Computer Science
University of Manchester

ActionConstraintMaintenance

YingZhang,HowardBarringer

TechnicalReportUMCS--94--9--1

Action Constraint Maintenance *

Ying Zhang, Howard Barringer

Department of Computer Science
University of Manchester
Oxford Rd., Manchester, U.K.
yzhang@cs.man.ac.uk, howard@cs.man.ac.uk

September 1994

Abstract

This report presents a novel approach to detect conflicts in nonlinear plans. Conventional methods detect conflicts in a nonlinear plan by checking that if preconditions of actions in the plan are possibly deleted by some other actions. Our novel method, *action constraint maintenance*, detects conflicts in a nonlinear plan by checking that if the nonlinear plan satisfies a set of specific constraints: action constraints. We prove that, under certain conditions, action constraint maintenance is sound and complete, and we show that action constraint maintenance is much more efficient than conventional methods. The most significant advantage of this method is that, if a nonlinear plan is not consistent with an action constraint, all its possible expansions will not be consistent with the action constraint. Therefore, action constraint maintenance can be employed to detect conflicts in a nonlinear plan during the plan construction, i.e. "on the fly", and backtracking from inconsistencies will not lose possible solutions. By adopting action constraint maintenance "on the fly" in our deductive nonlinear planning framework, we avoid computation for *modal truth criteria* in nonlinear planning, and we can achieve and maintain the necessary truth for all the goals involved in a planning process. Also, action constraint maintenance itself is a domain independent search heuristic in nonlinear planning.

1 Introduction

Nonlinear planning is believed to be exponentially more efficient than linear planning (Chapman, 1987). Nonlinear planning avoids most of the unnecessary backtrackings experienced by linear planning. However, some other computational problems arise in nonlinear planning (Yang, 1989). One of the problems is that it is more difficult to verify nonlinear plans than to verify linear plans. This difficulty is caused by the fact that a nonlinear plan is partially specified. A nonlinear plan is partially specified in two ways, the ordering of the actions are partially specified and the variables in the plan are partially described. A nonlinear plan is conflict free only if all the possible completions of the partially specified plan are conflict free. Conventionally, conflicts in a nonlinear plan can be detected by checking that if all the properties are *necessarily* true in the sates where they are needed (Chapman, 1987). This conventional method is computationally expensive.

This report presents a novel approach, *action constraint maintenance*, to detect conflicts in nonlinear plans. Our new approach detects conflicts in a nonlinear plan by checking that if the nonlinear plan satisfies a set of specific constraints, action constraints. In (Zhang *et al.*, 1994), we proposed a set of domain independent and human independent rules to regulate actions and we showed that these rules can prevent actions from being improperly defined. Under these rules, we proved that a nonlinear plan is conflict free if and only if the corresponding set of domain constraints is satisfied in the model of the nonlinear plan. We termed this conflict detecting method *domain constraint maintenance*. In this report, we show that, from a set of domain constraints, a set of temporal formulas, which are termed *action constraints*, can be constructed. We prove that, under certain conditions, a nonlinear plan is conflict free if and only if the nonlinear plan itself (not its model) is consistent with the corresponding set of action constraints. We show that action constraint maintenance is much more efficient than other conflict detecting methods.

In addition to the efficiency, the most significant advantage of action constraint maintenance over other conflict detecting methods is that, if a nonlinear plan is not consistent with an action constraint, all its possible expansions will not be consistent with the action constraint. This property allows action constraint maintenance to be employed to maintain conflict freeness of a nonlinear plan during the plan construction process, i.e. "on the fly", for the property guarantees that backtracking from inconsistencies will not lose possible solutions. By adopting action constraint maintenance "on the fly" in our deductive nonlinear planning framework, we avoid computation for *modal truth criteria* (Chapman, 1987; Christensen, 1990; Pednault, 1991) experienced in classical nonlinear planning systems. All we need to do is to progressively build up the nonlinear plan to achieve goals and to check if the plan under construction is consistent with the corresponding set of action constraints. By doing this we can achieve and maintain the necessary truth for all the goals involved in planning processes. Also, action constraint maintenance itself is a domain independent search heuristic in nonlinear planning and can improve planning efficiency.

2 Preliminaries

2.1 A Temporal Logic for Nonlinear Planning

For deductive planners, planning is a reasoning process within some formal logic. In (Zhang & Barringer, 1994), we presented a point-based, reified interval temporal logic for nonlinear planning. In the temporal logic, time is discrete and is in a partial order structure; time points are the only temporal primitives and, therefore, time intervals are referred to by their end-points; assertions of proposition types are interpreted over the biggest possible time intervals; a proposition type is associated with a time interval through a global predicate; there are two proposition types, property and action in the logic; to distinguish between these two types, we use two global predicates $Hold$ and $Exec$. The atomic formula $Hold(P, t_i, t_j)$ expresses that the property P holds true over the biggest possible time interval (t_i, t_j) and the atomic formula $Exec(A, t_i, t_j)$ expresses that the action A is executed over the biggest possible time interval (t_i, t_j). An interpretation of the temporal logic is a tuple $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}, \mathcal{D}, \pi_F, \pi_P \rangle$, where $\mathcal{W}$ is a set of time points; $\mathcal{R}$ is a partial order relation among the time points in $\mathcal{W}$; $\mathcal{D}$ is the domain of non-temporal variables; π_F is an interpretation of the function symbols and π_P is an interpretation of the proposition types. In appendix A, we give the formal definition of the temporal logic.

2.2 Restrictions on Actions

2.2.1 Actions and Domain Constraints

Basically, planning can be seen as an activity of manipulating actions. In the planning literature, actions are defined in an ad-hoc fashion. There do not exist human independent rules to distinguish between "correctly defined" actions and "incorrectly defined" actions. The correctness of definitions of actions affects the correctness of plans composed of the actions. For example, in the blocks world, suppose that $Puton(A, B, C)$ is an action which moves the block A from the top of block B to the top of block C. Then, conventionally, the action has $Clear(A)$, $Clear(C)$, $On(A, B)$ as its preconditions, $On(A, C)$, $Clear(B)$, $\neg On(A, B)$ and $Clear(A)$ as its postconditions under the assumption that all the blocks considered are of the same size. However, can we add some other properties, for example $Colour(D, White)$ (i.e. the block D is white), as one of the preconditions and/or add some other properties, for example $Father(Tom, Jack)$ (i.e. Tom is Jack's father), as one of the postconditions of this action? Can we remove some properties, for example $Clear(C)$, from the preconditions and/or remove some properties, for example $\neg On(A, B)$, from the post conditions of this action? Obviously, the answer is no. In (Zhang, 1994; Zhang *et al.*, 1994), we proposed a set of domain independent and human independent restrictions by which the improperness of the above intended modifications can be easily identified.

We believe that for any properly defined action, all its properties (i.e. preconditions and postconditions) have some kind of inherent connection among them. None of these properties can be detached from the definition of the action and none of other irrelevant properties can be attached to the definition. For example, we can't expect that an action can be properly implemented with one of its preconditions unsatisfied. Also, we can't expect that an execution of an action achieves some other properties that are not in its postconditions. We find that the inherent connection among the properties of an action can be captured by domain constraints.

Domain constraints specify impossible states of a planner's world in a specific domain. For example, in the blocks world, $(On(x,y), Clear(y))$ is a domain constraint which states that it is impossible that block y is clear when block x is on the top of it and vice-versa. In planning, domain constraints are used to improve planning efficiency by pruning the search space to avoid branches which will lead to impossible states specified by the domain constraints (Warren, 1974; Drummond & Currie, 1987; Currie & Tate, 1989; Currie & Tate, 1990; Allen, 1991). Kelleher et al. suggested that some of the domain constraints can be automatically constructed from actions (Kelleher & Cohn, 1992; Kelleher, 1990). This, from the other direction, confirms the mutually dependent relation between actions and domain constraints. The set of domain independent and human independent restrictions we proposed gives a formal description of the mutually dependent relation between the actions and the domain constraints in a problem domain. We also show that these restrictions can prevent actions from being improperly defined. The set of restrictions is also the foundation of the action constraint maintenance. In this report, we simply list the set of restrictions. Interested readers may refer to (Zhang, 1994; Zhang *et al.*, 1994) for details.

2.2.2 Restrictions on Actions

In a state-based language, a domain constraint expresses that two properties which are defined to be mutually exclusive are not allowed to hold true at the same state or time point. In our nonlinear interval temporal logic, if two properties are defined to be mutually exclusive, then the temporal intervals over which they hold true respectively must not *possibly* and properly overlap. Then, a set of domain constraints Σ_D for a problem domain is a set of formulas of the form:

$$\forall t_1, t_2, t_3, t_4.(Hold(p, t_1, t_2) \wedge Hold(q, t_3, t_4) \Longrightarrow t_2 \preccurlyeq t_3 \vee t_4 \preccurlyeq t_1)$$

If Σ_D contains the above formula, the properties p and q are said to be *mutually exclusive* according to Σ_D. In the following discussion, for convenience, we often use the notation $p \leftrightsquigarrow q$ to represent the domain constraint of the above form. It is not difficult to prove that if a nonlinear temporal model satisfies a domain constraint of the above form, then the intervals of the two properties involved will not possibly and properly overlap.

