Diagnosis of Plan Structure Violations

Nico Roos[1] and Cees Witteveen[2]

[1] Dept. of Computer Science, Universiteit Maastricht
P.O. Box 616, NL-6200 MD Maastricht
roos@micc.unimaas.nl
[2] Faculty EWI, Delft University of Technology
P.O. Box 5031, NL-2600 GA Delft
C.Witteveen@tudelft.nl

Abstract. Failures in plan execution can be attributed to errors in the execution of plan steps or violations of 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. While in previous work we have concentrated on the first type of failures, in this paper we introduce the idea of diagnosing plan structure violations. Using a formal framework for plan diagnosis, we describe how Model-Based Diagnosis can applied to identify these violations of plan structure specifications and we analyze their computational complexity.

Keywords: Model-Based Diagnosis, Plan execution, Coordination errors.

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 [1,2], equipment failures and malfunctioning agents causing the execution of plan steps to fail [3,4], and methods for assigning responsibility to agents in case plan execution failed [4]. In all these papers, however, we tacitly assumed that during plan execution the *plan structure* is not violated, i.e., all plan steps as specified in the plan are executed (correctly or incorrectly) and the order in which they are executed does not violate any precedence constraint.

In reality, however, violations of the plan structure may easily occur and might result in plan failure. For instance, consider a plan for loading a truck that has to visit several places to deliver cargo. Often, such a plan contains a specific ordering of loading actions guaranteeing that items are loaded in such a way that they can be unloaded in an efficient way. If, however, the structure of such a loading plan is violated, upon the delivery location it may force to unload other items in order to get the right item that must be unloaded, 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 occurring as a consequence of *synchronization problems* between agents. In such a multi-agent system a joint plan has to be executed by several agents each performing a subset of actions. Correct execution of the plan requires synchronization of their activities. Often, planning agents need not use special synchronization actions to synchronize their activities during plan-execution. Instead, 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, however, the execution of some crucial action is delayed or if observation errors lead to incorrect beliefs about the state of the world, violations of precedence constraints may easily occur. Also other failures of plan structure, as omitting or duplicating plan steps, might easily occur in multi-agent environments. For example, suppose that the set of plan steps to be executed by an agent overlaps with the set of plan steps to be executed by another agent. The first agent that is able to perform such a plan step will do so, enabling the other agent to skip the plan step. In such cases, without special synchronization actions, an agent might erroneously conclude that an action has already been performed (or not performed) by the other agent, taking the wrong action and causing the plan to fail. In any case, plan diagnosis should be able to identify such violations if they occur during the execution of a plan.

Remark. Identifying violations of precedence constraints is closely related to diagnosis of coordination errors. Kalech and Kaminka [5] apply classical model-based diagnosis [6,7] to identify coordination errors between *reactive agents*. Each reactive agent executes some *behavior* that may need to be coordinated with other agents. The coordination must ensure that certain constraints on behaviors are satisfied. Violation of these constraints implies that some agents behave abnormally in the sense that they fail to coordinate their behaviors.

The main difference with the work of Kalech and Kaminka is that here we have a (traditional non-behavior-based) plan in which coordination errors lead to violations of the plan's structure. Diagnosing plan structure violations also differs from other approaches to *plan diagnosis* proposed in the literature: [8,9,10,11,12,2,4].

In this paper, we extend the framework for plan diagnosis as described in [1,2]. In this model the state of the world is modeled by set of variables (objects) the values of which are changed by plan steps executed. This representation makes it possible to apply classical Model-Based Diagnosis (MBD) [6]) to identify anomalies in the execution of the plan. To simplify the presentation of diagnosis of violations of precedence constraints, we do not used the extension of the above mentioned model[1] presented in [3,4].

The remainder of this paper is organized as follows: Section 2 introduces the basic framework for plan-based diagnosis. Section 3 extends plan diagnosis to enable diagnosis of plan structure failures and Sect. 4 concludes the paper.

