An intelligent software agent for assisting database administrators

Magdi H. Nagi, Amani A. Saad and Zaki M. Zaki
Computer Science Dept., Faculty of Eng., Alexandria University, Alexandria, 21544 Egypt

The advances in modern database management systems have resulted in so complex systems that made the database administration *task* much more complicated. Although there are tools that are commercially available to assist the DataBase Administrator (DBA), there is still a wide gap between the DBA increasing needs and those tools. In the mean time, assisting computer users through the use of intelligent software agents has attracted much interest in research. Noting that the problem of assisting the DBA did not receive the attention it deserves, we propose the use of the technology of Intelligent Agents to aid in this process. In this paper, we report on our new Intelligent Software Agent, designed to assist the DBA especially in performance tuning. The agent has been implemented using Java, tested against an Oracle DBMS and have already shown promising results.

إن التطور في نظم إدارة قواعد البيانات الحديثة قد أدى إلى تعقد الأنظمة مما جعل مهمة مدير قواعد البيانات اكثر تعقيدا وبالرغم من وجود أدوات تجارية تستخدم كأدوات لمساعدة مديري قواعد البيانات مازالت الهوة واسعة بين تلك الأدوات واحتياجات مديري قواعد البيانات المتزايدة. فى الوقت نفسه، حظي موضوع مساعدة مستخدمي ألحواسب بصفة عامة على اهتمام الباحثين فى مجال برامج العميل الذكى، إلا انه قد لوحظ أن مساعدة مديرى قواعد البيانات خاصة لم تحظ بالاهتمام الذي تستحقه على الإطلاق. لذا فإننا نقترح من خلال هذا البحث توظيف تكنولوجيا برامج العميل الذكى في خدمة مديري قواعد البيانات.وهذا البحث يقترح عميلا ذكيا جديدا صمم من اجل مساعدة مديرى قواعد البيانات وخاصة في مجال أداء الأنظمة. ويستخدم فى تصميم هذا العميل الذكى نظام الاستنباط الهلامي حيث يستخدم فى تمثيل معارف إدارة قواعد البيانات بما يتلائم مع طبيعتها. وبالإضافة إلى ذلك فان برنامج العميل الذكى يمكنه أن.يتعلم من النتائج التي تحدث نتيجة تطبيق تلك التوصيات. هذا وقد تم تطبيق برنامج العميل الذكى عمليا باستخدام لغة جافا وتم اختباره مع نظام إدارة قواعد البينات الخاص بشركة أوراكل وقد أظهرت النتائج مؤشرات ناجحة.

Keywords: Database administration, Performance tuning, Intelligent agents, Fuzzy inference systems

1. Introduction

Database management systems play more critical roles in current systems. If database system fails there is a high probability that the entire company will be inconvenienced or even forced to stop. That is why, the term *Mission Critical Applications* is more popular these days. Companies are dealing with large volumes of data to a degree that was unimaginable just a few years ago. Users need to access data from anywhere and via different types of devices. The Internet has even placed more demands on maintaining the availability of database systems twenty-four hours a day, seven days a week. Moreover, the increase in using Decision Support Systems (DSS) with the wealth of ad hoc query tools that emerged during the 1990s added more complications to the database management systems. The demands on the database that are made by users of DSS applications vary dramatically. One moment they may enter a query that fetches only a few records and the next moment they may enter a massive parallel query that fetches and sorts millions of records from different tables. This is in contrast with Online Transaction Processing (OLTP) applications, which process thousands or even millions of very small transactions per day.

Behind the scene, the DataBase Administrator (DBA) plays a significant role in developing, deploying and maintaining these database systems. The DBA responsibilities include installations, upgrading, capacity planning, creating the database logical and physical structures, managing users and security, monitoring and optimizing database performance, planning and implementing backup and recovery, and troubleshooting database problems. These responsibilities require from the professional DBA to cover different areas of technical knowledge up to date such as: DataBase Management Systems (DBMSs), Data Modeling, Operating Systems, Computer Architecture, System Performance, Networking, OLTP, DSS, Web-based Applications and Software Engineering.

Optimizing the database performance as one of the major responsibilities of the DBA is neither a trivial nor a deterministic task. It is dependent in one of its aspects on imprecise linguistic rules and the expertise of the DBA. Monitoring the performance of production databases is essential for the DBA. Unfortunately, this is where most of the performance management tools end; and the DBA is left to figure out the means to correct the problem that was detected by the monitoring tool. Even though the DBA may realize he has a poorly tuned database environment, he may not have the luxury to resolve the problem, because:

- tuning the database environment may take more time than what could be afforded for this task,
- management may not want to disrupt business operations,
- the DBA may not want to risk introducing further performance degradation,
- the complexity of the problem may be beyond the skill level of the person assigned the problem,
- the tools may not be available to identify the cause of the problem.

Noting also that a considerable percentage of the Total Cost of Ownership (TCO) goes to the DBA, all of these factors may explain the motivation behind trying to support the DBA as much as possible.

In the mean time, Intelligent Software Agents is a rapidly developing area of research with a wide range of firms and universities pursuing agent technology. In 1995, it has been expected that within 10 years most new Information Technology (IT) development will be affected by agents and many consumer products will contain embedded agent-based systems. Assisting different levels of computer users through the use of Intelligent Agents (IA) has recently been one of the major research areas, however, assisting the DBA did not receive the attention it deserves. But how can an intelligent agent help the DBA? In fact an intelligent agent can help the database administrator in several ways, e.g. if the DBA faces an undocumented problem while installing a DBMS, an intelligent agent who is looking over the shoulder of the DBA can check the support knowledge base of the vendor of that DBMS and the related newsgroups for any similar problems, then, he will give the DBA a list of the solutions he found ordered with respect to how much he believes that these solutions will succeed.

