Geometric Ordering of Concepts, Logical Disjunction, and Learning by Induction

Dominic Widdows and Michael Higgins
MAYA Design
{widdows,higgins}@maya.com

Abstract

In many of the abstract geometric models which have been used to represent concepts and their relationships, regions possessing some cohesive property such as convexity or linearity have played a significant role. When the implication or containment relationship is used as an ordering relationship in such models, this gives rise to logical operators for which the disjunction of two concepts is often larger than the set union obtained in Boolean models. This paper describes some of the characteristic properties of such broad non-distributive composition operations and their applications to learning algorithms and classification structures.

As an example we describe a quad-tree representation which we have used to provide a structure for indexing objects and composition of regions in a spatial database. The quad-tree combines logical, algebraic and geometric properties in a naturally non-distributive fashion.

The lattice of subspaces of a vector space is presented as a special example, which draws a middle-way between 'non-inductive' Boolean logic and 'overinductive' tree-structures. This gives rise to composition operations that are already used as models in physics and cognitive science.

Closure conditions in geometric models

The hypothesis that concepts can be represented by points and more general regions in spatial models has been used by psychologists and cognitive scientists to simulate human language learning (Landauer & Dumais 1997) and to represent sensory stimuli such as tastes and colors (Gärdenfors 2000, §1.5). Of the traditional practical applications of such a spatial approach, the vector space model for information retrieval (Salton & McGill 1983) is notable, and its generalizations such as latent semantic analysis (Landauer & Dumais 1997), in which distributions of word usage learned from corpora become condensed into a low-dimensional representation and used, among other things, for discriminating between different senses of ambiguous words (Schütze 1998).

Schütze's (1998) paper exemplifies some of the opportunities and challenges involved in such a spatial approach — these include learning to represent individual objects as points in a geometric space (in this case, word vectors), combining these points into appropriate sentence or document vectors (in this case, using addition of vectors), and extrapolating from observed points of information to apportion the geometric space into cohesive regions corresponding to recognizable concepts (in this case, using clustering).

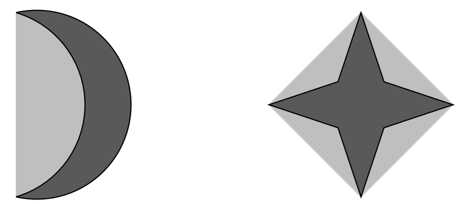

Figure 1: Two non-convex sets (dark gray) and the points added to form their convex closures (light gray)

The last question — how are empirical observations gathered into classes described by the same word or represented by the same concept? — is of traditional importance in philosophy and many related disciplines. The extrapolation from observed data to classifying previously unexperienced situations is implemented in a variety of theoretical models and practical applications, using smoothing and clustering, by exploiting a natural general-to-specific ordering on the space of observations (Mitchell 1997, Ch. 6, 7, 2), and by using similarity or distance measures to gauge the influence exerted by a cluster of observations upon its conceptual hinterland (Gärdenfors 2000, Ch. 3,4).

Mathematically, such extrapolation techniques are related to *closure conditions*, a set being *closed* if it has no tendency to include new members. A traditional example of closure is in the field of topology, which describes a set as being closed if it contains the limit point of every possible sequence of elements.

A more easily-grasped example is given by the property of *convexity*. A set S is said to be convex if for any two points A and B in S, the straight line AB lies entirely within S. The *convex closure* of S is formed by taking the initial set and all such straight lines, this being the smallest convex set containing S. Figure 1 shows two simple non-convex regions and their convex closures.

One of the best developed uses of such closure methods for obtaining stable conceptual representations is in Formal

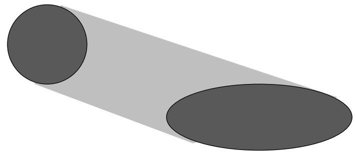

Figure 2: The convex closure of the union of two sets.

Concept Analysis, where the closure operation is given by the relationship between the intent and the extent of a concept (Ganter & Wille 1999, §1.1). An important closure operation we will consider later is the linear span of a set of vectors, which can also be thought of as the smallest subspace containing those vectors.

