The conventional relational database management systems (RDBMSs) are not flexible enough to support vague retrieval capabilities directly. That means when available data in a relational database do not match a user’s query precisely, the conventional RDBMSs will only return an empty result set to the user. This limits their applicability to domains where only crisp answers are meaningful. For many other application domains in the commercial world nowadays, however, the users also expect not only crisp results returned but also some other results close to the query in a certain sense. This capability of the applications is particularly very important as well for modern information retrieval systems in the digital age. In this paper, we discuss the lack and the need of supporting vague retrieval capabilities in the existing RDBMSs, introduce some previous prominent researches related to this topic in both the conventional RDBMSs and modern information retrieval systems; and eventually, we will present our most recent achievements in this field as well as introducing topics of interest for future researches.

1 Introduction

The conventional RDBMSs have been developed for a long time and database technology is one of the major contributions of computer science to the commercial world. The relational database model has been extremely successful in satisfying the requirements of most applications of the past. However, databases are facing new challenges evolve in step with the fast developments in computer science. One of those emerging challenges in the commercial world nowadays is how to process user’s queries not only efficiently but also flexibly. The traditional query processing model in the conventional RDBMSs, which return a result set matches a user’s query exactly, is insufficient and significantly inflexible. Nevertheless, almost existing RDBMSs still do not support

1 FAW-Institute, Johannes Kepler University of Linz, Austria, E-mail: {khanh, jkueng, rwagner}@faw.uni-linz.ac.at
vague retrieval capabilities directly. That means when available data in a relational database do not match a user’s query precisely, the system will only return an empty result set to the user. This limits their applicability to domains where only crisp answers are meaningful. In many other application domains, however, the users also expect not only the crisp results returned but also some other results that are relevant or close to the query in a certain sense. Such applications usually appear in domains as image/multimedia processing [32, 34, 6], CAD/CAM systems [2], geographical information systems (GIS) and tourist information systems [35], time-series databases [18], digital libraries [1, 22], electronic commerce and so on. A very simple example for tourist information systems, when a tourist is looking for a hotel with a rent price at 100 EUR a day and it would be located *in the city center*, he will fail to find such a hotel by means of the conventional RDBMS if the city center does not have any hotel rented at that price. In fact, the user might accept a hotel *near the city center* and the rent price can also be *a little lower or higher* 100 EUR per day.

A flexible system should solve this problem efficiently. We call a database system that supports this aspect a *flexible query answering system (FQAS)*. Specially, in the e-commerce systems nowadays, FQASs become more and more important because the customers need not touch the goods in advance (e.g. cars, clothes, etc.) but they see goods information by using computers. If the system does not directly support the vague retrieval capabilities, its users are forced to retry a particular query repeatedly with minor modifications until they get the satisfactory data. If the users do not know any alternative modifications to retry their queries, then this solution is infeasible [31]. Otherwise, even though the data items which the user really wanted cannot be found on the list of outputs, the output data items still provide the user with enough information to make the next trial effective [25].

Building a FQAS for an existing database system is totally not a simple work because there are many data types existing in the database that need to be managed and the FQAS must not conflict with the present functions of the system. There are some extensions proposed that facilitate the vague retrieval capabilities for the conventional RDBMSs such as ARES [25], VAGUE [31], VQS [27], etc. as well as for modern Information Retrieval (IR) systems as presented in [1]. A common problem of all these systems is to bear additional costs during the search process to gain the vague retrieval capabilities. The costs consist of both CPU-cost and IO-cost. Obviously, if the system must sustain an expensive additional cost then its performance will be decreased significantly. Both ARES and VAGUE have neglected this problem. Some of modern IR systems address the problem quite successfully, e.g. multimedia systems as QBIC [32], Garlic [8, 15], MARS [24, 34], CHITRA [33], [20, 5, 6], etc. The VQS has dealt with this problem to improve the search performance by an
incremental hyper-cube approach for finding the best match record for complex vague queries [29]. Here we define a complex vague query (CVQ) as a multi-feature query. This means that to answer a CVQ the system must search on some feature spaces. Unfortunately, this proposed approach is not general and has weaknesses lead to degenerate the search performance of the system. However, our recent research results have solved this problem efficiently and make the VQS become a full-fledged FQAS. In this paper we will discuss the importance of supporting the vague retrieval capabilities in the existing databases, introduce some previous researches related to this topic in both the conventional RDBMSs and modern IR systems; and eventually, we will present our most recent achievements in this field.

