APL 82, W. H. Janko, W. Stucky (ed.),
APL Quote Quad, Vol. 13 No. 1,
© 1982 by ACM, Inc.

INVITED ADDRESS

RECTANGULARLY ARRANGED COLLECTIONS OF COLLECTIONS

Trenchard More
staff member
IBM Çambridge Scientific Center
545 Technology Square
Cambridge, MA USA 02139
(617) 421-9234

Abstract

The concepts of collection, rectangular arrangement, and membership for nested arrays are discussed separately and independently of array-theoretic operations. The concepts are illustrated by examples generated in the Nested Interactive Array Language, Nial.

Introduction

The model of data used in APL [0], which here is called the **basic array** [1], has been extended to the **nested array** [2] in the Nested Arrays System [3,4] of the Scientific Time Sharing Company, Inc. (STSC) and to the **enclosed array** [5,6] by I. P. Sharp Associates Limited (IPSA). The nested array, which is the same as the **array** in the array-theoretic model of data [7-9], is used in Brown's work [10-12] on extensions to APL. The array of array theory is also the model of data used in the Nested Interactive Array Language, Nial [13-20].

The representation of tensors, linear transformations, and vectors by rectangular arrangements of real numbers suggests a two-sorted theory of collections based on two different universes of objects. One universe contains nothing but **elementary objects** [1], which correspond to the real numbers. The other universe contains nothing but **basic arrays** [1], which are collections of nothing but elementary objects gathered together in various rectangular arrangements. The elementary objects are considered to be indecomposable units that are not collections. The basic arrays are considered to be decomposable structures that are not elementary objects.

Even though the gathering of elementary objects into basic arrays provides a model of data that is essentially two-sorted, it is possible to do the work of all representations and manipulations in one of the sorts, the basic arrays, by letting the elementary objects correspond one-to-one with the basic arrays that have no axes. This is the fundamental insight of APL.

As will be made clear in the discussion below, a nilvalent basic array, which is called a scalar in APL, has no axes and therefore necessarily gathers together precisely one elementary object. APL treats a nilvalent basic array as if it were the unique elementary object that is gathered in the rectangular arrangement [21].

Orth [22] and Mercer [23] have compared the extensions of APL that have been made by STSC to nested arrays and by IPSA to enclosed arrays. The point of departure in these extensions is the basic array. The functions and terminology that have been used in APL for basic arrays are extended to include nested arrays or enclosed arrays.

Array theory and Nial have assimilated many of the principles of APL, LISP, and set theory but are not extensions of these systems. The point of departure is the visually perceived, spatially arranged collection of material objects. The reasoning and the terminology proceed from the general case of a rectangularly arranged collection of such collections to special cases, such as simple arrays that represent basic arrays and atomic arrays that represent elementary objects.

Explanations in APL of nested arrays and enclosed arrays tend to depend on a repertory of extended APL functions appropriate for such arrays. Explanations of the array, or nested array, the model of data used in array theory and Nial, have tended to develop the concepts of collection, rectangularity, and membership interdependently and simultaneously with the definition of array-theoretic operations. The purpose of the present paper is to separate these concepts for the nested array and to discuss them independently of array-theoretic operations.

Objects as collections

To **collect**, according to the Oxford English Dictionary (OED), means "To gather together into one place or group; to gather, get together." A **collection** is then "A number of objects collected or gathered together, viewed as a whole; a group of things collected and arranged: ..." OED As might be expected from the way in which language evolves to an economy of expression, nothing is said about the nature of the objects collected. No restriction is placed on the kinds of objects that may be gathered into a collection.

Since collections are viewed as wholes, collections may be treated as objects. All collections are objects. It is therefore possible to construct a collection of collections by gathering a number of collections together. The result of the gathering is viewed as a whole, in other words, as an object. No restriction is placed on the kinds of collections that may be gathered into a collection.

Since all collections are objects, one is led to consider whether there is an essential difference between a collection and an object. Is there an example of an object that cannot also be a collection? A collection, according to the OED definition, is the consequence of viewing as a whole the result of gathering together a number of objects. The question can be made more precise. Is there an example of an object that cannot be the consequence of viewing as a whole the result of gathering together a number of objects? The surprising answer is "no." Every object may be treated as a collection.

Given an object, such as a window, we may suspend analysis and take the object as a whole, or we may decompose the object into certain fragments, such as panes of glass, and consider the fragments as objects that have been gathered. An object perceived to have no fragments or two or more fragments is easy to regard a being a collection that results from gathering no objects or two or more objects. A gathering of no objects constitutes an empty collection. A gathering of two or more objects is necessarily a collection.

