Chapter 9

USER MODELING FOR COURSE PLANNING AND SCHEDULING

Fuhua Lin, M. Ali Akber Dewan, Alex Newcomb
School of Computing and Information Systems, Athabasca University

Abstract: Course offering determination (COD) is a complex task for the educational institutions by which they decides what subset of courses an academic department or program should offer in a given academic term or semester. For an effective COD, historic data in enrollment, constrained in budget, staff, and resources, and student's course selection preferences and priorities are needed to be taken into account. A poorly designed COD may lead to a low enrollment to the program, delayed graduation, and students increased dissatisfactions. A multi-agent based framework to facilitate COD is proposed by the authors in (Lin & Chen, 2013), which uses Contract Net Protocol, Single Transferable Voting, and Monotonic Concession Protocol. The multi-agent system (MAS) consists of an administrator (AD) agent, a group of student (SA) agents, and a student representative (SR) agent. In this paper, the modeling and implementation details of the SA agents are presented. The prototype of the system is implemented using Java Agent Development Framework (JADE). The system is expected to solve the problems in optimization and with flexibility in a fair and rational way, which can balance the competing needs of academic requirements, economics, and student preferences.

Key words: Multi-agent systems, agent-oriented software engineering, JADE, course-offering determination, Contract Net Protocol.

1. INTRODUCTION

Course offering determination (COD) is a process of deciding what courses of an academic program or school will be offering for the upcoming one or more semesters (Lin, Newcomp, & Armstrong, 2012). There are many factors that an administration needs to take into account in order to provide an effective list of offering courses. These factors include historic data in enrollments, budget, staffing and resource constraints, students preferences and priorities. A department of an academic institution may not be able to offer all courses in a program every semester, especially under contemporary fiscal and staffing constraints. However, a poorly designed COD can lead to several problematic situations, such as low enrollment, delayed graduation of the students, and students increased dissatisfactions. Some courses could only be arranged every other semester or even less frequently.

In the current course offering workflow, when the registration period for a new semester approaches, a course delivery schedule becomes available and the student can select courses to be taken in the coming semester. The competing or even adversarial goals of students and the department as well as the mutability of those goals indicate that COD is a complex constraint-satisfaction problem. Effective COD permits the efficient assignment of limited resources like faculty, labs, and classrooms, while satisfying the desires of most students.

Multi-agent system (MAS) allows the representation of every principal in a system as a single autonomous agent with unique goals and permits decision-making based on the preferences of multiple agents (Weiss, 1999) (Conitze, 2010). The MAS approach can be used to solve the constrained-satisfaction problem of COD because of the following reasons — (i) optimal solution of this problem can be changed during run time; (ii) relation between a user and the scheduling system lasts for a long period of time, which increases the possibility of learning by feedback; (iii) COD is a time consuming and tedious task using manual process; (iv) multiple-parties are involved in CODs (i.e., program administrators and students), all of which are required to be satisfied at least a minimum level from the provided solution; and finally, (v) unpredictable job market and student preferences needs to be taken into account fast and flexibly to environmental variables and their changes. Since the goals of the students and program administrators are different, therefore, there are conflicts of interests between them. These conflicts should be resolved in a fair cooperative decision making manner. The main research question of COD that is to be solved using MAS is "*what COD strategy of a program in an institution maximizes the satisfaction of the students and the enrollment of the courses within institutional constraints (i.e., limited budget, staff and teaching resources)?*"

In (Lin & Chen, 2013), authors modeled COD as a multi-agent constraint resource allocation problem and designed a mechanism to identify optimal solutions using voting and agent negotiation. Various theories of Agent-Oriented Analysis and Design (Wooldridge , Jennings, & Kinny, 2000) and

Agent-Oriented Software Engineering (AOSE) (Winikoff & Padgham, 2013) have been applied in modeling this framework. In agent-based recommendation applications, users need to feel easier and more comfortable to express their goals for obtaining items rather than specifying the features of the items. In view of this, this multi-agent framework is consisted with three different types of agents – Student Agent (SA), Student Representative Agent (SR), and Administrative Agent (AD), where the relations among these agents are modeled using contract net protocol (Smith, 1980), single transferable voting (Brams & Fishburn, 2002), and monotonic concession protocol (Endriss, 2006). Since, the overall system is very large and complex, this paper mainly focus on modeling and implementation details of SA within JADE and JASON framework.

The organization of this paper is as follows. A brief literature review relevant to this work is presented in Section 2. Section 3 provides details of SA modeling and implementation. Experimental results are presented in Section 4. Finally, the paper ends with the concluding remarks.

