Dartmouth Conference on Artificial Intelligence

Report on the 5th and 6th Weeks.

By Trenchard More

Attendance

5th week 16-20 July 1956

Etter, T.
McCarthy, J.
More, T.
Solomonoff, R.

6th week 23-27 July

Monday	Wednesday
Ashby, W.R.	McCarthy, J.
McCarthy, J.	Minsky, M. L.
Minsky, M. L.	More, T.
More, T.	Solomonoff, R.
Solomonoff, R.	

Tuesday	Thursday
Ashby, W. R.	McCarthy, J.
(Culver, R.)	Minsky, M. L.
McCarthy, J.	More, T.
McCulloch, W. S.	Robinson, A.
Minsky, M. L.	Solomonoff, R.
More, T.	
Solomonoff, R.	

Friday

McCarthy, J.
More, T.
Robinson, A.

This report will outline, under each name in alphabetic order, the current work of each person attending the conference.

Ashby, W. R.

Barnwood House
Gloucester, England

Ashby has just completed a new book "Introduction to Cybernetics", of which he has a printer's proof, and which should appear in a few months.

A point which Ashby stressed several times is the following. A simple machine appears to be extraordinary when viewed psychologically. When part of a mechanism is concealed from observation, the behavior of the mechanism seems remarkable.

After four years, Ashby has distilled two major criticisms of the homeostat which he described in his book "Design for a Brain".

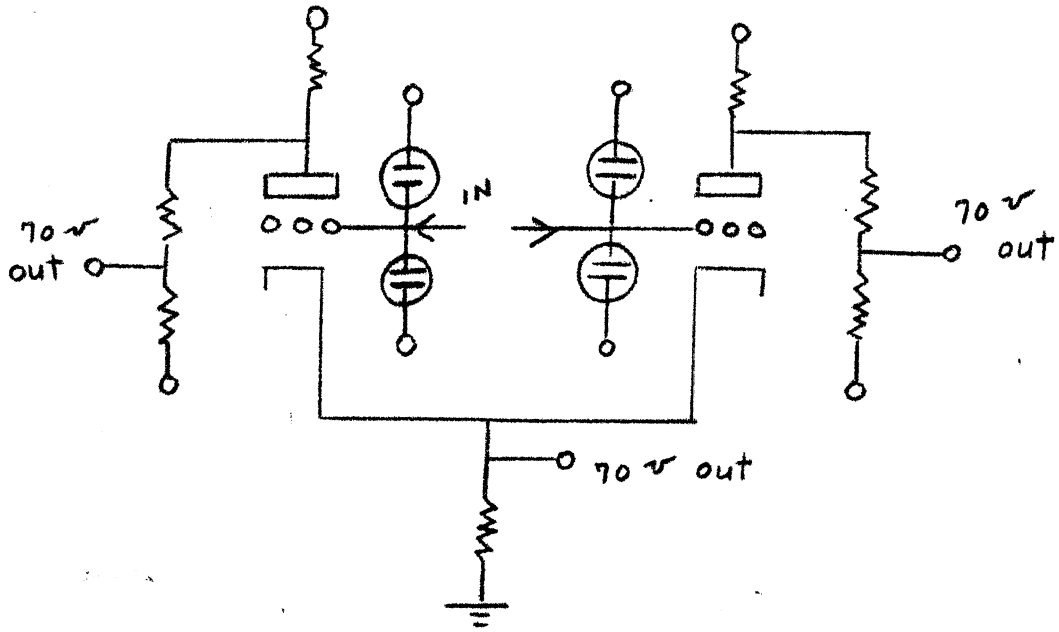
1. If, in working on problem A, the homeostat must solve problem B, then the homeostat will return to problem A as if it had never seen A before.

2. Time for problem solving goes up exponentially with the number of units in the homeostat.

In answer to criticism 1, Ashby suggested that a different part of a large memory be used to solve each problem. ?

In answer to criticism 2, Ashby suggested that random numbers be used to find new direct approaches to the problem. Not really.

In the original homeostat, if the system was stable, the needles of the four units of Ashby's machine would stay in the center. If unstable, one or more of the needles would diverge to hit a stop. When this happened, the ~~operator~~ machine would change one or more of the uniselectors, reorganizing the input connections, and the homeostat would search the new field for stability. The total process, giving rise to ultrastability, continued until the homeostat found a stable field. Recently, Ashby has been working on a network of 100 twin triodes to handle automatically the operation of the uniselectors. Each twin triode circuit has two inputs and three outputs. No!



