Volume 5, Number 1, January–March, 1981

COGNITIVE SCIENCE

A multidisciplinary journal of artificial intelligence, psychology, and language

ABLEX PUBLISHING CORPORATION
Norwood, New Jersey

COGNITIVE SCIENCE 5, 55–77 (1981)

Mathematical Model Building in the Solution of Mechanics Problems: Human Protocols and the MECHO Trace

GEORGE F. LUGER

Department of Artificial Intelligence
University of Edinburgh

Current Address: Dept. of Computer Science
University of New Mexico

This paper describes model building and manipulation in the solution of problems in mechanics. An automatic problem solver, MECHO, solving problems in several areas of mechanics, employs (1) a knowledge base representing the semantic content of the particular problem area, (2) a means-ends search strategy similar to GPS to produce sets of simultaneous equations and (3) a "focusing" technique, based on the data within the knowledge base, to guide the GSP-like search through possible equation instantiations. Sets of predicate logic statements are employed to describe this model building activity. These clauses are used to give information content to the knowledge base and provide both basis and guidance for the goal driven search.

It is hypothesized that human subjects solving mechanics problems employ similar model building techniques. Protocols of several subjects are presented and comparisons are drawn with the traces of the automated problem solver. Adjustments are made to the program to provide a better fit of traces to protocols. Some implications are presented for using a rule based model for describing human problem solving performance in solving mechanics problems.

I. INTRODUCTION

There has been considerable interest in recent years in the implementation of automated problem solvers in applied mathematics. Bundy (1978), Bundy et al. (1979), de Kleer (1975), and Novak (1977) have had primary interest in designing an automated solver. For this group of researchers success is measured by a large set of problems their programs can solve.

Another group of researchers, Bhaskar and Simon (1977), Hinsley, Hayes,

and Simon (1977), Marples (1974), Larkin (1979), McDermott and Larkin (1978) and Simon and Simon (1978) have had the elucidation of human problem solving processes in the solution of applied mathematics problems as their goal. Indeed, the Simon & Simon and Larkin studies have made some interesting hypotheses on the differences between novice and expert problem solvers.

The automatic problem solver at the central focus of this paper is the MECHO program developed at the University of Edinburgh (Bundy, 1978; Bundy et al., 1979). Although the primary goal of the MECHO program has been to develop a comprehensive problem solver for selected areas of mechanics such as pulley systems, moment-of-inertia, and distance-rate-time problems, the analysis of problem solving protocols and other clues obtained from the successful human problem solver have been valuable aids. Indeed, some heuristics originally suggested by Marples (1974) in his work with Cambridge undergraduates and the implications suggested by the Hinsley, Hayes, and Simon study noted above played a very important role in the design of the MECHO problem solver.

The argument of this paper, based on the results of analyzing collected sets of protocols of subjects solving groups of mechanics problems, is to show not simply a prima facie similarity but a model theoretic equivalence between the running computer program and the successful human solver of mechanics problems. The evidence for this equivalence rests on two aspects of the problem solving. First, it rests on the construction of a knowledge base. With its *schemata* the MECHO program builds sets of facts, inferences, and default values sufficient for solving a class of problems. Although the specific contents of this knowledge base (Section II) are open to criticisms of "lack of generality," or "arbitrary selection of systems," nonetheless, the representation of a problem's semantic content in a form easily changed or manipulated, with some generality across problem types, and testable for sufficiency, is an important advance. Further comments on projecting a semantic knowledge base sufficient for problem solving will be made in Section V.

Secondly, with the Marples algorithm, the MECHO program offers a general method for derivation of equations sufficient to solve mechanics problems. The Marples algorithm, to be explained in detail below, is a goal-driven search employing means-ends analysis in many ways similar to GPS (Newell & Simon, 1963). The Marples algorithm is guided by the semantic information provided by the sets of schemata.

Finally, what helps to make the human/automatic problem solver comparison fruitful is MECHO's use of PROLOG. This language, developed at Marseilles and Edinburgh (Warren et al., 1978), simulates programming with the first order predicate logic (Kowalski, 1979). PROLOG approximates a linear resolution theorem prover for Horn clauses that is augmented by features to optimize resolution and control backtracking.

The important notion from the above description is that PROLOG is structured and executed as a *rule based* language. This allows the facts, inferences,

and various default assignments that comprise the semantics of the mechanics problem domain to be directly translated for and executed by the computer. Thus, and this notion will be addressed in detail below, it may be hypothesized that the presence, absence, or order of rules to be executed by the computer will be matched by the competence and/or production order of the human subject.

Rather than introduce full PROLOG notation in this paper, an equivalent predicate logic formalism will be used. Predicate logic statements have three forms. The first is *facts* such as "mass (object1, 10, kg)." For the "mass of object1 is ten kilograms" or "slope (path1, 180)" for "the slope of path1 is horizontal" (0° or 180° by convention). The second form of predicate logic statements is the *goals* to be tested for truth: " ⊢ slope (path1, 180)" is "test whether the slope of path1 is horizontal" or " ⊢ mass (object1, X, kg)" for "test whether there is a value for X that will be the mass of object1 in kilograms." Finally, there are the *inferences* that may be used, with the facts, to prove the goals. These take the form A ⊢ B, C, D as "A is true if B, C, and D are found to be true." In this sense, B, C, and D may be seen as subroutines to be used to test the truth of A. An example of an inference rule from mechanics is: "Tension (string1, T) ⊢ friction (pulley, zero), attached (string1, string2, pulley), tension (string2, T)." Roughly translated: "to determine the tention T in string1 find a frictionless pulley, find another string, string2 attached to string1 over the pulley, and find the tension of string2."

The following section (II) will introduce the two problem areas to be considered in this paper, discussing each problem and outlining its solution. There will then be a more complete description of MECHO's use of schemata and the Marples algorithm to solve these problems.

Section III considers human protocols in some detail. Subjects will be seen solving the same two problems MECHO solves in Section II. Adjustments are made to the program in an attempt to provide a better fit of trace to protocol in Section IV, and a comparison is made between trace and protocol. Section V summarizes the model building in solutions of mechanics problems and points out advantages of using rule based computer languages to model human problem solving performance.

II. TWO MECHANICS PROBLEMS AND THE MECHO SOLUTIONS

II. A. The Problems

The first problem is a simple pulley problem. There is neither friction in the pulley nor stretch in the string passing over the pulley. Objects hang vertically downwards, and the strings are seen as long enough so that objects never get all the way to the pulley. None of these facts, however, are explicit in the problem statement as taken from a typical undergraduate text in Mechanics (Palmer & Snell, 1956):

A man of 12 stone and a weight of 10 stone are connected by a light rope passing over a pulley. Find the acceleration of the man.

