An Improved Bayesian Learning Method for Multi-agent System

http://dx.doi.org/10.3991/ijoe.v11i9.5071

Dai Sheng-Hui, Zhu Xue-Qin, Gui Ying,Xu Hong-Zhen
East China Institute of Technology, Shanghai, China

***Abstract*—A multi-agent coordinate ion is addressed in urban traffic control, which uses the recursive modeling method (RMM) that enables an agent to select its rational act ion by examining with other agents by modeling their decision making in a distributed multi-agent environment. Bayesian learning is used in conjunction with RMM for belief update. Based on this method, a multi-agent traffic control system is established and the results rated its effective.**

***Index Terms*—Multi-agent system, Intelligent agent, Agent learning, Bayesian learning**

I. INTRODUCTION

Multi-agent systems [MAS] have demonstrated their potential for solving complex problems in various domains. One attribute of agents which lends itself to problem solving is intelligence, which refers to the level of reasoning and learned behavior exhibited by an agent [12]. Higher levels of intelligence exist when agents can adapt to their environment [10]. Such adaptation involves learning about the user's objectives and resources available to the agent in its environment. Learning is also thought of as belief revision [1,3,18]. This is the actual mechanism by which adaptation, and thus learning occurs. Traditional machine learning has developed a wide variety of algorithms for providing single-agent systems with learning capacity [22]. Among the main classes of algorithms for traditional machine learning are induction of trees and rules, learning in neural nets, system classifiers and genetic algorithms, reinforcement learning, case-based learning, logic-based learning, and some others. However, these algorithms do not apply directly when used in MAS. Learning in multi agent systems has opened new challenges and opportunities for researchers.

Research has focused on developing learning agents that can adapt to their environment in order to achieve enhanced performance [4,16,19,23]. The concept of agent learning has also been applied to develop an adaptive interface agent in MAS. The research on agent learning has primarily addressed learning a user's preference model [8,11,21,29] and typically involves a single agent. MAS research on learning has been in the area of negotiation, and learning strategies of other agents [27]. Little research has been conducted to develop agents that learn free-text queries and keyword searches in MAS.

The objective of this paper is to address this open research area by presenting a Bayesian learning approach for Web-based MAS. Bayesian learning has previously been applied to MAS for performance enhancement [4,16,19], and has been identified as one of the most successful algorithms for classification [22]. Because the MAS presented in this paper seeks to resolve incoming queries in a timely and accurate fashion by learning to classify incoming queries based on past experience, a Bayesian approach is employed. A primary distinction from past work on Bayesian learning in MAS is that a negotiation problem is not used to illustrate learning. Instead, we use a cooperative system that does not involve negotiation between agents. In addition, the Bayesian approach presented in this paper learns to identify an appropriate agent to answer free-text and natural language queries as well as keyword searches submitted by users. The efficacy of MACS is determined by analyzing the accuracy of learning in the system. This work builds on past work by Liebowitz et al. [20], by extending MACS from a multi-agent system lacking the ability to learn from and adapt to its environment, to a truly intelligent multi-agent system.

The next section describes the MACS system. Then Section 3 provides an overview of Bayesian learning in multi-agent systems. Section 4 describes how Bayesian learning is implemented in MACS, while Section 5 analyzes the effectiveness of Bayesian learning in MACS. Finally, Section 6 draws conclusions from the analysis and suggests future research directions.

II. MACS

The MACS system is a MAS developed for procurement and acquisition of defense contracts. Specifically, it is designed to assist Acquisition Request Originators [AROs] and Contracting Officer's Technical Representatives [COTRs] with the pre-award phase of contracting and procurement [20]. The system architecture consists of nine agents—a User agent, a Facilitator agent, a Natural Language agent, a Machine Learning agent, and five specialty agents. The specialty agents are encoded with domain knowledge about the five general areas of expertise required of AROs/COTRs, and the user agent interfaces with AROs/COTRs. Interaction between AROs/COTRs and the system occur through either keyword searches or natural language queries. As shown in Fig. 1, the MACS architecture implements a three-tiered brokered architecture. The Facilitator agent coordinates agent activities and communicates with the agent(s) capable of responding to an incoming query.

The User agent interacts with the user/ARO/COTR to welcome the user, ask what pre-award questions the user has, and serve as the interface between the user/ARO/COTR and the other agents in the system [20]. Two business logic threads have been designed into the User agent. One thread supports the Natural Language capabil-

ity of the system, and the other supports the keyword search capability of the system. The User agent sends incoming user queries to the Facilitator agent, which is responsible for communicating with all of the other agents in the MACS system as illustrated in Fig. 1.

