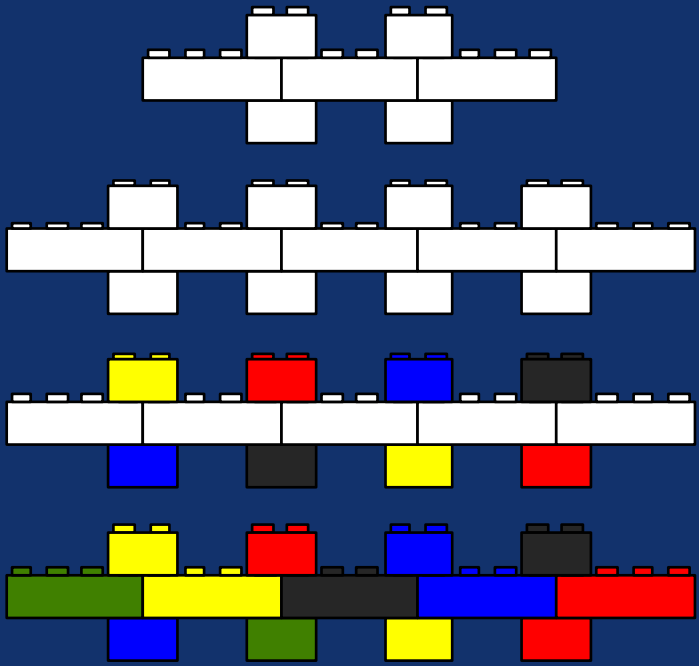
Mathematics of Big Data

Spreadsheets, Databases, Matrices, and Graphs

- course notes -

Jeremy Kepner and Hayden Jananthan



MIT LINCOLN LABORATORY BOOK SERIES

MIT PRESS

Foreword by Charles E. Leiserson

1 Introduction and Overview

Summary

Data are stored in a computer as sets of bits (0's and 1's) and transformed by data processing systems. Different steps of a data processing system impose different views on these sets of bits: spreadsheets, databases, matrices, and graphs. These views have many similar mathematical features. Making data rigorous mathematically means coming up with a rigorous definition of sets of bits (associative arrays) with corresponding operations (addition and multiplication) and showing that the combination is a reasonable model for data processing in the real world. If the model is accurate, then the same mathematical representations can be used across many steps in a data processing system, thus simplifying the system. Likewise, the mathematical properties of associative arrays can be used to swap, reorder, and eliminate data processing steps. This chapter presents an overview of these ideas that will be addressed in greater detail in the rest of the book.

1.1 Mathematics of Data

While some would suggest that data are neither inherently good nor bad, data are an essential tool for displacing ignorance with knowledge. The world has become "data driven" because many decisions are obvious when the correct data are available. The goal of data—to make better decisions—has not changed, but how data are collected has changed. In the past, data were collected by humans. Now, data are mostly collected by machines. Data collected by the human senses are often quickly processed into decisions. Data collected by machines are dormant until the data are processed by machines and acted upon by humans (or machines). Data collected by machines are stored as bits (0's and 1's) and processed by mathematical operations. Today, humans determine the correct mathematical operations for processing data by reasoning about the data as sets of organized bits.

Data in the world today are sets of bits stored and operated on by diverse systems. Each data processing system has its own method for organizing and operating on its sets of bits to deliver better decisions. In most systems, both the sets of bits and the operations can be described precisely with mathematics. Learning the mathematics that describes the sets of bits in a specific processing system can be time-consuming. It is more valuable to learn the mathematics describing the sets of bits that are in many systems.