Minor feedback for stability goes through the circuit without changing the neon bulbs. Major feedback for ultrastability changes the neon bulbs, and hence the effective connection of the circuit.

Ashby found that a completely randomized 10X10 twin triode field tended to lock into fixed states. He found that game theory did not help him organize the twin triode field.

It was surprising that Ashby shoudl find a cylindrical organization, a row of 10 twin triodes randomized into the adjacent rows, to yeild the best results obtained so far. This arrangement is similar to Rochester's organization of simulated neurons.

Ashby gave a lecture containing the above ideas on Monday afternoon, 23 July, 1956.

Culver, R. Rand Corporation

Culver accompanied McCulloch when they came to take Ashby to a conference at Colby College. Culver is interested not in artificial intelligence, but inertial navigation.

Etter, T. Not affiliated

Etter was interested in the decrease of TV Bandwidth by finding parameters that approximate the abstractions human beings use in describing pictures. Interpolation between pictures obtained in this way should approximate continuity.

McCarthy, J. Dartmouth College

McCarthy has become increasingly interested in the problem of writing a program that will write programs, the generative program being part of a general problem solving program. At present he is pursuing logic to find a language which will precisely express problems based mainly on an intuitive set theory.

He is also considering design aspects of a chess playing machine.

McCulloch, W.S.

McCulloch was present only for a few hours. In that time, however, he discussed Carnap's system of languages, and Turing machines.

Carnap's contention, as expressed by McCulloch, is that once a machine learns ostensibly the Sheffer Stroke function, it can construct the rest of logic, including quantification.

McCulloch claims that he can prove that a Turing machine can learn the English language, and belives that the human brain is a Turing machine. A Turing machine with a random element in the control unit, is the same as a Turing machine with a random tape.

Minsky, M. L. Harvard Fellow

Minsky's work is centered around "A Framework for an Artificial Intelligence", which is a block diagram approach to the construction of a problem solving machine, where no one of the blocks contains the intelligent part of the machine.

Minsky is also interested in a machine that controls a humanlike model in a simple physical environment of building blocks and gravity. The machine is expected to develop in its memory a model of the environment, and eventually a model of its own physical structure, thus gaining a certain degree of apperception.

More, T. IBM, MIT

More has been working on name theory, and the organization of blank memories. He found that he could tie together Craik's hypothesis on the nature of thought, name theories developed by Frege, Pierce, Quine, Church and Carnap, Morris's behavioral semiotic, Minsky's emphasis on models, methods of programming, and a symmetric machine, into a theory of machine semeiology. Semeiology is the science of signs, and machine semeiology is the science of signs applied to the design of machines, as well as the study of machines applied to the understanding of signs. More delivered a lecture on the subject on Tuesday afternoon, July 24, 1956.

Robinson, Abraham U. Toronto, Canada

Robinson is a logician and mathematician, who is interested in the heuristic suggested by logic in solving mathematical problems, especially problems in abstract algebra. He pointed out that the functional calculus of first order is not a comprehensive enough language to prove theorems in group theory, when McCarthy was trying to see if that calculus could be made into a general problem solving language.

Robinson finds that metalanguages about metalanguages, etc., is the source of much heuristic. He suggested that after a machine leaps through several levels of language, the machine should perform a generalization on the experience stored in its memory.

The preliminary outline of the report Solomonoff intends to write, is the following:

A. General description of how the machine operates.
B. Detailed description of how the machine operates.

1. Definition of terms.
2. Mode of operation.

a. How old n-grams that have been useful are transformed into new n-grams that have any a priori probability of being useful.

1. Factoring of n-grams into structures and n-tuples.
2. Creation of new n-tuples from old.
3. Creation of new structures from old.
4. Combination of n-tuples and structures to form new n-grams.

b. How a priori values of utilities of newly created n-grams are calculated from utilities of component structures and n-tuples.

c. How utilities are empirically determined.

d. How to get utilities of structures and n-tuples from empirical utilities of n-grams. This then feeds back to modify step b.

