Liet. matem. rink., **47**, *spec. nr.*, 2007, 273–278

Prielaidos automatiniam programavimo stiliaus vertinimui informatikos olimpiadose

Jūratė SKŪPIENĖ (MII)
el. paštas: jurate@ktl.mii.lt

Reziumė. Vertinant programą pagal kokybinius parametrus programavimo stilius yra svarbi vertinimo dalis. Lietuvos bei kai kurių kitų šalių nacionalinėse informatikos olimpiadose mokinių sukurtų programų programavimo stilius yra vertinamas naudojant žmogišką faktorių, tuo tarpu tarptautinėse olimpiadose jis nėra vertinamas, nes dėl daugia-kultūrinės aplinkos vertintojams sunku vienodai objektyviai įvertinti dalyvių sukurtų programų stilių per ganėtinai trumpą laiką. Tačiau universitetai, kuriuose dėstytojams tenka įvertinti didelį skaičių programų, dažnai programavimo stilių vertina remdamiesi įvairiomis automatinėmis sistemomis ir metodikomis. Straipsnyje analizuojama aukštųjų mokyklų patirtis automatiškai vertinant programavimo stilių ir pagrindžiama automatinio programavimo stiliaus vertinimo informatikos olimpiadose tikslingumas.

Raktiniai žodžiai: informatikos mokymas, algoritmavimas, informatikos olimpiados, programavimo stilius, automatinis testavimas.

1. Įvadas

Pastaruoju metu mokslinėje visuomenėje vis daugiau atgarsio susilaukia informatikos olimpiados ir ypač įvairūs darbų vertinimo jose aspektai. Pasaulinės informatikos olimpiados (IOI) yra vienas labiausiai prestižinių mokiniams skirtų kompiuterių mokslo (algoritmų kūrimo) konkursų (International Olympi..., 2007). Informatikos olimpiadų dalyviai turi sukurti algoritmus ir juos užrašyti nepriekaištingai veikiančiomis programomis viena iš leidžiamų programavimo kalbų (Paskaliu, C ar C++).

Vienas svarbiausių algoritmų, užrašytų programomis, vertinimo tikslų yra nustatyti koks algoritmas taikomas ir kiek jis tinkamas konkrečiam uždaviniui spręsti. Kadangi algoritmas užrašomas programa, taip pat reiktų įvertinti programavimo stilių, t.y. programos aiškumą, paprastumą, bendrumą.

Tarptautinėse informatikos olimpiadose algoritmų (programų) teisingumas ir efektyvumas vertinami automatiškai testuojant juodosios dėžės principu visiškai nebandant analizuoti paties algoritmo ar programos kokybės kitais aspektais. Lietuvos bei kai kurių kitų šalių nacionalinėse informatikos olimpiadose sprendimai vertinami ne tik pagal automatinio testavimo rezultatus, bet ir atsižvelgiama į algoritmo pagrindimą bei programavimo stilių (Skūpienė, 2004).

Tarptautinėse olimpiadose naudojama vertinimo schema pastaruoju metu susilaukia gan daug kritikos, nes vertinimas, paremtas vien programos vykdymo rezultatais neatskleidžia svarbiausių algoritmo savybių ir ypatybių. Netgi buvo iškelta mintis,

kad esant tokiam vertinimui mokslinė informatikos olimpiadų vertė yra abejotina (Verhoeff, 2006).

Programavimo stilius tarptautinėse olimpiadose nėra vertinamas ir toks požiūris netvarkingas, painias ir sunkiai skaitomas programas sulygina su tvarkingais ir elegantiškais darbais, jeigu tik sutampa kiti realizacijos parametrai, kurie turi įtakos vertinimui.

2. Programavimo stiliaus taksonomijos ir standartai

Atsiradus aukšto lygio programavimo kalboms greitai buvo suprasta, kad svarbu ne tik tai ką atlieka parašyta programa, bet ir tai, kaip ji yra parašyta. Juk programa retai kada rašoma „kompiuteriui“, dažnai ją tenka vertinti, koreguoti, tobulinti, papildyti pačiam autoriui arba kitiems asmenims.

