American Journal of Computational Linguistics *Microfiche 33 : 78*

Using Planning Structures to Generate Stories

James R. Meehan

Yale University
New Haven, Connectiuct 06511

ABSTRACT

TALE-SPIN is a program which makes up stories by using planning structures as part of its world knowledge. Planning structures represent goals and the methods of achieving those goals. Requirements for a particular method depend on static and dynamic facts about the world. TALE-SPIN changes the state of the world by creating new characters and presenting obstacles to goals. The reader / listener makes certain plot decisions during the telling of the story. The story is generated using the notation of Conceptual Dependency and is fed to another program which translates it into English.

INTRODUCTION TALE-SPIN is a computer program which makes up stories about characters who plan how to solve certain problems

---

This work was supported in part by the Advanced Research Projects Agency of the Department of Defense and monitored by the Office of Naval Research under contract N00014-75-C-1111.

and then carry out their plans. The planning procedures interact with a data base of knowledge about other characters and objects in the world, memory, and the personal relationships which exist between characters. The stories are represented in Conceptual Dependency and are passed to a program which expresses them in English. The reader is asked to make certain decisions about the story during the process of generation. Here is an example.

JOE BEAR WAS FAMISHED. HE DECIDED HE WOULD BE FULL IF HE ATE SOME HONEY. HE WANTED TO FIND OUT WHERE THE HONEY WAS. HE THOUGHT THAT IRVING BIRD WOULD TELL HIM WHERE THE HONEY WAS.
JOE BEAR WALKED TO THE TREE WHERE IRVING BIRD WAS. HE ASKED IRVING BIRD IF HE WOULD TELL HIM WHERE THE HONEY WAS.

>> DECIDE: DOES *IRVINGBIRD* AGREE? *NO

IRVING BIRD REFUSED TO TELL JOE BEAR WHERE THE HONEY WAS. JOE BEAR DECIDED IRVING BIRD WOULD LIKE HIM TO GIVE THE WORM TO HIM. JOE BEAR ASKED IRVING BIRD IF HE WOULD TELL HIM WHERE THE HONEY WAS IF HE GAVE THE WORM TO HIM.

>> DECIDE: DOES *IRVINGBIRD* AGREE? *YES

HE THOUGHT THAT HE WOULD LIKE JOE BEAR TO GIVE IT TO HIM. HE WANTED TO FIND OUT WHERE THE WORM WAS. HE THOUGHT THAT IRVING BIRD WOULD TELL HIM WHERE THE WORM WAS. JOE BEAR ASKED IRVING BIRD IF HE WOULD TELL HIM WHERE THE WORM WAS.

>> DECIDE: DOES *IRVINGBIRD* AGREE? *YES

IRVING BIRD DECIDED HE WOULD TELL JOE BEAR WHERE THE WORM WAS. IRVING BIRD TOLD HIM IT WAS AT A PATCH OF GROUND.
HE WALKED TO THE PATCH OF GROUND WHERE WORM WAS. HE TOOK THE WORM HE WALKED TO THE TREE WHERE IRVING BIRD WAS. HE GAVE THE WORM TO IRVING BIRD.

>> DECIDE: DOES *IRVINGBIRD* KEEP HIS PROMISE? *NO

HE REFUSED TO TELL JOE BEAR WHERE THE HONEY WAS. JOE BEAR TOLD IRVING BIRD HE IS GOING TO STRIKE HIM IF HE DOES NOT TELL HIM WHERE THE HONEY WAS.

>> DECIDE: DOES *IRVINGBIRD* IGNORE THE THREAT? *NO

IRVING BIRD DECIDED HE WOULD TELL JOE BEAR WHERE THE HONEY WAS. IRVING BIRD TOLD HIM IT WAS AT THE BEEHIVE.

JOE BEAR THOUGHT THAT HENRY BEE WOULD GIVE THE HONEY TO HIM. JOE BEAR WALKED TO THE BEEHIVE WHERE HENRY BEE WAS. HE ASKED HENRY BEE IF HE WOULD GIVE THE HONEY TO HIM.

