INDUCTIVE REASONING ON RECURSIVE EQUATIONS

by
Frank Malloy Brown
Department of Computer Sciences
The University of Texas at Austin
Austin, Texas USA

Sten - Ake Tarnlund
Department of Information
Processing & Computer Science
University of Stockholm,
Royal Institute of Technology
Stockholm, Sweden

TR-90 MARCH 1979

Inductive Reasoning on Recursive Equations

abstract

We investigate several methods of inductive reasoning in the domain of recursive equations, including the method of generalization with beliefs, the method of successive refinement, and temporal methods based on comparisons with previously solved problems.

CONTENTS

1. Introduction

We investigate several methods of inductive reasoning which may be of use in obtaining closed form solutions to recursive equations. This research is clearly related to the pioneering work of Polya (1,2,3) on inductive reasoning in mathematics. It differs from Polya's work in that it focuses in on a particular mathematical domain, and tries to analyze a few specific methods of inductive reasoning in that domain to a level detailed enough so as to be programmable. Also it differs in that we try to relate these inductive methods to the capabilities of current deductive systems.

We consider the problem of trying to find by inductive reasoning a closed-form solution, that is an algebraic solution, to a recursive function. That is, from a set K of equations of the form :

$\Psi_k(n,fn,f(n-h),f(n-2h),\ldots)=0$ or rather : $fn = \Psi_k(n,f(n-h),f(n-2h),\ldots)$

we wish to find an equation of the form :

$\forall n\ fn = \phi n$ where f does not occur in ϕ.

For example, given the recursive equation for the Fibonacci function :

$F(n+2) = F(n+1)+Fn$

$F1 = 1$

$F0 = 0$

We would like to find a theorem of the form :

$\forall n\ Fn = \phi n$

where ϕn is a sentence constructed from algebraic symbols such as numerals, plus (+), times (*), power ($\uparrow$), minus (-), division (/), logarithm (ln), sine (sin) and cosine (cos).

Or as another example, given the recursive equations for the minimum number of moves that must be made in the Tower of Hanoi puzzle of n discs (a description of this puzzle may be found in 4).

$H(n+1) = 2(Hn)+1$

$H0 = 0$

we would like to find a closed form solution :

$\forall n\ Hn = \phi n$.

1.2 Relevance of the Problem Domain

Even though we have restricted our research in the domain of recursive functions, it should be noted that this domain is important for mathematics and Computer Science. In particular it includes one of the most important areas of classical mathematics, namely the infinitesimal calculus, as can be seen from the fact that when h is an infinitesimal number:

$$\psi_k(n, f\ n,\ f(n-h),\ f(n-2h)\ldots) = 0$$

is the general form of a differential equation.

Furthermore a system for reasoning in this domain would be useful in computer science both for the synthesis (5) of efficient programs such as logrithemic (6) Fibonacci program which is based on the closed form solution of its recursive definition, and for the analysis of algorithms.

2. Elementary Inductive Methods

We first describe two basic methods of inductive reasoning, and then compare them.

2.1 The Method of Generalization with Beliefs

One popular theory of inductive reasoning (7,8,9,10,11,12) suggests that it is basically generalization.* Consider for example the Tower of Hanoi problem. Using the recursive equations we can deduce closed-form solutions for numerical

---

* We consider analogy to be an essential component of the method of generalization as here described, and hence do not define a separate : method of analogy.

instances of Hn.

H0 = 0

H1 = 2H0 +1 = 2·0 +1 = 1

H2 = 2H1+1 = 2·1+1 = 3

H3 = 2H2+1 = 2·3+1 = 7

H4 = 2H3+1 = 2·7+1 = 15

H5 = 2H4+1 = 2·15+1 = 31

H6 = 2H5+1 = 2·31+1 =63

I suspect the reader has already induced a closed-form solution for Hn, but let us investigate how this might be done. From

H0 = 0 H1 = 1 H2 = 3 H3 = 7 H4 = 15 H5 =31 H6 = 63

we wish to find

∀n Hn = ?

First let us note that a least general generalization method such as(7,8,9) does not work because the least general generalization :

∀n ∀m Hn = m

is false.

Furthermore, even if we include an algebraic matching facility similar to that used in (11,12)* which allows us to try to relate the arguments of the Hn function to its values by adding, multiplying, subtracting, and dividing, we still would not obtain a solution. For example, adding one to the sequence of values of Hn gives :

H0+1 = 1 H1+1 = 2 H2+1 =4 H3+1 = 8 H4+1 = 16 H5+1 = 32 H6+1 = 64

∀n Hn = ?

But still the remaining least general generalization is false.

So how can the closed form solution be induced? : Let us suppose that the

---

* In our case, however, we work directly on the numbers themselves rather than applying algebraic operations to the number of occurrences of symbols such as : the successor symbol.

inductive system has available a belief that any function which produces the sequence of values 1, 2, 4, 8, 16, 32, for the arguments 0, 1, 2, 3, 4, 5, is probably equal to the exponential function of base 2 :

(Belief 1, 2, 4, 8, 16, 32 is probably 2^n)

Then given this belief a system might try to compare the sequence 1, 2, 4, 8, 16, 32 to the sequence of values given by the recursive function by applying various algebraic operations pairwise to the elements of each sequence. For example, in the case of Hn we might subtract from each element of the belief sequence, the corresponding element generated by Hn, obtaining

$$\begin{array}{r} 1\ 2\ 4\ 8\ 16\ 32 \\ -\ 0\ 1\ 3\ 7\ 15\ 31 \\ \hline 1\ 1\ 1\ 1\ 1\ 1 \end{array}$$

a sequence of ones. This would give us the knowledge that

$H0 = 2^0 - 1$

$H1 = 2^1 - 1$

$H2 = 2^2 - 1$

$H3 = 2^3 - 1$

$H4 = 2^4 - 1$

$H5 = 2^5 - 1$

and via our belief, we induce that :

$Hn = 2^n - 1$

which in fact is true.