The most difficult case occurs when an object, such as a number or a truth-value, is perceived as being monolithic, a whole that is not decomposable into fragments. The solution to this case, which leads to certain theories of collections that are consistent relative to other such theories, is to consider the monolithic object as being a whole that results from gathering together precisely one object, which is the monolithic object itself [24].

Most objects considered in data processing are obviously collections, such as strings of characters, lists of words, tables of numbers, and files of records. If one assumes that every number, truth-value, character, and indivisible phrase is a collection that results from gathering together only the collection itself, then one can assert that all objects serving as data and processed by machine are finite collections in which the gathered objects themselves are again finite collections.

For a collection to contain only itself, the unique object that is gathered to make up the collection must be the very same collection. The device of self-containment brings all data under the purview of a one-sorted theory of collections: the only objects considered in the theory, the only objects that can be observed directly, are collections.

Within the confines of the theory, self-containment causes both "collection" and "object" to refer to the same concept. To paraphrase the OED definition, a **collection** is a number of collections gathered together, viewed as a whole. Although the word "object" is redundant, it is convenient to continue using the word to clarify the relationship between the things that are gathered together, the objects, and the thing that results from the gathering, the collection. A collection is a collection of objects. The objects are collections of objects. These objects, again, are collections of objects. This recursion is endless; it effectively terminates only in self-containment.

Objects as arrays

When stored or communicated by physical devices, collections become ordered; gathered objects fall into spatial or temporal arrangements. The kind of arrangement that is easiest to analyze and most likely to occur is **rectangular** in the sense that the gathered objects occur adjacent to one another in gangs of equally long parallel lines. All lines in a gang have the same extent in number of objects. The extents may differ from gang to gang.

Every object in a rectangular arrangement must occur in one line selected from each gang for the arrangement. For example, the panes of glass in a window, which may be treated as objects in a collection, occur simultaneously in a gang of horizontal lines, called **rows**, and a gang of vertical lines, called **columns**. Every pane occurs in one row and one column.

We now restrict our attention to objects that can be perceived to have finitely many fragments. All further consideration of collections is confined to those that have rectangular arrangement. To indicate this change, we use the word "array" instead of the word "collection." An **array**, according to the OED, is an "Arrangement in line or ranks, esp. martial order." In all of the preceding paragraphs we may substitute the word "array" for the word "collection."

To **array** means to gather together into one place or group, in rectangular arrangement. An **array** is a number of objects arranged or gathered together in rectangular arrangement, viewed as a whole. All arrays are objects. It is therefore possible to construct an array of arrays by gathering a number of arrays together. No restriction is placed on the kinds of arrays that may be gathered into an array.

Is there an example of an object (finite in number of fragments) that cannot also be an array? Again "no." All finite objects can be treated as arrays. A gathering of no objects in rectangular arrangement constitutes an empty array. A gathering of two or more objects in rectangular arrangement is necessarily an array.

A monolithic object is considered to be an **atomic array**, which results from gathering together in a particular rectangular arrangement precisely one object, which is the atomic array itself. The nature of the rectangular arrangement will be discussed later. (Atomic arrays have been called motes in previous papers on array theory. The change in terminology emphasizes that these recursively structured objects are arrays subject to the same general rules and operations that apply to all arrays.)

The device of using self-containment in certain arrays, called atomic arrays, brings all data under the purview of a one-sorted theory of arrays: the only objects considered in the theory, the only objects that can be observed directly, are arrays. An **array** is a collection of arrays that are gathered together in rectangular arrangement.

Rectangular arrangement

To each gang of lines in a rectangular arrangement there corresponds an implicit direction of adjacency, called an **axis**, that runs parallel to the lines and points from objects nearer the beginning of the arrangement toward objects nearer the end. The axes of an arrangement are referred to as the axes of the corresponding array. Unless drawn for the sake of illustration, axes do not appear explicitly in an array. Reference to an axis provides a convenient way to describe a property inherent in rectangular arrangement.

The number of different gangs, and therefore the total number of axes, is called the **valency** of the arrangement and of the corresponding array. Valency has the same meaning in tensor algebra [25]. Valency is the number of vector variables in a multilinear form or tensor, which equals the total number of upper and lower indices on a symbol representing the contravariant and covariant components of the tensor.

