International Journal of Database Theory and Application
Vol. 2, No. 4, December, 2009

Fuzzy based approach for Load balanced distributing database on Sensor Network

[1]Mohammad Zeynali, [2]Leili Mohammad Khanli, and [3]Amir Mollanejad

[1]Islamic Azad University - Bostanabad, Branch, Iran
[2]Department of Computer, Science University of Tabriz, Iran
[3]Islamic Azad University - Jolfa, Branch, Iran
[1]Mo_zeynali@yahoo.com, [2] L-khanli@tabrizu.ac.ir, [3]Amir.mollanejad@gmail.com

Abstract

A wireless sensor network consists of tiny sensing devices, with limited energy and processing ability. Some time we have to distribute a database on sensor network, because of limited energy in sensors, load balanced distributing database can increase the lifetime of this networks. In this paper we propose Fuzzy based approach for Load balanced distributing database on sensor Network that prolonging the network lifetime. We use vertical partitioning algorithms for distributing database on sensors. First we clustering the network and then distribute partitions on clusters. A simulator was built and Results of various simulation runs are consistent with the hypothesis.

Keywords*: Sensor Network, Database, Distributing, Partition, Load Balanced, Hit Ratio.*

1. Introduction

Wireless Sensor Network (WSN) comprises of micro sensor nodes with limited energy and processing ability. It is used in military as well as civil applications. From telescopes to microscopes, we have developed instruments to monitor and observe the world in ways that are not obvious to the human eye. With recent and projected advances in small, low cost microelectronic and mechanical systems (MEMS) with limited on-board processing and wireless communication capabilities, and the development of new sensor materials, we can envision a new generation of technology that enables us to build large collections of untethered, battery powered sensors with various sensing functions that are distributed densely over a geographic region. These sensor networks are able to measure traffic conditions, weather development, seismic activity, or track the movements of toxic fumes at a level of detail that was not possible before. The continued trend towards miniaturization and inexpensiveness of sensors makes it possible that such sensor networks are less than a cubic millimeter in size[13], and sensor networks are made up of thousands or even millions of sensors. At that scale, wireless sensors could permeate the physical world, and help us vastly increase to increase our understanding of our physical environment.
Sensor networks, however, have the following constraints that pose new challenges from a system and application development standpoint:

• Power consumption: Sensor nodes are limited with regard to their battery supply, and energy conservation is a major system design principle. Also, communication is a much larger battery drain than local computation on a node.

• Low-range communication: The bandwidth and range of wireless communication is limited. Since communication is much higher drain on the energy consumption than on-board

processing, optimizing communication within the sensor network is a major system design consideration.

• Limited computing and storage capabilities: Sensor nodes have, at least, for the foreseeable future limited on-board computational, and volatile and persistent storage capabilities. Thus, on-board data processing of using available memory and CPU is also limited.

• Self organization: Due to the large number of sensor nodes, the failure rates of nodes, and the often unattended deployment, task management and handling in sensor networks is decentralized and self organizing. Thus, some level of local autonomy must be provided for the devices. Over the last years, much research work has focused on the design of robust and energy-efficient communication protocols and the development of sensor materials and devices. More and more research and commercial prototype platforms become available to be deployed in real-world applications. First prototypical employments of sensor network can be found in the environmental domain, such as non-invasive habitat monitoring [11, 12]. From the application developer perspective, the main purpose of sensor networks is collection, aggregation and processing of data as well as the actuation of the environment. Thus, today simple, easy-to-use programming interfaces for sensor networks become important. A user, often a domain scientist, would like to define the necessary tasks in a user-friendly way, and delegate the optimization and ultimately self-adaptive execution to the run-time environment without having to worry about the details. Traditionally, database management systems (DBMS) have had the purpose of making managing and querying very large amounts of data simpler and robust. The current generation of DBMS, based on extended relational database technology, is well-understood, and this type of DBMS is used almost pervasively in business as well as in scientific applications. Relational DBMSs provide a simple, query language, SQL, to model the data and formulate queries over the data. The mathematical foundation of SQL on the relational algebra enables automated reformulation and optimization of user queries to speed up the execution of queries significantly. From a DBMS perspective, a sensor network can be viewed as a large distributed database system. Here, each sensor node behaves like a tiny DBMS running on the sensor node platform and accepts, processes and answers queries. Furthermore, a sensor node DBMS also participates in the global execution of distributed queries. Consequently, the user can interact with the sensor network as a whole. For example, imagine the storm petrel sensor network application on Great Duck Island in Maine (USA) conducted by scientists from UC Berkeley, Intel Research Labs and the College of the Atlantic in Bar Harbor [11]. Great Duck Island is a 90-hectare island consisting of rock and grass off the coast of Mount Desert Island and is home to one of the world's largest breeding colonies of Leach's storm petrels. Researchers installed a collection of monitoring devices (motes) in the petrels' nesting burrows with sensors that monitor light, humidity, pressure, and heat. The data is transmitted via a radio transceiver to nearby motes. The available sensor data is modeled via a relation called Sensors whose schema consists of the node identifier, attributes for a node's on-board sensors such as light and temperature, and a time stamp for each data tuple. The tuples are created on demand by nodes in the sensor network based on queries sent to the sensor network. Sensors nodes only start sampling based on the query. For example, a scientific observer might be interested in periodic measurements of pressure values to monitor the inhabitation of birds in nests with the following SQL query:

