MS. MALAPROP, A LANGUAGE COMPREHENSION PROGRAM

Eugene Charniak
Institute for Semantic and Cognitive Studies
University of Geneva, Switzerland

Abstract

This paper describes Ms. Malaprop, a program (currently being designed) which will answer questions about simple stories dealing with painting, where stories, questions and answers will be expressed in semantic representation rather than English in order to allow concentration on the inferential problems involved in language comprehension. The common sense knowledge needed to accomplish the task is provided by the frame representation of "mundane" painting found in Charniak (1976b). The present paper, after reviewing this representation, goes on to describe how it is used by Ms. Malaprop. Some specific questions of matching, correcting false conclusions, and search, will be discussed.

Descriptive Terms

Computer comprehension of language, representation of knowledge, semantic representation, read time inferencing, frame representation, question answering, common sense knowledge.

I. Introduction

For the purpose of this paper, I take language comprehension to be the process of fitting what one is told into the framework established by what one already knows. So, to take a simple example,

(1) Jack was going to paint a chair. He started to clean it.

our understanding of the second line of (1) is conditioned by two facts: the first is the story specific information provided by the first line, i.e., that Jack has the intention of painting the chair, while the second comes from our general fund of common sense knowledge and states that it is a good idea if the thing to be painted is clean before one starts. By tying the second line to such information a person, or computer, would "know" such related facts as why the action was performed, what might have happened if it hadn't been, and how far along Jack is in the process of painting the chair.

Ms. Malaprop is a computer program (currently being designed) which will answer questions about examples such as (1). Indeed (1) is a typical example in many respects. For one thing it stays quite close to our knowledge of everyday events. As such the story specific information serves only to tell the program which parts of its real world knowledge are relevant to the story situation; the story does not build up a complex setting of its own. Hence when Ms. Malaprop fits new story information into what she knows it is always by relating it to her store of common sense knowledge, and never by seeing how it relates to some complex plot supplied in the story. This is obviously unrealistic as far as stories go, but it is all too realistic given current understanding of language comprehension.

This example is also typical insofar as once we have seen the second line as an instance of a certain portion of the painting process, the typical questions one might ask to demonstrate understanding, such as "why" or "what would have happened if he hadn't", should not be too difficult to answer. Hence I shall ignore the problem of how questions actually get answered in order to concentrate on the problems of the initial integration which I will assume occurs at "read time" rather than "question time". (For discussion of this assumption, see Charniak (1976a).)

No example is completely typical however and one thing (1) does not indicate is that Ms. Malaprop, at least in her early versions, will not understand English, but rather will be given stories and questions already in semantic representation. This representation has been almost entirely designed (see Charniak (1976b)) but, except in those places where it is the topic of discussion, it will be replaced by English phrases throughout.

Also, while many of the examples which are being used to define Ms. Malaprop's capabilities are like (1) in that they call for the program to tell you what in some sense it already knows, other examples are considerably more complex. For example : "After Jack finished he did not wash the paint brush. He was going to throw it away. Question : Why didn't Jack wash the brush ?"

I should note that the foreseen first version of the program will handle all of the painting examples herein, given the caveat, repeated here for the last time, that Ms. Malaprop cannot handle actual English.

II. The Framed Painting

Evidently, a program which answers such questions will have to have at its disposal quite a bit of information about painting and its neighbouring concepts. This knowledge base is completely designed and is described in detail in Charniak (1976b). We can only give a brief overview here, but it should be stressed that this representation is a) completely formalized, b) fairly complete, and c) fairly deep. By this last comment I mean that I have striven to hook up the representation of painting knowledge to more basic knowledge whenever possible. So, the representation "knows" why one should wash a paint brush after use because it knows about what happens when paint dries on something. But this latter is based on its knowledge of the evaporation of liquids containing residues, which in turn is based on its knowledge of evaporation in general. Almost nothing of these properties can be presented here, and the interested reader is encouraged to consult the aforementioned article.