If all of the vector variables in a tensor or multilinear form are represented by lists of components and the number of components in each such list is the same, then this number is called the **dimension** of the underlying vector space. Dimension is a concept that corresponds to the number of positions on an axis, which is the common number of locations on each line in a gang.

Valency, the number of axes, is an intrinsic property of an array and of the rectangular arrangement for the array. If the valency of an array A is represented by a number N that is assumed to be an atomic array, then N is called the **valence** of A.

The gangs for an array are ordered; the corresponding axes are labeled as the 0-axis, 1-axis, 2-axis, etc. The lines in the last gang for an array are called **rows** and are drawn horizontally; the lines in the next to last gang are called **columns** and are drawn vertically; the lines in the next preceding gang are called **tiers** and again are drawn horizontally. the lines in the first gang for an array are called **pillars** and are drawn vertically if the valency is even and nonzero, horizontally if the valency is odd. If the valency is 2, then pillars are the same as columns. If the valency is 3, then pillars are tiers.

Tables and trivalent arrays

Figure 0 shows the rectangular arrangement of a **bivalent** array, which is an array that has 2 axes and therefore 2 gangs of parallel lines. Bivalent arrays are called **tables**. The row gang has 3 rows. Each row comprises 4 locations, which are visualized as corresponding to 4 positions on the row axis that points from left to right. The column gang has 4 columns. Each column comprises 3 locations, which are visualized as corresponding to 3 positions on the column axis that points from top to bottom.

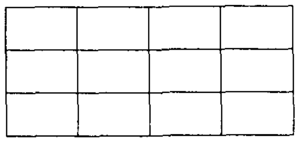


Figure 0. Each of the 12 cells in the 3-by-4 box represents a location in the rectangular arrangement for a bivalent array. The 3-by-4 shape indicates that the 0-axis has 3 positions and the 1-axis has 4 positions.

Figure 1 shows the rectangular arrangement of a **trivalent** array, which is an array that has 3 axes and therefore 3 gangs of parallel lines. The row gang in Figure 1 has 6 rows, each comprising 4 locations. The column gang has 8 columns, each comprising 3 locations. The tier gang has 12 tiers, each comprising 2 locations.

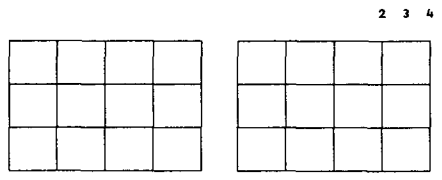


Figure 1. Rectangular arrangement for a trivalent array. The 2-by-3-by-4 shape indicates that the 0-axis has 2 positions, the 1 axis has 3 positions, and the 2-axis has 4 positions.

If a position on the tier axis is fixed and if the positions on all other axes are allowed to vary, then the effect is to sweep out a subarray encompassing 12 locations, which are the locations at the fixed position in each of the 12 tiers. This subarray is orthogonal to the tier axis and is called a **tier slab**. There are 2 tier slabs because the tier axis has 2 positions. Each tier slab amounts to a 3-by-4 table orthogonal to all 12 tiers. The tier axis points from the 3-by-4 slab on the left to the 3-by-4 slab on the right.

The column axis in Figure 1 has 3 positions and is orthogonal to 3 column slabs. Each column slab amounts to a 2-by-4 table orthogonal to all 8 columns. The row axis has 4 positions and is orthogonal to 4 row slabs. Each row slab amounts to a 2-by-3 table orthogonal to all 6 rows.

In general, the subarray that is swept out by fixing a position on one axis and letting positions on all other axes vary is called a **slab**. A slab is orthogonal to a line. Since a table has two axes, the row slabs of a table are columns and the column slabs are rows.

Singles

What are the characteristics of the rectangular arrangement for a **univalent** array, namely, an array that has only one axis and therefore only one gang of parallel lines? How many lines may be in a gang if there is only one gang? Figures 0 and 1 illustrate how a gang must have as many lines as there are locations in a slab orthogonal to the lines. If there is only one axis, then a slab orthogonal to the axis is a subarray having no axes. How many locations are possible in a rectangular arrangement that has no axes?

It may seem that a rectangular arrangement without axes is a strange limiting case of little interest to the arrangements of data commonly encountered in practice. Yet the univalent array, the array with one axis, is the most common of all arrangements and is itself a limiting case that depends directly on the rectangular arrangement of the **nilvalent** array, the array without axes. A determination of the rectangular arrangement for a nilvalent array will also determine the rectangular arrangement for a univalent array.

