

Received June 24, 2021, accepted August 16, 2021, date of publication August 20, 2021, date of current version September 3, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3106509

# Cronus: An Automated Feedback Tool for Concept Maps

MASRIK A. DAHIR<sup>1</sup>, SYED ALI QASIM<sup>1</sup>, AND IRFAN AHMED<sup>1</sup>, (Senior Member, IEEE)

Department of Computer Science, Virginia Commonwealth University, Richmond, VA 23284, USA

Corresponding author: Masrik A. Dahir (dahirma@vcu.edu)

**ABSTRACT** A Concept map is a pedagogical tool to help students understand the concepts and identify their misconceptions. Grading a concept map is a time-consuming manual task causing a severe bottleneck to use concept maps in a large class effectively. This paper presents Cronus that provides useful feedback on a student concept map similar to manual assessment by comparing it with an instructor concept map. The feedback includes identifying misconceptions, finding concepts, links, and branches that are (partially) matched or missed from a student concept map, generating summary statistics based on the feedback, and suggesting a grade of the map using predefined criteria (by the instructor) on the summary stats. Cronus is evaluated on a dataset of 74 student concept maps collected as homework assignments in an undergraduate (senior-level) course on introductory computer security. The evaluation results show that Cronus can provide accurate feedback on student concept maps compared to the manual evaluation of the maps and automatically suggest their correct grades.

**INDEX TERMS** Concept map, student misconception, automatic grading, cybersecurity education.

## I. INTRODUCTION

Concept maps are a pedagogical tool for visually organizing and representing knowledge. Figure 1 shows a simple example of a concept map on the data acquisition of digital forensic evidence, along with the key terms in Table 1. A concept map includes concepts represented as text boxes and relationships between pairs of concepts indicated by a connecting link (arrow) and a proposition, i.e., a word or a phrase describing the link. The most abstract concepts are placed at the top of the diagram, while progressively more specific ones are placed underneath them, creating a tree-like hierarchical structure. This simple design allows seamless and effective linking and exploration of concepts at different levels of detail.

Concept mapping is a cognitively intensive task that examines the level of a student’s understanding of concepts. It is beneficial for in-class activities and homework assignments and offers opportunities to improve instructional effectiveness. A poorly constructed map by a student has missing links and gaps in logic or incorrect information that can allow the instructor to correct misconceptions developed by a student. Conversely, instructors can use a correct map in class as the basis for in-class discussion. The map helps students actively

The associate editor coordinating the review of this manuscript and approving it for publication was James Harland.

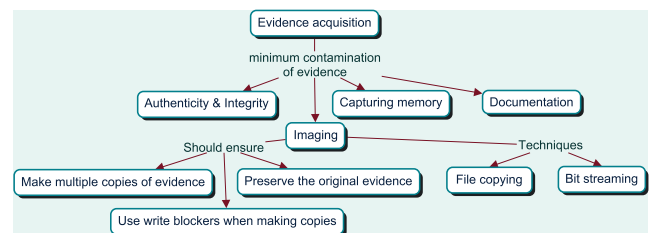


FIGURE 1. Concept map example of forensic data acquisition.

TABLE 1. Terminologies in concept map.

Term	Description
Concept	Node representing an idea
Link	Arrow connecting two concepts
Root concept	Concept with no parent
Leaf node	Concept with no children
Orphan	Concept not connected to other concepts
Branch	Complete connection from root to leaf node
Proposition	Concept-linking phrase-concept connection

build their understanding of foundational concepts and reason about the bigger picture and the connections among concepts.

Research has shown that concept mapping is beneficial for student learning if it is used as an integral, on-going feature of the learning process and not as an isolated activity at the

beginning or end of a semester [1]. Concept maps are useful for students to clarify their knowledge structures [2]. The students who learn through concept maps have better learning outcomes over traditional approaches [3].

Currently, grading and assessing student concept maps is a manual, tedious, and time-consuming task for an instructor, thereby posing a serious challenge to utilize concept maps for a large class effectively throughout a semester. Furthermore, the existing automated grading methods (e.g., [1], and [4]) are based on topological scoring and utilize only structural features of a concept map such as average words per concept, concept count, and linking-phrase count. Recently, Deshpande *et al.* [5] show that the accuracy of a topological scoring is not comparable to a manual rubric assessment.

In this paper, we propose *Cronus* that compares a student concept-map with an instructor (master) concept-map on a topic automatically and provides feedback similar to a manual assessment of a concept map. Specifically, *Cronus* identifies misconceptions in the student concept-map and finds the nodes, linking phrases, and branches matched or partially matched in the instructor's and student's concept map. It employs natural language processing to handle synonyms and different linguistic patterns.

In the end, *Cronus* generates an equivalent of the master concept-map that visually highlights the findings with different color schemes and line styles. An instructor can use the map to understand the quality of the student's concept map to grade it quickly and provide feedback to the student on misconceptions effectively. Furthermore, *Cronus* includes summary statistics of the findings on matched branches, partially matched-branches, extra-branches, concepts-matched, links-matched, etc. It utilizes predefined criteria (configurable by an instructor) using the summary stats to suggest a final grade of a student concept map automatically.

We evaluated *Cronus* on 74 concept maps developed by students of a computer security class. Our evaluation shows that the grading and feedback done by *Cronus* are significantly closer to the manual grading with the maximum average difference of 5.30%.

*Cronus* is written in Python and is released on GitHub at <https://github.com/Masrik-Dahir/Cronus>.

## A. CONTRIBUTIONS

- We present *Cronus*, a concept-map feedback tool to assist instructors in scoring based on statistical and analytical comparison with any grading templates.
- *Cronus* generates two PDF to display contextual and topological analytical data and graph.
- We evaluated *Cronus* with 78 student concept maps against 3 instructor concept maps and found an accuracy rate of 90% ( $R^2 = 0.91, 0.91, \text{ and } 0.88$ )
- We released our data set of over 1000 concept map comparison (both contextual analysis graph and topological analysis graph) and code base at GitHub

**Roadmap.** The rest of the paper is organized as follows: Section II provides the related work. Section III outlines

the problem statement and the proposed approach, *Cronus* followed by its implementation and evaluation in Sections III and IV. Section V concludes the paper.

## II. RELATED WORK

Concept map has proven to be an outstanding tool for education (i.e. self-evaluation, measuring the level of students' understanding). It has proven to be not only a good representation tool but also helps graduate students to become good learners [6]. Another study shows that there is a positive correlation between student understanding and concept mapping [7]. Tanner and Dampier [8] demonstrated the application of concept maps in a digital forensic investigation. Gwo-Jen [9] showed that collaborative u-learning activities via concept maps are effective in improving students' learning performances. Another experiment [10] shows that a concept-map-oriented mobile learning system with an instant feedback system can significantly improve student's learning mechanisms. The finding establishes that an automated concept map evaluation system is a catalyst for higher-order thinking and understanding the hierarchical composition of a topic. However, comparing two concept maps and a quick grading method is necessary for the instructor to grade them in a reasonable fashion. So far several studies have been completed, but Novak and Gowin [11] proposed the first evaluation rubric in the book called *Learning How to Learn*.

Novak [11] first proposed structural scoring; however other authors [7], [12]–[15] proposed more accurate scoring methods later on. The weakness in structure scoring is its incapability of identifying misconceptions, show the hierarchical difference, and providing analytical data of actual comparison. Yao [16] proposed a scoring technique based on the preposition chain (concept - linking phrase - concept relation). A preposition represents a logical relation between two concepts. The fallible analyzer [17], used for conceptual modeling, takes the preposition matching further by providing scores which are obtained by a student's concept map prepositions with an instructor's concept map prepositions.