What should be understood about this seemingly successful paradigmatic example of generalization with beliefs, is that other than for possibly the triggering of the belief that the closed form solution might involve an exponential function of base 2, we did not really need to produce those instances of the Hn function. Nor did we need to go through any generalization steps. There is a more direct method, which we call the Method of Successive

Refinement

2.2 The Method of Successive Refinement

Let us suppose that we believe a solution to Hn might involve an exponential function to a power of two.

(Belief Hn involves 2^n)

In other words, let us first form the hypothesis that

$Hn = 2^n$

and see what a hypothetical deductive system will do to this expression.

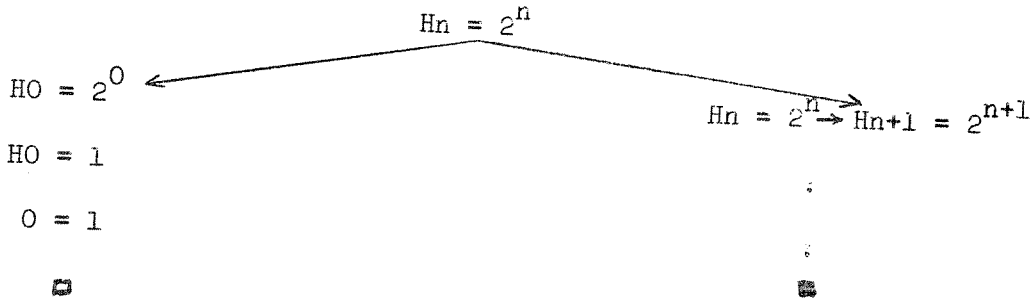
We assume that the deductive system includes a subsystem for simplifying algebraic expressions, and a subsystem for simplifying logical expressions which in particular eliminates trivial quantifiers:

$(\forall a\ a=t \rightarrow \phi a)$ is replaced by ϕt

$(\exists \alpha\ \alpha=t \wedge \phi\alpha)$ is replaced by ϕt

The rules of these two subsystems are to be applied as often as possible, the order in which rules are applied being essentially irrelevant. We further assume that the system includes an equality substitution step to eliminate the induction hypothesis, and a principle of mathematical induction which are applied only when the simplification rules are not applicable.

We now exemplify the method of successive refinement applied to $Hn = 2^n$. Inducting on n we get* :



Already we are in troube as the deductive system shows that our hypothesis is false, on the base case. Can we modify our hypothesis to induce a better one? Analysing our proof we see that if 0 = 1 were 0 = 1 - 1, then ■ and not □ would be produced. This means that

HO = 1 would have to be HO = 1 - 1,

HO = 2^0 would have to be HO = 2^0 - 1,

and Hn = 2^n would have to be Hn = 2^n - 1

Giving our new hypothesis to our deductive system we get :

---

* ■ is our symbol for true; and □ is our symbol for false.

$$Hn = 2^n - 1$$

$$H0 = 2^0 - 1$$

$$H0 = 1 - 1$$

$$H0 = 0$$

$$0 = 0$$

■

$$Hn = 2^n - 1 \rightarrow Hn+1 = 2^{n+1} - 1$$

$$2^n = Hn + 1 \rightarrow Hn+1 = 2 \cdot 2^n - 1$$

$$Hn+1 = 2(Hn+1) - 1$$

$$Hn+1 = 2Hn + 2 - 1$$

$$Hn+1 = 2Hn + 1$$

$$(2Hn)+1 = 2Hn + 1$$

■

which we deduce to be true.

We see then that we can obtain a proof by modifying a previous unsuccessful attempt at proving a theorem. This leads naturally to the question as to what is the space of all such modifications? Although, potentially this space may be quite large, we will only consider two simple modifications :

1) addition of a constant to an equation in order to make it true
2) multiplication of a constant to an additive part of an equation in order to make it true.

We have just seen an example of (1) in obtaining a solution to the Hanoi function An example of (2) applied to the false equation

$1 = \sqrt{5}$ would be to multiply $\sqrt{5}$ by $1/\sqrt{5}$

and an example of (2) applied to an additive part of the false equation

$0 = 1 + \sqrt{5}$

would be to multiply the additive part $\sqrt{5}$ by $-1/\sqrt{5}$

Note that modifications are only made to the right hand side of an equation. The reason for this is that that side contains the closed-form solution whereas the other side merely contains the recursive function.

The reader will see in the method of successive refinement an echo of Meltzer's (7) hypothesis that inductive reasoning is inverse deduction. But in our case, this is not so much a semantical hypothesis : if A→B then B is deduced from A, and A

is induced from B, as it is a notion of search strategy in the sense that if B is deduced from A in a proof then A is induced from B by analyzing in reverse the proof step consisting of the deduction of B from A.

We have seen how the closed-form solution for Hn may be obtained from the belief that the solution involves an exponential function of base two. But it turns out that this belief is stronger than what is necessary, and that the belief that some arbitrary exponential function is involved suffices :

$\exists a \forall n \; Hn = a^n$

$H0 = a^0$ $\qquad$ $Hn = a^n \rightarrow Hn+1 = a^{n+1}$

$H0 = 1$ $\qquad$:

$0 = 1$ $\qquad$:

□ $\qquad$ ■

Again by successive refinement of our hypothesis we prove :

$\exists a \forall n \; Hn = a^n - 1$

$H0 = a^0 - 1$ $\qquad$ $Hn = a^n - 1 \rightarrow Hn+1 = a^{n+1} - 1$

$H0 = 1 - 1$ $\qquad$ $a^n = Hn + 1 \rightarrow Hn+1 = a \cdot a^n - 1$

$H0 = 0$ $\qquad$ $Hn+1 = a \cdot (Hn+1) - 1$

$0 = 0$ $\qquad$ $Hn+1 = aHn+a - 1$

■ $\qquad$ $2Hn+1 = aHn+a - 1$

$2 = a$ $\qquad$ $1 = a - 1$

■ $\qquad$ $2 = a$

■

where a has been found to be equal to 2. Thus :

$Hn = 2^n - 1$

2.3 Comparison of the two Methods

So far, we have found that simple generalization methods are not sufficient and that sophisticated beliefs seem to be used in inductive reasoning. We have cast doubt on the suggestion that inductive reasoning is only generalization by describing another technique, the method of successive refinement, which seems to be more powerful (because it needs a weaker belief to produce the solution).

