Expert Systems With Applications 59 (2016) 119–144

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Hair-oriented data model for spatio-temporal data representation

Abbas Madraky[a,*], Zulaiha Ali Othman[b], Abdul Razak Hamdan[b]

[a] Computer Engineering Department, Kavosh University, Mahmoudabad, Mazandaran, Iran
[b] Data Mining and Optimization Research Group (DMO), School of Computer Science, Faculty of Information Science and Technology, National University of Malaysia (UKM), Malaysia

ARTICLE INFO

Article history:
Received 10 January 2015
Revised 21 April 2016
Accepted 22 April 2016
Available online 25 April 2016

Keywords:
Spatio-temporal data models
File size reduction
Query execution time
Hair-oriented data model
Nested tables

ABSTRACT

Having an effective data structure regards to fast data changing is one of the most important demands in spatio-temporal data. Spatio-temporal data have special relationships in regard to spatial and temporal values. Both types of data are complex in terms of their numerous attributes and the changes exhibited over time. A data model that is able to increase the performance of data storage and inquiry responses from a spatio-temporal system is demanded. The structure of the relationships between spatio-temporal data mimics the biological structure of the hair, which has a 'Root' (spatial values) and a 'Shaft' (temporal values) and undergoes growth. This paper aims to show the mathematical formulation of a Hair-Oriented Data Model (HODM) for spatio-temporal data and to demonstrate the model's performance by measuring storage size and query response time. The experiment was conducted by using more than 178,000 records of climate change spatio-temporal data that were implemented in implemented in an object-relational database using nested tables. The data structure and operations are implemented by SQL statements that are related to the concepts of Object-Relational databases. The performances of file storage and execution query are compared using a tabular and normalized entity relationship model that engages various types of queries. The results show that HODM has a lower storage size and a faster query response time for all studied types of spatio-temporal queries. The significances of the work are elaborated by doing comparison with the generic data models. The experimental results showed that the proposed data model is easier to develop and more efficient.

1. Introduction

Spatio-temporal data are complicated and exist in large sets, because this data type is usually allocated to large areas, and the volume of data generated tends to increase over time (Rakêt & Markussen, 2014). Furthermore, real-world geographic objects and their relationships increase the complexity of such data (Parent, Spaccapietra, & Zimányi, 2006a). One important concern regarding spatio-temporal models is the approach used to add the time and space dimensions to the model. This concept refers to the necessary independence among the modeling dimensions of the data structures, namely, space and time (Parent, Spaccapietra, & Zimányi, 1999). However, problem solving is required when using spatio-temporal data in decision making (Triglav, Petrovič, & Stopar, 2011). The demand of having a decision support system for spatio-temporal data leads to an increase in the system's ability about data mining. The use of un-normalized tables for data mining causes data redundancy (Han, Kamber, & Pei, 2006).

The data definition/manipulation concerned on how spatio-temporal data are defined, store and retrieved, the execution time for storing, retrieving and querying the data (Wikle, 2015). Design and development of robust spatio-temporal data structures are the fundamental issues for spatio-temporal data handling. As a first issue, the volume of spatio-temporal data has fast growing and the relationships between spatio-temporal values are also increase data volume due to data redundancy. Data redundancy is the repetition of data leads to data size increasing. If the data size is increased, data manipulation has more difficulties because of the data replication and complexity (Perumal, Velumani, Sadhasivam, & Ramaswamy, 2015). Reducing data redundancy is a general solution for data manipulation problems that leads to decrease data volume and index managing costs. The second issue is about performance of the data model while querying especially response time. The response time problem has occurred due to inappropriate indexing regards to large number of records for data processing. The time based values are gradually added to the spatio-temporal database and the index management will be more

* Corresponding author.
E-mail addresses: madraky@yahoo.com (A. Madraky), zao@ukm.edu.my (Z.A. Othman), arh@ukm.edu.my (A.R. Hamdan).

http://dx.doi.org/10.1016/j.eswa.2016.04.028
0957-4174/

difficult when the number of rows in a table is increased. Therefore, the main reason of query response time problem is about indexing (Guo, Papaioannou, & Aberer, 2014).

Therefore, the reduction of redundancy and inconsistency is an additional issue that should be considered by using suitable storage and retrieval systems. Reliable and on-time query responses are important aspects of very large databases (de Caluwe & Bordogna, 2004). This issue increases in importance with increasing data volume, because consistent query answering is an important data accuracy factor with respect to data redundancy (Del Mondo, Rodríguez, Claramunt, Bravo, & Thibaud, 2013). Additionally, the data access time increases with data volume, resulting in longer-than-expected query response times (Bertossi, 2011). A suitable data structure design is a common solution for data growth problems, because it leads to a reduction in data redundancy and inconsistency, resulting in increased data integrity and reliability (Ramakrishnan & Gehrke, 2003).

In this paper, we use a bio-inspired data model to reduce the above-mentioned problems. Many structures and functions in the various branches of science are designed based on biological principles (Fei & Ma, 2007). Experience has shown that nature adopted ideas that require fewer tests due to the problems being gradually removed due to evolution and optimization over time (Steer, Wirth, & Halgamuge, 2009). Additionally, bio-inspired ideas are interesting and attractive to proposers or users, but these plans also have other benefits in application. One of the advantages of bio-inspired models is understandability, because it is easier to explain a biological model to experts or users due to their previous knowledge of the natural subject (Floreano & Mattiussi, 2008).

Furthermore, system predictability is increased when using a bio-inspired model, because it is possible to forecast behavior or results by comparing the model to the natural system. This ability is very useful for system analysis or risk avoidance (de Castro, 2007). Therefore, many biological models are utilized in computing algorithms or when designing plans for improving or deploying the algorithms (Chiong, 2009). Some famous examples of such models are molecular, DNA-based and nano-scale structures that can be found in the design field and in neural networks, as well as algorithms based on ant colonies, bees and bats that are found in computing fields (Hasançebi, Teke, & Pekcan, 2013; Zomaya, 2006).

Spatio-temporal data possess two main properties. The first property refers to data being in a particular place, because the data are allocated to specific positions due to spatial properties, such as coordinates, GIS specifications and geographic values (Sagar, 2013). The second property refers to the continual insertion of new data over time (Parent et al., 1999). For example, daily temperature values of a region over a month could be saved and processed using a time-series. We will review several data models related to these issues in our concept-related literature study, which includes entity-relational, object-oriented and object-relational conceptual models.

The simplest general data structure for representing data models is the entity-relational category, which is designed based on tables and relationships. MADS (Modeling of Application Data with Spatio-temporal features) is an example of an entity-relational data model that exists in the spatio-temporal environment. The structural dimension of MADS includes attributes, functions, integrity constraints, n-ary relationships, is-a links and aggregation links (Parent et al., 1999). This model has the ability to store and retrieve hierarchical data and semantics of measurements. Using the MADS conceptual model allows for a multi-representation data model, which has been presented in previous research (Parent, Spaccapietra, & Zimányi, 2006b) and is named MurMur. The main goal of the MurMur project has been to propose a new approach to the manipulation of geographical databases. The MurMur query interface is supported by a query editor tool, which provides facilities that can be used to formulate a spatio-temporal query. MurMur also has MADS limitations. In regard to another entity-relational data model, a graph-oriented spatio-temporal data representation has been proposed in previous research (Del Mondo et al., 2013). The model is implemented by extending the relational database specification. Entity-relational data models focus on data manipulation and query processing, but they possess a collection of normalized tables that must be joined, aggregated and transformed in the process of data mining, representing costly and complex tasks (Ordonez, Maabout, Matusevich, & Cabrera, 2013). Previous reports also have not covered object-oriented concepts due to the complexity of spatio-temporal data types.

The second category of data models is object-oriented. This category is based on real-world object properties, which are suitable for spatio-temporal areas. A formal object-oriented data model for temporal information was proposed in 2003 (Grandi & Mandreoli, 2003). Types, subtypes and schema versions of this model are represented by semantic class. A fuzzy spatio-temporal model was presented as an object-oriented data model in previous research (Ribarić & Hrkać, 2011). This model focuses on knowledge representation in high-level Petri nets, and it is suitable for designing a knowledge base in real-time and multi-agent-based intelligent systems. The model may include expert or user human-like knowledge. The main feature of the model is its knowledge representation scheme, called FuSpaT, which supports representation and reasoning for domains that include imprecise and fuzzy spatial, temporal and spatio-temporal relationships. Other fuzzy spatio-temporal models have been presented in previous research (de Tré, de Caluwe, Hallez, & Verstraete, 2003; Sözer, Yazıcı, Oğuztüzün, & Taş, 2008; Verstraete, Tré, & Hallez, 2006). A newer object-oriented data model in the spatio-temporal field was proposed in 2011 for mining monitor-based data (Yang-Ming & Qin-Lin, 2011). This presentation is suitable for an object-oriented programming environment, and the data models in this category are also able to cover complex tasks with a corresponding increase in the costs of time and space (Park, Whang, Lee, & Song, 2001).

Object-relational modeling is the third category, and this approach is designed based on both features of the two previous categories. Most of the recent research in spatio-temporal modeling is concerned with object-relational data models (Chau & Chittayasothorn, 2008; Cuevas, Marín, Pons, & Vila, 2008; Harrington, 2010; Mok, 2007; Philippi, 2005; Yu, Davis, Wilson, & Cole, 2008). A conceptual model for temporal data warehouses was proposed in previous research (Malinowski & Zimányi, 2008). The conceptual model used multidimensional definitions for time values, in terms of other attributes. In 2009, Zhou et al. proposed another object-relational prototype for representation, organization and access to disaster information (Zhou, Liu, Fu, & Zhang, 2009). In 2013, Le et al. presented a geosciences data model that referred to space and time (Le, Gabriel, Gietzel, & Schaeben, 2013). This model is based on topology-generalized maps that use geo-objects, and it also uses a relational structure for events and space-time relationships. Madraky et al. proposed a spatio-temporal data model for data analysis that was named the Hair-Oriented Data Model (Madraky, Othman, & Hamdan, 2012). All of the object-relational data models are compatible with relational DBMS, and they also accept the object-oriented features in relationships and operations.

In this paper, we extend a certain approach to this model and manipulate spatial and temporal objects to reach better performances on the use of storage and data querying. We combine the spatio-temporal specifications with the natural properties and functions of hair to create a better knowledge representation that is based on object-relational features. The specific contributions of our work include a reduction of data redundancy and database size to obtain more integrity and a decrease in query execution time in relevant categories. Additionally, we present a formal definition of

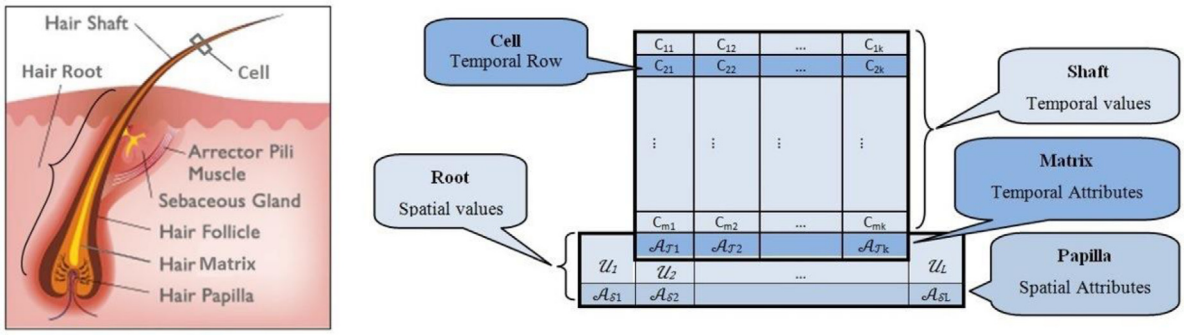

Fig. 1. Similarities between natural hair and hair structure in HODM.

data structure and operations in the proposed model and validate the model by implementing it in a case study.

Scope. This paper focuses on proving the performance of a hair-oriented data model by removing unnecessary redundancy, reducing file size and decreasing query execution times in spatio-temporal data models, which are later used for mining. This paper does not cover issues related to moving objects. This research only covers the definition and implementation of objects.

Outline. The rest of this paper is organized as follows. In Section 2, we define the hair-oriented data model as a spatio-temporal data model based on its attributes and operation specifications. We also present conceptual definitions in this section. Section 3 describes object-oriented and nested-table definitions as common concepts for implementing objects. In this section, we also implement input/output operations in the data structure of HODM by using Oracle-SQL statements regarding a standard data set. Section 4 is concerned with file size reduction and query execution performances in HODM, and Section 5 concludes the paper and discusses future work.

2. Hair-oriented data model

As mentioned in the introduction section, spatio-temporal data has two major specifications, which are location dependency and continual value insertion. Natural hair also has these specifications, because it is allocated to a particular position (the spatial feature), and it grows over time by the insertion of cells (the temporal feature). Other properties, such as strength, direction and type, are similar to spatial properties. Some operations, such as Grow, Cut, Fall, Implant and Wash, have corresponding meanings in the database systems in regard to the manipulation of data. Therefore, the structure of hair and its characteristics are suitable for use in spatio-temporal systems as a bio-inspired model that leads to better understandability and predictability.

To better explain the similarities of spatio-temporal data and hair, we briefly describe the structure of hair. The natural hair structure consists of two major parts, the shaft and the root. The part of the hair that emerges from the skin's surface is called the hair shaft, and the part of the hair that resides within the skin is called the hair root. One of the main parts of the hair root is called the papilla, which leads the generation and growth of the hair. The cells of the hair that are in contact with the dermal papilla are called the hair matrix, and they are responsible for the actual growth of the hair (Mitsui, 1997). The root, shaft and cell terms are used in the proposed data model. In the HODM, the spatio-temporal database includes a number of hairs. Each hair corresponds to a location and stores the spatio-temporal data about that location. The spatial values, such as coordinates and names, are stored in the root, and the temporal values are inserted into the shaft, which consists of a number of cells.

Table 1
Corresponding components of natural hair and HODM.

Natural hair component	HODM component
Root	Spatial values (Related to space)
Shaft	Temporal values (Related to time)
Cell	Temporal row (Event or observation)
Papilla	Spatial definitions
Matrix	Temporal definitions

Fig. 1 shows a schematic structure of a hair as a record in a spatio-temporal database. The connection between natural hair scheme ("www.hairformula37.com,") (left) and record structure (right) is stated in Table 1.

As mentioned in Fig. 1, each hair has spatial and temporal attributes. Spatial attributes specify the values that are related to the position and are chosen based on the requirements and available values. The number of spatial attributes (illustrated by s) depends on the need for query processing or data mining. We show spatial values using the terms R_1, R_2 … R_s, which are stored in the root. The shaft, as an array, is another part of the hair that is used for temporal values. The number of rows (illustrated by m) in the shaft depends on the number of observations or events that we want to save. Each row of the shaft is called a cell. Cells are inserted on the root side of the shaft. Each cell contains a number of attributes (illustrated by k), which are determined by temporal data. One cell of the shaft (C_{21}, C_{22} … C_{2k}) is displayed in Fig. 1.

The HODM is suitable for storing and processing unmovable objects in a spatio-temporal environment. Due to the complexity of real data processing, relational and object-oriented concepts are used in the spatial and temporal information system (Pinet, 2012), so we utilized the object-relational formalism in the structure of this data model. The HODM definitions contain two parts, the data structure and operations.

2.1. HODM attributes

As mentioned in previous research (Madraky et al., 2012), a hair contains a set of data regarding a specific location and determines information about its position. Fig. 2 shows that each hair consists of two parts, including the root and the shaft, which correspond to spatial and temporal attributes, respectively.

