Journal of Universal Computer Science, vol. 15, no. 13 (2009), 2701-2725
submitted: 31/10/08, accepted: 13/6/09, appeared: 1/7/09 © J.UCS

Interactive Learning of Independent Experts' Criteria for Rescue Simulations

Thanh-Quang Chu
(Institut de la Francophonie pour l'Informatique, Hanoi, Vietnam
Institut de Recherche pour le Développement, Bondy, France
thanh.quang@gmail.com)

Alexis Drogoul
(Institut de la Francophonie pour l'Informatique, Hanoi, Vietnam
Institut de Recherche pour le Développement, Bondy, France
alexis.drogoul@gmail.com)

Alain Boucher
(Institut de la Francophonie pour l'Informatique, Hanoi, Vietnam
alain.boucher@auf.org)

Jean-Daniel Zucker
(Institut de Recherche pour le Développement, Bondy, France
jdzucker@gmail.com)

Abstract: Efficient response to natural disasters has an increasingly important role in limiting the toll on human life and property. The work we have undertaken seeks to improve existing models by building a Decision Support System (DSS) of resource allocation and planning for natural disaster emergencies in urban areas. A multi-agent environment is used to simulate disaster response activities, taking into account geospatial, temporal and rescue organizational information. The problem we address is the acquisition of situated expert knowledge that is used to organize rescue missions. We propose an approach based on participatory design and interactive learning which incrementally elicits experts' preferences by online analysis of their interventions with rescue simulations. An additive utility functions are used, assuming mutual preferential independence between decision criteria, as a preference for the elicitation process. The learning algorithm proposed refines the coefficients of the utility function by resolving incremental linear programming. For testing our algorithm, we run rescue scenarios of ambulances saving victims. This experiment makes use of geographical data for the Ba-Dinh district of Hanoi and damage parameters from well-regarded local statistical and geographical resources. The preliminary results show that our approach is initially confident in solving this problem.

Keywords: Disaster Response, Multi-Criteria Decision Making, Decision Support System, Multi-agent Simulation, Interactive Learning, Preference Elicitation, Utility Function, Participatory Design.
Categories: L.1.0, L.1.1, L.3.3, L.5.0, L.5.1, M.0, M.4

1 Introduction

Disaster response (a phase of disaster management) provides for the immediate protection of life and property, re-establishing control and minimizing the effects of a disaster. Information technology provides capabilities that can help people grasp the dynamic realities of a disaster more clearly and help them formulate better decisions more quickly and information technology can help keep better track of the myriad details involved in all phases of disaster management (including disaster response) [Rao, 07].

Agent-based modeling (ABM) can serve as a powerful simulation technique for analyzing large-scale urban disasters. An agent-based emergency response model can utilize the large amount of information about the possible rules of behavior for people, hospitals, on-site responders and ambulances, without depending on the scarce knowledge about the efficacy of those rules or the global dynamics [Narzisi, 06a]. Furthermore, successful response starts with a map; all disasters have a temporal and geographic footprint that identifies the duration of impact and its extent on the Earth's surface [CPC, 07]. Many research projects in the literature rely on the approach of agent-based simulation with geographical information system (GIS) data to build, improve and evaluate rescue planning scenarios of decision support system for disaster response problem; we want to describe here two representative works of other research teams which use similar techniques of artificial intelligence in resolving rescue planning of multi-agent simulation.

The first one is the Large Scale Emergency Readiness (LaSER) project of New York University's Center for Catastrophe Preparedness and Response (CCPR), has explored how ABM can serve as a powerful simulation technique for analyzing large-scale urban disasters. The team effort in this direction has resulted in a new multi-agent based disaster simulation framework, able to model and simulate catastrophic scenarios [Narzisi, 06b].

[Narzisi, 06b] cast the tuning of an ABM for emergency response planning as a multi-objective optimization problem (MOOP). Planning can be seen as the problem of adjusting the controllable parameters in the interaction between different classes of agents (hospitals, persons, on-site responders, ambulances, etc.) and available resources, in order to minimize the negative consequences of a catastrophic event (the number of casualties (affected people), fatalities (mortalities), the average ill-health of the total population, average waiting time at the hospitals, etc.). They then propose the use of multi-objective evolutionary algorithms (MOEAs) for searching and selecting emergency response plans in the search space of all possible solutions.

Following this approach, we can see that users can control the static input parameters of response planning to attain some global objectives. However, it becomes impractical when there are many real-world parameters, a huge search space, large number of generic generations and considering the time consumed to find optimal solutions. Furthermore, controllable parameters cannot change from a given agent to the others in the same rescue group; their values are fixed at the beginning of the simulation (response planning is limited for only static solutions). On the other hand, good response planning allows for diversity in the decision strategies between agents (each agent has his own instance of controllable parameters) and its value changes dynamically during the simulation to adapt to unexpected situations.

The second one is a research project of Laval University, Canada for the international RoboCupRescue competition called Damas-Rescue. In this work, some problems of rescue agents are defined with their appropriate solutions. For instance: the development of a learning algorithm to improve the coordination between Fire-Brigade agents when they are choosing which fire to extinguish [Paquet, 04a], a selective perception reinforcement learning algorithm to estimate the number of Fire-Brigade agents necessary to extinguish a fire [Paquet, 04b], a scheduling algorithm for Ambulance agents to maximize the number of civilians that could be saved [Paquet, 05a], K-Nearest-Neighbors (KNN) approach for Ambulance agents to estimate the expected death time of civilians, and an algorithm for Police-Force agents based on a dynamic representation of the belief states to choose the best road to clear [Paquet, 06a].

[Paquet, 05b] have proposed Real-Time Belief Space Search (RTBSS) for Partially Observable Markov Decision Processes (POMDPs) problems wherein-which agents search to find the best action to execute at each cycle of simulation. Hence, each agent has his own dynamic decision strategy during simulation. [Paquet, 06b] have also proposed a pruning strategy, a hybrid algorithm for online search to avoid the overwhelming complexity of computing a policy for every possible situation the agents could encounter while they execute a reinforcement learning algorithm by autonomously evaluating local reward functions. So, this approach is feasible for an uncertain real-time multi-agent environment like RoboCupRescue simulation.

For our work presented in this paper, we resolve the same problem of emergency planning after urban disasters; the aim for building a reliable DSS is to simulate the relief effort and to learn human strategies from various disaster scenarios. The devastated infrastructures and human casualties are input GIS data for the rescue simulation. Rescue teams, such as ambulances, fire-fighters or policemen are modelled and simulated by agents along with their behaviours. The DSS in this case have to address two issues (figure 1): the first is the ability to simulate different disaster scenarios integrating all available information, coming from either GIS (Geographical Information System) data or other sources; the second is the ability to propose rescue solutions that are really realistic with respect to the experts' criteria.

Concerning the first issue, i.e. building a simulator for rescue teams acting in large urban disasters, the RobocupRescue community is proposing a multi-agent simulation environment for urban earthquakes. The goal of the agents (representing fire-fighters, policemen and ambulance teams) consists in minimizing the damages caused by the earthquake. Damages include buried civilians, buildings on fire and blocked roads. The RoboCupRescue simulation environment is a useful test-bed for evaluating cooperative multi-agent systems. However, a major problem with RoboCupRescue is its lack of support for standard GIS description, which prevents it from being compatible with existing GIS city descriptions. This paper will be mainly devoted to the presentation of our solution regarding the second issue. In order to capture the experts' experience and knowledge, we propose using an original combination of participatory design and interactive learning. In this approach, the experts are invited to interact with the simulation, thanks to a participatory design method, by trying out different rescue scenarios and representing their decision criteria or modifying the agents' behaviours in order to improve, as well as validate, the realism of the whole simulation; besides, the interactive learning method implemented allows rescue agents eliciting the informal preferences of experts in experimented scenarios and adapting agents' decision to experts' one in latter scenarios.

