ŠIAULIŲ UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS KATEDRA

Mantas Donelavičius

Informatikos specialybės II kurso magistrantūros studentas

PLAGIATO PATIKROS E. SISTEMA MOODLE APLINKAI

PLAGIARISM DETECTION SYSTEM FOR MOODLE

MAGISTRO DARBAS

Darbo vadovė:
Doc. S. Turskienė

Recenzentas:
Doc. G. Felinskas

Šiauliai, 2013

„Tvirtinu, jog darbe pateikta medžiaga nėra plagijuota ir paruošta naudojant literatūros sąraše pateiktus informacinius šaltinius bei savo tyrimų duomenis“

Darbo autoriaus _____

(vardas, pavardė, parašas)

Darbo tikslas ir uždaviniai

Tikslas

- Sukurti plagiato patikros e. sistemą Moodle aplinkai

Uždaviniai

- Ištirti esančius programinius produktus rinkoje
- Išanalizuoti naudojamų plagiatų aptikimo algoritmų tipus
- Suprojektuoti plagiatų aptikimo sistemą
- Sukurti plagiatų aptikimui skirtą Moodle įskiepį
- Atlikti sukurto įskiepio testavimą

Darbo vadovė _____

(vardas, pavardė, parašas)

Turinys

Įvadas

Plagijavimas – tai svetimo darbo ar jo dalies pasisavinimas. Plagijavimui galima priskirti ne vien tik rašto darbų tekstų kopijavimą, bet ir kitų darbų pristatymą savais (paveikslų, muzikos, programinio kodo...).

Su plagijavimu yra kovojama įvairiais būdais: leidžiami darbų autorius ginantys įstatymai, užsiimama švietėjiška veikla, taikomos sankcijos. Tačiau tam tikra plagijuotų darbų dalis taip ir lieka nepastebėta, kadangi darbą tikrinantis žmogus negali perskaityti ir prisiminti visų prieš tai parašytų straipsnių, analizių, knygų. [6]

Dėl to šiuo metu plagiatus padeda aptikti ir tam pritaikytos plagiatų aptikimo sistemos (kompiuterinės programos). Šios sistemos nėra pakaitalas darbą vertinančiam žmogui, jos yra tik pagalbinis įrankis, padedantis palyginti darbą su didesniu kiekių informacijos šaltinių. Šių sistemų efektyvumas priklauso nuo naudojamų paieškos algoritmų, duomenų bazėje esančių darbų kiekio ir žmogaus vertinančio rezultatus.

Dėl didėjančio interneto populiarumo ir darbų kopijavimo paprastumo buvo pasirinkta darbo tema: „Plagiato patikros e. sistema Moodle aplinkai“. Moodle pasirinkta todėl, kad tai yra pagrindinė virtuali mokymosi aplinka naudojama Šiaulių universitete.

Kuriamos plagiatų patikros sistemos tikslas – skatinti studentų savarankišką darbą. Įdiegus šios sistemos įskiepį į Moodle aplinką, kiekvienas įkeltas darbas (į Moodle) yra lyginamas su prieš tai įkeltais darbais - ieškoma plagiatų.

Plagiatų patikros sistema pasiekiama adresu: http://plag.distance.su.lt/.

Darbo pristatymas konferencijose:

2012 m. gruodžio 12 d. respublikinė mokslinė - praktinė konferencija: „Informacinių technologijų taikymas švietimo sistemoje 2012: E-studijų patirtis, aktualijos ir perspektyvos“ (priedas 1).

2013 gegužės 15 d. Šiaulių universiteto Technologijos fakulteto 8-oji Tarptautinė mokslinė konferencija: "Jaunųjų mokslininkų darbai" (priedas 2).

1 Teorinė dalis

1.1 Plagijavimo būdai

Plagijavimo būdus galima būtų suskirstyti į tris grupes pagal sudėtingumą (nuo paprasčiausio iki sudėtingiausiai aptinkamo):

1. Teksto kopijavimas, nenurodant šaltinių (specialiai ar netyčia).
2. Keičiant turinio išdėstymą, pvz.: keičiant žodžius vietomis arba naudojant sinonimus.
3. Svetimų minčių perpasakojimas savais žodžiais (arba darbas išverčiamas iš kitos kalbos ir pateikiamas kaip savo).

Sunkiausiai yra atpažinti paskutinio būdo naudojimą, kadangi:

Originalaus teksto pavyzdys:

„Critical care nurses function in a hierarchy of roles. In this open heart surgery unit, the nurse manager hires and fires the nursing personnel. The nurse manager does not directly care for patients but follows the progress of unusual or long-term patients. On each shift a nurse assumes the role of resource nurse. This person oversees the hour-by-hour functioning of the unit as a whole, such as considering expected admissions and discharges of patients, ascertaining that beds are available for patients in the operating room, and covering sick calls. Resource nurses also take a patient assignment. They are the most experienced of all the staff nurses. The nurse clinician has a separate job description and provides for quality of care by orienting new staff, developing unit policies, and providing direct support where needed, such as assisting in emergency situations. The clinical nurse specialist in this unit is mostly involved with formal teaching in orienting new staff. The nurse manager, nurse clinician, and clinical nurse specialist are the designated experts. They do not take patient assignments. The resource nurse is seen as both a caregiver and a resource to other caregivers. . . . Staff nurses have a hierarchy of seniority. . . . Staff nurses are assigned to patients to provide all their nursing care. (Chase, 1995, p. 156)“ [8]

Pakeistas tekstas (plagiatas):

„Chase (1995) describes how nurses in a critical care unit function in a hierarchy that places designated experts at the top and the least senior staff nurses at the bottom. The experts — the nurse manager, nurse clinician, and clinical nurse specialist — are not involved directly in patient care. The staff nurses, in contrast, are assigned to patients and provide all their nursing care. Within the staff nurses is a hierarchy of seniority in which the most senior can become resource nurses: they are assigned a patient but also serve as a resource to other caregivers. The experts have administrative and teaching tasks such as selecting and orienting new staff, developing unit policies, and giving hands-on support where needed.“ [8]

Nors pakeistame tekste ir yra minimas šaltinis, tačiau perpasakojant tekstą studentas naudoja frazes iš kito darbo ir tik suriša jas naudodamas savus žodžius.

Aukščiau parašytą sakinį irgi galėtume priskirti plagijatui, kadangi jis yra sutrumpinta šio teksto versija:

Originalas:

"This paraphrase is a patchwork composed of pieces in the original author's language (in red) and pieces in the student-writer's words, all rearranged into a new pattern, but with none of the borrowed pieces in quotation marks. Thus, even though the writer acknowledges the source of the material, the underlined phrases are falsely presented as the student's own." [8]

Galimai plagiatas:

"Nors pakeistame tekste ir yra minimas šaltinis, tačiau perpasakojant tekstą studentas naudoja frazes iš kito darbo ir tik suriša jas naudodamas savus žodžius."

Nors šis sakinys ir atitinka kai kuriuos plagiato kriterijus, tačiau nuorodos į šaltinį nėra būtina pateikti, nes jis gali būti priskirtas bendroms žinioms (angl. common knowledge).

Aukščiau pateiktas pavyzdys demonstruoja problemos sudėtingumą, kadangi net lyginant tik du tekstus tarpusavyje gali būti dviprasmybių nustatant ar tai plagiatas ar ne.

1.2 Plagiatų patikros programų veikimas

Šiuo metu yra sukurta daug įvairių plagiatų patikros programų. Keletas populiaresnių: Turnitin, WCopyfind, AntiPlagiarist, MyDropBox, CopyCatch, Eve, Findsame, Wordcheck, Viper, Crot...

Programas galima būtų suskirstyti į grupes, pagal tai, iš kur duomenys yra gaunami:

1. Tikrinamas darbas lyginamas su darbais esančiais tame pačiame kompiuteryje. T. y. tinka, kai reikia patikrinti ar darbe nėra panaudota informacija iš jau turimų darbų. Programos pavyzdys: Wcopyfind.
2. Lyginama su bendra duomenų baze esančia serveryje. Jos privalumas tame, kad šią duomenų bazę sudaro žymiai didesnis darbų kiekis, nes darbus įkelia daugiau nei vienas vartotojas. Programos pavyzdys: Viper.
3. Lyginama su paieškos sistemų turimais duomenimis. Nors nemažai darbų nėra paskelbiami internete, tačiau didžiausia tikimybė, kad darbas bus plagijuotas iš šaltinių rastų paieškos sistemų pagalba.

Plagiatų patikros programos dažniausiai naudoja kelis paieškos būdus (skirtingas duomenų bazes), nes kiekvienas darbų paieškos būdas turi savo privalumus.

Nemažiau svarbus yra ir paieškos algoritmas. Pvz.: Turnitin duomenų bazę 2011 metais sudarė 340 milijonų darbų ir 20 milijardų interneto svetainių puslapių. Vieno darbo patikrinimas vidutiniškai truko 13 s. [13]

Jei laikytume, kad 15 puslapių tekstinės informacijos užima 15 KB, tai vienas milijardas darbų užimtų apie 15000 GB (turnitin.com duomenų bazėje virš 2 milijardų dokumentų). Taigi naudojant paprastą žodžių palyginimo algoritmą negalėtume tokio duomenų kiekio apdoroti per protingą laiką. Todėl tinkamiausi tokiai užduočiai atlikti algoritmai yra indeksuojantys duomenis tokiu būdų, kad galima būtų vykdyti greitą paiešką tarp jų.

1.3 Moodle įskiepiai

Oficialioje Moodle svetainėje yra skelbiamos šios plagiatų patikros programos:

Crot – nemokama, atviro kodo.

Moss – nemokama. Mokama tik komerciniam naudojimui. Ši sistema skirta kodo plagijavimo paieškai.

Turnitin - mokama

URKUND - mokama [10]

Turnitin ir Urkund yra komercinės - mokamos programos. Moss – skirta tikrinti programiniam kodui. Crot – vienintelis nemokamas įskiepis siūlomas Moodle svetainėje.

Turnitin. Tikriausiai vienas iš populiariausių ir daugiausiai funkcijų turintis įrankis. Oficialiai kainos nėra skelbiamos, tačiau neoficialūs šaltiniai skelbia, kad yra taikomas $800 metinis mokestis + $1.50 už kiekvieną studentą ($2300 minimali užsakymo kaina). [7]

Urkund sistemos kainų apskaičiavimas šiek tiek skiriasi. 1 pav. Urkund įkainiai priklauso nuo studentų skaičiaus.

Number of pupils	Prices
< 99	675 EUR
100 - 199	790 EUR
200 - 299	860 EUR
300 - 399	925 EUR
400 - 599	1 030 EUR
600 - 799	1 155 EUR
800 - 999	1 260 EUR
1 000 - 1 299	1 525 EUR
1 300 - 1 599	1 785 EUR
1 600 - 1 899	1 995 EUR
1 900 - 2 199	2 230 EUR

For 2,200 or more pupils, please contact Urkund for a price.

1 pav. Urkund įkainiai. [14]

1.4 Plagiatų patikros e. sistemos Lietuvoje

Šiuo metu (2013 m.) nėra plačiai naudojamų lietuviškų plagiatų patikrai skirtų programų. Tačiau kai kurie universitetai jau pradeda diegti arba jau naudoja sistemas plagiatų aptikimui, pvz..: Vilniaus universitetas, Šiaulių universitetas, Mykolo Romerio universitetas, Žemės ūkio akademija. Pagrindiniai kriterijai stabdantys šių sistemų diegimą Lietuvos švietimo įstaigose – kaina ir plagiatų patikros sistemų integravimo sudėtingumas.

Kol yra ruošiama plagiatų patikros sistema, Šiaulių universitetas siūlo naudoti šias programas plagiatų aptikimui:

http://spore.vbi.vt.edu/dejavu/

http://www.articlechecker.com/

http://www.duplichecker.com/

http://www.plagium.com/

http://www.dustball.com/cs/plagiarism.checker/

http://www.plagiarismchecker.com/ [11]

Darbo rašymo metu, 2012m. rugsėjo mėnesį atsirado nauja lietuvių kurta plagiatų patikros sistema – plag.lt. Į ją įkelti darbai tikrinami tik su internete esančiais ir sistemos indeksuojamais dokumentais. Studentui išsamesni rezultatai apie jo darbo plagijavimo statusą yra mokami.

* Šiuo metu palaiko tik .docx formato dokumentus.
* Darbai tikrinami su internete esančiai dokumentais (delfi.lt, mokslai.lt...).
* Studentas nemato plagijavimo patikros rezultatų.
* Vieno darbo rezultatų peržiūra studentui kainuoja 3.69 - 5.46 lt.
* Darbas su programa vyksta per interneto naršyklę.
* Panašu, kad ieškoma tik identiškų tekstų atitikimų.
* Suindeksuota apie milijonas puslapių.
* Užregistruota dėstytojų: 1105. Studentų: 8655 (2013 m. sausio mėn.).

