CAES: A Model of an RBR-CBR Course Advisory Expert System

Emebo Onyeka, Daramola Olawande, Ayo Charles
Department of computer and information sciences, Covenant University, Nigeria
emebo.onyeka,dwande{@gmail.com},ckayome@yahoo.com

Abstract

Academic student advising is a gargantuan task that places heavy demand on the time, emotions and mental resources of the academic advisor. It is also a mission critical and very delicate task that must be handled with impeccable expertise and precision else the future of the intended student beneficiary may be jeopardized due to poor advising. One integral aspect of student academic advising is course registration, where students make decisions on the choice of courses to take in specific semesters based on their current academic standing. In this paper, we give the description of the design, implementation and trial evaluation of the Course Advisory Expert System (CAES) which is a hybrid of a rule based reasoning (RBR) and case based reasoning (CBR). The RBR component was implemented using JESS. The result of the trial experiment revealed that the system has high performance/user satisfaction rating from the sample expert population conducted.

1. Introduction

Advising plays an essential role in the retention and graduation of students in the university. One of the difficult and time-consuming tasks that university students and their advisors face today is individual course scheduling (assigning students to courses that satisfy their respective curricula). As the session progresses, the task becomes more complex due to the increase in the number of sequencing rules (e.g., prerequisites) that need to be satisfied by an advisee. Such a complex advising process may lead to decisions that can later inhibit a student from timely graduation. Thus, there is a need for a system that automates and simplifies the process for both students and advisors. It is important to realize that the course advisory expert system was not developed to replace the advisor but rather, it removes the time consuming tasks associated with course registration and allows advisor to concentrate on more complex advising functions.

Course advising is an activity in which faculty members advise students on which courses to take each semester in order to achieve their individual academic goals. For a university student to progress from one level to the next, he/she must meet up with certain numbers of credits as essential requirements.
Course Advisory exhibits characteristics favorable to an expert system approach in that– it is restricted to domain specific knowledge, uses voluminous data, is difficult to characterize accurately, curriculum changes constantly and decisions have to be made by stipulated rules of the university.
CBR is a concept of AI-problem solving that relies on knowledge gained from previous problem-solving episodes to resolve new problems once sufficient similarity between the current case (problem) and previously stored cases have been established. The justification for the CBR as our problem solving model is premised on contemporary experiences in the educational domain where a lot of similarities exist in the nature of academic problems and concerns that students have in the process of course registration. Hence our intent for implementing a RBR-CBR based expert system for student advising is to emulate human proficiency at drawing from experiences that are similar for solving problems at a reasonable level.
In this paper, we designed and implemented a course advisory expert system. The expert system is a hybrid of the rule based system and case-based reasoning. The main purpose of this system is to assist students and their advisors in providing timely and reliable course schedule for each student to register at the beginning of a new semester.

The remaining part of the paper is succinctly described as follows: In section 2 we elaborate on related work. Section 3 discusses CAES architecture. Section 4 gives a description of the course advisory process. Section 5 gives the decision algorithm for CAES. Section 6 details a case study report carried out in a tertiary institution. Section 7 reports a trial evaluation conducted. We conclude in section 8.

2. Related Work

A number of advising systems have been reported in literature that are mainly expert system based [1], or expert system and database hybrids [2,3].

A Software architecture of a new generation of advisory systems using Intelligent Agent and Semantic Web technologies was reported in [4]. The domain knowledge was modeled with the OWL ontology language. Using an inference engine the agents reason on the basis of their knowledge to make decisions or proposals.

A Planning Advisor on Curriculum and Enrollment was reported in [5]. A framework for an intelligent advisory system for college students that combine object-oriented and knowledge-based paradigms was presented. The model is presented for course advising based on students need to know "what to do" and "how to do it".

Interactive Virtual Expert System for Advising (InVEStA) was reported in [6]. InVEStA was proposed and developed to assist undergraduate students and their advisors in providing timely, accurate and conflict-free schedules. The proposed system was based on Java and object-relational database technologies and consists of the Database Layer, Transaction Layer, Scheduler and the web-based Front-End.