Let us start by considering a *yery* simplified version of the painting "frame" (term due to Minsky (1975)), expressed mostly in informal English, but with some formalism thrown in.

```
PAINTING (COMPLEX EVENT)
  VARS:
     (AGENT must be animate)
     (OBJECT must be a solid)
     (PAINT must be a liquid, usually is paint)
     (INSTRUMENT must be a solid, usually is
                 either a roller or a paint brush,
                 and should be absorbant)

GOAL:
  PAINTING-GOAL (OBJECT has a coat of PAINT on it)
    COMES-FROM:
       (PAINTING6 via rules which say that
        paint on INSTRUMENT will stick to OBJECT,
        partially fulfilling the goal)
EVENT:
 fPAINTINGI (OBJECT not dirty)
    COMES-FROM: (WASH-GOAL)
    LEADS-TO: (NOT DIRTY-0BJECT1)
 2PAINTING2 (nearby things covered with newspaper)
 ?PAINTING3
    (LOOP
     rPAINTING4 (get PAINT on INSTRUMENT)
        COMES-FROM:
           (rules which explain how immersing
           INSTRUMENT in PAINT will give the
     I                               desired result)
     ?PAINTING5 (GREATER DRIP-THRESHOLD than the
                amount of PAINT on INSTRUMENT)
        COMES-FROM:
           (rules showing how the regulation of
           pressure regulates the amount of PAINT)
     2PAINTING6 (INSTRUMENT is in contact
                                      with OBJECT)
     M>AINTING7 (GREATER amount of PAINT on
                INSTRUMENT than the
                                 STREAK-THRESHOLD)
     )
 PAAINTING8 (PAINT removed from INSTRUMENT)
    LEADS-TO:
      (rules expressing how if it were not
      removed INSTRUMENT would stiffen)
```

Approaching this in steps, we first note that it is divided into three sections. The first, labeled VARS, is simply a list of variables along with some specification of what sorts of things may be bound to these variables. Then comes the GOAL, which expresses the goal of the activity. Finally we have EVENT which is a description of what sorts of things have to be done in order to accomplish the goal. The arrows are to indicate rough time ordering.

Going down one level of detail we notice that the EVENT is made up of a series of "frame statements", each of which has a name, PAINTING1, etc., which is followed by an expression in parentheses. Here these are informal English like statements, but in the complete version they are simply predicate plus argument structures. Some of these have extra information following them (labeled COMES-FROM and LEADS-TO) but let us ignore these for the time being. If we just look at the frame statements in EVENT we see that they give an outline of how to paint. One portion of this outline is a LOOP (PAINTING3) which tells us to get paint on the instrument (PAINTING4), bring the instrument in contact with the object (PAINTING6), while at the same time keeping the volume of paint above the streak-threshold (PAINTING7), and below the drip-threshold (PAINTING5). Shifting our attention to the GOAL, we see that it too is a frame statement (named PAINTING-GOAL), and had we specified the variable restrictions in VARS more fully, we would have seen them to be frame statements also.

If we look now at a single frame statement, say PAINTING1, we see that it has various "tags". One of these expresses how the state described by the frame statement normally is achieved (or equivalently how it "comes about", or where it COMES-FROM) while the other gives the reason for doing this portion of the frame (or the results of the frame statement, or what it LEADS-TO). So PAINTING1 is brought about by the WASH frame (left to the reader's imagination). This is expressed by saying that PAINTING1 <u>matches</u> (in the normal pattern matching sense) the goal statement of the WASH frame, namely WASH-GOAL. In much the same way there is a COMES-FROM pointer from PAINTING-GOAL to PAINTING6 which states how it is that the goal is brought about by PAINTING6. Note that in this case PAINTING-GOAL does not match PAINTING6, so in a complete version there would be "intermediaries" or rules which, from a syntactic point of view explain how the two statements can be made to match, while from a semantic point of view they explain how it is that bringing the instrument in contact with the object can ultimately lead to the goal being achieved. For example, these rules would tell us that if there were no paint on the instrument the desired result would not be achieved.

