SISTEM OTOMATISASI PENGELOLAAN LABORATORIUM UNTUK PENILAIAN PRAKTIKUM PEMROGRAMAN DASAR DENGAN DETEKSI PLAGIARISME

Dewa Ade Andrea, Wijanarto

Abstract – The number of students who attend programming classes make this management becomes severe because the professors had to examine one by one the collected source code. Beside the similarity tasks that is collected is possible to occur. Professors do not know who is doing the action of plagiarism so he give same value on the task that proved same. It is very detrimental to the student who has been working on his own and benefit students who commit plagiarism. With automatic assestment and detection plagiarism based on SIM (based on program structure) with improved methods of Coding Model Algorithm (based on the writing of student program) can help automatically make an assestment and know the action of plagiarism also help professors in management especially assestment management. Is expected to provide an assestment in accordance with has been done by students

Keywords : Automation systems, laboratory management, plagiarism detection, coding model algorithm.

I. PENDAHULUAN

Dalam mempelajari ilmu komputer di butuhkan pengetahuan yang luas. Tidak hanya mengetahui teori tetapi bagaimana cara mahasiswa dalam menerapkan teori tersebut ke dalam praktiknya. Menurut [1] Seorang mahasiswa informatika harus mendapatkan pengetahuan teoritis dan praktek pemrograman dalam beberapa paradigma agar sudut pandang mahasiswa tidak sempit.¬¬ Untuk mendapatkan penguasaan pemrograman yang baik tidak mudah, harus dengan menulis banyak program dan itu dilakukan dari yang sederhana menuju ke kompleks [2]. Pada awal pembelajaran tidak bisa dijadikan acuan seorang mahasiswa dinyatakan baik atau tidak dalam menguasai pemrograman. Menurut [3], kemampuan mahasiswa di tahun pertama tidak

sesuai dengan harapan dosen dimana kegagalannya disebabkan oleh mereka sendiri.

Di kelas pemrograman, mahasiswa memiliki peran dimana harus menyelesaikan tugas pemrograman yang di berikan di kelas maupun di rumah secara tepat. Untuk mengetahui ketepatan suatu program harus melalui serangkaian proses percobaan pembangkitan secara urut [4]. Setiap tugas yang masuk harus di koreksi kemudian dilakukan penilaian. Banyaknya mahasiswa yang mengikuti kelas pemrograman, akan menyita waktu dosen dalam memberikan nilai dan memungkinkan terjadinya plagiarisme tugas pemrograman. Plagiarisme di kalangan mahasiswa masih menjadi masalah terbesar sampai saat ini di universitas seluruh dunia [5]. Plagiarisme merupakan tindakan menirukan atau menyalin atau menggunakan kreasi atau ide seseorang dan menyajikan hal tersebut adalah miliknya [6]. Dosen akan merasa kesulitan untuk menemukan siapa yang mengerjakan sendiri ataupun yang mengcopy pekerjaan teman. Dalam mencapai tujuan dari tugas pemrograman dapat dilakukan dengan banyak solusi sehingga setiap mahasiswa memiliki cara pemikiran tersendiri dalam menyelesaikan suatu masalah dan menuangkannya ke dalam bentuk program.

Di Universitas Dian Nuswantoro memiliki Laboratorium Dasar yang menampung mahasiswa untuk melakukan praktikum pemrograman. Laboratorium Dasar memiliki sebuah sistem dimana mampu memanajemen tugas – tugas pemrograman mahasiswa dan memberikan penilaian otomatis terhadap tugas yang di upload ke dalam sistem. Dengan adanya sistem tersebut, tugas seorang dosen menjadi lebih mudah, namun dari sistem tersebut belum mampu mengatasi masalah plagiarisme yang dilakukan oleh mahasiswa berupa source code tugas pemrograman. Plagiarisme mudah dilakukan tetapi sulit untuk di deteksi [7], karena tugas pemrograman mahasiswa di berikan dengan masalah yang sama, dan besar kemungkinan tugas pemrograman mereka akan sama satu sama lain. Umumnya mahasiswa merubah source code tanpa memperhatikan output dari program tersebut. Cara untuk mendeteksi plagiarisme yaitu dengan menemukan bagian dari program yang mungkin mirip dengan program lain yang terdeteksi menyalin di dalam kelompok program mahasiswa. Banyaknya kelompok program mahasiswa akan membuat kesulitan seorang dosen dalam memeriksa source code yang melakukan plagiarisme dengan cara manual.