3. Illustration of mechanisms used in learning $=$, $\sim$, $\oplus$, $\otimes$, +, -

C. Correspondence between elements of the machine, and elements in intuitive inductive processes.

1. Why operations performed by the machine have a reasonable correspondence to operations performed in the English language. End of outline.

At the present time, Solomonoff has just completed the definitions of terms.

In Solomonoff's opition, the most important problem that gives real ability and intelligence to the machine, concerns the machine's capacity to form sets of sets. For example, the machine may abstract the number 3 as the class of all triples. The concept "one greater than" can be defined as the set of all pairs of sets, such that the first set is the set of all n-tuples; the second set, the set of all (n+1) - tuples.

The arithmetic learning operations Solomonoff has and intends to investigate are

identity

= 1001
1001

Boolean product

⊗ 0011
0101
0001

complement

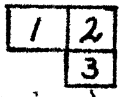
1001
0110

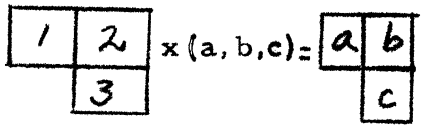
addition

\+ 1001
1101
10010
10110

Boolean sum

⊕ 0011
0101
0111

An example of a structure S_i has the form 

An example of an n-tuple N_j has the form (a, b, c).
An n-gram is the product of a structure and an n-tuple 

The utility U_{ij} of an n-gram $S_i \times N_j$ is found from the sum of the utility U_{S_i} of the structure S_i and the utility U_{N_j} of the n-tuple N_j. One of the problems is to find values of U_{S_i} and U_{N_j} that when added will give a good approximation value for U_{ij}, the empirically observed utility of the n-gram $S_i \times N_j$.

Summary

Newall and Simon, and More have approached the problem of heuristic from the point of view of formal rules of inference. In this approach, More has found that his program is guessing in a meta-language, rather than in the object language. Now he is searching for a way to handle several levels of language at once. Levels of language in this respect are different from levels of macro-instructions used in writing programs.

Craik and Minsky have approached heuristic from the point of view of constructing in a flexible memory models of the environment.

Robinson has approached heuristic in mathematics from a hierarchy of meta-languages.

Thus far, these are the principle approaches to a heuristic for problem solving.

Solomonoff, Minsky, and More spent Wednesday evening of July 25, 1956, working on the meaning of comparing two machines. Solomonoff contended that one builds a comparator that looks at two machine-environment complexes, the comparator being essentially a two place predicate. Minsky and More contended that the comparator must look at two machines and an environment common to both, the comparator being essentially a three place predicate. More approached the subject from his symmetric model of a machine and suggested that in complicated machines, the comparator might observe only machine effects on the environment, rather than the internal operation of each machine. Minsky suggested an example of machine comparison showing the necessity of including the environment in the third place of a predicate, and More found a numerical realization of the example.

Levels of Complexity in Machine, Program and Language

T. More, Jr.

Machine

component
circuit
unit
system

Program

(IBM
Tm. (big new super computer - 60 bit storage registers.
micro instruction say Boolean functions.
regular instruction — transfer, clear and add; mult.
macro instruction — Stretch, 704 mod. 3
find Pr. average of; dot multiply $\vec{A}$ and $\vec{B}$;
subroutine
routine
assembly routine
interpretive routine
compiling routine
program writing routine
block diagram
observe and learn routine
routine library

Language

object language
formal language
meta-language
semantics
elementary syntax
theoretical syntax
mathematics
scientific languages
vernacular
English language
world languages
language translation
universal language

{natural deduction} {games} {business operation programming} {formula translation}

The Primitive Symbols of a Modal Natural Deductive Pure Functional Calculus of First Order

T. More, Jr.

Primitive Symbols

Improper Symbols

it would seem that these are unnec.

Association Symbols

( ) , * (read "open paren", "close paren", "comma", "star")

Connectives

$\sim$ $\wedge$ $\rightarrow$ (read "not", "and", "from---infer---")

Operators

$\forall$ (read "for all") $\forall, \exists, S, D, \Pi, \Sigma$.

re. similarity between these 2 kinds of ops. isn't too clear.

