UNDERSTANDING AND SOLVING ARITHMETIC WORD PROBLEMS:

A COMPUTER SIMULATION

CHARLES R. FLETCHER

UNIVERSITY OF MINNESOTA

Technical Report No. 135

Institute of Cognitive Science
University of Colorado
Boulder, Colorado 80309

December, 1984

Understanding and Solving Arithmetic Word Problems: A Computer Simulation

Abstract

This paper describes a computer program called WORDPRO which simulates the psychological processes involved when third grade children understand and solve simple arithmetic word problems. Both the implementation of the program and its performance on a set of sample problems are presented. WORDPRO is a useful research tool in that it demonstrates the sufficiency of the theory upon which it is based, assists in communicating that theory to other researchers and provides a source of empirical predictions for experimental tests of the theory.

Understanding and Solving Arithmetic Word Problems: A Computer Simulation

WORDPRO is a computer program, written in Interlisp-D, which implements a theory of the comprehension and solution of simple arithmetic word problems which has been proposed by Kintsch and Greeno (1985). The goals of the program are to demonstrate the sufficiency of the theory, to assist in communicating it to other researchers, and to serve as a tool for exploring the consequences of the theory. This paper concentrates on the first two of these. It describes the behavior of WORDPRO on a set of problems and provides enough details about its implementation to allow other researchers who wish to obtain a copy of the program[1] to understand its output and (along with the internal documentation) test and modify it to their own needs.

The Domain of WORDPRO

WORDPRO is designed to simulate the comprehension and solution of a restricted set of word problems by third-grade children. Examples of the problems that make up this set are shown in Table 1. Each can be solved by a single addition or subtraction operation.

Insert Table 1 about here

WORDPRO does not currently accept as input the surface representation of a text. Instead it begins with a set of propositions which represent the text's meaning (see e.g., Turner & Greene, 1978; Bovair & Kieras, 1984). As an example, Table 2 shows the propositional representation of the first text from Table 1.

Insert Table 2 about here

Knowledge Structures

Given the propositional representation of a text, WORDPRO constructs a bi-level representation which it then uses to derive the solution. The two levels are referred to as the text base and the problem model. The text base is an organized set of propositions. Such a representation is assumed to result from reading or listening to any text, regardless of the domain. The problem model is a non-propositional, domain-specific representation which drives the problem solving process (van Dijk & Kintsch, 1983; Ch. 10).

Knowledge of sets is assumed to underly construction of both the text base and the problem model. This knowledge is captured by a set schema with three slots: an object slot which holds the intension of a set (e.g., marbles), a quantity slot which contains its cardinality (3), and a specification slot which holds information that distinguishes one set from another. At the present time WORDPRO's set schemata include specification sub-slots for owner (Joe) and time (past). Others, such as location, could be easily added. In the text base the slots of the set schema are filled with propositions. In the problem model, all information not essential to a slot is stripped away. As an example, if the object slot in the text base were filled with the proposition (P4 (3 MARBLE)), the same slot in the problem model would contain simply MARBLE.

Three higher-level schemata are used to organize set schemata into coherent problem representations. A transfer schema captures the relationship between a start-set, a transfer-set, and a result-set. A

superset schema is used to organize a superset and two subsets. Finally, a more-than (or less-than) schema describes the relationship between a small-set, a large-set, and a difference-set.

WORDPRO's schemata and the procedures that store and retrieve information from them are adapted from Winston and Horn (1981; Ch. 22).

Flow of Control

The basic structure of WORDPRO is shown in Figure 1. The heart of the system is an ordered set of production rules (Newell, 1973). Each rule consists of a set of conditions which must be met for the rule to be applied, and a set of actions which are then carried out. Each condition is a test on the current contents of short-term memory (STM). The actions, in turn, alter the state of STM by adding new information, reinstating old information from long-term memory (LTM), or deleting information. We thus assume that the set of production rules controls the flow of information through STM. The rules are checked against the contents of STM in order. When more than one rule matches the contents of STM, only the first one encountered is applied. The program is designed so that the user can watch STM change rule by rule.