Ordering, containment, implication and disjunction

The link between these geometric structures and logical operations arises because the ordering relationship given by containment or inclusion can be used to model logical implication, an equivalence introduced by Aristotle:

> That one term be included in another as in a whole is the same as for the other to be predicated of all the first.
>
> (*Prior Analytics*, Bk 1 Ch. 1)

That is to say, suppose the set A represents those situations where the assertion $\tilde{A}$ is true, and B represents those situations where the assertion $\tilde{B}$ is true. If A is contained in B, then every situation in which $\tilde{A}$ is true is also a situation in which $\tilde{B}$ is true, which is equivalent to saying that $\tilde{A}$ implies $\tilde{B}$. The similar treatment of classes of objects and propositions about them was pioneered by Boole (1854, Ch. 4).

Containment (between two sets) and implication (between two logical statements) are both *ordering* relationships, so both geometric regions and logical assertions have the basic structure of an *ordered set* (Davey & Priestley 1990, Ch. 1). The disjunction of two assertions (or their corresponding geometric regions) is given by their least upper bound in this ordered set. That is to say that for two statements $\tilde{A}$ and $\tilde{B}$, their disjunction $\tilde{A} \vee \tilde{B}$ is the most specific statement implied by both $\tilde{A}$ and $\tilde{B}$, which corresponds geometrically to the smallest region containing both A and B.

Now, suppose that concepts in our geometric model are represented only by *convex* sets. Then the least upper bound or disjunction of two concepts will be represented by the smallest convex set which contains them both, which is the convex closure of their set union (Figure 2). Note the similarity between Figure 2 and the convex closures in Figure 1, the only difference being that the initial (dark gray) set is no longer connected.

The resulting convex closure contains points which are in neither of the two original sets, and so this disjunction operation fails to obey the Boolean *distributive law*.[1] In this way, the behavior of concepts under logical connectives is determined by the closure condition, used to distinguish those regions which are cohesive enough to represent concepts from those which are not. The physical and philosophical consequences of relaxing the distributive law are discussed by Putnam (1976).

In language, we often encounter sets which do not correspond to any lexicalized concept. For example, there is no word in English for the set consisting of all *rabbits*, *pigs* and *dolphins*, and the most specific word which refers to all of these creatures (*mammals*) also refers to many other species. In this way, the concept *mammal* can be thought of as the disjunction of *rabbits*, *pigs* and *dolphins* in a concept lattice, and this disjunction does not obey the distributive law because there are many *mammals* which are not *rabbits*, not *pigs*, and not *dolphins*.

We have so far demonstrated that representing concepts using only those sets that satisfy some closure condition leads to the use of logical operators that violate classical Boolean assumptions such as the distributive law, and that there are linguistic situations where this divergence from the classical theory offers a reasonable interpretation, since many possible sets do not correspond to concepts represented in the lexicon. For more information on geometric ordering and its relationship with disjunction in a concept lattice, see Widdows (2004, Ch. 1,8).

Inductive bias for machine learning and non-distributivity

Machine learning algorithms are in various ways dependent on the *inductive hypothesis*, which more or less states that the situations we will encounter in the future are coherent with those we encountered in the past. However, we do not expect to encounter situations in the future which are *identical* to those we have encountered in the past (cf. the dictum of Heraclitus, "you can never step into the same river twice"), and so some means must be found for correctly interpreting future observations based on the general features they have in common with previous encounters.

In practice, this means that we always assume that the training data for any learning algorithm is incomplete, and the algorithm has to generalize from this training data to new situations. The method of generalization depends on the *inductive bias* of the algorithm, which is the set of premises which, when combined with the training data, provide deductive statements in novel situations (Mitchell 1997, §2.7).