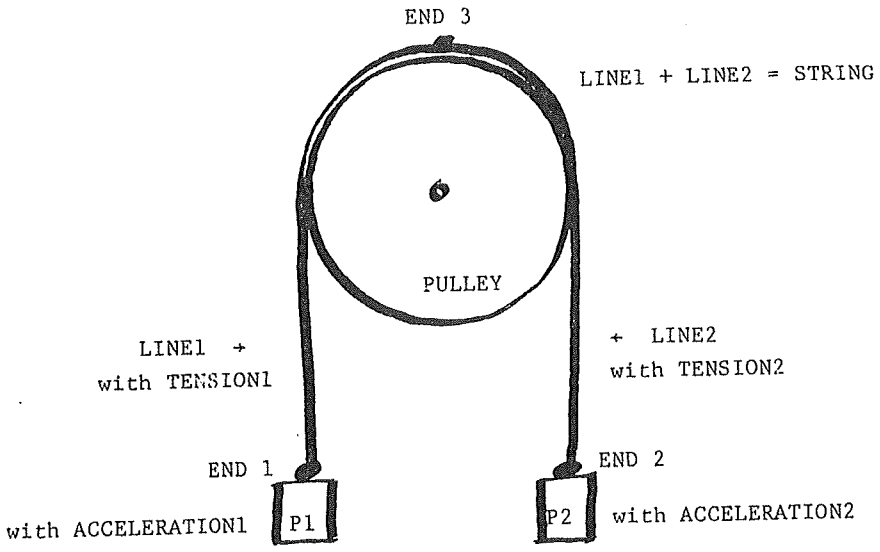


Figure 1. A representation of the entities created by the pulley schemata. TENSION1 = TENSION2 and ACCELERATION1 = ACCELERATION2 by schema inferencing.

The usual approach solvers take in attacking this problem is first to draw a figure similar to Figure 1, to construct a force diagram with tensions assigned to strings and accelerations assigned to objects, and finally to resolve forces at contact points of the string and hanging objects. The result of resolving forces at both contact points is to produce two independent equations sufficient to determine the acceleration of the man. Traces of two subjects solving this pulley problem may be found in the beginning of Section III. McDermott and Larkin (1978) describe a process very similar to this used by their subjects solving physics problems.

The second problem considered has a particle falling from the top of a tower. The solver is asked to determine the height of the tower when all that is known is the time t for the particle to fall the last h feet to the ground. The successful solver will know the acceleration due to gravity, will ignore friction in the air that might prevent unlimited constant acceleration, and will assume a straight path directly to the ground—ignoring, wind, rain, and the elements. These necessary assumptions are not stated in the problem, but must be understood for successful solution. The problem statement, taken from Palmer and Snell (1956) is:

A particle is dropped from the top of a tower. If it takes t seconds to drop the last h feet to the ground, find the height of the tower.

The usual solution has the solver create variables, perhaps similar to those in Figure 2, to represent total and partial distances, rates, and times for the falling object. Sets of simultaneous equations are then formed using these variables. Examples of three subjects solving the tower problem may be found in Section III.

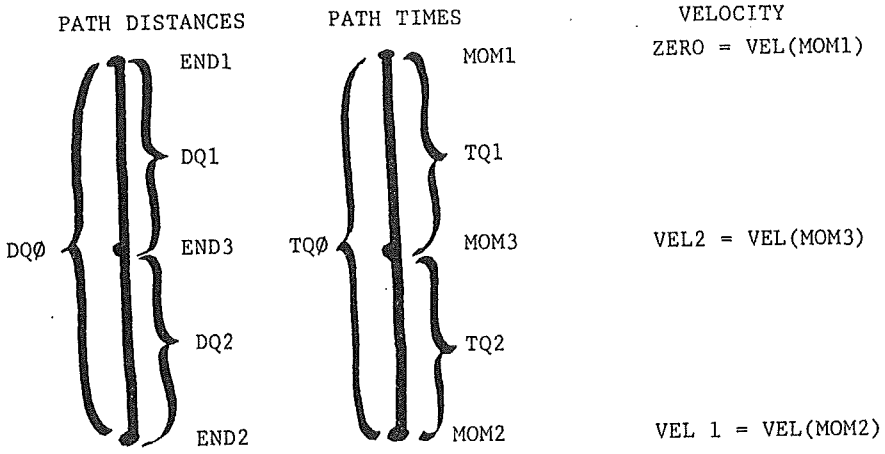


Figure 2. The representation of the tower problem created by the schemata.

II. B. MECHO Schemata and the Marples Algorithm

The MECHO problem solver may be divided into parts. The first part, the schemata, called by key-word-matching on the problem statement, "constructs" for the program the semantic relationships of the problem. The second, the Marples algorithm, is driven by the goal to be attained (the unknown to be solved), and reasons with the semantic relations in an attempt to construct sets of simultaneous equations sufficient to solve for the goal. Both these aspects of the MECHO program will be considered in some detail in this section.

MECHO creates the "semantics" of a problem situation through sets of schema calls (Luger & Bundy, 1977). The initial part of the schema, called "declarations," has two tasks. The first is to create data items and names to represent the semantic entities of the problem such as pulleys, strings, lines, ends for lines, etc. The second task is to call further related schemata. Thus, the *string* schema in pulley problems will call the *line* schema for various parts of the string.

The second part of the schema adds facts and inferences about the problem domain to the data base. The assertions for the string system, for example, divide the string into lines and assign a slope to each line. The inferences are also entered in the data base at this time. Thus, in a pulley problem, it is asserted that the acceleration found for one bit of the string may be assigned to the entire string if the string is not elastic.

The final part of the schema is a set of default values. These facts and inferences are placed at the bottom of the set of clauses of the same name in the data base, and they are to be called only if every other clause of that name has failed. Examples of default values are that a string is not elastic or that a string is light, i.e., has no mass. The default of a path is that it is straight and that it has no friction. Examples of the MECHO schemata for pulley and tower problems are given in the next subsection.

II. B. I. The Schemata. The schema calls used in the pulley and distance-rate-time problem are presented in Table 1. Using Table 1 and the discussion below, where schema names are italicized, the full set of clauses used by MECHO may be produced. (Luger, 1980, also has a full listing of these clauses.) In Table 1, the three lists "[. . .]" in each schema present the declarations, assertions, and default assignments mentioned in the previous section.

The pulley problem to be solved is recognized as a *standard* pulley problem. The *standard* problem has string and attached objects hanging vertically, moving at 90° and 270°, by convention. The *standard* pulley is a special instance of the more general pulley *major*. Pulley *major* has fixed contact between the ends of the string and the particles, asserts a constant acceleration of the objects, and infers that the acceleration of both particles is the same if the string is inextensible.

The pulley *major* is a particular instance of pulley *minor*. Pulley *minor* has constant contact of the string and pulley, but need not have fixed contact with the particle (suppose the man of the pulley problem above began to climb up the rope). The string in pulley *minor* has a direction tangent and tension assigned to the string on each side of the pulley. This tension is inferred to be common to the entire string if there is no friction in the pulley, and indeed, the pulley has default assignments of no friction and no mass.