Cmapanalysis [6] is another tool that analyzes the student concept map and provides statistical information to analyze a concept map. The tool generates an Excel file from a CXL format, an XML-based language. The paper evaluates concept map based on its *Size*, *Quality*, and *Structure*. The *Size* of a concept map is defined by three quantities: Number of Concepts, Number of Linking phrases, and the Number of Preposition. The quality of a concept map is measured by the Number of Correct Preposition (including the concepts and linking phrases used on the preposition). The instructor has to input the three most central concepts for each concept map. Cmapanalysis looks for those concepts in a preposition. *Structure* is an evaluation of four quantities - Centrality of Concepts, Number of Cross Links, Density, and Inter-Clustered Preposition Count. These quantities establish the hierarchy of a concept map and the centralization of core concepts. However, the actual comparison for instructor

and student concept map: synonyms, different structure of phrasing, misconception, and visual representation remains untouched. Also, the instructor has to manually input the essential core concepts to evaluate the quality of the student concept map. The sub-concepts from the instructor concept map are not compared with the student concept map which leaves a loophole for an accurate analytical score.

Compass [18], perhaps the best concept map analyzing tool, provides misconceptions, incomplete relationships, missing concepts, and linking phrases. Compass can identify two types of error - *Preposition Position Error* and the *Illegal Relation Error*. Also, the tool offers a personalized assessment process for the instructor to grade concept maps with their grading criterion. Even though Compass can identify misconceptions (incomplete relations), it does not isolate and display the missed and incomplete prepositions. Lack of a visual graph makes it harder for the instructor to isolate the critical misconception among the students. Besides, Compass requires the student to use concepts and linking phrases from two lists of options: list of available concepts, and list of available linking phrases. Therefore, the concepts from the instructor concept map are exposed to the student. This technique rules out the incident-like synonymous difference, but disregards the fact that a student might forget to mention a handful of concepts and linking phrases without the exposure. So, it would result in an advantage for the student where the only error a student could make is *misconception*. Additionally, the evaluating of the tool had been conducted on a concept map of 24 nodes. The accuracy of the tool for large and complex concept maps is a mystery.

Francisco [19] proposed to implement Ohlsson's theory to provide JIT (Just In Time) feedback while the student constructing the concept map. The student has to construct the concept map in a jigsaw puzzle [20] manner. Jigsaw Puzzle invoke to think logically and improve students' problem-solving skills. This technique is helpful for student to understand where a concept belong as he/she moved to complete the concept map. However, this technique does not evaluate a student's understanding of a topic. Therefore, the instructor cannot use this framework to grade their student. It is rather a study tool for students.

CohViz [21] is a feedback tool that demonstrates cohesion of written texts. It has been a popular tool to improve cohesion from a text passage, especially among students. The tool isolates semantic information from the text and creates a concept map structure. However, the tool is not capable of comparison, impairing instructors to use the tool to asses students' concept maps. On the other hand, CohViz can construct ambiguous and unambiguous references and moderately construct global and local cohesion. However, the height of the generated concept map is short. The tool is very accurate in constructing preposition for lower height concept map. However, the result would be defective when it comes to establishing a hierarchy for a larger concept map. Cronus considers both preposition (in terms of *misconception*) and *hierarchy* in comparison.

Recent approaches to evaluate a student's concept maps become futile when the student uses synonyms and different styles of language. Also, quantifying the hierarchical similarities between the instructor and student concept map has never taken into account, an essential for the understanding of a topic since not every concept of a concept map bears the same weight. Cronus has versatile uses - the instructor can use the student concept map to evaluate his/her understanding and retrieve analytical data to grade the concept map. Also the student can improve his conceptual understanding by building the concept map and comparing it with a standard version. Since the comparison is very time efficient for smaller number of nodes, the instant results would notify the student of missing concepts, linking phrases, and misconceptions.

Instructors prefer concept maps because it represents the related concepts in cohesion. However, when it comes to grading, the accuracy become a dominant factor on assessing the cohesion on student concept map. Andreas [22] sought to correlate the accuracy of concept map feedback tools with student's improvements on cohesion. The study found that students became frustrated with inaccurate feedback from any tool, disrupting students' preparation and impairing the instructor to grade properly. The researchers concluded that if the students receive any graphical or visual feedback, it help them the most to write cohesive explanatory texts. Cronus is capable of generating two visual PDF in response to each comparison. The first generated PDF is the comparison between the instructor and student concept map and the second PDF is the topological analysis of the student concept map. Subsection IV-C explores Cronus's accuracy with 78 concept maps in three groups from three separate modules. It has proven to be very accurate. Table 2 compares existing Concept map feedback tools with Cronus.

Cronus does not have an integrated concept map editor. It expects that the concept map is constructed on a different concept map editor and exported as a CXL, an XML-based language, file. The CXL format is necessary for building a dictionary of every possible branch of a concept map. The student and instructor sample we used to evaluate the accuracy of Cronus is generated by CmapTools [23]. It is developed by the Florida Institute for Human and Machine Cognition, is an advanced concept mapping tool that is intended for modular architecture and constructing knowledge models. The Cronus can represent the concept map of the instructor in a color-coded diagram visualizing which concepts and linking phrases are missing from the student concept map and analytical details including misconception.

## A. PROBLEM STATEMENT

Given two concept maps on a topic, one developed by a student while the other by an instructor, our goal is to compare both concept maps and provide visual feedback on the student concept map to the instructor. The feedback is a concept map that presents any misconceptions in the student concept map and highlights the nodes, linking phrases, and branches matched or partially matched between the instructor's and

TABLE 2. Comparing existing feedback tools with cronus.

Feedback Tools and Techniques	Topological Scoring	Contextual Scoring	Hierarchical Scoring	Prepositional Scoring	Graphical Display	Cohesion	Grading Template	Natural Language Processing	Misconception
Deshpande (2019)	Yes	No	No	No	No	No	No	No	No
Novak (1984)	Yes	No	Yes	No	No	No	No	No	No
Yao (2012)	No	Yes	No	Yes	No	No	No	No	No
Fallible Analyzer (2006)	Yes	No	No	Yes	No	No	Yes	No	No
Cmap Analysis (2013)	Yes	Yes	Yes	Yes	No	No	No	No	No
Compass (2005)	No	Yes	Yes	Yes	No	No	Yes	No	Yes
Cohviz (2020)	No	No	No	No	Yes	Yes	No	Yes	No
Cronus (2021)	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes

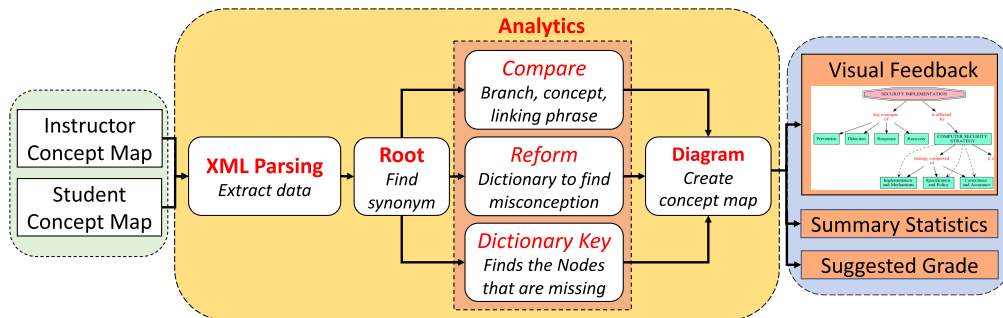


FIGURE 2. Cronus framework.

student’s concept map. Furthermore, we aim to quantify the feedback in summary statistics to suggest a grade for the student concept map based-on predefined criteria by the instructor.

