A Framework for Manipulating Vacuumed Data in Temporal Relational Database

Mohammad Shabanali Fami
Islamic Azad University of Arak
Arak,Iran

Elham Shabanali Fami, Mohammad Ali Montazeri
Isfahan University of Technology
Isfahan, Iran

Mohammad Taghi Isaai
Sharif University of Iran
Tehran,Iran

Abstract— **The Temporal database is one of the databases that manipulate by append-only policy instead of updating in-place. The data in these databases have two main features: valid-time and transaction-time. Since, the data aren't deleted in temporal database; instead they are increasingly expanded and grown up, it's necessary to adopt a mechanism for controlling the volume and capacity of the database. In such a database a large quantity of the information are fetched less, while some are fetched more, so that it is essential to use a vacuuming data method as well as physical deletion technique to control the database volume. In the present research, we introduce an intelligent vacuuming system based on an unintelligent model of SDVMT which attempts to vacuum the data based on the extent of data importance, transaction time and valid time using a distributed middleware platform. The intelligent model increased the accuracy of the unintelligent model. This model behaves intelligently by learning from the behavior of the system administrator, end user and the server's performance. Therefore, the importance of data is identified by analyzing the behavior of end users. In such a process, the servers are classified based on their performance by continuous monitoring of servers and observing the behavior of system administrators in data vacuuming.**

Keywords-temporal databases, machine learning, database models, database design, modeling and management.

I. INTRODUCTION

Temporal database is one of the most common types of databases in that its data have time references. Among many different applications of these databases, Portfolio management systems, Accounting, Banking, Aerology systems and Scheduling can be mentioned. In temporal databases in contrast with other databases, data will never remove from database. It means that temporal database uses append-only policy instead of update-in-place policy of other databases [1].

Temporal databases are tools for information storage and retrieval with the temporal nature [1]. While temporal data have infinite volume, computer systems due to their restriction in resources such as memory, calculation resources and communication resources can store and retrieve finite data. This topic needs specific approaches to deal with temporal data.

Jensen introduced temporal data vacuuming [2]. Skyt studied data management methods for physically removed data [3] and suggested a framework for vacuuming temporal data [4]. Roddick's aim was preventing some relations from removal in vacuuming process, so he searched about schema versioning [5] [6]. He also did researches about data mining on temporal database systems [7]. Jensen presented a framework for vacuuming temporal data. In this framework he vacuumed data base on organization's rules [8]. Grandi studied schema versioning on object oriented databases [9]. Skyt presented a method for removing data based on their features [10]. The whole of these methods classify data into active and inactive categories. Inactive data that stored in lateral storage devices always create costly queries with a lot of problems associated with time and availability.

On the other hand, Temporal databases are very huge because of the append-only politic. When you never delete any tuple from database the volume of database will increase by time. Thus this kind of database is a very large scale database. In this field there is a lot of research that shows how to act on this kind of database.

Today we see the SAN storage systems developed in hardware and we see a lot of developments in software like Google Big Table, Cassandra and so on to deal with this issue. These developments show that there is a big demand to have a larger data storage system. In this paper we did not consider having an expansive database; rather, we are demonstrating why we didn't use these developments and why our idea is needed for temporal databases.

As we know, E. Codd invented the relational model for database management systems [11]. This model is very common and the important point of the model that we consider is that it is row-oriented. In 2004 Google tried to develop a new database management system named Big Table. The major thrust in this kind of database was the column-oriented feature [12]. For using distributed database the idea of column-oriented database developed. It's a very good way that is used in Big Table and Cassandra [13], but there are some points that should be considered about differences between temporal database and other kinds of big databases.

When we are using such database management systems we store data column by column. The nature of temporal data is not similar to this idea because temporal database is a collection of tuples that each one labeled by time stamped and

its attributes need to fetch together. If some parts of data are fetched and some are not fetched, the data will not be useful. Thus we can't use the C-store base database management system to manipulate temporal database, thus, it's necessary to use a model based on a row-oriented model. All of the research on temporal database shows that they only use a row-oriented foundation.

