International Journal of Database Theory and Application
Vol.8, No.1 (2015), pp.197-204
http://dx.doi.org/10.14257/ijdta.2015.8.1.20

Query Algebra for the Very Loosely Structured Data Model

Ying Pan, Changan Yuan, Zhengqi Li and Wenjing Li

(College of Computer and Information Engineering, Guangxi Teachers Education University, Nanning 530023)

Abstract

In order to realize the idea of pay-as-you-go (PAYG) data management which dataspace emphasizes, the very loosely structured data model is usually used to describe the massive, heterogeneous and dynamic data in dataspace. However, the present study mainly concentrates on the applications of the data model, and the query theory research is less. The query algebra is a theoretical foundation for query and its optimization in a PAYG fashion, so that how to establish a complete algebra based on the characteristics of loosely structured data model is an important problem need to be solved. In this paper, a formal definition of very loosely structured data model is given, then the query model and query algebra based on the model are proposed, which support to not only the operations such as set operators, selection, projection and join, but also association query in dataspace.

Keywords: *dataspace, very loosely structured model, query algebra*

1. Introduction and Motivation

How to manage mass, heterogeneous and evolving data efficiently is the major problem in data management field. However, the existing database systems and data integration technology require the hard up-front investment before powerful functionalities can be provided, and they are thus difficult to management such heterogeneous and evolving data timely [1, 2].

Dataspace proposed in year 2005 [2], a new style of pay-as-you-go (PAYG) data management, addresses the above challenges. Dataspace offers the services (e.g., search and query) on data in an incrementally, PAYG fashion, without requiring the expensive effort, and automatically enhance services over time. That is, the management system firstly offers the simple keyword-based query on data without requiring the expensive up-front investment, and then gradually enhances services such as semantic search over time.

The very loosely-structured data model is generally used in dataspace. This data model does not enforce a schema over the data. In contrast to the existing data models, this model can describe the heterogeneous data sources more easily with little up-front cost, which is more helpful to realize the idea of PAYG data management which dataspace emphasizes. Dataspace has triggered the great attention, and a lot of relevant research papers have been published in major conferences such as SIGMOD and VLDB [3-6]. However, these researches mainly discussed the construction of the loosely structured model and its query efficiency from the perspective of application, and the query theory research is less. The query theory is the solid foundation to realize query and its optimization in PAYG fashion, and how to build a complete query algebra based on the characteristics of loosely structured data model is a very important problem need to be solved.

In this paper, we study query algebra based on the very loosely structured data model. The contributions of this paper can be summarized as follows: 1) We present a formal definition of very loosely structured data model; 2) We propose the query model and

ISSN: 2005-4270 IJDTA

query algebra, which support to not only the algebra operations such as set operators, selection, projection and join, but also association query in dataspace.

The remainder of this paper is organized as follows. The next section introduces the works which relevant to our research. Section 3 gives the formal definition of very loosely structured data model. Section 4 proposes query model and query algebra based on the data model. Section 5 proposes association query. Finally, Section 6 concludes the paper and outlines our future work.

2. Related Work

In [7], a data model similar to RDF was presented to describe the heterogeneous data as a set of triples. Then a system was proposed to support seamless search and querying on both structured and unstructured data. For example, a user may use a SQL query on unstructured data sources. The key to answer structured queries on unstructured data was as follows: Firstly, a given structured query was translated into a keyword query, and then the keyword query was answered over unstructured data. However, strictly speaking, RDF is not a very loosely structured data model [8], the data model in [7] is thus not a very loosely structured data model too.

The authors of [9] presented a very loosely structured data model named iDM, which described all personal information (*e.g.*, Word documents, relational data, XML, file content, folder hierarchies, email and data streams) into resource view graph. A resource view was a 4-tuple representing name component, tuple component, content component, and group component. The components of resource view can express structured, semi-structured and unstructured pieces of data, and these resource views are linked to each other to form resource view graph. The authors also presented a query language iQL, which was similar in spirit to NEXI [10], to query iDM model. iQL is composed of keyword expressions and XPath navigational restrictions, and it supports to query a resource view graph without requiring semantic integration. iQL also defines the extensible algebraic operations such as join and grouping. Moreover, iQL includes some features important for a dataspace management system, such as support to updates and continuous queries.

In [11], a general model of dataspace was presented, this model was composed of data objects, where each data object was a collection of data items, which were formed by attribute–value pairs. Then a theory of search queries was proposed, and the applications of the theory to classical database relations and to attribute–value dataspace were presented. Moreover, associative search and semijoin algebra were discussed.