One advantage of the generalization method is the possibility of triggering the 2^n belief by an algebraic matching of the 1, 2, 4, 8, 16, 32, sequence to the sequence generated by the Hanoi function : Hn. A second advantage is that it is quite easy to explain how the belief, that a function whose initial instances are 1, 2, 4, 8, 16, 32, is the exponential of base 2 :

(Belief 1, 2, 4, 8, 16, 32 is probably 2^n)

was produced by simply instantiating n in 2^n to successively 0, 1, 2, 3, 4, 5 whereas it does not seem quite as easy to explain the production of the initial hypothesis that Hn involved some exponential function :

(Belief Hn involves a^n)

Are we for example to believe that the closed-form solution of every recursive function involves an exponential function?

Although it is easier to explain the origins of a belief of the form :

(Belief 1, 2, 4, 8, 16, 32 is probably 2^n)

than of the form :

(Belief Hn involves a^n)

we note that if a recursive function is not actually equal to a simple algebraic function such as 2^n then it is quite improbable that the necessary belief of this first form could be available to be used in obtaining the closed-form solution. We will now give an example of such a function and suggest that the closed-form solutions of most recursive functions are of this character.

This example is crucial because it points out the inadequacy of inductive reasoning based only on the method of generalization even with sophisticated beliefs.

Our example is the Fibonacci function. Using the recursive equations we deduce :

F0 = 0

F1 = 1

F2 = F1 + F0 = 1 + 0 = 1

F3 = F2 + F1 = 1 + 1 = 2

F4 = F3 + F2 = 2 + 1 = 3

F5 = F4 + F3 = 3 + 2 = 5

F6 = F5 + F4 = 5 + 3 = 8

F7 = F6 + F5 = 8 + 5 = 13

F8 = F7 + F6 = 13+ 8 = 21

Thus from

F0 = 0 F1 = 1 F2 = 1 F3 = 2 F4 = 3 F5 = 5 F6 = 8 F7 = 13 F8 = 21

we wish to find : Fn = ?

But there is no simple algebraic function which produces an initial sequence anything like 0, 1, 1, 2, 3, 5, 8, 13, 21. In other words there could not be any belief of the form :

(Belief $a_1 \ldots a_m$ is probably a^n)

which would be helpful in solving this problem. For after all we cannot expect to have beliefs about every algebraic function in our system; only the simple ones. We can see that the closed-form solution is not simple, by actually exhibiting it. Concurrently we will take the opportunity to show how this closed-form solution could be obtained by the method of successive refinement using the belief that the solution involves two exponential functions :

(Belief Fn involves a^n, b^n)

From this belief we form an initial hypothesis that :

$\exists a \exists b \forall n \quad Fn = a^n + b^n$

and see what our deductive system will do to this expression. Inducting on n twice, because Fn does not recurse on Fn+1, we obtain :

$\exists a \exists b \forall n \quad Fn = a^n + b^n$

$F0 = a^0 + b^0$

$F0 = 1 + 1$

$F0 = 2$

$0 = 2$

□

$Fn = a^n + b^n \rightarrow Fn+1 = a^{n+1} + b^{n+1}$

$a^n = Fn - b^n \rightarrow Fn+1 = aa^n + b^{n+1}$

$Fn+1 = a(Fn - b^n) + b^{n+1}$

$Fn+1 = aFn + (b-a)\, b^n$

$Fn+1 = aFn + (b-a)b^n \rightarrow Fn+2 = aFn+1+(b-a)b^{n+1}$ **

$(b-a)b^n = Fn + 1 - aFn \rightarrow Fn+2 = aFn+1+(b-a)b^n b$

$Fn+2 = aFn+1+(Fn+1-aFn)b$

$Fn+2 = aFn+1+bFn+1-abFn$

$Fn+2 = (a+b)Fn+1-abFn$

$Fn+1+Fn = (a+b)Fn+1-a \cdot bFn$

$(a+b-1)Fn+1-(a \cdot b+1)Fn = 0$

$a+b-1 = 0 \wedge a \cdot b+1 = 0$

$F1 = aF0 + (b-a)\, b^0$

$F1 = a \cdot 0 + (b-a) \cdot 1$

$F1 = 0 + b - a$

$F1 = b - a$

$1 = b - a$ *

$\left(1 = \frac{1-\sqrt{5}}{2} - \frac{1+\sqrt{5}}{2}\right)$

$\left(1 = \frac{2\sqrt{5}}{2}\right)$

$\left(1 = \sqrt{5}\right)$

(□)

$a + b - 1 = 0$

$a = 1 - b$

$\left(a = \frac{1+\sqrt{5}}{2}\right)$

$a \cdot b + 1 = 0$

$(1-b) \cdot b + 1 = 0$

$b - b^2 + 1 = 0$

$b^2 - b - 1 = 0$

$b = \frac{1-\sqrt{5}}{2}$

□

Footnotes * and ** see on following page.

* In contemporary deductive systems (13,14,15) the equality or unification items would end up replacing all occurrences of say "a" by b-1. A system used in successive refinement should not do this because if Fn = $a^n + b^n$ is false a might not equal b - 1, and then we would not want to propagate this falsity into the branch of the proof which is the induction step which may after all be true as in this example. The reason we prefer to have □ derived on the branch of the proof which is the induction base rather than the induction step is that it is a simpler branch and will be easier to analyse why it produced □ , in order to induce a hypothesis.

** Unlike the equality item used in many contemporary deductive systems, we use one which does not replace the recursive function such as Fn whose solution one is trying to find by some term, but rather replaces some other function such as b^n of the induction variable n by some term. There are two reasons for this! First, sometimes as in the case of the Fibonacci, the function does not immediately recurse on the first induction hence there will be no occurrence of Fn to be replaced. Note that on the first induction we could and did replace a^n by some term. Second, the replacement of a recursive function, such as Fn, may make it difficult to decide when to apply the items to equate coeficients. For whereas we might expect Fn + 1 to be linearly independent of Fn and hence

1) $\alpha Fn+1 + \beta Fn = 0 \leftrightarrow \alpha = 0 \wedge \beta = 0$

is true, it seems less obvious that

2) $\alpha Fn + \beta b^n = 0 \leftrightarrow \alpha = 0 \wedge \beta = 0$

would be true. Note that the item (1) was used in the proof given here whereas if Fn had been replaced instead of b^n then item (2) would have been needed to obtain the solution.

