From: AAAI-80 Proceedings. Copyright © 1980, AAAI (www.aaai.org). All rights reserved.

ON WAITING

Arthur M. Farley
Dept. of Computer and Information Science
University of Oregon
Eugene, Oregon

ABSTRACT

Waiting is the activity of maintaining selected aspects of a current situation over some period of time in order that certain goal-related actions can be performed in the future. Initial steps toward the formalization of notions relevant to waiting are presented. Conditions and information pertinent to waiting decisions are defined.

Introduction

Waiting is the activity of maintaining selected aspects of a current situation over some period of time in order that certain goal-related actions can be performed in the future. Those aspects of the current situation which are maintained serve as preconditions for the desired future actions. Waiting extends from a decision to maintain these preconditions until (attempted) execution of the goal-related actions (or a decision to forego them). The act of waiting is closely associated with the future actions; a problem solver is said to be "waiting to" (do) these actions.

The primary function of waiting is to improve the efficiency with which problem solving plans are executed. Waiting attempts to overcome reestablishment of preconditions for desired, future actions. Not only may less effort be expended during problem solving, but performance can become more coherent. The problem solver can avoid frequent side- and back-tracking in multiple-goal situations. Waiting also allows for a certain degree of parallelism in problem solving. Waiting can be engaged in simultaneously with other actions which do not destroy satisfaction of the preconditions being maintained.

Waiting is an important and frequent problem solving activity in the real-world. By real-world, we mean an ongoing, continuing, schedule based context, within which cooperative, as well as competitive, efforts among groups of problem solving systems normally occur. Waiting minimally requires such a context to be effective. A decision to wait implies that other, as yet unsatisfied, preconditions of anticipated goal-related actions are expected to be met by means other than direct intervention by the waiting system.

Waiting as a problem solving activity has been largely (if not totally) ignored by AI research to date. This is primarily due to the fact that real-world contexts as defined here have only recently been considered. This paper outlines initial steps toward formalisms within which issues of waiting can be addressed. The research represents extensions to a knowledge-based problem solving system previously described by the author [3,4]. We briefly review important aspects of that system before describing straightforward extensions which aid our understanding of waiting. We conclude by discussing related research and suggesting future work.

Knowledge-based Problem Solving

The form of its representation of the environment influences all other aspects of a problem solving system. Let a situation of the relevant environment at any point in time (past, present, or future) be represented by a situation state. A situation state consists of a finite set of propositions which are true with respect to the specific environmental situations(s) which the state is said to represent. The current state represents the present environmental situation. A goal state is a situation state which the problem solving system desires the current state to satisfy (i.e. be consistent with). A problem exists for a problem solving system when the current state does not satisfy constraints specified by a goal state. Problem solving refers to any activity undertaken in attempts to eliminate differences between current and goal states. A problem is solved when differences between its goal state and the current state no longer exist.

A knowledge-based problem solving system solves most problems by instantiation and execution of known general solution plans. A general solution plan describes a process which is capable of satisfying a set of goal states from any of a set of current states. A general solution plan is represented as a rooted, directed, labelled tree of plan states. The root of the tree is the goal state. Plan states of the tree are interconnected by directed arcs, each labelled by an operator. An operator is a description of an action (or process), represented as sets of add, delete, and precondition propositions [7]. The operator labelling an arc is capable of transforming the plan state at the tail of the arc into the plan state at the head of the arc. The plan state at the tail satisfies preconditions of the operator, while the one at the head reflects the results of additions and deletions associated with the operator. The maximal directed path from any plan state ends at the goal state.

A plan state is a situation state whose propositions have been partitioned into three components. For each proposition in the SELFOP component, the problem solving has one or more operators capable of satisfying the proposition. Furthermore,

the system normally expects (prefers) to satisfy a SELFOP proposition itself by executing one or the operators. For each proposition in the OTHEROP component, (costly) operators may exist allowing the system to satisfy the proposition itself, but the system normally expects the proposition to be satisfied by other problem solving systems in the environment. Finally, for each proposition in the NOOP component, neither the system itself nor any other problem solving system has control over satisfaction of the proposition (i.e., weather conditions).