Insert Figure 1 about here

As Figure 1 also shows, the production rules are applied within a simple read-purge loop. This gives WORDPRO the cyclical character necessary in a model of text comprehension (Kintsch & van Dijk, 1978; Fletcher, 1981). During the read portion of the loop, those propositions corresponding to the next sentence are added to STM. The production rules then operate on the contents of STM until no rules apply. At this point STM is purged of all but one to three items and the cycle begins again. This prevents STM from

becoming overloaded but necessitates occasional searches of LTM for purged information. Two types of information are not purged: the most recently constructed set schema or higher level schema and requests by a higher level schema for set schemata to take up unfilled slots.

WORDPRO's Production Rules

WORDPRO's production rules fall into three categories: meaning postulates, arithmetic strategies and problem solving procedures. The first thirteen rules are the meaning postulates. Each of these is triggered by the occurrence of a proposition with a particular predicate (first term) which has not been added to the text base. A postulate's primary action is to add that proposition to the text base and corresponding information to the problem model. Two additional actions are carried out by subsets of these rules. Quantifying propositions (i.e., those with SOME, HOWMANY, or a number as their predicate) cause a new set schema to be added to STM. Propositions with GIVE, HAVE-ALTOGETHER, HAVE-MORE-THAN or HAVE-LESS-THAN as their predicate initiate searches of LTM under some conditions. English approximations of some representative meaning postulates are given in Table 3.

Insert Table 3 about here

The arithmetic strategies construct higher-level schemata. Their actions include adding new schemata to STM, filling the slots of those schemata, and adding requests to STM. Each request is for a set schema with a given specification which is needed to complete a higher-level schema. The conditions which trigger these rules are either set schemata with

certain specifications, or the simultaneous occurrence of requests and set schemata which satisfy them. WORDPRO includes twelve arithmetic strategies, two representative examples are shown in Table 4.

Insert Table 4 about here

The final eleven production rules comprise the problem solving procedures. Each of these rules is triggered by a completed higher-level schema. There are two types of problem solving procedures. Some derive solutions by performing a sequence of operations on mental blocks. The allowable operations include adding or deleting blocks from STM, counting blocks, matching blocks and moving them from one mental location to another. The remaining rules do not actually find solutions, rather they convert one type of higher-level schema into another (e.g., a transfer schema might be changed into a superset schema). Table 5 lists some representative problem solving procedures.

Insert Table 5 about here

The procedures which test each production rule's conditions and apply its actions (when appropriate) are modifications of those described in Winston and Horn (1981; Ch. 18).

An Example

As an example of how the program works, consider Table 6 which shows the contents of STM as WORDPRO begins to process the final sentence of the change-1 problem shown in Tables 1 and 2. At this point STM contains the propositions from the final sentence plus two chunks from earlier in the text, a transfer schema and a request for a result set. From here

processing proceeds as follows. First, three meaning postulates are triggered which build a new set schema, SET3. The first of these is triggered by (P10 (HOWMANY MARBLE)). This rule adds the new schema to STM and fills in its object and quantity slots. Next, (P11 (HAVE X P10)) activates a rule which adds owner information to the specification slot. Finally, (P12 (NOW P11)) completes the schema by filling the time sub-slot of the specification. This completed set schema matches the requirements of the request still active in STM. Thus, an arithmetic strategy is triggered which adds SET3 to the result slot of the transfer schema. This completes the transfer schema, as shown in Table 7, and evokes one of the problem solving strategies. This rule arrives at the correct solution to the problem by adding two sets of mental blocks to STM, one corresponding to the start-set and the other to the transfer-in set, and then counting the total number of blocks.

