Notre Dame Journal of Formal Logic
Volume 25, Number 3, July 1984

Mechanizing Logic II: Automated Map Logic Method for Relational Arguments on Paper and by Computer

JANET RYBAK and JOHN RYBAK*

This article is a continuation of our [12] and [13]. The methods of [12] enable us to mechanize arguments with premises containing complex terms such as 'All (X or Y) are (Z and non-W)'. Then [13] goes on to provide a basis for mechanizing relational arguments.

To the methods of [12] we add two new ideas:

(1) A rule for handling relational propositions we call Relational Conversion (RC). Here we state the rule in generalized form, and show how it works.
(2) A proposition-sorting routine, based on the moves used in our formal solutions, called "*streaming*" which allocates variables to Karnaugh maps.

Relational Conversion [10] Although Relational Conversion seems to be customarily used to describe an interchange of singular terms, we have employed it to cover also general terms. This process is not quite the same as change of quantifier order in predicate calculus.

The rule RC yields equivalent but formally different relational propositions for propositions having a relational term as predicate. (If only the subject is relational, a separate RC rule could be added, but it is ultimately simpler to use ex-

*We must acknowledge a debt to the Traditional and Modern Philosophy Department of the University of Sydney for financial aid with computing and to individual members of that department for helpful discussions.

Received March 6, 1981; revised July 21, 1982

actly the same rule applied after obverting and/or converting the proposition (cf. (8) below and (3) on the next page). RC exchanges:

(a) The subject with the *first (i.e., the dominant) relational-particle's* relatum
(b) The dominant relational-particle with its correlate
(c) The copula quantity with the relatum quantifier and vice versa; if affirmative a $\leftrightarrow d$ and i $\leftrightarrow u$, if negative e $\leftrightarrow d$ and o $\leftrightarrow u$.

For example, the proposition (α) 'Some mathematicians specialize in some branches of algebra', gives by RC (β) 'Some branches of algebra are specialized in by some mathematicians'. Symbolically, these two equivalent propositions are:

$$(\alpha)\ M \text{ i } S{-}B_u \quad \text{and} \quad (\beta)\ B \text{ i } S'{-}M_u.$$

(See [13], Appendix 1, for key to symbolism.)

The central point is that in (α) 'branches of algebra', '*B*' is only a dependent part of a relational term, whereas in (β) it is an independent nonrelational term.

Consider the case of the RC of a relational product:

(α) *Original proposition:* 'Some students respect anything favored by any radical professors.' Symbolically: S i $R{-}(F{-}P_d)_d$

(β) *RC with respect to R–:* 'Anything favored by any radical professors is respected by some students.' Symbolically: $F{-}P_d$ a $R'{-}S_u$

(γ) *RC with respect to F–:* 'No radical professors favor anything non-(respected by some students).' Symbolically: P e $F'{-}(\overline{R'{-}S_u})_d$

To obtain this RC, *obvert* (β) $F{-}P_d$ e $\overline{R'{-}S_u}$, *convert* $\overline{R'{-}S_u}$ e $F{-}P_d$ and RC P e $F'{-}(\overline{R'{-}S_u})_d$. (Every proposition has an equivalent obverse (cf. Double Negation). Only e/i propositions are directly converted in this system. To convert a/o propositions obvert them first to e/i propositions.)

Thus we have three equivalent but formally different propositions. RC is an extendible rule which can be applied to propositions of increasing complexity. Notice that the original quantities are retained throughout changes of order.

For a somewhat different account of RC see [10] or [13].

Boolean Form of Relational Propositions Relational propositions of this extended Traditional Logic are treated predicatively, just as are the nonrelational propositions; e.g., 'Some students fear some exams' is formalised as 'S i $F{-}E_u$' which is read traditionally as 'Some students are (beings or things) fearing some exams'. So, (just as 'Some A are B' is written in Boolean form as $A \cdot B \neq 0$) this is written $S \cdot F{-}E_u \neq 0$ and its RC as $E \cdot F'{-}S_u \neq 0$. This permits the diagramming of relational arguments in the same way as the nonrelational.

An Introduction to Streaming

Consider the argument: Some (things) observed by every guard are mere visitors. Certain guards are trained men. Therefore,

some mere visitors are observed by certain trained men.

Formally: 1. $O{-}G_d$ i V
2. $\underline{G \text{ i } T}$
$\therefore V$ i $O{-}T_u$

Dictionary: V = mere visitors
$O{-}$ = (things) observed by
G = guards
T = trained men
$O'{-}$ = observing

RCs: 3. G a $O'{-}V_u$ (RC of 1) ($O{-}G_d$ i V converts to V i $O{-}G_d$)
T i $O'{-}V_u$ (RC of conclusion)

Streaming is a relation-dependent technique. If all propositions (premises and conclusions) are nonrelational (monadic) then all variables lie in the one stream, appear on the one diagram. That is, streaming is normally directed by the relation-particles.

The first move is to initiate as many streams as possible by choosing a proposition with the largest number of relation-particles. So,

(1) having set up the argument in logical form
(2) RC all propositions that can be RC-ed (here (1) and the conclusion)
(3) Take from the given group any proposition having the greatest number of relations (in this example any one of the relational propositions) and arbitrarily enter its terms individually as Stream I and those of its RC as Stream II. It is essential to separate each relational proposition from its RC. All parts of the one proposition share the same stream. Here we could begin with premise 1 and its RC.

Stream I: $O{-}G_d$; V
Stream II: G; $O'{-}V_u$

(4) Now take any other relational proposition present, here the conclusion and its RC. The '$O{-}$' of the conclusion shows it belongs to Stream I and this is confirmed by the 'V' it has in common with the entries already made. Its RC shares '$O'{-}V_u$' with Stream II and so belongs there. It is also banned from Stream I by the 'V' already lodged there (see the section 'To Initiate Streaming', and Rule 3 following it)

Stream I: $O{-}G_d$; V; $O{-}T_u$
Stream II: G; $O'{-}V_u$; T

(5) This leaves only premise 2 to deal with. We see 'G' and 'T' are banned from Stream I since they appear there as relata and at the same time, in this case, happen to be already recorded in Stream II.

