Hand-OLAP: Semantics-aware Compression of Data Cubes for Effective and Efficient OLAP in Mobile Environments

Alfredo Cuzzocrea*, Domenico Saccà*

*ICAR-CNR and University of Calabria
Italy
cuzzocrea@si.deis.unical.it
*DEIS Dept., University of Calabria
Italy
sacca@deis.unical.it

Abstract—In this paper, we present a complete demonstration of Hand-OLAP, a Java-based distributed system that relies on intelligent data cube compression techniques for effectively and efficiently supporting OLAP in mobile environments. Hand-OLAP is based on an innovative systematic technique according to which first a two-dimensional OLAP view of interest is extracted from the target multidimensional data cube via the so-called OLAP dimension flattening process, and then this view is compressed by means of a meaningful semantics-based data cube compression approach. The compressed two-dimensional view is finally delivered to mobile devices, and used to support interactive OLAP exploration and querying tasks in an off-line manner.

I. INTRODUCTION

Data Warehousing (DW) systems offer integrated and consolidated views over heterogeneous repositories of business data in the form of multidimensional data cubes [8] that store SQL-based aggregations (e.g., SUM, COUNT) made available to decision makers via meaningful OLAP [4] tools. OLAP tools effectively and efficiently support advanced decision making processes via intuitive abstractions that are founded on a successful multidimensional and multi-resolution vision of data. This particular feature represents a critical add-in value in Business Intelligence (BI) scenarios, as it does not impose decision makers to have particular skills in Information Technology, thanks to nice and intuitive conceptual constructs like measures and dimensions, which are abstracted from the underlying data sources as attributes of interest and functional attributes for the target decision making process, respectively. Also, multi-level hierarchies, which are associated to dimensions, allow decision makers to meaningfully explore and mine huge amounts of multidimensional data on the basis of different-in-granularity perspectives of analysis. To give an example, sale data could be mined according to the granularity Month or Quarter, in an alternative or combined manner at the same, being Month and Quarter dimensional members of levels of the hierarchy Time. Finally, a wide family of powerful operators [4,8] (like drill-down, which allows us to increase the level of detail of multidimensional data while decreasing their level of abstraction, and roll-up, the opposite one) and query classes (e.g., range queries [9]) meaningfully completes the capabilities of OLAP tools in extracting summarized knowledge from huge amounts of multidimensional data by unquestionably overcoming the performance of conventional OLTP tools based on traditional SQL interfaces [5].

Despite the nice and well-understood characteristics of OLAP tools outlined above, the so-called curse of dimensionality problem [10], i.e. enormous size and very-high degree of multidimensionality of typical real-life data cubes, causes a natural disorientation in decision makers that, contrary to this, are instead very often interested in exploring and mining specific multidimensional ranges of the target data cube, depending on particular BI analysis goals, and more and more neglect to access and query a global view of the data cube. Two leading aspects clearly follow from this breaking evidence. First, data cube compression techniques [2], which aim at obtaining compressed representations of multidimensional data cubes in order to retrieve fast approximate answers to time-consuming OLAP queries, obviously allow us to achieve faster and more efficient data cube computation and query evaluation strategies at a cost of an approximation degree that is reasonably regarded as tolerable for OLAP applications, where decimal precision is not required (e.g., [5]). Second, these techniques are unfortunately still not enough to effectively deal with challenges posed by exploring, querying and mining very large and high-dimensional data cubes, as they indeed deliberately neglect the semantics exposed by data cube models and schemas, and try to adopt an “algorithmic” vision of the intrinsic data cube compression problem according to which a data cube is just viewed as a “bulk of data cells” [7].

These drawbacks get even worse when data cubes are processed, analyzed and mined in a mobile environment, where limitations of mobile devices [1], such as low computational capability, bounded memory capacity and limited visualization aptitudes, make traditional data cube compression approaches completely unsuitable to the mobile OLAP goal. On the other hand, mobile environments are more and more becoming leading application scenarios for BI, due to their clear aptitude and complete suitability for emerging computational paradigms and metaphors, like pervasive and ubiquitous information systems [12], information and knowledge dissemination approaches [14], location-aware and context-aware data warehouse management [13], and so forth.

