PERFORMING THE RETRIEVE STEP IN A CASE-BASED REASONING SYSTEM FOR DECISION MAKING IN INTRUSION SCENARIOS

Jesus Conesa and Angela Ribeiro
Instituto de Automatica Industrial, CSIC, Arganda del Rey, Madrid, Spain

Keywords: Case-based reasoning, Similarity function, decision tree, Cross-validation.

Abstract: The present paper describes implementation of a case-based reasoning system involved in a crisis management project for infrastructural building security. The goal is to achieve an expert system, capable of making decisions in real-time to quickly neutralize one or more intruders that threaten strategic installations. This article presents development of usual CBR stages, such as case representation, retrieval phase and validation process, mainly focusing on the retrieving phase, approaching it through two strategies: similarity functions and decision tree structures. The designing case, such as the discretization values that are adopted, will also discussed. Finally, results on the retrieving phase performance are shown and analyzed according to well-known cross-validations, such as k-validations or leave-one-out. This work is supported by project CENIT-HESPERIA.

1 INTRODUCTION

Due to the sensitive international situation caused by the still recent terrorist attacks, there is a common need to protect the safety of great spaces like government buildings. To support these kind of problems a CBR system has been designed which represents the decision making core of the security system under development, hence it will make decisions concerning several scenarios in real-time. The intrusion evaluation process and recovery of a solution that minimizes the consequences of the attack are both included within management of infrastructure safety. It is in this context that the proposed CBR will perform. Reaction time is an important parameter for these situations and, therefore, it would be very interesting to determine a solving action that can help the expert take the best decision process so speed intervention can be increased. This is the main reason for including a decision support system, in this case a CBR. In that concerning to the chosen CBR approach, the same allows identifying and analysing similar previous intrusion scenes (cases) and applying the associated solution to similar new cases or, even, construct a new solution from the previous one that is already stored. Finding the case that is most similar (from amongst all those contained in the cases database) to a new and unknown intrusion scene (case) is not an easy task, which is included in the CBR cycle (Lopez de Mantaras, 2006). For the retrieval phase, two approaches are developed in the present work: the first applies a similarity function and the second constructs a decision tree that groups information according to common characteristics, preparing it for a classification process. A knowledge base containing a set of cases with their associated solutions or actions for each one of them is referenced alongside this article. In the case of a new situation, the solution or action of the most similar case that is stored is copied in the reuse CBR phase, so it is in the retrieval phase where the adaptation process is focused.

2 CBR IMPLEMENTATION

2.1 Case Representation

The first implementation step is to define the case structure. Case representation can be viewed as the task of enabling the computer to recognize, store, and process past contextualized experiences (Shiu, 2004). These are those scene parameters that are, in general, inspected by an expert to decide from amongst the several actions that are possible. This subject represents a very important point because the chosen structure for a case will strongly determine the entire system design.

Conesa J. and Ribeiro A. (2009).
PERFORMING THE RETRIEVE STEP IN A CASE-BASED REASONING SYSTEM FOR DECISION MAKING IN INTRUSION SCENARIOS.
In *Proceedings of the 11th International Conference on Enterprise Information Systems - Artificial Intelligence and Decision Support Systems*, pages 343-346
DOI: 10.5220/0001996403430346

The initial requirements are the map of the building under surveillance, intruder locations, guards, exit and sensitive zone (in all probability the intruder target). Spatial representation of a possible scenario appears in figure 1. The red lines represent the bounds of constraint, such as obstacles, walls, closed doors. Since it is a map with obstacles, the well-known problem of finding an optimal path is certainly not trivial one. Our approach to resolve the path planning is based on a representation of the space known as Delaunay triangulation. For more details, see (Anglada, 1997).