Definition 1 (Complement Restriction) *Let Σ_D be a set of domain constraints and Γ be a set of actions. Γ is said to be* complementary for Σ_D *if the following conditions hold. For any property p, if p is an asserted (resp. deleted) property of an action A in Γ, then there exists at least one deleted (resp. asserted) property q of A such that $p \leftrightsquigarrow q \in \Sigma_D$.*

Given a set of domain constraints Σ_D and a set of properties $\mathcal{S}$, two properties $p, q \in \mathcal{S}$ are said to be *connective under Σ_D within $\mathcal{S}$* if $p \leftrightsquigarrow q \in \Sigma_D$ or there is a property $r \in \mathcal{S}$ such that $p \leftrightsquigarrow r \in \Sigma_D$ and r and q are connective under Σ_D within $\mathcal{S}$. If all the properties in the set $\mathcal{S}$ are connective under Σ_D within $\mathcal{S}$, then $\mathcal{S}$ is said to be connective under Σ_D.

Definition 2 (Connection Restriction) *Given a set of domain constraints Σ_D, an action is said to be* connective under Σ_D *if the set of all the deleted properties and all the asserted properties of the action is connective under Σ_D. If all the actions in a set Γ are connective under Σ_D, then Γ is said to be connective under Σ_D.*

Definition 3 (Consistency Restriction) *Given a set of domain constraints Σ_D, an action A is said to be* consistent with Σ_D *if for any two preconditions p and q of A, $p \leftrightsquigarrow q \notin \Sigma_D$ and for any two postconditions r and s of A, $r \leftrightsquigarrow s \notin \Sigma_D$. If all the actions in a set Γ are consistent with Σ_D, then Γ is said to be consistent with Σ_D.*

Definition 4 (Inclusion Restriction) *Given a set of domain constraints Σ_D, an action A is said to be* inclusive according to Σ_D *if the following condition hold. For every asserted (resp. deleted) property p of A, if q is a property such that $p \leftrightsquigarrow q \in \Sigma_D$, then either q is a deleted (resp. asserted) property of A or q is mutually exclusive with a precondition (resp. postcondition) of A according to Σ_D. If all the actions in an action set Γ are inclusive according to Σ_D, then Γ is said to be inclusive according to Σ_D.*

Definition 5 (Minimum Restriction) *Given a set of domain constraints Σ_D, a set of actions Γ is said to be* minimum *if for any two different actions A_1 and A_2 in Γ,*

either *the actions A_1 and A_2 do not have a common deleted property and they do not have a common asserted property,*

or *either at least one of the deleted properties of A_1 is mutually exclusive with one of the deleted properties of A_2 or at least one of the asserted properties of A_1 is mutually exclusive with one of the asserted properties of A_2.*

2.3 Domain Constraint Maintenance

In (Zhang *et al.*, 1994), we proposed another novel conflict detecting method for nonlinear plans: *domain constraint maintenance.* Domain constraint maintenance detects conflicts in a nonlinear plan by checking that the model of the nonlinear plan is consistent with the corresponding set of domain constraints. We proved that, under the above restrictions, a nonlinear plan is conflict free if and only if its model is consistent with the corresponding set of domain constraints. Domain constraint maintenance lays down the theoretical basis for action constraint maintenance. For convenience of reference, we list the main results of domain constraint maintenance in this section. Interested readers may refer to the original paper for the detailed proofs.

In the following discussion, whenever we state that a set of domain constraints Σ_D and a set of actions Γ satisfy the restrictions, we mean that Γ is complementary, connective, consistent, inclusive and minimum for Σ_D.

In our temporal logic, syntactically, a nonlinear plan can be expressed as a well formed temporal formula. Semantically, a nonlinear plan, together with the corresponding definitions of actions, describes a temporal model. A nonlinear plan consists of a set of time points at which some actions are executed and a set of temporal relations among the time points. Given a nonlinear plan and a set of actions Γ, based on the *closed world assumption*, a temporal model can be built such that the specification of actions in Γ are satisfied by the model.

Definition 6 (Temporal Model of Nonlinear Plan) *Given a nonlinear plan P which consists of a set of time points at which some actions in Γ are executed and a set of temporal relations among the time points. Then $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}, \mathcal{D}, \pi_F, \pi_P \rangle$ is said to be* the temporal model of P according to Γ *if it is constructed as follows.*

$\mathcal{W}$ = { t_i: t_i is a time point in P} $\cup\{T_0, T_\infty\}$ where T_0 represents the initial time point and T_∞ represents the universal ending point.

*$\mathcal{R} = \{t_i \prec t_j : t_i \prec t_j$ is a temporal relation of $P\} \cup$
$\{T_0 \prec t_i, t_i \prec T_\infty :$ for every time point t_i in $P.\}$*

For every action A in P
$\pi_P(A) = \{(t_i, t_i)$: A is executed at t_i in $P.\}$

For every property p which is either a precondition or a postcondition of an action in P
$\pi_P(p) = \{(t_i, t_j)$: if one of the following conditions holds.

1. *p is an asserted property of an action executed at t_i in P and*
   (a) *p is a deleted property of an action at t_j in P and $t_i \prec t_j$ holds in P, and there is no other action which deletes p and is executed at time points between t_i and t_j in P.*
   (b) *Otherwise $t_j = T_\infty$;*
2. *p is a precondition of an action executed at t_k and both the following conditions hold*
   (a) *p is an asserted property of an action executed at t_i and $t_i \prec t_k$ and there is no other action which asserts p and is executed at the time points between t_i and t_k in P. Otherwise, $t_i = T_0$.*
   (b) *p is an deleted property of an actions executed at t_j and $t_k \preccurlyeq t_j$ and there is no other action which deletes p and is executed at time points between t_k and t_j in P. Otherwise, $t_j = T_\infty$.}*

2.3.1 Domain Constraints via Necessary-Biggest-Interval Axioms

In our temporal logic for nonlinear planning, properties are asserted over the necessarily biggest possible time intervals. That is, if a property is asserted over an interval in an interpretation, then the interval should be the biggest possible for the property in all the completions of the interpretation[1]. This requirement can be captured by the following temporal formula.

$$\forall t_1, t_2, t_3, t_4.$$
$$(Hold(p, t_1, t_2) \wedge Hold(p, t_3, t_4) \Longrightarrow t_2 \prec t_3 \vee t_4 \prec t_1 \vee (t_1 = t_3 \wedge t_2 = t_4))$$

This formula states that, for any two intervals of the same property p, either they are equal or one of them is strictly before the other. We term such a formula *necessary-biggest-interval* axiom and we often use Σ_B to represent the set of necessary-biggest-interval axioms in a problem domain. The first theoretical result of domain constraint maintenance is as follows.

Theorem 1 *Let Σ_D be a set of domain constraints and Γ be a set of actions such that Σ_D and Γ satisfy the restrictions. Suppose that P is a nonlinear plan and $\mathcal{M}$ is the nonlinear temporal model of P. If $\mathcal{M}$ satisfies Σ_D, then $\mathcal{M}$ satisfies Σ_B.*

[1] If an interpretation $\mathcal{M}$ is the tuple $\langle \mathcal{W}, \mathcal{R}, \mathcal{D}, \pi_F, \pi_P\rangle$, then $\mathcal{M}' = \langle \mathcal{W}, \mathcal{C}(\mathcal{R}), \mathcal{D}, \pi_F, \pi_P\rangle$ is a *completion* of $\mathcal{M}$, where $\mathcal{C}(\mathcal{R})$ is a total order relation over the time points in $\mathcal{W}$ such that $\mathcal{R} \subseteq \mathcal{C}(\mathcal{R})$.

2.3.2 Soundness of Domain Constraint Maintenance

Theorem 2 (Soundness of Domain Constraint Maintenance) *Let Σ_D be a set of domain constraints and Γ be a set of actions such that Σ_D and Γ satisfy the restrictions. Let P be a nonlinear plan that consists of the actions in Γ and $\mathcal{M}$ be the nonlinear temporal model of P. If P is conflict free, then $\mathcal{M}$ satisfies Σ_D.*

Theorem 3 *Let Σ_D be a set of domain constraints and Γ be a set of actions such that Σ_D and Γ satisfy the restrictions. Let P be a nonlinear plan that consists of the actions in Γ, $\mathcal{M}$ be the nonlinear temporal model of P and Σ_B be the set of necessary-biggest-interval axioms. If P is conflict free, then $\mathcal{M}$ satisfies Σ_B.*

2.3.3 Completeness of Domain Constraint Maintenance

Theorem 4 (Completeness of Domain Constraint Maintenance) *Let Σ_D be a set of domain constraints and Γ be a set of actions such that Σ_D and Γ satisfy the restrictions. Let P be a nonlinear plan that consists of the actions in Γ and $\mathcal{M}$ be the nonlinear temporal model of P. If $\mathcal{M}$ satisfies Σ_D, then P is conflict free.*

3 From Domain Constraints to Action Constraints

3.1 Specification of Actions in the Interval Temporal Logic

In planning, a given problem domain can be described by a set of actions Γ and a set of domain constraints Σ_D. In this temporal language, each action in Γ can be formally specified by a set of formulas in Horn clause form. In this report, we consider point actions only.

