# MECHANIZING THE COMMON-SENSE INFERENCE OF RULES WHICH DIRECT BEHAVIOR†

Elliot M. Soloway
Edward M. Riseman

COINS Technical Report 76-2

April 1976

We shall present a system that augments its a priori general understanding of human behavior in a spatio-temporal domain of physical activity, goals, competition, and gaming. Observing seemingly novel situations within that domain, it uses common-sense reasoning to construct an interpretation of the observed actions. This understanding is expressed in terms of the goals of the actors executing the actions. The system has the ability to focus its attention, as well as generation, generalize, and verify hypotheses. The desired result is a consistent structure of generalized hypotheses which represent both the regularities and an understanding of the observed domain.

†To appear in the Proceedings of the Conference on Artificial Intelligence and Simulation of Behavior, University of Edinburgh, July 1976.

# MECHANIZING THE COMMON-SENSE INFERENCE OF RULES WHICH DIRECT BEHAVIOR†

Elliot M. Soloway
Edward M. Riseman

COINS Technical Report 76-2

April 1976

We shall present a system that augments its a priori general understanding of human behavior in a spatio-temporal domain of physical activity, goals, competition, and gaming. Observing seemingly novel situations within that domain, it uses common-sense reasoning to construct an interpretation of the observed actions. This understanding is expressed in terms of the goals of the actors executing the actions. The system has the ability to focus its attention, as well as generation, generalize, and verify hypotheses. The desired result is a consistent structure of generalized hypotheses which represent both the regularities and an understanding of the observed domain.

---

† To appear in the Proceedings of the Conference on Artificial Intelligence and Simulation of Behavior, University of Edinburgh, July 1976.

## INTRODUCTION

People seem able to face novel situations and within a short time understand them fairly well. For example, a child entering a new school often can quickly learn what the social structure is in his new classroom, e.g., if there is a bully or teacher's pet, who that individual is, etc. If that new student would observe a fellow classmate shoving a student out of the front of a line, and teasing another about his freckles, etc., he might reason that this fellow may be the class bully. What the child has done is use prior knowledge in order to structure his model of this new situation. By an inference process, which we would call *common-sense reasoning*, the child was able to piece together an understanding of seemingly isolated actions in terms of the common goals of the actor executing those actions. From this interpretation, the new student can predict future scenarios and develop a strategy to circumvent the power structure of the class bully. Similarly, in our daily lives, we are often faced with the behavior of humans in novel mini-situations. To explain such behavior we draw on our general understanding of people's goals in various situations in order to fit together a model of this new situation.

In this paper we will present some results of an application of this common-sense reasoning process to knowledge acquisition by the computer; it represents further development of some of the ideas previously outlined (Soloway and Riseman

1975). Our system operates in a domain of *actors* and *actions* varying over *space* and *time*. Specifically, we are developing a program that is initially given a high level description of action-oriented games (e.g., cricket, baseball, tennis, etc.). This description is expressed in terms of the *goals* and *intentions* involved in this situation, e.g., winning, scoring, etc. The description must also include the conditions that must be satisfied for actions to be counted as mediators of those goals. Driven by the observation of the activity in the game, which in our case is baseball, the system will use its general knowledge of action-oriented gaming in order to acquire an understanding of the particular goals of the people involved in this game.

As a second part of this reasoning process, the system must abstract regularities that it perceives in its world. These regularities serve as *rules* or *conventions* that govern the game and constrain the ways in which players can achieve their goals.

Not surprisingly, there is a high degree of similarity in the issues that our system must deal with and the issues that story understanding systems must face. Understanding human actions, whether read from a narrative or perceived directly, requires in addition to the understanding of the underlying goals and intentions of the actors, an understanding of the underlying causal relationships that link the actions of those actors together. Schank (1974) stresses the need for discovering via inferences the causal relationships between actions. Scripts (Schank and Abelson 1975) permit the system to 'fill-in' those causal links in stories that deal with stereotypic behavior scenarios. Since it is the task of our system to generate something like a script for baseball, our system is more akin to a constructive approach to behavior understanding (Schmidt 1976; Schmidt and Sridharan 1976; Schmidt and Goodson 1976). Schmidt argues that in order to deal with the complexity and infinite variations in human behavior, a system--be it human or machine--must ultimately be able to construct a plan(s) that serves to explain the observed behavior. This plan is expressed in such higher level terms as goals, motives, reasons, etc. Finally, in a later section we will discuss the relationship of our model of learning to those of Winston (1970), Sussman (1973), and Hayes-Roth (1976). Thus, our system uses its general knowledge about the kinds of goals and causal relationships important in action-oriented games to compose plans that explain the activity in baseball, and that capture regularities in that activity.

The following presentation will mirror the flow of information in our system, which is depicted graphically in figure 1. We will first discuss the representation of our mini-world and the systems' knowledge of the lowest level descriptors in this representation. This will be followed by a description of the various roles of *Attention Mechanism*: (1) focusing subsequent processing on 'interesting' action sequences; (2) abstracting recurrent action sequences; (3) watching for specific action sequences that have been fed back from the higher centers of processing. Next, examples will be given which illustrate the reasoning process of the Hypothesis Generator. It is in this phase that conjectures are made as to the causal relationships between the players' activities, goals, and corresponding successes and failures. Of most importance is the formation of a coherent interpretation of the composite activity. Finally, the Hypothesis Verification and Generalization section briefly describes techniques used to gather evidence for confirmation or rejection of hypotheses, and the process of abstracting important recurrent events during the grouping of similar sequences of activity.

