Involving Fuzzy Concepts in Active Mobile Databases*

Yücel Saygın, Özgür Ulusoy

Department of Computer Engineering and Information Science,
Bilkent University, Ankara 06533, TURKEY

Abstract. Current needs of industry required the development of advanced database models like mobile databases, active databases, and fuzzy databases. Fuzzy concepts are adapted to the field of databases in order to deal with ambiguous, uncertain data. Fuzziness comes into picture in mobile databases especially with moving objects. Incorporating fuzziness into rules would improve the effectiveness of active databases. Rules can be utilized in mobile databases to form a more powerful system, namely an active mobile database. In this paper we investigate the ways in which the concepts developed for fuzzy systems can be integrated to active mobile databases.

Key words: Active databases, mobile databases, rule execution, fuzzy databases, fuzzy triggers, fuzzy rule execution.

1 Introduction

Current needs of industry and business necessitated the development of advanced database models which are used to store and retrieve data in complex systems. Three of those advanced databases are of particular interest to us, namely mobile databases, active databases, and fuzzy databases. In this paper we propose an active mobile database platform and investigate the ways in which the concepts developed for fuzzy systems can be adapted to this platform. As a base for our investigation, the concepts of mobile, active and fuzzy databases are introduced in the following paragraphs, together with a description of the route which we take to achieve the proposed work.

Fuzzy concepts are incorporated to the field of databases in order to support queries closer to the natural language and to model data which is inherently fuzzy. An *active database management system (ADBMS)* allows users to specify actions to be executed when specific events are signaled [Day88]. In order for a conventional database management system to react to certain events, it should be incorporated with *rules*. Mobile database research aims to provide efficient access to data on both stationary data servers and mobile computers.

* This research is supported by the Research Council of Turkey (TÜBİTAK) under grant number EEEAG-246 and the NATO Collaborative Research Grant CRG 960648.

An *active mobile database* can be designed by incorporation of rules into a mobile database environment. We use in this paper an active mobile database platform to explain how fuzzy features can be integrated to active and mobile database systems. We adopt a battlefield environment to illustrate how the proposed approaches can be made use of in real applications. The primary contribution of our work is the incorporation of fuzziness into rule execution via fuzzy coupling modes and scenarios. We also describe how fuzzy primitive events can be combined to form fuzzy composite events, and how fuzzy rules and other fuzzy concepts can be utilized in MDBSs.

In the next section, an introduction to *fuzzy databases* is provided. Section 3 presents a mobile database system model that is supported with rules and fuzzy queries. In Section 4, a description of the current work on fuzzy triggers is provided together with our contributions. Finally in Section 5, conclusions and future work are discussed.

2 Fuzzy Databases

Uncertain nature of queries and real-life data has necessitated the development of *fuzzy databases*. Fuzzy database theory is based on the concepts of *fuzzy sets* and *fuzzy logic* which we discuss in the following.

The theory of fuzzy sets was introduced by Zadeh [Zad65]. For a crisp set (an ordinary set that we are familiar with) S, which is a subset of the universal set U, for any element $e \in U$, either $e \in S$ or $e \notin S$ where for a fuzzy set there is a degree of membership in the range $[0, 1]$ for each element belonging the the universal set. Crisp set theory is a special case of the fuzzy set theory where the membership degrees of any element belonging to the universal set is either 0 or 1. A fuzzy set is characterized by its membership function. This membership function, gives us the degree of membership of each element in the universal set to the fuzzy set. Membership function of a fuzzy set F on the universal set U is generally denoted by μ_F and maps each element $x \in U$ to a real number in the range $[0, 1]$, i.e.,

$\mu_F(x) : U \rightarrow [0, 1]$.

The fuzzy set theory is best understood with real life examples. Assume that we have a universal set U for all the ages a human being can have. We can define a fuzzy set *young* denoted by Y on U, and assign a membership function μ_Y to Y. A sample membership function can be defined as in Figure 1.