**B. PROPOSED APPROACH - CRONUS**

Cronus generates two graphs - contextual analysis graph and topological analysis graph. The contextual analysis graph is built on instructor concept map and the topological analysis graph is built on student concept map. Contextual analysis graph provides node (concept and inking phrase) comparison information in the generated PDF, including a visual of the concept map properly structured and color-coded. The Grade depends on three aspects: *Hierarchy Match*, *Concept Match*, and *Correct Conception*. To calculate *Hierarchy Match*, Cronus weight the concepts existing near the root concept higher and gradually degrade the value as it proceeds towards the leaf node. The *Concept Match* compares the percentage of matched concepts with the instructor concept map; The comparing mechanism takes account of synonyms and different formats of a sentence, clause, and phrase structure. The *Correct Conception* is the proper

relationship between two concepts in the concept map. *Correct Conception* portrays the proper use of concepts and highlights the misconceptions from the student concept map. By default, the weight of *Hierarchy Match*, *Concept Match*, and *Correct Conception* are distributed evenly, each worth one-third of a hundred percent. However, the instructor can input a unique weight distribution when calling the function, including a grade curving mechanism (set to 0 by default). The letter grade would add the grade curve before displaying it beside the grade.

Figure 2 presents an overview of the Cronus contextual framework. Cronus takes two concept maps as input in XML format developed by an instructor and a student on a topic. It parses the XML files, performs a series of analogies to identify misconceptions, and finds the concepts, links, and branches matched and missed by the student concept map. In the end, Cronus generates a sorted color-coded concept-map diagram and provides summary statistics and a suggested grade for the student concept map. Figure 3 shows an example diagram presenting a sorted graph with the instructor hierarchy. The red-color boxes show the unmatched concepts and linking phrases, while the matched concepts and

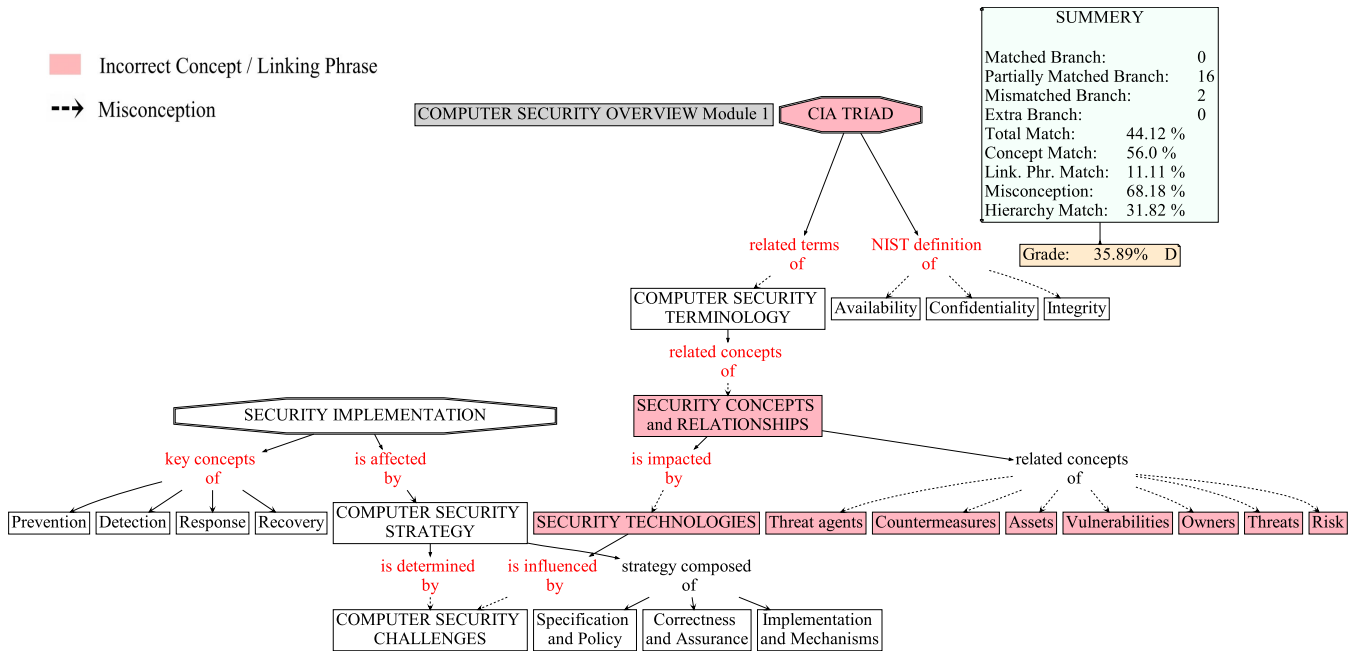


FIGURE 3. Cronus-generated contextual analysis graph.

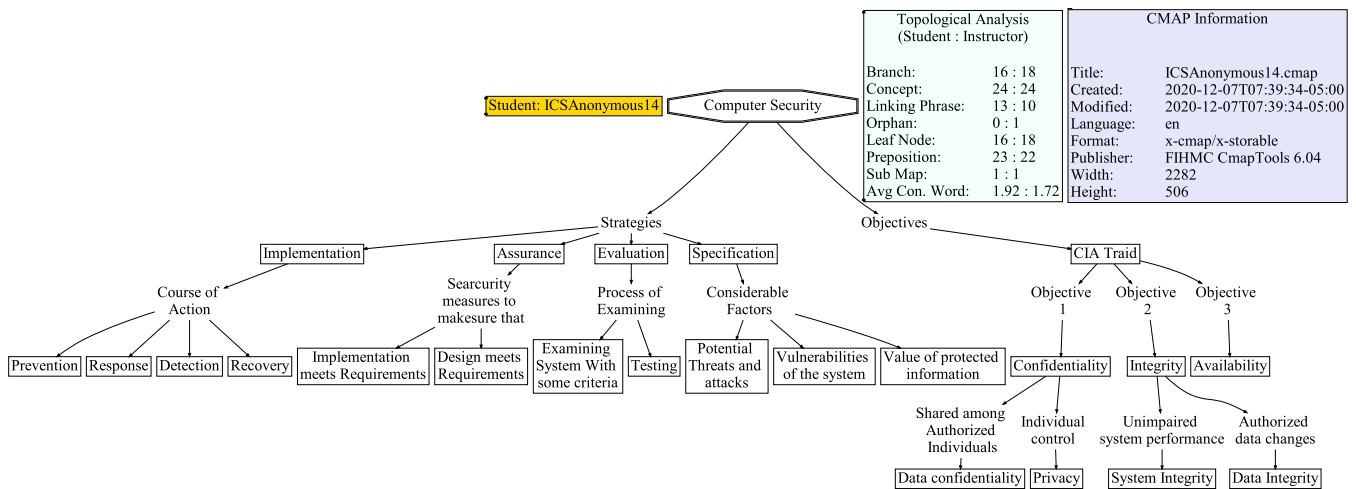


FIGURE 4. Cronus-generated topological analysis graph.

linking phrases are highlighted in green. The misconceptions are the incorrect connections between two concepts shown in dotted links.

Topological analysis graph includes the statistical comparison of the instructor’s and student’s concept map. The topological graph display the student concept map along with two information boxes. The first box includes the topological aspect of two concept map such as the number of *branches*, *concepts*, *linking phrases*, *orphans*, *leaf nodes*, *prepositions*, *sub maps*, and *average word per concept*. The second box display information about the creation and the modification of the student concept map. The topological graph also serve

the purpose of depicting the student concept map. Instructor might want to examine the student concept map for the better understating of the students’ cohesion. Therefore, both the contextual and topological graphs display unique comparison data of those two concept maps. The two graphs also serve the purpose of showing structural differences of the instructor’s and student’s concept maps.

Figure 4 presents an overview of the Cronus topological framework. Cronus displays the difference in topological identities in *student: teacher* format. By considering the differences in topological attributes, we can quantify the structural differences between student and instructor concept map.

The topological framework also generate a graph, structured from the student concept map. It helps us to identify graphical differences between two concept map (the contextual graph which is built on instructor concept map and the topological graph which is built on student concept map). Beside the topological analysis, the framework provides the concept map information such as title, when created, last modified, language, format, publisher, width, and height. While these information are unnecessary in the grading process, they can be beneficial in recording the name and version of concept map editor, and the last modification date.