With the aim of enabling OLAP in mobile environments, in [7] we assert that the data cube compression process must be driven by semantics kept in data cubes, thus making this process more effective, beyond that more efficient, and, above all, overcoming the natural disorientation of (mobile) decision makers during interactive OLAP exploration, querying and mining tasks. According to our vision, while this approach reasonably improves the performance of OLAP tasks over data cubes in conventional wired architectures, it plays an even more critical role in mobile environments, thus eliciting here the role of enabling database technology. These motivations and guidelines of OLAP research have convinced us to author the system Hand-OLAP [6], a Java-based distributed system that relies on
intelligent data cube compression techniques for effectively and efficiently supporting OLAP in mobile environments (see Fig. 1). In this paper, we present a complete demonstration of Hand-OLAP that shows its effective and efficient capabilities in a typical BI mobile OLAP environment.

II. THE HAND-OLAP APPROACH

Hand-OLAP is based on an innovative semantics-aware histogram-based data cube compression technique [6] (see Fig. 2), which is finally codified in the so-called Hierarchy-driven Indexed Quad-Tree Summary (HI-QTS), a novel histogram for (mobile) OLAP data cubes. Histograms [2] are well-recognized and well-understood synopsis data structures for OLAP, which pursue the idea of obtaining data cube compression by first generating a bucket-based partitioned representation of the input data cube, and then computing a compressed representation of the generated partition such that each bucket stores an SQL-based aggregation computed on data stored in the range it refers.

Traditional histograms are indeed not suitable to mobile environments [6]. Therefore, in order to deal with data cube compression challenges posed by these novel environments, Hand-OLAP introduces an innovative double-step data cube compression approach according to which, in the first step, a two-dimensional OLAP view \( V \) is extracted from the target multidimensional data cube \( D \) by means of the so-called OLAP flattening process. This process basically projects \( D \) onto \( V \) via (i) selecting two dimensions, called Visualization Dimensions (VD), from the collection of dimensions of \( D \), and (ii) modifying the original hierarchies of the VD by meaningfully merging these hierarchies with hierarchies of the remaining dimensions of \( D \), thus obtaining the so-called specialized hierarchies. The OLAP dimension flattening process finally allows us to represent a multidimensional domain \( D \) equipped with hierarchies by means of a two-dimensional domain \( V \) whose hierarchies are composed by sub-trees of \( D \)’s hierarchies, combined in a hierarchical fashion. Then, \( V \) is re-aggregated according to the so-generated specialized hierarchies. Both the selection and the merge phase are driven by particular BI analysis goals pursued by decision makers. It is easy to understand how, since this early editing stage, decision makers naturally select “isolated” and semantically-correlated two-dimensional ranges in \( V \), targeted to their specific BI purposes.

In the second step of the Hand-OLAP data cube compression approach, the histogram HI-QTS is finally computed by taking as input the two-dimensional OLAP view \( V \) and the bounded storage space \( B \) available to house its in-memory representation on the mobile device. In more detail, HI-QTS is obtained via generating a quad-tree based partition of \( V \) storing sub-domains that “follow” groups defined in the (specialized) OLAP hierarchies of \( V \), i.e. according to the semantics these hierarchies expose. This allows us to finally originate in H-IQTS the so-called semantics-aware buckets, i.e. buckets computed over semantically-correlated aggregate data of \( V \), still being this semantics determined by specific BI purposes of decision makers. Clearly, semantics-aware buckets are capable of supporting OLAP exploration, querying and mining tasks with higher effectiveness and efficiency than conventional ones, as traditional histogram-based data cube compression techniques are meant to generate the underlying partitioned representation of the input data cube \( D \) in dependence on intensional properties of data stored in \( D \), such as the variance of multidimensional ranges or the skewness of final buckets [2], and completely neglect semantic issues.