Concretely, the rest of the paper is organized as follows. Section 2 reviews some FQASs proposed in different domains as ARES, VAGUE, QBIC, Garlic, MARS, etc. For each of these systems, we will especially elaborate on what their most important features are to make them work efficiently on an existing database system and their viewpoint about the users’ information/data needs as well as how they define and create the relevant information to a given query. Section 3 is dedicated to introducing our new research results in a research project, called Vague Searches in Information Systems (VASIS) [16]. These achievements relate to multidimensional indexing structures and multi-attribute query processing [9, 10, 11], solving and optimizing complex vague queries in a semantic-based vague search system to get the relevant information efficiently [12, 13, 14]. Section 4 presents conclusions as well as some “hot” topics for future researches.

2 Flexible Query Answering Systems: An Overview

In our context, the flexibility can be interpreted as capabilities that provide easy, informative and intuitive access to data for every type of need. This section summarizes achievements of previous researches into database management systems and modern IR systems in support of the vague retrieval capabilities. As we know, with the increasing production and exchange of multimedia information through the Internet, the need of effective IR systems is a crucial issue nowadays [4]. The main aim of an IR system is to identify the information needs of a user. This is contrary to database management systems, which mainly aim at providing the data needs for the users. The database management systems are established on the use of a data model (relational or object-oriented) so as to express the data of the concerned universe by a given database. Afterwards, the data are exploited as they are stored in the system. IR aims at modeling, designing and implementing systems able to provide efficient and effective content-based access to a large amount
of information. This information is extracted from the data objects, e.g. images. The difference between systems in support of the flexible retrieval capabilities is the way to understand and process the users’ queries. In other words, it is their viewpoint about the users’ information/data needs as well as how they define and create the relevant information/data to a user’s query.

By this perspective, in the following sections, we introduce some most prominent systems and approaches to the discussing topic. They are classified into two main categories: (1) Supporting flexible retrieval capabilities in RDBMSs\(^2\) and (2) modern flexible IR systems.

## 2.1 Supporting Flexible Retrieval Capabilities in RDBMSs

### 2.1.1 ARES

The ARES (Associative Information Retrieval System), which was introduced by Ichikawa et al. [25], extends the relational data model with the so-called similarity relations in order to make capability of performing a flexible interpretation of queries. Each similarity relation stores the similarity between attribute values in a domain. Figure 1b illustrates a part of an example similarity relation for field job of a table Employee, in which the attribute values in the domain job in pairs are associated with a certain similarity degree (the smaller value means two jobs are more similar).

![Employee Table](image)

**Figure 1. The ARES’ similarity relations**

Depending on such similarity relations and a new operator “similar to”, which implies “approximately equal to”, the ARES can work on top of the conventional database systems to interpret user’s queries flexibly when they expect. However, this approach depending on similarity relations has some disadvantages. Each similarity relation needs \(n^2\) entries with respect to \(n\)

---

\(^2\) In section 2.1 of this contribution we only focus on the conventional RDBMSs, a similar research for the Object-Oriented DBMSs is also an aim of our future work.
different attribute values in the corresponding conventional relation and thus it leads to high storage cost. Maintenance cost of similarity relations is also high because when a new attribute value is added, $2n+1$ additional entries are necessary for the corresponding similarity relation [12]. Moreover, the ARES does not allow defining the similarities for infinite domains, for example the similarity between two given strings, because the similarities can only be defined by means of tables. The extensions of the relational data model in the ARES are unnecessarily complex [31]. Altogether, the ARES is a notable idea but the flaws have limited its usefulness.