If more than one object occurs in the arrangement for an array, then there exist at least two objects in the arrangement that are adjacent. Such adjacency establishes a direction of adjacency and therefore an axis. In other words, if two or more objects occur in the arrangement for an array, then the arrangement, and therefore the array, has at least one axis.

The law of contraposition in propositional logic asserts that if A implies B, then not B implies not A. It follows from the preceding paragraph, by contraposition, that if an array does not have at least one axis, then it is not possible for two or more objects to occur in the arrangement for the array. In other words, an arrangement without axes can involve at most one object. A nilvalent array can be the result of gathering together at most one object.

An array is said to be **nonempty** if it is the result of gathering together at least one object in rectangular arrangement. Every object so gathered must occur in as many lines as there are gangs for the arrangement -- one line from each gang. If an object occurs in a line of a gang, then all lines in that gang must have at least one location and the corresponding axis must have nonzero extent in number of positions. An array is nonempty if and only if every axis has nonzero extent. Using contraposition and the logic of quantifiers, it follows that an array is empty if and only if at least one axis has zero extent. Hence an empty array must have at least one axis. By contraposition, an array without axes must be nonempty.

According to the conclusions of the two preceding paragraphs, a nilvalent array must be the result of gathering together precisely one object. For this reason, a nilvalent array is called a **single**. A single is a collection constructed by gathering together one object in a rectangular arrangement that has no gangs of parallel lines. The object that is gathered in a nilvalent arrangement may be any array. Figure 2 shows the rectangular arrangement of a single.

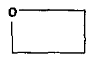


Figure 2. Rectangular arrangement for a nilvalent array. The small circle at the top left corner indicates that there are no axes. The unique cell within the box indicates that the arrangement has precisely one location for an object to occur in the arrangement.

Lists

A univalent array has one gang of parallel lines, which are called rows because the gang is necessarily the last gang. All rows of a univalent array are orthogonal to a row slab at each position on the row axis. Each row slab is a subarray that has no axes. In other words, at each position on the row axis of a univalent array, all rows are orthogonal to a single. A single has precisely one location in its nilvalent arrangement. Only one row can be orthogonal to a single.

It follows that a univalent array must have exactly one row. For this reason, a univalent array is called a **list**. The OED defines a list as "A catalogue or roll consisting of a row or series of names, figures, words, or the like."

Figure 3 shows the rectangular arrangement of a list. The axis implicit in Figure 3, which is necessarily the row axis, has 5 positions and is orthogonal to 5 row slabs. Each of these slabs amounts to a single that is orthogonal to the row.

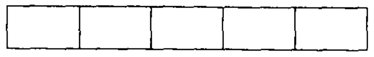


Figure 3. Rectangular arrangement for a univalent array. The shape indicates that the 0-axis has 5 positions.

Representation of tensors by arrays

What is the rectangular arrangement for an atomic array? As defined earlier, an atomic array is the result of gathering together in rectangular arrangement precisely one object, which is the atomic array itself. All axes, if any, for such an arrangement must have only one position. Each gang must have only one line and each line only one location. The structure of an atomic array is known if the following question can be answered: How many axes does an atomic array have? What is the valency of an atomic array?

A vector space involves an additive group of elements, called **vectors**, the field of real numbers, called **coefficients**, and certain rules for multiplying vectors by coefficients. The product of a vector with a coefficient is again a vector. If each vector in a given set of vectors is multiplied by a coefficient and if these products are summed, then the resulting vector is a **linear combination** of the given vectors. A **basis** for a vector space is a linearly independent set of vectors that spans the space: every vector in the space is a linear combination of the basis vectors.

A **tensor** of valency N is simply a real-valued function of N vector variables that is linear in each variable separately. For example, if x, y, and z are vectors and f is a trivalent tensor, then f(x,y,z) is a real number. Linearity in the variable x means that if u and v are vectors and a and b are coefficients, then

f(ua+vb,y,z) = f(u,y,z)a + f(v,y,z)b.

The application of the trivalent tensor f to 3 basis vectors results in a real number, which is called a purely covariant **component** of f. If the vector space is 4-dimensional, then there are 4 different basis vectors and 64 different ways f can be applied to a triple of basis vectors. The trivalent tensor f can be represented by a 4-by-4-by-4 trivalent array of components.

The function of two vectors x and y that results in the inner product of x with y is an example of a bivalent tensor, called the metric tensor, which in a 4-dimensional vector space is represented by a 4-by-4 array of components. Similarly, a univalent tensor, like a vector, can be represented by a list of 4 components.