2. LITERATURE REVIEW

Recommender systems have gained considerable interest since the 1990s as a means of helping users to deal with ever-increasing problems of information overload (Resnick & Varian, 1997) (Burke, 2002). In the field of education, researchers have used different recommendation techniques for different purposes, such as suggesting online learning activities or optimum browsing pathways to students (Farzan & Brusilovsky, 2006) (Tang & McCalla, 2005), making recommendations to courseware authors (Garcia, Romera, & Castro, 2009), and providing advice to high school students and college freshmen that are seeking a potential major (Grupe, 2002). There are three primary approaches for computing recommendations: content-based techniques, collaborative filtering, and demographic techniques. Content-based techniques rely on the availability of descriptive metadata that captures the essence of the items available for recommendation and compute similarities between items by comparing item characteristics. Collaborative filtering techniques provide an alternative strategy that replies on rating-based user profiles instead of descriptive meta-data (Schafer, Frankowski, Herlocker, & Sen, 2007) (Smyth & Cotter, 2001). Demographic techniques make recommendations based on demographic classes, by analyzing personal attributes of the user (Krulwich, 1997). Likely items for recommendation are identified because clusters of users with similar personal attributes have demonstrated similar needs or tastes.

COD is a kind of recommender system, where MAS has been proven to be as an effective solution (Lin & Chen, 2013). However, the majority of such works have been in the area of providing learner support and developing learning environments (Smyth, Shang, Shi, Chen, & Shing, 2001). In fact, the best sources of good research on applying MAS to problems of planning and scheduling, preference elicitation, and negotiation may be found outside the educational realm in sectors such as manufacturing (Shen, 2002) and health care (Kirn, Herzog, Lockemann, & Spaniol, 2006). The potential effectiveness of decision support system and other AI systems in supporting administration in an increasingly complex education sector have also been recognized (Kannan & Kasmuri, 2005). There has been a relative paucity of works focusing in the role of advisor or administrator in providing educational resources and support save for work on problems with a tradition of AI applications, such as automated course scheduling and computer-aided academic advising (Opera, 2007).

Hamdi (Hamdi, 2006) proposed a system Masacad that tackles the program planning problem using MAS approach and used neural network aiming at finding the correct agent function. It is a multi-agent information customization system that adopts machine-learning paradigm to advice students by mining web information. Vainio and Salmenjoki (Vainio & Salmenjoki, 2005) proposed an agent-based approach to designing and updating a personalized study plan in collaborative environment. The system is based on learning agents able to suggest a study plan and if needed identify potentially problematic choices in the future, thus bring dynamics in the system. It shows that collaborating with other student agents in a multi-agent environment, the chances of finding a mutually beneficial result is improved. Bruns (Bruns, 2006) presents software architecture of a new generation of advisory systems using intelligent agent and semantic web technologies. To the best of our knowledge, the system proposed in (Lin & Chen, 2013) by the authors of this paper is the only system that addresses COD problem within a MAS framework.

3. SYSTEM ARCHITECTURE

The overall architecture of the proposed system for COD based on MAS is shown in Figure 1. The proposed MAS system consists of an administrator agent (AD), a student representative agent (SR), and a group of student agents (SAs). This system correlates with an academic program, where a course scheduling process is initiated by having the program administrator determined the priority of courses available in the program based on expressed student needs, preferences, and goals. The agents have distinct areas of concern and intent, but collectively interact to generate a set of recommendations for courses to be offered that will be satisfactory to most students while fitting within the operational limitations of the offering program. Considering the large scale and complexity of the whole system, we have limited our scope on modeling and implementing the SAs of the proposed system in this paper.

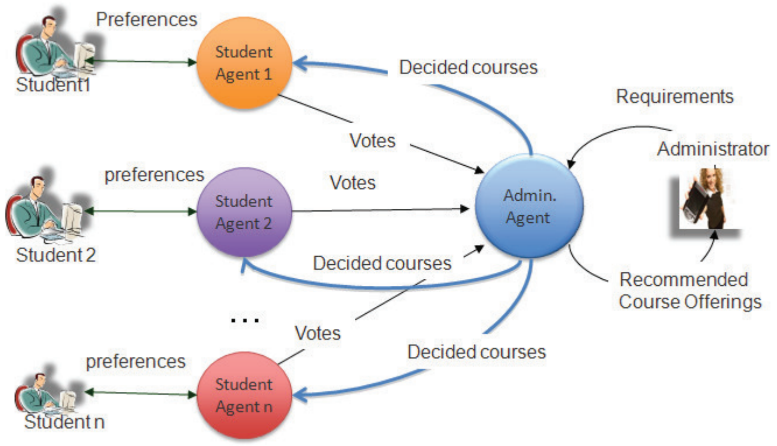


Fig. 1 The system diagram

When a student enters a program of study, the system creates a SA in the server. The SA runs in the background until the student is graduated from the program. The SA can be configured by the student with the basic information and profile of the student via a web-based interface. The process is started by the student who requires a study plan when s/he log into the system at the first time. Here we assume that the student has already chosen his/her program, and therefore, the curriculum of the program with a specific entrance year has been automatically decided by the system (Lin & Chen, 2013). This means that there are some mandatory courses the student must complete them to be graduated.