Berdasarkan latar belakang permasalahan tersebut penulis memilih judul : “ Sistem Otomatisasi Pengelolaan Laboratorium untuk Penilaian Praktikum Pemrograman Dasar dengan Deteksi Plagiarisme ”. Penelitian ini diharapkan mampu membantu untuk menghasilkan penilaian yang lebih sesuai dengan apa yang dikerjakan oleh masing – masing mahasiswa.

II. METODE YANG DI USULKAN

Laboratorium merupakan dasar penting dalam pengajaran dan penelitian di universitas [8]. Sehingga pembangunan dan manajemen

berdampak langsung dalam pengajaran, penelitian dan pelatihan mahasiswa. Kegiatan praktikum pada laboratorium dilakukan untuk melatih mahasiswa dalam penguasaan kompetensi (skills). Tehnik pengembangan pada perangkat keras maupun perangkat lunak membuat hal itu mungkin untuk mengotomatisasi beberapa aspek dari sistem.

Ini merupakan salah satu jenis alternatif dari alat manajemen otomatis agar mengurangi jumlah semua tugas yang harus dikoreksi pada laboratorium. Sistem pengelolaan praktikum merupakan suatu aplikasi yang dirancang untuk membantu melakukan pengelolaan praktikum berupa program, terutama untuk membantu mengelola tugas siswa ke dalam suatu sistem yang berbasis web dan penilaian tugas tersebut.

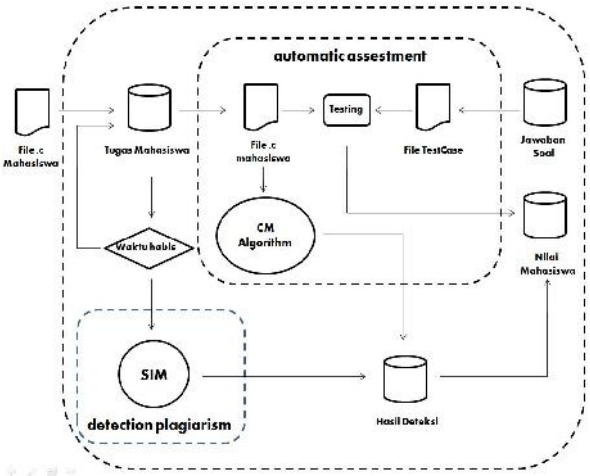
Menurut [9], Sudah ditekankan bahwa proses belajar pemrograman komputer yang dialami harus menyelesaikan berbagai skenario permasalahan. Karena hal itu benar benar penting bagi mahasiswa untuk mengembangkan ketrampilan sendiri dalam pemrograman. Hal ini sangat penting mencegah mahasiswa untuk melakukan plagiarisme terhadap karya orang lain.

SIIM merupakan aplikasi untuk mengukur kesamaan antara 2 atau lebih source code. Aplikasi ini sangatlah berguna untuk mendeteksi plagiarisme source code dalam jumlah banyak, untuk hal ini dapat membantu pengajar atau penilai dalam kelas pemrograman. Setelah di deteksi menggunakan sim jika hasil prosentasi melebihi batas yang ditentukan akan dilakukan deteksi dengan metode perbaikan coding model. Metode ini untuk mengukur gaya coding mahasiswa dengan satu set coding models yang telah digenerate sebelumnya. Metode ini menghasilkan coding model dimana mewakili gaya mengcoding setiap mahasiswa. Kemudian dengan memasukkan source code yang dikumpulkan oleh mahasiswa yang sama kemudian dibandingkan dengan coding models mahasiswa itu sendiri. Jika memang source code tersebut buatan mahasiswa itu sendiri maka menghasilkan ratio yang tinggi. Metode ini terbagi menjadi 3 bagian yaitu preparation, training dan author identification.