Similarly, fuzzy logic is an extension of crisp logic and based on fuzzy theory. Using fuzzy logic, we can reason about the degree of truth of imprecise propositions.

Fuzzy database models generally adopt the relational database model as their base since it has a well established theoretical framework. One approach by Buckles and Petry incorporates fuzziness into relational database model by replacing the ordinary equivalence relation by the notion of similarity [BP82]. Similarity relationships are constructed to allow the comparison of linguistic terms like

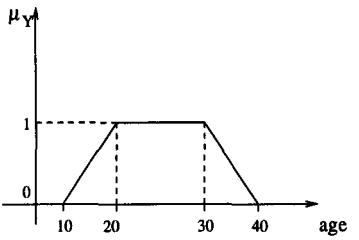

Fig. 1. Membership function of the fuzzy set young.

young, old, middle-aged in a particular domain. Similarity thresholds are explicitly defined in the queries in order to retrieve tuples in a certain similarity range.

3 Mobile and Active Databases in a Common Platform

There is a wide spectrum of applications of active mobile database management systems (AMDBMSs) from military to health and insurance. One such application in military is the management and control of vehicles in a battlefield environment [MB97], [MBM96]. In health, an active mobile computing system can be designed to reach the patients' previous records in the hospital from the moving ambulances [MB96].

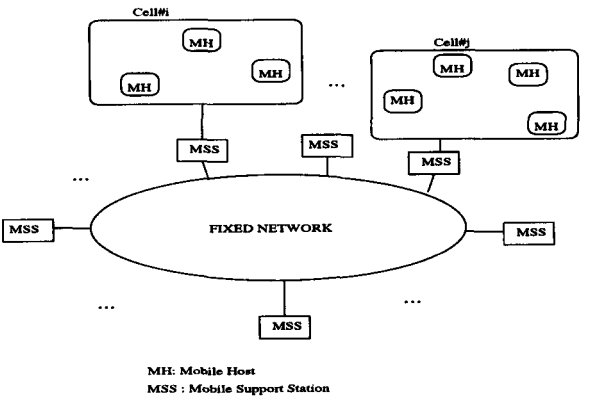

Fig. 2. A general architecture of a mobile computing system

A typical architecture for mobile computing systems which is inspired from [IB94] is depicted in Figure 2. In this architecture, there is a fixed network of Mobile Support Stations (MSSs). Mobile Hosts (MHs) are the computers which are portable and capable of mobile communication. Each MH is associated with a

MSS and MHs are connected to MSSs via wireless links. A MSS is generally fixed and it provides MHs inside a prespecified area called a cell with a wireless interface.

As an example application, a battlefield environment can be coordinated using a system based on the architecture provided in Figure 2 where the vehicles on land and aircrafts are moving objects which are also capable of issuing queries, i.e., they are MHs. In the fixed network there is a database management system supported with rules like:

- **event:** obj_1 is very close to obj_2
  condition: obj_1 belongs to enemy and obj_2 belongs to the alliances
  action: fire an alarm and inform obj_2
- **event:** send missile
  condition: there are objects close to the target that belong to the alliances
  action: move away those objects

An obvious property of the rules listed above is that they are close to natural language, and therefore very easy to write for the experts of war scenarios who are not much familiar with data management issues. These rules involve fuzzy queries on the database and some of them have fuzzy events.