>> DECIDE: DOES *HENRYBEE* AGREE? *YES

HENRY BEE DECIDED HE WOULD GIVE IT TO JOE BEAR. HENRY BEE GAVE IT TO JOE BEAR. HE ATE IT. HE WAS FULL. THE END.

Here is a story which TALE-SPIN generates which the translator is not yet capable of producing in English:

JOE BEAR WAS HUNGRY. HE THOUGHT THAT IRVING BIRD WUOLD TELL HIM WHERE SOME HONEY WAS. HE WALKED TO THE TREE WHERE IRVING BIRD WAS. HE ASKED IRVING BIRD TO TELL HIM WHERE THE HONEY WAS. IRVING BIRD TOLD HIM THE HONEY WAS IN A [certain] BEEHIVE.

JOE BEAR WALKED TO THE BEEHIVE WHERE THE HONEY WAS. HE ASKED HENRY BEE TO GIVE HIM THE HONEY. HENRY BEE REFUSED. JOE BEAR TOLD HIM WHERE SOME FLOWERS WERE. HENRY BEE FLEW FROM THE BEEHIVE TO THE FLOWERBED WHERE THE FLOWERS WERE. JOE BEAR ATE THE HONEY.

HE WAS VERY TIRED. HE WALKED TO HIS CAVE. HE SLEPT. THE END.

TALE-SPIN starts with a small set of characters and various facts about them. It also has a set of problem-solving procedures which generate the events in the story. Many decisions have to be made as the story is being told. Some are made at random (names of characters, for example); others depend on the relationships between characters (whom one asks for information, for example); others are made by the reader (whether a character keeps a promise, for example).

TALE-SPIN generates sentences using the representation system of Conceptual Dependency (Schank 1975). Some of the Conceptual Dependency (CD) structures are passed on to a program which expresses them in English. (The original version of that program was written by Neil Goldman for the MARGIE system. The present version has been modified by Walter Stutzman and Gerald

De Jong.) The sentences which are not passed to the translator are those which represent easily inferred ideas. Neither program yet worries about the style of expression; that is, we worry about whether to say a newly generated piece of the story, but not much about how to say it.

A TALE-SPIN story involves a single main character who solves some problem. To make the process interesting, obstacles are introduced, some by the reader if he chooses, and some at random. For instance, the reader's decision that Irving Bird is not going to tell Joe Bear what he wants to know produces an obstacle to Joe Bear's plan to find something out. Some obstacles are created when certain scenes are included in the story. For instance, the initial world state has no bees in it, but when it comes time in the story to conjure up some actual honey, we do so by creating a whole scene which includes some honey in a beehive in a tree and a bee who owns that honey. The bee may or may not be at home. If he is, Joe Bear is going to have another obstacle in his plan when he gets to the beehive.

The story is the narration of some of the events which occur during the solution (or non-solution) of the problem. (That is, more things happen in the solution of a problem than a storyteller says or needs to say.) TALE-SPIN differs from other problem-solving systems in several ways: (1) the problems it solves are those requiring interaction with other, unpredictable characters rather than with a data base of theorems or blocks or circuits; (2) the world inside TALE-SPIN grows: new characters are created with unpredictable effects on the story; (3)

obstacles are deliberately introduced; (4) an "unsuccessful" story, one in which the problem is not solved, can be just as interesting as a "successful" one.

PLANNING STRUCTURES Planning structures are what we use to organize knowledge about planful activity, which is represented in CD by a chain of causes and effects. The planning structures include delta-acts, planboxes, packages, scripts, sigma-states, rho-states, and pi-states.

A delta-act is used to achieve a particular goalstate. Delta-prox (written here as dPROX) is the procedure for becoming proximate to some location. A delta-act is defined as a goal, a set of planboxes, and a decision algorithm for choosing between planboxes.