In this paper, we report on our new Intelligent Agent, designed to assist the DBA by recommending to him the database administration actions to take according to: a predefined set of fuzzy (linguistic) rules. Also, the state of the system and the decision-making algorithm it uses are presented. The agent learns from the effects of applying his recommendations on the database system. At the core of that agent there is a fuzzy inference system that is used to capture the imprecise nature of the DBA knowledge and to conduct the reasoning process.

The paper is organized as follows: First the area of Intelligent Agents is introduced. Then some basic fuzzy modeling concepts are presented, followed by our proposed agent and its architecture. A case study that is based on an Oracle database with a workload drawn from the Wisconsin benchmark for relational databases is illustrated and a sample of the experiments conducted is presented and the results are discussed. Finally, the paper is concluded and some directions of future work are presented.

2. Intelligent software agents

Imagine that while you are working on your computer, your software assistant (agent) who is working in the background draws your attention to one particular paper that has been published on that moment on the web. Your assistant thinks that this paper is important to you as he noticed - from the paper you were writing in the last few weeks - that you are working in the same point. This scenario is just a sample of what an Intelligent Agent (IA) can do for you [1].

Although intelligent agents is a rapidly developing area of research [2], there is no Universal definition of IA. One of the definitions that may illustrate the functionality of IA is that proposed in [3]: "An autonomous agent is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future."

Intelligent agents may have several features but some of which are necessary which are [4]:

- Autonomy [1]: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state.
- Adaptability: an agent is able to improve over time, i.e., becomes better at achieving its goals with experience.

Some other optional features of Intelligent Agents are:

- Collaboration: The ability to collaborate with other agents to achieve their goals, so, they need to have the ability to interact with other agents and possibly humans via some communication language. This feature is sometimes thought to be mandatory based on the nature of the application.
- Mobility: The ability of an agent to move across networks and between different hosts to achieve its goals.

The applications of Intelligent Agents are quite large and cover many aspects of Information Technology. The following is just a sample of those applications:

- User Assistance (Interface agent) such as the e-mail interface agent of MIT [5], the Internet Softbot [6] and the LPA (Learning Personal Agent) [7],
- Information Retrieval such as SIMS [8] and St. Bernard [9],
- Financial Portfolio Management such as WARREN [10],
- Organizational Decision Making such as PLEIADES [10],
- E-commerce such as ShopBot [11], and
- File system maintenance such as SUMPY [12].

In the following section, we present some of the basic concepts of fuzzy modeling, as it is the technique we have used in our agent to model its environment.

3. Fuzzy modeling

Fuzzy modeling has been successfully used not only in research but also in a wide range of commercial products such as: smart washing machines, user-seeking electric fans and smart photocopiers. Fuzzy modeling is based on *Fuzzy logic* which can be viewed as a superset of the Boolean logic in the sense that it can handle the concept of partial truth [13]. Fuzzy logic solutions are generally characterized by being a human solution. A famous example introduced by Zadeh [14] is the car parking problem in which the objective is to place the car near the curb and almost parallel to it. A fuzzy logic solution of the parking problem would be a collection of fuzzy if-then rules that describe how a human parks a car. It is important to note that the problem of parking is easy for humans when it is formulated imprecisely and difficult to solve by traditional methods because such methods do not exploit the tolerance for imprecision [15].

In this section we list definitions [16,17] of some basic concepts in fuzzy modeling, as these terminology will be used through out the paper.

Definition 1: Fuzzy sets and membership functions

If X is a collection of objects denoted generically by x, then a fuzzy set A in X is defined as a set of ordered pairs:

$$A = \{(x, \mu_A(x)) \mid x \in X\},$$

where, X is referred to as the Universe of discourse and it may be discrete or continuous, and $\mu_A(x)$ is called the Membership Function (MF) of x in A. The membership function maps each element of X to a continuous membership value between 0 and 1. This is in contrast to the classical set (crisp set) in which an object either belongs to $\mu_A(x)$ or does not belong to the set A; there is nothing in between. Note that if is restricted to 0 or 1 only, then A is reduced to a classical set.

Definition 2: Linguistic variable and linguistic value

A linguistic variable is a variable that takes linguistic values such as height, age, speed, etc. Such variables can take linguistic values like *tall, young, fast, good,* etc. A linguistic value is a label for a fuzzy set.

Definition 3; t-norm [16,17]

A function T: $[0,1]^2 \rightarrow [0,1]$ is called t-norm, if the following conditions hold:

$T(a,1) = a$ *(unit 1)*

$a \leq b \Rightarrow T(a,c) \leq T(b,c)$ *(monotonically)*

$T(a,b) = T(b,a)$ *(commutativity)*

$T(a,T(b,c)) = T(T(a,b),c)$ *(associativity)*

Definition 4: t-conorm

A function $\perp$: $[0,1]^2 \rightarrow [0,1]$ is called t-conorm, *iff* $\perp$ is commutative, associative, monotonic, and 0 is its unit.

Definition 5: A fuzzy if-then rule (a fuzzy rule) assumes the form

if x is A then y is B

where A and B are linguistic values defined by fuzzy sets on the Universes of discourse X and Y, respectively, "*x is A*" is usually called the antecedent or premise, while "y is B" is usually called the consequent or conclusion.

Here is an example that describes the relationship between pressure and volume:

If pressure is high, then volume is small.

where, *pressure* and *volume* are linguistic variables, *high* and *small* are linguistic values that are characterized by membership functions.

Several types of fuzzy reasoning have been proposed in the literature. Depending on the types of fuzzy reasoning and fuzzy if-then rules employed, most fuzzy reasoning mechanisms can be classified into the three types illustrated by fig. 1.