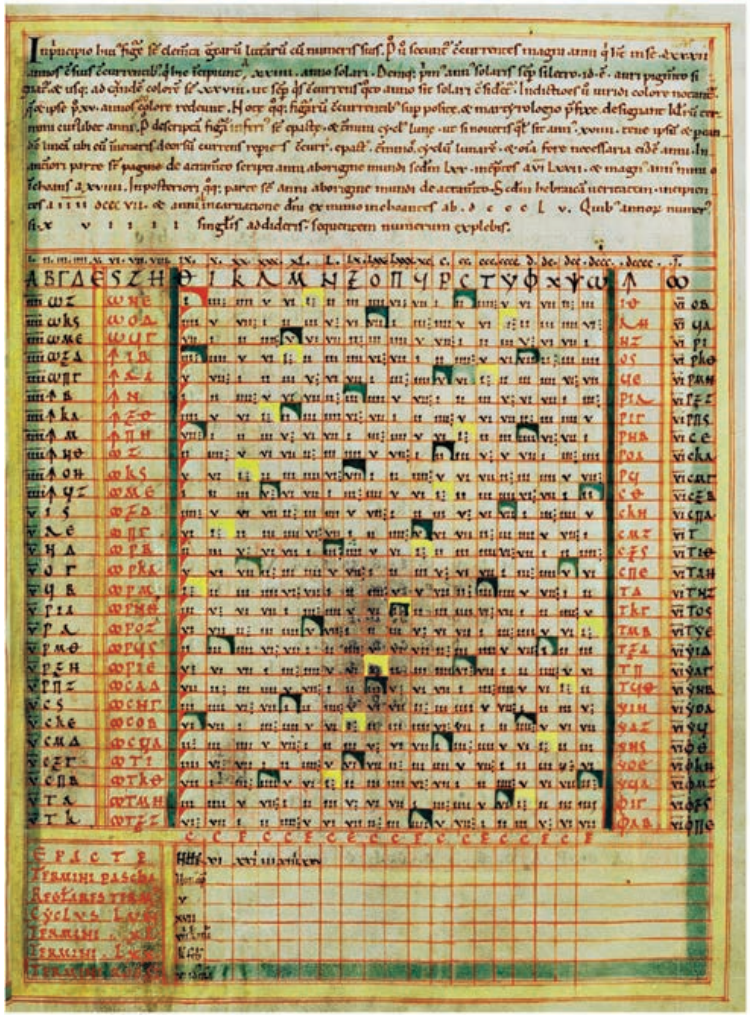


Figure 1.1
Tables have been used since antiquity as demonstrated by the Table of Dionysius Exiguus (MS. 17, fol. 30r, St. John's College, Oxford University) from the Thorney Computus, a manuscript produced in the first decade of the 12th century at Thorney Abbey in Cambridgeshire, England. ©2011 IEEE. Reprinted, with permission, from [1]

The mathematical structure of data stored as sets of bits has many common features. Thus, if individual sets of bits can be described mathematically, then many different sets of bits can be described using similar mathematics. Perhaps the most intuitive way to organize a set of bits is as a table or an associative array. Associative arrays consisting of rows, columns, and values are used across many data processing systems. Such arrays (see Figure 1.1) have been used by humans for millennia [2] and provide a strong foundation for reasoning about whole classes of sets of bits.

Making data rigorous mathematically means combining specific definitions of sets of bits, called associative arrays, with the specific operations of addition and multiplication, and showing that the combination makes sense. Informally, "makes sense" means that the combination of associative arrays and operations behave in ways that are useful. Formally, "makes sense" means that the combination of associative arrays has certain mathematical properties that are useful. In other words, utility is the most important aspect of making

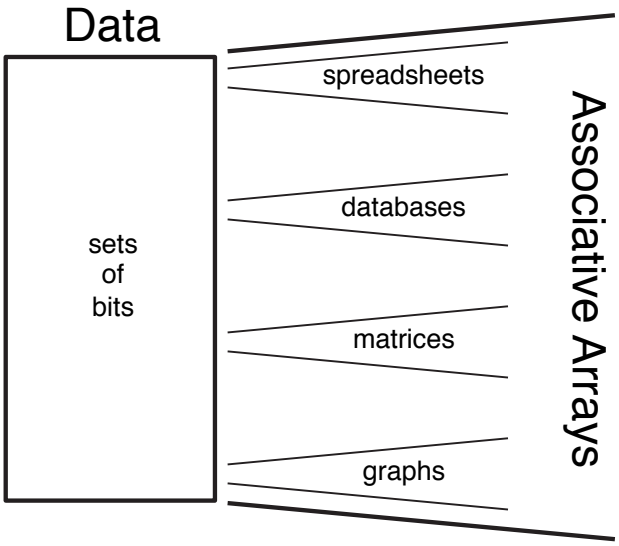


Figure 1.2
Data are sets of bits in a computer. Spreadsheets, databases, matrices, and graphs provide different views for organizing sets of bits. Associative arrays encompass the mathematical properties of these different views.

data rigorous. This fact should be kept in mind as these ideas are developed throughout the book. All the mathematics of data presented here have been demonstrated to be useful in real applications.

1.2 Data in the World

Data in the world today can be viewed from several perspectives. Spreadsheets, database tables, matrices, and graphs are commonly used ways to view data. Most of the benefits of these different perspectives on data can be encompassed by associative array mathematics (or algebra). The first practical implementation of associative array mathematics that bridges these perspectives on data can be found in the Dynamic Distributed Dimensional Data Model (D4M). Associative arrays can be understood textually as data in tables, such as a list of songs and the various features of those songs. Likewise, associative arrays can be understood visually as connections between data elements, such as lines connecting different elements in a painting.

Perspectives on Data

Spreadsheets provide a simple tabular view of data [3]. It is estimated that more than 100 million people use a spreadsheet every day. Almost everyone has used a spreadsheet at one time or another to keep track of their money, analyze data for school, or plan a schedule for an activity. These diverse uses in everyday life give an indication of the power and flexibility of the simple tabular view of data offered by spreadsheets.