### 1) XML PARSING

Extracting data from the concept maps is the first step of the process. An XML file can be extracted from a concept map file. It contains every node, linking phrase, and connection information which has a unique id. The strings (concepts and linking phrase) from the XML file is parsed and stored as a dictionary where each list represent a branch of the concept map and the keys for those lists are consecutive numbers starting from 1.

### 2) ROOT

When comparing two concepts or linking phrases from dictionaries, the words on strings are converted into root words, and the stop words are removed. A wide list of synonyms is listed for an individual concept or linking phrase of the instructor dictionary. To avoid word structural differences, every synonym of strings of instructor dictionary and strings of the student dictionary went through stemming and lemmatization to a point where all the non-stop words are in root format. Then they are compared with each other and check whether two strings are equivalent or synonymous.

### 3) ANALYTICS

After two dictionary is created from instructor and student concept map, and a system to compare two strings is established, Cronus would move forward to compare prepositions, branched, node and linking phrase.

#### a: COMPARE

All the branched of the instructor dictionary is compared with the student dictionary to find how many-branched are fully, partially, or did not match with the instructor dictionary including how many-branched are written extra in the student dictionary. Indexing the matched concepts and linking phrases, each branch can be grade hierarchically- the elements near the root concept weigh higher and the element near the leaf nodes weigh lower. The *mismatched branched* would be graded 0 while the branched matched completely would be graded 100. The average hierarchical grading for all the branched of the instructor concept map would give a score that represents the *hierarchy match* of the student concept map with the instructor concept map.

#### b: REFORM

The misconception is a critical error. To identify those errors the dictionary needs to be reformed to a concept-concept relation, a preposition without a linking phrase, dictionary. The values of the lists of the dictionary are two nodes that are connected hierarchically in the concept map- the first concept of a list has a higher rank than the second concept. The length of the dictionary is the number of the preposition in the concept map. The reformed instructor dictionary and reformed student dictionary are compared to find the percentage of correct concept-concept relation out of all the concept-concept relation in the reformed instructor dictionary.

#### c: DICTIONARY KEY

The total match, concept match, and linking phrase match quantify the percentage for all the elements, concepts, and linking phrases respectively. Dictionary from XML Parsing is utilized to isolate the concepts and linking phrases into two dictionary- one dictionary preserve the concepts in values and concepts' id in keys, and another dictionary preserves the linking phrases in values and its id in keys. The concept and linking phrase dictionary for the instructor and student concept map are compared. The unmatched instructor concepts and linking phrases are recorded into another dictionary.

### 4) DIAGRAM

The dictionary key from the unmatched concept and linking phrase dictionary is saved into a list. From the XML parsed data, another dictionary is formed from the instructor concept map that records the node and linking phrase connection ids on the list and a consecutive number in the key starting from 1. A diagram is created copying the connection id dictionary. If any id matches from the list of unmatched concepts and linking phrases, then the background of the node would be colored red. Otherwise, the background of the rest of the concepts would be green because those concepts were both present in the instructor and student dictionary. If a linking phrase present on the list on the mismatched concept and linking phrases, the font-color would be turned red; otherwise the font-color would be green because they are present in both instructor's and student's concept map. The correct conception (concept-concept connection) is depicted on the diagram with a dashed blue arrow. A summary box includes the summary statistics, including a suggested grade.

### 5) CRONUS OUTPUT

Cronus output comprises of three components:

1) Cronus provides visual feedback as a well-marked concept map after comparing the concept maps of a student and an instructor on a topic. Figure 3 provides an example of a Cronus generated concept map highlighting different types of concepts and links in colors, i.e., red and green for the unmatched and matched concepts and linking phrases respectively. The misconceptions are marked by dotted links between two concepts.

**Algorithm 1** Find\_Branch (*orphan*)

---

```

1: Clear the branch list
2: Set next to orphan node
3: while next! = None do
4:   Append the next node to branch
5: end while
6: Reverse the branch list
7: Return the branch list

```

---

**Algorithm 2** Process (*orphans*)

---

```

1: Clear the branch_dict dictionary
2: Set order equals to 0
3: while every element in a orphans list do
4:   Save find_branch(element) to branch
5:   Set branch_dict[n] equal to branch
6:   Increment n by 1
7:   Clear branch
8: end while
9: Return the branch_dict dictionary

```

---

**Algorithm 3** id\_dict (*directory*)

---

```

1: Declare an empty idConnection dictionary
2: Declare and set doc object to xml.dom.minidom.parse(directory)
3: Declare and set idConnectionList list to doc.getEle-
mentsByTagName('connection')
4: for every element in a idConnectionList list do
5:   Set idConnection[element.attributes['id'].value]
to [element.attributes['from-id'].value, element-
attributes['to-id'].value]
6: end for
7: return idConnection dictionary

```

---

**Algorithm 4** Clean (*phrase*)

---

```

1: Declare a no_stop_word list
2: Declare and set stopwords object to stopwords from nltk
corpus
3: Declare and set tokens list to all the words from phrase
4: for every element in a tokens do
5:   if element not in stopwords then
6:     Append it to no_stop_word
7:   else
8:     Nothing
9:   end if
10: end for
11: return no_stop_word

```

---

2) Cronus generates summary statistics based on the comparison. The stats parameters are described in Table 3.

3) Cronus utilizes predefined criteria (by the instructor) based-on the summary stats to suggest a grade for a student concept map, discussed further in Section IV-B.

**TABLE 3.** Parameters of summary statistics by cronus.

Parameter	Description
Matched Branch	The number of branches of the student's concept map fully matched with the instructor's concept map
Partially Matched Branch	The number of branches of the student's concept map partially matched with the instructor's concept map
Mismatched Branch	The number of branches in the instructor's concept map missing (not mentioned fully or partially) in the student's concept map
Extra Branch	The number of branches the student wrote extra that is not present on instructor's concept map
Total Match	The total percentage of the nodes in the student's concept map matched with the instructor's concept map
Concept Match	The percentage of concepts in the student's concept map matched with the instructor's concept map
Linking Phrase Match	The percentage of linking phrases in the student's concept map matched with the instructor's concept map
Misconception	The percentage of concept to concept connection in the instructor's concept map missing from the student's concept map (wrong connection or missing)
Correct Conception	The percentage of concept to concept connection in the instructor's concept map matched with the student's concept map (correct connection)
Hierarchy Match	The percentage of the hierarchy in the instructor's concept map matched with the student's concept map where the nodes closer to stem node have a higher weight than nodes close to the leaf node.

**Algorithm 5** Synonyms (*word*)

---

```

1: Declare an empty synonyms list
2: for every element_of_synsets in synsets(word) of word-
net do
3:   for every element_of_lemmas in lemmas() of ele-
ment_of_synsets do
4:     Append the name() of the element_of_lemmas
5:   end for
6: end for
7: Remove duplicates in synonyms list
8: return the synonyms list

```

---

**III. IMPLEMENTATION**

We provide sufficient implementation details and algorithms for reproducibility and reusability. Cronus is released on GitHub at <https://github.com/Masrik-Dahir/Cronus>.