In much the same way, PAINTING1 (OBJECT not dirty) LEADS-TO the prevention of flaking and cracking, which is expressed by a separate frame, DIRTY-OBJECT, given below.

```
DIRTY-OBJECT (SIMPLE-EVENT)
   VARS: ...
```

```
EVENT: r(AND DIRTY-0BJECT1  (object is dirty)
             DIRTY-0BJECT2  (paint is put over
                             the dirt) )
           CAUSES (after a year, plus or minus a
                             factor of four)
         U(0R   DIRTY-0BJECT3  (paint flakes)
                DIRTY-0BJECT4  (paint cracks) )
```

In effect, then, we are told that PAINTING1 will match the negation of DIRTY-0BJECT1, and hence prevent the causal relation described in DIRTY-OBJECT.

Note that the EVENT in DIRTY-OBJECT is of a different form than that of PAINTING, as the former expresses a simple cause and effect relation, while the latter gives a complex series of "commands" without any cause and effect relations. In fact, they are two different kinds of frames, as is indicated by the type marks appearing by their names, SIMPLE-EVENT and COMPLEX-EVENT. These are two of five types of frames allowed by the system.

Returning to our PAINTING frame we can now see how story statements like those of (1) can be integrated into Ms. Malaprop's knowledge of the world. The first line of (1) (Jack was going to paint a chair) will set up an instance of the PAINTING frame with the AGENT and OBJECT variables bound appropriately. Then the second line comes in (Jack started to clean the chair). If we assume that the input representation of this corresponds to "Jack started an activity which would cause the chair not to be dirty" we can see that part of this will match PAINTING!, a fact which will be recorded by a LEADS-TO pointer. (Remember, LEADS-TO pointers indicate reasons for doing things.) Then, if Ms. Malaprop is asked "why" she will simply follow the pointer from the story statement back to PAINTING1, and reply, in effect, "one should not paint a dirty object". If asked "why" again, the program would follow the LEADS-TO pointer from PAINTING! to DIRTY-OBJECT!, and reply "otherwise the paint might flake".

III. Search and Language Comprehension

The notion of search has seldom played an important role in common sense inference work. Recently, however, the problem of search (and its brother, problem solving) has come up, at least implicitly, in work designed to show how a comprehension program might infer a person's motives from his actions or vice versa (Schank and Abel son 1975, Rieger 1976). I have not considered this problem with respect to Ms. Malaprop, but the program does have search problems, if at a somewhat more modest level of complexity. The problems stem from the fact that Ms. Malaprop must find (and hence search for) frame statements which match incoming story statements.

In most of the cases we have considered so far, Ms. Malaprop's search is a *Very* simple sort. At any given time there is a list of "context frames" which are simply those complex event frames which have been mentioned in the story. Given a story statement, a list of frame statements with the same predicate is retrieved from each context frame, and matches are attempted against all of them. (Matches may be more or less good, which will be discussed in the next section.) While I am sympathetic to the view that even this amount of search will prove unacceptable in a system which is to deal with the complexities of real stories, a more pressing problem is that even this flagrant use of search proves not to be sufficient. But to see this, let us first consider another example where normal search is sufficient.

(2) Jack was painting the wall. At one point he had too much paint on the brush. Q : What happened ? A : Possibly the paint dripped.

In this example, Ms. Malaprop interprets "too much" as saying that some threshold has been exceeded and this leads to an undesirable effect. That is to say, Ms. Malaprop starts out by looking for a command in an active frame which matches (3).

(3) (GREATER THRESHOLD PAINT-VOLUME)

She will find PAINTING5. Once found, Ms. Malaprop simply states that this command was not "obeyed". Any deleterious results from this will then be specified by the LEADS-TO pointer from the command.

But now consider the following example :

(4) Jack was painting a wall. At one point he pressed the brush too hard against the wall. Question : What happened ?

This is, in fact, *very* similar to (2) only instead of saying there was too much paint on the brush, we are told that he pressed too hard. The result, of course, is the same, the problem, of course, is to figure this out.

Working in a parallel fashion to the "too much paint" example, Ms. Malaprop will attempt to find in one of the context frames a command matching

(5) (GREATER THRESHOLD PRESSURE!)

Were she able to find a match for (5) she would then proceed to say that the command was disobeyed, but given the search mechanism just explained she will not, in fact, find the match. The reasons are not precisely obvious but a little explanation should make them clear.