Example 1 *In the blocks world, suppose that the action $Puton(x, y, z)$, which moves block x from y to z, has the properties $On(x, y)$, $Clear(x)$, and $Clear(z)$ as its preconditions, deletes the properties $On(x, y)$ and $Clear(z)$, and asserts properties $On(x, z)$ and $Clear(y)$[2] This action can be specified by the following two clauses.*

$$
\begin{aligned}
&Hold(On(x,z), t_1, SKF1(t_1,x,y,z)) :- \\
&\quad Hold(Clear(x), t_2, t_2'), Hold(Clear(z), t_3, t_1), Hold(On(x,y), t_4, t_1), \\
&\quad Exec(Puton(x,y,z), t_1, t_1), t_2 \prec t_1 \prec t_2';
\end{aligned}
$$

$$
\begin{aligned}
&Hold(Clear(y), t_1, SKF2(t_1,x,y,z)) :- \\
&\quad Hold(Clear(x), t_2, t_2'), Hold(Clear(z), t_3, t_1), Hold(On(x,y), t_4, t_1), \\
&\quad Exec(Puton(x,y,z), t_1, t_1), t_2 \prec t_1 \prec t_2';
\end{aligned}
$$

The causal relations and the temporal relations of the properties and the action involved are clearly specified in the clauses. For example, the first formula expresses that if the blocks x and z are clear, x is on y and the action $Puton(x, y, z)$ is executed, then x

[2] In this definition, x is not the table; if y is the table, then $Clear(y)$ is not an asserted property; if z is the table, the $Clear(z)$ is not a precondition and a deleted property.

will be on z. The formula also expresses that the temporal intervals over which the deleted properties $Clear(z)$ and $On(x,y)$ hold true are finished at the time point t_1 at which the action is executed, the intervals over which the undeleted precondition $Clear(x)$ holds overrids the time point t_1, and the interval over which the asserted property $On(x,z)$ holds is started from t_1. SKF1 and SKF2 are Skolem functions.

Definition 7 (Planning Rule) *In a given planning problem domain, the set of (point) actions Γ is a set of* planning rules *of the form*

$$Hold(p, t, SKF(t, v_1, \ldots, v_m)) : -B_1, \ldots, B_n, Exec(A, t, t), C$$

where B_i, $i = 1, \ldots n$, the bodies of the planning rule are of the form $Hold(q_i, t_i, t'_i)$, A is the action of the rule and C is a formula consists of the temporal constraints. If the above planning rule is in Γ, then p is said to be an asserted property of A*, p is also said to be a* postcondition of A*. q_i, $i = 1 \ldots n$, are said to be the* preconditions of A*. For any B_i, if it is of the form $Hold(q_i, t_i, t)$, then it is said to be* a deleted property of A*. Otherwise, it is also said to be a postcondition of A. SKF is a Skolem function and $v_1, \ldots, v_m$ are all the universally quantified data variables that appear in p.*

The initial conditions of a planning problem can be expressed as special planning rules, each with a empty body:

$$Hold(p, T_0, SKC(v_1, \ldots, v_m)) : -$$

where SKC is a Skolem function. The formula expresses that the property p holds true from the initial time point T_0 to a time point at which it is deleted.

3.2 Domain Constraints via Action Constraints

A set of domain constraints is a set of property pairs such that the two properties in each pair are not allowed to properly and possibly overlap. Defined in the same way, a set of action constraints is a set of action pairs such that the two actions in each pair are not allowed to properly and possibly overlap. For point actions, the two actions of each pair in the set are not allowed to be possibly executed at the same time points.

In our interval temporal logic, a set of action constraints can be expressed as a set of temporal formulas of the form:

$$\forall t_1, t_2.(Exec(A_1, t_1, t_1) \wedge Exec(A_2, t_2, t_2) \Longrightarrow t_1 \prec t_2 \vee t_2 \prec t_1)$$

Obviously, the temporal relation $t_1 \neq t_2$ can not be used to replace the relation $t_1 \prec t_2 \vee t_2 \prec t_1$. They are equivalent in linear models, but not in nonlinear models. Like a domain constraint, if an action constraint of the above form is satisfied in a nonlinear plan, then the two actions in the constraint will never be executed at the same time point in any completion of the plan.

We intend to use action constraints to replace domain constraints to detect conflicts in nonlinear plans. However, given a domain constraint, it is obviously impossible to find a logically equivalent set of action constraints for the domain constraint. Fortunately, we do not expect a set of action constraints which is logically equivalent to the set of domain constraints. What we expect is to find a set of action constraints which can be used to replace the set of domain constraints to detect conflicts of a specific set of nonlinear plans, the nonlinear plans whose temporal models satisfy the following restriction.

Definition 8 (Planning Rule Consistency Restriction) *A temporal model $\mathcal{M}$ of a nonlinear plan P is said to be* planning rule consistent *if the following condition holds. For any two intervals over which the same property p holds true, if both the intervals are started by the same action at the same time point (or both of them start from the initial time point) in the model, then the two intervals must be finished at the same time point.*

Fig. 1 shows part of a temporal model which is not planning rule consistent. In the figure, the two intervals (t_i, t_j) and (t_i, t_k) of the property $Clear(B)$ are started by the same action $Unstack(A, B)$ at the same point t_i, but they are finished at the different time points t_j and t_k respectively. Therefore, the temporal model shown in this figure is not planning rule consistent.

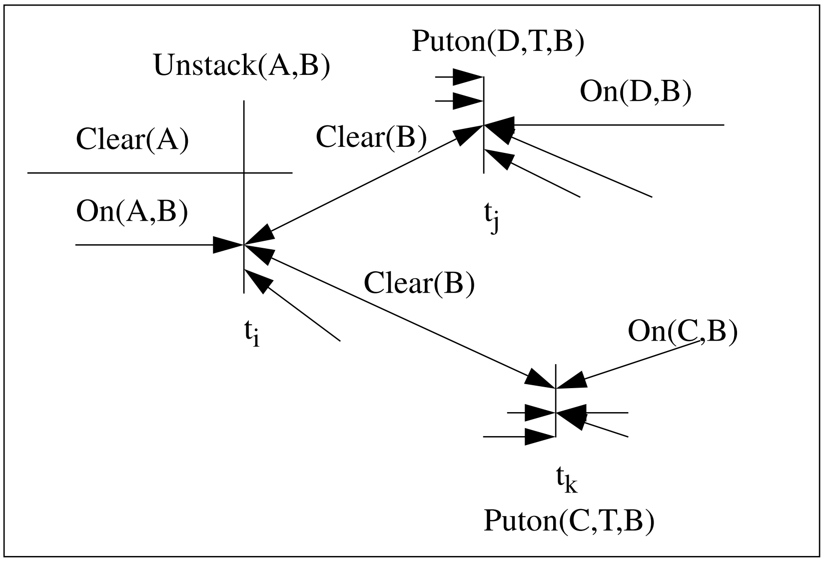

Figure 1: Part of a planning rule inconsistent model.

The temporal model of an arbitrary nonlinear plan is either planning rule consistent or planning rule inconsistent. In this report, however, we are only interested in the planning rule consistent temporal models because of the following two reasons.

1. They represent the temporal models of the nonlinear plans that are constructed using our deductive nonlinear planning method, constraint logic planning.

2. The temporal models that are planning rule inconsistent are models of the nonlinear plans that contain conflicts.

The first reason will be discussed later in this report. The second reason is justified by the following proposition.

Proposition 1 (Planning Rules Consistency) *Let Γ be a set of actions and Σ_D be the corresponding set of domain constraints such that Γ and Σ_D satisfy the restrictions. Let P be a nonlinear plan that consists of actions from Γ and $\mathcal{M}$ be its temporal model. If $\mathcal{M}$ is not planning rule consistent, then P contains conflicts.*

Proof Because the model is not planning rule consistent, there exist two intervals of the same property that start from the same time point and finish at different time points. Then, the necessary-biggest-interval axioms Σ_B is not satisfied in $\mathcal{M}$. By Theorem.3 (in page 6), P is not conflict free. ■

3.3 Transformation from Domain Constraints to Action Constraints

The following definition defines a transformation from a set of domain constraints and a set of actions to a set of action constraints. The intuitive meaning behind the transformation is that, if two properties are mutually exclusive with each other, then any two actions which assert them respectively or which delete them respectively should not be executed possibly at the same time point.

Definition 9 (Constraints Transformation) *Given a set of domain constraints Σ_D and a set of actions Γ, the set of action constraints Σ_Γ, defined as follows, is said to be constructed from Σ_D for Γ.*

$$\begin{aligned}\Sigma_\Gamma \;=\; & \{\forall t_i, t_j.(Exec(A_i, t_i, t_i) \wedge Exec(A_j, t_j, t_j) \Longrightarrow t_i \prec t_j \vee t_j \prec t_i) : \\ & \textit{if } p \leftrightsquigarrow q \in \Sigma_D \textit{ and} \\ & A_i\ A_j \textit{ assert the properties } p \textit{ and } q \textit{ respectively, or} \\ & A_i\ A_j \textit{ delete the properties } p \textit{ and } q \textit{ respectively.}\}\end{aligned}$$

Following the practice for domain constraints, we often use the notation $A_i \leftrightsquigarrow A_j$ to represent the action constraint of the form:

$$\forall t_i, t_j.(Exec(A_i, t_i, t_i) \wedge Exec(A_j, t_j, t_j) \Longrightarrow t_i \prec t_j \vee t_j \prec t_i)$$

Example 2 *In the blocks world, let the set Γ consist of only one point action schemata $Puton(x, y, z)$ given in Example 1. The set of domain constraints Σ_D according to which Γ satisfies the restrictions can be redefined as follows.*

$$\begin{aligned}\Sigma_D \;=\; & \{On(x, y) \leftrightsquigarrow On(x, y')|_{y \neq y'}, \\ & On(x, y) \leftrightsquigarrow On(x', y)|_{x \neq x'}, \\ & On(x, y) \leftrightsquigarrow On(y, x), \\ & On(x, y) \leftrightsquigarrow Clear(y)\}\end{aligned}$$

