Adaptive Multiagent System for Learning Gap Identification Through Semantic Communication and Classified Rules Learning

Kennedy E. Ehimwenma, Martin Beer and Paul Crowther
Communication & Computing Research Centre, Sheffield Hallam University, Sheffield, U.K.
{kennedy.k.ehimwenma@student., m.beer@, p.crowther@}shu.ac.uk

Keywords: Multiagent, Classification, Semantics, Learning, Pre-assessment, Knowledge Representation, Prerequisite, Jason Agentspeak.

Abstract: Work on intelligent systems application for learning, teaching and assessment (LTA) uses different strategies and parameters to recommend learning and measure learning outcome. In this paper, we show how agents can identify gaps in human learning, then the use of a set of parameters which includes *desired_concept*, *passed* and *failed* predicate attributes of students in the construction of an array of classified production rules which in-turn make prediction for multipath learning after pre-assessment in a multiagent system. The context in which this system is developed is structured query language (SQL) domain with concepts being represented in a hierarchical structure where a lower concept is a prerequisite to its higher concept.

1 INTRODUCTION

Different set of parameters or approaches have been used by researchers e.g. Chakraborty, Roy & Basu (2010), Mills & Dalgarno (2007) and Matsuda *et al.* (n.d.) to model intelligent systems for human learning. However, to the best of our knowledge no system has been modelled to: 1) diagnose the gap(s) in students learning using the set of modelling parameters (*desired_concept*, *passed* and *failed* predicates) used in this work, 2) use of predicate or description logic (DL) semantic literals that are understandable between agents for developing intelligent tutoring system (ITS), and 3) modelling production rules for learning material prediction.

1.1 Objectives of Research

- To use some set of parameters that comprises of *desired_Concept, Passed,* and *Failed* attributes in diagnosing knowledge gaps in students;
- To use symbolic representation and speech acts performatives (i.e. communicative acts) in supporting the development of an intelligent multiagent based Pre-assessment System; and
- To use production-rule multiple classification and learning technique by agents in the development of ITS on the platform of Jason API.

1.2 Research Problem

How can intelligent agent system identify gaps and guide students so that they are fully prepared for the next stage in their learning?

1.3 Related Work

Agents, machine learning and fuzzy logic approaches have been used in the development of computer based learning systems. For example, *SimStudent* (Matsuda *et al.,* n.d.) was developed using agent and machine learning for modelling the system. Chakraborty, Roy & Basu (2010) engaged the use of two-parameter attributes, namely: comprehensive ability and problem solving skill—<C, P>—to develop a student model using a combination of multiagent system (MAS) and machine learning. In Mills & Dalgarno (2007), a combination of machine learning technique and MAS were also used to provide hints to students on a current learning goal and to make performance prediction. In the foregoing analysis of agents and machine learning approaches to LTA technology, *SimStudent* (Matsuda *et al.,* n.d.) was used only for tutoring, Chakraborty, Roy & Basu (2010) was used for formative measurement, and Mills & Dalgarno

(2007) for making performance prediction. However, none of these have mentioned the same set of parameters, or the use of DL programming tool nor multiple classification ripple down rule: MCRDR (Kang et al., 1995) inference method approach in the development of intelligent learning and tutoring system as applied in this research.

2 STATE OF THE ART: THE PRE-ASSESSMENT SYSTEM

2.1 Agents & Multiagent System

An agent is a computer system that is situated in some environment, and capable of autonomous action in this environment in order to meet its design objectives (Wooldridge, 2009). A group of distributed agents with a collective goal constitutes a multiagent system. The pre-assessment system is a multiagent based system (Fig. 1) implemented in Jason AgentSpeak **(**Bordini et al., 2007). This system has five agents which are described as follows:

2.1.1 Agent agInterface

Observes the dynamic user inputs on the artifact.

2.1.2 Agent agModelling

This is referred to as the *classifier*. It learns and classifies the attributes received from the agent agSupport.

