International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 11, November-2014

A Survey: Different Approaches for Building Ontology and Algorithms used in Cache Optimization

B Jaibarathi,
ME (Computer Science Engg.),
Kumaraguru college of Technology,
India.

L Sowmya Devi
ME (Computer Science Engg.),
Kumaraguru college of Technology,
India.

M S Hema
Assistant Professor,
Dept of Computer Science Engg,
Kumaraguru college of Technology,
India.

Dr. S. Chandramathi
Dean,
Department of Electrical Science,
Sri Krishna College of Technology,
India.

***Abstract*— Data integration is an effective approach to combine data that reside in different sources for the purpose of analysis and decision making. Data federation is a data integration strategy used to create integrated virtual data store. Ontology is used to resolve semantic heterogeneity in data integration. This paper focus in different existing system that used ontology based data integration. This paper clearly explains the different existing system along with the comparison. In order to improve the access latency, cache is added in ontology based data integration process. Different cache optimization algorithms also explained along with the comparison. This paper attempts to summarize existing systems and major page replacement algorithms.**

Keywords — *Data integration, semantic heterogeneity, ontology, existing system, KRAFT, cache optimization algorithm, LRU.*

1. INTRODUCTION

Data integration provides the ability to manipulate data transparently across multiple data sources. The main advantage of data integration is to obtain a resultant data source with a complete and concise overview of all existing data without a need to access all sources separately. The data source is complete because no objects are missed out in the result and concise because no object is repeated twice in the result. The three types of data integration are data consolidation, data propagation and data federation. The creation of virtual view among different data sources is a challenging task due to the heterogeneities. Among all the heterogeneities, the semantic heterogeneity is caused by different meaning or interpretation of data which can be resolved by using ontology. Ontology is a formal explicit specification of a shared conceptualization. Conceptualization corresponds to an abstract model of a domain which identifies the relevant concepts and the relationship [1]. The ontology is used to represent domain knowledge to resolve semantic heterogeneities in data federation.

2. DATA INTEGRATION

Data Integration provides access to large amounts of data from alternative sources; data quality is becoming a first class property increasingly required by end-users.

i) Types of data integration

The different types of data integration are as follows in the table 1

S.No	Types of Integration	Process
1	Manual Integration	Integration process done manually by user.
2	Virtual data Integration	Provide Unified View.
3	Application Based Integration	Particular applications to implement.
4	Middleware Data Integration	It transfers the integration logic from applications to a new middleware layer.

Table: 1 Different types of data integration

ii) Data integration Strategies

New strategies to integrate data evolve from time to time, owing to the active research work being done in the field. The following are a few popular data integration strategies are compare in the table 2, and diagrams are in figure -1 and figure- 2.

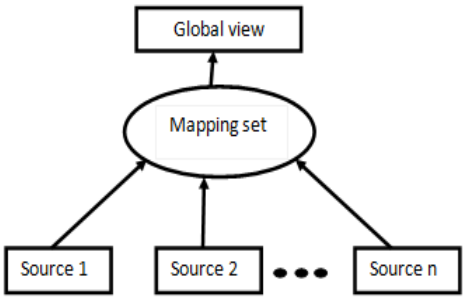

Figure: 1 Global As View

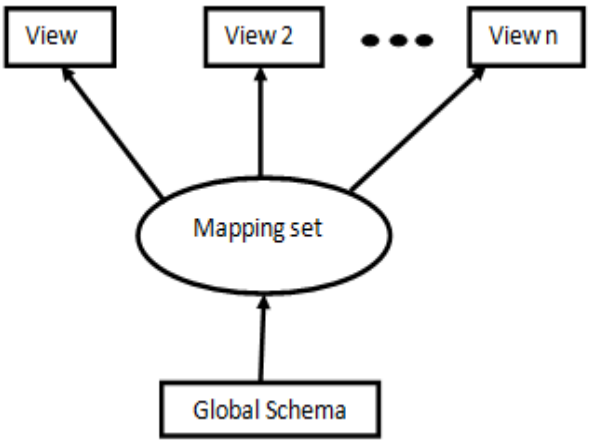

Figure: 2 Local As View

S.No	Strategies	Purpose	Advantage	Disadvantage
1	Global as View (GAV)	Set of views over source data	Query Processing	No of sources
2	Local as View (LAV)	Set of views over global schema	New source can added easily	Query Rewriting is complex
3	Global Local as view (GLAV)	Combination of global and local	Direct mapping, flexibility	Difficult to implement