Based on the data model in [11], the authors of [12] proposed an entity retrieval model for web data. This model is more expressive for describing semi-structured information found in distributed and heterogeneous web data sources. Then a boolean search model was proposed, and the query algebra similar to the relational query algebra was introduced.

In summary, there is not the unified standard for the definition of very loosely structured data model, and the corresponding query models for these data models also have their own characteristics. The study of the query theory is relatively few and not mature, therefore it still need further research.

3. Data Model

In recent years, some very loosely structured data models are proposed for different scenes. However, there are some weaknesses in these models. For example, iDM focus on describing a sequence of ordered relationship among data sources [9], and it is less convenient to describe more complex relationship. The model in [11] describes the edge (relationship) by attribute–value pairs, so it is not easy to distinguish between the attributes of entity and the relationship among entities. Therefore, in [13, 14] we propose

the definition of more general model, which using the concept of edge to describe the relationships. In this paper, we further refine the formal definition of very loosely structured data model VLDM based on our model in [13, 14].

Definition 1. **(**VLDM) The data in dataspace is described by a VLDM, denoted as $G := (N, E)$, where N is a set of nodes $\{N_1, \ldots, N_k\}$, each node N_i is a collection of attribute-value pairs, each value can be atomic, a bag of words or text content. E is a set of labeled, directed edges (N_i, N_j, L), where $N_i, N_j \in N, i \neq j$ and L is a label which can be a *null* value.

Definition 2. (attribute-value pairs**)** each node N_i is a set of attribute-value pairs, denoted as $N_i = \{(a_1^i, v_1^i), \ldots (a_n^i, v_n^i)\}$. A is a set of attributes, and V is a set of values, then $(a_k^i, v_k^i) \in A \times V$, $1 \leq k \leq n$.

Definition 3. (blank node) N_i called blank node if $N_i = \varnothing$. Specially, if node has attribute-value pair (a, v), and a or v is uncertain, then they may be *null* values, denoted as $(null, v)$ or $(a, null)$.

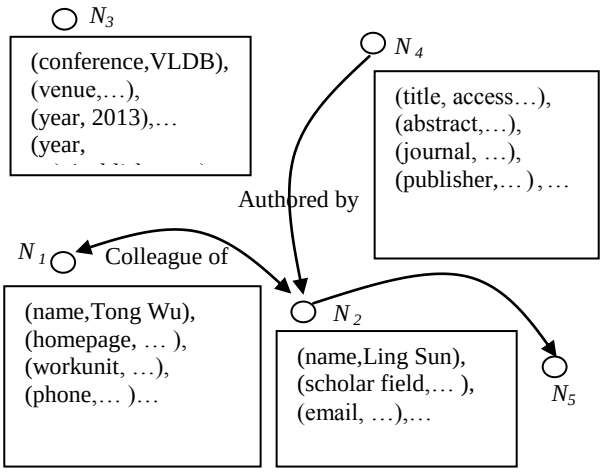

Figure 1. VLDM Describes Scholar Information In Dataspace

Figure 1 shows that VLDM describes scholar information in dataspace, where nodes N_1 and N_2 describe author information, N_3 describes conference information, N_4 describes paper information, and N_5 is a blank node. VLDM is a very loosely structured data model, in which the data is not necessarily to be conformed to a uniform schema. That is, the set of attributes of each node may be different, and even there exist the nodes without attribute-value pairs (blank nodes) which are used to describe uncertain information. Furthermore, edge describes any kind of relationships among nodes, and can be a *null* value if the edge has not an explicit label. Specifically, when there is not any connection between nodes, $E = \varnothing$.

4. Query Model and Query Algebra

Definition 4. (query model) For a given G, assume N and E are both finite sets, and G is a complete set, *i.e.*, there must be no node N_k, which satisfies conditions: $N_k \notin G$ and at the same time N_k in E (*e.g.*, edge (N_k, N_j, L)). Then a query Q returns the result $Q(G)$ which selects a subset of nodes, *i.e.*, $Q(G) \subseteq G$.

The best-effort query is requested for the dataspace system, that is, the system firstly provides the simple keyword-based query with little up-front investment, with time, more

powerful functionalities (*e.g.*, semantic query and search) are provided gradually when more efforts are invested. According to these query characteristics, we define the query expressions as follows.

Definition 5. (keyword query expression) The expressions of keyword query are given as follows:
exp := k | exp logop exp,
where k is a keyword, the logical operators are denoted as logop:= and | or | not. exp(G) is the result of exp, which returns those nodes containing the keyword k, i.e.,
exp(G):= $\{N_i \mid N_i \in G \wedge 'k' \in W_i\}$
where W is a set of terms (such as attribute, value and the label of edge) in G, and W_i is the terms set of node N_i.