1.5 Plagiatų patikros e. sistemos užsienyje

Užsienyje pirmauja Turnitin, skelbianti, kad jų sistemą naudojo 56% iš 100 geriausių aukštąjį mokslą teikiančių įstaigų Amerikoje (2010 m.). [13]

Nors tiksli statistika nėra pateikiama kiek ir kuria sistema naudojasi vartotojai, tačiau iš kiekvienų kūrėjų svetainės pateiktų duomenų galima apytiksliai nuspėti kiek skiriasi skirtingų sistemų vartotojų skaičius.

	Duomenu bazė			Vartotojų skaičius				
	Interneto puslapių	Žurnalai ir publikacijos	Studentų darbai	Destytojų	Studentų	Įstaigų	Darbuotojų	Kainą už 1 studentą
urkund.com	10000 mln.		1.5 mln. (viso iki 2009 m.)					~$2
turnitin.com	24000 mln.	120 mln.	250 mln.	1 mln.	20 mln.	10000	150	
scanmyessay.com (viper)	10000 mln.		2 mln.		0.38 mln.		6	
ephorus.com						4000	30	$0.6 - $5
plag.lt	1 mln.			1110	8686	34		

2 pav. lentelėje pateikti duomenys, kurie yra pateikti programų kūrėjų svetainėse.

1.6 Algoritmų patikimumas

Kai kuriems plagiatų aptikimo algoritmams galima pakeisti tam tikrus parametrus, kurie įtakos įtartinų žodžių junginių aptikimo slenkstį (threshold). Pvz.: naudojant algoritmą kuris žodžių junginius apdoroja maišos funkcija (hash) ir sukuria tam tikrus kodus (fingerprints), kurie yra toliau lyginami, vienas iš kintamųjų galėtų būti skaičius, kuris nusako po kiek žodžių sudarys

vieną žodžių junginį. Nuo to priklausys kiek žodžių junginių bus pažymėti kaip pasikartojantys kituose darbuose.

Jei algoritmas bus per “jautrus”, tai pažymės didelę dalį teksto, kuris nėra plagiatas. Taip pat esant atvirkštinei situacijai, kai yra nepažymimas tekstas kuris yra plagijuotas. Abu atvejai yra ypač svarbūs programos vartotojams, nes esant didesniam klaidų skaičiui vartotojas pradeda nebepasitikėti programos pateiktais rezultatais, tai savo ruožtu priveda prie to, kad programa nebesinaudojama.

Tai yra labai panašu į tyrimą atliktą su žemės drebėjimu išankstinio nustatymo sistema. Tyrimas analizavo žmones kurie buvo iš anksto perspėti apie žemės drebėjimą. Tačiau jei šis pranešimas pasirodydavo klaidingas (žemės drebėjimas neįvykdavo), tai žmonės sekantį kartą gavę pranešimą į jį reaguodavo su mažesne tikimybe. [3]

Dėl to algoritmas turėtų pranešti apie plagijavimo atvejus, kai yra didelė tikimybė, jog šis spėjimas pasitvirtins.

2 Plagiato patikros e. sistemos projektavimas

Vienas iš pagrindinių keliamų reikalavimų kuriamai plagiatų patikros sistemai yra galimybė tikrinti tūkstančius darbų tarpusavyje. Kadangi kaskart įkėlus naują darbą yra netikslinga jį tikrinti su visais esančiais duomenų bazėje (užima per daug laiko), bus naudojamas algoritmas, kuris galės efektyviai vykdyti paiešką tarp indeksuotų darbų. Įkėlus darbą pirmą kartą jis bus suindeksuojamas. Vėlesni kitų darbų tikrinimai naudos šį indeksą, o originalas nebebus naudojamas.

Pagrindinis darbų surinkimas vyks per Moodle įskiepį. Jo paskirtis yra įkeltų studentų darbų siuntimas patikrinimui. Jį nuspręsta naudoti dėl to, kad nereikėtų dėstytojui ir studentui atlikti papildomų veiksmų darbo tikrinimui. Įrašius šį įskiepį į Moodle mokymosi aplinką, visas darbų įkėlimo procesas išlieka toks pat, tik šalia įkelto darbo plagiatų patikros įskiepis dar rodys galimai plagijuoto teksto dalį (įvertinimą procentais). Paspaudus ant jo galima bus peržiūrėti detalesnę informaciją.

Serverinė šios sistemos dalis bus atsakinga už surinktų darbų apdorojimą. Pagrindinės jos funkcijos:

- Darbo gavimas iš Moodle įskiepio ar per interneto svetainę.

- Originalaus darbo išsaugojimas ir konvertavimas į tekstą (plain text) iš įvairių formatų (.doc, .pdf…).
- Darbo indeksavimas ir įtraukimas į duomenų bazę.
- Galimai plagijuoto teksto paiešką, bei rezultatų pateikimas.

2.1 Įrankių ir priemonių pasirinkimo analizė

Kadangi Moodle yra parašyta PHP kalba ir dažniausiai naudoja MySQL duomenų bazę, todėl ir įskiepio kūrimui pasirinktas šis programų derinys.

Dėl lankstumo ir didelio kiekio papildomų bibliotekų serverinei programos daliai bus naudojami PHP ir MySQL. Duomenų perdavimui tarp įskiepio ir serverinės programos dalies pasirinkta cURL biblioteka, ji turi paruoštas naudojimui funkcijas failų perdavimui.

HTML ir CSS realaus laiko redagavimui, bei problemoms aptikti bus naudojamas Firefox įskiepis – Firebug, kuris tikriausiai šiuo metu yra vienas populiariausių, bei daugiausiai funkcijų turintis įrankis savo grupėje.

Programos veikimo testavimui, bei svetainės dizainui testuoti bus naudojamos labiausiai paplitusios naršyklės:

- Firefox (3.5 – 20 versijos)
- Internet Explorer (8 – 10 versijos)
- Chrome (21 – 26 versijos)

Kodo rašymui pasirinktas Notepad++, todėl, kad jis turi visų dažniausiai naudojamų programavimo kalbų sintaksės paryškinimą (nereikės naudoti keletos redaktorių). Taip pat kitas jo privalumas – nedidelis kompiuterio resursų naudojimas.

Dokumentų konvertavimui iš įvairių formatų bus naudojama:

- .doc - obninsk_doc biblioteka
- .docx ir .odt – DOMDocument biblioteka
- .pdf - PDF2Text biblioteka
- .rtf - rtf2text biblioteka

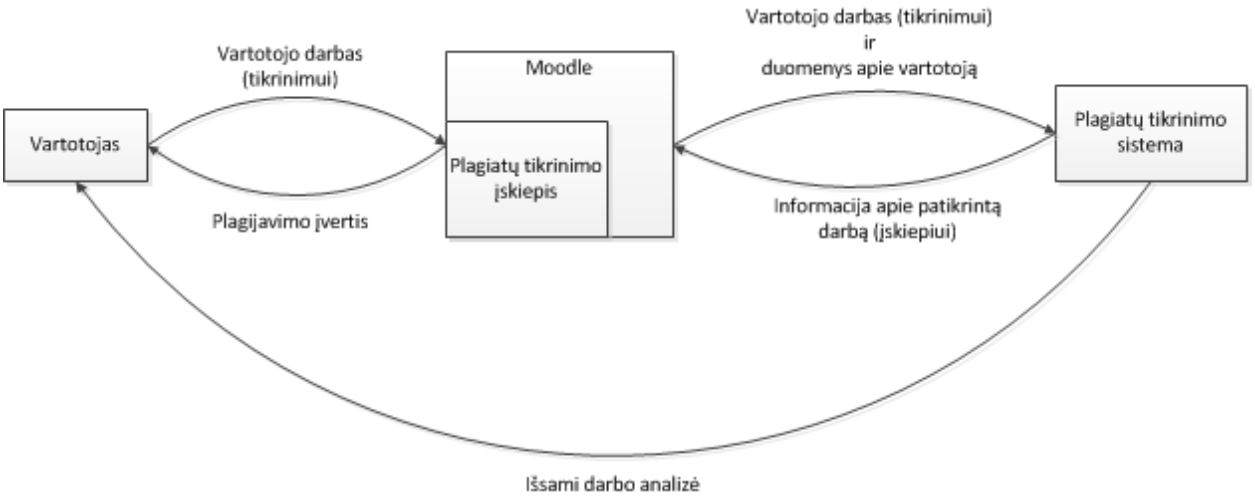
Darbo testavimui pasirinktas EasyPHP serveris. Jį sudaro Apache su įdiegtu PHP. Prie jo yra pridedamas MySQL, bei duomenų bazei skirtas valdyti įrankis – phpMyAdmin. Šio serverio privalumas yra tas, kad viskas yra sukonfigūruota ir paruošta darbui. Pvz.: iškaro įdiegti visi

pagrindiniai PHP moduliai. Taip pat EasyPHP stebi konfigūracinius failus. Jei juose yra atliekami pakeitimai, tai serveris yra automatiškai perkraunamas, kad įsigaliotų nauji nustatymai. Dar vienas šio serverio privalumas – galimybė jį perkelti į kitą kompiuterį paprasčiausiai nukopijavus failus.

2.2 Moodle įskiepis

Plagiato tikrinimo programa yra atskirta nuo Moodle įskiepio. Pasirinkus tokį sprendimą, plagiatų tikrinimo programa atlieka visus skaičiavimus ir siunčia duomenis į Moodle sistemą. Šio sprendimo privalumai yra tame, kad galima įdiegti plagiatų aptikimo įskiepį į kelias Moodle sistemas ir jos tarpusavyje turi bendrą duomenų bazę. Kitas privalumas – galimybė keisti pačios plagiatų aptikimo sistemos algoritmą nedarant pakeitimų Moodle įskiepiui (nereikia įrašinėti naujos įskiepio versijos Moodle aplinkai).

Viskas prasideda kai vartotojas įkelia savo darbą į Moodle sistemą (3 pav.). Įskiepis gavęs pranešimą (event) iš Moodle, įkeltą vartotojo dokumentą nusiunčia plagiatų tikrinimo sistemai. Taip pat iš Moodle gali būti siunčiami ir papildomi duomenys, kad vėliau galima būtų generuoti statistiką apie darbus. Plagiatų tikrinimo sistemai atlikus darbo analizę, rezultatai gražinami Moodle įskiepiui, kuris išsaugo gautus duomenis (kad nereikėtų daryti papildomų užklausų peržiūrint antrą syki tą patį darbą) ir rodo vartotojui rezultatą.

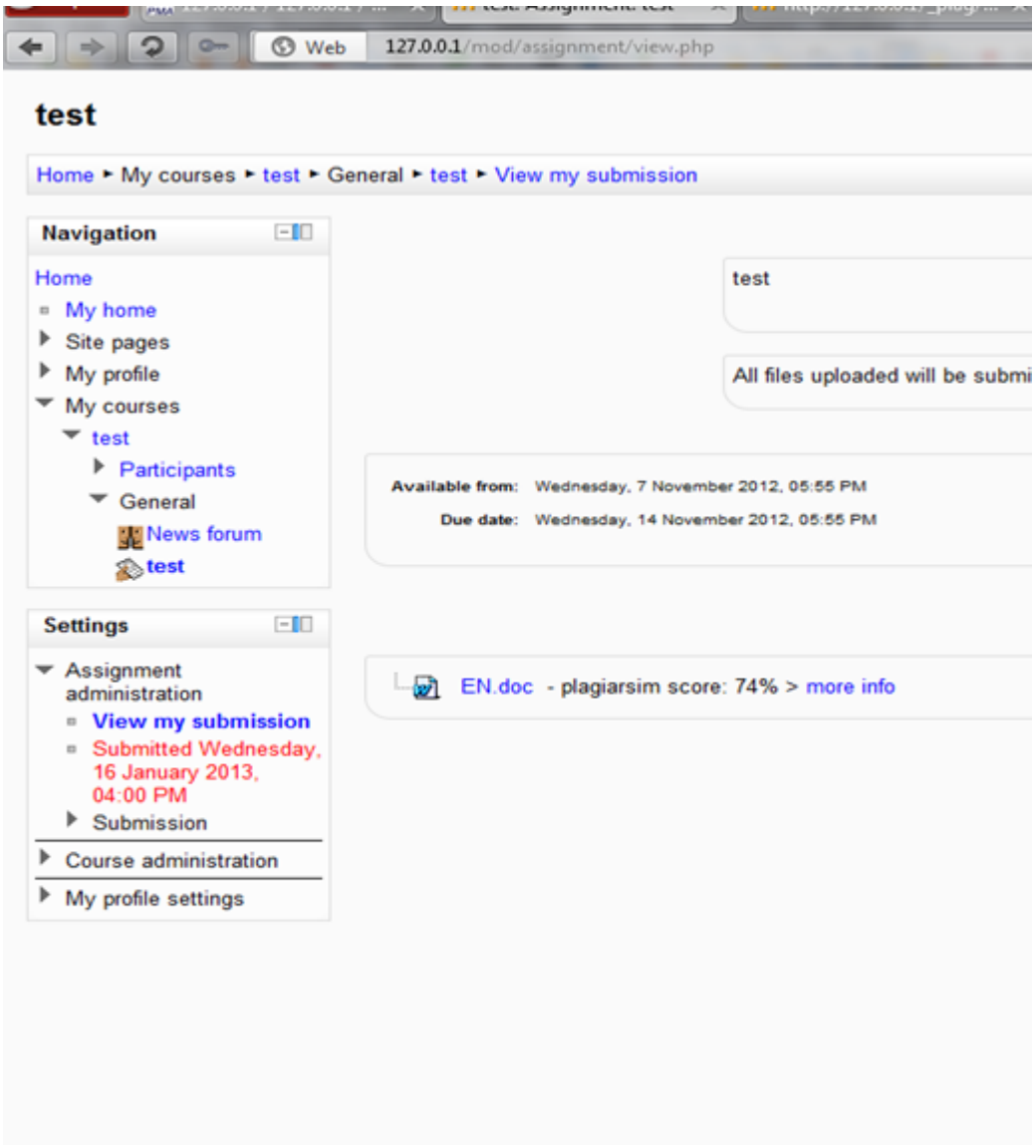


3 pav. Vartotojo sąveika per Moodle su plagiatų aptikimo sistema.