[1] In this extension we showed how our plan diagnosis approach can be conceived as a Discrete Event System (DES) [13,14,15,16]) of which the state is changed by unknown events causing anomalies in the plan execution. The here presented extensions of the former model that enable diagnosis of violations with respect to the structure of the plan can easily be incorporated in the latter, more elaborate, model.

2 Plans and Plan Execution

Before we discuss the idea of diagnosing plan structure failures, we start with a brief introduction to plan-based diagnosis.

2.1 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 one or more agents. By executing the plan 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.

We also 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 and they could be extended to the same complete state. That is,

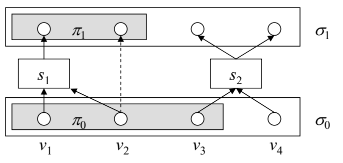
$$\pi \approx \pi' \text{ if } \forall v \in Var\ [(\pi(v) = \perp) \vee (\pi'(v) = \perp) \vee (\pi'(v) = \pi(v))].$$

Actions, Plan Operators and Plan Steps. In the preceding section we used the term 'actions' in a rather informal way. From now on we will distinguish *plan operators* and *plan steps*, which are both covered by the term 'actions'.

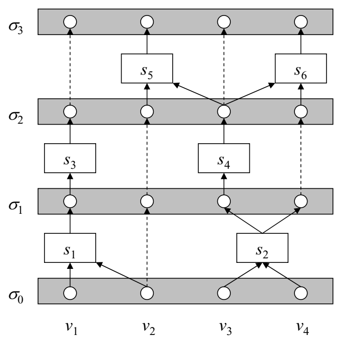
A *plan operator* refers to a description of an action in a plan. In our model, plan operators are *functions* mapping partial states to partial states. More exactly, a plan operator o is a function that replaces the values in its range $ran_{Var}(o) \subseteq Var$ by other values (dependent upon the 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 $1 \leq j_1 < \ldots < j_l \leq n$. Note that the set of variables in a plan operator's range $ran_{Var}(o)$ may differ from the variables in its domain $dom_{Var}(o)$. This property ensures that most planning formalisms, such as STRIPS, can be modeled using our plan operators. Also note that if a variable occurs in a plan operator's range but not in its domain, its value will be set by the application of the plan operator independently of its previous value.

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. We use $type(s)$ to denote plan operator o of which the plan step s is an instance: $s \in inst(o)$. Moreover, we use $dom_{Var}(s)$ for $dom_{Var}(type(s))$ and $ran_{Var}(s)$ for $ran_{Var}(type(s))$.

Example 1. Figure 1(a) depicts two states σ_0 and σ_1 (the white boxes) each characterized by the values of four variables v_1, v_2, v_3 and v_4. The partial states π_0 and π_1 (the gray boxes) characterize a subset of values in a (complete) state. The plan steps s_1 and s_2 are instances of the plan operators o_1 and o_2, respectively. Plan operators are used to model state changes. The domain of the plan operator o_1 is the subset $\{v_1, v_2\}$, denoted by the arrows pointing to s_1. The range of o_1 is the subset $\{v_1\}$, which is denoted by the arrow pointing from s_1. Finally, the dashed arrow denotes that the value of variable v_2 is not changed by the plan step s_1 causing the state change.



(a) Plan operators, states and partial states



(b) Plans and plan steps

Fig. 1. Plans, plan states and plan steps

Plans and Plan Execution. 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 these instances: 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*[2] of $\prec$ by $\ll$.

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 $\{s' \in S \mid s' \ll s\} = \varnothing$ and $depth_P(s) = 1 + max\{depth_P(s') \mid s' \ll s\}$, else. [3] 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

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

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

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 $P_{[t,t']} = \bigcup_{k=t}^{t'} P_k$.

Example 2. Figure 1(b) illustrates a plan with precedence relations: $s_1 \ll s_3$, $s_2 \ll s_4$, $s_4 \ll s_5$ and $s_4 \ll s_6$. In this plan, the depth of s_1 and s_2 is 0, the depth of s_3 and s_4 is 1, and the depth of s_5 and s_6 is 2. Therefore, $P_0 = \{s_1, s_2\}$, $P_1 = \{s_3, s_4\}$ and $P_2 = \{s_5, s_6\}$.

