Nalkki-project – Tool for Plagiarism Detection Using the Web

Petri Sirkkala **Sami Puonti**

Institute of Software Systems
Tampere University of Technology,
P.O.Box 553, FI-33101 Tampere, Finland,
Email: {petri.sirkkala, sami.puonti}@tut.fi

Abstract

Students achieve the best results in learning by writing and doing exercises. This mandates a large number of written exercises. However, limited resources and distribution of assessment work lead to problems when students' answers need to be checked for plagiarism. Plagiarism or copy pasting is difficult to notice in a large volume of documents. The demonstrated project focuses on computer-assisted plagiarism detection in medium to large volumes of text-based submissions. Moreover, the project supports automated search for web sources.

Keywords: Cheating, plagiarism, computer assisted assessment, text documents, web based

1 Introduction

A growing number of students who take programming courses in Tampere University of Technology use the Internet as a source for study material. Also many advanced courses point to the Internet for more up-to-date information on the subject of the course. It is estimated in Jones (2002) that nearly three-quarters of students in college use the Internet more than the library. As a result, the studied materials are easier to copy into exercise submissions. It was indicated by Culwin et al. (2001) that cases of plagiarism in initial programming courses were evident and currently less than 20% of the cohort. However, they added that plagiarism was clearly increasing. Students on a large mass course can easily produce hundreds of submitted text works. To save time, plagiarism detection needs to be automated as far as is reasonable and reliable.

Nalkki is a web application that helps to find plagiarism in students' submissions. Nalkki consists of a core and a web user interface that are written in Python. It can be installed locally in the university or department that wishes to use it. As the deployment is local it is easier to maintain security and to limit access to students' answers.

Automatic comparison is only the first step in determining if the submission really is plagiarised, and manual work is always required to make a decision on the findings (Surakka et al. 2006). For making this decision Nalkki offers two views. First it shows a listing of all documents ordered by relative similarity. After finding an interesting case in the similarity listing, the teacher opens a view to see the detailed comparison findings.

The demonstrated project is based on a Virtual university funded project, which was further enhanced with web aware features.

2 Problems in finding plagiarised text

Finding plagiarised parts of a text document is very slow labour for teachers. Even with a limited number of texts it relies on the teacher's ability to read and remember every submission. This is slow and ineffective. The Internet makes the problem even worse, since the teacher would be required to read and remember related material on the Internet to find the possible sources of copy and paste. Search engines can certainly help to find sources that have been copied, but picking suitable search terms and manually searching is tedious and repetitive.

As the process of finding plagiarised parts is based on the teacher's ability to remember all that he or she has read, the results may be incomplete. Some clear cases of copy and paste may easily be overlooked. And since the workload cannot be shared between multiple assistants, the monolithic toil is easily impossible for one teacher.

Plagiarism eats resources that would be better used in real work. It is wasting everyone's time without any gain on the course material. On mass courses, some students always play foul game. Without proper tools to handle plagiarism, the chances that those students will be penalised are close to nothing. We present Nalkki, a tool to help detect plagiarism.

3 Automated plagiarism search process

The process of finding plagiarism with Nalkki is generally done in four wizard-style steps. Adding submissions to Nalkki is the first step. Submissions are uploaded to Nalkki as individual files or as a zip file containing the source documents. The second step is running the comparison process. The comparison includes converting documents into a suitable text format, scanning for given references and making search queries to find sources that are not directly referenced. The third step is viewing the results in a listing. The last step is checking the findings in a detailed comparison view of each potential document.

3.1 Presenting the similarity listing

Nalkki presents two views of the results. The first view is a similarity listing of all source documents, sorted in relative similarity order. By relative similarity we mean that the number of words that are found to be copied from another source is divided by the length of the document in words. This listing allows the comparison view to be opened.