Most of the differences between these types of reasoning lie in the specification of the consequent part (monotonically non-decreasing or bell-shaped membership functions, or crisp functions) and thus the defuzzification schemes (weighted average, centroid of area, .. etc.) are also different.

"It is relatively easy to write down a set of fuzzy rules to describe a particular behavior. However, to calibrate these rules, i.e., to choose the type and characterization of the membership functions, is not a trivial problem." [13]. This issue will be raised in the discussion section, which explains the mapping between the database administrator knowledge and the fuzzy model.

Now, that the necessary background has been discussed. The rest of the paper presents our agent.

4. DBA assistant agent

The purpose of the DBA assistant is to help the DBA in his work by recommending to him actions to make. The agent will start his work with a set of predefined linguistic rules drawn from the database administration knowledge. To capture the imprecise nature of the DBA knowledge, there is a fuzzy inference system at the core of the agent architecture, so the linguistic rules will be represented in the agent as fuzzy rules. The architecture is designed to be open enough to cover different areas of database administration. Fig. 2 illustrates the architecture of the proposed agent-based system.

To assist the DBA in performance tuning for example, the agent monitors the performance of the database through a Database Interface Layer and infers recommendations when appropriate, based on his knowledge base and the database state. In order to illustrate the function of each module in our agent, the following example is presented. The example

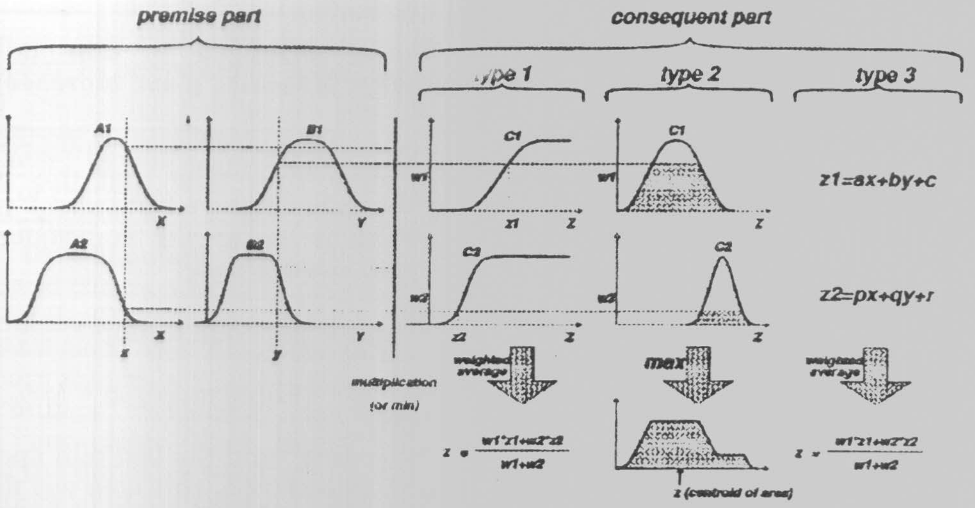

Fig. 1. Commonly used fuzzy if-then rules and fuzzy reasoning mechanisms.

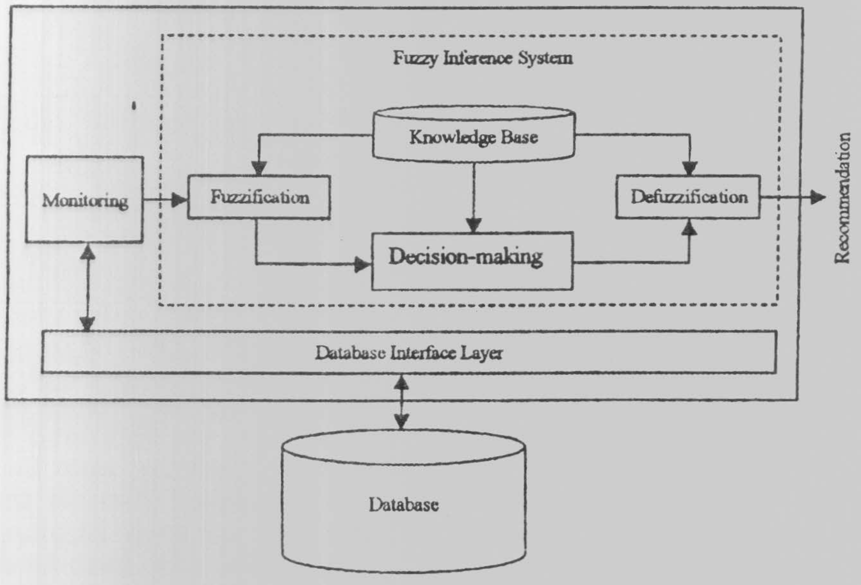

Fig. 2. DBA assistant agent architecture.

is for a rule that is used to tune the buffer cache in an Oracle database.

"IF Cache hit ratio is *Low* and Last increase in the buffer was *Effective* THEN *increase* the buffer."

From the rule, we can identify the following fuzzy variables: *Cache hit ratio* and *Last increase* in the antecedent and *Buffer Change* in the consequent. Note that if the consequent part is rewritten as "Buffer change is increase" or "Buffer change is Positive" then it would be more clear why the "Buffer Change" has been considered as a fuzzy variable in the consequent. So the rule can be rewritten as:

"IF Cache hit ratio is *Low* and Last increase in the buffer was *Effective* THEN *Buffer* change is Increase."

According to the rule the three fuzzy variables may have the corresponding fuzzy values *Low*, *Effective* and *Increase* respectively. Now, let us see how the agent will work with this rule.