Table: 2 Classifications of Strategies

3. DEVELOPMENT METHODOLOGY

This paper reviews the use on ontologies for the integration of heterogeneous information sources. Based on an in-depth evaluation of existing approaches to this problem we discuss how ontologies are used to support the integration task [3]. It also evaluates and compares the different existing ontology based projects. Some of the existing projects are discussed below are

3.1 KRAFT (Knowledge Reuse and Fusion / Transformation)

KRAFT is multisite research project conducted at University of Aberdeen, Cardiff and Liverpool in collaboration with BT in UK. It is just extension of Carnot Project to make legacy database system. The main objective is locating and extracting knowledge from multiple heterogeneous online sources and transform into union of knowledge [4] . Local ontologies allow the communication between heterogeneous resources that can maintain the intrinsic heterogeneity. It follow the two methods for execution there are building of shared ontologies and extracting of source ontologies It also detects set of ontology mismatch and establishing mapping between the shared ontology.

Workflow and coordinator transaction make use of intelligent agent.

3.2 INFOSLEUTH

Infosleuth is the system for integration of heterogeneous source developed by Microelectronics and Computer Technology Corporation, USA .It is the agent based system for information integration and analysis in distributed and dynamic environment with heterogeneous information sources. The main objective [5] is to retrieve and process information in heterogeneous information sources. TQML is used to create mapping between natural language query fragments. It is not human readable but constructed to reflect terminological expectation of prospective users. Infosleuth view information source at the level of relevant semantic concept. These information requests are independent of structure. This system processes the incoming document, extracting phrases that involve seed word and generate corresponding concept terms.

3.3 SIMS (Search in Multiple Sources)

SIMS Project was developed by research group at the University of South Carolina. It targeted to provide intelligent access to distributed and heterogeneous data source. Hierarchical terminological Knowledge base used with node representation of object, action and relation between nodes. It does not focus on building ontology instead of that it provide link between different ontologies. It uses the simple technique like Rewriting query, translation of query so it creates problem at the time of adding new source. Efficient handling of vocabulary sharing problem with no information loss.

3.4 COIN (Context Interchange)

COIN was initiated in 1991 with goal of achieving semantic interoperability among heterogeneous information source. This framework uses temporal contextual knowledge representation and reasoning capability to allow retrieval of data from multiple sources. Ontology used as formal knowledge for representing context knowledge and a mediation service to dynamically detect and reconcile semantic conflict. It offer more flexible and scaling when compare to other system.

3.5 CARNOT

CARNOT Project was initiated in 1990 by MCC with goal of addressing the problem of logically unifying physical distribution of heterogeneous information source. Varieties of techniques are used to address wide range of problem in achieving interoperation in heterogeneous environment. Processing of queries involve in heterogeneous query processing, relaxed transaction, workflow management, knowledge discovery and heterogeneous mode of integration. It work in five layer model, the layers are semantic service, distribution service, support service, communication service and access service.

3.6 MOMIS (Mediator Environment for Multiple Information Source)

MOMIS was joint Project among the Universita di Modenae Reggio Emilia, the Universita di Milano and University di Brescia with national Research Project "INTERDATA". MOMIS project [7] is closely related to SIMS based description logic. Semantic Approach was followed to integrate information based on conceptual schema or metadata of information source. This system support read only view of data that reside in multiple databases which make use of virtual and real world view.

3.7 OBSERVER (Ontology Based System Enhanced with relationship for vocabulary heterogeneity Resolution)

OBSERVER was conducted at the University of Zaregoza, Spain. This project mainly concern with resource discovery, structural/ format heterogeneity, modeling of information content, querying of information content and vocabulary problem. Pre-existing ontologies are used for evaluation. It provides broad knowledge about vocabulary use in different source. The major query processing tasks are connection between system, query Meta constructor, Resource discovery and Information Focusing, information access and correlation and Iteration.

