Knowledge about Knowledge: Making Decisions in Mechanics Problem Solving

KNOWLEDGE ABOUT KNOWLEDGE: MAKING DECISIONS IN MECHANICS PROBLEM SOLVING

Alan Bundy, George Luger,
Chris Mellish, Martha Palmer
Department of Artificial Intelligence,
University of Edinburgh,
Edinburgh, UK.

1. Abstract

In October 1975 we started a three year SRC supported project on "A Program to Solve Mechanics Problems Stated in English". This project is now beginning its third year and many of its initial objectives have been achieved. In this paper we briefly review the progress that has been made and discuss some of the more promising areas of research that we have uncovered.

The structure of the paper is as follows:

2. The current state of the project

The original objective of our project is summed up in the following quotation from the 1975 grant application:

"Our objective then, is to write a program which can solve Mechanics problems, of the kind given as examples in Humphrey (1957) p1-90. We will concentrate on writing a program which can extract equations, given a surface level meaning representation obtained from the statement of the problem in English. If we make suitable progress on this, we would like to extend the program, so that it would accept problem statements in English."

This program, entitled MECHO, now solves a number of mechanics problems. The program centres around an algorithm for extracting equations based on Marples' (1974) study of Cambridge Engineering Students (for further details, see Bundy et al 1976a). This algorithm in turn calls on various inference rules to draw conclusions from the input meaning representation. The equations extracted can be fed to an equation solving program written by R.K.Welham.

As might be expected, our current descriptive theory differs substantially from our ideas in 1975. The three stages of meaning representation (surface, deep and deep with quantities) have been merged into one, which is gradually added to in the course of problem solving. We now make use of far more high-level descriptions than envisaged in 1975 and have single predicates and associated schemas (Frames) for whole pulley systems (Luger and Bundy 1977b) and for particles in motion (Bundy 1977a).

The PROLOG programming language (Warren 1977) was chosen for all our programs and has proved quite successful. It has provided the important tools of an assertional database, pattern-directed invocation and a search facility as well as extra-logical evaluable predicates to modify the simple backtracking control structure. Our experiences with PROLOG are recorded in Bundy 1976b and Bundy and Welham 1977d.

Our original approach to problems of search control consisted in carefully designing and debugging inference rules within the basic PROLOG inference system. As more problems were tackled, generalisations were noted and some of the control information was incorporated in the inference system. This is leading to the design of a computational logic for natural reasoning based on our experience of mechanics problem solving (see section 4).

It was usually possible to rely on our own intuitions and experience of mechanics problem solving to design inference rules and descriptive terms. Occasionally, however, it was useful to investigate disputes or illuminate difficulties by examining protocols of expert problem solvers. The results of these protocol analyses are described in Luger 1977a.

We have progressed further than originally intended with the extension to allowing English input to the program. Since we were unable to find a consensus on the kind of meaning representation that might be output by a natural language "front end", we were forced to develop the natural language processing and problem solving programs in parallel. Our first programs took as input a syntactic parse from Soul's (1975) existing parser and produced a suitable meaning representation (see Stone 1976). Later efforts to produce a unified program have introduced interesting issues of syntax-semantics interaction (Palmer 1977 and Mellish 1977) and demonstrated the value of studying natural language processing and problem solving in the same system (see section 5).

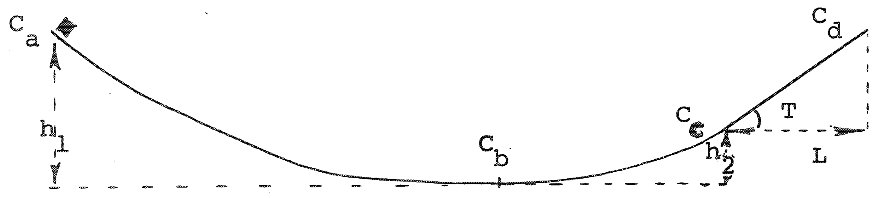
The three major problem areas that have been concentrated on so far are those involving pulleys, the motion of particles on complex paths and the motion of particles moving under constant acceleration. Each of these is described below with an indication of the representational issues involved and an account of one of the problems solved by the MECHO system. Most of these problems have been translated by hand into Predicate Calculus assertions suitable for input to the program, although some have been processed by the natural language programs.

A. Pulleys (from Humphrey, p75)

"Two particles of masses 6 and 10lbs are connected by a light string passing over a smooth pulley. Find their common acceleration and the tension in the string."