2.1.1. Spatial attributes

The root attributes store spatial values for the definition of a place or location. These attributes are declared in terms of requirements and conditions, but one of the attributes is compulsory. The mandatory value is stored in the Position attribute, which determines the coordinate in space. Other spatial attributes, which are displayed in Fig. 2, are optional. According to the specifications of natural hair and mining requirements, we consider other

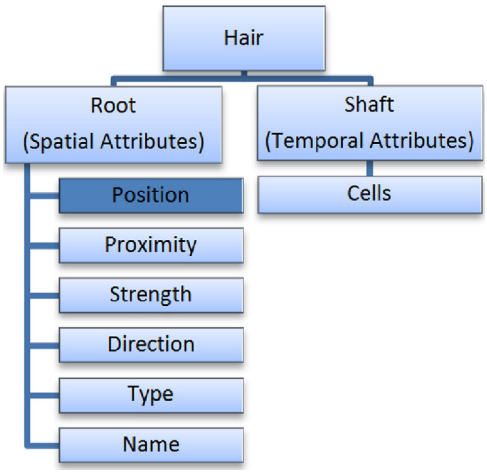

Fig. 2. The data structure view of a hair.

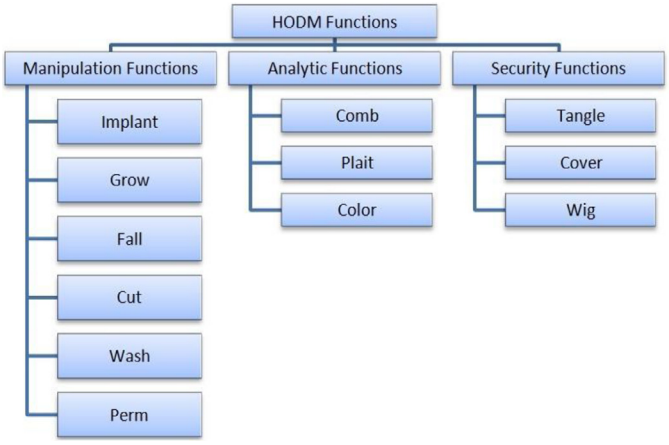

Fig. 3. The operations in the hair-oriented data model.

attributes, including `Proximity`, `Strength`, `Direction`, and `Type`. These attributes describe the properties and qualities of the data. Attribute values can be changed based on the time spent or different usages. The root's attributes for one position may be different from its attributes for another position, but all of the roots have a Position attribute. According to the attribute type, some attributes of the root are linguistic variables. A linguistic variable value is a word or sentence in a natural or artificial language (Zadeh, 1975). Having a realistic framework is the reasoning behind using these types of variables. For example, `Strength` (for determining the importance of position values) and `Proximity` (for defining the relationships among the positions) are linguistic variables because the values of `Strength` can be necessary, consequential, or insignificant, and the values of `Proximity` can be coincident, very near, near, away, etc. It is possible to determine these values numerically, if we have the metrics. We provide detailed specifications of the attributes and their conditions in the following:

`Position`: This attribute contains a pair of values, and it usually is stated in a geographic or geometric form. Geographic coordinates consist of longitude and latitude. They may be represented by degree, minute and second or by other forms of geometric position. This form is typically determined as an ordered pair (x, y). In the HODM, the position of a hair is static and unchangeable. The `Implant` operation defines the values of this attribute for each hair. For example, this operation may assign geometric values for the positioning of a city.

`Proximity`: A set of hairs around a point in a specific area defines the neighborhood of hairs. This attribute is initialized by the `Plait` operation that determines a set of closed locations. This set can be used for clustering or classification that continues or does not continue in an area.

`Strength`: Typically, in information systems, certain data can be more important than other data. The detection of data importance is a useful analytical process that is related to data mining and security preservation. In such systems, data importance is stated by a numerical coefficient. In the HODM, the importance of the point is considered. The default value assigned to this attribute depends on its information characteristics, and after analysis, certain points are chosen as the main points. For example, in the data analysis of climate change, information concerning certain places is more important.

`Direction`: This attribute is utilized for the arrangement of analytical processes. A set of hairs that has a common purpose based on the location of the hairs is defined by a destination or target position that is stated in the `Direction` attribute. The position of the hair can be used as a default value. In this situation, the hair has no direction. The `Comb` operation assigns value to this attribute, which depends on its goals or demands. The `Direction` value is usually made up of coordinates with neighbors, because comb operation works in regard to approximate locations.

`Type`: The characteristics of a hair are determined by this attribute. The characteristics are related to parameters, such as the category of data, the position of a location, the goals of the information, the user access level and surface determination. This parameter can be used for clustering and classification. The value of this attribute is primitive and is assigned while creating a hair, but it can be changed by the `Color` operation.

`Name`: This attribute is used for identifying a point. Although this parameter is not relevant to natural hair, nonetheless, in the computational system, the name may refer to the name of a point. This name can be a number or a string, based on the available values. For example, the name could be the name of a city in a certain country.

2.1.2. Temporal attributes

The remaining attributes are allocated to the shaft. The shaft always has cells that are named after an attribute, but the attribute referred to here is an array with several rows and columns. Each row of this array corresponds to an observation or event. The columns of this array are temporal attributes and are defined based on temporal variables.

`Cell`: Each row in the shaft is inserted into a hair over time. The shaft has one or more `cells`. The structure of a `cell` depends on the data values. For example, when using climate change data, we can define temperature, rain fall and humidity as some of the fields of each `cell`. The `cell's` sequence in the shaft is ordered by the time of insertion. The corresponding values in the rows have the same domain as the table's column. All of the relational definitions, including those of the keys, constraints and dependency rules, are able to define the shaft.

2.2. HODM operations

As illustrated in Fig. 3, the operations in this model are divided into three categories. The first section of operations is concerned with either the input/output data or is represented as manipulation operations, such as operations for the insertion, deletion, or modification of data. These operations are `Grow` (Insert cell), `Cut` (Delete cell), `Fall` (Delete hair), `Implant` (Insert hair), `Wash` (Update hair/cell) and `Perm` (Add/Remove attributes). When using

Table 2
HODM operations.

Manipulation operations	
`Grow`	
Nature role	Over time, the length of a hair increases.
Model role	Inserts cells into the hair from the root side.
Input attributes	Hair.cell
`Cut`	
Nature role	To shorten the hair.
Model role	For deleting old cells or removing useless information without structural modification. Summarization is also achieved by this task.
Input attributes	Hair.cell
`Fall`	
Nature role	Hair loss caused by time or illness.
Model role	For erasing data and the definitions of a position by weakening their importance over time.
Input object	Hair
`Implant`	
Nature role	A surgical technique for increasing the number of hairs.
Model role	For creating a new set of data and definitions that are allocated to a specific location.
Input object	Hair
`Wash`	
Nature role	Removes excess sweat and oil.
Model role	For cleaning noisy and unneeded attributes or values as a preprocessing or updating task.
Input object	Hair
`Perm`	
Nature role	Treat the hair to change the form for various aims such as good looking etc …
Model role	To add or remove spatial and temporal attributes based on requirements.
Input object	HODM
Analytic operations	
`Comb`	
Nature role	Hair combing is used for arrangement.
Model role	For specifying the orientation of information based on specific usage. The classification or clustering of neighboring data is performed by comb function.
Input attributes	Hair.direction
`Plait`	
Nature role	A plait is a complex structure that intertwines three or more strands of human hair.
Model role	For clustering data and data grouping.
Input attributes	Hair.proximity
`Color`	
Nature role	The color of the hair is changed to enhance its aesthetics.
Model role	For selecting data by changing certain data attributes for better data presentation or grouping, without changing the definition.
Input attributes	Hair.type
Security operations	
`Tangle`	
Nature role	Messy and disheveled hair.
Model role	For preserving security by converting data to sloppy cases. (The opposite of comb)
Input attributes	Hair.direction
`Cover`	
Nature role	To cover the hair using a veil, hat, etc.
Model role	For protecting data from unauthorized access or damage.
Input	HODM
`Wig`	
Nature role	A wig is a head of hair made from synthetic materials, wool, human hair, etc.
Model role	For achieving better security by defining non-real data so that the main data are difficult to identify through unauthorized access.
Input	HODM

the `Grow` operation, the new data are inserted into the hair, and by `Cut`, the old data are removed. In this model, data are inserted into the shaft from the root side. Therefore, the new data are located near the root, and the old data are located in the end of the hair. Over time, the importance of certain data (or `strength` of the hair) may be changed, and the data may be erased using the `Fall` operation. The `Implant` operation creates either a hair or a new set of data values and their properties. This operation is opposite of the `Fall` operation. The unneeded and noisy data or attributes can be removed or changed by utilizing the `Wash` operation. Data preprocessing is also performed using the `Wash` operation. The last operation in this category is named `Perm`. This operation is used for add/remove spatial and temporal attributes in the hair and cells, respectively. In Section 3, we explain these operations using SQL Statements.

The second group of tasks is used for data analysis. Comb, Plait, and Color operations are used for analytical or mining processing (Madraky, Othman, & Hamdan, 2014). Using Comb operation, the orientation (classification) of a hair can be changed, and we can define the membership function that is based on classes. We can also arrange the data using this operation. Other advantages of this operation include routing in a GIS environment and finding an answer that is not exactly described in the stored data. The Comb operation modifies the Direction attribute. Plait operation is used for

the designation (clustering) of a specific set of data. This grouping is not static, and it is changeable depending upon the request. The Plait operation uses the Proximity attribute to define the neighborhood positions. The Color operation changes the Type attribute to assign or select positions for grouping or virtual presentation. These operations are used for representation based on classification and clustering tasks and their features, according to the natural specifications of hair. These operations present a new way to record mining results and for the representation of such results according to new parameters. The proposed parameters lead to an expanded query definition and to a decrease in the query response time.

The third operation set includes Tangle, Cover and Wig, which are related to security preservation. In some cases, to provide better security, the regularity of the data is reduced. The Tangle operation eliminates the arrangement of data. This operation is the opposite of the Comb operation. Cover operation makes a protective layer for important data, so that unauthorized persons or software cannot access this data. The number of protective layers can be greater than one. The last security operation is Wig. This operation produces non-real data and stores this data type in a similar manner as the main data, but the authorized user can recognize the real data and use it. Manipulation operations do not affect non-real data. The operations in the HODM are inspired by the specifications of hair. We have described the specifications and properties of the operations in both the real mode and the data model, as shown in Table 2. As mentioned in the scope, this paper does not address analytic and security operations, although we define these operations briefly in Table 2 (Madraky et al., 2012).

2.1.3. HODM conceptual definitions

In the Hair-Oriented Data Model, each position is considered an object and is represented by a tuple that we define as a Hair. A spatio-temporal database is a set of hairs. The key idea of our proposed data model is the design of new abstract attributes for the representation of an analysis, in addition to designing new operations to manipulate, mine and protect the data. These operations are based on the biomaterial structure and properties of hair. To better represent spatio-temporal query processing, the following framework is defined according to its continual change. The following definitions introduce the components of a spatio-temporal data model that are able to address the entities of the hair and both the spatial and temporal attributes.

Definition 1. Database (*HODB*)

The *HODB* is a hair-oriented database that includes the spatial and temporal values of certain positions or locations. The *HODB* is a set of hairs (*H*).

HODB $= \{H_1, H_2 \ldots H_i \ldots H_n\}$, $1 \leq n < \infty$, *with n the number of hairs (locations) in a spatio-temporal database.*

Example 1. We illustrate the interest of the hair-oriented data model through a representative case study that shows climate change at certain locations on the Pacific Ocean's surface, as defined by buoys. Each floating buoy is shown as a hair with spatial and temporal attributes. All of the values for the buoys are structured and stored in a database.

Definition 2. Hair (*H*)

H includes a set $\mathscr{U}$ of possible spatial attributes and values, in addition to possible temporal attributes and values $\mathscr{T}$. The attribute names that accept relevant types of data are (i) the set $\mathscr{A}_{\mathscr{S}}$ of possible spatial attribute names and (ii) the set $\mathscr{A}_{\mathscr{T}}$ of possible temporal attribute names.

H is a spatio-temporal object that is defined by $H = (R, Sh)$, where:

R, or Root, is a two-dimensional domain, $R \subseteq \mathbb{R}^2$, that includes the possible spatial fields that take names in $\mathscr{A}_{\mathscr{S}}$ and values in $\mathscr{U}$.

Sh, or Shaft, is a continuously ordered set of possible temporal attributes that take names in $\mathscr{A}_{\mathscr{T}}$ and values in $\mathscr{T}$.

The three types of *H(R, Sh)*, which depend on the following conditions, are:

(i) ∃ (Hi=(Ri, Sh i) and Hj=(Rj, Sh j)) are called "Totally Compatible" if and only if Ri=Rj, and Sh i=Sh j
(ii) ∃ (Hi=(Ri, Sh i) and Hj=(Rj, Sh j)) are called "Spatially Compatible" if and only if Ri=Rj
(iii) ∃ (Hi=(Ri, Sh i) and Hj=(Rj, Sh j)) are called "Temporally Compatible" if and only if Sh i=Sh j

Where 1< = i, j < = n is the number of records (hair) in a database (HODB)

Example 2. Based on the contents in Example 1, the spatial specifications of a buoy, such as its latitude, longitude and name, are defined in the root (*R*). The temporal values, such as the humidity, temperature and winds, are recorded daily. A set of temporal values is stored by the shaft (*Sh*). The synthesis of the root and the shaft makes a hair (*H*), logically.

Definition 3. `Cell` (*C*)

C is a row of a shaft (*Sh*) that denotes evidence or observations. A shaft is an ordered set of cells, which could be empty. Each cell has temporal values in $\mathscr{T}$. The temporal attributes for all cells in a specific shaft are the same.

C is a record that is related to *Sh* by mapping (m: 1)

$\forall$ Sh $\subset$ H, $\exists$ C$\in \mathscr{T}$: C$\subseteq$ Sh, Sh $_i = (C_{i,1}, C_{i,2}, C_{i,3} \ldots C_{i,m})$,
$1 \leq m < \infty$, with m the number of temporal attributes in a shaft

Example 3. A cell shows climate change evidence at an identical time or time interval. For example, if we consider the first row of Table 6 in Section 3.1, the corresponding values with attributes of Obs, Y, M, D, Date_, ZonWinds, MerWinds, Humidity, AirTemp and SSTemp make up a cell, as follows:

$C_{1,1}$ =(177,915,98,1,1,980,101,–7,–6.7,86,26.19,26.54) [Day 1 for Buoy 1]

The next cell for $Buoy_1$ is recorded in the second row.

$C_{1,2}$ =(177,916,98,1,2,980,102,–6,–6.8,86,26,26.48) [Day 2 for Buoy 1]

The last cell for $Buoy_2$ is recorded in the last row.