III. IMPLEMENTASI

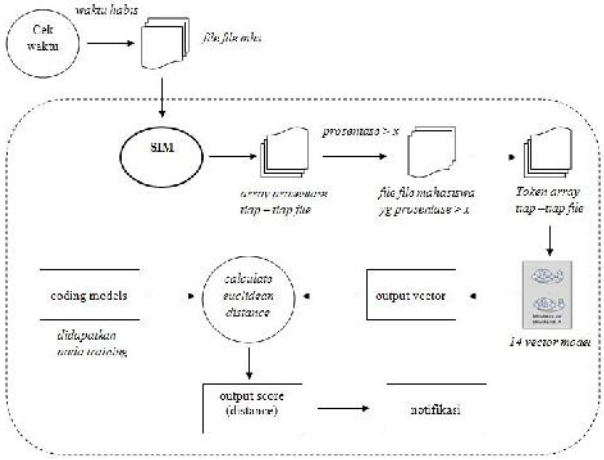
Pada tahap pertama, sudah terdapat suatu problem yang nantinya akan di selesaikan oleh mahasiswa. Semua jawaban dari soal tersebut sudah tersimpan di dalam database jawaban soal yang dibuat oleh dosen. Gambar 1 merupakan arsitektur sistem. Ketika seorang siswa mengerjakan problem yang sudah disediakan dan masuk ke dalam sistem maka alur kerja ketika sebuah source code memasuki kedalam sistem maka akan disimpan kedalam penyimpanan source code. Setelah itu source code akan dibandingkan dengan jawaban soal yang sebelumnya sudah di buat oleh dosen. Sistem akan langsung memproses sehingga akan mengeluarkan hasil berupa nilai dan disimpan kedalam database. Ketika itu juga, source code tersebut juga akan dites dengan menggunakan metode coding models algorithm pada tahap

training untuk mengetahui gaya mengcoding mahasiswa. Seperti kesalahan umum atau kesalahan program, jumlah mengerjakan, jumlah input, gaya mencoding mahasiswa akan disimpan kedalam database, dimana nantinya dapat membantu dosen untuk mengevaluasi kinerja mahasiswa dan keseluruhan kelas.



Gambar 1 Arsitektur Sistem

Hasil training tersebut nantinya disimpan untuk digunakan pada tahap pendeteksian plagiarisme saat ujian. Untuk gambar 2 merupakan skema deteksi plagiarisme. Ketika waktu pengerjaan soal habis, Asumsinya semua source code mahasiswa sudah terkumpul di penyimpanan. Disaat itulah terjadi proses deteksi plagiarisme source code menggunakan metode SIM. Hasil dari metode SIM ini menghasilkan prosentase – prosentase semua mahasiswa dikelas tersebut. Ketika prosentase melebihi batas yang ditentukan, akan dilakukan deteksi dengan coding model algorithm. Deteksi ini menghasilkan vector identifikasi dimana nanti akan dibandingkan dengan vector training masing – masing mahasiswa yang didapat ketika soal mingguan. Semakin kecil jarak yang dihasilkan dari dua vector tersebut artinya semakin mirip dengan model dirinya sendiri dimana mahasiswa tersebut mengerjakan sendiri. Berbanding terbalik jika semakin jauh jarak yang dihasilkan berarti mahasiswa tersebut tidak mengerjakan ujiannya sendiri.