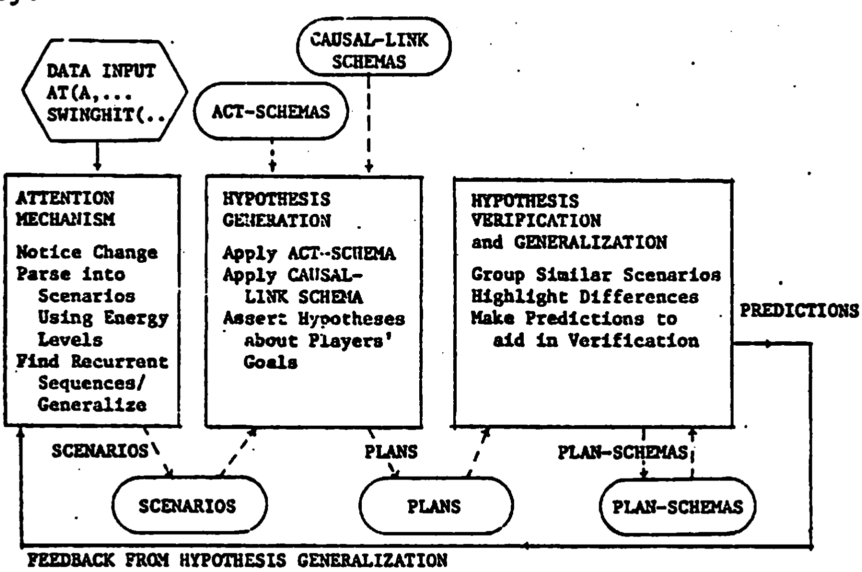


**Figure 1: System Overview**

**Ovals indicate Data Bases and Squares indicate Procedures.**

## REPRESENTATION OF THE ACTIONS

In order to understand sequences of actions over time, the system must be able to understand the necessary and probabilistic changes and constancies brought about by those actions, i.e., one aspect of the Frame Problem (McCarthy-Hayes 1969; Sridharan 1976). This requires a somewhat surprising amount of detail. (figures 2 and 3). For each action the following information is grouped into an ACT-SCHEMA: (1) the direct and indirect preconditions for execution of the action; (2) the direct and indirect consequences of the action; (3) the degree of skill and energy required to perform the action; (4) the (coarse) expected distribution of the probabilities of success/failure; (5) the general goals of the action.

| | |
|---|---|
| CATCH(PLAYER,OBJECT,LOCATION) | AT(PLAYER,LOCATION) |
| THROW(PLAYER,OBJECT,LOCATION) | ON(PLAYER,LOCATION) |
| WALK(PLAYER,STARTING-LOCATION) | TEAMBOX(TEAM-NAME,OUTS,RUNS-THIS-INNING,TOTAL-RUNS) |
| RUN(PLAYER,STARTING-LOCATION) | BATBOX(PLAYER-NAME,STRIKES,BALLS) |
| FAST,SLOW.GROUND,AIR- used as modifiers on actions | |
| SWINGHIT(PLAYER,OBJECT,LOCATION) | MOVING(OBJECT,STARTING-LOCATION,MODIFIERS) |
| SWINGMISS(PLAYER,OBJECT,LOCATION) | INNING(INNING-NUMBER) |

Figure 2: Listing of Primitive Descriptor Units

The system does not understand the 'baseball' meaning of any of the primitives. In particular, it understands only that INNING, BATBOX, and TEAMBOX are scoreboards. It does not know which events correlate with the counts and does not understand the concept INNING.

```
(SWINGHIT
 (PRIMACT ((PROPEL-INANOBJ ?SNAPNUM1 ?PERSON1 ?LOCATION1)
          (GENERAL-GOALS (PROPEL-INANOBJ (XOR (TO $LOCATION5) (AWAY $LOCATION1))))
          (DIFFICULTY (HIGH SKILL) (MEDIUM ENERGY))))
 (DIRECT-PEC
  ((MUST-EXIST ((MOVING-INANOBJ (BEFORE $SNAPNUM1 ) NIL ?LOCATION2)
              ((COUNTS-AS PROPEL-INANOBJ) (BEFORE $SNAPNUM1) ?PERSON2 $LOCATION2)))
  )
  (DELTA-ENERGY-SKILL (DIFFICULTY-INCREASES-IF (FASTER MOVING-INANOBJ))))
 (INDIRECT-PEC ((MUST-EXIST ((COUNTS-AS LOCATE-ANOBJ) $SNAPNUM1 $PERSON1 $LOCATION1))))
 (DIRECT-CONSEQ
  ((MUST-EXIST ((MOVING-INANOBJ (AFTER $SNAPNUM1) NIL ?LOCATION4)
              ((COUNTS-AS LOCATE-INANOBJ) (AFTER $SNAPNUM1) ?PERSON5 ?LOCATION5)))
   (DELTA-ENERGY-SKILL (CAN-EFFECT (FASTER MOVING-INANOBJ)(FARTHER MOVING-INANOBJ)))))
 (INDIRECT-CONSEQ ((MUST-EXIST ((COUNTS-AS LOCATE-ANOBJ) $SNAPNUM1 $PERSON1 $LOCATION1)))))
```