Pulley system problems were used to develop the descriptive theory especially the use of schemas to describe high-level objects like pulley systems. Also the problem above was the main working example for the natural language processing programs.

B. Motion of Particles on Smooth Complex Paths (de Kleer, 1975)



The particle starts from rest at C_a; given h_1, h_2, L and T will it reach point C_d?.

This problem was used to explore the representation of motion and the technique of hypotheses making and testing. These "roller coaster" problems rely heavily on diagrams for their description, and no attempt was made to input either diagrammatic or English descriptions of them.

C. Constant acceleration problems (Palmer and Snell, p20)

"The distance between two stations is 2000 yds. An electric train starts from rest at one station with a uniform acceleration of a1 ft/secs^2. It comes to rest at another station with a uniform retardation of a2 ft/sec^2. The speed for the intermediate portion of the journey is constant. Find the constant velocity if the journey is to be completed in three minutes."

These problems were used as a vehicle for exploring the representation of time. This example motivated a package for converting all units into a compatible set (e.g. yards and minutes above into feet and seconds). It also provided some ammunition for the natural language processing, i.e. how to parse "starts from rest" etc. recognizing the referent to "It" and associating 2000 yards with the path of the train.

3. Annotated Example

To give a clearer picture of how the system acts as a whole and how the individual parts operate, we will now briefly describe the steps it goes through for a concrete example. The example we have chosen is a simple pulley problem:

> Two particles of mass b & c are connected by a light string passing over a pulley. Find the acceleration of the particle of mass b.

The natural language analysis looks at each clause separately, interpreting it in two phases. As the clause is read, the syntactic structure is established and the objects referred to are identified. Then the meaning of the verb is consulted to determine how these objects are to be used to derive the meaning of the clause as a whole.

In the first clause, the word "two" starts the parsing of a noun phrase, the result being assigned the subject role because of its initial position in the sentence. Two new objects are created and predicated to be particles. The modifier "of mass b and c" is parsed, with a general-purpose routine distributing the masses - one to each particle. Then semantic routines scrutinise the particles to ensure their suitability to have the property of mass, creating new objects for the masses and ascribing them to the particles. The next major sentence constituent is the main verb group, which reveals the main verb "connect" as well as other information (for instance, that the sentence is in the passive voice). The preposition "by" initiates parsing of a prepositional phrase, and a new object, asserted to be a string, is created. This is asserted to have zero mass (it is "light") and takes the subject role in the reduced relative clause "passing over...". This second clause is now interpreted almost exactly as if it were the complete sentence "The string is passing over a smooth pulley", the main verb being "pass" and a pulley being created to fill the prepositional phrase slot. The full stop signals the end of all current clauses, which are now interpreted by semantic routines. These use syntactic roles like "logical subject" and "logical object" to access the appropriate participants in the relationships implied by the verb. The "pass" routine identifies its main participants as the string and the pulley and suggests contact between the midpoint of the string and the pulley (which is assumed to have no extent). Then the "connect" routine predicates contact between each particle and an appropriate contact point of the string, one particle being paired with each end. Notice that during this semantic analysis new objects, such as the ends of the string, have been created to satisfy the requirements of the meaning. Also the configuration is seen to satisfy sufficient constraints of the standard pulley system to cue in a pulley system schema.

The second sentence is seen to be an imperative when the first word is read, and the remaining work involves finding the logical object. The definite article in "the particle" reveals that it is referring to an object already known. Although there are two possible particles, there is only one known mass quantity with measure b and only the first particle has this mass. The definite article in "the acceleration" cannot be interpreted in the same sense as this, since the values of functions of objects (like accelerations) can be referred to by definite phrases without there being previous specific knowledge of them. However, in this case the acceleration has already been introduced by the pulley system schema, and so it is not necessary to create a new object. Finally the second sentence is interpreted as a whole. The meaning of "find" in the imperative involves marking certain quantities as "sought" for the equation solver. With the reaching of the end of the problem, all other quantities that have been introduced and specified can be marked "given".