Given a state σ at some time t and the set P_t of plan steps to be executed at time t we want to be sure that the next state σ' at time $t + 1$ is uniquely defined. If P_t contains two plan steps s and s' with overlapping ranges, i.e., if $ran_{Var}(s) \cap ran_{Var}(s') \neq \varnothing$, the final result of a variable v occurring in this intersection is not uniquely defined in σ'. We therefore assume the following condition to hold:

Determinism. If P is a plan and s, s' are plan steps in P such that $ran_{Var}(s) \cap ran_{Var}(s') \neq \varnothing$ then $depth_P(s) \neq depth_P(s')$.

It is not difficult to see that Determinism guarantees that a future plan state can be defined uniquely given a plan, the current time t and a partial state at time t.

2.2 Qualifications

As we already noted in the introduction, several types of failure can be distinguished: the execution of a plan step might fail, a plan step might be omitted, a plan step might be executed more than once (duplicated) or the precedence order between plan steps might be violated. If such a failure occurs, we say that a plan step is qualified as failed, missing or duplicated, or a precedence constraint is qualified as violated. Below we specify these qualifications in detail. In the next subsection we specify their consequence for plan execution. This will enable us to diagnose such failures.

Failing Plan Steps. 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 plan operator o is specified by the function f_o^{ab}, where $f_o^{ab}(d_{i_1}, d_{i_2}, \ldots, d_{i_k}) = (\bot, \bot, \ldots, \bot)$ for every partial state $(d_{i_1}, d_{i_2}, \ldots, d_{i_k}) \in dom(f_o)$.[4] 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 behavior of each plan step $s \in F$ is specified by the function f_o^{ab} where $s \in inst(o)$. The plan steps $S - F$ are qualified as normal and the behavior of the of each plan step $s \in (S - F)$ is specified by the function f_o^{nor}.

Omitted and Duplicated Plan Steps. At first sight it may seem that omitted (missing) and duplicated plan steps could be treated as special cases of failing plan steps.

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

For example, an omitted plan step $s \in inst(o)$ could be qualified as omitted by assuming a special health mode *omit* such that f_o^{omit} equals the identity function, while a plan step could be qualified as duplicated by assuming a health mode *dup* such that $f_o^{dup} = f^{nor} \circ f^{nor}$. The problem with this solution is that the execution of an omitted plan step still would take time and duplicating a plan steps would not increase execution time. Therefore, instead of assigning health modes, we propose another approach, where duplicated and omitted plan steps are indicated as the result of an explicit *plan transformation*. We specify this transformation using a special set D indicating the set of plan steps duplicated and M denoting the set of plan steps omitted (missing). The existing plan P then is transformed into a new plan $P_{M,D}$ reflecting the omitted and duplicated plan steps. This plan $P_{M,D} = \langle \mathcal{O}, S_{(M,D)}, \prec_{(M,D)} \rangle$ consists of the set of plan steps $S_{(M,D)} = S - M \cup \{s_{dup} \mid s \in D\}$, and the set of precedence constraints $\prec_{(M,D)} = \prec - \{(s,s'),(s',s) \mid s \in M\} \cup \{(s,s_{dup}),(s_{dup},s') \mid s \in D, (s,s') \in \prec\}$. 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, since $type(s) = type(s_{dup})$.

Precedence Constraint Violations. A precedence 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 the execution order of s and s' is *reversed*.[5] Instances of $\ll \subseteq \prec$ that are reversed are denoted by the set C. We have to take care that C is a closed set of violations. For example, if $s \prec s'$, $s' \prec s''$ then a violation of $s \prec s''$ not only implies that $(s,s'') \in C$, but also that $(s',s'') \in C$.[6] The plan $P^C = (\mathcal{O}, S, \prec \dagger C)$ is plan transformation from P where $\prec \dagger C$ is the updated set of precedence constraints generated by $\prec$ and the set of violations C: $\prec \dagger C = (\prec - C) \cup \{(s,s') \mid (s',s) \in C\}$.

