ATHABASCA UNIVERSITY

Course-Offering Determination with Combinatorial Auctions

By

ANTON VASSILIEV

An essay submitted in partial fulfillment

Of the requirements for the degree of

MASTER OF SCIENCE in INFORMATION SYSTEMS

Athabasca, Alberta

October, 2015

@ Anton Vassiliev, 2015
Course Offering Determination (COD) is a strategy for a program in an institution that maximizes the satisfaction of the students and maximizes the enrollment of the courses within budget and other resource constraints through intelligent and efficient course offering. COD is a dynamic, distributed resource allocation problem that offers significant challenges of continually changing and evolving environment due to changes in students and administrator’s preferences and resource constraints. An efficient solution to the COD problem can be achieved by gaining ability to improve with experience and learning from feedback and continues long standing relationship between the user and the COD system. In view of this, we approach the problem of COD from the Multiagent perspective and model it through the use of multiple collaborating agents. In our approach, the agents negotiate using a protocol based on a multi-round and multi-unit Combinatorial Auctions (CA). One of the main challenges in CA is to solve the Winner Determination Problem (WDP), which ensures the rational and fair representation of students, administrators, and all parties involved in the COD process. To solve WDP, we used a modified and extended version of Branch On Bids (BOB) algorithm, which accounts for multi-unit nature of courses as well as other constraints from students and administrators. The COD solution offered in this paper provides an approach that is both effective and efficient.
ACKNOLEDGEMENTS

I thank my research supervisor and mentor Prof. Fuhua (Oscar) Lin for his continuous support and for igniting my interest in Multiagent Systems.
CONTENTS

LIST OF TABLES .............................................................................................................. 7

LIST OF FIGURES ........................................................................................................... 8

CHAPTER I INTRODUCTION .............................................................................................. 9

CHAPTER II LITERATURE REVIEW .................................................................................. 12

  Multiagent Systems ........................................................................................................ 12

  Artificial Intelligence and Multiagent Systems in education ....................................... 14

  Combinatorial Auctions ................................................................................................ 18

  Case Study: Combinatorial Auctions for Transportation Services ................................ 20

    Context ........................................................................................................................ 21

    The Auction Design .................................................................................................... 21

    Results .......................................................................................................................... 22

    Relevance to Course Offering Determination .......................................................... 23

CHAPTER III METHODOLOGY ......................................................................................... 25

  Problem Statement and Motivation ............................................................................... 25

  Gaia Methodology for Agent Oriented Design .............................................................. 26

    Goals .......................................................................................................................... 30

    Agents ....................................................................................................................... 32

    Protocol Flow ............................................................................................................ 33

    Branch On Bids Algorithm ....................................................................................... 36
CHAPTER IV. Architecture ........................................................................................................40

Analysis .................................................................................................................................40

The Student Agent ..................................................................................................................40

The Auctioneer ......................................................................................................................42

The Program Administrator ...................................................................................................45

The interactions .......................................................................................................................47

Design .................................................................................................................................48

Bidding Language .................................................................................................................48

Combinatorial Auctions, Winner Determination Problem and Branch on Bids in Course Offering Determination context ..................................................................................52

Solution Quality ....................................................................................................................58

Performance ..........................................................................................................................59

Example ................................................................................................................................61

CHAPTER V. CONCLUSIONS AND RECOMMENDATIONS ..............................................65

Conclusions ..........................................................................................................................65

Future Research ......................................................................................................................66

Appendix A. Acronyms ...........................................................................................................67

REFERENCES .........................................................................................................................68
LIST OF TABLES

Table I Matching Gaia requirements to COD.................................................................27
Table II Branch On Bids variables and terms..............................................................38
Table III Branch On Bids Pseudo code........................................................................39
Table IV Student Agent Analysis..................................................................................42
Table V Auctioneer Agent Analysis...............................................................................45
Table VI Administrator Agent Analysis.........................................................................47
Table VII Course Offering Determination Interaction Model ..................................48
Table VIII Course Offering Determination Submission Variables .........................49
Table IX Course Offering Determination bid variables.................................................50
Table X Course Offering Determination bid. Example 1 ..........................................51
Table XI Course Offering Determination bid. Example 2 ..........................................51
Table XII Course Offering Determination CA. Variable definition............................53
Table XIII Branch on Bids for Course Offering Determination variables..................55
Table XIV Branch on Bids for Course Offering Determination pseudo code ...............58
Table XV. Branch On Bids Performance Variables......................................................60
Table XVI Branch on Bids Example Definition............................................................62
Table XVII Partial flow of Branch on Bids for Course Offering Determination example.................................................................................................................64
LIST OF FIGURES

Figure I. Graphical outline of Administrator, Student and Auctioneer agents in a COD process.................................................................12
Figure II Relationships between Gaia models..................................................28
Figure III Gaia Analysis Concepts................................................................29
Figure IV. Course Offering Determination goal model .....................................31
Figure V. Course Offering Determination Agents.............................................33
Figure VI. Negotiations Protocol for Course Offering Determination based on Combinatorial Auctions .................................................................35
Figure VII Course Offering Determination Interaction Model .......................47
Figure VIII Branch on Bids for Course Offering Determination example........64
CHAPTER I INTRODUCTION

Every semester or every year administrators of academic programs in colleges and universities work hard to determine the courses that their institution will be offering for the upcoming term or year. This is a challenging task that requires an intelligent strategy to maximize the satisfaction of the students and to maximize the enrollment, while keeping to budget and coping with other resource constraints. In the context of this paper we refer to the task of determining the selection of courses as Course Offering Determination (COD). The main question COD attempts to answer is: What course offering determination strategy for a program in an institution maximizes the satisfaction of the students and maximizes the enrollment of the courses within budget and other resource constraints [1]?

The importance of COD cannot be overestimated. For students, the courses offered drive their ability to graduate, the timelines for achieving selected majors, the prospects regarding the continuation of academic career and success in the job market. Moreover, students may have personal course preferences and the ability to take these courses affects their satisfaction with the program and the utility of the program for them.

For institutions and program administrators, student’s happiness with the offered courses leads to higher enrollment. Career and academic achievements upon
graduation of the students shape the school’s reputation. The courses offered also affect the school’s budget and consume other limited resources such as the availability of instructors, lecture halls, labs etc.

Today in practice, to determine the selection of courses, a program administrator may use past history of offered courses, current commitments and resource availability. Administrator’s best judgment and experience are also among the deciding factors. This process does not directly take into account student preferences and lacks precision in budget and resource allocation. This is why an improved COD system is necessary for the educational institutions.

A successful COD system must be able to address multiple challenges [1]:

• The continually changing and evolving environment (for example students may drop out or change preferences, instructors may leave, budgets may be cut or increased), affecting the optimal solution
• Gaining ability to improve with experience, learning from feedback and continues long standing relationship between the user and the COD system
• The time consuming nature of COD
• Rational and fair representation of students, program administrators and other parties involved in the COD process

In this paper, we propose and design a new COD solution based on Multiagent Systems (MAS) [2]. Representing all constituents of the COD problem with independent autonomous agents we use their unique preferences to solve the
COD problem in an optimal and efficient manner [3; 4]. A MAS-based solution offers continuous negotiation and the ability to apply machine learning techniques [5], addressing the challenges listed above.

The multiagent COD solution will be approached using the Gaia methodology for agent-oriented analysis and design [6] and agent-oriented software engineering [7]. Students and administrators will be assigned autonomous agents. We model the COD problem as a multiagent dynamic constraint resource allocation problem. We will seek for the optimal COD solution via a negotiation protocol based on Combinatorial Auctions (CA) [4].

The application of CA to COD aims to improve process transparency, lower transaction costs and introduce a robust method to account for the complementary nature of values for offered courses. This should result in increased long-term utility for students and in a more efficient application of school resources.

The proposed architecture uses the Auctioneer Agent (AA) and a protocol based on CA to enable a negotiation between multiple Student Agents (SA) and the Administrator Agent (AD). The result of negotiations is the optimal course offering that maximizes student's utility and satisfies administrator's resource constraints. Figure I shows a graphical outline of a COD process, where AA, SA and AD agents are involved in negotiation for the optimal course offering using CA.
This essay is organized as follows. Chapter II contains a brief survey of relevant research works. Chapter III covers the methodology and motivation for this research. Chapter IV provides the Gaia model as well as the detailed architecture of agents and the particulars of the CA-based negotiation protocol. Chapter V concludes with the findings and a brief discussion of future works.

CHAPTER II LITERATURE REVIEW

MULTIAGENT SYSTEMS
In this paper we are going to operate with the concept of an agent and with the concept of a multiagent system. According to [37], an agent is a computer program (or a group of computer programs) that is situated in some environment and that is capable of autonomous action in the environment to meet its objectives [37].
An agent that is reactive, proactive and capable of interaction with other agents (and possibly humans) within the hosting environment, is called an intelligent agent [37]. An intelligent agent acts rationally within its environment seeking to optimize the outcome of its efforts [39].