The following clauses are the result of the natural language analysis:

```
isa(particle, p1), isa(particle, p2), isa(string, s1)
isa(pulley, pull), end(s1, end1, right), end(s1, end2, left)
fixed_contact(end1, p1, period1), fixed_contact(end2, p2, period1),
midpt(s1, midpt1), fixed_contact(midpt1, pull, period1)
mass(p1, mass1, period1), mass(p2, mass2, period1), mass(s1, zero,
                                                           period1)
coeff(pull, zero), measure(mass1, b), measure(mass2, c),
accel(p1, a1, 270, period1), period(period1),
sought(a1), given(mass1), given(mass2),
```

in addition the following schema is cued:

```
pullsys_stan(sys, pull, s1, p1, p2, period1).
```

The backward reasoning Marples Algorithm then attempts to find an equation that expresses the sought quantity, a1, in terms of the given quantities b, c, and g (the gravitational constant). a1 is investigated and information is extracted which focusses the Marples Algorithm and narrows the range of equations to be considered. In this case the only equations considered are those obtained by resolving forces about the particle p1 and p2. Only one of these is needed at this stage, namely the resolution of forces about p1:

$$- b.g + tension1 = b.a1$$

This solves for a1 but introduces a new unknown tension1, the tension in the string.

This tension was not mentioned in the original problem statement, but was introduced by the pulley system schema in the same way as the accelerations of the particles. Other information introduced by the pulley system schema includes, for instance, that the string is divided into two straight segments with inclinations of 90° and 270° respectively. Thus the schemas are used to fill in "gaps" between what is given in the problem statement and what is needed to solve the problem.

The other "gap filling" mechanism is the backwards inference process initiated by equation extraction. In this example backwards inference is responsible for deducing that the tension of each of the string segments is tension1 and for working out the combined forces acting on the particles.

The Marples Algorithm now tries to find an equation which expresses tension1 in terms of a1, b, c, and g. Attention is focussed on resolving forces about p1, p2 or the pulley. Resolving about p1 is rejected because that equation has already been used. Resolving about p2 is preferred over resolving about the pulley because it introduces no new unknowns. The resulting equation is:

c.g - tension1 = c.a1

Before being solved the equations are checked to see that they are all in the same units. Solution in this case is very simple and merely involves eliminating tension1 to give

a1 = g.(c-b)/(c+b)

4. Towards a Computational Logic for Natural Reasoning

One of the original aims of the project was to see how domain specific knowledge could be used to control inference in a semantically rich domain. Inference rules were designed with care, to avoid superfluous search wherever possible. Program traces were examined to spot false trails and inference rules debugged to avoid them in the future. Eventually these rules were riddled with ad hoc control information.

It then became clear that much of this control information could be generalized and that it would then cease to be domain specific and become applicable to any inferential system. The two specific generalizations that we describe below are: (i) the exploitation of function properties (Bundy 1977c) and (ii) the use of similarity classes (Bundy 1978). We hope that this gradual process of generalization will lead to a computational logic for natural reasoning. That is, an inferential mechanism containing built in control primitives which have already proved their usefulness.

The first generalization is the exploitation of function properties. One of the properties of functions is that they give unique values. This uniqueness can be used both (i) to trap unachievable goals before they are called and (ii) to prune subsequent search after a first value has been found. For instance, if it is known that $f(a) = b$ then it is not possible to prove that $f(a) = c$ where $b \neq c$. In many cases it is possible to trap the unachievable goal $f(a) = c$ before it is called. Similarly if the value of $f(a)$ is requested and shown to be b it is silly to request alternative values of $f(a)$ on backtracking and the possibility can be pruned from the search tree.

It might be felt that functions occur only in mathematical reasoning. In fact they are common in everyday reasoning but often go unnoticed. For instance:

At is a function from objects and times to locations
Final is a function from periods of time to their final moments
Motherof is a function from mammals to their mothers.

In our inference system it is only necessary to specify that a particular relation is a function from some arguments to others for the appropriate control regime to be brought to bear. A relation can even be a function in more than one way e.g. the relation Timesys between a period and its initial and final moments can be a function in three ways.

The second generalization is the use of similarity classes. This was a technique originally developed to control the equality relation. In fact, it is applicable to any relation with transitive, symmetric and reflexive properties and again this is a far wider class of relations than is generally appreciated. The idea is to put similar objects in classes with a distinguished element (the root), so that similarity between two objects can be tested by looking to see whether they are in the same class (i.e. point to the same root). This test is highly efficient compared to the infamous explosive search properties of the transitive and symmetric laws.

The standard way to represent similarity classes is by trees, with the root as distinguished element. Below are some sample trees for (i) equality, (ii) the sameplace relation (two objects are in the same place) and (iii) the factor relation (giving the conversion factor between two units of the same dimension).