The induction step seems to be true, but the two base cases are false. Can we modify our hypothesis to induce a better one?

Analysing the F1 base case we see that if the $\sqrt{5}$ in $(1 = \sqrt{5})$ had been divided by $\sqrt{5}$ then ■ would have been produced. This means that the base case would have had to have looked something like :

$$F1 = aF0 + \left(\frac{b-a}{\sqrt{5}}\right)$$

$$F1 = a \cdot 0 + \left(\frac{b-a}{\sqrt{5}}\right) \cdot 1$$

$$F1 = 0 + \frac{b-a}{\sqrt{5}}$$

$$F1 = \frac{b-a}{\sqrt{5}}$$

$$1 = \frac{b-a}{\sqrt{5}}$$

$$1 = \left(\frac{1+\sqrt{5}}{2} - \frac{1-\sqrt{5}}{2}\right) / \sqrt{5}$$

$$1 = \frac{2\sqrt{5}}{2\sqrt{5}}$$

$$1 = 1$$

■

Working upwards we see that the five previous lines in the proof must have been something like :

$$Fn = a^n + \frac{b^n}{\sqrt{5}}$$

$$Fn = a^n + \frac{b^n}{\sqrt{5}} \rightarrow Fn+1 = a^{n+1} + \frac{b^{n+1}}{\sqrt{5}}$$

$$a^n = Fn - \frac{b^n}{\sqrt{5}} \rightarrow Fn+1 = a \cdot a^n + \frac{b^{n+1}}{\sqrt{5}}$$

$$Fn+1 = a\left(Fn - \frac{b^n}{\sqrt{5}}\right) + \frac{b^{n+1}}{\sqrt{5}}$$

$$Fn+1 = aFn + \frac{(b-a)}{\sqrt{5}} b^n$$

? ■ ?

So far we have been able to induce a new hypothesis which will probably make the F1-base branch of the proof result in ■. But, does this change the result of ■ we had previously obtained in the induction step branch of the proof? Since inducting on $Fn+1 = aFn + \frac{b-a}{\sqrt{5}} - b^n$ gives :

$$Fn+1 = aFn + \frac{b-a}{\sqrt{5}}\ b^n \rightarrow Fn+2 = aFn+1 + \frac{b-a}{\sqrt{5}}\ b^{n+1}$$

$$\frac{b-a}{\sqrt{5}}\ b^n = Fn+1-Fn \qquad \rightarrow Fn+2 = aFn+1+ \frac{b-a}{\sqrt{5}}\ b^n\ b$$

$$Fn+2 = aFn+1+ (Fn+1 - aFn)b$$

and since the third line is identical to our previous third line we see that the induction step branch of our proof will still result in ■.

Of course we don't actually have to analyse our proofsteps downward, as we could simply apply our deductive system to such a step and see what it does. For example, to see that $Fn+1 = aFn + \frac{b-a}{\sqrt{5}}\ b^n$ results in ■ all we need do is to apply our deductive system to it.

We have now produced a modification which results in ■ on both the F1 base, and induction step branches of our proof. Only the F0 base step remains. The F0 base branch resulted in □, so we know that some sort of modification is what was needed to make the F0 base branch result in ■ : We first check to see if the modification induced from the F1 base branch will suffice to make this branch result in ■ :

$$Fn = a^n + \frac{b^n}{\sqrt{5}}$$

$$F0 = a^0 + \frac{b^0}{\sqrt{5}}$$

$$F0 = 1 + \frac{1}{\sqrt{5}}$$

$$0 = 1 + \frac{1}{\sqrt{5}}$$

□

So there is still a problem. If, however, $0 = 1 + \frac{1}{\sqrt{5}}$ were $0 = \frac{1}{\sqrt{5}} - \frac{1}{\sqrt{5}}$ then

0 = 0, ■ would have been deduced. For this to be the case

$F0 = 1 + \frac{1}{\sqrt{5}}$ would have to be $F0 = \frac{1}{\sqrt{5}} - \frac{1}{\sqrt{5}}$

$F0 = a^0 + \frac{b^0}{\sqrt{5}}$ would have to be $F0 = \frac{a^0}{\sqrt{5}} - \frac{b^0}{\sqrt{5}}$

and finally :

$Fn = a^n + \frac{b^n}{\sqrt{5}}$ would have to be $Fn = \frac{a^n - b^n}{\sqrt{5}}$

and then our deductive system would easily be able to deduce that this new hypothesis :

$\exists a \exists b \; \forall n \; Fn = \frac{a^n - b^n}{\sqrt{5}}$

is true.

Replacing a and b by the algebraic terms that we have found them to be in our proof we find that the closed form solution for the Fibonacci function is :

$$\forall n \; Fn = \frac{\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n}{\sqrt{5}}$$

Thus we see that although it is plausible that we might be able to induce this solution by the inductive method of successive refinement, it is not very plausible that we could have induced this solution by the generalization method. The reason for this, as will be recalled, is because the generalization method would require prior beliefs that :

(Belief : 1, 1.618, 2.618, 4.236, 6.854, 11.090 is probably $\left(\frac{1+\sqrt{5}}{2}\right)^n$)

(Belief : 1, -.618, .382, -.236, .146, -.090 is probably $\left(\frac{1-\sqrt{5}}{2}\right)^n$)

And it simply does not seem plausible that such beliefs would be available.

2.4 The Supplementary Method of Existential Functions

We have one final point to make about the Fibonacci example : If we had a

slightly more sophisticated belief as to what the closed-form solution might be, then our deductive system would be able to obtain the solution in a very direct manner. Let us suppose that we have the belief that the closed-form solution of the Fibonacci involves a linear combination of two non-zero exponential functions :

(Belief Fn involves $\alpha a^n + \beta b^n$)

Then the hypothesis is : $\exists\alpha\exists\beta\exists a\exists b\ \forall n\ Fn = \alpha a^n + \beta b^n$

A solution is obtained by inducting twice on n .