As the number of rows and columns in a table grows beyond what can be easily viewed, then a database [4] can be used to store and retrieve the same tabular information. Databases

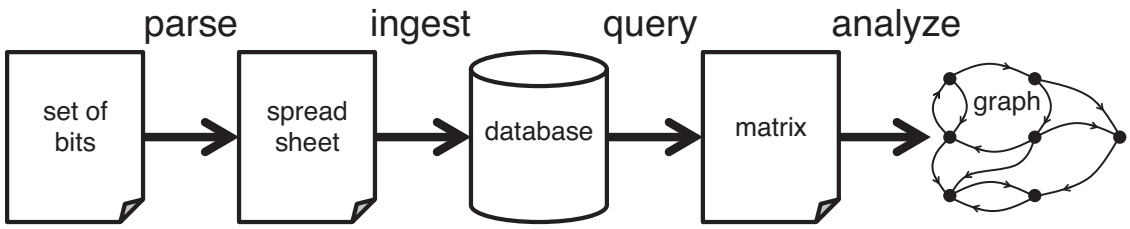


Figure 1.3
The standard data processing steps often require different perspectives on the data. Associative arrays enable a common mathematical perspective to be used across all the steps.

that organize data into large related tables are the most commonly used tool for storing and retrieving data in the world. Databases allow separate people to view the data from different locations and conduct transactions based on entries in the tables. Databases play a role in most purchases. In addition to transactions, another important application of databases is the analysis of many rows and columns in a table to find useful patterns. For example, such analysis is used to determine if a purchase is real or fake.

Mathematics also uses a tabular view to represent numbers. This view is referred to as a matrix [5], a term first coined by English mathematician James Joseph Sylvester in 1848 while working as an actuary with fellow English mathematician and lawyer Arthur Cayley [6]. The term matrix was taken from the Latin word for "womb." In a matrix (or womb of numbers), each row and column is specified by integers starting at 1. The values stored at a particular row and column can be any number. Matrices are particularly useful for comparing whole rows and columns and determining which ones are most alike. Such a comparison is called a correlation and is useful in a wide range of applications. For example, matrix correlations can determine which documents are most like other documents so that a person looking for one document can be provided a list of similar documents. Or, if a person is looking for one kind of document, a correlation can be used to estimate what products they are most likely to buy.

Mathematical matrices also have a concept of sparsity whereby numbers equal to zero are treated differently from other numbers. A matrix is said to be sparse if lots of its values are zero. Sparsity can simplify the analysis of matrices with large numbers of rows and columns. It is often useful to rearrange (or permute) the rows and columns so that the groups of nonzero entries are close together, clearly showing the rows and columns that are most closely related.

Humans have an intuitive ability to understand visual relationships among data. A common way to draw these relationships is a through a picture (graph) consisting of points (vertices) connected by lines (edges). These pictures can readily highlight data that are connected to lots of other data. In addition, it is also possible to determine how closely connected two data elements are by following the edges connecting two vertices. For example, given a person's set of friends, it is possible to suggest likely new friends from their friends' friends [7].

A

	Artist	Date	Duration	Genre
053013ktnA1	Bandayde	2013-05-30	5:14	Electronic
053013ktnA2	Kastle	2013-05-30	3:07	Electronic
063012ktnA1	Kitten	2010-06-30	4:38	Rock
082812ktnA1	Kitten	2012-08-28	3:25	Pop

Figure 1.4
Tabular arrangement of a collection of songs and the features of those songs arranged into an associative array **A**. That each row label (or row key) and each column label (or column key) in **A** is unique is what makes it an associative array.

Interestingly, graphs can also be represented in a tabular view using sparse matrices. Furthermore, the same correlation operation that is used to compare rows and columns in a matrix can also be used to follow edges in a graph. The duality between graphs and matrices is one of the many interesting mathematical properties that can be found among spreadsheets, databases, matrices, and graphs. Associative arrays are a tool that encompasses the mathematical properties of these different views of data (see Figure 1.2). Understanding associative arrays is a valuable way to learn the mathematics that describes data in many systems.

Dynamic Distributed Dimensional Data Model

The D4M software (d4m.mit.edu) [8, 9] is the first practical implementation of the full mathematics of associative arrays that successfully bridges spreadsheets, databases, matrices, and graphs. Using associative arrays, D4M users are able to implement high performance complex algorithms with significantly less effort. In D4M, a user can read data from a spreadsheet, load the data into a variety of databases, correlate rows and columns with matrix operations, and visualize connections using graph operations. These operations correspond to the steps necessary to build an end-to-end data processing system (see Figure 1.3). Often, the majority of time spent in building a data processing system is in the defining of the interfaces between the various steps, which normally requires a conversion from one mathematical perspective of the data to another. By using the same mathematical abstraction across all steps, the construction time of a data processing system is significantly reduced. The success of D4M in building real data processing systems has been a prime motivation for formalizing the mathematics of associative arrays. By making associative arrays mathematically rigorous, it becomes possible to apply associative arrays in a wide range of programming environments (not just D4M).