SELECT sensor.id, pressure FROM sensors WHERE pressure>threshold
SAMPLE PERIOD 60sec

Every 60 seconds, the query is evaluated at all participating sensor nodes, and those nodes with pressure values larger than the given threshold return a data tuple, and others return a null value. Instead of periodic observations, the scientist might only look for events such as"when and how long does a storm petrel occupy a burrow?" Events can be derived from Collected data outside of the sensor network based on periodic measurements or, they can be processed directly in the sensor network. A third group of queries are aggregation queries such as simple aggregates of"how long are nests occupied on average?" or a query to find the nest with the highest weight:

SELECT sensor.id, MAX (pressure) FROM sensors WHERE pressure>threshold SAMPLE PERIOD 60sec

Note, that queries define the request data in a declarative way, and the user does not need to know the the availability or node identifier of sensors, nor does he/she deal with the details of the execution of the query. The DBMS routes the query to the sensor nodes of interest. System-internally, multi-hop georouting or other routing algorithms ([9, 10]) can be applied to distribute the query to all relevant nodes, which contribute their measurements as partial query results, and routing algorithm to intelligently and energy-efficiently route partial result back to the users.

In summary, the sensor network database system translates the declarative user query, and responsible for generating a query execution and optimization plan, and control the query execution as well as its error handling. Although, a sensor network database system provides a similar, easy-to-use interface compared to the traditional database world and its data handling paradigm is promising when dealing with the complexity of data collection and processing in sensor networks, the database technology applied 'under the hood' of the front-end changes significantly from traditional database management systems strategies to deal with the constraints of sensor networks.

Some time we have to distribute a database on sensor network, because of limited energy in this sensor, load balanced distributing database can increase network lifetime. In this paper we propose Fuzzy based approach for Load balanced distributing database on sensor Network that prolonging network lifetime. In proposed algorithm first we clustering the network [2],[3],[4], and partitioning database then distribute partitions on clusters. The partitioning of a global schema into fragments can be performed in two different ways:
vertical partitioning and horizontal partitioning. This paper is concerned with vertical partitioning [1]. Partitioning based on attributes has been studied earlier in [5], [6], [7]. Navathe et al used a two-step approach for vertical partitioning. In the first step, they used the given input parameters in the form of an Attribute Usage Matrix (AUM) to construct an Attribute Affinity Matrix (AAM) for clustering [8]. After clustering, an empirical objective function is used to perform binary partitioning iteratively. In the second step, estimated storage cost factors are considered for further refinement of the partitioning process.
Eltayeb Salih Abuelyaman [1] proposes a scheme for vertical partitioning of a database at the design cycle. The scheme determines the hit ratio of a partition. As long as it falls below a predetermined threshold, the partition is altered. Although no proof is provided, experimental data showed that moving an attribute that is loosely coupled to a different Subset within a partition improves hit ratio. We use Eltayeb Salih Abuelyaman [1] algorithms for partitioning database. Only we change his mathematical formula to fuzzy sets as reduce The time of algorithm, also proposed algorithm can used in nondeterministic environment and on sensor networks. We know that skew distributing database on sensors cause inordinate use of some sensors, therefore reduced the life time of network, in this paper we effort to reduce the skew

of distributing database partitioning as possible insofar. The rest of this paper is organized as follows: In the next section we will introduce the related works. In section 3 we will discuss fuzzy sets, in section 4 we will introduce startphase. Simulation is given in Section 5. The conclusion is presented in sections 6.