A nilvalent tensor, which is called a **scalar**, results in a real number when applied to no vectors. This real number is always the same, regardless of the vectors to which the scalar is not applied. When not applied to any one of the basis vectors, a scalar results in the same component. Thus a scalar, or nilvalent tensor, is represented by a single, or nilvalent array, that gathers together precisely one real number, which is the component of the scalar.

Valency of an atomic array

In the case of a scalar, a tensor of valency 0, there appears to be no advantage in distinguishing the unique component of the scalar from the nilvalent array of components that represents the scalar. The monolithic nature of a number, the inability to visualize how a number can be decomposed into a spatial arrangement of fragments, persuades us to assume that a number is an atomic array. The object that is gathered together to create an atomic array is the atomic array itself. The component of a scalar is a real number that is assumed to be an atomic array.

The component of a scalar is now identified with the nilvalent array of components that represents the scalar. In other words, the atomic array that serves as the component of a scalar is now assumed to be the same as the nilvalent array that represents the scalar. This requires the atomic array to be nilvalent, to be without axes. We assume further that all atomic arrays are nilvalent. Thus an atomic array is a single that gathers together precisely one object, which is the single itself.

Given the assumptions that real numbers are atomic arrays and that atomic arrays are nilvalent, is there a difference between a single that gathers together one real number and the real number itself? No. Arrays are said to be the **same** or **equal** if they gather together the same objects at the same locations in the same rectangular arrangement.

Let N be a real number, which is an atomic array that is assumed to be nilvalent. Let S be the single that gathers N in nilvalent arrangement. N and S have the same rectangular arrangement; both are singles. Both N and S gather the atomic array N at this unique location. Thus N and S are equal; they cannot be distinguished by operations on arrays. A scalar, or nilvalent tensor, with component N is represented by the atomic array N.

Ordinary and exotic arrays

As mentioned earlier, an array is said to be **empty** if and only if at least one axis has no positions. All lines, if any, in at least one of the gangs of an empty array have no extent (extent 0) and therefore no locations. The rectangular arrangement for an empty array has no locations because no slabs can be orthogonal to an axis of no extent. An empty array is an empty collection that gathers together no objects in rectangular arrangement.

An array is said to be **singular** if at least one axis has precisely one position. All lines, if any, in at least one of the gangs of a singular array have singular extent (extent 1) and therefore precisely one location.

It is possible for a **multivalent** array, an array with two or more axes, to be both empty and singular because one axis may have extent 0 while another axis has extent 1. For example, a 0-by-1 table has no rows of extent 1 and one column of extent 0. There are no locations in the 0-by-1 table because there are no locations in the unique column. If there were a row, it would have one location, but there are no rows.

All empty arrays, all singular arrays, and all nilvalent arrays (singles) are said to be **exotic**. Atomic arrays are exotic. **Valent** arrays, arrays with at least one axis, for which all axes have extent 2 or greater are said to be **ordinary**. The ordinary arrays are those that are not exotic.

Examples of arrays

There are two kinds of arrays that occur most often in practical applications. The first kind comprises the atomic arrays, such as numbers, truth-values, characters, phrases, and faults. The second kind comprises the shortest ordinary lists, such as pairs and triples. Phrases include the indivisible words and sequences of words that occur in language. Faults are similar to phrases but are reserved for reporting error.

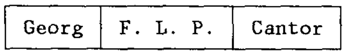
A **triple** is a collection that gathers together three objects in a univalent arrangement. A **pair** is a collection of two objects gathered in univalent arrangement. The objects gathered in a triple or a pair may be arbitrary arrays that have no necessary similarities or differences. Triples are lists because the objects gathered occur one after the other in one row parallel to one axis. The same is true of pairs.

The gathering of arrays into a rectangularly arranged collection is said to **nest** the arrays within the collection. The word "nesting" is used to describe this process because the objects that are gathered and the collection that results are all arrays. The gathering of an atomic array into a nilvalently arranged collection has no effect because the nilvalent array that results is the same in every respect as the atomic array gathered. The successive nesting of arrays within arrays effectively terminates in atomic arrays.

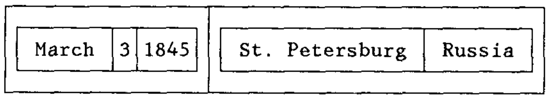
The simplest examples of arrays necessarily involve atomic arrays. Ordinary arrays cannot be described without also mentioning certain exotic arrays, namely, the atomic arrays. Figure 4 shows diagrams for (i) a triple, Person, that results from gathering three phrases; (ii) a pair, Birth, that results from gathering a triple followed by a pair; and (iii) a pair, Parents, that results from gathering two triples.