$\exists\alpha\exists\beta\exists a\exists b\ \forall n\ Fn = \alpha a^n + \beta b^n$

$F0 = \alpha a^0 + \beta b^0$

$F0 = \alpha \cdot 1 + \beta \cdot 1$

$F0 = \alpha + \beta$

$0 = \alpha + \beta$

$\alpha = -\beta$ ■

$(\alpha = 1/\sqrt{5})$

$Fn = \alpha a^n + \beta b^n \rightarrow Fn+1 = \alpha a^{n+1} + \beta b^{n+1}$

$\alpha a^n = Fn - \beta b^n \rightarrow Fn+1 = \alpha a^n a + \beta b^{n+1}$

$Fn+1 = (Fn - \beta b^n)a + \beta b^{n+1}$

$Fn+1 = aFn + (b-a)\beta b^n$

$F1 = aF0 + (b-a)\beta b^0$

$F1 = a \cdot 0 + (b-a)\beta \cdot 1$

$F1 = 0 + (b-a)\beta$

$F1 = (b-a)\beta$

$1 = (b-a)\beta$

$\beta = \frac{1}{b-a}$

$\left(\beta = \frac{1}{\frac{1-\sqrt{5}}{2} - \frac{1+\sqrt{5}}{2}}\right)$

$\left(\beta = \frac{2}{-2\sqrt{5}}\right)$

$\left(\beta = -\frac{1}{\sqrt{5}}\right)$ ■

$Fn+1 = aFn+(b-a)\beta b^n \rightarrow Fn+2 = aFn+1+(b-a)\beta b^{n+1}$

$(b-a)\beta b^n = Fn+1-aFn \rightarrow Fn+2 = aFn+1+(b-a)\beta b^n b$

$Fn+2 = aFn+1+(Fn+1-aFn)b$

$Fn+2 = aFn+1+bFn+1-abFn$

$Fn+2 = (a+b)Fn - abFn$

$Fn+1+Fn = a+bFn - abFn$

$(a+b-1)Fn+1-(a \cdot b+1)Fn = 0$

$a+b-1 = 0 \quad \wedge \quad a \cdot b+1 = 0$

$a + b - 1 = 0$

$a = 1 - b \cdots\cdots\rightarrow (1-b) \cdot b + 1 = 0$ ■

$\left(a = \frac{1+\sqrt{5}}{2}\right)$

$b - b^2 + 1 = 0$

$b^2 - b - 1 = 0$

$b = \frac{1-\sqrt{5}}{2}$ ■

This technique of using existential functions (in this case constants: a, b, α, β) has been used by Siklossy in [16]. Because it is so powerful that we shall raise it to the status of a third inductive method which we shall call : The supplementary method of existential functions. It should be noted that this technique is no substitute for the more general method of successive refinement, for its power depends crucially on knowledge possessed by the deductive system In the protocols using the method of existential functions we have assumed the ability to solve equations of the form :

$$\phi f = 0$$

for an arbitrary existential constant. This knowledge was not assumed when not using this method. We shall see more of this method in section 3.

2.5 Summary

In summary then we have described two basic inductive methods and one supplementary method; and have shown that the effectiveness of each method depends crucially on the availability of particular beliefs. We have suggested that the beliefs needed by the generalization method would only be available if the recursive function is equal to some very simple algebraic function, such as is the case with the Tower of Hanoi function. For most other recursive functions such as the Fibonacci functions we have suggested that the beliefs needed by the generalization method would not be available.

In section 3 we describe how the beliefs needed by the method of successive refinement might be automatically produced.

3. Temporal Inductive Methods

We now describe methods of inductive reasoning based on information obtained from the solutions of previously solved problems. Such methods are called temporal inductive methods. There are two temporal inductive methods, distinguished by the type of information on which they are based. The first temporal method is

based solely on information obtained from the theorem which expresses a previously solved problem, whereas the second temporal method is based on information obtained from the proof of a previously solved problem. Some early related research on temporal methods is given in [17].

Referring back to section 2.5 we shall see in section 3.2 how the second temporal method may be applied to producing the kind of beliefs needed by the method of successive refinement.

3.1 The Temporal Method based on Theorems

The basic idea of the temporal method based on theorems is for any given problem F to try to find a similar problem H which has already been solved, and then to use something similar to the solution for H as a hypothesis for the solution of F.

For example let H be the Tower of Hanoi problem whose solution we discovered in Section 2, and let F be the Fibonacci function whose solution we are trying to find. The theorem which expresses the solution to the Hanoi problem is then :

$$(\forall n\ H_{n+1} = 2H_n + 1 \wedge H0 = 0) \rightarrow H_n = 2^n - 1$$

and the (as yet incomplete) theorem which expresses the Fibonacci problem is then :

$$(\forall n\ F_{n+2} = F_{n+1} + F_n \wedge F1 = 1 \wedge F0 = 0) \rightarrow F_n = ?$$

We now try to compare these two theorems in order to produce a guess as to what ? should be :

$$(\forall n\ H_{n+1} = 2H_n + 1 \wedge H0 = 0) \rightarrow H_n = 2^n - 1$$

$$(\forall m\ F_{m+2} = F_{m+1} + F_m \wedge F1 = 1 \wedge F0 = 0) \rightarrow F_m = ?$$

We see that there is no reasonable analogy that we can make which will help us to solve F. For example if we form the analogy that m is n - 1 from (H_{n+1}, F_{m+2}), then we might conjecture that :

$$F_m = 2^{m-1} - 1$$

which is false. In any case, so much syntax has been left unexplained by this analogy (for example the 2°, +1, H0 = 0, +Fm, and F1 = 1) that we can have no confidence in this conjecture anyway.

Furthermore even if our generalization system were sophisticated enough to rewrite the H theorem as the equivalent theorem :

$$(\forall n\ H_{n+2} = 2\,H_{n+1} + 1 \wedge H1 = 1 \wedge H0 = 0) \rightarrow Hn = 2^n - 1$$
$$(\forall n\ F_{n+2} = F_{n+1} + Fm \wedge F1 = 1 \wedge F0 = 0) \rightarrow Fm = ?$$

The closest analogy would still not explain the 2°, +1, and +Fm.

