Detection and Incidence of Plagiarism in a Solid Modeling Course

Dr. Steven Joseph Kirstukas, Central Connecticut State University

Steve Kirstukas is an Associate Professor at CCSU, where he teaches courses in solid modeling, MATLAB programming, and engineering mechanics. He is exploring the use of computer-aided assessment of CAD files to give consistent, accurate, and quick feedback to students. He has degrees in civil and mechanical engineering, with a Ph.D. from the University of Minnesota. Steve has worked in industry as a civil engineer, software developer, biomechanics researcher, and mechanical design engineer.

Detection and Incidence of Plagiarism in a Solid Modeling Course

Abstract

This research paper presents the method used and results of a study in plagiarism detection of solid models. To aid in detecting plagiarism, a computer program was written that runs within a specific CAD package on the instructor's computer. When a solid model is opened, this program reads the complete history that is stored within the CAD file. This history contains all save dates and times, a hardware identifier corresponding to each save time, and previous file names. For files that were created at the same times on the same machines, are older than expected, or have unexpected originating names, these files can be placed into one of twelve different categories that that predict whether a particular file is probable plagiarism, near-certain plagiarism, near-certain self-plagiarism, or a false positive. In this paper, the program was used to examine the incidence of these types of plagiarism by retrospectively examining more than 3000 CAD files over fourteen semesters during a seven-year period. We have found plagiarism in this CAD course to be a relatively uncommon occurrence, but nevertheless present in 1-2% of all submitted files, and involving as many as 16% of all students in a particular section. Because we can now detect plagiarism, we hope that this will discourage future plagiarism and instead help students learn to be effective and efficient part modelers.

Introduction

The functional definitions of plagiarism can depend on the field in which they are used. The ASEE Policy on Plagiarism and Duplicate Publication [1] is written for authors of scholarly publications and says that, "plagiarism occurs when an author copies the words, illustrations, and ideas of others without identifying the sources." Another type of plagiarism is sometimes described as "redundant publication" or "self-plagiarism" and involves the reuse of significant similar portions of one's own work without acknowledging that one is doing so. While self-plagiarism does not involve the theft of the ideas of others, it is considered by many to be a form of academic dishonesty. A similar definition can be applied to student work: for assignments, students should be submitting their own original work, not the work of others or even their own old work.

Plagiarism detection software has been around for some time. Unlike plagiarism detection software for text documents that relies on matching of precise or very similar text phrases, plagiarism detection of engineering and technology files must operate differently. One such tool is Moss (Measure Of Software Similarity) which was developed in 1994 and is still available for use [2]. Moss can compare code written in more than 20 different computer languages and is often used for detecting plagiarism in programming classes. For Moss, plagiarism is defined as using copied code structure without attribution. Another plagiarism tool in the engineering realm is for the graphic comparison of integrated circuit layouts [3]. To date, it appears that no plagiarism-detection tools are available for solid models.

Background

All students at Central Connecticut State University in the programs of mechanical engineering, mechanical engineering technology, and manufacturing engineering technology take a course in 3D CAD where they create parts, assemblies of parts, and drawings of parts and assemblies. We use Siemens NX as the CAD package because some of Connecticut's biggest employers such as Pratt & Whitney and General Dynamics Electric Boat use NX, as do the large number of companies that work with them. While other CAD packages have an easier learning curve, we have observed that students who are fluent in NX can quickly make the transition to other CAD packages.

In our 3D modeling class, collaboration during in-class work and out-of-class homework is encouraged. There is only one instructor and many students, so having students assist one another is efficient and helps them learn by explaining, demonstrating, and discussing. However, the risk in allowing collaboration is that students may attempt shortcuts by submitting a file that is totally or partially based on another's work. This can result in the student who is working from another's work often having a significantly higher homework score relative to their score for in-class exams, where collaboration is not permitted. This kind of cheating does no one any good, so plagiarism is actively discouraged.

As part of the homework for the course, students create CAD models in response to tutorials and exercises. In tutorials, students follow instructions available on videos created with Camtasia [4], where the instructor creates the model in NX. The student sees what the instructor is doing, and more importantly, can hear why the instructor chooses to do things in some ordered sequence. In the exercises, the student has to come up with their own plan and create a CAD model.