Finally, the pulley system is made up of *string* and *line* schemata. These create lines to represent the string segments on each side of the pulley. End points are created for each of these segments. Default values make the string light, inextensible, and straight.

One of the goals of designing the system of schemata as described above was to represent the semantics of the pulley world in such a way as to be able to solve a large and diverse set of pulley problems. These include pulleys over wedges, at edges of tables, with objects on both friction and non-friction surfaces, and combinations of pulleys and objects, such as pulley systems themselves being attached to the ends of the strings of other pulley problems (Bundy et al., 1979).

The tower problem, as a distance-rate-time problem, calls *path*, *time*, *partition*, and *motion* schemata. The *path* schema calls the *line* schema (the same line schema used by the pulley problem). This creates an object called line and gives it a point at its leftend and a point at its rightend. Then the *path* schema

TABLE 1
Schemata for the Pulley and Tower Problem

Each schema is composed of three parts, each denoted by a list [A, ...]. Words with first letter capitalized are variables.

```
schema(pulleysystem-standard(Sys,Pulley,String,Particle1,Particle2,Time)
     [call(pulleysystem-major(Sys,Pulley,String,Particle1,90,Particle2,270,Time))]

     [  ]
     [  ] ).

schema(pulleysystem-major(Sys,Pulley,String,Particle2,Direction1,Particle2,Direction2,Time)


      [call(pulleysystem-minor(Sys,Pulley,String,Direction1,Direction2,Time)),
       end(String,Leftend,left),
       end(String,Rightend,right))]
      [constantacceleration(Particle1,Time),
       constantacceleration(Particle2,Time),
       contact(Particle2,Rightend,Time),
       (relativeacceleration(Particle1,Pulley,A1,Direction1,Time)
              ├─elastic(String,zero),
                 relativeacceleration(Particle2,A1,Direction2,Time)) ]
      (relativeacceleration(particle2),Pulley,A2,Direction2,Time)
              ├─elastic(String,zero),
                 relativeacceleration(Particle1,Pulley,A2,Direction1,Time)) ]

     [  ] ).


schema(pulleysystem-minor(Sys,Pulley,String,Direction1,Direction2,Time).
      [call(stringsystem(String,Leftbit,Pulleypt,Rightbit, Time)].
      [problemtype(pulley,Time),
       isa(particle,Pulley),
       contact(Pulley,Pulleypt,Time),
       tangent(Leftbit,Time)
       tangent(Rightbit,Time),
      (tension(Leftbit,T1,Time)
         ├─coefficient of friction(Pulley,zero),
           tension(Rightbit,T1,Time) ),
      (tension(Rightbit,T2,Time)
         ├─coefficient of friction(Pulley,zero),
           tension(Leftbit,T2,Time) ) ],
      [coefficient of friction(Pulley,zero),
       mass(Pulley,zero,Time) ] ).


schema(stringsystem(String,Leftbit,Pulleypt,Rightbit,Time),
      [call(linesystem(String,Leftend,Rightend)),
       call(linesystem(Leftbit,Leftend,Pulleypt)),
```

```
        call(linesystem(Rightbit,Pulleypt,Rightend)) ],
       [partition(String,(Leftbit,Rightbit)),
        isa(string,String),
        isa(string,Leftbit),
        isa(string,Rightbit) ],
       [concavity(Leftbit,straightline),
        concavity(Rightbit,straightline),
        elastic(String,zero),
        mass(String,zero,Time) ] ).


schema(linesystem(Line,Leftend,Rightend),
       [isa(line,Line),
        end(Line,Leftend,left) ,
        end(Line,Rightend,right)] .
       [isa(point,Leftend),
        isa(point,Rightend) ],
       [  ] ).


schema(timesystem(Time,Moment1,Moment2),
       [isa(period,Time),
        initial(Time,Moment1),
        final(Time,Moment2) ],
       [isa(moment,Moment1),
        isa(moment,Moment2) ],
       [  ] ).


schema(pathsystem(Name,Leftend,Rightend,Slope,Concavity),
       [call(linesystem(Name,Leftend,Rightend)) ],
       [isa(path,Name),
        concavity(Name,Concavity),
        slope(Name,Slope) ],
       [coefficient of friction(Name,zero) ]).


schema(motion(Particle,Path,Start,Side,Time).
       [farend(Path,Finish,Start) ,
        final(time,End) ,
        velocity(Particle,V,Direction,End) ,
        typoint(Path,Point) ,
        initial(Time,Begin) ],
       [at(Particle,Finish,End),
        contact(Particle,Finish,End),
        contact(Particle,Start,Begin),
        contact(Particle,Point,Time) ],
       [constantvelocity(Particle,Time),
        constantacceleration(Particle,Time),
        velocity(Particle1,zero,270,Start) ]).
```

assigns the path friction, concavity, and slope variables. In default the path is said to have zero friction, to be straight, and to be horizontal.

A partition schema is called that divides the *path* and *time* period into two segments so that leftend of the path is the leftend of the first segment, the rightend of the path is the rightend of the second segment, and a new point is created to be the rightend of the first and leftend of the second segment. Similarly for time, the first moment of the time period is asserted as the first moment of the first subperiod and the final moment of the entire period as the final moment of the second subperiod, with a new moment created and asserted as the last moment of the first subperiod and the first moment of the second supberiod. The set of default values of the path and time systems hold across the partitions of the path and time interval. Finally, a *motion schema* is called for a particle starting on a side of the path during the time period. This schema effectively unites for the particle the entities created by the path and time schemata and their partitions.

II. B. II. MECHO as Goal Driven. The Marples algorithm, first suggested for use in mechanics problem solving by D. Marples from his work with engineering students at Cambridge (Marples, 1974), represents the consequent or goal-driven inferencing of MECHO. This algorithm starts with the unknown of the problem and searches backwards through possible equation instantiations until a set of simultaneous equations sufficient to solve the problem is produced.

MECHO, however, is not blindly goal-driven; it has a "focusing" technique which "forces" the Marples algorithm to consider equations appropriate to a particular sought unknown, rather than thrash about through all possible equations. For example, focusing in the pulley problem forces the Marples algorithm to consider first the general resolution of forces equation at the contact points of the string and weights in order to determine the acceleration of the man, rather than considering other possible acceleration equations.

The Marples algorithm is also able to create new or "intermediate" unknowns in the process of solving for the desired unknowns of a problem. In the pulley problem, the desired unknown is the acceleration of the man. But, resolution of forces cannot solve the problem without creating a new unknown, representing the tension in the string at its end point. This tension is inferred to be common the entire string, so resolution of forces at the contact point of string and weight, where the acceleration is inferred to be the same as that of the man, gives two equations sufficient for solution of the problem $T - 10g = 10$ and $-(T - 12g) = 12a$.

