Automatinio mokinių programų vertinimo sistemų lyginamoji analizė

Bronius Skūpas

Matematikos ir informatikos instituto tyrėjas, doktorantas
Institute of Mathematics and Informatics, Researcher, PhD student
Akademijos g. 4, LT-08663 Vilnius
El. paštas: bskupas@ktl.mii.lt

Mokant programavimo jau seniai taikomas automatinis vertinimas. Tokiu atveju mokiniai (ar studentai) turi programas-sprendimus pateikti testavimo sistemai. Kad sistema galėtų patikrinti darbus, jie turi tenkinti gana griežtus reikalavimus. Savo ruožtu automatinė vertinimo sistema gali įvertinti dalį pamatuojamų kriterijų, patikrinti programos funkcionavimą su pradinių duomenų rinkiniais. Automatinių sistemų funkcionalumas kartais praplečiamas rankinėmis vertinimo galimybėmis, taip gaunant pusiau automatines sistemas. Sistemose paplitę du pagrindiniai programų analizės metodai: dinaminis ir statinis, o vertinimo būdų, kriterijų įvairovė yra gerokai didesnė. Automatinės mokinių programų vertinimo sistemos gelbsti ne tik vertinant kontrolines užduotis. Jos gali turėti ir mokomąją funkciją – mokinys gauna galimybę greitai, vos ne bet kuriuo momentu pasitikrinti, gauti kitą panašią užduotį, esti įsitikinęs vertinimo objektyvumu. Siekiant tobulinti automatinio vertinimo mokomąją funkciją reikia skirti daugiau dėmesio esamų sistemų analizei, jų pranašumams ir trūkumams. Straipsnyje lyginamosios analizės būdu vertinami sprendimai, siūlomos naujos sistemų tobulinimo sritys.

Įvadas

Jau susiklostė tradicija informacinių technologijų kurse mokykloje mokyti programavimo. Programavimo mokymas aktualus ir dėstant informatikos ar programų inžinerijos studentams. Mokant šių dalykų dažnai susiduriama su poreikiu tikrinti mokinių (ar studentų) parašytas programas. Šių programų vertinimas atima gana daug laiko. Vertinimas įgyja savo problemų valstybiniame informacinių technologijų brandos egzamine, kuriame praktinių užduočių sprendimus pateikia beveik tūkstantis abiturientų. Šiose situacijose taikomas pusiau automatinis programų vertinimas.

Automatinis programų vertinimas dažnai naudojamas informatikos olimpiadose, programavimo varžybose, programavimo mokymo sistemose. Jis padeda greitai apdoroti didelius kiekius pateiktų sprendimų, yra objektyvus. Tačiau daugeliu atvejų apsiribojama juodosios dėžės principu – programa išbandoma su parinktais duomenų rinkiniais ir patikrinami gauti rezultatai. Norint pasiekti tikslesnį programos vertinimą, tektų didinti programų formos apribojimus, taikyti sudėtingus statinės ir dinaminės analizės metodus. Tačiau net ir padidinus apribojimus tai neišsprendžia nemažos dalies klausimų: ar tikrai sprendimui taikomas reikalingas algoritmas, ar panaudota duomenų struktūra iš tiesų saugo reikiamus duomenis ir pan.

Tyrimo tikslas – palyginti keletą svarbių automatinio mokinių programų vertinimo sistemų, atrasti bendras šių sistemų savybes ir esminius skirtumus, pateikti jų naudojimo mokyklose ir galimo sistemų tobulinimo rekomendacijas.

Ankstesnių darbų apžvalga

Mokymo sferoje jau seniai tiriamos galimybės vertinti kompiuteriu (angl. *computer aided assessment – CAA*) (Carter, 2003). Tačiau šioje srityje dažniau analizuojamos galimybės naudoti kompiuterį teorinėms apklausoms ir testams nei praktinių darbų vertinimui. Kompiuteris šiuose darbuose daugiau duomenų kaupimo, apdorojimo įrankis ir komunikacijos priemonė nei sprendimus priimanti sistema (Ala-Mutka, 2005).