The Graduate Course Advisor (GCA), a Multi-Phase Rule-Based Expert System that advises graduate students in Computer Science was reported in [7]. It was implemented in Prolog, using an inference engine modeled after MYCIN. The advising task was divided into four phases, each of which may apply the inference engine to its own rule base and invoke other procedures.

In [8], AACORN (A CBR Recommender for Academic Advising), was introduced as a course recommendation system that uses the course histories as the basis for course advising. By reusing the experience embodied in historical student's transcripts, AACORN can make reasonable suggestions with only a limited amount of domain knowledge. The system uses the edit distance between two course histories, as well as other heuristics to determine the similarity between course histories.

Our approach in this work contrasts these previously reported approaches in that it is focused primarily on the aspect of student advising for online course registration using a combination of case-based reasoning (CBR) and rule based reasoning (RBR). In the next section, we discuss the hybrid architecture of CAES and how it works.

3. CAES Architecture

CAES comprises of the University database where information for each student are stored, the knowledge base where rules and structure of the courses are stored, the rule base engine that reasons on the available rules in the knowledge base and case-based engine that contains stored cases of previous advise.

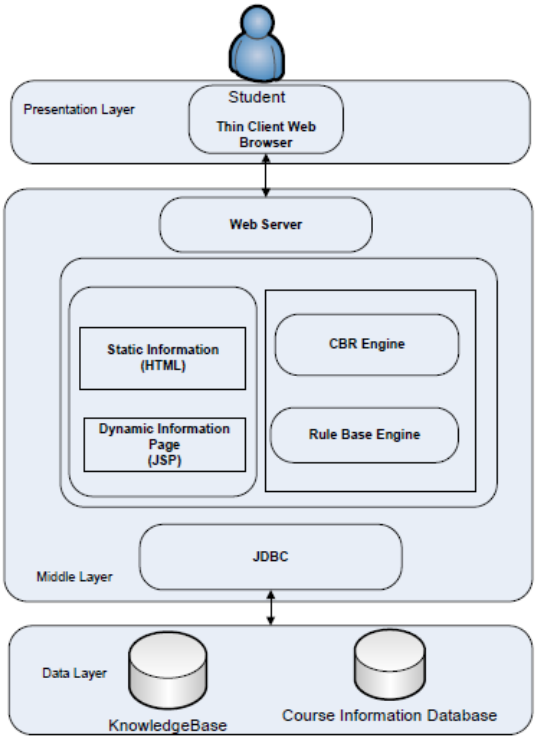

Figure 1. The 3-tier architecture of the system

The Figure 1 illustrates the architecture of the proposed system which is based on a 3-tier architecture. The Data Layer (bottom) contains all the knowledge sources that the system engages in order to provide information. This layer consist of a knowledge base that contains the facts and rules needed by the rule-based engine to provide advice and a relational database that contains details of course registration information of the university.

In the middle layer the most important component, the RBR engine implemented using Java Expert System Shell (JESS) [9], the CBR engine, and a web server with intrinsic capabilities to execute Java servlets and support JSP components. The web server handles communication with the external environment and routes external calls to appropriate components. In this particular case Apache Tomcat has been used as the web application server. Tomcat implements the Java Servlet and the JSP specifications, providing an environment for Java code to run in cooperation with a web server. Tomcat includes its own internal HTTP server. Communication with the data tier is through the JDBC (Java Data Base Connectivity) protocol.

The CBR engine which performs case base reasoning functionality and the rule base engine are solution deployed on the web application server.

The presentation layer contains forms which are used to interact with the system. Communication with the web server seated at the middle layer, is accomplished using HTTP protocol through a simple

web browser. Each student is able to connect into the system using the web browser available on the machine (laptop, desktop, PDA).