The equations to be applied in each problem are described by predicate calculus clauses where the equation is asserted if each of a set of conditions is met. Consider the clause for resolution of forces where words with first letter capitalized, such as Direction and Time, and single capital letters, such as F and M, represent variables:

```
isformula (F = M*A, resolve – (Particle, Direction, Time))
├— Mass (Particle, M, Time),
acceleration (Particle, A, Direct, Time),
sumforces (Particle, Direction, Time, F).
```

This clause asserts the formula F = M*A, where F, M, and A are instantiated variables when the three conditions after " ├—" are met. The equation is called "resolve" and is tied to a particular particle moving in some direction during a period of time (focusing determines these values for a particular goal). The three conditions are that the particle has a mass M, that an acceleration can be found or deduced for the particle in the required direction, and finally, that sumforces calculates the sum of all forces, such as the tension in the string, acting on the particle during the time period.

Another, and possibly more general, way of viewing the goal-driven search of MECHO is as a GPS machine (Newell & Simon, 1963; 1972). The givens and the goal of a particular problem are determined at the outset of the search. Each possible equation instantiation, such as the "resolve" clause discussed above, offers a procedure for reducing the differences between the givens and the goal. Usually, the goal is not immediately achieved by an equation instantiation and the givens. What results is that equations are found which reduce the givens/goal difference at the cost of introducing new intermediate unknowns required to link the givens and the goal. If the list of possible equation instantiations represent a "table of differences or connections" (Newell & Simon, 1963) that a GPS machine uses in goal driven search, then the focusing technique, by creating a queue or priority list, directs the GPS machine through the table of differences or connections.

Although the search for equations is short in the pulley problem, with immediate focusing on the general resolution of forces equation and the creation of only one intermediate unknown, it is not so simple with the tower problem, where several possible sets of equations may be produced. In the tower problem, when the length DQ0 of the path for the particle to travel is sought, only equations having "length" variables are considered, with particle, path, and time all known and fixed. Similarly, when TQ0, the total time the particle takes to travel the distance, is introduced as the first intermediate unknown, a queue of time equations is set up with particle, distance, and time fixed for that situation. Thus, the Marples algorithm reasons from the goal and is guided in its search by the semantics of the problem.

III. FIVE PROTOCOLS OF HUMAN SUBJECTS

In this section five protocols of subjects solving the pulley and tower problems are presented. The subjects were postgraduate students at the University of

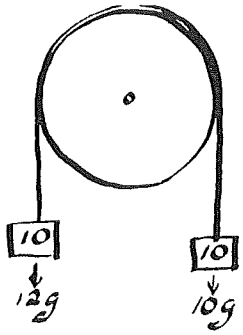
Edinburgh and all had had some mechanics or applied mathematics in their undergraduate education. The "figures" the subjects drew, at approximately the place in the protocol they drew them, are placed at the right of the protocol. Breaking of protocols into phrases represents the author's transcription of the tape recorded protocols. Sets of periods ". . ." represents pauses.

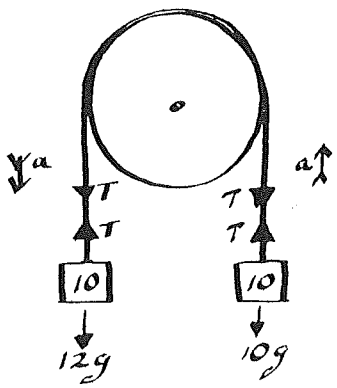
III. A. A Pulley Problem (Palmer & Snell, p. 21):

A man of 12 stone and a weight of 10 stone are connected by a light rope passing over a pulley. Find the acceleration of the man.

Protocol A

1. It's a standard pulley problem . . .
2. We'll draw a diagram, pulley with a light rope passing over.
3. Weight of 10 stone on one side so you draw a 10g force . . .
4. Don't worry about units 'cause they are the same.
5. On the other side you've got a man, draw another block there . . .
6. With a force of 12g on it . . .
7. So you got an acceleration of the man draw it downward for convention.
8. Tension in the rope is . . .
9. Doesn't say the pulley is smooth so assume it is.
10. So tension T in the rope on both sides . . .
11. We've got the 10 gram . . . 10 stone mass accelerating.
12. With the same acceleration as the man
13. In the opposite direction . . .
14. Assuming it's an inextensible rope . . .
15. So resolving that and that . . .
16. Forces want to match on either side of the pulley.
17. We've got T − 10g times the mass of the block which is 10.
18. No-nonsense . . . we've got force equals mass times acceleration.
19. That's T − 10g = 10A
20. and T − 12g = −12A
21. We want to eliminate T . . . etc. . . .



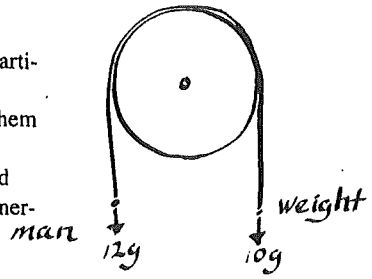


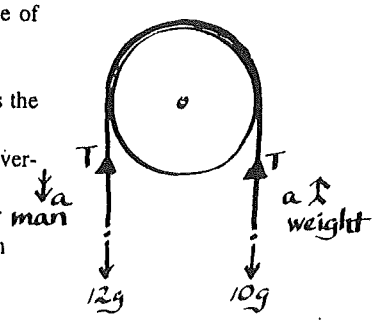
T − 10g = 10a

T − 12g = −12a

Protocol B

1. Mechanics problem . . .
2. We'll treat the man and weight both as particles, point masses . . .
3. I'll put two dots on the paper and join them by a rope looped over a pulley . . .
4. Ah, yes . . . this pulley is underspecified
5. I'll assume it's light, no moment of inertia . . .
6. and I'll assume it's frictionless . . .
7. Alternatively, it could be a smooth fixed pulley.
8. Perhaps this is all irrelevant . . .
9. The rope just passes over it with no friction
10. So man and weight . . . Man has a force of 12 stone vertically downwards
11. Unknown at the moment . . .
12. and assuming this frictionless pulley, T is the same on both sides . . .
13. We'll give the rope an acceleration . . . vertically down on the man's side
14. and vertically up on the weight's side
15. So resolving vertically down for the man $T - 12g = -12a$
16. The vertically downwards acceleration
17. And vertically upwards for the weight $T - 10g = 10a$
18. so the thing requires the acceleration of the man which is *a* . . .
19. So we just eliminate T from these two equations . . . etc.





$T - 12g = -12a$

$T - 10g = 10a$

III. B. Distance-Rate-Time Problems (Palmer & Snell, p. 21):

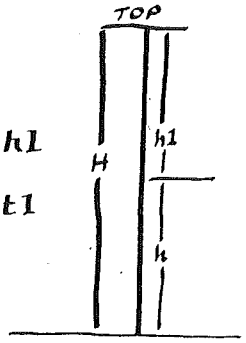
A particle is dropped from the top of a tower. If it takes *t* seconds to travel the last *h* feet to the ground find the height of the tower.

