1116 1

THE UNIVERSITY OF TEXAS AT AUSTIN
AUSTIN, TEXAS 78712

Department of Mathematics
RLM 8-100

Area Code 512
471-3361

August 30, 1982

Mr. C. W. Egberts
Principal Specialist
CAD/CAM Systems, Dept. K310
McDonnell Douglas Automation Company
P. O. Box 516
St. Louis, Missouri 63166

Dear Mr. Egberts:

Here finally is some material for your report, much too much. Please cut it down as you see fit. I did not have it all typed (yet) because that would further delay getting it to you. Call me at 512-471-1242 if you have questions.

Sincerely,

W W Bledsoe

W. W. Bledsoe

WWB:b

AUTOMATIC THEOREM PROVING

by Woody Bledsoe, University of Texas at Austin

A talk given before the Advanced Technical Planning Committee of CAU-I, in Dallas, Texas, August 24, 1982.

There is an active group of ATP researchers at UT-Austin under the direction of Woody Bledsoe, Bob Boyer, J. Moore, and Frank Brown, which includes 6 professors and about 12 graduate students. Also, Don Good heads a large group there working on Program Verification (the Gypsy project). Other AI researchers at UT-Austin include Bob Simmons (text understanding), Gordon Novac (Automatic Physics programs), and Elaine Rich (Expert Systems).

Woody Bledsoe and Michael Ballantyne are studying the feasibility of establishing an AI Laboratory at the Woodlands, a new city north of Houston. This laboratory (WAIL), if established, would be funded by the Mitchell Energy Corporations and other corporations in the Houston area. This would be part of HARC (Houston Area Research Center) at the Woodlands, which has already been given 110 acres of land and several million dollars.

A good introduction to ATP can be found in <u>What Can Be Automated</u>?, ed. Bruce Artin, MIT Press 1980, pp. 448-462. This appears as (*) in the list of references given here.

ATP Reference 20 Jan. 1981

BOOKS

Chang and Lee. Symbolic Logic and Mechanical Theorem Proving.
(see List of Reference in this book.)

Donald Loveland. Automated Theorem Proving: A Logical Basis. Careful definitions and proofs, especially on Resolution.

Nils Nilsson. Principles of Artificial Intelligence. Chaps. 4-6.

J. A. Robinson. Logic, Form and Function. (The inventor of the term RESOLUTION.) Especially Chapters 11-13.

Robert Kowalski. Logic for Problem Solving. (The inventor of Logic Programming, SL-RESOLUTION, Resolution graphs, etc.).

Boyer and Moore. A Computational Logic. The "Boyer-Moore" system.

Feigenbaum and Feldman. Computers and Thought. Section 3. Early papers on ATP by: Newell, Simon, and Shaw, and Gelernter and Loveland.

J. Siekmann and G. Wrightson. Collected Papers on Automatic Theorem Proving. Forthcoming from Springer. Three volumes. (Martin Davis' history of ATP will start the first volume; W. Bledsoe will write such a history for the third volume.)

INTRODUCTION

*) "Automatic Theorem Proving" (by W.W.Bledsoe), in What Can Be Automated? (NSF COSERS Study). Ed. Bruce Artin, MIT Press 1980, pp. 448-462.

*) "Non-resolution Theorem Proving," W. W. Bledsoe, A. I. Journal 9 (1977), 1-35.

JOURNALS

International Journal of Artificial Intelligence. (AI Jour.)

Journal of the Association for Computing Machinery. (JACM)

Machine Intelligence (MI-1 - MI-9)

IEEE Transactions on Computers

CONFERENCE REPORTS

Proceedings of the Fourth Workshop on Automated Deduction, Austin, Texas, Feb. 1-3, 1979.

5th Conference on Automated Deduction, Les Arcs, France, July 8-11, 1980.

6th Conference on Automated Deduction, New York, June 7-9, 1982.

Lecture Notes in Compurer Science 87, 138, Springer-Verlag 1980, 1982

Proceedings of IJCAI 1969-1981.

Proceedings of AAAI National Conferences, AAAI, 445 Burgess Dr., Menlo Park, California, 94025

What is ATP?

