Alityypityksen metateoria

Risto Niskakangas

Helsinki, 27. helmikuuta 2003
Tyyppiteoria ja ohjelmointikielet - seminaari
HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Sisältö

1 Johdanto

Tässä tekstissä käsittelen sitä kuinka aikaisemmin esitetty lambdakalkyyli alityypityksellä saadaan sopivaksi toteutukselle. Teksti ja esimerkit noudattavat tarkasti tässä seminaarissa käytettyä B.C. Piercen kirjan lukua 16 vain hieman tiivistettynä [Pie02]. Luvun 17 ML-kielistä alityypityksen toteutusta ei käydä läpi.

Aikaisemmista kalkyyleista poiketen tämän järjestelmä säännöt eivät ole *syntaksisesti ohjattuja* (syntax directed) eli niitä ei voi suoraan lukea alhaalta ylös tyypintarkistusalgoritmin muodostamiseksi. Suurimpina esteinä ovat säännöt tyypitysrelaation aliluokituksessa (T-SUB) ja transitiivisuus (S-TRANS) alityypitysrelaatiossa. T-SUB:ssa päätelmän termi on määritelty pelkkänä metamuuttujana t:

$$\frac{\Gamma \vdash t : S \qquad S <: T}{\Gamma \vdash t : T} \qquad \text{(T-SUB)}$$

Muut tyypityssäännöt määrittelevät tiettyä muotoa olevan termin kun taas T-SUB voidaan soveltaa kaikenmuotoisille termeille. Termin t tyyppiä laskettaessa on siis aina mahdollista soveltaa joko T-SUB:ia tai toista sääntöä, jonka päätelmä vastaa t:n muotoa.

S-TRANS:in kohdalla ongelma on sama kuin T-SUB:in eli sen johtopäätös menee päällekkäin toisten sääntöjen johtopäätösten kanssa.

$$\frac{S <: U \qquad U <: T}{S <: T} \qquad \text{(S-TRANS)}$$

S:n ja T:n ollessa pelkkiä metamuuttujia on mahdollista käyttää S-TRANS:ia päätössääntönä minkä tahansa alityypityslauseen päättelyssä. Siksi alityypistyssääntöjen alhaalta ylös toteutus ei tiedä käyttääkö ko. sääntöä vai jotain toista sääntöä, jonka tarkempi päätelmä myös vastaa kahta tyyppiä, joiden riippuvuuksia alityypitysrelaatiossa yritämme selvittää. Ongelmaksi muodostuu myös premisseissä mainittu metamuuttuja U, joka ei esiinny johtopäätöksessä. Alhaalta ylös luettaessa tulisi arvata U:n tyyppi osoittaessa että S<:U ja U<:T, mikä ei onnistu koska U:ita on ääretön määrä. Myös S-REFL-sääntö mene päällekkäin toisten alityypityssääntöjen päätelmien kanssa. S-REFL:lla ei kuitenkaan ole premissejä, joten tämä ei tuota ongelmia.

Ratkaisuna em. ongelmiin on korvata alkuperäinen alityypitys ja tyypitysrelaatiot *algoritmisella alityypityksellä* ja *algoritmisilla tyypitysrelaatioilla*, joiden päätelmäsäännöt ovat syntakstisesti ohjattuja. Tämä vaihdos oikeutetaan osoittamalla että alkuperäinen alityypitys ja tyypitysrelaatiot ovat yhtäpitäviä vastaavien algoritmisten esitysten kanssa.

2 Algoritminen alityypitys

Alityypityksen sisältävän kielen toteutuksen keskeisimpänä osana on alityypityksen tarkistava algoritmi. Tyypintarkistaja kutsuu tätä alityypin tarkistajaa kun kohdataan esim. sovellus t_1 t_2 jossa t_1 on tyyppiä $T \rightarrow U$ ja t_2 on tyyppiä S. Tehtävänä on silloin päättää onko $S <: T$ johdettavissa alityypityssäännöistä. Tyypintarkistaja tarkistaa kuuluuko (S,T) toiseen relaatioon, merkittynä $\vdash\triangleright S <: T$,mikä on määritelty siten että jäsenyys voidaan päättää seuraamalla tyyppien rakenteita ja niitä jotka sisältävät samat tyyppiparit alkuperäisenä alityyppirelaationa. Pääerona kuvaavalla (*declarative*) ja algoritmisella relaatiolla on se että algoritminen relaatio pudottaa S-TRANS ja S-REFL säännöt.

