Using Venn Diagrams to Perform Logic Reasoning: An Algorithm for Automating the Syllogistic Reasoning of Categorical Statements

Robbie T. Nakatsu*
Department of Computer Information Systems, Loyola Marymount University, One LMU Drive, Los Angeles, CA 90045

I describe a Venn diagramming technique used to perform syllogistic reasoning on categorical statements. The notation uses overlapping circles to represent relationships among two or three sets, shadings to represent emptiness, and x sequences to represent nonemptiness. These notations allow one to easily visualize logic problems. I then discuss rules of manipulation that can be used to transform one Venn diagram into another valid Venn diagram. These rules provide us with a formal procedure for performing syllogistic reasoning—that is to say, they provide us with an algorithm for proving or disproving the validity of a syllogism. I extend the Venn diagramming algorithm for syllogistic reasoning to allow for more than three sets of information at a time. This technique makes use of tables, which is also very intuitive and highly visual. The tabular technique described is capable of processing a much larger variety of logic statements. © 2013 Wiley Periodicals, Inc.

1. INTRODUCTION

The use of diagrams in formal logic reasoning has generated a great deal of interest in recent years due to the need to visualize complex logic problems that are difficult to understand. By diagram, I am referring to a graphical representation of how objects in a domain are interconnected or interrelated to one another.[1] In logic reasoning, the central diagram in question is the Venn diagram, a well-known diagramming technique that is used to graphically represent relations among sets. In this paper, I explore in greater depth the Venn diagram, and how it can be used to construct logic proofs for a class of problems in formal logic. From this discussion, a general algorithm is developed to automate the reasoning for a class of problems in logic, namely syllogistic reasoning involving categorical statements.

*Author to whom all correspondence should be addressed: e-mail: rnakatsu@lmu.edu.

INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS, VOL. 29, 84–103 (2014)

View this article online at wileyonlinelibrary.com. • DOI 10.1002/int.21628

The algorithm is powerful enough to work on more than three sets of information, unlike Venn diagrams, which are limited to representing at most three sets at a time.

In general, the use of diagrams in computer systems is an important topic today because of its potential in rendering systems that are more transparent and flexible to use. A common complaint of many systems today is that they are black boxes that are hard to understand and use. The use of diagrams and other graphical representations has been shown to foster more effective and efficient problem solving.[2–5]

A seminal paper in the literature on diagrammatic reasoning is the one by Larkin and Simon.[2] Although they define diagrams more broadly and apply it to solving problems in physics and proving geometry theorems and not to logic reasoning, their ideas are quite relevant to the discussion. They compare and contrast diagrammatic representations with what they refer to as sentential representations. A sentential representation is linguistic, and its expressions correspond to the sentences of a natural-language description of a problem. In the logic reasoning problems discussed in this paper, the sentential representation would typically be described by the notations and syntax of first-order logic, whereas the diagrammatic representation of choice would be a Venn diagram. Larkin and Simon note that the essential difference between the two types of representations is that sentential representations are sequentially organized, whereas diagrammatic representations are indexed by location on a plane.

While the applications of diagrammatic reasoning in the cognitive sciences focus on how to support learners in complex tasks typically with paper-based or more "static" diagrams,[6,7] in AI (artificial intelligence), the applications more typically involve how to program a computer to perform these tasks for you.[8] This is an important point to underscore because in order for diagrammatic reasoning to work in AI programs, the techniques not only need to be expressed in a visual way, but also need to be formalized. Harel[9] refers to this as "visual formalism"—visual because they need to easily manipulated and comprehended by human users; and formal, because they need to be automated by computer programs. Hence, the objective of this paper is to demonstrate a formal system for performing automated syllogistic reasoning for a class of logic problems using Venn diagramming.

Yet, to some traditional logicians, the use of diagramming techniques as a formal system is viewed with suspicion. Such logicians believe that diagramming can be used as a heuristic tool only, and should not serve as a formal system of proof. Such has been the thinking in geometry, in which certain invalid proofs were thought to be the result of the misleading diagrams that accompanied them.[10] So, the thinking went that the diagram should be discarded once the proof was found, and a "real" proof should be expressed in sentential form only. The work of Shin,[11,12] however, shows that this denigration of diagramming is misguided. She gives a rigorous analysis of Venn diagrams as a formal system, showing that its rules of inference are sound. In addition, she also proves completeness for the system. The importance of her work is that she shows that one can have a rigorous, logically sound, and complete formal system entirely based on diagrams.

2. VENN DIAGRAMMING OF CATEGORICAL STATEMENTS

In this section, I review the logic of categorical statements and introduce the notation for Venn diagramming. I then show how the categorical statements are expressed using this Venn notation.

2.1. The Logic of Categorical Statements

Here is an example of a syllogism containing categorical statements:

Some polygons are squares.
All squares have four sides.
∴ Some polygons have four sides.

A syllogism is an argument in which a conclusion is inferred from its premises. The syllogism is valid when the conclusion cannot be false when its premises are true. The reader can easily recognize the above syllogism as being valid.

The above syllogism is a specific type known as a *categorical syllogism*, a type of syllogistic reasoning that has been studied extensively in the field of logic. Categorical syllogisms are two-premise arguments that consist entirely of categorical statements of the following four distinct forms:

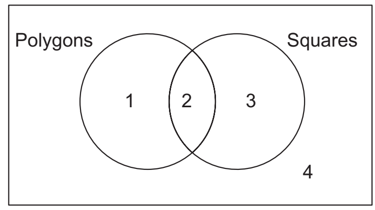
Form

A	All S are P.
E	No S are P.
I	Some S are P.
O	Some S are not P.