The problem here is that H and F are so dissimilar that no immediate analogy can be made. This fact, however, does not imply that the Temporal Method based on Theorems cannot be used to solve this problem, for it may be the case that H is related to F by a sequence of theorems each of which is similar to the next :

F is similar to X, which is similar to X2 .... H

In particular F might be similar to a function G which is similar to H.

This leads to two strategies for trying to find such a function G :

1) The bottom-up method which is to try to frame a problem G more similar to F than is H, and yet is similar enough to H so that the solution to H will be helpful in guessing the solution to G.
2) The top-down method which is to try to frame a problem G more similar to H than is F, and yet is similar enough to F so that if we could solve G, the solution to F may be forthcoming.

For example, using the bottom-up method we might try to modify H into a new function G which is more similar to F. Noting from our previous attempts to form an analogy that the 2° expression is unexplained, we might let G be obtained from H by replacing the numeral 2 by an existential constant a :

$$G_{n+1} = a \cdot Gn + 1$$
$$G_0 = 0$$

What is the closed-form solution to Gn? We compare this to our previous

$$H_{n+1} = 2H_n + 1 \wedge H0 = 0 \rightarrow H_n = 2^n - 1$$

$$G_{n+1} = aG_n + 1 \wedge G0 = 0 \rightarrow G_n = ?$$

Since G is a generalization of H, the solution of G should probably be a generalization of the solution of H. Since the recursive equation for Gn differs from the recursive equation of Hn only in having "a" where H has 2 and since 2 appears in the closed-form solution of H we might suspect that "a" should occur in the closed-form solution of G. Namely the hypothesis :

$$G_n = a^n - 1$$

In any case, even without regard to the function G, the fact that a symbol 2 which occurs in the recursive definition pops up again in the closed-form solution should lead us to the question as to whether this is mere chance or if there is some casual connection. And since the way to decide this question is to replace 2 by a different numeral, say 3, or even by an existential variable "a". Thus we are led to the function G and again to the hypothesis that :

$$\exists a \forall n \; G_n = a^n - 1$$

Let us see if this conjecture is true :

$$\exists a \forall n \; G_n = a^n - 1$$

$$G0 = a^0 - 1$$

$$G0 = 1 - 1$$

$$G0 = 0$$

$$0 = 0$$

■

$$G_n = a^n - 1 \rightarrow G_{n+1} = a^{n+1} - 1$$

$$a^n = G_n + 1 \rightarrow G_{n+1} = aa^n - 1$$

$$G_{n+1} = a(G_n + 1) - 1$$

$$G_{n+1} = aG_n + a - 1$$

$$aG_n + 1 = aG_n + a - 1^*$$

$$1 = a - 1$$

$$2 = a$$

■

* In retrospect, the fact that the item for equating the coefficients of like terms was not used here (the cancellation item was used) is, we feel, a qualitative indication that our hypothesis is very close to being true, and that a slight modification by the method of successive refinement will produce the correct solution. On the other hand if the item for equating coefficients had been used on the induction step branch of some protocol, and if that branch resulted in □, we believe that in this problem domain there would be little chance that the method of successive refinement alone could induce the correct solution. This is related to the restricting of the space of modification by successive refinement to the addition and multiplication by a constant.

We discover that a must equal 2. This means that if "a" were any other number but 2 then □ would have been derived on the induction step branch of this proof. Since it is the solution to G when "a" is not'2' that we are trying to discover, we ask if this proof can be modified so as to give us a solution in these cases.

We consider an instance of this proof where a is replaced by some particular numeral and apply the method of successive refinement. Replacing 'a' by '3' we get the protocol :

$\forall n \; Gn = 3^n - 1$

$G0 = 3^0 - 1$

$G0 = 1 - 1$

$G0 = 0$

$0 = 0$

□

$Gn = 3^n - 1 \rightarrow Gn+1 = 3^{n+1} - 1$

$3^n = Gn + 1 \rightarrow Gn+1 = 3 \cdot 3^n - 1$

$Gn+1 = 3 (Gn + 1) - 1$

$3Gn+1 = 3Gn + 3 - 1$

$1 = 3 - 1$

$1 = 2$

□

We see that if

$1 = 2$ had been $1 = \frac{2}{2}$

then ■ would have been produced on the induction step branch of the proof. Progressing upwards, this implies that :

$1 = a - 1$ would be $1 = \frac{3 - 1}{2}$

and $3Gn + 1 = 3Gn + 3 - 1$ would be $3Gn + 1 = \frac{6Gn + 3 - 1}{2}$

Continuing in this fashion we find that the entire protocol would now be :

$$\forall n \; G_n = \frac{3^n - 1}{2}$$

$$G0 = \frac{3^0 - 1}{2}$$

$$G0 = \frac{1 - 1}{2}$$

$$G0 = \frac{0}{2}$$

$$G0 = 0$$

$$0 = 0$$

■

$$G_n = \frac{3^n - 1}{2} \rightarrow G_{n+1} = \frac{3^{n+1} - 1}{2}$$

$$2G_n = 3^n - 1 \rightarrow G_{n+1} = \frac{3 \cdot 3^n - 1}{2}$$

$$3^n = 2G_n + 1 \rightarrow 3G_n + 1 = \frac{3 \cdot 3^n - 1}{2}$$

$$3G_n + 1 = \frac{3(2G_n + 1) - 1}{2}$$

$$3G_n + 1 = \frac{6G_n + 3 - 1}{2}$$

$$3G_n + 1 = \frac{6G_n + 3 - 1}{2}$$

$$3G_n + 1 = 3G_n + \frac{3-1}{2}$$

$$1 = \frac{3 - 1}{2}$$

$$1 = \frac{2}{2}$$

$$1 = 1$$

■

We now know that if "a" is 3 :

$$1 = 2H_n + 1 \;\wedge\; H0 = 0 \rightarrow H_n = (2^n - 1) / 1$$

$$1 = 3G_n + 1 \;\wedge\; G0 = 0 \rightarrow G_n = (3^n - 1) / 2$$

Applying the method of generalization to these examples (and others if we wish for we can let a be any other numeral and apply the same methods) we get :

$$G_{n+1} = aG_n + 1 \;\wedge\; G0 = 0 \rightarrow G_n = (a^n - 1) / (a - 1)$$

Recalling that our original problem is to solve the Fibonacci function F : we might rewrite G :

$$F_{n+2} = F_{n+1} + F_n \;\wedge\; F1 = 1 \;\wedge\; F0 = 0 \rightarrow F_n = ?$$

$$G_{n+2} = aG_{n+1} + 1 \;\wedge\; G1 = 1 \;\wedge\; G0 = 0 \rightarrow G_n = (a^n - 1)/(a - 1)$$

We see that if we let a = 1 then we will have made G look more similar to F than H because it will no longer have the "2" difference.