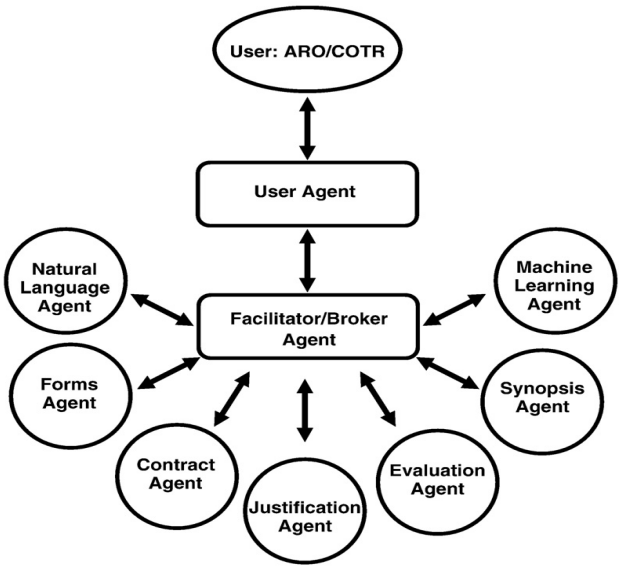

Figure 1. Agent architecture and communication channels.

Natural Language queries submitted by users are forwarded, by the Facilitator agent, to the Natural Language [NL] agent for parsing, and the parsed keywords are then forwarded to the Machine Learning [ML] agent. The detailed description of the NL parsing process is presented in Yoon et al. [30]. The user queries selected from a list of predetermined keywords are directly forwarded to the Machine Learning [ML] agent by the Facilitator. The ML agent implements Bayesian learning and creates an action plan, and that plan is then issued back to the Facilitator agent for completion. In MACS, the action plan is essentially a determination of which specialty agent(s) should be contacted to respond to an incoming query. The facilitator completes the action plan by performing the necessary low-level communication between the specialty agents. These communications lead to solutions being sent from a specialty agent (or agents) to the Facilitator agent and then from there to the User agent. As solutions to a query are collected, the ML agent updates its internal tables, making note of which agents responded to which user queries. This information is used to calculate the response plan for similar queries in the future.

The five specialty agents in the system relate to the pre-award phase of a contract and include the Forms, Justification, Evaluation, Synopsis, and Type of Contract agents. The Forms agent identifies the forms needed to complete a procurement request package. The Justification agent indicates situations where a justification and approval is required to complete a procurement request. The Evaluation agent provides guidelines for evaluating proposals. The Synopsis agent identifies the type of synopsis for a given procurement request. Lastly, the Type of Contracts agent identifies the type and nature of a contract based on conditions such as the source of contract, the nature of the work, etc.

Each specialty agent in MACS contains a rule base and has explicit goals. Its rule base describes how to achieve the goals under varying circumstances. The specialty agents respond to incoming queries by presenting necessary information and/or requirements for AROs/COTRs. For example, the Evaluation Agent can assist an ARO/COTR with information regarding how to evaluate a project and what criteria or weights to use for evaluation of a contract. If an ARO/COTR has a question regarding "determining weights on evaluation criteria," the Evaluation agent will reply with "You can develop your own weights on technical, qualifications, and cost criteria. Generally speaking, a weight of 40% (out of 100%) is given to cost." [20]. The knowledge contained within each specialty agent is independent of the knowledge contained within the other specialty agents. Thus, coordination between the specialty agents is not required for the current implementation. However, each specialty agent does coordinate with the user agent in order to answer queries. In the original system [20], the user agent broadcasts messages to all specialty agents. The learning capability that is now part of MACS allows the user agent to learn which specialty agent(s) should receive incoming messages. The user agent asks the ML agent to determine which specialty agent(s) should receive the query. The ML agent makes this determination probabilistically, by means of Bayesian learning. This is explained more fully in Section 4.

MACS has been implemented using the Open Agent Architecture [OAA], which is supported by the Artificial Intelligence Lab at the Stanford Research Institute. The Facilitator agent functions as part of MACS, but it is a specialized server agent that is part of OAA, and it performs many basic functions. The Facilitator agent has the ability to route messages, manage data, and fire registered triggers as well as accept incoming messages.

Furthermore, the rules for MACS have been encoded as XML documents. The XML encoding of the rules offers a flexible means for rule modification that was not possible in the earlier version of MACS [20]. Each specialty agent loads the XML document as its rule base. The rule base is used to compare against incoming queries to determine if a rule is true or false. The XML rule bases are simply ASCII documents that are served up by a web server. A series of web forms have been designed for modification of the rule bases and general system maintenance.