Insert Tables 6 and 7 about here

In the tables, as in the program itself, each schema is represented as a list. The first list element identifies the schema. The remaining elements are themselves lists which identify each slot and the information that fills it. When that information is itself a schema, it is represented in the same way. Each request is represented as a list with four elements. The first of these identifies it as a request, the second identifies the higher-level schema for which the request is being made, the third identifies the slot that needs to be filled, and the last contains information which a set schema's specification must contain for it to fill the request. Propositions are represented as in Table 2.

Evaluation and Future Directions

WORDPRO can be considered a success for a number of reasons. First of all, it successfully solves all fourteen problem types for which it was designed. Secondly, it makes the theory underlying it easier to understand by allowing one to watch it work. Finally, it allows us to test this theory by comparing the program's performance to that of human subjects. Run-time statistics such as those shown in Table 8 form the basis for this comparison. Each of the operations in the table is assumed to be resource consuming and should increase the time required to read and solve a problem while reducing the probability of a correct answer.

Insert Table 8 about here

Plans are now underway to expand WORDPRO in a number of directions. The first is to enlarge the domain of problems that the program can handle. The second is to adapt it to different levels of ability. The last is to interface WORDPRO with a parser which will allow it to accept the surface representation of a text as input.

Researchers interested in exploring WORDPRO are encouraged to acquire a copy of the program. It is hoped that this paper describes it in sufficient detail to make both the output and the code itself understandable.

References

Bovair, S. & Kieras, D.E. (1984). A guide to propositional analysis for research on technical prose. In B.K. Britton & J.B. Black (Eds.) Understanding expository text. Hillsdale, N.J.: Erlbaum.

Fletcher, C.R. (1981). Short-term memory processes in text comprehension. Journal of Verbal Learning and Verbal Behavior, 20, 564-574.

Kintsch, W. & Greeno, J.G. (1985). Understanding and solving word arithmetic problems. Psychological Review, 92, 109-129.

Newell, A. (1973). Production systems: Models of control structure. In W.G. Chase (Ed.), Visual information processing (pp. 463-526). New York: Academic Press.

Turner, A. & Greene, E. (1978). Construction and use of a propositional text base. JSAS catalogue of selected documents in psychology, MS 1713.

van Dijk, T.A. & Kintsch, W. (1983). Strategies of discourse comprehension. New York: Academic Press.

Winston, P.H. & Horn, B.K.P. (1981). LISP. Reading MA: Addison-Wesley.

Author Notes

This work was supported by National Science Foundation Grant No. BNS-8309075 to W. Kintsch and J.G. Greeno. I am grateful to Walter Kintsch, Denise Dellarosa, Suzanne Mannes and Bill Fox for their comments on an earlier draft of this paper. Reprint requests should be addressed to Charles R. Fletcher, Department of Psychology, University of Minnesota, 75 East River Road, Minneapolis, MN 55455.

Footnote

[1]Copies of WORDPRO are available in hardcopy or on diskette by writing to Walter Kintsch, Department of Psychology, University of Colorado, Boulder, CO 80309. The diskette contains two files; WORDPR.LSP which contains the program and TEXTS.LSP which contains some sample texts. To use the program one need only load these two files and type (Solve name), where name indicates the text to be solved, then follow the instructions on the screen. The program is written in Interlisp-D and will run on any XEROX lisp machine.

Table 1

Examples of the problem types handled by WORDPRO.

CHANGE: Result Unknown

1. Joe had three marbles. Then Tom gave him five marbles. How many marbles does Joe have now?
2. Fred had five boxes. Then he gave two boxes to Susan. How many boxes does Fred have now?

Change Unknown

3. Mary had four pencils. Then Sam gave her some pencils. Now Mary has nine pencils. How many pencils did Sam give Mary?
4. Betty had eleven puppies. Then she gave Debby some puppies. Now Betty has four puppies. How many puppies did Betty give Debby?

Start Unknown

5. Ed had some fish. Then Alan gave him eight fish. Now Ed has thirteen fish. How many fish did Ed have in the beginning?
6. Bob had some hats. Then he gave four hats to Danny. Now Bob has eight hats. How many hats did Bob have in the beginning?