Atvirųjų klausimų, algoritmų, programų vertinimui dažnai reikia gerokai daugiau laiko nei testo su uždaraisiais klausimais vertinimui. Todėl daugiau dėmesio turėtų būti skiriama praktinių užduočių bei atvirųjų klausimų automatinio vertinimo problemoms spręsti. Visgi šioje srityje tyrimų atliekama mažiau. Viena šios tendencijos priežasčių – bendrasis CAA yra svarbus ne vien informatikos mokymui.

Pastaruoju metu automatinių sistemų taikymu domimasi labiau. Pavyzdys – 2005 m. išleistas „ACM Journal on Educational Resources in Computing“ specialusis leidinys skirtas automatiniam vertinimui (ACM, 2005).

Lyginamoji analizė buvo pradėta remiantis Kirsti M. Ala-Mutka (2005) straipsniu, kuriame iš dalies klasifikuotos automatinės programų vertinimo sistemos. Skiriami du programų analizės metodai – statinis ir dinaminis.

Dinaminė analizė sudaro prielaidas stebėti vykdymo seką, atrasti pavojingą veikseną. Savo ruožtu vertinimui naudingiausi šie analizės rezultatai:

- funkcionalumo įvertis;
- efektyvumo įvertis;
- studento testavimo gebėjimų įvertis;
- kitos savybės (programavimo kalbos konstrukcijų naudojimas, duomenų struktūrų stebėjimas ir kt.).

Naudojant statinę analizę dažnai realizuojamos šios funkcijos:

- programavimo stiliaus vertinimas;
- programų sintaksinių ir semantinių klaidų paieška;
- programų matų skaičiavimas (vykdomų eilučių kiekis ir pan.);
- programos dizaino ir sąsajos vertinimas;
- programų struktūros ir veiksmų palyginimas;
- programų reikšminių žodžių paieška;
- programų struktūrogramų analizė;
- programų plagiato atpažinimas.

Sistemos taip pat gali būti skirstomos į visiškai automatines ir pusiau automatines; vertinimo formavimo mechanizmai gali būti skirtingi – sistema gali sumuoti taškus už skirtingas savybes arba gali reikalauti tol tobulinti, kol bus gautas visus reikalavimus tenkinantis rezultatas.

Objektas

Tyrime buvo analizuojamos viešai prieinamos automatinio mokinių programų vertinimo sistemos (toliau – sistemos). Dalis jų buvo pasiekiamos per internetą, kitos parsisiųstos su išeitiniais programų tekstais. Kai kurios jų yra iki šiol kūrimo ar tobulinimo stadijos.

Sistemos analizuotos pagal jų taikymo sritį:

- Valadolido universiteto programavimo mokymo sistema (*UVA Online Judge*) ir iš jos besivystanti EduJudge sistema (*EduJudge*), šioms sistemoms artimas JAV informatikos olimpiados treniruočių portalas (*USACO training program*).
- Warwicko universiteto studentų programų vertinimo sistema (BOSS Online Submission System), Notingemo universiteto studentų programų vertinimo sistemos (*Ceilidth, CourseMarker*).
- 2002 m. tarptautinės informatikos olimpiados programavimo varžybų aptarnavimo sistema (*IOI 2002 Contest and Grading System*) ir iš jos kilusi Lietuvos informatikos olimpiados programavimo varžybų aptarnavimo sistema (*Lietuvos informatikos olimpiados PVAS*).
- Valstybinio informacinių technologijų brandos egzamino praktinių užduočių vertinimo sistema (*ITBE PUVS*).

Tyrimo metodai

Darbui taikytas teorinis tyrimo metodas, atlikta lyginamoji analizė.

Pagrindinis sistemų parinkimo tyrimui argumentas buvo jų priklausomybė tam tikrai taikymo sričiai, informacijos apie jas gausa bei

šių sistemų minėjimas kituose šaltiniuose. Taip pat argumentas buvo sistemų populiarumas tarp mokinių ir studentų.

Apie sistemas informacija rinkta literatūroje, sistemų svetainėse viešai publikuojamoje dokumentacijoje (pvz., *CourseMaker*). Dalį sistemų pavyko išbandyti (*UVA Online Judge, USACO, IOI 2002 PVAS, LIO PVAS, ITBE PUVS*). Kitos buvo sunkiau prieinamos, pasiekiamos tik komerciniams vartotojams, laisvai neplatinamos (*BOSS, CourseMarker, Ceilidth*).

