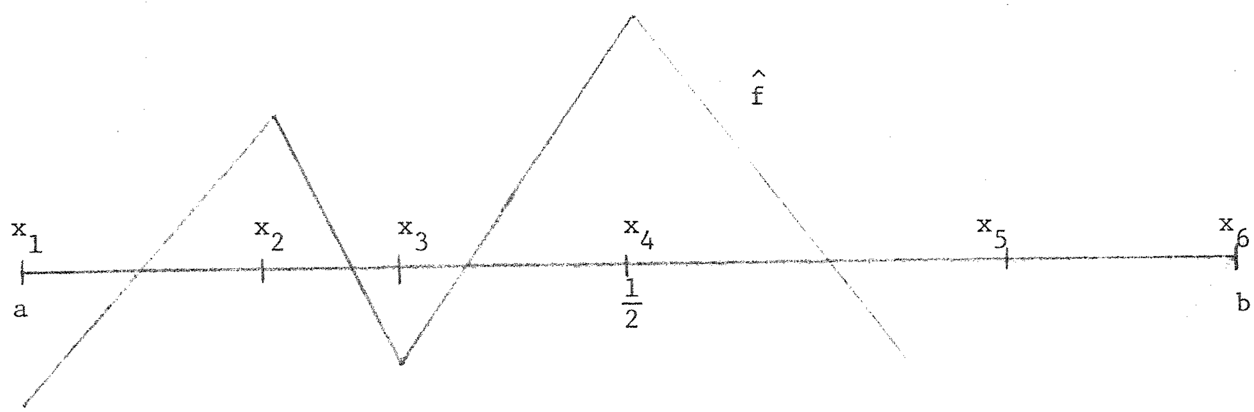
it is only necessary to check $\hat{f}(x) < f(.499)$ at each "corner" of $\hat{f}$, that is only for each x for which $(x,y) \varepsilon \hat{f}$. Thus in the algorithm GENERATE, Steps 3 and 4, where random points, $x_1,\ldots,x_n$, are selected, we might instead have used for the x_i's, the x's corresponding to the corners of $\hat{f}$.



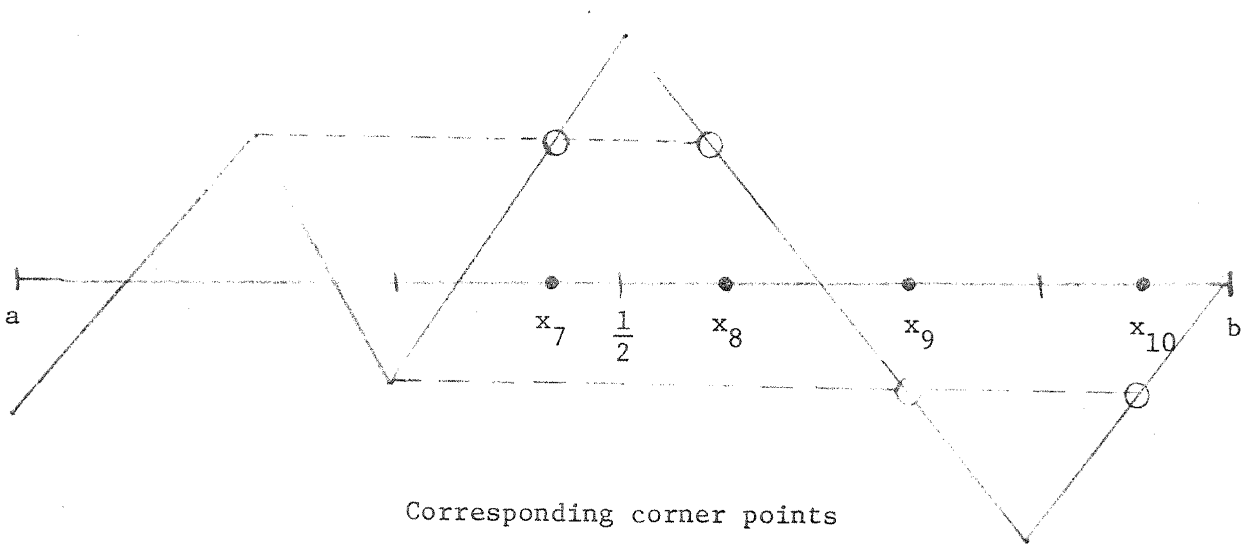
Corner points of $\hat{f}$

I.e., we need only check $\hat{f}(x) \leq f(1/2)$ for the points $x_1,x_2,\ldots,x_6$.

However, if we are using two variables x and y in the description of Q (see algorithm ALL-SOME, Section 3), and are trying to tally the formula

$$f(y) < f(x)$$

for x in interval I_1 and y in Interval I_2, it is also necessary to consider additional x's and y's corresponding* to the corner points of $\hat{f}$.