Students upload all tutorial and exercise files to the instructor through the moodle learning management system, and all filenames are expected to follow a standardized naming format. In the syllabus, the file naming convention is defined:

> All files that you create for this class must be saved with your last name as the prefix of the file name. Sample filenames:

Hunter_C2_T2.prt	(means Hunter created a model for Chapter 2, Tutorial 2)
Hunter_C3_E1.prt	(means Hunter created a model for Chapter 3, Exercise 1)

Use of standardized filename protocol is beneficial to both student and instructor. Over the semester, students will create many parts; the part filenames can be displayed in Windows File Explorer sorted by name, and the name identifies exactly what it is. When the instructor downloads a group of exercise files, they are all copied into the same folder. Without standardized filenames beginning with student last names, the instructor has a difficult time associating a file with a file owner.

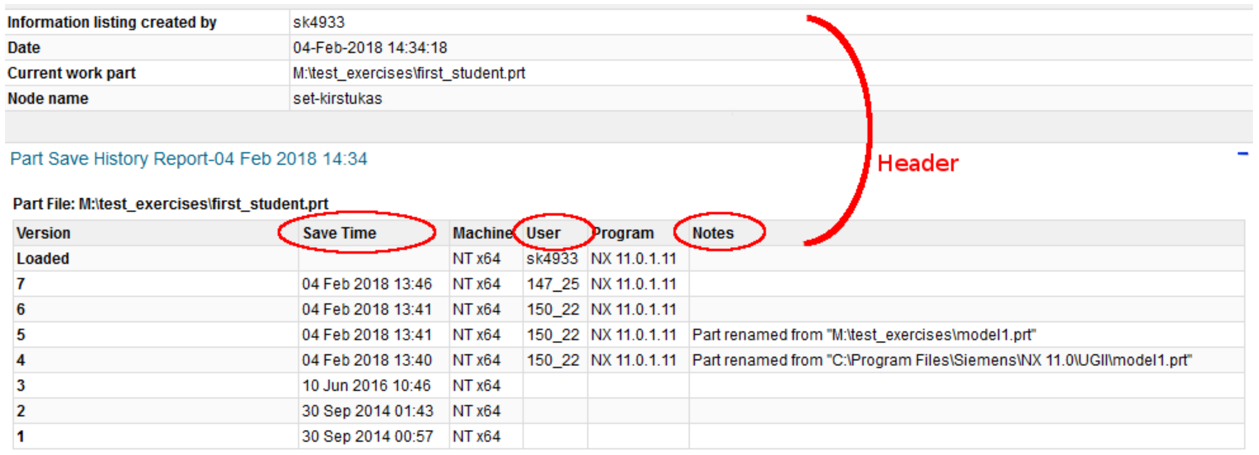
Even within our own engineering department, plagiarism is defined differently in different courses and across instructors. In the course syllabus for the CAD course, students are told that collaboration among fellow students is expected and encouraged during the tutorials and exercises, but the work they submit must be their own. This means that they are permitted to ask for help from anyone, but cannot submit work that is identical to, or even based on, the work of others. I would consider either case to be plagiarized work, as it uses a substantial portion of another's work without attribution.

In any given course section, up to 10% of the students may be repeating the course for various reasons. These students likely will still have access to their old exercise files. As the assigned exercises tend to be the same every semester, there is the temptation to submit old work, or modified versions of the student's own old work (self-plagiarism). While not at the same level of academic dishonesty, self-plagiarism harms the student because they don't go through the process of recreating the CAD model from scratch and relearn the sequence of steps. Also, the feedback they receive will not have much value as it pertains to work they did many months ago.

Extraction of File Properties to Detect Plagiarism

Superficial file properties obtained by right-clicking on a file — such as file size, or file creation, modification, or access dates — are typically of no use in detecting plagiarism. Examination of the CAD files to compare names of driving dimensions and the general structure of the CAD model is possible but is rather involved and is not being done at the moment. Instead, possibly plagiarized files are located using the file history that is part of every NX CAD file.

From the default software configuration, checking the history of any NX CAD file is a cumbersome manual process requiring five mouse clicks as one navigates through the menu system. But a script can be recorded of those actions and a button can be created to run the script. When the button is placed on the home tab, the file history can be accessed with a single click. A typical file history is shown in Figure 1.