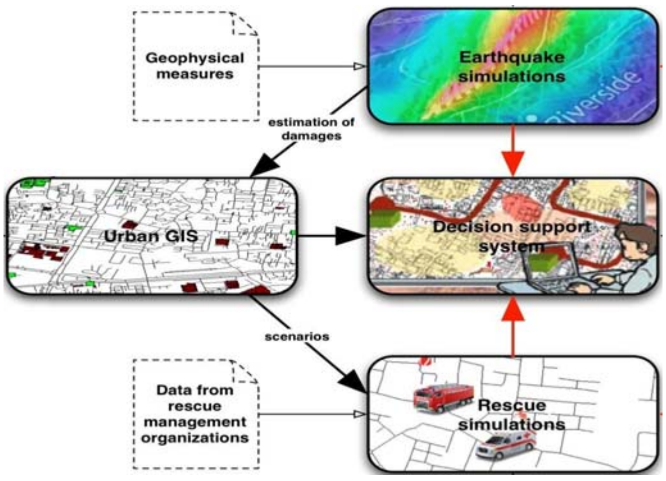


Figure 1: The rescue simulation and the decision support systems are the results of solving the first and second problems respectively.

For the next, we will present, in the second session, the participatory design applied in our rescue simulation. Because, we focus on the rescue activities of ambulances to illustrate our approach; so in the third section, we will present the decision-making problem of ambulances in rescue simulation. Afterwards, fourth section is our approach in elicitation of expert's preference with interactive learning algorithm for ambulances. Next, the experimental protocol and some preliminary results of our solution will be presented respectively in the fourth and fifth sections. Finally, in two last sections, we arrive at conclusions, discuss about perspectives and our future work to improve the solution.

2 Participatory Design

If we want to use rescue simulations as supports in decision-making processes, we not only need to make them as realistic as possible, but also to reflect, as much as possible, the decisions that would be taken by an expert in real situations. Most of these decisions are a compromise between existing regulations, written rules and, perhaps more importantly, the experience of the expert in similar situations.
Designing a process by which this latter aspect can be, with the help of learning techniques, incorporated in the behaviour of the agents will not only help in building more realistic simulations, but also increase the confidence of the expert in the support eventually provided by the simulation.

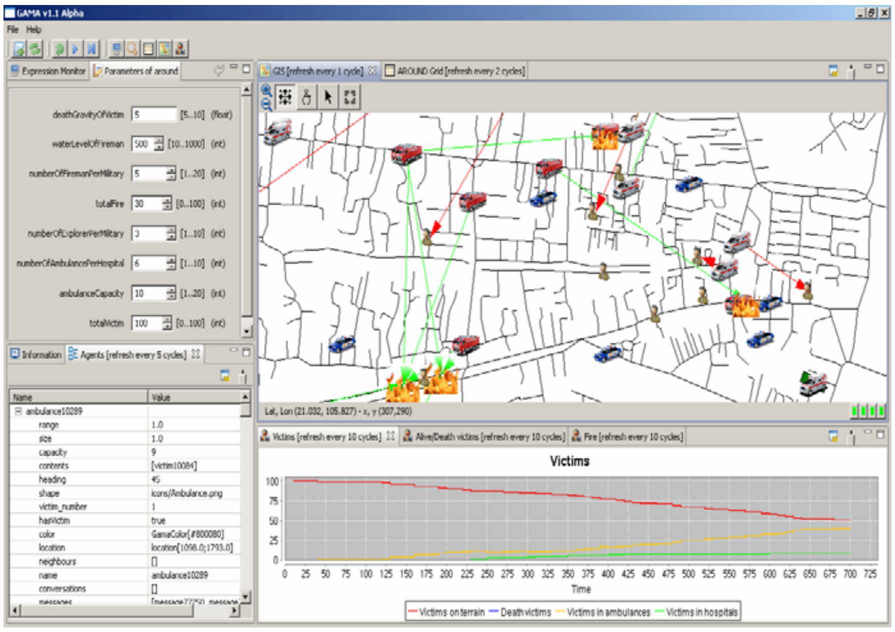
However, in order for this process to be successful, it has to be designed very carefully. [Nguyen-Duc, 07] has proposed, in a similar context, to apply methodological advices derived from those employed in practices like participatory design or user-centered design in order to control the experimental process by which agents can acquire knowledge from the experts.

Basically and similarly to [Nguyen-Duc, 07], our experiments will then rely on four major components: (1) the design of a flexible and ergonomic user-interface that would allow for real-time interactions between the expert and agents in the simulator; (2) the design of well-thought scenarios based on realistic conditions and corresponding to specific learning tasks and objectives; (3) the design of an experimental protocol composed of sessions organized around a goal, a set of experimental tasks and a set of support scenarios; (4) the design of a machine learning method for eliciting expert's knowledge.

2.1 User interface

Implementation of the interface between ambulances and users is necessary to ensure that ambulances can acquire expert knowledge during the learning process. This provides an excellent support for both situational awareness and user action during the course of the simulation.

For example, in figure 2, at any time, the objectives of the ambulance can be either victims or hospitals. The red arrow links the ambulance to its best objective. The user can pause upon the simulation and change the arrow to modify the objective of each ambulance.



deathGravityOfVictim	5	[5..10]	(float)
waterLevelOfFireman	500	[10..1000]	(int)
numberOfFiremanPerMilitary	5	[1..20]	(int)
totalFire	30	[0..100]	(int)
numberOfExplorerPerMilitary	3	[1..10]	(int)
numberOfAmbulancePerHospital	6	[1..10]	(int)
ambulanceCapacity	10	[1..20]	(int)
totalVictim	100	[0..100]	(int)

Name	Value
ambulance10289	
range	1.0
size	1.0
capacity	9
contents	[victim10084]
heading	45
shape	icons/Ambulance.png
victim_number	1
hasVictim	true
color	GamaColor[#800080]
location	location[1098.0;1793.0]
neighbours	[]
name	ambulance10289
conversations	[]
messages	[message77250, message

Figure 2: A prototype of the user interface that might be used during the learning sessions. This prototype has been designed in GAMA [Amouroux, 07]

Many of the lessons learned during these tests will serve for the implementation of the actual interface that will be used throughout the experiments with experts. The control buttons, not very surprising in this context, are that the interface should adapt, as much as possible, to the rhythm and needs of the user: giving him or her the possibility to change the speed of the simulation, to zoom in and out on the situations, to dynamically change the colours and shapes of the information displayed, to hide or reveal any pieces of information, to come back in time, etc. appears to be a cognitive (and not simply cosmetic) necessity for the user to get familiarized with the tool.

However, if we want to capture the knowledge mobilized in situation, we also need the interface to be as pressing and demanding as a critical real context could be. Once the user is familiarized with the manipulation of the simulator, all experiments should then take place in real-time. Since we cannot, of course, ask an expert to play his/her role during 12 or 24 hours (like in reality), the learning sessions will be cut into time-bounded, incremental episodes with their own goal, learning task, and time limit, which are presented below.

2.2 Scenario design

Even when the actors are already at ease with the user interface of the simulator, the quality of the knowledge that might be captured will strongly depend on: (a) The commitment and motivation of the user (which is known to decrease over time); (b) The realism of the scenarios provided and their understanding by the user; (c) The focusing (in term of task, or goal) of the sessions during which the users will play their role;

Therefore, a complete learning session will be organized as a succession of “episodes”, each of them being structured in the following way: (1) The task to be fulfilled by the agents and the timeframe within which they can accomplish it (for instance, save a maximum of victims in the minimum of time, save the most critical victims and communicate about the others, etc.) is communicated to the user and we make sure it is perfectly understood. Some episodes will of course share the same task. (2) For each task, a sequence of scenarios (see below) is then chosen, ranging from simple to complex ones. Each scenario will serve as a support for an “episode” of the session, and its results (in terms of machine learning) reused for the next episode in the sequence. (3) The set of criteria susceptible to be learnt (or ranked) during an episode depends on the complexity provided by the scenario. For instance, in basic scenarios, it may only contain the geographical location of the agents, while more advanced ones might want to take their communication (information received, etc.) into account.

There are many ways into which short-term focused scenarios could have been designed. Yet, we wanted a method that would allow for the learning episodes to act as different “layers” of increasing complexity, each of them focusing on the ranking of its own set of criteria and using the previous ones as starting points. As the criteria represent bits of information perceived, collected or received by the agents, we chose to base the progression of the scenarios on that of the “informational context” that the agents (and, therefore, the user) are facing. For instance, for a task like “locating and carrying a maximum of victims”, in a situation where only one ambulance and one hospital are being simulated (see figure 3), the decision of the agent will be based on a subset of the criteria used in a situation where several ambulances (or hospitals, or both) are present. And the criteria used in the latter situation will be themselves a subset of those necessary to take into account if all these agents are communicating or coordinating themselves.