Example 1. Let exp:= 'Ling Sun' and 'email', then
exp(G):= $\{N_i \mid N_i \in G \wedge$ 'Ling Sun' $\in W_i \wedge$ 'email' $\in W_i\}$.
exp(G) returns the node N_2 containing the keyword "Ling Sun "and "email "(see Figure 1).

Definition 6. (attribute–value pairs query expression) The query expressions of attribute–value pairs are given as follows:
exp:=(a : v) | exp logop exp,
where (a : v) is a attribute–value pair (a , v). Specially, we use a wildcard * to describe attributes or values. For example, we represent the attribute–value pair (*, v) and (a, *) as (*: v) and (a :*), respectively.
exp(G) is the result of exp, which returns those nodes containing the attribute–value pair (a , v) , *i.e.*,
exp(G):= $\{N_i \mid N_i \in G \wedge \exists(a,v) \subseteq A_i \times V_i\}$

Example 2. Let exp:= (name: Tong Wu), then
exp(G):= $\{N_i \mid N_i \in G \wedge \exists(\text{name},\text{Tong Wu}) \subseteq A_i \times V_i\}$
exp(G) returns the node N_1 containing the attribute-value pair (name, Tong Wu) (see Figure 1).

In order to utilize relational algebra theory, we describe VLDM by the classical relational databases.

Definition 7. (relational representation of VLDM) The nodes and their attribute-value pairs in VLDM are represented by relational schema $R(Nid, attr, val)$, $Nid \in N$, and $(attr, val) \in A \times V$, where *Nid*, *attr* and *val* represent node identifier, attribute and value, respectively. Otherwise, all edges in VLDM, which defined by the form of (N_k, N_j, L), are represented by relational schema $R'(Nid, Nid1, L)$. Dataspce thus may be represented by the set of triples.

According to relational algebra theory, we present the following definition:

Definition 8. (set operations) Intersection: $R_1 \cup R_2 = \{t \mid t \in R_1 \vee t \in R_2\}$,
Union: $R_1 \cap R_2 = \{t \mid t \in R_1 \wedge t \in R_2\}$,
Difference: $R_1 - R_2 = \{t \mid t \in R_1 \wedge t \notin R_2\}$,
where R_1 and R_2 are relations, t is a tuple.

The result of set operations can be seen as a subset of the dataspace graph G, where *Nid* represents node, (*attr*, *val*) represents the attribute-value pair of node, and $(Nid, Nid1, L)$ represents an edge.

Definition 9. (selection operation) the selection operation defines a view relation that contains only some of *R*'s tuples that satisfy some condition K . The selection operation can be denoted as

$\sigma_k(R) = \{t \mid t \in R \wedge K(t) = true\}$,

where K is the conditional expression, its result is either "true" or "false".

The result of selection operation can be seen as a subset of the dataspace graph *G* that satisfies some condition K . For example, $\sigma_{attr='conference' \wedge val='VLDB'}(R)$ returns the node N_3 which contains the attribute-value pair (conference, VLDB).

Definition 10. (projection operation) the projection operation produces a new relation from a relation *R*, this new relation contains only some of *R*'s columns, which can be denoted as:

$\pi_{j1,\ldots,jk}(R) = \{t \mid t = (t_{j1,\ldots,jk}) \wedge (A_{j1,\ldots,jk}) \in R\}$,

where $A_{j1,\ldots,jk}$ are the set of attributes.

For example, $\pi_{Nid}(R)$ returns the collection of nodes in *G*.

Definition 11. (join operation) the join operation defines a relation containing the tuples which satisfies the condition $B\theta C$ from the Cartesian product of R_1 and R_2. The join operation can be denoted as:

$R_1 \underset{B\theta C}{><} R_2 = \{t_1, t_2 \mid t_1 \in R_1 \wedge t_2 \in R_2 \wedge t_1[B]\, \theta\, t_2[C]\}$,

where θ may be one of the comparison operators (*e.g.*, <, >, =), *B* is the attribute from R_1, and *C* is the attribute from R_2.

Example 3. Find all nodes matching keyword "VLDB" and containing the attribute-value pair (author, Ming Li). The query expression and the algebra operations are given as follows:

exp:= 'VLDB' *and* (author : Ming Li)

exp(G)= $\pi_{Nid}(\sigma_{val='VLDB'}(R)) \cap \pi_{Nid}(\sigma_{attr=\text{author} \wedge val='\text{Ming Li}'}(R))$

Theorem 1. For relational representation of VLDM, its relational algebra operations are complete.

