Evolutionary Agents for Intelligent Transport Systems

Jong Yih Kuo, Shin Jie Lee, Chia Ling Wu , Nien Lin Hsueh and Jonathan Lee

Abstract

When agents are initially created, they have little knowledge and experience with relatively low capability. They strive to adapt themselves to the changing environment. It is an advantage if they have the ability to learn and evolve. This paper addresses evolution of intelligent agents in transport information system. Fuzzy theory and ontological reasoning approach are proposed as evolution mechanisms, and fuzzy soft goal is introduced to facilitate the evolution process. Genetic programming operators are employed to restructure agents in the proposed multi-agent evolution cycle. We have also built an agent system to demonstrate our approach.

Keywords: Intelligent Agent, Evolutionary, Intelligent Transportation Systems.

1. Introduction

Due to the growing number of cities and transportation networks, traffic congestion, accidents, transportation delays, and larger vehicle pollution emissions happens on a daily basis. To reduce these transportation problems, Intelligent Transportation System (ITS) is introduced. The purpose of ITS is to make transportation system more safe, comfortable, rapid, efficient and environmental friendly. ITS applies artificial intelligence, information, electronics, control, and communication technology to facilitate the traditional transportation engineering. In the ITS framework, components must have the abilities to understand, coordinate, and cooperate with each other sufficiently in a distributed environment. To manage such a complex software system in distributed environments, the research on Multi-Agent System (MAS) has invoked an increasing interest. The agent-based models are ideal in dealing with entities that are geographically and functionally distributed.

The intelligent agent acts on behalf of customers to carry out delegated tasks automatically. They have demonstrated tremendous potential in conducting various communication or coordination activities, such as route guidance, travel recommendation, traffic information provision, vehicles assistance control, and other similar activities [5][22]. In order to solve a problem, an agent has to have certain skills and the ability to rationalize with these skills [6]. However, when agents are initially created, they have little knowledge and experience with relatively low capability. They should strive to adapt their negotiation strategies and tactics to the changing environment [10]. It is also helpful if agents have the ability to learn and evolve. Many issues are essential in agent evolution. First of all, evolution of an agent is closely related with the agent structure. Therefore, a suitable agent structure is one of the several basic concerns in agent evolution. Second, agents should have their own mechanisms in advance evolution. Thirdly, in a multi-agent system, evolution of individual agents is also related with many social concerns, such as coordination, negotiation, and communication. Finally, there are tools that can be utilized to evaluate the fitness of agents in the evolution procedures.

In this paper, we address the multi-agent evolution for ITS network. Section 2 gives the literature review, while Section 3 summarizes our evolutionary agent model based on fuzzy theory and ontological reasoning. In Section 4, we propose our evolution mechanism. As in Section 5, we present an agent system prototype demonstrating our approach. Section 6 contains our conclusion.

2. Related Work

Various works in a number of fields have made their marks on our evolutionary agent system for ITS, including multi-agent system for ITS and agent evolution mechanism.

2.1. Multi-Agent framework for ITS

Nowadays, many researchers have attempted to apply MAS techniques to the ITS domain. This section reviews a diverse range of applications where multi-agent systems promise to create some great impact in ITS domain according to the difference of each research subjects and methods.

To produce the cooperative works of agents, Kohata et. al [8] proposed Soft DNA (Soft computing oriented

Corresponding Author: Jong Yih Kuo is with Department of Computer Science and Information Engineering, Fu Jen Catholic University, HsinChuang, 242, Taipei, Taiwan.
E-mail: jykuo@csie.fju.edu.tw.

Data driven fuNctional scheduling Architecture) to dynamically change agents' control blocks based on Fuzzy Associative Memory Organizing Units System (FAMOUS) and Conceptual Fuzzy Set (CFS). It is a function that allots each agent's role in the agent group and selects the role dynamically according to circumstances. InforMirror [7] is an ITS application framework that provides agent-based information assistance over the Internet to drivers through a car navigation system. Rosstti et al. [17] intended to improve the use of existing road capacity by influencing driver behavior. They also provided a multi-agent based extension to an existing microscopic simulation model. Wahle et al. [20] provided an anticipatory traffic forecast model, proposing the reasoning and reaction of the drivers have to be included in the agent model as well. The traffic messages are based on future predications which are in turn affected by the driver's reactions to the traffic messages received.

Ferreira et al. proposed a decentralized multi-agent strategy to control an urban traffic network [4]. Each agent is responsible for managing the signals of an intersection, and optimizing an index based on its local state, sensors, and "opinions" coming from adjacent agents. The comparison with a fix-time optimal signal control and a local decentralized adaptive controller both showed better results. Concurrently, Roozemond analyzed the feasibility of intelligent agents in urban traffic control (UTC) and proposed a UTC based on MAS [15]. The UTC model stemmed from the combination of several intersection controlled by Intelligent Traffic Signaling Agents (ITSA), authority agents and Road Segment Agents (RSA). The ITSA made decisions on how to control the assigned intersection based on its goals, capability, knowledge, perception and data. The task of authority agents was to control and coordinate the ITSA in order to ensure the optimization of the entire transportation system, whereas RSA helped other agents to control intersection signals. Park et al. brought forward an architecture-centric method for developing MAS [14]. This approach focused on agents' coordination and autonomy. They applied architectural styles and patterns to generate the overall design of MAS. They also developed a transportation information prototype to validate their approach.

