segðir málsins eru gefnar með eftirfarandi myndunarreglum:

(i) Stæða af gerðinni $\mathbf{x}$ þar sem $\mathbf{x}$ er breyta er λ_{HM}-segð.

(ii) Ef $\mathbf{M}$ er λ_{HM}-segð og $\mathbf{x}$ er breyta, þá er $(\lambda\mathbf{x}.\,\mathbf{M})$ λ_{HM}-segð.

(iii) Ef $\mathbf{M}$ og $\mathbf{N}$ eru λ_{HM}-segðir, þá er $(\mathbf{M}\,\mathbf{N})$ λ_{HM}-segð.

(iv) Ef $\mathbf{M}$ og $\mathbf{N}$ eru λ_{HM}-segðir og $\mathbf{x}$ er breyta, þá er $(\mathbf{x} \leftarrow \mathbf{N};\, \mathbf{M})$ λ_{HM}-segð.

Segð sem fæst með beitingu myndunarreglu (ii) nefnist *alhæfingarsegð*, segð sem fæst með beitingu myndunarreglu (iii) nefnist *beitingarsegð* og segð sem fæst með myndunarreglu (iv) nefnist *látsegð*.

Látsegð $(\mathbf{x} \leftarrow \mathbf{N};\, \mathbf{M})$ ætti að túlka þannig að breytunni $\mathbf{x}$ sé gefið gildið $\mathbf{N}$ í segðinni $\mathbf{M}$. Þegar horft er framhjá tagskiptingum má þannig nota látsegð $(\mathbf{x} \leftarrow \mathbf{N};\, \mathbf{M})$ á nákvæmlega sama hátt og segðina $((\lambda\mathbf{x}.\,\mathbf{M})\,\mathbf{N})$. Látsegðum er því ofaukið nema að tagskipting gefi einhvern kost á að nota þær. Þetta er gert með *Hindley-Milner-tagskiptingu* sem notar *tagmynstur*:

Skilgreining 4.6. Formlegt mál $\mathcal{S}'$ fyrir *tagmynstur* er skilgreint yfir stafrófið $\mathcal{V} \cup \{`\forall', `.', `\to', `(', `)'\}$, þar sem $\mathcal{V}$ er teljanlega óendanlegt mengi *tagbreyta*. Við köllum segðir þess *tagmynstur*, og eru þær gefnar með eftirfarandi myndunarreglum:

(i) Ef σ er einfalt tag, þá er σ tagmynstur.

(ii) Ef σ er tagmynstur og $\mathbf{a}$ er tagbreyta, þá er $(\forall\mathbf{a}.\,\sigma)$ tagmynstur.

Við skilgreinum, eins og áður, frjálsar og bundnar tagbreytur, innsetjanleg tagmynstur og innsetningu þeirra fyrir tagbreytur. Þetta má auk þess gera fyrir λ_{HM}, en þá er breytan $\mathbf{x}$ í látsegð $(\mathbf{x} \leftarrow \mathbf{N};\, \mathbf{M})$ talin vera bundin.

Skilgreining 4.7. Við skilgreinum *úthlutanir* og *samhengi* fyrir tagmynstur á nákvæmlega sama hátt og við gerðum fyrir einföld tög í skilgreiningu 3.4.

Ef Γ er samhengi fyrir tagmynstur þá segjum við að Γ *leiði af sér* úthlutunina A og ritum $\Gamma \vdash_{\text{HM}} A$ ef þetta má leiða út með því að beita útleiðslureglum (i)–(iii) í skilgreiningu 3.4 eða útleiðslureglum (iv) og (v) í skilgreiningu 4.2, (með $\vdash_{\text{HM}}$ í stað $\vdash$ og $\vdash_2$, og tagmynstur í stað fjölmóta tags í (v)), ásamt útleiðslureglu (vi) hér að neðan, endanlega oft.

(vi) Ef $\Gamma \vdash_{\text{HM}} \mathbf{N}\colon\sigma$ og $\Gamma, \mathbf{x}\colon\sigma \vdash_{\text{HM}} \mathbf{M}\colon\tau$, þá $\Gamma \vdash_{\text{HM}} (\mathbf{x} \leftarrow \mathbf{N};\, \mathbf{M})\colon\tau$

Athugasemd 4.8. Við fyrstu sýn gæti þetta virst vera öflugri tagskipting en sú með fjölmóta tögum, og að hún lendi því í sama vanda með ályktun tagmynsturs. Hér ætti hins vegar að taka eftir að útgáfa útleiðslureglu (v) fyrir $\vdash_{\mathrm{HM}}$ notar tagmynstur í stað fjölmóta tags, og að því megi t.d. ekki leiða út að $\vdash_{\mathrm{HM}} (\mathbf{x}\,\mathbf{x}) : (\forall \mathbf{a}.\,\mathbf{a})$ eins og gert var í dæmi 4.3(b).