So streaming is complete. (If it were desired to finish the whole process, all that remains is to draw two three-variable maps (one for each stream) and enter the variable labels thus sorted above. The premises and RCs are then plotted in Boolean form on the relevant maps in essentially the same way that Venn did with his nonrelational arguments, whereupon we are effectively presented with the results—here the (equivalent) RC of the given conclusion would appear on Map II (Map I being otiose)—a relational argument has been mechanized.) Com-

plete solutions of arguments will be given below but for the moment we wish to consider only streaming.

Streaming sorts given propositions by collecting their terms in groups ready for mapping. It is guided by a set of rules sufficiently precise and complete to have allowed the writing and successful operation of a noninteractive computer program.

To initiate streaming Translate the argument into logical form, and add all RCs for the relational propositions. Now, choose any one of the given propositions with the largest number of relations and arbitrarily assign its terms and the terms of its RCs to different streams. In assigning the terms of a relational proposition to a particular stream we are automatically banning (excluding) these terms from the stream(s) chosen for its RC(s).

Streaming rules Each proposition is considered for placement into a stream by the following rules:

(1) All terms of a given proposition belong together in the one stream
(2) Propositions with the same relation-particles are placed in the same stream. A relation with a relational relatum, e.g., S i $R-(F-P_d)_d$ is streamed by its first (dominant) relation-particle, here '$R-$'.
(3) A relation and its correlate(s) are always in different streams. This implies that the terms of a proposition containing the relational-particle are excluded or banned from the stream(s) containing the correlate(s) and vice versa.
(4) Repeated variables in the one proposition are treated as separate instances of the same variable. The importance of this idea is shown by two examples:
   a. A repeated relational-particle in one proposition: Take, for example, '$D-C_u$ a $D-F_u$' which has two RCs each having a dominant correlate '$D'-$', so in this case two different streams contain the same correlate.
   b. An identical subject and relatum:

Suppose streaming is initiated by: (1) X a $L-B_d$
and its RC: (2) B a $L'-X_d$.

This would cause B and $L'-$ to be banned from Stream I and X and $L-$ from Stream II. Now, if the next premise were, e.g.;

'B likes himself': (3) B a $L-B_d$
and its RC: (4) B a $L'-B_d$,

then the B of (3) won't be banned, as might be expected, from stream I, but will in fact be placed there by its predicate '$L-B_d$' while (4) goes into Stream II because of the shared '$L'-$'. So, the streaming for these four propositions would be:

I X; $L-B_d$; B
II B; $L'-X_d$; $L'-B_d$.

The double occurrence of B in (3) and (4) accounts for the unusual streaming together of B as an independent term and B as a relatum in both streams.

(5) While streaming is a relation-dependent technique, when the relations give inadequate guidance, nonrelational terms are used to decide to which stream a proposition should be assigned.
(6) Nonrelational propositions are placed in streams either already containing all its terms, or containing at least one term and having some other term banned from each of the alternative streams.

Final Steps: As each new term-placing adds more evidence for other term-placings several passes may be needed before all associated propositions are correctly located in some stream. When these movements have subsided, nonplaced propositions which have one term already in one or more streams and their other terms nonbanned from these streams are added to the streams. Remaining propositions unplaceable in already established streams have new streams started for them.

Why Stream? Streaming assigns terms (variables) to relevant Karnaugh maps. Streaming has grouped together propositions from which deductions (syllogisms, sorites, complex-term moves, etc.) are possible. By drawing Karnaugh maps for each stream, and plotting the assigned propositions on them, the Boolean form of each deduction licensed by this system is shown automatically.

A fuller understanding of this procedure is aided by a study of our formal solutions, where common terms usually suggest which pairs or groups of propositions should be considered together to produce useful results. RC changes the terms or is used to move deductions from one map (stream) to another. In the formal method discretion is needed to choose pairs, trios, etc. of premises with common terms from which useful subconclusions can be drawn and to decide which premises can be usefully RC-ed to free new terms for further deductions. Streaming is the mechanized version of these procedures, substituting rules for discretion.

However streaming is best understood by illustrations, which will follow below.

The Algorithm After examining the detailed working of the illustrative examples the algorithmic steps below will be easier to follow.

Step 1: Add RCs

The rule RC is used to add the equivalent relational converses for each given relational proposition (including conclusions) in the argument. This allows terms embedded by the initial form of the proposition to be released; a dependent relatum may thus become an independent term. The number of equivalent propositions is equal to the number of relations in the original proposition +1, i.e., the original proposition + an RC-ed form for each relation. With a simple dyadic relational proposition there will be a total of two equivalent forms.

Step 2: Stream

Once the complete set of propositions has been found by RC, streaming

can begin. This is done by applying the rules (described above and discussed in detail in the examples) designed to associate propositions with common relational-parts or terms, and to separate the equivalent RC propositions from each other, thus linking into groups propositions which can be combined to eliminate common terms.

Step 3: Draw the Karnaugh Maps

One Karnaugh map is drawn for each stream (group of associable propositions). Each variable (i.e., all terms not currently acting as relata or part of a relatum as these are merely fragments of terms) becomes a label or Boolean term on its appropriate map. Repeated or redundant variables do no harm other than expanding the map size. Map size is determined by the number of variables; n variables require 2^n cells.

Step 4: Translate into Boolean Form and Plot Premises

Each premise is changed into Boolean form and plotted on the appropriate map. It is represented by zeros if universal or a continuous line joining relevant cells if particular unless it covers only one cell, when it is shown by a '/' within that cell.

Step 5: Find Subconclusions

Each map is now scanned exhaustively for new valid universal combinations (all relevant cells are zero), and new valid particular combinations (a line or '/' lying within the scope of the cells selected by the new combination) for two terms, then three terms and so on. A limit to this process is set by the number of variables on the largest map. If no further conclusions or subconclusions have been found by the end of this exhaustive process then the program terminates; any proposed conclusions which have not been found to be valid are invalid (see step 7). If after an exhaustive scan for, say, two-term combinations, some valid cases are found, then step 6 is used or else an exhaustive scan for three terms follows.