- *Monitoring module*: the monitoring module calculates the (crisp) values of the fuzzy variables in the antecedent part. In this example, it will calculate the hit ratio of the buffer cache and the effectiveness of the last increase in buffer cache.
- *Fuzzification module*: The fuzzification module calculates the membership degrees of each fuzzy variable in the related fuzzy sets of the corresponding fuzzy values according to the crisp value of the fuzzy variable calculated by the monitoring module. So, if the hit ratio is 72%, the fuzzification module calculates the corresponding membership function MF_{Low}. Assuming that MF_{Low} (72) = 0.66, that means that 72% hit ratio belongs to the Low fuzzy set with 0.66 degree of membership. Denoting the hit ratio by X, it is expected that MF_{Low} (X) tends to 1 as X tends to 0 and tends to 0 as X tends to 100 according to the meaning of MF_{Low}.
- *Decision making module*: The decision-making module infers the value of the variable in the consequent part. In the previous example, it infers the percentage of increase in the buffer size, i.e., it generates the recommended percentage of increase of the buffer size. To do this, the module makes the following two steps:

- Application of the t-norm operator (such as min) on the membership degrees of the premise part to get what is called the firing strength of the rule. So, in the mentioned example this step will be as follows:

Firing Strength = min (MF_{Low} (Cache hit ratio), $MF_{Effective}$ (Last increase))

- Generation of the qualified consequent of each rule, note that this module gets the membership functions of the fuzzy values which are in the consequent part from the knowledge base.

In the previous example,

Firing strength = $MF_{Increase}(Z)$.

Assuming that the last min operator resulted in a value of 0.66, this step will find the value of Z that has $MF_{Increase}$ equals 0.66, where Z is the recommended percentage of increase in buffer cache.

- *Defuzzification module*: the defuzzification module aggregates the qualified consequents of the rules to produce the crisp output. The t-conorm operator (such as max operator) would be used to aggregate the recommendations (consequents) of all the rules.

As the example is limited to one rule for simplicity, this step will not be used in it. But if there are other rules that are used to tune the buffer cache, i.e., the consequent parts contain Z, each rule may give a recommendation for the value of Z, all these recommendations are aggregated via this step through a t-conorm operator such as max. So the maximum value recommended from all the rules will be the value that the agent recommends.

After we have introduced the architecture of the proposed agent, a few questions arise: Will this architecture be suitable to our problem domain? Will the fuzzy inference system be able to capture the necessary knowledge of database administration? How the membership functions will be chosen? What are the interpretations of the fuzzy membership functions?

5. Case study

Our case study is based on one of the major commercial database management systems, which is the Oracle DBMS [18]. We are going to limit our discussion to *performance tuning* and more specifically *parameter tuning,* as this is what have been modeled and tested at the time of writing this paper. In this section we give an overview of one of the major parameters of the Oracle database which is the *buffer cache* and present one of the linguistic rules that is practically used and the membership functions that are used in our fuzzy model.

In Oracle databases, the database buffer cache is a memory structure that consists of a number of buffers specified by the db_block_buffers parameter. Each buffer has the same size of the database block that is the most granular unit of data storage used by Oracle and can contain several rows of table data. The buffer cache stores data that is needed by SQL statements issued in user processes. The buffer cache improves the performance of subsequent SELECT statements on the same data, and allows Oracle users to make changes quickly in memory. We are going to consider one of the major linguistic tuning rules that is practically used [19,20,21] and which was introduced in the previous section, "IF Cache hit ratio is Low and Last increase in the buffer was Effective THEN buffer change is Increase."

We have considered the following membership functions to represent the fuzzy values Low, Effective and Increase.

$$MF_{Low}(X) = \begin{cases} 1 & \text{for } X < 60 \\ (95 - X)/35 & \text{for } X > 60 \end{cases}$$

where, X is the hit ratio.

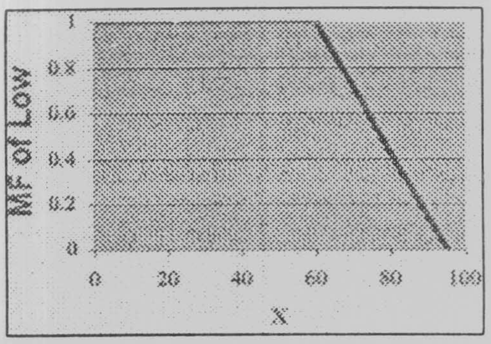

Fig. 3. Membership function of the fuzzy value *low*.

Note how the hit ratio is calculated, Oracle collects statistics that reflect data access and stores them in dynamic performance tables [22]. By retrieving the appropriate statistics through SQL statements [23,24], the hit ratio is calculated according to the following formula:

Hit Ratio = 1 - (physical reads / logical reads)

The SQL statements are issued through the Database Interface Layer as it is responsible for communication with the database. In our implementation the database interface layer is JDBC.

$$MF_{Effective}(Y) = Y/100;$$

where, Y represents the effectiveness of the last increase in buffer cache and we calculate it using the following formula:

$$Y = \begin{cases} \text{(New Hit Ratio} - \text{Previous Hit Ratio)/ Previous Hit Ratio} * 100, & \text{if previous hit ratio} \neq 0 \\ 100 & \text{otherwise.} \end{cases}$$

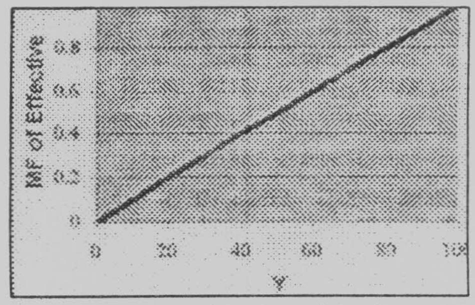

Fig. 4. Membership function of the fuzzy value *effective*.