The agent-hosting environments may be described with the help of the following properties [37][38]:

- Accessibility. The extent to which information about the environment is obtainable and accurate.
- Predictability. The extent to which the outcome of an agent auction can be predicted based on the current state.
- Continuity. The extent to which the environment can be described with a finite set of states, auctions and precepts.
- Control. The extent to which an agent may change the environment.
- Dynamics. The extent to which the environment can change without agent’s involvement.

Accessible, predictable, discrete, controllable and static environments tend to be the simplest. A good example of such environment may be an electronic checkers game [37]. On the other end of the spectrum are inaccessible, unpredictable, continues, uncontrollable and dynamic environments which tend to be the most complex. A good example of such environments is a highway for a driverless car.

With the help of the defined properties we can describe the COD environment as:
• Accessible. The historical bids and constraints are well-defined and known.
• Unpredictable. It is impossible to predict the bids of all students and the changing constraints.
• Discrete. The COD environment is described with a set of well-defined variables.
• Semi-controllable. The SA, AA and AD agents can influence the COD environment but cannot control it fully.
• Dynamic. Student’s bids and constraints change dynamically.

A Multiagent System (MAS) is a network of interacting, collaborating, intelligent agents that act within a distributed environment. Various research papers indicate that MAS solutions may be very effective at solving problems that are distributed in their nature, offering MAS as an ideal approach to COD with its highly distributed character [38].

MAS solutions are implemented with a dependency on multiagent platforms, which offer some sophisticated tools [40]. The previous COD research [1; 18] had utilized the JADE [31] and JASON [41] platforms for COD execution and testing.

MAS serves as a pivotal role in our approach. Multiple types of intelligent agents are introduced to fulfill various roles presented in the proposed MAS. The COD environment hosts these agents in a multiagent fashion, working to achieve the goal to have fair and optimized course offering.

ARTIFICIAL INTELLIGENCE AND MULTIAGENT SYSTEMS IN EDUCATION
There is a significant amount of research related to Artificial Intelligence and Multiagent Systems in education.

AutoTutor is a system that addresses pedagogical and tutoring tasks through emulation of a dialog with a human tutor [9]. The system displays an animated model of a human that interacts with the student through a series of presentations. Presentations are followed by open-ended questions that encourage lengthy and detailed answers and discourage superficial, episodic knowledge. AutoTutor was first implemented to teach the fundamentals of computer literacy and participating students showed improvements of almost a full letter-grade.

SQL-Tutor is another system that automates the tutoring task [10]. Instead of using sequential questions/dials like the AutoTutor, it personalizes tutoring with the use of the student’s progress model. The model is represented through a list of logical propositions that embody a set of constraints, passed or violated by the student. The system performed admirably when tested with a group of students learning the SQL language. The students reported a smooth learning curve and achieved better exam results than their peers who did not use the SQL-Tutor system. A notable problem of SQL-Tutor is that the constraints utilized in progress modeling are difficult to determine and may discourage from the use of the system.

The task of school selection was addressed by the SmartChoice program [11]. It specializes in providing personalized recommendations on public school
selection. SmartChoice models student’s achievements with the fixed-effects statistical model and uses it to predict student’s performance in a range of available schools.

A system called e-Advisor was proposed and developed by Athabasca University, which aims to help students to address the complex task of individualized study planning [12]. e-Advisor utilizes the MAS approach to implement multiple types of agents representing and acting for Students, Advisors, Instructors, and Administrators. The system relies on ontologies to represent knowledge of the environment.

Other research papers cover Multiagent course scheduling [13], a collaborative learning environment [14], a student advising application [15], a time-table scheduling system for educational institutions [16] and a collaborative personal study planning system [17].

Recently there has also been some research done on MAS-based solutions for COD that address the issue of balancing student preferences with school’s limited resources.

The web-based Intelligent System for Educational Program Planning and Scheduling (WisePPS) [18] is a COD solution design and implementation that combines distributed decision-making and one-to-one agent negotiation. It relies on the Student Agent, the Student Representative Agent, the Administrative Agent, the Instructor Agent and the Instructor Representative Agent to represent crucial roles in the system. Student preferences are modeled using three distinct
methods: precedence, grouping and progressions. The Single Transferable Vote (STV) protocol [19] is used to aggregate student preferences and the COD negotiation protocol is modeled via Petri Nets [20]. WisePPS was implemented and tested on the JADE platform and delivered an improvement over conventional course offering determination methods.

The second research paper [1] also relies on STV for aggregating student preferences and uses the precedence, grouping and progression methods to represent student’s course selections. The COD system and the negotiation protocol are modeled via Contract-Net Protocol (CNP) [21] and Monotonic Concession Protocol (MCP) [22]. The roles are fulfilled by the Administrator Agent, the Student Agent and the Student Representative Agent. This COD solution provides detailed agent design using the Gaia role model methodology and generates test results for JADE and Jason based implementations.

The WisePPS and the CNP-based COD both offer improvements to the COD problem and come up with course schedules that efficiently utilize available resources and satisfy student’s requirements. However the complexity of the voting and the course selection mechanisms may be seen as a deterrent to student participation.

The COD system proposed in this paper looks for inspiration in the existing COD research, and expands upon it with the use of CAs, which introduce bids on course bundles as well as offer simplicity and transparency to the end users, while retaining all the efficiency associated with the other approaches.
**COMBINATORIAL AUCTIONS**

In this paper, we rely on the existing MAS-based COD research and expand upon it using the CA [4; 24] to model the COD problem and the associated negotiations protocol.

CAs extend regular-style auctions by allowing bidders to bid on bundles of goods, instead of limiting to the bids on single items. This approach introduces a wide functional flexibility and CA’s offer a number of advantages over conventional auction design [4]:

- Bidding on bundles allows accounting for the complimentary values of the items composing the bundles. This results in increased economic efficiency and higher utility for buyers and sellers
- Expressive format of CAs allows for time saving and an efficient negotiation process for dealing with complex goods, thus lowering transaction costs
- Bid acceptance rules allow for high level of transparency and ensure fairness

CAs have been used in many real life, large scale auctions with great success. To name a few, Sears used CAs to design the auction of eight hundred and fifty four delivery routes and reducing its logistics costs by thirteen percent [25], the FCC used CAs for selling spectrum licenses for Wireless Services [26; 29], London Transport used CAs to distribute city bus routes between bidding contractor companies [24], and for allocating airport time slots [27].
An imperative topic for CAs is that of the bidding language. A bidding language assigns semantic meaning to syntactic constructs that are used for definition and representation of combinatorial bids. Naturally, the way we represent bids has a great influence on the CA protocol design and we must approach it with caution, keeping the specifics and the goals of our COD application in mind. Moreover, the cardinality of the bidding space in CAs is quite large (\(2^m\)), where m is the number of auctioned items, so a succinct language that does not require excessive bidding is of the great importance.

Logical languages and generalized logical languages [34] provide very powerful syntax with flexible semantics. They offer the ability to express complex preferences to a considerable depth with ease. Bidding languages for mixed multi-unit CAs expand that ability to include bids with quantity ranges [33]. While this expressive power injects the possibility for a wide range of bidding combinations, they introduce increased complexity to Winning Determination Problem algorithms. In this paper, we rely on a simpler version of a bidding language in order to streamline the initial design of the CA-based COD protocol, with little or no loss to COD functionality.

To determine the most efficient assignment of the bundles to the winning bidders, the CA auctioneer has to solve the Winner Determination Problem (WDP). The main challenge of WDP is that it is an intractable, NP-complete problem and it presents significant difficulties for time-efficient and space-efficient solutions.
There exist a large number of algorithms that offer both exact and approximate solutions for WDP [4]. The most obvious solution is the explicit, full enumeration of all possible combinations, however this approach is impractical and the computational effort quickly grows intractable.

The more sophisticated solutions include the Integer Programming algorithm which searches the extreme of the objective function that represents the CA [42], the Dynamic Programming algorithm which splits the CA into smaller problems using the bottom-up principle [43] and the Branch and Bound algorithm that minimizes the search spaces by pruning branches that would not provide a satisfactory solution [4].

The Branch on Bids algorithm stands out from the others because it offers an efficient method to model multi-unit auctions [23; 28] and a way to utilize problem-specific heuristics to prune search branches and improve efficiency. For the purpose of this research, the multi-unit auctions and problem-specific heuristics serve as compelling reasons for selecting the Branch and Bound algorithm as the way to solve WDP.

**CASE STUDY: COMBINATORIAL AUCTIONS FOR TRANSPORTATION SERVICES**

In this case study we review an example of a successful application of CA to Transportation Services, which is a complex, large scale and real-life problem. Our solution of CA for COD is largely based on the design used in this case study.