Protocol C

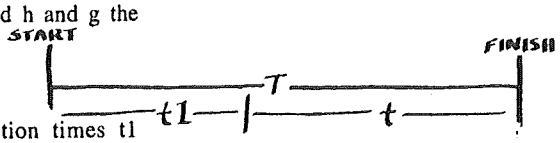
1. There is a tower, suppose it has height H
2. H is made up of h, the given height
3. and h1, the unknown portion of the tower
4. We also know time t
5. Call the total time of falling, T, T is made up of t plus t1 where t1 is the tine for the top part.

$H = h + h1$

$T = t + t1$



6. I need H and I'm given T and h and g the acceleration of the object
7. I know that H = h1 + h
8. Now to get h1, a distance
9. h1 equals one half acceleration times t1 squared h1 = ½g × t1²
10. and I have an equation with t1 already T = t1 + t
11. but I still need to find T, the total time . . .
12. Now, for the whole period, H = ½g × T²
13. Reviewing, I have one, two, three, four equations (indicates each)
14. and H, T, h1 and t1 are unknowns
15. That should do it . . .

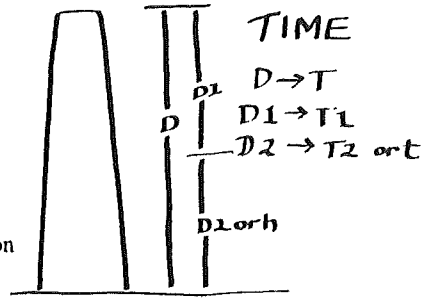


$h1 = \frac{1}{2} g \times t1^2$

$H = \frac{1}{2} g \times T^2$

Protocol D

1. For the tower t and h are known . . .
2. draw a tower, D is distance to top
3. made up of D1 and D2
4. Where D2 is given as h . . .
5. Call T the time for the ball to travel D,
6. and for completeness
7. T1 is the time the ball crosses D1
8. and T2 the time for D2, but T2 is t . . .
9. Now I want D, knowing g is the acceleration of the ball, D = ½g × T²
10. and I know T is T = T1 + t
11. So now I need T1 . . . in the top time period
12. The final velocity . . . Call it VM . . . is acceleration times time VM = g × T1
13. and now to get VM . . .
14. The total distance D . . .
15. That won't help . . .
16. The distance h and velocities
17. if VF is final velocity VF = VM + g × t
18. And I can easily get the final velocity VF VF = g × T
19. Because I've already got T.
20. I'll rewrite the equations and see if I have enough:
21. D = ½g T² D T
22. T = T1 + t T1
23. VM = g × T1 VM
24. VF = VM + g × t VF
25. VF = g × T
26. Yes, five equations in five unknowns.



$D = \frac{1}{2} g \times t^2$

$T = T1 + t$

$VM = g \times t1$

$VF = VM + g \times t$

$VF = g \times T$

$D = \frac{1}{2} g T^2$

$T = T1 + t$

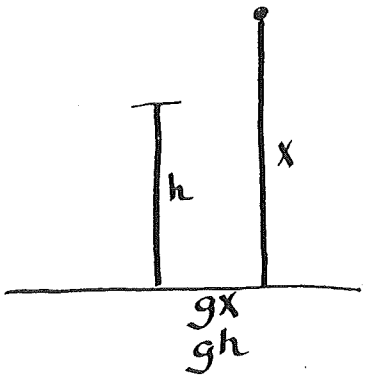
$VM = g \times T1$

$VF = VM + g \times t$

$VF = g \times T$

Protocol E

1. So we produce a diagram, body is h feet above the ground
2. no, the body starts at some other height above the ground, call it x . . .
3. Do this by energy . . .
4. It initially starts with potential energy mg
5. where m is the mass of the body, m is a constant . . .
6. we can forget about m, call it 1
7. gx is the initial potential energy gx
8. when it reaches height h, it will have potential energy gh
9. and it will have kinetic energy gh
10. so it will have $gx - hx = \frac{1}{2}v^2$
11. where v is its velocity at height h
12. Okay, we have velocity at height h
13. we want to know how long it's going to hit the ground starting at that velocity . . .
14. which equation do we want to use . . .
15. we want to use the constant acceleration which is v^2
16. no, we don't know what height . . .
17. what speed it is once it hits the ground
18. we want to use one of the other constant acceleration ones $v^2 = u^2 + 2aS$
19. we want to . . . an equation relating height, velocity and time, that is
20. under constant acceleration . . . ah . . . which is $S = vt + \frac{1}{2}at^2$
21. OK



$gx - hx = \frac{1}{2}v^2$

$v^2 = v^2 + 2as$

$S = vt + \frac{1}{2}at^2$

IV. COMPARING HUMAN AND MECHANICAL MODEL BUILDING

IV. A. Focusing and Altering the MECHO Trace

In the pulley problem, focusing produced immediate results. In fact, MECHO, like the human solvers, performed very little search in coming to the resolution-of-forces formula. Two applications of this formula and the problem is solved. The tower problem was much more interesting. In fact, there are several different sets of simultaneous equations (and of course permutations within each set) that will solve the problem. Each solver produced a different set of equations and each of these sets differed from MECHO's trace (Luger, 1980).

In this section the focusing technique and table of connections is altered in an attempt to produce traces similar to the human protocols of the tower problem.

First note that the semantic knowledge base developed by the schemata for the tower problem will not be changed. That is, the same set of clauses representing the semantics of the tower problem are used by MECHO throughout this section. Furthermore, the first part of focusing that binds situation variables for sought unknowns will remain unchanged. That is, DQ0 will remain the LENGTH of PATH during the TIME. What will change is the queue of possible equation instantiations for situations, and alterations will be made in the equations within the table of connections. These latter will be seen shortly.

The ten clauses below are used to form the queue of formulae to be attempted in any problem situation. Resolve, relative velocity, etc., refer to the equations available to MECHO. A full description of these equations may be found in Luger, 1980.

1. relates(resolve, [FORCE,ACCELERATION,MASS]).
2. relates(relative-velocity, [VELOCITY]).
3. relates(relative-acceleration, [ACCELERATION]).
4. relates(constant-acceleration-1, [ACCELERATION,VELOCITY,DURATION]).
5. relates(constant-acceleration-2, [ACCELERATION, LENGTH, VELOCITY, DURATION]).
6. relates(constant-acceleration-3, [VELOCITY, LENGTH, DURATION]).
7. relates(average-velocity,[VELOCITY,LENGTH,DURATION]).
8. relates(constant-velocity,[VELOCITY,LENGTH,DURATION]).
9. relates(lengthsum,[LENGTH]).
10. relates(timesum,[DURATION]).

After DQ0 is recognized as the LENGTH of the PATH during the EPISODE, the queue of possible equations is formed by searching the ten clauses above to find which formulae will solve for LENGTH. The queue is: constant acceleration-2, constant acceleration-3, average velocity, and lengthsum. These equations are tried in that order. Each, of course, fails because new unknowns are introduced. On the second pass, when new unknowns are allowed, constant acceleration-2 is accepted. This process continues until the problem is solved.