$C_{2,15}$ =(103,335,98,1,15,980,115,–7,-4.4,90.9,26.76,28.58) [Day 15 for Buoy 2]

Definition 4. Set operators (Union, Intersection and Difference)

Suppose *A* and *B* are two spatio-temporal relations about one location in two tables, $A, B \subseteq HODB$. A^s, B^s are the spatial set of *A, B,* and A^t, B^t are the temporal set of *A, B, respectively.* The set operators could be any of the following three types:

(1) **Type 1**: The spatial set operators that denote $\cup^s$ for union, $\cap^s$ for intersection and $-^s$ for difference **(for same locations)**
  (i) $A \cup^s B \equiv (A^s \cup B^s, A^t)$
  (ii) $A \cap^s B \equiv (A^s \cap B^s, A^t)$
  (iii) $A -^s B \equiv (A^s - B^s, A^t)$
(2) **Type 2:** The temporal set operators that denote $\cup^t$ for union, $\cap^t$ for intersection and $-^t$ for difference **(for same locations and time stamps)**
  (i) $A \cup^t B \equiv (A^s, A^t \cup B^t)$

```
Algorithm 1: Grow (H, C)
Input: The Hair, the new cell about an event or observation
Output: An updated Hair with a cell inserted

If there is a hair Hi of HODB such that Loc(H) = Loc(Hi) then
        insert (select h.cells from hair h where Loc(H)=(x,y))
        Values(C);
else
        Implant(H);
        Grow(H,C)
Endif
```

Fig. 4. Pseudo-code of grow operation.

```
Algorithm 2: Cut (H, n)
Input: The Hair with one or many cells about events or observations (Shaft), number of cells for deletion
Output: An updated Hair with one or many deleted cells
m: number of cells in H

If there is a hair Hi of HODB such that Loc(H) = Loc(Hi) then
        If m>=n
            for k:=1 to n
                    delete the oldest cell from shaft (select h.cells from hair h
                            where Loc(H)=(x,y));
            end for;
        else
                    'Cells Not Exist'
        endif
else
        'Hair Not Exist'
endif
```

Fig. 5. Pseudo-code of cut operation.

```
Algorithm 3: Fall (H)
Input: The Hair with one or many cells about events or observations (Root & Shaft)

If there is a hair Hi of HODB such that Loc(H) = Loc(Hi) then
        delete from hair where Loc(H)=(x,y));
else
        'Hair Not Exist'
endif
```

Fig. 6. Pseudo-code of fall operation.

```
Algorithm 4: Implant (H)
Input: The Hair with spatial values

If there is a hair Hi of HODB such that Loc(H) = Loc(Hi) then
        'Hair already Exist'
else
        insert hair
        Values(H);
endif
```

Fig. 7. Pseudo-code of implant operation.

(ii) $A \cap^{t} B \equiv (A^s, A^t \cap B^t)$
(iii) $A -^{t} B \equiv (A^s, A^t - B^t)$

(3) **Type 3:** The spatio-temporal set operators that denote $\cup^{st}$ for union, $\cap^{st}$ for intersection and $-^{st}$ for difference **(for same locations and time stamps)**
(i) $A \cup^{st} B \equiv (A^s \cup B^s, A^t \cup B^t)$
(ii) $A \cap^{st} B \equiv (A^s \cap B^s, A^t \cap B^t)$
(iii) $A -^{st} B \equiv (A^s - B^s, A^t - B^t)$

Example 4. These queries could be covered by set operators in regard to previous examples:

- $\prod_{a1,\ldots,an}(R)$ is projection of a table R, where $a_1,\ldots, a_n$ is a set of attribute names.
- $\sigma_{\varphi}(R)$ is selection of a table R, where φ is a condition.

Q1: Calculate $|\overline{Air\ Temp.\ of\ Buoy1} - \overline{Air\ Temp}|$: $H_1 \equiv Buoy1$, $H_2 \equiv Buoy2$
Q1 = Avg $(\prod_{Air\ Temp} \cdot (\sigma_{Name\ =\ Buoy1}\ (H_1)))$- Avg $(\prod_{Air\ Temp} \cdot (H_1 \cup^{t} H_2))$
Q2: Names of the positions in the areas A and B with radius<5′ from H_1

```
Algorithm 5: Wash (Hi, ci)
Input: The spatial part of Hair (Root) without cells / the hair with one or many cells
about events or observations (Shaft),
Output: An updated Hair with one or many cells changed

If there is a hair Hi of HODB such that Loc(H) = Loc(Hi) then
    If Root update are needed
        Update H set Root Attributes = New values;
    endif
    If Cells update are needed
        If update status is all
            for all cells
                Update H.cells set Shaft Attributes = New values;
            end for
        else
            Update H.cells set Shaft Attributes = New values;
        endif
    endif
else
    'Hair Not Exist'
endif
```

Fig. 8. Pseudo-code of wash operation.

```
Algorithm 6: Perm (Add/Remove, H, Attribute)
Input: Add/Remove operation, the spatial part of Hair (Root) without cells / the hair with
one or many cells about events or observations (Shaft), attribute name

Output: Hair modified structure in Root / Shaft part

If there is a hair Hi of HODB such that Loc(H) = Loc(Hi) then
    Alter H add/remove attribute name
else
    'Hair Not Exist'
endif
```

Fig. 9. Pseudo-code of perm operation.

$$Q2 = \Pi_{\text{Name}}(\sigma_{distance\ <\ 5}(A \cap^{S} B))$$

$$distance = acos(sin(H_1 \cdot Latitude) * sin(Latitude) + cos(H_1 \cdot Latitude) * cos(Latitude) * cos(Longitude - H_1 \cdot Longitude)) * 6371$$

Q3: Names of the positions with *Max (Humidity)>90* inside area *A* and outside *B*

$$Q3 = \prod_{\text{Name}} (\sigma_{\text{(Humidity)}>90}\ (A - {}^{st} B))$$

Definition 5. Manipulation operations (`Grow`, `Cut`, `Implant`, `Fall` and `Wash`)

A series of operators and operations are defined to insert, update and delete fields:

Let

H: Hair HODM as particular information about one positio
C: Cell $\mathcal{T}$ and C B as temporal values about evidence
R: Root Hair, R $\mathcal{U}$ as spatial values of a Hair
B: Body Hair, B∋C

(i) Addition evidence (Temporal values): **Grow (H, C) ≡ (R, Sh +C)**
(ii) Deletion evidence (Temporal values): **Cut (H, C) ≡ (R, Sh -C)**
(iii) Addition position (Spatio-temporal values): **Implant (H) ≡ new (H)**
(iv) Deletion position (Spatio-temporal values): **Fall (H) ≡ delete (H)**
(v) Update evidence(s) (Temporal values): **Wash (H,[C]) ≡ update (C)**

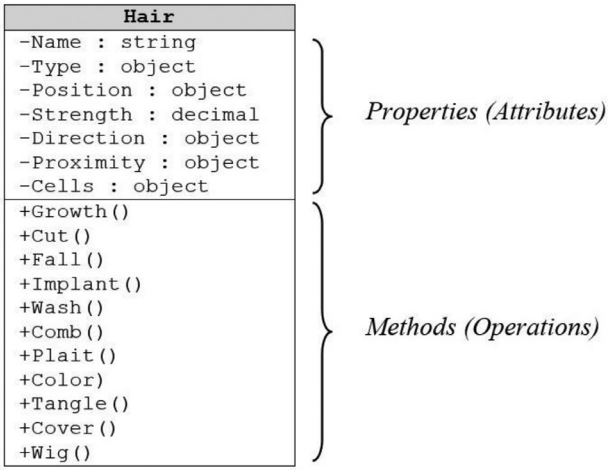

Fig. 10. The attributes and operations of a hair.

Example 5. Manipulation operations

(i) Insert the evidence for buoy 1 on Day = 14.
Grow (Buoy1, C14):
INSERT INTO HM.CELLSTABLE (DAY, ZONWINDS, MERWINDS, HUMIDITY, AIRTEMP, SSTEMP)
VALUES (14, −4.9, −6.4, 81.8, 26.08, 26.81);
(ii) Delete the evidence for buoy 2 from Day = 13.
Cut (Buoy2, C13):
DELETE FROM HM.CELLSTABLE WHERE Day = 13;

Parent Fields (Root) – Record 1				Child Fields (Shaft)			
P_1_Field$_1$	P_1_Field$_2$	...	P_1_Field$_s$	Temporal Attributes (Cells Values for X rows)			
				C_{11}_Field$_1$	C_{11}_Field$_2$	...	C_{11}_Field$_k$
				C_{12}_Field$_1$	C_{12}_Field$_2$	...	C_{12}_Field$_k$
				⋮			
				C_{1x}_Field$_1$	C_{1x}_Field$_2$	...	C_{1x}_Field$_k$
Parent Fields (Root) – Record 2				**Child Fields (Shaft)**			
P_2_Field$_1$	P_2_Field$_2$	...	P_2_Field$_s$	Temporal Attributes (Cells Values for Y rows)			
				C_{21}_Field$_1$	C_{21}_Field$_2$	...	C_{21}_Field$_k$
				C_{22}_Field$_1$	C_{22}_Field$_2$	...	C_{22}_Field$_k$
				⋮			
				C_{2y}_Field$_1$	C_{2y}_Field$_2$	...	C_{2y}_Field$_k$
⋮				⋮			
Parent Fields (Root) – Record i				**Child Fields (Shaft)**			
P_i_Field$_1$	P_i_Field$_2$	...	P_i_Field$_s$	Temporal Attributes (Cells Values for Z rows)			
				C_{i1}_Field$_1$	C_{i1}_Field$_2$	...	C_{i1}_Field$_k$
				C_{i2}_Field$_1$	C_{i2}_Field$_2$	...	C_{i2}_Field$_k$
				⋮			
				C_{iz}_Field$_1$	C_{iz}_Field$_2$	...	C_{iz}_Field$_k$

Fig. 11. Nested table structure view.

Obs	Y	M	D	Date_	Zonwinds	Merwinds	Humidity	AirTemp	SSTemp
177915	98	1	1	980101	-7	-6.7	86	26.19	26.54
177916	98	1	2	980102	-6	-6.8	86	26	26.48
177917	98	1	3	980103	-6.7	-4.9	87.3	26.04	26.27
177918	98	1	4	980104	-6.7	-4.9	87.7	25.96	26.21
177919	98	1	5	980105	-7.3	-4.8	88.1	25.96	26.13
177920	98	1	6	980106	-8	-2.6	87.3	26.08	26.06
177921	98	1	7	980107	-5.7	-2	88.1	26	26.12
177922	98	1	8	980108	-6.1	-3.2	86.9	25.88	26.1
177923	98	1	9	980109	-6	-4	86.9	25.8	26.05
177924	98	1	10	980110	-6.8	-5.3	82.6	25.73	26.01
177925	98	1	11	980111	-7.4	-5.2	82.6	25.69	25.94
177926	98	1	12	980112	-6.9	-6.5	84.7	25.8	25.91
177927	98	1	13	980113	-6.5	-6.1	83.5	25.73	25.87
177928	98	1	14	980114	-7	-6.7	83.9	25.65	25.8
177929	98	1	15	980115	-8.1	-6.3	83	25.76	25.74

Name	Type
Buoy1	Null

Position	Strength	Direction	Proximity
8.95, -140.33	Necessary	8.95, -140.33	Null

Fig. 12. The values for El Nino Buoy1 in HODM structure.

Obs	Y	M	D	Date_	Zonwinds	Merwinds	Humidity	AirTemp	SSTemp
177915	98	1	1	980101	-7	-6.7	86	26.19	26.54
17	Obs	Y	M	D	Date_	Zonwinds	Merwinds	Humidity	AirTemp	SSTemp
17	103321	98	1	1	980101	-4.1	-7.4	90.1	27.46	28.67
17	103322	98	1	2	980102	-3.2	-3.8	94.1	25.65	28.42
17	103323	98	1	3	980103	-4	-6.3	87.6	27.38	28.35
17	103324	98	1	4	980104	-5.2	-4.3	88	27.72	28.41
17	103325	98	1	5	980105	-3.9	-3.5	89.3	27.46	28.55
17	103326	98	1	6	980106	-5.3	-2.5	88	27.22	28.61
17	103327	98	1	7	980107	-4	-5.5	81.2	28.22	28.7
17	103328	98	1	8	980108	-4	-5.1	80	28.19	28.72
17	103329	98	1	9	980109	-3.6	-4	75.9	27.95	28.72
17	103330	98	1	10	980110	-3.1	-3.5	88.9	26.73	28.66
17	103331	98	1	11	980111	-4.4	-6.8	78.7	28.07	28.73
17	103332	98	1	12	980112	-2	-8	79.6	28.07	28.77
17	103333	98	1	13	980113	-2.8	-8.8	83.2	27.84	28.71
17	103334	98	1	14	980114	-5.3	-8.8	86	27.38	28.63
17	103335	98	1	15	980115	-7	-4.4	90.9	26.76	28.58

Name	Type
Buoy1	Null
Buoy2	Null

Position	Strength	Direction	Proximity
4.91, -139.84	Necessary	4.91, -139.84	Null

Fig. 13. The El Nino values for Buoy1 and Buoy2 in HODM structure.

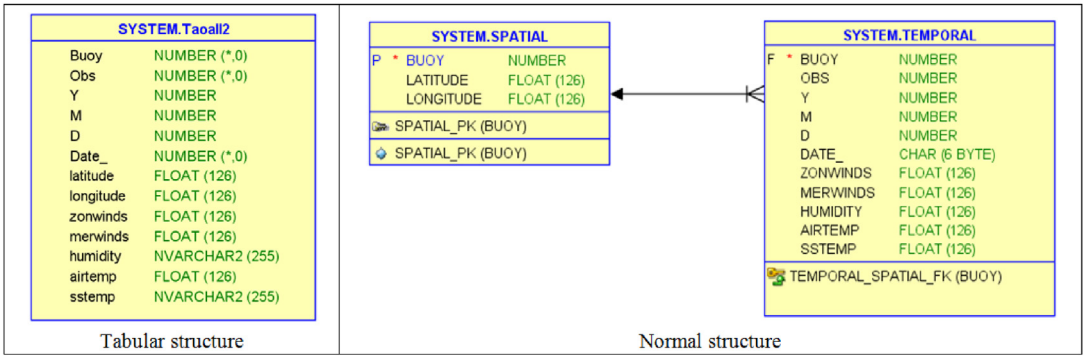

Fig. 14. Tables in tabular and normalize structures for climate change values (Created by Oracle SQL developer data modeler).

Spatial Part (Root)			Temporal Part (Shaft)
Buoy (int)	Latitude (float)	Longitude (float)	Cells (Obs, Y, M, D, Date_, [Zon.Winds], [Mer.Winds], Humidity, [Air Temp.], [S.S.Temp.]) (nested table)

Fig. 15. Attributes in HODM for climate change information.