Total Qualification. Having defined the plan steps that are qualifies as failed F, as missing M and as duplicated D, and the constraints that are qualified as violated by C, we define a total qualification of plan failures as: $Q = (F,M,D,C)$ and we denote a plan P with these qualifications by $P_Q = \langle \mathcal{O}, S, \prec, Q \rangle$. We keep in mind, however, that such a plan P_Q also implicitly defines a (complex) transformation of the original plan P.

2.3 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).

[5] Strictly speaking, a violation of $s \prec s'$ could also imply that s and s' are executed concurrently. Such a violation, however, leads to unpredictable outcomes because the *determinism* requirement is violated. This implies that we cannot distinguish between unplanned concurrent execution of plan steps and plan step execution failures. Therefore, this type of constraint violations will not be distinguished explicitly.

[6] In general, if $(s,s'') \in C$ then for all $s' \in S$ such that $s \prec s' \prec s''$ it holds that $(s',s'') \in C$.

This execution relation will be defined incrementally. We will start with a plan where the only failures that are allowed are plan step failures. Then the relation for plans with plan structure failures is defined by reducing them to the first case.

Execution of Failing Plan Steps. First, let us assume that M, D and C are empty sets, that is, we have a plan $P_{(F,\varnothing,\varnothing,\varnothing)}$. We will first specify how the derivability relation can be specified taking into account the set F of plan steps that might have failed [2].

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 $\pi'(v) = \pi(v)$;*
   Here, $ran_{Var}(P_t)$ is a shorthand for the union of the sets $ran_{Var}(s)$ with $s \in P_t$.
2. *if $v \in ran_{Var}(s)$ for some plan step $s \in P_t - F$ enabled in π (i.e., $dom_{Var}(s) \subseteq Var(\pi)$), then $\pi'(v) = f_o^{nor}(\pi)(v)$;*
3. *else $\pi'(v) = \bot$.*

Omitted and Duplicated Plan Steps. We now extend the direct derivability relation $\rightarrow_{(F,\varnothing,\varnothing,\varnothing);P}$ for normal and failing plan steps with missing and duplicated plan steps. As was pointed out in the previous subsection, the idea is that missing and duplicated plan steps transform the original plan P into an new plan P'. Hence, the direct derivability relation of the original plan P with qualification $(F, M, D, \varnothing)$ can be simply defined as follows:

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

That is, $(\pi, t) \rightarrow_{(F,M,D,\varnothing,);P} (\pi', t+1)$ iff $(\pi, t) \rightarrow_{(F,\varnothing,\varnothing,\varnothing);P^{M,D}} (\pi', t+1)$

Precedence Constraint Violations. Finally, we extend the direct derivability relation with a non empty set of violated precedence constraints. Constraint violations also modify the original plan P by eliminating constraints and by adding the reverse of the eliminated constraints. Hence, we define the execution relation analogous to the previous case:

Definition 3. *The timed state $(\pi', t+1)$ is (directly) generated from (π, t) by execution of the plan $P = \langle \mathcal{O}, S, \prec \rangle$ given the qualification $Q = (F, M, D, C)$, abbreviated by $(\pi, t) \rightarrow_{(F,M,D,C);P} (\pi', t+1)$, iff $(\pi', t+1)$ is (directly) generated from (π, t) by execution of the plan $P^{M,D,C} = \langle \mathcal{O}, S_{(M,D)}, (\prec \dagger C)_{(M,D)} \rangle$ given the qualification $(F, \varnothing, \varnothing, \varnothing)$.*

That is, $(\pi, t) \rightarrow_{(F,M,D,C);P} (\pi', t+1)$ iff $(\pi, t) \rightarrow_{(F,\varnothing,\varnothing,\varnothing);P^{M,D,C}} (\pi', t+1)$.

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

Definition 4. *For arbitrary values of $t \leq t'$ 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 $P_\varnothing$. Such a normal plan execution will also be denoted by $(\pi, t) \rightarrow^*_P (\pi', t')$.