The main module of the Cronus is *diagram.py*. The tool can analyze the CXL file only. At the beginning of the process, the concept map files need to be extracted into

**Algorithm 6** isSame(*instructor\_phrase*, *student\_phrase*)

---

```

1: Set  $n = 0$ 
2: if instructor_phrase is equals to student_phrase then
3:   return true
4: else
5:   for student_element in student_phrase do
6:     for instructor_element in instructor_phrase do
7:       Declare and set instructor_synonyms_list list
       to all synonyms set of the instructor_phrase
8:       for elements in instructor_synonyms_list do
9:         if any elements matches with
           student_element then
10:            increment  $n$  by 1
11:         else
12:           Nothing
13:         end if
14:       end for
15:     end for
16:   end for
17:   if ( $n$  is greater than or equal to the length of the
       instructor_phrase) and ( $2 * \text{length of } instructor\_phrase$ 
       greater than or equal to the length of student_phrase
       and ( $2 * \text{length of } student\_phrase$  greater than or equal to
       length of the instructor_phrase)) then
18:     true
19:   end if
20: end if
21: return false

```

---

**Algorithm 7** matched\_value\_advanced(*instructor\_list*, *student\_list*)

---

```

1: Declare an empty matched list
2: for every element_of_student in the set of student_list
   do
3:   for every element_of_instructor in the set of instructor_list
     do
4:     if Boolean result of two phrases are equal using
       justify() function of root module then
5:       Append element_of_instructor to matched
       list
6:     end if
7:   end for
8: end for
9: return the matched list

```

---

the CXL file. When the *diagram.dia()* function is called, the user has to input two directories on the parameter: the first parameter is the directory of the instructor, and the second is for the student. The function calls other functions from concurrent modules and generates two PDF to display results. The directory starts from the location of the/Cronus/library. The *diagram.dia()* function takes two types of parameters-string and list. In case the instructor needs to input two single, or listed concept map file directories, he/she can write those

**Algorithm 8** find\_match (*instructor\_dictionary*, *student\_list*)

---

```

1: Declare three high, h_key, h_point integer variables
2: for every instructor_key and instructor_value in the
   instructor_dictionary do
3:   Declare four variables point, val, instructor_list,
   student_list, matched_list and set the
   values 0, 0, instructor_value, student_list, and
   matched_value_advanced(instructor_list, student_list)
   respectively
4:   for every index, value in the enumerator of instructor_list
     do
5:     Increment point by the difference of the length of
     instructor_list and the value in index of instructor_list
6:     if point is greater than h_point then
7:       h_point is equals to point
8:     end if
9:   end for
10:  Declare a variable  $q$  and set it equal to 0
11: end for
12: for element in in the range of length of instructor_list -
   1, stopping at -1, and stepping -1 at a time do
13:   if length of instructor_list is equals to the length of
     student_list then
14:     Increment  $q$  by 1
15:     if  $q$  is equal to the length of instructor_list then
16:       Set h_key equals key
17:     else
18:       val equals to the length of matched_list
19:       if val is greater than high and
         h_point is greater than point then
20:         Set high to val
21:         Set h_key to key
22:       end if
23:     end if
24:   end if
25: end for
26: end for
27: Set val equals to length of the matched_list
28: if val is greater than high and
   h_point is greater than point then
29:   Set high to val
30:   Set h_key to key
31:   if val is greater than high then
32:     Set high to val
33:     Set h_key to key val is greater than high
34:     Set high to val
35:     Set h_key to key
36:   end if
37: end if
38: end if
39: return h_key

```

---

directories as a list on the first parameter and the same is true for the second parameter which is dedicated to the student directory.



**Algorithm 9** Concept (*instructor\_dictionary, student\_list*)

---

```

1: Declare three empty dictionary ins_c, con, extra_con
2: Declare two empty list variable con_values, con_values_extra
3: Declare two integer variable m, n and set those variables to 1, and 1.
4: for every instructor_key and instructor_value of instructor_dictionary do
5:   Declare a empty list dict_c_list
6:   for every index in instructor_value do
7:     Define and Set id to the index of instructor_value
8:     if the modulus of id is equals to 0 then
9:       Append index to dict_c_list
10:    end if
11:  end for
12:  Set the dict_c_list to the index of instructor_dictionary
13: end for
14: for every key and value of con do
15:   if value not in con_value then
16:     Append value to con_value
17:   end if
18: end for
19: for every key and value of extra_con do
20:   if value not in con_value_extra then
21:     if value not in con_value then
22:       Append value to con_value_extra
23:     end if
24:   end if
25: end for
26: Append the elements of con_value_extra to con_value
27: return con_value

```

---

**Algorithm 10** Comp (*instructor\_dictionary, student\_dictionary*)

---

```

1: Declare two tuples ins_concept and stu_concept and set the value from the concept(instructor_dictionary, student_dictionary)
2: Create four empty list variables result, result_no_duplicates
3: for every instructor_element in ins_concept do
4:   for every student_element in stu_concept do
5:     if the instructor_element and student_element are similar then
6:       Append instructor_element to result
7:     end if
8:   end for
9: end for
10: for every result_element in result do
11:   if result_element not in result_no_duplicates then
12:     Append result to result_no_duplicates
13:   end if
14: end for

```

---

**A. XML PARSING**

Once the directory is properly given, the program runs the *xmlPursing.py* module. The XML parsing libraries are used to parse the information from the CXL file in the *xmlPursing.py*. The *xmlPursing.find\_branch()* function find a complete branch for an orphan. The *xmlPursing.process()* turns the CXL files into dictionaries where the keys are a unique number (variable type: int) of branches and values are an entire branch of the concept map. The keys are consecutive numbers that start at 1; the number of elements of a dictionary is the length of the concept map. The *xmlPursing.py* module has a function called *xmlPursing.id\_dict()* which returns a dictionary that lists connections of concept-linking phrases or linking phrase-concept obtained from the CXL file. The *xmlPursing.py* uses *xml.dom.minidom, xml.etree.ElementTree*, and *re* modules.

**B. ROOT**

While comparing the concepts of student and instructor, several linguistic or synonymic differences are observed. Natural language processing (the *nlk* library) is used to compare possible synonyms. The *root.py* module is dedicated to comparing two strings and returns a boolean value indicating whether strings are similar or not. The root module uses string, word2number, nltk.corpus, and nltk libraries. The *root.clean()* function removes stop words from the strings after separating every word by space or special characters. The function also removes punctuations, special characters, extra space, and empty space from the string. The *root.synonyms()* return a list of synonyms for a word. The *root.isSame()* compares the instructor concept and student concept after the concepts (string) are passed through the *root.isSame()*. Every non-stop word on the instructor string goes through the *root.synonyms()* for any possible synonyms that match with the student non-stop words. The *root.isSame()* takes two strings (concept or linking phrase) and returns whether they are equal or not in Boolean.

**C. COMPARE**

The *compare.py* module compares two dictionaries and provides most of the analytical information. It does the most crucial task of calculating hierarchy match score for the student concept map and finds matched, partially-matched, mismatched, and extra branches. The *compare.find\_match()* function finds the student branch that matches close to the instructor branch. The *compare.matched\_value\_advanced()* finds the matched values of a branch that matches partially or fully to an instructor branch, obtained from the *compare.find\_match()* branch. The *compare.engine()* coordinates those matched strings (nodes or linking phrases) and follows a hierarchy equation to find a hierarchy match score for the student concept map.

---

**Algorithm 11** *Mismatched\_key\_list (instructor\_directory, student\_directory)*

---

```

1: Declare three empty list list_i, list_s, rt
2: Declare eight empty dictionary dict_key_i, dict_lf_i,
   dict_concept_i, dict_node_linking_i, dict_key_s,
   dict_lf_s, dict_concept_s, dict_node_linking_s
3: Set the value of keys from xmlParsing module to
   dict_key_s
4: Set the value of node from xmlParsing module to
   dict_concept_s
5: Set the value of lf from xmlParsing module to dict_lf_s
6: Update the dict_node_linking_s with the dict_concept_s
   list
7: Update the dict_node_linking_s with the dict_lf_s list
8: for every key and value in dict_node_linking_i do
9:   if value not in list_i then
10:     Append value to list_i
11:   end if
12: end for
13: for every key and value in dict_node_linking_s do
14:   if value not in list_s then
15:     Append value to list_s
16:   end if
17: end for
18: for every value_i in list_i do
19:   for every value_s in list_s do
20:     if value_i and value_s are similar then
21:       Append value_i to rt list
22:     end if
23:   end for
24: end for
25: Declare two integer variable num_node and num_lf and
   set them to 0
26: Declare two empty list variable num_node_list and
   num_lf_list
27: for every rt_element in rt do
28:   if the keys of instructor_directory is in
   dict_concept_i then
29:     if the keys of instructor_directory is in
   num_node_list then
30:       Append the key of instructor_directory to
   num_node_list
31:       Increment num_node by 1
32:     end if
33:   end if
34: end for
35: Remove duplicates from dict_concept_i and dict_lf_i
36: Declare a float variable per_node and set it to the
   num_node divided by the length of dict_concept_i and
   multiplied by 100
37: Round per_node with 2 decimal position
38: Declare a float variable per_lf and set it to the num_lf
   divided by the length of dict_lf_i and multiplied by 100
39: Round per_lf with 2 decimal position