As was assumed in the original "too much paint" example, the actual command PAINTING5 is to regulate the amount of paint - not the amount of pressure. PAINTING5 does state however that one way this is done is by regulating the pressure, but this does not allow us to make the match needed for (5). To see why, let us take a closer look at the relevant command.

(6) PAINTING5 (GREATER DRIP-THRESHOLD the amount of PAINT on the INSTRUMENT)
    C0MES-FR0M:((6a) (a rule which states that surface volume varies directly with pressure)
    (6b) (apply pressure to instrument))

The COMES-FROM here states that one applies pressure on the instrument (6b) and then by changing that pressure the volume will go up or down accordingly (6a). But to use this information to match (5) we must apply yet another rule, which states :

(7) If X varies (in)directly with Y, then X greater or less than a threshold (vice versa if indirectly) can be caused by Y being greater or less than a second threshold.

In terms of our example this means that the program must first look into the simple event frame which expresses rule (6a) to note that the volume varies directly with the pressure, and then apply rule (7) to infer that we must keep the pressure lower than some threshold. This last statement is the command which will match (5).

Given our previous search mechanism none of this would have taken place. For one thing the crucial information is not found in PAINTING, but in the frame for rule (6a), and secondly, we need to apply yet a second rule, namely (7) before this information yields a match. We could, of course, simply loosen our restrictions on search so that a) story statements will not only be matched against the context frames, but any frames which are pointed to by the context frames, and b) extra rules may be used in matching. However, if we were somewhat worried about the effects of the previous search technique in terms of search time, this new "restriction", or rather the lack of restrictions, on search, will be problematic to say the least.

The way in which Ms. Malaprop will actually handle this problem is quite different. It depends on the fact that in example (4) we not only know that there is "too much" of something, which is the portion of the input we have been emphasizing, but we also know that Jack is pressing the paint brush against the wall. This latter statement will, in fact, match (6b). Once this happens, Ms. Malaprop will then go into a different search mode, called the "restricted search" mode. In effect she assumes that any embellishments of the matched story statement should be matched against the same portion of the frame. Hence, in the attempt to match (5), she will concentrate all of her "energy" in the area of PAINTING5, and by so concentrating will allow herself to go considerably deeper into subframes than she would normally. I have yet to work out the exact restrictions which will apply here, but for our purposes it is sufficient to note that in the normal circumstance (i.e. when we do not have any previous match within the sentence to tell the program which part of which frame is being discussed), only the "normal", relatively superficial, frame search will be allowed.

Let us consider a second example, differing in detail, but with the same over-all problem.

(8) Jack was going to paint his chair green.
    He got some blue and yellow paint.
    Question : Why ?

The effect of the first line of (8) will be to set up an instance of the painting frame and note that the paint in question, although not yet mentioned in the story, will be green. Because people get things in order to use them, the second line of (8) will try to bind the blue and yellow paint to variables in a previously established frame. However, in the case of "paint" this will be narrowed down considerably, because one of the facts which we will know about paint is that its typical role in life is as the value of the variable PAINT in the PAINTING frame. Of course, in (8) this simple match will fail, for the colors are wrong, but note that we are again in a restricted search situation. That is, we know already which portion of PAINTING should match the second line of (8), so Ms. Malaprop will expend more effort in making the match. In particular, what should then occur is that the information about STUFF-COLOR (as opposed to simply COLOR which is "surface color") will be brought into play, and the search will be led to the frame for color mixing. Once there it is all downhill.

IV. The Problem of Matching

We have been talking about how Ms. Malaprop integrates a story statement by matching it against some frame statement. The last section was concerned with how potential matching frame statements are located, in this we will consider how the program decides that there is indeed a match.

One factor is time. For example, given sufficient disparities in time, some context complex event frames will not even be considered, simply because the events they describe were over so long before the new event that there cannot be any relation. In other cases the time disparity is not so great and time will only serve to eliminate certain possibilities within a particular frame. For example:

(9) Jack was going to paint the chair. He got some newspaper. Question : Why ? Answer : Presumably to put under the chair.