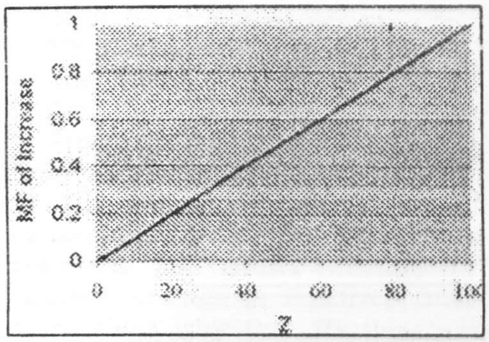

Fig. 5. Membership function of the fuzzy value *increase*.

Note that in the initial state (first inference), it is assumed that the previous hit ratio is zero, i.e, Y = 100.

$$MF_{Increase}(Z) = Z/100\,;$$

where, Z is the recommended increase percentage in buffer cache.

6. Experiments and results

In this section we present some of the experiments we conducted on our case study. The experiments are based on the Wisconsin benchmark for relational databases [25].

6.1. Experiment 1

In this set of experiments, the agent tunes the buffer assuming that the initial size of the buffer cache is 500 blocks.

In Experiment 1.1, the agent has monitored the workload and calculated the hit ratio according to the formula mentioned earlier. he has found that the hit ratio is 72 %. As shown in fig. 6, based on his knowledge base he has recommended to the DBA to increase the buffer by 66%, i.e., to 830 database blocks.

In Experiment 1.2, the buffer cache has been increased as recommended, the agent has monitored the workload again, he has calculated the new hit ratio and found that it has been increased to 73.57% and so recommended to increase the buffer cache by 2%, i.e., to 850 database blocks. Note that in this experiment the agent has used the knowledge that he has learned from Experiment 1.1, so he has just increased the buffer by 2% compared with 66% in Experiment 1.1. This is the effect of the fuzzy variable Last Increase. This experiment is illustrated in fig. 7.

In Experiment 1.3, the same process has been applied again but the agent has deduced that it will not be useful to increase the buffer again so he did not recommend any increase to the buffer. Table 1 summarizes the results of this set of experiments.

Although the hit ratio has just increased from 72% to 73.77%, the results has shown that the execution time has been reduced by 10% from 79,542 (msec.) to 71,202 (msec.)

6.2. Experiment 2

To make sure that the agent has worked properly, the relation between the size of the buffer cache and the hit ratio is studied through Experiment 2 by calculating the hit ratio that results from applying the same workload mentioned earlier to different buffer sizes. table 2 summarizes the results and table 3 shows a sample of the details of the collected statistics for each experiment conducted in our case study.

Table 1
Summary of results for experiment 1

Exp. No.	Buffer size	Hit ratio	MF_{Low}	Effective increase	$MF_{Increase}$	Deduced increase	New buffer size
1.1	500	72	0.657143	100	1	65.71429	830
1.2	830	73.57	0.612286	2.180556	0.021806	02.18056	850
1.3	850	73.77	0.606571	0.27185	0.002718	00.27185	850

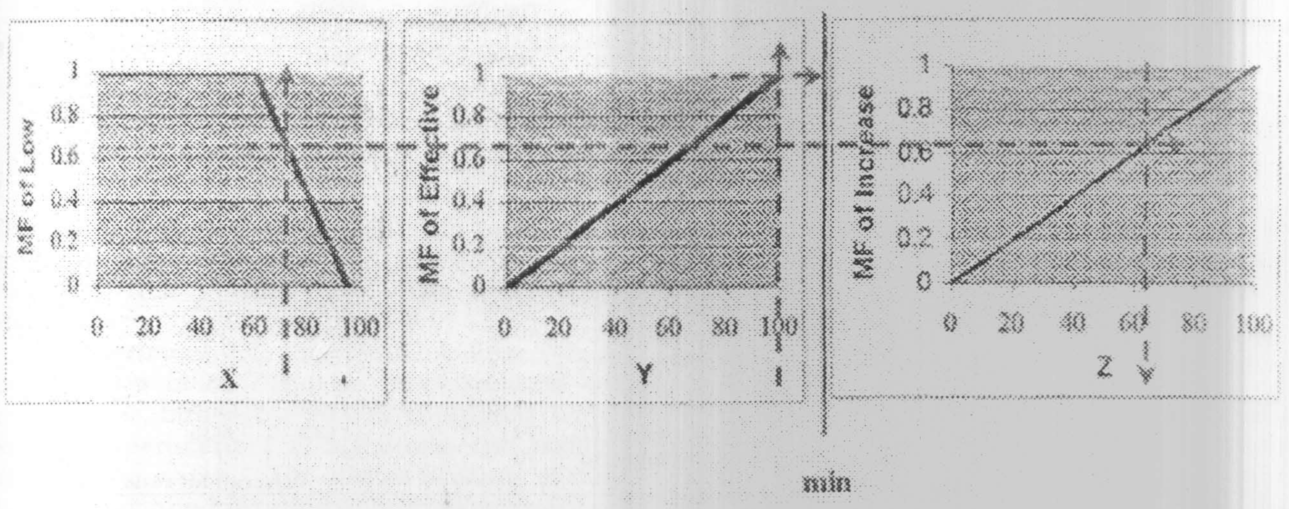

Fig. 6. Fuzzy reasoning of experiment 1.1.