Step 6: Supply Equivalent Propositions to Subconclusions by RC

The found valid combinations (subconclusions) if relational, are RC-ed to find their equivalent relational converses which are then transferred to the other relevant maps provided all the variables of their Boolean form are present on the other map, otherwise they are ignored as lying outside the field of the given.

Step 7: Iterate Subconclusion Finding and RCing

Again the maps are scanned because new information has been added to the other maps. New valid subconclusions are sought and, if found, are RC-ed and their RCs are placed on the appropriate maps (ignoring propositions with terms absent from the given set for that map). That is, steps 5 and 6 are iterated until either each proposed conclusion (or one of its RCs) is found to be valid, or no further information results from the scanning; i.e., nothing further can be deduced and so the program halts, having converged, and any proposed conclusion which has not been found to be valid, is invalid.

Notes on the Algorithm: Karnaugh maps are Venn diagrams extended to n variables—an accepted *effective* decision procedure. RC has well-known linguistic applications (e.g., active-passive transformations or correlates 'left of—right of' etc.). Streaming is a new mechanical analogue of the formal logicians' discretionary taking together groups of propositions to discover what they imply, if anything. Some refinements of the algorithm are best reserved for a separate publication. They have been set aside as needlessly complicating an initial presentation of a largely new approach.

Notice that the search for subsidiary conclusions aids the discovery of a range of nongiven valid conclusions (i.e., unforeseen links between variables are revealed). Karnaugh maps also help to focus attention on inconsistent premises. Map conclusions are always found in Boolean form and can, of course, be translated into many formally equivalent linguistic forms.

Small problems can be solved more concisely by hand using formal methods (which may be established or checked mechanically by Karnaugh map) but here we are concerned to show a reliable method which uses no intuition or ingenuity, and which always terminates for the data-set tested. (See the Appendix for discussion of the computer program.)

The algorithm is completely computerizable. To illuminate its essential features and to confirm its effectiveness, four examples are illustrated by hand.

Validity-Invalidity Check: If the conclusion is valid then mapping the premises (and subsidiary conclusions) *automatically* maps the conclusion or its RC on some iteration. If the conclusion is invalid, mapping the premises, etc., will not map the conclusion on any iteration.

That is, a/e conclusions are valid if all cells representing the conclusion contain zeros (actually premise numbers). If a/e conclusions do not fulfill this condition on any iteration they are invalid.

If a curved line (premise or subsidiary conclusion) lies entirely within the scope of the cells given by an i/o conclusion then the conclusion is valid. If the validity condition is not fulfilled on any iteration or if all cells given by the i/o conclusion are filled with zeros (a/e premise or subconclusion numbers) then the conclusion is invalid.

It is essential that a/e premises (and RC-ed subsidiary conclusions) are always plotted before i/o premises, etc., as cells with zeros are not accessible to curved lines. This means that maps are replotted by the computer after each iteration.

'Curved lines' are not always continuous on a Karnaugh map as a/e plottings may interrupt the curved lines.

As a relational proposition and its RC are equivalent, if one is found to be valid then the other is also valid for this system.

Other Automatic Methods: We have looked at the increasingly complicated literature of the prevailing automatic theorem-provers (for example, [3], [8], [4], and also [1] and [5]) and decided to test the effect of searching in an entirely different direction for a simpler solution to the problems of mechanizing logic.

The Algorithm at Work with Examples Done Formally in Mechanizing Logic I

Example 1. A Simple Illustration:
Argument: Some botanists are eccentric women.
Some botanists do not like any eccentric person.
Therefore, some botanists are not liked by all botanists. [6]

Symbolized English	Boolean Form	Comments
(1) B i $E \cdot W$	$B \cdot E \cdot W \neq 0$	
(2) B o $L-E_d$	$B \cdot \overline{L-E_d} \neq 0$	
$\therefore B$ o $L'-B_u$	$\therefore B \cdot \overline{L'-B_u} \neq 0$	
Added RCs		
(3) E e $L'-B_u$	$E \cdot L'-B_u = 0$	(RC of (2))
B o $L-B_u$	$B \cdot \overline{L-B_u} \neq 0$	(RC of conclusion)
Streams: I B; $L-E_d$; $L-B_u$		(3 variables)
II E; $L'-B_u$; B; W		(4 variables)

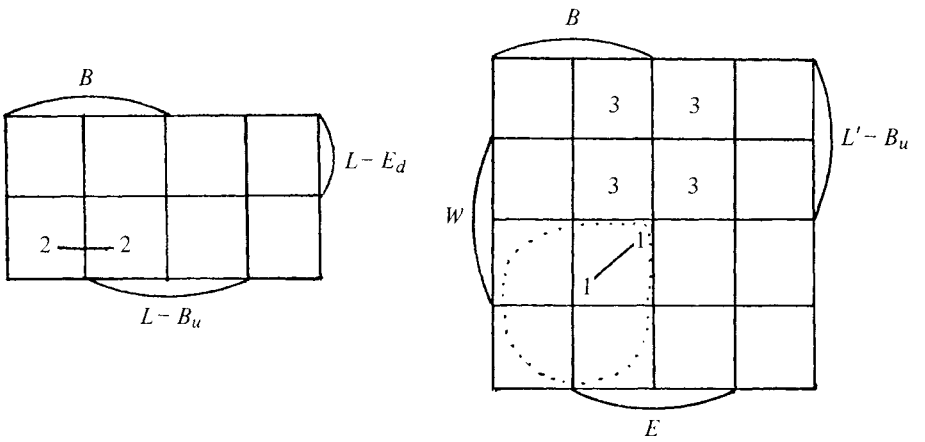
Streaming is initiated with the RC pair (2), (3); then because the conclusion contains the streamed term "$L'-B_u$" which is banned from Stream I the conclusion subject is added to Stream II. Premise (1) belongs to Stream II as 'E', like "$L'-B_u$", is banned from Stream I, so 'W' is added to Stream II. The RC of the conclusion is in Stream I, because of its relation-particle '$L-$'. Now all variables are streamed.