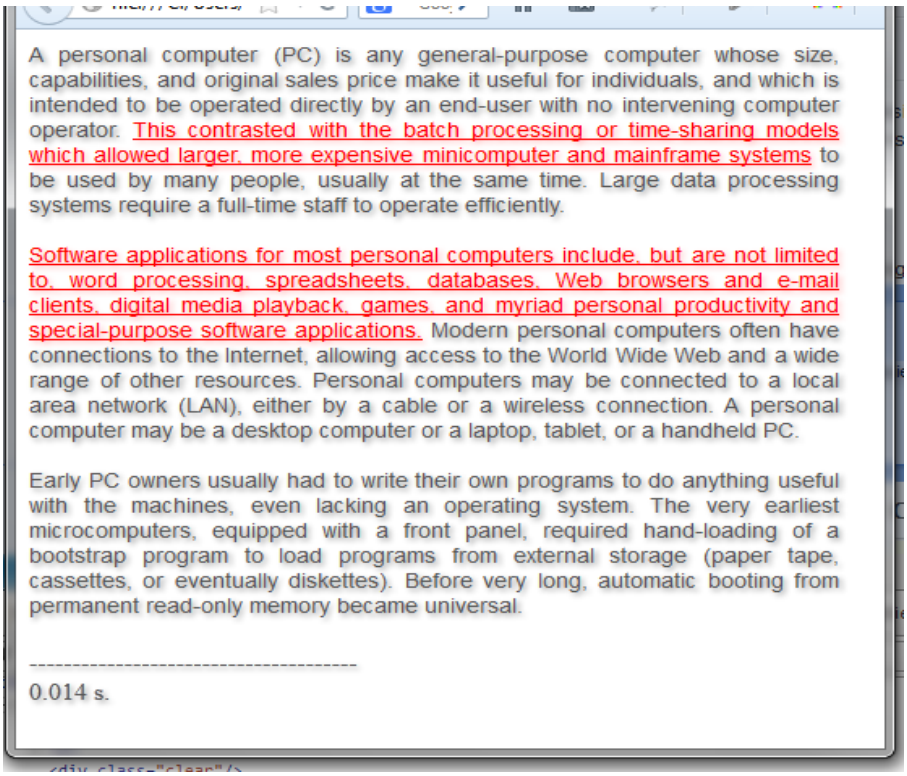
Moodle sistemoje plagiatų patikros įskiepiai yra talpinami į atskirą katalogą (“plagiarism”). Norint naudotis nauju Moodle įskiepiu reikia jį ne vien tik įrašyti, bet ir įjungti (enable). Kuriant plagiatų patikros įskiepį visas kodas yra rašomas tam tikrose funkcijose, kurias pati Moodle sistema iškviečia. Keletas funkcijų pvz., kurios gali būti iškviečiamos:

* public function get_links($linkarray)
* public function save_form_elements($data)
* public function print_disclosure($cmid)
* public function update_status($course, $cm)
* public function cron()
* function event_file_uploaded($eventdata)
* function event_files_done($eventdata)
* ...

Viena iš pagrindinių funkcijų, kuri atvaizduoja duomenis vartotojui, bei atlieka didžiąją įskiepio kodo daly yra – “public function get_links($linkarray)”. Šios funkcijos grąžintas tekstas ar HTML kodas yra rodomas prie studento prikabinto failo, kuris buvo patikrintas ar dar tik ruošiamas tikrinti plagiatų patikros įskiepio.



4 pav. kuriamas įskiepis rodys šalia studento įkelto dokumento vertę (procentais), kokia teksto dalis yra randama kituose dokumentuose. Taip pat papildoma nuorodą, kurią paspaudus galima peržiūrėti papildoma statistiką apie darbą.



5 pav. kopijuoto teksto paryškinimas pradinėje programos versijoje

Paspaudus nuorodą šalia darbo “more info”, galima matyti visą savo darbo tekstą, bei dalis, kurias plagiatų patikros algoritmas pažymėjo kaip galimai plagijuotus (5 pav.).

Kuriant Moodle įskiepį, reikėtų atkreipti dėmesį į tai, kad kai kurios Moodle funkcijos yra iškviečiamos tik CRON vykdymo metu (pvz.: event_file_uploaded – įvykis, kai naujas failas yra gautas Moodle sistemoje), o ne failo nusiuntimo metu, kaip galima tikėtis iš funkcijos pavadinimo.

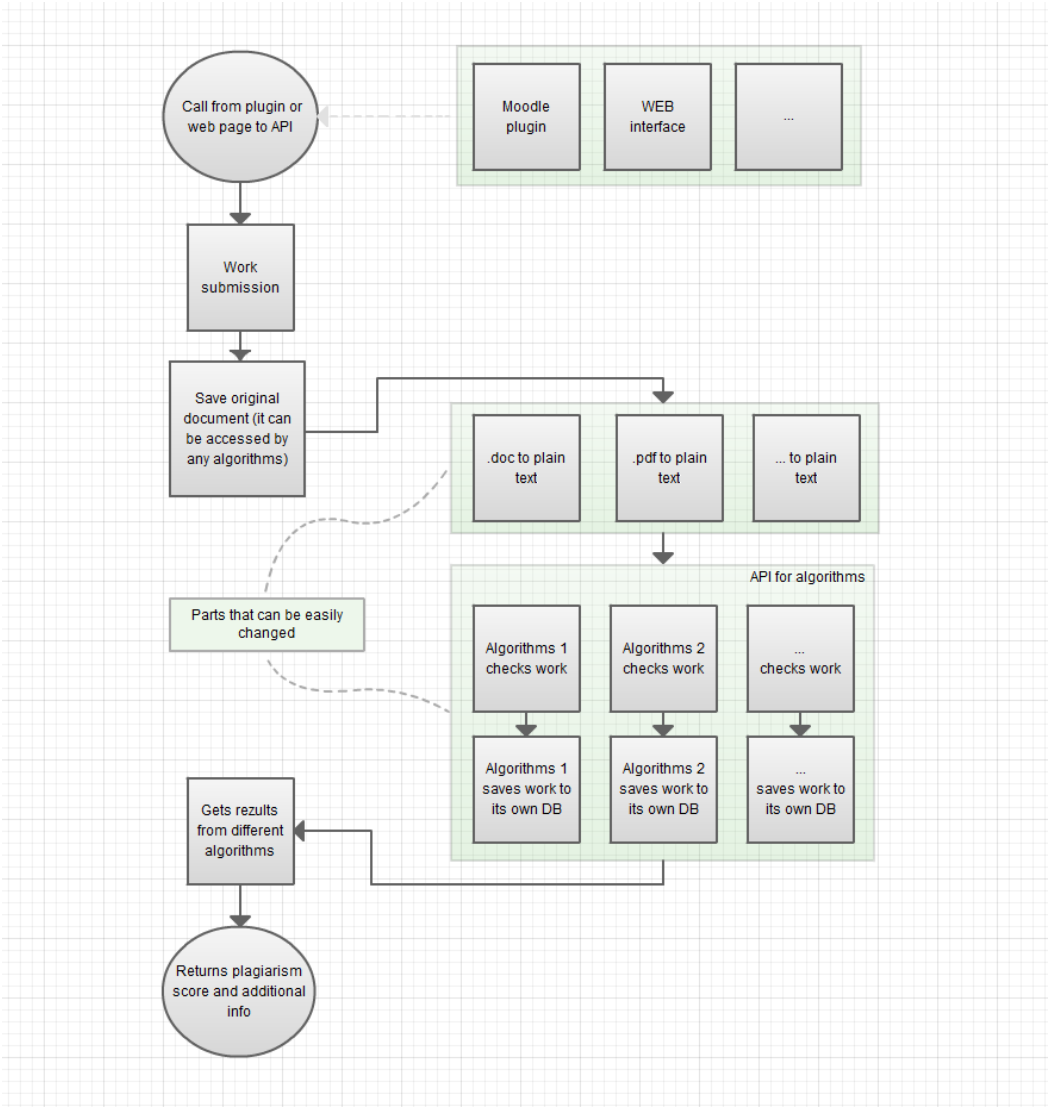
Taip pat nuo Moodle 2.0 versijos, failai yra saugomi Moodle failų saugykloje. Juos neįmanoma pasiekti įprastų būdu per HTTP užklausą. Prieigą prie failų dabar kontroliuoja Moodle. Todėl norint ką nors daryti plagiatų patikros įskiepyje tenka naudotis Moodle failų API.

2.3 Plagiatų aptikimo sistemos serverinė dalis

6 paveikslėlyje žaliai išskirtos dalys, kurių kodas yra parašytas tokiu būdu, kad jas keičiant nesutriktų visos programos veikimas.

Viršutinėje dalyje yra išskirtas API, kurio pagalba įskiepiai ir kitos programos gali bendrauti su sistema. Protokolo esmė – duomenų pateikimas per HTTP protokolą GET ir POST metodais. Tokiu būdu Moodle įskiepis gali pateikti studento parašytą failą, bei duomenis apie studentą, statistikos sudarymui.

Apatinėje dalyje yra išskirti plagiato paieškos algoritmai. Kiekvienas algoritmas yra klasė (OOP), kuri su sistema sąveikauja kai yra iškviečiamas tam tikras jos metodas. Duomenys taip pat yra gražinami baigus to metodo vykdymą (kaip funkcijos funkciniame programavime).



6 pav. Plagiato aptikimo sistema (serverinė dalis)

Norint panaudoti savo naują sukurtą algoritmą, reikia sukurti klasę ir joje aprašyti tris funkcijas, kurių pagalba algoritmas sąveikaus su sistema:

db_save($text, $file_id) – šią algoritmo funkciją plagiatų patikros sistema iškviečia tada, kai reikia suindeksuoti darbą ir išsaugoti duomenų bazėje. Kiekvienas algoritmo autorius gali pasirinkti pats kaip darbas bus indeksuojamas ir kokiu formatu išsaugomas.