3 Plan Diagnosis

In our framework, a diagnosis is a qualification that resolves conflicts between the observed and predicted values of variables. To establish 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 this 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 $t' > t \geq 0$. 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 $obs(t) \rightarrow^*_{(\varnothing,\varnothing,\varnothing,\varnothing);P} (\pi'_\varnothing, t')$.

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: in that case we should be able to find a state σ such that both the observed state π' and the predicted state $\pi'_\varnothing$ are contained in σ, that is, $\pi' \sqsubseteq \sigma$ and $\pi'_\varnothing \sqsubseteq \sigma$. Hence, $\pi'_\varnothing$ 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. [6]):

Definition 5. *Let $P = \langle \mathcal{O}, S, \prec \rangle$ be a plan with observations $obs(t) = (\pi, t)$ and $obs(t') = (\pi', t')$, where $t < t' \leq depth(P)$ 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 $\pi' \approx \pi'_Q$.*

It is easy to show that such a diagnosis can always be proven to exist if for every variable v there exists at least some plan step s and some time $t \leq t'' \leq t'$ such that $s \in P_{t''}$ and $v \in ran_{Var}(s)$.

Example 3. Consider the plan depicted in Fig. 2.a. Let $obs(0) = (\pi_0, 0)$, $obs(3) = (\pi'_3, 3)$ and let π'_3 be equal to π_3 except that there is a deviation in the value of v_1, v_2 and v_4 at time $t = 3$ (as indicated by the black dots).

Suppose that changing the execution order of plan steps s_4 and s_7 enables us to correctly predict the value of variable v_4, and omitting plan step s_6 enables us to predict the value of variable v_2. Then $Q = (\{s_5\}, \{s_6\}, \varnothing, \{(s_7, s_4)\})$ is a qualification diagnosis as depicted in Fig. 2.b.

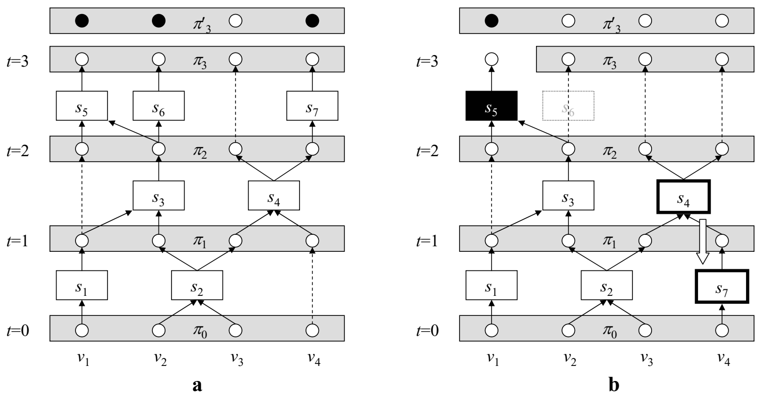


Fig. 2. Plan execution before and after a qualification diagnosis together with an observation deviating from the expected observation, as indicated by the black dot

3.1 Identifying Diagnoses

Let the size $||Q||$ of a qualitative diagnosis $Q = (F, D, M, C)$ be equal to the sum of the cardinalities of the sets involved, i.e. $||Q|| = |F| + |D| + |M| + |C|$. Intuitively, we should aim at finding diagnoses of minimum size. In general, finding such minimum diagnoses is an NP-hard problem, and it turns out that the same holds for plan diagnosis, too, even if we restrict our attention to the diagnosis of failing plan steps. In this section, therefore, we will restrict our attention to the complexity of finding *pure* F, D, M or C diagnoses.