This gives :

$$Gn+2 = Gn+1 + 1 \wedge G1 = 1 \wedge G0 = 0 \rightarrow Gn = n$$

Note that Gn does not equal $(1^n - 1) / (1 - 1)$ which is 0/0. So instead we replace a not by 1 itself, but by a number which is infinitesimally close to 1 and ask for the standard part of the resulting expression which is "n"

We could continue in this fashion, modifying previous solutions, so as to make the problem more and more similar to Fn. However, we believe that the modifications needed in order to solve F are of a nature that only a more sophisticated technique involving analogies on previous proofs and protocols are capable of solving this problem.

3.2 Temporal Method based on Proofs

The basic idea of the temporal method based on proofs is as follows :

1) First using the temporal method based on theorems find a problem H which is similar to the problem F you are trying to solve and conjecture a solution for F.
2) Using the deductive system produce the proof of H and a protocol of F with that hypothesis.
3) Finally compare the proof with the protocol to find out why no solution was obtained, and form a new hypothesis.

Suppose, now that we are trying to find a solution to the Fibonacci function F, given that we already know the solution to the Tower of Hanoi function H. We shall use the proof of the solution of H which is given on page 7 beginning with :

$$\exists a \; \forall n \; Hn = a^n - 1$$

The hypothesis for the solution of F is

$$\exists a \; \forall n \; Fn = a^n - 1$$

(We could throw in a few existential constants, for example : $Fn = \alpha \cdot a^n - \beta$, but it won't make any difference at this stage.)

Using our deductive system we obtain the following protocol :

$\exists a\ \forall n\ Fn = a^n - 1$

$F0 = a^0 - 1$ $\qquad$ $Fn = a^n - 1 \rightarrow Fn+1 = a^{n+1} - 1$

$F0 = 1 - 1$ $\qquad$ $a^n = Fn + 1 \rightarrow Fn+1 = a \cdot a^n - 1$

$F0 = 0$ $\qquad$ $Fn+1 = a \cdot (Fn + 1) - 1$

$0 = 0$ $\qquad$ $Fn+1 = aFn + a - 1$

■

$F1 = aF0 + a - 1$ $\qquad$ $Fn+1 = aFn + a - 1 \rightarrow Fn+2 = aFn + 1 + a - 1$

$F1 = a \cdot 0 + a - 1$ $\qquad$ $a(Fn+1) = (Fn+1) + 1 \rightarrow Fn+2 = aFn + 1 + a - 1$

$F1 = 0 + a - 1$ $\qquad$ $a = \dfrac{(Fn+1) + 1}{Fn + 1} \rightarrow Fn+2 = a(Fn + 1 + 1) - 1$

$F1 = a - 1$

$1 = a - 1$ $\qquad$ $Fn+2 = \dfrac{((Fn+1) + 1)^2}{Fn + 1} - 1$

$2 = a$

.
.
.

We now compare this protocol with the proof of Hn. We see that in both cases the induction principle was first applied. We then see that the steps in the base case of proof and protocol were the same, and that the first few steps on the induction step branch of the proof and protocol were the same. In fact we find that they differ exactly where Hn+1 is replaced by $2Hn + 1$

$Hn+1 = aHn + a - 1$ $\qquad$ $Fn+1 = aFn + a - 1$

$2 \cdot Hn + 1 = aHn + a - 1$ $\qquad$ $F1 = aF0 + a - 1$

$Fn+1 = aFn + a - 1 \rightarrow Fn+2 = a \cdot Fn+1 + a - 1$

This then is the problem. On F the theorem proven inducts because it can't recurse, because F recurses on n+2 not n+1, whereas H immediately recurses to obtain a solution.

So granted that F recurses on n+2 and that the theorem proven inducts twice the question, is : why is not a solution obtained after the second induction? Let us compare the steps beginning with the second induction in F to the steps

in Hn :

$\exists a\ \forall n\ Hn = a^n - 1$ $\qquad$ $\exists a\ \forall n.\ Fn+1 = aFn + a - 1$

We see that Hn involves some term "a" exponentiated by the induction variable n. Comparing Hn to Fn+1 this leads us to suspect that Fn+1 might also involve some term exponentiated by the induction term n+1. Thus we would hypothesize that Fn+1 involves some term b^{n+1} or rather that Fn involves some term b^n. The reader should bear in mind that the hypothesis is not a^{n+1} for this a would clash with the bound variable a already occuring in the above Fibonacci expression. The point is that the "a" in the Hanoi expression and "a" in the Fibonacci expression are two distinct bound variables, which do not necessarily refer to the same number.

There are two other indications that Fn involves a term of the form b^n. Inducting on n in each case we get respectively the base cases :

$H0 = a^0 - 1$	$F1 = aF0 + a - 1$
$H0 = 1 - 1$	$F1 = a \cdot 0 + a - 1$
$H0 = 0$	$F1 = 0 + a - 1$
$0 = 0$	$F1 = a - 1$
■	$1 = a - 1$
	$2 = a$

Note that in the proof of H a skolem function disappears by having a^0 replaced by 1 whereas in the protocol of F no skolem function disappears in this manner. Here then is a second indication that a term of the form b^0 or rather b^n is needed in the Fibonacci protocol.

Finally by comparing the induction steps of H and F we note that in H the equality item replaced a^n by an expression, whereas in F it merely replaced "a" by an expression. Here then is a third indication that Fn involves an expression of the form b^n (where b is not necessarily equal to a).