Dæmi 4.9. Skoðum segðirnar $\mathbf{M} := ((\lambda \mathbf{x}.\,(\mathbf{x}\,\mathbf{x}))\,\mathbf{I})$ og $\mathbf{M}' := (\mathbf{x} \leftarrow \mathbf{I};\,(\mathbf{x}\,\mathbf{x}))$. Þær gegna sama hlutverki, en $\mathbf{M}$ er ótaganleg í Hindley-Milner-tagskiptingu, á meðan $\mathbf{M}'$ er taganleg, enda má sýna

$$\mathbf{x} : (\forall \mathbf{a}.\,(\mathbf{a} \to \mathbf{a})) \vdash_{\mathrm{HM}} (\mathbf{x}\,\mathbf{x}) : (\forall \mathbf{a}.\,(\mathbf{a} \to \mathbf{a})),$$

sem ásamt $\vdash_{\mathrm{HM}} \mathbf{I} : (\forall \mathbf{a}.\,(\mathbf{a} \to \mathbf{a}))$ og reglu (vi) gefur $\vdash_{\mathrm{HM}} \mathbf{M}' : (\forall \mathbf{a}.\,(\mathbf{a} \to \mathbf{a}))$

Ekki er erfitt að sýna hins vegar að $\mathbf{M}$ er taganleg með fjölmóta tagi, sbr. dæmi 4.3. Sú takmörkun Hindley-Milner-tagskiptingar yfir þeirri með fjölmóta tögum felst þá í því að leyfa ekki að viðföng falla séu fjölmóta. Þegar þetta þarf má hins vegar nota látsegð til þess að tilgreina sérstaklega þær segðir sem gefa má tag með magnaranum '$\forall$'. Þessi takmörkun er því sjaldan stórmál fyrir forritara.

Nú kemur að helsta kosti Hindley-Milner-tagskiptingu yfir þeirri með fjölmóta tög, nefnilega ályktunareiginleika hennar. Ólíkt ályktun fjölmóta taga er ályktun tagmynsturs ákvarðanlegt verkefni. Til þess eru tvö reiknirit mest þekkt, nefnilega reiknirit J og W. Reiknirit J er oftast skilvirkara, en reikniriti W er auðveldara að lýsa. Við skoðum því einungis reiknirit W hér. Miðað er við framsetningu þess í [7, Chapter II, §4]. Reikniritið mun nota jöfnunarreikniritð í framkvæmd þess eins og höfuðtagsreikniritið. Lesandi má sjálfur sannfæra sig um að það sé ennþá vel skilgreint fyrir tagmynstur.

Skilgreining 4.10 (*Reiknirit W*)**.** Eftirfarandi endurkvæma reiknirit, nefnt *reiknirit W*, fær sem inntak samhengi Γ og λ_{HM}-segð $\mathbf{M}$, og skilar annað hvort pari $(\mathfrak{S}, \tau)$ af innsetningasafni $\mathfrak{S}$ og tagmynstri τ, eða $\bot$. Við táknum útkomu reikniritsins með $\mathrm{W}(\Gamma, \mathbf{M})$.

Látum fyrst $\mathbf{b}, \mathbf{b}_1, \ldots, \mathbf{b}_n$ vera ólíkar tagbreytur sem koma hvergi fyrir í Γ.

I. Ef $\mathbf{M}$ er breytusegð $\mathbf{x}$, og

$$\mathbf{x} : (\forall \mathbf{a}_1 \ldots \mathbf{a}_n.\,\tau') \in \Gamma,$$

þá er skilað $(\varnothing, \tau'_{\mathbf{a}_1,\ldots,\mathbf{a}_n}[\mathbf{b}_1, \ldots, \mathbf{b}_n])$.

II. Ef $\mathbf{M}$ er alhæfingarsegð $(\lambda \mathbf{x}.\,\mathbf{N})$, og

$$\mathrm{W}(\Gamma \cup \{\mathbf{x} : \mathbf{b}\}, \mathbf{N}) = (\mathfrak{S}', \tau'),$$

þá er skilað $(\mathfrak{S}', \mathbf{b}[\mathfrak{S}'] \to \tau')$.