Under this scheme, S stands for the subject term, and P for the predicate term. A, E, I, and O denote the specific form. (In addition, to qualify strictly as a categorical syllogism, the argument must consist of three class terms: the subject and predicate terms in the conclusion, and a third term called the middle term that occurs in both premises).[13]

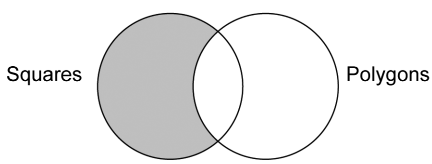
2.2. Venn Diagramming Notations

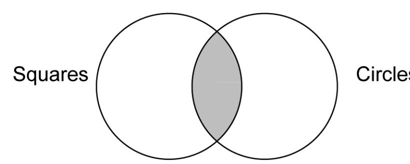
Let us begin with some basics of Venn diagramming. Suppose that we want to illustrate the relationships between the set of polygons and the set of squares. We begin by drawing two overlapping circles. Furthermore, we enclose the overlapping circles with a rectangle to represent the background set. In this example, the background set is the set of all closed figures.



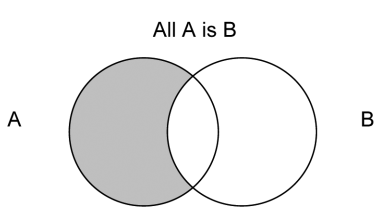
Each compartment (as numbered in the above Venn diagram) represents a possible combination of the two sets. For example, compartment 1 is the set of polygons that are not squares; compartment 2 is the set of polygons that are squares. (In the subsequent examples, the rectangle is omitted, as the background set is of no interest to the discussion).

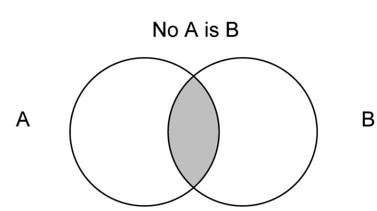
We further adopt the convention of shading those compartments that represent nothing, or the empty set.[14] For example:





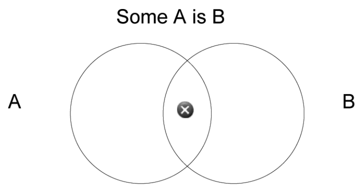
Because all squares are polygons, the compartment that represents squares that are not polygons is shaded. In a similar fashion, because no squares are circles, the compartment that represents the intersection between squares and circles is shaded. In general, we can represent *All A is B* (A form) and *No A is B* (E form), as follows:

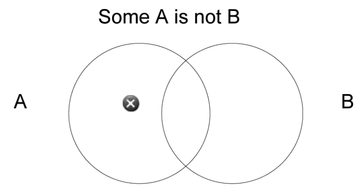




The A form and E form, which use the "all" and "no" quantifiers, are known as the universal statements.

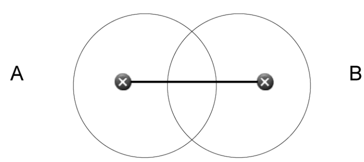
But how do we represent *Some A is B* (I form) and *Some A is not B* (O form)? It is clear that we need a new syntactic device to represent the fact that something exists in a particular compartment on a Venn diagram—that is, we require notation to denote a nonempty compartment. For this, we add an $\otimes$ to the compartment where there exists at least one occurrence. This notation is from Peirce.[15] Doing so yields the following two Venn diagrams:





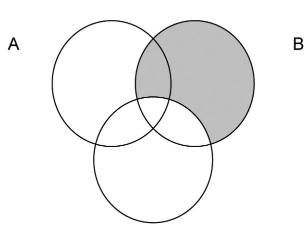
In the first diagram, we add an ⊗ to the compartment that represents the existence of at least one thing that is both A and B (A ∧ B). In the second diagram, we add an ⊗ to the compartment that represents the existence of at least one thing that is A, and not B (A ∧ ∼B). The I form and O form, which use the "some" quantifier, are known as existential statements.

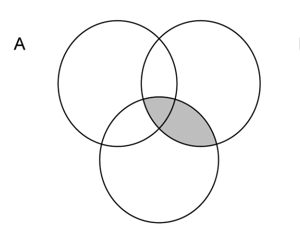
We now have a technique for representing many kinds of propositions on Venn diagrams, both universal statements and existential statements. To make our representational system even more expressive, we also need a way to represent disjunctive information. That is to say, we would like to express the fact that either one of two existential statements (or both) is true. The syntactic device we adopt is a line that connects the ⊗s. For example, suppose we wish to represent the statement: (*Some A is not B*) or (*Some B is not A*). We can do this by connecting the two ⊗s as the following diagram illustrates:



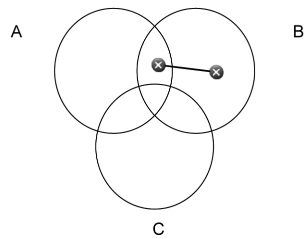
A series of interconnected ⊗s, linked to one another like a chain as an x sequence.[15] This notation is adopted from Shin's Venn–Peirce diagrams.[12]

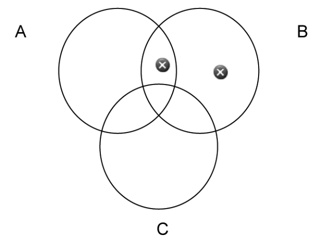
Like in the two-class Venn diagram case, three-class Venn diagrams also use shading to denote the empty set and an ⊗ to denote a nonempty set. The following three-class Venn diagrams represent the universal statements: (1) *All B is C* (diagram on the left), and (2) *No B is C* (diagram on the right).





Representing the existential statement *Some B is not C* requires that we specify a disjunction of ⊗s in two compartments.