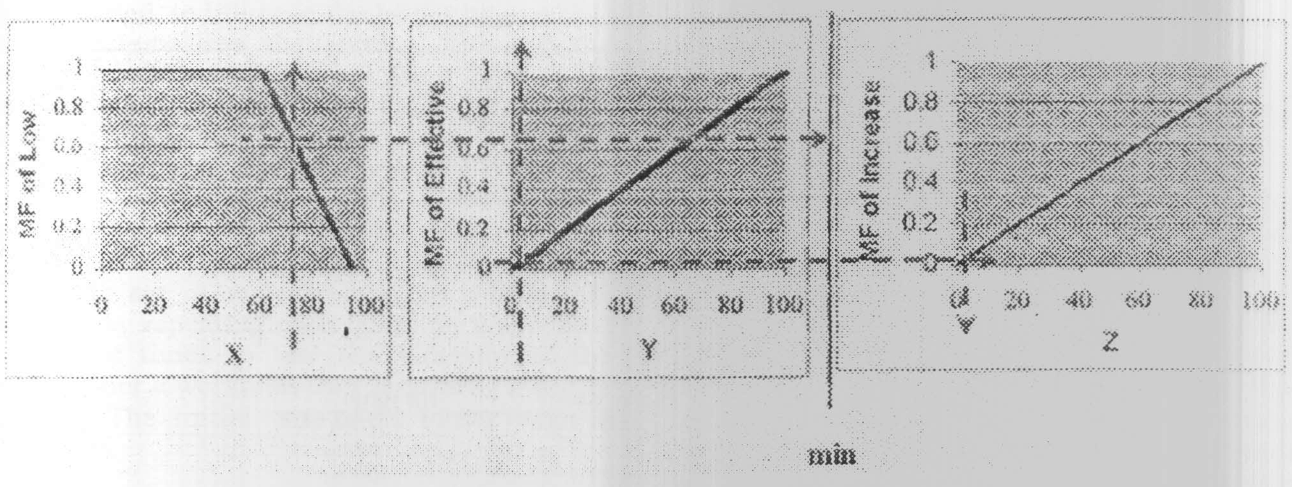

Fig. 7. Fuzzy reasoning of experiment 1.2.

Table 2
Summary of Results for Experiment 2

Exp. No.	Buffer size	Hit ratio
2.1	100	66%
2.2	200	68%
2.3	300	69%
2.4	400	71%
2.5	500	72%
2.6	600	72%
2.7	700	73%
2.8	800	73%
2.9	900	74%
2.10	1000	74%
2.11	1100	74%
2.12	1200	74%
2.13	1400	74%
2.14	1600	74%
2.15	1800	74%
2.16	2000	74%
2.17	2500	74%
2.18	3000	74%

Table 3
Summary of Results for Experiment 2.5 (buffer size = 500)

Query code	Execution Time(msec.)	Logical reads	Physical reads	Hit ratio
Wisconsin.001	851	483	42	91%
Wisconsin.002	2804	886	122	86%
Wisconsin.003	1131	184	90	51%
Wisconsin.004	2554	1922	1022	47%
Wisconsin.005	260	3	2	33%
Wisconsin.006	1753	28	11	61%
Wisconsin.007	6197	2712	483	82%
Wisconsin.008	3405	1786	211	88%
Wisconsin.009	5418	2293	399	83%
Wisconsin.010	5137	3798	457	88%
Wisconsin.011	7621	4745	355	93%
Wisconsin.012	5188	3785	354	91%
Wisconsin.013	4917	4308	1428	67%
Wisconsin.014	1552	1289	1091	15%
Wisconsin.015	28451	9446	3139	67%
Wisconsin.016	1512	1122	1088	3%
Wisconsin.017	721	1133	935	17%
Wisconsin.018	70	4	2	50%
			Overall hit ratio	72%

Table 4
Summary of results for experiment 3

Exp. No.	Buffer size	Hit ratio	MF_{Low}	Effective increase	$MF_{Increase}$	Deduced increase	New buffer size
3.1	500	86.74	0.236	100	1	23.6	620
3.2	620	88.67	0.180857	2.22504	0.02225	02.22504	630
3.3	630	88.67	0.180857	0	0	0	630

Table 5
Summary of results for experiment 4

Experiment	Buffer size	Hit Ratio
4.3.1	100	75%
4.3.2	500	87%
4.3.3	600	88%
4.3.4	700	89%
4.3.5	800	89%
4.3.6	900	89%
4.3.7	1000	89%
4.3.8	2000	89%
4.3.9	3000	89%

Table 6
Summary of Results for Experiment 5

Exp. No.	Buffer size	Hit ratio	MF_{Low}	Effective increase	$MF_{Increase}$	Deduced increase	New buffer size
5.1	500	77.48	0.500571	100	1	50.05714	750
5.2	750	79.24	0.450286	2.271554	0.022716	02.27155	770
5.3	770	79.28	0.449143	0.05048	0.000505	00.05048	770

Experiment 2 show that the optimum buffer size is greater than 800 and less than or equal to 900, and this verifies that the agent was working correctly in Experiment 1.

In Experiment 2.1, the agent has found that the hit ratio is 86.74%, he recommended to increase the buffer by 23.6% compared with 66% in Experiment 1.1, although, the buffer size was the same. He did so because of the difference in the hit ratios of both experiments. This is the effect of the membership function of the fuzzy variable **Low** shown in fig. 3. Through Experiments 2.2 and 2.3, the agent has recommended to increase the buffer by just 2%. table 4 summarizes the results. Although the hit ratio has just increased from 86.74% to 88.67%, the results has shown that the execution time has been reduced by 19% from 237,601 (msec.) to 191,997 (msec.).

6.3. Experiment 3

In this set of experiments, the workload is changed by repeating each query for a specified number of times to test the behavior of the agent under a workload that results in a higher hit ratio. The initial size of the buffer cache is 500 blocks.