In Γ, the actions $Puton(x, z, y)$ and $Puton(y, z', x)$ asserts the properties $On(x, y)$ and $On(y, x)$ respectively. Therefore, from the domain constraint $On(x, y) \leftrightsquigarrow On(y, x)$, we obtain an action constraint:

$$Puton(x, z, y) \leftrightsquigarrow Puton(y, z', x) \in \Sigma_\Gamma$$

Also, because the actions $Puton(x, y, z)$ and $Puton(y, x, z')$ delete the properties $On(x, y)$ and $On(y, x)$ respectively, from the same domain constraint, we obtain another action constraint:

$$Puton(x, y, z) \leftrightsquigarrow Puton(y, x, z') \in \Sigma_\Gamma$$

In a similar way, from the other three domain constraints, we can obtain another six action constraints. By properly rename the variables of the action constraints, the set of all the action constraints Σ_Γ obtained from Σ_D for Γ are listed as follows.

$$\begin{aligned}\Sigma_\Gamma = \{ & Puton(x,z,y) \leftrightsquigarrow Puton(x,z',y')|_{y\neq y'},\\ & Puton(x,y,z) \leftrightsquigarrow Puton(x,y',z')|_{y\neq y'},\\ & Puton(x,z,y) \leftrightsquigarrow Puton(x',z',y)|_{x\neq x'},\\ & Puton(x,y,z) \leftrightsquigarrow Puton(x',y,z')|_{x\neq x'},\\ & Puton(x,z,y) \leftrightsquigarrow Puton(y,z',x),\\ & Puton(x,y,z) \leftrightsquigarrow Puton(y,x,z'),\\ & Puton(x,z,y) \leftrightsquigarrow Puton(x',y,z'),\\ & Puton(x,y,z) \leftrightsquigarrow Puton(x',z',y)\}\end{aligned}$$

Suppose that, on average, one property can be asserted by n actions and can be deleted by m actions. According to the above definition, if there are k domain constraints, then the number of the corresponding action constraints is

$$(n^2 + m^2) \times k$$

In the above example, each property is asserted and denied only by one action schemata. The number of the action constraints is $(1^2 + 1^2) \times 4 = 8$.

For any specific problem domain, the transformation from domain constraints to action constraints needs to be done only once.

4 Action Constraint Maintenance

Action constraint maintenance detects conflicts in a nonlinear plan by checking if the corresponding set of action constraints is consistent with the nonlinear plan. In this section, we will prove that action constraint is sound and complete and we will show that, being applied in our nonlinear planning formal framework, action constraint maintenance can be significantly simplied.

4.1 Soundness of Action Constraint Maintenance

The action constraints consists of actions and the temporal relations among them only. Therefore, according to the definition of temporal model, the fact that a nonlinear plan satisfies a set of action constraints is equivalent to the fact that its temporal model satisfies the set of action constraints. The proof of the following proposition is straightforward.

Proposition 2 *Let P be a nonliner plan, $\mathcal{M}$ be its temporal model and Σ_Γ be a set of action constraints. P satisfies Σ_Γ if and only if the $\mathcal{M}$ satisfies Σ_Γ.*

Action constraint maintenance is said to be sound if, under certain conditions, the fact that a nonlinear plan P does not satisfy the corresponding set of action constraints implies that P contains conflicts. We will not prove this directly. We have already established the soundness for domain constraint maintenance, and we will make use of this result to establish the soundness for action constraint maintenance. The following theorem states that, under certain conditions, if a nonlinear plan P does not satisfy the corresponding set of action constraints Σ_Γ, its temporal model will not satisfy the corresponding set of domain constraints Σ_D.

Theorem 5 (Soundness of Action Constraint Maintenance) *Let Γ be a set of actions and Σ_Γ be a set of action constraints constructed from a set of domain constraints Σ_D for Γ. Suppose that Σ_D and Γ satisfy the restrictions. For the temporal model $\mathcal{M}$ of any nonlinear plan, if $\mathcal{M}$ satisfies Σ_D, then M satisfies Σ_Γ.*

Proof: By contradiction. Suppose that Σ_D is satisfied and the action constraint

$$(\forall t_i, t_j.(Exec(A_i, t_i, t_i) \wedge Exec(A_j, t_j, t_j) \Longrightarrow t_i \prec t_j \vee t_j \prec t_i)) \in \Sigma_\Gamma$$

is not satisfied in the model. According to Definition of Constraints Transformation (Definition 9 in page 9), there is a pair of properties p and q such that $p \leftrightsquigarrow q \in \Sigma_D$ and p and q are either asserted or deleted by A_i and A_j respectively. Suppose that p and q are asserted by A_i and A_j respectively, then there exist t'_i and t'_j such that $Hold(p, t_i, t'_i)$ and $Hold(q, t_j, t'_j)$ hold true in the model. Because Σ_D is satisfied in the model, then either $t_i \prec t'_i \preccurlyeq t_j$ or $t_j \prec t'_j \preccurlyeq t_i$ is satisfied in the model. This contradicts the assumption that the model falsifies $A_i \leftrightsquigarrow A_j$. The proof for the case that p and q are deleted by the actions is similar. ■

Here, we have proved that, under certain conditions, the fact that a nonlinear plan does not satisfy the corresponding set of action constraints implies that its temporal model does not satisfy the corresponding set of domain constraints. The fact that the temporal model of a nonlinear plan does not satisfy the corresponding set of domain constraints implies that the nonlinear plan contains conflicts (Theorem 2). Therefore, we proved the soundness of action constraint maintenance.

4.2 Completeness of Action Constraint Maintenance

Action constraint maintenance is said to be complete if the fact that a nonlinear plan contains conflicts implies that the nonlinear plan does not satisfy the corresponding set of action constraints. Before we prove this completeness, we prove a preparation proposition first.

Proposition 3 *Let Γ be a set of actions and Σ_D be a set of domain constraints such that Γ and Σ_D satisfies restrictions. Let Σ_Γ be the set of action constraints constructed from Σ_D for Γ. If two different actions A_i and A_j in Γ have at least one common deleted property or one common asserted property, then $A_i \leftrightsquigarrow A_j \in \Sigma_\Gamma$.*

Proof By the minimum restriction, either A_i has a deleted property which is mutually exclusive with a deleted property of A_j or A_i has an asserted property which is mutually exclusive with an asserted property of A_j. By Definition 9, $A_i \leftrightsquigarrow A_j \in \Sigma_\Gamma$. ■

As for soundness of action constraints maintenance, we will prove the completeness for action constraint maintenance indirectly through the corresponding result for domain constraint maintenance. The following theorem implies that, under certain conditions, if the temporal model of a nonlinear plan does not satisfy the corresponding set of domain constraints Σ_D, the nonlinear plan does not satisfy the corresponding set of action constraints.

Theorem 6 (Completeness of Action Constraint Maintenance) *Let Γ be a set of actions and Σ_Γ be a set of action constraints constructed from a set of domain constraints Σ_D for Γ. Σ_D and Γ satisfy the restrictions. Let P be a nonlinear plan such that its temporal model $\mathcal{M}$ is planning rule consistent. If $\mathcal{M}$ satisfies the set of action constraints Σ_Γ, then $\mathcal{M}$ satisfies the set of domain constraints Σ_D.*

Proof By contradiction. Suppose that Σ_Γ is satisfied in the model and Σ_D is not. Let (t_i, t_i') over which p holds and (t_j, t_j') over which q holds be the intervals that are the relatively nearest ones to the initial time point T_0 such that $p \leftrightsquigarrow q \in \Sigma_D$ and neither $t_i' \preccurlyeq t_j$ nor $t_j' \preccurlyeq t_i$ is satisfied in the model.

Because the initial conditions for any planning problems should be consistent, then, at least, either (t_i, t_i') of p or (t_j, t_j') of q is asserted by an action. Without loss of generality, we assume that (t_i, t_i') of p is asserted by action A_i at t_i and $t_i \prec t_j$ does not hold in the model. Then there are four possibilities about the temporal relation between t_i and (t_j, t_j').

Case 1. $t_i = t_j$. By the consistency restriction, (t_i, t_i') of p and (t_j, t_j') of q are asserted by two different actions A_i and A_j respectively at the same time point $t_i = t_j$. Because $p \leftrightsquigarrow q \in \Sigma_D$, by Definition 9 (in page 9), $A_i \leftrightsquigarrow A_j \in \Sigma_\Gamma$. This contradicts the assumption that the temporal model satisfies Σ_Γ.

Case 2. $t_j \prec t_i \prec t_j'$. By the inclusion restriction, A_i has a precondition r such that either $r \leftrightsquigarrow q \in \Sigma_D$ or $r = q$ and r is a deleted property of A_i. The fact that $r \leftrightsquigarrow q \in \Sigma_D$ contradicts the nearest to the initial point assumption. Therefore, $r = q$. See Fig. 2

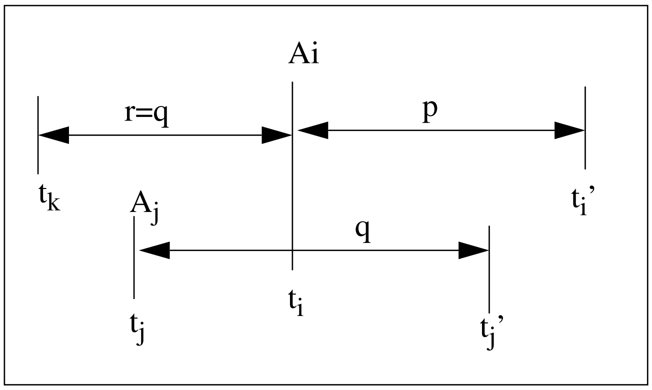

Figure 2: The temporal model in Case 2.