Information listing created by	sk4933
Date	04-Feb-2018 14:34:18
Current work part	M:\test_exercises\first_student.prt
Node name	set-kirstukas

Part Save History Report-04 Feb 2018 14:34

Part File: M:\test_exercises\first_student.prt

Version	Save Time	Machine	User	Program	Notes
Loaded		NT x64	sk4933	NX 11.0.1.11	
7	04 Feb 2018 13:46	NT x64	147_25	NX 11.0.1.11	
6	04 Feb 2018 13:41	NT x64	150_22	NX 11.0.1.11	
5	04 Feb 2018 13:41	NT x64	150_22	NX 11.0.1.11	Part renamed from "M:\test_exercises\model1.prt"
4	04 Feb 2018 13:40	NT x64	150_22	NX 11.0.1.11	Part renamed from "C:\Program Files\Siemens\NX 11.0\UGII\model1.prt"
3	10 Jun 2016 10:46	NT x64			
2	30 Sep 2014 01:43	NT x64			
1	30 Sep 2014 00:57	NT x64			

Figure 1: Typical file history

The header has no useful information regarding plagiarism detection and consists of all lines of the file above *Loaded*. The key fields below the header are *Save Time* (which includes a date and time), *User* (which for us tells us which machine altered the file), and an optional *Note* that may indicate a filename change. The save times are listed from most recent to oldest. The last save time that is paired with a user value is the creation date and time. All of the save times above the creation date and time are modification dates and times. We can see that the file corresponding to Figure 1 was created at 1:40 pm on computer 22 in Room 150. The file was then renamed and further modified, and then was last modified at 1:46 pm on computer 25 in Room 147. The history line that begins with *Loaded* indicates that the file is currently open on the instructor's computer.

Manual examination of the file history will ascertain whether the file creation date and subsequent modification dates were during the current semester. It also identifies previous filenames that were used with this file; this could indicate whether the file originated with another student. However, manually accessing the file history will not disclose if the save times and user identically match that of other files in the same course section, or of files that were previously submitted to that instructor.

Automated Inspection of CAD File Histories

Using an enhanced version of a previously described NX Open based program to compare CAD files [5], each CAD file is individually opened by the program, and its part history is saved as a text file in the local Documents directory. The text file is then examined and each row of the file below the header is processed one row at a time. The save times and user information (which generally identifies the computer workstation) are stored in a FileHistory object. For any file history, there will be the same number of save times and users, and there will be at least one save time and user.

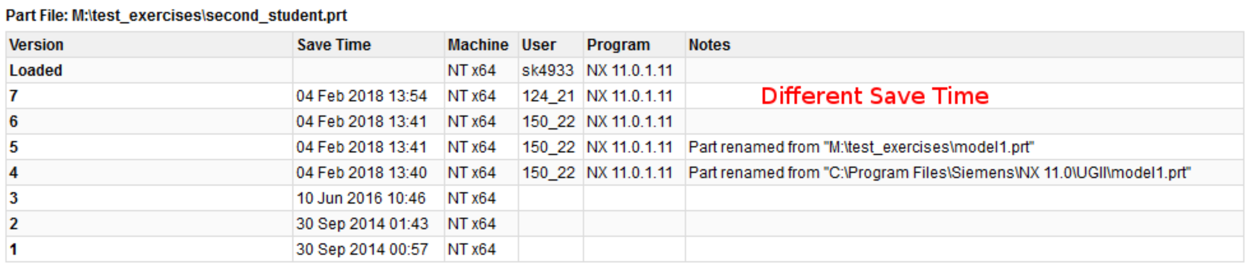
The current FileHistory object is compared to all previously processed FileHistory objects to see if there is similarity, meaning one or more identical save time and user pairs. Similarity is needed for a file to be a confirmed case of plagiarism. When we get a save time and user match, a message is displayed to the output window that indicates that file similarity was observed and also displays the number of matching modification times. Additionally, if the time gap between file creation date and the end of the relevant semester is longer than 120 days, then a message is written to the output window indicating that the file appears unexpectedly old.