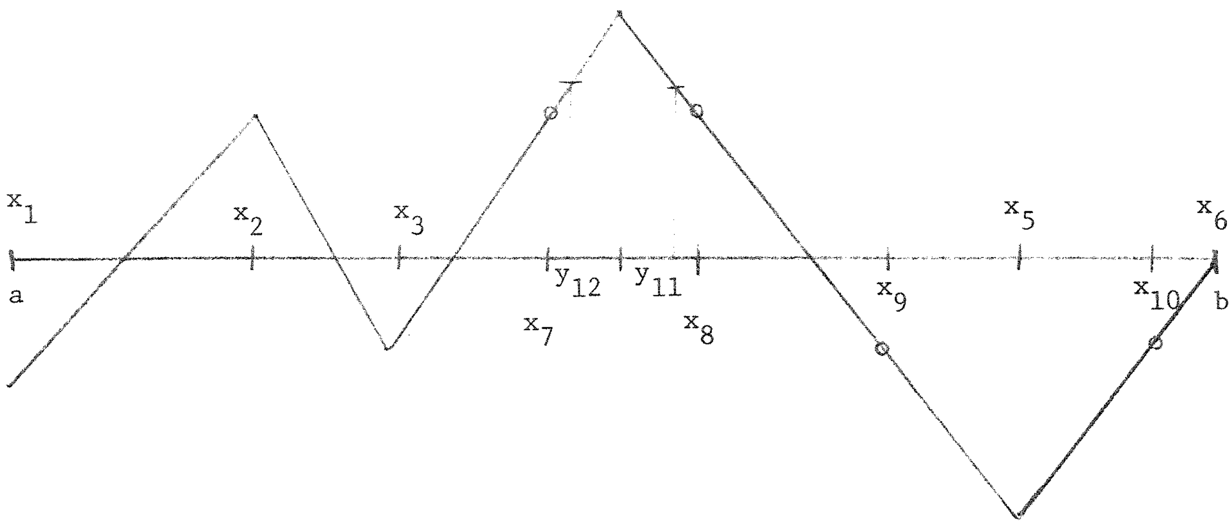
Corresponding corner points

Thus x is allowed to take the values $x_1, x_2, \ldots, x_{10}$, which lie within I_1 and once x is given such a value, $\bar{x}$, y is also allowed to take these values $x_1, \ldots, x_n$ (within I_2) as well as other values "close to" $\bar{x}$ (see Section 3.6) and points corresponding to them. Of course, this lacks generality but can be shown to be adequate in certain theorems about inequalities. For instance, it was successfully used in the examples of Section 4.

---

*We say a point x' corresponds to a point x'' (with respect to $\hat{f}$) if $\hat{f}(x') = \hat{f}(x'')$.

Using the example of the previous section (3.5), if I_1 is (1/2,b], and I_2 is [a,x), then x would be allowed to take the values $x_4 + \xi$, x_8, x_9, x_5, x_{10}, x_6. And once x is given the value x_8, then y is allowed to take the values x_1, x_2, x_3, x_7, y_{12}, x_4, y_{11}.



AM1. LUB $\wedge$ f is continuous on $[a,b] \wedge a \leq b \rightarrow \exists \ell \varepsilon [a,b] \forall x \varepsilon [a,b](f(x) \leq f(\ell))$,

where,

LUB. $\forall A \subseteq \mathcal{R} (A \neq 0 \wedge \exists r \forall x \varepsilon A(x \leq r)$

$\longrightarrow \exists \ell(\forall x \varepsilon A(x \leq \ell) \wedge \forall y[\forall z \varepsilon A(z \leq y) \longrightarrow \ell < y]))$.

Our objective in each of IMV and AM1, is to instatiate the set variable A of the hypothesis LUB, and thereby reduce both IMV and AM1 to theorems in first order logic.

Notice that they both have the form (1), p. 4 , where for IMV, H(f) is

$$\text{continuous } f[a,b] \wedge f(a) \leq 0 \wedge 0 \leq f(b)$$

and for AM1, H(f) is

$$\text{continuous } f[a,b].$$

and following the steps described on pages 4 - 6 we obtain formula (4), p. 6 , with

$$P(f,\ell) \equiv (f(\ell) \leq 0 \wedge 0 \leq f(\ell))$$

for IMV, and

$$P(f,\ell) \equiv x \varepsilon [a,b](f(x) \leq f(\ell))$$

for AM1.

Example 1 (IMV)