Proving theorems Automatically (by computer)

e.g., Pythagorean Theorem

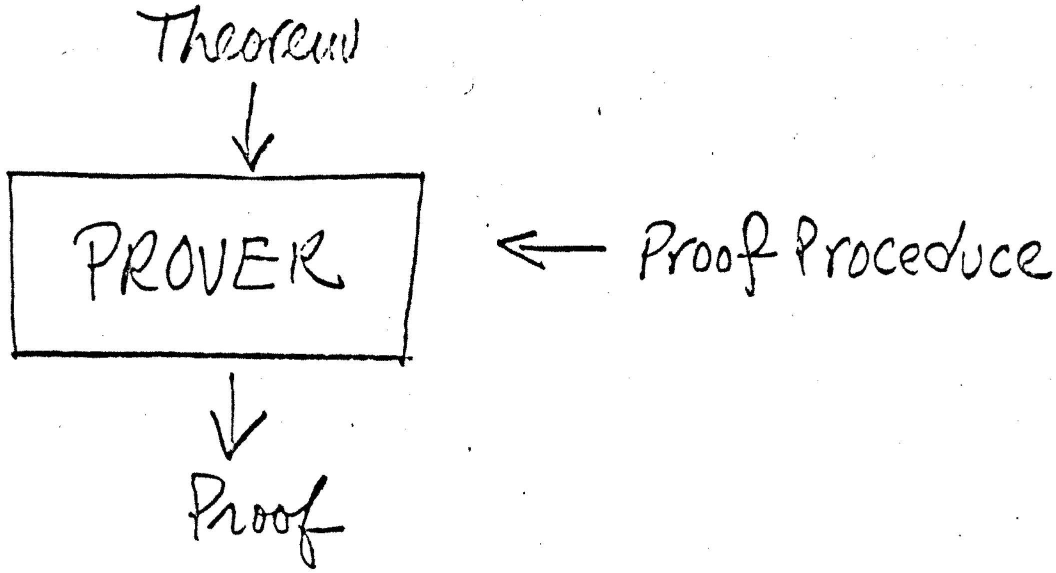
Heine-Borel Theorem

Schroeder Bernslein Theorem, etc.

new theorems

What has been done? Later

How? "



- SOUND : Does not "prove" non-theorems

- COMPLETE : Proves <u>all</u> theorems
  (semi-decision procedure)
  But may never finish on a non-theorem

- DECISION PROCEDURE :
  Can decide whether any formula
  is a theorem or not

Applications of ATP

Program Verification

esp. Man-machine theorem proving and proof checking

Now in use, somewhat (ISI, UT, Stanford, UT, --)

Program Synthesis (Waldinger-Manna, Balzer, ...)

Data Base Inference

Very important, but needs work

See Minker, et al. book (logic and data bases)

Truth maintenance?

Probabilistic inference

Logic Programming (Kowalski, et al.)

Expert Systems (MYCIN, PROLOGUE, PROSPECTOR, etc.)

Mathematics

Proof checking (see slide)

Man-machine (Assistant")

File of theorems

Any automatic decision maker

PROOF-CHECKING

J. Morris - "all" set theory theorems in A. P. Morse's book.

de Bruijn - all of Landau's book.

Boyer-Moore - prime factorization theorem, etc.

PV projects at UT, ISI, Stanford, SRI, ...

Suppes-Kreisel - CAI course in Set Theory to Gödel's Incomp. Th.

.
. Weyhrauch - FOL
.

Excellent application area, but has not been done right.

The user has to bend to the computer (let's change that).

Interesting, challenging, open problem.

Neveln - current APC project at UT.

These three pages are from Reference (*).

One of the earliest ATP programs was Galernter plane geometry prover. For example,

<u>Theorem</u>. Two vertices of a triangle are equidistant from the median to the side determined by those vertices.

