Perspects in astrophysical databases

Marco Frailis[a] Alessandro De Angelis[b] Vito Roberto[c]

[a]*Dipartimento di Fisica, Università di Udine, via delle Scienze 208, 33100 Udine, Italy*

[b]*INFN, Sezione di Trieste, Gruppo Collegato di Udine, via delle Scienze 208, 33100 Udine, Italy*

[c]*Dipartimento di Matematica e Informatica, Università di Udine, via delle Scienze 208, 33100 Udine, Italy*

Abstract

Astrophysics has become a domain extremely rich of scientific data. Data mining tools are needed for information extraction from such large datasets. This asks for an approach to data management emphasizing the efficiency and simplicity of data access; efficiency is obtained using multidimensional access methods and simplicity is achieved by properly handling metadata. Moreover, clustering and classification techniques on large datasets pose additional requirements in terms of computation and memory scalability and interpretability of results. In this study we review some possible solutions.

Key words: Multidimensional Indexing, Data Mining, Astrophysical Databases, Data Warehousing
PACS:

1 Introduction

At present, astrophysics is a discipline in which the exponential growth and heterogeneity of data require the use of data mining techniques. The primary source of astronomical data are the systematic sky surveys over a wide energy range (from 10^{-7} eV to 10^{13} eV). Large archives and digital sky surveys with dimensions of 10^{12} bytes currently exist, while in the near future they will reach sizes of the order of 10^{15} bytes. Numerical simulations are also producing comparable volumes of information.

Several scientific research fields require to perform the analysis on multiple energy spectra and consequently to get the data from different missions. Therefore, the use of data mining techniques is necessary to maximize the information extraction from such a growing quantity of data. This task is hardened by different issues, like the heterogeneity of astronomical data, due in part to their high dimensionality including both spatial and temporal components, due in part to the multiplicity of instruments and projects, or the use of traditional operational systems, in which the emphasis is on data normalization, to organize astrophysical data. Data mining for multi-wavelength analysis necessitates using an informational system, or data warehouse, as a model for data management, a definition of a common set of metadata to guarantee the interoperability between different archives and a more efficient data exploration.

2 Towards a data whareouse

Most of the online resources available to the astrophysicists community are simple data archives containing observational parameters (detector, type of the observation, coordinates, astronomical object, exposure time, etc.). Many astronomical catalogs can be accessed online, but it is still difficult to correlate objects in different archives or access multiple catalogs simultaneously. Some advances, in this direction, have been accomplished by projects like Vizier, Aladin and SkyView [1,2].

With an ideal astrophysical database, the users should be able to perform queries based on scientific parameters (magnitude, redshift, spectral indexes, morphological type of galaxies, etc.), easily discover the object types contained into the archive and the available properties for each type, and define the set of objects which they are interested in by constraining the values of their scientific properties along with the desired level of detail [3].

The aforesaid requirements can be satisfied organizing data in a data warehouse. A data warehouse can be defined as a *subject-oriented*, *integrated*, *time varying* and *non-volatile* data collection [4]. In a data warehouse, data are arranged in a structure that can be easily explored and queried, with fewer tables and keys than the equivalent relational model. You start from a relational model, but some restrictions are introduced by using *facts*, *dimensions*, *hierarchies* and *measures* in a characteristic star structure called *star schema* [5]. The central table is called "fact" table and it is the highest dimensional table of the scheme. It can represent a particular phenomenon that we want to study. This table is surrounded by a number of tables, called "dimensions", which represent entities related to the phenomenon to be studied and connected to the central table, forming the ends of the star. Within the dimensions, attributes are arranged in hierarchies, determining the "drill-down" and "roll-up" operations available on each dimension: the result is a tree that the user can visit from the root to the leaves, refining his query (drill-down) or generalizing it (roll-up).

Metadata play an important role: a researcher has to obtain information about the environment in which data have been gathered, in order to understand the respondence to the project requirements, like date and/or data acquisition method, internal or external error estimates, aim of data. Computing systems have to access metadata to merge or compare data from different sources. For instance, it is necessary that units are expressed unambiguously to allow comparisons between data with different units.

The astrophysicists community, in addition to using the FITS (Flexible Image Transport System) exchange format, is currently considering alternatives like XML to improve the interoperability. Some attempts to define a common standard are XSIL (eXtensible Scientific Interchange Language), XDF (eXtensible Data Format) and VOTable [6].

3 Multidimensional access methods

In the Astroparticle and Astrophysical fields, data is mostly characterized by multidimensional arrays. For instance, in X-ray and Gamma-ray astronomy, the data gathered by detectors are lists of detected photons whose properties include position (RA, DEC), arrival time, energy, error measures both for the position and the energy estimates (dependent on the instrument response), quality measures of the events . Source catalogs, produced by the analysis of the raw data, are lists of point and extended sources characterized by coordinates, magnitude, spectral indexes, flux, etc.