Associative Array Intuition: Text

Associative arrays derive much of their power from their ability to represent data intuitively in easily understandable tables. Consider the list of songs and the various features of those songs shown in Figure 1.4. The tabular arrangement of the data shown in Figure 1.4 is an associative array (denoted **A**). This arrangement is similar to those widely used in

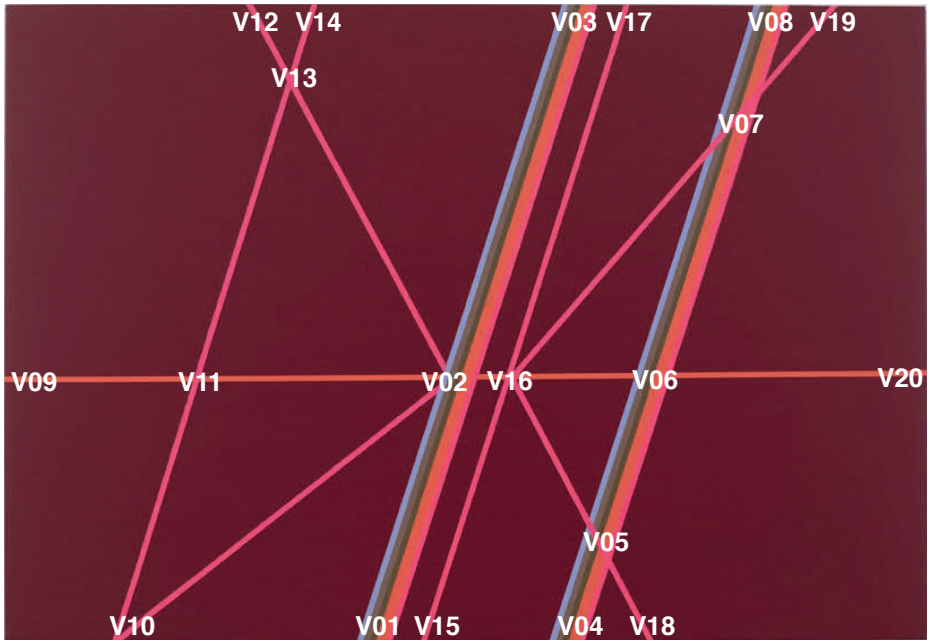


Figure 1.5
Abstract line painting (*XCRS* by Ann Pibal) showing various colored lines. The intersections and terminations of the lines are labeled vertices (V01,...,V20) and have been superimposed onto the painting in white letters.

spreadsheets and databases. Figure 1.4 does contain one property that distinguishes it from being an arbitrary arrangement of data in a two-dimensional grid. Specifically, each row key and each column key in **A** is unique. This property is what makes **A** an associative array and allows **A** to be manipulated as a spreadsheet, database, matrix, or graph.

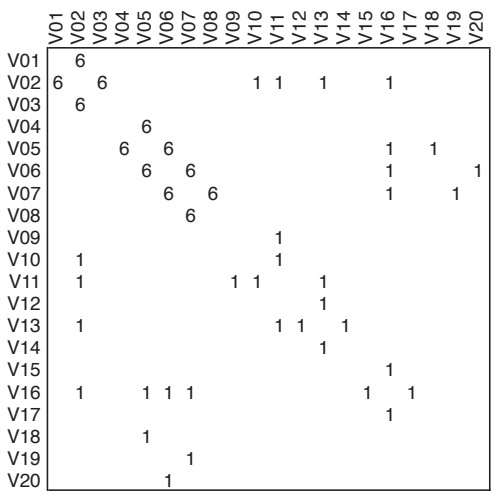
An important aspect of Figure 1.4 that makes **A** an associative array is that each row and column is identified with a string called a key. An entry in **A** consists of a triple with a row key, a column key, and a value. For example, the upper-left entry in **A** is

$$\mathbf{A}(\text{'053013ktnA1 '},\text{'Artist '}) = \text{'Bandayde '}$$

In many ways, associative arrays have similarities to matrices where each entry has a row index, column index and a value. However, in an associative array the row keys and the column keys can be strings of characters and are not limited to positive integers as they are in a matrix. Likewise, the values in an associative array are not just real or complex numbers and can be numbers or strings or even sets. Typically, the rows and columns of an associative array are represented in sorted order such as alphabetical ordering. This ordering is a practical necessity to make retrieval of information efficient. Thus, in practice, associative array row keys and column keys are orderable sets.

Associative Array Intuition: Graphics

Associative arrays can be visualized as relations between data elements, which are depicted as lines connecting points in a painting (see Figure 1.5). Such a visual depiction of



A

Figure 1.6
Square symmetric associative array representation of the edges depicted in Figure 1.5; each value represents the number of edges connecting each pair of vertices.

relationships is referred to mathematically as a graph. The points on the graph are called the vertices. In the painting, the intersections and terminations of the lines (or edges) are called vertices and are labeled V01,...,V20. An associative array can readily capture this information (see Figure 1.6) and allow it to be manipulated as a spreadsheet, database, matrix, or graph. Such analysis can be used to identify the artist who made the painting [10] or used by an artist to suggest new artistic directions to explore.

In Figure 1.6, each value of the associative array stores the count of edges going between each pair of vertices. In this case, there are six pairs of vertices that all have six edges between them

$$(\mathsf{V01},\mathsf{V02}),(\mathsf{V02},\mathsf{V03}),(\mathsf{V04},\mathsf{V05}),(\mathsf{V05},\mathsf{V06}),(\mathsf{V06},\mathsf{V07}),(\mathsf{V07},\mathsf{V08})$$

This value is referred to as the edge weight, and the corresponding graph is described as a weighted-undirected graph. If the edge weight is the number of edges between two vertices, then the graph is a multi-graph.