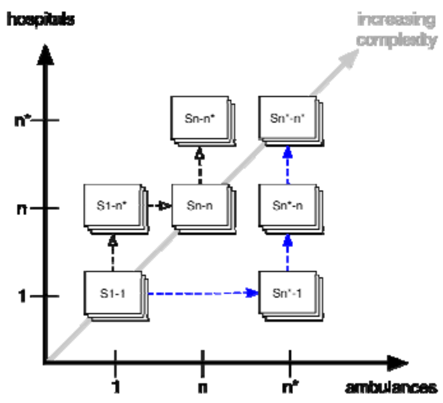


Figure 3: Description of multiple scenarios as incremental representations of informational contexts of increasing complexity; In the bottom-left corner, the context only implies one ambulance and one hospital and therefore few information on which to base a decision; in the top-right corner, n indicates n agents able to communicate (for sharing information or coordinating their task), which represents the most complex situation agents can face if we only take hospitals and ambulances into account.*

Of course, the scenarios space can grow as needed to account for other agents (firemen, civilians, victims themselves, etc.) or criteria (communication of orders, change in priorities, etc.). But we have to keep in mind that (1) not all of them are realistic; (2) no expert will be able to play them all. The path they will eventually follow, in their session, from one episode to the other, will be different from one expert to the other, and decided after each run through an interview with the modellers and an evaluation of their previous interactions with the agents.

2.3 Experimental Protocol

As we discussed in the session 2.2, we organize the experiments following the series of learning sessions and each of them contains a series of episodes (see figure 4) to ensure that the sequence of scenario is ranged from simple to complex ones. Each learning session contains a task to fulfil and a set of properties. We change from one session to another by changing the task and/or the properties. The scenarios of the episodes in the same learning session contain common session properties and some individual episode properties. We illustrate our experimental protocol in figure 5. The meaning of scenario properties is explicated in table 1. We use only one task "*control ambulances*" for all learning sessions in this example; other possible tasks of users are listed in table 2.

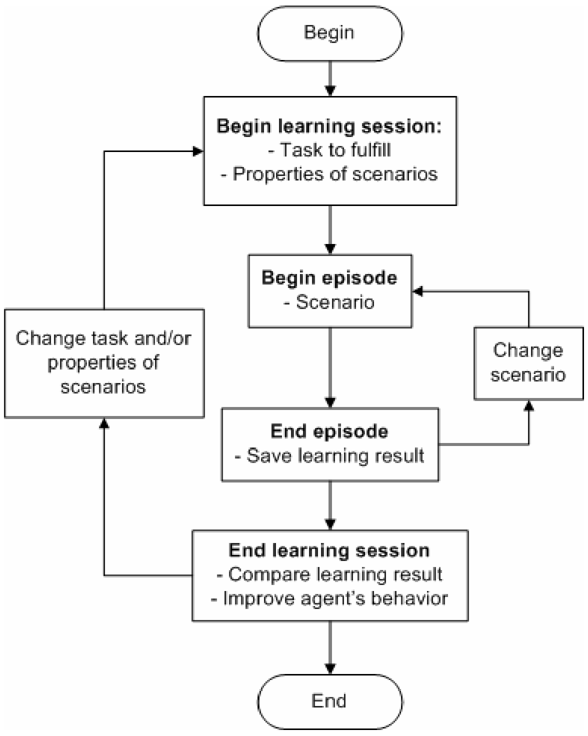


Figure 4: Experiment protocol

Properties of scenario	***Description***
Ambulance capacity	Maximum number of victims can be taken
Number of ambulances	
Number of hospitals	
Unlimited hospital capacity	Hospital capacity to take victims is unlimited
Limited hospital capacity	Hospital capacity to take victims is limited
Perfect situational awareness	Ambulances have list of all victims in the map
Partial situation awareness	Ambulances have only partial victim list
Decentralized communication	Ambulances inform directly to other ambulances for each time they choose one victim to take
Centralized communication	Ambulances inform to other ambulances via centre for each time they choose one victim to take
Unlimited energy	Ambulances have unlimited energy
Limited energy	Ambulances must refill energy
Explorer	Centre use explorer to search victims on the map

Table 1: List of possible properties of one scenario

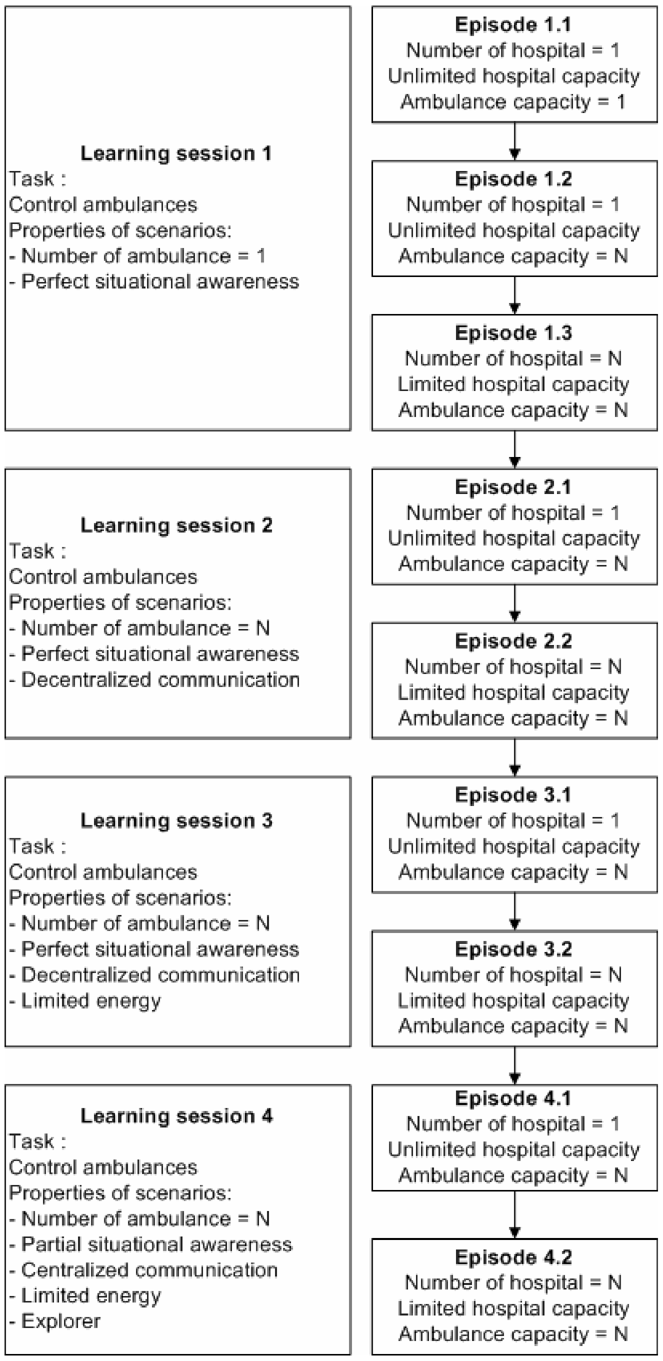


Figure 5: Example of sequence of scenarios in the experiment protocol.

Task of users	***Objective***
Control ambulances to save all victim	Minimize number of fatalities
Control hospitals to distribute victims to ambulances	Minimize overloaded ambulances
Control centre to distribute victims to hospitals	Minimize overloaded hospitals
Control firefighters to extinguish all fires	Minimize time to extinguish fires
Control firestations to distribute fires to firefighters	Minimize time to extinguish fires
Control centre to distribute fires to firestations	Minimize overloaded firestations
Control explorer to explore the map and send information about victims to centre	Maximize explored region of map

Table 2: List of possible users' task of one learning session

2.4 Eliciting experts' knowledge

Eliciting expert's knowledge is in fact a process of capturing and modelling preferences of human beings from informal experiences, essential in the context of decision-making. The experts' preferences are usually not elicitable or difficultly elicitable because following reasons: it depends on the experience of the expert in the domain, and are therefore highly subjective; it depends on situational awareness of experts (the completeness, accuracy and precision of the information they receives for decision situation); there are large number of parameters, criteria must be synthesized and these criteria are extremely variable and difficultly to be formalized in the understandable, reusable ways by artificial agents. For example, ambulances don't know always how imitate experts to calculate criteria such as "time-to-dead-of-victim", "distance-to-nearest-available-hospital", etc. Besides, another important thing of ambulance is how imitate experts to combine all criteria in a utility function for evaluating the value of a decision.