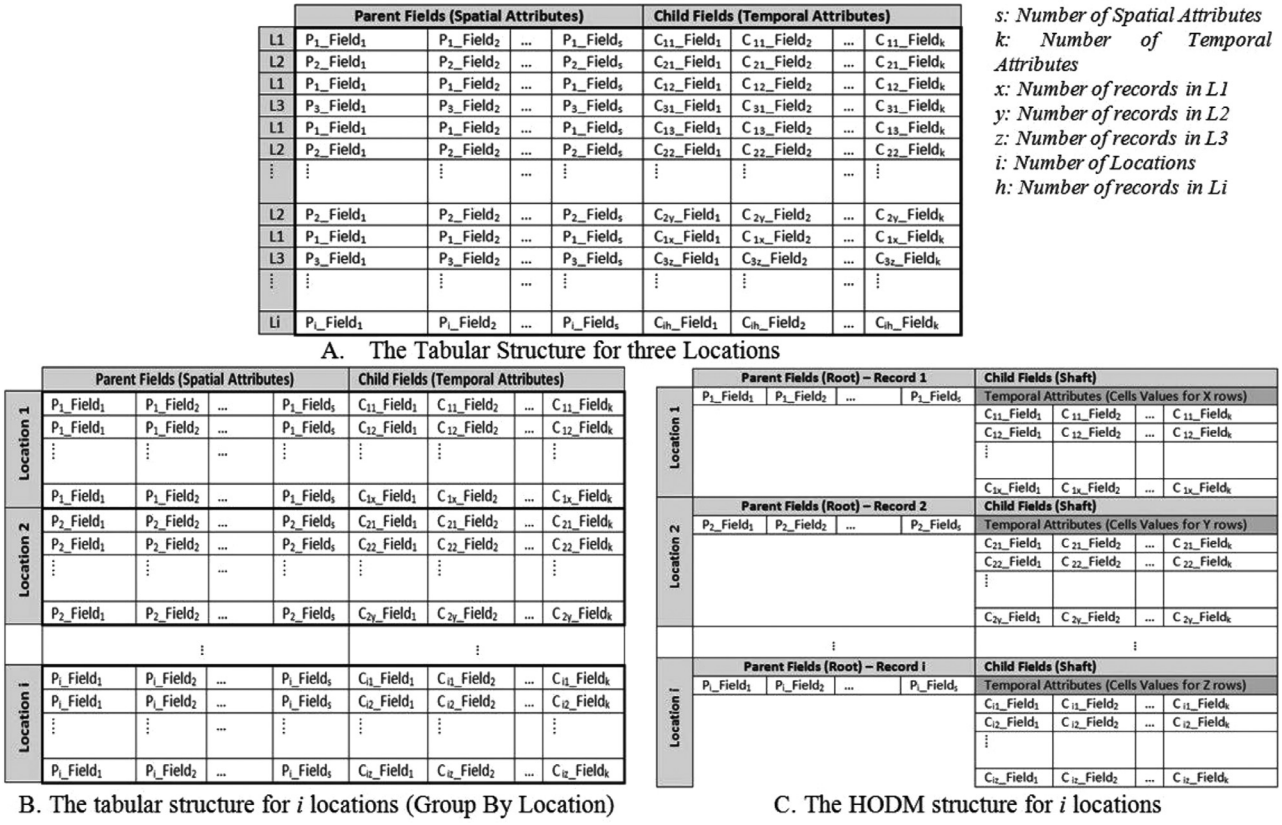

Fig. 16. The model structures for linear search comparisons.

(iii) Define buoy 3 with spatial values.
Plating (Buoy3):
INSERT INTO HM.HAIR (NAME, POSITION.X, POSITION.Y, STRENGTH)
VALUES (Buoy3, 2, -139.95, Null); /∗ Null is the default value for Strength of Hair} ∗/

(iv) Delete buoy4 with all of its spatial and temporal values.
Fall (Buoy4):
DELETE FROM HM.HAIR WHERE NAME = Buoy4;

(v) Convert all of the values for AirTemp for buoy1 from Celsius to degrees in Fahrenheit.
Wash (Buoy1, All):
UPDATE HM.HAIR SET AIRTEMP = AIRTEMP∗(9/5) + 32 WHERE NAME = Buoy1;

Above mentioned operations will be defined by designing some algorithms. In the following algorithms, we use Loc function to show the coordinates of a hair where:

$$\mathrm{Loc}(H) = (x, y) \tag{1}$$

Based on the conceptual definition, the manipulation algorithms for an event or observation about point are shown in Figs. 4–9

Definition 6. Mining Representation Operations (Comb, Plait and Color)

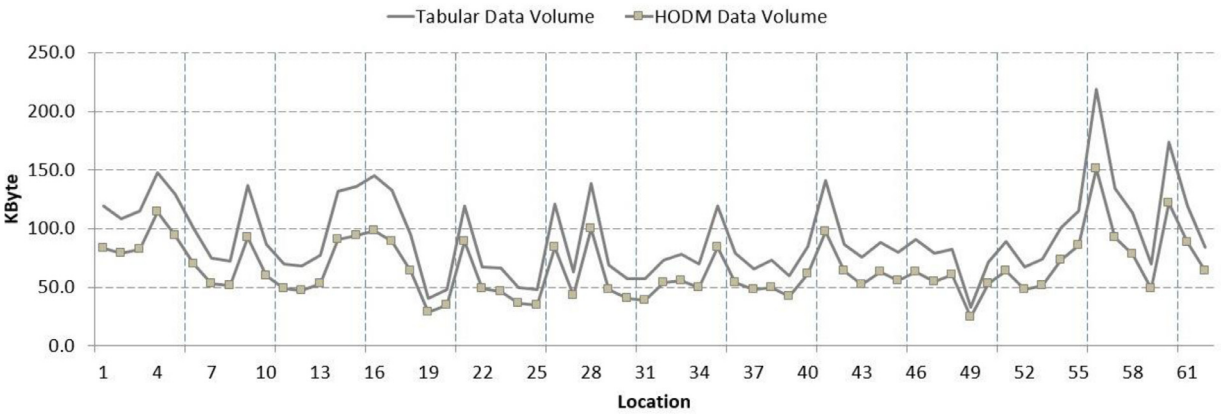

Fig. 17. A comparison of the file size in tabular and HODM in terms of positions.

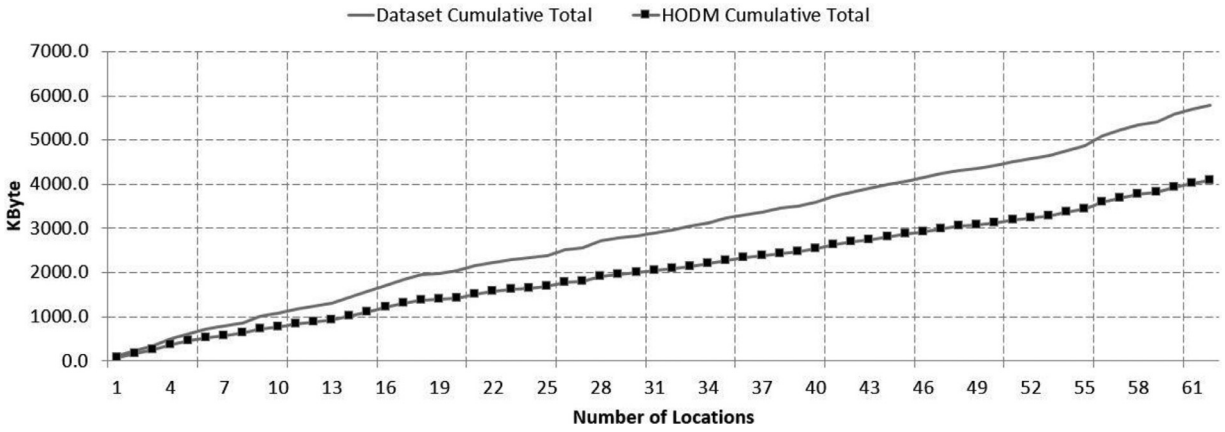

Fig. 18. A comparison of the cumulative file size in tabular and HODM.

Update Orientation (Spatial Values):

(i) *Comb (H, Direction) ≡ update (Direction⊆ R)*
The `Comb` operation defines the direction or orientation of a hair. `Direction` is an analytical attribute that determines the orientation of the data. Using the orientation, we can define the data groups or classifications.

Update Neighborhood (Spatial Values):

(ii) *Plait (H, Proximity) ≡ update (Proximity ⊆ R)*
`Plait` operation defines a region as a specific cluster through the use of neighborhoods of hair. This operation defines a group of data and allocates a hair to a group as a member.

Update Type (Spatial Values):

(iii) *Color (H, Type) ≡ update (Type ⊆ R)*
`Color` operation, like `Plait`, is used for grouping a specified set of hairs; however, `Color` operation is also able to cluster certain points that are scattered and not close to one another.

Definition 7. Security Operations (Tangling, Cover and Wig)
Change Direction (Spatial Value):

(i) *Tangle(H, Direction)≡ Random (Direction⊆ R)*

Change Access (Spatio-Temporal values):

(ii) *Cover (HODM) ≡Invisible (Database)*

Change Database (Spatio-Temporal values):

(iii) *Wig (HODM) ≡new (Database)*

3. Implementation

The implementation of the hair structure and associated operations is followed by object-oriented modeling. Fig. 10 shows the hair class that represents an object. In the proposed model, each point is defined by a hair. Each hair has certain properties or attributes that are divided into spatial and temporal variables. The spatial properties include the `Name`, `Type`, `Position`, `Strength`, `Direction`, and `Proximity`; these properties are dependent on a place or location. Some of these attribute types are complex and defined by specific data types. For example, the `Proximity` attribute is declared as an array that detects the neighbors of a Hair, and the `Position` attribute defines the geometry values that cover the coordinates of a particular point. Evidently, similar to whole structures that are inspired by nature, this model requires additional parameters to achieve more practical approaches. The `Name` attribute is an example of an additional parameter that can be added to the initial parameter set. This attribute is utilized to allocate the name of a location. For example, the `Name` could allocate the name of a city in the population information system, the name of a weather station in the climate change system or the name of a four-way in the urban transit system.

Temporal values, such as a time stamp or any other value that includes time, are stored in the `Cell` attribute. The data structure of the `Cell` is defined by the nested table concept. If we consider two tables as including a case table (parent table) with a primary key and a nested table (child table) with a foreign key, then a nested table is represented in the case table as a special column. For any particular case row, this kind of column contains

```
Q1: Display data values of each location
-----------------------------------------
/* Tabular Query */
SELECT * FROM Taoall2 WHERE Buoy=b;

/* Normal Query */
SELECT t.* FROM temporal t WHERE t.Buoy=b;

/* HODM Query */
SELECT c.* FROM hm.hair, table(cells) c WHERE name=b;
```

Fig. 19. A basic query.

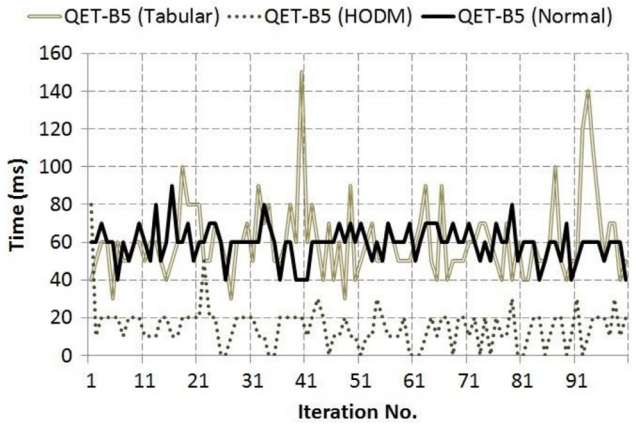

Fig. 20. Query execution time for particular locations (Buoy = 5).

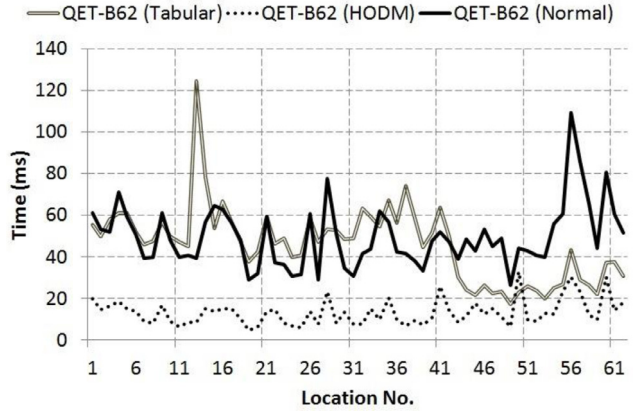

Fig. 21. The average query execution times for the 62 locations.

selected rows from the child table that pertain to the parent table. We present hairs according to the nested table structure in Fig. 11. Spatial values are inserted into parent fields, and temporal values are stored in child fields. The child field has several rows for each parent row. The number of parent fields (Root) and child fields (Shaft) are denoted by s and k, respectively. It is also possible to have a different number of rows for each parent row, as illustrated by x, y, and z in Fig. 11.

Using this nested solution, it is understood that the foreign key is removed and that the constraint specification is simplified. Despite listing the various operations for the HODM, this paper only focuses on the implementation of manipulation operations, including `Grow`, `Cut`, `Fall`, `Implant` and `Wash`.

3.1. Case study

We use the El Nino data set that is saved in the UCI Repository. The data set contains oceanographic and surface meteorological readings that are taken from a series of buoys positioned throughout the equatorial Pacific. The data are expected to aid our understanding and prediction of El Nino/Southern Oscillation (ENSO) cycles (UCI, 1999). The data set is categorized as spatio-temporal data. The El Nino data relate to climate change in terms of relevant positions. The data consist of variables such as the buoy number (Buoy), observation number (Obs), year (Y), month (M), day (D), date of collection (Date_), latitude, longitude, zonal winds (ZonWinds) (west<0, east>0), meridional winds (MerWinds) (south<0, north>0), relative humidity (Humidity), air temperature (AirTemp) and sea surface temperature (SSTemp). The data types of these attributes are shown in Table 3 (Madraky, Othman, & Hamdan, 2015).

The data were collected from buoys that are located in 62 locations. Each location contains climate observations from the time period of 1980 to 1998. The number of observations in terms of years and locations is illustrated in Tables 4 and 5 (Madraky et al., 2015), respectively. As displayed in Table 4, the collection of observations during the early years was lower, but it has increased in more recent years. The maximum number of observations is 22,238 in 1997, because the observations were only collected in the first half of the year during 1998.

We performed comparisons between the tabular and HODM data structures using selected sub-data values for 15 days from two particular locations (Buoy1 and Buoy2), sorted by location number. The data values are shown in a tabular structure in Table 6, which is suitable for data mining. The location number is saved in the Buoy column, and the location coordinates, which include Latitude and Longitude, are repeated in all of the rows. Redundant values are specified in gray in Table 6.

Using Definitions 1, 2 and 3, the structure and values based on the HODM are illustrated in Fig. 12, and the redundant location specifications that include `Name` and `Position` are removed. This causes the database size, response time and data transfer costs to decrease. As mentioned in previous research (Madraky et al., 2012), one advantage of the HODM is its ability to reduce redundancy by removing the repeated spatial data, which results in increased database integrity and consistency. The relationship between the data and definitions of a hair is shown in Fig. 12. The data, or shaft, has a relational structure and is represented in tabular form, and the definitions, or root, have object-oriented properties. The data are shown in relational form in Table 2. The number of locations saved in the Buoy column and the coordinates in terms of Latitude and Longitude are repeated in all of the rows. The HODM's structure is illustrated in Fig. 12, where location specification saves time in the `Name` and `Position` attributes. This decreases the database size, response time and data transfer costs in large databases.

The specifications of the attributes are completely explained in Section 2, although certain notes must be considered in regard to the values in the example. As stated, for the identification of a location, we can use a name in addition to the coordinates. In the example, the values are related to Buoy 1, as shown in Fig. 6; therefore, the name values are assigned to Buoy 1. Latitude and longitude are considered to be determinants of location, and they do not have any redundancy. `Strength`, `Direction` and `Proximity` are not defined in the real data, but these attributes are produced in the applications, thus they are defined and initialized in the HODM. For example, the importance of information from Buoy 1 is "Necessary," because there exists no additional orientation with respect to another point, which leads to the repetition of its coordinates in the Direction attribute. As a default value, "Null" is assigned to the `Proximity` and `Type` attributes. The second position is defined to be the same as the first point. The data values are stored entirely within one table, although some of the attributes are defined as a nested table. We explain the SQL statements for this definition in section 3.2. According to the

Table 3
The data types of attributes in the case study.

Row	Name	Type	Row	Name	Type
1	Buoy	NUMBER(38,0)	8	Longitude	FLOAT
2	Obs	NUMBER(38,0)	9	ZonWinds	FLOAT
3	Y	NUMBER	10	MerWinds	FLOAT
4	M	NUMBER	11	Humidity	NVARCHAR2(255 CHAR)
5	D	NUMBER	12	AirTemp	FLOAT
6	Date_	CHAR(6)	13	SSTemp	NVARCHAR2(255 CHAR)
7	Latitude	FLOAT			

Table 4
The number of observations in terms of years.

