Data Reduction and Compression in Database Systems

Alexander Thomasian
New Jersey Institute of Technology, USA

INTRODUCTION

Data compression is storing data such that it requires less space than usual. Data compression has been effectively used in storing data in a compressed form on magnetic tapes, disks, and even main memory. In many cases, updated data cannot be stored in place when it is not compressible to the same or smaller size. Compression also reduces the bandwidth requirements in transmitting (program) code, data, text, images, speech, audio, and video. The transmission may be from main memory to the CPU and its caches, from tape and disk into main memory, or over local, metropolitan, and wide area networks. When data compression is used, transmission time improves or, conversely, the required transmission bandwidth is reduced. Two excellent texts on this topic are Sayood (2002) and Witten, Bell, and Moffat (1999).

Huffman encoding is a popular data compression method. It substitutes the symbols of an alphabet with k bits per symbol, so that frequent symbols are represented with fewer than k bits, and less common symbols with more than k bits. A significant saving in space is possible when the distribution is highly skewed, for example, the Zipf distribution, because the average number of bits to represent symbols is smaller than k bits. Arithmetic coding is a more sophisticated technique that represents a string with an appropriate fraction. Lempel-Ziv coding substitutes a character string by an index to a dictionary or a previous occurrence and the string length. Variations of these algorithms, separately or in combination, are used in many applications.

Compression can be lossy or lossless. Lossy data compression is utilized in cases where some data loss can be tolerated, for example, a restored compressed image may not be discernibly different from the original. Lossless data compression, which restores compressed data to its original value, is absolutely necessary in some applications. Quantization is a lossy data compression method, which represents data more coarsely than the original signal.

General purpose data compression is applicable to data warehouses and databases. For example, a variation of the lossless Lempel-Ziv method has been applied to DB2 records. This is accomplished by analyzing the records in a relational table and building dictionaries, which are then used in compressing the data. Because relational tables are organized as database pages, each page on disk (and main memory) will hold compressed data, so the number of records per page is doubled. Data is uncompressed on demand, with or without hardware assistance.

Data reduction operates at a higher level of abstraction than data compression, although data compression methods can be used in conjunction with data reduction. An example is quantizing the data in a matrix, which has been dimensionally reduced via the SVD method, as I describe in the following sections. This concludes the discussion of data compression; the remainder of this article deals with data reduction.

MAIN THRUST

Recent interest in data reduction resulted in the New Jersey Data Reduction Report (Barbara, et al., 1997), which classifies data reduction methods into parametric and nonparametric. Histograms, clustering, and indexing structures are examples of nonparametric methods. Another classification is direct versus transform-based methods. Singular value decomposition –(SVD) and discrete wavelet transforms are parametric transform-based methods. In the following few sections, I briefly introduce the aforementioned methods.

SVD

SVD is applicable to a two-dimensional M×N matrix X, where M is the number of objects, and there are N features per object. For example, M may represent the number of customers, and the columns represent the amount they spend on any of the N products. M may be in the millions, while N is the hundreds or even thousands.

According to SVD we have the decomposition $X = USV^t$, where U is another M×N matrix, S is a diagonal N×N matrix of singular values, s_n, $1 \leq n \leq N$, and V holds the eigenvectors of principal components. Alternatively, the covariance matrix C, the product of the transpose of X times X divided by M, can be decompoed as: $C = V\Lambda V^t$,

where Λ is a diagonal matrix of eigenvalues, $\lambda_n = s_n^2/M$. We assume, without a loss in generality, that the eigenvalues are in nonincreasing order, such that the transformation of coordinates into the principal components will yield Y = XV, whose columns are in decreasing order of their energy or variance. There is a reduction in the rank of the matrix when some eigenvalues are equal to zero. If we retain the first p columns of Y, the Normalized Mean Square Error –(NMSE) is equal to the sum of the eigenvalues of discarded columns divided by the trace of the matrix (sum of the eigenvalues or diagonal elements of C, which remains invariant). A significant reduction in the number of columns can be attained at a relatively small NMSE, as shown in numerous studies (see Korn, Jagadish, & Faloutsos, 1997).