Analizuojamoms sistemoms buvo suteikiami reikšminiai žodžiai, kurie parodė jų vietą dabartinėje sistemų klasifikacijoje, taip pat išryškino jas skiriančius bruožus.

Tiriamų sistemų ypatumai

Šiuo metu Lietuvos mokyklose labiausiai žinoma *Lietuvos olimpiadų PVAS*. Ji yra viena paprasčiausių savo teikiamomis paslaugomis. Sistemai būdingas automatizuotas juodosios dėžės testavimas, taigi, dinaminė analizė. Esminiai sistemos patobulinimai *IOI'2002 PVAS* atžvilgiu – dviejų amžiaus grupių atskyrimas, automatinė registracija varžyboms, pusiau automatinio vertinimo modulis, skirtas sprendimų stiliui ir idėjai vertinti. Vienas svarbiausių sistemos privalumų, perimtų iš *IOI 2002*, tas, kad sistema dirba tinkle, turi kelių testavimo mašinų valdymo mechanizmą.

IT valstybinio brandos egzamino *PUVS* skiriasi platforma (dirba *Windows OS*), paprasta struktūrine schema su viena testavimo mašina. Esminis pranašumas – numatyta ne tik juodosios dėžės principu veikianti dinaminė analizė, bet ir statinė analizė. Ji atlieka programų matų skaičiavimus, vertina stilių, atlieka reikšminių žodžių paiešką. Taip pat joje ištobulintas pusiau automatinis interaktyvus vertinimas, leidžiantis eksperimentuoti su sprendimais.

Pirmiau minėtos sistemos neturi mokymo funkcijos – jos numatytos darbui ribotu laiku. Pateikiamos užduotys sprendžiamos per varžybas ar egzaminą, užduočių dalyviai rinktis negali, sistemos grąžinama informacija dalyviui ribota, vertinimo rezultatai skelbiami po tam tikro laiko. Testiniai duomenų rinkiniai slepiami iki skelbiant rezultatus.

Toliau minimos sistemos skirtos daugiau mokymui nei varžyboms, nors tai netrukdo jas naudoti ir varžyboms.

BOSS, *Ceilidth*, *CourseMarker* yra skirtos universitetų bendruomenėms, naudojamos programavimo mokymo kursuose. *BOSS* sistema daug dėmesio skiria atvirų klausimų testavimui. Automatinis programų vertinimas – tai tik vienas iš kelių jos modulių (Heng, 2005). Joje realizuota tik juodosios dėžės dinaminės analizės vertinimo metodika. Pagal galimybes naudoti įvairius vertinimo modulius, dinaminę ir statinę analizę toliausiai pažengusi sistema yra *CourseMarker*. Ši sistema išsivystė iš anksčiau naudotos sistemos *Ceilidth*, kuri buvo viena labiausiai paplitusių (ją naudojo apie 200 įstaigų). Kuriant *CourseMarker* sistema buvo perrašyta iš naujo stengiantis išlaikyti didžiąją dalį *Ceilidth* galimybių. Šiai sistemų grupei būdinga naudoti gausybę įvairių įrankių. Tai suteikia universalumo, tačiau nukenčia sistemos mobilumas. Ji darosi sunkiai perkeliama į kitus kompiuterius, sudėtingai integruojama. Šios sistemų grupės atstovams būdingos pastangos tikrinti darbus plagiato indikatoriais.

UVA Online Judge (Revilla, 2008) ir *USACO training program* (Kolstad, 2007) (toliau – *USACO*) yra prieinamos nemokamai. *UVA Online Judge* ir *USACO* buvo kuriamos norint parengti studentus ir mokinius programavimo varžyboms (atitinkamai *ACM ICPC* ir *IOI*). Todėl jų mokomoji funkcija truputį skiriasi nuo kitų universitetinių sistemų. Šiose sistemose mažiau dėmesio kreipiama į programavimo stilių, sprendimo elegantiškumą. Daugiau dėmesio skiriama algoritmų funkcionalumui, efektyvumui, jos yra visiškai automatinės, pritaikytos priimti daug lankytojų. Šiuo metu *USACO* užduočių bankas yra nedidelis (97 užduotys), geriau struktūrizuotas, užduotys mokiniams pateikiamos po kelias, jos suskirstytos temomis. *USACO* orientuojasi į savarankišką mokymąsi, todėl naudoti pamokose nėra patogu. *UVA Online Judge* suteikia gerokai daugiau laisvės renkantis užduotį (užduočių