2.1.3 Agent Student Model

This is the agent that constructs and keep track of all the student activity that is communicated from the agent agSupport. The Student Model agent uses *Jason TextPersistentBB class.* The TextPersistentBB is a persistent text file that captures all the activity or learning history of a student which are his: desired concept, pre-assessment questions, and correct and/or incorrect answers to questions.

2.1.4 Agent agSupport

This is the *teacher* in terms of machine learning. It pre-assesses the student based on his desired state concept received from the agent agInterface. The agent also communicate the outcome of pre-assessment to the agent agModelling and agModel, respectively, and connects the MAS to the MySQL database engine for result-set queries.

2.1.5 Agent agMaterial

This is the *ontology* agent that has all ontological relations initialised in its BB including the URL data value of any SQL concept. This agent outputs the appropriate URL learning material after it is communicated by the classifier—agent agModelling.

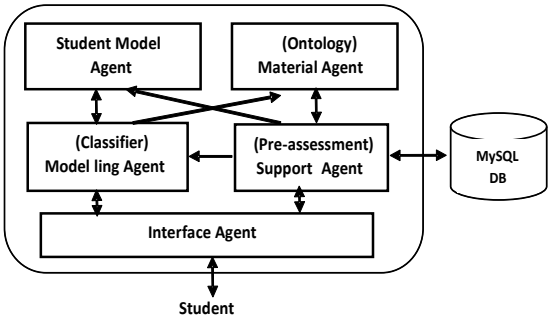

Figure 1: Multiagent System Architecture and Interaction.

2.2 Knowledge Representation

The domain content of the Pre-assessment System is SQL. This has been represented as an ontology of class and subclass concepts. To logically establish the relationships that exist between the concepts of the ontology, the SQL ontology was modelled, verified and validated with Protégé 4.3 as a structured hierarchy of learning content, and then initialised as internal knowledge model in the agent *agMaterial* beliefbase (BB). The agents in the MAS share, understand and communicate knowledge about these concepts as messages effectively. From the ontology, the MAS can:

- test (using a *test goal*) whether a desired topic entered by a student exists within the ontology after it is perceived by the agent *agInterface* with a reply confirmation expected for this request;
- trigger event to return recommended URL links (data values of leaf-nodes) to students after the agent *agModelling* classification of that student for appropriate learning from its array of predicated production rules and plans.

The following is a snippet of the knowledge representation (KR) of the SQL ontology with every relation semantically annotated as `[o(sql)]`. Class-to-class relations uses the *hasPrerequisite* predicate while the class-to-subclass relations uses the *has_KB* relation and has two leaf-nodes. The

subclass to data value relation uses the *hasContent* predicate.

```
hasPrerequisite(union, join)[o(sql)].
   has_KB(join, outer_join)[o(sql)].
   has_KB(join,inner_join)[o(sql)].
hasPrerequisite(join, update)[o(sql)].
   has_KB(update,update_select)[o(sql)].
   has_KB(update,update_where)[o(sql)].
hasPrerequisite(upadate, delete)[o(sql)].
   has_KB(delete, delete_select)[o(sql)].
   has_KB(delete, delete_where)[o(sql)].
hasPrerequisite(delete, insert)[o(sql)].
   has_KB(insert, insert_select)[o(sql)].
   has_KB(insert, insert_value)[o(sql)].
hasPrerequisite(insert, select)[o(sql)].
   has_KB(select, select_where)[o(sql)].
   has_KB(select, select_all)[o(sql)].
hasPrerequisite(select, select)[o(sql)].
```

2.3 Student Model

To identify gaps in students' learning, the model agent (shown as agent *student* in Fig. 2) is modelled to keep four parameter-information persistently about a given student in its BB. In a tuple, we represent this information as: M = <D, P, F, V> where

M: is the model
D: is a set of desired concepts i.e. desired state
P: is a set of passed pre-assessment i.e. current state gains
F: is a set of failed pre-assessment i.e. current state gaps
V: is the set of SQL query statements.