Points of interest:

1.1 'not every, not all' = some: u; ('not any' = d).
1.2 The two instances of B in the conclusion and in its RC cause B to be present in two streams. The predicate in each case selects the stream for the subject B.
1.3 Changing variable-name positions on the map will not affect the results.
1.4 On the maps, a/e premises are marked by premise numbers (instead of zeros), and i/o premises are shown by numbered curved lines, or a single straight line if only one square is involved.
1.5 [12] describes the method for drawing maps of any size.

Premise (2) is entered on Map I, and premises (3) then (1) are plotted onto Map II. The proposed conclusion (circled) is found to be present on the map, i.e., the conclusion is *valid*. Map I is not functional in this example, as it happens.

Other proposed conclusions can be checked on the maps. For instance, 'No women are liked by all botanists' (W e $L'-B_u$), i.e., $W \cdot L'-B_u = 0$. Checking the combination "$W \cdot L'-B_u$" on Map II, it will be seen that only two of the four possible squares contain premise numbers (zeros), therefore this proposed conclusion is *invalid*. 'All botanists are liked by some botanist' (B a $L'-B_u$), $B \cdot \overline{L'-B_u} = 0$, is also *invalid*. Or, on Map I, 'Some who like every eccentric do not like some botanists' ($L-E_d$ o $L-B_u$), $L-E_d \cdot \overline{L-B_u} \neq 0$, is *invalid* because nothing is shown by the specified cells.



Map I Map II

Figure 1

Example 2. Illustration of the Method:
Argument: Helen is an actress.
All actresses like Tony.
All men liked by Helen are good-looking.
Therefore, Tony, the man, is good-looking. [2]

Symbolized English	Boolean Form	Comments
(1) H a A	$H \cdot \overline{A} = 0$	
(2) A a $L-T_d$	$A \cdot \overline{L-T_d} = 0$	
(3) $M \cdot L'-H_d$ a G	$M \cdot \bar{G} \cdot L'-H_d = 0$	Explained below*
$\therefore T \cdot M$ a G	$\therefore T \cdot M \cdot \bar{G} = 0$	
Added RCs		
(4) T a $L'-A_d$	$T \cdot \overline{L'-A_d} = 0$	(RC of (2))
(5) H e $L-(M \cdot \bar{G})_d$	$H \cdot L-(M \cdot \bar{G})_d = 0$	(RC of (3))**

*Obvert: $M \cdot L'-H_d$ e $\bar{G}$; convert: $\bar{G}$ e $M \cdot L'-H_d$, Add and Drop SC: $M \cdot \bar{G}$ e $L'-H_d$.[1]

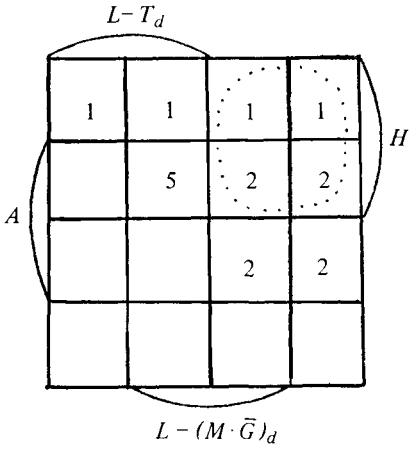
**'Helen doesn't like any non-(good-looking) man'.

Streams: I A; $L-T_d$; $L-(M \cdot \bar{G})_d$; H (4 variables)
II T; $L'-A_d$; M; $L'-H_d$; G (5 variables)

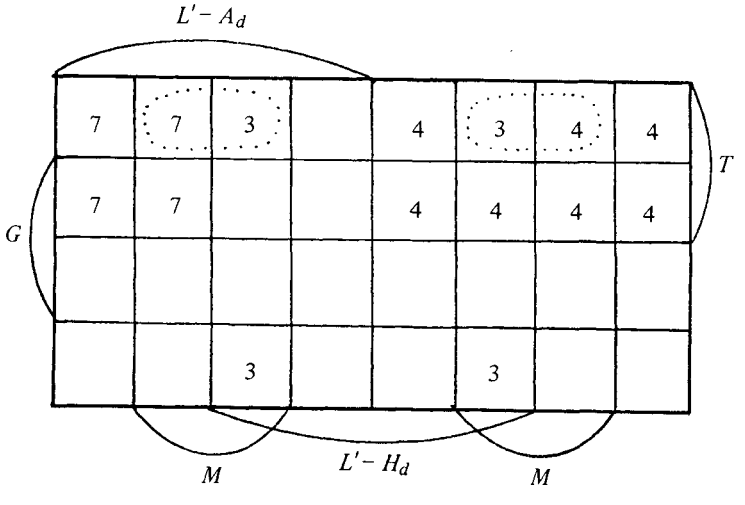
Streaming is initiated with the RC pair (2), (4); then (3), (5) are added to II and I respectively because of their relation-particle matches, $L'-$ and $L-$. Now all variables have been assigned to a stream, so, premise 1 is in Stream I.

Premise numbers instead of the Boolean 'O' are recorded on the maps to aid the selection of new combinations. If a cell has been marked by a premise number, other premise numbers falling in the same cell are omitted to simplify the figures in this paper, e.g., premise 5 on Map I.

After the maps have been plotted they are examined for new combinations.



Map I



Map II

Figure 2

Here the only subsidiary conclusion which is not just a mapped premise or part of a mapped premise is found on Map I, (6) "$H\cdot\overline{L-T_d}=0$", i.e., "H a $L-T_d$" whose RC is "T a $L'-H_d$" giving (7) "$T\cdot\overline{L'-H_d}=0$" to be plotted on Map II. The required conclusion (circled) "$T\cdot M\cdot\bar{G}=0$" is now seen to be present on Map II. Therefore the required conclusion "$T\cdot M$ a G" is *valid*.