Figure 3: ACT SCHEMA Representation of the act SWINGHIT

NOTE: In the actual implementation of the ACT SCHEMAS, the atom names preceded by ? or $ are actually function calls which serve to bind those atom names to the actual values in the action descriptor units of the scenario

## REPRESENTATION OF THE SPATIO-TEMPORAL ENVIRONMENT

The game of baseball is fed to the system in a discrete form. Frozen *snapshots* of the real activity are taken at successive event times during the game. Each action descriptor unit of a snapshot captures four fundamental features of a spatio-temporal domain: action, actor, location and time. Figure 4a gives a sample of 3 snapshots in which player A1 throws a ball. Unfortunately, space does not permit a full discussion of the descriptors chosen to represent the game. Suffice it to say that, though we do not represent the color of the players eyes, or the clouds moving, the combinatorics of the many descriptors that we did chose still make the problem far from trivial. Moreover, in discussing the rest of the system, it will be clear that many irrelevant features of this environment can be habituated out. Note that the machine does *not* initially understand in any operational sense what the meaning of the symbol INNING is. Semantic labels on locations, like home-plate, pitchers mound, etc., are equally mysterious; to the system they are (x,y) coordinates which happen to be locations of recurring activity.

## ATTENTION MECHANISM: FOCUS

As in any animal's brain, our computer program must filter out most of the incoming sense impressions and pass on to the higher centers only the most interesting ones. In particular, there are 22 action descriptor units per snapshot and about 6500 snapshots per game. Therefore, the job of the Attention Mechanism is to focus attention on interesting sequence of actions and pass those on for further analysis. The system's definition of 'interesting' is biologically motivated and embodied in two ways in this front-end preprocessor: (1) it attends to sequences of snapshots where there is activity and change, and (2) it notes in particular those subsequences of actions that recur.

The first characteristic translates into an algorithm which filters out non-activity (AT, ON in our case) while highlighting activity chains. The amount of data reduction using this algorithm is quite significant. Figure 4b illustrates the application of this filtering algorithm to the snapshots of figure 4a.

COMPLETE SNAPSHOTS:

| TIME: | 14 | 15 | 16 |
|---|---|---|---|
| | HOLDOBJ(A1,BALL,PM) | THROW(A1,BALL,PM) | AT(A1,PM) |
| | AT(A2,HP) | AT(A2,HP) | AT(A2,HP) |
| | AT(A3,FB) | AT(A3,FB) | AT(A3,FB) |
| | . | . | . |
| | . | . | . |
| | AT(A9,RF) | AT(A9,RF) | AT(A9,RF) |
| | AT(B1,HP) | AT(B1,HP) | AT(B1,HP) |
| | AT(B2,DUGOUTB) | AT(B2,DUGOUTB) | AT(B2,DUGOUTB) |
| | AT(B3,DUGOUTB) | AT(B3,DUGOUTB) | AT(B3,DUGOUTB) |
| | . | . | . |
| | . | . | . |
| | AT(A9,DUGOUTB) | AT(A9,DUGOUTB) | AT(A9,DUGOUTB) |
| | INNING(1) | INNING(1) | INNING(1) |

Figure 4a: Partial raw, prefiltered snapshots

REDUCED SNAPSHOTS:

| TIME: | 14 | 15 | 16 |
|---|---|---|---|
| | HOLDOBJ(A1,BALL,PM) | THROW(A1,BALL,PM) | AT(A1,PM) |

Figure 4b: Remaining primitive descriptor units after snapshots are filtered by attention mechanism.

This filtered data must now be further structured. The continuous action stream must be parsed into relatively small chunks, much like words in a paragraph are chunked into phrases or sentences. The heuristic to perform this task is suggested by the following observations of an action-oriented environment (game): (1) a flurry of activity often indicates that some cohesive process is taking place (competition), while (2) relative calm often indicates the completion of that process (resolution of competition) and the lull prior to the next spate of possibly relevant activity (another round of competition). This crude heuristic does partition snapshots into meaningful chunks. A semantic routine during a later phase will sharpen the boundary points of the activity.

In order to notice repetition of relatively similar sequences of events, generalization over various parameters of the action descriptor must be performed. For example, implicit in finding repetitive events is a generalization over absolute time. By using the *generalization operators* (figure 5), the system can abstract repetitive subsequences of actions within scenarios. This permits the system to build up, in a hierarchical structure, complex sequences of events into more complete scenarios. Then, instead of seeing isolated actions, the system can eventually perceive these complex sequences as if they were single action units, e.g., perceiving a batter's hitting and running as simply a 'hit' or a fielder's catching the ball and throwing it as a 'fielding play.'

| | |
|---|---|
| Original descriptor unit: | THROW(A1, FROM-PM, BALL) |
| Generalization of <u>descriptor unit</u> | |
| Person Operator: | THROW(ANY-PERSON, FROM-PM, BALL) |
| Place Operator: | THROW(A1, FROM-ANYPLACE, BALL) |
| Person and Place Operators: | THROW(ANY-PERSON, FROM-ANYPLACE, BALL) |

Figure 5: Syntactic generalization operations

NOTE: There is implicit generalization over time.