$$\forall f[f \text{ is continuous on } [a,b] \wedge a \leq b$$

$$\wedge\, f(a) \leq 0 \wedge 0 \leq f(b)$$

$$\longrightarrow \exists Q(\ell = \sup\{x \in [a,b]: Q(x)\}$$

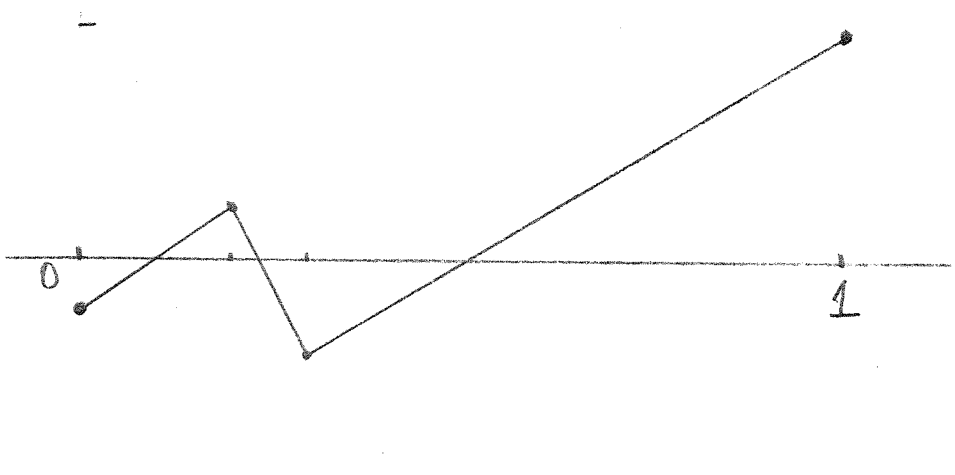
(7) $$\wedge\; \forall \ell' \varepsilon [a,\ell) \quad \exists x \in (\ell',\ell)\; Q(x)$$

$$\wedge \sim \exists x\; (\ell,b]\; Q(x)$$

$$\wedge \quad f(\ell) = 0)]$$

a call to INTERPRETATIONS yields $\hat{a} = 0$, $\hat{b} = 1$,

$$\hat{f} = \{(0\ -\ .2)(.2 .2)(.3\ -.4)(1\ 1)\} .$$



A call to CALCULATE-L yields $\hat{\ell} = 1/2$. (In the spirit of the "calculate vs. prove" remarks in Section 1, we of course call on the program to calculate a value of ℓ satisfying the theorem (for a particular $\hat{f}$)).

For these values of a, b, f and ℓ, the formula (7) reduces to

$$(7') \qquad \underbrace{\exists Q[\overbrace{\forall \ell' \varepsilon[0,1/2)\ \exists x\ \varepsilon\,(\ell',1/2)\ Q(x)}^{\psi_1} \wedge \underbrace{\sim\exists x\ \varepsilon\,(1/2,1]\ Qx}_{\psi_2}}_{\psi}$$

A call is made to GENERATE-1(ψ).

(We suppress the arguments n, $\mathcal{Z}$ and 'x) which (by Rule 1 of GENERATE) recalls itself on ψ_1 and ψ_2.

$$\text{GENERATE}(\underbrace{\forall \ell'\ \varepsilon[0,1/2)\ \exists x\ \varepsilon\,(\ell',1/2)\ Q(x)}_{\psi_1})$$

By Rule 3, the points $\ell'_1,\ldots,\ell'_4$ are selected as $\ell'_1 = 0$, $\ell'_2 = .2$, $\ell'_3 = .3$, $\ell'_4 = .499$,* and calls are made to

(8) GENERATE($\exists\, x\ \varepsilon\,(0,1/2)\ Q(x)$)

(9) GENERATE($\exists\, x\ \varepsilon\,(.2,1/2)\ Q(x)$)

(10) GENERATE($\exists\, x\ \varepsilon\,(.3,1/2)\ Q(x)$)

(11) GENERATE($\exists\, x\ \varepsilon\,(.499,1/2)\ Q(x)$)

---

*We are using the "corners" method described in Section 3.5.

These are intersected (by Rule 3 of GENERATE) to obtain

$$\{f(x) < 0\}$$

Recapitulating from (7'): a call was made to GENERATE(ψ_1) which yielded

(12) $$\{f(x) < 0\}$$

Next a call is made to

(7") $$\text{GENERATE}(\psi_2)$$
$$= \text{GENERATE}(\sim\exists x \in (1/2,1]\ Q(x))$$

By Rule 5, it calls first

$$\text{GENERATE}(\exists\ x \in (1/2,1]\ Q(x)),$$

which yields

$$\{0 < f(x)\}$$

and this is negated to obtain

(13) $$\{f(x) \leq 0\}.$$

Then by Rule 1, (12) and (13) are "intersected" to obtain

(12) $$\{f(x) < 0\}$$

as the final answer. (In intersecting (12) and (13) $\{f(x) \leq 0\}$ is treated as $\{f(x) < 0,\ f(x) = 0\}$).

For these values of a, b, f and ℓ, the formula (7) reduces to

(7')
$$\exists Q[\overbrace{\forall \ell' \varepsilon [0,1/2)\ \exists x \,\varepsilon\, (\ell',1/2)\ Q(x)}^{\psi_1}$$
$$\underbrace{\wedge \underbrace{\sim\exists x \,\varepsilon\, (1/2,1]\ Qx}_{\psi_2}}_{\psi}$$

A call is made to GENERATE-1(ψ).