Gambar 2 Arsitektur Deteksi Plagiarisme

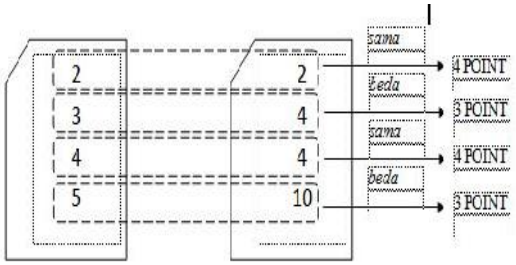
IV. HASIL & PEMBAHASAN

Proses Penilaian dilakukan dengan melakukan compile terhadap soure code mahasiswa dan dilakukan tes dengan testcase input dari dosen.

```
Z:\>Tin.exe < testin.txt > out.txt
```

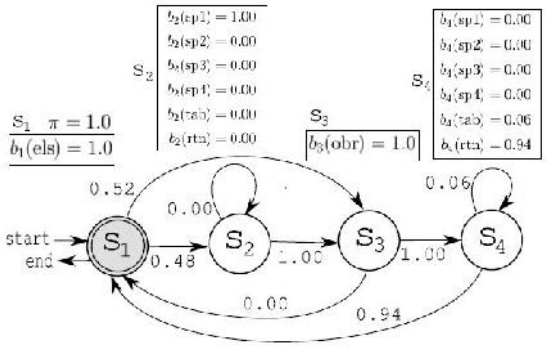
Gambar 3. Proses Percobaan Testcase

Out.txt merupakan output yang dihasilkan program mahasiswa ketika diberi inputan yang berada pada testin.txt. hasil output tersebut dilakukan pembandingan dengan testcase output dari dosen dengan mencocokan perline jawaban mahasiswa.



Gambar 3 Cara Penilaian

Untuk jumlah jawaban benar mendapat point 4 jika jawaban salah mendapat point 3. Jumlah point yang didapat dibagi dengan jumlah benar keseluruhan kemudian dikalikan 100. Untuk soal berupa tidak ujian maka akan dilakukan deteksi dengan menggunakan coding model algorithm dengan tahapan preparation dan training. Preparation dilakukan dengan cara menjadikan setiap karakter source code mahasiswa menjadi token yang sudah didefinisikan sebelumnya diantaranya seperti '{' atau obr, kemudian ':' atau colon. Token yang akan dideteksi kemunculannya hanya 14 token saja. Dimana pada souce code yang berupa token akan dicari masing-masing seperti '{' dilakukan dengan cara maju hingga menemui 'els' atau ke 13 token lainnya. Setelah itu dilakukan mundur hingga menemui 'els' ataupun ke 13 token lainnya selain '{'. Hal itu dapat dilakukan sebaliknya. Setelah masing-masing didapatkan baru dilakukan counting dan pembobotan untuk setiap token dari 14 token yang di identifikasi seperti pada gambar 4.



Gambar 4 Counting Parameter 'obr'

Tahap berikutnya dilakukan penghitungan probabilitas dengan rumus berikut :

$$P(O) = \Sigma P(O \mid M) P(M)$$

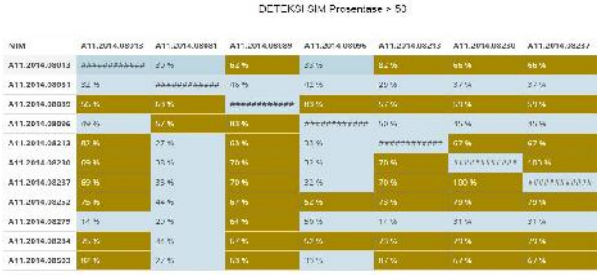
Dimana :

O merupakan sub urutan token pada 'obr'.

M merupakan state yang dikunjungi.

P(M) merupakan perpindahan yang dilalui sub urutan token.

Setiap mahasiswa memiliki 14 vector token dimana akan selalu di update ketika tidak ujian untuk memodelkan gaya mengcodingnya. Ketika ujian dilakukan deteksi dengan menggunakan SIM dimana menghasilkan prosentase seperti gambar 5.