III. BAYESIAN LEARNING IN MULTI-AGENT SYSTEMS

A. Background

Bayesian probabilistic inference has been widely used to reason and draw conclusions in the presence of uncertain knowledge in intelligent systems [24]. Bayesian inference uses a natural posterior probability update of a prior probability for a hypothesis to determine how likely it is that a hypothesis will be true given certain "evidence" exhibited by the data. Graphically, a Bayesian network may be used as a probability model that is represented as a directed acyclic graph [DAG], where nodes of the DAG represent random variables and the arcs represent dependencies among variables. The dependencies may represent causal relationships, where the arrows point from the causes to the effects. Bayesian inference is commonly used in machine learning for classification and prediction problems as well as information retrieval and user profiling tasks. Moreover, Bayesian inference provides a mathematically optimal method via maximum a posteriori (MAP) estimation (see [17] for details), which we choose

to employ here. Bayesian inference also provides a computationally tractable algorithm via naïve Bayes classification (see [17]), that scales easily and operates efficiently.

Bayesian inference and, more generally, Bayesian networks, are extremely convenient and useful for automated reasoning techniques where prior opinion influences the final outcome. In one approach to Bayesian modeling, human experts are responsible for designing the network structure (or DAG) by identifying the variables and finding the dependencies among the variables. They are also responsible for subjectively estimating and calculating the conditional probabilities associated with each node. In recent years, attempts have been made to automate both the design of the network structure and the calculation of the conditional probabilities from empirical data [2,7,14]. This automated process is known as Bayesian learning. In this case, prior probabilities are subjective, while conditional probabilities are unknown and rely on empirical data for their probability estimates. Ideally, in a fully automated Bayesian network, the DAG's structure and its conditional probabilities may be "learned" (or updated) automatically as new data arrives in order to facilitate Bayesian inference for a particular node in the DAG, given the observable results from a subset of the remaining nodes. In [13], a celebrated tutorial on Bayesian learning describes a Bayesian model as "an ideal representation for combining prior knowledge (which often comes in causal form) and data" as well as one which "readily handles situations where some data entries are missing." Furthermore, in a DAG where "attribute" nodes are assumed to be conditionally independent given the value of a "target" variable , thus producing a naive Bayes model, we have both the simplest paradigm for Bayesian learning and also the most realistic, given our underlying assumptions in MACS.

B. Multi-agent systems

Essentially, Bayesian learning in MAS allows agents to use new or updated knowledge about other agents in the system to enhance performance of the system as a whole. There is a plethora of prior work on multi-agent learning from many researchers [4,16,19,23,25,31]. Much of this research has addressed learning in autonomous negotiations and learning organizational roles (i.e., group dynamics). Our work differs by looking at a problem that does not involve negotiation. Rather than learning preferences of other agents, the MACS system learns abilities of other agents in order to enhance performance and efficiency of the system. Furthermore, the MACS system applies agent learning to the domain of defense contracting. Zeng and Sycara [31] present an approach to negotiation in MAS based on the sequential decision making paradigm, where decisions are dependent on each other and the decision maker has a chance to update his knowledge based on feedback after implementing the decision at each step in the sequence.

This allows for more informed decision making as the series of decisions progresses. Their sequential decision making model, Bazaar, learns by explicitly modeling beliefs about the negotiation environment and the participating agents under a Bayesian probabilistic framework. Li and Cao [19] present a Bayesian approach to learn the incomplete information of an opponent agent to enhance the negotiation efficiency. This method is applied to bilateral multi-issue negotiation in agent-mediated e-Commerce. Ishihara, Huang, and Sim [16] use Bayesian updating rules to learn opponents' eagerness in a market driven negotiation model. Recent work by Contizer and Garera [6] present a Bayesian approach to learn the optimal reward to motivate an agent to take a certain action useful to the principle agent. Their work demonstrates that the approach effectively learns the optimal reward quickly when the prior belief is limited to a certain class of distributions. Prasad, Lesser, and Lander [25] present a different type of learning in MAS — coordination. Their work looks at how agents can change their behavior as they learn about the MAS of which they are a part. As an agent's knowledge of the system configuration changes, it adjusts its behavior to help meet the common goals of the system and enhance its overall performance. Learning occurs as each agent learns which part(s) of the common goal it can work to solve. Chalkiadakis and Boutilier [4] present a Bayesian approach to model multi-agent reinforcement learning problems that enable agents to reason about their uncertainty regarding the underlying domain and the strategies of other agents. This Bayesian approach improves online performance of reinforcement learning agents in coordination problems, when compared to heuristic exploration techniques that attempt to induce convergence to optimal equilibrium. Chalkiadakis and Boutilier [5] also introduce a Bayesian "reinforcement learning" model for coalition formation under uncertainty. This model enables coalition participants to adjust their uncertainty regarding the value of potential coalitions and the capabilities of others. A formal model of implicit imitation for reinforcement learning introduced by Price and Boutilier [26] also uses the Bayesian method. This model learns useful behaviors by making intelligent use of the knowledge implicit in the mentors' behaviors. Sahin [28] also presents a Bayesian network approach for self-organization in a multi-agent system.