2.1.2 VAGUE

The VAGUE system that was introduced by Motro [31] resembles the ARES in overall goals. In fact, it is an extension to the relational data model with data metrics and the standard query language with a comparator “similar-to”. There are four types of data metrics in VAGUE and each of them is a definition of distance between values in the same domain. In addition, VAGUE also allows multiple metrics over each domain with ability to select the appropriate metric for each query. Motro also mentioned incomplete information problem and proposed ways to deal with it in the VAGUE system. Although VAGUE is a useful system, its design represents a compromise between the conflicting requirements for simplicity, flexibility and efficiency. For examples, the users of VAGUE cannot provide their own similarity thresholds for each vague qualification but when a vague query does not match any data, VAGUE increases double all related searching radii simultaneously and thus its search performance could be considerably deteriorated. In the other words, the performance problem in the VAGUE system has been neglected.

2.1.3 VQS

Depending on based ideas of the ARES and VAGUE, Kueng et al. has introduced the Vague Query System (VQS) [27] as an extension to a RDBMS in order to facilitate it with vague retrieval capabilities. Main difference between the VQS and the previous proposed approaches is method to model and deal with the problem. The VQS has not considered the similarity between tuples within a database relation by means of the pre-computed similarity values (e.g. the similarity relations in the ARES or tabular metric in the VAGUE) or only direct distance computations over the attribute values of each tuple, e.g. computational and referential metrics in the VAGUE, but rather than it employs numerical semantic information of the attributes. The VQS key feature is concept of Numeric-Coordinate-Representation-Tables (NCR-Tables) that keep semantic background information of the attributes of a query relation/view. Concretely, non-numeric attributes are
mapped into the Euclidean spaces and stored in the NCR-Tables, in which the attribute itself is the key (\textit{NCR-Key}) and the \textit{NCR-Columns} represent the space dimensions. Actually the NCR-Tables are also the conventional database relations and the mapped attributes are called \textit{Fuzzy Fields} [27]. Numeric attributes need no semantic background information; the computations and normalization are carried out over the domain itself. In addition, the VQS also supports multiple NCR-Tables over a single attribute. Figure 2 shows an example NCR-Table for color names.

<table>
<thead>
<tr>
<th>Colors</th>
<th>Name</th>
<th>Red</th>
<th>Green</th>
<th>Blue</th>
</tr>
</thead>
<tbody>
<tr>
<td>black</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>blue</td>
<td>0</td>
<td>0</td>
<td>255</td>
<td></td>
</tr>
<tr>
<td>blue violet</td>
<td>138</td>
<td>43</td>
<td>226</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>white</td>
<td>255</td>
<td>255</td>
<td>255</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2. An example NCR-Table for color names w.r.t their RGB values

With support of the NCR-Tables, the VQS can realize similarity search capabilities and return the best match for both simple and complex queries [29]. Specially, the VQS approach carries characteristics of a modern IR system as discussed in the section 2.2: the NCR-Tables play role as feature vector spaces in the IR systems. To deal with non-numeric attributes that are not expressed by semantic background information, the VQS, as the VAGUE, also defines a standard metric. The standard metric will return 0 if two attribute values match precisely, otherwise it returns 1. Even then, this system still has many shortcomings that have just been recently overcome. More discussions are deferred until section 3.

2.2 Modern Flexible Information Retrieval Systems

Although researches in IR include a numerous number of areas related to text databases, multimedia systems, digital libraries, etc., we introduce and discuss in this section only some most notable projects. More information can be found in the literature as [1], [4], etc.

2.2.1 QBIC

QBIC (Query By Image Content) is the first commercial content-based image retrieval system, which has been developed by \textit{IBM Almaden Research Center} [32]. QBIC can search for similar images to one that is supplied by a user. To achieve this functionality, QBIC has two main components: database population and database query. In the former process, images and videos are processed to extract features that described their content as colors, textures, shapes, location of an
object etc., and the features are stored in a database. In the latter process, the user can pose a query using a graphical user interface. Features are then extracted from the query and input to a matching engine to find images or videos from the database with similar features. Potential applications of the QBIC include medical, photojournalism, and many others in art, fashion, cataloging, retailing and industry [1]. There are also a lot of technical aspects related to the QBIC system, but due to the space limitation we do not detail them here. More information is referred to the QBIC home page: http://www.qbic.almaden.ibm.com/ and other literature such as [30], [19], [1], etc.