S-TRANS-säännön pudottamiseksi täytyy lisätä sääntö joka niputtaa yhteen syvyys-, leveys- ja permutaatioalityypityksen:

$$\frac{\{l_i\ ^{i\in 1..n}\} \subseteq \{k_j\ ^{j\in 1..m}\} \qquad k_j = l_i\ implies\ S_j <: T_i}{\{k_j : S_j\ ^{j\in 1..m}\} <: \{l_i : T_i\ ^{i\in 1..n}\}} \qquad \text{(S-RCD)}$$

LEMMA 1: Jos $S <: T$ on johdettavissa käyttäen alityypityssääntöjä S-RCDDEPTH, S-RCDWIDTH ja S-RCDPERM, (mutta ei sääntöä S-RCD), silloin $S <: T$ on johdettavissa myös käyttäen sääntöä S-RCD (mutta ei sääntöjä S-RCDDEPTH, S-RCDWIDTH ja S-RCDPERM) ja päinvastoin.

Lemma 1 oikeuttaa sääntöjen S-RCDDEPTH, S-RCDWIDTH ja S-RCDPERM eliminoinnit S-RCD:n eduksi.

LEMMA 2:

1. $S <: S$ voidaan johtaa jokaiselle tyypille S ilman S-REFL:ia.

2. Jos $S <: T$ on johdettavissa, niin se voidaan johtaa ilman S-TRANS:ia

MÄÄRITELMÄ 1: *Algoritminen alityypitysrelaatio* on vähäisin relaatio joka asettuu seuraavien sääntöjen alle

Algoritminen alityypitys $\boxed{\vdash\triangleright S <: T}$

$$\vdash\triangleright S <: T \qquad \text{(SA-TOP)}$$

$$\frac{\{l_i\ ^{i\in 1..n}\} \subseteq \{k_j\ ^{j\in 1..m}\} \qquad if\ k_j = l_i\ , then\ \vdash\triangleright S_j <: T_i}{\vdash\triangleright \{k_j : S_j\ ^{j\in 1..m}\} <: \{l_i : T_i\ ^{i\in 1..n}\}} \qquad \text{(SA-RCD)}$$

$$\frac{\vdash\!\triangleright\, T_1 <: S_1 \qquad \vdash\!\triangleright\, S_2 <: T_2}{\vdash\!\triangleright\, S_1 \to S_2 <: T_1 \to T_2} \qquad \text{(SA-ARROW)}$$

Voidaan sanoa että algoritmiset säännöt ovat 1. *eheitä*, koska jokainen algoritmisista säännöistä johdettavissa oleva lause voidaan johtaa myös kuvaavista säännöistä ja 2. *täydellisiä*, koska jokainen kuvaavista säännöistä johdettavissa oleva lause voidaan johtaa myös algoritmisista säännöistä.

VÄITE [EHEYS JA TÄYDELLISYYS]: $\mathrm{S} <: \mathrm{T}$ joss $\vdash\!\triangleright\, \mathrm{S} <: \mathrm{T}$.

Syntaksiohjatut algoritmiset säännöt voidaan lukea suoraan algoritmisen alityyppirelaation tarkistavana algoritmina. Seuraavassa algoritmi pseudokoodina:

$$\begin{aligned}
\mathit{subtype}(\mathtt{S},\mathtt{T}) = {} & \mathtt{if}\ \ \mathtt{T} = \mathtt{Top}, \mathtt{then}\ \ \mathit{true} \\
& \mathtt{else\ if}\ \ \mathtt{S} = \mathtt{S_1} \to \mathtt{S_2}\ \ \mathtt{and}\ \ \mathtt{T} = \mathtt{T_1} \to \mathtt{T_2} \\
& \quad \mathtt{then}\ \ \mathit{subtype}(\mathtt{T_1},\mathtt{S_1}) \wedge \mathit{subtype}(\mathtt{S_2},\mathtt{T_2}) \\
& \mathtt{else\ if S} = \{\mathtt{k_j} : \mathtt{s_j}\ ^{\mathtt{j}\in\mathtt{1..m}}\}\ \ \mathtt{and}\ \ \mathtt{T} = \{\mathtt{l_i} : \mathtt{T_i}\ ^{\mathtt{i}\in\mathtt{1..n}}\} \\
& \quad \mathtt{then}\ \ \{\mathtt{l_i}\ ^{\mathtt{i}\in\mathtt{1..n}}\} \subseteq \{\mathtt{k_j}\ ^{\mathtt{j}\in\mathtt{1..m}}\} \\
& \qquad \wedge\ \ \mathtt{for\ all}\ \ i\ \ \mathtt{there\ is\ some}\ \ j \in \mathtt{1..m}\ \ \mathtt{with}\ \ \mathtt{k_j} = \mathtt{l_i} \\
& \qquad\quad \mathtt{and}\ \ \mathit{subtype}(\mathtt{S_j},\mathtt{T_i}) \\
& \mathtt{else}\ \ \mathit{false}.
\end{aligned}$$

Vielä täytyy varmistaa että algoritminen alityyppirelaatio on totaalinen eli ts. algoritmisista säännöistä johdettu rekursiivinen funktio *subtype* palauttaa joko *true* tai *false* kaikille syötepareille äärellisessä ajassa.

VÄITE [PYSÄHTYVYYS]: Jos $\vdash\!\triangleright\, \mathrm{S} <: \mathrm{T}$. on pääteltävissä, niin *subtype*(S,T) palauttaa *true*. Jos ei ole johdettavissa, niin *subtype*(S,T) palauttaa *false*.

Tämä vakuuttaa yhdessä eheyden ja täydellisyyden kanssa että *subtype*-funktio on *päätösmenetelmä* kuvaavalle alityyppirelaatiolle.

3 Algoritminen tyypitys

Alityyppirelaation tapaan pitää myös tyypitysrelaatio saada hallintaan. Ainoa ei-syntaksisesti ohjattu tyypityssääntö on T-SUB. Kuten S-TRANS:n kanssa edellisessä kappaleessa, ei T-SUB:ia voida noin vain poistaa. Täytyy tutkia missä päin tyypitystä se on ratkaisesvassa osassa ja rikastaa muita tyypityssääntöjä saman mutta syntaksiohjatumman vaikutuksen aikaansaamiseksi.

Ratkaisevana ja itse asiassa ainoana tilanteena, jossa sisällytyksellä (*subsumption*) on ensiarvoinen rooli, on funktioiden olettamien ja sen saamien argumenttien tyyppien välisen kuilun kaventaminen. Kaikissa muissa sisällytystä käyttävissä tyypityksen todistuksissa, sama lause voidaan todistaa eri päättelyllä, jossa sisällytystä lykätään siirtämällä sitä kohti todistuspuun juurta. Seuraavassa kokeillaan erilaisilla puilla, joissa johtamisen lopetusta voidaan uudelleenjärjestää kun välitön alijohtopäätös loppuu T-SUB:lla.

Esimerkki 1. Tyypityspäättely päättyy T-ABS:lla, jonka edeltävä johtopäätös päättyy T-SUB:lla.

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma, x : S_1 \vdash s_2 : S_2} \quad \dfrac{\vdots}{S_2 <: T_2}}{\Gamma, x : S_1 \vdash s_2 : T_2}\,\text{(T-SUB)}}{\Gamma \vdash \lambda x : S_1 . s_2 : S_1 \to T_2}\,\text{(T-ABS)}$$

Tälläinen päättely voidaan uudelleenjärjestää siten että T-SUB:ia käytetään vasta T-ABS:n jälkeen:

$$\dfrac{\dfrac{\dfrac{\vdots}{S_2 <: T_2}}{\Gamma \vdash \lambda x : S_1 . s_2 : S_1 \to S_2}\,\text{(T-ABS)} \quad \dfrac{\dfrac{}{S_1 <: S_1}\,\text{(S-REFL)} \quad \dfrac{\vdots}{S_2 <: T_2}}{S_1 \to S_2 <: S_1 \to T_2}\,\text{(S-ARROW)}}{\Gamma \vdash \lambda x : S_1 . s_2 : S_1 \to T_2}\,\text{(T-SUB)}$$

Esimerkki 2. T-APP:n kohdalla voi kumpi tahansa välittömistä alijohtopäätöksistä päättyä T-SUB:lla. Otetaan tapaus, jossa T-SUB on vasemmalla puolen.