COMBINE: Superset Unknown

1. Lucy has two dimes. Sarah has six dimes. How many dimes do they have altogether?

Subset Unknown

2. Larry and Sally have twelve kittens altogether. Larry has three kittens. How many kittens does Sally have?

COMPARE: Difference Unknown

1. Dan has six books. Jill has two books. How many books does Dan have more than Jill?

2. Dennis has eleven apples. Fred has six apples. How many apples does Fred have less than Dennis?

Compared Quantity Unknown

3. Liz has four dollars. Sarah has three dollars more than Liz. How many dollars does Sarah have?
4. Anne has nine flowers. Kathy has five flowers less than Anne. How many flowers does Kathy have?

Referent Unknown

5. Larry has seven oranges. He has three oranges more than Jeff. How many oranges does Jeff have?
6. Alice has six horses. She has five horses less than Doris. How many horses does Doris have?

Table 2

Propositional representation of the Change-1 text from Table 1.

```
(Change-1 (((P1 (EQUAL X JOE))
           (P2 (PAST P3))
           (P3 (HAVE X P4))
           (P4 (3 MARBLE)))
          ((P5 (THEN P3 P7))
           (P6 (EQUAL Y TOM))
           (P7 (GIVE Y X P9))
           (P8 (PAST P7))
           (P9 (5 MARBLE)))
          ((P10 (HOWMANY MARBLE))
           (P11 (HAVE X P10))
           (P12 (NOW P11)))))
```

Table 3

English approximations of two representative meaning postulates.

Example 1

CONDITIONS: 1. Does STM contain a proposition?

2. Does that proposition have the predicate HOWMANY?

ACTIONS: 1. Add a new set schema to STM.

2. Put the proposition in the object slot of the text base.

3. Put the second term of the proposition in the object slot of the problem model.

4. Put GOAL in the quantity slot of the problem model.

Example 2

CONDITIONS: 1. Does STM contain a proposition?

2. Does that proposition have the predicate NOW?

ACTIONS: 1. Put the proposition in the specification slot of the text base.

2. Put (TIME:PRESENT) in the specification slot of the problem model.

Table 4

English approximations of two representative arithmetic strategies.

Example 1

CONDITIONS:
1. Does STM contain a set schema?
2. Does the text base of the set schema contain a proposition with GIVE in its specification?
3. Does STM contain a second set schema with the same owner as the first one?

ACTIONS:
1. Add a new transfer schema to STM.
2. Put the first set schema in the transfer-in slot.
3. Put the second set schema in the start slot.
4. Add to STM a request for a result set with the same owner as the first two sets.

Example 2

CONDITIONS:
1. Does STM contain a request?
2. Is the request being made by a transfer schema?
3. Is a result-set being requested?
4. Does STM contain a set schema whose specification matches the request?

ACTIONS:
1. Put the set schema in the result slot of the transfer schema.
2. Remove the request from STM.

Table 5

English approximations of two representative problem solving procedures.

Example 1

CONDITIONS:
1. Does STM contain a transfer schema?
2. Does the start-set of the transfer schema have GOAL as its quantity?
3. Does the transfer-in-set have a number as its quantity?
4. Does the result-set have a number as its quantity?

ACTIONS:
1. Change the transfer schema to a superset schema.
2. Put the start-set in one subset slot.
3. Put the transfer-in set in the other subset slot.
4. Put the result-set in the superset slot.

Example 2

CONDITIONS:
1. Does STM contain a transfer schema?
2. Does the start-set of the transfer schema have a number (X) as its quantity?
3. Does the transfer-in-set have a number (Y) as its quantity?
4. Does the result-set have GOAL as its quantity?

ACTIONS:
1. Add X blocks to STM.
2. Add Y blocks to STM.
3. Count the blocks in STM.

Table 6

The contents of STM at the beginning of the third processing cycle for the change-1 problem. At this point STM contains five chunks of information.