Unfortunately, the latter general approach clearly fails in capturing and supporting a subject-oriented analysis that is indeed typical in OLAP-based decision making [4], as it is well-understood and recognized that OLAP data are intrinsically clustered and correlated. To become convinced of this limitation of traditional histogram-based data cube compression techniques, consider Fig. 3, where two alternative partitioned representations of an example two-dimensional OLAP data cube \( D \) (respectively, view) having \( \text{Dim}(D) = \{\text{City}, \text{Product}\} \) as dimension set and \( \text{Mes}(D) = \{\text{Sale}\} \) as measure set, respectively, are depicted. Particularly, \( D \) stores sale data about summer and winter products sold in European and American cities, respectively. Fig. 3(a) shows an arbitrary quad-tree based partition of \( D \) that has been generated by splitting each dimension of \( D \) in two equal-sized halves, whereas Fig. 3(b) shows instead a semantics-aware quad-tree based partition of \( D \) that has been generated by splitting each dimension of \( D \) according to groups defined in the hierarchies of \( D \) (these hierarchies are not shown for space reasons, but they can be easily inferred from Fig. 3). From Fig. 3(a), it clearly follows that the arbitrary partition stores buckets computed on top of aggregate data that are not semantically-correlated, playing this aspect the role of a restrictive drawback for OLAP-based decision making. For instance, bucket (EU Cities, Summer Products) stores aggregate data related to the American city Chicago and the winter product Raincoat, which both do not have any semantic correlation with the remaining aggregate data stored in the same bucket. Contrary to this, the semantics-aware partition shown in Fig. 3(b) stores buckets computed on semantically-correlated aggregate data only, thus nicely and effectively supporting subject-oriented OLAP-based decision making.

A specific characteristic of Hand-OLAP is the ability of handling data cubes having both regular OLAP hierarchies, which are usually
represented as conventional $n$-ary trees such that each node models a
dimensional member of the hierarchy at a certain level, as well as
more probing irregular OLAP hierarchies [11], whose most popular
instance is the one exposing a lattice-based organization. To face-off
such irregular OLAP hierarchies, in Hand-OLAP we make use of a
unifying approach according to which, given a lattice-based OLAP
hierarchy $H^\ell$, $H^\ell$ is transformed in the corresponding tree-based
OLAP hierarchy $H^t$ by means of the so-called hierarchy
transformation process [6]. Basically, this process aims at finding an
“equivalent” regular OLAP hierarchy starting from the irregular one
via inspecting the equivalence of classes the dimensional members of
respective hierarchies describe [6].

In order to achieve high-efficiency during (range) query
evaluation, leaf buckets of the quad-tree based partition codified in
H-IQTS are equipped with Indices [3], which are high-performance
synopsis data structures providing a succinct description of actual

Fig. 3  Arbitrary (a) and semantics-aware (b) partition of the
example City $\times$ Product OLAP data cube storing sale data

Hand-OLAP fully supports interactive exploration and querying
tasks over the compressed two-dimensional OLAP view in an off-
line manner, via high-performance query algorithms over H-IQTS.
These algorithms are able to deal with compressed two-dimensional
data materialized in H-IQTS, and retrieve approximate answers to
OLAP queries. In more detail, Hand-OLAP supports range queries [9]
over the compressed view. Range queries apply an SQL aggregation
operator to a sub-domain of the target view, and are widely
recognized as a very popular way of extracting summarized
knowledge from data cubes. It should be noted that these queries can
be also used as baseline operations to support meaningfully and
intuitive mining tasks over OLAP data cubes, according to
consolidated OLAM paradigms.

Fig. 4  The Hand-OLAP logical architecture

In order to achieve high-efficiency during (range) query
evaluation, leaf buckets of the quad-tree based partition codified in
H-IQTS are equipped with Indices [3], which are high-performance
synopsis data structures providing a succinct description of actual
distributions of buckets they are associated to. Indices can be stored
in few bytes, with limited dependency on the effective bucket size,
thanks to a meaningful summarization approach [3]. Basically,
Indices allow us to significantly improve the so-called intra-bucket
query estimation, which, as we prove in [3], could seriously decrease
the approximate query answering capabilities of general data cube
compression approaches like histograms and wavelets [2]. It should
be noted that, since Indices occupy in memory a limited amount of
storage space, they turn to be particularly suitable to the mobile
OLAP context, where size of synopsis data structures plays a critical
role. Finally, the query performance of Hand-OLAP has been
extensively proven by us in [7] against synthetic, benchmark and
real-life data cubes, and under several perspectives of experimental
analysis.

Fig. 5  The Hand-OLAP software architecture

III. THE HAND-OLAP SOFTWARE ARCHITECTURE

Fig. 4 shows the logical architecture of Hand-OLAP, where we
conceptually isolate each logical layer of the system. According to
well-known multi-layer software design patterns, each (logical)
component of the system corresponds to a specific application logic
of the overall knowledge management process underlying the system
itself, and it can be separately designed and developed by formally
specifying its (software) interface for component interaction
purposes. This design pattern provides meaningful separation,
maintenance and re-engineering functionalities. The logical layers of
Hand-OLAP are the following:

- **Data Sources Layer:** it models the collection of (i) OLAP
  servers from which the desired view can be retrieved, and (ii)
  wrappers that extract metadata on the available multimodal
dimensional databases and data cubes as well as the actual
multidimensional data.