There are two possibilities about the actions which assert the q over (t_j, t_j') and the r over (t_k, t_i).

Subcase 1. They are asserted by the same action.

In this case, if the actions that assert the two intervals are executed at the same time point, i.e. $t_k = t_j$, or both r over (t_k, t_i) and q over (t_j, t_j') are initial conditions, i.e. $t_k = t_j = T_0$, then by the planning rule consistency restriction, the two interval should be finished at the same time point, that is $t_i = t'_j$. This contradicts the assumption that $p \leftrightsquigarrow q$ is not satisfied. If the two actions are executed at different time points, then $t_k \neq t_j$, $t_k \neq T_0$ and $t_j \neq T_0$ according to Definition. 6 (in page 4). Then, no matter what the temporal relation between t_k and t_j is, it is not difficult to show that the case contradicts the nearest to the initial point assumption.

Subcase 2. The r over (t_k, t_i) and the q over (t_j, t'_j) are asserted by two different actions A_k and A_j or, at most, one of them is an initial condition.

If the two actions are executed at the same time point, $t_k = t_j$, then by Proposition 3 , $A_k \leftrightsquigarrow A_j \in \Sigma_\Gamma$. This contradicts the assumption that Σ_Γ is satisfied in the model.

If the two actions are executed at different time points, $t_k \neq t_j$, then no mater what the temporal relation between t_k and t_j is, it is straightforward to show that the case contradicts the nearest to the initial point assumption.

Case 3. $t_j \prec t_i$ and t_i has no relation with t'_j. The proof of this case is similar to that of Case 2.

Case 4. Neither $t_j \prec t_i$ nor $t_i \preccurlyeq t_j$ is satisfied in the model. In this case, neither t_i nor t_j can be unified with the initial point. Then the q over (t_j, t'_j) must be asserted by an action A_j. Because the case that the precondition r of A_i is mutually exclusive with q contradicts the nearest to the initial point assumption, r must be equal to q. By the complement restriction, r is mutually exclusive with a deleted property r' of A_j. This also contradicts the nearest assumption.

■

The above completeness result is only suitable for the nonlinear plans whose temporal models are planning rule consistent. We have already proved that, if the temporal model of a nonlinear plan is not planning rule consistent, the nonlinear plan contains conflicts (Proposition 1). Now, we prove that, for all the nonlinear plans that are constructed using our deductive nonlinear planning method, their temporal models are planning rule consistent.

In (Zhang, 1994), we proposed a novel deductive nonlinear planning approach, *constraint logic planning*. For a given planning problem, the main planning process of this method can be informally described as follows.

> The process concerns a set of goals and a nonlinear plan under construction. The initial contents of the set of goals are that of the planning problem and the initial plan under construction contains the temporal restrictions for the goals.
>
> To achieve a goal of the form $Hold(p, t_1, t_2)$, if it is unifiable with an initial condition, then bind the variables in the goal and the corresponding variables in the plan under construction as long as the plan under construction remains consistent. Otherwise, try to resolve it with a planning rule. If there exists a planning rule in Γ of the form:
>
> $$Hold(p', t, SKF(t, v_1, \ldots, v_m)) :- B_1, \ldots, B_n, Exec(A, t, t), C;$$
>
> such that

1. there exists a most general unifier θ for p and p' and
2. the plan under construction P with the new binding $P\theta$ is consistent with the action $Exec(A,t,t)\theta$, the temporal binding $t_1 = t \wedge t_2 = SKF(t, v_1, \ldots, v_m)\theta$ and the temporal constraint $C\theta$,

then resolve the goal with the rule, add $B_1\theta, \ldots, B_n\theta$ to the set of the goals and expand the plan under construction P to $(P \wedge Exec(A,t,t) \wedge C \wedge t_1 = t \wedge t_2 = SKF(t, v_1, \ldots, v_m))\theta$. Unify the actions and the temporal constraints in the resulting plan if possible. Repeat this process until all the goals are achieved.

We will not discuss our constraint logic planning in detail here. The above brief description is enough for the purpose of justifying action constraint maintenance. Before we justify the planning rule consistency property of the nonlinear plans constructed by the above process, we prove the following preparation proposition.

Proposition 4 *Let Γ be a set of actions and Σ_D be a set of domain constraints such that Γ and Σ_D satisfy the restrictions. Let Σ_Γ be the set of action constraints constructed from Σ_D for Γ, P be a nonlinear plan constructed out of Γ by the planning process described above and P is consistent with Σ_Γ. For any action A in P and for any two different time points t_i t_j, if A is executed at t_i and t_j in P, then either $t_i \prec t_j$ or $t_j \prec t_i$ holds true in P.*

Proof By contradiction. We can assume that A at t_1 and A at t_2 be the relatively nearest pair to the T_0 in P such that neither $t_1 \prec t_2$ nor $t_2 \preccurlyeq t_1$ holds in P. Let p be a precondition of A, then p for A at t_1 and p for A at t_2 must be achieved by two actions A_i at t_i and A_j at t_j such that either $t_i \prec t_j$ or $t_j \prec t_i$ holds in P.

By the inclusion restriction, the complement restriction, the nearest assumption, the above planning process and the condition that P satisfies Σ_Γ, there exists a chain over P such that, in any action-free section of the chain, the two overlapping intervals are either of the same property or of the two different properties which are mutually exclusive. See Figure 3. This chain will finally reaches the initial time point T_0. The case that the last two overlapping intervals on the chain are of two mutually exclusive properties contradicts the consistency requirement for initial conditions. Therefore, the last pair of overlapping intervals on the chain are of the same property. By the above planning process, the Skolem constant of the corresponding initial condition for the property will unify the next pair of actions, which finish the two intervals, in the chain. Then either such a unification contradicts the condition that P satisfies Σ_Γ, or such unifications go on backward to unify t_1 and t_2. ■

Now we can prove that, if a nonlinear plan is constructed from the above process and it satisfies the corresponding set of action constraints, its temporal model is planning rule consistent.

Proposition 5 *Let Γ be a set of actions and Σ_D be a set of domain constraints such that Γ and Σ_D satisfy the restrictions. Let Σ_Γ be the set of action constraints constructed from Σ_D for Γ and P be a nonlinear plan constructed out of Γ by the planning process described above. If P satisfies Σ_Γ, then the temporal model of P is planning rule consistent.*

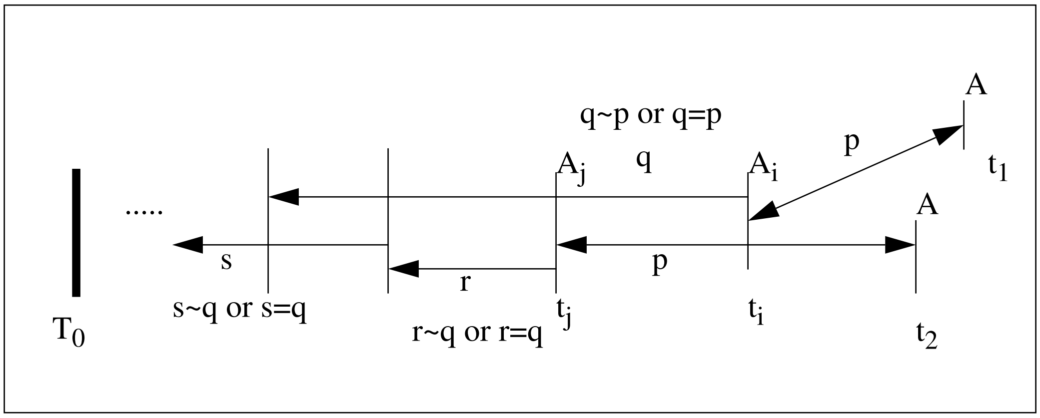

Figure 3: A mutual exclusiveness/equal chain.

Proof By contradiction. If the temporal model $\mathcal{M}$ of P is not planning rule consistent, then P must contain some actions A_i at t_i that asserts a property p together with some A_i' at t_i' and some A_i'' at t_i'' that delete the property p such that, either $\mathcal{M}$ contains a part shown in Fig. 4a or $\mathcal{M}$ contains a part as shown in Fig. 4b.

