Diagnosis of Plan Step Errors and Plan Structure Violations

Cees Witteveen
Faculty EEMCS
Delft University of Technology
P.O.Box 5031
NL-2600 GA Delft
The Netherlands
C.Witteveen@tudelft.nl

Nico Roos
Dept. of Computer Science
Universiteit Maastricht
P.O.Box 616
NL-6200 MD Maastricht
The Netherlands
roos@cs.unimaas.nl

Adriaan ter Mors &
Xiaoyu Mao
Almende B.V.
Westerstraat 50
NL-3016 DJ Rotterdam
The Netherlands
{adriaan,xiaoyu}@almende.com

ABSTRACT

Failures in plan execution can be attributed to errors in the execution of plan steps or violations of the plan structure. While in previous work we have concentrated on the first type of failures, in this paper we introduce the idea of diagnosing violations in the plan structure. The structure of a plan prescribes which actions have to be performed and which precedence constraints between them have to be respected. Especially in multi-agent environments violations of plan structure might easily occur as the consequence of synchronization errors. Using a formal framework for plan diagnosis, we show how Model-Based Diagnosis can applied to identify these violations of plan structure specifications and we analyze the computational complexity of the associated diagnostic problems.

Categories and Subject Descriptors

I.2.8 [**Problem Solving, Control Methods, and Search**]: Plan execution, formation, and generation; I.2.11 [**Distributed Artificial Intelligence**]: Coherence and coordination, Intelligent agents, Multiagent Systems

General Terms

Performance, Reliability, Theory

Keywords

Model-Based Diagnosis, Plan execution, Planning

1. INTRODUCTION

Plan diagnosis deals with the identification of errors occurring during the execution of a plan. In previous work, we have presented methods for identifying such errors as failed executions of plan steps in multi-agent plans [4, 6], equipment failures and malfunctioning agents causing the execution of plan steps to fail [1, 2], and methods for assigning responsibility to agents in case plan execution failed [1]. In all these papers, however, the tacit assumption was that the plan structure itself is not violated, i.e., all plan steps as specified in the plan have been executed and the order in which they are executed did not violate any precedence constraint. In reality, however, violations of the plan structure may easily occur. For instance, consider a plan for loading a truck that has to visit several places to deliver cargo. To guarantee an efficient unloading procedure, often the plan requires a specific precedence order between the items to be loaded. If such a loading order is violated, unloading at an intermediate delivery location may require unloading of other items besides the ones intended for this location, causing unnecessary delay and even violation of time constraints. Another example would be a plan for loading a ship that ensures a correct weight distribution by carefully ordering the items that have to be loaded. In this case, an incorrectly loaded ship may even disturb the stability of the ship in rough seas causing a total transport plan failure.

Multi-agent systems are particularly susceptible to such violations of plan structure. Often precedence constraints between plan steps to be executed by different agents are ensured by *synchronizing* the agents involved. In many cases such synchronization is achieved by relying on specific starting times of actions specified in the plan itself or by relying on observations that indicate the completion of tasks performed by other agents. If, in such a case, the execution of some crucial action is delayed or observation errors lead to incorrect beliefs about the state of the world, violations of precedence constraints may easily occur. Of course, other plan structure failures such as omitting or duplicating plan steps, can also easily occur in multi-agent environments as a result of synchronization problems.

2. PLANS AND PLAN EXECUTION

Plans as Systems We consider plan-based diagnosis as a simple extension of the model-based diagnosis approach, where the model is not a description of an underlying physical system but a *plan* of an agent. By executing plans we change a part of the world.

To keep representational issues as simple as possible, we assume that for the planning problem at hand, the world can be simply described by a set $Var = \{v_1, v_2, \ldots, v_n\}$ of variables and their respective *value domains* D_i. A *state of the world* σ then is a value assignment $\sigma : Var \rightarrow \bigcup_{i=1}^{n} D_i$

to the variables. We will denote such a state simply by an element of $D_1 \times D_2 \times \ldots \times D_n$, i.e. an n-tuple of values.

Therefore, we introduce a *partial state* as an element $\pi \in D_{i_1} \times D_{i_2} \times \ldots \times D_{i_k}$, where $1 \leq k \leq n$ and $1 \leq i_1 < \ldots < i_k \leq n$. We use $Var(\pi)$ to denote the set of variables $\{v_{i_1}, v_{i_2}, \ldots, v_{i_k}\} \subseteq Var$ specified in such a partial state π. The value $\sigma(v_j)$ of variable $v_j \in Var(\pi)$ will be denoted by $\pi(v_j)$. The value of a variable $v_j \in Var$ not occurring in a partial state π is said to be *undefined* (or *unpredictable*) in π, denoted by $\perp$. Including $\perp$ in every value domain D_i allows us to consider every partial state π as an element of $D_1 \times D_2 \times \ldots \times D_n$.