2.2. Agent Evolution

During the past several years, a great deal of attention has been dedicated to the study of evolving systems consisting of agent population adapting their behavior to the environment through evolution and/or through learning. Evolution occurs at the scale of the entire population and involves genetic mechanisms that act over successive generations [3]. Genetic Algorithms are a class of optimization technique capable of finding solutions to hard non-linear problems. A selective reproduction bias, when iterated, results in fitter structures that replace less fit structures in the population, namely survival of the fittest. The reproduction process produces offspring that are slightly altered from their parents. This allows for the exploration of new and hopefully improved structures, while still passing on the necessary information to succeed.

In [13], Maskell et al. proposed an approach to evolve agent behaviors according to their environment. By evolving control code for agents with a genetic algorithm (GA), it promises agents capable to adapt their behavior to local network conditions, and hence re-program themselves as the circumstance changes. They applied GA to focus on finding good behavioral sequences, instead of coding the individual behaviors themselves. In [3], Cristea et al. presented preliminary results in exploring the Evolutionary Intelligent Agents (EIA) concept. EIA is a cognitive and genetically controlled version of ants and bees, with links to both artificial intelligence and neural networks. The evolution of the population is task-dependent and favors the predominance of the individuals best fitted for the problem at hand, while preserving genetic variety. EIA is provided with a genotype that controls their capabilities to carry out various tasks.

In [19], Tanaka et al. proposed an approach to the evolving mobile agents based on dynamic objects capable of being attached, interpreted, carried, exchanged, and structured by other agents. A mobile agent can evolve itself with dynamic objects by repeating a sequence of operations. If a mobile agent associates with a dynamic object, it serializes and carries this dynamic object and all of its children during any subsequent network migration. The dynamic object can assist the mobile agents in itinerary rescheduling, computation check-pointing, per-site load balancing, and server functionality updating. In [2], Bonarini presents two approaches based on evolutionary and reinforcement learning algorithms. The approach allows evolution of behavior modules in real-time fuzzy models that actually control behaviors. A behavior module is in charge of implementing an agent behavior in order to perform a task. Moreover, they have developed different approaches to learn the fuzzy rules which composed the behavior modules, the interaction of behavior modules within other agents, and the activation conditions of behaviors in a dynamic environment.

On the other hand, an agent-based evolutionary approach is proposed to extract interpretable rule-based

knowledge in [21]. In the multi-agent system, each fuzzy agent autonomously determines its own fuzzy information, such as the number and distribution of the fuzzy sets. The fuzzy agents can cooperate with each other to exchange their fuzzy information and generate offspring agents. The parent agents and their offspring compete with each other through the arbitrator agent based on the criteria associated with accuracy and interpretability, allowing them to remain competitive enough to move into the next population.

3. Evolutionary Agent Model

This paper addresses the evolution of intelligent agents and their mental skills. Obviously, there is no limit to be included under so-called "mental skills". We agree that BDI model [16] provides a simple but powerful formalism for the representation, that is the specification and the analysis of the mental attributes of intelligent agent includes belief, desire and intention.

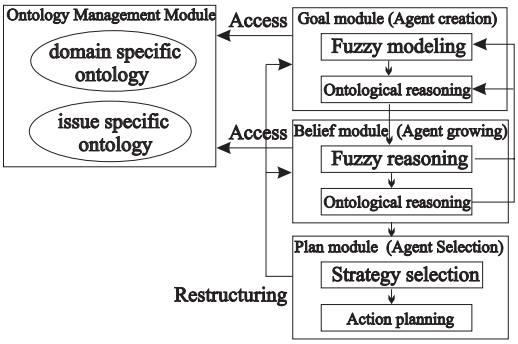


Figure1 Agent Model

3.1. Agent Architecture

In our model, an agent can be completely specified by the events that it perceives, the actions it performs, the beliefs it holds, the goals it adopts, and the plans to fulfill its intentions 0[10]. Figure 1 represents the relationships of agent components.