4 Fuzzy Rules in Active Databases

Although incorporating fuzziness to active databases introduces much flexibility, not much attention has been paid so far to this issue. To the best of our knowledge, only a research group in VTT (Finland) has worked on fuzzy triggers [BW96], [BW97], [BKPW97]. In [BKPW97], a Condition-Action(CA) fuzzy trigger is proposed which means that fuzziness is introduced to the CA part of an ECA rule. In a later work [BW97], the concept of CA trigger is extended to a fuzzy ECA rule by introducing the notion of fuzzy events. A CA fuzzy trigger consists of a fuzzy predicate (i.e., a predicate that has linguistic hedges) on the database as its condition, and a fuzzy action where updates are performed on a fuzzy database. A rule with a fuzzy condition and a crisp (i.e., not fuzzy) action is called a C-fuzzy trigger. The C-fuzzy trigger model is based on linguistic hedges. C-fuzzy trigger is a rule which has a fuzzy predicate as its condition. Max-min inference method is applied to the rule set to determine the truth value of the fuzzy predicates. In fuzzy ECA rules, an event may fire a set of rules. Fuzzy events are defined as fuzzy sets and use linguistic hedges like *high, low,* and *strong* [BW97]. Formally a primitive fuzzy event is represented as a tuple $< e_c, e_f >$ where e_c is a crisp event, and e_f is a fuzzy event predicate. When a crisp event is signaled (such as a database update), the current value v produced upon the operation causing the crisp event is fed into the membership function of e_f. The output of the membership function is called the event match factor, and the fuzzy event is signaled only if the event match factor is greater than zero [BW97]. Upon the signaling of the fuzzy event, the corresponding rules are fired and their conditions (which are fuzzy predicates on the database) are

checked. The action of a rule may be started to execute depending on the result of condition evaluation.

The methods discussed in [BW97] and [BKPW97] try to incorporate fuzziness into the event and condition of a rule. In this paper, we deal with the rules belonging to particular fuzzy sets, together with the coupling modes and scheduling of rules. The rest of this section is devoted to the detailed discussion of our approach.

4.1 Fuzzy Events

Different events and their categorization together with composite events are explained in [GGD94], [CM91], and [Buc94]. Among these references, the most comprehensive event set and composition semantics are provided in [Buc94]. Primitive events are categorized in [Buc94] as: method execution events, state transition events, temporal events, transaction and flow-control events, and abstract events. Method execution events are raised when the specified methods are executed. State transition events are signaled when the corresponding state changes occur in the database, for example location updates of moving objects. Temporal events are either absolute or relative. An absolute temporal event is something like, "at 13:45", and a relative temporal event is like "5 seconds before the firing of a missile". Transaction and flow-control events are related with the beginning, commit and abort of transactions. Finally, abstract events are defined by user and therefore signaled explicitly by the user.

Events which are important from the point of fuzzy rule execution are state transition events, and temporal events. There is a high level of potential for incorporating fuzzy concepts into those kinds of events.

Temporal events are widely used in many active database systems and can be applied to critical jobs in real time systems. Fuzzy concepts can be incorporated to temporal events by adding fuzzy modifiers to exact time values. For example, instead of the absolute temporal event, "at 13:43", we may have a fuzzy absolute temporal event like, "at about 13:43" which is more flexible. Relative temporal events can also be modified in order to convert them to fuzzy relative temporal events. Calculating the membership degrees of fuzzy temporal events can be done using the membership functions of the fuzzy terms and the concept of fuzzy numbers which is explained in more detail in [KF88]. Membership degree of crisp events is taken as one.

Primitive events can be combined to form composite events. Composition of primitive events can be done with different event constructors, like conjunction, disjunction, closure, sequence, history, and negation [Buc94], [GGD94], [CM91]. Disjunction of two events E_1 and E_2 is raised when one of E_1, or E_2 is raised. Conjunction of two events E_1 and E_2 is raised when both E_1 and E_2 have occurred, regardless of the order of occurrence. Sequence is similar to conjunction but the order of occurrence of the events is important with sequence. Closure constructor is used when multiple occurrences of the same event in a period of time (such as, during the execution of a transaction) is considered together as a composite event. History event constructor is a more restricted case of the

closure event constructor where the number of occurrences of the same event is specified. Negation of an event can also be considered as a composite event and it is raised when the negated event has not occurred in a specified period of time. Events composed by multiple event constructors are composite events as well.