When we see the benefits of memory Hierarchy we try to use this idea in temporal database management to deal with memory limitations. Thus, at first we proposed a model that is combined of a distributed system middle ware and a memory hierarchy with new level conceptual [14]. SDVMT that is contracted of Semi-Distributed Vacuuming Model on Temporal Databases, contrary to previous methods uses three levels for vacuuming temporal data. The first being *active data* level. The data in this level is very critical and fetched with more frequents. The second level is *vacuumed data*, the data in this level is not very critical but the data in this level will store on the distributed network of systems. Therefore, this level has expanded storage. The third level is *offline data*. The data in this level is rarely fetched and this level is the same as inactive data in old models. With the wide storage space in level two this method can serve more requests. This method may face a storage limitation but this will posteriorly accrue unlike old models[15].

For increasing the benefits of the SDVMT model we present an intelligent model that we named as IVMT as abbreviation of Intelligent Vacuuming Model on Temporal Databases. The difference between these two models is the method of selecting data to be assigned to a particular level. Another difference is the data can migrate in IVMT. When we use a better method to know which data will fetched more, our model will work better. User desired data is different each time. Thus in IVMT model data can migrate between levels. This will improve the SDVMT model[15].

In this paper, first we briefly introduce SDVMT model in section II. Then IVMT model will be presented in section III. Then IVMT model will be trained to target the user's behavior as presented. This model is based on allotting more significant data to more powerful servers. In next section performance investigation of this model will be done by referring to performance of methods that is used in the memory hierarchy [14]. We show the comparison between SDVMT and IVMT model by logical sentences, although this is very clear.

II. SDVMT MODEL

In all CM methods, some parts of data have been physically removed from the database and partitioned the data in active and inactive parts. Inactive data is maintained in lateral storage devices, while active data will be remained in the online system. In this method, inactive data will removed from online system physically. The important point is this division based on organizational rules. For example in a hospital system, records of patients that were admitted two years prior will be inactive. This division is not a very good method because some patients may be local and some are not local. Most likely, the local patients will come back and their related data would have been archived. This model does not have good perception about which data should be active and which should be in-active. This model has not enough flexibility to perceive changes that may accrue in the user's desired data[15].

The major issue concerning CM models is that these models need manual support to retrieve data from inactivated data. This is indeed costly for organizations because of technology changes that will require organizations to pay for experts to work on the new technology. When manual support is needed, another problem is the time required for support. An interval of time is needed to seek the inactive data, as well as to activate the inactivated data prior to conducting a search of the inactive data. This is a time-consuming and inefficient method to support critical applications in a hospital setting[15].

The most important problems of the CM Methods were incapability for answering most of the temporal queries and its high response time for other temporal queries. SDVMT method shown in Fig. 1 has been designed to solve these problems with the objective of optimum utilization of resources. In this model, the concepts of distributed systems are used to improve on pre-designed models[15].

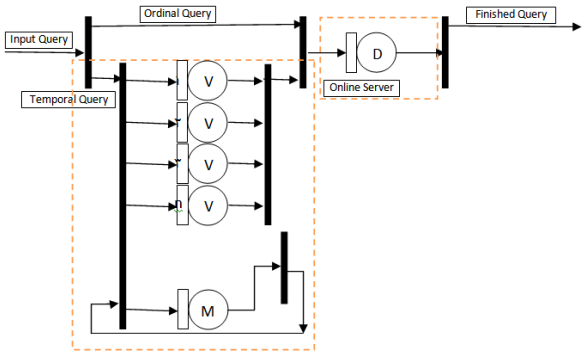

Figure 1. Proposed model for the vacuuming temporal database. In this model, vacuum will be kept actively in vacuum servers.

In this system, the temporal vacuums data rather than being kept in inactive storage resources will be kept in on-line servers. Since most organizations usually provide the appropriate hardware infrastructure that does not allow for optimum use, presenting this model provides a method for utilizing the maximum power of resources to troubleshoot problems concerning serving the users that need information on inactive data[15].

In fact, SDVMT suppose that there is a network in organization that is not too far-fetched. In organizations we always have a lot of PCs that are idle and some busy servers. The level one in the SDVMT model is a server that is costly and limited, and the second level is the idle PCs in the network. These PCs have storage that is needed. For example, the admission's PCs are idle for long periods and only use less than five percent of the processor and storage. Thus, we use a middleware that connects these PCs together by socket connection. Now we have the second level of vacuumed data. This level has a wide storage and this is not costly to the organization. The level three is the same as inactive storage in CM models[15].