III. Ef $\mathbf{M}$ er beitingarsegð $(\mathbf{P}\,\mathbf{Q})$, og

$$\begin{aligned} & \mathrm{W}(\Gamma, \mathbf{P}) = (\mathfrak{S}_1, \tau_1), \\ & \mathrm{W}(\Gamma[\mathfrak{S}_1], \mathbf{Q}) = (\mathfrak{S}_2, \tau_2) \\ \text{og} \quad & \mathrm{U}(\tau_1[\mathfrak{S}_2]; (\tau_2 \to \mathbf{b})) = \mathfrak{U} \neq \perp, \end{aligned}$$

þá er skilað $(\mathfrak{S}_1 \rhd \mathfrak{S}_2 \rhd \mathfrak{U}, \mathbf{b}[\mathfrak{U}])$.

IV. Ef $\mathbf{M}$ er látsegð $(\mathbf{x} \leftarrow \mathbf{Q};\, \mathbf{P})$, og

$$\begin{aligned} & \mathrm{W}(\Gamma, \mathbf{Q}) = (\mathfrak{S}_1, \tau_1) \\ \text{og} \quad & \mathrm{W}(\Gamma \cup \{\mathbf{x} : \forall \mathbf{a}_1 \ldots \mathbf{a}_n .\, \tau_1\}, \mathbf{Q}) = (\mathfrak{S}_2, \tau_2), \end{aligned}$$

þar sem $\mathbf{a}_1, \ldots, \mathbf{a}_n$ er upptalning þeirra tagbreyta sem koma fyrir frjálst í τ en ekki í Γ, þá er skilað $(\mathfrak{S}_1 \rhd \mathfrak{S}_2, \tau_2)$.

V. Annars er skilað $\perp$.

Eins og fyrir höfuðtagsreikniritið má tilgreina hvernig velja skal tagbreyturnar $\mathbf{b}, \mathbf{b}_1, \ldots, \mathbf{b}_n$ með því að gefa sér upptalningu á mengi allra tagbreyta.

Setning 4.11. *Að gefnu samhengi Γ og λ_{HM}-segð $\mathbf{M}$ skilar reiknirit W pari $(\mathfrak{S}, \tau)$ sem uppfyllir $\Gamma[\mathfrak{S}] \vdash_{\mathrm{HM}} \mathbf{M} : \tau$ ef $\mathbf{M}$ er taganleg með tagmynstri, og $\perp$ ef hún er það ekki.*

Jafnframt má sýna að reiknirit W gefur ávallt „almennasta“ tagmynstur segðar, hliðstætt við höfuðtagsreikniritið. Þetta er gert í [7, Chapter II, §5].

Heimildir

[1] Arnar Ágúst Kristjánsson, *Curry-Howard samsvörunin*, Samæfingar í stærðfræði, (2023).

[2] H. P. Barendregt, *The Lambda Calculus: Its Syntax and Semantics*, North-Holland, 1984.

[3] ——, *Lambda calculi with types*, in Handbook of Logic in Computer Science (Vol. 2): Background: Computational Structures, Oxford University Press, Inc., USA, 1993, p. 117–309.

[4] Bjarki Baldursson Harksen, *Höfuðtagsreikniritið*. `https://github.com/mrbjarksen/principal-type`, 2023.

[5] CWoo, *Ackermann function is not primitive recursive.* `https://planetmath.org/ackermannfunctionisnotprimitiverecursive`, 2013. Sótt 7. mars 2023.

[6] ——, *Ackermann function is total recursive.* `https://planetmath.org/ackermannfunctionistotalrecursive`, 2013. Sótt 7. mars 2023.

[7] L. Damas, *Type assignment in programming languages*, PhD thesis, The University of Edinburgh, 1984.

[8] J. R. Hindley, *Basic Simple Type Theory*, Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, 1997.

[9] J. R. Hindley and J. P. Seldin, *Lambda-Calculus and Combinators: An Introduction*, Cambridge University Press, 2 ed., 2008.

[10] S. C. Kleene, *λ-definability and recursiveness*, Duke Mathematical Journal, 2 (1936), pp. 340 – 353.

[11] Y. Komori, N. Matsuda, and F. Yamakawa, *A simplified proof of the Church-Rosser theorem*, Studia Logica: An International Journal for Symbolic Logic, 102 (2014), pp. 175–183.

[12] Reynir Axelsson, *Inngangur að stærðfræðilegri rökfræði*, Raunvísindadeild, Háskóli Íslands, 2020.

[13] P. Selinger, *Lecture notes on the lambda calculus*, 2013.

[14] Stack Overflow, *Stack Overflow developer survey 2022.* `https://survey.stackoverflow.co/2022/`, 2022. Sótt 19. mars 2023.

[15] J. Wells, *Typability and type checking in System F are equivalent and undecidable*, Annals of Pure and Applied Logic, 98 (1999), pp. 111–156.

[16] Wikibooks, *Fix and recursion.* `https://en.wikibooks.org/wiki/Haskell/Fix_and_recursion`, 2022. Sótt 7. mars 2023.