IV. LEARNING IN MACS

In this section, we describe how learning is undertaken in MACS. Bayesian learning is applied in the ML agent so that it can learn which specialty agents should receive incoming messages. In particular, we use a naïve Bayes classifier to allow the "user agent" in MACS to redirect each input query to its appropriate "specialty agent," given our available "ground truth" in the form of actual human-verified queries with their "correctly judged" specialty agents as respondents. Our basic idea then is to establish patterns among input queries as they are being submitted in order to accurately predict (with high probability) which specialty agent should most likely respond to a new (and possibly different) query. Our approach is similar to that of Heckerman and Horvitz [15] where user goals are inferred from user queries using a naïve Bayes classifier. However, their approach [15] allows for free-text queries to a database, whereas MACS is designed for free-text and natural language queries as well as keyword searches in a multi agent system. Our approach is also flexible enough to "score" (in the sense of MAP estimates) new and unseen queries because it relies on our assumption of statistical independence among the specific keywords that are selected from each of five (set) menus associated with each of the five specialty agents. This independent assumption is the key to a computationally feasible and conveniently realistic Bayesian model.

Essentially, the user agent employs a Bayesian model to identify which of five specialty agents, a_i, i=1, 2, 3, 4, 5, is most likely to respond correctly (as judged from ground truth) to a query, q, given the evidence, e_j, appearing in the query q, where q=∩ e_j, j=1, 2, 3, 4, 5, and e_j refers to the specific keyword taken from the jth menu, where menu items are fixed and depend on the particular specialty agent. So, in MACS we may take each incoming query, $q_{incoming}$, to be the Boolean "and" of five specific keywords that may be found on each of the five different menus. Our Bayesian learning procedure simply updates the probability, $P(a_i)$, for each of the five specialty agents, as $P(a_i |q_{incoming})$, in light of the "evidence" provided by the incoming query, i.e., $q_{incoming}$=∩ e_j, j=1, 2, 3, 4, 5, conditional on the ith agent a_i (note that $P(a_i)$=0.2 initially for all i). We may then summarize our resultant action in terms of Bayes rule by stating that the largest posterior probability, maxi $P(a_i |q_{incoming})$, i=1, 2, 3, 4, 5 (after we have observed the specific query, $q_{incoming}$), is our maximum a posteriori (MAP) probability estimate that determines to which of the five specialty agents the incoming query, $q_{incoming}$, should most likely be directed (see [9] for a discussion of MAPs). Our use of MAP estimation as a Bayesian classification method is indeed mathematically optimal in that it achieves the minimum error rate for all possible classifiers (refer to [17, p. 207] and [22, p. 174] for proofs of this fact). We further notice that computing and updating MAPs is quite easy via direct application of Bayes Rule since maximization of the posterior-to-prior probability ratio, given by $P(a_i |q_{incoming})$ / $P(a_i)$, is mathematically equivalent to maximization of the likelihood function, given by $P(q_{incoming} |ai)$, i=1, 2, 3, 4, 5. Under our assumption of statistical independence of the evidence as represented by keywords, e_j, where $q_{incoming}$=∩ e_j, j=1, 2, 3, 4, 5, conditional on the ith agent a_i, we may conveniently write the likelihood function as a product of conditional probability estimates of P(ej |ai) for fixed I and for each specific ej. To obtain these individual probability estimates, we initially "train" our Bayesian learning model by relying on ground truth queries that have been associated with their "correct" specialty agents. We then compute the probability estimates for each keyword used on the jth menu by simply keeping track of the number of times each particular keyword was used in an incoming query, which was known to be associated with one of the five specialty agents. In this manner we compute and update likelihood "scores" for each of the five specialty agents as users input their queries.

Since a menu item is either present or absent in a particular incoming query, we may represent each query as an "exploded" binary vector, where ones in this vector do the necessary bookkeeping for us and also allow us to easily estimate and update the probabilities of keywords used in each specialty agent's menu. Furthermore, since the number of menu items (keyword choices) is fixed, we may estimate the probabilities of unseen keywords (i.e., those keywords not chosen by users) from the jth menu in a particular query as simply $1 / [n_j(m+1)]$, where n_j, j=1, 2, 3, 4, 5, is the total number of menu items for the jth menu in MACS and m is the total number of user input queries. This type of smoothing of probability estimates [17, p. 219] facilitates our "scoring" of incoming queries consisting of one or more previously unseen menu items and also prevents any score from ever being equal to zero. Unless certain menu items are very rarely used, we will generally have non-uniform probability estimates for each menu item and will update them for each new incoming query, as our Bayesian learning paradigm guarantees.