The main difference between SDVMT method and CM methods is in optimum usage of organization resources to deliver better services to applicants. More resources are possessed in the SDVMT method. Parallel seeking in vacuums, and scalability of it that obtained from its distributed nature, make higher accountability for this method. If required resources of SDVMT method were not provided, the organization would have to use CM methods. In this situation, however some part of data will be kept inactive, there are more resources to return vacuums and maintain them online for organization[15].

For instance, consider a small hospital that has 20 workstations with normal capabilities along with its online server. This hospital can use its workstations as servers for vacuums. These workstations always have some amount of computational capacity and free storage that can be used for storing and retrieving vacuums data. It is obvious that there are limitations concerning these resources, and after awhile the organization will need inactive storage. Therefore, optimum usage of resources that were costly for the organization, the severity of the problem and the number of inactive vacuums will reduce[15].

In the SDVMT model, rather than lateral storage devices, data will be stored actively in some servers called vacuum servers. Vacuum servers are always slower and weaker than online server but they are much stronger than the manual supports methods. When a temporal query arrives, it is sent to the related vacuum server. Then the online server gathers and combines all results and answers the applicant[15].

III. IVMT MODEL

IVMT is the abbreviation of Intelligent Vacuuming Model on Temporal Databases. The goal of designing this model is to increase the performance of SDVMT model by directing more significant data to more powerful servers. As it is displayed in Fig. 2 this model has three types of intelligence. Firstly, this system is intelligent concerning the user's behavior by checking the user's desired data. Secondly, the system logs the actions of the administrator for automatic migrating data. Thirdly, the system is intelligent about the changes that accrue in server performance. Server performance will change because of data traffic, hardware problems and operating system's fails. So this is very important in identifying which server is important at the time.

In this model we suppose that we have a data atom. Data atom is the smallest significant data in system. Data atom may be a tuple. Data atoms have an integer variable that values increase by user visiting it. So the system will know which data is more important for users. As time passes, this variable decreases. This means that over time the importance of data decreases. Thus, only the newer data will have a high value in this variable and will demonstrate to be more important. Data atoms have two variables for saving the transaction time of data, as well as two variables to save the valid time of data. In this model the administrator can use a command to migrate data from a server to another server by organizational roles. Every server has an indicator that shows how powerful the server is. This indicator is based on network latency of node to the core, node processing usage and free memory in node. In time this indicator will refresh.

Thus, the system knows about user's behavior, the level of power of the server and the behavior of the administrator. By using the neural network engine that is embedded in this model, the system learns the behavior of users and administrators. This model system also knows about the server's status. Thus some data will automatically migrate to other servers to increase performance. This model can uses some open source systems such as hyper table, Hadoop and Hbase to develop. But this paper is only considering introduction of this model.

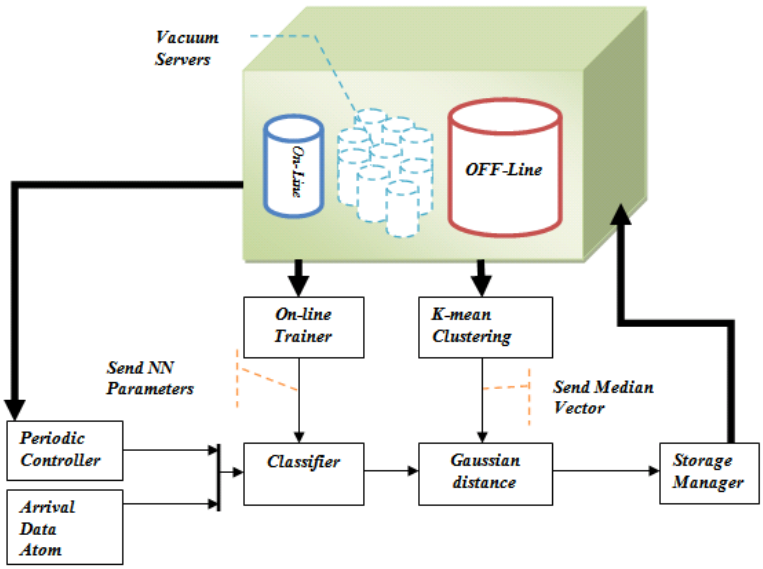

Figure 2. IVMT model block diagram. The IVMT model has a middleware that composed of seven parts. These parts show in above diagram.