```

---



---

**Algorithm 11 (Continued.)** *mismatched\_key\_list (instructor\_directory, student\_directory)*

---

```

40: for every element in list_i do
41:   if element not in rt and element not in result then
42:     Append element to result
43:   end if
44: end for
45: for every element in result do
46:   Append the value of element in instructor_dictionary
   to result_final
47: end for
48: Return result_final, per_node, per_lf, per_avg,
   num_i_node

```

---



---

**Algorithm 12** *Dia (instructor\_file\_directory, student\_file\_directory)*

---

```

1: Convert the CXL file to dictionary using xmlPursing
   module
2: Use compare module to calculate hierarchical_score
3: Find the misconceptions from the dictKey module
4: Retrieve the concept_match, total_match from reform
   module
5: for every element in the dictionary do
6:   Create a node and connect it to the next node
7:   Highlight missed concepts with Red background
8:   Change the front color for the missed linking phrases
9:   Dot the arrows that are flagged as misconceptions
10:  create a node to provide summery statistics
11:  create another to show suggested grade and connect
   it to the summery node
12:  Save the graph to a PDF and display it
13: end for

```

---

#### D. REFORM

The *reform.py* module is dedicated to finding misconceptions. Since node-relation is a unique relation, a unique dictionary needs to be formed. The *reform.concept()* function creates a dictionary of a single concept-relation- the key preserves the id numbers of the nodes and the value contains two concepts that are connected in the concept map. The *reform.comp()* function takes two concept-relation dictionaries and returns which concept relations are present in the student dictionary.

#### E. DICTIONARY KEY

The *dictKey.find\_match()* function takes a list of the instructor dictionary and runs it through the student dictionary and finds the closed list the algorithm can find. The algorithm first looks for the list from the student dictionary with the highest number of matched concepts for a particular list from the instructor dictionary. In case of a tie, the algorithm looks for the list with the foremost concepts in the

hierarchy (closest concept to the root node). Every list (branch) of the instructor dictionary is run through a loop in `dictKey.unmatched_key_list()` function. The primary function of the `dictKey.mismatched_key_list()` function is to find the matched concepts and unmatched concepts and append them to two separate lists and return those values.

**F. DIAGRAM**

The `diagram.py` module draws a diagram using the `Graphviz` library that demonstrates the hierarchy and a visual representation of the student’s understanding and an analytical summary. The correct concepts are indicated with the background color green, and the wrong concepts are indicated with the background color red. The linking phrases, when represented in red, characterizes that the student missed it and if it is represented in green, it characterizes that the student got it correct. The `diagram.dia()` generates a PDF file in the results folder with the same name as the concept map file of the student.

**IV. EVALUATION**

**A. CONCEPT MAP DATASET**

We evaluated `Cronus` on 74 concept maps on three modules, i.e., computer security introduction, user authentication, and cryptographic tools. The maps were created by the undergraduate (senior-level) students as homework assignments in an introductory computer security course. There were 30 students enrolled in the class and every student was required to develop one concept map for each module. Table 4 presents the summary of the dataset.

**TABLE 4. Concept map dataset for computer security course.**

Modules	Students	# Concept Maps	Avg. Concepts & Links
Introduction to Computer Security	30	26	50.34
User Authentication	30	23	69.16
Cryptographic Tools	30	25	80.04
Total	30	74	-

**B. COMPARISON & DISCUSSION**

Unlike other tools, one of the main advantages of `Cronus` is that it performs a thorough comparison of student’s and instructor’s concept maps and provides feedback similar to manual assessment. `Cronus` performs a thorough analysis and evaluates different parameters described in table 3. These parameters cover all the aspects of a student’s concept map including statistics, quality, and hierarchical features. Figure 3 show the output of `Cronus` for evaluating student concept map. The concepts and linking phrases are color-coded in red and green. The green color indicates that the student correctly identified the concept or linking phrase and its presence in both student’s and instructor’s concept map while the red color indicates that the student missed the concept or linking phrase present in the instructor’s

concept map. This statistical information helps in calculating the concept match, linking phrase match, total match, and hierarchy match parameters.

`Cronus` also compares the branches of an instructor’s concept map with the student’s concept map to check that the student fully understood the logical relations between concepts. When every concept and linking phrase of the student branch is matched with the instructor branch, the branch is considered as matched; For no matches and less than complete matches, it is considered as respectively mismatched and partial matched. This information is used in calculating the branch parameters in table 3

The misconception is another common error students do while developing a concept map. A misconception means linking two concepts that are logically not related to each other. Due to the importance of misconceptions in grading, `Cronus` highlights the correct concept relations (shown with blue-dotted arrows) and provides a misconception percentage as shown in figure 3.

**1) GRADING**

After evaluating different parameters mentioned in table 3, `Cronus` also suggests a grade. In order to calculate this grade, it focuses on three important aspects of a concept map:

- Organization: Logical format, proper hierarchy etc
- Concepts & Terminology: The concept map include appropriate concepts and linking phrases.
- Connection and Knowledge of the relationships among Concepts: No misconceptions and concepts are accurately connected.

So to calculate the grade, `Cronus` uses Total match, hierarchy match, and correct conceptions (the logical connection between two concepts, the opposite of misconception). The grade is calculated according to the formula in eq1. The instructor can also change the grading formula according to need.

$$Grade = \frac{1}{3}(TotalMatch) + \frac{1}{3}(HierarchyMatch) + \frac{1}{3}(100 - Misconception) \quad (1)$$

Time is a crucial factor in any automated grading tool. `Cronus` is time-efficient. The occupied time correlates with the total number of nodes (concepts and linking phrases) of instructor’s and student’s concept map.

Since every concept in the student concept map looks for a match from the entire instructor concept map while considering every possible synonym, the node and time relation becomes a power relation. The approximate time can be calculated with a power equation. Since time can impede lengthy concept maps to be graded quickly, we recommend calculating the estimated time for comparing concept maps above 500 nodes.

$$Time = 0.0035 * (node)^{1.98} \quad (2)$$

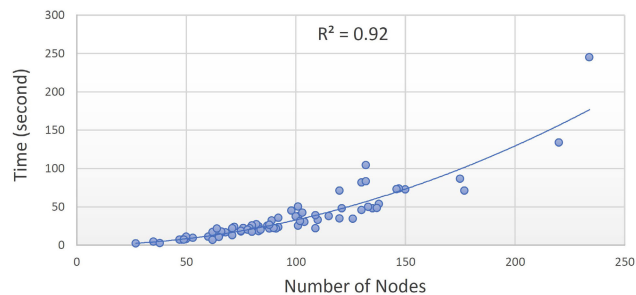


FIGURE 5. Node and time correlation.

TABLE 5. Difference between manual grading and cronus grading.

Module	Avg. difference in Total Match %	Avg. difference in Concept Match %
Introduction to Computer Security	2.56	3.85
User Authentication	2.54	3.80
Cryptographic Tools	5.6	11.2

C. ACCURACY

To measure the accuracy of the Cronus, we manually graded all 74 concept maps developed by students of the Computer security course. For each concept map, we calculated values of different parameters presented in the Cronus output. For most of the parameters, Cronus produced exactly the same value as calculated by manual grading except for “Total Match” and “Concept Match.”

As shown in table 5, in “Total Match (%)” we found an average difference of 2.56 between the manual grading and Cronus grading for the first module, 2.54 for the second module, and 5.6 for the third module. Similarly, in “Concept Match (%)” the average difference between the manual grading and Cronus grading for the first module was 3.85, 3.80 for the second module, and 11.2 for the third module. This small difference in the manual and Cronus grading for the above two parameters can be isolated into five categories.