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash s_1 : S_{11} \to S_{12}} \quad \dfrac{\vdots}{S_{11} \to S_{12} <: T_{11} \to T_{12}}\,\text{(S-ARROW)}}{\Gamma \vdash s_1 : T_{11} \to T_{12}}\,\text{(T-SUB)} \quad \dfrac{\vdots}{\Gamma \vdash s_2 : T_{11}}}{\Gamma \vdash s_1\, s_2 : T_{12}}\,\text{(T-APP)}$$

Aikaisempien tulosten perusteella voidaan olettaa että johtopäätöksen $S_{11} \to S_{12} <: T_{11} \to T_{12}$ päättelyssä käytetty sääntö voi olla vain S-ARROW, jolloin puusta tulee seuraavan näköinen.

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash s_1 : S_{11} \to S_{12}} \quad \dfrac{\dfrac{\vdots}{T_{11} <: S_{11}} \quad \dfrac{\vdots}{S_{12} <: T_{12}}}{S_{11} \to S_{12} <: T_{11} \to T_{12}}\text{(S-ARROW)}}{\Gamma \vdash s_1 : T_{11} \to T_{12}}\text{(T-SUB)} \quad \dfrac{\vdots}{\Gamma \vdash s_2 : T_{11}}}{\Gamma \vdash s_1\, s_2 : T_{12}}\text{(T-APP)}$$

Uudelleenkirjoitetaan puu T-SUB:n ilmentymän poistamiseksi...

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash s_1 : S_{11} \to S_{12}} \quad \dfrac{\dfrac{\vdots}{\Gamma \vdash s_2 : T_{11}} \quad \dfrac{\vdots}{\Gamma \vdash T_{11} <: S_{11}}}{\Gamma \vdash s_2 : S_{11}}\text{(T-SUB)}}{\Gamma \vdash s_1\, s_2 : S_{12}}\text{(T-APP)} \quad \dfrac{\vdots}{S_{12} <: T_{12}}}{\Gamma \vdash s_1\, s_2 : T_{12}}\text{(T-SUB)}$$

Alkuperäisen S-ARROW:n ilmentymän oikeanpuoleinen alipäätelmä on työnnetty puun pohjalle, jossa uusi T-SUB:n ilmentymä korottaa koko sovellussolmun tyyppiä. Toisaalta, vasemmanpuoleinen alipäättely on työnnetty ylös argumentin s_2 päättelyyn.

Oletetaan että siirrettävä T-SUB:n ilmentymä onkin T-APP:n oikeanpuoleisen alipäätelmän viimeisenä päätelmänä.

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash s_1 : T_{11} \to T_{12}} \quad \dfrac{\dfrac{\vdots}{\Gamma \vdash s_2 : T_2} \quad \dfrac{\vdots}{T_2 <: T_{11}}}{\Gamma \vdash s_2 : T_{11}}\text{(T-SUB)}}{\Gamma \vdash s_1\, s_2 : T_{12}}\text{(T-APP)}$$

T-SUB voidaan ainoastaan siirtää vasemmanpuoleiseksi alipäätelmäksi.

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash s_1 : T_{11} \to T_{12}} \quad \dfrac{\dfrac{\dfrac{\vdots}{T_2 <: T_{11}} \quad \dfrac{}{T_{12} <: T_{12}}\text{(S-REFL)}}{T_{11} \to T_{12} <: T_2 \to T_{12}}\text{(S-ARROW)}}{\Gamma \vdash s_1 : T_2 \to T_{12}}\text{(T-SUB)} \quad \dfrac{\vdots}{\Gamma \vdash s_2 : T_2}}{\Gamma \vdash s_1\, s_2 : T_{12}}\text{(T-APP)}$$

Aliluokituksen käyttämistä sovelluksen tulostyypin korottamiseksi voidaan siis siirtää ohi T-APP:n kohti juurta, mutta argumentin tyypin sovittamisessa funktion tyyppialaan T-SUB:n käyttöä ei voi poistaa.