4. Description of the Course Advisory process

The student accesses the Expert system Interface Online with a valid identification number. On successful access the University course information database displays the Student details which entails his current standing on failed/dropped course if any and the set of course to register for the current semester. The University course information database is maintained by the University's database administrator. The knowledge base which is a component of the expert system is maintained by the knowledge engineer who models the rules as used by the human course adviser in advising student. The Inference engine comprising of the rule base engine and the Case-based engine are used to generate recommendation of courses to be registered in the current semester. The Inference mechanism checks to see if there are previous cases that are similar to the current case and uses such as a case for generating advise for the student. The report is sent back to the student via the Expert system interface. If no such advice case exists then the rule base inference engine processes the request and stores the recommended solution as a new case in the case base reasoning engine.
The Figure 2 is a schematic representation of the recommendation process using program flowchart.

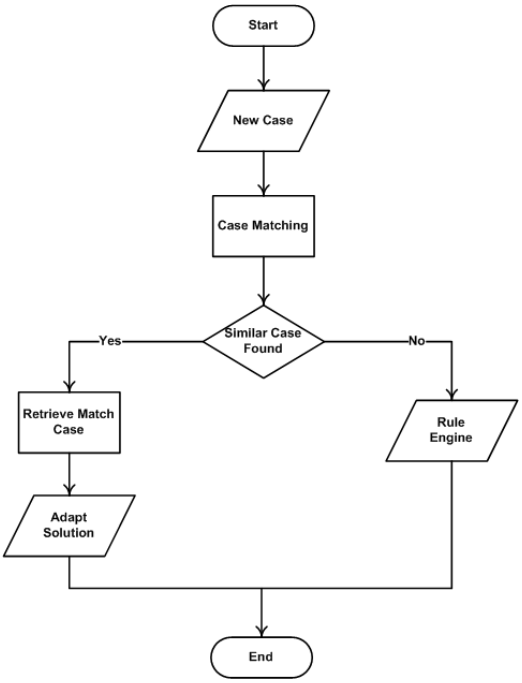

Figure 2. Schematic representation of CAES recommendation process

5. The Decision Algorithm for CAES

When CAES starts, the student course information is considered as a new case. CAES then computes a similarity score for the new case using the algorithm.

$$\text{Similarity (NC, OC)} = \frac{\text{common}}{\text{common + different}}$$

Where NC is the new case, OC is the old case present in the case base.
Common refers the matching pair between the new case and that of the old case.
Different refers the mismatch pair between the new case and that of the old case.

The case with the highest similarity score is picked as the candidate for adaptation in recommending courses to register else an appropriate decision algorithm based on the rule engine is executed.

6. Case Study and Discussion

A case study of Covenant University a tertiary institution based in Ota, Ogun State of Nigeria was carried out using the Computer science study program of the department of computer and Information science. For a student intending to register a course at the beginning of a new semester this scenarios exist.

- The student could have just the current semester course to register.
- The student could have failed course(s) alongside the current semester courses.
- The student could have dropped course(s) alongside the current semester courses.
- The student could have failed and dropped course(s) alongside the current semester courses.

In recommending the set of courses to register for the current semester, CAES uses the scenario above that is applicable to that particular student together with the set of rules outlined by the University policies for course registration, putting into consideration the different course status (course perquisites, compulsory or elective courses).
These rules were put together in the form of an algorithm as modeled in the rule base of CAES.

The REGISTERDROPPEDFAILEDCOURSE Algorithm caters for the first 3 mentioned scenarios.

Algorithm
REGISTERDROPPEDFAILEDCOURSE (**V**, **E, S**)

Input: A vector **V** of courses failed and/or dropped in the previous session of the same semester, E a vector of Elective courses and **S** a vector of courses to register in the current session of the same semester.
***Output*:** A vector **R** containing the list of courses recommended for registration by the student in that semester.

Initialize R.
[Considering Failed and Dropped courses]
for all courses v_i ϵ V ordered by coursecode in ascending order
while registeredCredit < maxRegistrable AND i < count(V)
Add v_i to R.
registeredCredit ← registeredCredit + courseCredit(v_i)
increment i.
[Considering failed prequisite course]
If registeredCredit < maxRegistrable
for each course C_j ϵ S
while registeredCredit < maxRegistrable **AND** j < count(S)
if prequisite(C_j) is failed **OR** dropped
then Add C_j to D
else
Add C_j to R
S ← S- C_j
registeredCredit ← registeredCredit + courseCredit(C_i)
increment j.