1.3 Mathematical Foundations

Data stored as sets of bits have many similar mathematical features. It makes sense that if individual types of sets can be described mathematically, then many different sets can be described using similar mathematics. Perhaps the most common way to arrange a set of bits is as a table or an associative array. Associative arrays consisting of rows, columns, and values are used across many data processing systems. To understand the mathematical

structure of associative arrays requires defining the operations of addition and multiplication and then creating a formal definition of associative arrays that is consistent with those operations. In addition, the internal structure of the associative array is important for a range of applications. In particular, the distribution of nonzero entries in an array is often used to represent relationships. Finally, while the focus of this book is on two-dimensional associative arrays, it is worth exploring those properties of two-dimensional associative arrays that extend into higher dimensions.

Mathematical Operations

Addition and multiplication are the most common operations for transforming data and also the most well studied. The first step in understanding associative arrays is to define what adding or multiplying two associative arrays means. Naturally, addition and multiplication of associative arrays will have some properties that are different from standard arithmetic addition

$$2+3=5$$

and standard arithmetic multiplication

$$2\times 3=6$$

In the context of diverse data, there are many different functions that can usefully serve the role of addition and multiplication. Some common examples include max and min

$$\max(2,3)=3$$
$$\min(2,3)=2$$

and union, denoted $\cup$, and intersection, denoted $\cap$

$$\{2\}\cup\{3\}=\{2,3\}$$
$$\{2\}\cap\{3\}=\emptyset$$

To prevent confusion with standard addition and multiplication, $\oplus$ will be used to denote associative array element-wise addition and $\otimes$ will be use to denote associative array element-wise multiplication. In other words, given associative arrays $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$, that represent spreadsheets, database tables, matrices, or graphs, this book will precisely define corresponding associative array element-wise addition

$$\mathbf{C}=\mathbf{A}\oplus\mathbf{B}$$

associative array element-wise multiplication

$$\mathbf{C}=\mathbf{A}\otimes\mathbf{B}$$

and associative array multiplication that combines addition and multiplication

$$\mathbf{C} = \mathbf{AB}$$

The above array multiplication can also be denoted $\oplus.\otimes$ to highlight its special use of both addition and multiplication

$$\mathbf{C} = \mathbf{A} \ \oplus.\otimes \ \mathbf{B}$$

Finally, array transpose is used to swap rows and columns and is denoted

$$\mathbf{A}^{\mathsf{T}}$$

That these operations can be defined so that they make sense for spreadsheets, databases, matrices, and graphs is what allows associative arrays to be an effective tool for manipulating data in many applications. The foundations of these operations are basic concepts from abstract algebra that allow the ideas of addition and multiplication to be applied to both numbers and words. It is a classic example of the unforeseen benefits of pure mathematics that ideas in abstract algebra from the 1800s [11] are beneficial to understanding data generated over a century later.

Formal Properties

It is one thing to state what associative arrays should be able to represent and what operations on them are useful. It is another altogether to prove that for associative arrays of all shapes and sizes that the operations hold and maintain their desirable properties. Perhaps the most important of these properties is coincidentally called the *associativity* property, which allows operations to be grouped arbitrarily. In other words,

$$(\mathbf{A} \oplus \mathbf{B}) \oplus \mathbf{C} = \mathbf{A} \oplus (\mathbf{B} \oplus \mathbf{C})$$
$$(\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C} = \mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C})$$
$$(\mathbf{AB})\mathbf{C} = \mathbf{A}(\mathbf{BC})$$

The associativity property allows operations to be executed in any order and is extremely useful for data processing systems. The ability to swap steps or to change the order of processing in a system can significantly simplify its construction. For example, if arrays of data are entering a system one row at a time and the first step in processing the data is to perform an operation across all columns and the second requires processing across all rows, this switching can make the system difficult to build. However, if the processing steps possess the property of associativity, then the first and second steps can be performed in a different order, making it much easier to build the system. [Note: the property of associativity should not be confused with the adjective *associative* in associative array; the similarity is simply a historical coincidence.]

Another powerful property is *commutativity*, which allows arrays in an operation to be swapped

$$\mathbf{A} \oplus \mathbf{B} = \mathbf{B} \oplus \mathbf{A}$$
$$\mathbf{A} \otimes \mathbf{B} = \mathbf{B} \otimes \mathbf{A}$$
$$(\mathbf{AB})^{\mathsf{T}} = \mathbf{B}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}$$

If operations in data processing systems are commutative, then this property can be directly translated into systems that will have fewer deadlocks and better performance when many operations are run simultaneously [12].

To prove that associative arrays have the desired properties requires carefully studying each aspect of associative arrays and verifying that it conforms to well-established mathematical principles. This process pulls in basic ideas from abstract algebra, which at first glance may feel complex, but when presented in the context of everyday concepts, such as tables, can be made simple and intuitive.

Special Arrays and Graphs

The organization of data in an associative array is important for many applications. In particular, the placement of nonzero entries in an array can depict relationships that can also be shown as points (vertices) connected by lines (edges). These diagrams are called graphs. For example, one such set of relationships is those genres of music that are performed by particular musical artists. Figure 1.7 extracts these relationships from the data in Figure 1.4 and displays it as both an array and a graph.