2. Related works

Because of the criticality of the database performance, several researchers have contributed enormously to vertical partitioning.. Database partitioning has been applied in centralized relational databases [16,19,25,32,35], distributed databases[15,17,19,27,29,33], Data Warehouse Design [20,22,26], and Object-Oriented Database design [21,23]. Hoffer and Severance [25] consider the vertical partitioning problem by applying bond energy algorithm on similarity of attributes, which are based on access patterns of transactions. Their work was extended by Navathe, Ceri, Widerhold, and Dou [30] by presenting vertical partitioning algorithms for three contexts: a data base stored on devices of a single type; in different memory levels; and a distributed database. They used affinity between attributes for partitioning, which is based on number of disk accesses. An alternate graphical approach was proposed by Navathe and Ra [31]. Cornell and Yu [19] used an optimal binary-partitioning algorithm to obtain vertical partitioning, which is iteratively applied to obtain more partitions. The study uses number of accesses to evaluate partitions. Chu and Ieong [18] develop a transaction-based approach to vertical partitioning, in which transaction rather than attribute is used as the unit of analysis. Song and Gorla [35] used genetic algorithms to obtain solutions simultaneously for vertical partitions and access paths for those partitions. They also used the number of disk accesses as the partitioning evaluation criterion. Cheng, Lee, and Wong [17]use genetic searchbased clustering algorithm based on traveling salesman problem to obtain vertical partitions in distributed databases. With reference to object-oriented database design, Gorla [23]used genetic algorithm to determine the instance variables that should be stored in each class/ subclass in a subclass hierarchy, so that the total cost of database operations is minimized. More recently, Ailamaki et al [14] proposed Partition Attributes Across (PAX) model by improving cache performance, while Ramamurthy et al [34] proposed fractured mirrors partitioning scheme based on Decomposition Storage Model and N-ary Storage Model. Fung, Karlapalem, and Li [21] analyze vertical partitioning of classes/ subclasses for class composition hierarchy and subclass hierarchy and develop the associated cost functions for query processing under the cases of large memory and small memory availability. Ng et al [32] proposed a combined vertical partitioning and tuple clustering using genetic algorithm. We extend previous research of single relation cases by providing a procedure for vertical partitioning of relations in a multi-relation database environment. An important characteristic that distinguishes multi-relation schema from single relation case is referential integrity constraints enforcement due to update transactions. Our approach makes use of a 2-attribute affinity, as used in previous studies of Navathe et al [30] and Cornell and Yu [19]. However, we differ from their approach in that our attribute affinity metric is based on differential access times of transactions rather than number of disk accesses. There is a substantial difference between these two methods of evaluation, since fetching an additional block of records from disk in a sequential scan takes much less time than fetching an arbitrary block randomly, which takes even less time than the time for inserting/deleting a record. Our access time computations are baaed on disk IO service times. Furthermore, our algorithm is similar to “hill climbing” [24] in that our algorithm groups attributes such that the objective function keeps increasing; our approach differs theirs in that we have two steps to our algorithm – grouping and verifying. Thus, we extend previous

research on vertical partitioning by including referential integrity constraints and join transactions, and by using a comprehensive cost function to evaluate fragmentation scheme that is based on access time rather than count of accesses. Physical database design provides truly optimal performance when the design is made to fit specific disk characteristics and is only optimal on the given hardware architecture [28]. Our proposed methodology includes disk access characteristics as part of our attribute affinity measure.

2.1 Vertical partitioning

The vertical partitioning problem in a multi-relation environment is stated as follows: Given a relational schema, the retrieval/update/join transactions on the schema, the referential integrity constraints among relations, and the disk access parameters, the objective is to determine stored fragments for each relation, which results in the minimum total database access costs. The partitioning problem is computationally complex. Consider a relational schema with N relations, with Ai attributes for relation i. A relation with A attributes can be partitioned in B(A) different ways [16], where B(A) is the Ath Bell number (for A=30, B(A) = 1015). Using exhaustive enumeration, the number of possible fragmentations for the N-relation schema is approximately B(A1)B(A2) ... B(AN). Yu et al [34]find out that the number of attributes for base tables and views in a typical relational environment are 18 and 41 respectively. Even if we consider a small schema of 10 relations with 15 attributes per relation, the number of possible fragments is approximately (109)10 =1090. Since the problem is intractable, solving large problems requires the use of heuristic techniques. Our procedure consists of three steps.
First, database transactions on the logical schema are transformed into transactions on individual relations. Second, an attribute grouping benefit index (AGBI) is computed. Third, a clustering algorithm using AGBI is applied to derive effective fragments.