Identifying F-diagnoses. Identifying an arbitrary F-diagnosis is trivial: qualify every plan step as abnormal. This will constitute a diagnosis. Finding a subset-minimal F-diagnosis is also easy, but finding a minimum (cardinality-minimal) F-diagnosis is NP-hard [17].[7]

Minimizing the *size* of a diagnosis, however, is only one option in finding a suitable diagnosis. It is usually preferred if the normal health state of a plan step is more likely than an abnormal one. Another criterion that is useful is the *information content* of a diagnosis. We say that a qualification diagnosis Q is more informative than another one

[7] It is not difficult to show that for every qualification diagnosis of size m there exists an F-diagnosis (where every component except the F-component is empty) with size less than or equal to m, explaining the same set of observations.

Q' iff $Var(\pi'_{Q'}) \subset Var(\pi'_Q)$, where $obs(t) \rightarrow^*_{Q;P} (\pi'_Q, t')$. A diagnosis Q is maximally informative (maxi-diagnosis) if no diagnosis Q' is more informative than Q.[8] Likewise, we can define a *minimal maximal informative* diagnosis (mini-maxi diagnosis):

Definition 6. *Let $\langle P, obs(t), obs(t')\rangle$ be a diagnostic problem with observations $obs(t) = (\pi, t)$ and $obs(t') = (\pi', t')$, and let $obs(t) \rightarrow^*_{Q;P} (\pi'_Q, t')$, given a qualification Q. Then Q is said to be a* maximally informative diagnosis *of $\langle P, obs(t), obs(t')\rangle$ iff (i) $\pi' \approx \pi'_Q$, and (ii) $Var(\pi_Q)$ is maximal among all diagnoses.*

Q is said to be a minimal maximally informative diagnosis *(mini-maxi diagnosis) iff the qualification Q is a minimal diagnosis among the maximally informative diagnoses.*

Mini-maxi diagnoses should be preferred if it is unlikely that a faulty plan step produces correct results. Quite surprisingly, as we have shown in a recent paper (see [17]), *mini-maxi* diagnoses can be found in polynomial time (polynomial in the size of the plan).

The following example gives an illustration:

Example 4. Reconsider the plan depicted in Fig. 2.a. If we only consider failing plan steps, then there are seven qualifications that are minimal diagnoses according to Definition 5. Among these seven diagnoses $Q = (\{s_2\}, \varnothing, \varnothing, \varnothing)$ is a *minimum* diagnosis, and $Q' = (\{s_3, s_7\}\varnothing, \varnothing, \varnothing)$ is a *mini-maxi* diagnosis. Let π'_Q denote the state derived at time $t = 3$ by using Q as a qualification. Then $Var(\pi'_Q) = \varnothing$, $Var(\pi'_{Q'}) = \{v_3\}$.

Consider a diagnosis Q of a plan with observations. Suppose that the qualification Q only consists of missing or duplicated actions and of violated constraints; i.e., $Q = (\varnothing, M, D, C)$. Then all plan steps are executed normally and the set of variables for which known values are predicted is maximal. Moreover, since the qualification Q is a diagnosis, these predicted values are compatible with the observations. Since coincidental compatibility of values is unlikely, this diagnosis must be preferred to any diagnosis in which less known values are predicted. Only plan steps that are qualified as failed reduce the predicted set of variables with known values. Hence, we can determine the missing and duplicated plan steps and the violated constraints by preferring mini-maxi diagnoses.

Example 5. Reconsider the plan depicted in Fig. 2.a. Suppose that changing the execution order of plan steps s_4 and s_7 enables us to correctly predict the value of variable v_4, and omitting plan step s_6 enables us to predict the value of variable v_2. Then $Q = (\{s_5\}, \{s_6\}, \varnothing, \{(s_7, s_4)\})$ is a qualification diagnosis. Figure 2.b depicts this diagnosis. Let π'_Q denote the state derived at time $t = 3$ by using the diagnosis Q. Then the set of correctly predicted variables given this diagnosis is: $Var(\pi'_Q) = \{v_2, v_3, v_4\}$.