## HYPOTHESIS GENERATOR: PLANS

The major goals of the competitors in an action-oriented game can be expressed (roughly) as follows:

GOAL: Both teams are trying to win. A team can win only by 'scoring' more than the opposing team. This implies both offense and defense.

GOAL: The players on each team try to help members of their own team and try to hinder members of the opposing team from achieving their goals.

We characterize competition in the following way: (1) acts that are considered competitve often require a medium to high degree of skill and/or energy; (2) causal relationships that link the actions of opposing teams will highlight subgoals. They are also the basis for determining the successes and failures of the teams with respect to those subgoals. Examples of such causal relationships, called *CAUSAL-LINK SCHEMAS* (sometimes denoted CLS) are given in figure 6.

| CAUSAL-LINK SCHEMA & TRIGGERING CONDITIONS | HYPOTHESES MADE |
|---|---|
| PHYSICAL-CONFLICT (P-CONFLICT)<br>a. Action ACT1 executed by P1 was the direct physical enabling condition for action ACT2 executed by P2<br>b. P1 and P2 are on opposing teams<br>c. DIFFERENTIAL-ANALYSIS returns T by finding some way that P1 could have performed ACT1 so as to (decrease) the likelihood of P2 executing ACT2 | a. P1 did not intend that P2 execute ACT2<br>b. P1 failed to prevent P2 execute ACT2<br>c. P2 intended to execute ACT2<br>d. P2 succeeded |
| PHYSICAL-COOPERATION (PHYS-COOP)<br>a. same as a. above<br>b. P1 and P2 are on the same team<br>c. same as c. above, except substitute (increase) | a. P1 intended to help P2<br>b. P1 succeeded if P2 succeeded, but not necessarily conversely |
| LOGICAL-COOPERATION (LOG-COOP)<br>a. NOT a. above, yet ACT1 must precede ACT2<br>b. P1 and P2 are on the same team, and may in fact be the same person | a. execution of ACT2 required the execution of ACT1<br>b. P1 intended to execute ACT1<br>c. P1 succeeded |
| RELATIVE-TIME (REL-TIME)<br>a. P2 executed ACT2 after P1 executed ACT1, and ACT1 and ACT2 are not linked by physical enabling conditions<br>b. P1 and P2 are on opposing teams<br>c. there exists actions ACT1* and ACT2* that DIFFERENTIAL-ANALYSIS says could have permitted P1 and P2 to execute acts ACT1 and ACT2 sooner | a. P1 was allowed to execute ACT1 because P1 executed ACT1 before P2 executed ACT2<br>b. P2 intended to execute ACT2 before P1 executed ACT1<br>c. P1 succeeded<br>d. P2 failed |
| LOGICAL-CONFLICT (L-CONFLICT)<br>A. Change of Action<br>a. P2 changed from executing ACT2 to ACT2'<br>b. P1 executed ACT1 concurrently with ACT2<br>c. P1 and P2 are on opposing teams | A. Change of Action<br>I. a. P1 succeeded by executing ACT1 for some goal which forced P2 to execute ACT2'<br>b. P2 did not intend to execute ACT2'<br>c. P1 succeeded<br>d. P2 failed<br>or, II. a. P2 succeeded by executing ACT2 for some goal, and therefore intended to execute ACT2'<br>b. P1 failed to do something which could have prevented P2 from succeeding<br>c. P2 succeeded<br>d. P1 failed<br>or, III. a. P2 succeeded by executing ACT2 for some goal, and therefore intended to execute ACT2'<br>b. P1's ACT1 is NOT causally linked to the actions of P2, i.e., ACT1 and ACT2 are independent |

Figure 6: List of CAUSAL-LINK SCHEMAS

All the hypotheses for a CAUSAL-LINK SCHEMA are asserted when all the triggering conditions are satisfied.

In the analysis of the scenario passed to it by the Attention Mechanism, Hypothesis Generation proceeds by first applying the appropriate ACT-SCHEMA to each action descriptor unit in the scenario. This process establishes the precondition links for the action. It is followed by the application of the CAUSAL-LINK-SCHEMAS (e.g., PHYSICAL-CONFLICT, RELATIVE-TIME, PHYSICAL-COOPERATION, LOGICAL-COOPERATION, etc.). These demon-like routines (Charniak 1972) search for sequences of actions that satisfy conditions specified in each CLS. Meeting of the conditions implies that the causal relationship specified by the particular CLS *may* exist between those actions. Once triggered, they make hypotheses about the goals of the players and about the success or failure of the players with respect to those goals.

During a third phase, isolated actions and their hypothesized goals are grouped by player and team into *PLANS*. A plan is a very important aspect of human behavior. It permits a coherent *interpretation* of a sequence of actions. Intermediate actions serve as means for attaining subgoals, while subgoals are executed in order to achieve the final goal (often indicated by the last action).