2.3 Graph based vertical

A new algorithm has been developed by Navathe and Ra based on a graphical technique (Navathe and Ra, 1989). This algorithm starts from the attribute affinity matrix by considering it as a complete graph called the “affinity graph” in which an edge value represents the affinity between the two attributes,
and then forms a linearly connected spanning tree. By a “linearly connected tree” we imply a tree that is constructed by including one edge at a time such that only edges at the “first” and the “last” node of the tree would be considered for inclusion. We then form “affinity cycles” in this spanning tree by including the edges of high affinity value around the nodes and “growing” these cycles as large as possible. After the cycles are formed, partitions are easily generated by cutting the cycles apart along “cut-edges”.

The major feature of this algorithm is that all fragments are generated by one iteration in a time of O(n2) that is more efficient than the previous approaches.

Recalling table 1 of page 3, the attribute usage matrix for a relation containing 10 attributes with respect to 8 transactions, namely, T1 through T8 that are initiated by the applications. Table 2 shows an example of an attribute affinity matrix. Figure 1 shows the result of applying the algorithm to the attribute affinity matrix. In Figure 1 the nodes refer to attributes of the relation. The resulting vertical fragments are: 1. (a1, a5, a7) 2. (a2, a3, a8, a9) 3. (a4, a6, a10)

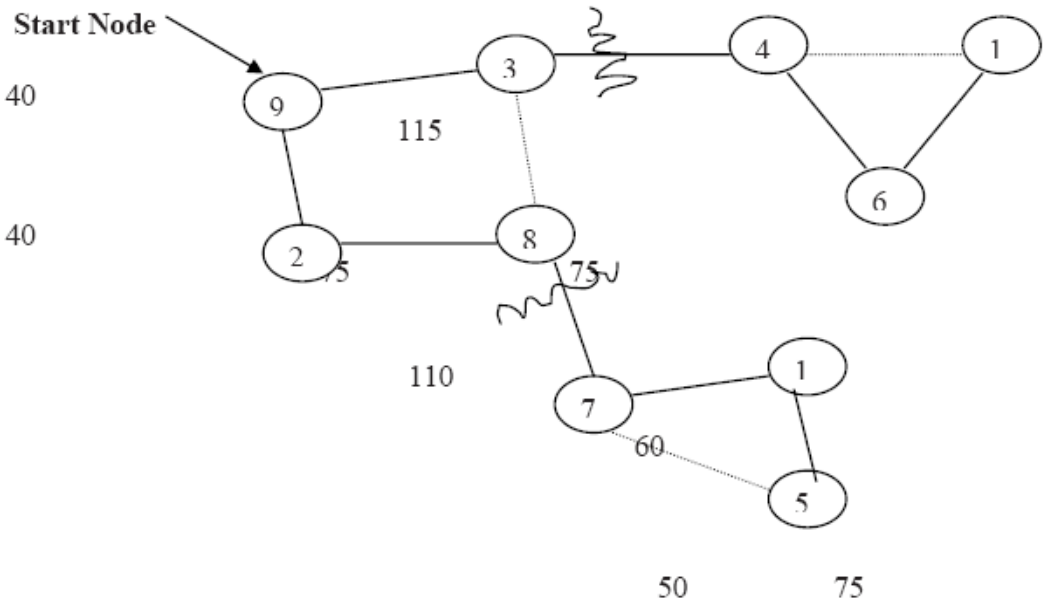

Figure 1.Vertical fragments generated by graph theoretic algorithm

We can summarize the major advantages of this method over the previous approaches in the following:

1. There is no need for iterative binary partitioning. The major weakness of iterative binary partitioning used in (Navathe, Ceri, Weiderhold, 1984) is that at each step two new problems are generated increasing the complexity; furthermore, termination of the algorithm is dependent on the discriminating power of the objective function.
2. The method obviates the need for using any empirical objective functions as in (Navathe, Ceri, Weiderhold, 1984). As shown by (Cornell and Yu, 1987) the "intuitive" objective functions used in (Navathe, Ceri, Weiderhold, 1984) do not necessarily work well when an actual detailed cost formulation for a specific system is utilized.
3. The method requires no complementary algorithms such as the SHIFT algorithm of (Navathe, Ceri, Weiderhold, 1984) that shifts the rows and columns of the affinity matrix.
4. The complexity of this approach is O (n2) as opposed to O (n2log(n)) in (Navathe, Ceri, Weiderhold, 1984).