3.8 GARLIC

Garlic system was built on complex wrapper architecture to describe local source with object oriented language. These complex objects are manually unifies the local source to define global schema. It provide object oriented schema to interprets object queries and execution plans for sending piece of queries to data servers are created. Queries are expressed in SQL like query language It interpreted the objects like image, video, audio and text files.

4. COMPARISON OF EXISTING METHODOLOGIES

The following table summarizes the different features of different projects are given in the table 3.

S.No	Projects	Source	Architecture type	Ontology Type	Language	Query Processing	Optimization
1	SIMS	HTML pages	Wrapper/Mediator	Single Ontology	Loom Language	Query transformation	Semantic query optimization
2	MOMIS	Structured and unstructured	Global Schema Builder	Global Virtual Schema	Description Logic	*SIM,*SLIM and ARTEMIS	Semantic optimization
3	OBSERVER	Repository	Wrapper/ontology Server	Multiple Ontology	Description Logic on classic or Loom	Query Construction	Division of sub queries may or may not loss information
4	TSIMMIS	Structure and semi structured	Wrapper/Mediator	Global Ontology	*OEM-QL	Wrapper/Mediator	Translation of query
5	DOME	Structure and semi structured	Ontology Server/ Mapping server	Multiple/ shared ontology	Description Logic on classic	Query Engine	Query Optimization
6	KRAFT	Norman Database	Wrapper/Mediator/ Facilitator user agent	Hybrid	Description Logic on classic	Query Facilitation	No Mechanism
7	Carnot and Infosleuth	Repository	Broker agent/Resource agent	Infosleuth: Multiple Carnot: Single	Description Logic on classic	Query Graph	Data flow model execution
8	COIN	Semi structured	Wrapper/context mediator	Hybrid	Object Oriented data mode F-logic	Query Rewriting	Semantic query optimizer

Table: 3 Comparison of existing approaches

*OEM-Object Exchange Model, *SIM- Source Integrator Module, *SLIM- *Schemata Lessical Integrator Module*

5. ONTOLOGY BASED DATA INTEGRATION USING CACHE

Ontology based data integration combine with cache provides improved access latency. The resultant data are cached and necessary cache optimization techniques are performed to enhance the access speed. There are various cache optimization techniques which work efficient in various situations.

6. CACHE OPTIMIZATION ALGORITHM

Caching and perfetcting are the most popular techniques that play a key role in [9] improving the performance by keeping object that are likely to be visited in the near future. As cache size is limited, a cache replacement policy is needed to handle the cache content. There are some of the cache replacement policies and they are classified based on certain factors. The important factors are given below in the table 4

a. Recency: time of (since) the last reference to the the object.
b. Frequency: number of requests to an object.
c. Size: size of the object.
d. Cost of fetching the object: cost to fetch an object.

e. Function based: Access latency of object.

S.No	Types	Description	Algorithm
1	Recency based	Recently used object as primary factor	LRU
2	Frequency based	Frequent count as primary factor	LFU
3	Size based	Object size as primary factor	SIZE
4	Function based	It associate each object in the cache with a utility value.	GD-SIZE

Table: 4 Types of cache algorithms Strategies

7. CACHE REPLACEMENT ALGORITHMS

Cache replacement algorithms play an extremely important role in caching. There are some of the algorithms that are based on above factors. They are

7.1. BELADY'S ALGORITHM

Belady's algorithm is one of the most efficient algorithm would be always discard the information that will not be needed in future. It also has lowest page fault rate because it simply replace the page that are not used for long time. It is popularly known as optimal page replacement algorithm as it always yields optimal result. However it is impossible to predict how long the future information is needed, this causes generally not implemented in practice.

7.2 LRU (LEAST RECENTLY USED)

Least recently used page replacement used algorithm is simple and easy to use. LRU follow the strategy of rejecting the least recently used object first. It also keep the path of what was used when the substance are used. LRU works on time stamp. It remove recently used object that are not used for longest time when the cache exceeds its maximum size and put a new object in place of evicted object. If the cache is not full then it will insert the object in cache.

7.3 LFU (LEAST FREQUENTLY USED)

LFU algorithm removes the objects which are least frequently used and put a new request page in free space. In this, it maintains a counter which will count the frequency of object. Based on the counter value the frequency of each object is determined. The page with least frequency value is replaced first by new incoming page when the cache exceeds its maximum size.