$text – kintamajame yra pateikiamas darbo tekstas (plain text)

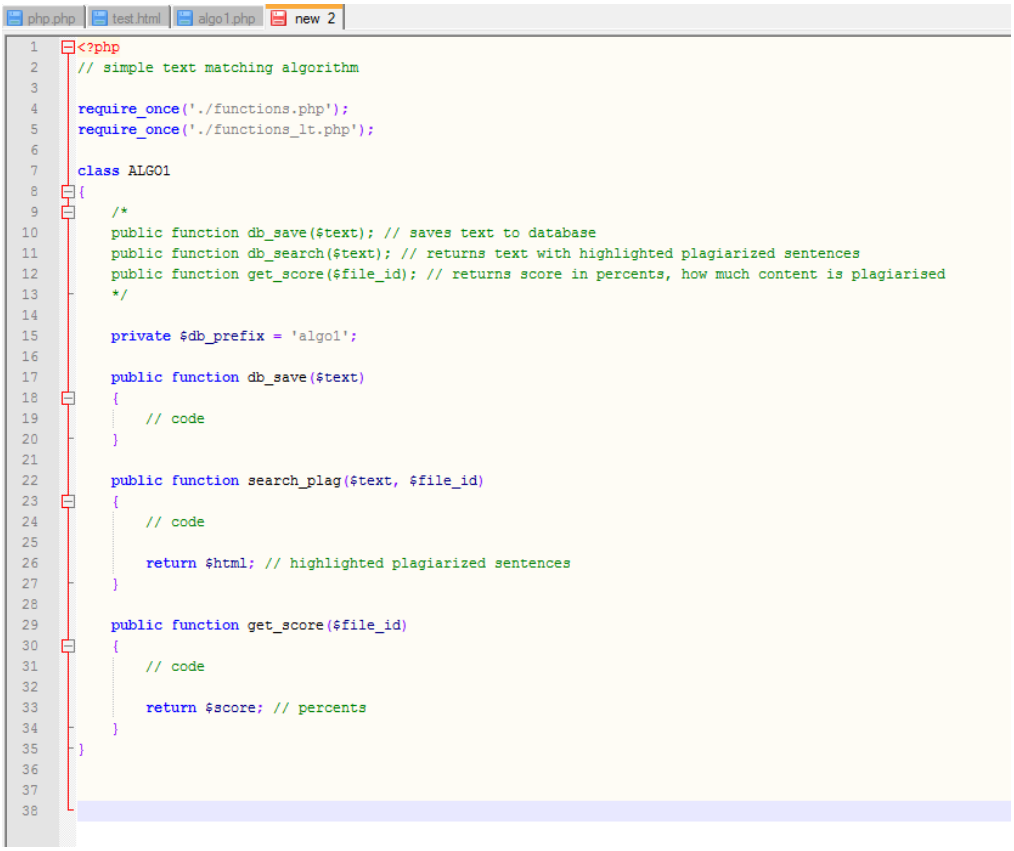
$file_id – unikalus darbo numeris (integer tipo)

search_plag($text, $file_id) – funkcija iškviečiama kai sistema gauna naują darbą tikrinimui. Šis metodas turi grąžinti skaitinę vertę procentais, kokia darbo teksto dalis yra rasta kituose šaltiniuose. Ji taip pat gali išsaugoti papildomus duomenis apie darbą, kurie yra naudingi studentui ar dėstytojui peržiūrint rezultatą (pvz.: paryškinamos vietos kuriose rasti teksto sutapimai).

get_score($file_id) – metodas iškviečiamas kai reikia gauti tam tikro darbo įvertį (procentais).

Kadangi kiekvienas paieškos algoritmas yra kaip atskira programa, todėl jam nėra keliama jokių papildomų reikalavimų sąveikajaunt su sistema. Algoritmo autorius gali pats rinktis kaip ir kokiu formatu nori indeksuoti ir išsaugoti darbą, kaip vykdyti paieška ir kt.

Pastaba: sistema pati išsaugo originalų atsiųsto darbo dokumentą. Kai algoritmas yra patobulinamas ar norima išbadyti kitą, visi prieš tai atsiųsti darbai gali būti perindeksuoti, kad veiktų su nauju algoritmu.



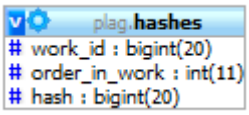
7 pav. Plagiato paieškos algoritmo klasės pavyzdys.

2.4 Duomenų saugojimas MySQL

Kuriant sistemą kuri lygina darbą su tūkstančiais ar milijonais kitų darbų, negalime kiekvieną kartą darbą lyginti su visa turima informacija. Pvz.: jei bandytumėme lyginti atsiųsto darbo kiekvieną sakinį su turimu milijonu įrašų, todėl tenka naudoti išankstinį indeksavimą ir algoritmą kuris panašius sakinius (galimai plagijuotus, bet pakeistus) paverstų tuo pačių kodu (hash). Teoriškai toks algoritmas turėtų vykdyti paiešką greičiausiai, kadangi užtektų rasti tik du sutampančius kodus (hash).

Duomenų saugojimas yra realizuotas SQL duomenų bazėje. Lentelės struktūra parodyta 8 pav. Joje yra saugoma teksto atkarpos pakeistos maišos algoritmu paimtos iš vartotojų darbų (angl. hash). Stulpelis, kuriame saugomos maišos algoritmo reikšmės, turi būti indeksuotas. Tokiu būdu vykdant paiešką nėra peržiūrimos visos reikšmės, o tik pagal indeksą randama reikalinga reikšmė. MyISAM duomenų basėje INTEGER tipo kintamiesiems naudojamas B – Tree indeksavimas. B – Tree indeksavimo vienas iš pagrindinių privalumų yra greita paieška. Tai yra

pasiekiama saugant kelias indekso reikšmes viename atminties bloke, ko pasekoje kietojo disko galvutė nuskaitydama duomenis (reikalingus paieškai) atlieka mažiau peršokimo operacijų (angl. seek).

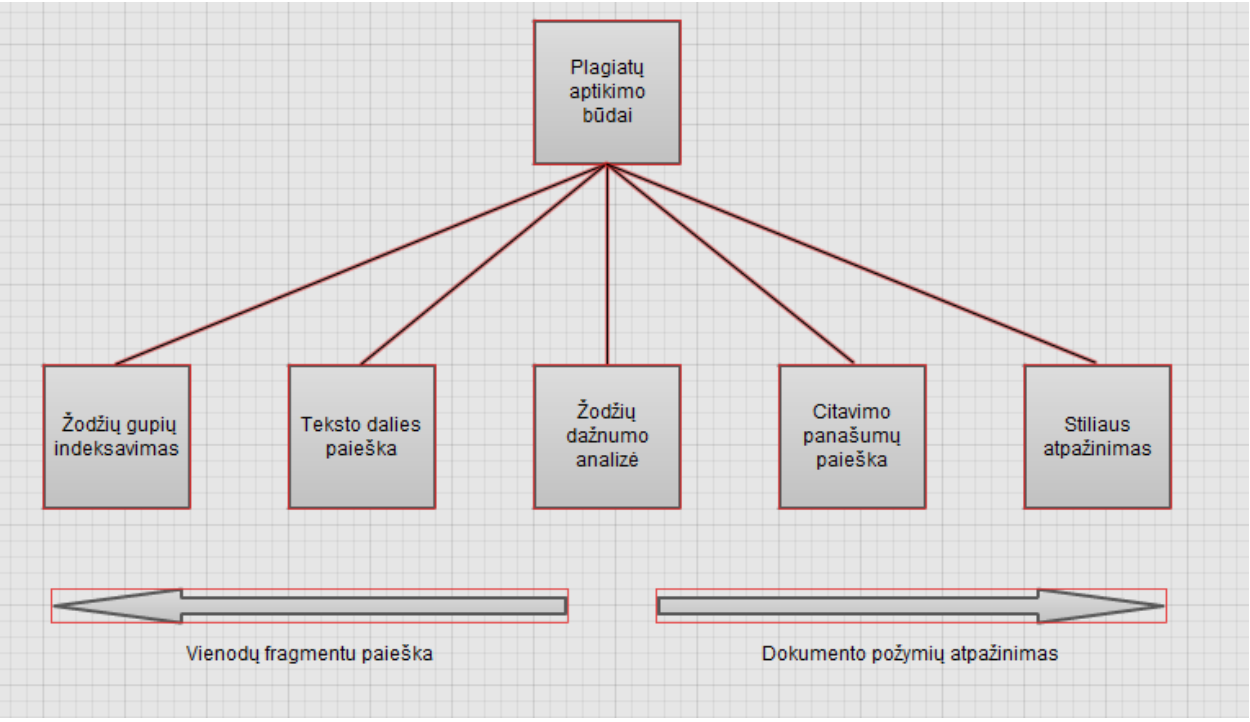


8 pav. Kodus (hash) saugančios ir daugiausiai užklausų sulaukiančios lentelės struktūra. "work_id" ir "hash" stulpeliai indeksuojami B – tree algoritmu.

Prieš naudojant maišos algoritmą tekstą reikia padalinti į atkarpas. Tekstą dalinti į atkarpas galima pagal skirtingus požymius, bet paprasčiausias – suskirstyti į tiek dalių, kiek yra sakinių. Tokio skirstymo trūkumas tame, kad naudojant paiešką pagal sugeneruotą maišos algoritmo kodą sistemą galima apgauti paprasčiausiai kitaip sudedant skyrybos ženklus.

2.5 Plagiatų paieškos algoritmai

Pagal naudojamą paieškos algoritmą sistemas galima būtu suskirstyti į kelias pagrindines grupes. 9 paveikslėlyje kairėje esantys algoritmai atpažįsta tik tikslų teksto atitikimą, o link dešinės einantys algoritmai naudojasi teksto požymiais. Kairėje esantys beveik visada randa kopijuotą tekstą. Dešinėje esantys algoritmai atvirkščiai, jų rezultatai nėra tokie patikimi, tačiau jie gali rasti ir sudėtingesnius plagijavimo atvejus.



9 pav. Plagiatų paieškos algoritmų tipai.

Žodžių grupių indeksavimas – šis algoritmas suskirsto tekstą į žodžių grupes ir ieško atitikimų duomenų bazėje. Pavyzdys: tekstas yra padalintas kas penktą žodį, algoritmas ieško tokių pačių penkių žodžių junginių kituose darbuose.

Teksto dalies paieška – algoritmas lygina tam tikras teksto dalis, kuriuose gali būti praleisti arba įterpti žodžiai. Paprasčiausiuose šio algoritmo realizacijose yra tikrinamas tekstas su kiekvienu jau esančiu duomenų bazėje. Didėjant darbų skaičiui proporcingai didėja ir paieškos laikas.

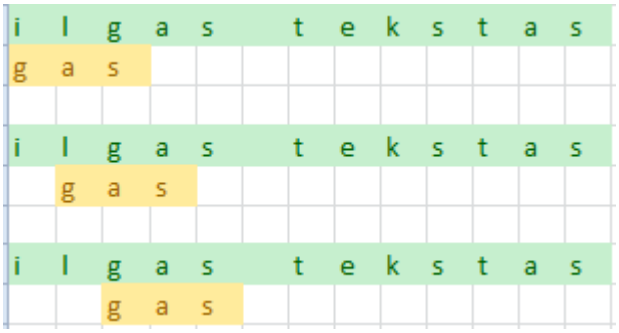
Žodžių dažnumo analizė – paprasčiausias pavyzdys, tai kai tekste ieškoma kiek kartų pasikartoja žodžiai. Kad sumažinti kitiems algoritmams paieškos erdvę, šis algoritmas gali būti naudojamas ne vien tik plagijuotų vietų paieškai, bet ir darbų skirstymui pagal tam tikras temas.

Citavimo panašumų paieška – kai darbai yra plagijuojami, plagiato autoriui tenka nurodyti ir šaltinius iš kur paimta informacija. Pastebėjus, kad skirtingi darbai naudoja tuos pačius šaltinius galima įtarti plagijavimo galimybę.

Stiliaus atpažinimas – šis algoritmas remiasi idėja, kad kiekvienas autorius tekstą rašo savaip: naudoja tam tikro ilgio sakinius, dažniau jo tekste pasitaiko kai kurie jungtukai ir terminai. Paprasčiausias algoritmo įgyvendinimas būtų suskirstyti visą darbą pastraipomis ir pagal tam tikrus požymius ieškoti panašiai parašytų pastraipų kituose darbuose. Taip galima identifikuoti ar nėra plagijuotos pastraipos iš kitų informacijos šaltinių.

2.6 Teksto paieškos algoritmai

Pats primityviausias būdas aptikti tam tikrą raidžių kombinacija tekste – lyginti tą raidžių kombinaciją pradedant nuo pirmos ir baigiant paskutiniu teksto simboliu.

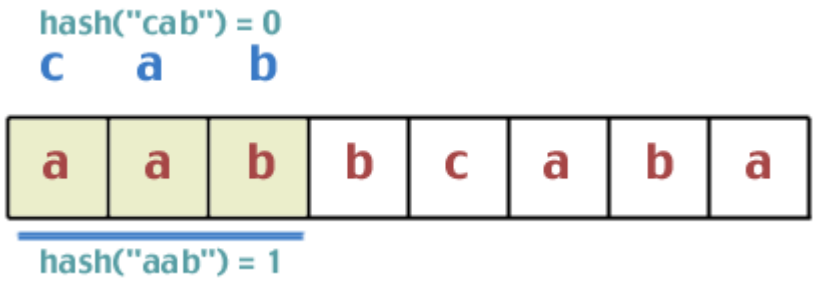


10 pav. Teksto paieška kitame tekste.

10 pav. ieškomas tekstas pažymėtas geltona spalva yra stumiamas dešinėn, kol yra randamas atitikmuo. Šis paieškos būdas yra pats lėčiausias, kadangi jei ieškoma raidžių kombinacija yra “aaaaaaaaaaaab”, o ieškoma tekste “aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaab”, tai norint patikrinti ar pirmasis tekstas kartojasi antrajame, teks tikrinti visas pirmojo teksto raides.