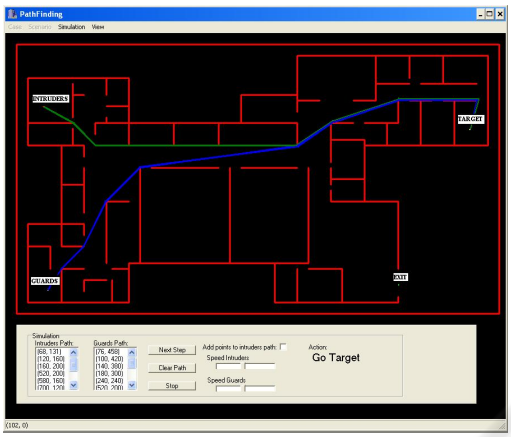

Figure 1: Shortest paths between intruders and target and between guards and target.

The figure 1 have been taken from the simulator implemented to test the developed CBR system. The optimal (shortest) paths between intruders and target and between guards and target are drawn in red and blue respectively. Information about the alarm system is also available. This device can be in three different states: On (ringing), Off and Not Available. The direction of the intruders is supposedly detected at each time by a wireless sensor network integrated in the scenario and the intruder behaviour could be predicted, that is, if intruders are going to a certain target or if they are going to the exit. The number of intruders and guards are also considered. Likewise distances corresponding to the different places are also available. Based on this information, a natural abstraction for an intrusion scene could as follows:

$$< ratio, state, door, alarm, dIT, dIE, dGI, dGT, dGE, opSol >$$

where ratio is guard number/intruder number; state is intruder direction/purpose; alarm is alarm state; dIT is distance (intruders,target); dIE is distance (intruders,exit); dGI is distance (guards,intruders); dGT is distance(guards,target); dGE is distance (guards,exit); opSol is Solution or action to apply.

Table 1: Values for variables in the characteristic vector of a scene.

Property	Value
ratio	LESS ‖ EQUAL ‖ MORE
state	TARGET ‖ EXIT
alarm	NOT_AVAILABLE ‖ OFF ‖ ON
distances	CLOSE ‖ MEDIUM ‖ FAR ‖ NOT_REACHABLE
opSol	GO_EXIT ‖ GO_TARGET ‖ TURN_ON_ALARM ‖ CAPTURE ‖ GO_INTRUDERS

2.2 Value Discretization

Property values corresponding to the components / variables of the characteristic vector that has been previously defined are listed in table 1.

Distances are calculated over the triangulation: a spatial representation of the map (Anglada, 1997). Its discrete value is set according to the size of the own map. Given two places on the space, if there is no path that connects them, then the value assigned is *NOT_REACHABLE*. If a path exists and if it is shorter than 1/3 of the diagonal map, then the *CLOSE* value is assigned. For more than 1/3 and less than 2/3 the value is *MEDIUM*, while in other cases, *FAR* is set.

Taking into account that outlined so far, the scene in figure 1 would be represented by the following characteristic vector:

$$< EQUAL, TARGET, ON, MEDIUM, CLOSE, MEDIUM, MEDIUM, MEDIUM, UNDEFINED >$$

Please note that solution-property (opSol) is set to UNDEFINED because this is the default value until the case can be evaluated by the expert or by the CBR system in the retrieval phase.

2.3 Similarity Function

Establishing an appropriate similarity function is an attempt at handling the deeper or hidden relationships between the relevant objects associated with the cases. There are two major retrieval approaches (Liao, 1998). The first is based on the computation of distance between cases, where the most similar case is determined by evaluation of a similarity measure. The second approach is related to indexing structures and our approach is described in the following section. At any rate, the global similarity function should check all the variable values and combine them through local similarity results to obtain a global similarity measurement. In other words, given cases A and B, the

function could be expressed as:

$$SIM(A,B) = \frac{1}{p}\sum_{i=1}^{n}\frac{1-|ord(a_i)-ord(b_i)|}{card(O)} \quad (1)$$

In which p is the number of compared attributes In which *card(O)* is the cardinality of the set of possible category values for a and b and *card(O)* is the cardinality of the set of possible category values. In this first approach, the weight coefficient is the same for all the properties *(1/p)*. It is important to highlight that the similarity function coefficients or weight are not adjusted to the knowledge base, thus the global function does not measure properly, that is, according to its true weight and each local characteristic. Each property bears the same importance to discern the cases. To resolve this matter, another strategy is applied to the similarity, such as inductive reasoning with decision trees discussed in the next section.