$Hn = a^n - 1 \rightarrow Hn+1 = a \cdot a^n - 1$

$a^n = Hn+1 \rightarrow Hn+1 = a \cdot a^n - 1$

$Hn+1 = a \cdot (Hn+1) - 1$

$Fn+1 = aFn+a-1 \rightarrow Fn+2 = a(Fn+1)+a-1$

$aFn+1 = Fn+1+1 \rightarrow Fn+2 = a(Fn+1)+a-1$

$a = \frac{((Fn+1)+1)}{Fn+1} \rightarrow Fn+2 = a((Fn+1)+1)-1$

$Fn+2 = \frac{((Fn+1)+1)^2}{Fn+1} - 1$

Thus, for three reasons we are lead to the belief that Fn involved some term b^n where be is not necessarily a. We now go back and modify our original belief : that Fn involves a^n :

(Belief Fn involves a^n)

to the new belief that Fn involves both a^n and b^n :

(Belief Fn involves $a^n + b^n$)

And this new belief, as we have already seen in section 2.3, suffices to find the closed-form solution to the Fibonacci function.

3.3 Summary

In summary we have described two temporal inductive methods, one based on comparing the statement of a problem with the statement of a similar previously solved problem, and the other based on comparing the unsuccessful attempts to solve a problem with the proof of an analogous problem. In either case we have found that inductive reasoning depends curcially on the ability to make use of previous solutions.

We have also seen how the type of belief needed to apply the method of succession refinement to the Fibonacci problem might be generated.

4. Conclusion

We have investigated and described several feasible methods of inductive reasoning which are useful in the domain of solving difference equations. We stress that these methods use no mathematical knowledge in terms of lemmas, strategies or rules other than that which would be available to a simple algebra theorem prover which has only the following rules *:

1) Rules to simplify algebraic expressions.

2) The logical rules needed to put formulas in conjuctive and disjunctive normal form.

3) The two equality rules :

$$(\forall x \quad x = y \rightarrow \phi\, x) \leftrightarrow \phi\, y$$

$$(\exists x \quad x = y \wedge \phi\, x) \leftrightarrow \phi\, y$$

4) A mathematical induction principle.

5) Rules to equate the coefficients of various types of expressions. (This ability is needed by the method of existential functions).

Of course, if we were to allow the use of sophisticated mathematical techniques we could easily solve the problems we have considered in this paper. In particular, we could implement a theorem prover based either on the rules of the finite difference calculus (18, 19), or on the method of generating functions (6, 19, 20, 21).

But this is just what we do not want to do, for this would not explain how the mathematical knowledge embedded in these techniques was actually created. In our view of mathematics, we see that this knowledge is basically created by trying to solve a number of problems using only one's general inductive abilities and the current status of one's deductive abilities. Then one generalizes

---

* These rules are imbedded in an algebra theorem prover which one of the authors (Brown) implemented about a year ago.

and organizes the methods that one has found useful in solving these problems into some sort of deductive calculus, such as for example the theory of finite difference equations or the method of generating functions. One then adds this calculus to one's deductive abilities, henceforth making what were difficult problems quite easy, thus allowing one to try to solve problems in more difficult domains by repeating this process.

Acknowledgements

We thank Professor B Meltzer for commenting on a draft of this paper and for arranging for our collaboration on this research at Edinburgh. We also thank the reviewers for their helpful comments.

References

1. G. Polya Induction and Analogy in Mathematics, Mathematics and Plausible Reasoning, Vol.I. 1968. Oxford University Press, London.
2. G. Polya Patterns of Plausible Inference, Mathematics and Plausible Reasoning, Vol.II. 1968. Oxford University Press, London.
3. G. Polya Mathematical Discovery, Vol.I,II. John Wiley & Sons, Inc.1967.
4. George Luger "The use of the state space to record the behavioral effects of subproblems and symmetrics in the Tower of Hanoi Problem." Int.J. Man Machine Studies, 1976. 8, 411-421.
5. Marvin Minsky, "Form and Content in Computer Science" JACM Vol. 17, no. 2, April 1970.
6. Donald E. Knuth Fundamental Algorithms, The Art of Computer Programming, Vol. I, Addison-Wesley Publishing Company, Inc. 1969.
7. B. Meltzer "Power Amplification for Theorem-Provers, Part 2 : A hypothetico deductive approach to programming induction." MI5.
8. Gordon D. Plotkin "A note on Inductive Generalization", MI5, 1969.
9. Gordon D. Plotkin "A Further Note on Inductive Generalization", MI6, 1971.
10. R.J. Popplestone "An Experiment in Automatic Induction", MI5, 1969.
11. J. Feldman, J. Gips, J. Horning & S. Reder. "Grammatical complexity and inference", Tech. Report No CS 125, Stanford AI Project, Stanford University, 1969.
12. Steven Hardy "Synthesis of LISP Functions from Examples." Proceedings of AISB Summer Conference, 1976.
13. F.M. Brown "Doing Arithmetic without Diagrams." Artificial Intelligence, Vol. 8, p175-200, 1977.
14. F.M. Brown "Towards the Automation of Set Theory and its Logic". Artificial Intelligence, Vol. 10, p281-316, 1978
15. R.S. Boyer, J S. Moore. "Proving Theorems about LISP Functions", pp. 486-473. Proceedings of IJCAI3, Stanford, 1973, and Journal of ACM, January 1975.
16. W. Bibel "Synthesis of Strategic Definitions and their Control" Report 7610 Tech. Univ. Munchen, 1976.

17. R. E. Kling "A Paradign For Reasoning by Analogy," Artificial Intelligence, Vol. 2, No. 2, p. 147-178, 1971.

18. Murray R. Spiegel Finite Differences and Difference Equations. McGraw-Hill Inc. 1971.

19. Charles Jordan Calculus of Finite Differences. Chelsea Publishing Co. New York, NY. 1965. Library of Congress Catalog Card No. 58-27786.

20. C.L. Liu Introduction to Combinatorial Mathematics. McGraw-Hill Inc. 1968.

21. Edwin F. Beckenbach (editor) Applied Combinatorial Mathematics. University of California Engineering and Physical Sciences Extension Series, John Wiley & Sons, Inc. 1964.