Year	Number of observations	Year	Number of observations	Year	Number of observations	Year	Number of observations
80	166	85	1684	90	8437	95	21,947
81	545	86	3780	91	8800	96	21,825
82	505	87	4688	92	16,011	97	22,238
83	406	88	6136	93	20,609	98	10,076
84	947	89	7929	94	21,351	Sum	178,080

Table 5
The number of observations in terms of location.

Buoy No.	Number of observations	Buoy No.	Number of observations	Buoy No.	Number of observations	Buoy No.	Number of observations
1	3614	17	3962	33	2216	49	988
2	3308	18	2789	34	2011	50	2303
3	3656	19	1267	35	3544	51	2830
4	4820	20	1506	36	2270	52	1921
5	3922	21	3787	37	1975	53	2309
6	2791	22	2092	38	2133	54	3082
7	2161	23	1975	39	1770	55	3656
8	2103	24	1499	40	2531	56	7586
9	4483	25	1402	41	4389	57	4159
10	2419	26	3664	42	2520	58	3512
11	1953	27	2042	43	2108	59	2301
12	2090	28	4326	44	2520	60	5795
13	2431	29	1892	45	2231	61	3916
14	4147	30	1714	46	3140	62	2871
15	4212	31	1665	47	2656	Sum	178,080
16	4249	32	2127	48	2769		

HODM specifications, the data values for both buoys are illustrated in Fig. 13.

3.1.1. SQL Statements for hair data type and operations

In the first step, we created a database named the HODM and schema named HM using SQL*PLUS 11.2.0.0 in Oracle 11 g. The object-relational features of the Oracle DBMS are used in the implementation phase. These features are, namely, the possibility of declaring object types with associated operations, the aggregation structure of nested tables, and the implementation of object referencing by using an attribute of the object reference data type. The SQL statements for declaring data types, tables and procedures based on the dataset are shown in Table 7. One of the common objects among the tables is PointType. PointType is used to determine the `Position`, `Direction` and `Proximity` variables. PointType is also used by some functions to define lines or surfaces, where X and Y are the coordinates of a point. The Z variable covers the 3-dimensional position.

Using the `Grow` operation, a row is inserted into this table, and the `Cut` operation is defined as a procedure for deleting a row of cells. The `Wash` operation is also used for data cleaning. The `Cell` attribute stores temporal values. We used the "Nested Table" ability of Oracle to create relationships between cells and spatial values (Root). The HM.PointSetType and HM.CellsSetType collections are used to define nested tables in the hair table. `Proximity` is also defined as a nested table that reserves some neighbored positions. The primary data set is stored in the text file, but we converted the file into an Access DB form so that it could be imported into Oracle. The SQL statements for data additions include the `Implant` and `Grow` operations, as defined in Table 6. The hair definitions are initialized by the `Implant` operation, and the `Grow` operation is used for data insertion. The `Implant` operation initializes certain attributes of hair, and other attributes, such as the `Type`, `Direction` and `Proximity`, are assigned by other operations. The `Grow` operation is defined according to the temporal data structure. Default values are allocated to other attributes in the first stages of defining a hair. The `Fall` and `Cut` operations are used for data removal. `Fall` is utilized to erase a hair (the complete data for a position), and `Cut` deletes a row from the `Cell` attribute. Finally, the `Wash` operation is used to update the data in a hair by changing some of its values.

4. Performances of the HODM

The normalized structure is not suitable for data mining operations, because it is essential to consider all of the relationships between attributes that play a role in data classification or clustering (Mousavi, Bakar, & Vakilian, 2015). Therefore, data redundancy is accepted for the detection of similarities and dissimilarities. Data redundancy causes a decrease in time and file size performances. The HODM has the benefits of an implicit normalized structure without having table decomposition, because it uses nested tables. The physical data size is decreased by the HODM due to its reduction in replicated data. The spatio-temporal data are divided into

Table 6
El Nino sub-data in the tabular structure.

Buoy	Obs	Y	M	D	Date_	Latitude	Longitude	ZonWinds	MerWinds	Humidity	AirTemp	SSTemp
1	177,915	98	1	1	980,101	8.95	−140.33	−7	−6.7	86	26.19	26.54
1	177,916	98	1	2	980,102	8.95	−140.33	−6	−6.8	86	26	26.48
1	177,917	98	1	3	980,103	8.95	−140.33	−6.7	−4.9	87.3	26.04	26.27
1	177,918	98	1	4	980,104	8.95	−140.33	−6.7	−4.9	87.7	25.96	26.21
1	177,919	98	1	5	980,105	8.95	−140.33	−7.3	−4.8	88.1	25.96	26.13
1	177,920	98	1	6	980,106	8.95	−140.33	−8	−2.6	87.3	26.08	26.06
1	177,921	98	1	7	980,107	8.95	−140.33	−5.7	−2	88.1	26	26.12
1	177,922	98	1	8	980,108	8.95	−140.33	−6.1	−3.2	86.9	25.88	26.1
1	177,923	98	1	9	980,109	8.95	−140.33	−6	−4	86.9	25.8	26.05
1	177,924	98	1	10	980,110	8.95	−140.33	−6.8	−5.3	82.6	25.73	26.01
1	177,925	98	1	11	980,111	8.95	−140.33	−7.4	−5.2	82.6	25.69	25.94
1	177,926	98	1	12	980,112	8.95	−140.33	−6.9	−6.5	84.7	25.8	25.91
1	177,927	98	1	13	980,113	8.95	−140.33	−6.5	−6.1	83.5	25.73	25.87
1	177,928	98	1	14	980,114	8.95	−140.33	−7	−6.7	83.9	25.65	25.8
1	177,929	98	1	15	980,115	8.95	−140.33	−8.1	−6.3	83	25.76	25.74
2	103,321	98	1	1	980,101	4.91	−139.84	−4.1	−7.4	90.1	27.46	28.67
2	103,322	98	1	2	980,102	4.91	−139.84	−3.2	−3.8	94.1	25.65	28.42
2	103,323	98	1	3	980,103	4.91	−139.84	−4	−6.3	87.6	27.38	28.35
2	103,324	98	1	4	980,104	4.91	−139.84	−5.2	−4.3	88	27.72	28.41
2	103,325	98	1	5	980,105	4.91	−139.84	−3.9	−3.5	89.3	27.46	28.55
2	103,326	98	1	6	980,106	4.91	−139.84	−5.3	−2.5	88	27.22	28.61
2	103,327	98	1	7	980,107	4.91	−139.84	−4	−5.5	81.2	28.22	28.7
2	103,328	98	1	8	980,108	4.91	−139.84	−4	−5.1	80	28.19	28.72
2	103,329	98	1	9	980,109	4.91	−139.84	−3.6	−4	75.9	27.95	28.72
2	103,330	98	1	10	980,110	4.91	−139.84	−3.1	−3.5	88.9	26.73	28.66
2	103,331	98	1	11	980,111	4.91	−139.84	−4.4	−6.8	78.7	28.07	28.73
2	103,332	98	1	12	980,112	4.91	−139.84	−2	−8	79.6	28.07	28.77
2	103,333	98	1	13	980,113	4.91	−139.84	−2.8	−8.8	83.2	27.84	28.71
2	103,334	98	1	14	980,114	4.91	−139.84	−5.3	−8.8	86	27.38	28.63
2	103,335	98	1	15	980,115	4.91	−139.84	−7	−4.4	90.9	26.76	28.58

two parts, which are related to the spatial and temporal specifications. Spatial attributes, such as the coordinates, name and importance, are changed based on each position, but temporal attributes are produced and stored based on the time values. For example, climate change attributes, such as temperature, pressure and humidity, vary over time at each position. Spatial and temporal attributes are separated in the normalized tables of relational databases. For example, the normalized data structure in the abnormal table that is illustrated in Fig. 14 is converted into two tables to store the spatial and temporal values.

In the HODM, using the nested table feature, we can define a column of a record as a table without using key redundancy, even though the foreign key replication is mandatory in the normal structure. The structural diagram of climate change information based on HODM is shown in Fig. 15.

Reducing the data file size in the HODM depends on the size of spatial attributes, with the performance being increased whenever the spatial data size is larger. Data file size reduction is calculated using the following formula:

$$\begin{aligned} \textit{Data file reduction} &= \textit{Size of Tabular table-Size of HODM table} \\ &= n \times (S_s + S_t) - (n \times S_t + P \times S_s) \\ &= S_s \times (n - P) \end{aligned} \quad (2)$$

Where:

n: Number of observations
S_s: Size of spatial data
S_t: Size of temporal data
P: Number of positions

Data file reduction based on the data values in Table 5 and Fig. 7 is defined as:

n = 30, P = 2
S_s = Size of [Buoy(int-2), Latitude (float-4) and Longitude (float-4)] = 10 Byte
S_t = Size of [Obs(int-2), Y(int-2), M(int-2), D(int-2), Date_(char-6), [Zon.Winds] (float-4), [Mer.Winds] (float-4), Humidity (float-4), [Air Temp.] (float-4), [S.S.Temp.] (float-4)] = 34 Byte

$$\textbf{Data file reduction (for 2 locations)} = 10 \times (30 - 2) = 280\ \textbf{Byte}$$
$$\textbf{Data file performance} = \frac{\textbf{Data file reduction}}{\textbf{Tabular file size}} \times 100 = \frac{280}{1320} \cong 21\%$$

As shown in the above calculations, the data file size performance in this example is approximately 21%. This performance increases with the number of observations (temporal rows). In large databases, if it is assumed that the size of the spatial fields is equal to that of the temporal fields, the file reduction performance is closer to 50%. The size reduction performance could be more than 50%, if the size of the temporal fields is more than the size of the spatial fields, as illustrated in Table 8. In this table, the second and third columns are the proportions of the spatial and temporal data sizes, respectively. The forth column is the average number of observations in each location. In the HODM structure, this value is the average number of `Cells` in the row. The last column is calculated using formula (1) to obtain a reduced file size. For example, if the spatial and temporal data sizes are proportionally equal, the file size reduction will be 45% for 10 observations at each location (Row 5), 49.5% for 100 observations at each location (Row 14) and 49.95% for 1000 observations at each location (Row 23). It is necessary to explain that the attribute size is different from the data size, because some attributes may not have any value or may not initialize by Null, thus the file size is calculated using the sum of the stored data size.

According to the query execution time, indexing is one of the most effective issues with respect to data access. The index is a data structure that improves the speed of data retrieval operations in a database table. Typically, the index is defined based on certain attributes that are used for data searching. Using the index significantly improves the data access time. For index optimization, there are certain parameters, such as the number, size and level of index,

Table 7
A summary of SQL statements.

Row	Definition category	Name	Main SQL statements
1	Data structure (Type – Object)	POINTTYPE	CREATE OR REPLACE TYPE HM.POINTTYPE AS OBJECT (X Number, Y Number);
2	Data structure (Type – Object)	CELLSTYPE	CREATE OR REPLACE TYPE HM.CELLSTYPE AS OBJECT (... {Attribute definitions}c ... MEMBER PROCEDURE Grow, MEMBER PROCEDURE Cut, MEMBER PROCEDURE Wash); ...
3	Data structure (Type – Table)	PointSetType	CREATE TYPE HM.PointSetType AS TABLE OF POINTTYPE;
4	Data structure (Type – Table)	CellsSetType	CREATE TYPE HM.CellsSetType AS TABLE OF CellsType;
5	Data structure table	HAIR	CREATE TABLE HM.Hair (... {Attribute definitions} ... Position HM.POINTTYPE, Direction HM.POINTTYPE, Proximity HM.POINTSETTYPE, Cells HM.CELLSSETTYPE) NESTED TABLE Proximity STORE AS PointsTable, NESTED TABLE Cells STORE AS CellsTable; ...
6	Manipulation operation	IMPLANT	CREATE OR REPLACE PROCEDURE HM.IMPLANT (... {Parameter definitions} ... INSERT INTO HM.HAIR ((Attribute Names}) VALUES ({Attribute Values}); ...
7	Manipulation operation	GROW	CREATE OR REPLACE PROCEDURE HM.GROW (... {Parameter definitions} ... INSERT INTO HM.CELLSTABLE ((Attribute Names}) VALUES ({Attribute Values}); ...
8	Manipulation operation	FALL	CREATE PROCEDURE HM.FALL (... {Parameter definitions} ... DELETE FROM HM.HAIR ...
9	Manipulation operation	CUT	CREATE OR REPLACE PROCEDURE HM.CUT (... {Parameter definitions} ... DELETE FROM HM.CELLSTABLE ...
10	Manipulation operation	WASH	CREATE PROCEDURE HM.WASH (... {Parameter definitions} ... UPDATE HM.HAIR ... CREATE PROCEDURE HM.WASH (... {Parameter definitions} ... UPDATE HM.CELLSTABLE ...

which should be used. Index values, or Entries, are sorted to reach a better access time when searching. Sometimes, indexes just use key values, and other values are searched using a linear search. We explain both linear and binary search time orders in the tabular and HODM structures.

Binary Search (With Index):

The binary search algorithm compares the search key value with the Entry of the middle record in the index. If the keys match, then a matching element has been found, and its index is returned. Otherwise, if the search key is less than the Entry value, then the algorithm repeats its action on the sub-index to the top of the middle Entry, or if the search key is greater, the action is performed on the sub-index to the bottom of the middle Entry. If the remaining index to be searched is empty, then the key cannot be found in the array, and a special "not found" indication is returned. By considering the tabular and HODM structures, which are shown in Fig. 16, the time order of the binary search algorithm is calculated for the worst case.

Table 8
A file size reduction based on spatial and temporal ratios and the number of observations.

Row	Spatial data size ratio (in percent)	Temporal data size ratio (in percent)	Number of observations in each location (Average)	File size reduction (in percent)	Row	Spatial data size ratio (in percent)	Temporal data size ratio (in percent)	Number of observations in each location (Average)	File size reduction (in percent)
1	10	90	10	9.000	19	10	90	1000	9.990
2	20	80	10	18.000	20	20	80	1000	19.980
3	30	70	10	27.000	21	30	70	1000	29.970
4	40	60	10	36.000	22	40	60	1000	39.960
5	50	50	10	45.000	23	50	50	1000	49.950
6	60	40	10	54.000	24	60	40	1000	59.940
7	70	30	10	63.000	25	70	30	1000	69.930
8	80	20	10	72.000	26	80	20	1000	79.920
9	90	10	10	81.000	27	90	10	1000	89.910
10	10	90	100	9.900	28	10	90	10,000	9.999
11	20	80	100	19.800	29	20	80	10,000	19.998
12	30	70	100	29.700	30	30	70	10,000	29.997
13	40	60	100	39.600	31	40	60	10,000	39.996
14	50	50	100	49.500	32	50	50	10,000	49.995
15	60	40	100	59.400	33	60	40	10,000	59.994
16	70	30	100	69.300	34	70	30	10,000	69.993
17	80	20	100	79.200	35	80	20	10,000	79.992
18	90	10	100	89.100	36	90	10	10,000	89.991

The number of binary search steps using the tabular structure is defined as:

$$\boldsymbol{O}(\boldsymbol{Binary\ Search_{Tabular}}) = 2\log_2 \mathbf{n} \tag{3}$$

Where:

n: Number of observations (records)

The number of binary search steps for the HODM structure is defined as:

$$\begin{aligned}\boldsymbol{O}(\boldsymbol{Binary\ Search_{HODM}}) &= 2\log_2 \boldsymbol{p} + 2\log_2 \frac{\boldsymbol{n}}{\boldsymbol{p}} \\ &= 2(\log_2 \boldsymbol{p} + \log_2 \boldsymbol{n} - \log_2 \boldsymbol{p}) \\ &= 2\log_2 \boldsymbol{n}\end{aligned} \tag{4}$$

Where:

n: Number of observations
p: Number of locations (records)

As mentioned above, the number of steps when using the index in either the tabular or HODM structures is equal.