It can be hypothesized that the human subject has a stack of equations that relate to specific situations. These equations are employed when attempting to reduce the given-goal differences in a specific problem. In fact, the stack might be quite similar to that produced by the ten clauses above. This clause queue need *not* contain the full equations, only their names referencing the actual equation formulae which are stored with their full sets of conditions for instantiation (cf. earlier example of the "resolve" formula, II.C), in "long-term" store.

This proposed implementation of the GPS model of problem solving may be tested by making simple alterations in the ten clauses above to see if the different queue of equations to be formulated can produce a different set of simultaneous equations. In particular, we attempt to produce traces similar to the human protocols C,D, and E.

Subject C used the lengthsum equation with top priority when solving for

LENGTH, and timesum when a DURATION was sought. Thus, if clause 9 and 10 are placed before the constant acceleration clauses, the order of "relates" above becomes 1, 2, 3, 9, 10, 4, 5, 6, 7, 8. When this re-arrangement of clauses (and consequent re-arrangement of equation queues) was run in MECHO the following trace occurred:

F. 1. Attempting to solve for DQ0 in terms of [g,DQ2, TQ2].
2. DQ0 = DQ1 + DQ2 solves for DQ0 but introduces [DQ1].
** DQ1 is a LENGTH, the lengthsum cannot be used again, so the second in the queue, constant acceleration-2 is used**
3. DQ1 = ZERO × TQ1 + ½ × g × $TQ1^2$ solves for DQ1 but introduces [TQ1]
** TQ1, a DURATION, creates a queue with timesum first**
4. TQ0 = TQ1 + TQ2 solves for TQ1 but introduces [TQ0].
** constant acceleration-2 is now the first on the queue for TQ0 since timesum may not be used again, and MECHO always tries to solve without further introduction of unknowns**
5. DQ0 = ZERO × TQ0 + ½ × g ½ $TQ0^2$
6. Equations F 2-5 solve the Tower problem.

If the ZERO term (initial vel. × time) is removed from 3 and 5 these equations are exactly those produced by subject C above.

In an attempt to produce a trace similar to protocol D, the "relates" clause (9) for lengthsum is returned to its original position, (i.e., 1, 2, 3, 10, 4, 5, 6, 7, 8, 9). The subject of protocol D does not use the constant acceleration-2 equation to its full potential, that is, he only used the equation when the initial velocity is zero. Marples (1974) comments on this use of equations by engineering students when he notes they often apply an equation without knowing its full power. In this instance, the subject uses constant acceleration-2 for relating acceleration, time, and distance and not in its full use of relating initial velocity, acceleration, time, and distance. If it is conjectured that this happens with subject D, the constant acceleration-2 equation is changed to this limited use by rewriting 5 above to "relates (constant acceleration-2,[ACCELERATION, LENGTH, DURATION])" and removing the initial velocity component of the equation itself.

MECHO is now run with these changes and the following trace results:

G. 1. trying to solve for DQ0 in terms of [g, TQ2, DQ2].
2. DQ0 = ½ × g × TQ0 solves for DQ0 but introduces [TQ0].
3. TQ0 = TQ1 + TQ2 solves for TQ0 but introduces [TQ1].
4. VEL2 = ZERO + g × TQ1 solves for TQ1 but introduces [VEL2].
** Recall that the schema for motion named the final velocity at the bottom VEL1 and the velocity at the midpoint VEL2. The solver using constant acceleration-2 properly would now be done. Our subject, with limited resources, grinds on**
5. VEL1 = VEL2 + g × TQ2 solves for VEL2 but introduces [VEL1].
6. VEL1 = ZERO + g × TQ0 solves for VEL1.
7. Equations G 2-6 solve the Tower problem.

This trace is remarkably similar to protocol D.

The subject of protocol E, from the start of problem solving, decided to use energy equations. This was not expected by the investigator taking protocols or designing MECHO to solve distance-rate-time problems. But in principle there was no reason why energy equations could not be used. They were, in fact, already in the system and used to solve de Kleer's problems (Bundy, 1978). A new entry for the "relates" table was constructed: No. 11 relates (conserve energy, [VELOCITY,LENGTH]) and this was given priority over all other LENGTH relation clauses. Further, constant acceleration-3 is equivalent to conserve energy and was removed. Finally, subject E favored constant acceleration-2 over the constant acceleration-1 formula for solving VELOCITY problems, so this order was changed. The "relates" list was 1, 2, 3, 4, 10, 11, 6, 8, 9, 5.

MECHO was run in this situation; its trace:

H. 1. Trying to solve for DQ0 in terms of [g, TQ2, DQ2].
2. $\frac{1}{2} \times VEL1^2 - \frac{1}{2} \times ZERO^2 = g \times DQ0$ solves for DQ0 but introduces [VEL1].
** The energy equation is attempted again. This time to solve for VEL1**
3. $\frac{1}{2} \times VEL1^2 - \frac{1}{2} \times VEL2^2 = g \times DQ2$ solves for VEL1 but introduces [VEL2].
4. $DQ2 = VEL2 \times TQ2 + \frac{1}{2} \times g \times TQ2^2$ solves for VEL2.
5. H. 2-4 solve the Tower problem.

It can be seen that this trace is very close to the protocol of subject E above. Further comments will be made in the next section.

IV. B. Comparison of Traces and Protocols

This paper has argued that humans solving problems in applied mathematics use a knowledge base representing the semantic content of the problem domain. Previous work in this area was cited (Hinsley, Hayes, & Simon, 1977; Bhaskar & Simon, 1977), and our own protocols were offered to demonstrate the existence and to hint at the composition of this knowledge base. In Section II a computer program, the MECHO problem solver, was seen to create such a semantic knowledge base. In this section the problem solvers' protocols will be compared with the trace of the MECHO program. This comparison will be in two general areas: the use of a semantic knowledge base and of a goal-driven search.

IV. B. I. Schemata and Problem Semantics. When the human subject is given a paper and pencil and asked to solve the pulley or distance-rate-time problems, each subject interviewed sketched out the physical problem and then attached force, acceleration, time, etc., variables to the appropriate locations. This is what occurs in lines 2–10 of protocol A, 2–6 of B, 1–5 of C, 2–8 of D, 1–2 of E. In these statements of the protocol, the relationships of the domain are sometimes explicitly made, and sometimes implicitly, i.e., without a corresponding verbal utterance. The statement is explicit in Protocol A, "a pulley and a rope passing over . . . weight of 10 stone on one side . . . on the other you've got a

man.'' But many of the relationships here are implicit, i.e., a fixed-contact between man and rope and a sliding contact between rope and pulley will be drawn on the paper but not described verbally.