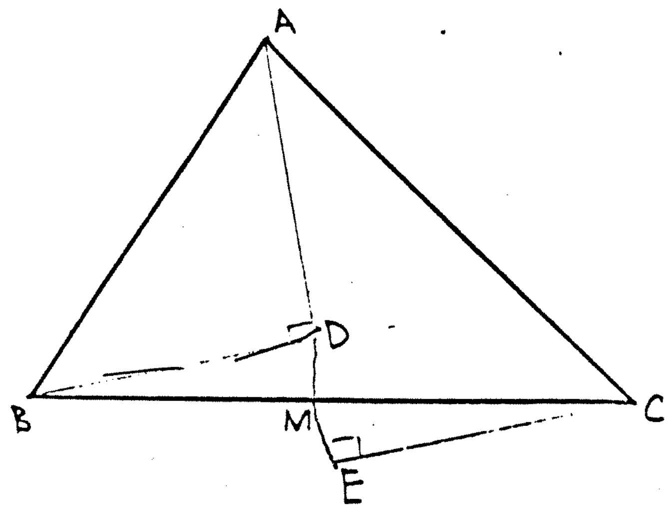


Figure 1

GIVEN: Segment BM = Segment MC, BD ⊥ AM, CE ⊥ ME.

GOAL: Segemtn BD = Segment EC.

SOLUTION:

Angle DMB = Angle EMC	Verticle Angles
Angle BDM = Angle CEM	Right Angles are equal
Segment BM = Segment MC	Given
CEM is a triangle	Assumption based on diagram
BDM is a triangle	Assumption based on diagram
$\triangle$ CEM $\cong$ $\triangle$ BDM	Side-angle-angle
Segment BD = Segment EC	Corresponding elements of congruent triangles

This is the machine's proof though we have omitted some of its steps for simplicity of presentation.

In this proof the machine proceeds ("reasoning backwards") as follows:

Its goal is

G1 Segment BD = Segment EC .

So it consults a list of solutions for this type of goal and finds (among others), that two segments can be proved equal by showing that they are corresponding parts of congruent triangles. Since BD is in Δ BDM and EC is in Δ CEM, it selects the subgoal

G2 Δ CEM $\approx$ Δ BDM .

Now it consults another list for ways of proving two triangles congruent. It finds: (a) three-sides, (b) side-angle-side, (c) side-angle-angle. It sets the subgoal

G3 "three-sides" for Δ's CEM and BDM .

This fails (after a good deal of work). So it sets the subgoal

G5 "side-angle-angle" .

There are several ways this can be achieved, one of which requires the three subgoals

G6 Segment BM = Segment MC

G7 Angle DMB = Angle EMC

G8 Angle BDM = Angle CEM .

The machine finds subgoal G6 among its premises. In solving subgoals G7 and G8 it consults a list of methods for making two angles equal, and finds (among others): "verticle angles are equal", and "all right angles are equal." Since it detects from the diagram that angles DMB and EMC are verticle angles, and that angles BDM and CEM are right angles it successfully concludes the proof of subgoals G6 - G8, and therefore G5, G2 and G1. In the second step of the proof when subgoal G2 was selected, the machine could have selected any of the following subgoals:

G2.1	Δ CEA $\approx \Delta$ BDA
G2.2	Δ CEA $\approx \Delta$ BDM
G2.3	Δ CEM $\approx \Delta$ BDA
G2	Δ CEM $\approx \Delta$ BDM .

But, by constructing in its memory a "general" diagram of the situation (which is its representation of the drawing in Figure 2), the machine easily checked by <u>measurements</u> that subgoals G2.1 - G2.3 could not be true, but that G2 seems alright. Thus it selected only subgoal G2 and thereby drastically reduced the search time.

This idea of filtering out false subgoals is generalized and used in many areas of automatic theorem proving. For example, in group theory a false subgoal can be discarded by testing it on known groups (such as the Klein four groups).