All other subsidiary conclusions read from Map II have RCs which contain a term lying outside the range of the data-set (e.g., "$T\cdot G$ a $L'-H_d$" RCs to "H a $L-(T\cdot G)_d$" and "$L-(T\cdot G)_d$" is not a given term on any map). That is, no further information is exchanged between maps: the iterative process has converged. As convergence has been reached, the maps can be inspected to see whether proposed conclusions using terms on the maps are valid or invalid. For

example, "$M \cdot \overline{L' - A_d} = 0$" (All men are liked by any actress) is *invalid.* Or, on Map I, "$A \cdot \overline{L - (M \cdot \bar{G})_d} = 0$" (Every actress likes every non-(good-looking) man)—not demonstrated on the map—is *invalid.*

Example 3. A More Complicated Example with Four Maps and Three Iterations:

Argument: Everyone reading some good biology books is interested in every good biology book.
All zoologists read some good biology books.
All Darwin's main works are good biology books.
All good biology books interesting every zoologist contain some firmly established doctrines.
All firmly established doctrines are seminal.
$\therefore$ All Darwin's main works contain something seminal.

	Symbolized English	Boolean Form	Comments
(1)	$R - B_u$ a $I - B_d$	$R - B_u \cdot \overline{I - B_d} = 0$	(R = reading, I = interested in)
(2)	Z a $R - B_u$	$Z \cdot \overline{R - B_u} = 0$	
(3)	D a B	$D \cdot \bar{B} = 0$	(D = Darwin's main works)
(4)	$B \cdot (I' - Z_d)$ a $C - F_u$	$B \cdot (I' - Z_d) \cdot \overline{C - F_u} = 0$	(C = containing)
(5)	F a S	$F \cdot \bar{S} = 0$	(F = firmly established doctrines)
	$\therefore D$ a $C - S_u$	$\therefore D \cdot \overline{C - S_u} = 0$	
	Added RCs		
(6)	B a $I' - (R - B_u)_d$	$B \cdot \overline{I' - (R - B_u)_d} = 0$	(RC for '$I-$' for (1))
(7)	B o $R' - (\overline{I - B_d})_d$	$B \cdot \overline{R' - (\overline{I - B_d})_d} \neq 0$	(RC for '$R-$' for (1))
(8)	B i $R' - Z_d$	$B \cdot R' - Z_d \neq 0$	(RC for (2))
(9)	F i $C' - (B \cdot I' - Z_d)_d$	$F \cdot C' - (B \cdot I' - Z_d)_d \neq 0$	(RC for '$C-$' for (4))
(10)	Z e $I - (B \cdot \overline{C - F_u})_d$	$Z \cdot I - (B \cdot \overline{C - F_u})_d = 0$	(RC for "$F'-$" for (4))
	S i $C' - D_d$	$S \cdot C' - D_d \neq 0$	(RC for conclusion)

Streams:			
	I	$R - B_u$; $I - B_d$; Z; $I - (B \cdot \overline{C - F_u})_d$	(4 terms)
	II	B; $I' - (R - B_u)_d$; $I' - Z_d$; $C - F_u$; $C - S_u$; D	(6 terms)
	III	B; $R' - (\overline{I - B_d})_d$; $R' - Z_d$; D	(4 terms)
	IV	$C' - (B \cdot I' - Z_d)_d$; F; S; $C' - D_d$	(4 terms)

Points to notice:

3.1 Premises 1 and 4 have two relations and hence two RCs each. (7) is (1) obverted, converted, and RC-ed: $R - B_u$ e $\overline{I - B_d}$; $\overline{I - B_d}$ e $R - B_u$; B o $R' - (\overline{I - B_d})_d$. (10) comes from (4); obvert, convert, and add SC, B, to subject, drop SC, B, from the predicate then RC: $B \cdot (I' - Z_d)$ e $\overline{C - F_u}$; $\overline{C - F_u}$ e $B \cdot (I' - Z_d)$; $B \cdot \overline{C - F_u}$ e $I' - Z_d$; Z e $I - (B \cdot \overline{C - F_u})_d$.

3.2 Where there are two relation-particles forming a relational product the stream is decided by the *first* of the pair, e.g., $R'-$ for premise 7.

3.3 On Map II, "D a $I' - (R - B_u)_d$", e.g., can be seen to be valid, but

$I-D_d$ (in its RC) is not a data item, so this adds no new information for the next iteration.

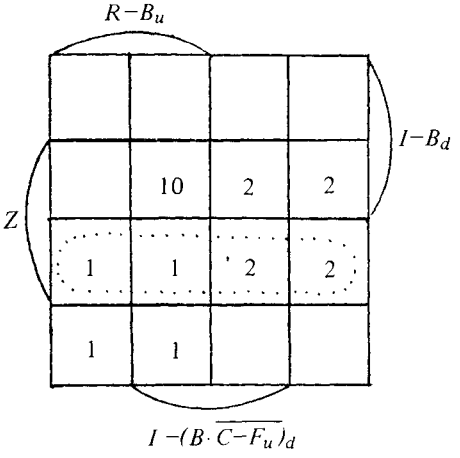
3.4 i/o premises are plotted with curved lines (identified by their premise numbers) through the relevant cells (see Map IV).

3.5 Map III is otiose here and is omitted.

3.6 Conclusions other than the one named in the argument can be found on the maps, e.g., S i $C'-(B \cdot I'-Z_d)_d$, on Map IV, the RC of which could be transferred to Map II to produce further conclusions.