2.2.2 Garlic

In fact, Garlic [8] is an object-oriented multimedia middleware query system. It allows existing data management components, such as a RDBMS, a full text search engine or an image retrieval system, e.g. the QBIC system, to be integrated into an extensible information management system. Garlic provides their applications with a common interface in order to access any of the data in the underlying data sources and the applications themselves can exploit the special query capabilities of those sources. During developing Garlic, there are a lot of query optimization and processing and IR techniques to be introduced. In particular, in [15] the author presented a notable algorithm for Combining Fuzzy Information from Multiple Systems, which has been used and improved by many researches afterwards as [6, 33, 20, 17]. More discussion about the Garlic project is referred to [8] and its homepage: http://alme1.almaden.ibm.com/cs/garlic/homepage.html.

2.2.3 MARS

Having similar goals to the QBIC, MARS (Multimedia Analysis and Retrieval System) project was started [24]. MARS attempts to exploit the achievements of a large number of retrieval models in the IR literature for content-based retrieval over images. As in the QBIC, an image in MARS is represented as a collection of low-level image features as well as a manual text description of the image. With support of a graphical user interface, a user can construct a query by choosing certain images from the collection. This query is then interpreted as a Boolean expression over image features and a Boolean retrieval model (adapted for image retrieval) is employed to retrieve a set of images ranked based on their degree of match. To support a ranked retrieval for queries, MARS adopts an approach based on a variation of the Boolean model. This approach bases on probabilistic and fuzzy interpretations of distances between images and the query [34]. Moreover, MARS allows its users to adjust the weights associated with the atomic features according to their own emphasis
as in the Virage system [3]. There are also a numerous number of related publications, which are referred to the MARS homepage: http://www-db.ics.uci.edu/pages/research/mars/index.shtml.

2.3 Discussion

There are a vast amount of other proposals, approaches and systems that have been introduced for supporting the vague retrieval capabilities in the RDBMSs as well as the storage and retrieval of the information in modern IR systems, e.g. fuzzy databases, [36], [7], [21], etc. Although in this paper we just outline and overview some most prominent ones related to some areas, such FQASs are really an indispensable requirement for searching applications in the commercial world nowadays. Because of the needs, we have carried out a research project, called VASIS, at the FAW institute and obtained a considerably large number of achievements so far. Section 3 briefly surveys the VASIS project and its achieved results.

3 VASIS Project – Vague Searches in Information Systems

3.1 Motivation and Objectives

The VASIS project has been started from 1998 [16]. Its implemented prototype is the VQS (cf. section 2.1.3). The main purpose of the VASIS project is obvious and intuitive: We want to deal with possibilities for the modeling of semantic metadata and similarity searches in the conventional RDBMSs. We have realized that building a FQAS like the VQS needs to first solve two essential problems as follows: (1) Efficient management of semantic metadata, which are called semantic background information in the VQS; (2) make use of these metadata to facilitate the RDBMSs with the vague retrieval capabilities efficiently and effectively. Researching into integration capability of these theoretic results with a commercial RDBMS should be followed. With this perspective, we have intensively carried out our researches into multidimensional access methods, similarity search algorithms, complex vague queries processing and optimization as well as approximate queries answering. In section 3.2 we present our promising results with respect to these topics.

3.2 Achievements

3.2.1 Multidimensional Access Methods (MAMs) and Similarity Search Algorithms

Designing an efficient MAM will contribute to the efficient management of semantic metadata [9] because semantic background information of the attributes is in common multidimensional data.
Our research in this topic has resulted in the SH-tree, a super hybrid index structure for multidimensional data [10]. The SH-tree is a flexible, well-combined structure of both space partitioning-based and data partitioning-based techniques. The tree operations in the SH-tree are similar to the R-tree family but there are many modifications to efficiently adapt them for the new structure. We also introduced a new concept for the SH-tree called extended balanced tree (EBT). It implies that the SH-trees are unnecessary to be exactly balanced, but the search performance is still not deteriorated and the maintenance cost for the tree balance is reduced significantly.

In [11] we presented two adapted algorithms, which are originated from the state-of-the-art research results, for the SH-tree to efficiently process k-nearest neighbor (k-NN) queries in spatial/multidimensional databases. The experiments were conducted on both uniformly distributed and real data sets. The results have shown that the SH-tree with these adapted algorithms efficiently processes k-NN queries and outperforms the SR-tree [26] by orders of magnitude in all the experiments. Our experimental evaluations also confirm conclusions in the previous researches with respect to the optimality in terms of the IO-cost of the original algorithm presented in [23]. Besides, these results also prove the correctness of our theoretical analyses in [10]: the SH-tree can efficiently scale to high dimensional spatial databases.