…ematics can be divided as follows:

FIRST ORDER LOGIC		HIGHER ORDER LOGIC
PROPOSITIONAN LOGIC		

↑
Decision Procedure

$\underbrace{\text{FIRST ORDER LOGIC}}_{\text{Complete}}$ $\underbrace{\text{HIGHER ORDER LOGIC}}_{\text{incomplete}}$

- PROPOSITIONAN LOGIC – Rather trivial

  eg $[P \wedge (P \rightarrow Q) \wedge (\sim Q \vee R) \rightarrow R]$

  No quantifiers: $(\forall, \exists)$

  Easily handled by Computers. (Tr Tab)

- FIRST ORDER LOGIC – difficult
- HIGHER " " – more "

FIRST ORDER LOGIC

- Examples See next two pages.
- Quantification of individual variables
- This is the challenge of this age, to prove all theorems in first-order logic.
- A complete proof procedure was devised by Herbrand in 1930

Herbrand Procedure

So we are finished?

No. It was too slow!

- Can methods be devised so that computer provers can compete with humans? surpass them?

That's the challenge.

Where are we now?

<u>Theorem</u>. $a \le l \le b \wedge a \le t_0 \le l$

$\wedge \forall \varepsilon(\varepsilon > 0 \rightarrow \exists r(r < l \wedge \forall s(r \le s \le l \rightarrow f(l) \le f(s) + \varepsilon$
$\wedge \forall y(a \le y < l \rightarrow \exists z(y < z \le l \wedge \forall t(a \le t < z \rightarrow f(z) \le f(t)$

$\longrightarrow f(l) \le f(t_0))$

<u>Theorem</u>. The sum of two continuous functions continuous:

$\forall \varepsilon(\varepsilon > 0 \rightarrow \exists \delta(\delta > 0 \wedge \forall y(|x_0 - y| < \delta \rightarrow |f(x_0) - f(y)| \le \varepsilon$
$\wedge \forall \varepsilon(\cdot \quad - \qquad\qquad |g(x_0) - g(y)| \le \varepsilon)$
$\wedge \ \varepsilon_0 > 0 \longrightarrow$

$\exists \delta(\delta > 0 \wedge \forall x(|x_0 - x| < \delta \rightarrow |[f(x) + g(x)] - [f(x_0) + g(x_0)]| < \varepsilon_0$

Example : AM5

C1. $\forall x \in [a,b]\ \forall \varepsilon > 0\ \exists r < x\ \forall s\,(r \le s \le x \rightarrow f(x) \le f(s) +$

C2. " " $\exists r > x\ \forall s\,(x \le s \le r \rightarrow$ "

LUB1. $\forall x \in [a,b]\ \big(\forall t\,(a \le t \le x \rightarrow f(t) \le f(x)) \longrightarrow x \le \ell\big)$

LUB2. $\forall y\,(y < \ell \longrightarrow \exists z \in [a,b]\ [\forall t\,(a \le t \le z \rightarrow f(t) \le f(z)) \wedge y < z \le$

GLB1. $\forall w \in [a,b]\ \exists g\ \forall x \in [a,b]\ (f(x) \le f(w) \longrightarrow g \le x)$

GLB2. $\forall w \in [a,b]\ \exists g\ \forall y\,(g < y \longrightarrow \exists z \in [a,b]\,(f(z) \le f(w) \wedge g \le z < y)$

<u>Theorem</u> (AM5) $a \le b \wedge C1 \wedge C2 \wedge LUB1 \wedge LUB2 \wedge GLB1 \wedge GLB$

$$\longrightarrow \exists u \in [a,b]\ \forall t \in [a,b]\ (f(u) \le f(t)).$$

CHRONOLOGY

1930	"Herbrand Procedure"	
		(also Skolem Presburger, etc.)
/ / / / / /	COMPUTERS / / / / / / / / / /	
1955	Logic Theorist	NSS Rand (Principia Math.)
1959	Geometry Machine	Gelernter
1960	Herbrand Procedure Won't Work. Too Slow.	Gilmore
1960-65	Improved Hilbert Procedure	Davis, Putnam, Prowitz, Russians
"	Wang's System	
1965	RESOLUTION	J. A. Robinson efficient, excitement
1965-70	Refinements of RESOLUTION	
1970	"Natural Deduction Systems"	Bledsoe, Nevins, C. Hewitt, Loveland, etc.
1970's	Both types Applications	

- Newell-Simon-Shaw (Chap. 2 of (1) 38 of 52 theorems)