Let us illustrate the application of the CAUSAL-LINK SCHEMAS. Consider the scenario in Figure 7, which depicts an infield single. In the analysis of action descriptor #6 (the SWINGHIT by player B1), the PHYSICAL-CONFLICT SCHEMA finds itself applicable. As a result of the ACT-SCHEMA, the system knows that the THROW by opponent A1 'direct-physically' enabled B1 to hit the ball, and that both actions require a high degree of skill. Before labeling success or failure, PHYSICAL-CONFLICT must be sure that the performance of the player who threw the ball definitely had some effect upon the performance of the action of the hitter. DIFFERENTIAL-ANALYSIS confirms this possibility by accessing data in the ACT-SCHEMA, and accessing general facts about actions in the data base. In this case DIFFERENTIAL-ANALYSIS infers that A1 could have thrown the ball faster by applying an increase of energy and skill. (Note that skill or energy is usually expended by a person whose motive is to achieve an action-oriented goal.) This would have had the effect of requiring a corresponding increment of skill for B1's action. The point is that A1 could have done something to decrease the likelihood of B1's hitting the ball. In this case PHYSICAL-CONFLICT makes the following hypotheses: (1) A1 did not intend that the effects of his actions should allow B1 to hit the ball, therefore A1 failed with respect to his goal; (2) B1 intended to execute the act SWINGHIT, therefore B1 succeeded with respect to this goal.

Consider act descriptor #20 (ON firstbase by B1). The CAUSAL-LINK SCHEMA of RELATIVE-TIME asserts the possibility that the reason B1 was allowed to execute that action was because he did it *before* action descriptor #22 (CATCH ball at firstbase by A3). Again DIFFERENTIAL-ANALYSIS is called to confirm that the time of execution of both acts could have been influenced by a change in the skill or energy expended. Yes, A6 could have made the ball arrive at firstbase sooner if he could have thrown the ball faster; and B1 could have arrived at firstbase sooner if he had run faster. The hypotheses of this CLS are: (1) B1 did intend to execute act ON firstbase, therefore B1 succeeded with respect to his goal; (2) A1 did not intend for B1 to execute that act, therefore he failed with respect to his goal.

| Team A | Team B | Ball |
|---|---|---|
| **(1 1 HOLDOBJ (A 1) PM BALL) | | |
| **(2 2 THROW (A 1) PM BALL)<br>((PHYS-COOP SUCCEED)(PEC 1 2)) | | |
| **(3 3 AT (A 1) PM NIL)<br>((LOG-COOP SUCCEED)(LOGICAL 2 3)) | **(4 3 AT (B 1) HP NIL) | **(5 3 MOVING NIL PM BALL (FAST AIR)) |
| | **(6 4 SWINGHIT (B 1) HP BALL)<br>((P-CONFLICT SUCCEED)(SUCCEED 6)(FAIL 2))<br>((LOG-CONF SUCCEED)(CHANGE-LINK 3 4)) | |
| | **(7 5 RUN (B 1) HP (FAST))<br>((LOG-COOP SUCCEED)(LOGICAL 6 7)) | **(8 5 BOUNCE NIL HP BALL) |
| **(9 6 AT (A 6) TB NIL) | **(10 6 RUN (B 1) HP (FAST))<br>((PHYS-COOP SUCCEED)(PEC 7 10)) | **(11 6 MOVING NIL HP BALL (FAST GRND)) |
| **(12 7 CATCH (A 6) TB BALL)<br>((P-CONFLICT SUCCEED)(SUCCEED 12)(FAIL 6<br>((LOG-CONF SUCCEED)(CHANGE-LINK 10 9)) | **(13 7 RUN (B 1) HP (FAST))<br>((PHYS-COOP SUCCEED)(PEC 10 13))<br>((LOG-CONF SUCCEED)(NOCHANGE-LINK 10 9)) | |
| **(14 8 THROW (A 6) TB BALL)<br>((PHYS-COOP SUCCEED)(PEC 12 14))<br>((LOG-CONF SUCCEED)(CHANGE-LINK 13 12)) | **(15 8 RUN (B 1) HP (FAST))<br>((PHYS-COOP SUCCEED)(PEC 13 15))<br>((LOG-CONF SUCCEED)(NOCHANGE-LINK 13 12)) | |
| **(16 9 AT (A 6) TB NIL)<br>((LOG-COOP SUCCEED)(LOGICAL 14 16))<br>((LOG-CONF SUCCEED)(CHANGE-LINK 15 14)) | **(17 9 RUN (B 1) HP (FAST))<br>((PHYS-COOP SUCCEED)(PEC 15 17))<br>((LOG-CONF SUCCEED)(NOCHANGE-LINK 15 14)) | **(18 9 MOVING NIL TB BALL (FAST AIR)) |
| **(19 10 AT (A 3) FB NIL) | **(20 10 ON (B 1) FB NIL)<br>((PHYS-COOP SUCCEED)(PEC 17 20))<br>((REL-TIME SUCCEED)(SUCCEED 20 FAIL 22))<br>((LOG-CONF SUCCEED)(CHANGE-LINK 16 17))<br>((EOC SUCCEED)(END-OF-COMPETITION)) | **(21 10 MOVING NIL TB BALL (FAST AIR)) |
| **(22 11 CATCH (A 3) FB BALL)<br>((PHYS-COOP SUCCEED)(PEC 14 22))<br>((LOG-CONF SUCCEED)(CHANGE-LINK 20 19)) | | |
| **(23 12 HOLDOBJ (A 3) FB BALL)<br>((PHYS-COOP SUCCEED)(PEC 22 23))<br>((EOC SUCCEED)(END-OF-COMPETITION)) | | |
| **(24 13 THROW (A 3) FB BALL)<br>((PHYS-COOP SUCCEED)(PEC 23 24)) | | |
| **(25 14 AT (A 3) FB BALL)<br>((LOG-COOP SUCCEED)(LOGICAL 24 25)) | | **(26 14 MOVING NIL FB BALL (SLOW AIR)) |
| **(27 15 AT (A 1) PM BALL)<br>((LOG-CONF SUCCEED)(NOCHANGE-LINK 3 4)) | | **(28 15 MOVING NIL FB BALL (SLOW AIR)) |
| **(29 16 CATCH (A 1) PM BALL)<br>((PHYS-COOP SUCCEED)(PEC 24 29)) | | |
| **(30 17 HOLDOBJ (A 1) PM BALL)<br>((PHYS-COOP SUCCEED)(PEC 29 30)) | | |