---

1 The case study is based on [25]
owing to its proven success and the similarity of the business requirements between the two problems.

**CONTEXT**
Sears Logistics Services (SLS) is a subsidiary of Sears Inc. and handles the logistics distribution chain connecting production locations with Sears retail stores and distribution hubs. SLS does not employ its own, private fleet of trucks but instead contracts out to trucking and carrier vendors. This is a very large-scale operation as the Sears logistics chain is one of the biggest in the world.

Prior to the integration of CAs, the vendor procurement process was handled by the SLS field agents who had established relationships with suppliers and continuously worked on maintaining the contracts. In order to cut costs, improve timelines, efficiency and service, the SLS decided to consolidate the vendor procurement process. Besides the stated improvements the new process had to offer a fair, transparent and easy to understand negotiations protocol in order to keep existing successful vendors.

A conventional auction could not suffice for the requirements since it necessitated breaking up existing supply chains and profitable routes into small auctionable chunks. This approach is discouraging to the bidding vendors because they run the risk of losing pieces of the business (routes) that make their offers profitable.

**THE AUCTION DESIGN**
To address the requirements and problems listed above, the CA design was proposed\(^2\). The bidders were allowed to place bids on any combination of SLS auctioned routes, thus allowing coordination of placed bids with their existing infrastructure and other business commitments. The proposed design is an iterative CA that avoids sealed-bid auctions in favor of progressive auctions to simplify the bidding and improve transparency.

In progressive auctions, at the beginning of each CA iteration all bidders are allowed to submit any number of bids on any combination of routes. Each bid constitutes a contract proposal, and if accepted by SLS the bidding vendor must honor it. The winning bids are recorded as provisional winners and communicated to the bidding parties. The next iteration starts with bids submitted against provisional winners and carries on through the normal bidding process.

The CA iterations are terminated (the stop rule) when the cost improvement resulting from the current iteration is less than predetermined value.

**RESULTS**

The results of the first (proof-of-concept) execution of the auction were overwhelmingly positive. The combinatorial feature of the auction was used by fifty percent of the bidders. Thirty percent of the bidders using the combinatorial feature successfully won the auction. The solution to the WDP was found in fifteen to thirty minutes in all of five auction iterations. Participants were generally satisfied with the CA process and expressed interest in future auctions.

---

\(^2\) [25] Refers to “Combinatorial Auctions” as “Combined Value Auctions”. We are using the term “Combinatorial Auctions” to remain consistent with the rest of our paper.
Some complaints were reported for the prolonged time of execution of the CA from start to finish. The auction was completed in five iterations, requiring one month between iterations, thus spanning five months.

Throughout all five rounds of the auction, the bid counts remained in the thousands, with acquisition costs gradually declining from $187149 to $165371. Round five had a 1.9% decrease of the acquisition costs, which was below the threshold, thus terminating the auction.

The subsequent large-scale application of the designed auction to procurement of vendors for five hundred and thirty six supply lanes resulted in thirteen percent in savings and a total saving of thirteen million dollars.

The successful application of CA principles and the similarity of the problem of supply lane procurement auctions to that of the COD, serves to us as the motivation for adopting a similar auction design to solve our problem.

**Relevance to Course Offering Determination**

In this paper, we model COD using MAS and CA principles, where the courses are the commodities being auctioned, and the students form bundles of courses on which they bid in the following manner:

- The students are represented by SA agents that will act as buyers in CA.
- The school administrator is represented by the AD agent that will act as the seller in CA.
- An AA agent will be used for the collection of bids and the calculation of WDP; essentially it will act as a CA auctioneer.
• The courses are the commodities being auctioned and they form the bundles for which the students place their bids.

A single course is a multi-unit commodity, with the number of units equal to the maximum student enrollment capacity. Therefore to model COD with CA we will need to consider multi-unit CAs where each auctioned commodity is associated with the number of available units. Multi-unit CAs can be solved with generalized algorithms such as Branch On Bids (BOB) [23; 28] via the extension of the counters on the items used in selected bids. BOB is a variation of Branch and Bound [4] that utilizes a branch-search tree instead of an item search tree.

To ensure efficient negotiation, the COD protocol will be modeled using multi-round CAs [25; 29], by repeating the single round of CA multiple times until the time allotted for the auctions is expired or if no more bids were placed. The below protocol follows the same strategy outlined in the previous sections (COMBINATORIAL AUCTIONS FOR TRANSPORTATION SERVICES case study), and builds upon it. A single auction round will consist of the following steps (the rounds repeat until the auction expires or no more bids are placed)

• Students select preferred courses and submit their bids on course bundles (multiple bundles and bids are allowed per student)
• The AA agent collects the bids and solves WDP, taking into account AD agent preferences and resource constraints
• Winning bid allocation (the provisional winning bids) is made available to students
• Students correct and resubmit their bids with the last winning allocation in mind
• The new bids are combined with the provisional winning bids and the new round is executed
• Auction ends when the time allotted for the auction is expired or there are no more bids

CHAPTER III METHODOLOGY

PROBLEM STATEMENT AND MOTIVATION

At its core, COD is a dynamic constraint satisfaction problem, with students and school administrators, courses and other resources as its subjects and actors. A student’s course selection is governed by the student’s goal of graduation, prospects on the job market, personal preferences, financial and scheduling constraints, course inter-relationships and other factors. School administrators are concerned with availability of school resources (such as lecture halls, labs, equipment, course instructors, etc.), school’s commitments, finances, reputation, students’ satisfaction and enrollment.

The problem is made more difficult by the fluid nature of the constraints. Students may fail, drop or pickup new courses. A student’s personal preferences evolve with experience. A student’s financial situation may improve or decline and job markets often present new skill demands. For school administrators the course
offering requirements are also in a state of constant transformation: Budgets change on a regular basis; Instructors leave and new instructors are hired; School resources break down, get repaired, get added or removed.

Without automation, COD relies on a program administrator’s best judgment and experience to balance the past history of offered courses, current commitments and resource availability. This process does not account for student preferences and lacks precision in budget and resource allocation as well as student satisfaction. Improving the COD process, the economics of course offering and student satisfaction serves as the motivation for this research, with the main goal of finding the optimal course offering that will maximize student satisfaction and school reputation while keeping to the school’s limited resources.

**GAIA METHODOLOGY FOR AGENT ORIENTED DESIGN**

In this paper we rely on the Gaia [6; 30; 31] methodology to analyze and design the COD solution. Gaia offers agent-oriented software engineering methods and it provides a comprehensive approach to analyzing, designing, and building multi agent systems, from abstract high-level concepts to somewhat low-level details.

Blow we prepared a table (Table I [6]) outlining Gaia’s system requirements and matched them to those of COD to show the applicability of Gaia to our problem.

<table>
<thead>
<tr>
<th>Gaia requirements</th>
<th>COD requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agents are coarse-grained</td>
<td>Agents will represent students, school</td>
</tr>
</tbody>
</table>
computational systems | administrators and auctioneers

The goal of the system is to maximize some global quality measure | COD has to maximize students' utility and minimize resource consumption

Agents are heterogeneous | No limitations from COD

Organizational structure of agents is static | In COD agents do not change their role

The abilities of agents and services they provide are static | In COD agents do not change their services

The system contains less than a hundred agent types | There are three agent type in COD

**Table I: Matching Gaia requirements to COD**

Gaia methodology centers on gradually bringing system concepts from abstract to concrete and detailed. When compared to Object Oriented Design (OOD), the abstraction level in Gaia is assumed to be higher, since OOD concentrates on objects that can be directly coded using a programming language. In Gaia the most concrete concepts are agents and their service and acquaintance models, which in turn can be modeled using Object Oriented Design and then implemented.

Abstract Gaia concepts include System, Roles, Permissions, Responsibilities, Protocols, Activities, Liveness and Safety properties. Concrete concepts are Agent Types, Services and Acquaintances (Figure II [6]).
During the analysis phase, Gaia methodology prescribes the development of an abstract model for the problem it is trying to solve. The object of analysis is to build a better understanding of the problem and the concepts remain abstract with no reference to implementation details. Please refer to the diagram below that outlines the analysis of abstract concepts advocated by Gaia, following from more abstract to more concrete (see figure III [6]).
The System is the most abstract concept operated on by Gaia, and it encompasses the complete solution. Roles outline functional positions available in the system, and are roughly equivalent to job titles in business organizations. Interactions represent communications between roles. Responsibilities cover the functions that roles fulfill, with permissions explaining the access levels to restricted resources. The safety concept enforces the "must not happen" conditions and liveness brings about the properties that we strive to achieve.

During Gaia's design phase the problem model is taken to a more concrete level, enough to apply conventional design methodologies (i.e. Object Oriented Design). During this phase Gaia attempts to answer the question of how a society of agents operates in order to solve the stated problem. The design phase results in the generation of three models: the Agent Model, the Services Model and the Acquaintance Model (Figure II), outlining all of the types of agents involved, the
services offered by the agents and their properties, and the communication links between the agents.