L e n t e l ė . ***Automatizuoto vertinimo sistemų palyginimas***

Sistemos		Skirtos mokymui					Trumpo naudojimo laikotarpio		
		Pasirengimo varžyboms			Universitetinės		Varžybų		Egzaminų
		UVA Online Judge	*EduJudge*	*USACO training program*	*BOSS Online Submission System*	*Ceilidh ir Course-Marker*	*IOI 2002 Contest and Grading System*	*Lietuvos informatikos olimpiados PVAS*	*IT BVE prakt. už. vertinimo sistema*
Dinaminė analizė	Funkcionalumo ir efektyvumo įverčiai	+	+	+	+	+	+	+	+
	Studento testavimo gebėjimų įvertis (galima testuotis su savo testais)	-	-	-	-	+	+/-	+/-	-
	Kitos savybės (programavimo kalbos konstrukcijų naudojimas, duomenų struktūrų stebėjimas, ...)	-	+[2]	-	-	+[2]	-	-	+[2]
Statinė analizė	Programavimo stiliaus vertinimas	-	+[1]	-	+[2]	+	-	+[3]	+
	Sintaksinių ir semantinių klaidų paieška[4]	-	-	-	-	+[2]	-	-	+[2]
	Programų matų skaičiavimas (vykdomų eilučių skaičius, ...)	-	+[1]	-	+[2]	+	-	-	+
	Programos dizaino ir sąsajos vertinimas	-	+[3]	-	+[2]	+[3]	-	-	+[3]
	Programų struktūros ir veiksmų palyginimas	-	-	-	+[2]	+	-	-	+
	Programų reikšminių žodžių paieška	-	+[1]	-	+[2]	+	-	-	+
	Programų struktūrogramų analizė	-	+[1]	-	+[2]	+[2]	-	-	-
	Programų plagiato atpažinimas	-	+[1]	-	+	+	-	-	-
Vertin. automat.	Vertinimas visiškai automatizuotas	+	+	+	+	+	+	+	+
	Kai kurie komponentai gali būti vertinami rankiniu būdu	-	+	-	+	+	-	+	+
Vertinimo formavimo mechanizmas	Reikalauja veikiančių visų teisingų testų	+	+	+	+	+	-	-	-
	Vertinimas formuojamas pagal skirtingą teisingų testų vertę	-	+	-	-	+	+	+	+
	Vertinimas grupuojant testus	-	+	-	-	-	-	-	-
	Vertinimas gali priklausyti nuo laiko momento ir konkurentų pasiekimų	-	+	-	-	-	-	-	-
	Vertinama tik praėjus varžybų momentui	+	+	-	+	+	+	+	+
Užd. bankas	Yra parengtas užduočių bankas	+	+	+	-	-	-	+/-	+/-
	Gali pildyti kiti vartotojai (pvz., mokytojai)	-	+	-	+	+	-	-	-
Testo klaus.	Atviri	-	+	-	-	+	-	-	-
	Uždari	-	+	-	-	+	-	-	-
Vertimas	Yra galimybė išversti	-	+	-	-	+	+	+	-
	Išversta	-	+/-	-	-	-	-	+	+
Kaina	Naudotis galima nemokamai	+	+	+	-	-	+	+	-
	Atvirasis kodas	-	+	-	-	-	+/-	+/-	+/-

[1] Kreipimosi į kitas sistemas būdu.
[2] Galima naudojant papildomus įrankius.
[3] Atliekant pusiau automatinį vertinimą.
[4] Sintaksinis tikrinimas kompiliavimo metu realizuojamas visose sistemose. Čia akcentuojama papildomos analizės galimybė.

bankas yra didesnis – apie 2000 užduočių), pateikiama šios užduoties sprendimo statistika. Internete yra nemažai sistemos naudotojų svetainių, analizuojančių, klasifikuojančių užduotis, – taip stengiamasi sumažinti sistemos mokomojo vaidmens trūkumą. Šiuo metu vystomas *EduJudge* projektas siekia išplėsti *UVA Online Judge* galimybes, išspręsti lokalizavimo, sistemos moduliškumo, integravimo į mokymo sistemas problemas. Tai bus daroma integruojant automatizuoto programų vertinimo galimybes į virtualiąją mokymosi aplinką *Moodle*. Užduočių bankas bus perimtas iš *UVA Online Judge*. Jis bus struktūrizuotas, patogesnis mokymui.