3.7 Note, incidentally, the method of drawing a map for six variables. [9]

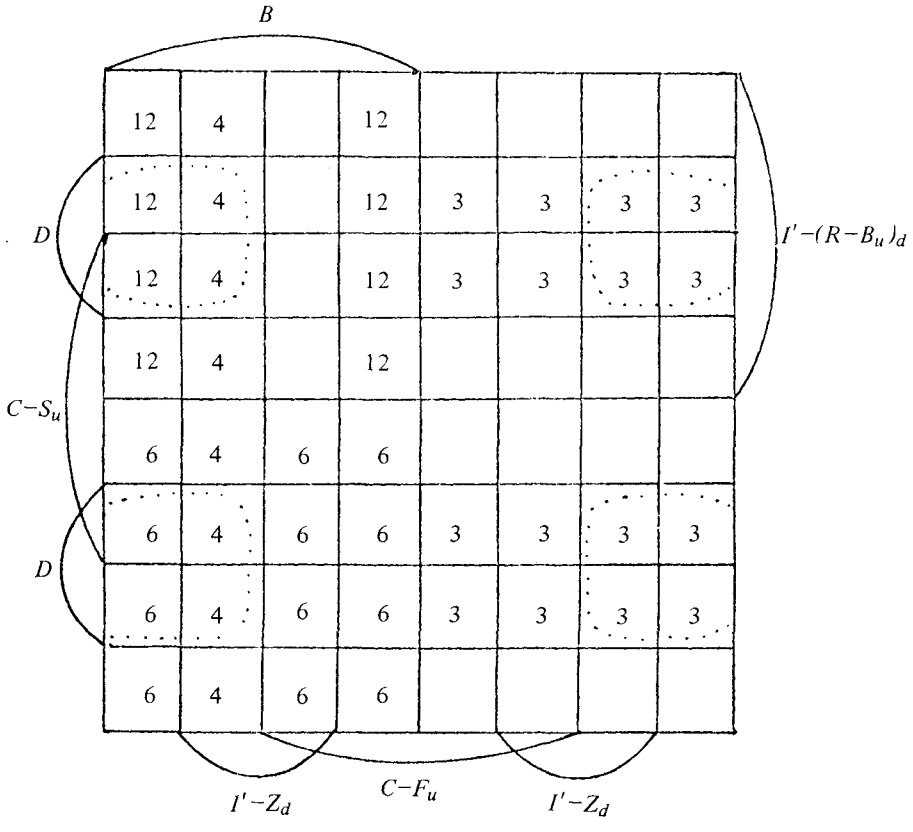
The Algorithm at Work: (1), (6), and (7) are used to initiate Streams I, II, and III, respectively. (2) and (8) are located next in I and III respectively because of $R-$ and $R'-$. (10) is now placed in Stream I as its relation-particle is '$I-$', and 'Z' also lies in Stream I. Premise 3, being a nonrelational proposition, is placed in both Streams II and III as they already contain B. Because of B and $I'-$ in its subject, premise 4 belongs to Stream II; this adds '$C-F_u$' to that stream. The conclusion is in Stream II because both D and $C-$ are there. (5), (9), and the RC of the conclusion are associated with each other (because of $C'-$, S and F) but not with any other stream, therefore Stream IV is started for them and streaming is complete.



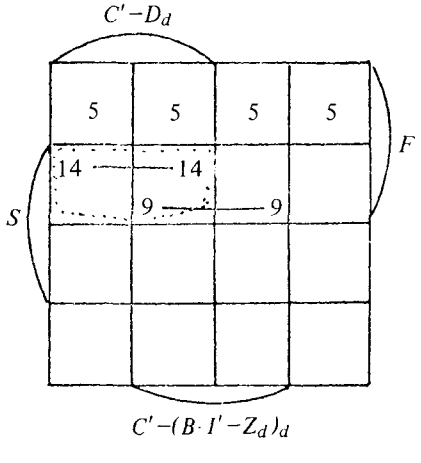
Map I

Figure 3a

Reading the Valid and the Invalid from the Maps: Subsidiary conclusion (11), 'Z a $I-B_d$', found on Map I gives as RC, (12) 'B a $I'-Z_d$' which can be transferred to Map II. The second iteration conclusion, (13), 'D a $C-F_u$', now found on Map II has as RC, (14) 'F i $C'-D_d$' which is added to Map IV giving (15) 'S i $C'-D_d$'. Thus, the RC of (15) 'D a $C-S_u$', the required conclusion, appearing on the third iteration, is *valid*.



Map II



Map IV

Figure 3b

An illustration of the second invalidity condition (see the earlier section "Validity-Invalidity Check" for i/o conclusions can be seen on Map II. If we wished to check the possible conclusion 'Some good biology books do not contain all firmly established doctrines', i.e., '$B \cdot \overline{C-F_u} \neq 0$' we would find every cell for this Boolean combination is occupied by a zero (a premise number only); the conclusion is *invalid.* We might also ask whether '$I-B_d$ a Z' (All (beings) interested in every good biology book are zoologists) is valid? Map I shows it has not been implied—is *invalid.* Is 'All firmly established doctrines are contained in Darwin's main works' (F a $C'-D_d$) to be found on Map IV? No; it is *invalid.*

Example 4. An Illustration with Four Particular Conclusions, Two Valid and Two Invalid:

Argument: All contributions to this publication are reports of original research work. Anyone who produces a report of original research work is hard-working or nonconformist. Some very obscure persons produced some of the material which is a contribution to this publication. Therefore, (a) some very obscure persons are hard-working or nonconformist, (b) some very obscure persons are not producing some reports of original research work, (c) some very obscure persons are neither hard-working nor nonconformist, and (d) some reports of original research work are produced by some very obscure persons. [2].

	Symbolized English	Boolean Form	Comments
(1)	C a R	$C \cdot \bar{R} = 0$	
(2)	$P-R_u$ a $(H \vee N)$	$P-R_u \cdot \bar{H} \cdot \bar{N} = 0$	(P = producing)
(3)	$\underline{O \text{ i } P-(M \cdot C)_u}$	$\underline{O \cdot P-(M \cdot C)_u \neq 0}$	
	$\therefore O$ i $(H \vee N)$	$\therefore O \cdot H \vee O \cdot N \neq 0$	
	O o $P-R_u$	$O \cdot \overline{P-R_u} \neq 0$	
	O o $(H \vee N)$	$O \cdot \bar{H} \cdot \bar{N} \neq 0$	
	R i $P'-O_u$	$R \cdot P'-O_u \neq 0$	
	Added RCs		
(4)	R o $P'-(\bar{H} \cdot \bar{N})_d$	$R \cdot \overline{P'-(\bar{H} \cdot \bar{N})_d} \neq 0$	(RC of (2))
(5)	$M \cdot C$ i $P'-O_u$	$M \cdot C \cdot P'-O_u \neq 0$	(RC of (3))
	R o $P'-O_u$	$R \cdot \overline{P'-O_u} \neq 0$	(RC of conclusion (b))
	O i $P-R_u$	$O \cdot P-R_u \neq 0$	(RC of conclusion (d))