Following the Gaia principles outlined in this subsection, the Architecture chapter details the CA-based COD solution. The Architecture chapter begins with the Gaia Analysis Phase by talking about the COD Role Models, their associated Safety and Liveness Properties, Permissions and Interactions. It then continues to the Gaia Design Phase, which provides the low-level details of the COD bidding language, the CA COD algorithm and examples.

GOALS
As illustrated in Figure IV, the goals of students and program administrators are not aligned. The course selection process for students is high-contingency and has a cumulative effect that influences the student’s ability to graduate from the program on schedule, achieve chosen majors and progress with their academic career or on the job market. The ideal course offering for a student would offer maximum course selection and would contain all courses included in the program’s curriculum, which of course would contradict the administrator’s constraints.

The program administrator’s goal is to find a course selection that minimizes the costs while maximizing student enrollment and student satisfaction and is subject to school resource constraints. While it has some common ground with the goals of the students, the dependency on the constraints introduced by the school’s limited resources and the requirement to minimize costs introduces a conflict with the needs of students.
Figure IV: Course Offering Determination goal model³

Please consider the diagram in Figure IV. The student's main goal is efficient graduation while studying desirable courses. The sub-goals include keeping to budget, enabling academic and/or professional career, and taking courses selected due to personal preferences to study with fun and ease.

The administrator's main goal is the determination of course offering schedule. The sub-goals include the calculation of cost and revenue to make a cost effective decision, the aggregation of student preferences and the selection of courses that minimize expenses while maximizing student satisfaction and keeping to budget.

³ Based on the goal-model diagram from [1]
The COD solution introduced in this paper is designed to address the conflicting goals of students and administrators by using CAs. The proposed COD CA-based negotiations protocol runs multiple rounds, continuously reevaluating the most efficient solution under the changing constraints of the COD environment.

AGENTS
The COD solution design proposed in this paper relies on three types of agents. The Student Agent (SA) represents a student and is assigned to each student at the beginning of the program for the duration of the program. The AD agent represents a program administrator and the AA agent represents the CA auctioneer. The SA and AD agents work in the best interest of the human actors they represent and the AA agent coordinates their effort via a CA-based protocol to achieve a mutually agreeable and efficient solution. The COD negotiation is triggered by the AD agent which prepares a set of must offer courses and a set of negotiable courses and forwards them to the AA and SA agents. The SA agent generates bids for the CA protocol and the AA agent manages the auction. The responsibilities of these agents are covered in details in the Architecture chapter. Figure V. below provides a high-level outline of the relationships between the agents.
The CA-based negotiations protocol for COD begins before course registration for a school year. Program administrators notify their respective AD agents to begin building a set of must offer courses, must not offer courses and negotiable courses. The list of must offer courses and must not offer courses depends on previous faculty commitments and does not depend on student preferences.

Program courses that are not included in the must offer and must not offer sets comprise the negotiable courses set. The set of negotiable courses and the set of must offer courses are forwarded to the AA agent. The AA agent forwards these courses to all SA agents, thus starting CA-based negotiations protocol.

The SA agent begins negotiations by working with the student to update the study plan based on the student's preferences and must offer courses and negotiable courses.
courses, thus determining the set available for bidding. The student is allowed to select bundles of courses that he or she will prefer to take over the school year and places bids on these bundles. Bidding is done using course bidding points that are allotted to each student at the beginning of a program and are never refilled. A winning bid will withdraw the spent points from the student’s account, thus emulating the use of money in conventional auctions. Each student is allowed to place multiple bids for each round, and once the set of bids is determined, it is forwarded to the AA agent.

The AA agent aggregates the student bids and uses BOB algorithm to solve the WDP. The BOB algorithm is modified to account for multi-unit nature of courses and for other constraints from SA and AD agents.

For each selection of a bid, the winning set of bids is verified with AD agent to ensure that the winning solution does not break AD’s constraints. The BOB algorithm is executed until completion or until AD’s constraints are broken, at which point the winning bids are forwarded to SA agents and the next cycle of the protocol begins.

The protocol terminates when the time allotted for the auction is expired or if no more bids were placed. The last selected set of winning bids, combined with must offer courses, form the course offering for the year.

Figure VI below provides a swim lane diagram to visualize the flow.
Figure VI. Negotiations Protocol for Course Offering Determination based on Combinatorial Auctions
**Branch on Bids Algorithm**

The core of COD solution proposed in this paper relies on the BOB algorithm to find the best bids. This subsection formalizes of CAs, WDP and the BOB algorithm [23; 28]. The architecture chapter will build on the BOB algorithm presented here and will adopt it to solving the COD problem.

**Definitions**

A CA is defined by

1. The set of items M available for bidding
   
   \[ M = \{ i_1, i_2, \ldots, i_m \}, \quad |M| = m \]

2. The set of bids B generated by the bidders
   
   \[ B = \{ b_1, b_2, \ldots, b_n \}, \quad |B| = n \]

   With each bid \( b_i \) defined as a two-tuple
   
   \[ b_i = \langle S_i, p_i \rangle \]

   Where \( S_i \) is the set of bidding items and \( p_i \) is the bidding price such that
   
   \[ S_i \subseteq M, \]
   
   \[ p_i \geq 0 \]

To identify the winner in CA we must solve the WDP, which is defined as follows:

\[
\text{max} \sum_{j=1}^{m} p_j x_j
\]

\[
\text{s.t.} \quad \sum_{j \in S_i} x_j \leq 1, \quad i = 1, 2, \ldots, m
\]

\[ x_j \in \{0, 1\} \]

where \( x_j = 1 \) signifies a winning bid \( b_j \) and \( x_j = 0 \) signifies a losing bid \( b_j \)
This means that we have to maximize the sum of winning bids, while ensuring that the sets of items in winning bids do not intersect. The last condition is imposed by the assumption that each offered item is unique. The unique item requirement will have to be relaxed for solving COD.

**Algorithm**
Prior to introducing the details of BOB algorithm and the pseudo code listing, we are going to define the variables and terms (Table II Branch On Bids variables and terms).

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$IN$</td>
<td>Winning set of bids in the current search path</td>
</tr>
<tr>
<td>$g$</td>
<td>The revenue from the bids in $IN$</td>
</tr>
<tr>
<td>$IN^*$</td>
<td>The set of winning bids overall</td>
</tr>
<tr>
<td>$f^*$</td>
<td>The revenue from the bids in $IN^*$</td>
</tr>
<tr>
<td>$e_j$</td>
<td>The exclusion count for bid $b_j$</td>
</tr>
<tr>
<td>$M^*$</td>
<td>The set of unallocated item in the current search path</td>
</tr>
<tr>
<td>$h$</td>
<td>The upper bound on the revenue from the unallocated items ($M^*$)</td>
</tr>
<tr>
<td>$c(i)$</td>
<td>The admissible heuristic for estimating revenue for item $i \in M^*$</td>
</tr>
<tr>
<td>$c(i) = \max_{j \in S_i, e_j = 0} p_j /</td>
<td>S_j</td>
</tr>
<tr>
<td>$ChooseBranch$</td>
<td>The heuristic algorithm for selecting a bid $b_k$ to branch on.</td>
</tr>
</tbody>
</table>
Let $B_0$ be a set of bids $|B_0| \leq |B|$ and $\forall b_j \in B_0 \ e_j = 0$

Then $b_k \in B_0$ and $\forall b_j \in B_0 \quad \frac{p_k}{\sqrt{|S_k|}} \geq \frac{p_j}{\sqrt{|S_j|}}$

The square root of cardinality of item subsets has been shown to be effective at selecting bids with balance between high price bids, but with large number of items, verses bids with low item count, but also with low bidding value [23; 32]. This is an important feature that under proper circumstances may result in faster convergence to the winning bid allocation.

Table II Branch On Bids variables and terms

The BOB algorithm takes $M^*$ and $g$ as its input. Upon completion of BOB execution, $IN^*$ contains the set of winning bids. The first call is BOB($M,0$), please refer to “Table III Branch On Bids Pseudo code” for details.