Certain special patterns of relationships appear frequently and are of sufficient interest to be given names. Modifying Figure 1.7 by adding and removing some of the relationships (see Figure 1.8) produces a special array in which each row corresponds to exactly one column. Likewise, the graph of these relationships shows the same pattern, and each genre vertex is connected to exactly one artist vertex. This pattern of connections is referred to as the *identity*.

Modifying Figure 1.7 by adding relationships (see Figure 1.9) creates a new array in which each row has a relationship with every column. Likewise, the graph of these relationships shows the same pattern, and each genre vertex is connected to all artist vertices. This arrangement is called a biclique.

In addition, to the identity and the biclique patterns, there are a variety of other patterns that are important because of their special properties. For example, the square-symmetric pattern (see Figure 1.6), where the row labels and the column labels are the same and the pattern of values is symmetric around the diagonal, indicates the presence of an undirected graph. Understanding how these patterns manifest themselves in associative arrays makes it possible to recognize these special patterns in spreadsheets, databases, matrices, and graphs. In a data processing system, recognizing that the data have one of these special

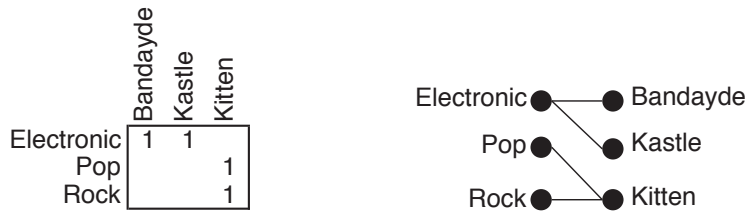


Figure 1.7
Relationships between genres of music and musical artists taken from the data in Figure 1.4. The array on the left shows how many songs are performed for each genre and each artist. The graph on the right shows the same information in visual form.

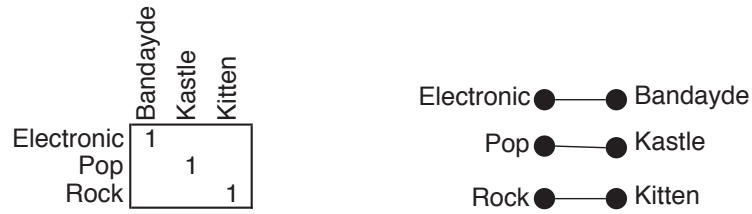


Figure 1.8
Modifying Figure 1.7 by removing some of the relationships results in a special array where each row corresponds to exactly one column. The graph of these relationships has the same pattern, and each genre vertex connects to exactly one artist vertex. This pattern is referred to as the identity.

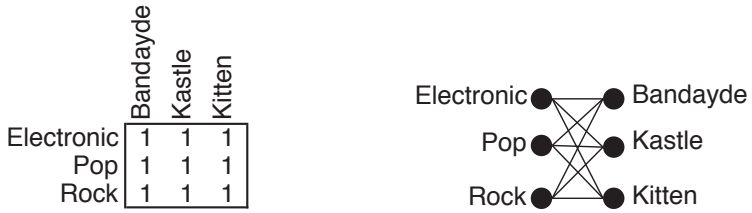


Figure 1.9
Modifying Figure 1.7 by adding relationships produces a special array in which each row has a relationship with every column. The graph of these relationships shows the same pattern, and each genre vertex is connected to all artist vertices. This collection is referred to as a biclique.

patterns can often be used to eliminate or simplify a data processing step. For example, data with the identity pattern shown in Figure 1.7 simplifies the task of looking up an artist, given a specific genre, or a genre, given a specific artist, because there is a 1-to-1 relationship between genre and artist.

Higher Dimensions

The focus of this book is on two-dimensional associative arrays because of their natural connection to spreadsheets, databases, matrices, and graphs, which are most commonly two-dimensional. It is worth examining the properties of two-dimensional associative arrays that also work in higher dimensions. Figure 1.10 shows the data from Figures 1.7,

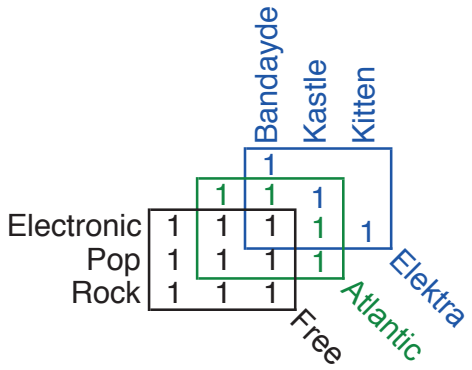


Figure 1.10
Data from Figures 1.7, 1.8, and 1.9 arranged in a three-dimensional array, or tensor, using an additional dimension corresponding to how the music was distributed.

1.8, and 1.9 arranged in a three-dimensional array, or tensor, using an additional dimension corresponding to how the music was distributed. Many of the structural properties of arrays in two dimensions also apply to higher-dimensional arrays.