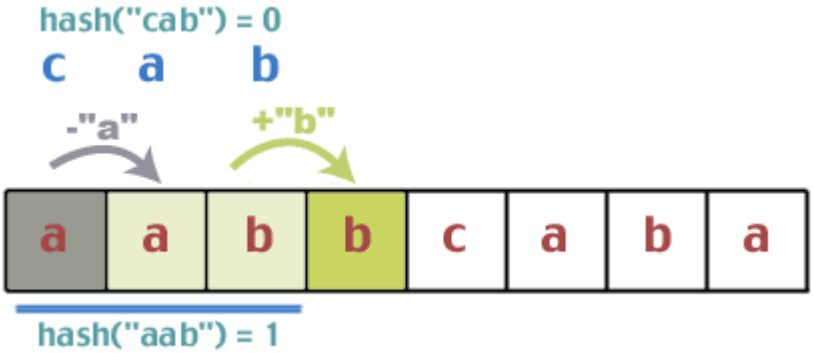
Šiek tiek greičiau veikiantis algoritmas - “Rabin–Karp“. Jo modifikacijos yra naudojamos plagiatų patikros sistemose, pvz.: Jplag, YAP3. Jo esmė simbolių grupių apdorojimas maišos algoritmu (hash). Michael O. Rabin atrado, kad „hash“ reikšmę įmanoma apskaičiuoti per pastovų laiką. [5]

Jei kiekvienai raidei priskirtume skaičių tokiu būdu: a – 1, b - 2, c – 3... , o „hash“ funkcija būtų skaičių sumos šaknis iš trijų, tai cab „hash“ reikšmė būtų lygi 0. aab „hash“ reikšmė lygi 1. Kadangi 0 nelygu 1, tai tekstai cab ir aab yra skirtingi.



11 pav. „Hash“ funkcijos rezultato pavyzdys.

Visas algoritmo greičio privalumas yra „hash“ funkcijos skaičiavime. Kad apskaičiuoti sekančią „hash“ reikšmę (12 pav.), užtenka iš turimo „hash“ reikšmės (11 pav.) atimti nebenaudojamą pirmąją raidę „a“ ir prie sumos pridėti naują raidę „b“.



12 pav. „Hash“ funkcijos rezultato pavyzdys antrajame žingsnyje.

Taip skaičiuojant “hash” visą laiką reikės vienodo skaičiaus aritmetinių operacijų. Tai žymiai sutrumpia paieškos laiką, jei tekste yra pasikartojančių, bet ne visiškai vienodų simbolių sekų. [2] [12]

Kita svarbi sistemos dalis – duomenų saugojimas. Jei sistemoje yra laikomi tūkstančiai darbų, jie užima nemažai vietos. Taip pat ir saugant duomenų indeksus, juos galima suspausti. Geriausiai tam tinka algoritmai kurie nepraranda informacijos (lossless compression). Vienas iš geriausiai žinomų algoritmų yra – LZ77, sukurtas 1977 metais. Remiantis šiuo algoritmu buvo sukurti kiti algoritmai: LZW, LZSS, LZMA ir kiti. LZ77 algoritmo patobulintos versijos yra naudojamos GIF ir PNG paveikslėlių formatuose. [1]

2.7 Plagiato aptikimas naudojant interneto paieškos sistemas

Plagiatų paiešką naudojantis interneto paieškos sistemomis dažniausiai naudoja programos, kurios atrenka tik keletą sakinių iš darbo ir juos patikrina. Arba tai yra kaip programos papildymas, kuri tikrina naudodama savo algoritmą, bet papildomai įtartinas vietas patikrina ir paieškos sistemose.

Nors šis būdas ir turi daug privalumų, tačiau didžioji dalis paieškos sistemų apmokestina automatinį užklausų siuntimą. Automatinės užklausos yra siunčiamos per specialų API (application programming interface). Dviejų populiariausių paieškos sistemų įkainiai už paiešką:

Google – 100 užklausų per diena nemokamai. Jei norima siųsti didesnį užklausų kiekį - $5 už 1000 užklausų ($1 – 200 užklausų).

Bing – leidžia atlikti apie 166 užklausas nemokamai per dieną. Pigiausias planas suteikia 500 užklausų už $1. Tai yra beveik dvigubai daugiau nei Google.

Užklausų kiekis (vnt.)	Kaina (doleriais)
5000	0
10000	20
20000	40
50000	100
100000	200
250000	500
500000	1000
1000000	2000
1500000	3000
2000000	4000
2500000	5000

Lentele 1. Bing užklausų kaina.

Jeigu laikytumėme, kad viename referato darbe yra vidutiniškai apie 250 sakinių, tai tikrinant kiekvieną sakinį Bing paieškos sistemoje (kaip atskirą užklausą), kainuotų apie 139 litus ($0.2 už užklausą * 250 užklausų * 2.78 dolerio kursas = 139 Lt).

3 Plagiato patikros sistemos kūrimas

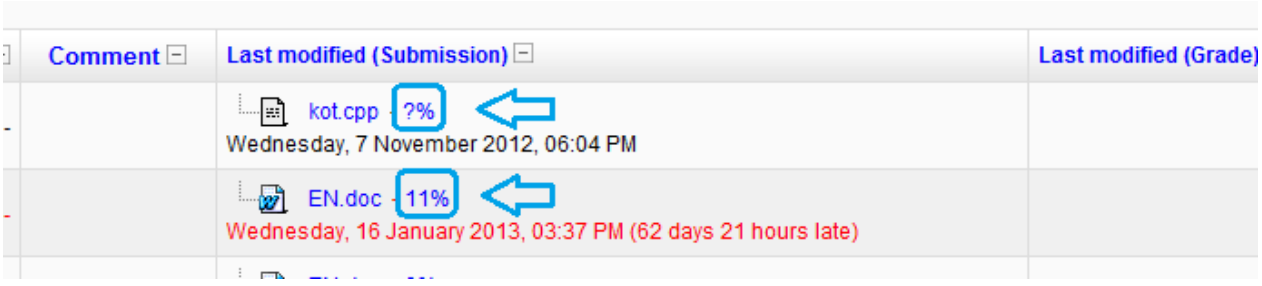
3.1 Moodle įskiepis

Sukurtas Moodle įskiepis veikia Moodle 2.0 ir naujesnėse versijose. Senesnės Moodle versijos saugo failus kitokiu būdų.

Pagrindinė problema kuriant įskiepį buvo failo perdavimas. Kadangi skirtingose Moodle sistemose failų katalogas gali būti ne toje pačioje vietoje, reikėjo naudoti Moodle File API. Failo persiuntimui buvo pasirinkta cURL biblioteka, kuriai užtenka nurodyti kelią iki failo ir serverio adresą kur bus siunčiamas failas, o ši biblioteka rūpinasi tolimesniu failo siuntimu.

Kad kiekvieną kartą peržiūrint darbą Moodle sistemoje nebūtų kreipiamasi į serverį ir rezultatai būtų pateikiami greičiau, Moodle įskiepis gautą rezultatą išsaugo vietinėje Moodle duomenų bazėje. Sekantį kartą peržiūrint tą patį darbą duomenys yra imami iš vietinės duomenų bazės.

Kuriant plagiatų patikros įskiepį galima naudoti Moodle dokumentacijoje pateiktą pavyzdį (template). Nors jame ir yra pateiktos visos pagrindinės funkcijos reikalingos sėkmingam sukurto įskiepio veikimui, tačiau pavyzdžio pasirinktas pavadinimas yra netinkamas - "new". Kadangi norint pakeisti įskiepio pavadinimą į savo, galima būtų paleisti paiešką, kuri automatiškai visuose failuose tam tikra žodį pakeistų į norimą. Tačiau to negalima padaryti su Moodle pasirinktu pavadinimu. PHP programavimo kalboje (taip pat ir kitose), žodis “new” yra rezervuotas kalbos reikmėms, pvz.: kuriant naują objektą. Todėl bandant pakeisti šį žodį yra sugadinamas ir kodas (pvz.: objekto kūrimo). Dėl to tenka kiekvieną failą peržiūrėti atskirai, kas tokią paprastą užduotį kaip pavadinimo keitimas padaro gana ilgu ir nuobodžiu procesu. Todėl geriau būtų buvę pasirinkti pavadinimą, kuris niekur daugiau nebūtų sutinkamas kode, pavyzdžiui “repme” (replace me).



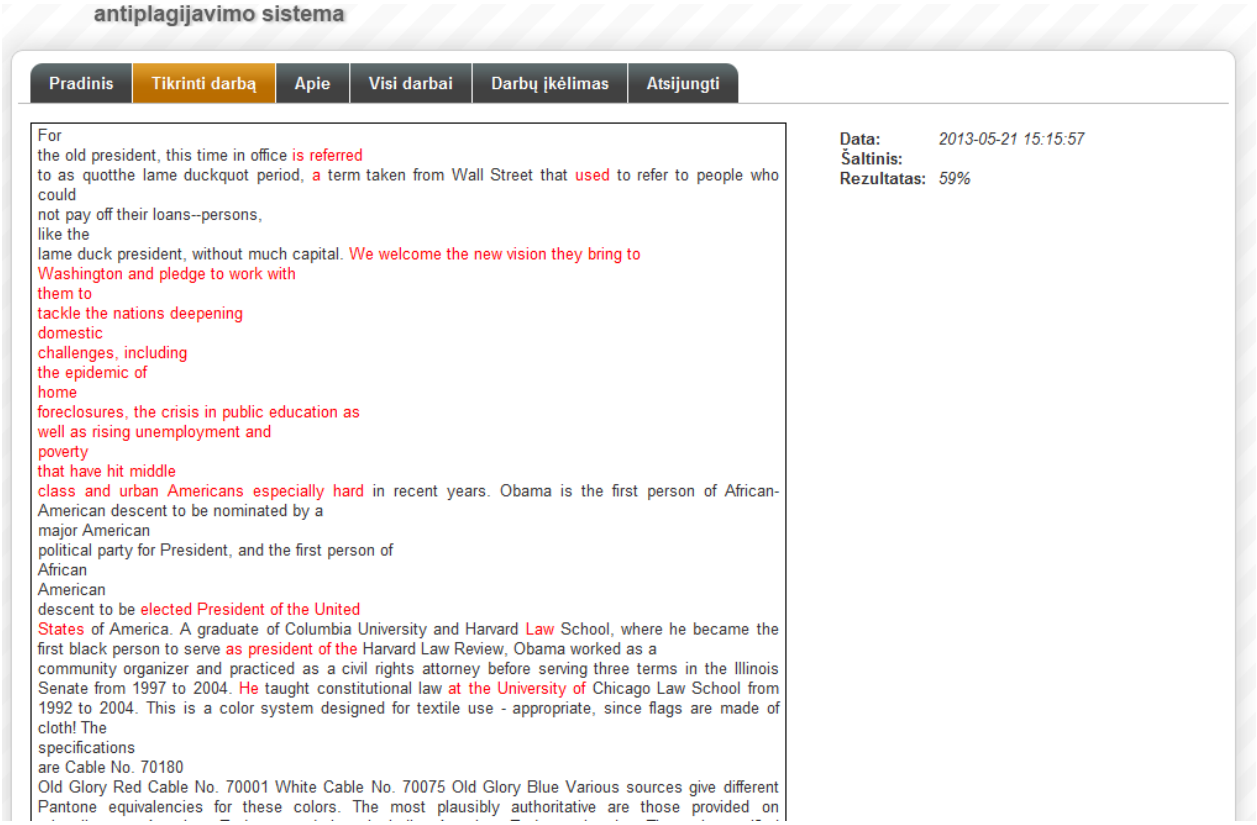
13 pav. Plagiatų patikros sistemos grąžintas rezultatas atvaizduojamas Moodle.

Vienintelė vieta kur galima pastebėti šio plagiatų patikros įskiepio veikimą – Moodle aplinkoje šalia įkeltų dokumentų (13 pav.). Pradžioje šalia jų buvo rodomi ne vien tik patikros rezultatai, bet ir klaidų pranešimai. Tačiau taip yra bereikalingai apkraunama vartotojo sąsaja. Dėl to buvo pasirinktas tik trumpų pranešimų rodymas. Dabar yra rodomas arba patikros rezultatas, jei patikra buvo sėkminga, arba klaustuko ženklas ant kurio vartotojas paspaudęs gali sužinoti išsamesnę informaciją (kodėl darbas nebuvo patikrintas ir įvertintas).

Paspaudus šias nuorodas vartotojas yra nukreipiamas į plagiatų patikros sistemos tinklalapį, kuriame jau yra rodomas rezultatas (14 pav.). Dažniausiai klaidos pranešimas yra dėl nepalaikomo failo formato:

14 pav. Palaikomi tekstinių failų formatai.