Fuzzy composite events can be constructed by combining crisp primitive events listed above and fuzzy primitive events (i.e., fuzzy temporal, fuzzy state change, and fuzzy method execution events). The membership values of fuzzy composite events can be calculated depending on the semantics of the event constructors. In case of the conjunction event constructor, the event with minimum membership degree among the component events is selected, and its membership degree determines the membership degree of the composite event. When disjunction is used as the event constructor, then the maximum membership value among the membership degrees of the component events determines the membership degree of the composite event. In case of negation, the membership degree, μ_n, of the composite event is calculated as,

$$\mu_n = 1 - \mu_e$$

where μ_e is the membership degree of the event being negated.

Computation of the membership degrees of composite events constructed by history and closure is done by using the following formula:

$$\mu_c = \frac{\sum_{i=1}^{i=n} \mu_{e_i}}{n}$$

where μ_c is the membership degree of the composite event, μ_{e_i} is the membership degree of the i^{th} occurrence of event e, and n is the number of occurrences of event e. Here we should note that different occurrences of the same event may result in different membership degrees depending on the crisp parameter of the event. Membership degrees of the composite events formed by the sequence constructor are computed similar to that of the conjunction constructor.

We define the strength of a primitive or fuzzy event as the membership degree of the corresponding event parameter. For example, an event like "obj_1 is close to obj_2" can have different strengths depending on how close obj_1 is to obj_2 in a particular situation. Closer the objects, more strong is the fuzzy event.

4.2 Inter-rule Fuzziness Via Scenarios

There can exist a finite set of events that can be signaled in an active database system. We partition the whole event set E into event groups called scenarios (not necessarily disjoint). The idea of scenarios comes from the need to group rules into sets corresponding to different situations. There can be only one active scenario at a time. Switching among scenarios is performed by rules as well. Consider the battlefield application we discussed in Section 3, where there can

be *emergency* situations as well as *normal* situations. An emergency situation corresponds to the events which may have serious effects like a serious damage and should urgently be handled whereas a normal situation corresponds to the events with a low level of importance. Switching from a normal scenario to an emergency scenario is performed by rules which detect emergency situations. Each rule may be subscribed to more then one scenario. If a rule is not subscribed to a scenario, then it is called an *idle rule*. Each scenario behaves like a fuzzy set, i.e., it has a membership function that maps the rules to a real number in the range $[0, 1]$. Events belonging to a scenario are fuzzy events as described in Section 4.1. Event signaling is done by considering the membership degree of the event parameter in the fuzzy event. The fuzzy event structure described in [BW97] is utilized where a primitive event is a tuple, $e :< e_c, e_f >$, consisting of a crisp part e_c which is the crisp parameter coming from the system and a fuzzy part e_f which denotes the fuzzy event set. In order to decide whether a rule r will be fired in response to the signaling of a fuzzy event $e :< e_c, e_f >$, the membership value of r is multiplied by the membership value of e_c in the fuzzy set e_f. The result is compared with a threshold value associated with the rule. If the result is greater than or equal to the threshold value, then the rule is fired. Threshold values of rules can be changed dynamically to tune to particular scenarios.

Assume that in our battlefield application, we have *emergency* and *normal* scenarios which are considered to be fuzzy sets. Each rule belongs to one or two of the scenarios with a membership degree. Consider the following rule:

event: obj_1 is very close to obj_2
condition: obj_1 belongs to enemy and obj_2 belongs to the alliances
action: fire an alarm and inform obj_2

which belongs to the emergency scenario with a membership degree of 0.9. Assume that its event is signaled, and the membership degree of the event parameter (i.e., the distance between the objects) in the fuzzy set *close* is 0.7. If the rule has a threshold value 0.6 for that scenario, then the rule will be fired since $0.9 * 0.7 = 0.63 \geq 0.6$. The threshold parameters and the membership functions for the fuzzy rules can be determined according to the results of a priori simulations.

4.3 Similarity Based Event Detection

Signaling of similar events upon an event detection is something very useful when the cost of missing events is very high in supported applications, like a nuclear reactor control system. Assume that an event such as *update in temperature level* is detected. Events with a high similarity degree, like *update in pressure level* should also be signaled automatically. This way, the risk of events escaping from detection is reduced.