7.4. MRU (MOST RECENTLY USED)

MRU removes the most recently used object from the cache. This algorithm is used where the access of historical information are takes place. It work similar to LRU but it replace the object which has highest frequency.

7.5 LVC (LOWEST RELATIVE VALUE)

As the name suggest that lowest relative object value is first removed from the cache. Relative cost value is calculated for each object. The object with lowest relative value is removed first.

7.6 GREEDY DUAL-SIZE (GD SIZE)

Greedy-Dual algorithm was introduced by Young .It is the general version of LRU. This algorithm concerned when the equal size of object and each object has different cost to fetch. The algorithm associates a value, H, with each cached object p. Initially, when an object is brought to cache, H is defined to be a cost to bring the object into the cache (note that the cost is non-negative). When a replacement needs to be made, the object with the lowest H value, MIN_H, is replaced, and then all the objects reduce the cost values H by MIN_H. If an object p is accessed again its current cost value H is restored to the original cost of bringing this object to the cache.

7.7. SIZE

This strategy tries to minimize a miss ratio by replacing one large object rather than a bunch of small ones. However, some of the small object brought to a cache may never be accessed again. The SIZE strategy does not provide any mechanism to evict such object, which leads to pollution of the cache.

7.8 ARC (ADAPTIVE REPLACEMENT CACHE)

Megiddo and Modha introduced the ARC which has some similarity to 2Q algorithm. ARC gives ghost list to dynamically adapt the size depending on the workload. ARC keeps in track of both commonly used and lately used pages along with some of past data. It maintains two lists which follow LRU and LFU algorithm.

7.9. LRU-MIN

LRU-MIN algorithm minimize the replacement in LRU.LRU-MIN keeps the object which are smaller in size. If any objects with size S in cache, then it follow the LRU algorithm to evict least recently used object from the cache. If there is no object which is having size S in cache, then this algorithm evict the object of size S/2 in LRU order.

7.10 CLOCK

In CLOCK, all page frames are visualized to be arranged in form of a circular list that resembles a clock. The hand of the clock is used to point to the oldest page in the cache. CLOCK algorithm decreases the overhead cost.

7.12 CAR (CLOCK WITH ADAPTIVE REPLACEMENT)

CAR attempts to merge the adaptive policy of ARC with the implementation efficiency of CLOCK. The algorithm maintains four doubly linked lists C1, C2, L1, and L2. C1 and C2 are CLOCKs while L1 and L2 are simple LRU lists. The size of C1 roughly equals L2 and size of C2 roughly equals L1. It determines both recency and frequency of object.

8. CLASSIFICATION OF DIFFERENT CACHE REPLACEMENT ALGORITHM

The following table summarizes the different features of cache replacement algorithms are given in table 5.

S.No	Cache Replacement algorithms	Description	Advantage	Disadvantage
1	Belady's Algorithm	Discard information that are not used in future	Remove information that is not used longer.	Not suitable for practical implementation.
2	LRU	Least recently objects are removed first.	Logic is easy to understand, performs better as the cache size increases.	Need larger cache to increase efficiency.
3	LFU	Least frequently objects are removed first.	Lower runtime complexity	Large storage of object.
4	MRU	Most recently objects are removed first.	Simple to implement. Suitable for particular application.	Low access latency
5	GD-SIZE	Equal size of object and each object has different cost to fetch.	Removal of object which are no longer requested	Not taken account of previous frequency of object.
6	SIZE	Replacing large object by smaller object.	Keep small object in cache	Never used small objects are also stored.
7	ARC	Maintains to list for calculating LRU and LFU.	Lower overhead, self-tuning.	Two list need to maintain.
8	CLOCK	Clock hand point to oldest page in cache.	very low overhead on cache, removes serialization problem.	Does not take care of temporal filtering.
10	CAR	Combine both clock and adaptive algorithms.	Adaptively and dynamically captures the "recency" and "frequency" features of a workload.	Difficult to implement.