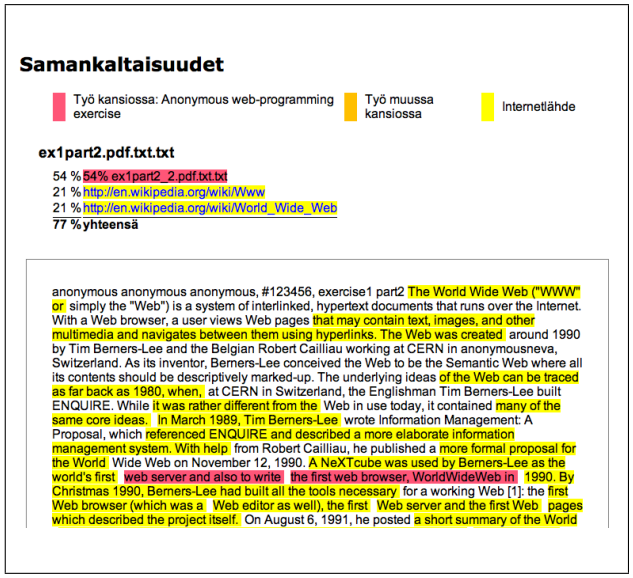

Figure 1: Comparison view reveals details about copied content

In the similarity listing the first column tells how much of the whole submission is found to have matches in other documents. The second column shows what was the biggest single match between this submission and another document. The third column leads to the comparison view that is explained in the next section. In the fourth column is the original name of the submission.

3.2 Viewing detailed comparison of a document

Any document that is listed in the similarity view has a link to open the document for closer inspection. This link leads to the comparison view, where the document text is presented. Any copied parts are identified by colours. Figure 1 shows an example of the comparison view.

The comparison view uses three colours to highlight the matches Nalkki has found. Red highlight is used to denote a match that was found only in another document in this submission set. Yellow means that the match was found on the Internet. If a match leads both to another document in this submission set and to the Internet, the Internet colour dominates as it is assumed that both submissions have copied the same Internet source. An orange highlight is used if the match is made with a submission in a previous submission set, for example last year's submission of the same subject.

4 Discussion

Automated plagiarism detection works well when enough comparison material is available. When the exercise is short or there are few participants the results are not so good. Fortunately, these cases are better handled by manual inspection. As noted by Ahoniemi & Reinikainen (2006), some assignments tend to produce similar results. For example, a teacher might give students a base document to work on. The text in the base document would therefore be present in most of the submissions. The unwanted comparison matches could be compensated by letting the teacher upload a negative match document that could be used to eliminate those matches.

The conversion to text format is limited to certain document types. This limits Nalkki in two ways. First, the submissions are required to use formats that are suitable for text conversion. Second, external web documents that Nalkki cannot convert to text are not used in comparison. It might be possible to support more formats for web documents by allowing partial failure in conversion. Currently Nalkki supports only essay-style answers, and is therefore not directly usable for finding plagiarism in program code.

Nalkki only provides clues to potential cases of plagiarism. It is necessary to refer to the original document to prove the case. As stated by Surakka et al. (2006), finding potential plagiarism cases takes only 10-40% of the time, while an estimated 60-90% of the time is spent on manual work after detection. Nalkki has proven to be effective in the courses it has been piloted on. Some cases of plagiarism have already been detected with Nalkki. The feedback from teachers has been positive. Moreover we have been asked to present the system at faculties' teacher meetings. Just because Nalkki exists the teachers have already warned students not to try plagiarism, as the chances of getting caught are high.

Since plagiarism is a delicate subject and there are legal limitations to sharing information, the reports are not generally available. Plagiarism is a very serious accusation and therefore each case must always be double-checked by hand.

5 Conclusions

Plagiarism is a growing issue. To manage the issue, automated detection tools help to find cases of copy and paste more efficiently in medium to large volumes of submissions. Teachers have found Nalkki to be a useful tool in detecting plagiarism and used it as a threat to prevent plagiarism. Since plagiarism is such a serious accusation, the final decision must always be made manually.

References

Ahoniemi, T. & Reinikainen, T. (2006), Aloha - a grading tool for semi-automatic assessment of mass programming courses, *in* 'A. Berglund & M. Wiggberg (Eds.) Proceedings of the 6th Baltic Sea Conference on Computing Education Research', Uppsala, Sweden, pp. 139–140.
URL: *http://cs.joensuu.fi/kolistelut/*

Culwin, F., MacLeod, A. & Lancaster, T. (2001), 'Source Code Plagiarism in UK HE Computing Schools, Issues, Attitudes and Tools', *South Bank University Technical Report SBUCISM-01-02* .

Jones, S. (2002), 'The Internet goes to college', *Pew Internet & American Life* **15**.

Surakka, S., Ahtiainen, A. & Rahikainen, M. (2006), Plaggie: Gnu-licensed source code plagiarism detection engine for java exercises, *in* 'A. Berglund & M. Wiggberg (Eds.) Proceedings of the 6th Baltic Sea Conference on Computing Education Research', Uppsala University, Uppsala, Sweden, pp. 141–143.
URL: *http://cs.joensuu.fi/kolistelut/*