Following [Boutilier, 05], preference elicitation is generally required when making or recommending decisions on behalf of users whose utility function is not known with certainty. Although one can engage in elicitation until a utility function is perfectly known, in practice, this is infeasible. Because user preferences are always incomplete initially, and tend to change in different context, in addition to user's cognitive and emotional limitations of information processing, preference elicitation methods must also be able to avoid preference reversals, discover hidden preferences, and assist users making tradeoffs when confronting with competing objectives [Li Chen and Pearl Pu, 04].

The theoretical basis of user preference models can be found in decision and utility theory. Multi-attribute utility theory focuses on evaluation of choices, outcomes (or alternatives) for a decision problem. For example: an ambulance needs to decide, for a given moment, saving firstly one among many victims of the simulation in regarding the distance from him to each victim, injured level of each victim, etc. The value function reflects the decision maker's preferences on a particular outcome. In case of uncertain decision scenarios, where the outcomes are characterized by probabilities, a more complex function, utility function, is need to evaluate the "utility" of a decision [Li Chen and Pearl Pu, 04].

Many decision support systems have made various assumptions concerning preferences structures. The normally applied assumption is additive independence [Keeney and Raiffa, 76], where the value (or utility) of any given outcome can be broken down to the sum of individual attributes. The assumption of independence allows for the reduction of the number of outcomes for consideration and the construction of less complicated and more manageable value functions.

In many cases, attributes are preferentially dependent and thus assumptions of decomposability are incorrect. In order to elicit full utility function as well as save user's effort as much as possible, some research works have proposed to elicit the preferences of a user using the closest existing preference structures as potential default. They don't make any restrictive assumptions on the form of the underlying value functions, but make assumptions about the existence of complete or incomplete preference structures elicited from a population of users [Li Chen and Pearl Pu, 04].

Utility independence leads to less convenient decompositions, such as multi-linear [Keeney and Raiffa, 76] or hierarchical [Von Stengel, 88], [Wellman, 92]. Most previous efforts in the artificial intelligence community to adapt modern graphical modelling to utility functions employ the generalized additive decomposition [Bacchus and Grove, 95], [Boutilier, 01], [Gonzales and Perny, 04]. In contrast, our work continues the other thread, based on the weaker utility independence assumption. We assume that criteria are preferentially independent for multi-criteria decision making problem of rescue agents.

Therefore, we propose an interactive learning method which allows rescue agents adapting their preferences to expert's intervention. Each agent's preference is formulated as an additive utility function, collecting decision criteria. The learning algorithm adjusts, for each time expert intervene to agent's decision, the coefficients of the agent's function by resolving an incremental linear programming problem. The details of this method will be presented in the next sections.

3 Decision-Making Problem of Ambulances

3.1 Decision situation of ambulances by example

The ambulance's task is to take care of victims in its defined area. Their goal is to provide assistance to a maximum of injured victims thusly assuring as few deaths as possible. They must urgently perform on-the-fly first-aid, and/or transporting, with the least delay, injured victims to hospitals. An ambulance can normally carry several injured at the same time.

Figure 6 shows an example of an emergency situation. In this example, we assume that the ambulance A_1 (in the centre of figure) has access to all of the information inherent in the situation. If A_1 carries no victim, it must decide to which victim it will provide first-aid and to take the victim to the hospital if necessary. For example, A_1 may decide to go to the nearest victim V_5, and then take him to hospital H_2. But if A_1 knows that the victim V_6 is more seriously injured than V_5, it could also decide to go first to V_6. It is also possible that the victims V_1, V_2, V_3 are seriously injured and because they are all three near one-another, the ambulance A_1 may decide to go to this group, and then to take them to hospital H_2, this choice will probably save three victims V_1, V_2, V_3 instead of one victim V_6. Another possibility can happen: if the ambulance A_1 knows that the ambulance A_3 will take care of the group of victims V_1, V_2, V_3 and that the group of victims V_9, V_{10}, V_{11} are seriously injured, the ambulance A_1 may decide go to the group V_9, V_{10}, V_{11} and then take them to hospital H_3. In this case, the solution will probably save six victims V_1, V_2, V_3 and V_9, V_{10}, V_{11} instead of three victims V_1, V_2, V_3.

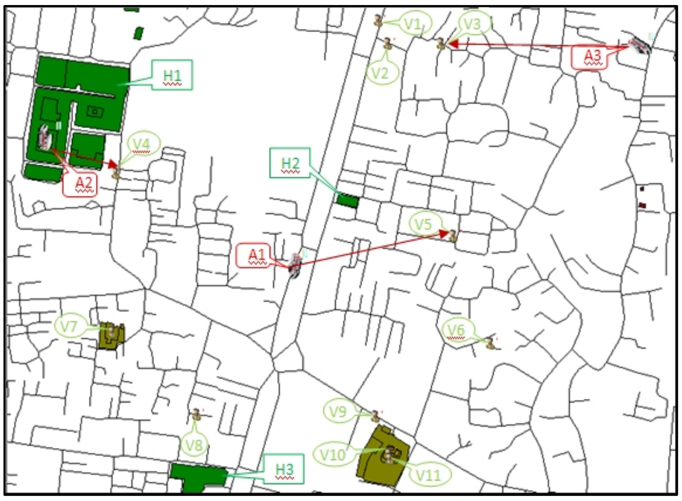


Figure 6: Example of a possible emergency situation. Ax represents ambulances, Vx represents victims awaiting rescue and Hx are hospitals in the neighbourhood.

3.2 Decision criteria of ambulance

The ambulance's decision may depend on a lot of information, so decisions must follow certain criteria to improve their relief activities. For example, after experiencing the above situations, the ambulance may acquire preference as follows: the ambulance prefers going to "hot" positions on the map, where there are more victims who require assistance. The ambulance must take the victims' injury severity level into account: the more seriously injured victims should be tended to earlier. The places, where other ambulances will eventually arrive, will be not preferred by ambulance. All of these decision criteria of ambulances will be combined as decision strategy and used in later situations.

Normally an ambulance does not have access to all the information related to its situation. Some possible choice can thus occur at the same time. Let us consider a situation where the ambulance is in the process of taking one or more seriously injured victims to the nearest hospital but is informed (by the information centre or the other ambulance via some communication channels) that a group of victims in need of rescue was spotted right along the road he is travelling. The decision is currently: either go to the hospital, or go to the group of outside victims to practice first-aid. Example of several criteria might be involved in this decision, as shown in Table 3.

Decision criteria of the ambulance to choose a victim	**Min/Max**	**Ab.**
Time-from-ambulance-to-victim	(-)	C^1
Time-to-death-of-victim	(-)	C^2
Time-from-victim-to-nearest-other-victim	(-)	C^3
Time-from-victim-to-nearest-other-available-ambulance	(+)	C^4
Time-from-victim-to-nearest-available-hospital	(-)	C^5

Table 3: Example of decision criteria of the ambulance to choose a victim: the minus sign (-) indicates a criteria to minimize while the plus (+) sign indicates a criteria to maximize

We have two types of criteria: criteria to maximize (+) which show that the victim having greater values for these criteria will have higher priority in the ambulance's decision process; and criteria to minimize (-) which show that the victim or the hospital having lesser values for these criteria will have higher priority in the ambulance's decision process.

3.3 Expert's intervention in the ambulance's decision

Making appropriate decisions for the ambulances raises difficulties for two reasons. Firstly, there are too many criteria involved in this decision. Secondly, each ambulance only has partial knowledge about the situation (local view) at the moment of decision-making; thus, the ambulances lack both necessary information and the strategies to take good decisions. Many approach in literature of DSS use searching algorithm in uncertainty (like in DamasRescue and LaSER reseach project) to find good decision strategies for agents. However, these approaches have high complexity in both memory and time consumed (the reinforcement learning of DamasRescue needs not only many space to keep large number of states in Makov Decision Process but also great quantity of examples for training agents; besides, the genetic algorithm of LaSER project have to collect so many parameters and execute so many generations to obtain optimal solutions for only initial response planning).

Taking into account human aspects help ambulances to make use of experts' capacity; human experts can provide not only supplementary information about the situation but also their useful knowledge and experience to aid in the ambulances' decision, and/or they may force the ambulance to take a particular decision with or without explanation.