A. IVMT Model details

Temporal data have two temporal dimensions, *valid time* and *transaction time* [16]. The importance of each datum is determined by considering the final user's usage. In this paper, the smallest in frangible units of data is called data atom. With regard to the more applicable architecture of BCDM [17] [18], in temporal data models it can be assumed that a data tuple is a data atom. Each data atom has a temporal information header for its data that consists of data valid interval, transaction interval and data significance. Data significance will be calculated regarding user's usage of that data. By retrieving data, its significance will increment by a unit, inserting a tuple add a unit to its significance and deletions increment it. Also by updating a tuple, its significance will increase a +1 unit. As time passes, for each interval distance equal to t, the data's importance will decrease by a unit. K is another parameter in the data header and it specifies the number of data usage to answer a user's question.

It is assumed that temporal data are inputs of artificial neural network. Start valid time, valid time distance, start transaction time, transaction time distance and data importance are inputs of neural network. Stochastic Gradient Descent algorithm is used to train a neural network to support online training [19] [20]. As time passes, the data obtained from user behavior will be used for neural network inputs and it will be trained. In this supervised learning method, it is determined which data atoms are partnered in answering a user's question. In such a way, k is network's output and other parameters of data header are its inputs. In different time slices the system determines which data should be kept in the current server and which data should be transferred to other servers.

With the aim of proving the viability of IVMT method's performance, first the model that is shown in Fig. 1 with three levels of servers must be considered. Then this model can be compared with memory hierarchy. It was practically proved that more applicable data should be maintained in quicker memories, so more applicable information will be maintained in the more powerful server in the IVMT method.

B. Founding a server responsible for maintaining the data in IVMT model

In this area we were faced with unsupervised learning. An index determined the amount of partnership of each data atom in answering a user's query which is obtained from the neural network. K stands for the algorithm used to cluster data to classes whose number is equal to the number of vacuum servers plus one [21] [22]. Each cluster's center can be calculated, therefore, the Gaussian distance of data that do not belong in a cluster from the cluster's center can be obtained. Data will be assigned to a cluster with minimum Gaussian distance and the responsible server for each datum will be obtained.

IV. A COMPARISON BETWEEN SDVMT METHOD AND MEMORY HIERARCHY

With the goal of proving the viability of a method's performance, first the model that is shown in Fig. 1 with three levels of servers will be considered. Then this model can be compared with memory hierarchy using Table I. It was practically proved that more applicable data should be maintained in quicker memories, so more applicable information will be maintained in the more powerful server in the method. The conclusion of this comparison is the superiority of IVMT model than SDVMT model.

In the SDVMT method, users send their ordinary and temporal queries to the online server and wait to be answered. On the other hand, in memory hierarchy the operating system sends arithmetic queries and retrieves information from RAM or Hard Disk to the central processing unit and wait to be serviced. The CPU will answer quickly to queries from its memory, but if the query needs to retrieve information from RAM memory or Hard Disk, the CPU will take some time to transfer information from them to its memory. This time will be referred to as transfer time. Also in the IVMT model, if a temporal query arrives, it needs an online server to gather data from the vacuum server or archive to process them.

In the SDVMT method, vacuum servers make middle memories between online servers and archives to keep temporal data. In memory hierarchy, RAM acts as a middle memory between the CPU and the disk. Vacuum servers deliver services more quickly than archives, just as RAM in the memory hierarchy has a higher speed in accessing data than a disk. Considering cost, maintaining information on disk is less costly than on RAM. Additionally, vacuum servers are more costly than archives.

In the SDVMT method, the archive has a high capacity to store data and is also less costly than other servers, but its swiftness in accessing data is less. In memory hierarchy, disk has more storage capacity to store information. Disk is a less expensive memory than other levels of memory in the hierarchy. With regard to cost, Disk is less expensive than other levels.

The memory hierarchy has existed for a long time from the early usage of computer systems and its good performance has been proven practically. In this research, by using this proved feature, a method for responding to the queries was presented. With respect to the memory hierarchy, when a request for retrieving data is received, at first the data stored in processor memory, such as cache memory, is searched. Next, the data in the main memory will be searched and last, if there is failure in these steps, the data stored in the hard disk will be searched.