Similarly for the tower problem, line 5–8 of protocol D says ''Call T the time for the ball (sic) to travel D and for completeness T1 is the time the ball crosses D1 and T2 the time for D2.'' Nowhere does the solver explicitly state that the last second of time period T1 is immediately prior to the first second of period T2, and yet later the velocities of the particles at these two times are equated.

In a further example of the richness of the problem domain, variables are introduced in the solver's description of the problem that were not mentioned in the problem statement. ''Tension in the string'' for the pulley problem and the acceleration constant ''g'' in both problem domains are introduced with no other justification than that they are part of the semantics of the domain (A8–10; B12; C6; D9), and are necessary for the solution. They are not mentioned in the statements of either problem.

MECHO's pulley problem trace creates entities to represent the string, its endpoints, the pulley, etc. It also makes fixed contact assertions and assigns slope variable to represent concavities and friction. The tower problem trace also creates a path for travel and a system of partitions of path segments and time periods to represent the physical situation. This is very similar to the figures subjects drew and to references made in lines C1–6 and D2–8 of the protocols.

The problem solvers' protocols contain several explicit ''inferences'' about their problem domains that relate the objects and properties such as accelerations, tensions, and time periods. In lines A8–10 ''Tension in the rope is . . . doesn't say the pulley is smooth so assume it is . . . so tension T in the rope on both sides. . . .'' Again A11–14 ''We've got . . . mass accelerating with the same acceleration as the man . . . assuming it's an inextensible rope . . .'' Similar ''inferences'' are made in B12; in B13–14 no explicit mention of ''inextensible rope'' is made, but the same acceleration is assigned to the rope on either side of the pulley. These inferences correspond very closely to the inferences asserted as part of the pulley problem schema (Table 1).

There is also a set of ''default'' facts that are part of the knowledge base of the experienced problem solver. Thus A9 ''doesn't say the pulley is smooth, so assume it is . . .,'' A14 ''assuming it's an inextensible rope,'' B4–6 ''This pulley is underspecified . . . I'll assume it's light . . . no moment of inertia . . . and I'll assume it's frictionless.'' These default values correspond closely to pulley schema assertions (Table 1).

The subjects also use default assignments such as the velocity of the particle is zero at the start of the path C9—implicit in the use of this equation—and D12 and D17. Furthermore, the acceleration of the object falling is ''g'' and ''downwards.'' This ''g'' is also constant over the path. The *path*, *time*, and *motion* schemas of Table 1 have the same default values.

Much of the semantic content of the human problem solvers' knowledge

base is only implicit in what the solver says and the figures that are drawn. Besides the similarities of trace and protocol, an identification of the "sufficiency" of the semantic knowledge base created by MECHO is its ability to support the Marples algorithm for generation of sets of simultaneous equations. This is sufficient with the pulley problem, but even more impressive for the tower problem when, depending on the "focus" used by the Marples algorithm, four different sets of simultaneous equations are created.

IV. B. 2. The Marples Algorithm. Goal-driven search using means-ends analysis for problem solving is not new (Polya, 1945; Newell & Simon, 1972; Ernst & Newell, 1969), but its use as part of a problem solver in applied mathematics is. A "givens" list and "unknowns" list are determined from the problem statement. Possible equation instantiations are used to "reduce" the differences between these two lists. A "focusing" technique forms a queue of possible equations for consideration. This queue is examined to see if any of the equations will solve for the unknowns using only the givens. If this fails the queue is examined again and new intermediate unknowns may be created to attempt to reduce the givens/unknowns differences.

In the pulley problem, both protocols (A & B) indicate that the subject goes immediately to the resolution of forces equation. This is used to create the intermediate unknown of the tension in the string. Forces are again resolved at the other end of the string to solve for tension and two simultaneous equations are produced sufficient to solve the problem.

The Marples algorithm proceeds in exactly the same fashion with the important difference that the first resolution of forces equation, the first equation considered, is rejected because it introduces a new unknown (tension). All other equations in the queue trying to find acceleration for a particle in a time period are examined and rejected before the return to the resolution of forces equation and introduction of the tension as a new unknown. The difference between the experienced human and MECHO is obvious and interesting. The human realizes immediately a new unknown must be introduced, accepts it, and goes on, while MECHO tries as long as possible to refrain from this course of action. Other than this, the protocol of the human and MECHO's trace for solving pulley problems are remarkably similar.

The similarities between trace and protocol are even stronger with the tower problem. Using MECHO's possible equations many different sets of simultaneous equations sufficient for solving the tower problem may be produced. In MECHO the formation of any set of these equations depends on the use of the focusing technique, that is, the queue of possible equation instantiations that is generated. In Section IV.A, systematic alterations were made to the focus and the result indicates the robustness of the Marples algorithm and focusing to generate different sets of simultaneous equations to fit closely the traces of the human subjects.

V. CONCLUSIONS

The goal of this paper has been to describe model building activity in the solution of applied mathematics problems. MECHO's solutions consisted first of invoking sets of schemata that provided semantic knowledge for the particular problem domain. Predicate calculus statements were presented (Table 1) that actually represented the information content of the sets of schemata. After creation of the knowledge base, the Marples algorithm was invoked and using means-ends analysis, in many ways similar to GPS, produced sets of simultaneous equations sufficient for solving the problem.

Experienced human subjects solving the same problems were presented in Section III. These subjects called on a store of general semantic information related to the problem domain to solve each problem. Both general and specific comparisons were made between the information used by subjects and the content of the MECHO schemata. Furthermore, it was shown that the subjects' strategies of producing sets of simultaneous equations sufficient to solve a problem were in many ways similar to the Marples algorithm. Finally, with slight changes, the Marples algorithm could produce sets of equations almost identical to those produced by the human subjects.

The expertise of the subjects of this study was somewhere between that of the novice and expert solvers described by Larkin et al. (1980). The subjects had little trouble solving the problems and derived their knowledge from general principles (Newton's laws, energy) known about the problem domain. The subjects were aware of the unknown throughout the problem solving and with little trouble put together a sequence of equations with intermediate unknowns allowing them to solve for the unknown. In MECHO the Marples algorithm provides a goal-driven search but with the presence of the "focusing" and "relates" predicates (IVA) there is very little search that actually takes place. Indeed, except for MECHO's attempts to solve the problem with the introduction of as few as possible intérmediate unknowns, it can go directly to the appropriate equations.

One of the principal advantages of writing the MECHO problem solver in PROLOG (here represented by predicate logic assertions) is that PROLOG actually computes using the predicate logic statements themselves. In fact, as mentioned in the introduction, PROLOG was designed as a predicate logic theorem prover. Facts, inferences, and default values are entered into the program as potential "meaningful bits of behavior." Thus the inference "to find a tension in a string on one side of a pulley, find the tension in the string on the other side of pulley and check if the pulley is smooth" is in the knowledge base in the form of a predicate logic inference. This allows removing the rule from the program, substituting another rule, or simply changing the order of the rules and then checking the results of these changes on the running computer program. Thus, a PROLOG fact, inference rule, or default assignment may be paired with the corresponding competency in the human subject and the effect of its presence

or absence in the human subject may be simulated by the running program. This can be seen when the "conservation of energy" equation was added for solution of the tower problem (IVA). The added new rule resulted in the exhibition of a new competency, and conversely, the absence of the rule marked the absence of the related ability. Indeed, it should be feasible to enter the 50,000 patterns that Larkin et al. (1980) suggest make up the index set of the experts' factual knowledge, actions, and strategies.