[For the remaining courses]
If registeredCredit < maxRegistrable
for each course K_p ϵ S that is compulsory ordered by course credit in descending order
while registeredCredit < maxRegistrable **AND** p < count(S)
Add K_p to R
registeredCredit ← registeredCredit + courseCredit(K_p)
increment p.
If registeredCredit < maxRegistrable
for each course M_e ϵ E that is elective
while registeredCredit < maxRegistrable **AND** e < count(E)
Add M_e to R
registeredCredit ← registeredCredit + courseCredit(M_e)
increment e.
return the vector **R** containing the list of recommended course for the semester.

Algorithm REGISTERCOURSE (E, S) caters for the last scenario.

Algorithm REGISTERCOURSE (**E, S**)

Input: A vector **E** of Elective courses and **S** a vector of courses to register in the current session of the same semester.
***Output*:** A vector **R** containing the list of courses recommended for registration by the student in that semester.

R ← NULL [initialize R]
for each course C_i ϵ S
while registeredCredit < maxRegistrable AND i < count(S)
if prequisite(C_i) is passed
Add C_i to R
registeredCredit ← registeredCredit + courseCredit(C_i)
increment i
If registeredCredit < maxRegistrable
for each course K_j ϵ S that is compulsory ordered by course credit in descending order
while registeredCredit < maxRegistrable AND i < count(S)
Add K_j to R
registeredCredit ← registeredCredit + courseCredit(K_j)
increment j
If registeredCredit < maxRegistrable
for each course M_e ϵ E that is elective
while registeredCredit < maxRegistrable AND e < count(E)
AddM_e to R
registeredCredit ← registeredCredit + courseCredit(M_e)
increment e
return the vector **R** containing the list of recommended course for the semester..

7. Evaluation

A usability evaluation of the prototype was conducted using human-expert evaluation to determine the level of performance/user satisfaction of the system and validated by using a direct method as used by Salim et al in [10].

A small experiment to test the system's recommendations against those of human advisors was conducted using the direct method. Course Advisors across each level from the Department of Computer and Information Sciences of Covenant University were asked to participate in the survey. Each received an identical set of questionnaire alongside a copy of CAES. The course advisors were asked to rank from one to five (TRUE to FALSE) the recommendation of CAES.
A description of the direct method test instrument completed by each evaluator is as follows:

1. The evaluator obtains demonstration or sample copies of the software packages to be evaluated.

2. The evaluator selects a benchmark problem, based on his experience, and runs this problem on CAES.
3. After running the bench-mark problem, the evaluator responds to the 14 questions in the instrument and estimates a quantitative answer to each question on a 0 to 5 scale with 5 being very true and 0 being very false.
4. Each numerical result is multiplied by a weighting factor as given in the weight column.
5. The weighted values are summed and then divided by 19 the sum of the weights to give a result in the numerical range of 0 to 5.

The Figure 3 gives a computation of the evaluation experiment conducted by one of the evaluator.

	Question	Assessment Value	Weight	Value X Weight
	Correctness of Answer			
1	Is there enough information to evaluate the software?	4	(2)	8
2	Does the software give the same answer that a human advisor would give?	5	(2)	10
3	Does the software provide the right answer for the right reasons?	5	(2)	10
	Accuracy of Answer			
4	Is the software accurate in its answer(s)?	5	(2)	10
5	Is the answer complete? Does the user need to do additional work to get a usable result?	4	(2)	8
	Correctness of reasoning technique			
6	Does the answer change if new but irrelevant data is entered into the software?	5	(1)	5
7	Does the system require a lot of irrelevant question to reach its answer?	0	(1)	0
	Sensitivity			
8	Does the answer change if irrelevant changes are made to the system rules?	5	(1)	5
	Reliability			
9	Does the software crashes or hang ups in its host computer?	2	(1)	2
10	Does the system give warnings for cases involving incomplete data or rules?	5	(1)	5
	Cost Effectiveness			
11	Does the software still provide answers with incomplete knowledge	2	(1)	2
12	Is the cost of the system justified by its performance?	5	(1)	5
	Limitations			
13	Can limitations of the system be detected at this point in time?	4	(1)	4
14	Can the system learn from increased data or experience?	2	(1)	2
	Result = Σ (weight x value) / Σ (weight)		19	76
			4.00	