Within PAINTING there are two uses of newspaper : Once at the beginning to put around the object to be painted, and then again near the end for wiping the brush (not shown in the version of PAINTING given earlier). In (9), the fact that Jack has just started painting suggests that the earlier use is more reasonable. Hence we must still examine this context frame, but many of the later statements can be disregarded on the basis of time considerations.

But the major influence on matching is the binding of variables. This being a quite complex problem, let us start with the most simple situations and work our way up.

(10) Jack was going to paint the chair with a paint brush and some green paint. He dipped the paint brush in the paint.

The first line of this example tells the program to set up an instance of PAINTING, and further to make the following bindings :

(11) AGENT = JACK! OBJECT = CHAIR!
PAINT = GREEN-PAINT1
INSTRUMENT - PAINT-BRUSH1

Now, when we see the second line we will be trying to match :

(12) (LIQUID-IN INSTRUMENT PAINT)
from the painting frame
(13) (LIQUID-IN PAINT-BRUSH1 GREEN-PAINT)
from the story

Here LIQUID-IN is a predicate which expresses the fact that an object is submerged in a liquid. Given that both INSTRUMENT and PAINT are already bound to the objects they are supposed to match there is no difficulty and the match will be made.

But we will not always have situations where all of the variables are bound in advance. For example :

(14) Jack was painting a chair. He dipped a brush into the paint.

In trying to integrate the second line of (14) we have exactly the same situation as in matching (12) with (13), only this time neither of the relevant variables are bound. In such cases the bindings must be the result of the match. To see how this occurs, let us concentrate on INSTRUMENT.

Once Ms. Malaprop notes that INSTRUMENT is unbound she will examine the variable entry for the variable in PAINTING, and will find, roughly speaking, the following :

```
(15) (INSTRUMENT (15a) (SOLID INSTRUMENT)
                 NORMAL
                 (15b) (PAINT-BRUSH INSTRUMENT)
                 (15c) (ROLLER INSTRUMENT)
                 (15d) (ABSORBANT INSTRUMENT) )
```

The statement before the NORMAL states that the instrument must be a solid. Those after give various statements which are normally true of the variable. In the case of Jack's paint brush, it will certainly satisfy (15b) and if it is a typical brush will satisfy (15d) as well. In fact, simply satisfying (15b) would have been sufficient to ensure a match between INSTRUMENT and PAINT-BRUSH1.

But now let us consider a still more difficult example :

(16) Jack was going to paint the chair. He dipped a sponge into the paint.

We have here an anomalous situation, and if asked Ms. Malaprop would be quite right to say that she did not know what Jack was doing. But suppose we add :

(17) Then he wiped the sponge across a leg of the chair. Question : Why ? Answer : I guess he is painting the chair with his sponge.

The point of this example is that when first confronted with (15) the program tries to match (12) against :

(18) (LIQUID-IN SP0NGE1 PAINT!)

In the attempt to match SP0NGE1 to INSTRUMENT it will not only match the strict requirements, but will also match one of the normal conditions, that of absorbancy. However, this is not sufficient for Ms. Malaprop to make a match because a distinction is made between "normal object conditions" (which state what sort of object the thing is) which are sufficient to match the variable, and "normal property conditions" (like "this thing is normally absorbant") which are not. However, Ms. Malaprop will remember in the case of (16) that there was at least some positive evidence for the match, and when (17) comes in and the same variable match is tried again, this will be deemed sufficient and the match will be made.

This is somewhat complex, but it is not complex enough. In fact, the problem of matching as developed here is simply a special case of the recognition or diagnosis problem, and as such, given the notorious difficulties of these issues, is sure to remain beyond our grasp for quite some time.

V. <u>Guessing, and</u> Guessing Wrong

I stated at the outset that I would not try to justify here the decision to have Ms. Malaprop "integrate" new information at read time rather than waiting for user questions. Nevertheless, there are certain problems entailed by such a decision, and some discussion of these would be appropriate.