Rezultatai

Sistemų lyginimas parodė, kad daugumai jų būdingas specifinis užduočių rengimo formatas. Tai sukelia papildomų problemų norint pakeisti dabar naudojamą sistemą kita ar papildyti dabar naudojamos sistemos užduočių sąrašą. Nors *IMS Global Learning Consortium* yra pasiūlęs klausimams ir testams saugoti XML schemą *IMS QTI* (2005), tačiau ji be papildymų programavimo užduotims nėra tinkama. Mokymo objekto modelis *IEEE LOM* (2002) kaip pagrindas programavimo užduotims turėtų tikti, tačiau dėl programavimo užduočių specifikos (testų duomenys, rezultatai, tikrinimo programa, laiko ribojimai ir t. t.) jį reikia plėsti. EduJudge projekte kuriama programavimo užduočių specifikacijos schema (Leal, Queirós, 2008), kuri išplečia *IEEE LOM*. Taigi artimiausiu metu turėtų nusistovėti galimai standartinė užduočių mokomojo objekto metaduomenų schema. Turėtų pagerėti galimybės užduotis kilnoti iš vienų programų automatinių vertinimo sistemų į kitas. Kuriant kitas sistemas reiktų atsižvelgti į šią iniciatyvą ir stengtis naudoti vienodus užduočių apibrėžimo standartus.

Kita bendra sistemų savybė ta, kad dauguma naudoja juodosios dėžės dinaminės analizės metodiką. Jose programos veikimas vertinamas pagal tai, ar pateiktiems duomenų rinkiniams programa duoda teisingus atsakymus. Todėl rengiant užduotis nemažai laiko reikėtų skirti vertinimo schemos ir gerų testinių duomenų parinkimui (Skūpienė, 2007). Jie turėtų tikrinti ne vien programos funkcionalumą su dideliais ar mažais duomenų rinkiniais. Duomenų sudėtingumas didėjant testo numeriui turėtų kisti eksponentiškai – taip parinkti duomenys demonstruoja sprendimo efektyvumą. Naudojant tuos pačius testų rinkinius daug metų iš eilės iškyla problemų: testų efektyvumas krinta, nes mokinių kartos keičiasi tarpusavyje duomenimis. Todėl svarstytina bent nesudėtingų testų dinaminio generavimo idėja.

Norint naudoti automatinio programų vertinimo sistemas Lietuvos mokyklose, daugiau dėmesio reikėtų skirti pasirenkamos sistemos mokomosios funkcijos egzistavimui. Gerai susistemintas užduočių bankas, galimybė susitvarkyti savo užduočių rinkinį tikrai svarbi mokymui.

Didelė dalis aptariamų sistemų neturi numatytos lokalizavimo galimybės. Šiuo aspektu daug vilčių teikia *EduJudge* projektas[*], kuriame yra numatyta ne tik aplinkos vertimas, bet ir sąlygų keliomis kalbomis galimybė.

Sistemos vertę padidina pusiau automatinis vertinimas. Nei dinaminė, nei statinė analizė šiuo metu dar neturi tokio pajėgumo, kad galėtų įvertinti programas visais aspektais. Todėl svarbi galimybė mokytojui įsikišti į vaiko darbo vertinimą.

Iš sistemų palyginimo lentelės matome, kad mokykloms galima rekomenduoti pasidomėti automatinio vertinimo sistemomis. Nemokamų sistemų išbandymas niekuo neįpareigoja, o tarp jų yra tokių, kurios padeda individualizuoti mokymą. *EduJudge* projekto tebekuriama sistema savo galimybėmis nenusileis populiariausioms komercinėms sistemoms, todėl į ją vertės atkreipti dėmesį ateityje.

Automatinės programų testavimo sistemos efektyvina individualizuotą mokymą, kuris būtinas dėstant programavimą. Galimybės pasitikrinti sprendimą be mokytojo įsikišimo, gauti kitą užduotį, pagalbinį patarimą sprendžiant, įvertinimą pagal sprendimui sugaištą laiką ir kitos rodo, kad šiuolaikinės sistemos dar turi erdvės tobulėti.