a) Ontology management module: There are two types of ontology which provide the domain- and issue-specific knowledge [9] respectively. All domain-specific concepts are defined in domain specific ontology. In terms of ITS, the domain-specific ontology means tour information that the user requires. In this case is namely the information for hotel accommodation. In Figure 2, OWL is used to annotate its ontology because hotel semantics varies in different domains. From Figure 2, we know that the hotel information contains name, room type, price, address, and phone number. The issue-specific ontology describes the elastic constraints for the soft goal or the non-functional goal. In this example, the quality of hotel is shown in Figure 3.

```
<?xml version="1.0" encoding="big5"?>
<rdf:RDF xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs ="http://www.w3.org/2000/01/rdf-schema#"
xmlns:xsd = "http://www.w3.org/2001/XMLSchema#"
xmlns:owl="http://www.w3.org/2002/07/owl#">
<owl:ObjectProperty rdf:ID="hasHotel">
      <rdfs:range rdf:resource="#Hotel"/>
</owl:ObjectProperty>
<owl:DatatypeProperty rdf:ID="name">
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdfs:domain rdf:resource="#Hotel"/>
      <rdfs:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="roomType">
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdfs:domain rdf:resource="#Hotel"/>
      <rdfs:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="price">
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdfs:domain rdf:resource="#Hotel"/>
      <rdfs:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="address">
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdfs:domain rdf:resource="#Hotel"/>
      <rdfs:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="phone">
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <rdfs:domain rdf:resource="#Hotel"/>
      <rdfs:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
</owl:DatatypeProperty>
```

Figure 2 the ontology of the hotel via OWL

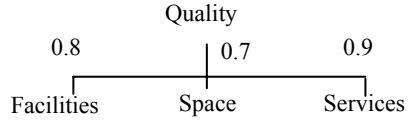


Figure 3 ontology of the quality of hotel

b) Goal module: A goal module describes the goals that an agent may possibly adopt, and the events to which it can respond. It consists of a goal set which specifies the goal, the event domain, and one or more goal states. Goal states, or sets of ground goals, are used to specify an agent's initial mental state. Soft and rigid goals are specified by the users. Fuzzy logic is used to represent the goals [11]. Based on domain- and issue-specific ontology, we propose a goal structure to analyze the user's requirement, and in turn compose the goal hierarchy.

c) Belief module: A belief module describes the information about the environment and internal state that an agent of a certain class may hold, and the strategies or tactics it may perform. According to the environmental information and the goal hierarchy of the goal module, we can construct the belief module by defining some facts and fuzzy rules. Fuzzy rules can constrain the usage of strategies of the plan module, whereas facts or reasoning consequences will then refine the goal module.

d) Plan module: A plan module describes the plans that an agent may possibly employ to achieve its goals. A plan is a sequence of actions or strategies derived through reasoning mechanism. A strategy is the combination of tactics with various weights. By using the goal hierarchy of the goal module, and the fuzzy rules of the belief module, the intelligent agent can then plan some useful strategies. These strategies constitute

a series of active actions which will attempt to satisfy the goals listed in the goal module. If successful or failed results are returned, these messages will be passed on to the belief module. The belief module utilizes that feedback to modify the related fuzzy rules.

3.2. Goal-driven Analysis

To model the goal module, we use GDUC (goal-driven use case) approach [11][12] to structure the goal hierarchy and to analyze the plans or strategies devised to achieve these goals. The steps are described below.

a) Identify actors and user's goals to construct belief module: First, we must analyze the organization or the e-commerce environment to extract basic knowledge for the agent. Then this knowledge can be built into a general common ontology. We also identify the users and their preferences to build corresponding user-defined ontology. This ontology hierarchy can be stored into the ontology management system.

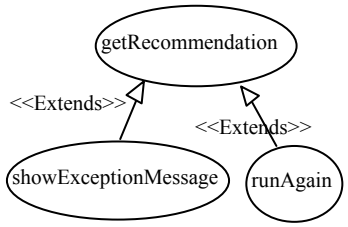
b) Analyze goal hierarchy to build the goal module: A multi-faceted classification is proposed for identifying goals from domain descriptions and system requirements. Each goal can be classified under four aspects: competence, view, content, and constraints. The competence aspect is related to whether a goal has to be completely satisfied or only to a degree. A rigid goal describes a minimum requirement from the user, which demands utter satisfaction. A soft goal describes a desirable property for the user, and can be satisfied to a degree. The view aspect is concerned with whether a goal is user-specific or agent-specific. User-specific goals are objectives of the user; whereas agent-specific goals are requirements on services provided by the agent system. The content illustrates whether the requirements represented by this goal are functional or non-functional. A functional goal can be achieved, ceased, or impaired while performing actions related to the functional requirements. As for non-functional requirements, it refers to goals that the target system needs to fulfill, such as optimization or maintenance. Lastly, constraints represent the pre-/post-condition that must be completed before and after the achievement of a goal. We use the "use case" to structure goal hierarchy. In this paper, we address the first two aspects.

c) Analyze goal module to build a matching plan module: According to the user's goal and use cases, we can construct the use case scenario and the possible plans to achieve the goals. Then we evaluate the degrees of satisfaction achievable by the plans. The ability of context sensitivity and evolution help agents to correctly adapt to the negotiation strategies to achieve user's goals.