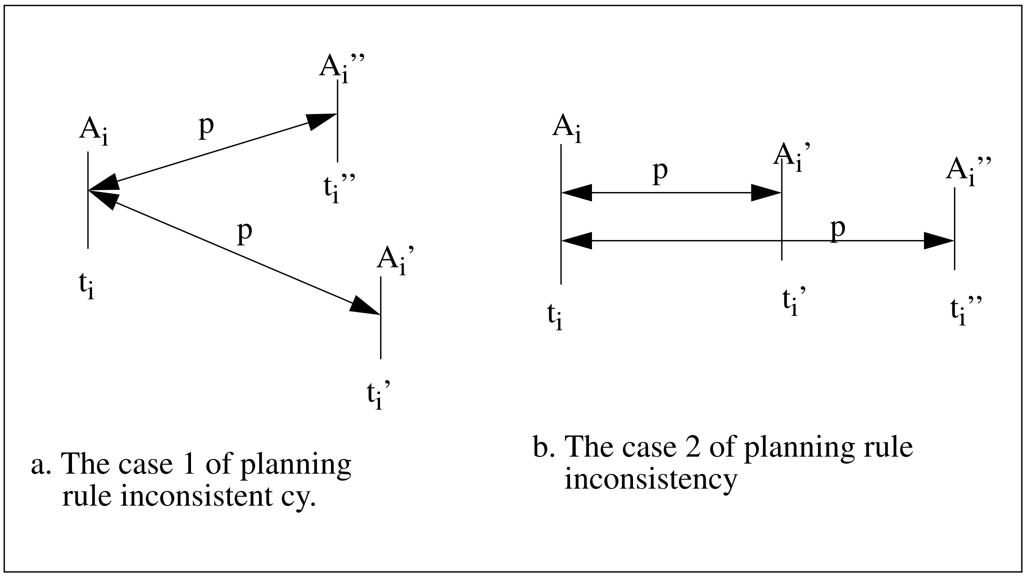

Figure 4: The possible cases that $\mathcal{M}$ is not planning rule consistent.

Case 1. $\mathcal{M}$ contains a part as shown in Fig. 4a. In this case, by Proposition 4, A_i' and A_i'' must be two different actions. Because A_i' and A_i'' delete one common property p, by Proposition 3 (in page 11), $A_i' \leftrightsquigarrow A_i'' \in \Sigma_\Gamma$. Because P satisfies Σ_Γ, either $t_i' \prec t_i''$ or $t_i'' \prec t_i'$ which contradict the premiss of this case.

Case 2. $t'_i \prec t''_i$ holds in $\mathcal{M}$. According to Definition of Temporal Model (Definition 6 in page 4) and the description of the planning process, there must be an action A_j at t_j in P such that $t_j \prec t'_i$ (or $t_j \prec t''_i$) holds in P and A_j at t_j is introduced in the planning process to assert p for A'_i at t'_i (or A''_i at t''_i). Otherwise, $t'_i = t''_i$ holds in P. For the same reason, if A_j and A_i are the same action, $t_j \prec t_i$ must hold in the model. If A_j and A_i are different actions, by Proposition 3 and the assumption that P satisfies Σ_Γ, $t_j \prec t_i$ also must hold in P. Therefore, in either case, $t_j \prec t_i$ holds in P.

Without loss of generality, we can assume that, in the planning process, the precondition p of A'_i at t'_i is achieved by introduction of the action A_j at t_j and that of A''_i at t''_i is achieved by introduction of A_i at t_i and the temporal relations $t_j \prec t_i \prec t'_i \prec t''_i$ hold in the plan P.
By the complement restriction, A_i must have a deleted property q such that $q \leftrightsquigarrow p \in \Sigma_D$ and q either is asserted by an action A_k at t_k or is an initial condition started from $t_k = T_0$. Because $p \leftrightsquigarrow q \in \Sigma_D$, by Definition 9 (in page 9) and the assumption that P satisfies Σ_Γ, either $t_j \prec t_k$ or $t_k \prec t_j$ holds in P. See Fig. 5.

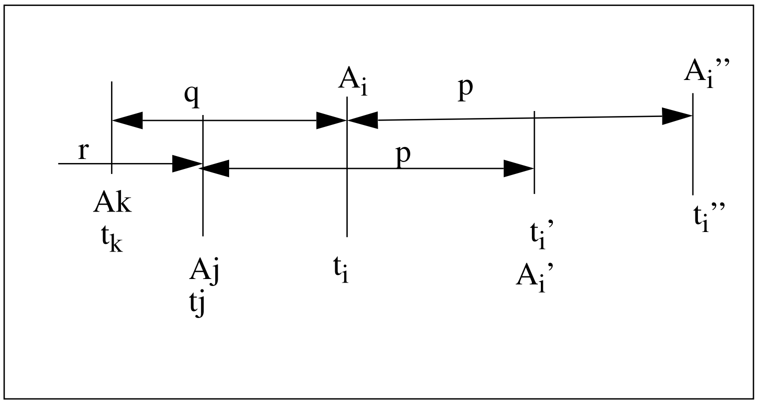

Figure 5: The figure for the case 2.

Therefore, if the temporal model of P is not planning rule consistent, there must exist two actions A_i at t_i and A'_i at t'_i in P which delete q and p respectively such that $p \leftrightsquigarrow q \in \Sigma_D$. These two properties must be achieved in the planning process by the introduction of two other actions A_k at t_k and A_j at t_j such that the intervals (t_k, t_i) and (t_j, t'_i) necessarily and properly overlap in P. We prove this is an impossible case by contradiction.

Suppose that $t_k \prec t_j \prec t_i \prec t'_i$ holds in P and this is one of the such cases that is the nearest to the initial time point in P.

If both p and q are initial conditions, then the case contradicts the requirement that the set of initial conditions for a problem be consistent. Without of loss generality, we can assume that $t_k \prec t_j$ and the p over (t_j, t'_i) is not an initial condition. Then, by the inclusion restriction, A_j must have a precondition r such that either $r \leftrightsquigarrow q \in \Sigma_D$ or $r = q$ and r is a deleted property of A_j. Suppose r is started from

t_l, then it is straightforward to show that either $t_l \prec t_k$ or $t_k \preccurlyeq t_l$. The case that $r \leftrightsquigarrow q$ contradicts the nearest assumption; therefore, $r = q$. By the complement restriction, both the case $t_l \prec t_k$ and $t_k \prec t_l$ contradict the nearest assumption. The only possibility left is that $t_l = t_k$. By Proposition 3, r over (t_l, t_j) and q over (t_k, t_i) either are asserted by the same action or are both initial conditions. In both cases, according to the description of the planning process, the Skolem function or the Skolem constant of the corresponding planning rule binds t_j and t_i, $t_i = t_j$, and this contradicts the assumption that (t_j, t_i') of p and (t_k, t_i) of q necessarily and properly overlap.

■

Now we can prove the main theorem for action constraint maintenance. This theorem states that, for any nonlinear plans that are constructed from our constraint logic planning, it is conflict free if and only if it satisfies the corresponding set of action constraints.

Theorem 7 (Action Constraint Maintenance) *Let Γ be a set of actions and Σ_D be a set of domain constraints such that Γ and Σ_D satisfy the restrictions. Let Σ_Γ be the set of action constraints constructed from Σ_D for Γ and P be a nonlinear plan constructed out of Γ by the planning process in our formal framework. Then P is conflict free if and only if P satisfies Σ_Γ.*

Proof Suppose that $\mathcal{M}$ is the temporal model of P.

$\Longrightarrow$

The fact that P is conflict free
implies that $\mathcal{M}$ satisfies Σ_D by Theorem 2 (in page 6) and this
implies that $\mathcal{M}$ satisfies Σ_Γ by Theorem 5 (in page 11) and this
implies that P satisfies Σ_Γ by Proposition 2 (in page 10).

$\Longleftarrow$

The fact that P satisfies Σ_Γ
implies that $\mathcal{M}$ satisfies Σ_Γ by Proposition 2 (in page 10) and
implies that $\mathcal{M}$ is planning rule consistent by Proposition 5 (in page 14).
The fact that $\mathcal{M}$ satisfies Σ_Γ and $\mathcal{M}$ is planning rule consistent
implies that $\mathcal{M}$ satisfies Σ_D by Theorem 6 (in page 12) and this
implies that P is conflict free by Theorem 4 (in page 6).

■

4.3 Simplification of Action Constraint Maintenance

It is not difficult to see that, if a nonlinear plan does not satisfy an action constraint, some of its possible expansions may still satisfy the constraint. A nonlinear plan P is said to be an expansion of another nonlinear plan P' if P' is a subplan of P or P' is a instance of a subplan of P. In this section, we will show that action constraints can be replaced by *simplified action constraints*, and, if a nonlinear plan does not satisfy a simplified action constraint, all its possible expansions will not satisfy the constraint.

An action constraint $a_1 \leftrightsquigarrow a_2$ has been defined to be of the form

$$\forall t_1, t_2.(Exec(a_1, t_1, t_1) \wedge Exec(a_2, t_2, t_2) \Longrightarrow t_1 \prec t_2 \vee t_2 \prec t_1)$$

The following theorem shows that, applied in our planning framework, such an action constraint can be replaced by a much simpler formula of the form:

$$\forall t_1, t_2.(Exec(a_1, t_1, t_1) \wedge Exec(a_2, t_2, t_2) \Longrightarrow t_1 \neq t_2)$$

Checking such a simplified action constraint for a nonlinear plan, we need not calculate the transitive closure for the partial order relation of the plan. We only need check if the two actions of the action constraint are executed at the same time point in the plan and this is a simple task.