Characters, such as letters, digits, punctuation marks, and symbols, are assumed to be atomic arrays. Characters and nonsingular lists of characters are called **strings**. Examples of strings are the triple 'A5+', which gathers the characters 'A' and '5' and '+' in univalent arrangement, the pair 'A5', which gathers the first two of these characters in univalent arrangement, the single 'A', which is the atomic array that gathers the character 'A' in nilvalent arrangement, and the empty list '', which gathers no characters in univalent arrangement.

```
First_name Surname ←"Georg "Cantor
Middle_initial ← phrase'F. L. P.'
Person ← First_name Middle_initial Surname
Person
```



```
Month Day Year ← "March 3 1845
Date ← Month Day Year
City Country ← (phrase'St. Petersburg') "Russia
Birthplace ← City Country
Birth ← Date Birthplace
Birth
```



```
Father ← "Georg "W. "Cantor
Mother ← "Maria "_ "Bohm
Parents ← Father Mother
Parents
```

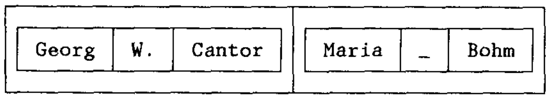


Figure 4. Diagrams of a triple and two pairs. The assignments above each of these diagrams show how the triple and pairs are constructed.

The operation **phrase** converts a string to a phrase. For example, phrases representing Georg Cantor's surname, middle initials, and city of birth are constructed as follows:

```
phrase 'Cantor'
phrase 'F. L. P.'
phrase 'St. Petersburg'.
```

A phrase spelled without blanks, such as Cantor's surname, can also be named by quoting it. Thus the string '"Cantor', which is the list of the seven characters between the tick marks, mentions the phrase "Cantor , which is the indivisible surname spelled by the six letters between the leading quote mark and the trailing blank. "Cantor is the same atomic array as phrase 'Cantor'. Every phrase is an atomic array.

Cantor's year of birth can be represented either by the phrase "1845 or by the integer 1845. These two atomic arrays are unequal because they differ in type.

Item connotes membership of arrays

In all of the preceding discussion, nothing has been said about elements or items. An array is a particular kind of collection in which objects are gathered in rectangular arrangement. In a one-sorted theory, the objects gathered in any rectangular arrangement are themselves arrays. Although there is no conceptual difference between an object and an array, it is convenient, as mentioned before in the case of collections, to use the word "object" in addition to the word "array" to distinguish between the things that are gathered in rectangular arrangement, the objects, and the thing that results from the gathering, the array.

For the purposes of a one-sorted theory, all arrays are objects; all objects are arrays. Arrays are the only objects that are observed, described, used, and manipulated. Instead of relying on connotations of the words "array" and "object" to communicate indirectly the relationship that exists between an object (array) that is gathered in a rectangular arrangement and the object (array) that results from such a gathering, we shall use the words "hold" and "item" to communicate the relationship directly.

An array is the consequence of viewing as a whole the result of gathering together a number of objects in rectangular arrangement: each object gathered in this way is said to be an **item** of the array; the array is said to **hold** each of the objects. In other words, given any two arrays A and B, the array A either does or does not occur as one of the objects gathered in rectangular arrangement to make the array B. If A does have this relationship to B, then A is an item of B and B holds A. "Is-an-item-of" and "holds" are converse ways of expressing the same relationship between two arrays.

An **item**, according to the OED, is "An article or unit of any kind included in an enumeration, computation, or sum total; an entry or thing entered in an account or register, a clause of a document, a detail of expenditure or income, etc." The important point to notice about each part of this definition is that the word "item" refers to an object relative to a collection in which the object occurs. The statement that the array A is an item is meaningful only in the context of an array B, which is not necessarily different from A. Once the array B is identified as the context, then A is an item or not an item according as B holds or does not hold A.

The following are the only statements of membership that can be made about the three arrays in Figure 5: Parents holds Father; both Father and Surname hold Surname. These statements can be made in converse form: Surname is an item of itself and of Father; Father is an item of Parents.

Since rectangular arrangement orders the objects that are gathered together in an array, it is possible to say not only whether an object occurs as an item but where an object occurs as an item. In the arrays of Figure 5, Surname is the third (and last) item of Father and the first (and last) item of Surname. Father is the first item of Parents.