Inductive bias can thus be viewed as a kind of closure operation on the set of training examples. Insofar as the training examples provide a set of different situations *any* of which are possible, the final classifier learned from the training data combined with the inductive hypothesis can be thought of as a disjunction, whose arguments are given by the training examples and whose inductive bias is given by the closure condition needed to smooth out the training examples to form a cohesive conceptual representation. For example, Li and Abe (1998) use the closure condition that

[1] Boole's distributive law states that for all classes x, y and z, $z(x + y) = zx + zy$ (Boole 1854, Ch. 2, eq. 4). In modern notation, the distributive law reads $z \wedge (x \vee y) = (z \wedge x) \vee (z \wedge y)$ (Davey & Priestley 1990, Ch. 4).

any cohesive set of nouns is given by a tree-cut in a noun hierarchy, and use the minimum description length algorithm to approximate such tree-cuts from training data in order to predict what kinds of nouns are usually taken as arguments by different verbs.

Introducing inductive bias therefore dispenses with the distributive law, which would impose the condition that the disjunction of the training examples should simply be the set of all training examples, preventing the generalization to new situations. Learning by induction is thus incompatible with Boolean logic. (In fairness, Boole's own examples in *The Mathematical Analysis of Logic* (1847) and *The Laws of Thought* (1854) are almost entirely of a deductive nature, and he does not seem to have proposed that the scheme be used for learning premises from empirical observations.) Since Boolean logic is not well-suited as a model for learning from incomplete experience, it is incumbent upon us to find more appropriate models for learning and building classes. As this paper demonstrates, non-distributive models *can* generalize effectively from partial data, while retaining the deductive power of logical representations.

Quad-tree representations in spatial databases

Geometric ordering relationships are important for building and querying spatial databases. Geographic applications including map-servers, gazetteers and local business directories need to support a variety of spatial queries, such as

- Find objects near to the point x.
- Find objects of a given type in region A.
- Find a path joining points x and y.

this last example also implying that the query

- Find paths in a region that includes points x and y.

must be addressed.

The notion of spatial ordering is naturally entailed in such examples. Since there are an infinite number of possible 'regions' of different shapes and sizes, it is never possible to keep an index for every definable region, and the shapes and granularities of regions for which special indexes are precomputed and stored must be chosen in advance. Design goals typically include maximizing the speed and flexibility for the user while minimizing storage and update requirements. Such goals entail construction around Aristotle's predication-inclusion principle almost inexorably: if region B is contained in region A, then a predicate (defined by a user query) satisfied by an object in region B will naturally be satisfied by the same object in region A. It follows that an index kept for the subregion B must be accessible to an index kept for the region A — in other words, the indexing structure must reflect the spatial ordering of containment.

Quad-tree representations

Quad-trees provide a simple representation for such indexes. A quad tree is a tree in which each node has exactly four children. In spatial database (Rigaux, Scholl, & Voisard 2002) and computer graphics (Foley *et al.* 1990) applications, the nodes are taken to be rectangular areas, and a node's children are computed by bisecting the parent along each axis. The child nodes' areas thus form an equal partition of the parent node's area. (Quad-trees have an obvious generalization in higher dimensions: the number of children at each level is equal to 2^N, where N is the number of dimensions.)

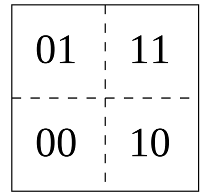

Figure 3: A parent cell in a quad-tree and its four indexed children

Quad-trees are an example of an index driven by spatial decomposition. While this approach can waste storage space when compared to data driven approaches such as R*-trees and their variants (Beckmann *et al.* 1990), the quad-tree has some attractive characteristics. One is its implementation simplicity. Another is that when used as an index for a spatial database, index keys can be computed directly and without prior reference to the database. We have found that this property is particularly handy in a distributed index where many disconnected agents may be performing index updates simultaneously. (Note that this sense of "distributed" is unrelated the Boolean distributive law. We mean here a database that is implemented across many cooperating computers, with possibly intermittent network connectivity.) The simplicity, stability, and predictability of the quad-tree structure makes distributed updates much more feasible.