To say that Ms. Malaprop "integrates" incoming story statements at read time is simply another way to say that she makes inferences at read time. So, in matching statements like "Jack got some newspaper" against our frames we *are* assuming, that is, inferring that the newspaper will be used in the course of painting. We then confront two problems. First, since in theory the number of such inferences is infinite, which ones should the program make. Secondly, such inferences, while quite reasonable, may upon occasion be wrong - Jack may have fetched the paper because he was going to pack some glasses in a box immediately after he finished painting. So if we *are* to make such inferences we must be able to unmake them as well. But how ?

About the first of these I shall have nothing to say here. Indeed, about the second of these I have little to say - except for one point. If one hopes to correct mistaken beliefs, one should have some record of how this belief came about and what, if any, influence it has had on future computation. It is this question I wish to address, although we will approach it from a somewhat oblique angle.

Consider the following example :

(19) Jack was painting a chair. He dipped a brush into the paint. Then he drew the brush across the chair. Question : Could this step be left out ? Answer : No. Question : Why not ? Answer : Because it is this step which gets the paint on the chair, and that's what painting is about.

The interesting point about this second question is that it is a *yery* different sort of "why" question than the others we have considered. Formerly we have been asking, "what were the person's goals when he did this activity". This time the request is rather "how did you infer that this step could not be left out". So derivational information is not only needed to correct mistaken inferences, but to answer questions as well.

If we go back and look at how the program actually did infer that the step could not be left out, we see that it used a rule, which in simplified form looks like :

```
(20)   (AND OBLIGATORY!  (LEADS-TO X G)
             0BLIGAT0RY2  (GOAL G A)     )
       IFF
       0BLIGAT0RY3  (OBLIGATORY X A)
```

This states that if action X leads to the goal of an action A, then X is obligatory with respect to A. The user of this rule in the present case will produce :

```
(21) (OBLIGATORY PAINTING6 PAINTING)
          C0MES-FR0M:  (0BLIGAT0RY3)
```

(Actually, the relevant derivation would relate the story statement instantiating PAINTING6 with that instantiating PAINTING, but as usual, this introduces complications I would rather avoid.) In (21) C0MES-FR0M has been extended from its normal role of indicating how a state of affairs came about, and rather indicates how this particular fact was inferred. (The next example however will serve to indicate how close these two notions are within the system.) So, to answer the second question of (19) we simply follow the C0MES-FR0M pointer back to (20) and give the conditions which were used to make the inference as our answer. We will extend LEADS-TO in the analogous fashion.

Now let us turn to a case of mistaken assumption.

(22) Jack finished painting. He did not clean the brush. He left the brush in the paint. Question : Will the paint dry on the brush ?

After the second line Ms. Malaprop will infer the bad results from the failure to wash the brush. (I think I can justify this action, but not in the available space.) The problem with this inference in the case of (22) is that the last line tells us we jumped to a false conclusion.

The crucial knowledge here concerns evaporation, and in a simplified form would look like :

```
(23)   (AND EVAP1  (liquid on some surface)
            EVAP2  (liquid exposed to air)  )
      CAUSES
      EVAP3  (liquid dries)
```

Using this fact, by the time we reach line three of (22) we will have :

```
(24) (paint is sticking to the brush)
        C0MES-FR0M:  (the story itself)
        LEADS-TO:  (EVAP1)
(25) (paint exposed to air)
        C0MES-FR0M:  (air exposure rule, see below)
        LEADS-TO:  (EVAP2)
(26) (paint will dry)
        C0MES-FR0M:  (EVAP3)
```

In (25) the "air exposure rule" is a rule which states that, if there is no reason to believe the contrary, everything is assumed to be exposed to air. This rule, of course, is fallible, and will be marked as such.

Given such a structure, we can see, in principle, the program's reaction to the last line of (22). Concluding that the paint on the brush will not be exposed to air, it follows the results of this assumption to the rule of evaporation, and negates it, hence negating the fact that the paint brush will be unabsorbant the next time Jack wishes to use it (a fact not included in (24)-(26), but which would be in a complete version).