Streams: I $P-R_u$; H; N; $P-(M \cdot C)_u$; O (5 variables)
II R; $P'-(\bar{H} \cdot \bar{N})_d$; M; C; $P'-O_u$ (5 variables)

Points to notice:

4.1 Complex terms (terms linked by 'and' and 'or') fit neatly into the system.

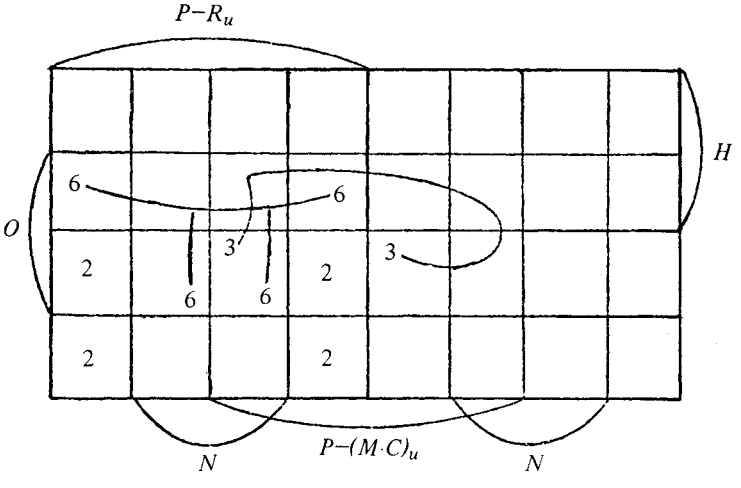
4.2 (4), the RC of (2), is obtained by obversion and De Morgan's Law then conversion and RC; i.e., $P-R_u$ e $(\overline{H \vee N})$; $P-R_u$ e $\bar{H} \cdot \bar{N}$; $\bar{H} \cdot \bar{N}$ e $P-R_u$; R o $P'-(\bar{H} \cdot \bar{N})_d$.

4.3 a/e premises are plotted first, then the i/o premises which are indicated

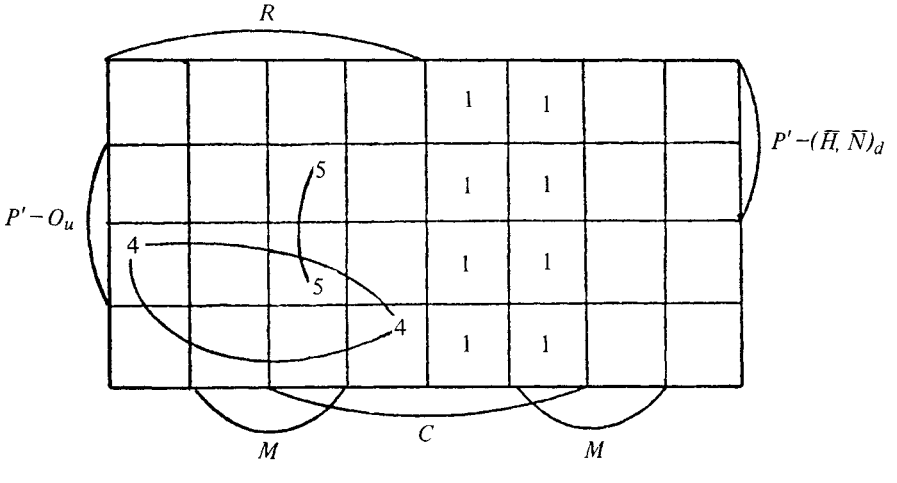
by drawing curved lines through the remaining relevant cells. On Map I the range of premise 3 is confined by premise 2.

4.4 A single term may represent many words, e.g., '*C*' = 'contributions to this publication' (see also the later section, 'Other Points').

The Algorithm at Work: Streaming starts with (2) and (4). (3) and (5) are then placed in Streams I and II, respectively, because of their relation-particles '*P*–' and '*P*′–'. All terms are now streamed.



Map I



Map II

Figure 4

Considering only two-term combinations, and ignoring part premises (e.g., '$P'-O_u \cdot M \neq 0$' on Map II arising from premise 5) and nontransferable new combinations (e.g., '$R \cdot M \neq 0$' on Map II) the only new RC-able subsidiary conclusion is '$R \cdot P'-O_u \neq 0$' or 'R i $P'-O_u$' from Map II, which is conclusion (d), i.e., conclusion (d) is *valid.* Conclusion (d) RCs to (6) 'O i $P-R_u$' or '$O \cdot P-R_u \neq 0$' which is plotted onto Map I.

The Algorithm seeks two-term combinations first. If at least one RC-able subsidiary conclusion is found then the program passes back to the RC routine, and hence the next iteration. But if no two-term combinations are found, then three-term combinations are sought, and so on, up to the total number of terms on the largest map. Here the second iteration finds no two-term subsidiary conclusions, but conclusion (a) '$O \cdot H \vee O \cdot N \neq 0$' is now present on Map I. So, 'O i $(H \vee N)$' is *valid.*

New three-term subsidiary conclusions are '$R \cdot M \cdot P'-O_u \neq 0$' and '$R \cdot C \cdot P'-O_u \neq 0$' and '$R \cdot M \cdot C \neq 0$'. The last is nonrelational and RCing the first or the second leads to nongiven Map I terms. The only four-term subsidiary conclusion '$R \cdot M \cdot C \cdot P'-O_u \neq 0$' also leads to a nongiven Map I term. No five-term subsidiary conclusions are found. Now that convergence has been reached, it can be seen that conclusions (b) and (c) are not supported by the given premises. That is, they are *invalid.*

Reading subsidiary conclusions from the maps has, with this example, almost reached the point of complexity where it is sensible to hand the job over to a computer. A computer is particularly well-suited to this phase of the algorithm. When the detail becomes excessive and strenuous for the human mind to handle, then it is time to use the computer, though eventually exponential growth will defeat the computer and breaking the problem up into smaller units may be the answer. The algorithm incorporates the basic instructions needed by the computer program to carry out mechanized deduction.