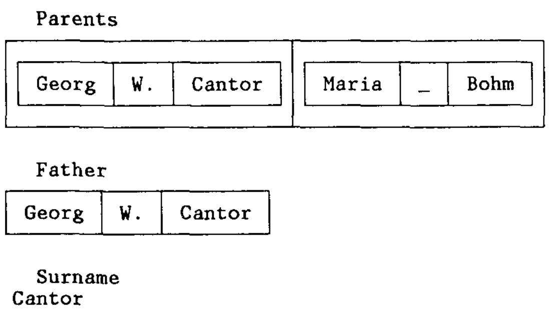


Figure 5. Diagrams of the pair, Parents, of the triple, Father, and of the atomic array, Surname.

Element connotes membership of sets

A **set** is the consequence of viewing as a whole the result of gathering together a number of objects without regard to arrangement: each object gathered in this way is said to be an **element** of the set; the set is said to **contain** each of the objects. A set is a collection of sets in a one-sorted theory of sets; an array is a collection of arrays in a one-sorted theory of arrays. An atomic set, also called an individual, contains itself as sole element. An atomic array holds itself as sole item.

The OED defines an **element** to be "A component part of a complex whole." As in the case of "item," the word "element" refers to an object relative to a collection in which the object occurs. The statement that the set A is an element is meaningful only in the context of a set B, which may or may not differ from A. When the set B is identified as the context, the set A is an element or not an element according as B contains or does not contain A.

The concept of membership is essentially the same for sets and arrays. (The chief difference occurs in the structure of empty collections. Although the empty set is unique, empty arrays differ with infinite variety not only in rectangular arrangement but also in type of object not held.)

Conclusion

In view of the extensive use of sets to represent all kinds of mathematical structures, it is important to preserve the close similarity between sets and arrays. The similarity permits arrays to represent sets quite directly. The concepts of collection and membership are used in set theory and array theory in the same way that they are used in plain language.

An array serves the purposes of a set when the only questions asked of the array concern the occurrence or nonoccurrence of objects as items in the array. An array is fundamentally a collection, like a set. Arrays are said to be **like** if they hold the same items, regardless of where the items occur in the arrangements or how many times any given item occurs. Likeness of arrays corresponds to equality of sets.

In addition to having the properties of a collection, an array also has the properties of rectangular arrangement, which are ignored by certain array-theoretic operations that correspond to some of the set-theoretic operations. As far as these operations are concerned, an array amounts to a set.

Figures 6 and 7 show how the terminology of membership for arrays parallels that for sets. The reason for adopting different but parallel terminologies is to permit array theory and set theory to be used in the same discussion. At each point in such a discussion, the choice of words indicates which kind of collection is being considered.

collective object	**relationship**	**collected object**
collection	**gathers**	**member**
array	**holds**	**item**
set	**contains**	**element**

Figure 6. Parallel terminology for collections in general, rectangularly arranged collections, and collections without regard to arrangement.

collected object	**relationship**	**collective object**
member	**gathered in**	**collection**
item	**occurs in**	**array**
element	**belongs to**	**set**

Figure 7. Parallel terminology, converse to that of Figure 6.

Acknowledgments

I am indebted to Flemming Schmidt, Neil Sorensen, and particularly Mike Jenkins for technical discussions concerning various points of this paper. Comprehensive acknowledgments are given at the end of Reference 16.

Notes and references

0. A. D. Falkoff and K. E. Iverson, **APL/360 User's Manual**. Thomas J. Watson Research Ctr., IBM Corp., Yorktown, NY, July, 1968.

1. T. More, "Nested rectangular arrays for measures, addresses and paths," ACM STAPL/Sigplan APL79, also **APL Quote Quad 9**, 4 - part 1, June 1979, pp. 156-163. "Each component of a grounded, rectangular, nested array is either another such array or an elementary object, such as a number or character, that is not a grounded

array. A **basic array** is a grounded array in which all components are elementary objects." [p. 158] I prefer the word "basic" to the word "flat" because "flat" has technical meaning in affine spaces and linear algebra.