Septintajame-aštuntajame dešimtmetyje pasirodė ne vienas straipsnis kuriame buvo analizuojamas programavimo stilius, pateikiami konkretūs pasiūlymai kaip turėtų būti rašoma programa (Baecker, 1988; Crider, 1978; Gilmore, 1984; Grogono, 1979). Vienas žymiausių ir plačiausiai cituojamų to laikmečio darbų, turėjusių didelės įtakos programavimo stiliaus standartų formavimuisi buvo B.W. Kernighan ir P.J. Plauger knyga „Programavimo stiliaus elementai“ (Kernighan, 1974).

Tačiau apskritai tuo metu spausdinamuose darbuose vyravo stilių įvairovė, pateikiami (siūlomi) programavimo stiliai pasižymėjo nenuoseklumu, neišsamumu ir netgi prieštaringumu. Deja, nei programavimo stiliui skirtuose darbuose, nei pirmuosiuose automatiniuose programavimo stiliaus analizatoriuose, nei automatinėse programos teksto tvarkymo programose nebuvo rimto teorinio ar empirinio pagrindimo kodėl laikomasi (ar siūloma laikytis) vienokių ar kitokių programavimo stiliaus taisyklių (Oman, 1990).

Didelis žingsnis į priekį buvo padarytas 1990 metais sukuriant programavimo stiliaus taksonomiją. Programavimo stiliaus taksonomija buvo sukurta atidžiai analizuojant ir sisteminant įvairias programavimo stiliui skirtus žinynus, knygas, automatinius programos teksto analizatorius, realias programas.

Gautoji programavimo stiliaus taksonomija yra suskaidyta į keturias svarbiausias kategorijas: projektavimo stilius, tipografinis stilius, valdymo struktūrų stilius, informacinių struktūrų stilius. O kiekviena šių kategorijų buvo dar suskaidyta į makro ir mini kategorijas (Oman, 1990).

Programavimo stiliaus taksonomija neišsprendė prieštaravimų tarp skirtingų stilių, tačiau apibendrino ir susistemino programavimo stiliaus komponentus bei atvėrė kelią programavimo stiliaus standartų kūrimams, nes remiantis taksonomija jau buvo galima sukurti išsamų ir neprieštaringą programavimo stiliaus standartą. Kartu taksonomija palieka galimybę įtraukti naujus elementus ar perkelti elementą iš vienos kategorijos į kitą.

Taksonomijos sukūrimas taip pat padėjo apibrėžti kas vis tik įeina į programavimo stiliaus sampratą. Dažnai laikoma, kad programavimo stilius apima visa tai kas susiję su programos „išvaizda“, kuri svarbi programuotojui, tačiau kuri neturi jokios įtakos kompiliatoriaus darbui ar programos veikimui. Pavyzdžiui, identifikatorių vardų parinkimas, programos komentavimas, programos teksto išdėstymas. Tuo tarpu P.W. Oman ir C.R. Cook programavimo stilių supranta plačiau. Taksonomija

apima ir kitus programos aspektus, kurie turi įtakos svarbiausiems geros programos savybėms – aiškumui, paprastumui ir bendrumui. Pavyzdžiui, modulių įdėtumas, globalių ir lokalių duomenų naudojimas.

Tolesni tyrinėjimai programavimo stiliaus tema apima gan daug specifinių situacijų, konkrečių taikymų, standartų kūrimą vienos ar kitos grupės konkretiems produktams, tačiau nebuvo bandymų toliau plėtoti taksonomijos sampratą, analizuoti ir suderinti skirtingu standartus, apibendrinti.