Parameters <D, P, F> which are also simultaneously communicated by the agent *agSupport* to the agent *agModelling* are gathered, learned as *pre-conditions* within which the appropriate plan is selected to classify a student and make prediction for his

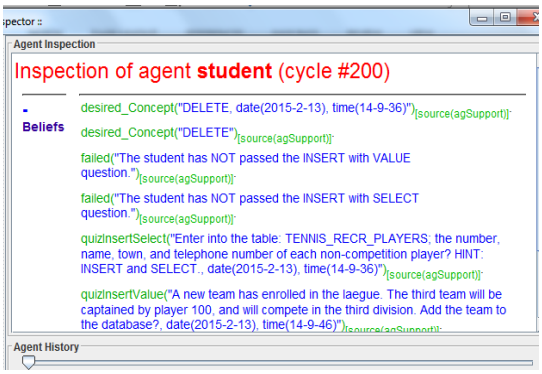

Figure 2: A snapshot of the model agent "***student***" Persistent BB that the human tutor can access and assess to identify skills gain or gap 24/7 including date and time stamp of each activity.

learning material. The rule in a Jason plan format for this classification is given as:

+recommend_material : set_of_profile_parameters
<- recommended_material.

After the agent agSupport pre-assesses a student, it communicates the student attributes to the agent *agModelling*. On receipt of these attributes, the agModelling classifies and predict some learning path for the student depending on a *desired concept*, *passed* or *failed* set of parameters received.

3 METHODOLOGY

3.1 Systems Development

The systems development approach to the pre-assessment system design is the Agent-Oriented Analysis and Design (AOAD)—agent software engineering cycle that involves: systems specification, analysis, design and implementation. The AOAD is agent development methodology that is based on goals, plans and belief. In this work, we have closely followed the *Prometheus* methodology for developing intelligent agent systems (Padgham and Winikoff, 2004). As a set of guidelines, the Prometheus methodology involves three major agent software development steps, given below:

3.1.1 System Specification

This is the phase that presents a scenario of the problem, identification of goals and basic functionalities of the system along with inputs (percepts) and outputs (actions).

3.1.2 Architectural Design

This is the phase where the number and type of agents are decided. It also consist of the system overall (static) structure using system overview diagram, and the description of the dynamic behaviour of the system using interaction diagram and interaction protocols.

3.1.3 Detailed Design

This phase is focused on the description of responsibility and development of the internal structure of each agent, and how they will achieve their task within the system.

3.2 Goal Specification

As natural construct, goals are central to the

functioning of intelligent agent that are going to realise a functioning system (Padgham and Winikoff, 2004). The use of goals or subgoals breakdown a scenario into units of achievable design steps which map details into later design and implementation. The following outlines the entire goal of the MAS that split up into responsibilities for the various agents:

- Receive Concept
- Fetch subConcept (pre-requisite) quiz
- Test student
- Receive answer from student
- Analyse and pre-assess answer
- Feedback to student
- Update model agent KB
- Classify the student
- Fetch Concept or subConcept materials
- Tutor the student

3.3 Multiple Classification Learning

As the *classifier*, the agent *agModelling* learns every attribute of the parameters it receives from the agent *agSupport* during the course of pre-assessment. Below is an exemplary code in Jason AgentSpeak from the agent *agModelling* plan library for a pre-assessment on the INSERT prerequisite if a DELETE is received as the desired concept:

```
/* Prediction rules for DELETE concept */
@d1
+!recommendMaterial[source(agSupport)] :
desired_Concept("DELETE")[source(agSupport)]
      & passed("The student has passed the
      INSERT with SELECT question.")
      & passed("The student has passed the
      INSERT with VALUE question.")
<- .send(agMaterial, achieve,
    hasPrerequisite(delete, insert)).

@d2
+!recommendMaterial[source(agSupport)] :
desired_Concept("DELETE")[source(agSupport)]
      & passed("The student has passed the
      INSERT with SELECT question.")
      & failed("The student has NOT passed
      the INSERT with VALUE question.")
<- .send(agMaterial, achieve, has_KB(insert,
    insert_value)).

@d3
+!recommendMaterial[source(agSupport)] :
desired_Concept("DELETE")[source(agSupport)]
      & failed("The student has NOT passed
      the INSERT with SELECT question.")
      & passed("The student has passed the
      INSERT with VALUE question.")
<-.send(agMaterial, achieve, has_KB(insert,
    insert_select)).

@d4
+!recommendMaterial[source(agSupport)] :
desired_Concept("DELETE")[source(agSupport)]
      & failed("The student has NOT passed
      the INSERT with SELECT question.")
      & failed("The student has NOT passed
      the INSERT with VALUE question.")
<-.send(agMaterial, achieve,
    hasPrerequisite(insert, select)).
```