We present our Bayesian learning algorithm in the following steps:

1. 1llow user to enter a query, $q_{incoming}$.
2. Employ Bayesian reasoning to determine which a_i should receive the user's query.
3. For each a_i (i=evaluation, synopsis, justification, forms, type of contracts):
   a. CALCULATE the percentage of time each keyword appears in all $q_{existing}$ (e.g., evaluation criteria appears in 80% of existing queries sent to an evaluation).
   b. CALCULATE the probability P (evidence $|a_i$) by multiplying all percentages calculated in the immediately preceding step that correspond to $q_{incoming}$. P (evidence $|a_i$) represents the likelihood that a query actually corresponds to the domain knowledge of that a_i. This is a causal relationship from the cause ai to the effect $q_{incoming}$.
   c. EMPLOY the Bayesian formula by: (i) multiplying the prior probability, $P(a_i)$, by $P(evidence|a_i)$, (ii) obtain the sum of products found in (i), and (iii) divide each individual product computed in (i) by the sum in (ii) to normalize results. This will yield the probability P (a_i | evidence). The prior probability, $P(a_i)$, represents the probability that $q_{incoming}$ should be sent to a particular a_i given no evidence. Initially, $P(a_i)$=0.2 for all i. Then P(a_i | evidence) is the posterior probability of a_i given the keyword evidence. This probability assesses the likelihood that a specialty agent will address $q_{incoming}$ based on the evidence provided by $q_{incoming}$ itself.
   d. DIVIDE the probability P(a_i | evidence) by $P(a_i)$ to obtain the likelihood scores for each i.
4. After completing the calculations for each a_i, do the following:
   a. IDENTIFY which result in 3(d) has the largest value.
   b. SEND $q_{incoming}$ to the corresponding a_i.
   c. ADD $q_{incoming}$ to the list of $q_{existing}$, given the corresponding a_i, for updating purposes.

The learning model is displayed in Fig. 2. This model is a simple Bayesian network in that it shows a causal relationship between the specialty agents and the evidence in the query. a_i is a random variable with a probability distribution over the five specialty agents, assumed to be a uniform distribution a priori. Evidence is a random variable gathered from $q_{incoming}$, with a probability distribution over the attributes associated with each specialty agent, where attributes are assumed to be independent. Evidence consists of keywords that are extracted from a user's natural language query and used to probabilistically determine which specialty agent(s) should most likely receive $q_{incoming}$. An example of how Bayesian learning is implemented in MACS is provided in subsequent paragraphs. The prior probability for each specialty agent represents the likelihood that particular agent will respond to an incoming query. The prior probabilities are adapted over a series of runs, and the values displayed below are from a point in time prior to the sample run used in this example. In this example, the Contracts agent has the greatest probability, when compared to the other specialty agents, of respond-

ing to a new query. This indicates that a majority of the incoming queries contain terms that are used by the Contracts agent. It is important to note that all of the prior probabilities sum to one; the probability displayed as 1.0 for the Contracts agent is rounded up and is not actually 1.0.

Agent	Prior probabilities
Forms	2.5822858689540867 E−242
Contracts	1.0
Justification	1.0240324973261967 E−166
Evaluation	1.9181461523891187 E−214
Synopsis	3.6206243551615097 E−190

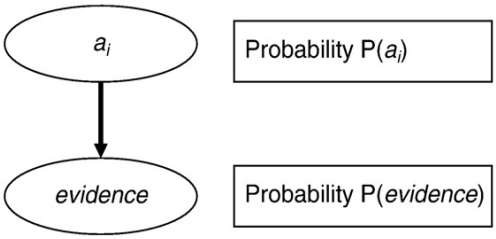

Figure 2. Bayesian network for the user agent.

The example run includes the following four keyword terms:

1. Not on GSA Schedule
2. Cost plus fixed fee
3. Synopsis format
4. Sole source (non-competitive) procurement

The terms are sent to the Machine Learning agent and the Bayesian formula is applied. The learning procedure is detailed below.

Agent	Probability (evidence/a_i)
Forms	1.0000000000000004E−20
Contracts	3.698224852071007E−14
Justification	1.0526315789473686E−16
Evaluation	1.0000000000000004E−20
Synopsis	1.5306122448979592E−12

The probability(evidence/a_i) values for each specialty agent are then multiplied by their respective prior probabilities and summed. This value is used in a subsequent calculation.

(2.5822858689540867E-242*1:0000000000000004E-20)
+(1.0*3:698224852071007E -14)
+(1.0240324973261967E-166*1.0526315789473686E-16)
+(1.9181461523891187E-214*1.0000000000000004E-20)
+(3.6206243551615097E-190*1.5306122448979592E-12)
=3.698224852071007E -14

The next step is to calculate the updated prior probability values, as well as the probability P(a_i/ evidence). These values are used to determine the order in which the agents will be contacted to answer the incoming query. The calculations used for the Forms agent are included here. The same calculations are made for each agent (Table 1).