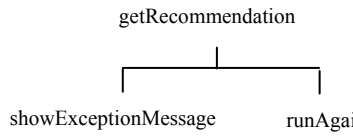
For our ITS example, the"getHotelRecommendation" use case is show in Table 1. This "use case" has two alternative courses, meaning that two other use cases extend the original use case as shown in Figure 4(a). Therefore, we can construct the goal hierarchy shown in Figure 4(b). The stable kernel system must focus on the "getRecommendation" goal.

Table 1 "getHotelRecommendation" use case

Primary Actor	USER
Description	The user inputs preference
Preconditions	The user inputs his/her preference.
Post-conditions	1.The JSP Form successfully transfers data to agent system. 2. The agent returns the recommended hotel, scenery, and snack information to the user.
Basic Flows	1. The user inputs his/her preference, including the required hotel quality and price, the preferred type of scenery and snack. 2. JSP Form receives the data and transfers the data to agent system. 3. Agent evolves and automatically creates the result. 4. Agent saves the result into ontology repository. 5. A JSP renders the received result from agent, and parses it to the user.
Alternative Flows	2.1. If the connection fails, the error message will be returned to JSP Form. 5.1. If the user is not satisfied with the solution, he can execute the agent system one more time.



(a)



(b)

Figure 4 (a) use case diagram (b) goal hierarchy

3.3. Fuzzy Modeling of User's Goals

To model user goals, we apply GDUC to get a set of soft and rigid goals, use cases, and plans. To achieve these goals, agents must use particular strategies to change their mental states. We can continuously change the problem state in order to attain the goal state. In consequence, we can apply the soft requirement [11] so to formally represent the user goals. A user goal, g, is specified by the properties of agent's mental state-transition $<b, g, a>$, where b is the state before a plan, and a is the state after invoking the plan. A plan or strategy can thus be specified using the pair $<precondition, post\text{-}condition>$. The precondition and

the post-condition describe properties that should be held by states *b* and *a*. A rigid goal describes state properties that must be completely satisfied. The soft goal describes state properties that can be satisfied to a degree. We use Zadeh's test-score semantic [23] to represent the user goals. The basic idea underlying test-score semantics is that a proposition p in a natural language may be viewed as a collection of elastic constraints, C_1, ..C_k, which restricts the values of a collection of variables X =(X_1, ..X_n). In fuzzy logic, this is accomplished by representing *P* in the canonical form:

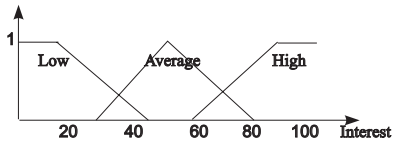
$G \Rightarrow R(P)\ IS\ A$

in which *A* is a fuzzy predicate. The canonical form of *G* implies that the possibility distribution of *R(P)* is equivalent to the membership function of *A*, namely, $\Pi_{R(p)} = \mu_A$. In the "getRecommendation" use case, the agent recommends the user a high quality hotel which is nearer to snack spot and can be represented using the canonical form below:

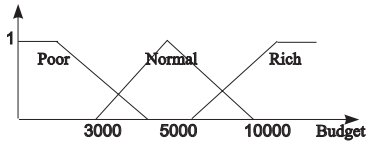
$G_1 \Rightarrow Preference(Snack)\ IS\ HIGH$

$G_2 \Rightarrow Quality(Hotel)\ IS\ HIGH$

Where *HIGH* is a fuzzy predicate. Fuzzy linguistic terms, *HIGH*, is defined in Figure 4.



(a) High, Average, and Low



(b) Rich, Normal, and Poor

Figure 4 fuzzy sets of the user's goals

The rigid goal is a specialization of the soft goal, of which the membership function of fuzzy predicate is 1.0. For this example: $G_3 \Rightarrow MaxPrice(Hotel)\ IS\ M_{max}$, where M_{max} is the maximum price that the user is willing to pay.

4. Agent Evolution Mechanism

For agent evolution, we will focus on goal, knowledge, and plan modules. Based on our goal-driven approach [11][12], we apply ontological reasoning and fuzzy modeling to precisely refine a set of soft and rigid goals. To achieve these goals, agents must use particular strategies to continuously evolve their plan and actions.

4.1. Fuzzy Reasoning Model

We can construct the fuzzy module by means of fuzzy theory and ontology. The inference mechanism of fuzzy reasoning on a rule base employed in this paper is extended from the Sugeno controller model [24]. Suppose we have a simple rule base as followed:

R_1 : if *x* is A_1 and *y* is B_1 then $z = c_1$

R_2 : if *x* is A_2 and *y* is B_2 then $z = c_2$

fact: *x* is x_0 and *y* is y_0

consequence: *z* is z_0

where A_1, A_2, B_1, and B_2 are fuzzy sets, and c_1, c_2 are real numbers. The firing levels α_1 and α_2 of rules R_1 and R_2 are computed by the *Min* operator. According to Sugeno controller definition, the control action of the rule base is obtained by

$$z_0 = \frac{\alpha_1 c_1 + \alpha_2 c_2}{\alpha_1 + \alpha_2}$$

We identify the user's preferences to build a specific user profile. In the example, according to the user's goals and various fuzzy rules, we will compute the degree of the hotel class (DH). In this case, we use some heuristic rules described in Table 2. Fuzzy linguistic terms, *VL(Very Large)*, *L(Large)*, *N(Normal)*, *S(Small), and VS(Very Small)* are defined in Figure 5. The ontology management module can provide the fuzzy rules with knowledge. The learning mechanism will be able to evolve the fuzzy rules of agents.