3. Fuzzy sets overview

Fuzziness [7] is a way to represent uncertainty, possibility and approximation. Fuzzy sets are an extension of classical set theory and are used in fuzzy logic. In classical set theory the membership of elements in relation to a set is assessed in binary terms according to a crisp condition- an element either belongs to or does not belong to the set. By contrast, fuzzy set theory permits the gradual assessment of the membership of elements in relation to a set; this is described with the aid of a membership function:

$$\mu \rightarrow [0, 1]$$

The domain of the membership function, which is the domain of concern and from which elements of the set are drawn, is called the '*universe of discourse'*. For example, the Universe of discourse of the fuzzy set 'High Income' can be the positive real line [0, ∞).
The notion central to fuzzy systems is that truth values (in fuzzy logic) or membership values (in fuzzy sets) are indicated by a value on the range [0. 0, 1. 0], with 0. 0 representing absolute false and 1. 0 representing absolute truth. For example, let us take the statement: "Jane is old. " If Jane's age was 75, we might assign the statement the truth value of 0. 80. The statement could be translated into set terminology as "Jane is a member of the set of old people. " This statement would be rendered symbolically with fuzzy sets as :

$$\mu \text{ OLD (Jane)} = 0.80$$

Where μ is the membership function, operating in this case on the fuzzy set of old people, which returns a value between 0. 0 and 1. 0. The modifiers of fuzzy values are called Hedges. To transform the statement, "Jane is old" to "Jane is very old", the hedge "very" is usually defined as follows:

$$\mu\text{"very" } A(x) = \mu A(x) \wedge 2.$$

For example, If μ OLD (Jane) =0. 8 then μ VERYOLD (Jane) =0. 64. Every input value is associated with a linguistic variable. A linguistic variable represents a concept that is measurable in some way either objectively or subjectively, like temperature or age. Linguistic variables are characteristics of an object or situation. For each linguistic. Variable it should be assigned a set of linguistic terms (values) that subjectively describe the variable. Most of the times, linguistic terms are words that describe the magnitude of the linguistic variable, as "hot" and "large", or how far they are from a goal value as in "exact" or "far". Each linguistic term is fuzzy set and has its own membership function. It is expected that for a linguistic variable to be useful the union of the support of the linguistic terms cover its entire domain.

4. Startphase

The design of efficient database systems is not an exception because database partitioning is based on Frequencies of Queries (FOQ). In a distributed database, data must be collected from a large number of queries before partitioning. To avoid as a constraint, dependency on FOQ must be eliminated. One way to do so is to perform database partitioning at the design phase and immediately after completion of the schema. Comfortably, partitioning can be decided even before database tables are populated. For illustration of the proposed partitioning, the following definition will be necessary.

Description:
a) PNa : the total probability of attributes.
b) PNk : the probability of queries in the Set of Kickoff Queries SKQ[1].

5. Simulation phase

The simulator has the following two modules. Each of the modules is discussed separately.
a) fuzzy reflexivity
b) fuzzy symmetry

5. 1. The fuzzy reflexivity module

The module prompts a user to enter values for each of the first three parameters in above Description. The module then prompts the user to enter a percentage C that controls the number of attributes appearing in each query. If the designer enters the value 30 for example, then the module will generate a value of [0. 6, 1] with probability of 0. 3 and a value of [0, 0. 5] with probability of 0. 7 every one of entries in fuzzy RM matrix represent probability of existence one attribute in a query.