Identifying Omitted and Duplicated Plan Steps. A plan step omitting diagnosis (M-diagnosis) and a plan step duplicating diagnosis (D-diagnosis) are defined by qualification diagnoses $Q = (\varnothing, M, \varnothing, \varnothing)$ and $Q = (\varnothing, \varnothing, D, \varnothing)$, respectively. It is easy to see that both M- and D-diagnosis are maximal informative diagnoses. They are both, in general, hard to compute:

[8] Note that a maximal informative diagnosis is also maximum informative diagnosis. Q is a maximum informative diagnosis if for no diagnosis Q': $|Var(\pi_{Q'})| > |Var(\pi_Q)|$.

Proposition 1. *Let $P = \langle \mathcal{O}, S, \prec \rangle$ be a plan with observations $obs(t) = (\pi, t)$ and $obs(t') = (\pi', t')$, where $t < t' \leq depth(P)$. Deciding whether an M-diagnosis exists as well as deciding whether a D-diagnosis exist is NP-hard.*

Proof. Easy reduction of KNAPSACK to an M- and to a D-diagnosis problem.[9]

Identifying Constraint Diagnoses. A constraint diagnosis is a qualification diagnosis $Q = (\varnothing, \varnothing, \varnothing, C)$. It is easy to see that every constraint diagnosis is a maximal informative diagnosis. It turns out that also these maxi-diagnoses are hard to compute:

Proposition 2. *Let $P = \langle \mathcal{O}, S, \prec \rangle$ be a plan with observations $obs(t) = (\pi, t)$ and $obs(t') = (\pi', t')$, where $t < t' \leq depth(P)$. Deciding whether a constraint diagnosis Q exists is NP-hard.*

Proof. Reduction of TSP to a C-diagnosis problem.[11]

3.2 Approximations

The above results are of course rather disappointing. However, assuming that the omitted or duplicated plan steps and constraint violations occur in unrelated parts of a plan (or occur only once), diagnoses can be determined efficiently. Given a diagnostic problem $\langle P, obs(t), obs(t') \rangle$ with observations $obs(t) = (\pi, t)$ and $obs(t') = (\pi', t')$, for each observed variable $v \in Var(\pi')$ at time point t' we can determine the set of plan steps and the set of precedence constraints on the value of the variable v at time point t' depends. Let $Dep^{steps}(v, t') \subseteq S$ and $Dep^{constr}(v, t') \subseteq \ll$ be the set of plan steps and the set of precedence constraints between pairs of plan steps, respectively, on which the value of the variable v at time point t' depends. The dependency sets $Dep^{steps}(v, t')$ of the observed variables in $Var(\pi')$ can be used to determine mini-maxi diagnoses in polynomial time.

A missing or duplicated plan step can also be determined using the dependency sets $Dep^{steps}(v, t')$. For each dependency set $Dep^{steps}(v, t')$ of an observed variable $v \in Var(\pi')$ of which the predicted value $\pi'_{\varnothing}(v)$ is incompatible with the observed value $\pi'(v)$, we can perform the following tests. Check for every plan step $s \in Dep^{steps}(v, t')$ whether omitting s or duplicating s enables us to predict compatible values for every observed variable $v' \in Var(\pi')$ such that $s \in Dep^{steps}_{P^{M,D}}(v', t')$. Here $P^{M,D}$ denotes either the modified plan $P^{\{s\},\varnothing}$ or the modified plan $P^{\varnothing,\{s\}}$ depending on whether we check for the omission or duplication of the plan step s. Similarly we can use the constraints $s \ll s' \in Dep^{constr}(v, t')$ to check for constraint violations.

Note that a group of agents can efficiently determine the dependency sets $Dep^{steps}(v, t')$ and $Dep^{constr}(v, t')$. A multi-agent protocol for determining dependency sets in general diagnostic problems has been presented in [19].

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

[9] The proof is omitted due to lack of space. It can be found in [18].

multi-agent plan execution where such constraint violations can easily be caused by coordination errors. We have pointed out that, like maximally informative diagnosis of failing actions, constraint diagnosis and diagnosis of missing plan steps and duplicated plan steps also try to establish a maximally informative explanation of the observations made. Unlike a maximally informative diagnosis, however, identifying these M-, D- and C-diagnoses turns out to be an NP-hard problem. A heuristic that enables an efficient search for constraint diagnoses in some restricted cases has been presented.

