Control Knowledge in POPLAR, a Personality-Oriented Planner.

Sergei Nirenburg, Irene Nirenburg and James H. Reynolds

Colgate University, Hamilton, NY 13346

ABSTRACT

This paper presents an overview of a personality-oriented planner, and then describes in detail the types of knowledge it uses to make control decisions. POPLAR is a model of an intelligent actor capable of planning sequences of control and domain actions in a simulated world that exists independently of the planner. The world is a simplification of the "Dungeon" computer game environment. The actor makes control decisions on the basis of situational knowledge as well as its personality characteristics (character traits, physical and mental states) and its beliefs about personality of other cognitive entities in the world. POPLAR is a step toward an AI system whose behavior is psychologically justified and can provide the basis for an experimental testbed in cognitive modeling. POPLAR is implemented as described in PEARL on a Vax/Unix system.

Making control decisions in AI systems more knowledge-dependent is a general trend in the field. Among the crucial choices in the control problem are:

a) what types of knowledge to use: e.g., knowledge about the problem-solving situation, about the actor's goals, or both;

b) how control knowledge is to be represented: e.g., procedurally vs. declaratively; using metalevel knowledge as well as domain knowledge, etc.;

c) what decision procedures to use: e.g., choosing between static evaluation functions or dynamic lookahead strategies for scheduling, or using both.

POPLAR is a model of an intelligent actor that

a) makes control decisions using knowledge about its goals, plans, processing history and personality in addition to knowledge about situation;

b) uses both domain and metalevel plans, and encodes them in a frame-based language (PEARL);

c) uses static evaluation of alternatives to guide processing; the processor treats domain and metaplan similarly.

These choices were made on the basis of the following 3 interrelated hypotheses:

1. in everyday situations humans do not tend to consciously perform lookahead in choosing a goal to pursue and a plan to achieve that goal;

2. humans do not tend to spend a lot of energy considering how the choice of a plan will affect the eventual processing of other plans (competing or cooperating) that can be instantiated in the system;

3. humans strive to minimize the expenditure of not only the physical but also the mental resources available to them. The concrete minimizing function is determined by the personality and mental and physical state parameters of the intelligent actor.

POPLAR consists of two components: a model of an intelligent actor and a model of an objective world. In the current Vax/Unix implementation, the world is a simplified environment of the "Dungeon" computer game. In it POPLAR's Actor finds and reacts to enemies, treasures, tools, weapons, food and other objects. Actor has a personality -- a set of beliefs, character traits, etc. -- that influences its reactions. The human user, taking on the role of the "objective world", can experiment with the model by supplying it with
external-world inputs and by permitting or forbidding certain primitive operations to simulate Actor's pragmatic experience. The user can also vary Actor's personality. (Note that this dichotomy between Actor as model and User as world reverses the roles of human and computer in "Dungeon"-like games.)

POPLAR's world is simulated by the user with the help of the world blackboard, WBB, cf. Figure 1. The user adds new inputs to WBB for Actor to "perceive". WBB also contains a clock that guides POPLAR's time-related activities, such as automatic updates to the level of Actor's hunger.

Actor consists of three major modules: the cognitive processor (CP), the short-term memory (STM) and the long-term memory (LTM).

CP consists of three components: the input/output monitor, the executor and bookkeeping demons. It is an infinite loop that blindly executes the highest-rated plan on Actor's agenda. We assume CP is itself non-cognitive. All knowledge necessary for making control decisions is stored in plans themselves.

Activated instances of domain plans are stored in STM, in the AGENDA and CURRENT-PATH slots of its Actor blackboard (ABB) component. In addition, ABB contains knowledge about activated object instances. STM also contains the single-instance metaplans: the scheduler and the three goal-generator plans (cf. Fig. 2). All of this knowledge is directly used for control.

Actor's LTM contains:

-- Actor's beliefs about physical and mental OBJECTS in the world; entries for itself and other cognitive entities include data on personality: character traits, physical and mental states;

-- the three basic GOALS used by Actor: a) "Don't get killed", Preserve-Self-1 (PS1); b) "Don't die of hunger, thirst or fatigue", Preserve-Self-2 (PS2); and c) "Collect as much treasure as possible", Get-Treasure (GTR);

-- a self-decaying history of processing where past states and decisions are recorded and used as an additional type of control knowledge;

-- Actor's knowledge of plans for achieving the above goals. (In the absence of a powerful learning mechanism it is irrelevant whether the dynamic knowledge in an AI system is represented as (backward) subgoaling knowledge or (forward) planning knowledge that uses "canned" parametrized plans. Subgoaling alone (cf. e.g., Barber, 1983) will not lead to "creative" planning, i.e. finding new ways of achieving same goals. Actor must consciously decide to look for an alternative solution. And that requires additional knowledge.) Plans are encoded using a modified version of Event Description Language EDL (Bates et al., 1981).

To highlight control knowledge it will suffice to describe the slots "IS", "RATING-FUNCTION" and "OPTIONAL" in a POPLAR plan frame.

The IS slot contains the temporal and causal expansion of the plan into lower-level plans. The IS slots of all the plans in the system form a grammar (cf. Figure 2). Generating a path through this grammar requires control decisions in the cases of a) disjoined and b) optional constituents.

The RATING-FUNCTION slot contains knowledge used to make decisions in cases of disjoined constituents. In these plan choice points CP executes the scheduler. The main action of the scheduler plan is to rate candidate plans by executing the code in their respective RATING-FUNCTION slots. The top-rated plan is chosen for execution.