Proper Symbols

Constants

Individual Constants (none initially)
Propositional Constants (none initially)
Singulary Functional Constants (none initially)
n-ary Functional Constants (none initially)

Variables

Individual Variables

$w\ x\ y\ z\ w_1\ x_1\ y_1\ z_1\ w_2$ ---

Propositional Variables

$p\ q\ r\ s\ p_1\ q_1\ r_1\ s_1\ p_2$ ---

Singulary Functional Variables

$F^1\ G^1\ H^1\ F^1_1\ G^1_1\ H^1_1\ F^1_2$ ---

$f(x)$; x is sick ; $f(\ )$; $(\ )$ is sick.

n-ary Functional Variables

$F^n\ G^n\ H^n\ F^n_1\ G^n_1\ H^n_1\ F^n_2$ ---

(Each category of variables is listed here in alphabetic order. We adopt that F, G and H may be any of these functional variables)

the convention

The Formation Rules of a Modal Natural Deductive Pure Functional Calculus of First Order

T. More, Jr.

Formation Rules (wf = well-formed)
(wff = well-formed formula)

1. A propositional variable (or propositional constant) is a wff.
   Example: p

2. If f is an n-ary functional variable (or n-ary functional constant), and if a_1, ---, a_n are individual variables (or individual constants or both mixed and not necessarily all different), then $f(a_1, ---, a_n)$ is a wff.
   Examples: F (x) ; G (x, y)

3. If C is wf then $\sim$ C is wf.
   Examples: $\sim p$; $\sim\sim F(x)$; $\sim(p \wedge q)$; $\sim\forall x\ G(x, y)$

4. If C and D are wf then (C $\wedge$ D) is wf.
   Examples: $(p \wedge q)$; $(\sim p \wedge \forall x \sim F(x))$

5. If C and D are wf then (C $\rightarrow$ D) is wf.
   Examples: $(p \rightarrow q)$; $(\sim p \rightarrow \sim\forall x \sim F(x))$.

6. If C is wf and b is an individual variable then $\forall$ b C is wf.
   Examples: $\forall x\, p$; $\forall y\ H(x, y)$;
   $\forall x\ \forall y\ \sim \forall z \sim (F(x, y, z) \rightarrow \sim G(x, y))$.

7. If (C) is a wff standing alone (not a wf part of a larger wff), then C is a wff only when standing alone. (Redundant parentheses outside the total formula may be removed by this rule.)
   Example: '((p $\wedge$ q) $\rightarrow$ p)' becomes '(p $\wedge$ q) $\rightarrow$ p'

8. If C is a wff standing alone then ---*C is a wff only when standing alone.
   (The dots indicate a zero or finite number of stars.)
   Examples: *p; *** $\forall$xF (x); **p $\rightarrow$ q; * $\sim$ q

The Axiom Schemata of a Modal Natural Deductive Pure Functional Calculus of First Order

Let A, B and C be syntactical variables whose range is the well-formed formulas of the object language.

Let a and b be syntactical variables whose range is the individual variables of the object language.

Let $S^b_B A|$ be the result obtained by substituting B for all free occurrences of b in A.

Axioms of the propositional calculus.

$(A \wedge B) \to A$ example $(p \wedge q) \to p$

$(A \wedge B) \to B$

$A \to \sim\sim A$ example $p \to \sim\sim p$

$(\sim A \to \sim B) \to (B \to A)$

$(A \wedge \sim(A \wedge \sim B)) \to B$

Added axioms for the functional calculus.

$\forall a (A \to B) \to (A \to \forall a B)$

where a is not a free variable of A.

Examples $\forall x (p \to F(x)) \to (p \to \forall x F(x))$

$\forall z (F(x,y) \to G(z)) \to (F(x,y) \to \forall z G(z))$

$\forall a A \to S^a_b A|$

where no free occurrence of a in A is in a well-formed part of A of the form $\forall b C$.

Examples $\forall x F(x) \to F(y)$

$\forall x F(x) \to F(x)$

$\forall x G(x,y) \to G(y,y)$

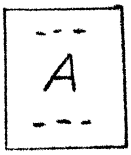
The Rules of Inference of a Modal Natural Deductive Pure Functional Calculus of First Order

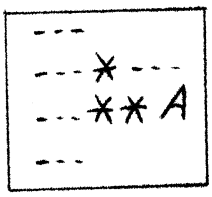
The squares below may be thought of as windows in a deductive proof. The dots standing alone represent an undetermined finite number of initial, intermediate, or terminal lines in the deductive proof. The symbol '---*' represents a zero or finite number of stars. The number of stars in ' ---* ---* ' is greater than or equal to the number of stars in ' ---* '. Let A and B be syntactical variables whose range is the well-formed formulas of the object language.