Theorem 8 *Let Γ be a set of actions and Σ_D be a set of domain constraints such that Γ and Σ_D satisfy the restrictions. Let Σ_Γ be the set of action constraints constructed from Σ_D for Γ and P be a nonlinear plan constructed out of Γ by the given planning process. For any action pair A_i and A_j such that $A_i \leftrightsquigarrow A_j \in \Sigma_\Gamma$, P satisfies*

$$\forall t_1, t_2.(Exec(A_1, t_1, t_1) \wedge Exec(A_2, t_2, t_2) \Longrightarrow t_1 \neq t_2)$$

if and only if P satisfies

$$\forall t_1, t_2.(Exec(A_1, t_1, t_1) \wedge Exec(A_2, t_2, t_2) \Longrightarrow t_1 \prec t_2 \vee t_2 \prec t_1)$$

Proof It is straightforward that the fact that P satisfies the formula $\forall t_1, t_2.(Exec(A_1, t_1, t_1) \wedge Exec(A_2, t_2, t_2) \Longrightarrow t_1 \prec t_2 \vee t_2 \prec t_1)$ implies that P satisfies $\forall t_1, t_2.(Exec(A_1, t_1, t_1) \wedge Exec(A_2, t_2, t_2) \Longrightarrow t_1 \neq t_2)$.

Now, we prove the opposite implication of this theorem by contradiction. Let A_i at t_i and A_j at t_j be the relatively nearest pair of actions to the initial time points such that $A_i \leftrightsquigarrow A_j \in \Sigma_\Gamma$ and neither $t_i \prec t_j$ nor $t_j \prec t_i$ is satisfied in the nonlinear plan. Without loss of generality, we can assume A_i asserts p and A_j asserts q such that $p \leftrightsquigarrow q \in \Sigma_D$.

First, we prove that A_i has a precondition r and A_j has a precondition s such that either $r \leftrightsquigarrow s \in \Sigma_D$ or $r = s$. Because $p \leftrightsquigarrow q \in \Sigma_D$, by the inclusion restriction, A_i has a precondition r such that $r \leftrightsquigarrow q \in \Sigma_D$ or $r = q$. By the complement restriction, $r = q$ implies that A_j has a precondition s such that $r \leftrightsquigarrow s \in \Sigma_D$. By the inclusion restriction, $r \leftrightsquigarrow q$ implies that A_j has a preconditions s such that $r = s$ or $r \leftrightsquigarrow s \in \Sigma_D$. Now we prove the theorem for these two possible cases.

Case 1. $r = s$.

If r and s are initial conditions, then the corresponding Skolem constant in the initial condition will unify the time points t_i and t_j. This contradicts the assumption that $t_i \neq t_j$. Suppose that r is asserted by an action A'_i at t'_i. Because of the nearest assumption, no matter whether s is an initial condition ($t'_j = T_0$) or not (s is asserted by an action at t'_j), we can assume either $t'_j \prec t'_i$ or $t'_j = t'_i$. For the case $t'_j \prec t'_i$, because $r = s$, by the complement restriction, A'_i has a precondition u such that $u \leftrightsquigarrow s \in \Sigma_D$. By the nearest assumption, the temporal interval of u and the temporal interval of s necessarily and properly overlap. Also by the nearest assumption, for any action pair A_k at t_k and A_l and t_l such that $A_k \leftrightsquigarrow A_l \in \Sigma_\Gamma$ and $t_k, t_l \prec t_j$, either $t_k \prec t_l$ or $t_l \prec t_k$. Then the case that the temporal interval of u and the interval of s necessarily and properly overlap is impossible. The proof of this is similar to the proof for Proposition 5 (Case 2).

The proof for the case that $t'_j = t'_i$ is also similar to that of Proposition 5 (Case 2).

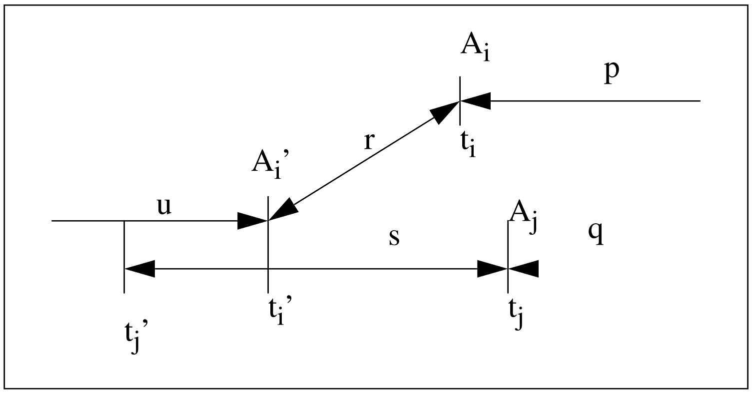

Figure 6: The temporal modal of Case 1.

Case 2. $r \leftrightsquigarrow s$.

The case that both r and s are initial conditions contradicts the consistency assumption of initial conditions. Then, we can assume that, at least, r is asserted by an action A_i' at t_i'. Then the proof for this case if similar to that for Case 1 and to that for Proposition 5 (Case 2). ■

The importance of this theorem comes from the fact that, if a nonlinear plan is not consistent with such a simplified action constraint, all its possible expansions will not be consistent with that constraint. Therefore, action constraint maintenance can be employed to maintain the conflict freeness of a nonlinear plan through the plan construction process, and the above property guarantees that backtracking from inconsistencies will not lose possible solutions. The proof of the following theorem is straightforward.

Theorem 9 *Let P be a nonlinear plan and Σ_Γ be a set of action constraints (in simplified form). If P is not consistent with Σ_Γ, all possible expansions of P will not be consistent with Σ_Γ.*

5 Discussion

In this report, we propose a novel conflict detecting method, action constraint maintenance, for nonlinear plans. We proved that action constraint maintenance is sound and complete, i.e. under certain conditions, a nonlinear plan is conflict free if and only if it satisfies the corresponding set of action constraints.

Action constraint maintenance is much more efficient than other conflict detecting methods for nonliner plans. According to Theorem 8, to check that a nonlinear plan satisfies an action constraint, we just need to check if the two actions of the action constraint are executed exactly at the same time point in the plan. For a given nonlinear plan P, this conflict detecting method has the worst case time complexity of $O(|P|^2)$. This is much better than that for domain constraint maintenance ($O(|P|^4)$ (Zhang *et al.*, 1994)).

Conventionally, conflicts in a nonlinear plan can be detected by checking that if all the properties are *necessarily* true in the sates where they are needed (Chapman, 1987). Adopting this method to detect conflicts in a nonlinear plan P, one needs to check that, for each precondition p of each of the $|P|$ actions A_i in the plan, there exists an action A_j in the $|P|$ actions and A_j necessarily asserts p and A_j necessarily precedes A_i; and for every other action A_k (in the $|P|$ actions) which may possibly deny p, either A_k is not possibly between A_j and A_i, or there exists another action A_l ("white knight") in the $|P|$ actions such that A_l necessarily asserts p and A_l is necessarily between A_k and A_i. The worst case time complexity of this method is also $O(|P|^4)$ and is worse than that of action constraint maintenance.

For a given nonlinear plan, each of its completions is a linear plan. A nonlinear plan is conflict free if all its possible completions are conflict free. Therefore, conflicts in a nonlinear plan P can also be detected by checking that, for every precondition p of each of the $|P|$ actions A_i in each of the $|P|! \times n^m$ (the worst case) possible completions, where m is number of variables in P and n is the average number of possible values for each variable, there exists an action A_j in the $|P|$ actions such that A_j asserts p and A_j precedes A_i; and all the other actions ($|P|-2$ in the worst case) which are between A_j and A_i do not deny p. The worst case time complexity of this method is $O(|P|! \times |P|^3 \times n^m)$ and is, therefore, much worse than other methods.

In addition to the efficiency, the most significant advantage of action constraint maintenance over other conflict detecting methods is that if a nonlinear plan is not consistent with an action constraint, all its possible expansions will not be consistent with the action constraint. This property allows action constraint maintenance to be employed to maintain conflict freeness of a nonlinear plan during the plan construction process.

Directly based upon action constraint maintenance, in (Zhang, 1994), we built a deductive nonlinear planning framework: constraint logic planning. Constraint logic planning employs the *least commitment strategy* for both object selection and temporal relation decision. The theoretical results of action constraint maintenance provide the theoretical basis for soundness and completeness of constraint logic planning. Briefly, to solve a planning problem in constraint logic planning is to progressively construct the nonlinear plan by introducing actions and the related temporal relations by means of reasoning about goals and planning rules, and keeping the constructed plan consistent with the corresponding set of action constraints.

In comparison with classical nonlinear planning systems, our constraint logic planning avoids the computation for modal truth criteria and achieves conflict freeness.

In classical nonlinear planning, to "necessarily" achieve a goal, a planner needs to introduce actions and the related temporal relations to assert the goal and to take measures to prevent the goal from being possibly denied by other actions in the plan; i.e., to implement various modal truth criteria (Chapman, 1987; Christensen, 1990; Pednault, 1991) to achieve "necessary truth". By implementing modal truth criteria, what classical nonlinear planners can achieve is only to prevent the most recently achieved goal from being destroyed by the previously introduced actions, not to prevent previously achieved goals from being destroyed by newly introduced actions or by modification of the existing plan.

In our constraint logic planning, what we need to do is to introduce actions and the related temporal relations to achieve goals and to check if the constructed plan satisfies the corresponding action constraints. Action constraint maintenance is, at least, not more computationally complicated than preventing the most recently achieved goal from being destroyed by the previously introduced actions using modal truth criteria. However, constraint logic planning achieves and maintains the necessary truth for all the goals involved in planning processes and, therefore, guarantees conflict freeness.