In similarity based event detection, when an event is signaled, other events which are similar to it should also be fired. In order to facilitate this, each

scenario has a similarity matrix, where the similarity values of the events in that scenario are stored. Similarity matrices which show the similarities between events in a pairwise manner can be provided by the experts of the particular application; in our application they are military experts. Similarity matrices can be dynamically constructed and updated by the system by examining the event history. Signaling of two events consecutively in a short period of time implies that those events may be similar. As the consecutive signaling of two events is seen more frequently in the event history, the similarity of these events should be increased in the similarity matrix. This way, system learns the similarity values as the event history grows.

An example would be helpful in explaining similarity based event detection. Assume that event e_1 is raised. Other events whose similarity to e_1 is greater than or equal to the similarity threshold for the current scenario also need to be considered. If, for example, the similarity threshold for a scenario s is 0.7, and e_1 is signaled (which belongs to s) and another event e_5 is similar to e_1 with degree 0.8, then event e_5 should also be signaled since $0.8 \geq 0.7$. But the membership value of e_5 is multiplied by its degree of similarity (in this case 0.8) in order to determine which rules are going to be fired as a result of e_5.

Grouping of rules into scenarios restricts the number of rules to be considered when an event is raised, improving the efficiency of rule execution especially in case of emergency when efficient use of resources is very important.

4.4 Fuzzy Coupling Modes

In ECA rules coupling modes between event and condition, and between condition and action determine when the condition should be executed relative to the occurrence of the event, and when the action should be executed relative to the satisfaction of the condition, respectively. There are three basic coupling modes: *immediate*, *deferred*, and *detached* (or *decoupled*) [Day88]. If the condition is specified to be evaluated in *immediate* mode, then it is executed right after the triggering operation that caused the event to be raised. If the action part is specified to be executed in immediate mode then it is executed immediately after the evaluation of the condition. In case the condition is specified to be in *deferred* mode, its evaluation is delayed until the commit point of the transaction, and similarly if the action is in deferred mode relative to the condition, again it is executed right before the transaction commits. Finally, in *detached mode*, condition is evaluated or action is executed in a separate transaction.

Coupling modes is a very important concept for rule execution in active database systems and should also be considered for fuzzy rule execution. In fuzzy ECA rules, the coupling modes between event and condition, and between condition and action can be determined depending on the strength of the event and *credibility* of the condition respectively in case the coupling mode is not specified explicitly. We define the *credibility* of a condition as the truth value of the fuzzy predicate or the combination of the fuzzy predicates. Determination of the truth values of the fuzzy predicates is explained in [KF88]. A high credibility implies immediate or detached coupling mode and a low credibility

implies deferred coupling mode in case the coupling modes are not specified explicitly. Each coupling mode should be assigned a credibility threshold which will determine the coupling mode between the event and condition, and condition and action. That way, implicit priorities are assigned to the condition and action depending on the credibility of the corresponding event and condition. Strength of an event signaled due to its similarity to another event is calculated by multiplying its similarity value by its membership value. Assume that, in an emergency scenario, two of the events are $e_1 :< e_{c1}, e_f >$ and $e_2 :< e_{c2}, e_f >$. If the membership value of e_{c1} in e_f is 0.8, the membership value of e_{c2} in e_f is 0.4, threshold values for immediate, detached and deferred coupling modes are, 0.7, 0.5, and 0.0[2], respectively. Assuming that both e_1 and e_2 are signaled, the condition of the rule whose event is e_1 will be evaluated in immediate mode where the condition of the rule whose event is e_2 will be evaluated in deferred mode. The same techniques can be applied to the rules with composite events using the membership calculation methods proposed in Section 4.1.

Assume that we have a rule with the event "obj_1 is very close to obj_2" and no coupling mode was assigned for the rule. If the strength of the event is 0.95 which means that when obj_1 gets very close to obj_2, then the condition should be evaluated immediately, suspending the transaction that signaled the event, or should be evaluated in detached mode. But if the strength of the event is 0.6 then the evaluation of the condition can be deferred to the end of the transaction since obj_1 is not dangerously close to obj_2.