Rating functions use both situation and personality knowledge. For example, the knowledge used to rate the plan Fight is the following ratio:

\[(\text{div} \\text{times adversary.weapon-against.efficiency} \\text{actor.courage} \\text{actor.power} \\text{adversary.injury} (\text{expt actor.aggression 2})) \\text{times} (\text{calculate-fear actor adversary}) \\text{adversary.power} \\text{actor.injury} \text{adversary.fearsomeness} \text{actor.fatigue})\]

where, e.g. adversary.power is Actor's belief about the current adversary's power, stored in the corresponding object frame in STM; calculate-fear is an auxiliary function of 10 arguments that produces the value of Actor's fear of this particular adversary; adversary.fearsomeness is Actor's acquired value for fear against adversaries of this type (for instance, Actor1 may be afraid of snakes and Actor2 may not).
The choice of character traits has been an empirical process of gradual refinement. However, in parallel with the implementation of POPLAR, extensive psychological experiments are being conducted (cf. Reynolds & Nirenburg, in preparation) seeking to establish a set of "primitive" personality characteristics and their mapping into more complex notions that are used in rating and control functions.

Knowledge to handle optional subpaths is stored in the OPTIONAL slot of the plan frame. The type of knowledge used here is comparable to that used by the rating functions.

Consider now a simplified example of the use of control knowledge in POPLAR. Suppose the present situation involves Actor, a snake, a troll, a gold nugget, a stick and a sword. ABB contains the GG plans, which are executed by CP, as a result of which all of the above object instances are added to ABB.objects and the top-level plan instances PS1(snake1), PS1(troll1) and GT1(gold-nugget1) are added to agenda in ABB. The scheduler is then executed, producing ratings for the above plan instances.

Suppose that PS1(troll1) obtained the highest rating and is selected for execution. To select one of the possible PS1 expansions (Fight, Hide or Wait-and-See), the scheduler is run again. Suppose that Fight is picked. The expansion of Fight has the following semantics: i) look for a weapon in your inventory; if none, or if the weapon is not good enough, ii) look for one around you; if none, abort the whole Fight; if found, iii) get it; iv) move within the range of the weapon from the adversary; v) attack. Actor believes that swords (but not e.g. fists!) are weapons against trolls. Step i) fails because Actor's inventory is empty. The OPTIONAL predicate in Fight returns "true", and the optional path ii) and iii) is executed. The sword is found, which leads to the success of the Get and hence of the optional path. Finally, the move and attack plans are performed.

In the present implementation these primitive plans are executed by having Actor ask permission of the User to perform them. In this way the User can witness Actor's cognitive processes, and force it to select an alternative plan of action. Changes in the world introduced through the execution of permitted plans are then made by the bookkeeping demons.

None of the theoretical or design decisions in POPLAR were made without the comparison with, and often influence of, previous related work. A very limited survey follows.

The use of character traits has been discussed in AI literature, though, unlike POPLAR, none of the approaches has made personality a crucial source of control knowledge. Carbonell (1979) suggests the use of personal goals in understanding stories. Slocan & Croucher (1981) discuss the role of motives, moods, attitudes and emotions in natural and artificial intelligent systems, without incorporating this knowledge in a system.

We agree with Hayes-Roth (1984) that the control problem is a real-time planning problem. POPLAR achieves or is in principle capable of achieving most of the "behavioral" goals that Hayes-Roth sets for the control capability of an AI system. The point where POPLAR differs is the absence of the dynamic lookahead as a source of control decisions, cf. Hypothesis 1. POPLAR shares Hypothesis 2 with Carver, Lesser and McCue (1984), but differs from Wilensky (1983) and Hayes-Roth (1984) on this point. Extensive experimentation with POPLAR will show whether our hypotheses have been sound. We believe that POPLAR's world, being different from that in, say, Hayes-Roth and Hayes-Roth (1979), will require the different treatment we suggest.

POPLAR generates and executes a relatively small number of plans in a knowledge-rich, though simulated, environment. Experimentation with the current version of POPLAR aims at a) enriching Actor's beliefs; b) developing, through psychological experimentation, a compositional calculus of personality characteristics (cf. Reynolds & Nirenburg); c) calibrating the rating functions; d) extending the use of history in control decisions.

The more far-reaching possibilities of development include (though are not restricted to) a) adding plan understanding to plan production, which necessitates b) the introduction of verbal behavior into POPLAR; c) making POPLAR a testbed, with multiple cognitive actors, for modeling "real-life" situations of cooperation, conflict, etc. Work on these topics has been started (cf. Nirenburg & Pustejovsky).
References.


Figure 1. A schematic view of POPLAR architecture.

I ::= PS1 | PS2 | GTR | GG ("Goal-Generator") | S ("Scheduler")
PS1 ::= FIGHT | HIDE | WS ("Wait-and-See")
PS2 ::= EAT | DRINK | SLEEP
GTR ::= (FIGHT | find) GET
GG ::= gg-input | gg-objects-perceived | gg-physical-states-perceived
FIGHT ::= find (find GET) move attack
HIDE ::= find move
WS ::= do-nothing
EAT ::= find (find GET) ingest
DRINK ::= find (find GET) ingest
SLEEP ::= find do-nothing
GET ::= move take

Space is a concatenation symbol. Vertical bars separate disjoined plans (the one whose rating function returns highest value is selected in processing). Curly brackets enclose optional paths. Plans shown in lower case are primitive.

FIGURE 2. A grammar of plans in POPLAR.