The correct specification of this statement is given by the left diagram. This diagram represents the fact that there could exist an occurrence in either one of the two compartments or in both in order for *Some B is not C* to be true. The diagram on the right does not accurately represent this statement because it represents the fact that there is at least one occurrence in each of the two compartments. For more information on these Venn diagramming notations, see Ref. 1.

3. SYLLOGISTIC REASONING WITH VENN DIAGRAMS

Let us begin this section with a formal procedure for testing the validity of a syllogism. (A similar procedure is found in references by Nakatsu[1] and Shin,[12] but has been modified somewhat in this discussion).

1. Draw a separate diagram that represents the facts that each premise of a syllogism conveys. (We are not limited to just two premises, but can have any number of premises).
2. Unify each of the diagrams created above into one diagram. We will call this unified diagram D_P (see Section 3.1).
3. Resolve any conflicts on the unified diagram (see Section 3.2).
4. Draw a diagram that represents the facts that the conclusion of a syllogism conveys. We will call this diagram D_C. Check whether we can "read off" diagram D_C from D_P. If we can, the syllogism is valid; if we cannot, the syllogism is invalid. By "read off," we mean that we can legitimately transform D_P into D_C (see Section 3.3).

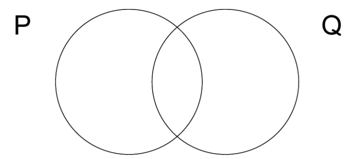
3.1. Unification Rules

How do we unify two separate Venn diagrams into a single diagram? Let us refer to D_1 and D_2 as the diagrams that contain the knowledge contained in two individual premises. The unified diagram will be referred to as D_P. The first unification rule enables us to represent sets in a single diagram.

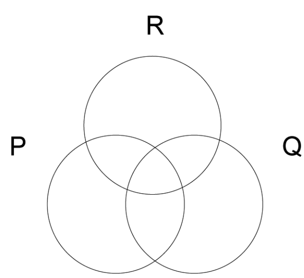
RULE U.1 (Copying circles to the unified diagram).

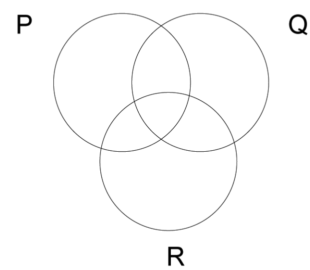
We first copy the circles from Diagram D_1. If there are circles on Diagram D_2 that do not exist in D_1, we must copy these circles as well, observing the partial overlapping rule. This rule states that: a new circle introduced onto a Venn diagram must overlap each and every compartment of that diagram once and only once.[12]

For example, suppose we have a two-set Venn diagram with P and Q, represented by two overlapping circles:



The following two Venn diagrams illustrate the correct addition of a third circle (in both cases, R overlaps each compartment in P and Q):





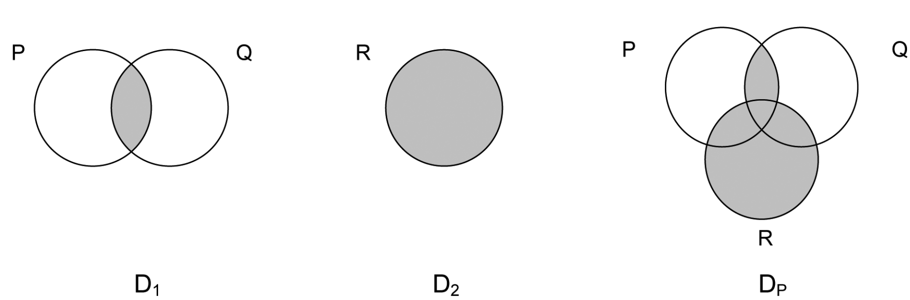
Of course, a limitation of Venn diagrams is that we cannot graphically represent more than three circles at a time. For $n > 3$ circles, we would need to construct a program that could enumerate all possible compartments of Venn diagrams containing more than three classes. In general, a Venn diagram with n classes will contain 2^n compartments. Later, in Section 4, I will discuss syllogistic reasoning involving more than three sets of information.

We also require unification rules that enable us to add shadings and x sequences to a unified diagram. The second and third unification rules enable us to add this information to the unified diagram.

RULE U.2 (Copying corresponding shadings to the unified diagram).

For any region shaded in D_1 or D_2, the corresponding region in D_p should be shaded.

Example 1



The second unification rule is relatively straightforward to apply. In the above example, we copy all the circles that are contained in D_1 and D_2, namely P, Q, and R, observing the partial overlapping rule. By Rule U.2, we then shade the corresponding regions in D_P that are also shaded in D_1 and D_2.

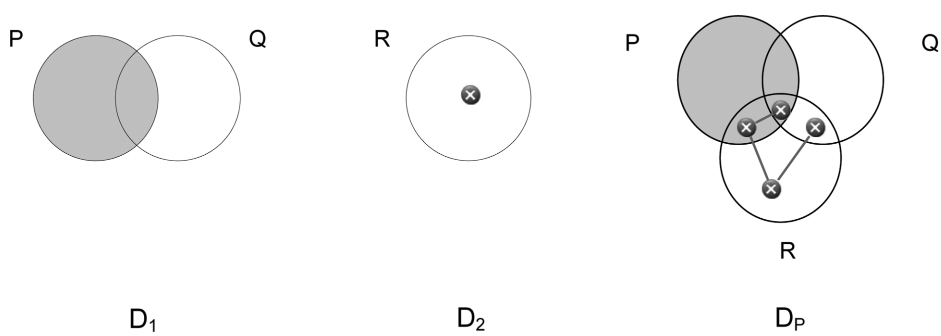
RULE U.3 (Copying corresponding x sequences to the unified diagram).