Index keys are computed by describing the path through the tree to the appropriate node. The empty string is used to denote the root node in the tree, which corresponds to the total space being indexed. We label each of the children 00, 10, 11, and 01, as depicted in Figure 3. We can extend these labels recursively in the obvious fashion: 1100 represents a node two levels deep: the first child of the third child of the root. Such a node has an area $1/16$ the size of the root node. Three levels deep, there are 4 top level quadrants, 16 second level quadrants, and 64 third level quadrants. Some of the available quadrants are labeled in Figure 4, along with their binary index keys in the quad-tree. Note that long index keys correspond to small regions, and vice versa.

Thought of as a path in a tree, the choice of labels is fairly obvious. To be useful in an application context often requires the ability to compute these labels (and hence keys) on the basis of Cartesian coordinates. Imagine a mobile user equipped with a small computer or PDA and a GPS. This user may wish to update a spatial index with information about her location, expressed in latitude and longitude from the GPS, but may not have access to the full index since she is disconnected from a network. Fortunately, knowing only her latitude and longitude (to some finite precision) she can compute an index key.

The algorithm is as follows: for any set of bounded Carte-

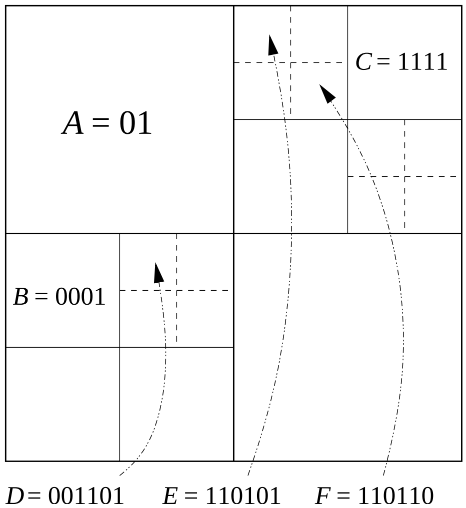

Figure 4: Addresses for smaller and smaller cells are generated by extending the addresses of their parents.

sian coordinates (such as latitude and longitude), normalize the coordinates so that they lie in $[0, 1)$. Each coordinate is then a fraction. Convert the decimal representation of each fraction to its binary representation. Interleave the bits of the two sequences with the x coordinate preceding the y coordinate. The resulting string is the index key, or to put it another way, the path to the correct node in the spatial tree, or to think of it yet another way, an alternating series of bifurcations in space that result in a particular area.

Any finite sequence of this form represents an area. (Points cannot be finitely represented. This is generally acceptable in real world applications since no real measuring device has perfect precision.) It should be noted that while all finite binary sequences correspond to areas in this model, not all areas correspond directly to binary sequences. We return to this rigidity later.

Joins or disjunctions in quad-trees

From an application standpoint we must often find the smallest node in the tree that completely encloses a given set of points. This is useful for queries like, "What are all the objects of interest within 5 miles of my present location?" In many spatial index structures this requires interrogating many levels of the index and calculating many distances. Because of the stable structure of our index, we can do it entirely algebraically, based on the structure of the keys.

For example, suppose we wish to compute the index key of the smallest cell that contains both the cells $E = 110101$ and $F = 110110$ in Figure 4. This is given by the string 1101 representing the second-level quadrant that contains both E and F. This can be written algebraically using the join notation

$$E \vee F = 110101 \vee 110110 = 1101.$$

Similarly, the join of B and D is found to be

$$B \vee D = 0001 \vee 001101 = 00.$$

In both cases, the join is found algebraically by taking the longest common prefix (of even length) of the two index keys. This method turns out to be true in general, as follows.

Theorem 1 *For any set of index keys, the index key of the smallest area containing the areas of all the given index keys is given by the longest common prefix (of even length) of the given keys.*

Proof. For any region A, the index keys of the children of A are formed by adding a suffix to the index key of A. For any cell B such that $B \subset A$, it follows that A's index key is a prefix of B's index key. Since each node in the quad-tree has a unique path to the root node, the converse (if B is not contained in A, the index key of A can not precede that of B) also holds. The cells in the quad-tree (ordered by geometric inclusion) and their index keys (ordered lexicographically by the prefix ordering) are therefore isomorphic as ordered sets.