(We suppress the arguments n, $\not\approx$ and 'x) which (by Rule 1 of GENERATE) recalls itself on ψ_1 and ψ_2.

$$\text{GENERATE}(\underbrace{\forall \ell' \varepsilon [0,1/2)\ \exists x \,\varepsilon\, (\ell',1/2)\ Q(x)}_{\psi_1})$$

By Rule 3, the points $\ell'_1,\ldots,\ell'_4$ are selected as $\ell'_1 = 0$, $\ell'_2 = .2$, $\ell'_3 = .3$, $\ell'_4 = .499$,* and calls are made to

(8) GENERATE($\exists$ x ε (0,1/2) Q(x))

(9) GENERATE($\exists$ x ε (.2,1/2) Q(x))

(10) GENERATE($\exists$ x ε (.3,1/2) Q(x))

(11) GENERATE($\exists$ x ε (.499,1/2) Q(x))

---

*We are using the "corners" method described in Section 3.5.

and each of these, by Rule 4, select points $x_1,\ldots,x_n$ from $(\ell_i',1/2)$ and calls

GENERATE(Q(x_i))

which in turn calls TALLY.

For example, when ℓ' takes the various values shown below, the corresponding values of $x_1,\ldots,x_n$ are chosen and calls to GENERATE(Q(x_i) and TALLY yield the results shown.

ℓ'	x_i	Result of Call to TALLY (actually a singleton set in each case)
0	.001	$f(x) < 0$
	.2	$0 < f(x)$
	.3	$f(x) < 0$
	.4999	$f(x) < 0$
.2	.2001	$0 < f(x)$
	.3	$f(x) < 0$
	.4999	$f(x) < 0$
.3	.3001	$f(x) < 0$
	.4999	$f(x) < 0$
.499	.4991	$f(x) < 0$
	.4999	$f(x) < 0$

Then from Rule 4 of GENERATE, we obtain the union of these various subgroups for (8), (9), (10), and (11). I.e., $\{f(x) < 0,\ 0 < f(x)\}$ is returned from (8); $\{f(x) < 0,\ 0 < f(x)\}$ is returned from (9); $\{f(x) < 0\}$ is returned from (10); $\{f(x) < 0\}$ is returned from (11).

These are intersected (by Rule 3 of GENERATE) to obtain

$$\{f(x) < 0\}$$

Recapitulating from (7'): a call was made to GENERATE(ψ_1) which yielded

(12) $$\{f(x) < 0\}$$

Next a call is made to

(7") $$\text{GENERATE}(\psi_2)$$

$$= \text{GENERATE}(\sim\exists x \in (1/2,1]\ Q(x))$$

By Rule 5, it calls first

$$\text{GENERATE}(\exists\ x \in (1/2,1]\ Q(x)),$$

which yields

$$\{0 < f(x)\}$$

and this is negated to obtain

(13) $$\{f(x) \leq 0\}.$$

Then by Rule 1, (12) and (13) are "intersected" to obtain

(12) $$\{f(x) < 0\}$$

as the final answer. (In intersecting (12) and (13) $\{f(x) \leq 0\}$ is treated as $\{f(x) < 0,\ f(x) = 0\}$).

This answer (13) is checked against other values of $\hat{f}$, and, finally the set A in Theorem IMV is given the value

$$\{x: a \leq x \leq b \wedge f(x) < 0\} \tag{14}$$

which is a correct instantiation.

Incidently, when A is instantiated with this value (14), the theorem IMV is reduced to the first order theorem:

$$\begin{aligned} &\forall x(a \leq x \leq b \wedge f(x) < 0 \longrightarrow x \leq \ell) \\ &\wedge \forall y(\forall z(a \leq z \leq b \wedge f(z) < 0 \longrightarrow z \leq y) \longrightarrow \ell \leq y) \\ &\wedge \ f \text{ is continuous on } [a,b] \wedge f(a) \leq 0 \wedge 0 \leq f(b) \\ &\qquad \longrightarrow \ f(\ell) \leq 0 \wedge 0 \leq f(\ell). \end{aligned}$$

Of course we would need to add the definition of continuity and, unless one uses a general inequality prover like [4], also add the axioms for the ordered reals.

Example 2. (AM1).

$\forall f[f \text{ is continuous on } [a,b] \wedge a \leq b$

$\longrightarrow \exists Q(\ell = \sup\{x \in [a,b]: Q(x)\}$

$\wedge \forall \ell' \in [a,\ell) \ \exists x \in (\ell',\ell) \ Q(x)$

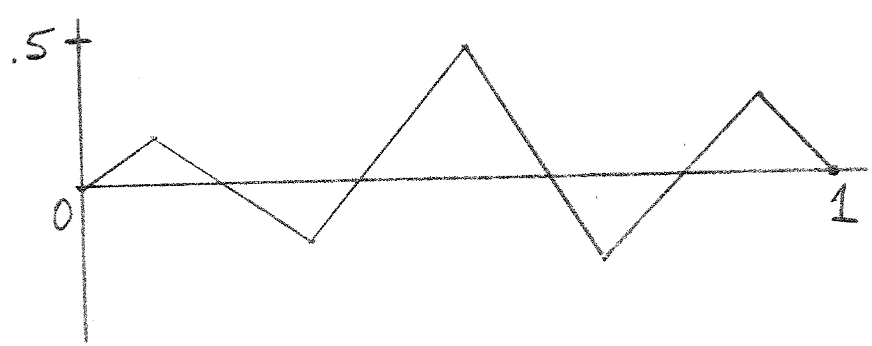
$\wedge \sim \exists x \in (\ell,b] \ Q(x)$