The program also examines the file note of the file history and looks for the word *renamed*. If it finds the word *renamed*, it then reads further into the note and extracts the name of the originating file. Many files are renamed at some point, so being renamed by itself does not indicate anything conclusive; however, sometimes the originating filename can identify another student via the filename protocol. To decrease the number of false positives due to renaming, an originating filename undergoes several checks. First the full path of the originating file is checked to determine if it was saved to the default file save location which begins with *C:\Program Files\Siemens\NX*. If not, the filename is examined to see if it's similar to the default file name, which is of the form *model#.prt*, where # indicates an integer between 1 and 9, such as *model1.prt* or *model2.prt*. A final check examines if the file was renamed from a file that begins with the same student last name associated with the current file. For instance, when dealing with a file called smith_C6_E1.prt, the program extracts the last name *smith* and if the part was renamed from smith_E1_C6.prt (an inversion of the requested file name convention) the program would ignore that name change. Because the current filename and the original filename reference the same student last name, it is highly unlikely that this type of renaming indicates plagiarism. Alternatively, if one of the previous filenames has neither the default path location or a filename of expected structure, the program output window will display a message indicating that the originating filename requires further manual review to determine if it possibly indicates plagiarism.

The program was developed for use primarily on exercise files. With tutorials, the student simply needs to follow provided video directions, so tutorial files are expected to be good and no further review is done. In this paper, only single parts submitted as exercises were examined — typically 6 to 8 exercise files per student per semester. Assemblies, drawings, and individual projects were excluded from this study. However, the program has been tested against assembly and drawing files and is fully capable of extracting history data from these files.

Demonstration of the Program

A second CAD file was created by copying the file *first_student.prt* after it was saved in Room 150. The file was then renamed in Windows File Explorer to *second_student.prt* and further modified in Room 124 (Figure 2). Somewhat interestingly, the file history does not mention that both files were once named *first_student.prt*, as the rename from *first_student.prt* occurred outside of the program.



Part File: M:\test_exercises\second_student.prt

Version	Save Time	Machine	User	Program	Notes
Loaded		NT x64	sk4933	NX 11.0.1.11	
7	04 Feb 2018 13:54	NT x64	124_21	NX 11.0.1.11	Different Save Time
6	04 Feb 2018 13:41	NT x64	150_22	NX 11.0.1.11	
5	04 Feb 2018 13:41	NT x64	150_22	NX 11.0.1.11	Part renamed from "M:\test_exercises\model1.prt"
4	04 Feb 2018 13:40	NT x64	150_22	NX 11.0.1.11	Part renamed from "C:\Program Files\Siemens\NX 11.0\UGII\model1.prt"
3	10 Jun 2016 10:46	NT x64			
2	30 Sep 2014 01:43	NT x64			
1	30 Sep 2014 00:57	NT x64			

Figure 2: A file history of a second file that borrows heavily from the previous file

These files were placed in the same directory and NX was opened, and the executable file was loaded. The directory was selected, and the program finished 12 seconds later. Program output is displayed in Figure 3.

```
Processing all parts in folder M:\test_exercises
   Note: second_student.prt has same creation date and creator as first_student.prt and 2
of 3 same modification dates

Total number of files processed: 2
Total Time Elapsed: 0 minutes and 12 seconds.
```

Figure 3: Program output after comparing the two files

Note that the program ignored the file rename notes because the full file path and/or filenames are default values and don't supply any useful information. However, it noted that the two files were created on the same machine at the same time and then modified on that machine two more times. This would be a case of near-certain plagiarism.

Detection of Plagiarism

The program identifies three criteria, each of which will be represented in this paper by a single word: similarity, rename, and old. *Similarity* means that at least one file save time and user match identically. *Rename* means the file was renamed from another filename which may be suspect. *Old* identifies a file that was created more than 120 days before the end of a particular semester. These three criteria can be grouped together in various different combinations. After examining the output of thousands of student files, it was noted that some combinations suggest plagiarism only if certain conditions are met. We will examine each of twelve categories that have been useful in classifying some files and discuss when a file's category indicate probable plagiarism.

1. *Similar* or (*Similar* and *Rename*)

1a. Two files were created by the same student and have the same save time and user. This can happen when a student submits multiple versions of the same exercise file, for instance, an initial version and a final version. Only the final version should be uploaded, but early users of the course software sometimes submit more than one exercise file. This type of similarity would be categorized as a plagiarism false positive.