6.4. Experiment 4

In this set of experiments, we examine the relationship between the buffer size and the hit ratio for the workload of Experiment 3. Table 5 summarizes the results.

Note that these experiments show that the optimum buffer size is greater than 500 and less than or equal to 1000, and this verifies that the agent was working correctly in Experiment 3.

6.5. Experiment 5

In this set of experiments, the workload is a random generation of the queries mentioned earlier 75 times. The initial size of the buffer is 500 database blocks. Through the three experiments, the agent has recommended to increase the buffer to 770 database blocks. Table 6 summarizes the results.

7. Discussion

7.1. Weighting the effect of each of the collected statistics

In this section, we discuss how the DBA knowledge can be mapped to the fuzzy model, as this is a major step in using our agent.

Notice that the DBA may consider the 95% hit ratio as not Low, but for the 72% and 85% he may consider them Low but with different weights. This difference in weights for these different values will have its effect on how much he will increase the buffer (and even how much he thinks he has a problem at all). Suppose we are going to express these weights by figures, let us say that he will weight the 72% as 0 7 Low and 85% as 0.3 Low. This is typically what the MFLow is, each point in which represents how much he weights the effect of the associated statistics (or variables in general). Hence, in fig. 8 each MF represents a different model. For a 72% hit ratio, each MF will give a different value. This can be seen as modeling the knowledge of different database administrators or the same database administrator in different environments. For example if the DBA will consider that any hit ratio below 60% (say) is completely Low, that is what is represented in MF1. MF2 says that the DBA assumes that any hit ratio below 80% is completely Low. That means that he treats any value in that range in the same way, i.e., he will treat 59% in the same way as he treats 45%, e.g. he will increase the buffer with the same amount in both cases. This is in contrast with MF3 which has a different value for each hit ratio.

7.2. Weighting the combined effect of collected statistics

Sometimes the DBA takes his decision based on more than one statistics, so he combines all the results somehow to produce his decision. Considering our example for tuning the buffer, in addition to the Hit ratio, he will consider also how effective was the Last increase in the buffer size. Let us say that the hit ratio is very low and the DBA has found that the last increase in buffer was effective, so he will increase the buffer. In which membership function is this knowledge? In fact, it is not in a single membership function, but in the relationship between the two membership functions. Each value in MF_{Low} is related somehow to $MF_{Increase}$, .We can see this as if each point in MF_{Low} divides $MF_{Increase}$ into two ranges, one which is less than Y1 and the other which is greater than Y1. This is shown in fig. 9.

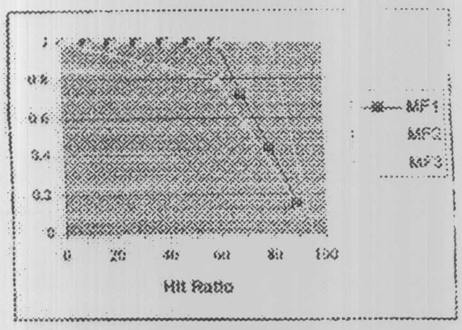

Fig. 8. Different membership functions.

So if we have a hit ratio X1, then we will favor the effect of the Cache hit ratio to that of the Last increase when Y > Y1 and vice versa. Therefore, not only that each membership function represents some database administrator knowledge, but also the relationship between different membership functions of different fuzzy variables represents another sort of knowledge.

8. Conclusion and future work

This paper described a new intelligent agent that assists database administrators. The agent captures the knowledge of the DBA through fuzzy modeling. It is already implemented in Java and tested against an Oracle database

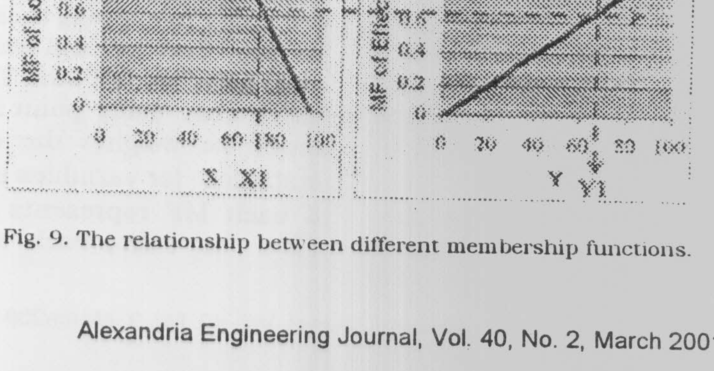

Fig. 9. The relationship between different membership functions.

considering tuning the buffer cache as a case study.

Preliminary results have shown that the architecture of the agent and the use of fuzzy modeling are promising in the area of supporting database administrators. Simplifying the task of database administrators through supporting them via intelligent agents can help in reducing the Total Cost of Ownership of database systems and encouraging more people to use modern database management systems through reducing the complexity of database systems.

Many other directions of future work remain open such as:

- extending the functionality of the agent to cover other areas of database administration,
- studying the enhancement of the reasoning mechanism of the agent through using neuro fuzzy techniques [26,27,28] such as NEFCON [29,30] and ANFIS[31,32],
- studying the extraction of the fuzzy rules of database administration and the corresponding membership functions from the history of database administration actions taken by database administrators and their results,
- extending the features of the agent to be able to share his knowledge with other agents through an Agent Communication Language (ACL) such as Knowledge Query and Manipulation Language (KQML) [33].

References

[1] Wooldridge M. and Jennings N., "Intelligent Agents: Theory and Practice", Knowledge Engineering Review, Vol. 10(2), pp. 115-152 (1995).