Proof. By literature [15], the set of relational algebra operations $\{\sigma, \pi, \cup, -, ><\}$ is complete. Relational algebra operations of VLDM can simulate these five basic operations (See the definition of 8-11). Therefore, its relational algebra operations are also complete.

5. Association Query

Association query based on the edges (relationships) is one of the important characteristics for dataspace query in a PAYG fashion. Association query can find out not only the nodes satisfy the query conditions, but also the nodes that are related to those nodes.

For example, we can use association query to find out the scholar Tong Wu's colleague (see Figure 1). Firstly, we find the node N_1 which contains the attribute-value pairs (name, Tong Wu), and then we find that this node having an edge labeled "Colleague of", so, the query result is N_2, the other node of this edge, which having an attribute-value pairs (name, Ling Sun). That is, Ling Sun is Tong Wu's colleague.

In the above example, we determine whether one pair of nodes having the colleague relationship by an edge labeled "Colleague of". The edges which actually exist in the graph G called extensional edges. In contrast to extensional edges, intensional edges are computed in query processing. The data relationships in dataspace require to be described gradually and dynamically, thus, it is very important that intensional edges may be computed lazily. To this end, we define the following association query expressions for querying (computing) extensional edges and intensional edges.

Definition 12. (association query expression)The association query expressions are defined as follows:

$\exp := \text{Edge}(\text{lable}, [\theta(l)])\ \exp 1$,

where $\theta(l)$ is optional, it is used to query intensional edges, and it discovers all node pairs which satisfy the condition.

$\exp(G) := \{N_i \mid N_i \in G \wedge N_j \in \exp 1(G) \wedge ((N_i, N_j, lable) \in E \ or\ (N_i, N_j) \in \theta(l))\}$,

the result of exp(G) is the nodes which having an edge with a given label, or satisfying the condition $\theta(l)$.

We can use a wildcard * to describe an uncertain label. For example, the query expressions "Edge (*) exp1" return those nodes which are connected with the nodes in exp1(G)

Example 4. 1) exp:= Edge (Colleague of) (name: Tong Wu)

Then the query result exp(G) returns the nodes which are connected with the nodes containing the attribute-value pairs (name, Tong Wu) and having an edge with a label "Colleague of".

2) exp:= Edge (Colleague of, θ(Colleague of)) (name: Tong Wu)

Where $\theta(\text{Colleague of}) := (N_i.workunit = N_j.workunit)$,which is used to judge if there exists a colleague relationship in the node pair (N_i, N_j).

Then the query result exp(G) returns not only the nodes which are connected with those nodes containing the attribute-value pairs (name, Tong Wu) and having an edge with a label "Colleague of", but also the nodes which satisfy the query conditions $\theta(\text{Colleague of})$.

Note that, $\theta(\text{Colleague of})$ can be computed in query processing. For each pair of nodes which satisfy the condition $\theta(\text{Colleague of})$, we may add one intensional edge labeled "Colleague of" between two nodes in the graph G, thus, intensional edges can be added dynamically in a PAYG fashion.

In the following, we present the relational algebra of association query. The definition of semijoin operation is similar to the traditional relational algebra. Specially, we define θ-semijoin as follows:

Definition 13. (θ-semijoin) θ-semijoin which satisfies the condition θ is denoted as

$R_1 \underset{B\theta C}{>\!\!<} R_2 = \{t_1 \in R_1 \mid \exists t_2 \in R_2 \wedge t_1[B]\,\theta\, t_2[C]\}$.

Theorem 2. (algebra operations of association query) for a given query expressions "exp:= Edge(lable,[$\theta(l)$]) exp1 ", assume the nodes in exp1(G) are represented by relational schema R_1, then exp(G) is given by:

1) if exp:= Edge(lable) exp1, then

exp(G):= $\pi_{Nid1}((\sigma_{l=lable}(R'))_{R'.Nid1=R_1.Nid}><R_1))$ ①

2) if exp:= Edge(lable,$\theta(l)$) exp1, then

exp(G):= $\pi_{Nid1}((\sigma_{l=lable}(R'))_{R'.Nid1=R_1.Nid}><R_1)) \cup \pi_{Nid}(R_1 \underset{R_1.Nid \neq R.Nid \wedge R_1.B\ \theta\ R.C}{><} R)$

Proof. By Definition 12, exp(G) in the first case returns the nodes which are connected with those nodes in R_1 having an edge with a given label, i.e., the node set in the exp(G) satisfies $\{N_i \mid N_i \in G \wedge N_j \in \exp1(G) \wedge (N_i, N_j, lable) \in E\}$. Obviously, exp(G) can be computed by the algebra operations $\pi_{Nid1}((\sigma_{l=lable}(R'))_{R'.Nid1=R_1.Nid}><R_1))$, then Formula is proved.

exp(G) in the second case returns not only the nodes which satisfy the conditions $\theta(l)$, but also the nodes from the query results in the first case. On the other hand, the nodes which satisfy the conditions $\theta(l)$ can be computed by the algebra operations $\pi_{Nid}(R_1 \underset{R_1.Nid \neq R.Nid \wedge R_1.B\ \theta\ R.C}{><} R)$, then Formula is proved.