1b. A first file was created by a student who then went on to drastically revise it and save it as a second file. For example, the student opened an exercise 1 file and then deleted all model features such as sketches and extrudes and then recreated an exercise 2 model. These files would have one or more identical save time and user pairs. This type of model creation is discouraged but is not plagiarism.

1c. Otherwise, the originating file was created by another student. Since the file was not identified as old, both files came from the same semester. The first file read would not be flagged, but the program would note the similarity when processing the second file and would link the two files where plagiarism was noted. Currently, the program makes no attempt to identify the originator or receiver of the file. Both files would be categorized as being involved in near-certain plagiarism.

2. (*Similar* and *Old*) or (*Similar* and *Old* and *Rename*)

2a. The student is resubmitting their own work from a previous semester that was previously seen by the program. This would be categorized as self-plagiarism.

2b. Otherwise, the student is submitting someone else's work from a previous semester that was previously seen by the program. This would be categorized as near-certain plagiarism.

3. Old

3a. The student is resubmitting work from a previous semester that has not yet been seen by the program. This would be categorized as probable plagiarism (however, it cannot be determined with certainty if it is plagiarism of others or self-plagiarism).

4. *Old* and *Rename*

4a. The file history points to another student's file from a previous semester that was never seen by the program. This would be categorized as near-certain plagiarism.

4b. The original file save time is from a previous semester but the originating filename does not identify a student by name. This would be categorized as probable plagiarism.

4c. The originating filename points to an old instructor-provided file. The student went on to drastically revise it by removing features and building it up as a new file. This type of model creation is discouraged but is not plagiarism.

4d. The student submitted a file built upon an old template (that could involve things like setting datum planes). While using old templates would be discouraged, provided that the originating filename does not point to likely plagiarism, this would be categorized as a plagiarism false positive.

5. Rename

5a. The originating filename was changed in an attempt to conform to the requested filename protocol. This would be categorized as a plagiarism false positive.

5b. The filename may be referencing another student in the same semester. But the program has not yet processed the other student's file to note the save time and user similarity. This would be categorized as probable plagiarism that would be further confirmed after analysis of the other student's file.

Results

The program was assigned to examine the master folder where the instructor keeps all subdirectories corresponding to seven years of coursework. Only subdirectories that contained the word *exercise* were included. Additionally, if the file path contained something that would indicate assembly or drawing files (such as *C9, C10*, or *C13*), then that subdirectory and all subdirectories below that would be excluded. The program creates a sorted list of all the files (path and filename) to be examined. Beginning with the first file from year 2011, each *.prt file was opened by NX and its file history read and processed. If the program detected anything requiring manual review, a message was sent to the output window. A total of 3055 student files were processed during a program run lasting a couple of hours (approximately three seconds per file). Of these, 119 files were flagged for manual evaluation of plagiarism. Program output was pasted into an Excel spreadsheet, and each of the 119 files was categorized from 1a thru 5b. These data were collected into false positive, probable plagiarism, near-certain plagiarism, and near-certain self-plagiarism groups. Also, the data was summed over all 14 semesters (Table 1).

Semester	number of students	number of files	number of files per category												number of files FALSE POS	number of files PROBABLE PLAGIARISM	number of files NEAR-CERTAIN PLAGIARISM	number of files NEAR-CERTAIN SELF-PLAGIARISM	number of students doing unpermitted activities
			1a	1b	1c	2a	2b	3a	4a	4b	4c	4d	5a	5b					
2011 Spring	22	201	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
2011 Fall	59	422	5	0	0	0	0	0	0	0	0	0	5	0	10	0	0	0	0
2012 Spring	40	329	2	0	2	2	0	0	0	0	0	0	9	0	11	0	2	2	3
2012 Fall	18	121	0	0	0	0	0	0	0	0	0	0	3	0	3	0	0	0	0
2013 Spring	17	126	0	1	0	0	0	0	0	0	0	0	8	0	9	0	0	0	0
2013 Fall	18	151	0	0	0	0	0	4	0	0	0	0	1	0	1	4	0	0	1
2014 Spring	19	156	1	0	0	0	0	0	0	0	0	0	1	0	2	0	0	0	0
2014 Fall	22	192	0	1	0	0	0	0	0	0	0	0	1	0	2	0	0	0	0
2015 Spring	43	309	0	1	0	0	0	0	0	0	0	0	10	0	11	0	0	0	0
2015 Fall	40	263	0	0	0	0	0	0	0	0	0	3	6	0	9	0	0	0	0
2016 Spring	39	223	2	0	4	2	0	6	3	0	0	0	0	0	2	6	7	2	5
2016 Fall	24	150	2	0	0	0	1	3	0	0	0	0	7	0	9	3	1	0	3
2017 Spring	25	192	1	0	6	0	1	3	2	1	0	0	1	2	2	6	9	0	4
2017 Fall	26	220	0	0	0	0	0	0	0	1	2	0	2	0	4	1	0	0	1
Totals	412	3055	14	3	12	4	2	16	5	2	2	3	54	2	76	20	19	4	17