In future work we intend to take the diagnosis one step further: By looking at the constraint violations an agent is responsible for, we may identify a pattern that indicates a flaw in the behavior of the agent.

References

1. Roos, N., Witteveen, C.: Diagnosis of plans and agents. In: Pěchouček, M., Petta, P., Varga, L.Z. (eds.) CEEMAS 2005. LNCS (LNAI), vol. 3690, pp. 357–366. Springer, Heidelberg (2005)
2. Witteveen, C., Roos, N., van der Krogt, R., de Weerdt, M.: Diagnosis of single and multi-agent plans. In: AAMAS 2005, pp. 805–812 (2005)
3. de Jonge, F., Roos, N., Witteveen, C.: Primary and secondary plan diagnosis. In: 17^{th} International Workshop on Principles of Diagnosis, DX'06, Universidad de Valladolid, pp. 133–140 (2006)
4. de Jonge, F., Roos, N., Witteveen, C.: Diagnosis of multi-agent plan execution. In: Fischer, K., Timm, I.J., André, E., Zhong, N. (eds.) MATES 2006. LNCS (LNAI), vol. 4196, pp. 86–97. Springer, Heidelberg (2006)
5. Kalech, M., Kaminka, G.A.: Towards model-based diagnosis of coordination failures. In: AAAI 2005, pp. 102–107 (2005)
6. Reiter, R.: A theory of diagnosis from first principles. Artificial Intelligence 32(1), 57–95 (1987)
7. Kleer, J.d., Williams, B.C.: Diagnosing multiple faults. Artificial Intelligence 32(1), 97–130 (1987)
8. Kalech, M., Kaminka, G.A.: On the design of social diagnosis algorithms for multi-agent teams. In: IJCAI-03, pp. 370–375 (2003)
9. Kalech, M., Kaminka, G.A.: Diagnosing a team of agents: Scaling-up. In: AAMAS 2005, pp. 249–255 (2005)
10. de Jonge, F., Roos, N.: Plan-execution health repair in a multi-agent system. In: Proc. 23rd Annual Workshop of the UK Planning and Scheduling SIG (PlanSIG 2004) (2004)
11. Carver, N., Lesser, V.: Domain monotonicity and the performance of local solutions strategies for CDPS-based distributed sensor interpretation and distributed diagnosis. Autonomous Agents and Multi-Agent Systems 6(1), 35–76 (2003)
12. Horling, B., Benyo, B., Lesser, V.: Using self-diagnosis to adapt organizational structures. In: Proc. 5th Int'l. Conf. on Autonomous Agents, pp. 529–536. ACM Press, New York (2001)
13. Baroni, P., Lamperti, G., Pogliano, P., Zanella, M.: Diagnosis of large active systems. Artificial Intelligence 110(1), 135–183 (1999)
14. Cassandras, C.G., Lafortune, S.: Introduction to Discrete Event Systems. Kluwer Academic Publishers, Dordrecht (1999)
15. Debouk, R., Lafortune, S., Teneketzis, D.: Coordinated decentralized protocols for failure diagnosis of discrete-event systems. Journal of Discrete Event Dynamical Systems: Theory and Application 10, 33–86 (2000)

16. Pencolé, Y., Cordier, M.: A formal framework for the decentralised diagnosis of large scale discrete event systems and its application to telecommunication networks. Artificial Intelligence 164(1-2), 121–170 (2005)
17. Roos, N., Witteveen, C.: Models and methods for plan diagnosis. In: Formal Approaches to Multi-Agent Systems (FAMAS'06), ECAI 2006, Workshop Notes (2006)
18. Roos, N., Witteveen, C.: Diagnosis of plan structure violations. Technical Report MICC 07-05, Universiteit Maastricht (2007)
19. Roos, N., ten Teije, A., Witteveen, C.: A protocol for multi-agent diagnosis with spatially distributed knowledge. In: AAMAS 2003, pp. 655–661 (2003)