A planbox is a particular method for achieving a goalstate. All the planboxes under a delta-act achieve the same goalstate. Each planbox has a set of preconditions (some of which may be delta-acts), and a set of actions to perform. "Unconscious" preconditions are attached to planboxes which would never occur to you to use. If you're trying to become proximate to X, you don't even think about persuading X to come to you when X is an inanimate object. "Uncontrollable" preconditions cannot be made true if they're not already true. (The assumption is that they are sometimes true.) "Plantime" preconditions are the things you worry about when you're making up the plan. You don't worry about "runtime" preconditions until you're executing the plan. ("Planning" is a mental activity. PLAN is, in fact, one of the primitive ACTs of CD. "Executing a plan" is performing a

logically structured sequence of actions to achieve the goal of the plan.) If I'm planning to get a Coke out of the machine upstairs, I worry about having enough money, but I don't worry about walking up the stairs until I'm at the stairs. That the machine actually has some Coke is an uncontrollable runtime precondition: I don't worry about it until I get there, and there's nothing I can do if it is empty when I get there.

A package is a set of planboxes which lead to a goal act rather than state. The PERSUADE package, for instance, contains planboxes for X to persuade Y to do some act Z. The planboxes include asking, giving legitimate reasons, offering favors in return, threatening, and so on.

Goalstates come in various flavors. There are the goals which are associated with the delta-acts: the goal of dPROX is to be somewhere, the goal of dKNOW is to find out the answer to some question, the goal of dCONTROL is to possess something. But there are also goals of satiation, called sigma-states. For example, sHUNGER organizes the knowledge about satisfying hunger (invoking dCONTROL of some food, eating). TALE-SPIN also uses sigma-state knowledge in the bargaining process; offering someone some food in return for a favor is legitimate since it will satisfy a precondition for sHUNGER. There are also goals of preservation, called pi-states, which are most interesting when they are in danger of being violated. The logic of the THREATEN planbox in the PERSUADE package, for example, derives from the fact that physical violence conflicts with pHEALTH.

A SAMPLE DELTA-ACT: dPROX TALE-SPIN does not include all nine

delta-acts described by Abelson (1975). It contains the three which closely correspond to primitive acts: dPROX (PTRANS), dCONTROL (ATRANS), dKNOW (MTRANS).

Here is an outline of dPROX:

dPROX(X,Y) -- X wishes to be near Y

Planbox Ø: if X is already near Y, succeed.

Planbox 1: X goes to Y

uncontrollable precondition: can X move himself?

plantime precondition: dKNOW(location of Y)

runtime precondition: dLINK(location of Y)

action: PTRANS to location of Y

runtime precondition: is Y really there? (We may have gotten false information during the dKNOW.)

Planbox 2: Y comes to X

unconscious precondition: is Y animate?

uncontrollable precondition: is Y movable?

action: PERSUADE Y to PTRANS himself to X (PERSUADE package)

Planbox 3: Agent A brings X to Y

uncontrollable precondition: is X movable?

action: X gets AGENT to bring X to Y (AGENCY package)

Planbox 4: Agent A brings Y to X

unconscious precondition: is Y animate?

uncontrollable precondition: is Y movable?

action: X gets AGENT to bring Y to X (AGENCY package)

Planbox 5: X and Y meet at location Z

unconscious precondition: is Y animate?

uncontrollable precondition: is Y movable?

actions: PERSUADE Y to PTRANS himself to Z and dPROX(X,Z)

THE DATA BASE Planning structures are essentially procedural. The non-procedural data base used by the planning structures is divided into five classes.

1. Data about individual PPs (Picture Producers, nouns) where applicable: height; weight; where their home is; who their acquaintances are.

2. Data common to classes of PPs (e.g., data common to all birds) where applicable: what they eat; what their goals (sigma-states) are; whether they are animate (capable of MBUILDing), movable, self-movable; how they move around.