For any region with an x sequence in either D_1 or D_2, an x sequence should be drawn in the corresponding compartments of D_P.

RULE U.3.a (Inclusion of all compartments of a region in an x sequence).

If there is an x sequence in the original diagram, an ⊗ must be added to each new compartment that is created in D_P, which is part of the region containing the ⊗ in the original diagram. The ⊗s are joined together to form a single x sequence.

Example 2



We first copy all circles to D_P, namely P, Q, and R, observing the partial overlapping rule. By Rule U.2, we copy the shadings onto the unified diagram. By Rule U.3, we copy the x sequence from R. Because circle R contains four compartments in the unified diagram, we need to include four ⊗s joined together as one x sequence (Rule U.3.a).

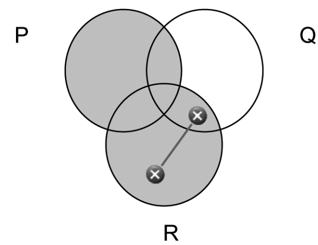
3.2. Resolution Rules

A conflict occurs when a compartment contains both an ⊗ and a shading. The first resolution rule indicates under what situations a diagram represents inconsistent information.

RULE R.1 (Inconsistent diagram).

If there is an x sequence in which each and every ⊗ is in a shaded compartment, then the diagram represents inconsistent information. The premises, which are represented on the diagram, need to be reformulated.

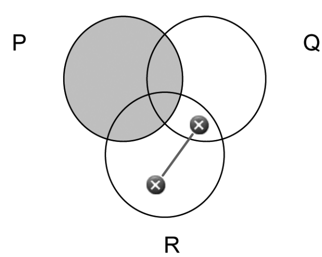
Example 3



The diagram in Example 3 illustrates a situation in which the x sequence is completely contained within a shaded area. Hence, this diagram represents an inconsistency that something cannot exist and be empty at the same time.

RULE R.2 (Erasure of part of an x sequence in a shaded compartment).
If the diagram is not inconsistent, as given by Rule R.1, then we may erase any ⊗ of an x sequence if that ⊗ is in a shaded compartment. If the ⊗ is in the middle of the x sequence, then after we erase the ⊗, we must connect both of the remaining parts so that we have one x sequence.

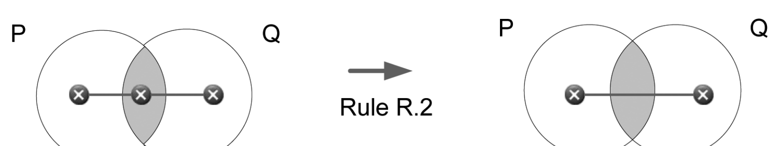
In Example 2 above, two ⊗s are contained in shaded compartments in diagram D_P. Rule R.2 allows us to eliminate these ⊗s, thereby resulting in the following diagram:



D_P

Sometimes, the eliminated ⊗ occurs in the middle of the x sequence. If this is the case, we need to connect the remaining parts together, as Example 4 illustrates.

Example 4



3.3. Transformation Rules

Once we have combined information from the premises into one diagram using the unification rules, and then resolved any conflicts using the resolution rules, the next step is to determine whether the conclusion, as represented by the diagram D_C follows from the premises, as represented by diagram D_P. If such a transformation is possible, then our syllogism is valid; otherwise it is invalid. Therefore, we require a set of transformation rules that will enable us to transform D_P into D_C.

There are basically two types of permissible transformations: one that erases information from the diagram, and one that allows us to extend an x sequence. Let us begin with the erasure rules.

RULE T.1 (Erasure of shading in a compartment).

We may erase the shading in a compartment.

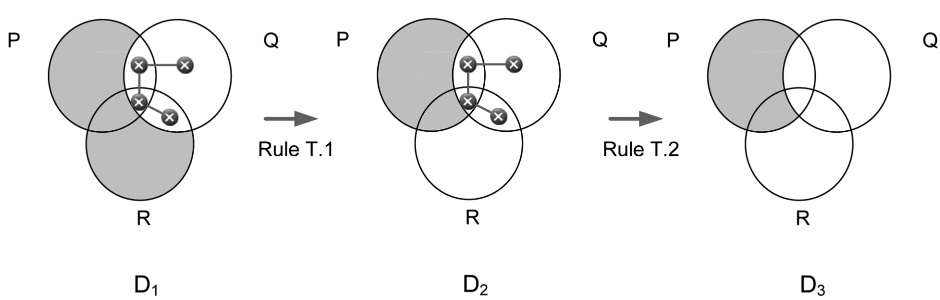
RULE T.2 (Erasure of a whole x sequence).

We may erase a whole x sequence.

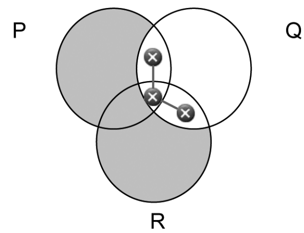
RULE T.3 (Erasure of a circle).

We may erase an entire circle.

Example 5



In the above diagram, we first erase a shading (diagram D_2), and then an entire x sequence (diagram D_3). We are permitted to do this because whenever D_1 is true, it is certainly the case that D_2 and D_3 are true. On the other hand, the following manipulation to D_1 is not permissible:



D_4

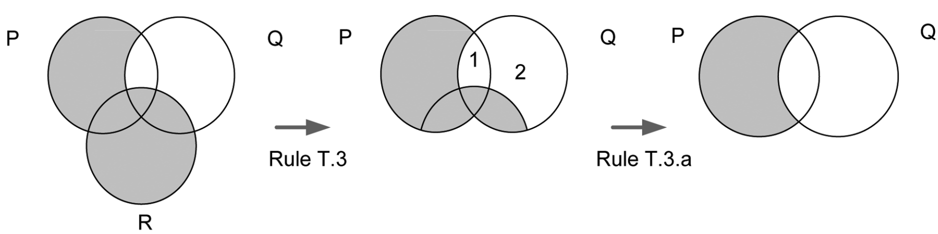
This is because we cannot erase a portion of an x sequence—we may only erase an entire x sequence. Hence, D_4 does not logically follow from D_1.