The COD problem is now concerned with the optimal offerings of courses for the upcoming semesters to meet the needs of the students in the program within budget and with scarce departmental resource, and maximize the course enrollment, while the goal for the students is to minimize the waiting time for desired courses to complete his/her graduation. In the system, human participants include students and program administrators, who exchange information and proposals in order to solve course selection tasks, and program planning tasks of the students, and the COD tasks of the program administrators. As shown in Figure 1, a society of software agents, including AD and a group of SAs work together to find an optimal solution for COD. In this context, the students and the program administrator can be identified as the coordination entities, which are modeled as autonomous agents, where the SAs try to acquire the desired courses to take for some semesters from the AD agent. The primary responsibilities of a SA are of two folds: first, eliciting and reasoning about and learning the planning requirements and program preferences from students and generating personal plan; second, cooperating with AD agent to iteratively generate course delivery schedules collecting course selection preferences from students. On the other hand, the responsibility of an AD is conducting COD with SAs to generate course delivery schedules. The process of performing the responsibilities of SAs is detailed in the following two subsections:

3.1 Planning Program Preferences of SAs

For the planning of program preferences for SAs, a similar method as used in (Linden, Hanks, & Lesh, 1997) has been adopted. Let $\mathbf{C} = \{c_1, \dots, c_m\}$ be a set of m elective courses in a given program of study. Let $\mathbf{P} = \{p_1, \dots, p_n\}$ be a set of n attributes of the courses. For example, three attributes about assessment style are $p_1 = "Exam"$, $p_2 = "Project"$, and $p_3 = "Assignment"$. For a course c_i, an attribute value p_j is defined as a real number in $[0, 1]$, which represents the degree of correlation between c_i and p_j, $\{r_{ij}: i = 1, \dots, m; j = 1, \dots, n\}$. Therefore, course metadata for **C** and **P** forms a relation matrix, denoted as $\mathbf{R}(\mathbf{C}, \mathbf{P})$ representing domain knowledge between a set of courses **C** and a set of attributes **P**. $\mathbf{R}(\mathbf{C}, \mathbf{P})$ can be acquired course-by-course from the course syllabi or from course instructors. For example, metadata for three courses, $\mathbf{C} = \{c_1, c_2, c_3\}$, and three attributes for each of them, $\mathbf{P} = \{p_1, p_2, p_3\}$, are presented as follows:

$$\mathbf{R}(\mathbf{C}, \mathbf{P}) = \begin{pmatrix} \mathbf{R}(c_1, \mathbf{P}) \\ \mathbf{R}(c_2, \mathbf{P}) \\ \mathbf{R}(c_3, \mathbf{P}) \end{pmatrix} = \begin{pmatrix} 0.3 & 0.5 & 0.2 \\ 0.5 & 0.2 & 0.3 \\ 0.2 & 0.3 & 0.5 \end{pmatrix} \tag{1}$$

From the relational matrix **R**, it can be observed that a student preference for an attribute $p_j$with respect to a course c_i is expressed as a weight,$w_{ij} \in [0,1]$, indicating the degree of preference for c_i to p_j. The larger the weight is, the more preferable the student thinks. The weight is updated by a machine-learning algorithm, which is detailed in the next sub-section. Thus, for each student, s, it elicits a weight vector, $\mathbf{w}_p(s) = \{w_j: j = 1, \dots, n\}$. For example, for student "John", $\mathbf{w}_p(John) = \{1.0, 0.5, 0.1\}$ indicates that "John" prefers "*Exam*" to "*Project*", and the "*Assignment*" is less preferable.

To ranking a list of courses, a metric called Degree of Desirability (*DoD*) is used. For a course c_i, *DoD* for a student *s* is calculated by summing up the normalized product of the student's preference weights $\mathbf{w}_p(s)$ and course metadata $R(c_i, P)$, and then dividing by the sum of the student's preference item weights for a weighted sum. By the assumption that the preference structure is additive independence, we construct an error function which provides a partial ordering over all solutions. For course c_i, its preference weight, $DoD(s, c_i)$ is determined by the formula:

$$DoD(s, c_i) = \frac{1}{\sum_{j=1}^{n} \mathbf{w}_{pj}} \left(\mathbf{w}_p(s). R(c_i, P) \right) \tag{2}$$

For example, in the above example, student "*John*" specified his preference $\mathbf{w}_p(John) = \{1.0, 0.5, 0.1\}$, we have, $DoD(John, c_1) = 0.36$, $DoD(John, c_2) = 0.39$, $DoD(John, c_3) = 0.25$. Thus, if only considering John's preferences about the assessment style, the most preferable course is c_2.

3.2 Preference Weight Update Using Machine Learning

For updating preference weights, machine learning technique has been used. For the students start at an initial value for all preference attributes, which we set 1. In general, we want to have such a modification setup that the preference weights increase/decrease sharply with initial changes, but more slowly with the similar changes later on. This assures that trends change rapidly enough so that there will be significant differences in plans generated in the early phases of machine learning, but that once stronger trends are created they will not be greatly offset by any false assumptions. For learning the model of student preference weights, $\mathbf{w}_p(s)$, we use a tangential model (equation 3) with a maximum (horizontal asymptote) at 5 and minimum at 0. This range was chosen because the advisors who set the weights for the importance of preferences to individual courses are given the same range.

$$y = \frac{5}{3} \times \tan^{-1}\frac{x}{2} + 2.5 \qquad (where\ x \in \mathbb{Z}) \tag{3}$$