(i) <u>equality</u>

$$\begin{tikzcd}
\text{Sandra's husband} \arrow[dr] & & \text{David's father} \arrow[dl] \\
& \text{Robert} &
\end{tikzcd}$$

(ii) <u>sameplace</u>

$$\begin{tikzcd}
\text{Mary} \arrow[dr] & & \text{John} \arrow[dl] \\
& \text{the lamppost} \arrow[d] & \underline{\text{last night}} \\
& \text{the corner} &
\end{tikzcd}$$

(iii) <u>factor</u>

$$\begin{tikzcd}
\text{inches} \arrow[dr, "12"] & & & \\
& \text{feet} \arrow[dr, "3"] & & \text{miles} \arrow[dl, "1/1760"'] \\
& & \text{yards} &
\end{tikzcd}$$

Slight modifications of the original techniques are needed for some relations. For instance, in the case of sameplace a different set of classes is needed for each time period or moment. For factor it is necessary to label the arcs with conversion factors. These need to be multiplied together as the arcs are traversed, to find, say, the conversion factor between inches and miles.

The MECHO program contains a generalization of the similarity class machinery and appropriate relations can be defined simply in terms of it.

5. Investigations of syntax-semantics interaction

In the area of natural language processing, one of our main aims is to study the process-control issues of efficient natural language parsing - the type of decisions that have to be made and the ways that syntactic and semantic processes can interact to best advantage. It is the process of reference evaluation that has concerned us mainly.

Winograd (1972) has shown that evaluating references early in the parsing can be useful in resolving syntactic ambiguity. Thus in the sentence:

Put the pyramid on the block in the box,

which is apparently ambiguous, only one reading is acceptable in a context where there are no pyramids on blocks. The absence of a referent to 'the pyramid on the block' is used to ensure that another interpretation is taken.

However, there is not always enough information in a noun phrase to determine the referent uniquely (for instance, if it is a pronoun, this is rarely so). Thus Winograd was forced into adopting discourse heuristics to make premature decisions (for most definite noun phrases) or carrying forward all possibilities in the hope that the rather crude semantic marker lists in the verb meaning will eliminate some (in the case of pronouns).

The system that we are investigating involves considering reference evaluation as an incremental process, with references becoming progressively more instantiated as the analysis proceeds. In this, pronouns are treated in the same way as other definite noun phrases, as the same mechanism is used to express both definitional information and 'semantic checks'. Both types of semantic constraints are interpreted as filters on possible referents and may be enforced at any time in the analysis, with the result that there can be a close interaction between syntactic and semantic processes. Reference evaluation still provides a check on syntactic hypotheses because a filtering process which has eliminated all the possibilities for a referent causes the abandonment of the current hypotheses. Moreover, in this framework, there is much less risk of carrying out extensive syntactic manipulations on the basis of faulty reference evaluations.

An important feature of the system developed is that all semantic tests are in terms of actual referents and not in terms of the structure of referring phrases. Thus we are able to express a much wider range of constraints than can be captured by the use of traditional semantic markers of the Katz and Fodor (1964) type. Those of the simplest type involve single references and express ideas like 'an object cannot be a part of itself'. More complicated constraints impose dependencies between the possible values of several references. Thus the constraint that two objects be in contact causes interactions between the two evaluation processes - further information relevant to one may have repercussions for the value of the other. A great deal of knowledge of possible physical configurations cannot be expressed with a mechanism like semantic markers but requires something of this type. An investigation into how to make the best use of this powerful system will be an important further step in our work on mechanics problems.

As a simple example of how this kind of knowledge can help resolve ambiguity, consider the following problem:

> "AB is a uniform rod, of length 8a, which can turn freely about the end A, which is fixed;
> C is a smooth ring, whose weight is twice that of the rod, which can slide on the rod and..."

In this example, there is apparent ambiguity as to whether the final relative clause ("which can...") should be attached to "a smooth ring" or to "the rod". A human being who reads this sentence and attempts to visualise the scene has no difficulty with this problem. In order to resolve it mechanically, it suffices to notice the fact that an object cannot slide on itself. This information ensures that the phrase cannot be attached to the rod and thus must qualify the ring. However, the restriction needed (the irreflexivity of a particular sliding relation) cannot be expressed in terms of semantic markers.

There is a danger that the number and complexity of such semantic restrictions could be completely unmanageable, but several factors combine to ease this problem. Firstly, there is no reason why restrictions should have to be immediately verifiable - in general, deductions may be necessary to satisfy them. This means that similar restrictions can share pieces of deductive machinery; moreover in some cases complex restrictions can be pieced together from more primitive tests. Secondly, many restrictions express mathematical properties of particular relations and so are generalizable. These involve functional properties between relation arguments (c.f. section 4) and properties like symmetry and transitivity (e.g., the relation of 'support' is asymmetric and transitive). It is interesting that these are the kinds of properties that we are exploiting in the quest for general mechanisms of search control. There seems to be a close similarity between controlling deduction by pruning inappropriate goals from the search tree and resolving ambiguity by avoiding consideration of semantically inappropriate hypotheses. Thus, in particular, the mechanisms we have developed to exploit function properties and similarity classes for search control may well be of direct use to the natural language programs. Conversely, mechanisms that have suggested themselves for the natural language tasks (such as the particular use of irreflexivity shown above) are likely to provide ideas for general search control techniques.