The 'smallest containing area' gives the least upper bound of two regions under the geometric ordering, and the 'longest common prefix' gives the least upper bound of two index keys under the lexicographic prefix ordering. Since order-isomorphism preserves least upper bounds, it follows that the smallest containing area under the geometric ordering corresponds to the longest common prefix of the index keys.

The extra condition that this prefix be 'of even length' prevents the inclusion of regions whose x-coordinates coincide but whose y-coordinates do not (since we are using a binary alphabet to model a 4-way branching tree). In a higher dimensional space, the extra condition is: 'of length that is a multiple of the dimensionality.' ∎

This claim may not seem intuitive at first, but it becomes quite obvious when it is recalled that the keys are paths in a tree. Each of the given index keys represents a path to a node in the tree. Since every node of the tree contains all its children, the smallest node containing each of the given nodes is the one that lies furthest along the common portion of the paths in tree. Because of the way we labeled paths, it must be identical to the longest common prefix of the keys. There is always such a path, since the root node contains every node. There can be no smaller node that contains all the children, because the space is partitioned with no overlap at every level of the tree.

Because we find an area which must contain the given areas, but may also contain other points, we have another non-distributive disjunction. When representing a collection of disconnected objects, a distributive Boolean disjunction may be appropriate, but in geographic situations requiring us to display a connected *region* containing the desired objects, non-distributive logical structures are much more natural.

Significance of this approach

We believe it is important to consider this method in a historical context. None of the ingredients are new, but the combination is novel and powerful.

The method of using numbers to represent points in the plane is due largely to Descartes (1598-1650). The adaptation of such methods to describe *regions* as well as points will be addressed further in the next section. While Cartesian coordinates use tuples of numbers, since the 19th century the discovery of *space-filling curves* (eg. by Peano and Hilbert) makes it possible to represent parts of the plane using a *single* number, as we have done here. These parts are single points only in the theoretical limit (where the representation may be of infinite length), and in practice are usually cells of a uniform size (unlike the quad-tree, which naturally represents cells of several nested sizes). Space-filling curves are useful for building spatial indexes (Rigaux, Scholl, & Voisard 2002, pp. 221-223) though they can never be completely faithful to the geometric structure of the space being indexed.

Another important influence on our method is the use of numbers to represent different levels of *specificity* rather than different magnitudes. This owes much to the practical work of Stevin (1548-1620) and Napier (1559-1617), who introduced and popularized the decimal point as a way of using Hindu-Arabic numerals to represent numbers in the range $[0, 1)$ with unlimited specificity. By convention in the practical sciences, such numbers often refer to regions, the appropriate size of which is determined by the accuracy and precision of available measuring techniques: for example, it is taught that the decimal 0.6 should be used to represent measurements in the region $[0.55, 0.65)$ to one significant figure. If we adopt the convention of rounding down rather than rounding to the nearest significant figure so that 0.6 represents the interval $[0.6, 0.7)$, we recover the practice of using the prefix ordering to give a tree-structure on regions in the real line of different levels of specificity.

The quad-tree indexing structure described in this section combines these two important ideas, using representations of numbers of increasing specificity as coordinates and interleaving these coordinates to give a spatial indexing structure for any compact region of N-dimensional space. With this structure, the geometric ordering of regions of space is naturally encoded by the lexicographic ordering of the index keys.

Examples such as these show that the Cartesian synthesis of algebra and geometry can be naturally extended to incorporate *logical* operations within a common theoretical and practical framework. The logic of such structures will often be non-Boolean, not because Boolean logic is 'wrong,' but because Boole's decision to limit the possible values of coordinates to 0 and 1 (Boole 1854, p. 37) is appropriate for modeling some situations and not others.

In particular, Boole's choice of discrete binary values leads to the distributive law because the disjunction of 0 and 1 becomes the discrete set $\{0, 1\}$. A change to continuous values is represented by a mere change in bracket symbols to give the interval $[0, 1]$, which represents a non-distributive disjunction of 0 and 1. That one of these operations is considered part of standard logic and the other is considered part of analysis or geometry is for historical reasons. Our quad-tree index successfully combines logic and geometry, not by inventing a complicated new bridge between them, but by ignoring this artificial historical division.