2.4 Decision Tree

A decision tree is used to classify the cases in order to their common features. Each node of the tree stores one of the properties of the characteristic vector and each arc covers one of the possible alternative values for the property(Buckinx04). One of the most well-known algorithms is ID3, and that is the proposed approach. For the selection of the best attribute, the entropy and gain concepts defined in (Mitchell, 1997) will be used. It is interesting to verify that the solution-attribute is not a concept capable of holding two values (positive and negative), and even more, its cardinality is 5 (please see table 1). It should be highlighted that if there is no case associated to the final node, the solution that is returned is the most usual one on the parent node. Since the final node does not reference any case, the need for another method to retrieve a proper solution arises, and since it keeps common characteristics in that regarding nodes at the same level, it is natural to select the most common solution of them.

3 RESULTS

Results for the two retrieval process approaches are detailed and compared in this section. The measurement that determines performance of the retrieval phase is the number of properly recovered cases per the total number of cases that are stored in the cases database. Different cross validations that modify the number of partitions from the examples space have been applied to check these results. In each contrast operation, one of the example sets is selected to test the system and those remaining are used as training sets for the CBR.

Parameter K specifies the number of partitions over the examples space. It should be highlighted that in $K = N$ (N is the total number of examples), cross validation is known as leave-one-out validation. For K = 10 the validation seems to be especially accurate (Kohavi, 1995).

In rows "F" and "F*" (Function) corresponding to table 2, the number of well retrieved cases with a similarity function (and its percentage) is displayed for a knowledge case base without and with noise respectively.

In row "T" and "T*" (Tree) the cases that are well recovered by the decision tree are displayed for knowledge case base without and with noise respectively.

Table 2: Data without noise F and T rows and with noise F* and T* rows . NC=Number of Cases.

NC	10(%)	30(%)	50(%)	70(%)	100(%)
K = 5					
F*	4(40)	14(46)	27(54)	52(74)	74(74)
F	6(60)	15(50)	31(62)	54(77)	80(80)
T*	2(20)	18(60)	31(62)	50(71)	82(82)
T	3(30)	18(60)	33(66)	55(79)	86(86)
K = 10					
F*	2(20)	13(43)	25(50)	51(73)	72(72)
F	5(50)	14(47)	29(58)	53(76)	78(78)
T*	1(10)	16(53)	32(64)	55(79)	80(80)
T	3(30)	17(57)	35(70)	60(86)	85(85)
K = Number of cases					
F*	2(20)	13(43)	25(50)	50(71)	72(72)
F	5(50)	14(47)	29(58)	52(74)	78(78)
T*	1(10)	13(43)	31(62)	48(69)	83(83)
T	3(30)	13(43)	35(70)	53(76)	88(88)

In the figure 2a, it is shown the better decision tree performance for high numbers of examples. Since 50 cases, there is enough stored knowledge to classify properly a new case with the decision tree. For smaller examples sets, the accurate with the similarity function is close to the performance of the decision tree, even improves it for some cases.

In that pertaining to the results outlined in figure 2b, the first considerations may assume a stable number of proper recoveries, however, the results highlight a slight increase according to the number of cases, exactly the same as in the decision tree. The reason for this is that all the variables have an important role and no particular variable dominates the rest. Likewise, there is no redundancy among them. Also, there is more diversity and retrieving cases without similarity to the queried one is more difficult.