$\wedge \forall y \in [a,b] \ (f(y) \leq f(\ell)))]$

A call to INTERPRETATIONS yields $\hat{a} = 0$, $\hat{b} = 1$,

$\hat{f} = \{(0\ 0)(.1\ .2)(.3\ -.2)(.5\ .5)$

$(.7\ -.3)(.9\ .3)(1\ 0)\}$



a call to CALCULATE-L yields $\hat{\ell} = 1/2$

As in Example 1 we obtain (exactly) the formula (7'), and calls are then made successively to

(15) GENERATE(ψ)

(16) GENERATE(ψ_1)

(17) GENERATE(ψ_2)

The call (16) returns the list

(18) $\{\forall y \varepsilon [a,x)(f(y) < f(x)),\ \exists y \varepsilon (x,b](f(x) < f(y)), \exists y \varepsilon (x,b](f(y) < f(x))\}$

and the call to (17) returns the list

(19) $\{\forall y \varepsilon [a,x)(f(y) \leq f(x))\}$

and (18) and (19) are intersected to obtain

(20) $\{\forall y \varepsilon [a,x)(f(y) < f(x))\}$

which is returned by the call (15).

We will now examine some of the details.

The answer (19) may seem strange, but recall that it is a list of formulas (about x) which <u>do not</u> hold for x in $[\ell,b]$, which incidently (see Rule 5) is the list of the <u>negations</u> of formulas which <u>do</u> hold for x in $(\ell,b]$.

Let us now examine the call (16).

(16) GENERATE($\forall \ell\ \varepsilon[a,\ell)\ \exists x\ \varepsilon(\ell',\ell)\ Q(x)$)

Rules 3 and 4 are used to select $(\ell_1',\ldots,\ell_4')$ equal to (0, .1, .3, .499), and for each ℓ', $x_i,\ldots,x_n$ as shown below.

In the following table

$$ALL_1 \text{ is } \forall y \,\varepsilon\, [a,x) \;\; f(y) < f(x)$$

$$ALL_2 \text{ is } \forall y \,\varepsilon\, [a,x) \;\; f(x) < f(y)$$

$$ALL_3 \text{ is } \forall y \,\varepsilon\, (x,b] \;\; f(y) < f(x)$$

$$ALL_4 \text{ is } \forall y \,\varepsilon\, (x,b] \;\; f(x) < f(y)$$

$$SOME_1 \text{ is } \exists y \,\varepsilon\, [a,x) \;\; f(y) < f(x)$$

$$SOME_2 \text{ is } \exists y \,\varepsilon\, [a,x) \;\; f(x) < f(y)$$

$$SOME_3 \text{ is } \exists y \,\varepsilon\, (x,b] \;\; f(y) < f(x)$$

$$SOME_4 \text{ is } \exists y \,\varepsilon\, (x,b] \;\; f(x) < f(y)$$

If one of these is primed, then $\leq$ is used instead of $<$. For example,

$$ALL_2 \text{ is } \;\; y \,\varepsilon\, [a,x) \;\; f(x) \leq f(y)$$

ℓ'	x_1	GENERATE($Q(x_i)$)	GENERATE($x \,\varepsilon\, (\ell',\ell)$ $Q(x)$)
0	.001	$\{ALL_1 SOME_3 SOME_4\}$	
	.1	{ " " " }	$\{ALL_1 ALL_2 SOME_3 SOME_4\}$
	.3	$\{ALL_2$ " " }	
	.4999	$\{ALL_1$ " " }	
.1	.1001	$\{SOME_1 SOME_2 SOME_3 SOME_4\}$	
	.3	$\{ALL_2 SOME_3 SOME_3\}$	$\{ALL_1 ALL_2 SOME_1 SOME_2$ $SOME_3 SOME_4\}$
	.4999	$\{ALL_1$ " " }	
.3	.3001	$\{SOME_1 SOME_2 SOME_3 SOME_4\}$	$\{ALL_1 SOME_1 SOME_2$ $SOME_3 SOME_4\}$
	.4999	$\{ALL_1 SOME_3 SOME_4\}$	
.4991	.4991	$\{ALL_1 SOME_3 SOME_4\}$	$\{ALL_1 SOME_3 SOME_4\}$
	.4999	{ " " " }	

These are now intersected to yield

(18) $\{ALL_1 SOME_3 SOME_4\}$

from (16) GENERATE(ψ_1)

The call (17) GENERATE(ψ_2),

(17) GENERATE($\sim\exists x \varepsilon (1/2,1)\ Q(x)$),

causes (by Rule 5) a call first to

(17') GENERATE($\exists x \varepsilon (1/2,1)\ Q(x)$).

The negation of the result from (17') is returned for (17).

Rule 3 is used to select (.5001 .7 .9 1.) for (x_1 x_2 x_3 x_4), and GENERATE)Q(x_i)) is called for each to obtain the results shown below.

x_i	Result from GENERATE(Q(x_i))
.5001	$\{SOME_1\ SOME_2\ ALL_3\}$
.7	$\{ALL_2\ ALL_4\}$
.9	$\{SOME_1\ SOME_2\ ALL_3\}$
1.0	$\{SOME_2\ SOME_1\}$

The union of these

$$\{SOME_1\ SOME_2\ ALL_2\ ALL_3\ ALL_4\}$$

is then returned from the call (17'), and this is negated to yield

(19) $$\{ALL'_2\ ALL'_1\ SOME'_1\ SOME'_4\ SOME'_3\}$$

from call (17).