3.2.2 Complex Vague Queries Processing and Optimization

In [29] Kueng et al introduced an Incremental hyper-Cube Approach (ICA) for finding the nearest record/tuple for CVQs in the VQS. Again, we define a multi-feature vague query as a CVQ. This indicates that to answer a CVQ, the system must search on some feature spaces of the mapped attributes and then combine these search results to return the ranked ones to the user. Nevertheless, the ICA is not general and has weaknesses lead to degenerate the search performance of the system. In [12] we introduced a new approach called Incremental hyper-Sphere Approach (ISA), which was then improved in [13], that can be applied to the VQS system for solving CVQs more efficiently and generally. The ISA is superior to the ICA in both concepts of CPU-cost and IO-cost. Besides, we introduced an incremental algorithm adapted for range queries. It has been also proven to be an optimal algorithm in terms of the disk access number as the original one in [23]. This adapted algorithm considerably contributes towards accelerating search for the nearest neighbors of CVQs in the ISA. Moreover, the ISA is more general and practical than the ICA in whatever sense and can be applied to all information systems that the metadata has form likes the NCR-Tables. Specially, the ISA is also usable in systems that the mapping one-to-one between keys (Fuzzy Fields) in the
NCR-Tables does not exist. This is a notable advantage of the ISA over Fagin’s algorithm [15] and all its improvements [33, 20].

3.2.3 Approximate Queries Answering

Most recently, in [14] we introduce an approach called \(\varepsilon\)-ISA for efficiently solving the approximate nearest neighbor problem for CVQs, which are also called multi-feature nearest neighbor (M-FNN) queries. To the best of our knowledge, the \(\varepsilon\)-ISA is one of the vanguard solutions for dealing with this problem, i.e. approximate M-FNN queries. Although the \(\varepsilon\)-ISA has been presented in the context of the VQS, it can also be applied to a variety of application domains such as multimedia databases, GIS and tourist information systems, data mining and so on. In general the \(\varepsilon\)-ISA is very useful for application domains that the returned results need not to be exact but similar or approximate similar (with a certain tolerant error) to a given query. The experimental results have proven this. With a suitable tolerant error value \(\varepsilon\), the \(\varepsilon\)-ISA can save a very high percentage of the costs including both IO-cost and CPU-cost while it still preserves the accuracy of the returned results at a particularly high value. Specially, the \(\varepsilon\)-ISA, as the ISA, is also applicable to general feature spaces including ones that do not satisfy three conditions, which are pointed out in [38], to make Fagin’s algorithm [15] and all its variants as [33, 20] become utilizable.

3.3 Discussion

As mentioned above, our final goal is to facilitate the RDBMSs with the vague retrieval capabilities. Therefore, integration capability of these theoretic research results with a commercial RDBMS must be engaged in the research work. Our future work will concentrate on deploying these theoretic achievements for the real world applications as GIS and tourist information systems [35, 37] as well as some other applications in the modern IR area. Besides, developing other new theoretic concepts, e.g. complex join operation [28], is also the research work of great interest.

4 Conclusions and Future Work

This contribution discussed an important issue in the commercial world and the digital age nowadays: supporting flexible query answering capabilities in the RDBMSs as well as modern IR systems. We presented the indispensable importance of the problem and briefly overviewed some previous prominent approaches proposed for dealing with it. Specially, we also introduced our most
recent achievements in this field resulting from a research project VASIS – Vague Searches in Information Systems. There are some related research directions being the “hot” discussing topics. They include flexible and efficient web searching supporting, building efficient and flexible meta-search engines for the World Wide Web, searching in digital libraries, etc. Specially, in these fields, we need to develop general and extensible FQASs that can be applied to both homogeneous and heterogeneous environments of a lot of data types in different databases.

References

[16] The VASIS project at FAW institute: http://www.faw.uni-linz.ac.at/cgi-pub/e_showprojekt.pl?projektnr=10