An important notion in plan diagnosis is the notion of *compatibility* between partial states. Two states π and π' are said to be compatible, denoted by $\pi \approx \pi'$, if there is no essential disagreement about the values assigned to variables in the two states. That is, $\pi \approx \pi'$ if $\forall v \in Var\ [\pi(v) = \perp \vee \pi'(v) = \perp \vee \pi'(v) = \pi(v)]$ A *plan operator* refers to a description of an action in a plan. In our model, plan operators are *functions* mapping partial states to partial states. More exactly, a plan operator o is a function that replaces the values in its range $ran_{Var}(o) \subseteq Var$ by other values (dependent upon the values of the variables in its domain $dom_{Var}(o) \subseteq Var$). Hence, every plan operator o can be modeled as a (partial) function $f_o : D_{i_1} \times \ldots \times D_{i_k} \rightarrow D_{j_1} \times \ldots \times D_{j_l}$, where $1 \leq i_1 < \ldots < i_k \leq n$ and $1 \leq j_1 < \ldots < j_l \leq n$.

A plan operator o may be used at several places in a plan. A specific occurrence s of o is called a *plan step* mapping a specific partial state into another partial state. A plan step s as an occurrence of o then describes a specific *function application* of the function f_o at a specific place in the plan. Therefore, given a set $\mathcal{O}$ of plan operators, we consider a set $S = inst(\mathcal{O})$ of *instances* of plan operators in $\mathcal{O}$, called the set of plan steps. A plan step will be denoted by a small roman letter s_i.

Plans An executable plan is a tuple $P = \langle \mathcal{O}, S, \prec \rangle$ where $S \subseteq Inst(\mathcal{O})$ is a set of plan steps occurring in $\mathcal{O}$ and $(S, \prec)$ is a partial order. The partial order relation $\prec$ specifies an *execution relation* between plan steps: for each $s \in S$ it holds that s is executed immediately after all plan steps s' such that $s' \prec s$ have been finished. We will denote the *transitive reduction* of $\prec$ by $\ll$.[1]

Without loss of generality, we assume that every plan step $s \in S$ takes one unit of time to execute and the execution of the first plan step starts at time $t = 0$. Using this assumption and the definition of the execution ordering $\prec$, the time t at which a plan step s will be executed is uniquely determined: Let $depth_P(s)$ be the depth of plan step s in plan $P = \langle \mathcal{O}, S, \prec \rangle$. Here, $depth_P(s) = 0$ if $\{s' \in S\ | s' \ll s\} = \varnothing$ and $depth_P(s) = 1 + max\{depth_P(s') \mid s' \ll s\}$, else.[2] Then the time t_s at which the plan step s is executed is $t_s = depth_P(s)$ and s will be completed at time $t_s + 1$. Let P_t denote the set of plan steps s with $depth_P(s) = t$, let $P_{>t} = \bigcup_{t' > t} P_{t'}$, $P_{<t} = \bigcup_{t' < t} P_{t'}$ and finally, let $P_{[t,t']} = \bigcup_{k=t}^{t'} P_k$.

Qualifications The correct execution of a *plan step may fail* either because of an inherent malfunctioning, or because of a malfunctioning of an agent responsible for executing the action, or because of unknown external circumstances. In all these cases, we model the effects of a failed execution of a plan-operator by introducing a set of *health modes* H_s for each plan step $s \in S$. This set H_s contains at least the normal mode *nor*, the mode *ab* indicating the most general abnormal behavior, and possibly several other specific fault modes. The most general abnormal behavior of action o is specified by the function f_o^{ab}, where $f_o^{ab}(d_{i_1}, d_{i_2}, \ldots, d_{i_k}) = (\perp, \perp, \ldots, \perp)$ for every partial state $(d_{i_1}, d_{i_2}, \ldots, d_{i_k}) \in dom(f_o)$.[3] To keep the discussion simple, we distinguish only the health modes *nor* and *ab*. We will use the set of plan steps $F \subseteq S$ to denote the plan steps that are qualified as abnormal (failed). The plan steps $S - F$ are therefore qualified as normal.

Missing and duplicated plan steps may at first sight be treated as special cases of failing plan steps. However, the execution of an omitted plan step still would take time and duplicating plan steps would not increase execution time. Therefore, we will qualify these plan steps using a special set D indicating the set of plan steps duplicated and M denoting the set of plan steps omitted. Instead of assigning health modes, we use these sets to *transform* the existing plan P into a new plan $P_{M,D}$ reflecting the omitted and duplicated plan steps. The resulting plan $P_{M,D} = \langle \mathcal{O}, S_{(M,D)}, \prec_{(M,D)} \rangle$ thus obtained will consist of the set of plan steps $S_{(M,D)} = S - M \cup \{s_{dup} : s \in D\}$, and the set of precedence constraints $\prec_{(M,D)} = \prec - \{(s, s'), (s', s) : s \in M\} \cup \{(s, s_{dup}), (s_{dup}, s') : s \in D, (s, s') \in \prec\}$. Here, the idea is that the duplicating plan step s_{dup} will be executed immediately after the original plan step s. Moreover, s_{dup} and s have the same behavior.