Table1. A randomly generated fuzzy Reflexivity Matrix

Attribute / Query	A	B	C	D	E	F	G	H
a	2 .0	8 .0	2 .0	2 .0	9 .0	1 .0	7 .0	6 .0
b	2 .0	1 .0	9 .0	7 .0	2 .0	9 .0	1 .0	4 .0
c	7 .0	1 .0	9 .0	7 .0	2 .0	1 .0	1 .0	4 .0
d	1 .0	1 .0	1 .0	7 .0	7 .0	9 .0	9 .0	7 .0
e	1 .0	9 .0	9 .0	3 .0	8 .0	9 .0	9 .0	4 .0
f	1 .0	1 .0	1 .0	3 .0	2 .0	2 .0	4 .0	3 .0
g	1 .0	9 .0	8 .0	3 .0	2 .0	8 .0	7 .0	6 .0
h	1 .0	2 .0	7 .0	9 .0	9 .0	8 .0	5 .0	4 .0

The (see table. 1) provides the relationship between queries and attributes

5. 2 The fuzzy symmetry module

The following equations were used to compute the fuzzy *Symmetry Matrix* (SM) on (see table. 2) which defines the desired relationships among attributes. There we change Eltayeb Salih Abuelyaman [1], mathematical formula to fuzzy formula that firstly reduces the time of his algorithm and secondly we can use this formula in nondeterministic environment. Our proposed fuzzy based algorithm is:
Algorithm (1):

$$SM[i,j] = MAX_{K=1}^{N}(Min(RM[k,i], RM[k,j])) \quad \text{for j=1 to Na}$$

$$\text{for i =1 to Na (for j= 1 to Na)} \quad i \neq j$$

Table 2. fuzzy Symmetry Matrix generated from the Reflexivity Matrix and equations 1

Attribute / Query	A	B	C	D	E	F	G	H
a	7 .0	2 .0	7 .0	7 .0	2 .0	2 .0	2 .0	4 .0
b	2 .0	9 .0	3 .0	3 .0	8 .0	9 .0	9 .0	6 .0
c	7 .0	9 .0	7 .0	7 .0	8 .0	9 .0	9 .0	6 .0
d	7 .0	3 .0	9 .0	9 .0	9 .0	8 .0	7 .0	7 .0
e	2 .0	8 .0	9 .0	9 .0	9 .0	8 .0	8 .0	7 .0
f	2 .0	9 .0	8 .0	8 .0	8 .0	9 .0	9 .0	7 .0
g	2 .0	9 .0	7 .0	7 .0	8 .0	9 .0	9 .0	7 .0
h	0. 4	6 .0	6 .0	7 .0	7 .0	7 .0	7 .0	7 .0

for example, for computing SM[E,F] in (see table. 2), from (see table. 1), i=E, j=F and E=(0. 9,0. 2,0. 2,0. 7,0. 8,0. 2,0. 2,0. 9) و F=(0. 1,0. 9,0. 1,0. 9,0. 9,0. 2,0. 8,0. 8) from formula (1) we have : RM[E,F]=MAX(0. 1,0. 2,0. 1,0. 7,0. 8,0. 2,0. 2,0. 8)=0. 8 RM[E,F] represent the percent of query's that have attribute's e and f.

5.3. Partition forming state

In general, the success of an algorithm depends on the strategy that sets criteria for choosing the best start point and the smartest move thereafter. we used the fuzzy SM on (see table. 2) to produce the partition P.
Initially As is equal to {(A, B, C, D, E, F, G, H)}.
from Eltayeb Salih Abuelyaman[1],partitioning Strategy and fuzzy SM matrix :
(a) S = {A} and As = {(B, C, D, E, F, G, H)}
(b) S = {(A,C)} and As = {(B, D, E, F, G, H)}
(c) S = {(A,C,F)} and As = {(B, D, E, G, H)}
(d) S = {(A,C,F,D)} and As = {(B, E, G, H)}
P = { (A, C, D, F) ; (B, E, G, H) }

In this state we compute the hit ratio using algorithm(1) if the value of hit ratio less than predetermined threshold (51%) then we move attribute's that closely coupled with partition to an other partition.