Paspaudus darbo įvertinimo nuorodą naujai atsidariusiame lange rodomas visas darbo tekstas (15pav.). Tekstas, kuris yra sutinkamas kituose darbuose, yra pažymėtas raudonai. Dešinėje rodoma kita papildoma informacija:

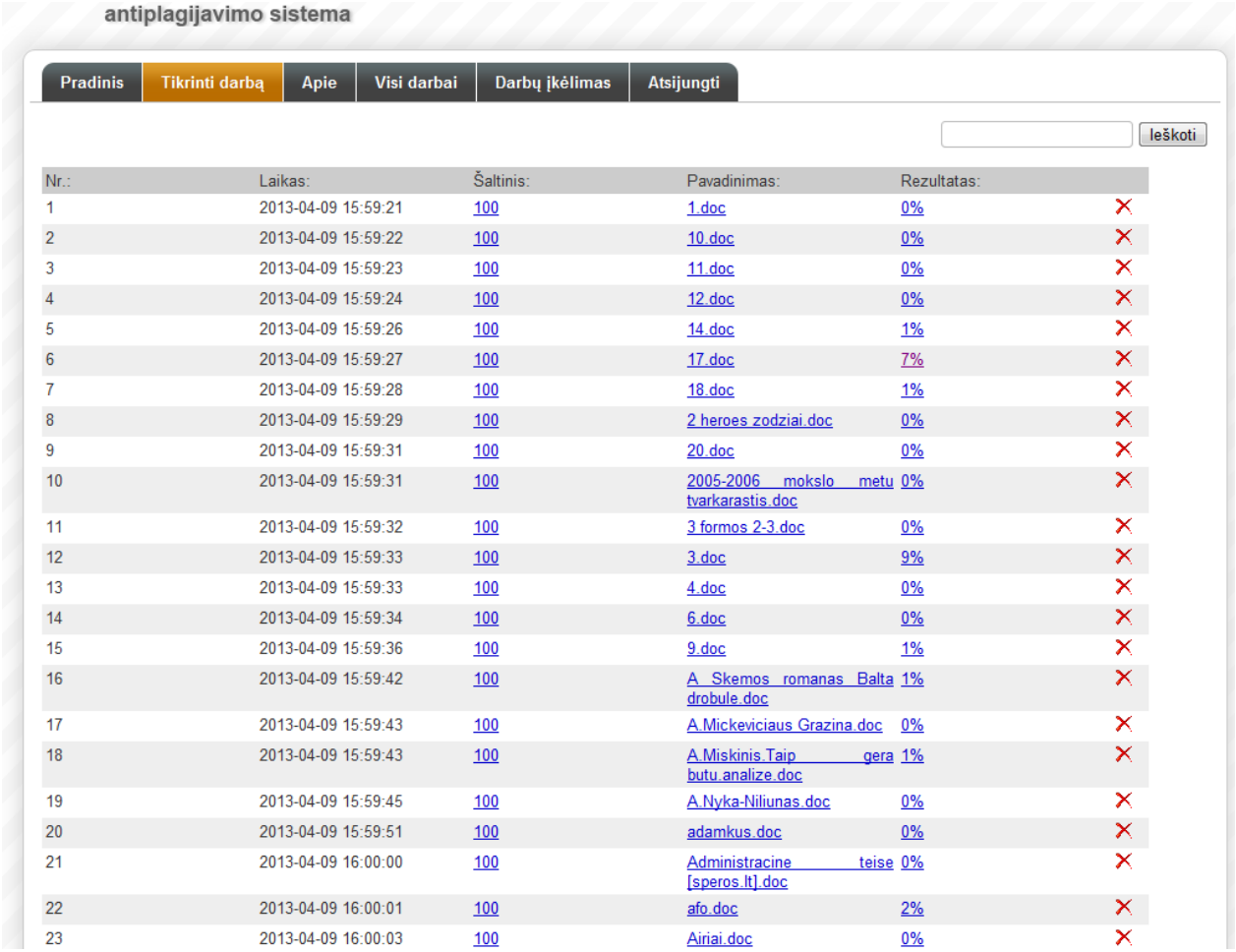


15 pav. Rasto plagijuoto teksto pavyzdys.

3.2 Serverinė programos dalis

Šios sistemos dalies pagrindinė funkciją – organizuoti duomenų keitimąsi tarp kitų sistemos komponentų: algoritmo, Moodle įskiepio, dokumentų konvertavimo, vartotojo sąsajos.

Serverinė programos dalis dokumentus gali gauti ne vien tik iš Moodle įskiepių, tačiau ir įkeltus per interneto svetainę. Darbus įkelti per interneto svetainę gali pats studentas. Tokiu būdu jis gali tikrinti tik po vieną darbą. Prisijungusiam prie sistemos vartotojui (administratoriui) yra padaryta galimybė importuoti didelius darbų kiekius suarchyvuotus .zip formatu (16 pav.). Jis gali įkelti per ankstesnius metus sukauptą darbų archyvą. Kita funkcija kuri yra pasiekiama tik administratoriui – visų įkeltų darbų peržiūrą, bei galimybė pašalinti iš sistemos:



16 pav. Prisijungusiam prie sistemos administratoriui rodomi visi įkelti darbai.

Įkeliant darbą studentui, jis gali pasirinkti po kiek dienų sistemą turėtų jį suindeksuoti (Moodle sistemos administratorius gali tai pasirinkti įskiepio nustatymuose). Ši funkcija reikalinga tam, kad įkeliant tą patį darbą jis nebūtų pažymėtas kaip plagiatas.

3.3 Plagiatų paieškos algoritmas

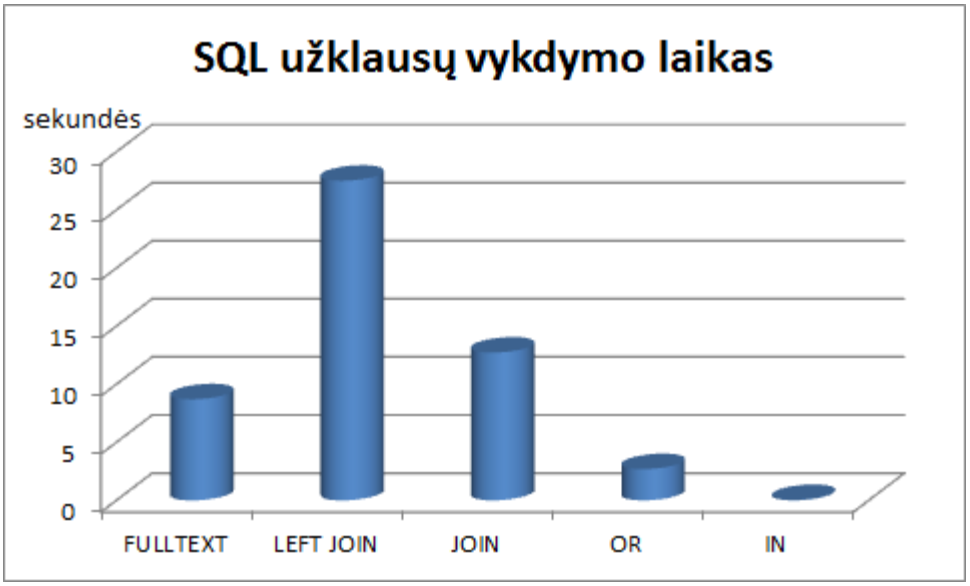
Vienas iš pagrindinių kriterijų naudojamam algoritmui buvo veikimo greitis. Kadangi duomenų bazėje bus saugomi tūkstančiai darbų tarp kurių kurių reikės vykdyti paiešką, buvo orientuotasi į minimalų vieno darbo tikrinimo laiką.

Iš įvairių algoritmo rūšių buvo pasirinktas “fingerprinting” metodas. Jo privalumas, kad iš tam tikrų teksto atkarų yra sugeneruojami “fingerprints” tik vieną syki. Toliau jie yra naudojami lyginti darbams tarpusavyje. [4]

“Figerprints” saugojimui buvo pasirinkta MySQL duomenų bazė dėl galimybės duomenis indeksuoti ir tokiu būdų vykdyti greitą paiešką tarp jų. Pirmiausiai buvo išbandytas įrašų įrašymo greitis į duomenų bazę. Kiekvienas įrašas buvo įrašomas siunčiant MySQL atskirą komanda:
INSERT INTO test_db (ID, www) VALUES ('', 'random text')

MySQL duomenų bazė gali naudoti skirtingus variklius duomenų saugojimui ir paieškai. Pagrindiniai iš jų: MyISAM ir InnoDB.
Bandymo metu buvo įrašoma po 100 įrašų naudojant skirtingus duomenų saugojimo varikliukus:
0.011 s. – MEMORY
0.013 s – MyISAM
8.65 s – InnoDB

InnoDB lėtas įrašymo greitis yra dėl palaikomo ACID (Atomicity, Consistency, Isolation, Durability) standarto. Jo esmė – duomenų saugumas. Kiekvienos operacijos duomenys iškart yra įrašomi į kietąjį diską, taip pat kuriami yra “log” failai duomenų atstatymui, pavyzdžiui jei dingtu elektra duomenų rašymo metu. Kadangi plagiatų patikros sistema neturi kritinių kodo vietų, nuo kurių veikimo priklauso sistemos darbas, tai InnoDB šiai sistemai netinka.
MEMORY – duomenų bazė yra saugoma tik kompiuterio RAM atmintyje ir tinka tik laikinam duomenų saugojimui. Perkrovus serverį per naują, visi duomenys buvę MEMORY duomenų bazėje dingsta. Šio tipo duomenų bazės pagrindinis privalumas - greitas operacijų vykdymas.
MyISAM duomenų bazė turi FULLTEXT indeksą, skirtą teksto paieškai straipsniuose. Naudojantis šiuo indeksu galima visą straipsnio tekstą įrašyti kaip vieną MySQL lentelės reikšmę ir toliau naudojantis šiuo indeksu greitai rasti reikiamą tekstą. Be FULLTEX indekso tektų naudoti tokią MySQL užklausą:
SELECT * FROM tekstai WHERE tekstas LIKE “%zodziu junginys%”
Šis užklausos tipas yra vienas iš lėčiausių, kadangi vykdant paiešką būtų peržiūrima kiekviena lentelės eilutė. Taip pat kiekvienos eilutės tekstiniame laukelyje būtų ieškoma nurodyta reikšmė.



17 pav. SELECT vykdymo greitis priklausomai nuo naudojamas SQL komandos struktūros.

17 pav. Pavaizduota SELECT komandos vykdymo trukmė skiriasi keliasdešimt kartų tarp lėčiausiai vykdytos paieškos naudojant LEFT JOIN ir greičiausiai – IN. Pasirodo, kad norint išrinkti kelias reikšmes iš lentelės su indeksuotu stulpeliu greičiausiai tai yra atliekama naudojant tokią komandą:

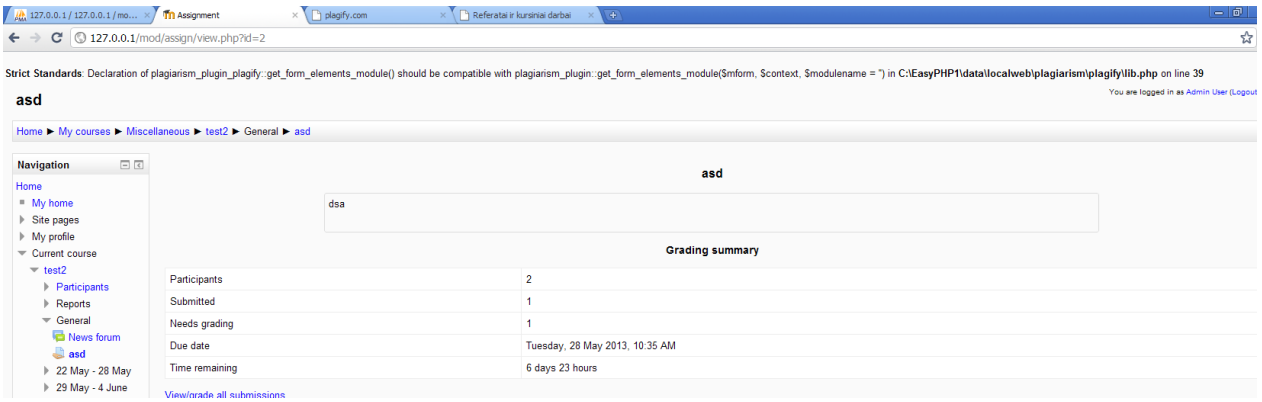
SELECT * FROM lentele WHERE reikšmė IN (reikšmė1, reikšmė2, reikšmė3, reikšmė4, ...)

4 Sistemos testavimas

4.1 Įskiepio palaikomos Moodle versijos

Įskiepis buvo kuriamas ir testuojamas su Moodle 2.3 versija. Joje visos įskiepio funkcijos veikia.