The erasure of an entire circle from a Venn diagram, permitted by Rule T.3, requires two corollary rules, which are stated below, and are followed by an illustrative example.

RULE T.3.a (A partial shading in a compartment).

When a circle is erased, and the resulting Venn diagram contains a partial shading in a compartment, then the shading in that compartment should be erased.

Example 6



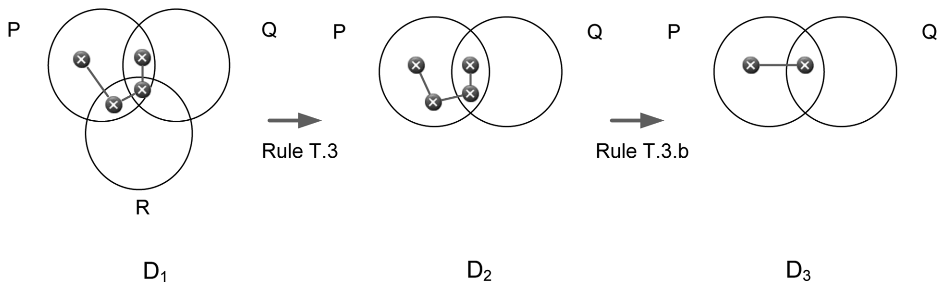
D_1 D_2 D_3

By Rule T.3, we may erase the circle R, resulting in partial shadings in compartments 1 and 2. By Rule T.3.a, these partial shadings are removed. The correct transformation is given by D_3.

RULE T.3.b (An x sequence with more than one ⊗ in a compartment).

When a circle is erased, and the resulting Venn diagram contains an x sequence in which there is more than one ⊗ in a compartment, then that part of the x sequence should be replaced by one ⊗ and should be connected to the rest of the x sequence.

Example 7

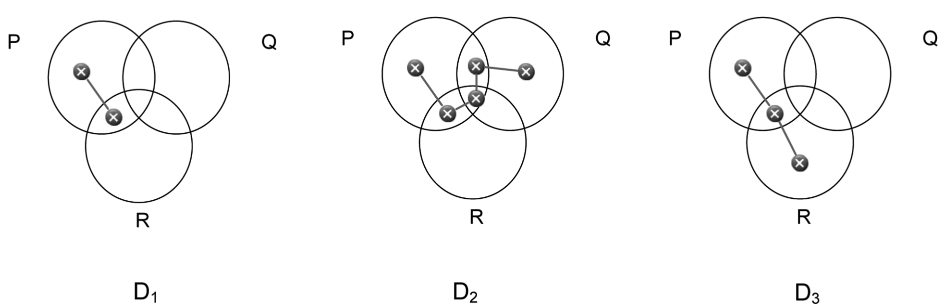


RULE T.4. (Lengthening an x sequence).

We may add any number of ⊗s to an existing x sequence.

This rule allows us to add any number of disjuncts to an already-existing statement. Thus, for example, if we know that A is true, then we can certainly say that (A or B) is true, or that (A or B or C) is true as well. Lengthening an x sequence is equivalent to adding disjuncts to what we already know to be true.

Example 8



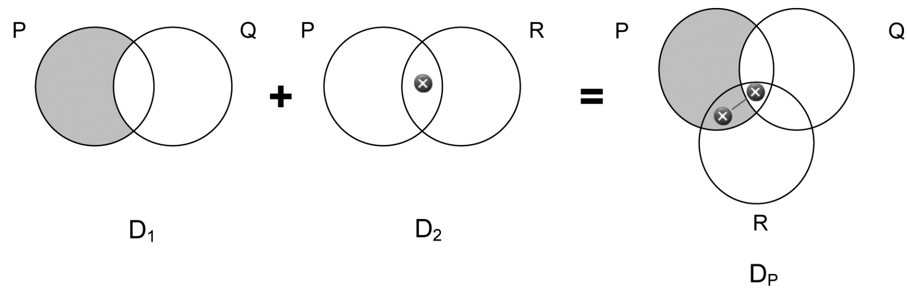
In Example 8, D_2 and D_3 follow from D_1. If D_1 is true, then it is always the case that D_2 and D_3 are true.

3.4. Proving the Validity of Syllogisms using the Rules of Manipulation

Now that I have described and illustrated all the rules of manipulation, let us now look at two examples on how to prove the validity of a syllogism.

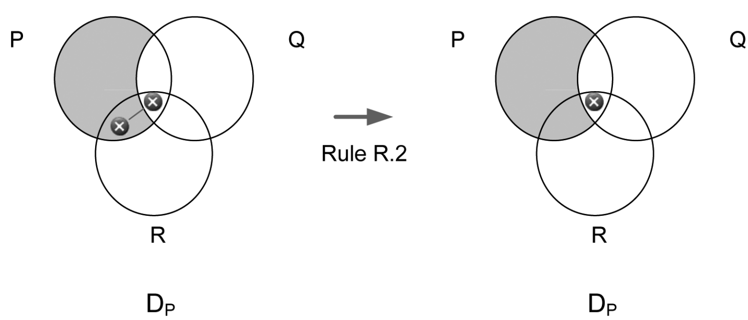
Example 9

P_1: All P are Q.
P_2: Some P are R.
∴ Some Q are R.