Įvairios programavimo grupės įvairioms programavimo kalboms yra sukūrę įvairių programavimo stiliaus standartų. Yra išleista nemažai knygų skirtų programavimo stilių standartams nemažai dėmesio skiriant naujesnėms, plačiai paplitusioms kalboms (Cargill, 1992; Vermeulen, 2000). Tuo tarpu tarptautinėse informatikos olimpiadose leidžiama rašyti programas Paskalio, C ar C++ kalbomis. Paskalio kalba naudojama praktiškai mokymui ir tėra trys pagrindiniai šios kalbos standartai: GNU Paskalio kalbos standartas, Borland/Delphi objektinio Paskalio stiliaus vadovas, Delphi programuotojų stiliaus standartas (Delphi 2007). C/C++ kalboms yra sukurta daugiau standartų. Tai ir GNU C/C++ programavimo stiliaus standartas, Ellemtel C++ programavimo stiliaus standartas, ir Europos Laboratorijos C++ programavimo stiliaus specifikacija ir t.t. (C Coding, 2007).

Nežiūrint standartų gausos daugeliu atveju yra pabrėžiama, kad nėra svarbus koks konkrečiai standartas bus pasirinktas ir naudojamas, programuotojas gali net pats pasirinkti savo standartą, daug svarbiau yra išlaikyti stiliaus vientisumą.

3. Automatinis programavimo stiliaus koregavimas ir vertinimas

Automatinis programavimo stiliaus tvarkymas ar vertinimas dažniausiai sutinkamas trijų tipų programinėje įrangoje. Automatiškai tvarko programos tekstą teksto tvarkymo programos (angl. pretty-printers, beautifiers). Jos stengiasi programos tekstą išdėstyti taip, kad jis būtų lengviau suprantamas ir skaitomas. Nemažai automatinių teksto tvarkymo programų buvo sukurta septintajame- aštuntajame dešimtmetyje (Bates, 1981; Hansen, 1989; Winter, 1988). Pataruoju metu tokių programų kuriama mažiau ir jos dažniau skirtos atviro kodo programinei įrangai (Open 2007). Galima spėti, kad viena to priežasčių yra automatinės programos teksto tvarkymo programos integruotos į programavimo kalbos aplinką. Kuriant programą integruotoje aplinkoje dažnai iš karto pasiūlomas teksto išretinimas, kai kurie kiti stiliaus elementai.

Automatiniu programavimo stiliaus vertinimo sistemų yra sukūrę įvairūs universitetai, kurie dirba su dideliais studentų srautais ir norima užtikrinti, kad studentų sukurtos programos būtų pakankamai aukštos kokybės (Schorsch, 1995; Ala-Mutka, 2004). Pirmosios programavimo stiliaus vertinimo schemos buvo sukurtos aštuntajame dešimtmetyje. Buvo išskirta daug konkrečių programavimo stiliaus kriterijų ir kiekvienam kriterijui buvo nustatyta minimali ir maksimali riba į kurią turi patekti kriterijus. Pavyzdžiui, rekomenduojamas komentaro eilučių skaičius turėtų sudaryti 20–40% programos teksto eilučių (Rees, 1982). Remiantis ir vėliau plėtojant šią metodiką buvo sukurta ir daugiau programavimo stiliaus vertinimo sistemų.

4. Automatinio programavimo stiliaus vertinimo informatikos olimpiadose galimybės

Ankstesni darbai ir universitetų patirtis automatinio programavimo stiliaus vertinimo temoje sudaro galimybes tai realizuoti ir informatikos olimpiadose. Tačiau informatikos olimpiados turi savo specifiką į kurią būtina atkreipti dėmesį. Jeigu universiteto dėstytojai gali parekomenduoti besimokantiems programuoti naudoti vieną ar keletą stilių ir automatinį vertinimą suderinti su rekomenduojamais stiliais, tai informatikos olimpiadose to padaryti neįmanoma. Į informatikos olimpiadas susirenka dalyviai iš viso pasaulio ir kiekvienas jų programuoja savu stiliumi, atspindinčiu jo asmeninę patirtį, mokyklą, jo aplinkai būdingą programavimo stilių. Tad norint įvesti automatizuotą programavimo stiliaus vertinimą būtina sukurti vertinimo metodiką, kuri nebūtų susieta su jokiu konkrečiu programavimo stiliumi, remtųsi pačiais bendriausiais stiliaus reikalavimais bei tikrintų ar dalyvis nuosekliai laikosi pasirinkto programavimo stiliaus.