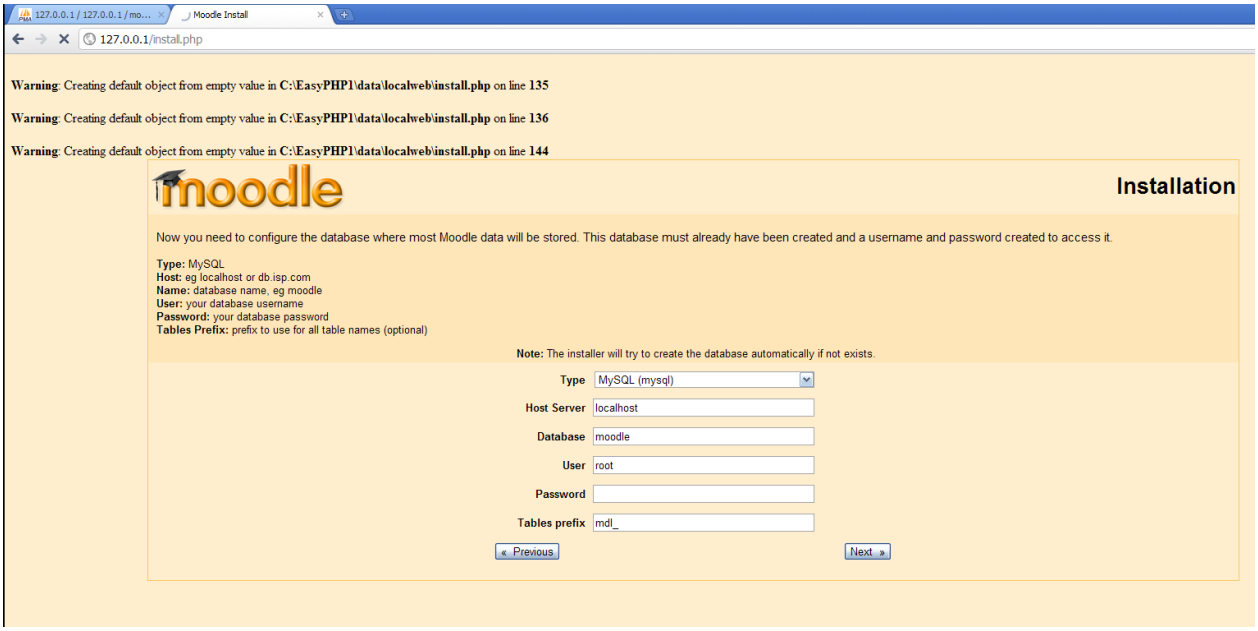
Išbandžius įskiepį Moodle 2.4 versijoje, taip pat įskiepis veikė, tačiau Moodle rodė perspėjimą (18 pav.), kad pasikeitė vienos iš jos funkcijų aprašymas.



18 pav. Klaidos pranešimas naujesnėje Moodle versijoje, kai buvo įdiegtas plagiatų patikros įskiepis.

Pakeitus funkcijos aprašymą, klaidos pranešimas neberodomas nei vienoje Moodle versijoje.

Su Moodle 1.9 versijos įrašymu kilo problemų, kadangi Moodle kodas turėjo klaidų (PHP 5.4 versija). Serverio nustatymuose (php.ini) klaidų rodymo nustatymai neturėjo įtakos klaidų rodymui, kadangi Moodle kūrėjai specialiai instaliaciniame faile įjungė klaidų rodymą, o tos klaidos neleido įrašyti Moodle mokymosi aplinkos. Dėl to teko pataisyti Moodle kodą, kad nebūtų rodomos klaidos (19 pav.). Tačiau instaliacijos procesas vis tiek pastrigdavo.



19 pav. Klaidos pranešimai diegiant Moodle 1.9 į serverį naudojantį PHP 5.4.

Naudojant PHP 5.2 versija Moodle 1.9 įsirašė be klaidų. Tačiau dėl kitokios failų struktūros, plagiatų patikros įskiepis Moodle 1.9 versijoje neveikia.

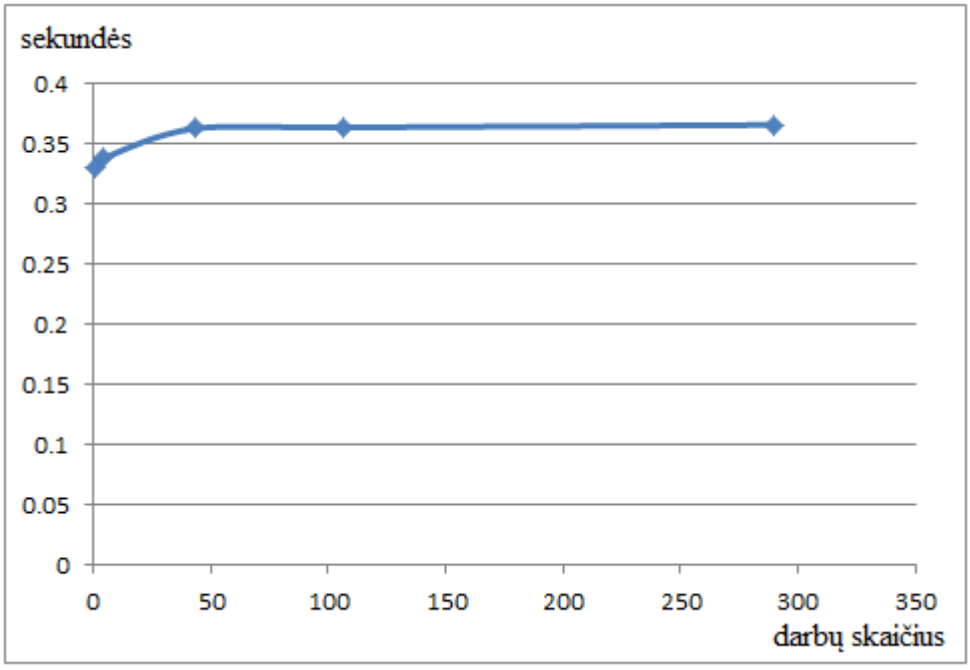
Testuojant Moodle 2.5 versija nepavyko nustatyti į kurį serverį kreiptis plagiatų patikros įskiepiui, tai buvo dėl to, kad Moodle 2.5 neišsaugodavo įskiepio nustatymų.

Moodle versija	1.9	2.3	2.4	2.5
Ar veikia sukurtas įskiepis?	ne	taip	taip	ne (neišsisaugo įskiepio nustatymai)

Lentelė 2. Įskiepio veikimas skirtingose Moodle versijose.

4.2 Paieškos algoritmo veikimo sparta

Darbo testavimui buvo pasirinktas bakalauro darbas, kurio apimtis 3083 žodžiai (24 puslapiai). Buvo tiriama kaip kinta plagiatų paieškos algoritmo vykdymo laikas priklausomai nuo darbų skaičiaus duomenų bazėje (20 pav. ir lentelė nr. 3). Duomenų bazė buvo sudaroma iš referatų ir panašaus pobūdžio darbų skelbiamų internete.

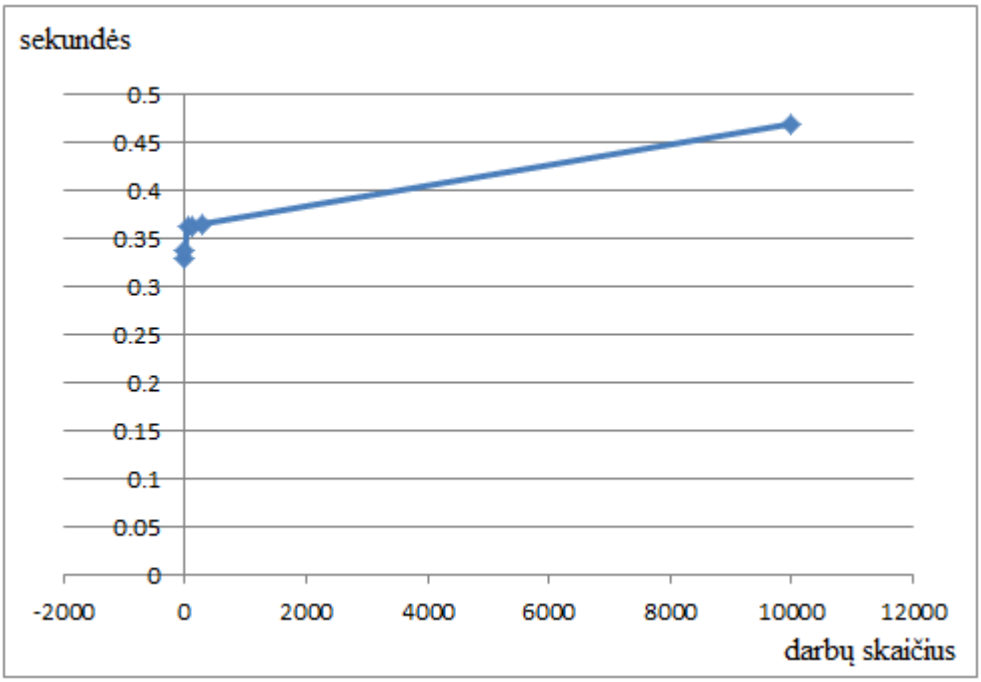


20 pav. Vieno darbo tikrinimo laikas priklausomai nuo esančio darbų skaičiaus duomenų bazėje.

darbų sk.	1	4	43	106	290
sekundės	0.33	0.338	0.363	0.364	0.366

Lentelė 3. Vieno darbo tikrinimo laikas priklausomai nuo esančio darbų skaičiaus duomenų bazėje (tie patys duomenys kurie yra pavaizduoti aukščiau esančiame grafike)

Pagal aukščiau pateiktus empirinius duomenis galime pastebėti, kad dvigubėjant saugomų darbų skaičiui, darbo tikrinimo laikas kinta tik keliomis tūkstantosiomis sekundės.
Ekstrapoliavus duomenis su Excel programos funkcija TREND, paieška tarp 10000 darbų gali trukti apie 0.47 sekundės.



21 pav. Ekstrapoliuoti plagiato aptikimo patikros trukmės duomenys

Galima teigti, kad didėjant darbų skaičiui saugomam duomenų bazėje, vieno darbo patikros laikas kinta nežymiai.

4.3 Sistemos diegimas plag.distance.su.lt serveryje

Diegiant plagiatų patikros sistemą į plag.distance.su.lt serverį su didesnėmis problemomis nebuvo susidurta, kadangi sistema naudoja standartines PHP bibliotekas, kurios praktiškai visuose serveriuose yra įdiegtos iš anksto.

Testuojant įdiegtą sistemą su įvairiais darbais, buvo pastebėta, kad keliant tam tikrą darbą kurio dydis apie 8 MB, PHP skriptui pritrūksta atminties (RAM). Tai įvyksta dėl naudojamos bibliotekos konvertuojančios .doc formato failą į tekstą. Kad nepasitaikytų tokių atvejų ateityje, buvo sumažintas įkeliamų failų dydis iki 5 MB (22 pav.). Daug užimančius darbus galima vis tiek patikrinti nukopijavus jų tekstą į pradiniame puslapyje pateikiama formą.

22 pav. Klaidos pranešimas kai bandoma įkelti per didelį failą. Taip pat pateikiama trumpa instruksija kaip patikrinti tokį darbą.

4.4 Kurtos plagiatų patikros sistemos palyginimas su kitomis programomis

Programos pavadinimas	Crot	Viper	Kurta plagiatų patikros sistema	WCopyfind	AntiPlagiarist 2.6
Lietuvių kalbos palaikymas	taip	taip	taip	ne	ne
Tikrinimas su darbais esančiais tame pačiame kompiuteryje	ne	taip	ne	taip	taip
Tikrinimas su darbais esančiais serveryje	taip	taip	taip	ne	ne
Tikrinimas naudojantis paieškos sistemomis	taip	taip	ne	ne	taip
1000 darbų indeksacijos laikas	darbus galima įkelti tik po vieną	90 s	428 s*	nėra darbų indeksacijos	nėra darbų indeksacijos
Paieška tarp 1000 darbų	-	<1 s	0.38 s*	3.16 s	209 s
Paieška tarp 2000 darbų	-	<1 s	0.39 s*	6.19 s	-
Pastabos	neįmanoma atlikti darbų tikrinimo palyginimo nes nėra galimybės kelti darbus ne po vieną	~2 s, paieška tarp 10000 darbų			
Kaina	nemokama	nemokama (galima nusipirkti papildomų paslaugų)	-	nemokama	mokama (galima nemokamai išbandyti)

Lentelė 4. Plagiatų patikros programų palyginimas (- ekstrapoliuoti duomenys)*

Pagrindiniai sukurtos plagiatų patikros sistemos privalumai lyginant su kitomis programomis (lentelė 4), tai gana greita paieška, Moodle mokymosi aplinkos palaikymas, žinoma kur ir kokie

duomenys yra saugomi. Tačiau lyginant su Viper programa, pastarosios darbų indeksavimas vykdomas beveik 5 kartus greičiau. Viena iš pagrindinių to priežasčių, kad kurta plagiatų patikros sistema yra parašyta PHP kalba.