Finally, (18) and (19) are intersected to yield

$$\{ALL_1\} = \{\ y \varepsilon [a,x)\ f(y) < f(x)\}$$

from the call (15).

This is checked against other values of $\hat{f}$, and finally, the set A in Theorem AM1 is given the value

$$\{x\colon a \leq x \leq b \wedge \forall\ y \varepsilon [a,x)\ f(y) < f(x)\}.$$

Probing a bit deeper, let us see how the values shown were obtained from the calls, GENERATE($Q(x_i)$).

First a call is made to TALLY which fails and then (by Rule 6 of GENERATE) a call is made to ALL-SOME, which selects points $y_1,\ldots,y_k$ in the intervals $[\hat{a},x_i)$ and $y'_1,\ldots,y'_k$ in $(x_i,\hat{b}]$, and calls TALLY for each such y_j and y'_j.

For example, when $x_i = .3$, the points (0 .1 .2999) are selected for ($y_1\ y_2\ y_3$), and (.3001 .5 .7 .9 1) are selected for ($y'_1\ y'_2,\ldots,y'_5$).

x_i	y_j	Result from TALLY	Result from ALL-SOME
.3	0	$f(x) < f(y)$	$\{\forall y \varepsilon [a,x)\ f(x) < f(y),$ $\exists y \varepsilon (x,b]\ f(y) < f(x),$ $\exists y \varepsilon (x,b]\ f(x) < f(y)\}$
	.1	"	
	.2999	"	
	y'_j		
	.3001	$f(x) < f(y)$	
	.5	$f(x) < f(y)$	
	.7	$f(y) < f(x)$	
	.9	$f(x) < f(y)$	
	1.0	"	

and similarly for the other x_i.

For the particular ψ used in Examples 1 and 2, it was not necessary to choose the x_i's throughout the interval (ℓ', ℓ), but in fact we needed only select <u>one</u> such x_i "close" to ℓ. This is true because we are dealing with the least upper bound axiom, which gives a set A for which

$$\ell = \sup A ,$$

and hence the description of A need depend only on points x "near" ℓ. (Of course, this can also be determined by the quantifier structure of ψ).

B. Other "cute" examples.

In each of these examples the routine GENERATE is given a function $\hat{f}$ and a value $\hat{\ell}$, and it is expected to provide a predicate Q(x) for which

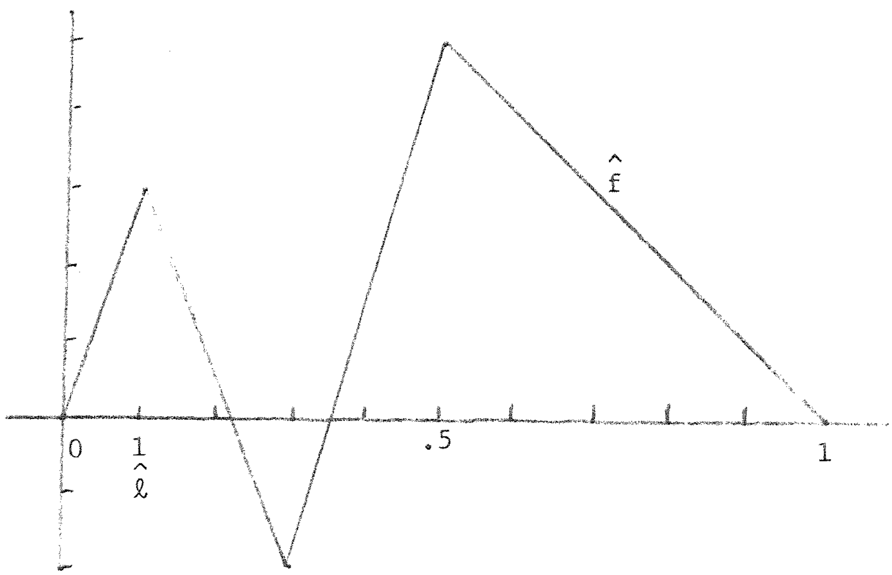
$$\hat{\ell} = \sup\{x: Q(x)\} .$$

These examples are strange (cute) in that the value $\hat{\ell}$ does not represent a zero of $\hat{f}$, or is maximum on [0,1], etc.

In each case we give only the results and show none of the details.