  Wang

  (all of (1); > 350 Theorems)

  (1) Principia Mathematica (Whitehead & Russell)

  (2) Propositional Logic

- Gelernter

  A number of theorems in plane geometry

  (not requiring constructions)

Some Theorems Proved Automatically

19

- $x^3 = x$ (in a Ring) $\longrightarrow$ The RING is Commutative
- Unique factorization Theorem (with some input Lemmas)
- The sum of two continuous functions is continuous (δ, ε)
- Intermediate Value Theorem

* f continuous on $[a,b] \longrightarrow f$ u. cont. on $[a,b]$
* using non-standard analysis
(+ similar Theorems: Bolzano Weierstrass, etc)

SOME TO PROVE

- Schroeder-Bernstein Theorem

* f cont on $[a,b] \longrightarrow f$ u. cont on $[a,b]$
* without Non-stand An

- Heine Borel Theorem
- Hahn-Banach Theorem

Two main Types of Provers (to be discussed shortly).

- RESOLUTION
- NATURAL DEDUCTION

Two Types of ACTIVITY (of Research)

- Devising Proof procedures and testing them, using Computers
- Proving Completeness results (mathematics

OR

- Man-machine
- Machine alone

Much of the research in ATP during the last fifteen years has been stimulated by J. A. Robinson's introduction of RESOLUTION in 1965 (see the books by Chang and Lee, Loveland, or Robinson). A succinct easy-to-read, introduction in RESOLUTION is given in Reference (*).

Another kind of ATP research utilizes the "Natural Deduction" Method (see reference (**)).

Natural Deduction is governed by a set of (production) rules. They use the implication symbol "$\rightarrow$". For example,

John is a boy $\rightarrow$ John is a male,

or more generally

$$P \rightarrow Q$$

where P and Q are statements which are either true or false.

Some Natural Deduction Rules

(for the Ground Case - no variables)

$$\frac{P \;,\; P \rightarrow Q}{Q}$$

$$\frac{P \rightarrow Q \;,\; P \rightarrow R}{P \rightarrow Q \,\&\, R}$$ AND-SPLIT

$$\frac{R \rightarrow S \;,\; P \rightarrow Q}{P \,\&\, (Q \rightarrow R) \rightarrow S}$$ BACK CHAIN

$$\frac{P \rightarrow R \;,\; Q \rightarrow R}{P \text{ or } Q \rightarrow R}$$ CASES

$$\frac{\quad}{P \rightarrow P}$$ match

~~John is a boy → John is Male~~

~~P → Q~~

When variables are admitted, we have expressions of the form

$$\text{For all } x \quad (x \leq 0 \rightarrow x < 1).$$

and write this

$$\forall x \ (x \leq 0 \rightarrow x < 1).$$

Thus we use the symbol "$\forall$" as a shorthand for "for all", and similarly use "$\exists$" for "for some". The following rules for the IMPLY natural deduction prover, are taken from reference (**).

NATURAL SYSTEM

$$(H \Rightarrow G)$$

H IS A SET OF HYPOTHESES.

G IS A GOAL.

TO FIND A SUBSTITUTION θ FOR WHICH

$$(H\theta \rightarrow G\theta)$$

IS A VALID PROPOSITIONAL FORMULA.

<u>EXAMPLE</u>.

$$P(A) \wedge (P(x) \rightarrow Q(x)) \Rightarrow Q(A)$$

ANSWER: $\theta \equiv A|x$

25

IMPLY RULES

A PARTIAL SET FROM [12]

I4. $(H \Rightarrow A \wedge B)$ "SPLIT"

IF $(H \Rightarrow A)$ RETURNS θ

AND $(H \Rightarrow B\theta)$ RETURNS λ

THEN RETURN $\theta \circ \lambda$

I3. $(H_1 \vee H_2 \Rightarrow C)$ "CASES"

IF $(H_1 \Rightarrow C)$ RETURNS θ

AND $(H_2\theta \Rightarrow C)$ RETURNS λ

THEN RETURN $\theta \circ \lambda$

I5. $(H \Rightarrow C)$

PUT $C' := \text{REDUCE}(C);\ H' := \text{REDUCE}(H)$

CALL $(H' \Rightarrow C')$

I7. $(H \Rightarrow (A \rightarrow B))$ "PROMOTE"

CALL $(H \wedge A \Rightarrow B)$.