Figure 7: Infield Single Scenario with Asserted Hypotheses

This listing reflects only successful CAUSAL-LINK SCHEMAS; not shown are the reasons returned when they fail. The syntax of the action descriptor unit is:

(descriptor-number snapshot-number action actor location modifiers)

The columns represent all actions of a team or object, while rows reflect concurrent actions (equal snapshot numbers).

Note that the two CLS's above 'perceive' competition on two different levels. PHYSICAL-CONFLICT actually observes the physical interaction between the actions of the players, while RELATIVE-TIME must posit the existence of a relationship between the actions of the players. Of course, both require that their respective relationships exist between members of opposing teams. PHYSICAL-CONFLICT deals with a specific feature of the physical environment, e.g., skill and energy, while RELATIVE-TIME deals with a specific feature of the logical environment, e.g., time precedence as a relevant relationship.

There may be additional features in either (or both) levels that the system cannot directly perceive, but which are nonetheless important to the specific game under observation. For example, since the system does not perceive the specific placement of the ball as it passes over the homeplate, it will not be able to perceptually distinguish between a 'called strike' and a 'ball'. (To be a 'called strike', the ball must pass over homeplate at a height somewhere between the batter's shoulders and his knees.) The general CAUSAL-LINK SCHEMA, LOGICAL-CONFLICT, deals with this type of situation. This general CLS looks at the changes (and non-changes) in players' actions, and tries to explain those changes (or non-changes) in terms of the existence of some causal relationship, even though it cannot directly perceive one. This decision is based on an understanding of the kinds of causal links that might be necessary to explain a player's actions. With this ability, the system has a flexible and powerful technique for dealing with novel situations.

Generation of plans will ultimately require the binding of many of the local hypotheses distributed across the action scenarios. The system must find interlocked and globally consistent subsets of inferences which might explain the observed situation. For example, during the PLAN building phase, another hypothesis generator, EOC, attempts to find the end of a competitive epoch so it can highlight the final goals of the two teams. In figure 7, EOC hypothesizes that RUN-ON and CATCH-HOLDOBJ are the last competitive acts of the two teams. A process called PLAN-BUILD then backs up the final goals and attempts to relate them--the SWINGHIT of B1 was executed in order to enable B1 to execute the act ON firstbase. Similarly, the A team's goal now has become: prevent B1 from executing ON firstbase. We are presently investigating other more dynamic techniques to assist in this analysis and transformation of local hypotheses into globally consistent plans.

This is only the first stage of hypothesis generation. We have not used all the concepts from action-oriented gaming. For example, we have not as yet made hypotheses about what counts as scoring, or what counts as failed opportunities to score. Nor have we started to keep track and tally up these kinds of actions, usually an important facet of scoring. Nor have we introduced the high information cue of spectator cheering. This latter stage of hypothesis generation will build on the hypotheses made so far, but will have to wait until after the next phase, hypothesis verification and generalization, where evidence will be gathered to support or reject those earlier hypotheses. Note, however, that our model of knowledge acquisition, using a general description of action-oriented gaming, has already moved from the perception of actions to the possible goals intended by those actions.

## HYPOTHESIS VERIFICATION AND GENERALIZATION: PLAN-SCHEMAS

Exemplar learning models (e.g., Winston 1970, Sussman 1973, Hayes-Roth 1976), usually have the following characteristics. First, they may require a partially ordered training sequence with presentations of positive and negative instances of the class in order for the desired concepts to be properly learned. Next, such models require that the system be told to which class an instance belongs. This is usually done either explicitly by associating the class name with the presentation, or implicitly by requiring that the trainer present examples of only one class at a time. Third, the set of relationships used in generalization is basically the same as used in the examples. Finally, a local similarity measure (e.g., frequency of occurrence) relating examples of a class is used to define a generalized class description.