A *precedence constraint violation* occurs if a plan P specifies that some plan step s' is dependent upon a plan step s (i.e. $(s, s') \in \prec$) and either s and s' are executed concurrently or s is executed after s' is executed. A violation of the first type is considered to be an element of the set $C_=$ indicating a concurrent execution of dependent plan steps and $C_\prec$ indicates a stronger violation where the dependency is reversed. Note that a violation often implies other violations as well. For example, if $s \prec s'' \prec s'$ and (s, s') is violated then it is clear that (s'', s') is violated as well.

We use such a specification of violations $C = C_= \cup C_\prec$ to compute an updated set of precedence constraints $\prec \dagger C$ generated by the original precedence order $\prec$ and the set of violations C. That is, $\prec \dagger C = (\prec - C) \cup \{(s, s') : (s', s) \in C_\prec\}$.

Having defined the failed plan steps F, the missing steps M, the duplicated steps D, and the set C of violated constraints, we define a *total qualification* of plan failures as: $Q = (F, M, D, C)$ and we denote a plan P with these qualifications by $P_Q = \langle \mathcal{O}, S, \prec, Q \rangle$.

Plan execution In general, a plan P executed in a given initial state π_0 will induce a sequence of states $\pi_0, \pi_1, \ldots, \pi_k$, where π_{t+1} is generated from π_t by applying the set of plan steps P_t to σ_t. To define this relation between partial states at different time points we denote a partial state π at a given time t by a tuple, also called a *timed state*, denoted by (π, t).

The effect of executing a plan depends of course on the qualification $Q = (F, M, D, C)$. First, let us assume that both C and D and M are empty sets, that is, we have a

[1]So $\ll$ is the smallest subrelation of $\prec$ such that the transitive closure $\ll^+$ of $\ll$ equals $\prec$.

[2]If the context is clear, we often will omit the subscript P.

[3]This definition implies that the behavior of abnormal actions is essentially unpredictable.

plan $P_{(F,\varnothing,\varnothing,\varnothing)}$. We will first specify the *derivability relation* specifying the effect of a plan execution only taking into account the set F of plan steps that might have failed.

We define the execution of a plan step as follows:

Definition 1. *We say that $(\pi', t+1)$ is (directly) generated by execution of the F-qualified plan $P_{(F,\varnothing,\varnothing,\varnothing)}$ from (π, t), abbreviated by $(\pi,t) \rightarrow_{(F,\varnothing,\varnothing,\varnothing);P} (\pi', t+1)$, iff for every $v \in Var$ the following conditions hold:*

1. *if $v \notin ran_{Var}(P_t)$ then $\pi'(v) = \pi(v)$;*[4]
2. *if $v \in ran_{Var}(s)$ for some plan step $s \in P_t - F$ enabled in π, then $\pi'(v) = f_o^{nor}(\pi)(v)$;*
3. *else $\pi'(v) = \bot$.*

We now extend the direct derivability relation $\rightarrow_{(F,\varnothing,\varnothing,\varnothing);P}$ to a full derivability relation $\rightarrow_{(F,M,D,C);P}$ with missing and duplicated plan steps, and constraint violations, as follows:

Definition 2. *The timed state $(\pi', t+1)$ is (directly) generated from (π, t) by execution of $P = \langle \mathcal{O}, S, \prec \rangle$ under the qualification $Q = (F, M, D, C)$, abbreviated by $(\pi, t) \rightarrow_{Q;P} (\pi', t+1)$, iff $(\pi, t) \rightarrow_{(F,\varnothing,\varnothing,\varnothing);P'} (\pi', t+1)$ where $P' = \langle \mathcal{O}, S_{(M,D)}, (\prec \dagger C)_{(M,D)} \rangle$.*

We extend the direct derivability relation to a general derivability relation in a straightforward way:

Definition 3. *For arbitrary values of $t \le t'$ we say that (π', t') is (directly or indirectly) generated by execution of P^Q from (π, t), denoted by $(\pi, t) \rightarrow^*_{Q;P} (\pi', t')$, iff the following conditions hold:*

1. *if $t = t'$ then $\pi' = \pi$;*
2. *if $t' = t+1$ then $(\pi, t) \rightarrow_{Q;P} (\pi', t')$;*
3. *if $t' > t+1$ then there must exist a unique state $(\pi'', t'-1)$ such that $(\pi, t) \rightarrow^*_{Q;P} (\pi'', t'-1)$ and $(\pi'', t'-1) \rightarrow_{Q;P} (\pi', t')$.*

Note that $(\pi, t) \rightarrow^*_{(\varnothing,\varnothing,\varnothing,\varnothing);P} (\pi', t')$ denotes the normal execution of a normal plan $P_\varnothing$.