For example, we assume that ambulance agents are constrained by an initial decision strategy. As long as the user considers the decision taken by the ambulances is optimal, s/he need not interact. On the other hand, if the user identifies a "better" solution, s/he may interact, by specifying or not upon which criteria he is basing his interaction (for instance: user can show that s/he chooses to go to hospital because one victim transported by ambulance can die immediately in short time). Thus, the user can “play the role” of the ambulance by forcing a decision (to continue to the hospital or to stop and treat roadside victims). The objective is for the ambulance agents to gradually acquire, by collecting enough of these couples “criteria to combine-–decision to perform” and generalizing these cases (through adapted machine learning algorithms), a decision strategy that can be reused independently in close or similar circumstances. The building of decision strategy of ambulances to imitate experts’ decision is elicitation process of human preferences.

4 Learning to capture experts’ experiences

To elicit experts’ preference, we propose an interactive learning algorithm which combines, at the same time, two methods: decision tree and utility function for representing agent’s behaviour. In fact, we distinguish the behaviour of agents at two decision levels: the first one is “***what***” level which is used to identify what type of action that will be done by agent, for example: an ambulance have to choose one between following actions to execute *“goto_victim*”, *“goto_hospital*”, *“goto_fuel*”, etc. We use a decision tree to represent the agent behaviour at this level; the second level is “***which***” level is used to identify the detail parameter for each action type, for example: if the ambulance has decided to execute *“goto_victim*”, then *which victim* between several victims the ambulance should go to? To respond to this type of question, we use a utility function to represent agent decisions at this level.

4.1 Representation of “what” level by decision tree

The knowledge and experience used by the human experts is usually expressed as rules, with priorities between them and some exception cases; or that humans have some generic (or prototypical) situations with associated actions in their mind. The decision tree as well as rule-based representation is appropriate ways for the human actor and the agent can understand each other. We show, in figure 7, an example of ambulance’s behaviour in form of a decision tree. Thus, the ambulance can identify the action corresponding to its current situation based on this tree. For instance, the ambulance always verifies its energy level. According to its estimation if energy will be soon empty, then it decide *“goto_fuel”* to fill the energy, else it continues check the inside victim number and serious victim number to choose one action between two ones: *“goto_hospital”, “goto_victim”*. Moreover, we can obtain easily a rule set from decision tree, for instance:

- IF (emptyEnergy) THEN (goto_fuel)
- IF (Not(emptyEnergy) and (haveSeriousVictims or fullVictims)) THEN (goto_hospital)
- IF (Not(emptyEnergy) and Not(haveSeriousVictims) and Not(fullVictims)) THEN (goto_victim)

The type of decision tree that we use in our approach is the binary tree; each non-leaf node expresses a Boolean function, the left sub-tree of this node corresponds to “true” value of this function and the right sub-tree of this node corresponds to “false” value of the function; each leaf node expresses a decision to make (or an action to execute) of agent.

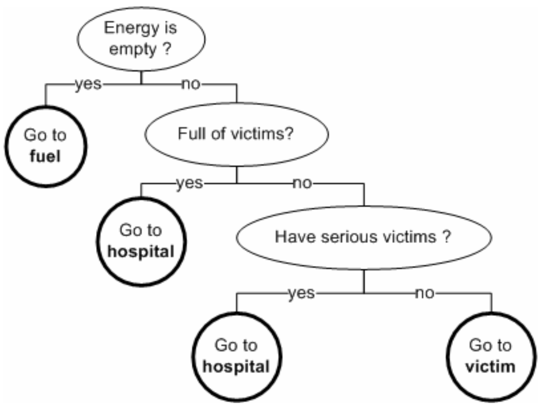
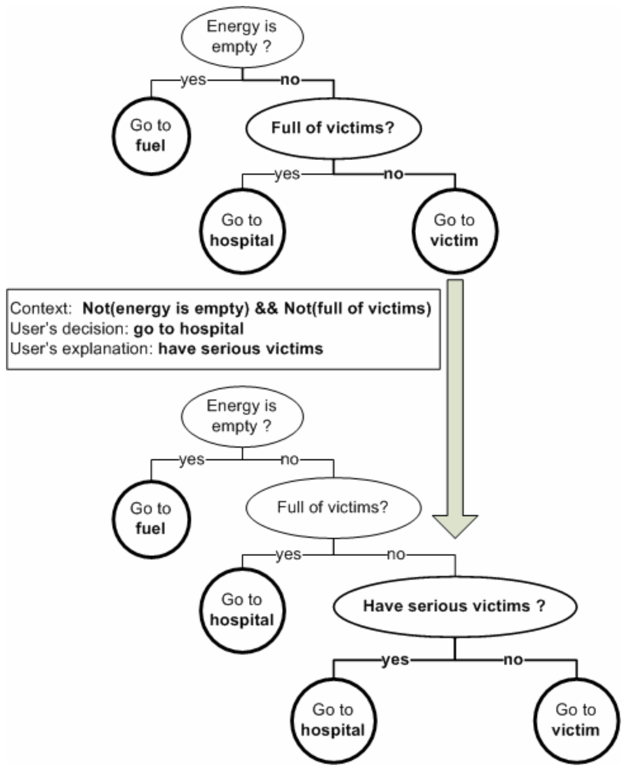


Figure 7: The behaviour of an ambulance is represented by a decision tree



Context: **Not(energy is empty) && Not(full of victims)**
User's decision: **go to hospital**
User's explanation: **have serious victims**

Figure 8: Learning algorithm modifies the structure of the decision tree based on user's decision and explanation

4.2 Learning decision tree with user interaction

The idea behind the learning algorithm, at this level, is the incremental construction of a decision tree for agent through expert's intervention. During the experiments of simulation, if the human expert does not agree with the agent's decision he will make his own decision and express his explanation in the form of a Boolean function. The learning algorithm based on this decision and this explanation to change the structure of decision tree. Figure 8 illustrates an example of a change in learning agent's decision tree with user's decision and explanation.

The change happens when the ambulance's energy level is not empty and the ambulance is not full of victims. The ambulance decides "*goto_victim*" but the expert does not agree with this choice because he prefers that the ambulance *"goto_hospital".* The explanation represented by the Boolean function called "*have serious victims*". So, the algorithm replaces the leaf node "*goto_victim*" by a binary sub-tree beginning with non-leaf node of Boolean function "*have serious victims*", left leaf node "*goto_hospital*" (corresponding to the expert's choice when the Boolean function of expert satisfies) and right leaf node "*goto_victim*" (corresponding to the previous agent's choice when the Boolean function was not satisfied).

4.3 Representation of "which" level by an utility function

For each type of action that will be executed (for example: "goto_victim" of ambulance), the agent has to identify the parameters for this action (which victim in current situation the ambulance needs to go for rescuing). For that reason, an agent uses a utility function for each type of action. The general form of the utility function that we have decided to consider is a weighted linear combination of criteria represented as follows:

$$F(V_k) = \sum w^i * C_k^i \qquad \textbf{(1)}$$

Where V_k is the k^{th} victim; w^i : the weight of the i^{th} criteria and C_k^i: the value of the i^{th} criteria for the k^{th} victim

The victim Vmin will be selected if:

$$F(V_{min}) = \mathrm{Min}_k\{F(V_k)\} = \mathrm{Min}(\sum w^i * C_k^i) \qquad \textbf{(2)}$$

$$V_{min} = \mathrm{ArgMin}\{F(V_k)\} \qquad \textbf{(3)}$$

To calculate the minimum value of the utility function, the value signs of the criteria to maximize are reversed or the weights of these criteria are negative. For example: this is the case for the criteria: *Time-from-victim-to-nearest-other-available-ambulance* in the table 3.

4.4 Learning utility function by incremental linear programming

Decision model for each action type of ambulance is formulated by a utility function; all history of decision situations is stored in set of constraints called "inequalities system" of ambulance. For each time ambulance has a new situation to decide, the constraints number of inequalities system is increased, taking into account new constraints generated from new situation. If the decision on the new situation between expert and ambulance are not the same (we called "conflict decision"), then the new utility function is obtained by resolving updated inequalities system. In fact, set of all coefficients of the utility function is a feasible solution of the inequalities system; we have applied the "Phase I" of simplex method to find this solution. Figure 9 shows the diagram of learning algorithm.