Table 2 the heuristic rules

HR1: IF	Max(Budget)	IS	Rich	AND	Min(Budget)	IS	High	
	THEN	DH	IS	VL				
HR2: IF	Max(Budget)	IS	Rich	AND	Min(Budget)	IS	Average	
	THEN	DH	IS	L				
HR3: IF	Max(Budget)	IS	Rich	AND	Min(Budget)	IS	Low	
	THEN	DH	IS	N				
HR4: IF	Max(Budget)	IS	Normal	AND	Min(Budget)	IS	High	
	THEN	DH	IS	L				
HR5: IF	Max(Budget)	IS	Normal	AND	Min(Budget)	IS	Average	
	THEN	DH	IS	N				
HR6: IF	Max(Budget)	IS	Normal	AND	Min(Budget)	IS	Low	
	THEN	DH	IS	S				
HR7: IF	Max(Budget)	IS	Poor	AND	Min(Budget)	IS	High	
	THEN	DH	IS	N				
HR8: IF	Max(Budget)	IS	Poor	AND	Min(Budget)	IS	Average	
	THEN	DH	IS	S				
HR9: IF	Max(Budget)	IS	Poor	AND	Min(Budget)	IS	Low	
	THEN	DH	IS	VS				

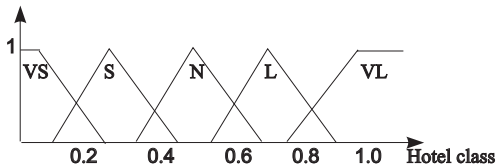


Figure 5 degree of hotel class

4.2. Goal-driven Ontology Reasoning

Ontology is a formal description of entity, entity relationships, entity attributes, and entity behaviors. In short, ontology is a method that describes conceptualized matters and supports auto-reasoning. It sets the foundation for the sharing and reusing of

domain knowledge. This gives a consistent way to express interchanging information and description of agent collaboration in distributed applications. In practical terms, developing ontology not only signifies using ontology markup language, but also arranging the classes in a taxonomic hierarchy. In order to compare resources and requirements based on their semantics, the inference mechanism quantifies the confidence level of two matching classes by computing a similarity between the two in a class hierarchy.

First, we apply a goal hierarchy [12] to analyze the goal structure. A goal structure can be built dynamically by analyzing the concepts defined in the domain- and issue-specific ontology. If sub-goals are generated, a goal confirmation form can be generated to return feedback for the user. Meanwhile, the OWL inference engine employs the domain- and issue-specific ontologies to derive the similarity between concepts. For this example, the relations between concepts are given with the pre-defined relevance value in Figure 3. Also, the semantic relation path is a directed path composed by the same type of relations from one concept to the other. The calculation of similarity between two concepts is the product of relevance values of relations that constitutes the semantic relation path. The ontology rules are specified by RuleML [25]. Thus, G_2 can be refined as followed.

$G_{21} \Rightarrow$ *Equipments (Hotel) IS HIGH*

$G_{22} \Rightarrow$ *Space (Hotel) IS HIGH*

$G_{23} \Rightarrow$ *Service (Hotel) IS HIGH*

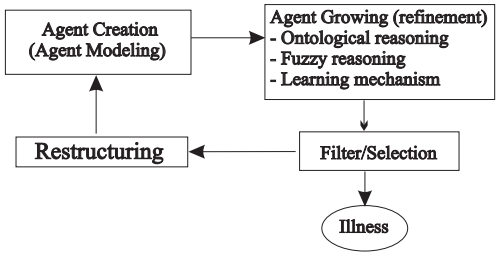


Figure 6 agent evolution process

4.3. Agent Evolution Process

In its lifespan from creation to termination, an agent experiences several states [10]. We construct an agent evolution life cycle to demonstrate the transition between these states, as shown in Figure 6.

a) Agent creation: A new agent is created for specific purposes. Based on the mental model of agents, agent factory is responsible for producing new agents. An agent is assigned to the user's goal, which may be imprecise. The agent has primitive knowledge and strategies. In our case, the hotel, snack and scenery in the region are numbered from 1 to N and represented by a binary string. Each gene is stored in an array y_i (i=1, 2, .., n), and carries the numbers of hotel/snack/scenery recommended by the agent. The chromosome carries genetic information of an individual which in our case corresponds to a solution.