<table>
<thead>
<tr>
<th>Step</th>
<th>Pseudo code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>//if current search is better, remember it as all-time best [\text{If } g &gt; f^<em>, \text{ then } IN \rightarrow IN^</em> \text{ and } g \rightarrow f^*]</td>
</tr>
<tr>
<td>2</td>
<td>//Calculate the upper bound [h = \sum_{i \in M^*} c(i)]</td>
</tr>
<tr>
<td>3</td>
<td>// Check if this branch can produce better results than we already have. // If not – bound [\text{If } g + h \leq f^*, \text{ return } ()]</td>
</tr>
<tr>
<td>4</td>
<td>Use ChooseBranch algorithm to select a bid $b_k$ to branch on [\text{If no such bid exists, return}]</td>
</tr>
</tbody>
</table>
// Prepare to branch on \( b_k \)

\[
IN \cup \{b_k\} \rightarrow IN
\]

\( e_k = 1 \)

// Update the exclusion count for all the bids that share items with \( b_k \)

\[
\forall \ b_j \in B \mid b_j \neq b_k \text{ and } S_j \cap S_k \neq \emptyset
\]

\( e_j = e_j + 1 \)

// Branch in

BOB(M\(^{-}\), g + \( p_k \))

// Prepare to branch out

\[
IN - \{b_k\} \rightarrow IN
\]

// Update the exclusion count for all the bids that share items with \( b_k \)

\[
\forall \ b_j \in B \mid b_j \neq b_k \text{ and } S_j \cap S_k \neq \emptyset
\]

\( e_j = e_j - 1 \)

// Branch \( b_k \) out

BOB(M\(^{-}\), g)

// Done with this branch

\( e_k = 0 \)

return

**Table III Branch On Bids Pseudo code**
CHAPTER IV. ARCHITECTURE

Following the Gaia methodology, the Architecture chapter is divided into two subsections: Analysis and Design. The Analysis subsection provides an abstract COD model, aiding with understanding of COD. The Design subsection provides the more concrete details of the solution, most importantly explaining the particulars of the CA-based negotiations protocol and the application of the WDP and the BOB algorithm to determining the set of courses to offer.

Analysis
In this section we seek to identify the following parts of the COD system: roles, interactions between the roles, a role’s responsibilities and permissions. The responsibilities include safety properties (things that should not happen) and liveness properties (things that should happen).

Prior chapters identify three important roles in COD: a student, an auctioneer and a school administrator. An agent type is assigned to each role respectively: the SA agent, the AA agent and the AD agent.

The Student Agent
As the name Student Agent suggests, the main responsibility of this role is the representation of the student it has been assigned to in the context of the COD System. The SA agent also serves in a reverse role as the student’s entry portal into the COD System. With the SA’s help the student keeps track of the study plan. The SA Agent is responsible for sending CA round beginning and ending notifications and for identifying the student’s biddable courses. During the
process of CA negotiations, the historical bids are kept by the SA Agent to help the student decide on new bids.

When a student enters a program and an SA agent is assigned, the course bidding points are transferred to the student’s account. These bidding points are intended to be used for course bidding throughout the duration of the program and are never replenished, thus serving as the limited resource emulating money. The SA agent is in charge of tracking these points: generating them at the beginning of the program, and transferring them if the student’s bids win.

The SA agent’s safety properties ensure that the student does not miss votes, bids only on courses that fit within their study program and satisfy inter-course relationships and that the potential winning bids to not exceed the amount of available bidding points.

The two role-defining goals for SA agents are the student’s completion of the program (long-term goal) and the student’s ability to take preferred courses (short-term goal). Winning the COD CA bids promotes the satisfaction of the student and increases the likelihood of achieving both goals.

SA Agents will need permissions to access the student’s academic records, the student’s course bidding points account and the program’s course database. In addition SA Agents have to be permitted to place bids and accept auction results on behalf of the student.

Table IV below enumerates the analysis of the SA agent.
Responsibilities

Generation of course bidding points at the beginning of a program.

Representing students in the COD System.

Serving as the student’s entry point to the COD System.

Tracking the student’s personal study plan.

Notifying the student of CA negotiation stages.

Tracking all historical bids.

Accepting student bids and submission of bids to AA agent.

<table>
<thead>
<tr>
<th>Safety Properties</th>
<th>Student cannot miss votes.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Bids are placed only on eligible courses.</td>
</tr>
<tr>
<td></td>
<td>Bidding values do not exceed available course points.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Liveness Properties</th>
<th>Student’s preferred courses are offered</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Student completes the program successfully and on schedule</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Permissions</th>
<th>Placing bids on behalf of the student</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Interactions</th>
<th>AA</th>
</tr>
</thead>
</table>

Table IV Student Agent Analysis

*THE AUCTIONEER*

The Auctioneer Role, represented by the AA agent, is responsible for the management and coordination of the CA auctions. Arguably the most challenging and important task the AA agent has to address is the solution of the WDP. It also enforces the auctions rules, communicates the beginning and ending of the
auction and auction rounds, and publishes the winning bids. The AA agent is also responsible for ensuring the winning bids comply with school resource constraints by acquiring bid approvals from the AD agent. As the central point of the CA protocol, the AA agent manages all communication flow between the System’s constituent agents.

From the Safety Properties point of view, the AA agent must ensure that the CA protocol does not execute indefinitely and has a reasonable termination strategy (the stopping point). Bids by outsiders should not be allowed, only SA agents may participate in the bidding process. The resource restriction may only be applied by the AD agent and the winning bids may not contradict the program’s and school’s resource constraints, as indicated by the AD agent.

Liveness Properties are limited to a successful, fair and efficient execution of the CA protocol. The conclusion of execution should be marked by determination of the course offering that generates the highest student utility and satisfies imposed constraints.

To fulfill its responsibilities the AA agent will need permissions to send CA notifications to SA and AD Agents. Notifications will include the start and ending of the auctions and the winning bids, including provisional winning bids.

Please refer to “Table V Auctioneer Agent Analysis” for the enumeration of AA agent analysis.

<p>| Responsibilities | Management, coordination and enforcement of CAs rules. |</p>
<table>
<thead>
<tr>
<th>Safety Properties</th>
<th>WDP solution in the context of COD.  Communication of negotiable and must offer courses to SAs  Initiation of CA negotiation.  Initiation of new CA rounds.  Termination of CA negotiation.  Communication of final course offering to SAs and ADs.  Notification of CA negotiation termination.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA may not execute indefinitely (terminates after decided number of rounds).  Bids on behalf of a student may only be placed by the student’s agent.  A student may win only a single bid per round.  Provisionary winning bids may not contradict AD’s constraints.  Constraints may be applied by AD only.</td>
<td></td>
</tr>
<tr>
<td>Liveness Properties</td>
<td>Successful execution of the CA protocol.  Course offering maximizes student’s satisfaction within school’s constraints.</td>
</tr>
<tr>
<td>Permissions</td>
<td>SA’s notifications of the beginning of CA negotiations.  SA’s notification of the beginning of a new CA negotiation round.  Forwarding provisionary winning bids to SAs.  Forwarding provisionary winning bids to AD.</td>
</tr>
</tbody>
</table>
As the school representative, the AD agent will be responsible for keeping track of school commitments and the set of program courses. Using this information the AD agent will need to prepare a set of must offer courses a set of must not offer courses and a set of negotiable courses. Communication of these sets to the AA agent would signify the beginning of the CA process and trigger all the steps leading to successful completion of COD negotiations. It is also the AD agent’s responsibility to track the school’s resource availability and constraints as well as validating the winning bids with the determined constraints.

As a safety responsibility the AD agent must ensure that the courses contained in the winning bids do not break the school’s resource constraints.

Through the process of CA negotiations, the AD agent is striving to find the course offering that maximizes student utility and enrollment, while minimizing the costs, ultimately, to satisfy the resource constraints.

Permissions required by the AD agent to fulfill its responsibilities include access to the program’s course database, access to information required to determine the school’s resource constraints and the selection of must offer, must not offer courses and negotiable courses. The agent must also have rights to send sets of
the courses to the AA agent and to respond to AA agent winning bid verification requests.

Please refer to “Table VI Administrator Agent Analysis” below for the enumeration of AD agent analysis

<table>
<thead>
<tr>
<th>Responsibilities</th>
<th>Track school commitments.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Track program courses.</td>
</tr>
<tr>
<td></td>
<td>Determine the set of must offer courses.</td>
</tr>
<tr>
<td></td>
<td>Determine the set of must not offer courses.</td>
</tr>
<tr>
<td></td>
<td>Determine the set of negotiable courses.</td>
</tr>
<tr>
<td></td>
<td>Communicate negotiable, must offer and must not offer courses to AA.</td>
</tr>
<tr>
<td></td>
<td>Ensure that provisionary winning bids satisfy school’s resource constraints.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Safety Properties</th>
<th>Selected courses may not break school’s resource constraints.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Liveness Properties</th>
<th>Determine course offering that maximizes student’s satisfaction while keeping to school’s resource constraints</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Permissions</th>
<th>Access to program’s courses database</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Access to information required to determine resource constraints</td>
</tr>
<tr>
<td></td>
<td>Sending course sets to AA</td>
</tr>
<tr>
<td></td>
<td>Approving/Denying provisionary winning bids</td>
</tr>
</tbody>
</table>
For the visual interactions model please review Figure VII below:

**Table VI Administrator Agent Analysis**

**THE INTERACTIONS**

Table VII enumerates all COD interactions that happen during CA negotiations and provides their descriptions.