This multidimensional (spatial) data tend to be large (sky maps can reach sizes of Terabytes) requiring the integration of the secondary storage, and there is no total ordering on spatial objects preserving spatial proximity [7]. This characteristic makes difficult to use traditional indexing methods, like B-trees or linear hashing.

Data mining applied to multidimenisonal data analyzes the relationships between the attributes of a multidimensional object stored into the database and the attributes of the neighboring ones. Typical queries required by this kind of analysis are: *point queries*, to find all objects overlapping the query point; *range queries*, to find all objects having at least one common point with a query window; *nearest neighbor queries*, to find all objects that have a minimum distance from the query object. Another important operation is the *spatial join*, needed to search multiple source catalogs and cross-identify sources from different wavebands. Some of the following indexing methods can be used to improve the queries efficiency.

HTM. Data gathered by all sky survays are distributed on an imaginary sphere. The HTM [8] indexing method maps triangular regions of the sphere to unique identifiers keeping a certain degree of locality. The technique for subdividing the sphere in spherical triangles is a recursive process. The starting point is a spherical octahedron which identifies 8 spherical triangles of equal size. In a recursion step, a triangle is further subdivided into 4 triangles by connecting the side midpoints. At each level of the recursion, the area of the resulting triangles is roughly the same and each triangle is uniquely identified by a 2 bit value. This method as been used to index the Sloan Digital Sky Survay, a catalog of 200 M objects in a multi-terabyte archive. A level-5 HTM index is used to partition the bulk data. A database for each level-5 leaf node of the HTM (defining the database file name) has been built. Each database, containing tuples in a 5-dimensional color space, is indexed by a KD-tree.

KD-tree and its variants. The KD-tree [9] is a binary tree that stores points of a k-dimensional space. In each internal node, the KD-tree divides the k-dimensional space into two parts with a $(k-1)$-dimensional hyperplane. The direction of the hyperplane, that is the dimension on which the division is performed, alternates between the k possibilities from one tree level to the following one. The subdivision process is recursive and terminates when the size of a node (its longer side) or the number of points contained into it is below a certain threshold. Given N data points, the average cost of an insertion operation is $O(\log_2 N)$. The tree structure and the resulting hierarchical division of the space depends on the *splitting rule*. A drawback of KD-trees is that they have to be completely contained into the main memory. With large datasets this is not feasible. KD-B-trees [10] and hB-trees [11] combine properties of KD-trees and B-trees to overcome this problem.

R-tree and its variants. The R-trees [12] are hierarchical dynamic data structures meant to efficiently index multidimensional objects with a spatial extent. They are used to store not the real objects but their minimum bounding box (MBB). Each node of the R-tree corresponds to a disk page. Similar to B-trees, the R-trees are balanced and they guarantee an efficient memory usage. Due to the overlapping between the MBBs of sibling nodes, in an R-tree a range query can require more than one search path to be traversed. Search performances depend on the insertion algorithms. Some variants have been proposed to improve the disjointness among regions: the R*-tree [13], which uses a new insertion policy, the SR-tree [14], which uses the intersection of bounding spheres and bounding rectangles to keep small the diameters and volumes of the regions, and the A-tree [15], which improves the fanout of the nodes using an approximation of the MBRs.

Usually, the analysis of astrophysical data is performed on a static dataset. In this case, an optimized index (in terms of memory and query performances) can be built using a priori information on the dataset. Several bulk loading techniques have been proposed in the literature. We have followed a top-down construction method called VAMSplit algorithm, described in [16], to build and optimized R-Tree. The main idea is to find a split strategy that minimizes the number of buckets used and provides a good query performance. This is achieved by recursively splitting the dataset on a near median element along the dimension with maximum variance. To adapt it to a large dataset, we had to implement an external selection algorithm. The implementation uses a sampling method suggested by [17] to find a good pivot value and reduce the number of I/O operations; a caching strategy explained in [18] has been adopted to partition the data into the secondary memory.

4 Clustering algorithms on large datasets

Clustering algorithms have to locate regions of interest in which to perform more detailed analysis and point out correlations between objects. An important issue, in large datasets, is the efficiency and scalability of the clustering algorithms with respect to the dataset size.

Many scalable algorithms have been proposed in the last ten years, including: BIRCH, CURE, CLIQUE [19].

In particular, BIRCH is a hierarchical clustering algorithm. The main idea behind the algorithm is to compress data into small subclusters and then to perform a standard partitional clustering on the subclusters. Each subcluster is represented by a *clustering feature* which is a triplet summarizing information about the group of data objects, that is the number of points contained into the cluster and the linear sum and the square sum of the data points. This algorithm has a linear cost with respect to the number of data points.

CURE is an hierarchical agglomerative algorithm. Instead of using a single centroid or object, it selects a fixed number of well-scattered objects to represent each cluster. The distance between two clusters is defined as the distance between the closest pair of representatives points and at each step of the algorithm, the two closest clusters are merged. The algorithm terminates when the desired number of clusters is obtained. To reduce the computational cost of the algorithm, these steps are performed on a data sample (using suitable sampling techniques). Its computational cost is not worse than the BIRCH one.