[*] Straipsnis rengtas dalyvaujant *EduJudge* projekte, kurį remia Europos Komisija (paramos skyrimo numeris – 135221-LLP-1-2007-1-ES-KA3-KA3MP)

LITERATŪRA

ACM Journal on Educational Resources in Computing, 2005, vol. 5, no. 3.

ALA-MUTKA, Kirsti M. (2005). A survey of automated assessment approaches for programming assignments. *Computer Science Education* vol. 15, p. 83–102.

BOSS Online Submission System [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.dcs.warwick.ac.uk/boss/>.

CARTER, J.; ALA-MUTKA, K.; FULLER, U.; DICK, M.; ENGLISH, J.; FONE, W.; and SHEARD, J. (2003). How shall we assess this? Iš *WG reports* (Thessaloniki, Greece, June 30 – July 02, 2003). D. Finkel (Ed.) ITiCSE-WG '03. ACM, New York, NY, p. 107–123.

Ceilidth [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.cs.nott.ac.uk/~ceilidh>.

CourseMarker [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.cs.nott.ac.uk/~cmp>.

EduJudge [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.edujudge.eu>.

HENG, P.; JOY, M.; BOYATT, R.; GRIFFITHS, N. (2005). *Evaluation of the BOSS online submission and assessment system*. Tech. Rep. CS-RR-415, Department of Computer Science, University of Warwick Coventry, UK [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.dcs.warwick.ac.uk/reports/cs-rr-415.pdf>.

IEEE LOM (2002). IEEE Standard for Learning Object Metadata IEEE 1484.12.1-2002 [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.ieeeltsc.org/standards/1484-12-1-2002/>.

IMS QTI (2005). IMS Question and Test Interoperability v2 Final specification [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.imsglobal.org/question/index.html>.

IOI 2002 Contest and Grading System [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://theory.snu.ac.kr/ioi/>.

KOLSTAD R.; PIELE D. (2007). USA Computing Olympiad (USACO). *Olympiads in Informatics*, vol. 1, p. 105–111 [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.mii.lt/olympiads_in_informatics/pdf/INFOL016.pdf>.

LEAL J. P.; QUEIRÓS R. (2008). Integration of E-Learning Systems With Repositories of Learning Objects. Iš *ECEL 2008 – 7th European Conference on e-Learning*, Agia Napa, Cyprus, November 2008 [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.edujudge.eu/pdfs/ecel_2008.pdf;jsessionid=KLMICMIBAGLH>.

Lietuvos informatikos olimpiados PVAS [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.mif.vu.lt/olimp/>.

REVILLA M. A.; MANZOOR S.; LIU R. (2008). Competitive Learning in Informatics: The Uva Online Judge Experience. *Olympiads in Informatics*, vol. 2, p. 131–148 [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.mii.lt/olympiads_in_informatics/pdf/INFOL035.pdf>.

SKŪPIENĖ J. (2007). Automatinio sprendimų vertinimo informatikos olimpiadose raida ir perspektyvos. *Informacijos mokslai*. t. 42–43, p. 43–49 [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://www.leidykla.eu/fileadmin/Informacijos_mokslai/42-43/43-49.pdf>.

USACO training program [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://train.usaco.org/usacogate>.

UVA Online Judge [žiūrėta 2009 m. gegužės 17 d.]. Prieiga per internetą: <http://uva.onlinejudge.org/>.

COMPARATIVE ANALYSIS OF AUTOMATIC STUDENTS' PROGRAM EVALUATION SYSTEMS

Bronius Skūpas

S u m m a r y

Practice in programming is typical for graduators of secondary schools and for programming courses in universities. Students usually have programming assignments that need to be assessed. The assessment can be done using automatic assessment systems. There are several areas where such systems can be used: programming competitions and preparations for them, evaluation of maturity exam in programming, teaching of programming in courses. Static and dynamic assessment of programs is discussed. Article compares several different assessment systems described in literature and available from Internet. Comparative analysis shows main strength and weakness of automatic assessment systems. Requirements, possibilities and trends for future assessment systems are discussed.