Table 1: Summary of fourteen semesters of data

Of the 119 files flagged for manual evaluation, 76 of these (64%) were quickly identified as false positives, as it was clear that these files had been renamed by the original author to correct spelling errors or to conform to the filename protocol.

At least a few files were identified in every one of the 12 defined categories. A total of 43 files were identified as probable or near-certain plagiarism, including self-plagiarism, and each of these groups comprised less than 1% of the total file population. For 8 of my 14 semesters, I saw no evidence of plagiarism of any kind. In the other 6 semesters, I saw between 4% and 16% of students involved in some sort of plagiarism.

Discussion and Conclusion

For files that exhibit save time and user similarity to others, are older than expected, or have unexpected originating names, these files can be placed into one of twelve different categories that predict whether a particular file is probable plagiarism, near-certain plagiarism, near-certain self-plagiarism, or a false positive. After a run of the program, a manual review of the output is needed to place the files in the appropriate category. Overall, 96% of files had no indications of suspicious behavior, and the majority of those that required manual review were found to be false positives.

Going into this study, it was unknown whether some previously observed incidences of plagiarism were the full extent of the issue or merely the tip of the iceberg. Outside of a high of 16% students that were involved at one time or another in plagiarism in one course section, it was a pleasant surprise to see so few cases of plagiarism. Only 1.4% of all files indicated plagiarism. Many of the noted cases of plagiarism were already known, although a few new cases were identified.

Although three of the last four semesters had relatively higher incidences of plagiarism, the numbers are too small to make any definitive statements about the incidence of plagiarism changing over time.

Because we can now detect plagiarism as part of an automated file grading process, both the giver and receiver of suspect digital files can be quickly assembled for a discussion. The use of this program in the teaching environment will encourage students to learn early to avoid plagiarism, teaching them instead to be effective and efficient part modelers in their work.

Future Work

An improvement in the next phase of code revisions will increase the program's usefulness by writing the full master list of save times and users to a local file. A master list saved to a local file would make it much faster to compare the current semester's files to past files. When new files are processed, their information will be appended to the master file.

As this was a retrospective study concerning the development and verification of a plagiarism detection tool, it is unknown what the effect of the program will have on upcoming class sections. Areas for further study include ascertaining whether plagiarism is reduced when students are aware of the program's existence, and determining the appropriate measures to be taken when plagiarism is detected.

References

[1] ASEE Policy on Plagiarism and Duplicate Publication, https://www.asee.org/public/conferences/106/papers/plagarism_policy accessed April 29, 2018.

[2] Aiken, Alex, "Moss: A System for Detecting Software Similarity," https://theory.stanford.edu/~aiken/moss/ accessed February 2, 2018.

[3] Kasprowicz, Dominik, Wada, Hilekaan, "Methods for automated detection of plagiarism in integrated-circuit layouts", Microelectronics Journal, September 2014, Vol.45(9), pp.1212-1219.

[4] Amaya-Bower, L., Kirstukas, S. (2016) "Effect of Video Guided Tutorials in a Standard Curriculum and in a Flipped Classroom for a 3D-CAD Course," Proceedings of the 2016 ASEE Annual Conference & Exposition, New Orleans, LA, June 26-29, 2016, 10.18260/p.27295, 11 pages.

[5] Kirstukas, S. (2016) "Development and Evaluation of a Computer Program to Assess Student CAD Models," Proceedings of the 2016 ASEE Annual Conference & Exposition, New Orleans, LA, June 26-29, 2016, 10.18260/p.26781, 13 pages.