TABLE I.
CALCULATION RESULT FOR EACH AGENT

Agent	Posterior *P*	Score
Forms	6.98250098965185E−249	2.704E−7
Contracts	1.0	1.0
Justification	2.914719866073721E−169	0.0028463157894736833
Evaluation	5.186667196060177E−221	2.704E−7
Synopsis	1.4984951412790896E−188	41.3877551020408

Posterior probability:
1.0000000000000004E -20*2.5822858689540867E - 242 /3.698224852071007E -14 =6:98250098965185E- 249
Score:
6.98250098965185E -249/2.5822858689540867E -242 =2.704E- 7

As is indicated by the Score values, the Machine Learning agent has determined that the Synopsis agent is the most likely agent to have a response for the incoming query. In fact, the Synopsis agent is the correct agent to respond in this example. The posterior probability values are then used to update the prior probability values for each specialty agent for the next iteration of MACS.

V. ANALYSIS OF LEARNING IN MACS

C. Evaluation method

In order to evaluate the performance of the ML agent, it is tested against twenty-six historical queries. This is accomplished by having the User agent send incoming queries to all specialty agents in addition to the ML agent (i.e., broadcast). Because all specialty agents receive the queries when broadcast, all correct responses are collected. The results from the broadcast query are then used as the basis for evaluating the results from the ML agent in order to determine how well the ML agent is performing (i.e., whether the ML agent identifies the correct agent to receive the incoming query). The level of effort required of the ML agent in order to obtain a correct response from keyword searches in MACS is the focus of this paper. The evaluation method is summarized as follows:

1. A user enters a query, $q_{incoming}$.
2. The query is broadcast by the User agent to all specialty agents.
3. All correct results are enumerated and collected.
4. The User agent sends the original $q_{incoming}$ to the ML agent.
5. Data on where the ML agent suggests routing $q_{incoming}$ are collected.
6. Results from the specialty agents contacted via machine learning are collected.
7. Results from the machine learning response are measured against all correct results per the broadcast query.

Results are analyzed using descriptive statistics. First is the accuracy of the ML agent. Here, the specialty agent to which the ML agent sends a query is compared to which specialty agent(s) replied to that same query when the query was broadcast to all. This statistic targets accuracy of the system, but not learning. Learning is handled by the second statistic computed.

The second set of data collected is the number of specialty agents to which the ML agent sent a query before reaching the correct agent. The term correct used here refers to the agent with the domain expertise to answer a user's question. This statistic targets the ability of the system to learn. Learning is evaluated over time using this statistic, where messages should be sent to fewer and fewer specialty agents before reaching the correct one over time. That is, the Bayesian probabilities are updated after every new query so that over time the ML agent eventually "learns" which specialty agents can answer which types of questions.

D. Results

1) Accuracy

Accuracy of MACS is 74.31%. That is, 74.31% of the time, an incoming query was sent to the correct specialty agent and fully resolved. The accuracy rate of 74.31% represents only those cases where the query was fully resolved by a single specialty agent's response. In many cases, multiple specialty agents may have to respond to a query in order to resolve it fully, where each specialty agent provides a different area of domain expertise. 22.94% of the time there was a second specialty agent that should also have received the query, 1.83% of the time there were two additional specialty agents that should also have received the query and 0.92% of the time there were three additional specialty agents that should have also received the query.

The current design of MACS allows the ML agent to designate only one specialty agent to receive an incoming query. Thus, the system design will need to be changed before the accuracy rate can be improved above 74.31%. However, it should also be noted that Bayesian learning identified a correct specialty agent in every case. That is, there may have been multiple specialty agents necessary to fully resolve the query, but at least one of those was identified in every case. Thus, queries were not sent to specialty agents with no expertise relevant for the incoming query.

2) Learning

The data on how many attempts the ML agent made before reaching the correct agent to which queries should be sent suggest the MACS system is in fact "learning." Fig. 3 illustrates MACS learning over time. As the trend line shows, over time, fewer tries were needed before the correct specialty agent was identified, indicating the ML agent sufficiently taught the appropriate specialty agents to which incoming queries should be sent. The y-axis identifies the number of tries the ML agent made before learning the correct agent.