VI. Conclusion

Aside from the frames themselves, almost none of the program described here has been implemented. There does exist a "frame checker" which checks the frames for correct syntax and translates them into the internal structure used by Ms. Malaprop. That is all. I would estimate that the version which will handle all of the examples herein is six months off, but previous experience tells me that such estimates are likely to be too ambitious by factors of two or three.

I have no illusions that when completed Ms. Malaprop will handle these examples "correctly".

Ms. Malaprop can only be an approximation to "the truth" because she will have embedded in her approximate, or rather trivialized, solutions to the many standard AI problems which have come up in the course of this paper : search, matching, diagnosis, visual recognition, problem solving, etc.

AI workers have always believed in the essential unity of cognitive processes. At one time this was expressed in the belief that some simple idea (like heuristic search) underlay all of our mental abilities. Ms. Malaprop on the other reflects what I take to be the now emerging view - what we once took to be peculiar complexities of particular AI domains occur in the rest of AI as well. It is here that the unity lies.

Ac knowLedgements

This research was supported by the Dalle Molle Foundation and the Department of computer science, University of Geneva. Many of the ideas presented here were "batted around" in the Institute's Thursday afternoon seminars, and may properly "belong" to other people. But I doubt that any of us could remember. I would like to thank all the members of the seminar, and in particular Walter Bischof who displayed exemplary courage while being shot at.

IRe_fj?rences

Charniak, E. Inference and knowledge, part I. In E. Charniak and Y. Wilks (Eds.), Computational semantics, Amsterdam, North-Holland, 1976a, pp. 1-21.

Charniak, E. A framed PAINTING : the representation of a common sense knowledge fragment. Working paper 29, Institute for semantic and cognitive studies, 1976b.

Minsky, M. A framework for representing knowledge. In P.H. Winston (Ed.), The psychology of computer vi_sjon, New York, Mc'Graw-Hi 11 , 1975, pp." 21f-277.

Rieger, C. An organization of knowledge for problem solving and language comprehension. In : Artificial intelligence, 7, 1976, pp. 89~-12~7~. ~

Schank, R. and Abelson, R. Scripts, plans and knowledge. In : Advance papers of the fourth international conference on artificial inteiligence, Tbil isi ,1975, pp. 151-158.

Appendix

To give the reader some idea of the actual representation, I have included here two examples of frames as they are handed to Ms. Malaprop. Before actually being used however they are converted to a somewhat more convenient property list format, and some redundant pointers are added.

```
(ABSORBED-BY (STATE)
   ; This describes the state which typically
   ; results from the process of absorption.
  VARS: (FLU       (PHYS-OB FLU)
        ; Note that a fluid, such as paint, could
        ; be absorbed, only to dry and become a
        ; solid. This is why FLU is not required
        ; to be a liquid.
          NORMAL: (LIQUID FLU)  )
         (SOL      (SOLID SOL)   )
         (SUR-FLU  (PART-OF SUR-FLU FLU))
         (SUR      (SURFACE SUR SOL))
  RELATIONS:
     (ABSORBEDI (ABSORBED-BY FLU SOL)
        IMPLIES
      ABS0RBED2 (SUPPORT SOL FLU)   )
        ; This states that a solid which has ab-
        ; sorbed a fluid supports it.
     (ABS0RBED3 (ABSORBED-BY FLU SOL)
        IMPLIES
      ABS0RBED4 (STICKY-ON SUR-FLU SUR) )
        ; Some of the absorbed fluid will be on
        ; the surface.
  COMES-FROM: (ABS0RPTI0N4)
  LEADS-TO:    (AND (PRESS2) (V0L-INC3))         )

(ABSORPTION (SIMPLE-EVENT)
   ; This frame describes how having a liquid
   ; adhere to the surface of an absorbant solid
   ; will cause absorption.
  VARS: (LIQ  ABS0RPTI0N1 (LIQUID LIQ))
           ; ABSORPTION, as opposed to ABSORBED-BY,
           ; requires a 1iquid.
         (SOL  ABS0RPTI0N2 (ABSORBANT SOL))
  EVENT: ABS0RPTI0N3 (STICKY-ON LIQ (OBJ SOL))
           CAUSES
         ABS0RPTI0N4 (ABSORBED-BY LIQ SOL)    )
```