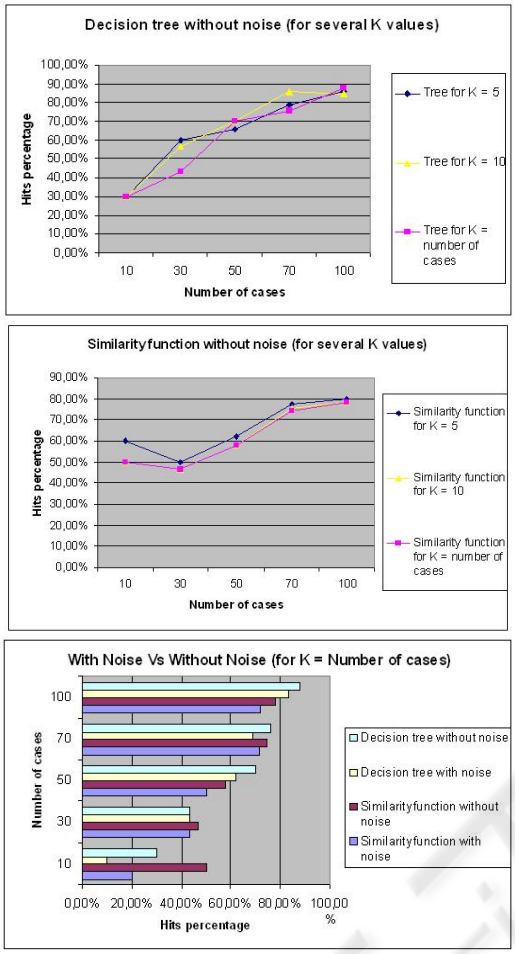

Figure 2: (a) Results without noise for the decision tree (b) Results without noise for the similarity function (c) Results without noise and with noise for K = number of cases.

Also, in figure 2c what is highlighted is that noise negatively affects the system. For said data the system was performed with 3 improperly classified cases for each knowledge base that was used. All the results that were obtained were worse, in particular for small-sized knowledge bases (number of cases = 10), with the reason being that the ratio for incorrectly classified examples is larger. Even so an appreciable hit (accurate decreases 3%) can be discerned in big knowledge bases ($numberofcases = 100$).

4 CONCLUSIONS

The intrusion evaluation process and recovery of a solution that minimizes the attack consequences are included within management of the safety infrastructure. A CBR is proposed as the decision making core of the security system in this context. The performed system offers the following advantages: a) The system is able to learn complex rules with a rather small number of training examples; b) It can achieve good performance with a well-selected training set; c) The decision tree achieves more complex rules than similarity functions; d) The gain information measurement reduces the decision tree depth and this allows improved translation from the decision tree structure to a rules-based structure. In order words, the learnt knowledge can be expressed as a set of rules. Finally the decision tree has nodes without any references to cases. Hence, it is possible that no similar past experience to the one queried can be found and, therefore, any solution that matches. This situation is resolved in the current approach by returning the most common solution that is present on the nodes at the same level (a context that is more similar to the queried case). However, this is a solution based on statistics, which could be slightly wrong in some cases. Future versions should deal with these kinds of nodes.

REFERENCES

Lopez de Mantaras, R., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M., Cox, M., Forbus, K., Keane, M., Watson, I., 2006. *Retrieval, Reuse, Revision, and Retention in CBR*. The Knowledge Engineering Review, vol. 20, no. 3, pp. 215-240

Shiu, S., Pal, S. K. , 2004. *Foundations of Soft Case-Based Reasoning*. John Wiley & Sons.

Liao, T. W., Zhang, Z., Mount, C. R., 1998. *Similarity measures for retrieval in case-based reasoning systems*. Applied Artificial Intelligence, vol. 12, pp. 267-288.

Mitchell, T., 1997. *Machine Learning*. McGraw-Hill.

Anglada, M. V., 1997. *An Improved Incremental Algorithm for Constructing Restricted Delaunay Triangulations*. Computer & Graphics, 21(2):215-223.

Buckinx, W., Moons, E., Poel, D. V. D., & Wets, G., 2004. *Customer-adapted coupon targeting using feature selection*. Expert Systems with Applications, 26, 509-518.

Kohavi, R., 1995. *A study of cross-validation and bootstrap for accuracy estimation and model selection*. Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence 2(12): 1137-1143. (Morgan Kaufmann, San Mateo).