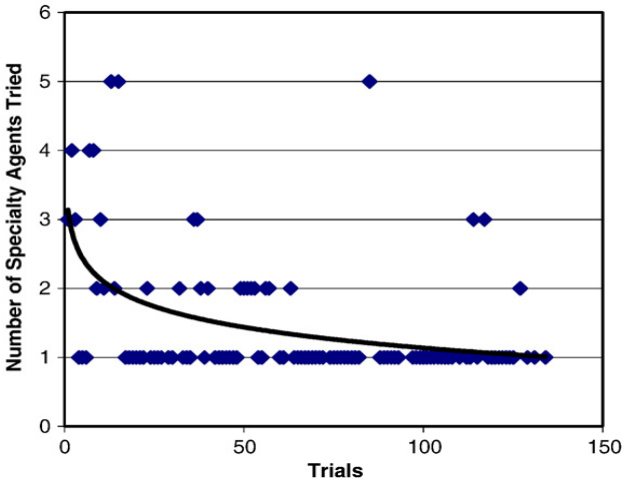

Figure 3. Learning in MACS over time.

VI. CONCLUSIONS AND FUTURE DIRECTIONS

The multi-agent system, MACS, presented in this paper illustrates how Bayesian learning can be built into MAS. A real-world problem in the area of defense contracting is used to illustrate the usefulness of the system design and learning techniques built into the system. MACS has been extended beyond the system presented by the authors in previous research in that it now possesses the ability to learn from and adapt to its environment. Furthermore, the system has been migrated to OAA and uses XML coding.

There is also a natural language agent that is in the process of being developed and expanded. The system has been tested against known, historical data in order to show definitively that it is learning. The data presented in Section 4 support the authors' claims that Bayesian learning is a meaningful approach, and that learning is in fact occurring in the system. However, the authors believe the results would be far more dramatic in a larger system where there are many more specialty agents that may need to respond to queries. In such systems, performance enhancements from Bayesian learning will be more pronounced because resources allocated to agent communication increase as the system size increases. Clearly, as the number of agents increases, the value of targeted brokering versus communicating with all agents in the system increases.

While the research presented in this paper contributes to the existing MAS literature by building learning into MAS, it also lays a foundation upon which future work can build. The primary direction for future work is in the area of more comprehensive learning than what is currently achieved by the MACS system. This will occur in two dimensions. First, future research will look at MAS in which the specialty agents cooperate with each other to more completely answer user queries. This is in contrast with the existing system where specialty agents communicate with a user agent, but not with each other. Second, reinforcement learning will be built into the system so that MACS can learn from user feedback in order to learn new keywords.

REFERENCES

[1] A. Baltag, M. Sadrzadeh, The algebra of multi-agent dynamic belief revision, LCMAS 2008 Preliminary Version, http://www.ecs.soton.ac.uk/~ms6/Brev-LCMAS05.pdf.

[2] W.L. Buntine, Operations for learning with graphical models, Journal of Artificial Intelligence Research 2 (1994) 159–225.

[3] T.R. Burns, A. Gomolinska, Socio-cognitive mechanisms ofbelief change—applications of generalized game theory to belief revision, social fabrication, and self-fulfilling prophesy, Cognitice Systems Research 2 (1) (2001) 39–54. http://dx.doi.org/10.1016/S1389-0417(01)00014-6

[4] G. Chalkiadakis, C. Boutilier, Coordination in multi-agent reinforcement learning: a Bayesian approach, Proceedings of Autonomous Agent and Multi-agent Systems, Melbourne, Australia, July 14–18 2008.

[5] G. Chalkiadakis, C. Boutilier, Bayesian reinforcement learning for coalition formation under uncertainty, Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS-04), New York, NY, 2004, pp. 1090–1097.

[6] V. Contizer, N. Garera, Learning algorithms for on-line principle agent problem (and selling goods online), Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, 2008.

[7] G.F. Cooper, E. Herskovitz, A Bayesian method for the induction of probabilistic networks from data, Machine Learning 9 (1992) 309–349. http://dx.doi.org/10.1007/BF00994110

[8] E. Crawford, M. Veloso, Learning dynamic time preferences in multi-agent meeting scheduling, Technical Report, CMU-CS-05-153 (2008).

[9] O.R. Duda, P.E. Hart, G.D. Stork, Pattern Classification, 2nd edition, Wiley Interscience, Inc., New York, NY, 2008.

[10] G. Elofson, M. Beranek, P. Thomas, An intelligent agent community approach to knowledge sharing, Decision Support Systems 20 (1) (1997) 83–97. http://dx.doi.org/10.1016/S0167-9236(96)00077-2

[11] M. Fleming, R. Cohen, User modeling in the design of interactive interface agents, Proceedings of the Seventh International Conference on User Modeling, 1999, pp. 21–32. http://dx.doi.org/10.1007/978-3-7091-2490-1_7

[12] D. Gilbert, M. Aparicio, B. Atkinson, S. Brady, J. Ciccarino, B. Grosof, P. O'Connor, D. Osisek, S. Pritko, R. Spagna, L.Wilson, IBM Intelligent Agent Strategy, IBM Corporation, 1995.