Oletetaan että viimeisenä päättelysääntönä on T-SUB ja myös välitön alipäättelysääntö on T-SUB. Tällöin nämä lähekkäiset aliluokituksen käytöt voidaan sulauttaa yhdeksi eli mikä tahansa muotoa:

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash s : S} \quad \dfrac{\vdots}{S <: U}}{\Gamma \vdash s : U}\,\text{(T-SUB)} \quad \dfrac{\vdots}{U <: T}}{\Gamma \vdash s : T}\,\text{(T-SUB)}$$

oleva päätelmä voidaan uudelleenkirjoittaa muotoon:

$$\dfrac{\dfrac{\vdots}{\Gamma \vdash s : S} \quad \dfrac{\dfrac{\vdots}{S <: U} \quad \dfrac{\vdots}{U <: T}}{S <: T}\,\text{(S-TRANS)}}{\Gamma \vdash s : T}\,\text{(T-SUB)}$$

Näitä muunnoksia toistuvasti soveltamalla voidaan mielivaltainen tyypityspäätelmä uudelleenkirjoittaa muotoon, jossa T-SUB esiintyy vain oikeanpuoleisen alipäättelyn lopussa ja koko päättelyn lopussa. Lisäksi jos poistamme lopussa olevan päättelyn niin jäljelle jää siltikin samalle termille tyypin antava päättely, jossa erona on vain tyypin pienempi koko. Tästä syystä aliluokituksen käytölle jää ainoastaan paikka sovelluksessa. Sen käsittelemiseksi voidaan sovellussääntö korvata hieman tehokkaammalla sisällyttäen premissehin yhden aliluokituksen ilmentymän:

$$\dfrac{\Gamma \vdash t_1 : T_{11} \to T_{12} \qquad \Gamma \vdash t_2 : T_2 \qquad T_2 <: T_{11}}{\Gamma \vdash t_1\ t_2 : T_{12}}$$

Kaikki muotoa "sisällytys edeltää sovellusta"alipäättelyt voidaan korvata tällä säännöllä, jolloin T-SUB:ia ei tarvitse käyttää ollenkaan. Lisäksi rikastettu sääntö on syntaksiohjatumpi: johtopäätöksen termin muoto estää sääntöä menemästä päällekkäin toisten sääntöjen kanssa.

MÄÄRITELMÄ 2: *Algoritminen tyypitysrelaatio* on vähin relaatio, joka asettuu seuraavien sääntöjen alle:

Algoritminen tyypitys $\boxed{\Gamma \vdash\!\triangleright\, S <: T}$

$$\frac{x : T \in \Gamma}{\Gamma \vdash\rhd x : T} \quad \text{(TA-VAR)}$$

$$\frac{\Gamma, x : T_1 \vdash\rhd t_2 : T_2}{\Gamma \vdash\rhd \lambda x : T_1.t_2 : T_1 \to T_2} \quad \text{(TA-ABS)}$$

$$\frac{\Gamma \vdash\rhd t_1 : T_1 \qquad \Gamma \vdash\rhd t_2 : T_2 \qquad T_1 = T_{11} \to T_{12} \qquad \vdash\rhd T_2 <: T_{11}}{\Gamma \vdash\rhd t_1\, t_2 : T_{12}} \quad \text{(TA-APP)}$$

$$\frac{\text{for each } i\ \Gamma \vdash\rhd \mathrm{t_i} : \mathrm{T_i}}{\Gamma \vdash\rhd \{l_1 = t_1 ... l_n = t_n\} : \{l_1 : T_1 ... l_n : T_n\}} \quad \text{(TA-RCD)}$$

$$\frac{\Gamma \vdash\rhd t_1 : R_1 \qquad R_1 = \{l_1 : T_1 ... l_n : T_n\}}{\Gamma \vdash\rhd t_1.l_i : T_i} \quad \text{(TA-PROJ)}$$

Jäljellä on vielä algoritmisten tyypityssääntöjen vastaavuuden tarkistaminen alkuperäisten kuvaavien sääntöjen kanssa. Algoritmisen tyypitysrelaation tulee siis olla eheä ja täydellinen alkuperäisten sääntöjen valossa.