We use Eltayeb Salih Abuelyaman[1], algorithm to computeing hit ratio:Algorithm(1)
1. Compute the partition hit ratio (PHR)
2. If PHR is less than the predefined threshold
then
a) Find the attribute with the minimum hit to miss ratio and move it to a different subset using the attribute association table in the process
b) Repeat from step (2)
3. End partitioning

Table 3. Attribute associate for P

	A	B	C	D	E	F	G	H
hit	6 .1	2. 3	2. 3	2. 2	2. 3	1. 9	2. 4	2
miss	1	2. 3	3. 2	2. 6	2. 7	3. 3	2. 7	2. 4

Hit ratio computing formula is: (hit/ (miss+hit))%100
result's equal (%45), less than preditermined thresold (%50) thus the partitioning will be changed (see table. 3). We know that skew distributing database on sensor cause inordinate use of some sensors as reduce the life time of network, therefore Now we move C attribute to an other partition and againe compute the new hit ratio.
P = {(A, C, D, F); (B, C, E, G, H)}

Table 4. Attribute associate for new P

	A	B	C	D	E	F	G	H

hit	0. 9	4. 1	3. 2	1. 5	1 .3	1	3 .3	2. 6
miss	1. 7	4 .1	2. 3	3. 3	1. 9	4. 2	1. 8	1. 8

the new hit ratio is become (%52) that greater than preditermined threshold (%50).
Total partitioning is (see table. 4):
New P = {(A, D, F); (B, C, E, G, H)}
Now we distribute the (A, D, F) on one cluster and (B, C, E, G, H) on an other cluster.

6. Conclusion

In this paper we propose Fuzzy based approach for Load balanced distributing database on sensor Network. Same as we know that skew distributing database on sensor cause inordinate use of some sensors as reduce the life time of network in proposed algorithm, fragment's replaced in suitable cluster as reduce the energy consumptions of sensors and prolonging network lifetime. The result of our proposed algorithm evaluated in (see table. 3 and table. 4) that suitable for databases that attributes in query nondeterministic, for this reason we change the Eltayeb Salih Abuelyaman [1] algorithm's.

References

[1]. Eltayeb Salih Abuelyaman.: An Optimized Scheme for Vertical Partitioning of a DistributedDatabase. IJCSNS International Journal of Computer Science and Network Security, VOL. 8 No. 1, January (2008)
[2]. A. A. Abbasi, and M. Younis.: A Survey on Clustering Algorithms for Wireless Sensor networks. Computer Communications 30: 2826-2841(2007)
[3]. O. Younis, M. Krunz, and S. Ramasubramanian.: Node Clustering in Wireless Sensor networks: Recent Developments and Deployments Challenges. IEEE Network, May/June (2006)
[4]. D. Wei, and H. A. Chan.: Clustering Ad Hoc Networks: Schemes and lassifications. IEEE (2006)
[5]. H. Abdalla and M. AlFares.: Vertical Partitioning for Database Design: A Grouping Algorithm. to appear in SEDE (2007)
[6]. M. Özsu and P. Valduriez.: Principles of Distributed Database Systems. 2nd edition (1 st edition 1991)
[7]. L. A Zadeh.: Fuzzy sets. Information and control. vol. 8,pp. 338-353 (1965)
[8]. S. Navathe, S. Ceri, G. Weiderhold, and J. Dou. Vertical Partitioning Algorithms for Database Design ACM Transactions on Database Systems, Vol. 9, No. 4(1984)
[9] Braginsky, D. and Estrin, D., *Rumor Routing Algorithm For Sensor Networks*, First Int'lWorkshop onWireless Sensor Networks and Applications, Atlanta, Georgia, 2002.
[10] Intanagonwiwat, C., Govindan, R. and Estrin, D., *Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks*, Proc. of ACM Conference Mobile Computing and Networking (MobiCOM), Boston,MA, August, pp. 56-67, 2000.
[11] Mainwaring, A., Polastre, J., Szewczyk, R. and Culler, D., *Wireless sensor networks for habitat monitoring*, First Int'l Workshop on Wireless Sensor Networks and Applications, Atlanta, Georgia, September 28, 2002.
[12] Cerpa, A., Elson, J. , Estrin, D., Girod, L., Hamilton, M. and Zhao, J., *Habitat monitoring: Application driver forwireless communications technology*, Proc. of the 2001 ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean, April, 2001.
[13] Kahn, J.M. and Katz, R.H. and Pister, K.S.J., *Next Century Challenges: Mobile Networking for "Smart Dust"*, International Conference on Mobile Computing and Networking (MOBICOM), 1999, pp. 271–278.
[14]. Ailamaki, A; Dewitt, D.J.; Hill, M.D. and Skounakis, M.,"Weaving Relations for Cache Performance," *Proceedings of the 27th VLDB Conference*, 2001
[15]. Baiao, F; Mattoso, M and Zaverucha, G., "A Distribution Design Methodology for Object DBMS," Journal of Distributed and Parallel Databases, 16 (6), 2004, 45-90
[16]. Ceri, S., Navathe, S., and Wiederhold, G., "Distribution Design of Logical Database Schemas", IEEE Trans. Soft. Eng. SE-9, 4, (July 1983)
[17]. Cheng, C-H; Lee, W-K; Wong, K-F, "A Genetic Algorithm-Based Clustering Approach for Database Partitioning," *IEEE Transactions on Systems, Man, and Cybernetics,* 32(3), 2002, 215-230.
[18]. Chu, W. W. and Ieong, I.T., "A Transaction-Based Approach to Vertical Partitioning for Relational Database Systems", IEEE Transactions on Software Engineering, 19-9, August 1993.