<u>EX 3.</u> $\hat{f} = \{(0\ 0)(.1\ .3)(.3\ -.2)(.5\ .5)(1\ 0)\}$

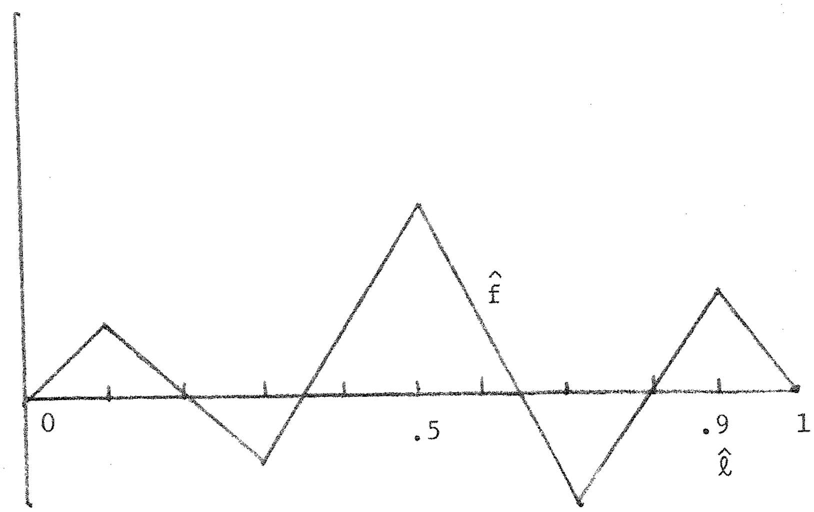
$\hat{\ell} = .1,\ \hat{a} = 0,\ \hat{b} = 1$



Result: $Q(x) \equiv \forall y \in [a,x)\ \exists z \in (y,x)\ f(y) < f(z)$

EX 4. $\hat{f} = \{(0\ 0)(.1\ .2)(.3\ -.2)(.5\ .5)(.7\ -.3)(.9\ .3)(1\ 0)\}$

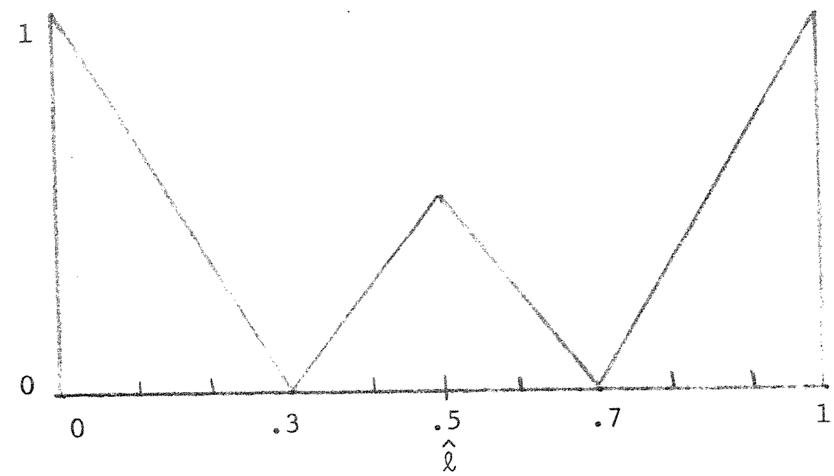
$\hat{\ell} = .9,\ \hat{a} = 0,\ \hat{b} = 1$



Result: $Q(x) \equiv \exists y \in (x,b] f(x) < f(y)$

<u>EX 5.</u> $\hat{f} = \{(0\ 1)(.3\ 0)(.5\ .5)(.7\ 0)(1\ 1)\}$

$\hat{\ell} = .5,\ \hat{a} = 0.\ \hat{b} = 1$



Result: $Qx \equiv \exists y \in (x,b]\ \exists z \in (y,b](f(x) < f(z) < f(y))$

This is a simplified part of a more complicated answer returned by the program (see Comment at the end of Section 2), namely

$$
\begin{array}{lll}
Q(x) \equiv & & \\
\exists y \in (x,b](\ \exists z \in [a,x) & & f(z) < f(x) < f(y) \\
 & \wedge \quad '' & f(x) < f(z) < f(y) \\
 & \wedge \quad '' & f(x) < f(y) < f(z) \\
 & \wedge\ \exists z \in (x,y) & f(x) < f(z) < f(y) \\
 & \wedge \quad '' & f(x) < f(y) < f(z) \\
 & \wedge\ \exists z \in (y,b] & f(z) < f(x) < f(y) \\
 & \wedge \quad '' & f(x) < f(z) < f(y) \\
 & \wedge \quad '' & f(x) < f(y) < f(z))
\end{array}
$$

Other simplified parts are

$$\exists\, y \in (x,b]\ \exists\, z \in (x,y)(f(x) < f(y) < f(z)),$$

and

$$\exists\, y \in (x,b]\ \exists\, z \in (y,b](f(z) < f(x) < f(y)),$$

but note that

$$\exists\, y \in (x,b]\ \exists\, z \in (y,b](f(x) < f(y) < f(z)$$

is not a solution.

5. Comments

5.1 Higher Order Logic.

Since instantiating set variables is a part of higher order logic one could also use procedures like those of Andrews [5], Huet [6], Darlington [7], or possibly Bledsoe [8]. In general we would expect these to be less efficient than the technique discussed here, but further experience is needed.

5.2 Conjecturing

The central component of this work is the routine GENERATE which attempts to general (describe) a predicate $Q(x)$ satisfying a particular form $\psi(Q(x))$. This is much in the spirit of Lenat's work [2], where various conjectures are derived from examples.