1) PLEONASM AND MULTI-CONCEPT

When a student includes a concept into a detailed description, the representation of the concept changes. Cronus mismatches a student concept if over half of the words of a student concept do not match with a single instructor concept. The algorithm is written in a restrictive fashion to discourage pleonasm and including multiple concepts in a sentence in hope of a match. The phrases in a concept should be short and precise. To avoid this mistake, students are discouraged from writing wordy phrases and including multiple concepts.

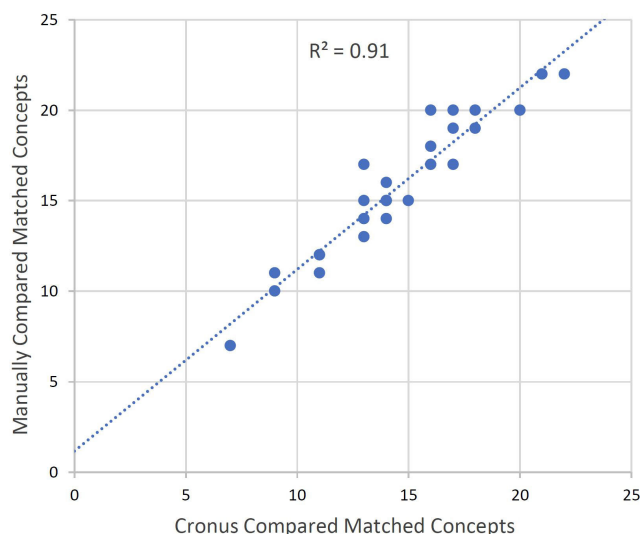


FIGURE 6. Grading comparison for “introduction to computer security” module.

2) ABBREVIATION OF CONCEPTS

If a student uses an unusual abbreviation that is not a standard synonym, the result would be a false negative. Missing Nonstop words does not impact the result because they are eliminated by Cronus during natural language processing. However, Some students inappropriately abbreviate nonstop (essential) words in a concept. Since they are not a standard replacement, Cronus counts it as a mismatch.

3) MISSPELL

Misspelling a concept keyword is a critical error. However, the grammatical error does not cause a false negative because every word on the concept is converted into a root word. The acceptance of a misspelled concept is a matter of the instructor’s judgment. The algorithm of Cronus can be perfected to flag misspelled concepts in the future.

4) VAGUE SYNONYM

While Cronus takes account of relevant synonyms, inapplicable synonyms are not accepted. A very thin margin of students used vague and poor synonyms to represent a concept. By definition, they cannot be counted as a synonym, but we counted them as a match for manual grading.

5) IRRELEVANT ORPHAN

A couple of orphan nodes in the instructor’s concept maps included the name of the module, instructor name, class name, or date. These orphan nodes are ignored in manual grading because they are not related to the actual topic and only serve as metadata. However, Cronus considers them relevant to the concept map, and try to find them in the student’s concept map and end up reporting them as missing nodes. This false negative is a result of irrelevant orphans in the instructor concept map; there are no issues with

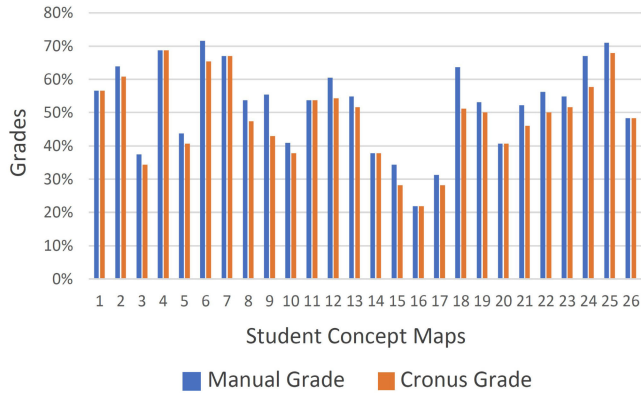


FIGURE 7. Grading comparison for “user authentication” module.

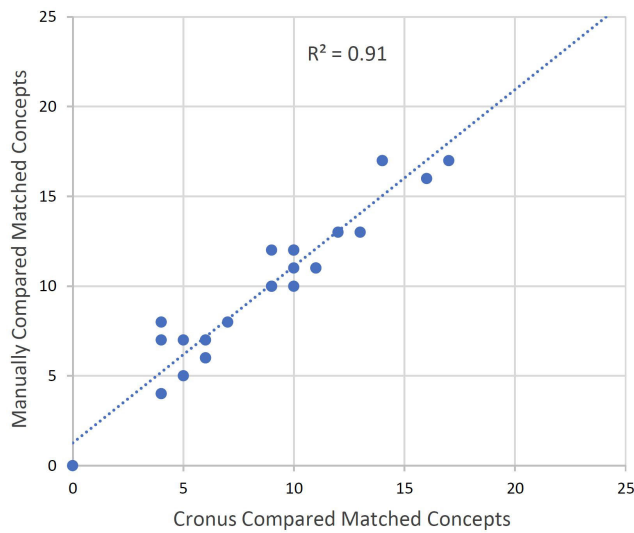


FIGURE 8. Concept Match comparison for “user authentication” module.

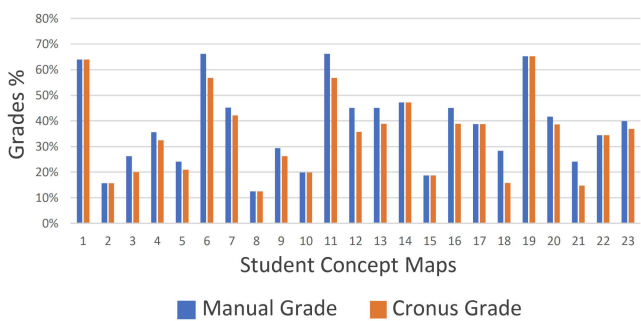


FIGURE 9. Grading comparison for “user authentication” module.

relevant orphans. So to get the perfect result, we suggest that the user removes such irrelevant nodes from the instructor’s concept map (master-concept map).

6) GRADING ACCURACY

Since there is minimal difference in the values of parameter evaluated manually, and by Cronus, the suggested grades are also similar. Figure 10, 9 and 11 show the comparison

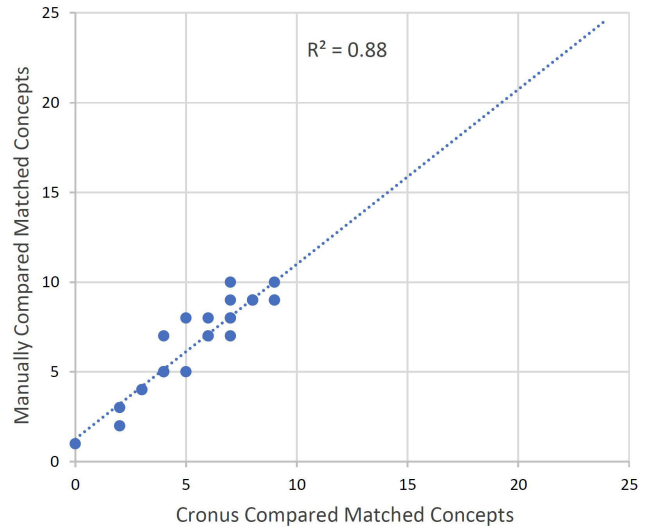


FIGURE 10. Concept match comparison for “cryptographic tools” module.