<table>
<thead>
<tr>
<th>Step</th>
<th>From</th>
<th>To</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AD</td>
<td>AA</td>
<td>AD sends to AA the sets of must offer, must not offer and negotiable courses</td>
</tr>
<tr>
<td>2</td>
<td>AA</td>
<td>SA</td>
<td>AA sends to SAs the sets of must offer, must not offer, negotiable courses and the call for bids</td>
</tr>
<tr>
<td>3</td>
<td>SA</td>
<td>AA</td>
<td>SAs submit the bids (each CA round)</td>
</tr>
<tr>
<td>4</td>
<td>AA</td>
<td>AD</td>
<td>Each time a new bid is considered as a provisional winning</td>
</tr>
</tbody>
</table>
bid, AA sends the set of provisional winning bids to AD for verification against resource constraints.

5. AD approves or denies the provisional winning bids proposed in step 4. The decision is based upon available school resources.

6. Upon completion of each round, AA sends all provisionary winning bids to SAs. New round begins by cycling to step 3. Students place new bids against provisionary winning bids in step 6.

7. AA notifies SA of CA completion.

8. AA notifies AD of CA completion.

9. AA sends the final course offering to SA.

10. AA sends the final course offering to AD.

<table>
<thead>
<tr>
<th>Step</th>
<th>Sender</th>
<th>Receiver</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>AD</td>
<td>AA</td>
<td>AD approves or denies the provisional winning bids proposed in step 4. The decision is based upon available school resources.</td>
</tr>
<tr>
<td>6</td>
<td>AA</td>
<td>SA</td>
<td>Upon completion of each round, AA sends all provisionary winning bids to SAs. New round begins by cycling to step 3. Students place new bids against provisionary winning bids in step 6.</td>
</tr>
<tr>
<td>7</td>
<td>AA</td>
<td>SA</td>
<td>AA notifies SA of CA completion.</td>
</tr>
<tr>
<td>8</td>
<td>AA</td>
<td>AD</td>
<td>AA notifies AD of CA completion.</td>
</tr>
<tr>
<td>9</td>
<td>AA</td>
<td>SA</td>
<td>AA sends the final course offering to SA.</td>
</tr>
<tr>
<td>10</td>
<td>AA</td>
<td>AD</td>
<td>AA sends the final course offering to AD.</td>
</tr>
</tbody>
</table>

Table VII Course Offering Determination Interaction Model

**DESIGN**

**BIDDING LANGUAGE**

We begin our design discussion with the introduction of the bidding language.

Bellow we assign semantic meaning to syntactic constructs that can be later used to define student’s course bids.

**Definition 1**

A Submission is a three-tuple \(<B, sa, po, co>\)

For variable definition please reference “Table VIII Course Offering Determination Submission Variables”.

48
Variable | Definition
--- | ---
\(B\) | \(\{b_1, b_2, ..., b_n\}\) is a set of student bids, \(|B|=n\), the bids are a composite value defined below
\(\text{sa}\) | the name or identification of the SA agent submitting the bid
\(\text{po}\) | the total amount of course bidding points available to the SA agent
\(\text{co}\) | the maximum number of courses the student is allowed to take and therefore the maximum number of course the student can win for successfully placed bids

Table VIII Course Offering Determination Submission Variables

**Definition 2**

A bid \(b_j\) is a two-tuple \(<S_j,p_j>\)

For variable definition please reference “Table IX Course Offering Determination bid variables”.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>(S_j)</td>
<td>({s_1, s_2, ..., s_k}), (S_j \subseteq M), (</td>
</tr>
<tr>
<td>(p_j)</td>
<td>the amount of course bidding points assigned to this bid by the student</td>
</tr>
<tr>
<td>(M)</td>
<td>({i_1, i_2, ..., i_m}), (</td>
</tr>
<tr>
<td>(O)</td>
<td>({o_1, o_2, ... , o_l}), (</td>
</tr>
</tbody>
</table>
\[ N = \{ n_1, n_2, \ldots, n_t \}, |N|=t \]
a set of negotiable courses

Table IX Course Offering Determination bid variables

Submission Interpretation
The set of bids \( B \) is interpreted by the CA protocol as a non-exclusive OR concatenation of the bids, baring the following rules

- Winning bids from the same student may not contain intersecting sets of courses.
- Winning bids may not add up to a total exceeding the amount of course bidding point available to the student
- The number of courses comprising winning bids may not exceed maximum number of courses the student is allowed to take
- The courses comprising winning bids may not contradict course interrelationships (for example there may not be any antirequisites)

Example 1

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Submission</td>
<td>(&lt;B, &quot;sa1&quot;, 10, 6&gt;)</td>
</tr>
<tr>
<td>(B)</td>
<td>({b_1, b_2, b_3})</td>
</tr>
<tr>
<td>(b_1)</td>
<td>(&lt;S_1, 6&gt;)</td>
</tr>
<tr>
<td>(S_1)</td>
<td>{&quot;comp501&quot;,&quot;comp503&quot;,&quot;comp504&quot;}</td>
</tr>
<tr>
<td>(b_2)</td>
<td>(&lt;S_2, 6&gt;)</td>
</tr>
<tr>
<td>(S_2)</td>
<td>{&quot;comp504&quot;,&quot;comp505&quot;,&quot;comp506&quot;}</td>
</tr>
<tr>
<td>(b_3)</td>
<td>(&lt;S_3, 2&gt;)</td>
</tr>
</tbody>
</table>
In this example a student, represented by the SA agent “sa1”, submits three bids: \(\{b_1, b_2, b_3\}\). The student has ten course bidding points available and can take six courses. The following combination may comprise winning bids: \(\{b_1\}, \{b_2\}, \{b_3\}\), \(\{b_1, b_3\}\), \(\{b_2, b_3\}\). Note that \(\{b_1, b_2\}\) is not a viable combination because the the sets of courses \(S_1\) and \(S_2\) intersect and because the sum of bids for \(b_1\) and \(b_2\) exceeds the number of available points.

**Example 2**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Submission</td>
<td>(&lt;B, \text{&quot;sa2&quot;}, 10, 3&gt;)</td>
</tr>
<tr>
<td>B</td>
<td>({b_1, b_2, b_3})</td>
</tr>
<tr>
<td>(b_1)</td>
<td>(&lt;S_1, 2&gt;)</td>
</tr>
<tr>
<td>(S_1)</td>
<td>{&quot;comp601&quot;,&quot;comp602&quot;}</td>
</tr>
<tr>
<td>(b_2)</td>
<td>(&lt;S_2, 2&gt;)</td>
</tr>
<tr>
<td>(S_2)</td>
<td>{&quot;comp604&quot;,&quot;comp605&quot;}</td>
</tr>
<tr>
<td>(b_3)</td>
<td>(&lt;S_3, 2&gt;)</td>
</tr>
<tr>
<td>(S_3)</td>
<td>{&quot;comp607&quot;,&quot;comp610&quot;}</td>
</tr>
</tbody>
</table>

In example 2 a student represented by the SA agent “sa2” again submits three bids: \(\{b_1, b_2, b_3\}\). The student has ten course bidding points available and can take three courses. Here the only viable options to consider are: \(\{b_1\}, \{b_2\}, \{b_3\}\). All other combinations of the bids \(\{\{b_1, b_3\}, \{b_2, b_3\}, \{b_1, b_2\}\}\) are not allowed.
because they would exceed the maximum number of courses allowed for the student.

**COMBINATORIAL AUCTIONS, WINNER DETERMINATION PROBLEM AND BRANCH ON BIDS IN COURSE OFFERING DETERMINATION CONTEXT**

In this subsection we are going to discuss the application of CAs and the WDP to the COD problem. We will expand and build upon the definitions and algorithms introduced in the Methodology chapter and modify them as needed to fit the COD environment.

A notable change is the need to adjust to the fact that multiple students can bid on the same course or on the same combination of courses and win. This is allowed because multiple students can take the same course or the same combination of courses at the same time, the only limit on the number of students taking the same course is the maximum course enrollment requirement enforced by the AD agent. We address this challenge through the introduction of multi-unit auctions [23], with the number of available course units equal to the enrollment limit for the course. Using this approach we can apply the Branch-on-Bids algorithm with only minor modifications.

**Definitions**

Reusing and adjusting the formal definition of CAs from the Methodology chapter we define CA in the context of COD as a three-tuple

\[<M,U,B>\]

Where variables are defined in “Table XII Course Offering Determination CA. Variable definition”.

52
<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>M (i_1, i_2, \ldots, i_m), (</td>
<td>M</td>
</tr>
<tr>
<td>U (u_1, u_2, \ldots, u_m), (</td>
<td>U</td>
</tr>
<tr>
<td>B (b_1, b_2, \ldots, b_n)</td>
<td>is a set of student bids</td>
</tr>
</tbody>
</table>

Table XII Course Offering Determination CA. Variable definition

A WDP is defined as follows

\[
\max \sum_{j=1}^{n} p_j x_j \\
\text{s.t. } \sum_{j \in \mathcal{S}_i} x_j \leq u_i, \quad i = 1, 2, \ldots, m \\
\quad x_j \in \{0, 1\}
\]

*Other constraints and limitations from SA and AD agents.*

Note that in the context of COD an item (a course) within a bid will always have quantity of exactly one because a student can only enroll in a course once per semester. Therefore we use a simplified version of the multiunit WDP from [23].