Conclusion

We hope that this paper has given an understandable account of the current state of the MECHO project and some of the more interesting issues that we are following up. Although our work covers a range of subjects that are not commonly studied in close conjunction, there is an underlying theme running through the whole project. This is the idea that only through carefully establishing exactly where and how individual pieces of knowledge should be used to best effect can one construct a system which performs an intellectual task of any complexity. Even in the relatively simple domain of mechanics problem solving, extremely sophisticated techniques have to be introduced in order to come close to the expert human's ability to make decisions and discard false trails and faulty hypotheses. It is our belief that a significant number of the ad-hoc solutions produced for specific problems can be generalized and that there are useful domain-independent principles of search and process control to be found. One of the most promising sources of relevant generalizations seems to involve the province of second-order relations like functionality, transitivity and irreflexivity and it is in this direction that our investigations are currently proceeding.

References

Bundy, A., Luger, G. and Stone, M., 1975. "A Program to Solve Mechanics Problems Stated in English", SRC Grant Application.

Bundy, A., Luger, G., Stone, M. and Welham, R., 1976a. "MECHO: Year One", Procs. of the 2nd AISB Conference, ed. Brady,M., Edinburgh, p94-103, also DAI Research Report No.22.

Bundy, A. 1976b. "My Experiences with PROLOG", DAI Working Paper No. 12.

Bundy, A. 1976c. "The Role of Inference in the Solving of Mechanics Problems", DAI Working Paper No. 16.

Bundy, A. 1977a. "Will it reach the top? Prediction in the Mechanics World", Artificial Intelligence Journal (forthcoming), also DAI Research Report No. 31.

Bundy, A. 1977b. "Can Domain Specific Knowledge be Generalized?", (Short Note), p496, Procs. of the 5th IJCAI, ed. Reddy,R., Cambridge, Mass.

Bundy, A. 1977c. "Exploiting the Properties of Functions to Control Search", DAI Research Report No. 45.

Bundy, A. and Welham, R.K. 1977d. "Utility Procedures in PRO-LOG", DAI Occasional Paper No. 9.

Bundy, A. 1978. "Similarity Classes", DAI Working Paper No.25.

de Kleer, J. 1975. Qualitative and quantitative knowledge in classical mechanics, MIT AI Lab. Report AI-TR-352, Cambridge, Mass.

Humphrey, D. 1957. "Intermediate Mechanics, Dynamics", Longman, Green & Co., London.

Katz, J.J. and Fodor, J.A. 1964. "The Structure of a Semantic Theory" in J.A. Fodor and J.J. Katz (Eds.) "The Structure of Language", Engelwood Cliffs, New Jersey: Prentice Hall.

Luger, G. 1977a. "The Use of Protocols and the Representation of Semantic Information in Pulley Problems", DAI Research Report No. 36.

Luger, G. and Bundy, A. 1977b. "Representing Semantic Information in Pulley Problems", p500 (Short Note), Procs. of the 5th IJCAI, ed. Reddy, R., Cambridge, Mass.

Marples, D. 1974. "Argument and technique in the solution of problems in mechanics and electricity, CUED/C - Educ/TRI, Dept. of Engineering Memo, University of Cambridge.

Mellish, C. 1977. "Preliminary Syntactic Analysis and Interpretation of Mechanics Problems Stated in English", DAI Working Paper.

Palmer, A.H.G. and Snell, K.S. 1956. "Mechanics", Univ. of London Press Ltd., London.

Palmer, M. 1977. "Where to Connect? - Solving Problems in Semantics", DAI Working Paper No. 22.

Soul, M. "Parsing and reference determination", unpublished Ph.D. Thesis, Essex.

Stone, M. 1976. "PAT - Pulleys and Things", DAI Working Paper No. 18.

Warren, D. 1977. "Implementing PROLOG - compiling predicate logic programs", Vol. 1 and Vol. 2, DAI Research Report No. 39 and 40, Edinburgh.

Winograd, T. 1972. "Understanding Natural Language", Academic Press.