3. Sigma-state knowledge indicating how to achieve a sigma-state and what the plantime preconditions are that someone other than the planner can achieve. This is used in the bargaining process. Joe Bear offers to bring Irving Bird a worm because dCONTROL(FOOD) is a plantime precondition for sHUNGER which Joe Bear can achieve for Irving Bird. There are no plantime preconditions for sREST that he can achieve for Irving Bird (except maybe to leave him alone).

4. Memory: what everybody knows (thinks, believes); what Joe Bear knows; what Joe Bear thinks Irving Bird knows; etc. Planbox Ø of dKNOW, for example, accesses Memory to test whether Joe Bear already knows the answer to the question being asked, or whether it is public knowledge. Since both the question and the facts in Memory are represented in CD, the pattern match is very simple, taking advantage of CD's canonical representation of meaning.

5. Personal relationships. The relationship of one character to another is descibed by a point on each of three scales: COMPETITION, DOMINANCE, and FAMILIARITY. Scale values range from -10 to +10. The relation "is a friend of" is represented by a certain range on each of the three scales. The relation "would act as an agent for" is represented by a different range. The sentence "Joe Bear thought that Irving Bird would tell him where the honey was" comes from the "Ask a Friend" planbox of dKNOW. There is a procedure which goes through a list of Joe Bear's acquaintances and produces a list of those who qualify as "friends", i.e., those who fit somewhere within the "friend" range.

Relations are not symmetric: Joe Bear may think of Irving Bird as his friend, so he might ask him where the honey is, but Irving Bird may not think of Joe Bear as his friend at all, in which case he might refuse to answer Joe Bear.

Relationships can change. If Joe Bear becomes sufficiently aggravated at his "friend" Irving Bird and has to threaten to bash him in the beak in order to get him to tell him where the honey is, then the relationship between them deteriorates.

We plan to extend this feature to describe a character's "default" relationship: how he relates to total strangers. This would not necessarily be the point (0,0,0) but rather some point which would be used to give a rough indication of the character's "personality". Big bad Joe Bear might rate at (+6,+9,+4), where small meek Bill Worm might rate at (-6,-10,-4).

Changing a relationship is a type of goal we haven't yet

considered in much detail, although goals of relationships (rho-states) clearly exist. The procedure for getting someone to like you (rLIKE) might contain planboxes for ATRANSing gifts, MTRANSing sweet nothings, etc., in addition to changing your own feelings toward that person so that if he (she) asks you to do something, you don't refuse.

Information gets into the data base in several ways. Memory data gets produced directly by the planning structures. Changes in relations are side-effects of the present set of planning structures. But things have to start somewhere. There is a function CREATE(X) which invents a new item of type X (e.g., bear, flower, berry). Associated with each type of item is a small procedure called a picture which invents the desired item and others as required. For example, when we create some honey, we also create a beehive, a tree, and a bee. The honey is "owned" by the bee and is inside the beehive which is in the tree. The bee may or not be at home. Randomly chosen names, heights, weights, etc., are attached. All this data is then added to Memory.

The CREATE function is called when needed; remember that TALE-SPIN models the process of making up a story as you go along. We will now follow, in detail, the production of the second sample story.

CREATE a bear, which invokes a picture procedure which invents a bear. Assume the bear is named Joe; although since the name is chosen at random from a list of first names, it is just as often Irving. A cave is also invented, and has Joe in it.

Joe's location becomes public knowledge.

CREATE a bird, named Irving, and a tree which is his home. Irving's location is also now public knowledge.

Assert that Joe is hungry. This fact enters Joe's Memory. We also "say" this; that is, we pass it to the English translator which then produces the sentence "JOE BEAR WAS HUNGRY".

Invoke sHUNGER.