TEOREEMA [EHEYS]: Jos $\Gamma \vdash\rhd \mathrm{t} : \mathrm{T}$, niin $\Gamma \vdash\rhd \mathrm{t} : \mathrm{T}$.

Todistus: Suora induktio algoritmisille tyypityspäätelmille.

TEOREEMA [TÄYDELLISYYS, TAI MINIMAALINEN TYYPITYS]: Jos $\Gamma \vdash \mathrm{t} : \mathrm{T}$, niin $\Gamma \vdash\rhd \mathrm{t} : \mathrm{S}$ jollekin $\mathrm{S} <: \mathrm{T}$.

4 Liitokset ja katteet (Joins And Meets)

Tyypintarkistuslauseet joilla on useita tuloshaaroja vaativat alityypityksellisessä kielessä lisää huomiota. Käytetään esimerkkinä kuvaavaa ehdollisuuden tyypityssääntöä.

$$\frac{\Gamma \vdash t_1 : Bool \qquad \Gamma \vdash t_2 : T \qquad \Gamma \vdash t_3 : T}{\Gamma \vdash if\ t_1\ then\ t_2\ else\ t_3 : T} \quad \text{(T-IF)}$$

Sääntö vaatii että kahden haaran tyypit ovat samoja ja välittää saman tyypin koko ehdollisuudelle. Kuitenkin alityypiksen yhteydessä voi olla useita tapoja antaa kahdelle haaralle sama tyyppi. Yleisesti ottaen mielivaltaisen ehdollisen lauseen minimaalisen tyypin laskemiseksi täytyy ensin laskea **then** ja **else** haarojen tyypit ja vielä laskea saatujen tyyppien epätavanomaisin supertyyppi, jota usein kutsutaan haarojen tyyppien liitokseksi.

MÄÄRITELMÄ: Tyyppiä J kutsutaan tyyppien S ja T *liitokseksi*, merkittynä $\mathrm{S} \vee \mathrm{T} = \mathrm{J}$, jos $\mathrm{S} <: \mathrm{J}$, $\mathrm{T} <: \mathrm{J}$ ja kaikille tyypeille U, jos $\mathrm{S} <: \mathrm{U}$ ja $\mathrm{T} <: \mathrm{U}$ niin $\mathrm{J} <: \mathrm{U}$. Samoiten tyyppi

M on tyyppien S ja T *kate*, merkittynä S $\wedge$ T = M, jos M <: S, M <: T ja kaikille tyypeille L, jos L <: S ja L <: T niin L <: M.

Riippuen alityypityksellisen kielen alityyppirelaation määrittelystä, jokaisella tyyppiparilla on tai ei ole liitosta. Annetulla alityyppirelaatiolla sanotaan olevan liitokset, jos jokaiselle S ja T on olemassa jokin S:n ja T:n liitos J. Samoiten alityyppirelaatiolla sanotaan olevan katteet, jos jokaiselle S ja T on jokin S:n ja T:n kate M.

Tyyppiparin S ja T sanotaan olevan *alhaalta rajattu*, jos on olemassa tyyppi L siten että L<:S ja L<:T. Annettulla alityyppirelaatiolla sanotaan olevan *rajatut katteet*, jos jokaiselle alhaalta rajatulle parille S ja T on olemassa S:n ja T:n kate M.

VÄITE [LIITOSTEN OLEMASSAOLO JA RAJATUT KATTEET]:

1. Jokaiselle tyyppiparille S ja T, on olemassa jokin tyyppi J siten että S $\vee$ T = J. 2. Jokaiselle tyyppiparille S ja T yhteisellä alityypillä, on olemassa jokin tyyppi M siten että S $\wedge$ T = M.

Liitosoperaatiota käyttäen saadaan alityypitykselliselle if-konstruktiolle algoritminen sääntö.

$$\frac{\Gamma \vdash t_1 : T_1 \qquad T_1 = Bool \qquad \Gamma \vdash t_2 : T_2 \qquad \Gamma \vdash t_3 : T_3 \qquad T_2 \vee T_3 = T}{\Gamma \vdash \text{if } t_1 \text{ then } t_2 \text{ else } t_3 : T} \qquad \text{(TA-IF)}$$

Viitteet

[Pie02] Benjamin C. Pierce. *Types and Programming Languages* the MIT Press, Cambridge, Massachusetts, 2002.