6. Conclusions and Future Work

In this paper, we present a formal definition of very loosely structured data model VLDM, then we propose a query model and query algebra for VLDM, which support to not only the basic algebra operations but also association query in dataspace.

In the future, we plan to study the query optimization in a PAYG fashion. VLDM is a special kind of graph model, and the key to the study is how to find the appropriate edge (relationship) for querying paths. Therefore, we will research this issue by referencing the query optimization theory on graph model.

Acknowledgements

This research is supported by the National Natural Science Foundation of China (Grant Nos. 61363074, 61363037, 61163012), Natural Science Foundation of Guangxi Province of China (Grant No. 2013GXNSFAA019346), Scientific Research Fund of Guangxi Education Department of China (Grant No. 2013YB148), and Doctoral Scientific Research Foundation of Guangxi Teachers Education University (Grant No. 20110819).

References

[1] Y. Li, X. Meng and X. Zhang, "Research on dataspace", Chinese Journal of Software, vol. 19, no. 8, **(2008)**, pp. 2018-2031.
[2] M. Franklin, A. Halevy and D. Maier, "From databases to dataspaces: a new abstraction for information management", ACM SIGMOD Record, vol. 34, no. 4, **(2005)**, pp. 27-33.
[3] A. Das Sarma, X. Dong and A. Halevy, "Bootstrapping pay-as-you-go data integration systems", Proceedings of ACM Sigmod, **(2008)**, pp. 861-874.
[4] S. R. Jeffery, M. J. Franklin and A. Y. Halevy, "Pay-as-you-go user feedback for dataspace systems", Proceedings of ACM SIGMOD, **(2008)**, pp. 847-860.
[5] M. A. Salles, J. P. Dittrich, S. K. Karakashian, O. R. Girard and L. Blunschi, "iTrails: pay-as-you-go information integration in dataspaces", Proceedings of VLDB, **(2007)**, pp. 663-674.
[6] C. Hedeler, K. Belhajjame, N. W. Paton, A. A. Fernandes, S. M. Embury, L. Mao, C. Guo, "Pay-as-you-go mapping selection in dataspaces", Proceedings of ACM SIGMOD, **(2011)**, pp. 1279-1282.
[7] X. Dong, "Providing best-effort services in dataspace systems", University of Washington, **(2007)**.

[8] J. Dittrich, "The iMeMex dataspace management system: architecture, concepts, and lessons learned", Dataspace: the Final Frontier, vol. 5588, **(2009)**, pp. 7-12.
[9] J. P. Dittrich and M. A. Salles, "iDM: a unified and versatile data model for personal dataspace management", 32nd International Conference on Very Large Data Bases. Seoul, Korea: VLDB Endowment, **(2006)**, pp. 367-378.
[10] A. Trotman and B. Sigurbj Rnsson, "Narrowed extended xpath i (NEXI)", Advances in XML Information Retrieval, **(2005)**, pp. 16-40.
[11] G. H. Fletcher, G. H. Van and G. D. Van, "Towards a theory of search queries", 12th International Conference on Database Theory. Saint Petersburg, Russia: ACM, **(2009)**, pp. 201-211.
[12] R. Delbru, S. Campinas and G. Tummarello, "Searching web data: An entity retrieval and high-performance indexing model", Web Semantics: Science, Services and Agents on the World Wide Web, vol. 10, **(2012)**, pp. 33-58.
[13] Y. Pan and Y. Tang, "Pay-as-you-go web services discovery", Journal of Computer Research and Development, vol. 49, no. 12, **(2013)**, pp. 2549-2558.
[14] Y. Pan, Y. Tang and H. Liu, "Access control in very loosely structured data model using relational databases", Acta Electronic Sinica, vol. 40, no. 3, **(2012)**, pp. 600-606.
[15] R. Elsmari and S. Navathe, "Fundamentals of database systems", Third Edition, Addison Wesley, USA, **(2000)**.