Choose at random a food that bears eat: honey. Assert that Joe is now planning to achieve the goal (sigma-state) of satisfying his hunger. Assert that he has decided that eating the food can lead to the achievement of his goal.

sHUNGER calls dCONTROL(honey). This forms a new goal, namely, that Joe have some honey. dCONTROL's "Planbox Ø" asks Memory if the goal is already true: does Joe already have some honey? The answer comes back: no. A plantime precondition is to know the location of some honey, so dCONTROL calls dKNOW(where is honey?). (The question is represented in CD, not English.)

dKNOW forms the new goal. dKNOW's "Planbox Ø" asks Memory whether Joe knows the location of any honey. Memory says no. Planbox 1 tests whether the question can be answered by consulting a standard reference (e.g., "What time is it?"). That fails. Planbox 2 tests whether the question requires expertise: no. Planbox 3 tests whether this is a "general information" question. It is, so we assert that Joe is planning to answer this question using Planbox 3 ("Ask a Friend").

Planbox 3 starts. Choose a friend: Irving. dKNOW calls

the PERSUADE package to try to get Irving to answer Joe's question.

PERSUADE asks Memory whether Joe thinks that Irving cannot answer the question. Answer: no. Irving is a "friend", so we try the ASK planbox. Assert that Joe thinks that Irving will tell him where the honey is. PERSUADE calls dPROX(Irving), since Joe needs to speak to Irving.

dPROX asks Memory whether Joe is already near Irving. Memory says no. Planbox 1: is Joe self-movable? Yes. Assert that Joe is planning to be near Irving by going there himself. dPROX calls dKNOW(where is Irving?).

dKNOW's "Planbox Ø" asks Memory whether Joe already knows where Irving is. The answer comes back: yes, Irving is in a certain tree. dKNOW returns this to dPROX. (We will omit future references to "Planbox Ø".)

dPROX asserts that Joe walks to the tree where Irving is. We ask Memory whether Irving is actually there. He is, so dPROX has achieved its desired goal; his change in location is added to Memory. dPROX returns to PERSUADE.

Joe asks Irving where some honey is. The reader now gets to decide whether Irving agrees to do so. Assume the reader says yes. We ask Memory whether Irving actually knows where any honey is. If he did, we would have Irving tell him, but he doesn't, so we CREATE some honey: a storyteller can create solutions to problems as well as obstacles! Some honey is invented, along with a beehive, a tree, and a bee (Henry) who is at home. Irving tells Joe that the honey is in the beehive. ASK succeeds, so

PERSUADE succeeds, so dKNOW succeeds: Joe knows where some honey is.

Back in dCONTROL, we ask Memory whether [Joe thinks that] anyone owns the honey. Memory says that Henry does, so dCONTROL's Planbox 1 ("Free for the taking") fails. Planbox 2 is to PERSUADE Henry to give the honey to Joe.

Given no relation between Joe and Henry (they don't know each other), the only planboxes in PERSUADE which can be used are ASK and INFORM REASON.

We try ASK first. This calls dPROX(Henry) which succeeds since Joe knows where Henry is; we omit the details here. Joe asks Henry to give him the honey, and the reader decides that Henry refuses.

We try INFORM REASON next. We choose a goal of Henry's and build a causal chain backwards from the goal. For example, one of Henry s goals is to "eat" flowers. (TALE-SPIN thinks that what bees do to flowers is equivalent to eating.) In order to eat a flower, you have to "control" a flower, which results from someone (possibly you yourself) ATRANSing the flower to you. We test whether what Joe is trying to PERSUADE Henry to do matches ATRANSing a flower. It doesn't. (Joe is trying to PERSUADE Henry to ATRANS the honey to him.) We then consider that in order to ATRANS a flower, you have to be near the flower, which results from someone PTRANSing you to the flower. Does this match? No. We repeat this process a few times, trying to construct a short inference chain which connects what Joe is trying to persuade Henry to do with one of Henry's goals. INFORM

REASON fails, and we return to dCONTROL.