The multiple classification code classifies a student for learning material into one of four categories for any given concept e.g. the DELETE. In the codes the attributes of the students which forms the production-rules (otherwise known as the *context* in Jason agentSpeak) or pre-conditions must be true and satisfied before classification can be completed. Recall that the set of parameters that is devised to construct this multiagent based Pre-assessment System is given in the turple M = <D, P, F, V>. Logically, based on the *Passed* and *Failed* two-state predicate attributes of a student, if a set of attributes are all <P> (e.g. label @d1) then we say the student has *positive ability,* but if all <F> (e.g. label @d4) we say the student has *negative ability*. But if the set of attribute is a mix of <P> and <F> (e.g. label @d2 and @d3) then we say it is *partial ability*.

3.4 Agent Learning Hypothesis

In this production rules classification learning, let $\boldsymbol{C}$ be the number of prerequisite concept(s) to a desired concept $\boldsymbol{D}$, $\boldsymbol{T}$ a binary-state value for student pre-assessment outcome and $\boldsymbol{N}$ the equal number of leaf-nodes across each parent node, then the total number of classified production rules $\boldsymbol{R}$ for a given ontology tree is determined by:

$$\boldsymbol{R = CT^N + 1}$$

where

$$\boldsymbol{C} \in \{0, 1, 2, \ldots, k^{-1}, k\}$$
$$\boldsymbol{T} = 2, \text{ for a pass or fail state}$$
$$\boldsymbol{N} \in \{1, 2, 3, \ldots, k^{-1}, k\}$$

For any SQL rules set that would need to be added to the array of production rules, the agent *agModelling* would increment the number of classified rules for a given concept with:

$$\boldsymbol{R' = R + T^N;}$$

where C = 0, 1, 2,…, k in

$$\boldsymbol{R = CT^N + 1}$$

and conversely decrements by removing rules for a concept that is no longer needed with:

$$\boldsymbol{R' = R - T^N;}$$

where C ≠ 0 in

$$\boldsymbol{R = CT^N + 1.}$$

From each learning algorithm, the number of rules to

be added or removed is determined by the number of leaf-nodes T^N in the ontology. The code snippet in Section 3.3 gives a picture of how the classified rules makes prediction for learning. Since $T^N = 2^2$ then the number of classified rules equals 4 for each class concept. In the example for the DELETE concept received, the agModelling classifies the student and make prediction for appropriate learning URL through semantic literal communication to the agent *agMaterial* using the *tell* or *achieve* performative.

4 EXPECTED OUTCOME

The main purpose of this system is to ascertain whether a student is prepared to learn a concept he so desired to study, and to identify gaps in the student knowledge. Figure 3 and 4 shows the Pre-assessment System user interface and a minimized output console. As students would enter concepts on the input interface, the MAS outputs a couple of questions of the immediate lower (or prerequisite) concept. Depending on the answers received by the MAS, the student is shown the URL link(s) of the concept to learn after his/her classification.