Vector lattices and composition

The quad-tree of the previous section has several drawbacks. For example, two points may in fact be close to one another, but if they are on opposite sides of the boundary separating two top-level cells, then the whole of the space will be returned as the quad-tree disjunction of these points. (For example, in Figure 4 the disjunction $001111 \vee 110000$ has empty common prefix and returns the whole space.) While some of these problems are peculiar to the quad-tree methodology, similar insensitivity is a common feature of tree structures generally. If there is no mechanism for multiple-inheritance (and therefore cross-classification), the closure condition on a collection of leaf-nodes is often very strong and may return an uninformative generalization.

Such taxonomic classification is in some ways an opposite extreme from the Boolean set-theoretic model. Boolean logic permits *any* collection of objects to represent a concept (technically, to be the *extension* of a concept; see Ganter & Wille (1999, Ch. 1)), so there are no closure conditions at all. Adding any single element v to a subset U (assuming that $v \notin U$) creates a set $U \cup \{v\}$, and since $U \subset U \cup \{v\}$, any set of the form $U \cup \{v\}$ is a parent of the set U under the inclusion ordering. At the other extreme, taxonomic classification allows each node in the tree to have only a *single* immediate parent, and thus a unique path up to the root node.

In this final section, we describe the closure conditions and ordering in vector spaces and their subspaces, as a fertile middle ground between the Boolean and taxonomic extremes. Vector spaces (sometimes called 'linear spaces') are generalizations of (and include) the traditional Euclidean notions of lines and planes (in particular, lines and planes through the origin).

The standard closure condition in a vector space V is given by the property of *linearity* — a subset $U \subseteq V$ is considered to be a *vector subspace* of V if for all a and b in U, the linear combination $\lambda a + \mu b$ is also in U. The simplest subspaces are lines and planes which pass through the origin. The corresponding closure operation on a set of vectors is to take their linear span,

$$\mathrm{span}\{a_1, \ldots a_n\} = \{\lambda_1 a_1 + \ldots + \lambda_n a_n\}.$$

For example, the linear span of the lines OA and OB in Figure 5 is the plane OAB. Thus the plane OAB is the disjunction of the lines OA and OB. Using the vector sum for disjunction, the intersection for conjunction, and the orthogonal complement for negation, the collection of subspaces of a vector space naturally forms a logic, which has been used especially to model experimental postulates in quantum mechanics (Birkhoff & von Neumann 1936; Putnam 1976).

Vector space models for information are a standard part of information retrieval (Salton & McGill 1983; Baeza-Yates & Ribiero-Neto 1999). However, vector models in information retrieval have mainly treated words and documents as points rather than regions (though as stated, the

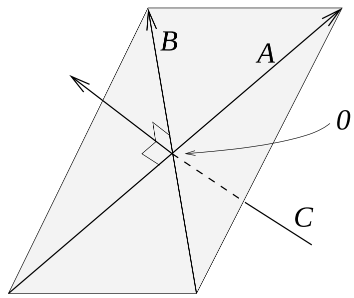

Figure 5: Lines and planes in a vector space

process of clustering and classifying these points leads to a consideration of regions in the space (Schütze 1998)). The drawback of using only this most basic unit is that no point is contained in any other, and the logical structures based upon geometric containment that we have discussed in this paper are not exploited. Using the lattice of closed subspaces is an appealing alternative since it gives a natural ordering on concepts: so far, experiments have shown that unwanted content can be removed from search results more effectively by representing the unwanted content as the vector disjunction (plane spanned by) the unwanted keywords, rather than treating these keywords as separate points and removing their average (Widdows 2003; 2004, Ch. 7).