The next Planbox is called "Steal". We ask Memory whether Henry is home; if he weren't, Joe would simply take the honey. But Memory tells us that Henry is home, so STEAL calls PERSUADE to get Henry to leave home; that is, Joe is now going to try to persuade Henry to PTRANS himself from the hive.

In the context of STEAL, the ASK planbox is not used. Joe tries INFORM REASON again and succeeds in producing the following chain: we get to the idea of someone PTRANSing himself to a flower again as we did before, but we notice that this does match what we are trying to persuade Henry to do: the connection is that Henry will PTRANS himself from the beehive to the flower. Joe now considers the precondition for Henry's PTRANSing himself to the flower, namely, that Henry has to know where the flower is. Memory does not indicate that Joe thinks that Henry knows where a flower is, nor does Joe know where a flower is, but rather than invoke dKNOW(where is a flower?), we CREATE a flower: this is legitimate in a plan to steal something. Joe now tells Henry that there is a flower in a certain flowerbed, and then asks Henry if he would like to fly to that flower. Henry agrees and flies away. PERSUADE succeeds, and returns to dCONTROL.

Joe now takes the honey from the hive, so dCONTROL succeeds and returns to sHUNGER. Memory is modified to indicate that Joe knows that he has the honey, but that Henry does not.

Joe now eats the honey, and has achieved the sigma-state of not being hungry. But, when bears eat, they become tired, so sREST is invoked.

sREST is very short. It requires a dPROX(cave), which is easily achieved, and then Joe goes to sleep.

Since the main goal has been achieved, and the goal produced as a consequence of that goal has also been achieved, the story ends.

What distinguishes stories from simple sequences of events? Coherency is important: there has to be a logical flow from one event to the next. This is represented in CD as a chain of acts which result in states which enable further acts and so on. Interest is important: something interesting or unusual has to happen or else the reader will begin to wonder what the point of the story is. TALE-SPIN creates impediments to goals, on the assumption that the overcoming of obstacles can make an interesting story. "One day Joe Bear was hungry. There was a jar of honey right next to him. He ate it. The end" is not a story. It shouldn't be that easy.

On the other hand, it shouldn't be too hard either. In theory at least, there is a cost-effectiveness calculus which people employ when deciding how much energy to expend on a subgoal, based on how much the goal is worth to them. This process prevents the plans from being too complicated.

As the story is generated, various plot decisions have to be made. Some decisions are made at random, others are made by the reader. When Joe Bear threatens Irving Bird because Irving Bird won't tell him where the honey is, the reader gets to decide whether Irving Bird is going to ignore the threat.

We use planning structures because any program which reads

or writes a story, whether of the folktale variety or the New York Times variety, must have a model of the logic of human activity. It might be easier to simulate the generation of a highly stylized form of story, as Klein (1974) has done using Propp's analysis of a class of Russian fairy tales, but there is little generality there. One could use any of the well-known problem-solving systems like MICRO-PLANNER, but the story is the proof procedure, and the procedure used there does not correspond to my conception of how people solve problems. That's not a criticism of MICRO-PLANNER as a problem-solver, but only as a model of human problem-solving.

User interaction was included for two reasons. First, the interactive feature now serves as a heuristic for placing bounds on the complexity of the story. Beyond some number of obstacles to the goal, a story becomes a kind of joke. Second and more important, extensions to TALE-SPIN will include more sophisticated responses than the present yes/no variety.

THE FUTURE OF TALE-SPIN. There are a lot of things that TALE-SPIN doesn't do yet that would improve it as a storyteller. Here are some of the theoretical problems we will be working on in the immediate future. (1) Bargaining, as it exists now in TALE-SPIN, is a pretty one-sided affair, with the main character making all the proposals. Irving Bird is just as likely to suggest that Joe Bear go get him a worm as Joe is to offer to do so. Counter-proposals are certainly common enough. (2) Future stories should include planning on the part of more than one character. The present stories are all "about" the bear, and