Note: In the tabular structure, each observation is considered to be a record, but in the HODM, each location is defined as a record.

Example in the climate change data set:

n (Number of observations) = 178,080

The worst number of steps in the binary search based on the index (Tabular and HODM) $= 2\log_2 178080 = 2 \times 18 = 36$

Linear Search (Without Index):

The linear search algorithm is used when the records are not sorted or when an index is lacking. The algorithm consists of checking each record, one at a time and in sequence, until the desired record is found. By considering the tabular and HODM structures, which are shown in Fig. 15, in the worst case, the number of linear search steps for the tabular structure is defined as:

$$\boldsymbol{O}(\boldsymbol{Linear\ Search_{Tabular}}) = \boldsymbol{n} \tag{5}$$

Where:

n: Number of observations (records in tabular data structure)

The number of linear search steps for the HODM structure is defined as:

$$\boldsymbol{O}(\boldsymbol{Linear\ Search_{HODM}}) = \boldsymbol{p} + \frac{\boldsymbol{n}}{\boldsymbol{p}} = \frac{\boldsymbol{p}^2 + \boldsymbol{n}}{\boldsymbol{p}} \tag{6}$$

Where:

n: Number of observations
p: Number of locations (records in HODM data structure)

By comparing the steps in the tabular and HODM structures, we find that the number of steps in HODM is less than the number in the tabular model, because the number of locations (p) is less than the number of observations (n).

Example for the climate change data set:

n (Number of observations) = 178,080
p (Number of locations) = 62

The worst number of steps in the linear search (Tabular) = 178,080

The worst number of steps in the linear search (HODM) $= \frac{62^2+178080}{62} = \frac{181924}{62} \cong 2934$

4.1. Experimental results on file size reduction

In this sub-section, we validate the proposed data model by implementing the model in a real case study. We created the HODM data structure in Oracle, according to its spatial and temporal attributes. One of Oracle's features, which is referred to as a 'Nested table', is used to define the relationships between the spatial and temporal parts without using a foreign key. We defined the data structures in tabular and HODM databases based on the specification that was mentioned in Section 3. After importing data to the tabular database, the spatial specifications were extracted and imported to the HODM database. In the next step, the temporal values were separated and imported to the temporal part for each spatial position. With the goal of comparing the file sizes of the tabular and HODM databases, we obtained the file sizes using SQL statements. For better comparison, the file size values are calculated based on location which is illustrated in Table 9.

In Table 9, the number of locations in the data set is 62. The counts of temporal observations for each position are illustrated in the second column. The third and fourth columns refer to the file sizes in bytes for the tabular and HODM databases, and the file size reduction per location is calculated in the next column. The cumulative values are calculated in the other columns. As can be seen, the cumulative file size reduction in the last row (position 62) is 29.326%. It means that the reduction percent of file size by

Table 9
The file size reduction for 178,080 records (El Nino) by position.

Position	Count	Tabular file size (Byte)	HODM file size (Byte)	File size reduction (per position)	Tabular cumulative total (Byte)	HODM cumulative total (Byte)	File size reduction (Cumulative)
1	3614	119,301	83,170	30.286%	119,301	83,170	30.286%
2	3308	108,554	78,790	27.419%	227,855	161,960	28.920%
3	3656	115,368	82,472	28.514%	343,223	244,432	28.783%
4	4820	147,744	114,010	22.833%	490,967	358,442	26.993%
5	3922	129,164	93,874	27.322%	620,131	452,316	27.061%
6	2791	100,204	69,513	30.629%	720,335	521,829	27.557%
7	2161	74,397	52,796	29.035%	794,732	574,625	27.696%
8	2103	72,141	51,120	29.139%	866,873	625,745	27.816%
9	4483	137,203	92,382	32.668%	1,004,076	718,127	28.479%
10	2419	86,183	59,585	30.862%	1,090,259	777,712	28.667%
11	1953	69,852	48,380	30.739%	1,160,111	826,092	28.792%
12	2090	68,239	47,349	30.613%	1,228,350	873,441	28.893%
13	2431	77,612	53,312	31.310%	1,305,962	926,753	29.037%
14	4147	131,808	90,348	31.455%	1,437,770	1,017,101	29.258%
15	4212	136,313	94,203	30.892%	1,574,083	1,111,304	29.400%
16	4249	145,290	98,562	32.162%	1,719,373	1,209,866	29.633%
17	3962	132,877	89,306	32.790%	1,852,250	1,299,172	29.860%
18	2789	94,953	64,285	32.298%	1,947,203	1,363,457	29.979%
19	1267	39,997	28,603	28.487%	1,987,200	1,392,060	29.949%
20	1506	48,290	34,745	28.049%	2,035,490	1,426,805	29.904%
21	3787	119,620	89,332	25.320%	2,155,110	1,516,137	29.649%
22	2092	67,342	48,523	27.945%	2,222,452	1,564,660	29.598%
23	1975	66,035	46,295	29.893%	2,288,487	1,610,955	29.606%
24	1499	49,726	36,244	27.113%	2,338,213	1,647,199	29.553%
25	1402	48,279	34,269	29.019%	2,386,492	1,681,468	29.542%
26	3664	120,778	84,148	30.328%	2,507,270	1,765,616	29.580%
27	2042	63,413	43,003	32.186%	2,570,683	1,808,619	29.644%
28	4326	138,897	99,972	28.024%	2,709,580	1,908,591	29.561%
29	1892	68,523	47,722	30.356%	2,778,103	1,956,313	29.581%
30	1714	57,185	40,055	29.955%	2,835,288	1,996,368	29.589%
31	1665	56,927	38,623	32.153%	2,892,215	2,034,991	29.639%
32	2127	73,370	54,236	26.079%	2,965,585	2,089,227	29.551%
33	2216	78,019	55,869	28.391%	3,043,604	2,145,096	29.521%
34	2011	69,488	49,388	28.926%	3,113,092	2,194,484	29.508%
35	3544	119,500	84,070	29.649%	3,232,592	2,278,554	29.513%
36	2270	78,617	53,658	31.748%	3,311,209	2,332,212	29.566%
37	1975	65,729	47,963	27.029%	3,376,938	2,380,175	29.517%
38	2133	73,198	49,746	32.039%	3,450,136	2,429,921	29.570%
39	1770	59,392	41,702	29.785%	3,509,528	2,471,623	29.574%
40	2531	84,484	61,714	26.952%	3,594,012	2,533,337	29.512%
41	4389	141,005	97,125	31.119%	3,735,017	2,630,462	29.573%
42	2520	86,333	63,662	26.260%	3,821,350	2,694,124	29.498%
43	2108	75,439	52,262	30.723%	3,896,789	2,746,386	29.522%
44	2520	88,261	63,071	28.540%	3,985,050	2,809,457	29.500%
45	2231	79,823	55,293	30.730%	4,064,873	2,864,750	29.524%
46	3140	91,113	62,862	31.007%	4,155,986	2,927,612	29.557%
47	2656	78,960	55,065	30.262%	4,234,946	2,982,677	29.570%
48	2769	82,388	60,244	26.878%	4,317,334	3,042,921	29.519%
49	988	33,211	24,328	26.747%	4,350,545	3,067,249	29.497%
50	2303	71,593	53,177	25.723%	4,422,138	3,120,426	29.436%
51	2830	89,121	63,660	28.569%	4,511,259	3,184,086	29.419%
52	1921	67,448	48,248	28.466%	4,578,707	3,232,334	29.405%
53	2309	74,237	51,157	31.090%	4,652,944	3,283,491	29.432%
54	3082	100,906	73,177	27.480%	4,753,850	3,356,668	29.391%
55	3656	114,872	85,632	25.454%	4,868,722	3,442,300	29.298%
56	7586	219,083	150,818	31.159%	5,087,805	3,593,118	29.378%
57	4159	134,026	92,446	31.024%	5,221,831	3,685,564	29.420%
58	3512	113,216	78,106	31.012%	5,335,047	3,763,670	29.454%
59	2301	69,615	48,915	29.735%	5,404,662	3,812,585	29.457%
60	5795	173,883	121,737	29.989%	5,578,545	3,934,322	29.474%
61	3916	119,270	87,950	26.260%	5,697,815	4,022,272	29.407%
62	2871	84,124	64,034	23.881%	5,781,939	4,086,306	29.326%

HODM in comparison to tabular model is about 30%. Based on the results in Table 9, a size comparison chart is shown in Fig. 17 and Fig. 18 displays a chart of the size comparisons based on the locations.

4.2. Experimental results on query execution time

Several query types are requested in the spatio-temporal databases. These queries can be divided into four categories, including 1) Basic, 2) Spatial, 3) Temporal, and 4) Spatio-temporal (Deshpande, Ives, & Raman, 2007). We considered certain queries that were selected from a benchmark (Werstein, 1998); these ex-

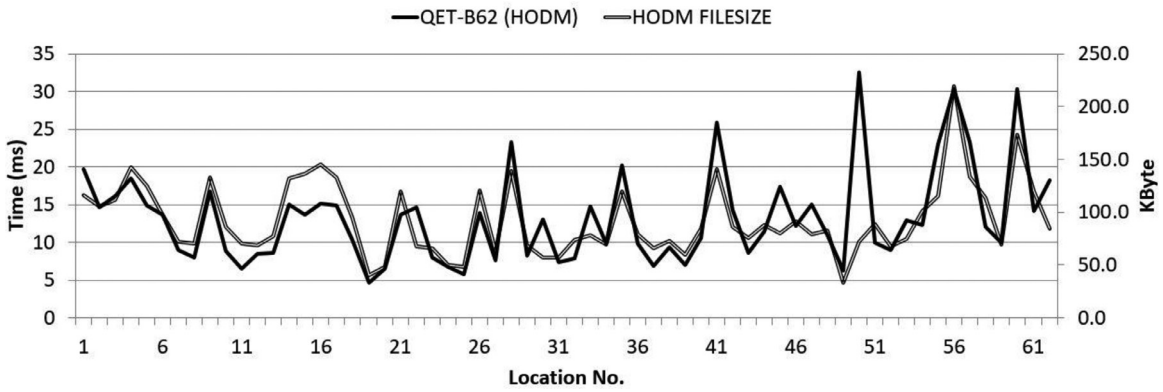

Fig. 22. A comparison of the file sizes and query execution times.

```
Q2: Find the nearest buoys to the buoy=5
-----------------------------------------------------------
/* Tabular Query */
SELECT buoy,
 to_char((ACOS(SIN(-2)*SIN(LATITUDE) +COS(-2) *COS(LATITUDE) *COS(LONGITUDE+140)) * 6371),'999999.99')
AS dist
FROM Taoall2
WHERE buoy<>5
ORDER BY dist;

/* Normal Query */
SELECT buoy,
 to_char((ACOS( SIN(-2)*SIN(LATITUDE) +COS(-2) *COS(LATITUDE) *COS(LONGITUDE+140)) * 6371),'999999.99')
AS dist
FROM SPATIAL
WHERE buoy<>5
ORDER BY dist;

/* HODM Query */

SELECT  h.NAME,
to_char((ACOS( SIN(-2)*SIN(h.position.x) +COS(-2) *COS(h.position.x) *COS(h.position.y +140)) * 6371),'999999.99')
AS dist
FROM HM.HAIR h
WHERE h.NAME<>5
ORDER BY dist;
```

Fig. 23. Spatial query.

```
Q3: Find the locations and dates where humidity>95% at least for One Week continuously
-------------------------------------------------------------------------------------------
/* Tabular Query */
SELECT t1."Buoy",TO_DATE(t1."Date_",'YYMMDD') Bdate, TO_DATE(t1."Date_",'YYMMDD')+7 Edate
FROM "Taoall2" t1
WHERE t1."humidity">95 and t1."humidity"<=all
(select t2."humidity"
 from "Taoall2" t2
 where TO_DATE(t2."Date_",'YYMMDD') BETWEEN
  TO_DATE(t1."Date_",'YYMMDD') and TO_DATE(t1."Date_",'YYMMDD')+7 and t1."Buoy"=t2."Buoy")
order by t1."Date_";

/* Normal Query */
SELECT t1.Buoy, TO_DATE(t1.date_,'YYMMDD') Bdate, TO_DATE(t1.date_,'YYMMDD')+7 Edate
FROM TEMPORAL t1
WHERE t1.HUMIDITY>95 and t1.HUMIDITY<=all
(SELECT t2.HUMIDITY
 FROM Temporal t2
 WHERE TO_DATE(t2.date_,'YYMMDD') BETWEEN
    TO_DATE(t1.date_,'YYMMDD') and TO_DATE(t1.date_,'YYMMDD')+7 and t1.buoy=t2.buoy)
ORDER BY t1.date_;

/* HODM Query */
SELECT h1.Name, TO_DATE(c1."DATE",'YYMMDD') Bdate,TO_DATE(c1."DATE",'YYMMDD')+7 Edate
from HM.HAIR h1,Table(CELLS) c1
WHERE c1.HUMIDITY>95 and c1.HUMIDITY<=all
(select c2.HUMIDITY
 from HM.HAIR h2,Table(CELLS) c2
 where TO_DATE(c2."DATE",'YYMMDD') BETWEEN
   TO_DATE(c1."DATE",'YYMMDD') and TO_DATE(c1."DATE",'YYMMDD')+7 and h1.Name=h2.Name)
order by c1."DATE";
```

Fig. 24. A temporal query.