In Figure 2, the *y* axis is the weight and the weight modification process moves the position linearly along the *x* axis. It can be observed from this curve that when the weight is between 1.5 and 3.5, it is changed quickly and almost linearly with the changes in *x* positions, while at the higher and lower ranges changes are much slower. As mentioned before, the default starting value for all preference weights is $y = 1$, and so the starting *x* position on the weight curve is -2.52. During the preference elicitation stage, the student can specify a preference attribute value as a level: primary, secondary, third, and forth. If a user selects a preference attribute as a primary preference, the x position for that preference is shifted right by 3, which brings the weight to nearly $y \cong 3\ (2.92)$. If it is a secondary preference, the x position for that preference is shifted right by 2, so its weight is around 2 (2.07). During the learning/training stage of preference weights, the weight y is updated with the plan selection and course selection from the selected plans.

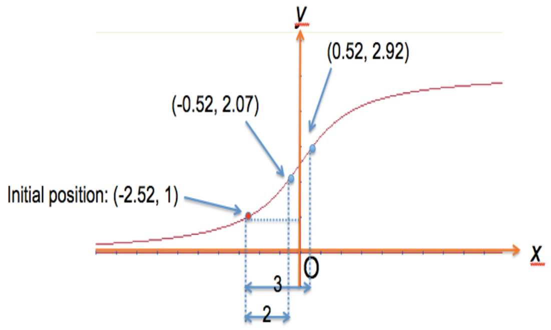
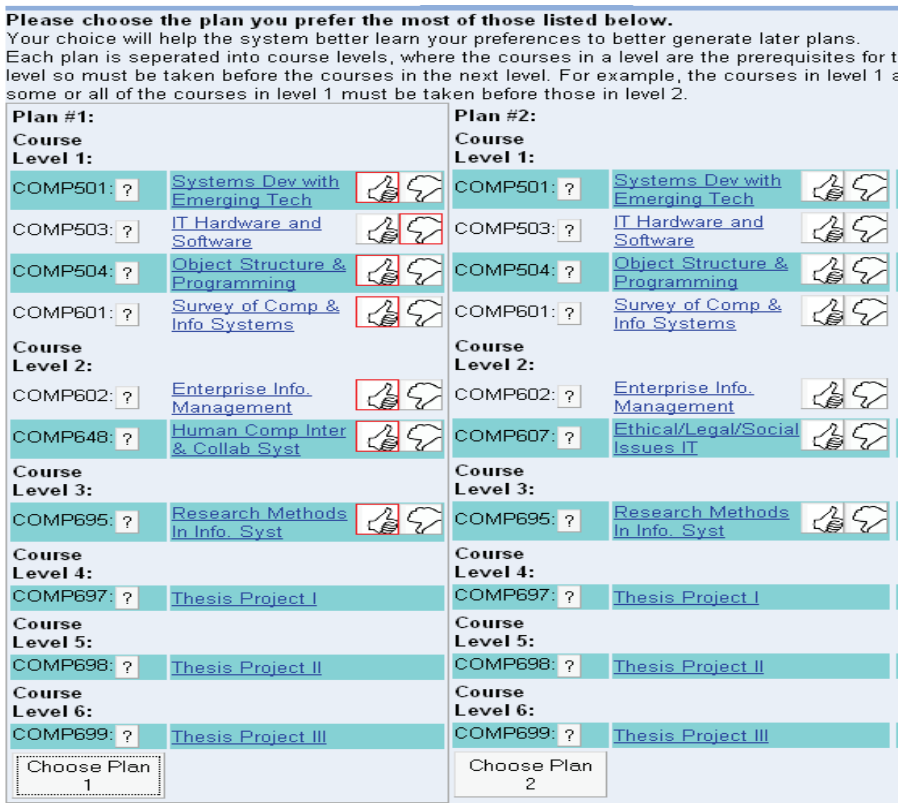


Figure 2: Proposed preference weight learning function

First, during the plan selection, the student is presented with several most preferable plans so that the student can choose a plan s/he prefers most. The interface will send the selection to the student agent who will determine the differences between the selected plan and the other plans. The agent will examine both plans and adjust preference weights depending on courses on the selected plan and the other plans. For a course c in the selected plan of student s, the weight $\mathbf{w}_p(s)$ for preference attribute p will be increased, if course-attribute relation $r(c_i, p_j)$ is non-zero. Similarly, for a course c_i in the not-selected plan of student s, the weight $\mathbf{w}_p(s)$ for preference attribute p_j will be decreased, if course-attribute relation $r(c_i, p_j)$ is non-zero. Second, after the student selected the most preferable (possibly not satisfactory) plan, the SA provides the students a way to approve or disapprove of courses individually (by buttons next to the course with "thumbs up" and "thumbs down" icons). Figure 3 shows a screenshot about this explicit feedback.

The preference weights are increased or decreased with respect to the x values along the weight curve by the following: $0.1 \times r(c_i, p_j)$ for the preference of the given course c_i and preference attribute p_j.