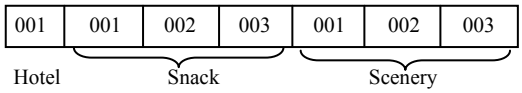


Figure 7 chromosome representation

b) Agent growth: Our approach uses random number to choose which individuals from the current population should go into the mating pool. The mating pools forms the basis of the next population. The growth mechanism is based on learning mechanism, fuzzy reasoning, and ontological reasoning.

c) The selection mechanism: We use a SOM-based (Self Organization-Based Optimization System, SOM) [18] selection approach to select the more fitting agents. It shows the performance of an agent and its ability to survive and adapt to the environment. It is also an indicator of the trend of evolution. In terms of this example, the fitness function is shown as followed.

$\Sigma (w_i * d(H, Sc_i) + w_j * d(H, Sn_j))$

H is the selected hotel, Sc_i is the *ith* scenery, and Sn_j is the *jth* snack. The w signifies the weight and d is the distance function.

d) Restructuring phase: An agent is composed of various kinds of modules. Each module has a special feature for a particular plan. In multi-agent system, there are two phases of restructuring: the inter-agent and intra-agent crossover, respectively. With the inter-agent crossover, an agent exchanges a module with another agent. On the other hand, the intra-agent crossover exchanges some parameters or strategies between modules in the same agent. As an example, we use the restructuring process which exchanges genetic material between individuals. We randomly select two individuals from the population. Crossover points are then randomly chosen and sorted in ascending order. Then the genes between successive crossover points are alternately exchanged between the individuals, depending on probability. Mutation process works by randomly selecting and modifying some of the genes present in the population. In our case, one entry is selected at random from the array of numbers and is exchanged with another number selected randomly from 1 to N.

e) The illness phase: When an agent is damaged or tampered with by malicious agents or hosts, it will fall to the illness state. After recovery, it can be treated as a growing agent again.

5. Case Study

To demonstrate our evolutionary agents in ITS, we have built a subsystem of advanced traveller information system – the hotel recommendation system. This agent system helps users to search for hotels that are close to captivating scenic spots and convenient snack places.

5.1. Simulation Environment

In this system, the user can input the budget P_r that he is willing to pay and three preferred locations. We wrote two web services to read the location point data of snack and scenery from a GPS database including hotel, scenery, and snack information. There are five target web sites of hotels running in the environment. We have also implemented respective web services of hotels to acquire the information from those web sites. The hotel information includes hotel name, room types, cost, and the current reservation state. The outputs are the hotel name, the room type, and the reservation date.

5.2. Agent System Architecture

The system architecture is shown in Figure 8. At first, we acquire the user input via a JSP Form. The JSP module transfers the information to Travel Planning Web Service (TPWS). The TPWS also obtains the data from Web Service Simplifier (WSS). This information will be parsed and transferred to the Agent Evolution Module (AEM). The AEM use GA to generate the optimized solution, and store the knowledge into the Ontology Revolution Module (ORM).

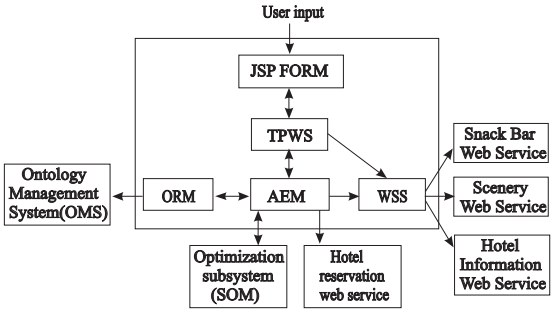


Figure 8 agent system architecture

a) TPWS: The module takes the user profile as input, and transforms the profile into a demand degree of the facilities by the user. The processes include (1) establishing the fuzzy linguistic terms for the profile, (2) building fuzzy rules and use fuzzy inference mechanism, and (3) doing defuzzification to get a demand degree.

b) ORM: This module can construct rules for ontological reasoning, and store them into the ontology management system. According to the ontology, the inference rules, and the demand degree derived from the fuzzy module, the ontological reasoning module can generate a product list for the user. This module also store user's profile in the OMS.

c) AEM: This module uses GA and SOM technique to evolve adapting actions for hotel reservation web service, which is included in another sub-project.

d) WSS: This module can get the hotel, scenery, and snack information from the related web services.

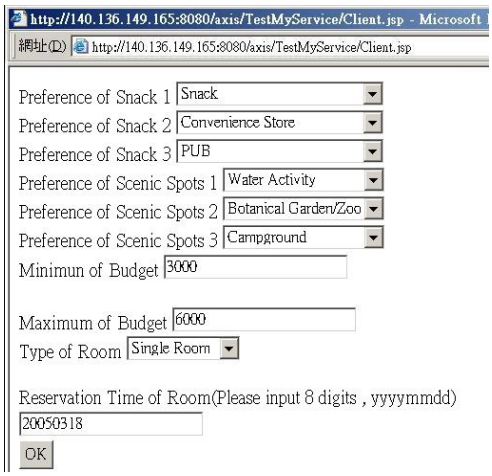


Figure 9 the JSP form

5.3. Experimental Results

The user can input his goals into the agent system as shown in Figure 9. We apply the fuzzy decision making approach [10] to compute the fitness value of the agents. Then we apply the evolution approach to generate the next generation agents. In this particular experiment, the mutation rate is 0.01. The simulations stop when the population is stable (i.e. when 95% agents have the same fitness) or the number of iterations is bigger than a predetermined maximum value (100 in our case). In table 3, we summarized the number of snack places and scenic spots, the number of evolution generation, and the execution time.