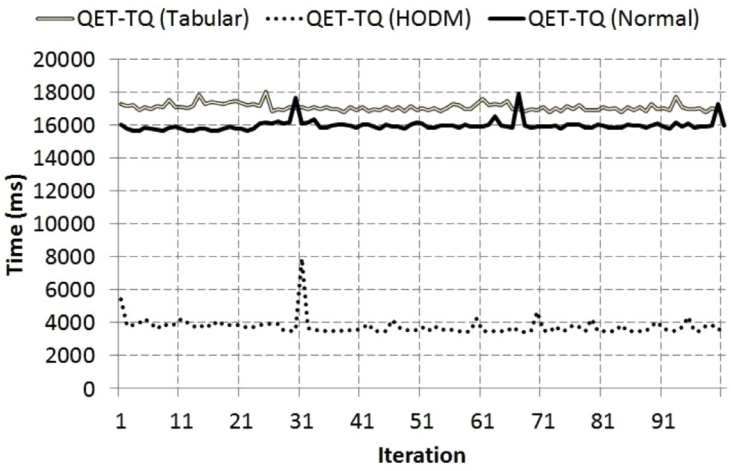

Fig. 25. The average query execution times for the temporal query.

```
Q4: Find all locations where temperature less than 20 degrees Celsius from 20 March to 21
June of each year
-----------------------------------------------------------------------------------------------
/* Tabular Query */
select "Buoy", "latitude", "longitude",
  Count(*) Days, to_char(Avg("airtemp"),'99.99') MeanTemp
from "Taoall2"
where "airtemp"<25 and
  to_char(TO_DATE("Date_",'YYMMDD'),'MM-DD') between '03-20' and '06-21'
group by "Buoy", "latitude", "longitude"
order by "Buoy";

/* Normal Query */
select T."BUOY",S."LATITUDE",S."LONGITUDE",
  Count(*) Days, to_char(Avg(T."AIRTEMP"),'99.99') MeanTemp
from "TEMPORAL" T, "SPATIAL" S
where T."AIRTEMP"<25 and
       to_char(TO_DATE(T."DATE_",'YYMMDD'),
  'MM-DD')
Between '03-20' and '06-21' and
  T."BUOY"=S."BUOY"
group by T."BUOY", S."LATITUDE",S."LONGITUDE"
order by T."BUOY";

/* HODM Query */
select  h.Name, h.position.x, h.position.y, Count(*) Days, to_char(Avg(c.airtemp),'99.99')
MeanTemp
from HM.HAIR h, Table(CELLS) c
where  c.airtemp<25 and  to_char(TO_DATE(c."DATE" ,'YYMMDD'),'MM-DD') between
     '03-20' and '06-21'
group by h.Name, h.position.x , h.position.y
order by h.Name;
```

Fig. 26. A spatio-temporal query.

ample queries cover all of the categories but have been adopted to the climate change case study. The first category includes the basic and relevant queries, which are popular in all of the databases. For this type of query, certain conditions are applied to the stored data to obtain the desired information. It is possible to perform simple calculations, including sum, average, maximum, minimum, etc. For example, the list of climate change points at a location and the minimum humidity in a zone are two queries in the basic category. The second category is related to spatial data. The spatial data are coordinates, neighborhoods, borders and specifications that refer to geography (Yeung & Hall, 2007). The nearest neighborhood subjects are in this category. For example, we can include finding the climate stations that are nearest to the central station as a spatial query. The third category relates to temporal queries that use data related to time. This type of query could be related to a time interval. For example, we can run this query in a temporal database: find the periods by using the date when the humidity in zone 1 is between 90 and 95. The last category is concerned with spatio-temporal queries. These queries use spatial and temporal parameters simultaneously. For example, we can run this query: find all of the locations where the temperature is less than 20 degrees Celsius from 20 March to 21 June of each year (Menon, Jayaraman, & Govindaraju, 2012). A spatio-temporal data model should be able to run queries according to all of the mentioned categories. We created three data models, including Tabular, Normal and HODM, in Oracle (based on Fig. 8) and imported data

Table 10
The query execution time for particular locations (Buoy = 5).

Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)
1	40	60	80	26	50	40	0	51	50	70	0	76	50	70	20
2	50	60	10	27	30	60	10	52	60	60	10	77	40	60	10
3	60	70	20	28	60	60	20	53	70	50	10	78	70	60	10
4	60	60	20	29	60	60	20	54	50	60	30	79	40	80	30
5	30	60	20	30	70	60	20	55	50	50	20	80	60	50	0
6	60	40	20	31	50	60	20	56	70	70	10	81	40	60	0
7	50	60	10	32	90	60	10	57	60	60	10	82	40	60	10
8	50	50	20	33	70	80	10	58	50	60	10	83	60	60	20
9	60	60	20	34	80	70	0	59	50	60	20	84	50	40	20
10	60	70	20	35	50	60	0	60	50	70	0	85	50	50	0
11	50	60	10	36	50	40	20	61	60	50	0	86	60	60	10
12	60	50	10	37	60	60	20	62	70	60	0	87	100	60	20
13	60	80	10	38	80	60	20	63	90	70	10	88	50	50	20
14	50	50	20	39	60	40	20	64	50	70	20	89	40	70	0
15	40	60	20	40	150	40	20	65	40	70	10	90	50	40	10
16	50	90	10	41	60	40	10	66	90	60	20	91	50	50	30
17	60	60	10	42	80	60	20	67	40	60	20	92	120	60	0
18	100	60	20	43	60	60	30	68	50	70	0	93	140	60	10
19	80	70	20	44	40	60	20	69	50	60	20	94	110	60	20
20	80	50	20	45	70	60	0	70	50	60	20	95	80	60	20
21	80	60	20	46	40	60	10	71	60	70	10	96	50	50	20
22	50	60	50	47	60	70	10	72	60	60	20	97	70	60	10
23	50	70	20	48	30	60	20	73	70	50	0	98	70	60	30
24	70	70	20	49	90	70	10	74	70	60	20	99	40	60	10
25	60	60	0	50	40	60	10	75	60	50	0	100	50	40	20
												Avg.	60.9	59.5	14.9

Table 11
The average query execution times for the 62 locations.

Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)
1	55.6	60.9	19.7	17	56.9	56.4	14.9	33	59.1	43.8	14.8	49	17.2	26.2	6.2
2	49.9	53.1	14.7	18	47.3	47.8	10.3	34	54.6	62	9.8	50	22.7	44.3	32.6
3	57.9	52	16.1	19	37.6	29.1	4.6	35	67.2	56.7	20.2	51	25.8	42.9	9.9
4	60.9	71.1	18.5	20	42.4	32.1	6.5	36	56.4	42.5	9.8	52	23.6	40.6	8.9
5	60.9	59.5	14.9	21	59.5	59.3	13.7	37	74.3	41.4	6.8	53	19.8	39.9	12.9
6	52.1	50.7	13.6	22	46.1	37	14.7	38	58.5	38.1	9.3	54	24.8	55.7	12.3
7	45.8	39.2	8.9	23	49.1	36.5	8	39	44.5	33.4	7	55	26.9	60.5	23.1
8	47.5	39.6	8	24	39.7	30.6	6.7	40	51.7	47.5	10.6	56	43.4	109.2	30.3
9	56.6	61	16.7	25	40.8	31.3	5.8	41	63.6	52	25.9	57	28.9	85.7	23.3
10	49.6	48.2	8.8	26	58.6	60.5	13.9	42	49.9	47	14.3	58	26.5	65.6	12
11	47	39.6	6.5	27	47	28.9	7.6	43	30.1	38.9	8.6	59	21.9	44	9.9
12	44.8	40.5	8.5	28	53.1	77.4	23.3	44	24	48.5	11.4	60	37.1	80.8	30.4
13	124.7	39.4	8.6	29	52.6	51	8.2	45	21.7	42.8	17.4	61	37.4	60.6	14.2
14	78.5	56.6	15	30	48.5	34.5	13.1	46	26.4	53.4	12.2	62	30.5	51.5	18.2
15	53.8	64.5	13.6	31	48.9	30.5	7.4	47	22.5	44.9	15	**Avg**	**46.09**	**49.55**	**13.26**
16	66.6	62.6	15.2	32	63.3	41.5	7.8	48	23.1	49	10.9				

Table 12
The results of the spatial query.

Row	Buoy	Distance	Row	Buoy	Distance	Row	Buoy	Distance
1	4	2.2	21	35	33.03	41	21	50.81
2	6	3.24	22	16	33.03	42	24	51.22
3	3	4.3	23	15	34.12	43	25	51.36
4	2	7.5	24	36	34.12	44	20	51.74
5	27	11.12	25	37	34.22	45	19	52.19
6	1	12.04	26	17	34.38	46	46	287.45
7	43	16.91	27	60	34.38	47	48	297.56
8	10	17.89	28	34	34.52	48	47	298.18
9	42	18.13	29	38	34.75	49	51	308.42
10	44	18.38	30	18	34.76	50	53	308.83
11	11	18.97	31	33	34.81	51	22	309.21
12	8	19.02	32	14	34.91	52	62	309.34
13	9	19.72	33	13	35.14	53	50	309.47
14	12	20.17	34	32	35.34	54	49	309.87
15	45	20.33	35	28	44.19	55	57	318.54
16	40	20.67	36	29	44.32	56	59	318.92
17	41	20.92	37	30	45.12	57	58	319.32
18	7	21.07	38	31	45.56	58	61	319.61
19	39	22.29	39	26	45.64	59	55	319.79
20	52	22.95	40	23	50.03	60	54	319.92
						61	56	319.99

Table 13
The results of the temporal query.

Row	Buoy	BDate	EDate
1	59	24-MAY-93	31-MAY-93
2	59	25-MAY-93	01-JUN-93
3	59	26-MAY-93	02-JUN-93
4	59	27-MAY-93	03-JUN-93
5	59	30-MAY-93	06-JUN-93
6	59	14-OCT-93	21-OCT-93
7	23	05-MAY-94	12-MAY-94
8	24	20-APR-96	27-APR-96
9	24	23-APR-96	30-APR-96
10	23	28-APR-96	05-MAY-96
11	23	21-DEC-96	28-DEC-96

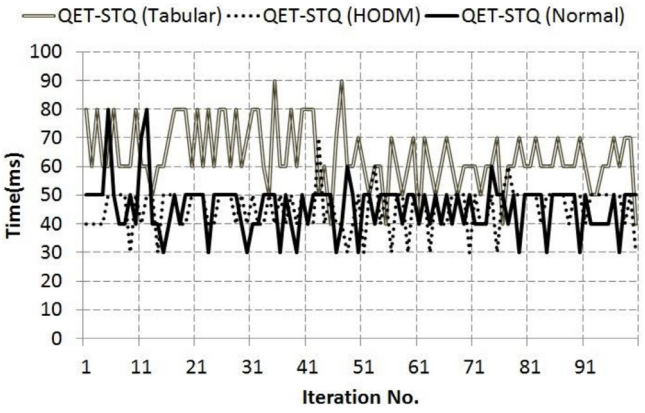

Fig. 27. The average query execution times for the spatio-temporal query.

values from the dataset. The query execution times were obtained and compared in regard to query types. To do this, the oracle SQL Developer version 4.0.1.14 was used, and the computations were performed on an Intel Pentium 4, 1.86 GHz computer, with 6 GB RAM and Windows 7 Enterprise Service pack 1.

4.2.1. Basic query

The first query type is simple and is called the basic query. We consider a simple query to be the identification of location specifications in the tabular, normal, and HODM databases, as illustrated in Fig. 19. The tabular database has a table that is named Taoall2. The Normal database consists of two tables that are named Spatial and Temporal. The HODM database has a table that identifies Hair and includes a nested table that is called Cells.

To reduce head position dependency, each query is run 100 times, and the query execution times are compared. For every location, these queries are performed, and the averages of the iterations are calculated. 'b' in Fig. 19 is a counter for determining locations that have values between 1 and 62. For example, the results of all of the iterations for Buoy = 5 are illustrated in Table 10.

The average query execution time for Buoy = 5 is inserted in the last row of Table 10. The time unit is in milliseconds. Values that are less than a millisecond are considered to be zero. This occurs when the physical and logical data sequences are similar, which is referred to as a strong data locality. The results show that in the Tabular and Normal databases, the execution time is approximately four times larger due to redundancy in the Tabular structure and data scattering in the Normal structure. The results in Table 10 are displayed in Fig. 20

The query execution times (QET) for Tabular, Normal and HODM are identified with blue, green and red curves, respectively. As displayed in Fig. 20, the values obtained with HODM are smaller than those from the other data models for all of the iterations, except the first. The averages of QET for all of the locations are shown in Table 11. The obtained results show that the QET in HODM were three times better than in the other structures. This result is also displayed in Fig. 21.

We also find that the QET (regardless the data model) is directly proportional to the data size. In Fig. 22, show a comparison of the QET and data size, based on the 62 locations. The left axis refers to the execution time in milliseconds, and the right axis determines the data size in bytes.

4.2.2. Spatial query

The second query category is concerned with spatial values, such as coordinates, distances and borders. We consider a spatial query example, taken as the calculation of location distances, as defined in Fig. 23. We use a mathematical formula to calculate the distance of each location from buoy =5 with latitude =−2 and longitude =−140. The query finds the nearest locations to this position.

The response to this mentioned query is the same in all of the three models. In Table 12, the locations are sorted by the distance from buoy =5, in ascending order.

Table 14
The query execution time for the temporal query.

Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)
1	17,280	16,010	5410	26	16,810	16,070	3930	51	17,050	16,070	3760	76	16,960	16,020	3860
2	17,180	15,810	3800	27	16,940	16,210	3960	52	16,890	15,820	3510	77	17,210	16,040	3760
3	17,220	15,680	3890	28	16,910	16,110	3530	53	17,020	15,870	3830	78	16,880	15,870	3500
4	16,930	15,630	3990	29	17,100	16,130	3480	54	16,820	15,960	3530	79	16,920	15,840	4210
5	17,120	15,820	4220	30	17,000	17,670	3560	55	17,050	15,960	3590	80	16,910	16,010	3490
6	16,990	15,790	4010	31	17,070	16,090	7910	56	17,270	15,990	3580	81	17,100	15,990	3480
7	17,140	15,690	3610	32	16,960	16,150	3710	57	17,240	15,870	3480	82	16,950	15,830	3470
8	17,080	15,640	3900	33	17,080	16,310	3530	58	16,940	16,000	3460	83	17,010	15,840	3500
9	17,530	15,820	3830	34	16,950	15,820	3530	59	16,990	15,930	3500	84	16,770	15,830	3860
10	17,080	15,880	3950	35	17,100	15,850	3510	60	17,260	15,890	4250	85	17,080	16,030	3490
11	17,110	15,750	4230	36	16,990	15,970	3500	61	17,590	15,930	3500	86	16,930	15,940	3480
12	17,020	15,660	4010	37	16,980	16,020	3550	62	17,190	16,000	3470	87	17,080	15,970	3490
13	17,150	15,680	3790	38	16,780	16,030	3490	63	17,270	16,500	3480	88	16,870	15,860	3480
14	17,830	15,760	3680	39	17,070	15,960	3540	64	17,210	15,990	3480	89	17,270	15,990	3830
15	17,300	15,770	3930	40	16,880	15,850	3540	65	17,440	15,890	3490	90	16,990	16,060	4040
16	17,410	15,670	3760	41	17,060	16,010	3680	66	16,990	15,850	3750	91	17,030	15,880	3610
17	17,330	15,670	4130	42	16,820	16,000	3960	67	16,980	17,880	3480	92	16,910	15,800	3570
18	17,300	15,770	3860	43	16,990	15,880	3500	68	16,870	15,960	3460	93	17,730	16,140	3480
19	17,410	15,930	3850	44	16,880	15,790	3500	69	16,970	15,850	3500	94	17,060	15,880	3740
20	17,450	15,790	3840	45	17,080	16,000	3490	70	16,920	15,900	4670	95	16,970	16,100	4380
21	17,340	15,760	3850	46	16,900	15,920	4150	71	17,070	15,900	3540	96	16,950	15,830	3480
22	17,230	15,650	3680	47	17,070	15,880	3730	72	16,790	15,920	3460	97	17,050	15,880	3490
23	17,260	15,750	3770	48	16,850	15,790	3540	73	17,000	15,970	3790	98	16,780	15,910	3780
24	17,150	16,110	3860	49	17,130	16,010	3530	74	16,840	15,790	3480	99	17,000	15,990	3860
25	18,010	16,140	3960	50	16,890	16,150	3550	75	17,140	16,040	3540	100	16,900	17,280	3560
												Avg.	17,082.5	15,968.4	3761.8

Table 15
The results of the spatio-temporal query.