I15. $(H \Rightarrow C)$

PUT $C' := \text{DEFINE}(C)$

CALL $(H \Rightarrow C')$

H2. $(H \Rightarrow C)$ "MATCH"

IF $H\theta \equiv C\theta$, RETURN θ.

H6. $(A \wedge B \Rightarrow C)$ "OR-FORK"

IF $(A \Rightarrow C)$ RETURNS θ (NOT NIL), RETURN θ.

ELSE CALL $(B \Rightarrow C)$

H7. $H \wedge (A \rightarrow D) \Rightarrow C$ "BACK-CHAIN"

IF $(D \Rightarrow C)$ RETURNS θ,

AND $(H \Rightarrow A\theta)$ RETURNS λ,

THEN RETURN $\theta \circ \lambda$

H9. $H \wedge (A = B) \Rightarrow C$ "SUB = "

PUT $A' :=$ CHOOSE (A, B), $B' :=$ OTHER(A, B)

CALL $(H(A'/B') \Rightarrow C(A'/B'))$.

$(\forall x \; P(x) \longrightarrow P(a))$

Skolemize (*Eliminate Quantifiers, $\forall$, $\exists$*)

$(P(x) \Longrightarrow P(a))$

Returns a/x H2

h. $(Q \wedge P(a) \longrightarrow \exists x\,(P(x) \wedge Q))$

$(Q \wedge P(a) \Longrightarrow P(x) \wedge Q)$

AND-SPLIT I4

1) $(Q \wedge P(a) \Longrightarrow P(x))$

$(Q \Longrightarrow P(x))$ H6

FAILS

$(P(a) \Longrightarrow P(x))$

Returns a/x H2

2) $(Q \wedge P(a) \Longrightarrow Q)$

$(Q \Longrightarrow Q)$ H6

Returns "T" H2

Returns a/x.

~~Example Using IMPLY~~

<u>Theorem</u>. $(P(A) \wedge \forall x(P(x) \rightarrow Q(x)) \rightarrow Q(A))$.

$P(A) \wedge (P(x) \rightarrow Q(x)) \Rightarrow Q(A))$ — Eliminate Quantifier

$(P(A) \Rightarrow (Q(A))$ — H6

FAILS

$(P(x) \rightarrow Q(x)) \Rightarrow Q(A))$

$(Q(x) \Rightarrow Q(A))$ — H7

Return $\theta \equiv A|x$ — H2

$(P(A) \Rightarrow P(x)(A|x))$

Return TRUE.

Returns $A|x$.

Boyer-Moore (UT-Austin)

RECURSIVE FUNCTION PROVER

e.g. Proving Theorems about LISP functions

EX. ORDERED (SORT L)

For Hard theorems, the user suggests a series of lemmas which it proves (like proof-checking)

Ex. Prime Factorization Theorem

Ex. "Verified" a simple compiler for algebraic expressions (McCarthy)

Ex. Halting Problem (unsolvability) 1982

.
.
.

Applications: PV, Proof-checking, related to programming

Uses: Induction, Generalization, etc., etc.

ATP is a part of AI, but more than that.

Earliest Provers had AI features

(1) knowledge base

(2) reasoning rules

Later provers tended toward (2) alone.

Why is there still a problem?

Why not use EMYCIN and TEIRESIAS?

Ans.: These (EMYCIN and TEIRESMS) are best for applications needing

(1) much expert knowledge, and

(2) shallow reasoning.

This is fine for many of life's problems, but ATP's needs are more severe:

(1) much expert knowledge,

(2) deeper reasoning.

- Expert knowledge is hard to encode for advanced mathematics. It is
  - easy to prove <u>all</u> geometry theorem of a certain type.
  - hard to <u>discover</u> the proof of a new theorem.
  - hard to discover a new theorem.

- Ongoing research in ATP is exciting. We will not have time to even mention much of it here.