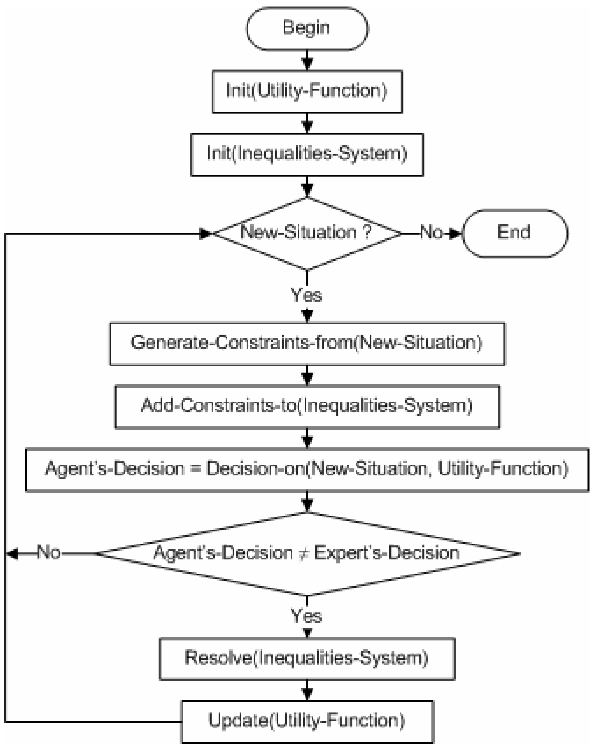

Begin
Init(Utility-Function)
Init(Inequalities-System)
New-Situation ?
No
End
Yes
Generate-Constraints-from(New-Situation)
Add-Constraints-to(Inequalities-System)
Agent's-Decision = Decision-on(New-Situation, Utility-Function)
No
Agent's-Decision ≠ Expert's-Decision
Yes
Resolve(Inequalities-System)
Update(Utility-Function)

Figure 9: Block diagram of learning algorithm

We illustrate more details of this algorithm in following example: to make decision, an ambulance regards to the current situation of the rescue simulation. A situation contains a set of vectors and each vector represents the criteria set of a victim as follows in table 4.

	Victim1	**Victim2**	**Victim3**	**Victim4**	**Victim5**	**Victim6**
C1	18.8	23.77	21.28	20.72	6.45	18.52
C2	330.0	259.0	51.0	16.0	189.0	125.0
C3	9.36	15.96	9.36	4.66	12.74	4.66
C4	0.0	0.0	0.0	0.0	0.0	0.0
C5	3.19	4.97	0.83	8.04	1.62	4.66

Table 4: Example of a situation for ambulance's decision making; the meaning of decision criteria C1, C2, C3, C4, C5 are explicit in table 3

An expert chooses always the victim who optimizes his objective. For example, the situation in table 4, the expert has chosen the *Victim4* as his objective. Thus, the ambulance's utility function, at this time, has to satisfy the following condition:

$$F(V_4) = \mathrm{Min}_k\{F(V_k)\}$$

$$\Rightarrow \quad F(V_k) \geq F(V_4) \quad \textit{for all } k = 1, 2, 3, 4, 5, 6$$

$$\Rightarrow \quad F'(V_k) = F(V_k) - F(V_4) \geq 0 \quad \textit{for all } k = 1, 2, 3, 4, 5, 6$$

$$F'(V_k) = \sum w^i * (C_k^i - C_4^i) \geq 0 \quad \textit{for all } k = 1, 2, 3, 4, 5, 6 \qquad \mathbf{(4)}$$

$$\textit{Where } i = 1, 2, 3, 4, 5$$

We can represent the ambulance's function at time t by a vector F_t with 5 elements $F_t = (w_t^1, w_t^2, w_t^3, w_t^4, w_t^5)$. The initial values may be chosen arbitrary, for example, $F_0 = (1, 0, 0, 0, 0)$.

$$F_0(V_k) = 1*C_k^i \qquad \mathbf{(5)}$$

$$F_0'(V_k) = 1*(C_k^i - C_k^4) = 1*(C'^i_k) \qquad \mathbf{(6)}$$

If the victim V_4 does not minimize its utility function, then the condition (4) is not satisfied for all *k* as illustration in table 5, because all values of $F_0'(V_k)$ are normally positives for all victims.

	Victim1	**Victim2**	**Victim3**	**Victim4**	**Victim5**	**Victim6**
$C'^1 = C^1 - C^4$	-1.91	3.05	0.55	0.0	-14.27	-2.2
$C'^2 = C^2 - C^4$	314.0	243.0	35.0	0.0	173.0	109.0
$C'^3 = C^3 - C^4$	4.69	11.3	4.69	0.0	8.08	0.0
$C'^4 = C^4 - C^4$	0.0	0.0	0.0	0.0	0.0	0.0
$C'^5 = C^5 - C^4$	-4.85	-3.07	-7.21	0.0	-6.42	-3.38
$F_0'(V_k)$	-1.91	3.05	0.55	0.0	-14.27	-2.2

Table 5: Let $C'^i = C^i - C^4$, we have the normalized situation in regarding the expert's decision. Victim4 is choice of expert but it has not optimized utility.

The goal of learning algorithm is to change the current utility function of the ambulance to minimize the expert decision. It means that the weights w_i will be adjusted to adapt to the condition expressed in formula (4). Thus, the algorithm adds to each w^i a value Δw^i as follows:

$$w_{t+1}^i = w_t^i + \Delta w^i$$

$$F_{t+1}'(V_k) = \sum(w_t^i + \Delta w^i) * C'^i_k \geq 0 \quad \textit{for all } k = 1, 2, 3, 4, 5, 6$$

$$F_{t+1}'(V_k) = F_t'(V^k) + \sum \Delta w^i * C'^i_k \geq 0 \quad \textit{for all } k = 1, 2, 3, 4, 5, 6 \qquad \mathbf{(7)}$$

Appling equation (7) for the example in table 5, we obtain the following inequality system:

$$\begin{array}{l} (-1.91) + (-1.91)*\Delta w^1 + 314.0*\Delta w^2 + 4.69*\Delta w^3 + 0.0*\Delta w^4 + (-4.85)*\Delta w^5 \geq 0 \\ (3.05) + (3.05)*\Delta w^1 + 243.0*\Delta w^2 + 11.3*\Delta w^3 + 0.0*\Delta w^4 + (-3.07)*\Delta w^5 \geq 0 \\ (0.55) + (0.55)*\Delta w^1 + 35.0*\Delta w^2 + 4.69*\Delta w^3 + 0.0*\Delta w^4 + (-7.21)*\Delta w^5 \geq 0 \\ (-14.27) + (-14.27)*\Delta w^1 + 173.0*\Delta w^2 + 8.08*\Delta w^3 + 0.0*\Delta w^4 + (-6.42)*\Delta w^5 \geq 0 \\ (-2.2) + (-2.2)*\Delta w^1 + 109.0*\Delta w^2 + 0.0*\Delta w^3 + 0.0*\Delta w^4 + (-3.38)*\Delta w^5 \geq 0 \\ \textit{Where } \Delta w^i \geq 0 \textit{ for all } i=1, 2, 3, 4, 5 \end{array} \quad (\mathbf{8})$$

We can prove that the condition (4) is satisfied if and only if the inequality system (8) has the feasible solution (Δw^1, Δw^2, Δw^3, Δw^4, Δw^5) and vice versa. It means that if we can find a feasible solution for inequality system (8) then we will succeed in correcting the weights of the ambulance's utility function to adapt to the new situation.
To resolve (8) we can apply the "Phase I" of simplex method which is used to find a feasible solution for a linear programming problem [Dantzig and Thapa, 03], [Vanderbei, 08].

By solving (8), we obtain a solution: (Δw^1, Δw^2, Δw^3, Δw^4, Δw^5) = (0, 0.082, 0, 0, 0). Thus, the updated utility function is (1, 0.082, 0, 0, 0) and we have an updated table as follows:

	Victim1	**Victim2**	**Victim3**	**Victim4**	**Victim5**	**Victim6**
$C'^1 = C^1 - C^4$	-1.91	3.05	0.55	0.0	-14.27	-2.2
$C'^2 = C^2 - C^4$	314.0	243.0	35.0	0.0	173.0	109.0
$C'^3 = C^3 - C^4$	4.69	11.3	4.69	0.0	8.08	0.0
$C'^4 = C^4 - C^4$	0.0	0.0	0.0	0.0	0.0	0.0
$C'^5 = C^5 - C^4$	-4.85	-3.07	-7.21	0.0	-6.42	-3.38
$F_1'(V_k)$	23.99	23.1	3.43	0.0	0.0	6.78

Table 6: After update, the utility F' of Victim4 is minimized.