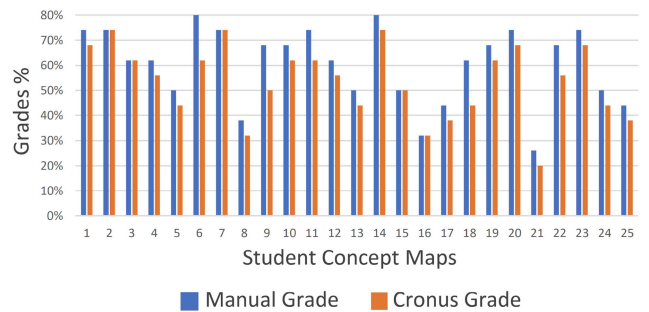


FIGURE 11. Grading comparison for “cryptographic tools” module.

between manual grade and Cronus grade for the three-course modules. The average difference between manual and Cronus assigned grades for the first module is 3.85%, 3.80% for the second module, and 6.72% for the third module.

V. CONCLUSION

In this paper, we presented Cronus, an automated tool for evaluating concept maps. Unlike other tools, Cronus provided a comparison of student’s concept maps with the instructor’s concept maps and generated visual feedback for quick assessment of student concept maps. It further quantified the feedback into useful summary statistics of evaluation parameters and suggested grades based on the grades and pre-defined instructor criteria for the maps. Our Results showed that the grading done by Cronus was significantly closer to the manual grading and can be used by instructors to evaluate concept maps for larger classes.

REFERENCES

[1] A. J. Cañas, J. W. Coffey, M.-J. Carnot, P. Feltovich, R. R. Hoffman, J. Feltovich, and J. D. Novak, “A summary of literature pertaining to the use of concept mapping techniques and technologies for education and performance support,” Inst. Hum. Mach. Cognition, Chief Naval Educ. Training, Tech. Rep., 2003.

- [3] C. Akinsanya and M. Williams, "Concept mapping for meaningful learning," *Nurse Educ. Today*, vol. 24, no. 1, pp. 41–46, Jan. 2004. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0260691703001205>
- [4] P. Kim and C. Olaciregui, "The effects of a concept map-based information display in an electronic portfolio system on information processing and retention in a fifth-grade science class covering the earth's atmosphere," *Brit. J. Educ. Technol.*, vol. 39, no. 4, pp. 700–714, Jul. 2008.
- [5] A. Anohina-Naumeca and J. Grundspenkis, "Scoring concept maps: An overview," in *Proc. Int. Conf. Comput. Syst. Technol. Workshop PhD Students Comput.*, Jan. 2009, p. 78.
- [6] P. Deshpande and I. Ahmed, "Topological scoring of concept maps for cybersecurity education," in *Proc. SIGCSE*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 731–737.
- [7] A. Cañas, L. Bunch, J. Novak, and P. Reiska, "Cmapanalysis: An extensible concept map analysis tool," *J. Educ., Teachers Trainers*, vol. 4, no. 1, pp. 36–46, 2013.
- [8] J. R. McClure, B. Sonak, and H. K. Suen, "Concept map assessment of classroom learning: Reliability, validity, and logistical practicality," *J. Res. Sci. Teaching*, vol. 36, no. 4, pp. 475–492, Apr. 1999.
- [9] A. Tanner and D. Dampier, "Concept mapping for digital forensic investigations," in *Advances in Digital Forensics V*, G. Peterson and S. Sheno, Eds. Berlin, Germany: Springer, 2009, pp. 291–300.
- [10] G.-J. Hwang, Y.-R. Shi, and H.-C. Chu, "A concept map approach to developing collaborative mindtools for context-aware ubiquitous learning," *Brit. J. Educ. Technol.*, vol. 42, no. 5, pp. 778–789, Sep. 2011, doi: [10.1111/j.1467-8535.2010.01102.x](https://doi.org/10.1111/j.1467-8535.2010.01102.x).
- [11] G.-J. Hwang, P.-H. Wu, and H.-R. Ke, "An interactive concept map approach to supporting mobile learning activities for natural science courses," *Comput. Educ.*, vol. 57, no. 4, pp. 2272–2280, Dec. 2011.
- [12] J. D. Novak and D. B. Gowin, *Learning How to Learn*. Cambridge, U.K.: Cambridge Univ. Press, 1984.
- [13] K. M. Markham, J. J. Mintzes, and M. G. Jones, "The concept map as a research and evaluation tool: Further evidence of validity," *J. Res. Sci. Teaching*, vol. 31, no. 1, pp. 91–101, Jan. 1994.
- [14] N. R. Pearsall, J. E. J. Skipper, and J. J. Mintzes, "Knowledge restructuring in the life sciences: A longitudinal study of conceptual change in biology," *Sci. Educ.*, vol. 81, no. 2, pp. 193–215, Apr. 1997.
- [15] J. D. Wallace and J. J. Mintzes, "The concept map as a research tool: Exploring conceptual change in biology," *J. Res. Sci. Teaching*, vol. 27, no. 10, pp. 1033–1052, Dec. 1990.
- [16] D. C. West, J. K. Park, J. R. Pomeroy, and J. Sandoval, "Concept mapping assessment in medical education: A comparison of two scoring systems," *Med. Educ.*, vol. 36, no. 9, pp. 820–826, Sep. 2002.
- [17] Q. Yao, K. Yang, G. Zhao, R. Huang, and J. Novak, "A concept mapping scoring algorithm based on proposition chains," Tech. Rep., 2012.
- [18] T. Conlon, "Formative assessment of classroom concept maps: The reasonable fallible analyser," *J. Interact. Learn. Res.*, vol. 17, pp. 15–36, Jan. 2006.
- [19] E. Gouli, A. Gogoulou, K. Papanikolaou, and M. Grigoriadou, "Evaluating learner's knowledge level on concept mapping tasks," in *Proc. 5th IEEE Int. Conf. Adv. Learn. Technol. (ICALT)*, Jul. 2005, pp. 424–428.
- [20] F. J. Álvarez-Montero, F. Sáenz-Pérez, and A. Vaquero-Sánchez, "Using datalog to provide just-in-time feedback during the construction of concept maps," *Expert Syst. Appl.*, vol. 42, no. 3, pp. 1362–1375, Feb. 2015.
- [21] L. E. Altener, "The learning curve in solving a jig-saw puzzle: A teaching device," *J. Educ. Psychol.*, vol. 26, no. 3, pp. 231–232, 1935.
- [22] C. Burkhart, A. Lachner, and M. Nückles, "Assisting students' writing with computer-based concept map feedback: A validation study of the CohViz feedback system," *PLoS ONE*, vol. 15, no. 6, Jun. 2020, Art. no. e0235209.
- [23] A. Lachner, I. Backfisch, and M. Nückles, "Does the accuracy matter? Accurate concept map feedback helps students improve the cohesion of their explanations," *Educ. Technol. Res. Develop.*, vol. 66, no. 5, pp. 1051–1067, Oct. 2018.
- [24] A. Canas, G. Hill, R. Carff, N. Suri, J. Lott, T. Eskridge, G. Gomez, M. Arroyo, and R. Carvajal, "CmapTools: A knowledge modeling and sharing environment," Tech. Rep., Sep. 2004.



**MASRIK A. DAHIR** is currently pursuing the bachelor's degree in computer science with Virginia Commonwealth University. He is also researching with the Security and Forensics Engineering (SAFE) Lab and the RamSec: Cybersecurity Lab. His research interests include malware, virtual machine introspection, cloud engineering, natural language processing, artificial intelligence, DDoS attack mitigation, and graph theory (applied mathematics).



**SYED ALI QASIM** received the B.S. degree in computer science from Lahore University of Management Sciences (LUMS), Lahore, in 2016. He is currently pursuing the Ph.D. degree with Virginia Commonwealth University (VCU). He is also working as a Research Assistant with the Security and Forensics Engineering (SAFE) Lab, VCU. His research interests include cybersecurity, industrial control systems, digital forensics, and the IoT.



**IRFAN AHMED** (Senior Member, IEEE) is currently an Assistant Professor of computer science with Virginia Commonwealth University (VCU). He is also the Director of the Security and Forensics Engineering (SAFE) Lab and a Faculty Fellow of the VCU Cybersecurity Center. His research interest includes the area of cybersecurity. His current research interests include digital forensics, malware, cyber-physical systems, and cybersecurity education.

• • •