We also add to the set of constraints all limits induced by limitations of SA agents (see Submission Interpretation) and the resource limitations enforced by the AD agent.

**BOB Algorithm for COD**
The BOB for COD algorithm relies on the variables outlined in “Table II Branch On Bids variables and terms” and the adjusted variables outlined in “Table XIII Branch on Bids for Course Offering Determination variables”

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M'$</td>
<td>The set of available courses (courses that have available enrollment places). Can be thought of as a set of auction items that still have available inventory.</td>
</tr>
<tr>
<td>$U'$</td>
<td>The set of course units matching $M'$. When branching in on a bid, the course units for the courses in the bids get subtracted a unit. When branching out on a bid, the course units for the courses in the bid get an extra unit. In other words, this variable tracks available enrollment places in a course.</td>
</tr>
</tbody>
</table>
| $c(i)$   | The admissible heuristic for estimating revenue for item $i \in M'$ $c(i) = \max_{j | i \in S, e_j = 0} u_i * p_j / |S_j|$  
(this formula can be interpreted as the number of available course units multiplied by the maximum per-item bidding price among all bids containing the item $i$ and that are not part of $IN$ or $IN^*$) |
| $d$      | The depth of the current branch |
| $EX$     | A set of “exclusion” two-tuples $<d,b>$  
A tuple represents a bid $b$ excluded at branch depth $d$ due to violation of Administrator or Student constraints. |
**ChooseBranch**  
The heuristic algorithm for selecting a bid $b_k$ to branch on.

Let $B_0$ be a set of bids s.t. $b_j \in B_0$ iff

$b_j \in B$ and $e_j = 0$ and $<d, b_j> \notin \mathcal{E}X$ for any integer $d$

Then $b_k \in B_0$ and $\forall b_j \in B_0$, \[ \frac{p_k}{\sqrt{|S_k|}} \geq \frac{p_j}{\sqrt{|S_j|}} \]

**SA_Constraints**  
Winning bids from the same student may not contain intersecting sets of courses.

Winning bids may not add up to a total exceeding the amount of course bidding point available to the student

The number of courses comprising winning bids may not exceed maximum number of courses the student is allowed to take

The courses comprising winning bids may not contradict course interrelationships (for example there may not be any antirequisites)

**AD_Constraints**  
For the purpose of this paper, the AD agent will enforce the maximum number of courses that the school can offer

| BOB takes on input the set of available courses, their associated available enrollment limits, current search revenue and branch depth. The first call looks like this: BOB(M, U, 0, 1). The pseudo code listing for the BOB application to COD is provided in table “Table XIV Branch on Bids for Course Offering Determination pseudo code”. |  |

Table XIII Branch on Bids for Course Offering Determination variables
<table>
<thead>
<tr>
<th>Step</th>
<th>Pseudo code</th>
</tr>
</thead>
</table>
| 1    | //if current search is better, remember it as the all-time best  
      | If \( g > f^* \), then \( IN \rightarrow IN^* \) and \( g \rightarrow f^* \) |
| 2    | //Calculate the upper bound  
      | \( h = \sum_{i \in M^c} c(i) \) |
| 3    | // Check if this branch can produce better results than we already have.  
      | // If not – bound  
      | If \( g + h \leq f^* \), return () |
| 4    | Use the ChooseBranch algorithm to select a bid \( b_k \) to branch on  
      | If no such bid exists, return |
| 5    | // Prepare to branch on \( b_k \)  
      | \( IN \cup \{b_k\} \rightarrow IN \)  
      | \( e_k = 1 \) |
| 6    | Verify that \( IN \) set satisfies the SA_Constraints and the AD_Constraints  
      | If not then  
      | \( IN - \{b_k\} \rightarrow IN \)  
      | \( e_k = 0 \)  
      | \( EX \cup \{<d, b_k>\} \rightarrow EX \)  
      | Return to step 4 |
| 7    | // Update the set of available course units \( U^c \).  
<pre><code>  | For each course in \( S_k \) subtract 1 from the matching course unit in \( U^c \) |
</code></pre>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
| 8 | // Update the set of available courses $M^\prime$.  
If an item $i_j \in M^\prime$ and $u_j = 0$ then  
$M^\prime = M^\prime - \{i_j\}$ |
| 9 | // Update the exclusion count for all the bids that share courses with $b_k$  
// and have no enrollment places available  
$\forall \ b_j \in B \mid b_j \neq b_k \text{ and } S_j \cap S_k \neq \emptyset \text{ and } \exists \text{ an item } i_e \in S_j \cap S_k \text{ s.t. } u_e = 0$  
then $e_j = e_j + 1$ |
| 10 | // Branch in  
$\text{BOB}(M^\prime, U^\prime, g + p_k, d+1)$ |
| 11 | // Prepare to branch out  
$\text{IN - \{b_k\} \rightarrow IN}$ |
| 12 | // Update the exclusion count for all the bids that share courses with $b_k$  
// and have no enrollment places available  
$\forall \ b_j \in B \mid b_j \neq b_k \text{ and } S_j \cap S_k \neq \emptyset \text{ and } \exists \text{ an item } i_e \in S_j \cap S_k \text{ s.t. } u_e = 0$  
then $e_j = e_j - 1$ |
| 13 | // Update the set of available course units $U^\prime$.  
For each course in $S_k$ add 1 to the matching course unit in $U^\prime$ |
| 14 | // Update the set of available courses $M^\prime$.  
If an item $i_j \notin M^\prime$ and $u_j \geq 0$ then  
$M^\prime = M^\prime + \{i_j\}$ |
| 15 | // Update the exclusions set EX  
Remove all exclusions for branch d |
\[ \forall \; ex_j \in EX \; s.t. \; d_j = d \]

\[ EX = EX - \{ex_j\} \]

16  //Branch \( b_k \) out

BOB(M', U', g, d)

17  //Done with this branch

\[ e_k = 0 \]

return

<table>
<thead>
<tr>
<th>Table XIV Branch on Bids for Course Offering Determination pseudo code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upon the completion of the algorithm execution the ( IN^* ) set will contain the set of winning bids.</td>
</tr>
<tr>
<td><strong>SOLUTION QUALITY</strong></td>
</tr>
</tbody>
</table>

Distributed MAS solution quality is a complex, multifaceted topic that involves, amongst others, those of social welfare of agent societies, individual rationality, stability and efficiency [35].

The practical side of solution quality should be measured through a pilot project that would follow a group of participating students through their academic progress for a period of one or more semesters. The student’s subjective utility with the program and academic performance may be compared to those of non-participating students, drawing the conclusions on the quality of proposed solution.

In lieu of a pilot program, we follow up with a discussion based on the quality analysis principles outlined in [35].
The CA-based COD algorithm is a form of a negotiation mechanism that through the use of CAs provides a transparent and economical solution for COD. The mechanism aims to benefit the crucial players of the system: the students and the administrators, increasing social welfare through the mutually satisfactory solution. We measure social benefit of the students through the value they assign to the courses in their bids, and the administrator’s benefit through student enrollment (subject to the resource constraints). It follows that since CAs maximize the economical payoff through increasing the course bidding points revenue, the CA-based COD algorithm should lead to the course selection most beneficial within the constructed environment.

Moreover, the proposed solution encourages student’s participation in the bidding process through the potential to improve the Course Offering for the bidding student. Participation in the bidding process will always lead to the results that are at least as good as for non-participating students thus providing the individual rationality.

The bidding process and the limited course bidding points inject stability into the protocol, promoting truthful bidding for the courses the students are interested in.

**Performance**
An unsatisfactory performance may render even the best designs unusable in practice. It is most noticeable in on-line, real-time applications, where a slow response time may dwindle user performance or lead to customer attrition. However in off-line applications performance plays much less of an important role.
In the case of the CA-based COD algorithm, the WDP is solved off-line and therefore is not expected or required to generate results in milliseconds. The only expectation we may impose is that the problem is solved in a reasonable time and the solution process does not consume the school’s computational resources in a manner significantly detrimental to the rest of the services. The looseness of this performance requirement is possible because the COD needs to be addressed only once a semester or once a year, and the bidding rounds are executed once a day (or possibly on some other prolonged schedule) during the COD process and do not require immediate response.