Theorems which do not contain variables to be instantiated (bound) are called <u>ground</u> theorems.

State of the art remark:

> All ground theorems (that arise naturally) are easy to prove by modern ATP programs. But much needs to be done to handle theorems with variables.

Assertion: Much of the difficulty in ATP will be eliminated if we have programs that can

- successfully fetch the appropriate lemmas (and not useless ones)
- properly bind these lemmas' variables.

Assertion: Many of the concepts used successfully by human provers have yet to be property exploited by ATP programs:

- use of examples
  - as counterexamples (some done)
  - as guides to proof discovery (a little has been done)
- conjecturing (Lenat's work, little else)
- analogy (very little)
- Agenda Mechanism - to control the search (two Ph.D.'s theses)
- Special-purpose subprovers
  - equality packages (lots has been done) (see slide)
  - inequality packages (lots has been done)
- Domain specific heuristics

Many other ideas that we are considering are not mentioned here. These have much in common with AI research.

EQUALITY

$(A = B \wedge H \longrightarrow C)$

1. CHOOSE either A or B to replace the other.

   (Replacing both ways is disastrous)

2. REDUCTIONS

   A ---> B   A is always replaced by B

   Ex. $A \cap \emptyset \dashrightarrow \emptyset$
   $P \wedge T \dashrightarrow P$
   (Car (Cons x y)) ---> x

   Rewrite Rules

3. Complete Sets of Reductions

   Convert <u>all</u> equality units to Reductions

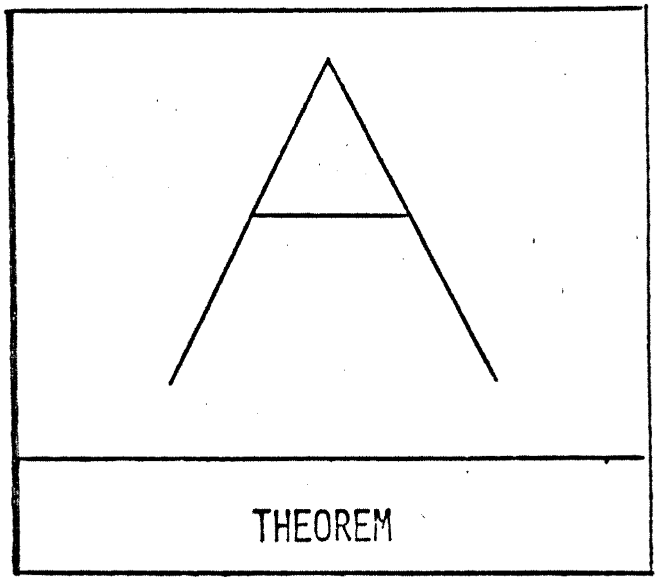
   If possible the saving is enormous.

4. EQUALITY PACKAGE

Man-machine interactive prover and proof checkers are expected to play an important role in future ATP. Examples of these are

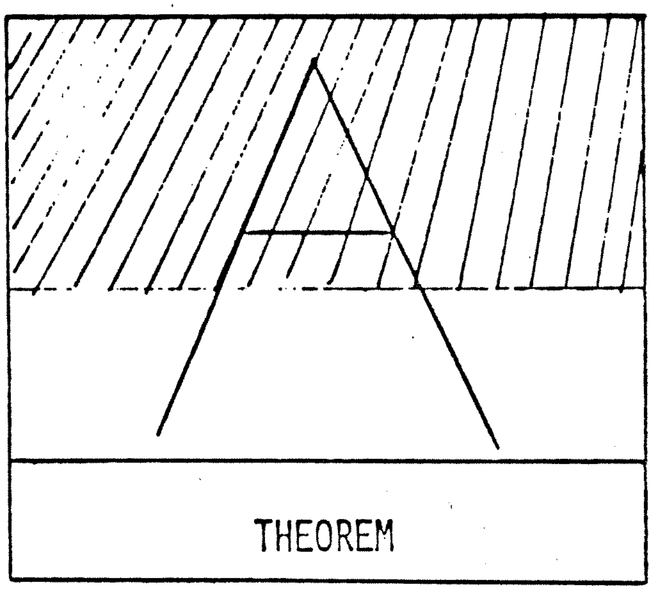
- Program Verification (mentioned earlier)
- Wos and Winker's Prover at Argonne National Laboratory
- The Boyer and Moore prover for recursive function theory
- The proof of the four-color problem in topology
- Our current attempts to prove the Poincare conjecture.