[2] Nwana H., "Software Agents: An Overview", Knowledge Engineering Review Vol. 11 (3), pp. 205-244 (1996).

[3] Franklin S., Graesser A., "Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents", Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, Springer Verlag, (1996).

[4] Maes P., "Modeling Adaptive Autonomous Agents", Artificial Life Journal, edited by C. Langton, MIT Press, Vol. 1(1, 2), pp. 135-162 (1994).

[5] Lashkari et. al., "Collaborative Interface Agents", Proceedings of the Twelfth National Conference on Artificial Intelligence, Vol. 1, AAAI Press, Seattle, WA, August (1994).

[6] Etzioni, O., Lesh, L. and Segal,R., "A Softbot-Based Interface to the Internet", Communications of the ACM, Vol. 37 (7), pp. 72-76 (1994).

[7] Pannu A. and Sycara K., "A Learning Personal Agent for Text Filtering and Notification", Proceedings of the International Conference of Knowledge-Based Systems (KBCS 96), December (1996).

[8] Knoblock C., et. al., "Cooperating Agents for Information Retrieval", Proceedings of the Second International Conference on Cooperative Information Systems, Toronto, Canada, May, pp. 122-133 (1994).

[9] Etzioni, O., Lesh L. and Segal R., "Building Softbots for UNIX", Technical report, Department of Computer Science and Engineering, University of Washington, (1992).

[10] Sycara K. et. al., "Distributed Intelligent Agents", IEEE Expert, December (1996).

[11] Etzioni, O. et. al., "Learning to Understand Information on the Internet: An Example-Based Approach" (1997).

[12] Song H., et al., "SUMPY: A Fuzzy Software Agent," Proceedings of the ISCA 5th International Conference on Intelligent Systems, June, pp. 124-129 (1996).

[13] Azvine B., Azarmi N., and Tsui K., "An Introduction to Soft Computing: A Tool for Building Intelligent Systems; Concepts and Applications", in Nwana H. and Azarmi N., editors, "Software Agents and Soft Computing - Towards Enhancing Machine Intelligence", Springer Verlag, pp.191-210 (1996).

[14] Zadeh L., "Fuzzy Logic, Neural Networks, and Soft Computing", Comm. of ACM. Vol. 37 (3), pp. 77-84 (1994).

[15] Zadeh A., "The Roles of Fuzzy Logic and Soft Computing in the Conception, Design and Deployment of Intelligent Systems", in: Nwana, H. and Azarmi, N., editors, Software Agents and Soft Computing Towards Enhancing Machine Intelligence, Lecture Notes in Artificial Intelligence (1198), Springer-Verlag, pp. 183-190 (1996).

[16] Nauck D., Klawonn F., and Kruse R., "Fuzzy Sets, Fuzzy Controllers, and Neural Networks", workshop in Postsdam, Germany (1992).

[17] Jang R., " Neuro Fuzzy Modeling Architecture, Analyses and Applications", University of California, Ph.D. Thesis, June (1992).

[18] L. Leverenze and Rehfield D, "Oracle8i Concepts, Release 8.1.5", Oracle8i Documentation, February (1999).

[19] Bauer M., "Oracle8i Tuning, Release 8.1.5", Oracle8i Documentation, February (1999).

[20] J. Couchman "Oracle8 Certified Professional, DBA Certification Exam Guide", Oracle Press (1999).

[21] Fee J., "Administrator's Guide, Release 8.1.5", Oracle8i Documentation, February (1999).

[22] Aronoff E., et al., "Oracle8 Advanced Tuning and Administration", Oracle Press, (1998).

[23] J. Durbin"Oracle8i Reference, Release 8.1.5", Oracle8i Documentation, February (1999).

[24] J. Trezzo "The V$ View: Critical Knowledgeable for DBAs", Oracle Magazine, March/April, pp.87-95 (1999).

[25] J. Gray "The Benchmark Handbook for Database and Transaction Processing Systems", Morgan Kaufmann Publishers, (1993).

[26] D. Nauck R. and Kruse "Choosing Appropriate Neuro-Fuzzy Models", Proceedings of EUFIT'94, pp. 552-557 (1994).

[27] D. Nauck "Beyond Neuro-Fuzzy Perspectives and Directions". Proceedings of the Third European Congress on Intelligent Techniques and Soft Computing (EUFIT'95), August, pp. 1159-1164 (1995).

[28] D. Nauck "Neuro-Fuzzy Systems: Review and Prospects", Proceedings of the fifth European Congress on Intelligent Techniques and Soft Computing EUFIT'97 (1997).

[29] D. Nauck R. and Kruse "A Fuzzy Neural Network Learning Fuzzy Control Rules and Membership Functions by Fuzzy Error Backpropagation", Proceedings of IEEE International Conference of Neural Networks ICNN'93, San Francisco, March 18–April 1, pp. 1022–1027 (1993).

[30] A. Nurnberger D. Nauck and Kruse R. "Neuro-Fuzzy Control Based on the NEFCON-Model Under MATLAB/ SIMULINK", 2nd On-Line World Conference on Soft Computing in Engineering Design and Manufacturing WSC2, (1997).

[31] R. Jang C. and Sun "Neuro-Fuzzy Modeling and Control", Proceedings of IEEE, Vol. 83, pp. 378 – 406 (1995).

[32] R. Jang "ANFIS: Adaptive-Network-Based Fuzzy Inference System", IEEE Transactions On Systems, Man and Cybernetics, Vol. 23, (3) pp. 665-685 (1993).

[33] T. Finin Y., Labrou and J. Mayfield "KQML as an agent communication language", in Bradshaw J., editor, "Software Agents", MIT Press, Cambridge (1997).

Received August 28, 2000
Accepted March 14, 2001