We can see that $F_1'(V_k) \geq 0$ for all k=1, 2, 3, 4, 5, 6. Currently, the updated function is satisfied of the current situation.

For the following decision situation, the ambulance has to choose one optimal victim among m victims. The algorithm will keep all the current constrains and add (m – 1) additional constraints to inequality system. By this way, the algorithm allows the updated function to be always adapted to all historical decision situations.

4.5 Resolving user's contradiction by linear least square method

The simplex method is only useful in the case where experts have their own utility function and they always respect these function when deciding. Nevertheless, it is not very realistic because the choice of human experts is usually emotive and intuitive and they have no explicit function they rely upon during their decision making. They can even contradict themselves from time to time. So, it is quite possible that the inequality system (8) has no solution. In this case, we have to find an approximate solution for the maximum number of inequalities instead of finding an exact solution for all inequalities.

This problem is quite similar to the regression problem, curve fitting or least square problem which aim at finding the best-fitting equation (function) with a minimal deviation from all example data. Unfortunately, we have not yet found an efficient algorithm to resolve this problem.

4.6 Supporting user-defined criteria

Decision making for complex problem needs take into account many criteria; we cannot identify all these criteria at the same time. So, the principle of our approach is to discover incrementally the criteria set for agents with the support of human experts. In the "what" level, the user's explications in form of Boolean function are also the criteria on which the agents based to make decision; these criteria are defined by users following different experiment situations. In the "which" level, the criteria set for each utility function is also able to dynamically increase.

For supporting user-defined criteria, our approach allows users manipulate directly with the modelling language of GAMA [Amouroux, 07], called GAML to define the Boolean criteria for "what" level as well as numeric criteria for "which" level during the experiment scenarios.

In the "what" level, we return to the example of leaning decision tree of ambulance (figure 8); for intervening to ambulance's decision, user have to choose one of many possible actions and for clarifying explication of this choice, user define a Boolean function, such as:

- *Full_of_victims = (length contents = capacity):* Where *"contents"* is list of victims in the ambulance; "*length*" is number of victims in the list; "*capacity*" is maximum number of victims that the ambulance is able to take.
- *Have_serious_victims = (length (contents where each.isSerious) > 0):* Where "*contents where each.isSerious*" is list of serious victims in the ambulance.

In the "which" level, we return to the example of the utility function for action "*goto_victim*". For intervening to ambulance's decision, user want sometimes add a new criteria on which he based to choose one victim, for example:

- *Number_of_near_victims = length (victims where ((each distance_to target) < 50m))*
- *Distance_to_nearest_other_victim = target distance_to (first (victims sort_by (each distance_to target)))*

We can see that GAML play, in this context, the role of query a language to help users describe their criteria; for experimenting various scenarios, users can find sometimes new useful criteria and define them in GAML; agents keep its and extend the criteria set to use in later situations for decision making.

5 Preliminary Results

To experiment with our algorithm, we have used firstly a simple rescue scenario with two ambulances rescuing 100 victims and GIS data of the Ba-Dinh district, Hanoi city. The behaviour of the ambulance at “what level” (represented by a decision tree) is described before experiment and does not change during the experiment; besides, the decision criteria set of ambulances is fixed in this experiment. The goal of our first experiment is to test only the change of utility function at “which level”. We replace the human expert by an agent “oracle”. This agent has an exact utility function called “oracle function”. At each step, the ambulance chooses one on the map to save among many victims by using his utility function. The oracle also chooses by himself his victim. In the case of conflict of decisions (the choice of ambulance is not the same with the oracle’s one), the ambulance has to update its function using the learning algorithm; the objective of the learning process is to make ambulance’s decision conformed to oracle’s one in as many situations as possible. Therefore, we use two parameters to evaluate the algorithm: the first one is the convergence of ambulance’s function to oracle function and the second one is decision conflict rate between ambulance and oracle.

5.1 Difference between agent function and oracle function

The difference between two functions is calculated by comparing two weight series of two functions by the following formulate:

$$Diff(k_{min}) = \sum | a^i - k_{min} * w^i | \quad \text{with } k_{min} = ArgMin\{Diff(k)\}$$

Where a^i are coefficients of oracle function: $Fo(V_k) = \sum a^i * C_k^i$
And w^i are coefficients of ambulance’s function: $Fa(V_k) = \sum w^i * C_k^i$

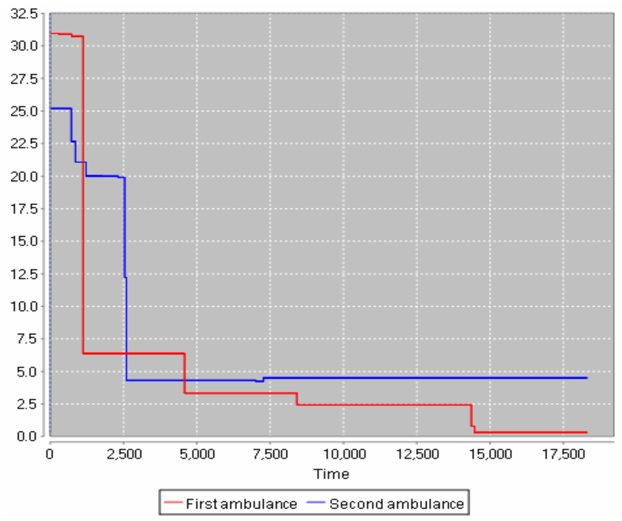


Figure 10: The difference between the function of the ambulance and of the oracle

In figure 10, each step of the curve corresponds to a time of decision conflict when the ambulance function has to be updated and the difference between two functions has to be reduced. The reduction of this difference during the simulation show that the ambulance's function converges slowly to oracle function. For that reason, the more the ambulance learns the more its function is similar to the oracle function.

5.2 Conflict between agent decision and oracle decision

Decision of ambulance and oracle are the same in most situations. However, the ambulance's decision is different from the oracle's one in some cases. These cases are called "conflict" of decision between ambulance and oracle. The rate of conflict is calculated by percentage of situations having conflict in total number of decision situations. In figure 11, we observe that the rate of conflict of decision reduces during the simulation. It means that the more the ambulance learns the more its decision converges to the decision of the oracle. Finally, all victims are saved and the number of decisions is 100 and the number of conflicts is x.

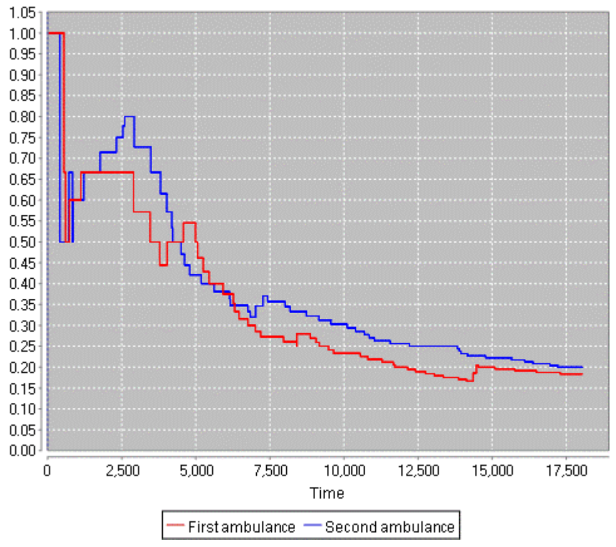


Figure 11: Rate of conflict decision between ambulance and oracle

6 Conclusions

Urban disaster management is an extremely complex problem. In this context, we are particularly interested in the use of information technology, GIS and ABM tools to address the vital problem of resource allocation for disaster response activities. Our research is intended to provide means for building efficient decision support systems that would be easily usable by non-computer scientists.

There are several research projects (e.g. RoboCupRescue Simulation, LaSER Project) which attempt to address similar questions relying on multi-agent models to optimize the planning of rescue teams, the combination of partial GIS and planning methods, and so on. However, there are only a few works that take into account the human (and subjective) aspects of decisions made during the disaster response.