Figure 3. Evaluator's questionnaire

A subset of the summary result in calculating the experimental evaluation of the evaluators is given in the Table 1 below.

Table 1. Result of Evaluation Experiment

Evaluator	Computed Satisfaction Level
1	4.00
2	4.16
3	4.21
4	3.52
5	3.57
Mean Satisfaction Level	3.89

From the statistical analysis of the results obtained from the evaluation of the human experts that per took in the experiment, CAES had a mean satisfaction level score of 3.89, which is indicative of a 77.8% level of user satisfaction.

The result revealed that the system has a performance rating/user satisfaction of 77.8% from the sample human expert population used for the trial experiment.

8. Conclusion

The proposed CAES system is intended for use in mid-range universities. Currently, its experimental version is launched at the Department of Computer and Information Sciences of Covenant University, its modular structure and web based design makes it possible to be launched and used elsewhere.
In our future work we hope to elaborate more on the case revision and case adaptation of the case based reasoned and also issues relating to data security and database mapping, in order to prevent unauthorized access to data.

As its contribution, this project offers a demonstration of application of modern AI-approaches for evolving important computer-based systems that can be used to resolve crucial business and operational concerns in the educational domain. While an expert system will not replace the need for wise and sympathetic counsel from human advisors, CAES focuses students more clearly on issues to consider and let them have unhindered access to the expert system before seeking further advise, thus alleviating academic staff of part of their burden.

9. References

[1] Harlan R. M, "The Automated Student Advisor: A Large Project for Expert Systems Courses," *ACM SIGCSE Bulletin Vol.* 26(1), pp. 31- 35, 1994.

[2] Rao T. M, Coleman S., Hollenbeck C., "ADVISOR – An Expert System for Student Advisement," *Proc. 15th Annual Conference on Computer Science, St Louis*, pp. 32-35, 1987.

[3] Murray W. S, LeBlanc L. A., "A Decision Support System for Academic Advising," *Proc. 1995 ACM Symposium on Applied Computing*, pp. 22-26, 1995.

[4] Dunkel J., Bruns R., "Software architecture of advisory systems using agent and semantic Web technologies" Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence, Volume 19, pp. 418 – 421, 2005.

[5] Gunadhi H., Kwang-Hui Lim, Wee-Yong Yeong, "PACE: a planning advisor on curriculum and enrolment",

Proceedings of the Twenty-Eighth Hawaii International Conference on System Sciences, Vol. 3, pp. 23 – 31, 1995.

[6] Pokrajac D., Rasamny M., "Interactive Virtual Expert System for Advising (InVEStA)", 36th Annual Frontiers in Education Conference. pp. 18-23, 2006.

[7] Valtorta M.G., Smith B.T., and Loveland D.W., "The graduate course advisor: a multi-phase rule-based expert system" Proceedings of the IEEE Workshop on Principles of Knowledge-Based Systems, 1984.

[8] Sandvig, J.J. and Burke, R. "AACORN: a CBR recommender for academic advising", Technical Report, TR05-015, DePaul University, 2005.

[9] Friedman-hill, E., "JESS, The rule engine for the java platform", from http://herzberg.ca.sandia.gov/jess/. Access date: November 28, 2009.

[10] Salim MD, Villavicencio A., Timmerman M.A., "A Method for Evaluating Expert System Shells for Classroom Instruction", *Journal of Industrial Technology*, Vol. 19(1), 2002.