The version of BOB utilized in this paper is a search algorithm, that builds and traverses a binary bid-search tree, branching on bids [23]. The performance of BOB is proportional to the number of leaves in our search tree which is not greater than

\[
\left( \frac{n}{m/k} + 1 \right)^{\lfloor m/k \rfloor} \quad [23]
\]

The variables are described in “Table XV. Branch On Bids Performance Variables”.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n )</td>
<td>The number of bids</td>
</tr>
<tr>
<td>( m )</td>
<td>The number of items</td>
</tr>
<tr>
<td>( k )</td>
<td>The smallest number of items among all bids</td>
</tr>
</tbody>
</table>

Table XV. Branch On Bids Performance Variables
The number of leaves is exponential on the number of items (courses available for bids), but polynomial on the number of bids. This is an important and positive observation because program administrators can control the number of biddable items (courses) in case of unsatisfactory algorithm performance. The number of bids, however, depends on the number of students and their personal preferences and involvement, and therefore is harder to control.

The original BOB algorithm was implemented by its authors and the performance was analyzed [28]. Bid sets containing from five hundred to two thousand bids for sets of ten items were solved in less than six seconds. While bid sets containing four hundred and fifty bids for sets of forty five items were solved in less than twenty seconds.

In the context of COD these results appear to be very promising. We can estimate that the number of items (courses) in most COD auctions is unlikely to exceed a few dozen. The number of bids will vary between programs and schools and will greatly depend on student enrollment, however for the majority of the programs it will remain under a few thousand. It appears likely that WDP for COD may be solved within a reasonable time-frame, measuring seconds or minutes.

**Example**
In this subsection we will provide a small example that illustrates the partial listing of the BOB for COD algorithm flow, terminating the listing at the first leaf. Please note the simple form of the student preference submissions, which represent a clear improvement over precedence, grouping and progression model used in [1] and [18]. Rationality for the winning bids is also easy to understand, which should
lead to students expressing their preferences through bidding rounds with higher precision.

Let us begin by assuming that due to resource constraints (AD_Constraints) we can offer at most five courses. Next we define the CA (“Table XVI Branch on Bids Example Definition”).

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>{c1, c2, c3, c4, c5}</td>
</tr>
<tr>
<td>U</td>
<td>{u1=1, u2=2, u3=2, u4=1, u5=1}</td>
</tr>
<tr>
<td>Submission1</td>
<td>{(c1,c2),4&gt;,(c3,c4),3&gt;,sa1,10,4&gt;</td>
</tr>
<tr>
<td>Submission2</td>
<td>{(c4,c5),4&gt;,sa3,10,4&gt;</td>
</tr>
</tbody>
</table>

Table XVI Branch on Bids Example Definition

The execution flow is provided below (“Table XVI Branch on Bids Example Definition”)

<table>
<thead>
<tr>
<th>Execution Step</th>
<th>BOB Step</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>n/a</td>
<td>BOB({c1, c2, c3, c4, c5}, {u1=1, u2=2, u3=2, u4=1, u5=1}, 0, 1)</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>n/a</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>h = c(c1) + c(c2) + c(c3) + c(c4) + c(c5) = 2 + 4 + 3 + 2 + 2 = 13</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>13 &gt; 0 -- proceed</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>Following the ChooseBranch algorithm we select bid</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>
| 6 | 5 | IN=\{\langle c_1,c_2\rangle,4\}\>
|   |   | \(e_1 = 1\) |
| 7 | 6 | No constraint violations -- proceed |
| 8 | 7 | U^\prime=\{u_1=0, u_2=1, u_3=2, u_4=1, u_5=1\} |
| 9 | 8 | M^\prime=\{c_2, c_3, c_4, c_5\} |
| 10 | 9 | No bids intersect with \(b_1\) -- proceed |
| 11 | 10 | BOB(\{c_2, c_3, c_4, c_5\}, \{u_1=0, u_2=1, u_3=2, u_4=1, u_5=1\}, 4, 2) |
| 12 | 1 | IN^\ast=\{\langle c_1,c_2\rangle,4\}, f^\ast=4 |
| 13 | 2 | \(h=c(c_2) + c(c_3) + c(c_4) + c(c_5) = 0 + 3 + 2 + 2 = 7\) |
| 14 | 3 | \(4 + 7 = 11 > 4\) -- proceed |
| 15 | 4 | Following ChooseBranch algorithm we select bid \(\langle c_4,c_5\rangle,4\) |
| 16 | 5 | IN=\{\langle c_1,c_2\rangle,4\},\langle c_4,c_5\rangle,4\}\>
|   |   | \(e_3 = 1\) |
| 17 | 6 | No constraint violations -- proceed |
| 18 | 7 | U^\prime=\{u_1=0, u_2=1, u_3=2, u_4=0, u_5=0\} |
| 19 | 8 | M^\prime=\{c_2,c_3\} |
| 20 | 9 | \(e_2 = 1\) |
| 21 | 10 | BOB(\{c_2, c_3\}, \{u_1=0, u_2=1, u_3=2, u_4=0, u_5=0\}, 8, 3) |
| 22 | 1 | IN^\ast=\{\langle c_1,c_2\rangle,4\},\langle c_4,c_5\rangle,4\}, f^\ast=8 |
| 23 | 2 | \(h=c(c_2) + c(c_3) = 0 + 3 = 3\) |
| 24 | 3 | \(8+3=11>8\) -- proceed |
| 25 | 4 | At this point there are not more bids available -- return |
Table XVII Partial flow of Branch on Bids for Course Offering Determination example

We terminated our example at step 25, however the algorithm will continue to explore remaining branches, determining \( \{c_1,c_2\},4;\{c_4,c_5\},4 \) as the winning bids. Figure VIII shows the complete progress of the algorithm in a graphical form. The bright orange rectangles represent bids in the IN set, the grey rectangles represent the excluded bids.

![Figure VIII Branch on Bids for Course Offering Determination example](image-url)
CHAPTER V. CONCLUSIONS AND RECOMMENDATIONS

CONCLUSIONS
COD is a dynamic, distributed resource allocation problem. We based our research on existing designs for the MAS-based COD solutions that built the COD model through the use of the Contract Net Protocol and the Single Transferable Vote.

The notable innovation of the research was the introduction of CAs to the process of the COD negotiations. CAs bring a number of benefits to distributed negotiation protocols, for instance: accounting for the complimentary values of the course bundles, increased economic efficiency, expressive negotiations format that lowers transaction costs and high level of transparency that ensures fairness.

The challenge of efficient negotiation was met with introduction of multi-round auctions, and the ability for multiple students to win bids on the same courses was handled through multi-unit auctions. The intractable WDP was tackled using the BOB algorithm that offers polynomial performance on bids and generalization that includes the multi-unit solutions.

The architecture was approached with reliance on the Gaia methodology for Agent Oriented Design and this research paper contains the high level CA and MAS-based COD analysis, the model and the communication flows, as well as
the details for the CA bidding language and the pseudo code for the WDP in the COD context.

The expected performance of the CA-based COD solution promises to be within reasonable and practical limits. The most computation-costly part of the algorithm, the WDP, is solved off-line and is expected to complete execution within seconds or minutes for the most extreme cases.

The implementation of the designed CA-based COD solution as well as the completion of a full round of tests should provide an experimental basis for our future research. The solution may be implemented with Java, employing the JADE platform for Multiagent Systems.

FUTURE RESEARCH
A potential area of interest for CA-based COD research may be the application of it to the Massive Open Online Courses (MOOC). Some of the popular MOOC providers offer upwards of five hundred courses, catering to over four million students [36]. The scale and popularity of online programs serve as a powerful argument in favor of application of CA-based COD, which aspires to improve economic efficiencies.

A further elaboration of the CA-based COD solution may also improve its efficiency and flexibility:

- The bidding language may be extended to include complex logical combinations of bids, such as “AND” and “EXCLUSIVE OR”
• Student and Administrator constraints may be improved to provide more realistic rules. For example Administrators may not only limit the total number of courses, but also courses that may not be offered together.

• The ChooseBranch and c(i) heuristics may be improved from their general form to more applicable algorithms that take into account the specifics of COD.

• The SA and AD agents may be extended with the use of Machine Learning techniques to self-customization for the purpose of providing a more personalized service to the students and administrators they represent.

• A data warehouse and data mining system may be designed to learn the effects of the CA-based COD on the welfare of the system, and empower research for further improvements.

APPENDIX A. ACRONYMS.

AA – Auctioneer Agent
AD – Administrator Agent

BOB – Branch On Bids

CA – Combinatorial Auctions

CNP – Contract Net Protocol

COD – Course Offering Determination

MCP – Monotonic Concession Protocol

MAS – Multiagent Systems

MOOC – Massive Open Online Courses

SA – Student Agent

WDP – Winner Determination Problem

REFERENCES


http://dtpr.lib.athabascau.ca/action/viewdtrdesc.php?cpk=495&id=48806


31. Moraitis P., Petraki E., Spanoudakis N., Engineering JADE Agents with the Gaia Methodology, Chapter extract from book: Agent Technologies,