3. PLAN DIAGNOSIS

In our framework, a diagnosis is a qualification that resolves conflicts between the observed and predicted values of variables. To establish a plan diagnosis in our framework we need to make *observations*. Our framework provides a natural candidate for representing such observations: an observation *obs*(t) at time t can easily be represented by a timed state (π, t). Note that his implies that we do not require observations to specify a complete state.

Suppose that during the execution of a plan P we have an observation *obs*$(t) = (\pi, t)$ and an observation *obs*$(t') = (\pi', t')$ at some later time $t' > t \ge 0$. We would like to use these observations to infer a qualification $Q = (F, D, M, C)$ for the plan. First, assuming a normal execution of P, we can predict the partial state of the world at a time point t' given the observation *obs*(t): if all plan steps behave normally, no plan steps are omitted or duplicated and no constraint is violated, we predict the timed state $(\pi'_\varnothing, t')$ such that *obs*$(t) \rightarrow^*_{(\varnothing,\varnothing,\varnothing,\varnothing);P} (\pi'_\varnothing, t')$.

[4] Here, $ran_{Var}(P_t)$ is a shorthand for $\bigcup_{s \in P_t} ran_{Var}(s)$.

Such a prediction has to be compared with the actual observation *obs*$(t') = (\pi', t')$ made at time t'. It is easy to see if the predicted state and the observed state match, $\pi'_\varnothing$ and π' are *compatible* states, i.e. $\pi' \approx \pi'_\varnothing$ holds.

If this is not the case, the execution of some plan steps must have gone wrong, some plan steps might have been omitted or duplicated, or some precedence constraint might have been violated. Therefore, we have to determine a qualification $Q = (F, M, D, C)$ such that the predicted state π'_Q derived using Q is compatible with π'. Hence, we have the following straight-forward extension of the diagnosis concept in MBD to plan diagnosis (cf. [3]):

Definition 4. *Let $P = \langle \mathcal{O}, S, \prec \rangle$ be a plan with observations obs$(t) = (\pi, t)$ and obs$(t') = (\pi', t')$, where $t < t' \le$ depth(P) and let obs$(t) \rightarrow^*_{Q;P} (\pi'_Q, t')$ be a derivation using the qualification Q. Then Q is said to be a* qualification diagnosis *of $\langle P, obs(t), obs(t') \rangle$ iff $\pi' \approx \pi'_Q$.*

Finding diagnoses Since model-based diagnosis is an NP-hard problem, it comes as no surprise that the same holds for diagnosis of plan-execution. In [5], we have shown that diagnosis of failing plan steps can be determined in polynomial time for the important sub-class of *maximal-informative* diagnoses.

The diagnosis of missing and duplicated plan steps and of constraint violation all turn out to be NP-hard and we do not have an important sub-class that can be determined in polynomial time yet. However, we have developed polynomial time heuristics for the general cases.

4. CONCLUSION AND FURTHER WORK

We have extended previous work on plan diagnosis in order to incorporate the identification of violations of the plan structure. This extension is particularly important for multi-agent plan execution where such constraint violations can easily be caused by coordination errors. Future work will focus on developing heuristics that work for a larger class of problems than our current heuristics, and on extending diagnosis in order to identify agents responsible for plan execution failures.

5. REFERENCES

[1] F. de Jonge, N. Roos, and C. Witteveen. Diagnosis of multi-agent plan execution. In *Multiagent System Technologies: MATES 2006, LNCS 4196*, pages 86–97, 2006.

[2] F. de Jonge, N. Roos, and C. Witteveen. Primary and secondary plan diagnosis. In *The International Workshop on Principles of Diagnosis, DX-06*, 2006.

[3] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.

[4] N. Roos and C. Witteveen. Diagnosis of plans and agents. In *Multi-Agent Systems and Applications IV: CEEMAS 2005, LNCS 3690*, pages 357–366, 2005.

[5] N. Roos and C. Witteveen. Models and methods for plan diagnosis. In *Formal Approaches to Multi-Agent Systems (FAMAS'06)*, 2006.

[6] C. Witteveen, N. Roos, R. van der Krogt, and M. de Weerdt. Diagnosis of single and multi-agent plans. In *AAMAS 2005*, pages 805–812, 2005.