The Computer Program The authors actually have a somewhat inelegant but nevertheless working computer program written in accordance with this algorithm. The program is described in more detail in the Appendix.[2] A data-set of some seventy varied problems each with valid *and* invalid proposed conclusions taken from different modern logic texts (e.g., [2], [7], etc.) presented no difficulties to the program. This would seem to support the soundness of the algorithm, and its theoretical basis.

Program Restrictions: The program has so far been confined to propositions with a maximum of two relations: a relational product, two relations connected by 'and' or 'or', or a relation governing a complex relatum of which one term is relational. The authors' next program would allow relational propositions of any complexity consistent with storage limitations, as there seems to be no difficulty, apart from exponential growth problems with computer storage and time, in expanding the algorithm to accommodate more complicated propositions.

Other Points: We have to concede, as every system of logic has to, that translation into logical form can involve serious doubts and difficulties. However the

relative closeness to ordinary English and the conciseness of the symbolism of this system are attributes in its favor.

Conclusion Map Logic is an applied logic which has a convergent (terminating) iterative procedure which shows the validity of the valid conclusions and equally the invalidity of the invalid conclusions. This has been confirmed by our successful testing of the repertoire of representative logic-text examples. The system seems to be viable in this field.

Considerations of space would require us to leave for a future article some topics which might include, e.g., some refinements of streaming, triadic and other polyadic relations, tentative applications to elementary algebra, and generally problems even more complicated than those exemplified in this article (cf. [11]).

Appendix: The Program The authors have written in Fortran IV (4.7 extended) a Map Logic program, developed over the last five years, to run on the University of Sydney's CDC Cyber-170-730. It was written to test the soundness of their theory. Data are read as characters but immediately converted to a numeric form by using a dictionary of terms and of relation-particles. The program expects to perform algebraic Boolean operations on propositions. Each Karnaugh map is a vector of 2^n elements where n is the number of variables on the map. $(n+2)$ is restricted to the integer digit-size of the computer word (here 17) as each variable is assigned a value 0, 1, or 3 (positive, negative, or not present) and there is a propositional-value-indicator with a value 0, 1, or 2 ('=0', '≠0' for 1 cell, i.e., '/'; or '≠0' for more than 1 cell, i.e., a curved line) plus an 'and/or' term-indicator for propositions with 'and/or' terms. The RC routine, at present, handles up to two dyadic relations per proposition. That is, relational-products, two relations linked by 'and' or 'or', or a relation governing a complex relatum of which one term is a relation (in one of the equivalent RCs).

The present pilot program has a total maps-size area of 4096 words, i.e., 12 variables for a nonrelational problem, or, for example, 10 variables on each of four maps. This area can be increased by full use of core stores, disk storage, or magnetic tapes in exchange for some job run-time loss, but as the problem-size expands, storage requirements increase rapidly as map-vector size is an exponential function of the number of variables. This is a difficulty which haunts all methods attempting to solve similar problems, but is somewhat alleviated in this case by the sharing of the variables over several maps.

The pilot program was written to test our theoretical position and so contains much redundant text built in to check that the program was behaving self-consistently. Now the authors are prepared to write a more generalized version, adding programming improvements and storage economies to enhance its efficiency, as time and money allow. Then its present execution (CPU) time of 3.0 sec for Example 2, 12.9 sec for Example 3, and 28.4 sec for Example 4 should be greatly improved (assuming the same computer and systems). Significant information for each step of the algorithm is printed, and a special option-call supplies Karnaugh maps and other diagnostic information.

NOTES

1. For further details, see [13] under headings Add SC and Drop SC.
2. The authors will be happy to enter into correspondence about the computer program with those interested.

REFERENCES

[1] Ackermann, W., *Solvable Cases of the Decision Problem*, North-Holland, Amsterdam, 1954.

[2] Copi, I. M., *Symbolic Logic*, 3rd Ed., Macmillan, New York, 1967.

[3] Cook, S. A., "The complexity of theorem-proving procedures," *Proceedings of the 3rd ACM Symposium on Theory of Computing*, 1971, pp. 155–157.

[4] Joyner, W. H., Jr., "Resolution strategies as decision procedures," *Journal of the ACM*, vol. 23, no. 3 (1976), pp. 398–417.

[5] Kneale, W. and M. Kneale, *The Development of Logic*, Oxford University Press, London, 1962, pp. 725–728.

[6] Lemmon, E. J., *Beginning Logic*, Nelson, 1965, p. 134.

[7] Quine, W. V. O., *Methods of Logic*, 2nd Ed., Routledge & Kegan Paul, London, reprinted 1966, pp 175–189, etc.

[8] Rabin, M. O., "Theoretical impediments to artificial intelligence," *Proceedings of the I.F.I.P Congress*, Stockholm, Sweden (1974), pp. 615–619.

[9] Rybak, J., "Diagrams for set theory and probability problems of four or more variables," *The American Statistician*, vol. 29 (1975), pp. 91–93.

[10] Rybak, J. and J. Rybak, *Map Logic*, self-published, December 1973. Out of print, but copies in Library of Congress, Princeton, Yale, The British Library, Bodleian, and Cambridge University Library, etc.

[11] Rybak, J. and J. Rybak, "Map logic," *Australasian Journal of Philosophy*, vol. 53, no. 3 (1975), pp. 257–259.

[12] Rybak, J. and J. Rybak, "Venn diagrams extended: Map logic," *Notre Dame Journal of Formal Logic*, vol. 17, no. 3 (1976), pp. 469–475.

[13] Rybak, J. and J. Rybak, "Mechanizing logic I: Map logic extended formally to relational arguments," *Notre Dame Journal of Formal Logic*, vol. 25, no. 3 (1984), pp. 250–264.

Janet Rybak
Department of Economic Statistics
The University of Sydney
New South Wales, Australia 2006

John Rybak
Department of Traditional and Modern Philosophy
The University of Sydney
New South Wales, Australia 2006