Please choose the plan you prefer the most of those listed below.
Your choice will help the system better learn your preferences to better generate later plans.
Each plan is seperated into course levels, where the courses in a level are the prerequisites for t
level so must be taken before the courses in the next level. For example, the courses in level 1 a
some or all of the courses in level 1 must be taken before those in level 2.

Plan #1:		**Plan #2:**	
Course Level 1:		**Course Level 1:**	
COMP501: ?	Systems Dev with Emerging Tech	COMP501: ?	Systems Dev with Emerging Tech
COMP503: ?	IT Hardware and Software	COMP503: ?	IT Hardware and Software
COMP504: ?	Object Structure & Programming	COMP504: ?	Object Structure & Programming
COMP601: ?	Survey of Comp & Info Systems	COMP601: ?	Survey of Comp & Info Systems
Course Level 2:		**Course Level 2:**	
COMP602: ?	Enterprise Info. Management	COMP602: ?	Enterprise Info. Management
COMP648: ?	Human Comp Inter & Collab Syst	COMP607: ?	Ethical/Legal/Social Issues IT
Course Level 3:		**Course Level 3:**	
COMP695: ?	Research Methods In Info. Syst	COMP695: ?	Research Methods In Info. Syst
Course Level 4:		**Course Level 4:**	
COMP697: ?	Thesis Project I	COMP697: ?	Thesis Project I
Course Level 5:		**Course Level 5:**	
COMP698: ?	Thesis Project II	COMP698: ?	Thesis Project II
Course Level 6:		**Course Level 6:**	
COMP699: ?	Thesis Project III	COMP699: ?	Thesis Project III
Choose Plan 1		Choose Plan 2	

Figure 3: Plan selection and course selection interface

So, if a student *s* selects a plan with course *A* which has a preference, let us say the job objective "CTO," a course-attribute relation value of 5, the *x* position on the weight curve would shift right by 0.5 for the weight of that job objective in the student's preferences.

Similarly, for course *B*, if it has preference for the career track "Consulting" $r(B, "consulting")$ of 4, the *x* position of the weight curve would shift left by 0.4 for the weight of that career track in the student's preferences. This is done in two steps, first finding the *x* position for the old weight w'_p according to the formula,

$$x = 2 \times \tan\left(\frac{3}{5} \times \left(w'_p - 2.5\right)\right) \tag{4}$$

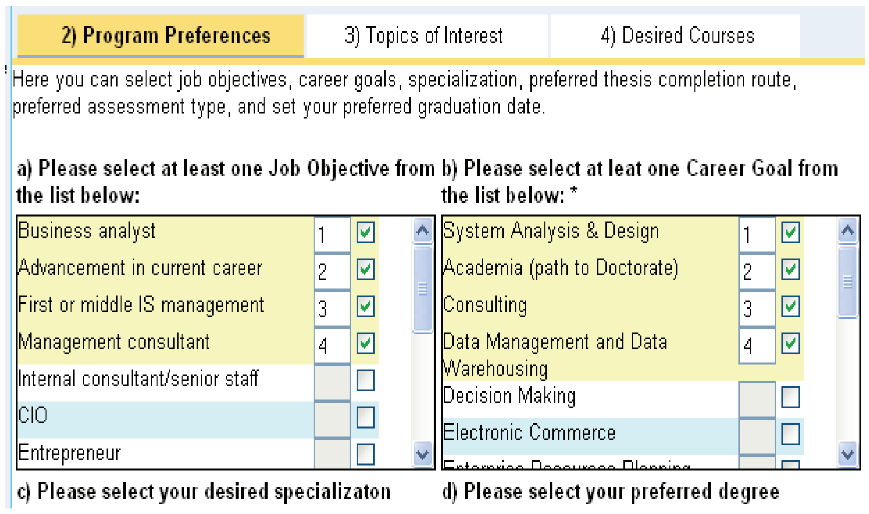
and then, the new weight w''_p is calculated using the formula below.

$$w''_p = \frac{3}{5} \times \tan^{-1}\left(x + \frac{\Delta x}{2}\right) + 2.5 \tag{5}$$

So, if the student has preference for “CTO” a weight of 3, the old *x* position would be determined to be 0.61 through using (4). From that a new weight would be calculated to be 3.34 according to (5). Similarly, if the student’s weight for “Consulting” started at 2, it would decrease to 1.71.

4. PROTOTYPE IMPLEMENTATION

For the evaluation and benchmark of the proposed COD system, a prototype multi-agent system was implemented using JADE (http://jade.tilab.com). To test and evaluate the prototype under real-world conditions, a simulated environment was built, which allows simulating different scenarios by varying several parameters, such as the number of courses in the program, the divergence of preferences for course selection, and the must-offer courses in emergence cases. Figure 4 shows a screen shot of entering students’ course selection preferences.