PROPOSITIONS:

1. (P10 (HOWMANY MARBLE))
2. (P11 (HAVE X P10))
3. (P12 (NOW P11))

SCHEMATA:

```
4. (TRANSFER-SET
       (TRANSFER-IN
          (SET 2 (MODEL (QUANTITY (5))
                        (OBJECT (MARBLE))
                        (SPECIFICATION ((OWNER:JOE)
                                        (TIME:PAST)
                                        (TIME:AFTER P3))))
                 (T-BASE (OBJECT ((P9 (5 MARBLE))))
                         (SPECIFICATION ((P7 (GIVE Y X P9))
                                         (P6 (EQUAL Y TOM))
                                         (P8 (PAST P7))
                                         (P5 (THEN P3
                                                   P7)))))))
       (START
           (SET1 (MODEL (QUANTITY (3))
                        (OBJECT (MARBLE))
                        (SPECIFICATION ((OWNER:JOE)
                                        (TIME:PAST))))
```

```
(T-BASE (OBJECT ((P4 (3 MARBLE))))
        (SPECIFICATION ((P3 (HAVE X P4))
                        (P1 (EQUAL X JOE))
                        (P2 (PAST P3))))))))
```

REQUESTS:

5. (REQUEST TRANSFER-SET RESULT (OWNER:JOE))

Table 7

The contents of STM before the final production rule is applied to find the solution of the Change-1 problem. At this point STM contains a single chunk, the transfer schema.

SCHEMATA:

```
1.  (TRANSFER-SET
       (TRANSFER-IN
          (SET 2 (MODEL (QUANTITY (5))
                        (OBJECT (MARBLE))
                        (SPECIFICATION ((OWNER:JOE)
                                        (TIME:PAST)
                                        (TIME:AFTER P3))))
                 (T-BASE (OBJECT ((P9 (5 MARBLE))))
                         (SPECIFICATION ((P7 (GIVE Y X P9))
                                         (P6 (EQUAL Y TOM))
                                         (P8 (PAST P7))
                                         (P5 (THEN P3
                                                   P7)))))))
       (START
          (SET1 (MODEL (QUANTITY (3))
                       (OBJECT (MARBLE))
                       (SPECIFICATION ((OWNER:JOE)
                                       (TIME:PAST))))
                (T-BASE (OBJECT ((P4 (3 MARBLE))))
                        (SPECIFICATION ((P3 (HAVE X P4))
                                        (P1 (EQUAL X JOE))
                                        (P2 (PAST P3)))))))
```

```
(RESULT
   (SET3 (MODEL (QUANTITY (GOAL))
                (OBJECT (MARBLE))
                (SPECIFICATION ((OWNER:JOE)
                                (TIME:PRESENT))))
         (T-BASE (OBJECT ((P10 (HOWMANY MARBLE))))
                 (SPECIFICATION ((P11 (HAVE X P10))
                                 (P12 (NOW P11)))))))))
```