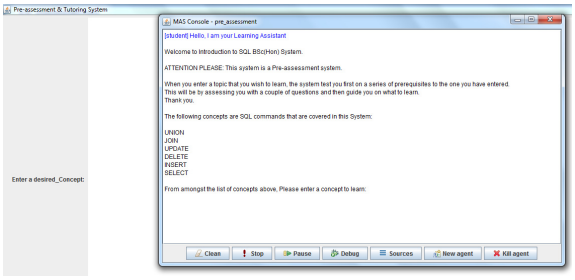

Figure 3: Starting the Pre-assessment System. The system outputs a list from which a SQL concept can be chosen to be studied.

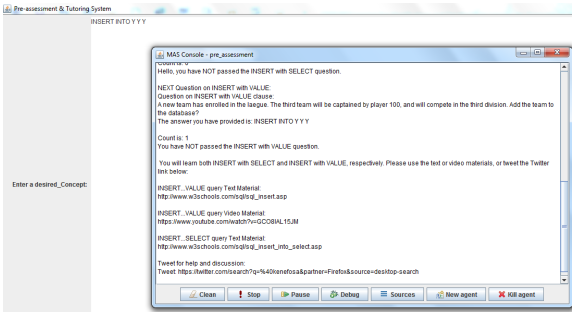

Figure 4: The user is diagnosed to have a knowledge gap in INSERT SELECT and INSERT VALUE statement. So not quite ready for DELETE. Output of INSERT URL links.

5 STAGE OF THE RESEARCH

The next stage of this work is the evaluation of the multiagent based Pre-assessment System for: fitness of purposed, evidence of use, and collation of feedback data from potential participants and analysis of the data. The collection of feedback data shall be through a structured qualitative questionnaire method, followed by the tutor's task to view the *student* Model agent BB to unveil successful construction of SQL queries by students and/or the technical difficulties they are facing which we have called the gaps between what they have learnt and what they want to learn in order to address the research question. Finally write my *Thesis* which is potentially titled: *Helper Agents for Learning in Structured Ontology Interoperability Based System.*

6 CONCLUSIONS & CONTRIBUTION

This paper has demonstrated how a multiagent tool can be used to design an intelligent learning system, and how the agents can cooperatively diagnose gap(s) between a student's desired knowledge and his previous knowledge using a devised set of parameters. As well as the foregoing, its contribution is the use predicate or description logic semantic literals that are understandable between agents for modelling classified production rules and predicting learning materials. The *Expected Outcome* is that the system should be able to pre-assess students, identify students' strengths (i.e. gains) for projection to the next module, identify weaknesses (i.e. gaps) in order to fill the gaps and prepare them for the next module; and recommend learning materials.

REFERENCES

Bordini, R. H., Hübner, J. F., & Wooldridge, M. (2007). *Programming multi-agent systems in AgentSpeak using Jason.* John Wiley & Sons, West Sussex, England.

Chakraborty, S., Roy, D., & Basu, A. (2010). Development of knowledge based intelligent tutoring system. *Advanced Knowledge Based Systems: Model, Applications & Research*, *1*, 74-100.

Kang, B., Compton, P., & Preston, P. (1995, February). Multiple classification ripple down rules: evaluation and possibilities. In *Proceedings 9th Banff knowledge*

acquisition for knowledge based systems workshop (pp. 17-1).

Matsuda, N., Cohen, W. W., Sewall, J., Lacerda, G. & Koedinger, K. R. (n.d.). SimStudent: Building an Intelligent Tutoring System by Tutoring a Synthetic Student. Pp.1-19. http://www.cs.cmu.edu/~wcohen/postscript/simstudent-authoring-submitted.pdf (Accessed: January 3rd, 2015).

Mills, C., & Dalgarno, B. (2007). A conceptual model for game-based intelligent tutoring systems. *Proceedings of the 2007 Australasian Society for Computers in Learning in Tertiary Education*, 692-702.

Padgham, L., & Winikoff, M. (2004). *Developing intelligent agent systems: A practical guide* John Wiley & Sons. West Sussex, England.

Wooldridge, M. (2009). An Introduction to MultiAgent Systems. John Wiley & Sons Ltd, UK.

View publication stats