Here you can select job objectives, career goals, specialization, preferred thesis completion route, preferred assessment type, and set your preferred graduation date.

a) Please select at least one Job Objective from the list below:

Business analyst	1	✓
Advancement in current career	2	✓
First or middle IS management	3	✓
Management consultant	4	✓
Internal consultant/senior staff		☐
CIO		☐
Entrepreneur		☐

b) Please select at leat one Career Goal from the list below: *

System Analysis & Design	1	✓
Academia (path to Doctorate)	2	✓
Consulting	3	✓
Data Management and Data Warehousing	4	✓
Decision Making		☐
Electronic Commerce		☐

c) Please select your desired specializaton

d) Please select your preferred degree

Figure 4: Program planning preferences

Table 1 shows simulated students’ course selection preferences participated in the election process, where among 22 students 9 are freshman students f_1~f_9, 7 are sophomore or junior students s_1~s_6, and 6 are senior students e_1~e_6. There are 24 courses in the program. The simulated course Hits are shown in column H1 of Table 2. Utility for required courses is \$11,500. The candidate courses are denoted by C_1. By considering the curriculum and priority of some courses, the administrator determines a set of courses denoted by C_0 that must be offered in the next term to meet the requirements. The administrator also prepares an exclusion list containing

Table 1: Simulated students' course selection preferences

S	**Course Priority**	**Course Grouping**	**Course sequences**
f_1	$c_{503}\ c_{601}$	$(c_{607}\ c_{610}\ c_{689})$	$\{(c_{503}\ c_{601}), (c_{607}\ c_{610}), (c_{689}\ c_{667})\}$
f_2	$c_{501}\ c_{503}\ c_{691}\ c_{504}$	$(c_{501}\ c_{503}), (c_{601}\ c_{504})$	$\{(c_{501}\ c_{503}), (c_{504}\ c_{691}), (c_{648}\ c_{602})\}, \{(c_{602}\ c_{604}), (c_{648}\ c_{607})\}$
f_3	$c_{501}\ c_{503}\ c_{691}\ c_{601}$	$(c_{501}\ c_{503}), (c_{689}\ c_{667})$	$\{(c_{501}\ c_{503}), (c_{601}\ c_{691}), (c_{602}\ c_{689}\ c_{667})\}$
f_4	$c_{503}\ c_{504}\ c_{601}$	$(c_{503}\ c_{504}\ c_{601})$	$\{(c_{503}\ c_{504}\ c_{601}), (c_{604}\ c_{607}\ c_{648}), (c_{689})\}$
f_5	$c_{601}\ c_{501}\ c_{504}\ c_{691}$	$(c_{501}\ c_{504})$	$\{(c_{501}\ c_{504}),(c_{601}),(c_{691})\}$
f_6	$c_{501}\ c_{503}\ c_{601}\ c_{691}$	$(c_{503}\ c_{501}\ c_{601}\ c_{691})$	$\{\}$
f_7	$c_{501}\ c_{504}\ c_{601}$	$(c_{504}\ c_{501}), (c_{602}\ c_{607})$	$\{\}$
f_8	$c_{501}\ c_{601}\ c_{691}$		$\{(c_{501}), (c_{601}), (c_{602})\}$
f_9	$c_{503}\ c_{504}\ c_{601}$	$(c_{504}\ c_{503}), (c_{607}\ c_{603})$	$\{(c_{601}), (c_{602}), (c_{603}\ c_{607})\}, \{(c_{674}), (c_{695}), (c_{696})\}$
s_1	$c_{602}\ c_{607}\ c_{691}$	$(c_{602}\ c_{691})$	$\{(c_{602}\ c_{607}), (c_{695})\}$
s_2	$c_{601}\ c_{691}$		$\{(c_{601}), (c_{691}\ c_{602}), (c_{689})\}$
s_3	$c_{695}\ c_{691}$		$\{(c_{695}), (c_{617})\}$
s_4	$c_{602}\ c_{691}$		$\{(c_{602}), (c_{691}\ c_{695})\}$
s_5	$c_{610}\ c_{691}$	$(c_{610}\ c_{691})$	$\{\}$
s_6	$c_{667}\ c_{607}\ c_{691}$		$\{(c_{667}\ c_{691}), (c_{607})\}$
s_7	c_{605}	$(c_{504}\ c_{503}), (c_{602}\ c_{604})$	$\{(c_{602}), (c_{603}), (c_{604})\}, \{(c_{605}\ c_{607}), (c_{636}),(c_{667})\}$
e_1	$c_{691}\ c_{695}$		$\{(c_{695}), (c_{696})\}$
e_2	c_{695}		$\{(c_{695}), (c_{674}\ c_{696})\}$
e_3	$c_{695}\ c_{691}\ c_{636}$	$(c_{691}\ c_{695})$	$\{(c_{691}\ c_{695}), (c_{636})\}$
e_4	c_{695}	(c_{695})	$\{\}$
e_5	$c_{695}\ c_{660}\ c_{691}$	(c_{695})	$\{(c_{660}), (c_{695})\}$
e_6	$c_{695}\ c_{691}$	$(c_{691}\ c_{695})$	$\{(c_{691}\ c_{695}), (c_{617}), (c_{696})\}$

courses denoted by C_{-1} that absolutely may not be offered (perhaps because the responsible professor is on sabbatical or the course has a requirement that is only being offered at certain times of a year).