Though the structure of a general solution plan is made clear by its tree, representing the plan as a production system can facilitate its execution. A production system [6] is a collection of condition-action pairs called rules. In the production system representation of a general solution plan, each plan state serves as the condition part of a rule whose action part is the operator labelling the arc leaving that plan state. In [4], the author describes how this representation can be useful in a selective approach to coordinating the execution of multiple plans. Rules from general solution plans associated with as yet unsatisfied goals are combined to form one production system. A rule classification scheme is defined which provides for the avoidance of most inter-plan conflicts while allowing for responsive, efficient problem solving. For example, a goal state is classified as ungrounded if it denies satisfaction of conditions from a (critical) rule of another plan (in a conjunctive goal set). Rules from plans for ungrounded goal states are not executed until conflicting plans have been completed.

Waiting

We are concerned here with the control of waiting during plan executions by knowledge-based problem solving systems operating in real-world contexts. We first consider the question: what situations trigger consideration of waiting? A rule is self-satisfied if all propositions in its SELFOP component are satisfied. Whenever a self-satisfied rule exists during plan execution, a problem solving system may consider waiting, maintaining those conditions of the rule it has satisfied until the rule can be executed. A rule whose conditions are completely satisfied, and thus can be executed during a current system cycle, is classified as ready. Ready rules can suggest waiting. The system may wait to execute one ready rule, selecting another which leaves the rule ready (through conflict resolution).

During plan execution, a rule corresponding to a plan state whose NOOP component is not satisfied is classified as irrelevant. Such a rule can only fire if uncontrollable aspects of the environment change favorably. An irrelevant rule which is otherwise satisfied is a contingent rule. Being self-satisfied, contingent rules suggest consideration of waiting. Any NOOP condition must be satisfied occasionally, otherwise a rule would be fantasy and not considered part of a problem solving system's executable plans.

An important class of NOOP propositions are those which deal with time. In a general solution plan reflecting real-world time constraints (a scheduled plan), the NOOP component of a plan state will contain one or both of the propositions ISNOWBEFORE(t) and ISNOWAFTER(t), where t is a time specification. The truth of a time proposition is determined relative to the current time, considered to be an ever present aspect of the current state. Time propositions of plan states derive from time constraints placed upon goal state satisfaction. They are propogated to other states (rules) of a scheduled plan, with time parameters adjusted to reflect time estimates for traversed operators.

In a scheduled plan, time propositions alone may determine the relevancy status of rules. Rules which are irrelevant solely due to unsatisfied time constraints have their NOOP components classified as early or late, depending upon which time proposition is not satisfied. A rule with early NOOP component, but which is otherwise ready, is classified as imminent. Whenever an imminent rule exists during plan execution, the problem solving system may consider waiting while time passes until the rule becomes relevant and can be fired. One source of imminent rules are overestimations of time requirements for completed, prior operations from a scheduled plan. For example, you estimate a 30 minute drive to a shopping mall; but it only takes 15; you arrive before the mall opens. The rule by which you would enter the mall is imminent.

Another situation which may prompt waiting is the existence of a dependent rule. A relevant, self-satisfied rule corresponding to a plan state whose OTHEROP component is not satisfied is classified as dependent. Whenever a dependent rule exists during plan execution, the system may consider waiting until the expected (necessary) assistance arrives and the rule can be fired. Dependent rules can often arise during cooperative problem solving efforts. For example, you are painting a house with a friend and turn around, expecting him to hand you a new can of paint, but he hasn't finished opening it. The rule by which you would take the paint is dependent.

Finally, rules which have only their SELFOP component satisfied are classified as needy. Needy rules arise frequently when coordinating with public, problem solving support systems, such as mass transportation. To board a bus, a problem solver must be at a bus stop (SELFOP), at the scheduled time (NOOP), with the bus being at the stop (OTHEROP). If a problem solver arrives at a stop five minutes early, the rule by which it boards the bus is needy.