only incidentally involve the bird and other characters. The stories are more concerned with reaction than interaction. (3) For every plan, there may be a counter-plan, a plan to block the achievement of a goal: a plan for keeping away from something or someone; a plan not to find out something, or to be convinced that it isn't true; a plan to get rid of something you own. (4) How much of a plan do people consider in advance? We have made some efforts in this area by making the distinctions between kinds of preconditions. Certainly the most important improvement here will be the cost-effectiveness reasoning. (5) The theory of telling stories (what to say) now implemented in TALE-SPIN is to express violations of sigma-states ("Joe Bear was hungry"), physical acts, and those mental acts which provide motivation or justification for later events. The reader is assumed to be able to infer the rest. This seems to work reasonably well for the present simple stories, but may have to be modified to suit longer, more complicated stories.

REFERENCES

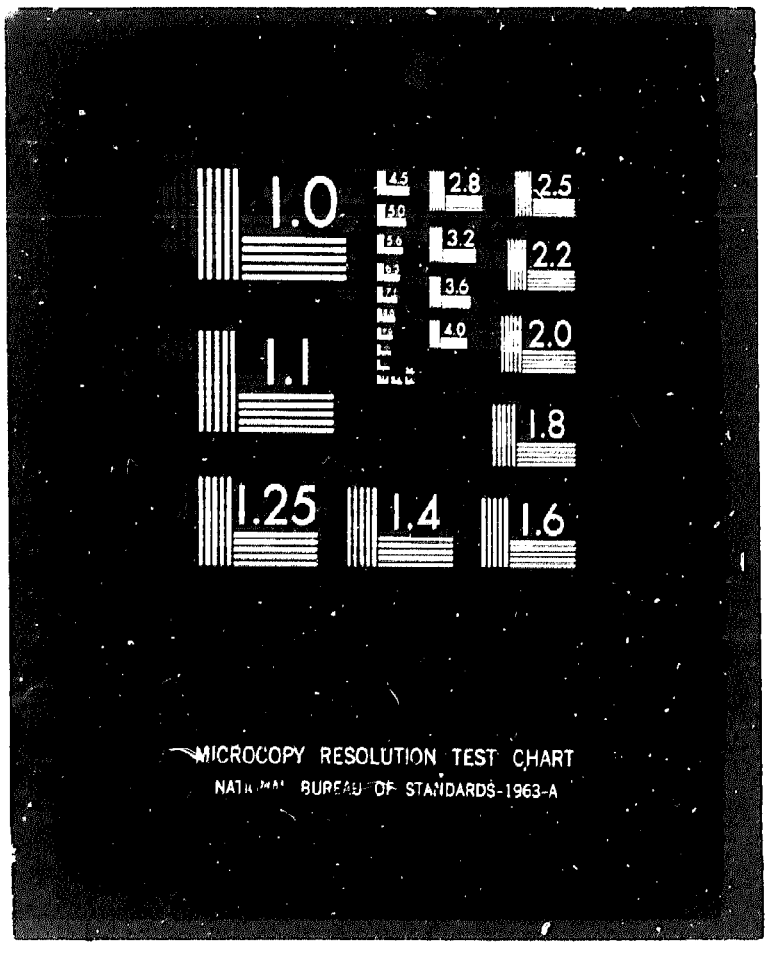
Abelson, R. P. (1975). Concepts for representing mundane reality in plans. In D. Bobrow and A. Collins, eds. Representation and understanding: Studies in cognitive science. Academic Press, New York.

Klein, S. et al (1974). Modelling Propp and Levi-Strauss in a meta-symbolic simulation system. Technical Report 226, University of Wisconsin at Madison.

Schank, R. C. (1975). Conceptual Information Processing. American Elsevier, New York. This includes contributions by Neil M. Goldman, Charles J. Rieger III, and Christopher K. Riesbeck.

Schank, R. C. and Abelson, R. P. (1975). Scripts, plans and knowledge. In Proceedings of the 4th International Joint Conference on Artificial Intelligence.

END


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
7.1
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A