- **Application Server Layer:** it models the layer that elaborates
  users’ requests via (i) interacting with OLAP servers, (ii)
extracting the two-dimensional OLAP view $V$ from the target
multidimensional data cube $D$ by means of the OLAP
dimension flattening process, (iii) computing the compressed
representation of $V$, and (iv) sending the compressed view to
the mobile device.

- **User’s Layer:** it comprises the client-side tool running on the
  mobile device that allows users to access and process the
  desired two-dimensional OLAP view, by also enabling useful
  functionalities such as connectivity services, browsing OLAP
  metadata, editing of the two-dimensional OLAP view, browsing
  and querying the compressed view, and so forth.

Fig. 5 shows the Hand-OLAP software architecture where the
logical components depicted in Fig. 4 are detailed and described in
their reference software platforms. In particular, the software
architecture of Fig. 5 shows the prototype of Hand-OLAP when
Analysis Services 2000 is adopted as OLAP server platform, and
iPAQ as mobile device, respectively. All the other software
components of the Hand-OLAP are completely independent by both the particular OLAP server and the particular mobile device software platform, respectively, meaning that they can be interfaced with any other OLAP server platform (provided that the specific OLAP wrapper is designed) and any other mobile device software platform (provided that its Application Programming Interface (API) can be interfaced with Java). Server-side and client-side components are based on Java2 Enterprise Edition (J2EE) and Java2 Micro Edition (J2ME), respectively, hence being completely deployable on top of any arbitrary data/software platform.

At the Data Sources Layer, in order to handle the proper format of multidimensional data exposed by the OLAP server platform Analysis Services 2000, we designed and developed a set of Win32 libraries based on the API ADOMD that are able to (i) extract the target two-dimensional OLAP view from the OLAP server via the multidimensional query language MultiDimensional eXpressions (MDX), (ii) write the output OLAP view in a set of XML files (according to a specific partitioned in-memory-representation), and finally (iii) send the XML-formatted OLAP view to the Java-based server-side components at the Application Server Layer by means of serialization. Here, the view can be parsed and processed easily, thanks to conventional Java/XML libraries (e.g., JDOM). All considering, this realizes a specific OLAP wrapper for Analysis Services 2000, and, above all, a “neutral” XML-based protocol that allows us to make the COM-compliant Analysis Services 2000 and the J2EE-compliant Application Server communicant. The same approach can be devised for any other OLAP server platform that cannot be interfaced with Java directly. By exploiting the client-side functionalities of Hand-OLAP, users are allowed to connect the OLAP server and access OLAP metadata on the server (e.g., exploring the collection of multidimensional databases and data cubes available in the server), as well as OLAP metadata on data cubes that are regarded as relevant for their specific decision making processes (see Fig. 6(a)). Having selected a target multidimensional data cube, users are allowed to define the two-dimensional OLAP view of interest by means of intuitive tools that hierarchically explore the available dimensions in order to progressively select specific semantically-correlated multidimensional ranges (see Fig. 6(b)), on the basis of particular BI analysis goals. The so-generated view definition is thus sent to the Application Server that is in charge of compressing the view and re-sending it to the mobile device. Finally, users are allowed to browse (see Fig. 7(a)) and query (see Fig. 7(b)) the compressed two-dimensional OLAP view in an off-line manner, yet being capable of refreshing the view when updates occur in the underlying data sources alimenting the data cubes of interest.

IV. DEMONSTRATION

In our demonstration, we show all the functionalities of Hand-OLAP throughout a complete case study running on the OLAP server platform Analysis Services 2000, and particularly focusing on the multidimensional data cube Sales built on top of the multidimensional database FoodMart 2000, which is released in bundle with the OLAP server platform. FoodMart 2000 stores information about an hypothetical company selling foods in a geographically-distributed area, and captures several typical sale entities like customers, employers, stores, warehouses, inventories, regions etc. Sales is a SUM-based 12-dimensional data cube storing 5 measures that fully supports OLAP analysis over sale data stored in FoodMart 2000 via simultaneously aggregating them according to the following conceptual OLAP dimensions: Time, Customer, Promotion, Product, and Store, which well describe a typical sale scenario.