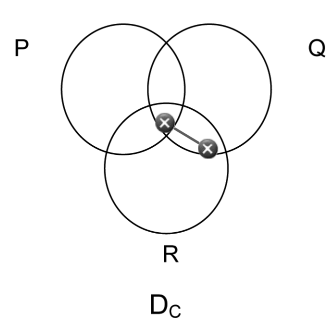


We begin by drawing separately the Venn diagrams that represent each of the premises—D_1 is the diagram for P_1 and D_2 is the diagram for P_2. By the unification rules, D_P is generated by adding the three overlapping circles (P, Q, and R) that represent the three sets used in P_1 and P_2. Furthermore, we copy over the corresponding shadings and x sequences into D_P.

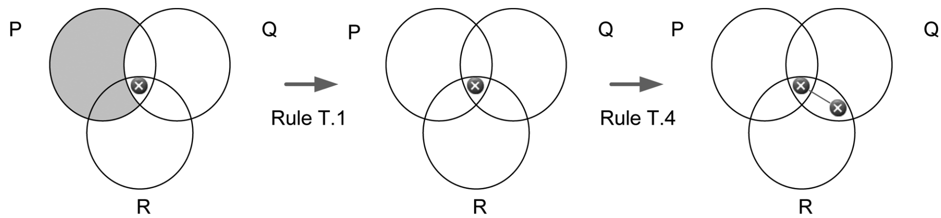
Resolution rule R.2 is then used to resolve any conflicts in which a shading and an $\otimes$ occurs in the same compartment as is indeed the case in D_P. By eliminating the $\otimes$, we now have the following unified diagram:



D_C, which represents the conclusion of the syllogism, is given next:



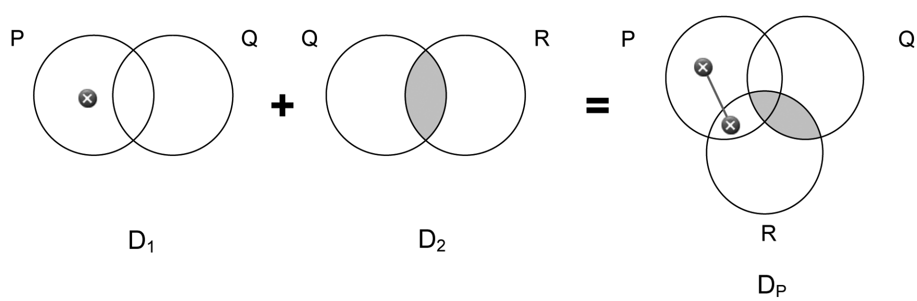
Can we transform D_P into D_C? If the transformation is possible, then our syllogism is valid.



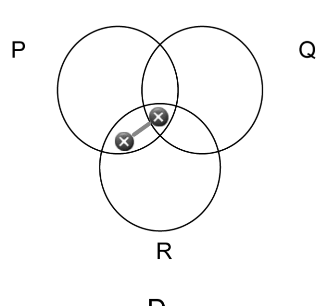
By first applying Rule T.1, we can transform D_P into the second diagram by eliminating the shading. By Rule T.4, we can lengthen an x sequence, so that we obtain the third diagram, which is precisely our goal D_C. Hence, we have proven the validity of the syllogism.

Example 10

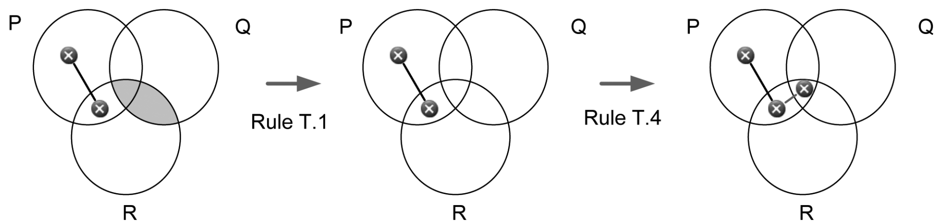
P_1: Some P are not Q.
P_2: No Q are R.
$\therefore$ Some P are R.



Once again, we begin by drawing separately the Venn diagrams that represent each of the premises P_1 and P_2. By the unification rules, D_P is generated by adding the three overlapping circles (P, Q, and R) that represent the three sets used in P_1 and P_2. Furthermore, we copy over the corresponding shadings and x sequences into D_P. This time, we do not need to use the resolution rules because there are no conflicts on D_P. D_C is given by the following diagram:



Using the transformation rules, we will now attempt to transform D_P into D_C:



First, by Rule T.1, we can eliminate the shading from D_P to obtain the second diagram. Then, by Rule T.4, we can lengthen the x sequence by adding another x (third diagram). However, we cannot transform this diagram any further toward D_C because that would entail eliminating the first ⊗ in the x sequence. This transformation is not permitted: remember that we can add ⊗s to an x sequence, but cannot subtract them out. Therefore, this syllogism is invalid.

4. A GENERALIZED ALGORITHM FOR SYLLOGISTIC REASONING

The preceding section suggests a procedure for syllogistic reasoning that can be applied to more than three sets, as well as used on a greater variety of categorical statements. While Venn diagramming is a useful technique for visualizing the relationships among sets, they are limited to three sets only. In this section, a more generalized algorithm is presented.