Automatiškai būtų galima vertinti tik nemažą dalį programavimo stiliaus elementų, bet ne visumą (pavyzdžiui, sudėtinga būtų automatiškai vertinti kintamųjų vardų parinkimo mnemoniką). Kuriant programavimo stiliaus automatinio vertinimo metodiką vertėtų patyrinėti kiek vertinimas pagal kuriamą metodiką būtų artimas programavimo stiliaus įvertinimu, kai jį vertina ekspertai. Tai padėtų nustatyti ar ta automatiškai nevertinama programavimo stiliaus dalis yra statistikai reikšminė bendro stiliaus įvertinimo atžvilgiu.

Programa, sukurtą olimpiados metu nuo programos, sukurtos ne varžybose skiriasi laiko ribojimu. Olimpiadose dažniausiai per penkias valandas tenka parašyti tris programas, tad vieno uždavinio sąlygos perskaitymui, sprendimo apgalvojimui, programos parašymui ir suderinimui vidutiniškai tenka šimtas minučių. Tai atsispindi ir programavimo stiliuje. Net ir geru stiliumi programuojant kai tenka taupyti laiką vienas ar kitas stiliaus kriterijus yra paaukojamas. Pavyzdžiui, kintamųjų vardai gali būti trumpesni ir dėl to prastesnė jų mnemonika. Į tai būtina atkreipti dėmesį kuriant stiliaus vertinimo metodiką.

Programavimo stiliaus vertinimo įtraukimas į bendrą programos vertinimo schemą, t.y. stiliaus įvertinimo susiejimas su programos teisingumo ir efektyvumo vertinimu taip pat yra gana svarbus. Jeigu dalyvis neieškojo gudresnio algoritmo, o parašė trumpą ir gražią kelių eilučių programą generuojančią atsitiktinį rezultatą arba parašė programą, kuri apskritai nesprendžia uždavinio, tai balai už programavimo stilių neturėtų būti skiriami. Kita vertus, pripažįstama, kad esamas vertinimo schema nėra pakankamai objektyvi, t.y. būna atvejų, kai gera programa dėl nedidelės klaidos pagal esamą vertinimo schemą nesurenka taškų (Skūpienė, 2006). Stiliaus įtraukimo į bendrą vertinimo schemą metodika ypatingai svarbi dar ir todėl, kad pasaulinėse mokinių informatikos olimpiadose dažnai net vieno ar dviejų taškų skirtumas (iš 600 galimų) nulemia medalio spalvą ir apskritai ar medalis būtų skiriamas, ar ne.

5. Išvados

Automatinis programavimo stiliaus vertinimas taikomas universitetuose vertinant studentų darbus, yra sukurta automatinių vertinimo sistemų. Šia patirtimi galima pasi-

naudoti ir informatikos olimpiadose. Tam būtina sukurti tinkamą programavimo stiliaus vertinimo metodiką bei programinę įrangą. Kuriant stiliaus vertinimo metodiką svarbu atsižvelgti į informatikos olimpiadų specifiką, t.y. programavimo stiliaus vertinimas jokiu būdu neturėtų būti susietas su konkrečiu stiliaus standartu, nes olimpiadose dalyvauja įvairių pasaulio šalių atstovai, programuojantys įvairiausiu stiliumi. Kitas svarbus aspektas – reiktų ieškoti objektyviausio būdo kaip įtraukti programavimo stiliaus vertinimą į egzistuojančią vertinimo schemą, t.y. kaip susieti stiliaus vertinimą su sprendimo teisingumu.