Table 3 part of experimental results

Snack	Scenery	Generation	Time (ms)
100	75	4	156
100	88	11	266
103	12	3	125
115	75	16	343
128	78	7	203
131	15	5	172
140	85	11	266
146	77	3	125
191	79	15	328
228	79	11	250
256	78	31	16
256	86	23	422
292	77	15	297

6. Conclusion

In this paper, we present a new approach for evolving intelligent agents in ITS. A goal-driven approach can construct the user's soft and rigid goals based on fuzzy set theory. A sub-goal approach is applied to evolve the agent's goals based on ontological reasoning and learning mechanism. The proposed agent model is to facilitate and control the process of evolution of agent knowledge. We construct multi-agent evolution cycle as well, which includes states of restructuring, selection, and growth. Finally, we have built an agent system to demonstrate our approach.

7. Acknowledgement

This research is partially sponsored by the Ministry of Education Program for Promoting Academic Excellence of Universities under the grant EX-91-E-FA06-4-4 and National Science Council under the grant NSC93-2213-E-030-008.

References

[1] M. Barbuceanu and M.S. Fox, Cool: "A language for describing coordination in multi agent system." In Proceedings of the International Conference on Multi gent Systems, ICMAS-5, pp. 17-24, 1995.

[2] A. Bonarini, " Evolutionary learning, reinforcement learning, and fuzzy rules for knowledge acquisition in agent-based systems. " Proceedings of the IEEE, Vol. 89, No. 9, pp.1334-1346, 2001.

[3] P. Cristea, A. Arsene, and B. Nitulescu. " Evolutionary intelligent agents. " In Proceedings of the Congress on Evolutionary Computation, Vol. 2, pp. 1320-1328, 2000.

[4] E. D. Ferreira and E. Subrahmanian, and D. Manstetten, " Intelligent agents in decentralized traffic control. " In Proceedings of the IEEE Conference on Intelligent Transportation Systems, pp. 705-709, 2001.

[5] R. H. Guttman and P. Maes. "Agent mediated negotiation for retail electronic commerce. In Proceedings of Agent mediated Electronic Commerce,"1st International Workshop on Agent Mediated Electronic Trading, pp. 70-90, 1999.

[6] D.C. H. Han and N. Parameswaran. " Multi agent problem solving with mental state. " In Proceedings of the 2nd Australian and New Zealand Conference on Intelligent Information Systems, Vol. 29, pp. 288- 292, Australian, 1994.

[7] N. Kase, M. Hattri, A. Ohsuga, and S. Honiden. " Informirror - agent-based information assistance to drivers. " In Proceedings of the IEEE/IEEJ/JSAI International Conference on Intelligent Transportation Systems, pp.734-739, 1999.

[8] N. Kohata, T. Yamaguchi, M. Sato, T. Baba, and H. Hashimoto. " Dynamic formation generating for intelligent transport systems using algorithm to select function by environmental information. " In Proceedings of IEEE/IEEJ/JSAI International Conference on Intelligent Transportation Systems, pp. 798-803, 1999.

[9] J. Y. Kuo. " Ontology supported fuzzy intelligent agent. " In Proceeding of the 15th Workshop on Object-oriented Technology and Application, Taiwan, 2004.

[10] J. Y. Kuo and J. Lee. " Evolution of intelligent agent in auction market. " In Proceedings of 2004 IEEE International Conference on Fuzzy Systems, Vol. 1, pp.331-336, 2004.

[11] J. Lee and J.Y. Kuo. "New approach to requirements trade-off analysis for complex systems. " *IEEE Transactions on Knowledge and Data Engineering*, Vol. 10, No. 4, pp. 551-562, 1998.

[12] J. Lee, N.L. Xue, and J.Y. Kuo. " Structuring requirement specifications with goals. " *Information and Software Technology*, Vol. 43, pp.121-1350, 2001.

[13] B. Maskell and M. Wilby. " Evolving software agent behaviors. " In Proceeding of the Global Telecommunications Conference on : The Key to Global Prosperity, Vol. 1, pp. 90-94, Nov. 1996.

[14] S. Park, V. Sugumaran, and S. Lee. " An architecture-Centric approach for multi-agent system development and application. " In Proceedings of the Third International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems, pp.160-169, 2001.

[15] D. A. Roozemond. " Using intelligent agents for practice, real-time urban intersection control. " *European Journal of Operational Research*, Vol. 131, No. 2, pp.293-301, 2001.