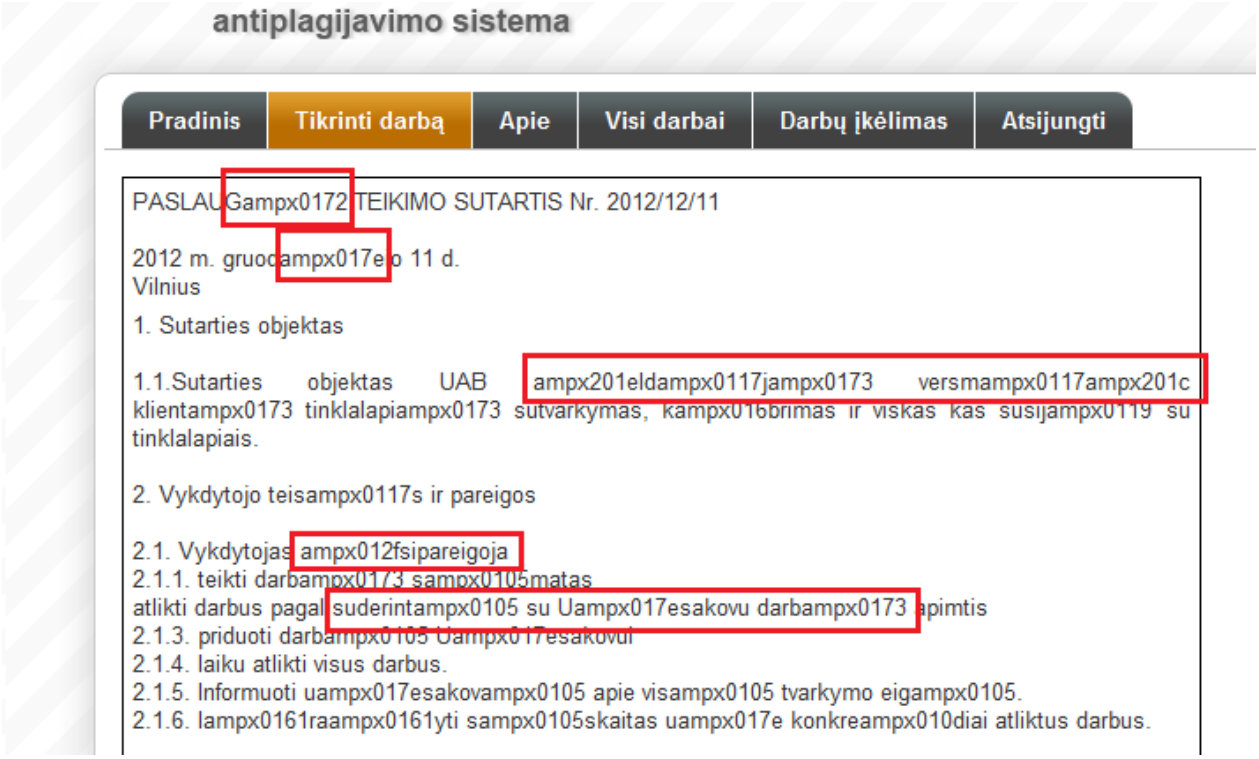
4.5 Dokumentų konvertavimas

Dokumentų konvertavimui buvo specialiai pasirinktos bibliotekos parašytos PHP kalba, kad nereikėtų keisti standartinių Apache ir PHP nustatymų, nes vėliau atnaujinant serverio programinę įranga gali kilti problemų su plagiatų patikros sistemos nustatymų derinimu.

Testuojant buvo pastebėta, kad ne visi dokumentai yra teisingai konvertuojami iš .doc formato - kai kurių dokumentų naudojamai bibliotekai konvertuoti nepavykdavo.

Testavimui buvo pasirinkta populiaraus Moodle plagiatų patikros įskiepio (Crot) bibliotekos. Jos taip pat yra parašytos PHP kalba. Išbandžius su jomis tuos pačius dokumentus, joms irgi nepavyko gauti juose esančio teksto.

Taip pat buvo pastebėta, kad konvertuojant dokumentus Crot biblioteka lietuviški simboliai yra pakeičiami HTML atitikmenimis (HTML Entities).



23 pav. Konvertuojant tekstą su Crot naudojama biblioteka lietuviškos raidės yra pakeičiamos HTML atitikmenimis.

4.6 Patarimai, pastebėjimai, rekomendacijos

Kadangi plagiatų patikros sistemos pagrindinės funkcijos yra sukurtos, galima būtų ateityje susitelkti dėmesį į algoritmo tobulinimą. Yra trys pagrindinės kryptis, kuriomis galima būtų tai atlikti:

1) patobulinti plagiatų aptikimo būdą

2) sumažinti dokumentų indekso dydį

3) pagreitinti paiešką ir indeksavimą

Užimamą duomenų bazėje indekso dydi galima sumažinti saugant viename baite daugiau nei vieną simbolį. Šiuo metu duomenų bazėje yra įrašomi tik lotyniškos abėcėles simboliai. Kadangi nėra įrašomi kableliai, taškai ir kiti simboliai, tai vienam atvaizduojamam ženklui galima būtų skirti 5 bitus.

Kitas būdas, tai naudoti maišos funkciją (hash), kuri indeksuojamų žodžių junginius paverstų į trumpesnę simbolių seką.

Paieškai ir indeksavimui didžiausią įtaką turėtų turėti kritinių kodo dalių perrašymas C kalba, kadangi ją yra rašomos PHP kalbai moduliai, tai perrašytas kodo dalis galima būtų prijungti kaip PHP modulius.

Ne tokia svarbi, tačiau irgi galinti pagreitinti sistemos veikimą dalis – dokumentų konvertavimas į tekstą. Galima būtų vietoje PHP kalba parašytų bibliotekų naudoti tam pritaikytas programas.

Išvados

1) Atlikus plagiatų aptikimo e. sistemų analizę paaiškėjo, kad nėra programų kurios galėtų tikrinti darbus tarp kelių Moodle mokymosi aplinkų ir būtų nemokamos.

2) Išnagrinėjus plagiatų paieškos algoritmus (skirtus dideliam darbų kiekiui tikrinti) tinkamiausias pasirodė „fingerprints“ naudojantis algoritmas.

3) Sukurta plagiatų patikros sistema. Tobulinant jos plagiatų paieškos algoritmą buvo testuoti skirtingi duomenų saugojimo būdai MySQL duomenų bazėje. Taip pat buvo sukurtas Moodle įskiepis, kuris integruoja plagiatų aptikimo sistemą į Moodle mokymosi aplinką.

4) Pagal 5.2 skyriuje atlikto eksperimento rezultatus galima teigti, jog dvigubėjant darbų skaičiui duomenų bazėje, vieno darbo patikros laikas kinta nežymiai (didėja tik keliomis tūkstantosiomis sekundės).

5) Lietuvoje kaip ir užsienyje turėtų paplisti plagiatų patikros sistemų naudojimas, tai savo ruožtu turėtų paskatinti Lietuvos kalbai pritaikytų programų didesnio skaičiaus atsiradimą.

Literatūros ir informacinių šaltinių sąrašas

1. Jacob Ziv ir Abraham Lempel. A universal algorithm for sequential data compression. Transactions on information theory, Vol. IT-23, NO. 3. 1977 metai.
2. Karp, Richard M. Efficient randomized pattern-matching algorithms. IBM Journal of Research and Development. Vol. 31, issue: 2, pages 249 – 260. 1987 metai.
3. L. Erwin Atwood ir Ann Marie Major. Exploring the „Cry Wolf" hypothesis. The Pennsylvania State University. 1996 metai. Prieiga per internetą: <http://www.ijmed.org/articles/337/download/>. Žiūrėta 2013 m. vasario 11 d.
4. Louis A. Bloomfield. How WCopyfind and Copyfind Work. 2011 metai. Prieiga per internetą: <http://plagiarism.bloomfieldmedia.com/How_WCopyfind_and_Copyfind_Work.pdf>. Žiūrėta 2013 m kovo 6d.
5. Michael J. Wise. YAP3: Improved Detection Of Similarities In Computer Program And Other Texts. 1996 metai. Prieiga per internetą: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.49.947>. Žiūrėta 2013 m vasario 9 d.
6. Nevinskaitė L., Trumpulytė S., et al. Etd dokumentų plagijavimo patikros galimybių studija. Vilniaus universitetas. 2008 metai. Prieiga per internetą: <http://senas.labt.lt/naujienos/ETD_dokumentu_plagijavimo_patikros_galimybiu_studija.pdf >. Žiūrėta 2013 m kovo 6 d.
7. Academic Integrity. Plagiarism and Related Issues. 2006 metai. Prieiga per internetą: <http://www.hpcnet.org/peru/facultysenate/academicintegrity>. Žiūrėta 2013 m balandžio 11 d.
8. Avoiding Plagiarism. Prieiga per internetą: <http://writing.wisc.edu/Handbook/QPA_paraphrase.html>. Žiūrėta 2013 m gegužės 3 d.
9. Bing Search API prices. Prieiga per internetą: <https://datamarket.azure.com/dataset/5BA839F1-12CE-4CCE-BF57-A49D98D29A44>. Žiūrėta 2012 m gruodžio 15 d.

10. Moodle plagiarism prevention plugins. Prieiga per internetą: <https://moodle.org/plugins/browse.php?list=category&id=35>. Žiūrėta 2013 m. kovo 8 d.
11. Šiaulių universiteto rekomenduojamos antiplagijavimo sistemos. Prieiga per internetą: <http://distance.su.lt/?page_id=197>. Žiūrėta 2013 m balandžio 2 d.
12. SparkNotes Editors. SparkNote on Hash Tables. Prieiga per internetą: <http://www.sparknotes.com/cs/searching/hashtables/>. Žiūrėta 2013 m gegužės 4 d.
13. Turnitin comany information. Prieiga per internetą: <http://turnitin.com/en_us/about-us/our-company>. Žiūrėta 2013 m sausio 24 d.
14. Urkund product prices. 2010 metai. Prieiga per internetą: <http://ebookbrowse.com/urkund-products-prices-eur-pdf-d44645856>. Žiūrėta 2013 m kovo 16 d.

Anotacija

Tema: „Plagiato patikros e. sistema Moodle aplinkai“.

Baigiamajame magistro darbe nagrinėjamos plagiatų aptikimas, vertinami skirtingi algoritmai, užsienio šalių patirtis šioje srityje. Išnagrinėti pagrindiniai būdai, kuriais remiantis nustatomi plagiatai. Darbo metu sukurta plagiatų patikros sistema ir įskiepis skirtas Moodle mokymosi aplinkai. Sistemoje naudojamo algoritmo efektyvumas matuojamas atsižvelgiant į duomenų saugojimo, bei gavimo būdus iš duomenų bazės, taip pat ir jo patikimumas. Baigiamajame darbe nagrinėjama kitų plagiatų patikros programų tikrinamų darbų skaičiaus įtaka jų darbo efektyvumui, bei lyginama su sukurtu produktu.

Summary

Topic: „Plagiarism detection system for Moodle“.

Main goal of this work was to create antiplagiarism system for Moodle. Before system development every aspect of it was researched. Hardest thing to design was algorithm for plagiarism detection, because it must be quite fast and accurate. After that plagiarism detection system was developed in PHP and SQL languages there were performed different speed tests. Created algorithm's one work processing time almost doesn't depend on work count in database, so this system is limited only by the amount of RAM and HDD space server has.

Priedai

1. Dalyvavimo tarptautinėje moklinėje konferencijoje pažymėjimas

KAUNO KOLEGIJA

PAŽYMĖJIMAS

Nr.____34-523_____

Mantas Donelavičius

2012 gruodžio 12 d. dalyvavo respublikinėje mokslinėje – praktinėje konferencijoje

INFORMACINIŲ TECHNOLOGIJŲ TAIKYMAS ŠVIETIMO SISTEMOJE 2012: E-STUDIJŲ PATIRTIS, AKTUALIJOS IR PERSPEKTYVOS

KAUNO KOLEGIJA

Direktorius dr. Mindaugas Misiūnas

Kaunas
2012 12 12

24 pav. Dalyvavimo tarptautinėje moklinėje konferencijoje pažymėjimas.

2. Dalyvavimo tarptautinėje moklinėje konferencijoje sertifikatas.

ŠIAULIŲ UNIVERSITETAS
TECHNOLOGIJOS FAKULTETAS

SERTIFIKATAS

Reg. Nr. TFP-27

2013 gegužės 15 d.
Šiauliai

Mantas Donelavičius

dalyvavo
Šiaulių universiteto Technologijos fakulteto 8-oje Tarptautinėje mokslinėje konferencijoje „Jaunųjų mokslininkų darbai“ ir skaitė pranešimą tema

Plagiato patikros e. sistema Moodle aplinkai

Dekanas dr. Sergėjus Rimovskis

Organizacinio komiteto pirmininkas dr. Nerijus Ramanauskas

25 pav. Dalyvavimo tarptautinėje moklinėje konferencijoje sertifikatas.