Table: 5 Classification of different cache replacement algorithms

9. DATA STRUCTUREUSE IN CACHE REPLACEMENT ALGORITHMS

There are different data structures used to represent the nodes in trie search. The choice of data structure depends on cache characteristics as well as the fan out of the node. Different data structures are used in different implementation of nodes in trie. Three different types of data structures used commonly are

1) Partitioned array
2) B-tree
3) Hash table
4) Queue

9.1 HASH TABLE

Hash table (hash map) is a data structure used to implement an associative array, a structure that can map keys to values. A hash table uses a hash function to compute an index into an array of buckets or slots, from which the correct value can be found. A good hash function and implementation algorithm is essential for good hash table performance. The basic requirement is that the function should provide a uniform distribution of hash values. The Perfect hash function can be used to create a perfect hash table that has no collisions. The running time complexity of hash table is given in the below in the table 6

	Average	Worst case
Search	O(1)	O(n)
Insert	O(1)	O(n)
Delete	O(1)	O(n)

Table 6 Running time of hash table

10. CONCLSION

Integrating existing databases is certainly not an easy task. Still, it is something that enterprises probably cannot avoid if they want to launch new applications or to reorganize the existing information system. In this paper we presented a comparison of several systems that use ontologies to solve the problem involved in data integration. It also discussed various famous cache replacement policies like LRU, LFU, FIFO, ARC, and CAR and so on. It also explains the variant characteristics of different algorithms, which helps to characterize the behavior and development of new cache page replacement techniques in future development.

REFERENCE

1. Rodrigo Fernandes Calhau, Ricardo de Almedia Falbo (2010), 'An Ontology-based Approach for Semantic Integration', IEEE International Enterprise Distributed Object Computing Conference, EDOC 2010.
2. Hajmoosaei, A., & Abdul-Kareem, S. (2007, December), " An ontology-based approach for resolving semantic schema conflicts in the extraction and integration of query-based information from heterogeneous web data sources", In Proceedings of the Third Australasian Workshop on Advances in Ontologies ,Volume 85 (pp. 35-43).
3. Wache, H., Voegele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., & Hübner, S. (2001, august)," Ontology- based integration of information-a survey of existing approaches", In IJCAI-01 workshop: Ontologies and information sharing (vol. 2001, pp. 108-117).(Wache H et al 2001)
4. Preece, A., Hui, K. Y., Gray, A., Marti, P., Bench-Capon, T., Jones, D., & Cui, Z. (2000). The KRAFT architecture for knowledge fusion and transformation.Knowledge-Based Systems, *13*(2), 113-120.
5. Bayardo Jr, R. J., Bohrer, W., Brice, R., Cichocki, A., Fowler, J., Helal, A., & Woelk, D. (1997, June). InfoSleuth: agent-based semantic integration of information in open and dynamic environments. In *ACM* SIGMOD Record (Vol. 26, No. 2, pp. 195-206). ACM.
6. Arens, Y., Chee, C. Y., Hsu, C. N., & Knoblock, C. A. (1993). Retrieving and integrating data from multiple information sources. International Journal of Intelligent and Cooperative Information Systems, 2(02), 127-158.
7. Beneventano, D., Bergamaschi, S., Castano, S., Corni, A., Guidetti, R., Malvezzi, G.,& Vincini, M. (2000, September). Information integration: the MOMIS project demonstration. In *VLDB* (pp. 611-614).
8. Mena, E., Illarramendi, A., Kashyap, V., & Sheth, A. P. (2000). OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. Distributed and parallel Databases, 8(2), 223-271.
9. Amit S. Chavan, Kartik R. Nayak, Keval D. Vora, Manish D. Purohit and Pramila M. Chawan," A Comparison of Page Replacement Algorithms", IACSIT , Vol.3, No.2, April 2011, pp (171-174). (Amit S et al 2011)
10. Bansal, S., & Modha, D. S. (2004, March). CAR: Clock with Adaptive Replacement. In *FAST*, Vol. 4, pp. 187-200.
11. Megiddo, N., & Modha, D. S. (2004). Outperforming LRU with an adaptive replacement cache algorithm. Computer, *37*(4), 58-65.
12. Kavar, C. C., & Parmar, S. S. (2013, February) , "Performance Analysis of LRU Page Replacement Algorithm with Reference to different Data Structure", International Journal of Engineering Research and Applications (IJERA), Vol. 3, Issue 1, January - February2013,pp(2070-2076).