1.4 Making Data Rigorous

Describing data in terms of rigorous mathematics begins with combining descriptions of sets of bits in the form of associative arrays with mathematical operations, such as addition and multiplication, and proving that the combination makes sense. When a combination of sets and operations is found to be useful, it is given a special name so that the combination can be referred to without having to recall all the necessary definitions and proofs. The various named combinations of sets and operations are interrelated through a process of specialization and generalization. For example, the properties of the real numbers

$$\mathbb{R} = (-\infty, \infty)$$

are in many respects a specialization of the properties of the integers

$$\mathbb{Z} = \{\ldots, -1, 0, 1, \ldots\}$$

Likewise, associative arrays $\mathbb{A}$ are a generalization that encompasses spreadsheets, databases, matrices, and graphs. To prove this generalization requires building up associative arrays from more fundamental combinations of sets and operations with well-established mathematical properties. These combinations include well-defined sets and operations, such as matrices, addition and multiplication of matrices, and the generalization of matrix entries to words and numbers.

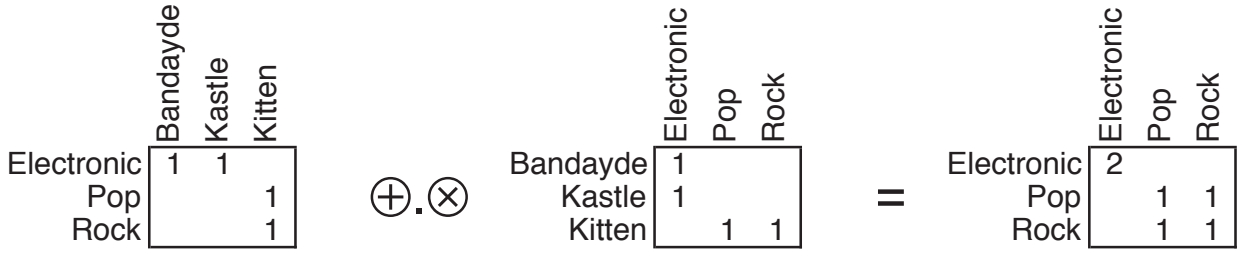


Figure 1.11
Correlation of different musical genres using associative array multiplication $\oplus.\otimes$.

Matrices, Combining Matrices, and Beyond

If associative arrays encompass the matrices, then many of the useful behaviors that are found in matrices may also be found in associative arrays. A matrix is a two-dimensional arrangement of numbers with specific rules on how matrices can be combined using addition and multiplication. The property of associativity allows either addition or multiplication operations on matrices to be performed in various orders and to produce the same results. The property of distributivity provides a similar benefit to certain combinations of multiplications and additions. For example, given matrices (or associative arrays) **A**, **B**, and **C**, these matrices are distributive over addition $\oplus$ and multiplication $\otimes$ if

$$\mathbf{A} \otimes (\mathbf{B} \oplus \mathbf{C}) = (\mathbf{A} \otimes \mathbf{B}) \oplus (\mathbf{A} \otimes \mathbf{C})$$

An even stronger form of the property of distributivity occurs when the above formula also holds for the matrix multiplication that combines addition and multiplication

$$\mathbf{A}(\mathbf{B} \oplus \mathbf{C}) = (\mathbf{AB}) \oplus (\mathbf{AC})$$

As with the associativity property, the distributivity property enables altering the order of steps in a data processing system and can be used to simplify its construction.

The property of distributivity has been proven for matrices in which the values are numbers and the rows and columns are labeled with positive integers. Associative arrays generalize matrices to allow the values, rows, and columns to be numbers or words. To show that a beneficial property like distributivity works for associative arrays requires rebuilding matrices from the ground up with a more general concept for the rows, columns, and values.

Multiplication

Multiplication of associative arrays is one of the most useful data processing operations. Associative array multiplication can be used to correlate one set of data with another set of data, transform the row or column labels from one naming scheme to another, and aggregate data into groups. Figure 1.11 shows how the different musical genres can be correlated by artist using associative array multiplication.

For associative array multiplication to provide these benefits requires understanding how associative array multiplication will behave in a variety of situations. One important situation occurs when associative array multiplication will produce a result that contains only zeros. It would be expected that multiplying one associative array by another associative array containing only zeros would produce only zeros. Are there other conditions under which this is true? If so, recognizing these conditions can be used to eliminate operations.

Another important situation is determining the conditions under which associative array multiplication will produce a result that is unique. If correlating musical genre by artist produces a particular result, will that result come about only with those specific associative arrays or will different associative arrays produce the same result? If multiplying by certain classes of associative arrays always produces the same result, this property can also be used to reduce the number steps.

Eigenvectors

Knowing when associative array multiplication produces a zero or unchanging result is very useful for simplifying a data processing system, but these situations don't always occur. If they did, associative array multiplication would be of little use. A situation that occurs more often is when associative array multiplication produces a result that projects one of the associative arrays by a fixed amount along a particular direction (or eigenvector). If a more complex processing step can be broken up into a series of simple eigenvector projection operations on the data, then it may be possible to simplify a data processing system.

1.5 Conclusions, Exercises, and References

This chapter has provided a brief overview of the remaining chapters in the book with the goal of making clear how the whole book ties together. Readers are encouraged to refer back to this chapter while reading the book to maintain a clear understanding of where they are and where they are going.