$$C_l = C_0 / C_{-1} \tag{6}$$

There are 88 participating students for the COD of the coming semester. Negotiation in *Round 1* assumes that $C_0 = \{c_{501}, c_{503}, c_{504}, c_{601}, c_{695}\}$; $C_{-1} = \{c_{602}, c_{604}, c_{617}, c_{636}\ c_{637}, c_{682}\}$; and $C_1 = \{c_{501}, c_{503}, c_{504}, c_{601}, c_{602}, c_{605}, c_{607}, c_{610}, c_{648}, c_{660}, c_{667,} c_{689}, c_{691,} c_{695}\}$. For U_{AD}, the actual cost to be paid for the courses offering is calculated with the following formula:

$$U_{AD} = Cost(C) - C_{ideal} = (b \times num_c + r \times \sum_{i=1}^{num_c} h_i) - C_{ideal} \tag{7}$$

where b is the base salary for one course to be paid to the instructor (e.g.,

b=\$5,000); r is the amount of money to gain $course\ fee - payment\ to\ the\ instructor$ for one course registration (e.g. r = \$500); h_i is the number of the registrations of course c_i (i = 1, 2, …, num_c), and C_{ideal} = \$75,000.

In negotiation round R2, c_{691} is chosen with a voting result of 60.77 getting Course Hits as shown in H2 in Table 2. For example, 64 people are going to take c_{691} since it is the first course chosen, and many students have it as their first pick due to it being on many plans and not having many prerequisites. Utility for the current course offering is \$15000.0. Comparing C_{ideal} (\$75,000), the administrator is still not satisfied with the required courses. Students are also unsatisfied with course offering as students' weight is 10.31.

Table 2: Course hits and votes in the simulated environment

	R_1	R_2		R_3		R_4		R_5	
	H_1	V_2	H_2	V_3	H_3	V_4	H_4	V_5	H_5
c_{501}	**23**	0.2	12		9		9		9
c_{503}	**17**	0.2	14		13		13		13
c_{504}	**11**	0.2	9		8		8		8
c_{601}	**20**	0.2	15		13		11		11
c_{695}	**6**	0.5	6		6		5		5
c_{607}		0.2		0.2		0.2		1.2	
c_{605}		5.0		**12**	10		10		10
c_{602}		1.6		1.6		1.6		1.6	
c_{610}		0.7		0.7		0.7		1.7	
c_{648}		2.0		2.0		**2.0**	4		3
c_{691}		**60**	64	0.5	64		64		64
c_{660}		1.0		1.0		1.0		1.0	
c_{689}		1.0		1.0		1.0		**2.0**	11
c_{667}		1.7		2.0		**2.0**	12		12

Now we select ΔU_{SR} = *3*. In negotiation round R3, the required courses are set as six: c_{501}, c_{503}, c_{504}, c_{601}, c_{695}, and c_{691}, votes see column "V_3" in Table 2. c_{605} is chosen in this round with a result of 12.0 by getting Course Hits shown in Column H3. Utility for current course offering is U_{AD} = \$21500.0. Comparing C_{ideal} (\$75, 000), the AD agent is not satisfied with the required courses. SA is also unsatisfied with course offering as weight is 7.34.

In negotiation round R4, the required 7 courses: c_{501}, c_{503}, c_{504}, c_{601}, c_{695}, c_{691}, and c_{605}. Fractional votes are shown in "V4" in Table 2. Courses c_{648} and c_{667} are chosen in this round, with a fractional result of 2.0. Course hits are shown in column H4 of Table 2. U_{AD} for current course offering is \$38000. Comparing C_{ideal} (\$75,000), AD agent is not satisfied with the course-offering result. SA is also unsatisfied with course offering as weight is 4.2.

In negotiation round R5, the required 9 courses: c_{501}, c_{503}, c_{504}, c_{601}, c_{695}, c_{691}, c_{605}, c_{648}, and c_{667}. The fractional votes for this round are shown in V5 of Table 2. c_{689} is chosen in this round with a result of 2.0. U_{AD} = \$48000.0. So, AD agent is still not satisfied with the course offering. SA satisfied with the course offering as the SA weight is 0.86. Negotiation concluded with a course offering list: $C_1 = \{c_{501}, c_{503}, c_{504}, c_{601}, c_{695}, c_{691}, c_{605}, c_{648}$, and $c_{667}\}$. It can be seen from this example that $H_1 = 77$, $H_2 = 110$, $H_3 = 123$, $H_4 = 136$, $H_5 = 145$. As the num_c increases, the total course registrations increase accordingly. Here, we do not consider the size limit of a class. The negotiation and voting processes are described in details in (Lin & Chen, 2013).

