VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS

Programavimo kalbos su priklausomais tipais transliavimas į Go programavimo kalbą

Dependently typed programming language translation to Go programming language

Bakalauro baigiamojo darbo planas darbas

Atliko: 4 kurso studentas
Justas Tvarijonas (parašas)

Darbo vadovas: Partn. Doc. Viačeslav Pozdniakov (parašas)

Vilnius – 2021

Temos aktualumas

Programavimas yra instrukcijų davimas kompiuteriui, kuris jas vykdo nepaisant to ar jos yra prasmingos ar ne. Kadangi žmonija tampa vis labiau priklausoma nuo kompiuterių, kurie atsiranda kiekviename mūsų gyvenimo aspekte, tampa vis svarbiau kuo labiau sumažinti klaidų kiekį programiniame kode. Tai bandoma įgyvendinti įvairiais metodais, tokiais, kaip detalus projektavimas ar testų rašymas. Pastarasis yra vienas dažniausiai naudojamų metodų, tačiau testavimas parodo ne klaidų nebuvimą, o tik tai, kad jos yra [Hau15].

Tipai programavimo kalbose leidžia programuotojui nurodyti numatytą programos elgseną tipų pavidalu. Tipai ne tik padeda programuotojui rašyti teisingą programą, bet taip pat, kompiuteris gali patikrinti ar sukurta programa veikia taip, kaip buvo užrašyta. Paprasčiausias pavyzdys būtų programa, kuri gavusi sąrašą skaičių, prie kiekvieno elemento prideda vienetą, šios programos tikslas yra priimti sąrašą bei sąrašą gražinti. Tai labai abstraktus apibrėžimas, ką ši programa atlieka, tačiau tai suteikia tam tikrą informaciją apie šios programos veikimą.

Priklausomų tipų sistemos [Zha92] leidžia tipams būti priklausomais nuo konkrečių reikšmių. Programavimo kalbos [Sit18; Stu16], kuriose yra naudojami priklausomi tipai leidžia sukurti tikslesnius tipus, kurie padeda daugiau klaidų aptikti kompiliavimo metu, vietoje to, kad tos pačios klaidos išliktų nepastebėtos iki programos veikimo pradžios. Su tipais, kurie turi daugiau informacijos mes daugiau žinome apie galimus programos parametrus bei rezultatus. Taip pat, priklausomi tipai programavimo kalbose suteikia galimybe rašyti įrodymus.

Viena iš programavimo kalbų su priklausomais tipais yra Agda. Tai funkcinė programavimo kalba, kurios sintaksė yra panaši į plačiau žinomos funkcinės programavimo kalbos Haskell [Lip11] sintaksę. Dabartinė Agda implementacija tipų patikrinimais siekia užtikrinti, kad programos bei įrodymai būtų teisingi. Tuo tarpu Go programavimo kalba [WP14] yra greitai kompiliuojama, bet ne taip griežtai tipizuota, taigi atsiranda poreikis tam tikras vietas suprogramuoti su Agda programavimo kalba, verifikuoti jas Agda viduje, bei tada sugeneruoti Go kodą, kurį jau galėtų panaudoti egzituojantis Go kodas.

Tikslas, uždaviniai bei laukiami rezultatai

Tikslas

Realizuoti Agda kalbos transliatorių į Go programavimo kalbą

Uždaviniai

1. Sukurti Agda kalbos transliatorių į Go programavimo kalbą
2. Verifikuoti įgyvendintą Agda kalbos transliatorių
3. Identifikuoti įgyvendinto transliatoriaus trūkumus

Laukiami rezultatai

1. Aprašyta Go poaibė į kurią yra transliuojamas Agda kodas
2. Sukurtas Agda kalbos transliatorius į Go programavimo kalbą
3. Sukurtas Agda kalbos transliatorius yra verifikuotas
4. Identifikuoti ir aprašyti įgyvendinto transliatoriaus trūkumai
5. Nustatyta sėkmingai kompiliuojama Agda standartinės bibliotekos dalis

Literatūra

[BD08] A. Bove ir P. Dybjer. Dependent types at work. *LerNet ALFA Summer School*, 2008.

[BDN09] A. Bove, P. Dybjer ir U. Norell. A brief overview of agda – a functional language with dependent types, 2009.

[DP20] P. Dybjer ir E. Palmgren. Intuitionistic type theory. Metaphysics Research Lab, Stanford University, 2020.

[GF11] D. Gustafsson ir O. Fredriksson. A totally epic backend for agda. 2011.

[Hau15] P. Hausmann. The agda uhc backend. 2015.

[HDS15] P. Hausmann, A. Dijkstra ir W. Swierstra. The utrecht agda compiler. 2015.

[HKM13] G. Huet, G. Kahn ir C. P. Mohring. The coq proof assistant a tutorial, 2013.

[Jef13] A. Jeffrey. Dependently typed web client applications - frp in agda in html5. *PADL*, 2013.

[Lip11] M. Lipovača. Learn you a haskell for great good!, 2011.

[NAD+] U. Norell, A. Abel, N. A. Danielsson, M. Takeyama ir C. Coquand. Agda readthedocs puslapis. URL: `https://agda.readthedocs.io/en/v2.6.1.1/overview.html`.

[Nor07] U. Norell. Towards a practical programming language based on dependent type theory. 2007.

[Nor08] U. Norell. Dependently typed programming in agda. *Advanced Functional Programming*, 2008.

[NPS90] B. Nordström, K. Petersson ir J. M. Smith. *Programming in Martin-Löf's Type Theory: An Introduction*. Clarendon Press, 1990.

[Sit18] B. Sitnikovski. *Gentle Introduction to Dependent Types with Idris*. Amazon/KDP, 2018.

[Stu16] A. Stump. *Verified Functional Programming in Agda*. Association for Computing Machinery, Morgan ir Claypool, 2016. ISBN: 9781970001273.

[WKS20] P. Wadler, W. Kokke ir J. G. Siek. *Programming Language Foundations in Agda*. 2020-07. URL: `http://plfa.inf.ed.ac.uk/20.07/`.

[WP14] E. Westrup ir F. Pettersson. Using the go programming language in practice, 2014.

[XP99] H. Xi ir F. Pfenning. Dependent types in practical programming. *Proceedings of the 26th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, p. 214–227, 1999.

[Zha92] L. Zhaohui. A unifying theory of dependent types: the schematic approach. *Logical Foundations of Computer Science*, p. 293–304. Springer Berlin Heidelberg, 1992.