This book will proceed in three parts. Part I: Applications and Practice introduces associative arrays with real examples that are accessible to a variety of readers. Part II: Mathematical Foundations provides a mathematically rigorous definition of associative arrays and describes the properties of associative arrays that emerge from this definition. Part III: Linear Systems shows how concepts of linearity can be extended to encompass associative arrays.

Exercises

Exercise 1.1 — Refer to the array in Figure 1.4.

(a) Compute the number of rows m and number of columns n in the array.

(b) Compute the total number of entries mn.

(c) How many empty entries are there?

(d) How many filled entries are there?

Remember the row labels and column labels are not counted as part of the array.

Exercise 1.2 — Refer to the painting in Figure 1.5.

(a) Count the total number of vertices.

(b) One line passes through six vertices, list the vertices on this line.

(c) There are six triangles in the pictures, list the vertices in each triangle.

Exercise 1.3 — Refer to the associative array in Figure 1.6.

(a) Why is the array called square?

(b) Why is the array called symmetric?

(c) Sum the rows and the columns and find the row and column with the largest sum?

Exercise 1.4 — Refer to the array and graph in Figure 1.7.

(a) Compute the number of rows m and number of columns n in the array.

(b) How many genre vertices are in the graph? How many artist vertices are in the graph?.

(c) Compute the total number of entries mn in the array.

(d) How many empty entries are there in the array?

(e) How many filled entries are there in the array?

(f) How many edges are in the graph?

Exercise 1.5 — Consider arrays $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ and element-wise addition denoted by $\oplus$ and element-wise multiplication denoted by $\otimes$.

(a) Write an expression that illustrates associativity among $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$.

(b) Write an expression that illustrates commutativity among $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$.

(c) Write an expression that illustrates distributivity among $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$.

Exercise 1.6 — List some of the earliest examples of humans using tables to store information. Discuss how those instances are similar and different from how humans use tables today.

Exercise 1.7 — List some of the different perspectives that can be used to view data. How are these different perspectives similar and different?

Exercise 1.8 — What is the main goal of a data processing system? What are the advantages to a common mathematical view of data as it flows through a data processing system?

Exercise 1.9 — What are the two main mathematical operations that are performed on data? Speculate as to why these are the most important operations.

Exercise 1.10 — What is the main goal of rigorously defining mathematical operations on associative arrays?

References

[1] J. Kepner, W. Arcand, D. Bestor, B. Bergeron, C. Byun, V. Gadepally, M. Hubbell, P. Michaleas, J. Mullen, A. Prout, A. Reuther, A. Rosa, and C. Yee, "Achieving 100,000,000 database inserts per second using Accumulo and D4M," in *High Performance Extreme Computing Conference (HPEC)*, IEEE, 2014.

[2] F. T. Marchese, "Exploring the origins of tables for information visualization," in *15th International Conference on Information Visualisation (IV), 2011*, pp. 395–402, IEEE, 2011.

[3] D. J. Power, "A history of microcomputer spreadsheets," *Communications of the Association for Information Systems*, vol. 4, no. 1, p. 9, 2000.

[4] K. L. Berg, T. Seymour, and R. Goel, "History of databases," *International Journal of Management & Information Systems (Online)*, vol. 17, no. 1, p. 29, 2013.

[5] "Khan academy - intro to the matrices." http://www.khanacademy.org/math/algebra2/alg2-matrices/basic-matrix-operations-alg2/v/introduction-to-the-matrix. Accessed: 2017-04-08.

[6] A. Cayley, "A memoir on the theory of matrices," *Philosophical Transactions of the Royal Society of London*, vol. 148, pp. 17–37, 1858.

[7] M. Zuckerburg, "Facebook and computer science." Harvard University CS50 guest lecture, Dec. 7 2005.

[8] J. V. Kepner, "Multidimensional associative array database," Jan. 14 2014. US Patent 8,631,031.

[9] J. Kepner, W. Arcand, W. Bergeron, N. Bliss, R. Bond, C. Byun, G. Condon, K. Gregson, M. Hubbell, J. Kurz, A. McCabe, P. Michaleas, A. Prout, A. Reuther, A. Rosa, and C. Yee, "Dynamic Distributed Dimensional Data Model (D4M) database and computation system," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5349–5352, IEEE, 2012.

[10] C. R. Johnson, E. Hendriks, I. J. Berezhnoy, E. Brevdo, S. M. Hughes, I. Daubechies, J. Li, E. Postma, and J. Z. Wang, "Image processing for artist identification," *IEEE Signal Processing Magazine*, vol. 25, no. 4, 2008.

[11] R. Dedekind, "Über die komposition der binären quadratischen formen," in *Über die Theorie der ganzen algebraischen Zahlen*, pp. 223–261, Springer, 1964.

[12] A. T. Clements, M. F. Kaashoek, N. Zeldovich, R. T. Morris, and E. Kohler, "The scalable commutativity rule: Designing scalable software for multicore processors," *ACM Transactions on Computer Systems*, vol. 32, no. 4, p. 10, 2015.