Literatūra

1. R. Baecker, Enhancing program readability and comprehensibility with tools for program visualization, in: *Proceedings 10th International Conference on Software Engineering*, Singapore (1988), pp. 356–366.
2. R.M. Bates, A Pascal prettyprinter with a different purpose, *ACM SZGPLAN Notices*, **16**, 10–17 (1981).
3. W.P. Oman, C.R. Cook, *Journal of Systems and Software*, **15**, 287–301 (1991).
4. T. Cargill, *C++ Programming Style*, Addison-Wesley Professional (2002).
5. W.P. Oman, C.R. Cook, *Journal of Systems and Software*, **15**, 287–301 (1991).
6. *Coding Style Standards* (2007). `http://www.textrush.com/coding-standard.htm`
7. J.E. Crider, Structured formatting of Pascal programs, *ACM SIGPLAN Notices*, **13**, 15–22 (1978).
8. D.J. Gilmore, T.R. Green, Comprehension and recall of miniature programs, *International Journal of Man-Machine Studies*, **21**, 31–48 (1984).
9. P. Grogono, On layout, identifiers and semicolons in Pascal programs, *ACM SIGPLAN Notices*, **14**, 35–40 (1979).
10. *International Olympiads in Informatics*, official page (2007). `http://olympiads.win.tue.nl/ioi/.`
11. J. Hansen, B. Sands, Some design considerations for a C source code pretty printer, *ACM SIGSMALL/PC Notes*, **11**, 16–22 (1985).
12. W.P. Oman, C.R. Cook, *Journal of Systems and Software*, **15**, 287–301 (1991).
13. B.W. Kernighan, P.J. Plauger, *The Elements of Programming Style*, McGraw-Hill, New York (1974).
14. W.P. Oman, C.R. Cook, *Journal of Systems and Software*, **15**, 287–301 (1991).
15. *Open Source Java Code beautifiers written in Java* (2007). `http://www.roseindia.net/opensource/javacodebeautifiers.php.`
16. M.J. Rees, Automatic assessment aids for Pascal programs, *SIGPLAN Notices*, **17**(10), 33–42 (1982).
17. T. Schorsch, CAP: an automated self assessment tool to check Pascal programs for logic, in: *Proceedings of the twenty-sixth SIGCSE Technical Symposium on Computer Science Education* (1995), pp. 168–172.
18. J. Skūpienė, Automatinis testavimas informatikos olimpiadose, kn.: *Informacinės technologijos 2004*, Konferencijos pranešimų medžiaga, Technologija, Kaunas (2004), pp. 37–41.
19. J. Skūpienė, Programming style – part of grading scheme in informatics olympiads: Lithuanian experience, in: *Infomatics Education – the Bridge between Using and Understanding Computers. Proceedings of International Conference in Informatics in Secondary Schools – Informatikon Technologies at School*, Vilnius, TEV (2006), pp. 545–552.
20. T. Verhoeff, The IOI is (not) a science olympiad, *Informatics in Education*, **5**(1), 63–76 (2006).
21. A. Vermeulen, S.W. Ambler *et al.*, *The Elements of Java Style*, Cambridge University Press (2000).
22. K.A. Winter, *A Prototype Intelligent Prettyprinter*, Master's Thesis, Oregon State University Computer Science Dept. (1988).

SUMMARY

J. Skūpienė. Possibilities for automated programming style assessment in informatics olympiads

Programming Style is an important part of program quality and it should be taken into account while assessing programs designed by competitors in informatics. In International Olympiad in Informatics grading is automated and based on testing results only, while programming style is not taken into account. However there exists practice in universities in programming courses where programming style of submitted programs is evaluated automatically. The paper reviews existing experience and discusses possibilities for automated grading of programming style in informatics olympiads.

Keywords: teachning programming, algorithms, programming style, automated grading.