In proposing a method enabling experts to interact directly with the agents of the simulation to teach them "good" behaviours, we hope to (1) improving the realism of these simulations (and thus improve the strategies that can be learnt/proposed), (2) increasing the confidence of decision-makers in the supporting decision tools; (3) facilitating the training of the same decision-makers to these tools.

7 Future Works

In solving multi-criteria decision making problem, by using additive utility function and supervised learning algorithm we support the elicitation human expert's preferences while taking into account their independent decision criteria. This method allows decision making of agents conforming to decision strategy of users in decision support system. However, we have used some assumptions to reduce the complexity of the problem. These assumptions show some limitations which need to be overcame.

Firstly, decision criteria are assumed preferentially independent; this is a necessary condition for using additive utility function; but in many cases, decision criteria are not preferentially independent. Future work include extending our algorithm with some more sophisticated preference models like CP-Nets (Conditional Preference Networks) [Boutilier, 04], GAI-Nets (Generalized Additive Independence Networks) [Gonzales and Perny, 04], CUI-Nets(Conditional Utility Independence Networks) [Engel and Wellman, 06.

Secondly, in experimenting of our work, we use a simulated expert called oracle function to train the agent. So, the preference of oracle is always exact as it is a formula. In fact, it is very different from real situations where the emotion and intuition of human expert play a role in the decision. A second perspective is to identify another learning algorithm that do accommodate for expert's inconsistencies.

Acknowledgements

We are very grateful to our partners, and particularly to Pr. Nguyen Hong Phuong (Institute of Geophysics, Vietnamese Academy of Science and Technology – Hanoi, Vietnam), for their advices on the input data for our model.

References

[Amouroux, 07] Amouroux, E., Chu, T.Q., Boucher, A., Drogoul, A. : GAMA: an environment for implementing and running spatially explicit multi-agent simulations, Proceedings of the 10th Pacific Rim International Workshop on Multi-Agents (PRIMA), Bangkok, Thailand, 2007.

[Bacchus and Grove, 95] Bacchus, F. and Grove, A.: Graphical models for preference and utility, Eleventh Conference on Uncertainty in Artificial Intelligence, Montreal, 1995, 3-10.

[Boutilier, 01] Boutilier, C., Bacchus, F. and Brafman, R. I.: UCP-networks: A directed graphical representation of conditional utilities, Seventeenth Conference on Uncertainty in Artificial Intelligence, Seattle, 2001, 56-64.

[Boutilier, 05] Boutilier, C., Patrascu, R., Poupart, P. and Schuurmans, D.: Regret-based utility elicitation in constraint-based decision problems, Proceedings of the International Joint Conference on Artificial Intelligence, 2005, 929-934.

[Boutilier, 04] Boutilier, C., Brafman, R. I., Domshlak, C., Hoos, H. H. and Poole D.: CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements, Journal of Artificial Intelligence Research, volume 21, 2004.

[CPC, 07] Committee on Planning for Catastrophe: Successful Response Starts with a Map: Improving Geospatial Support for Disaster Management, A Blueprint for Improving Geospatial Data, Tools, and Infrastructure, National Research Council, Washington, D.C, 2007.

[Dantzig and Thapa, 03] Dantzig, G. B. and Thapa, M. L.: Linear Programming 2: Theory and Extensions, Springer Series in Operations Research, 2003.

[Engel and Wellman, 06] Engel, Y. and Wellman, M. P.: CUI Networks: A Graphical Representation for Conditional Utility Independence, Proceedings of The Twenty-First National Conference on Artificial Intelligence and the Eighteenth Innovative Applications of Artificial Intelligence Conference, Boston, Massachusetts, USA, 2006.

[Gonzales and Perny, 04] Gonzales, C. and Perny, P.: GAI networks for utility elicitation. Ninth International Conference on Principles of Knowledge Representation and Reasoning, Whistler, BC, Canada, 2004, 224-234.

[Keeney and Raiffa, 76] Keeney, R. L., and Raiffa, H.: Decisions with Multiple Objectives: Preferences and value tradeoffs, Cambridge University Press, 1976.

[Li Chen and Pearl Pu, 04] Li Chen and Pearl Pu: Survey of preference elicitation methods, Technical Report IC/2004/67, Swiss Federal Institute of Technology in Lausanne (EPFL), 2004.

[Narzisi, 06a] Narzisi, G., Mysore, V., Nelson, N., Rekow, D., Triola, M., Halcomb, L., Portelli, I. and Mishra, B.: Complexities, Catastrophes and Cities: Unraveling Emergency Dynamics, Proceedings of the 6th International Conference on Complex Systems (ICCS), Boston, MA, 2006.

[Narzisi, 06b] Narzisi, G., Mysore, V. and Mishra, B.: Multi-Objective Evolutionary Optimization of Agent-based Models: An Application to Emergency Response Planning, Proceedings of the Second IASTED International Conference on Computational Intelligence, San Francisco, California, USA, 2006, 228-232.

[Narzisi, 07] Narzisi, G., Mincer, J. S., Smith, S.and Mishra, B.: Resilience in the Face of Disaster: Accounting for Varying Disaster Magnitudes, Resource Topologies, and (Sub) Population Distributions in the PLAN C Emergency Planning Tool, Holonic and Multi-Agent Systems for Manufacturing (LNCS, volume 4659. Springer, Heidelberg), 2007, 433-446.

[Nguyen-Duc, 07] Nguyen-Duc, M., Drogoul, A.: Using Computational Agents to Design Participatory Social Simulations, Journal of Artificial Societies and Social Simulation vol. 10, no. 4 5, 2007.

[Paquet, 06a] Paquet, S.: Distributed Decision-Making and Task Coordination in Dynamic, Uncertain and Real-Time Multiagent Environments, PhD thesis, Faculté de Sciences et Génie, Université Laval, Québec, 2006.

[Paquet, 06b] Paquet, S., Chaib-draa, B. and Ross, S.: Hybrid POMDP Algorithms, Proceedings of The Workshop on Multi-Agent Sequential Decision Making in Uncertain Domains (MSDM), Hakodate, Hokkaido, Japan, 2006.

[Paquet, 05a] Paquet, S., Bernier, N. and Chaib-draa, B.: Multiagent Systems Viewed as Distributed Scheduling Systems: Methodology and Experiments, Proceedings of the 18th Canadian Conference on Artificial Intelligence (AI'2005), Victoria, Canada, 2005.

[Paquet, 05b] Paquet, S., Bernier, N. and Chaib-draa, B.: An Online POMDP Algorithm for Complex Multiagent Environments, Proceedings of The 4th International Joint Conference on Autonomous Agents & Multi Agent Systems (AAMAS'2005), Utrecht, The Netherlands, 2005, 970-977.

[Paquet, 04a] Paquet, S., Bernier, N. and Chaib-draa, B.: Comparison of Different Coordination Strategies for the RoboCupRescue Simulation. Proceedings of The 17th International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems, LNAI 3029, Springer-Verlag, Ottawa, Canada, 2004, 987-996.

[Paquet, 04b] Paquet, S., Bernier, N. and Chaib-draa, B.: Multi-Attribute Decision Making in a Complex Multiagent Environment using Reinforcement Learning with Selective Perception. Proceedings of the 17th Canadian Conference on Artificial Intelligence, (AI'04), London, Ontario, Canada, 2004, 416-421.

[Rao, 07] Rao, R. R., Eisenberg, J. and Schmitt, T.: Improving Disaster Management: The Role of IT in Mitigation, Preparedness, Response, and Recovery, Committee on Using Information Technology to Enhance Disaster Management, National Research Council, Washington, D.C, 2007.

[Satty, 80] Satty, T. L.: The Analytic Hierarchy Process, McGraw-Hill International, New York, USA, 1980.

[Satty, 94] Satty, T. L.: Fundamentals of Decision Making and Priority Theory with the AHP, RWS, Publications, Pittsburgh, PA, USA, 1994.

[Vanderbei, 08] Vanderbei, R. J.: Linear Programming: Foundations and Extensions, 3rd Edition, International Series in Operations Research & Management Science, 2008.

[Von Stengel, 88] Von Stengel, B.: Decomposition of multi-attribute expected utility functions, Annals of Operations Research, Volume 16, 1988, 161-184.

[Wellman and Doyle, 92] Wellman, M. P. and Doyle, J.: Modular utility representation for decision-theoretic planning, First International Conference on Artificial Intelligence Planning Systems, College Park, MD, 1992, 236-242.