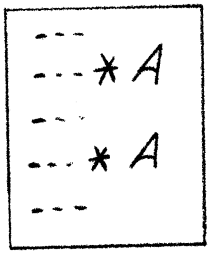
Schematic Rules of Inference or "Stencils"

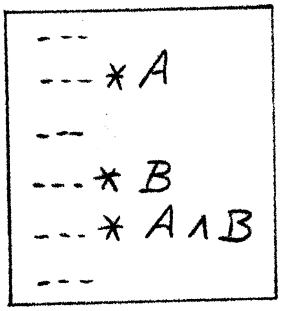
--- where A A is an axiom --- or theorem	Rule of axiom and theorem introduction An axiom or theorem may be introduced at any line in a deductive proof.
--- ---*--- ---**A ---	Rule of assumption introduction An arbitrary assumption may be introduced at any line provided only that the assumed wff begins a new column.
--- ---*A ---*B ---*A ∧ B ---	Rule of conjunction introduction Two consecutive lines containing A and B in the same column may be abbreviated to a conjunction.
--- ---*--- ---**A --- ---**B ---*A ⊰ B ---	Rule of strict implication introduction or rule of assumption elimination If A is assumed and later B is derived from A in column n+1, then the derivation may be summarized in column n. The column n+1 is thus terminated.
--- ---*A ⊰ B ---*---*A --- ---*---*B ---	Rule of modus ponens or rule of strict implication elimination If it is known in an earlier column depending on fewer assumptions that B is inferred form A, then in any later column from A we may infer B in a later line. The arbitrary number of intermediate lines between A and B allows vertical mobility in a deductive proof.
--- where A A is an axiom ∀aA or theorem ---	Rule of generalization or rule of universal quantification introduction If A is an axiom or theorem, as represented by being in column zero, then A is universally a theorem with respect to any individual variable.