Higher dimensional data, as in the case of data warehouses, for example, (product, customer, date) (dollars) can be reduced to two dimensions by appropriate transformations, for example, with products as rows and (customer × date) as columns.

The columns of dataset X may not be globally correlate —for example, high-income customers buy expensive items, and low-income customers buy economy items — so that the items bought by these two groups of customers are disjoint. Higher data compression (for a given NMSE) can be attained by first clustering the data, using an off-the-shelf clustering method, such as k-means (Dunham, 2003), and then applying SVD to clusters (Castelli, Thomasian, & Li, 2003). More sophisticated clustering methods, which generate elliptical clusters, may yield higher dimensionality reduction. An SVD-friendly clustering method, which generates clusters amenable to dimensionality reduction, is proposed in Chakrabarti and Mehrotra (2000).

K-nearest-neighbor (k-NN) queries can be carried out with respect to a dataset, which has been subjected to SVD, by first transforming the query point to the appropriate coordinates by using the principal components. In the case of multiple clusters, we first need to determine the cluster to which the query point belongs. In the case of the k-means clustering method, the query point belongs to the cluster with the closest centroid. After determining the k nearest neighbors in the primary cluster, I need to determine if other clusters are to be searched. A cluster is searched if the hypersphere centered on the query point, with the k nearest neighbors inside it, intersects with the hypersphere of that cluster. This step is repeated until no more intersections exist.

Multidimensional scaling – (MS) is another method for dimensionality reduction (Kruskal & Wish, 1978). Given the pair-wise distances or dissimilarities among a set of objects, the goal of MS is to represent them in k dimensions so that their distances are preserved. A stress function, which is the sum of squares of the difference between the distances of points with k dimensions and the original distance, is used to represent the goodness of the fit. The value of k should be selected to be as small as possible, while stress is maintained at an appropriately low level. A fast, approximate alternative is FASTMAP, whose goal is to find a k-dimensional space that matches the distances of an N×N matrix for N points (Faloutsos & Lin, 1995).

WAVELETS

According to Fourier's theorem, a continuous function can be expressed as the sum of sinusoidal functions. A discrete signal with n points can be expressed by the n coefficients of a Discrete Fourier Transform –(DFT). According to Parseval's theorem, the energy in the time and frequency domain are equal (Faloutsos, 1996).

The DFT consists of the sum of sine and cosine functions. I am interested in transforms, which can capture a vector with as few coefficients as possible. The Discrete Cosine Transform –(DCT) achieves better energy concentration than DFT and also solves the frequency-leak problem that plagues DFT (Agrawal, Faloutsos, & Swami, 1993).

The Discrete Wavelet Transform –(DWT) is also related to DFT but achieves better lossy data compression. The Haar transform is a simple wavelet transform that operates on a time sequence and computes the sum and difference of its halves, recursively. DWT can be applied to signals with multiple dimensions, one dimension at a time (Press, Teukolsky, Vetterling, & Flannery, 1996). To illustrate how a single dimensional wavelet transform works, consider an image with four pixels having the following values: [9,7,3,5] (Stollnitz, Derose, & Salesin, 1996). We obtain a lower resolution image by substituting pairs of pixel values with their average: [8,4]. Information is lost due to down sampling. The original pixels can be recovered by storing detail coefficients, given as 1=9–8 and –1=3–4, that is, [1,–1]. Another averaging and detailing step yields [6] and [2]. The wavelet transform of the original image is then [6,2,1,–1]. In fact, for normalization purposes, the last two coefficients have to be divided by the square root of 2. Wavelet compression is attained by not retaining all the coefficients.

As far as data compression in data warehousing is concerned, a k-d DWT can be applied to a k-d data cube to obtain a compressed approximation by saving a fraction of the strongest coefficients. An approximate computation of multidimensional aggregates for sparse data using wavelets is reported in Vitter and Wang (1999).