[16] A. S. Rao and M. P. Georgeff. " Modeling rational agents within a bdi architecture. " In Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91). pp. 473-484, 1991.

[17] R. J. F. Rossetti, S. Bampi, R. Liu, D. Van. Vliet, and H.B.B. Cybis. " An agent-based framework for the assessment of drivers' decision making. " In Proceedings of the 2000 IEEE International Conference on Intelligent Transportation Systems, pp. 387-392, 2000.

[18] M. C. Su, Y. X. Zhao, and J. Lee. " SOM-based

optimization. ” In Proceeding of the 2004 International Joint Conference on Neural Networks (IJCNN), Vol. 1, pp. 25-29, 2004.

[19] Y. Tanaka, M. Fukuda, and N. Suzuki. “ Dynamic objects to support evolution of mobile agents. ” In Proceedings of the IEEE Pacific Rim Conference on Intelligent Transportation Systems, Vol. 2, pp. 748- 751, 2001.

[20] J. Wahle and M. Schreckenberg. “ A multi agent system for online simulations based on real world traffic data. ” In Proceedings of the 34th Hawaii International Conference on Systems Sciences, pp. 1-9, 2001.

[21] T. H. Wang, S. Kwong, Y. Jin, W. Wei, and K.F. Man. “ Agent-based evolutionary approach for interpretable rule-based knowledge extraction. ” *IEEE Transactions on Systems*, Man, and Cybernetics-Part C: Applications and Reviews, Vol. 99, pp.1-13, 2005.

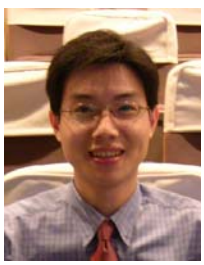
[22] T.H. Wang, S.U. Guan, and S.H. Ong. “ An agent-based auction service for electronic commerce. ” In Proceedings of International ICSC Symposium on Interactive and Collaborative Computing, Australian, CD #1524-045, 2000.

[23] L.A. Zadeh. “Test-score semantics as a basis for a computational approach to the representation of meaning. ” *Literacy Linguistic Computing*, Vol. 1, pp.24-35, 1986.

[24] H.J. Zimmermann. *Fuzzy Set Theory and Its Applications*. Kluwer Academic, MA, 1996.

[25] RuleML Website. http://userpp..umbc.edu/#mgandh1/2002/06/damlruleml/.

Jong-Yih Kuo received his BS degree from National Tsing Hua University, Taiwan, Republic of China, in 1991, and his PhD degree from the National Central University, Taiwan, in 1998. He is now an Assistant Professor in the Software Engineering Laboratory of the Department of Computer Science and Information Engineering at the Fu Jen Catholic University in Taiwan. His research interests include agent-based software engineering and fuzzy logic.



Shin-Jie Lee received his B.S. degree in Mathematics from National Changhua University of Education, Taiwan, in 2000. He is currently a Ph.D. student in the Department of Computer Science and Information Engineering of National Central University. His current research interests include agent-based software engineering, service-oriented computing, and software engineering.

Chia-Ling Wu received his B.S. degree in Computer Science and Information Engineering from National Chiao Tung University, Taiwan, in 1999. He is currently a Ph.D. student in the Department of Computer Science and Information Engineering of National Central University. His current research interests include agent-based software engineering, service-oriented computing, and software engineering with computational intelligence.

Nien-Lin Hsueh is an assistant professor in the Department of Information Engineering and Computer Science at Feng Chia University in Taiwan since 2003. Before joining Feng Chia University, he is an assistant professor at Shu-Te University from 2001 to 2003. His research interests include fuzzy logic, agent-based software engineering, object-oriented software engineering, design pattern, component-based software engineering, service-oriented architecture and software quality. Now he is also a researcher in the Office of Information Technology at Feng Chia University, where his responsibilities include implementing CMMI-based software process improvement project in the software development division. CMMI is a software process improvement framework proposed by Software Engineering Institute at Carnegie Mellon University. He received his Bachelor of Computer Science degree in Soochow University, Taiwan, in 1993, Master and PhD degree in computer science from National Central University, Taiwan, in 1995 and 1999 respectively. Contact him at nlhsueh@fcu.edu.tw.

Jonathan Lee is a professor in the Computer Science and Information Engineering at National Central University (NCU) in Taiwan, and was the department chairman from 1999 to 2002. He is currently the director of Software Research Center at NCU. His research interests include agent-based software engineering, service-oriented computing, and software engineering with computational intelligence. He has authored more than 100 journal and refereed conference papers, and is the editor-in-chief of International Journal of Fuzzy Systems and in the editorial boards of Fuzzy Sets and Systems, International Journal of Artificial Intelligence Tools, and Fuzzy Optimization and Decision Making. He received his Ph.D in computer science from Texas A&M University in 1993. He is the president of Taiwan Software Engineering Association, a senior member of the IEEE Computer Society and a member of the ACM.