CLIQUE has been designed to locate clusters in subspaces of high dimensional data. This is useful because generally, in high dimensional spaces, data are scattered. CLIQUE partitions the space into a grid of disjoint rectangular units of equal size. The algorithm is made up of three phases: first, it finds subspaces containing clusters of dense units, than identifies the clusters, and finally generates a minimum description for each cluster. Also this algorithm scales linearly with the database size.

5 Novelty detection: Support Vector Clustering

Support Vector Machines and the related kernel methods are becoming popular for data mining tasks. In many real problems, the task is not classifying but novelties or anomalies detecting. In astrophysics, possible applications are the research of anomalous events or new astronomical sources. An approach is finding the *support* of a distribution (rather than estimating the density function of the data), thus avoiding the need of an a priori parameterized model of the distribution. A method to solve this problem is represented by the Support Vector Clustering (SVC) algorithm [20], in which data are mapped to a higher dimensional space by means of a Gaussian kernel function. In the new space, the algorithm finds the minimum sphere enclosing the data. The mapping of the sphere to the original input space generates a set of contours enclosing the data and corresponding to the support of the distribution. Outliers are defined as the Bounding Support Vectors (BSV).

6 Conclusions

In this work we have studied some data management and mining issues related to astrophysical data, aiming at a complete data mining framework. In particular, we have justified the need for a data warehousing approach to handle astrophysical data and we have focused on multidimensional access methods to efficiently index spatial and multidimensional data. A second issue concerns clustering techniques on large datasets, and we have discussed about some scalable algorithms with linear computational complexity. Finally, we have outlined the usefulness of non-parametric clustering algorithms, like the SVC, for novelty detection.

References

[1] *CDS*, `http://cdsweb.u-strasbg.fr`

[2] *SkyView - The Internet's Virtual Telescope*, `http://skyview.gsfc.nasa.gov`

[3] P. Dowler, D. Schade, R. Zingle, D. Durand, S. Gaudet, *Scientific Data Mining* ASP Conf. Ser., Vol. 216, Astronomical Data Analysis Software and Systems IX, eds. N. Manset, C. Veillet, D. Crabtree (San Francisco: ASP), 211 (2000)

[4] W. H. Inmon, *What is a Data Warehouse?*, Prism Tech Topics 1(1) (1997)

[5] S. Peterson, *Stars: A Pattern Language for Query Optimized Schema*, Portland Pattern Repository (1999)

[6] R. Stamper, *XML for STP data*, Tech. Report (2002)

[7] V. Gaede, O. Gunther, *Multidimensional access methods*, ACM Comput. Surv., 30:170-231 (1998)

[8] P. Z. Kunszt, A. S. Szalay, A. R. Thakar, *The Hierarchical Triangular Mesh*, Proc. of the MPA/ESO/MPE workshop in Mining the sky, 631-637, Springer Verlag (2001)

[9] J. L. Bentley, *Multidimensional binary search trees used for associative searching*, Communications of the ACM, 18(9), 509-517 (1975)

[10] J. T. Robinson, *The KD-B-tree: a search structure for large multidimensional Dynamic Indexes*, Proc. ACM SIGMOD Int. Conf. on Management of Data, 10-18 (1981)

[11] D. B. Lomet, B. Salzberg, *A Multiattribute Indexing Method with Good Guaranteed Performance*, ACM Trans. on Database Systems (1990)

[12] A. Guttman, *R-trees: A Dynamic index Structure for Spatial Searching*, Proc. ACM SIGMOD Int. Conf. on Management of Data, 47-57 (1984)

[13] N. Beckmann, H.-P. Kriegel, R. Schneider, B. Seeger, *The R^*-tree: An Efficient and Robust Access Method for Points and Rectangles*, Proc. ACM SIGMOD Int. Conf. on Management of Data, 322-331 (1990)

[14] N. Katayama, S. Satoh, *An Index Structure for High-Dimensional Nearest Neighbor Queries*, Proc. ACM SIGMOD Int Conf on Management of Data, 369-380 (1997)

[15] Y. Sakurai, M. Yoshikawa, S. Uemura, H. Kojima, *Spatial Indexing of High-Dimensional Data Based on Relative Approximation*, VLDB Journal, 11(2):93-108 (2002)

[16] D.A. White, R. Jain, *Similarity Indexing: Algorithms and Performance*, Proc. SPIE, 2670:62-73, San Diego, USA (1996)

[17] C. Martínez, S. Roura, *Optimal Sampling Strategies in Quicksort and Quickselect*, SIAM J. on Comput., 31:683-705 (2001)

[18] C. Böhm H.-P. Kriegel, *Efficient Bulk Loading of Large High-Dimensional Indexes*, DaWaK '99, 31:251-260 (1999)

[19] P. Berkhin, *Survey Of Clustering Data Mining Techniques*, Tech Report (2002)

[20] A. Ben-Hur, D. Horn, H. T. Siegelmann, V. Vapnik, *Support Vector Clustering*, Journal of Machine Learning Research 2(Dec), 125-137 (2001)