5 Conclusion

In this paper we have discussed the potential of adapting fuzzy database concepts in active database rules and mobile database systems. In doing this, we assumed an active mobile database management system which is a mobile database system supported with active features. In Section 4, we have tried to illustrate how fuzziness can be introduced to different aspects of rule execution. Lots of interesting research issues have been raised mostly on the incorporation of membership degrees for the dynamic determination of coupling modes of rules and priority assignment. Partitioning of the rule set into scenarios has also been discussed as an example of inter-rule fuzziness.

As a future work, a fuzzy database system can be designed for a mobile system application like a battlefield environment together with all possible types of generic queries. The concepts developed for the incorporation of fuzziness into active databases can be applied to a real active database system to measure the effectiveness of the proposed methods. Fuzzy rules may also provide a high degree of flexibility to mobile systems. Design of mobile database system applications incorporating fuzzy rules is an important research issue that requires further work.

[2] A value greater then zero as a credibility threshold for deferred mode means that some rules may not be fired even in deferred mode.

Acknowledgment

The authors wish to thank Dr. Adnan Yazıcı for his help in establishing the fuzzy concept background of this work.

References

[BKPW97] Tarik Bouaziz, Janne Karvonen, Anton Pesonen, and Antoni Wolski. Design and implementation of tempo fuzzy triggers. In *Lecture Notes in Computer Science*, volume 1308, pages 91–100, Toulouse, Fran, September 1997. Springer.

[BP82] B. Buckles and F. Petry. A fuzzy model for relational databases. *International Journal of Fuzzy Sets and Systems*, 7:213–226, 1982.

[Buc94] Alejandro Buchmann. Active Object Systems. In Asuman Dogac, M. Tamer Ozsu, Alex Biliris, and Timos Sellis, editors, *Advances in Object-Oriented Database Systems*, pages 201–224. Springer-Verlag, 1994.

[BW96] Tarik Bouaziz and Antoni Wolski. Incorporating fuzzy inference into database triggers. Technical report, VTT Information Technology, P.O. Box 1201, November 1996.

[BW97] Tarik Bouaziz and Antoni Wolski. Applying fuzzy events to appoxiamte reasoning in active databases. In *Proc. Sixth IEEE International Conference on Fuzzy Systems(FUZZ-IEEE'97)*, Barcelona, Catalonia, Spain, July 1997.

[CM91] S. Chakravarthy and D. Mishra. An event specification language (snoop) for active databases and its detection. Technical report, University of Florida at Gainesville, September 1991.

[Day88] Umeshwar Dayal. Active Database Management Systems. In *Proceedings of the Third International Conference on Data and Knowledge Bases*, pages 150–169, Jerusalem, June 1988.

[GGD94] Stella Gatziu, Andreas Geppert, and Klaus R. Dittrich. The samos active dbms prototype. Technical report, Institut fur Informatik, Universitat Zurich, October 1994.

[IB94] Tomasz Imielinski and B. R. Badrinath. Mobile wireless computing: Challenges in datata management. *Communications of the ACM*, pages 19–27, October 1994.

[KF88] George J. Klir and Tina A. Folger. *Fuzzy Sets, Uncertainty and Information.* Prentice Hall, 1988.

[MB96] S. Morton and O. Bukhres. Mobile transaction recovery in distributed medical databases. In *LASTED Eighth International Conference on Parallel and Distributed Computing and Systems*, 1996.

[MB97] S. Morton and O. Bukhres. Mobile computing in military ambulatory care. In *The 10th IEEE Symposium on Computer-Based Medical Systems(CBMS'97)*, 1997.

[MBM96] S. Morton, O. Bukhres, and M. Mossman. Mobile computing architecture for a battlefield environment. In *International Symposium on Cooperative Database Systems for Advanced Applications*, 1996.

[Zad65] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.