Rule classifications which trigger consideration of waiting are summarized in Table I.

TABLE I: Waiting Rule Classifications

Class	Component SELFOP	OTHEROP	NOOP
self-satisfied	S	-	-
ready	S	S	S
contingent	S	S	N
imminent	S	S	E
dependent	S	N	S
needy	S	N	N

Component Status: S-satisfied, N-not satisfied, E-early.

Given the existence of a self-satisfied rule, what information is pertinent to subsequent waiting decisions? With each self-satisfied rule, the problem solving system can associate three values: a waiting period, a set of compatible rules, and a set of contending rules. The waiting period is an estimate of the length of time before a self-satisfied rule will become ready. A compatible rule is a ready rule which requires less (estimated) time than the waiting period, and would not destroy satisfaction of the self-satisfied rule's conditions. A contending rule is a ready rule from another scheduled plan which would destroy the other's self-satisfaction, requires more time than the waiting period, but will become classified late if it is not fired within the waiting period. Determining these two sets of rules would not require dramatic additional computational effort. They only contain ready rules, rules which the system always determines before selecting the next rule to execute.

A simplest policy for waiting can be stated in terms of these two sets of rules. If there is a contending rule associated with a self-satisfied rule, do not wait; otherwise wait, firing a compatible rule, if any exist. Though this may produce effective behavior in many circumstances, a moment's thought suggests further considerations. A goal state is nearby to a self-satisfied rule if time estimates indicate that the system could satisfy the goal state and reestablish conditions of the self-satisfied rule within the expected waiting period. A more complex waiting policy could have a system elect to wait in the face of contending rules, especially when compatible rules and/or nearby goal states allow active waiting. Relative importances of goal states would influence such a policy. Finally, rather than an expected waiting period, a cumulative probability function representing the likelihood that a self-satisfied rule will be able to fire within a given period of time could add further sophistication to waiting policies.

Conclusion

Research on distributed problem solving has used waiting as a coordination primitive [1]. Coordination is realized in our model through ungrounded goal states and unsatisfied OTHEROP components. Postponing actions until others have completed their responsibilities may or may not result in waiting as defined here. Research on medical diagnosis has dealt with time duration propositions in rule conditions [2]. These suspend judgement until time lends its support. Again, time-based postponement differs from issues of waiting addressed here.

In this paper we have specified formalisms which add waiting to the repertoire of capabilities of problem solving systems dealing with real-world contexts. An (incomplete) set of conditions under which waiting may be reasonably considered are defined, as are aspects of the subsequent waiting decision process. We are continuing investigation of waiting by considering a particular context -- that of satisfying a set of work and personal goals on a given day within an office and city setting. This context has received attention in recent research on plan formulation [5]. It appears to be just as fruitful a source of ideas on plan execution and the role of waiting. Meeting schedules and office hours, going shopping, and using public transportation are actions which require frequent, even planned, waiting. A simulation to evaluate various waiting policies is planned.

REFERENCES

[1] Corkill, D. "Hierarchical planning in a distributed environment", Proceedings IJCAI-79, Tokyo, 1979, p. 168-175.

[2] Fagan L. etal., "Representation of dynamic clinical knowledge", Proceedings IJCAI-79, Tokyo, 1979, p. 260-262

[3] Farley, A.M. "The coordination of multiple goal satisfaction", Proceedings IJCAI5, MIT, 1977, p. 495.

[4] Farley, A.M. "Issues in knowledge-based problem solving" to appear in I.E.E.E. Transactions on Systems, Man, and Cybernetics, August, 1980.

[5] Hayes-Roth, B. and Hayes-Roth F. "A cognitive model of planning", Cogntive Science, 3 (1979), pp. 275-310.

[6] Newell, A. and Simon, H.A. Human Problem Solving, Prentice-Hall: Englewood Cliffs, NJ, 1972.

[7] Sacerdoti, E.D. "Planning in a hierarchy of abstraction spaces", Artificial Intelligence, 5, 1974, pp. 115-135.