Row	Buoy	Latitude	Longitude	Days	Mean temperature	Row	Buoy	Latitude	Longitude	Days	Mean temperature
1	1	8.99	−140.26	5	24.82	14	17	−4.99	−109.97	13	24.84
2	3	2	−140.12	53	24.31	15	18	−8.04	−109.93	1	24.92
3	4	0	−140	102	24.10	16	20	5.1	−94.92	6	24.77
4	5	−2	−140	50	24.37	17	21	0.73	−95.09	328	23.70
5	6	−4.99	−139.93	1	24.88	18	23	−1.98	−95.05	219	24.06
6	7	5.13	−124.92	8	24.46	19	24	−5	−95.1	85	24.16
7	8	1.96	−125.1	24	24.23	20	25	−8.01	−95.09	53	24.46
8	9	−0.1	−124.42	292	23.85	21	26	5.89	−179.87	1	23.74
9	10	−2.04	−124.91	8	24.81	22	27	7.01	−140.15	3	24.48
10	12	−7.98	−125	1	24.96	23	39	7.97	−154.98	4	22.94
11	14	5.02	−109.97	2	24.96	24	41	1.89	−156.06	1	24.91
12	15	2.06	−110.22	150	23.98	25	47	4.98	147.02	3	24.01
13	16	−2.03	−110.09	143	24.37	26	56	−0.18	165.83	1	24.09
						27	60	0.01	−109.89	440	23.60

The spatial query is also run 100 times by SQL Statements in the three data models, and the obtained averages are 745.2, 0.8 and 2.3 milliseconds for the Tabular, Normal and HODM databases, respectively. The QET in the Tabular model is significantly higher than the other models. The main reason for this difference is due to the mixing of spatial and temporal values, which requires the model to read complete records (178,080 rows) for each calculation, while the temporal values are not required for reading. The total number of records that must be read to answer this query is 62, according to the number of locations. The Normal structure has a better QET when compared to the HODM, because the temporal values are completely separated from the spatial values in the Normal structure, while the temporal values are considered through nested tables in the HODM.

4.2.3. Temporal query

The temporal query refers to values that change over time. Time intervals and repetitious patterns of change are examples of temporal query types. The SQL statements in the three data models are displayed in Fig. 24. The results of the temporal query are illustrated in Table 13.

The QET of the temporal query for the three data models is obtained within milliseconds, as shown in Table 14. The HODM delivers a better result than the Tabular and Normal structures. The chart of QET for the temporal query is displayed in Fig. 25.

4.2.4. Spatio-temporal query

The spatio-temporal queries use spatial and temporal parameters simultaneously. Of course, spatio-temporal queries are not always more complex than other types of queries, although they often must consider all of the records in the spatio-temporal database. We include a sample spatio-temporal query based on the climate change dataset in Fig. 26.

The answer to the mentioned query is presented in Table 15.

The query execution times are shown in Table 16. In this case, the HODM also performs better than the other data models. The curves of the spatio-temporal query execution times are displayed in Fig. 27.

5. Conclusion and future work

In this paper, we proposed a novel natured-inspired conceptual definition based on the structure and properties of hair for storing and processing spatio-temporal data with specific attributes and operators has been proposed, namely hair-oriented data model. The experimental results demonstrated that the HODM has a smaller data storage size than the un-normalization (tabular) entity-relationship model, which is demonstrated either through the use of a formula or an experiment. The HODM has also been shown to have good performance in some types of querying, such as basic, temporal and spatio-temporal. This study also demonstrates that the application of nested table is useful for reducing the file size and improving the querying time. The main reason for this is due to the storage of data in indexes and due to the fact that no primary key is needed for the relationship between spatial and temporal data. However, many other issues must be taken into consideration, especially when querying all of the spatio-temporal data sorted that is not based on time. The proposal for using HODM aims to develop a data model for the spatio-temporal data warehouse that can be used to develop quality spatio-temporal data and that is easy to maintain with various operations being given to it in regard to the specific domain of data mining. Using these criteria, the spatio-temporal data warehouse has supported the development of real-time spatio-temporal data mining.

However, the model in its current form has a limitation in that it cannot accept movable objects with variable spatial specifications. This problem is inevitable as the model's features are derived from the characteristics of natural hair. A number of issues merit attention in the future work. Other advantages and benefits of Hair Oriented Data Model such as better predictability, data analysis performance with better quality are not proved in this research. In addition, security operations such as Tangle, Cover and Wig will also be defined, implemented and tested in the proposed data model. Model developing is essential through preparing soft tools with graphic user interfaces. Obviously, preparing model application in different environments such as urban transition system, gas and petroleum resource management, educational information systems and many other cases cause to improve structural and mining definitions.

To achieve this result, our next study will focus on the mining operation by investigating spatio-temporal databases such as MADS and Geoscience. We will utilize Comb, Plait and Color as clustering, classification and association operations, respectively. This study aims to see how the mining operation is implemented directly from the database and to measure its performance in terms of accuracy and processing times, in comparison to the previous model. The HODM has been successfully shown to be sufficient to use for spatio-temporal climate change, and it is believed that the HODM will be successful when used in other spatio-temporal domains.

Table 16
The query execution time for the spatio-temporal query.

Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)	Iteration No.	Tabular time (ms)	Normal time (ms)	HODM time (ms)
1	80	50	40	26	80	50	50	51	60	50	30	76	40	50	50
2	60	50	40	27	60	50	50	52	50	50	50	77	60	40	60
3	80	50	40	28	80	50	40	53	60	40	60	78	60	50	50
4	60	50	40	29	60	40	50	54	60	50	40	79	70	30	50
5	60	80	50	30	70	30	40	55	40	50	50	80	60	50	50
6	80	50	50	31	80	40	50	56	70	50	30	81	60	50	50
7	60	40	50	32	80	40	40	57	60	50	50	82	70	50	50
8	60	40	50	33	60	50	40	58	50	40	50	83	60	50	40
9	60	50	30	34	50	50	50	59	60	50	30	84	60	30	50
10	80	40	50	35	90	50	40	60	70	50	50	85	60	50	50
11	60	70	40	36	60	30	50	61	40	40	50	86	70	50	50
12	60	80	50	37	60	50	40	62	70	50	50	87	60	50	50
13	50	40	50	38	80	40	50	63	60	40	30	88	60	50	40
14	60	40	30	39	60	30	40	64	50	50	50	89	60	50	50
15	60	30	50	40	80	50	50	65	60	40	40	90	70	30	50
16	70	40	50	41	80	40	40	66	70	50	40	91	60	50	40
17	80	50	50	42	80	50	40	67	60	40	50	92	50	40	50
18	80	40	40	43	50	50	70	68	50	50	50	93	50	40	50
19	80	50	40	44	60	50	40	69	60	40	50	94	60	40	50
20	60	50	50	45	40	50	50	70	60	50	30	95	60	40	50
21	80	50	50	46	70	30	40	71	60	40	50	96	70	50	50
22	60	50	50	47	90	40	40	72	50	40	40	97	60	30	50
23	80	30	40	48	60	60	30	73	60	40	40	98	70	50	40
24	60	50	40	49	60	50	40	74	60	60	50	99	70	50	50
25	80	50	50	50	70	30	50	75	70	50	30	100	40	50	30
												Avg.	63.8	46	45.3

References

Bertossi, L. (2011). Database repairing and consistent query answering. *Synthesis Lectures on Data Management, 3*, 1–121.

Chau, V. T. N., & Chittayasothorn, S. (2008). A temporal object relational SQL language with attribute timestamping in a temporal transparency environment. *Data & Knowledge Engineering, 67*, 331–361.

Chiong, R. (2009). *Nature-inspired algorithms for optimisation*: Vol. 193. Chennai, India: Springer.

Cuevas, L., Marín, N., Pons, O., & Vila, M. A. (2008). pg4DB: A fuzzy object-relational system. *Fuzzy Sets and Systems, 159*, 1500–1514.

de Caluwe, R., & Bordogna, G. (2004). *Spatio-temporal databases: Flexible querying and reasoning*. New York, USA: Springer.

de Castro, L. N. (2007). Fundamentals of natural computing: An overview. *Physics of Life Reviews, 4*, 1–36.

de Tré, G., de Caluwe, R., Hallez, A., & Verstraete, J. (2003). Modelling of fuzzy and uncertain spatio-temporal information in databases: A constraint-based approach. In B.-M. Bernadette, F. Laurent, L. F. Ronald, R. YagerA2- Bernadette Bouchon-Meunier, & R. Y. Ronald (Eds.), *Intelligent systems for information processing* (pp. 117–128). Amsterdam: Elsevier Science.

Del Mondo, G., Rodríguez, M., Claramunt, C., Bravo, L., & Thibaud, R. (2013). Modeling consistency of spatio-temporal graphs. *Data & Knowledge Engineering, 84*, 59–80.

Deshpande, A., Ives, Z., & Raman, V. (2007). Adaptive query processing. *Foundations and Trends in Databases, 1*, 1–140.

Fei, K.L.M., & Ma, G.W.I.S. (2007). Bio-Inspired Computational Intelligence and Applications.

Floreano, D., & Mattiussi, C. (2008). *Bio-inspired artificial intelligence: Theories, methods, and technologies*. Massachusetts, USA: The MIT Press.

Grandi, F., & Mandreoli, F. (2003). A formal model for temporal schema versioning in object-oriented databases. *Data & Knowledge Engineering, 46*, 123–167.

Guo, T., Papaioannou, T. G., & Aberer, K. (2014). Efficient indexing and query processing of model-view sensor data in the cloud. *Big Data Research, 1*, 52–65.

Han, J., Kamber, M., & Pei, J. (2006). *Data mining: Concepts and techniques*. Massachusetts, USA: Morgan kaufmann.

Harrington, J.L. (2010). SQL Clearly Explained.

Hasançebi, O., Teke, T., & Pekcan, O. (2013). A bat-inspired algorithm for structural optimization. *Computers & Structures, 128*, 77–90.

Le, H. H., Gabriel, P., Gietzel, J., & Schaeben, H. (2013). An object-relational spatio-temporal geoscience data model. *Computers & Geosciences, 57*, 104–115.

Madraky, A., Othman, Z. A., & Hamdan, A. (2015). Hair-oriented data model for spatio-temporal data mining. *International Review on Computers and Software, 10*, 90–101.

Madraky, A., Othman, Z. A., & Hamdan, A. R. (2012). Hair data model: A new data model for spatio-temporal data mining. In *Data mining and optimization (DMO), 2012 4th conference on* (pp. 18–22). IEEE.

Madraky, A., Othman, Z. A., & Hamdan, A. R. (2014). Analytic methods for spatio-temporal data in a nature-inspired data model. *International Review on Computers and Software, 9*, 547–556.

Malinowski, E., & Zimányi, E. (2008). A conceptual model for temporal data warehouses and its transformation to the ER and the object-relational models. *Data & Knowledge Engineering, 64*, 101–133.

Menon, V., Jayaraman, B., & Govindaraju, V. (2012). Spatio-temporal querying in smart spaces. *Procedia Computer Science, 10*, 366–373.

Mitsui, T. (1997). *New cosmetic science*. Amsterdam, The Netherlands: Elsevier.

Mok, W. Y. (2007). Designing nesting structures of user-defined types in object-relational databases. *Information and Software Technology, 49*, 1017–1029.

Mousavi, M., Bakar, A. A., & Vakilian, M. (2015). Data stream clustering algorithms: A review. *International Journal of Advances in Soft Computing & Its Applications, 7*, 2–4.

Ordonez, C., Maabout, S., Matusevich, D. S., & Cabrera, W. (2013). Extending ER models to capture database transformations to build data sets for data mining. *Data & Knowledge Engineering, 89*, 38–54.

Parent, C., Spaccapietra, S., & Zimányi, E. (1999). Spatio-temporal conceptual models: Data structures+ space+ time. In *Proceedings of the 7th ACM international symposium on advances in geographic information systems* (pp. 26–33). ACM.

Parent, C., Spaccapietra, S., & Zimányi, E. (2006). *Conceptual modeling for traditional and spatio-temporal applications: The MADS approach*. Heidelberg: Springer.

Parent, C., Spaccapietra, S., & Zimányi, E. (2006). The MurMur project: Modeling and querying multi-representation spatio-temporal databases. *Information Systems, 31*, 733–769.

Park, C.-M., Whang, K.-Y., Lee, J.-J., & Song, I.-Y. (2001). A cost-based buffer replacement algorithm for object-oriented database systems. *Information Sciences, 138*, 99–117.

Perumal, M., Velumani, B., Sadhasivam, A., & Ramaswamy, K. (2015). Spatial data mining approaches for GIS–A brief review. In *Emerging ICT for bridging the future-proceedings of the 49th annual convention of the computer society of India CSI volume 2* (pp. 579–592). Springer.

Philippi, S. (2005). Model driven generation and testing of object-relational mappings. *Journal of Systems and Software, 77*, 193–207.

Pinet, F. (2012). Entity-relationship and object-oriented formalisms for modeling spatial environmental data. *Environmental Modelling & Software, 33*, 80–91.

Rakêt, L. L., & Markussen, B. (2014). Approximate inference for spatial functional data on massively parallel processors. *Computational Statistics & Data Analysis, 72*, 227–240.

Ramakrishnan, R., & Gehrke, J. (2003). Database Management Systems.

Ribarić, S., & Hrkać, T. (2011). A model of fuzzy spatio-temporal knowledge representation and reasoning based on high-level Petri nets. *Information Systems, 37*(3), 238–256.

Sagar, B. S. D. (2013). *Mathematical Morphology in Geomorphology and GISci*. Boca Raton, FL, USA: CRC Press.

Sözer, A., Yazıcı, A., Oğuztüzün, H., & Taş, O. (2008). Modeling and querying fuzzy spatiotemporal databases. *Information sciences, 178*, 3665–3682.

Steer, K. C., Wirth, A., & Halgamuge, S. K. (2009). The rationale behind seeking inspiration from nature. In *Nature-Inspired Algorithms for Optimisation* (pp. 51–76). Berlin, Germany: Springer.

Triglav, J., Petrovič, D., & Stopar, B. (2011). Spatio-temporal evaluation matrices for geospatial data. *International Journal of Applied Earth Observation and Geoinformation, 13*, 100–109.

UCI. (1999). El Nino Data. In D. O. Statistics, & I. S. University (Eds.), *El Nino Data*. University of California, Irvine: Iowa.

Verstraete, J., Tré, G. D., & Hallez, A. (2006). Fuzzy spatial data modeling: An extended bitmap approach. In B.-M. Bernadette, C. Giulianella, G. C. Ronald, R. YagerA2- Bernadette Bouchon-Meunier, & R. Y. Ronald (Eds.), *Modern information processing* (pp. 321–331). Amsterdam: Elsevier Science.

Werstein, P. (1998). A performance benchmark for spatiotemporal databases. In *Tenth annual colloquium of the spatial information research centre, 16-19 Dec, Dunedin, New Zealand* (pp. 1365–1374).

Wikle, C. K. (2015). Modern perspectives on statistics for spatio-temporal data. *Wiley Interdisciplinary Reviews: Computational Statistics, 7*, 86–98.

Yang-Ming, J., & Qin-Lin, D. (2011). Mine evolution dynamic monitor based on dynamic object-oriented model. *Procedia Engineering, 26*, 2181–2190.

Yeung, A. K., & Hall, G. B. (2007). *Spatial database systems: Design, implementation and project management*: Vol. 87. Dordrecht, The Netherlands: Springer.

Yu, H., Davis, M., Wilson, C. S., & Cole, F. T. (2008). Object-relational data modelling for informetric databases. *Journal of Informetrics, 2*, 240–251.

Zadeh, L. A. (1975). The concept of a linguistic variable and its application to approximate reasoning—I. *Information Sciences, 8*(3), 199–249.

Zhou, Y.-X, Liu, G.-J., Fu, E.-J., & Zhang, K.-F. (2009). An object-relational prototype of GIS-based disaster database. *Procedia Earth and Planetary Science, 1*, 1060–1066.

Zomaya, A. Y. (2006). *Handbook of nature-inspired and innovative computing: Integrating classical models with emerging technologies*. New York, USA: Springer.