However, abstracting regularities of human behavior by simply observing that activity in a natural setting requires a more sophisticated model of unsupervised learning. The complexity of our problem domain requires an extrapolation of the above issues in the following ways. First, natural experience is often a fickle teacher. A model that learns from experience must be flexible enough to accept an unordered training sequence and impose its own order. Second, in a new experiential domain, the system cannot expect to know or be told to which class an example belongs; it must be able to *infer* the classes, using both a priori knowledge of what could count as a class type and the observations of specific examples. Third, given the multiplicity and non-specificity of features in any given real-world situation, a priori semantic knowledge is required in order to hypothesize the existence of higher level relationships that serve to highlight relationships that are important to a specific interpretation. For example, in our domain, the CLS posit the existence of relationships that are important to interpreting that activity in the context of action-oriented games. Other relationships would need to be hypothesized if the system were trying to interpret that activity as a religious ceremony. The above properties characterize the 'experiential model of learning' employed in the present system. We believe that it examines issues underlying human developmental learning that previous systems have not addressed.

In order to first generate classes and then generalize within those classes, two types of similarity measures are required. A more global one that can partition examples into classes, and another more local one that can abstract the important characteristics within a class. The global criteria under which we have chosen to group scenarios stems from a simple but powerful observation: events that begin the same but end differently, events that begin differently and end the same, and events that begin and end the same but have different middles--are cues to the structure of the general scenarios which govern those situations.

Using these principles the system will be able to group together a set of scenarios that will eventually be labeled as infield singles, and a set of scenarios that will eventually be labeled as infield groundouts. Comparing these two groups using the 'begin the same, end differently' heuristic, we note that they do only differ at the end. Looking at the hypothesized interpretations in each group, we saw that the team which was labeled as having succeeded in one group is the same team that was labeled as having failed in the other group (figure 8). This result not only lends support to a correct partitioning of the scenarios, but also supports the correctness of the hypotheses.

A powerful input during this phase of analysis comes from the CAUSAL-LINK SCHEMAS themselves. If their hypotheses about the observed actions are correct, then they should expect to see action sequences in which the team that was previously hypothesized as being successful should now fail, and vice versa. In fact, predictions are made of precisely these complementary outcome conditions, and prime the Attention Mechanism via feedback. From our example (figure 7), PHYSICAL-CONFLICT will predict that A1 will throw the ball, and some B team member will not hit it. RELATIVE-TIME will predict that A3 will catch the ball before some B team member reaches firstbase, and that some B team member will not be permitted to execute the act 'ON firstbase'.

The convergence of both techniques to similar conclusions is strong evidence for the correctness of the hypotheses and the partitioning. Finally, just as the Attention Mechanism is building up generalized, repetitive subsequences of actions, this stage of the system will generalize PLANS into *PLAN-SCHEMAS*, based on its grouping of scenarios. These latter structures will be the final output from the system. They will represent both an understanding of the goals and intentions involved in the scenario, and also the rules or regularities observed in the scenarios.