2. T. More, "The nested rectangular array as a model of data," invited address, ACM STAPL/Sigplan APL79, also **APL Quote Quad 9**, 4 - part 1, June 1979, pp. 55-73. I use the word "array" instead of the phrase "general array," as logicians use "set" instead of "general set," to indicate that the general case is the fundamental concept. Special kinds of arrays, such as simple arrays and atomic arrays, are named by qualifying the root word "array." The phrase "general array" suggests that a fundamental concept has been generalized. I have used the terms **nested rectangular array** and **nested array** in a descriptive rather than a qualifying sense to mean **array**: "The purpose of array theory or list theory is to carry the mathematics of nested arrays or nested lists far into the province of algorithms before using programming techniques to bestow the full power of effective computability." [p. 55]

3. C. M. Cheney, **APL*PLUS™ Nested Arrays System Reference Manual**. STSC, Inc., Bethesda, MD, 1981.

4. R. A. Smith, "Nested arrays, operators, and functions," ACM SIGAPL APL81, also **APL Quote Quad 12**, 1, Sept. 1981, pp. 286-290.

5. R. Bernecky and K. E. Iverson, "Operators and enclosed arrays," APL USERS MEETING, I. P. Sharp Associates Limited, Toronto, 1980.

6. R. Bernecky, "Representations for enclosed arrays," ACM SIGAPL APL81, also **APL Quote Quad 12**, 1, Sept. 1981, pp. 42-46.

7. T. More, "A theory of arrays with applications to databases," Tech. Rep. G320-2106, IBM Scientific Ctr., Cambridge, MA, Sept. 1975.

8. T. More, "Types and prototypes in a theory of arrays," Tech. Rep. G320-2112, IBM Scientific Ctr., Cambridge, MA, May 1976.

9. T. More, "On the composition of array-theoretic operations," Tech. Rep. G320-2113, IBM Scientific Ctr., Cambridge, MA, May 1976.

10. J. A. Brown, "APL language extensions," Proceedings of SEAS 1978 anniversary meeting, Stresa, Italy, vol. 1, pp. 335-353.

11. J. A. Brown, "Evaluating extensions to APL," ACM STAPL/Sigplan APL79, also **APL Quote Quad 9**, 4 - part 1, June 1979, pp. 148-155.

12. J. A. Brown and M. A. Jenkins, "The APL identity crisis," ACM SIGAPL APL81, also **APL Quote Quad 12**, 1, Sept. 1981, pp. 62-66.

13. A. Hassitt and L. E. Lyon, "Array theory in an APL environment," ACM STAPL/Sigplan APL79, also **APL Quote Quad 9**, 4 - part 1, June 1979, pp. 110-115.

14. M. A. Jenkins, "A development system for testing array theory concepts," ACM SIGAPL APL81, also **APL Quote Quad 12**, 1, Sept. 1981, pp. 152-159.

15. W. G. Bouricius and N. R. Sorensen, "An informal introduction to array theory with applications to a language and a database." Oyvind Bjorke and Ole. I. Franksen (eds.), **Structures and Operations in Engineering and Management Systems**. Tapir Publishers, Trondheim, Norway, 1981, pp. 447-496.

16. T. More, "Notes on the diagrams, logic and operations of array theory." Oyvind Bjorke and Ole. I. Franksen (eds.), **Structures and Operations in Engineering and Management Systems**. Tapir Publishers, Trondheim, Norway, 1981, pp. 497-666.

17. N. Jacobsen, S. W. Poulsen, R. Stockner, and P. H. Thygesen, "Nial applications, an introductory investigation," Electric Power Engineering Department, Technical University of Denmark, Publ. 8202, April 1982.

18. M. A. Jenkins, **The Q'Nial™ Reference Manual**, Queen's University, Kingston, Ontario, April 1982.

19. W. S. Adams, "Plain programming in Nial," Queen's University Tech. Rep., Kingston, Ontario, June 1982.

20. F. Schmidt and M. A. Jenkins, "Systems design and the Nial approach," Queen's University Tech. Rep., Kingston, Ontario, June 1982.

21. W. E. Gull and M. A. Jenkins, "Recursive data structures in APL," **Comm. ACM 22**, Feb. 1979, pp. 79-96.

22. D. L. Orth, "A comparison of the IPSA and STSC implementations of operators and general arrays," **APL Quote Quad 12**, 2, Dec. 1981, pp. 11-18.

23. R. Mercer, "A based system for general arrays," **APL Quote Quad 12**, 2, Dec. 1981, pp. 18-21.

24. W. V. Quine, "Unification of universes in set theory," **Journal of Symbolic Logic** **21**, 3, Sept. 1956, pp. 267-279.

25. J. C. H. Gerretsen, **Lectures on Tensor Calculus and Differential Geometry**. P. Noordhoff N.V. Groningen, 1962.