In Lenat's work as well as ours, there is given a set of examples and the program is asked to determine "what is true" about them. There is a difference however: whereas Lenat asks <u>all</u> that is true (about his examples), we ask what is <u>specifically</u> true about certain objects in $\mathcal{F}$, such as f.

Such conjecturing seems to play an important role in all of human endeavor, and we would expect a prominant place for it in future automatic reasoning systems.

5.3 Calculate vs. Prove

We cannot over emphasize the importance of being able to <u>calculate</u> properties about a particular $\hat{f}$ rather than <u>prove</u> the same properties about the uninstantiated variable f.

For example, for the continuous function

$$\hat{f} = \lambda x\,(4x - 4x^2)$$

it is rather easy to automatically <u>calculate</u> that $\hat{\ell} = 1/2$ is the maximum of $\hat{f}$ on the interval $[a,b] = [0,1]$. However, it is indeed difficult to <u>prove</u> automatically that <u>any</u> continuous function of $[a,b]$, with $a \leq b$, attains its maximum on that interval.

5.4 Other Example Theorems.

We hope to extend these results to other example theorems such as

Heine-Borel Theorem

Nestled Interval Theorem

Baire Category Theorem

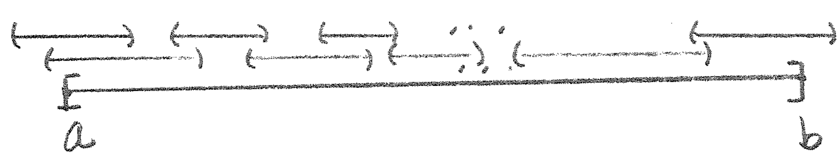
Balzano-Weierstrass Theorem

etc.

In many of these one will work with families of sets (intervals) instead of functions. There is a natural analogy between piecewise-wise linear continuous functions (plcf's) and <u>finite</u> families of intervals. Accordingly we would expect instantiations which consist of a finite family of intervals, to suffice for many applications. But infinite families will also be needed.

Let us consider briefly the Heine-Borel theorem.

<u>Theorem</u> (Heine-Borel Theorem). If F is a family of open intervals covering $[a,b]$, then there is a <u>finite</u> subfamily of F which covers $[a,b]$.



It turns out that one can use the Least Upper Bound axiom to prove this by putting

$$A = \{x: a \leq x \leq b \wedge \underbrace{\exists H \subseteq F \text{ (H is finite} \wedge \text{H covers } [a,x])}_{Q(x)}\}$$

But how does one generate (automatically) this Q(x)? For example, suppose we are given

$$[a,b] = [0,1]$$

$$\hat{F} = \{(-\tfrac{1}{4}, \tfrac{1}{4}) (\tfrac{1}{2}, \tfrac{3}{2}) (\tfrac{1}{n}, \tfrac{1}{2} - \tfrac{1}{n})\}$$

$$n = 5,6,7,\ldots$$

$$\hat{x} = \frac{1}{2} - \frac{1}{1000}.$$

Question: What is true about $\hat{x}$ and $\hat{F}$ (in terms of "finite subfamilies of F").

Answer: $\exists \subseteq F$ (H is finite $\wedge \forall y(0 \leq y < \hat{x} \rightarrow \exists B \in H\ (y \in B)))$

But it is not clear at this time how such a description can be efficiently generated.

References

1. A. Michael Ballantyne and W. W. Bledsoe. On Generating and Using Examples in Proof Discovery. Machine Intelligence 10, Ellis Harwood Limited, Chichester, 1982, pp. 3-39.

2. R. Davis and D. Lenat. Knowledge-Based Systems in Artificial Intelligence. McGraw-Hill, 1982.

3. E. Rissland and E. Soloway. Generating Examples in LIPS: Data and Programs. COINS Technical Report 80-07, Amherst: Department of Computer and Information Science: University of Massachusetts at Amherst, 1980.

4. W. W. Bledsoe and Larry Hines. Variable Elimination and Chaining in a Resolution-Based Prover for Inequalities. Fifth Conference on Automated Deduction, Les Arcs, France, July 12, 1980, Springer Lecture Notes in Computer Science.

5. P. B. Andrews. Resolution in Type Theory. Jour. of Symbolic Logic, 36, 1971, pp. 414-432.

6. G. P. Huet. Constrained Resolution: A Complete Method for Higher Order Logic. Ph.D. Thesis, Jennings Computer Center Report 1117, Cleveland, Case Western Reserve University, 1972.

7. J. L. Darlington. Deduction Plan Formation in Higher Order Logic, Machine Intelligence 7, 1972, pp. 129-137. (eds. Meltzer, B. and Michie, D.), Edinburgh University Press.

   J. L. Darlington. Talk at Oberwolfach Conference on Automatic Theorem Proving, 1976, Oberwolfach, Germany.

8. W. W. Bledsoe. A Maximal Method for Set Variables in Automatic Theorem Proving, Machine Intelligence 9, Ellis Harwood Ltd., Chichester, 1979, pp. 53-100.