[19]. Cornell, D.W. and Yu, P.S., "An Effective Approach to Vertical Partitioning for Physical Design of Relational Databases", IEEE Transactions on Software Engineering, 16-2, (Feb 1990)
[20]. Ezeife, C.I., "Selecting and materializing horizontally partitioned warehouse views," Data and Knowledge Engineering, 36, 2001, pp 185-210
[21]. Fung, C-w; Karlapalem, K. and Li, Q., "An Evaluation of Vertical Class Partitioning for Query Processing in Object-Oriented Databases," *IEEE Transactions on Knowledge and Data Engineering,*14(5), 2002, 1095-1118.
[22]. Furtado, C; Lima, A.A.B.; Pacitti, E; Valduriez, P. and Mattoso, M., "Physical and virtual partitioning in OLAP database cluster," 17th International Symposium on Computer Architecture and High Performance Computing, 2005, pp 143-150
[23]. Gorla, N., "An Object-oriented database design for improved performance," Data & Knowledge Engineering, 2001.
[24]. Hammer, M., and Niamir, B. "A Heuristic Approach to Attribute Partitioning", ACM SIGMOD International Conference on Management of Data (1979).
[25]. Hoffer, J.A. and Severance, D.G. "The Use of Cluster Analysis In Physical Data Base Design", International Conference On Very large Databases (1975).
[26]. Labio, W.J., Quass, D., and Adelberg, B., "Physical Database Design for Data Warehouses, IEEE Conference on Data Engineering, 1997, pp 277-288.
[27]. Lim, S-J and Ng, Y-K, "Vertical Fragmentation and Allocation in Distributed Deductive Database Systems," Information Systems, vol. 22, No. 1, 1997, pp 1-24.
[28]. Mannino, M.V., Database Design, Application Development, and Administration. McGraw-Hill, Third Edition, 2007
[29]. March, S.T. and Rho, S., "Allocating Data and Operations to Nodes in Distributed Database Design," IEEE Trans on Knowledge and Data Engineering, vol. 7, no. 2., 1995, pp 305-317.
[30]. Navathe, S., Ceri, S., Wiederhold, G., and Dou, J. "Vertical Partitioning Algorithms for Database Design", ACM Trans. Database Syst. 9, 4 (Dec. 1984). 680-710.
[31]. Navathe, S and Ra, M. "Vertical Partitioning for Database Design: A graphical algorithm", Proceedings of ACM SIGMOD, 1989.
[32]. Ng, V; Gorla, N.; Law, D.M. and Chan, C.K., "Applying Genetic Algorithms in Database Partitioning," Proceedings of the 2003 ACM Symposium on Applied Computing (SAC) 2003, pp 544-549.
[33]. Ozsu, M. and Valduriez, P., Principles of Distributed Database Systems, Prentice Hall, 1996.
[34]. Ramamurthy, R; Dewitt, D.J. and Su, Q., "A Case for Fractured Mirrors," *Proceedings of the 28th VLDB Conference*, 2002
[35]. Song, S.K. and Gorla, N., "A genetic Algorithm for Vertical Fragmentation and Access Path Selection" The Computer Journal, vol. 45, no. 1, 2000, pp 81-93.
[36]. Yu, P.S., Chen, M-S, Heiss, H-U, and Lee, Sukho, "On Workload Characterization of Relational Database Environments," IEEE Trans. Software Engineering, vol.18, no. 4, April 1992, pp 347-355.