In the SDVMT method, at first the data stored in the online server will be searched, then the data stored in vacuum servers will be searched and, finally, archived data will be chosen to search. Since, In terms of memory and speed of response for each of these sectors, the proposed model is equivalent to the memory hierarchy. The IVMT method, similar to the methods used by hierarchical memory in the computer systems, seems to have much better performance than a system that only worked with online server and archived data. These kinds of systems are similar to computer systems without the RAM memory that search a slow hard disk to answer each request. Consequently, this architecture slows down the system and is inefficient.

TABLE I. A COMPARISON BETWEEN SDVMT METHOD AND MEMORY HIERARCHY

SDVMT Method				**Memory Hierarchy**			
Dataset level	***Access Time***	***Storage Capacity***	***Cost***	***Memory level***	***Access Time***	***Storage Capacity***	***Cost***
Level 1 Online Server	Very High	Very Low	Very High	***SRAM, CACHE and Register***	Very High	Very Low	Very High
Level 2 Vacuum Server	High	Low	High	***DRAM&RAM Memory***	High	Low	High
Level 3 Archive	Very Low	Very High	Low	***Magnetic disk***	Medium	High	Medium
An average of Data volume	About Exabyte and increase as time passed			***An average of Data volume***	Some multiple of storage capacity		

Since from the early computerization memory hierarchy was used, viability of its performance is obvious and has been proven. Using the SDVMT method a schema like memory hierarchy approach can be considered as a solution for temporal database systems. Due to insufficient resources in the SDVMT model, efficiency will reduce again after some time. To resolve this problem, intelligence parameter was added to the SDVMT method. Therefore the powerful servers are responsible to store more important data. By using this method, the problem is resolved.

V. CONCLUSIONS

In this study, the CM methods for vacuuming data were reviewed. The SDVMT method improved CM methods by using existing idle resources in the organization but when there were insufficient amount of resources, its response to temporal queries decreased. In the CM models, there are two servers that will be named as active and inactive servers. Data that is stored in an active server, needs less service time but the other data stored in inactive servers needs more service time. Ordinary queries need active server's data while temporal queries need data from active and inactive servers. The result was that ordinary queries need less time to be served, but for some parts of the temporal queries that need data from inactive servers, more service time is required. The server which serves a user query in less time has better performance. Thus the CM model's performance for ordinary queries is good but its performance concerning temporal queries with data on inactive servers is not satisfying.

The IVMT method has three kinds of servers which include online server, vacuum servers and inactive servers. Online server response time is the least. Vacuum server response time is a lot more than online server but inactive servers have longer response time. In the IVMT method, more required data by the final users have more importance. In this method, more important data will be stored on the server with higher performance and vice versa. In this method, ordinary queries will be answered by the online server just as CM methods, and causes less service time and higher performance for the method in this case. Temporal queries should be answered by all three levels of servers. If the query needs data from the online server, service time will be the least and performance will be satisfying. When it needs data from vacuum servers, its response time is a little more but less than the online server. In this situation the performance of the model is less than the previous condition. As a result, this method performance in this case is good but just a little less than the online server situation. The third position occurs when the temporal query needs data from inactive servers. In this method it was assumed that more important data are stored in more efficient servers and user's requests need more important data. This important data are in more efficient servers so that in inactive servers there is a little important datum there, and for these rare data requests, performance is less. If the user needs less important data, inactive servers should be searched so the performance will decrease and the response time will increase. As it was illustrated, just for temporal query from unimportant data the performance is like CM models and for all other situation it works better.

If data importance is increasing, firstly this method is like CM methods, but as time passes this unimportant data will be requested more often and its importance will be increased. At that point the data will transfer to vacuum servers with higher performance. After a period of time, the systems efficiency will increase. In summary, IVMT model is satisfactory in all conditions except for the temporal request for unimportant data. When these types of requests repeat, importance of data will modify. Therefore, by using the online learning method, after a period of time the performance increases. In the end, its performance is weak concerning unimportant data requests.

VI. FUTURE WORKS

This paper just introduces a framework for manipulating vacuuming in temporal database. For developing this framework we can use some open source systems same as Hadoop and Hbase. Before the implementation of this framework we should implement the learning cores. In this core the main point is which feature should be selected. Once the selection is made we can use a neural network to learn the network by using data originating from the administrator and user behaviors. We should then define the instruction sets that nodes in this network will use.