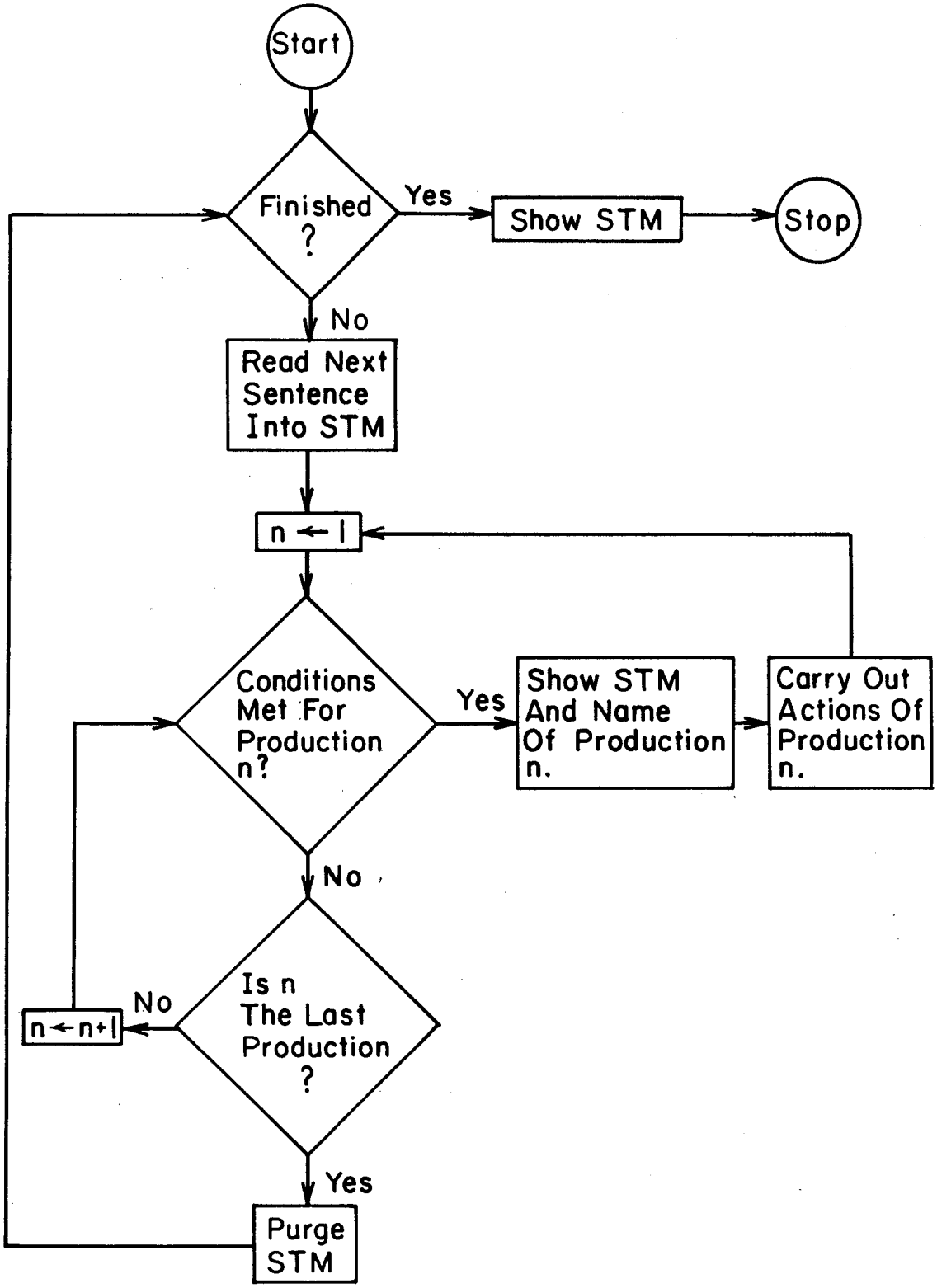
Table 8

Number of production rules fired, number of conversions from one schema to another, number of LTM searches and maximum number of chunks held over from one processing cycle to the next for each of the problems from Table 1.

TEXT	RULES FIRED	CONVERSIONS	LTM SEARCHES	MAXIMUM CHUNKS HELD
Change-1	15	0	0	
2Change-2	15	0	0	2
Change-3	19	0	0	2
Change-4	19	0	0	2
Change-5	20	1	0	2
Change-6	20	1	0	2
Combine-1	12	0	1	3
Combine-2	12	0	0	2
Compare-1	12	0	1	3
Compare-2	12	0	1	3
Compare-3	13	1	0	2
Compare-4	13	1	0	2
Compare-5	13	1	0	2
Compare-6	13	1	0	2

Figure Caption

Figure 1. The basic control structure of WORDPRO.


Start
Finished ?
Yes
Show STM
Stop
No
Read Next Sentence Into STM
n ← 1
Conditions Met For Production n?
Yes
Show STM And Name Of Production n.
Carry Out Actions Of Production n.
No
Is n The Last Production ?
No
n ← n+1
Yes
Purge STM