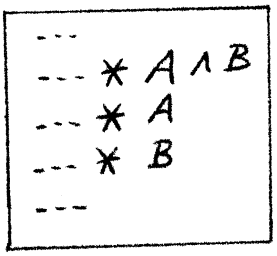
Initial Machine Rules of Inference

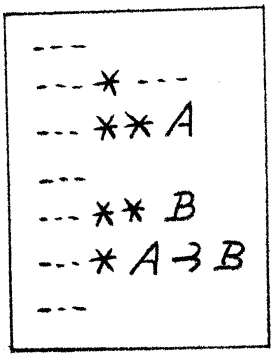


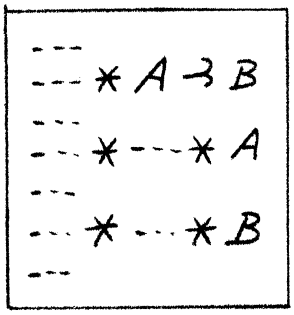


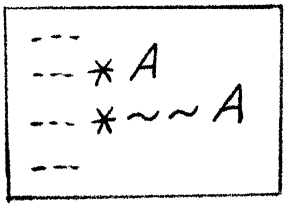


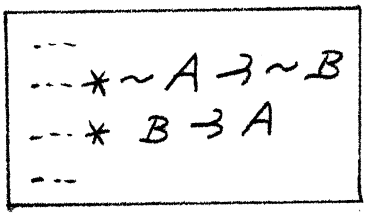


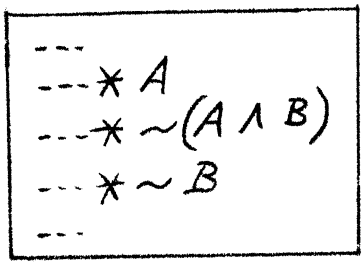


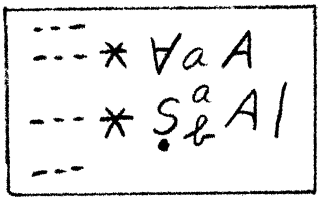




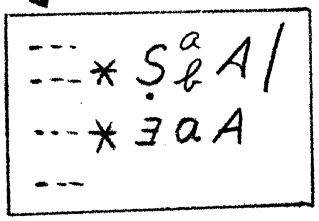




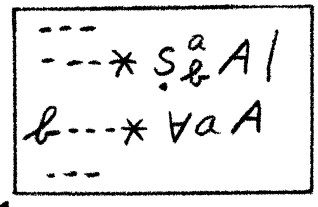




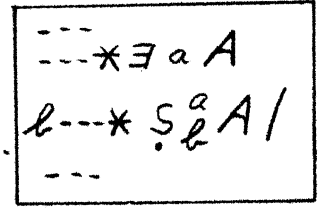
↖ where no free occurrence of a in A is in a wf part of A of the form $\forall bC$



No variable may be flagged more than once in a deductive proof. No flagged variable may occur free in the last line of a finished deductive proof.



↖ where b is alphabetically later than all free individual variables in $\forall aA$ or $\exists aA$



Definition Schemata

$A \equiv B$	$\rightarrow (A \rightarrow B) \wedge (B \rightarrow A)$	equivalent
$A \vee B$	$\rightarrow \sim(\sim A \wedge \sim B)$	or
$A \supset B$	$\rightarrow \sim(A \wedge \sim B)$	if --- then ---
$A \equiv B$	$\rightarrow (A \supset B) \wedge (B \supset A)$	if and only if
$\exists a\, A$	$\rightarrow \sim \forall a \sim A$	there exists
0	$\rightarrow A \wedge \sim A$	null
I	$\rightarrow \sim 0$	realm
$\Diamond A$	$\rightarrow \sim(A \rightarrow 0)$	possible
$\Box A$	$\rightarrow I \rightarrow A$	necessary

Infrequently used definitions

$A \leftarrow B$	$\rightarrow B \rightarrow A$	inferred from
$A \subset B$	$\rightarrow B \supset A$	if
$A \circ B$	$\rightarrow \Diamond(A \wedge B)$	consistent

$A \tilde{m} B \rightarrow \sim(A\, m\, B)$ where m is any binary connective.

Symbols

$\rightarrow$	$\leftarrow$	$\leftrightarrow$	&	or	intuitive
$\Rightarrow$	$\Leftarrow$	$\Leftrightarrow$	$\wedge$	$\vee$	propositional
$\supset$	$\subset$	$\equiv$	$\cap$	$\cup$	set or class
$\rightarrow$	$\vdash$	$\equiv$	$\frown$	$\smile$	lattice
$\vdash$	$\dashv$	$\#$	$\cdot$	$+$	switching
$\geq$	$\leq$	$=$			
$\subseteq$	$\supseteq$				

Various Proofs in the Deductive System

Categorical proof of $\sim\sim p \to p$

$\sim p \to \sim\sim\sim p$ axiom

$(\sim p \to \sim\sim\sim p) \to (\sim\sim p \to p)$ axiom

$\sim\sim p \to p$ modus ponens

Hypothetical machine-proof of $\sim\sim p \to p$

$* \sim\sim p$
$** \sim p$
$** \sim\sim\sim p$
$* \sim p \to \sim\sim\sim p$
$* \sim\sim p \to p$
$* p$
$\boxed{\sim\sim p \to p}$

$* p$
$* \sim\sim p$
$\sim\sim p \to p$
$* p$
$\boxed{p \to p}$

$* p \to q$
$** \sim q$
$*** \sim\sim p$
$\sim\sim p \to p$
$*** p$
$*** q$
$*** \sim\sim q$
$** \sim\sim p \to \sim\sim q$
$** \sim q \to \sim p$
$** \sim p$
$* \sim q \to \sim p$
$\boxed{(p \to q) \to (\sim q \to \sim p)}$

$*(p \wedge q) \wedge r$
$* p \wedge q$
$* r$
$* p$
$* q$
$* q$
$* r$
$* p$
$* q \wedge r$
$* p \wedge (q \wedge r)$
$\boxed{((p \wedge q) \wedge r) \to (p \wedge (q \wedge r))}$

$* p \to (q \to r)$
$** p \to q$
$*** p$
$*** q$
$*** q \to r$
$*** r$
$** p \to r$
$*(p \to q) \to (p \to r)$
$\boxed{(p \to (q \to r)) \to ((p \to q) \to (p \to r))}$