4.1. The Algorithm

For $n > 3$ sets, it is first necessary to enumerate all the combinations of overlapping sets. In general there will be 2^n combinations of n sets (or 2^n compartments). For example, for the four sets A, B, C, and D, we need to enumerate all combinations of (A, ∼A), (B, ∼B), (C, ∼C), and (D, ∼D). Table I shows the enumeration, which results in 16 possible compartments. In general, for n sets $A_1, A_2, \ldots A_n$, we would like to look at all combinations of A_j and $\sim A_j$ for sets $j = 1$ to n.

The procedure for testing the validity of a syllogism involves steps that correspond to those involving the manipulations on Venn diagrams as described in Section 3. Specifically, we need to do the following to prove the validity of a syllogism:

1. Draw a table that enumerates all compartments of n sets. We refer to this table as the Venn table.
2. Indicate on separate columns on the Venn table, the facts that each individual premise conveys.
3. Unify the facts, contained in each premise, in a single column (Unified column).
4. Resolve any conflicts and write the result in a separate column (Resolved column).
5. Indicate on a separate column on the Venn table, the facts that a conclusion conveys.

Table I. Syllogistic reasoning on four sets using a Venn table.

Compartment				P_1: All A are B	P_2: No A are C	P_3: Some B are C	P_4: Some C are ~D	Unified	Resolved	C_1: Some B are D	C_2: Some B are not A	C_3: No A are D	C_4: No A~B are C
A	B	C	D		e	x_1		e x_1	e	x		e	
A	B	C	~D		e	x_1	x_2	e x_1 x_2	e				
A	B	~C	D							x		e	
A	B	~C	~D										
A	~B	C	D	e	e			e	e			e	e
A	~B	C	~D	e	e		x_2	e x_2	e				e
A	~B	~C	D	e				e	e			e	
A	~B	~C	~D	e				e	e		x		
~A	B	C	D			x_1		x_1	x_1	x	x		
~A	B	C	~D			x_1	x_2	x_1 x_2	x_1 x_2		x		
~A	B	~C	D							x	x		
~A	B	~C	~D										
~A	~B	C	D										
~A	~B	C	~D				x_2	x_2	x_2				
~A	~B	~C	D										
~A	~B	~C	~D										
Result										**Invalid**	**Valid**	**Invalid**	**Valid**

6. Determine whether the conclusion follows from the premises. If we can transform the Resolved column into the conclusion column, then the conclusion is valid; otherwise it is invalid.

Step 1 corresponds to the partial overlapping rule. By enumerating all of the compartments in Step 1, we are ensuring that we account for each and every overlap among the *n* sets. Like the procedure using Venn diagrams, the generalized procedure also involves unification (Step 3), resolution (Step 4), and transformation (Step 6). The unification, resolution, and transformation rules are all the same, except that we manipulate columns on a table, rather than x sequences and empty shadings on a Venn diagram.

Step 2 requires that we indicate on the table all facts that each premise conveys. Table I illustrates how the four categorical statement forms (A, E, I, and O) are represented in the Venn table.

P_1: All A are B (A form)

The first premise P_1 is represented by placing an "e" in each cell which represents a compartment in which A and ∼B hold. (The e means that the compartment is empty; it corresponds to a shading on a Venn diagram). There are four such compartments in Table I. In general, for the A-form statement,

$$\text{All } A_j \text{ are } A_k$$

an e is placed in the cell which represents the compartment in which A_j and $\sim A_k$ hold.

P_2: No A are C (E form)

The second premise P_2 is represented by placing an e in each cell which represents a compartment in which A and C hold. There are four such compartments in Table I. In general, for the E-form statement

$$\text{No } A_j \text{ are } A_k$$

is represented by placing an e in the cell which represents the compartment where A_j and A_k hold.

P_3: Some B are C (I form)

The third premise P_3 requires that we represent an x sequence in the table. In Table I, an "x_1" is placed in the cell that represents a compartment in which B and C hold. The x_1's thus created correspond to an x sequence created on a Venn diagram.

In general, for the I-form statement,

$$\text{Some } A_j \text{ are } A_k$$

is represented by placing x_i's in the cells that represent the compartment where A_j and A_k hold.

P_4: Some C are not D (O form)

The fourth premise P_4 also requires that we represent an x sequence in a table. This time, we place an "x_2" in the cells in Table I which represent compartments where C and ∼D hold. The general form

$$\text{Some } \mathrm{A_j} \text{ are not } \mathrm{A_k}$$

means that we must place an x_i in the cells that represent the compartment where A_j and $\sim A_k$ hold.

Step 3 is the unification step, in which P_1, P_2, P_3, and P_4 are unified. We do this by copying over all e's and x sequences to the corresponding cells into a single column labeled "Unified" in Table I. Here we have six compartments that are empty, and two x sequences, namely x_1 and x_2.

Step 4 involves resolving conflicts in the Unified column. Whenever there is an x_i and an e in the same cell, and that x_i is part of an x sequence, we may eliminate the x_i in Table I, we may eliminate x_1 from compartment ABCD, x_1 and x_2 from compartment ABC∼D, and x_2 from A∼BC∼D. See the "Resolved" column, which is shaded in Table I for the final result.

Now we are ready to test the validity of the syllogism, that is, whether the conclusion follows logically from the premises. Step 5 involves creating a separate column for the conclusion. Like the representation of premises, we will represent the facts that the conclusion conveys with e's and x's. In Table I, we will test separately four conclusions—C_1, C_2, C_3, and C_4—and hence, there are four columns to represent each conclusion.

We may attempt to transform the resolved column into the conclusion column via the transformation rules given earlier for Venn diagrams. In particular, we may erase any e or an entire x sequence from the resolved column; or we may lengthen an x sequence from the resolved column. (Note that we do not need to erase a circle because we will assume that we are leaving all sets and compartments intact, and so the transformation rule T.3 is not necessary in this discussion). If we are able to transform the resolved column into the conclusion column, then the conclusion is valid; otherwise it is invalid.