[13] D. Heckerman, A tutorial on learning with Bayesian networks, Technical Report MSR-TR-95-06, Microsoft Research Corporation, March 1995 (Revised November 1996).

[14] D. Heckerman, D. Geiger, D.M. Chickering, Learning Bayesian networks: the combination of knowledge and statistical data, Mach. Learn. 20 (1995) 197–243. http://dx.doi.org/10.1007/BF00994016

[15] D. Heckerman, E. Horvitz, Inferring informational goals from text-free queries: a Bayesian approach, Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, 1998, pp. 230–237.

[16] Y. Ishihara, R. Huang, K.M. Sim, Learning opponent's eagerness with Bayesian updating rule in a market-driven negotiation model, Proceedings of the 19th International Conference on Advanced Information Networking and Application, March 28–30, Taipei, Taiwan, 2005, pp. 903–908.

[17] D. Larose, Data Mining: Methods and Models, Wiley Inter science, Inc., New York, 2006, 204–236 pp.

[18] R. Lau, Context-sensitive text mining and belief revision for intelligent information retrieval on the web, Web Intelligent Agent Systems 1 (3–4) (2003) 151–172.

[19] J. Li, Y. Cao, Bayesian learning in bilateral multi-issue negotiation and its application in MAS-based electronic commerce, Proceedings of IEEE/WIC/ACM International Conference on Intelligent Agent Technology, 2004, pp. 437–440.

[20] J. Liebowitz, B. Rubenstein-Montano, M. Adya, V. Yoon, J. Buchwalter, M. Imhoff, MACS: multi-agent COTR system for defense contracting, Knowledge-Based Systems Journal 13 (5)(2000) 241–250. http://dx.doi.org/10.1016/S0950-7051(00)00084-8

[21] V.K. Mavromichalis, G. Vouros, Building intelligent collaborative interface agents with the ICAGENT development framework, Auton. Agents Multi-agent Systems 13 (2) (2006) 155–195. http://dx.doi.org/10.1007/s10458-006-6212-9

[22] T. Mitchell, Machine Learning, McGraw-Hill, New York, 1997.

[23] L. Panait, S. Luk, Collaborative multi-agent learning: the state of the art, Auton. Agents Multi-agent Systems 11 (3) (2005) 387–434. http://dx.doi.org/10.1007/s10458-005-2631-2

[24] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, San Mateo, CA, 1988.

[25] M.V.N. Prasad, V.R. Lesser, S.E. Lander, Learning organizational roles in a heterogeneous multi-agent system, International Journal of Human-Computer Studies 48 (1) (1998) 51–67. http://dx.doi.org/10.1006/ijhc.1997.0160

[26] B. Price, C. Boutilier, Accelerating reinforcement learning through implicit imitation, Journal of Artificial Intellegence Research 19 (2003) 569–629.

[27] Z. Ren, C.J. Anumba, U.U. Ogwu, Multi-agent system for construction claims negotiation, Journal of Computing in Civil Engineering 17 (3) (2003) 180–188. http://dx.doi.org/10.1061/(ASCE)0887-3801(2003)17:3(180)

[28] F. Sahin, A Bayesian network approach to the self-organization and learning in intelligent agents, Thesis, Virginia Polytechnic and State University, August (2000).

[29] D.S. Weld, C. Anderson, P. Domingos, O. Etzioni, K. Gajos, T. Lau, S. Wolfman, Automatically Personalizing User Interface, IJCAI, 2003.

[30] V. Yoon, T. Wilson, S. Lowry, B. Rubenstein-Montano, J. Liebowitz, Natural language interface for multi-agent COTR

AUTHORS

Dai Shenghui is with the Department of Computer Science and Technology, East China University of Technology, China, 344000 (e-mail: shhdai@ecit.cn)

Zhu Xueqin is with the Department of Computer Science and Technology, East China University of Technology, China, 344000 (e-mail: zhuxq@ecit.cn)

Gui Ying is with the Department of Computer Science and Technology, East China University of Technology, China, 344000 (e-mail: ygui@ecit.cn)

Xu Hongzhen is with the Department of Computer Science and Technology, East China University of Technology, Chin, 344000a (e-mail: xuhz@ecit.cn)

The paper is supported by the National Natural Science Foundation of China under Grant No.61262001 and the Program of Training Young Scientists of Jiangxi Province under Grant No.20142BCB23017, and the Invention Patent Industrialization Technology Demonstration Project of Jiangxi Province of China under Grant No. 20143BBM26115, and the Science & Technology Support Program of Nanchang City of China under Grant No.2014HZZC010. Submitted 21 September 2015. Published as resubmitted by the authors 20 October 2015.