ACKNOWLEDGMENT

We wish to thank everyone who helped us complete this dissertation. Without their continued efforts and support, we would have not been able to bring our work to a successful completion. We specially thank Mrs Kathryn L.Chavez from

Smart Worldwide Solution Pros in Oklahama City, Oklahama, USA, who helped us for editing this research paper.

REFERENCES

[1] C. S. Jensen, “Temporal Data Management” IEEE Trans. Knowledge and Data engineering, Vol. 11, No. 1, pp. 36-44, 1999.

[2] C. S. Jensen, “Vacuuming,” In Proc. The TSQL2 Temporal Query Language, Kluwer, pp. 447-460, 1995

[3] J. Skyt and C. S. Jensen, “Managing Aging Data Using Persistent Views,” In Proc. Int. Conf. on Cooperative Information Systems, Israel, pp. 132-137, 2000

[4] J. Skyt and C. S. Jensen and L. Mark, “A foundation for vacuuming temporal databases” ELSEVIER, Data & Knowledge engineering, Vol. 44, No. 1, pp. 1-29, 2003.

[5] J. F. Roddick, “Schema Versioning,” In Proc. The TSQL2 Temporal Query Language, Kluwer, pp. 425-446, 1995

[6] J. F. Roddick, “Schema Vacuuming in Temporal Databases” IEEE Trans. Knowledge and Data engineering, Vol. 21, No. 5, pp. 744-747, 2009.

[7] J. F. Roddick and M. Spiliopoulou, “A Survey of Temporal Knowledge Discovery Paradigms and Methods” IEEE Trans. Knowledge and Data engineering, Vol. 14, No. 4, pp. 750-767, 2002

[8] C.S. Jensen and L. Mark, “A Framework for Vacuuming Temporal Databases,” Technical Report CS-TR-2516, Univ. of Maryland, College Park, 1990

[9] F. Grandi and F. Mandreoli, “A Formal Model for Temporal Schema Versioning in Object-Oriented Databases,” ELSEVIER, Data & Knowledge engineering, Vol. 46, No. 2, pp. 123-167, 2003.

[10] J. Skyt, C.S. Jensen, and T.B. Pedersen, “Specification-Based Data Reduction in Dimensional Data Warehouses,” In Proc. Int. Conf. on Data Eng, USA, p. 278, 2002.

[11] E. F. Codd, A relational model of data for large shared data banks, Communications of the ACM journal, vol 13, no 6, pp 377-387, 1970.

[12] F. Chang and , Bigtable: a Distributed storage system for structured data, ACM Transactions on computer systems, Vol. 26, No 2,Article no 4, 2008.

[13] A. Lakshman, Cassandra- A Decentralized Structured Storage System, Facebook, 2008.

[14] Koltsidas, Ioannis and Muller, "Sorting hierarchical data in external memory for archiving," In Proc. VLDB Endow., Vol. 1,No 1, pp. 1205-1216, Aug, 2008.

[15] M. S. Fami, E. S. Fami, M. Montazeri and M. Isaai, "Semi-Distributed Vacuuming Model on Temporal Database (SDVMT)", Database Systems Journal, vol III, no 4, pp 73-79, 2012.

[16] R. Sondgrass and I. Ahn, " A Texonomy of time in databases," In Proc. ACM SIGMOD int. Conf. on Management of data., New York, USA, pp. 236-246, May, 1985.

[17] C.S. Jensen, R. Sondgrass and M. Soo,"The TSQL2 Data Model," Kluwer Academic Publishers, pp. 157-240, 1995.

[18] C.S. Jensen, M.D. Soo and R. Sondgrass, "Unifying temporal data model via a conceptual model," Information System, Vol. 11, no. 1, pp. 513-547, 1994.

[19] Bottou and Leon, "Online Algorithms and Stochastic Approximations," Cambridge University Press, Cambridge, UK, 1998.

[20] D.P. Bertsekas, A. Nedic and A.E. Ozdaglar, "Convex Analysis and Optimization," Athena Scientific, 2003.

[21] D. Aloise, A. Deshpande, P. Hansen and P. Popat, " NP-hardness of Euclidean sum-of-squares clustering," Springer Machine Learning, Vol. 75, no. 2,pp. 245-248, 2009.

[22] A. Choromanska and C. Monteleoni, "Online Clustering with Experts," In Proc. Of 15th Int. Conf. on Artificial Intelligence and statistics (AISTATS), 2012.