A man-machine prover must allow the user easy access. The user must not be asked to prove easy things, the machine must be able to detect when it needs help from the user and to communicate with him on what is needed without excessive work on the user's part. Such an interface is used in the Don Good PV program at UT-Austin but needs much improvement.



AXIOMS AND SUPPORTING THEOREMS NEEDED IN THE PROOF.

THEOREM BEING PROVED



BUILT IN PROCEDURES AND REDUCTION TABLES

GIVEN ONLY WHEN NEEDED

If the axioms and supporting theorem ("lemmas") shown in the slide are to be supplied ahead of time by the user, then the user would have to prove the theorem before he asks the computer to do so. Ridiculous! Whereas a really good man-machine prover will have many such lemmas "built-in" and will elicit others from the user as needed in the proof.

Several proof checkers have been built but most suffer from the fact that the user cannot submit his proofs in natural form. Work is underway to partially remedy that.

WOODLANDS AI LABORATORY
WAIL

37

3 August 1982

Mike Ballantyne and Woody Bledsoe are conducting a study on the feasibility of establishing an AI Laboratory at the Woodlands.

The Woodlands is a new city located about 40 miles northwest of Houston on Interstate 45, which has been under planning for twelve Years, is now partially built, and promises to be one of the loveliest communities in the world. It is being built on a plan that provides for the environmental, social, and employment needs of its citizens: extensive wooded parks which permeate all of the housing areas, golf courses (the Houston Open is played yearly on one of the woodlands' golf courses), tennis courts, swimming polls, ice skating, etc, housing areas (in all price ranges), schools, churches, community centers, businesses, high technology industry (including energy and medical), research and development laboratories, etc. It is designed to provide all levels of housing needs and jobs for every adult who lives there. We feel that this will be one of the choicest places to live and work.

The woodlands AI Laboratory (WAIL) will be part of HARC (Houston Area Research Center - See attached brochure) which is associated with the University of Houston, Rice University, and Texas A&M, and which is attempting to bring research and development laboratories to The Woodlands. The Woodlands Corporation, which is principally owned by The Mitchell Energy Corporation, has donated a 150 acre site to HARC and provided several million dollars in start-up funding for the next few years. It is envisioned that other Energy and Medical related industries in the Houston are would sustain the funding for the long run. (A 150 acre site has also been donated to the Texas Medical Center).

The Woodlands AI Laboratory would initially concentrate on applied AI, such as expert systems, industrial robotry, etc, which will be useful to businesses and industries in the Woodlands and Houston areas, especially those related to energy, medical, and computing research, development and applications, and later expand to others such as Natural Language interfaces, program verification, Vision, problem solving and search, knowledge represention and acquisition, theorem proving, program synthesis and understanding, etc.

Initial housing and funding for WAIL will come from those provided to HARC. We feel that the existing and projected funding is very secure, and that WAIL will be able to survive the incubation stage and become a strong, well known, laboratory.

As part of our feasibility study we will talk with a number of individuals throughout the country and abroad. These include prominent AI researchers from Universities, research laboratories, and industrial AI groups, and others in research and development laboratories throughout industry:

Stanford, MIT, CMU, U. Md, U. Texas, Rutgers, Rochester, U.Penn., U. Illinois, Yale, etc.

SRI, ISI, SUMEX, BBN, etc

Schlumberger, TI, Fairc d, Hewlett-Packard, Machine Intelligence Corporation, etc

Texas Medical Center, MD Anderson Hospital, etc

We are seeking advise on the following points:

- Possible Projects for WAIL
  - Applications oriented
  - Long range research projects
- Existing AI Projects (in other Laboratories)
- Prospects for heading and staffing WAIL
- General Advice