JUMPING COLUMNS

$* p$
$** q$) fallaciously
$** p$
$* q \to p$
$p \to (q \to p)$

Further Examples of Proof

$* \forall x(F(x) \supset G(x))$
$* F(y) \supset G(y)$
$** \forall x F(x)$
$** F(y)$
$** G(y)$
$y ** \forall x G(x)$
$* \forall x F(x) \supset \forall x G(x)$

$$\boxed{\forall x(F(x) \supset G(x)) \supset (\forall x F(x) \supset \forall x G(x))}$$

$* \forall x(F(x) \wedge G(x))$
$* F(y) \wedge G(y)$
$* F(y)$
$y * \forall x F(x)$
$* F(z) \wedge G(z)$
$* G(z)$
$z * \forall x G(x)$
$* \forall x F(x) \wedge \forall x G(x)$

$$\boxed{\forall x(F(x) \wedge G(x)) \supset (\forall x F(x) \wedge \forall x G(x))}$$

Stencil for proof by contradiction

--- $*$ ---
--- $** A$

--- $** O$
--- $* A \supset O$
--- $* I \supset \sim A$
I
--- $* I$
--- $* \sim A$

Organization of logical registers

	10	6	10	10 ← # of bits in each field
( ) [	( )	( )	( )	( )]
loc	ant	con	suc	ref
location	antecedent	connective	succedent	reference

→ ( ) [( ) ( ) ( ) ( )] → ( ) [( ) ( ) ( ) ( )] →

Example of storage

2 4 1 3 6 5 7 8 9
$\sim p \supset \sim (p \wedge \forall x F(x))$

(1) [(2) ($\supset$) (3) (2)]
(2) [(D) ($\sim$) (4) (3)]
(3) [(D) ($\sim$) (5) (4)]
(4) [(D) (p) (S) (5)]
(5) [(6) ($\wedge$) (7) (6)]
(6) [(D) (p) (S) (7)]
(7) [($\forall$) (x) (8) (8)]
(8) [(J) (F) (9) (9)]
(9) [(7) (x) (S) (1)]

6 July 56
T. More

MACHINE LETTER	LOGICAL LETTER		
A	$\forall$	"for every"	universal quantifier
B			
C	$\Rightarrow$	"if then"	conditional
D	$\Leftrightarrow$	"if and only if"	biconditional
E	$\exists$	"there exists"	existential quantifier
F	F	} predicate letters	
G	G		
H	H		
I	I	"realm"	tautological truth
J	$\Diamond$	"possible" (modal)	
K	$\wedge$	"and"	conjunction
L	$\vdash$	"infers"	deducibility
M	$\dashv\vdash$	"equivalent"	mutual deducibility
N	$\sim$	"not"	negation
O	O	"null"	tautological falsity
P	$\mathcal{P}$	} sentential letters	
Q	$\mathcal{Q}$		
R	$\mathcal{R}$		
S	$\mathcal{S}$		
T	T	"true"	contingent truth
U	U	"false"	contingent falsity
V	$\vee$	"or"	disjunction, alternation
W	w	} variable letters	
X	x		
Y	y		
Z	z		

Propective logical letter assignment 15 June 56

T. Moore, J.

internal unpunched code letters

other letter wheel assignments

MACHINE		FORTRAN LETTERS			LOGICAL LETTERS	
0		0	0	digits	0	digits
1		1	1		1	
2		2	2		2	
3		3	3		3	
4		4	4		4	
5		5	5		5	
6		6	6		6	
7		7	7		7	
8		8	8		8	
9		9	9		9	
0+	&	+	+	plus	+	plus
	@	-	output	minus	-	minus
0-		—	input			
		.	.	decimal point	.	
		*	×	times	*	star (for column indication)
		/	/	divided by	/	divides
	%	(	(	open paren	(	open paren
	¤	)	)	close paren	)	close paren
	#	=	=	equals	=	identity
		,	,	comma	,	comma
			not allowed		$	successor
		$	on input			
+ +		BLANK	BLANK		BLANK	

Total number of available letters is 48

There are three type wheel assignments

Commercial	#	&	@	%	¤
General purpose	+	+	—	%	¤
Fortran	=	+	—	(	)