The use of linear models in physics has for a long time been much more sophisticated, and is becoming gradually unified in a combined *geometric algebra* (Lasenby, Lasenby, & Doran 2000). This is based upon the works of Grassmann (1809–1877) and Clifford (1845–1879), who developed a product operation on vectors which can be used to build larger representations from primitive components. For example, the *outer product* of two vectors a and b is the *bivector* $a \wedge b$ which represents the directed region swept out by a and b. This has the useful property that $b \wedge a = -a \wedge b$, so these operators form a useful counterpart to traditional commutative compositional operators such as vector addition or disjunction.[2] A related operation on vectors is the *tensor product* $a \otimes b$, used to represent the interaction between particles in quantum mechanics, and used by Plate (2003) for composition in connectionist memory models,

Vector models provide mathematically sophisticated tools which combine the geometric and logical approaches for representing and combining concepts. As a classification scheme, they take a middle road: the closure conditions on a linear space can be used to generate well-formed concepts, but since a line can belong to many different planes, a rich structure of cross-classification arises naturally.

[2] The outer product operator $\wedge$ should not be confused with the logical conjunction, and indeed it behaves more like a directed disjunction. The precise relationship between the composition operations of geometric algebra and those of quantum logic, and developing a suitable notation for combining these, may be a useful research topic.

Acknowledgments

This work was supported by Defense Advanced Research Projects Agency grant SB031-008 for the Cluster-Based Repositories and Analysis project, and by European Community / National Science Foundation grant IST-1999-11438 for the MUCHMORE project.

The first author would like to thank Pentti Kanerva for challenging, stimulating and encouraging conversations.

References

Baeza-Yates, R., and Ribiero-Neto, B. 1999. *Modern Information Retrieval*. Addison Wesley / ACM Press.

Beckmann, N.; Kriegel, H.-P.; Schneider, R.; and Seeger, B. 1990. The R*-tree: An efficient and robust access method for points and rectangles. In *SIGMOD Conference*, 322–331.

Birkhoff, G., and von Neumann, J. 1936. The logic of quantum mechanics. *Annals of Mathematics* 37:823–843.

Boole, G. 1847. *The Mathematical Analysis of Logic*. Macmillan). Republished by St Augustine's press, 1998, introduction by John Slater.

Boole, G. 1854. *An Investigation of the Laws of Thought*. Macmillan. Dover edition, 1958.

Davey, B. A., and Priestley, H. A. 1990. *Lattices and Order*. Cambridge University Press.

Foley, J. D.; van Dam, A.; Feiner, S. K.; and Hughes, J. F. 1990. *Computer Graphics*. Addison Wesley.

Ganter, B., and Wille, R. 1999. *Formal Concept Analysis: Mathematical Foundations*. Springer.

Gärdenfors, P. 2000. *Conceptual Spaces: The Geometry of Thought*. Bradford Books MIT Press.

Landauer, T., and Dumais, S. 1997. A solution to Plato's problem: The latent semantic analysis theory of acquisition. *Psychological Review* 104(2):211–240.

Lasenby, J.; Lasenby, A.; and Doran, C. 2000. A unified mathematical language for physics and engineering in the 21st century. *Phil. Trans. R. Soc. Lond.* A 358:21–39.

Li, H., and Abe, N. 1998. Generalizing case frames using a thesaurus and the MDL principle. *Computational Linguistics* 24(2):217–244.

Mitchell, T. 1997. *Machine Learning*. McGraw-Hill.

Plate, T. 2003. *Holographic Reduced Representations: Distributed Representation for Cognitive Structures*. CSLI Publications.

Putnam, H. 1976. The logic of quantum mechanics. In *Mathematics, Matter and Method*. Cambridge University Press. 174–197.

Rigaux, P.; Scholl, M.; and Voisard, A. 2002. *Spatial Databases*. Morgan Kauffmann / Academic Press.

Salton, G., and McGill, M. 1983. *Introduction to modern information retrieval*. McGraw-Hill.

Schütze, H. 1998. Automatic word sense discrimination. *Computational Linguistics* 24(1):97–124.

Widdows, D. 2003. Orthogonal negation in vector spaces for modelling word-meanings and document retrieval. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*.

Widdows, D. 2004. *Geometry and Meaning*. CSLI Publications.