5. CONCLUSIONS

In this paper, we presented the modeling and implementation details of student agent within a multi-agent based framework (Lin & Chen, 2013) for course-offering determination. It includes modeling of agent goals and behaviors and the protocols of agent-based coordination in dynamic decision making of individual students and the group decision-making of program administrators. One of the contributions of this work is the identification of a novel problem domain of determining a group of courses offering in educational institutions. The work can stimulate discussion of alternative points of view for how to solve the problem and lead to further discoveries. The second contribution of this work is the proposed architecture of multi-agent system that includes the algorithm to incorporate reasoning capabilities in the student agent, preference elicitation, and inference algorithm. Finally, there is a significant value to this mechanism design, where the administrative agent gets the ability to recommend suitable courses offering to departments in academic terms coordinating multiple student preferences, the course budget of the department, and the derived cost of the courses offered. Each student's preferences are translated into fractional votes that inform a negotiation process bounded by academic and resource constraints. The future work will focus on testing and deploying the system to turn it into a practical application. We will also explore the multi-winner election problem with exogenous constraints in other application domains.

REFERENCES

Brams, S., & Fishburn, P. (2002). *Voting procedure.* Handbook of Social Chice and Welfare .

Bruns, A. (2006). *Towards produsage: futures for user-led content production.* Murdoch University.

Burke. (2002). Hybrid recommender systems: survey and experiments. *User Modeling and Adapted Interaction, 12*(4), 331-370.

Conitze, V. (2010). Making decisions based on the preferences of multiple agents. *Communications of the ACM , 53*(3), 84-94.

Endriss, U. (2006). Monotonic concession protocols for multilateral negotiation. *International Joint Conference on Autonomous Agents and Multiagent Systems* . NY, USA.

Farzan, R., & Brusilovsky, R. (2006). Social navigation support in a course recommendation system. *International Conference on Adaptive Hypermedia and Adaptive Web-based Systems.* Dublin, Ireland.

Garcia, E., Romera, C. V., & Castro, C. (2009). An architecture for making recommendations to courseware authors using association rule mining and collaborative filtering. *User Modeling and User-Adapted Interaction, 19*, 99-132.

Grupe, F. (2002). Student advisement: applying a web-based expert system to the selection of an academic major. *Academic Journal, 36*(4), 573.

Hamdi, M. (2006). MASACAD: a multiagent-based approach to information customization. *IEEE Intelligent Systems, 21*(1), 60-67.

Kannan, S., & Kasmuri, E. (2005). Empowering education managers in schools via a multi-agent system. *Malaysian Online Journal of Instructional Technology*, 17-24.

Kirn, S., Herzog, O., Lockemann, O., & Spaniol, O. (2006). *Multiagent engineering: theory and applications in enterprises.* Springer.

Krulwich, B. (1997). Lifestyle finder: intelligent user profile using large-scale demographic data. *MI Magazine, 18*(2), 37-45.

Lin, F., & Chen, W. (2013). Designing a multiagent system for course-offering determination. *Principles and Practice of Multi-Agent Systems.*

Lin, F., Newcomp, A., & Armstrong, A. (2012). A MAS approach to course offering determination. *IEEE International Conference on Web Intelligence and Intelligent Agent Technology.* Macau, China.

Linden, G., Hanks, S., & Lesh, N. (1997). Interactive assessment of user preference models: the automated travel assistant. *In the Proceedings of User Modeling.* Alberta, Canada.

Opera, M. (2007). MAS_UP_UCT: a multi-agent system for university course timetable scheduling. *International Journals of Computers, Communications and Control, II*(1), 94-102.

Resnick, P., & Varian, H. (1997). Recommender systems. *Communications of the ACM, 40*(3), 56-58.

Schafer, J. B., Frankowski, D., Herlocker, J., & Sen, S. (2007). Schafer, J. B., D. Frankowski, J. Herlocker, & S. Sen,. *Mehtods and Strategies of Web Personalization in the Adaptive Web*, (pp. 291-324).

Shen, W. (2002). Distributed manufacturing scheduling using intelligent agents. *IEEE Intelligent Systems, 17*(1), 88-94.

Smith, R. G. (1980). The contract-net protocol: high-level communication and control in a distributed problem solver. *IEEE Transactions on Computers, C-29*(12), 1104-1113.

Smyth, B., & Cotter, P. (2001). Personalized electronic program guides for digital TV. *AI Magazine, 22*(2), 89-98.

Smyth, B., Shang, Y., Shi, H., Chen, & Shing, S. (2001). An intelligent distributed environment for active learning. *ACM Journal of Educational Resources in Computing, 22*(2), 89-98.

Tang, T., & McCalla, G. (2005). Smart recommendation for an evolving e-learning system: architecture and experiment. *International Journal of E-Learning, 4*(1), 105-129.

Vainio, A., & Salmenjoki, K. (2005). Improving study planning with an agent-based system. *Informatica, 29*, 453-459.

Weiss, G. (1999). *Multiagent systems, a modern approach to distributed artificial intelligence.* MA, USA: MIT Press.

Winikoff, M., & Padgham, L. (2013). *Agent oriented software engineering.* MA, USA: MIT Press.

Wooldridge , M., Jennings, N., & Kinny, D. (2000). The Gaia Methodology for Agent-Oriented Analysis and Design. *Journal of Autonomous Agents and Multi-Agent Systems, 3*, 385-312.

ACKNOWLEDGEMENTS

We thank Natural Sciences and Engineering Research Council (NSERC), Canada, and Research Incentive Grant (RIG), Athabasca University, Canada for their support to this research project.