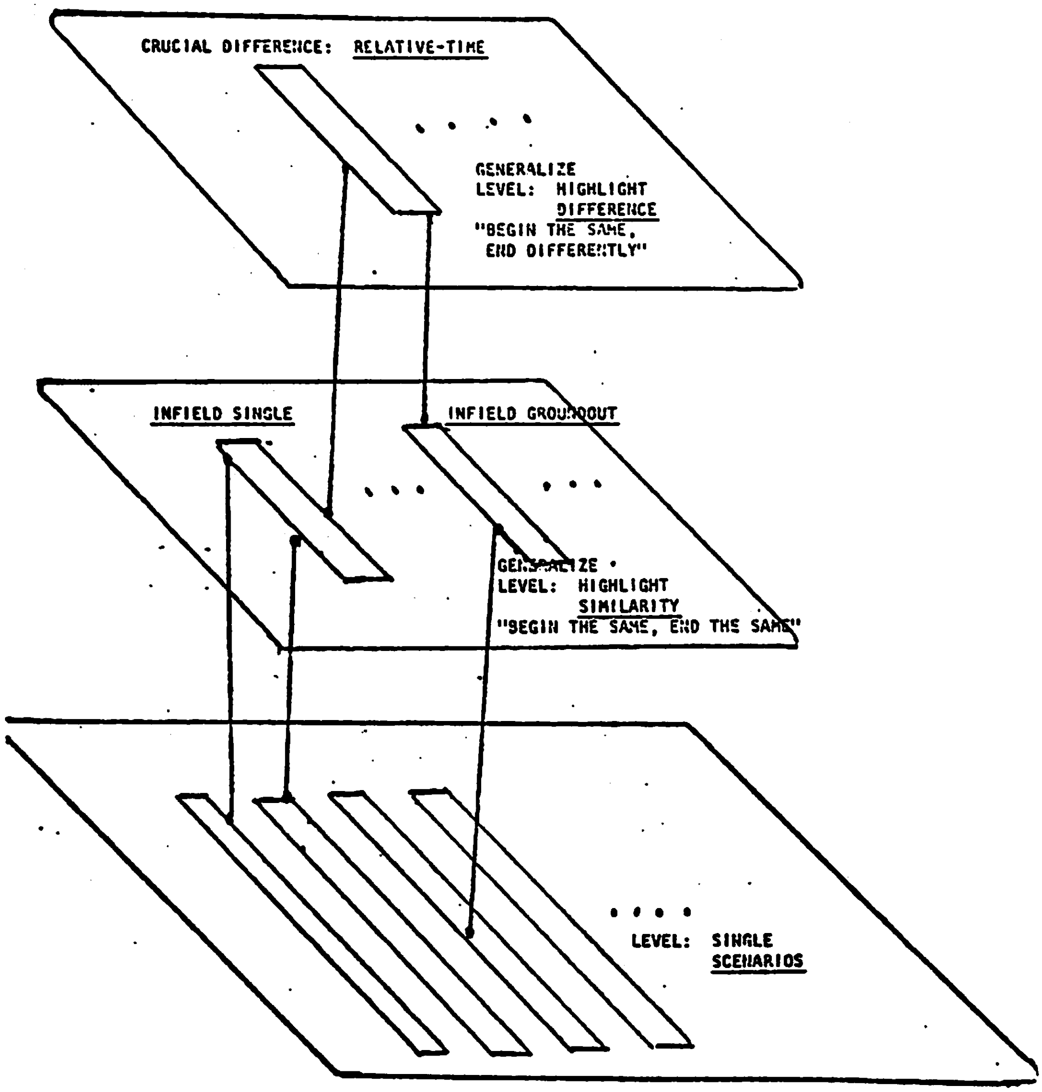


**Figure 8: Hierarchical Generalization of Scenarios**

## SUMMARY

The implementation of the Attention Mechanism was done in SNOBOL4. The output from this subsystem is used by the Hypothesis Generation phase which is written in LISP. In the analysis of a typical scenario, this program uses approximately 27K of the CDC6600 and requires about 30 seconds of processor time. A preliminary version of the Hypothesis Generalization and Verification system is presently being built and tested. Results from the Hypothesis Generation part of the system confirm its inferential power, while early results from the hypothesis generalization part of the system are also encouraging.

What we have explored is a way in which a system can add to its present knowledge base an understanding of some new situations. Though initially driven by the perception of knowledge about actions, the system develops a consistent structure of hypotheses about the goals of the actors. This conceptual representation permits the system to abstract the commonalities from a multiplicity of somewhat varying situations. This generalization process provides the general rules governing behavior in the observed situations of this environment.

## ACKNOWLEDGEMENTS

This work has been aided by the stimulating dialogue within the Cybernetics group at the University of Massachusetts. The many discussions with Chuck Schmidt of Rutgers University has served to significantly sharpen our understanding of the issues. We also wish to thank W. Kilmer, M. Arbib, J. Lowrance, and D. Lawton for helpful comments during the preparation of this paper.

## REFERENCES

Arbib,M.A. (1975) Two papers on schemas and frames. Univ. of Mass. COINS Technical Report 75C-9, Amherst.

Charniak,E. (1972) Toward a model of children's story comprehension. MIT AI Lab Technical Report AI-TR266, Cambridge.

Hayes-Roth,F. (1976) Uniform representation of structured patterns and an algorithm for the induction of contingency-response rules. *Information and Control*, in press.

McCarthy,J. & P.Hayes (1969) Some philosophical problems from the standpoint of artificial intelligence, in *Machine Intelligence 4* (ed D.Michie). Edinburgh: Edinburgh Univeristy Press.

Rieger,C. (1973) The common-sense algorithm as a basis for computer models of human memory, inference, belief and contextual language comprehension. Theoretical Issues in Natural Language Processing, June, Cambridge.

Rumelhart,D.E. (1975) Notes on a schema for stories, in *Representation and Understanding* (eds D.Bobrow & A.Collins). New York: Academic Press.

Schank,R.C.(1974) Understanding paragraphs. Instituto per gli Studi Semantici e Cognitivi, Technical Report 5, Castognola, Switzerland.

Schank,R.C. & R.P.Abelson (1975) Scripts, plans and knowledge. Advanced Papers *IJCAI 4*, USSR.

Schmidt,C.F. (1976) Understanding human actions: recognizing the plans and motives of other persons, in *Cognition and Social Behavior* (eds J.Carroll & J.Payne). Erlbaum Press, in press.

Schmidt,C.F. & J.L.Goodson (1976) Summarizing observed actions: a structural hypothesis about the nature of summaries. Dept. of Computer Science Technical Report CBM-TR62, Rutgers Univ., New Brunswick.

Schmidt,C.F. & N.S.Sridharan (1976) The representation of plans. Dept. of Computer Science Technical Report CBM-TR61, Rutgers Univ., New Brunswick.

Soloway,E.M. & E.M.Riseman (1975) Common-sense theory formation: towards understanding baseball. Univ. of Mass COINS Technical Report 75C-5, Amherst.

Sridharan,N.S. (1976) The architecture of BELIEVER, PART II: the frame problem. Dept. of Computer Science Technical Report CBM-TR47, Rutgers Univ., New Brunswick.

Sussman,G.J. (1973) A computational model of skill acquisition. MIT AI Lab Technical Report TR-297, Cambridge.

Winston,P.H. (1970) Learning structural descriptions from examples. MIT Project MAC Technical Report 76, Cambridge.