A modular set of rules also allows general purpose algorithms, such as the Marples algorithm discussed above, to be implemented and the effects of the presence of this algorithm to be seen by running the program. In this manner Larkin et al. (1980) and Simon and Simon (1978) have run sets of production rules in an attempt to simulate the differences in skills of the expert and novice problem solvers. Young (1976) has also used the modularity of production rules to describe children's acquisition of seriation skills.

The presence of production or behavior rules also provides a model for the interpretation of missing or ambiguous behavior of the human subject. Take for example protocol E above. E3 "Do this by energy" indicates an energy equation will be called to find the height of the tower. E7 states "gx is the initial potential energy" . . . and E8 "when it reaches height h, it will have potential energy gh . . ." and finally in E10–11 "so it will have $gx - gh = \frac{1}{2} V^2$ where V is the velocity at height h . . ."—what does this all mean? The protocol E3 to E11 is at best confusing. Reading MECHO's trace on the same problem goes a long way towards sorting out these vagaries. H2 gives a full description of gx "g* DQ0 = $\frac{1}{2}$ VEL1²" (where h* DQ0 is gx and VEL1 is the final velocity). Similarly for gh, H3 says "g* DQ2 = $\frac{1}{2}$ VEL1² − $\frac{1}{2}$ VEL2²" (where g* DQ2 is gh, VEL1 is final velocity and VEL2 is the velocity at the top of h). Now simply subtract H3 from H2 and line E10 of the protocol is precisely understood. The subject of protocol E was a particularly bright individual who liked to skip steps and simplify as he went along. Although MECHO is not able to imitate this behavior completely, its rule system does give a precise and complete performance and often provides data sufficient to disambiguate the human subject's behavior.

There are, of course, many things that MECHO, as presently designed, does not do that human subjects do quite easily. We have already seen how subject E simplified as he went along and omitted "unnecessary" bits of equations. Similarly, other subjects left out terms of equations that were zero—this could be easily added to MECHO. Also, MECHO is exhaustively thorough in its search while human subjects are not. Thus, MECHO will never use five equations where four are sufficient. However, as noted in IVA, MECHO can offer an explanation of precisely why a subject needed five equations when four would have been sufficient.

There have been two goals of this paper (1) to understand and elucidate the model-building activities at work when human subjects solve mechanics problems and (2) to describe a computer program that models closely aspects of that

performance. The goodness of fit of (1) and (2) rests in comparing the protocol of the former with the trace of the latter. Both trace and protocol are products or artifacts of the information processing systems that created them. We feel they offer important insights into the organisms they represent and their comparison offers deeper understanding of both.

ACKNOWLEDGMENTS

I would like to thank Alan Bundy and Lawrence Byrd, of the MECHO project, for useful criticism and assistance on preparing this paper. I would especially like to thank Richard Young for comments and encouragement on earlier drafts. Thanks also to the British Science Research Council for supporting this work.

REFERENCES

Bhaskar, R., & Simon, H. A. Problem solving in semantically rich domains: An example from engineering thermodynamics. *Cognitive Science*, 1977, *1*, 193–215.

Bundy, A. Will it reach the top? Prediction in the Mechanics World. *Artificial Intelligence*, 1978, *10*, 111–122.

Bundy, A., Luger, G., Mellish, C., & Palmer, M. Knowledge about knowledge: Making decisions in mechanics problem solving. *Proceedings of AISB/GI Conference*, 1978, 71–82.

Bundy, A., Byrd, L., Luger, G., Mellish, C., & Palmer, M. Solving mechanics problems using meta-level inference. *Proceedings of the IJCAI-79*, 1017–1027.

de Kleer, J. Qualitative and quantitative knowledge of classical mechanics. Technical Report AI-TR-352, MIT AI Lab, 1975.

Ernst, G., & Newell, A. *GPS: A case study in generality and problem solving*. New York: Academic Press, 1969.

Hinsley, D., Hayes, J., & Simon, H. From words to equations: Meaning and representation in algebra work problems. In P. A. Carpenter and M. A. Just (Eds.), *Cognitive processes in comprehension*. N.J.: Erlbaum, 1977.

Kowalski, R. *Logic for problem solving*. New York: North Holland, 1979.

Larkin, J. H. Skill acquisition for solving physics problems. C.I.P. Report 409, Department of Psychology, Carnegie-Mellon University, Pittsburgh, 1979.

Larkin, J. H., McDermott, J., Simon, D. P., & Simon, H. A. Expert and novice performance in solving physics problems. *Science*, June 20, 1980, *28*.

Luger, G. Problem solving in mechanics: Human protocols and the MECHO Trace. Working Paper 82, Department of Artificial Intelligence, University of Edinburgh, 1980.

Luger, G., & Bundy, A. Representing semantic information in pulley problems. In *Proceedings of 5th International Joint Conference on Artificial Intelligence*, II, 500, 1977.

Marples, D. *Argument and technique in the solution of problems in mechanics and electricity*. Department of Engineering, Cambridge (United Kingdom, CUED/C, EDUC/TRI), 1974.

McDermott, J., & Larkin, J. H. Re-representing textbook physics problems. In *Proceedings of the Canadian Society for Computational Studies of Intelligence*, 156–164, University of Toronto Press, 1978.

Newell, A., & Simon, H. GPS: A program that simulates human thought. In Feigenbaum & Feldman (Eds.), *Computers and Thought*. New York: McGraw-Hill, 1963.

Newell, A., & Simon, H. Human problem solving. N.J.: Prentice-Hall, 1972.

Novak, G. S. Computer understanding of physics problems stated in natural language. In *Proceedings of 5th International Joint Conference on Artificial Intelligence*, I, 286–291, 1977.

Palmer, A. & Snell, K. *Mechanics*. London: University of London Press, 1956.

Simon, D. P., & Simon, H. A. Individual differences in solving physics problems. In Siegler (Ed.), *Children's thinking: What develops*. N.J.: Erlbaum, 1978.

Warren, D., & Pereira, L. PROLOG—The language and its implementation compared with LISP. In *Proceedings of Symposium on Artificial Intelligence and Programming Languages*, (ACM) SIGPLAN Notices, 1978, *12*(8).

Young, R. *Seriation by children: An artificial intelligence account of a Piagetian task*. Basel, Switzerland: Birhauser, 1976.

e-mail

~~Sarah Mayer~~

Ben

4 weeks!

finish draft of paper
get out of office @ VN
get subs organized
do promotion screen

Do SNL stuff - Michael