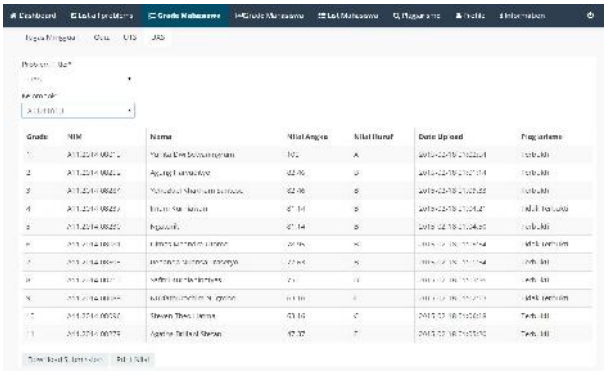
Gambar 5 hasil deteksi sim

Setelah dilakukan deteksi menggunakan sim, prosentase yang melebihi 50% akan dibawa untuk di deteksi dengan coing model agorithm. hal yang sama dilakukan pada tahap ini yaitu preparation dan training tetapi tanpa mengupdate 14 vector

trainingnya. Vector yang didapatkan pada tahap ujian akan dilakukan penghitungan untuk mengetahui jarak sehingga ketika jarak yang ditimbulkan semakin kecil maka mirip dengan modelnya sendiri.

Gambar 6 Hasil deteksi cm algorithm

Hasil deteksi tersebut disertakan pada halaman nilai mahasiswa berupa terbukti melakukan atau tidak terbukti seperti gambar 7.



Gambar 7 Hasil Penilaian

V KESIMPULAN

Dari hasil penelitian, coding model algorithm mampu membaca kebiasaan dalam penulisan coding suatu mahasiswa dari souce code yang dikumpulkan ketika soal bukan ujian dimana akan selalu diperbarui dan dapat memotivasi mahasiswa untuk mengerjakan sendiri soal pemrograman. Hasil penilaian mampu dilakukan secara otomatis dan sesuai dengan hasil dari program yang benar. Dengan adanya deteksi ini mampu mengetahui konsistensi mahasiswa dalam penulisan coding dan pengajar dapat mengetahui dengan mudah apakah suatu mahasiswa mengerjakan sendiri atau melakukan tindakan curang tanpa melakukan penjagaan yang ketat. Hasil deteksi menggunakan SIM dan CM algorithm dapat dijadikan keputusan pendukung dalam memberikan penilaian. Keputusan akhir tetap berada di tangan pengajar.

VI DAFTAR PUSTAKA

[1] Inggriani Liem, *Diktat Kuliah Dasar Pemrograman*. Bandung, Indonesia, 2008.

[2] Leslie Fife, Andrew Thompson Don Colton, "A Web-based Automatic Program Grader," *Information Systems Education Journal*, p. 114, 2006.

[3] V. Almstrum, D. Diaz, M. Guzdial, D. Hagan, Y. B.-D. Kolikant, C.Laxer, L.Thomas,I.Utting, and T. Wilusz M. McCracken, "A Multi-national, multiinstitutional study assestment of programming skills of first-year cs students," *SIGCSE Bull*, vol. 33, no. 4, pp. 125-180, 2001.

[4] Anita Singh Rishabh Kaushal, "Automated Evaluation of Programming Assignments," *ieee*, p. 2012.

[5] B Gaines B Braumoeller, *Actions do speak louderthan words : Deterring plagiarism with the use of plagiarism-detection software.*, 2002, pp. 835-839.

[6] S Hannabus, "Contested texts: issues of plagiarism," pp. Vol 22(6-7), 2001.

[7] N. Wagner. (2001) Plagiarism by student Programmers. [Online]. HYPERLINK "http://www.cs.utsa.edu/~wagner/pubs/plagiarism0.html" http://www.cs.utsa.edu/~wagner/pubs/plagiarism0.html

[8] Bo Li, "Design and implementation of web-based laboratory management system in colleges and universities," *IEEE*, 2009.

[9] Natasja Bautista Rachel Edita Roxas Nathalie Rose Lim, "Automatic generation of plagiarism detection among student programs," *IEEE*, pp. 2-6, 2006.