Let us look separately at each conclusion given in Table I. C_1: *Some B are D* is an invalid conclusion because there is no way to transform the Resolved column into the C_1 column. We may eliminate both the e's and the x_2's, but the x_1's cannot be extended to match the x's in the conclusion column.

However, C_2: *Some B are not A* is valid because once we eliminate the es and x_2s in the resolved column, the x_1 sequence can be extended into compartments A∼B∼C∼D and ∼AB∼CD so that it matches the C_2 column exactly.

In C_3: *No A is D*, we are dealing exclusively with e's; hence, as a first step, we will eliminate all x sequences. In further transforming the resolved column, we may eliminate the unnecessary e's from the resolved column, namely those found in compartments ABC∼D, A∼BC∼D, and A∼B∼C∼D. But the compartment AB∼CD contains an e in the C_3 column but nothing in the resolved column. Because we are not permitted to add an e, there is no way to transform the resolved column into the C_3 column. Hence, this conclusion is invalid.

C_4: *No A∼B is C* is an example of more complex example of the E Form (*No S is P*) where S is replaced by A∼B. This statement means that *No A that is not B is C*. It is represented in column C_4 by placing an e in two compartments: A∼BCD and A∼BC∼D. Like C_3, we first eliminate all x sequences, and we are left with six e's in the resolved column. It can readily be seen that we can eliminate four of the es to match exactly the two es in C_4. Hence, C_4 is valid.

5. DISCUSSION

The algorithm for performing syllogistic reasoning on Venn diagrams discussed in Section 3 has taken us, step by step, through a procedure that enables one to construct logic proofs in a graphical and visual way, as opposed to the traditional way of constructing logic proofs using first-order logic. A central feature of these Venn diagrams is that they can be manipulated via rules of manipulation, the effect of which is to transform one Venn diagram into another valid Venn diagram. The rules tell us what the permissible transformations are and serve the role of performing syllogistic reasoning for us. By doing so, they provide us with a formal procedure for proving (or disproving) the validity of a syllogism.

In Section 4, I have extended the algorithm for syllogistic reasoning. Here, the procedure allows for more than three sets of information. Try as you may, it is very difficult to draw more than three sets of overlapping circles at a time on a Venn diagram. The generalized algorithm instead employs a Venn table that enumerates all possible combinations of overlaps of n sets. Although no longer diagrammatic in nature, the algorithm replicates the steps of the Venn diagramming technique—it uses the same unification, resolution, and transformation rules—except that it transforms shadings (as denoted by e's) and x sequences on a table, rather than shadings and x sequences on a Venn diagram. Like the Venn diagramming technique, the Venn table technique is very intuitive and easy to master and learn. The tabular format also is highly visual, and it is easy to prove or disprove the validity of a conclusion given any number of premises.Yet, clearly, the algorithm cannot solve many different kinds of logic problems. First-order logic is a system that can solve many more types of logic problems—it has more expressive power—and this in part might explain why it is the logician's first choice in proving logic theorems. On the other hand, first-order logic is very difficult to learn and is not very intuitive in terms of showing one how to construct a logic proof. Indeed, it may well be the case that just because we cannot construct a proof does not mean the syllogism is invalid; rather we may not be skillful enough to produce the proof. One advantage of Venn diagrams and Venn tables over first-order logic is that they are much easier to learn and master. This is because they are highly visual and allow us to see set relationships in a perceptually more obvious way.

References

1. Nakatsu RT. Diagrammatic reasoning in AI. Hoboken, NJ: Wiley; 2010.
2. Larkin JH, Simon HA. Why a diagram is (sometimes) worth ten thousand words. Cognit Sci 1987;11:65–99.
3. Bower GH, Clark MC, Lesgold AM, Winzenz D. Hierarchical retrieval schemes in recall of categorized word lists. J Verb Learn Verb Behav 1969;8:323–343.
4. Kieras DE, Bovair S. The role of a mental model in learning to operate a device. Cognit Sci 1984;8:255–273.
5. Nakatsu RT, Benbasat I. Improving the explanatory power of knowledge-based systems: An investigation of content and interface-based enhancements. IEEE Trans Syst Man Cybern, Part A 2003;33(3):344–357.
6. Mayer RE. Thinking, problem solving, and cognition, 2nd Ed. New York: W.H. Freeman; 1991.
7. Mayer RE. Multimedia learning, 2nd Ed. Cambridge: Cambridge University Press; 2009.
8. Glasgow J, Narayanan NH, Chandrasekeran B, editors. Diagrammatic reasoning. Menlo Park, CA: The AAAI Press; 1995.
9. Harel D. On visual formalisms. Commun ACM 1988;13(5):514–530.
10. Barwise J, Etchemendy J. Heterogeneous logic. In Glasgow J, Narayanan NH, Chandrasekeran B, editors. Diagrammatic reasoning. Menlo Park, CA: The AAAI Press; 1995. pp 211–234.
11. Shin SJ. An information-theoretic analysis of valid reasoning with Venn diagrams. In Barwise J, et al. editors. Situation theory and its applications, part 2. New York: Cambridge University Press; 1991. pp 581–605.
12. Shin SJ. The logical status of diagrams. Cambridge: Cambridge Univeristy Press; 1994.
13. Nolt J, Rohayton D, Varzi A. Logic, 2nd Ed. New York: McGraw-Hill; 1998.
14. Venn J. On the diagrammatic and mechanical representation of proposition and reasonings. Philos Mag 1880;9:1–18.
15. Peirce C. In Hartshorne C, Weiss P. editors, Collected papers, Cambridge, MA: Harvard University Press; 1933.