Furthermore, in constraint logic planning, action constraint maintenance also serves as a domain independent search heuristic. We have showed that a nonlinear plan is inconsistent with an action constraint implies that the temporal model of the plan is not consistent with a domain constraint. By backtracking from inconsistencies of a constructed plan with the corresponding set of action constraints, constraint logic planning can prune the search space to avoid some of the branches which will lead to impossible states specified by the domain constraints.

Because of the space limitation, in this report, we only concern point actions. In (Zhang, 1994), we gave all the corresponding results for interval actions.

6 Acknowledgements

We would like to thank Behnam Bani-Eqbal, David Carlisle, Clare Dixon, Rajeev Goré and Ian Pratt who read an early version of this report and gave valuable comments.

This paper is a modified version of part of Zhang's Ph.D. thesis (Zhang, 1994). The work was partially supported by the British Council and the State Education Commission of China

References

Allen, James F. 1991. Planning as Temporal Reasoning. *Pages 3–14 of: Proceedings of the 12th International Joint Conference on Artificial Intelligence.*

Chapman, David. 1987. Planning for Conjunctive Goals. *Artificial Intelligence*, **32**, 333–377.

Christensen, Jens. 1990. *Automatic Abstraction in Planning.* Ph.D. thesis, Stanford University.

Currie, Ken, & Tate, Austin. 1989 (May). *O-Plan: the Open Planning Architecture.* Tech. rept. AIAI-TR-67. Artificial Intelligence Applications Institute, University of Edinburgh, Edinburgh.

Currie, Ken, & Tate, Austin. 1990. *O-Plan:Task Formalism User Manual.* Tech. rept. AIAI-SR-12. Artificial Intelligence Applications Institute, University of Edinburgh, Edinburgh.

Drummond, Mark, & Currie, Ken. 1987. *Temporal Coherence in Non-linear Plan Construction.* Tech. rept. AIAI-TR-23. Artificial Intelligence Applications Institute, University of Edinburgh, Edinburgh.

Kelleher, G. 1990. *An Automatic Planner for the Support of Users of Information Processing Systems.* Ph.D. thesis, University of Leeds.

Kelleher, G., & Cohn, A. G. 1992. Automatically Synthesising Domain Constraints from Operator Descriptions. *Pages 653–655 of: Proceedings of the 10th European Conference on Artificial Intelligence.*

Pednault, E. P. D. 1991. Generalizing Nonlinear Planning to Handle Complex Goals and Actions with Context-Dependent Effects. *In: Proceedings 12th International Joint Conference on Artificial Intelligence.*

Warren, D. H. D. 1974. *WARPLAN: a System for Generating Plans.* Tech. rept. Department of Computational Logic Meno No. 76. University of Edinburgh, Edinburgh.

Yang, Q. 1989. *Improving the Efficiency of Planning.* Ph.D. thesis, Univ. of Maryland, Computer Science Dept.

Zhang, Ying. 1994 (May). *Constraint Logic Planning: A Formal Framework for Nonlinear Planning.* Ph.D. thesis, Manchester Univ., Computer Science Dept.

Zhang, Ying, & Barringer, Howard. 1994 (July). *A Temporal Logic For Nonlinear Planning.* Tech. rept. MUCS-94-7-1. Department of Computer Science, University of Manchester, Manchester, M13 9PL.

Zhang, Ying, Barringer, Howard, & Carlisle, David. 1994 (August). *Domain Constraint Maintenance.* Tech. rept. MUCS-94-8-1. Department of Computer Science, University of Manchester, Manchester, M13 9PL.

A A Temporal Logic for Nonlinear Planning

A.1 Syntax

Given

DV: a set of data variables,

TV: a set of temporal variables which is disjoint from DV,

DF: a set of data function symbols,

TF: a set of temporal function symbols which is disjoint from DF,

$\mathcal{A}$: a set of primitive action symbols,

$\mathcal{P}$: the set of primitive property symbols which is disjoint from $\mathcal{A}$,

the set of data terms, T_D, is defined recursively as follows

1. if $t \in DV$, then $t \in T_D$,
2. if $f \in DF$, $arity(f) = n$ and $t_1, \ldots, t_n \in T_D$, then $f(t_1, \ldots, t_n) \in T_D$.

The definition of the set of *temporal terms*, T_T, is similar to that of data terms except that the arguments of temporal functions are not restricted to be temporal terms. The set of temporal terms, T_T, is defined as follows:

1. if $t \in TV$, then $t \in T_T$,
2. if $f \in TF$, $arity(f) = n$, and $t_1, \ldots, t_n \in T_T \bigcup T_D$, then $f(t_1, \ldots, t_n) \in T_T$.

The set of *well-formed formulas(wffs)* is defined inductively as follows:

1. If $dt_1, \ldots, dt_n \in T_D$, $tt_1, tt_2 \in T_T$, $P \in \mathcal{P}$, $arity(P) = n$, then $Hold(P(dt_1, \ldots, dt_n), tt_1, tt_2)$ is a wff,
2. if $dt_1, \ldots, dt_n \in T_D$, $tt_1, tt_2 \in T_T$, $A \in \mathcal{A}$, $arity(A) = n$, then $Exec(A(dt_1, \ldots, dt_n), tt_1, tt_2)$ is a wff,
3. if $tt_1, tt_2 \in T_T$, then $tt_1 \prec tt_2$ and $tt_1 \preccurlyeq tt_2$ are wffs,
4. if φ and ψ are wffs, then so are $\neg\varphi$, $\varphi \wedge \psi$,
5. if φ is a wff and $v \in DV \bigcup TV$ is a free variable of φ, then $\forall v.\varphi[v]$ is a wff.

We assume the usual definitions of $\vee$, $\Longrightarrow$, $\Longleftrightarrow$ and $\exists$.

A.2 Semantics

Definition 10 *A* partial order time structure *is a pair* $(\mathcal{W}, \mathcal{R})$ *where* $\mathcal{W}$ *is a non-empty universe of time points and* $\mathcal{R}$ *is a* partial order relation *between the time points in* $\mathcal{W}$.

Definition 11 *An* interpretation *of a temporal language is a tuple :* $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}, \mathcal{D}, \pi_F, \pi_P \rangle$ *where*

$(\mathcal{W}, \mathcal{R})$ *is a partial order time structure,*

$\mathcal{D}$ *is the domain of* data *for all the time points in* $\mathcal{W}$.

π_F *is a interpretation of the function symbols. If* $f \in DF \bigcup TF$ *and* $arity(f) = n$*, then* $\pi_F(f) \in (\mathcal{D} \bigcup \mathcal{W})^n \longrightarrow \mathcal{D} \bigcup \mathcal{W}$.

π_P *is a interpretation of the proposition types (actions and properties). If* $P \in \mathcal{P} \bigcup \mathcal{A}$ *and* $arity(P) = n$*, then* $\pi_P(P) \in \mathcal{D}^n \longrightarrow 2^{\mathcal{W} \times \mathcal{W}}$. *We require that if* $(t_1, t_2) \in \pi_P(P)(dt_1, \ldots, dt_n)$*, then* $(t_1, t_2) \in \mathcal{R}$. *To characterize that assertions are interpreted over the necessarily biggest possible intervals,* π_P *has the property that, if* $(t_1, t_2), (t_3, t_4) \in \pi_P(P)(dt_1, \ldots, dt_n)$ *and* (t_1, t_2) *and* (t_3, t_4) *do not refer to the same time interval, then either* $(t_2, t_3) \in \mathcal{R}$ *or* $(t_4, t_1) \in \mathcal{R}$.

A *variable assignment* ν for a given interpretation $\mathcal{M}$, is a mapping from variable symbols to the elements of the domains, $\nu : (DV \bigcup TV) \longrightarrow (\mathcal{D} \bigcup \mathcal{W})$ which satisfies sort restriction, i.e. $\nu(x) \in \mathcal{D}$ if $x \in DV$ and $\nu(x) \in \mathcal{W}$ if $x \in TV$.

For a given interpretation $\mathcal{M}$ and variable assignment ν, a *term assignment* τ is a mapping, $\tau : (T_D \bigcup T_T) \longrightarrow (\mathcal{D} \bigcup \mathcal{W})$ which can be defined as follows.

1. If $t \in DV \bigcup TV$, then $\tau(t) = \nu(t)$,
2. if $f \in DF \bigcup TF$ and $arity(f) = n$, then $\tau(f(t_1, \ldots, t_n)) = \pi_F(f)(\tau(t_1), \ldots, \tau(t_n))$.

The semantics of the wffs is given with respect to an interpretation $\mathcal{M} = \langle \mathcal{W}, \mathcal{R}, \mathcal{D}, \pi_F, \pi_P, \rangle$ and a variable assignment ν. The *satisfaction relation* ($\models$)of the well formed formulas under such a pair is defined recursively as follows.

$$
\begin{array}{lcl}
\langle \mathcal{M}, \nu \rangle & \models & Hold(P(dt_1, \ldots, dt_n), tt_1, tt_2) \text{ iff} \\
 & & (\tau(tt_1), \tau(tt_2)) \in \pi_P(P)(\tau(t_1), \ldots \tau(t_n)) \\
\langle \mathcal{M}, \nu \rangle & \models & Exec(A(dt_1, \ldots, dt_n), tt_1, tt_2) \\
 & & \text{ iff } (\tau(tt_1), \tau(tt_2)) \in \pi_P(A)(\tau(t_1), \ldots \tau(t_n)) \\
\langle \mathcal{M}, \nu \rangle & \models & t_1 \prec t_2 \text{ iff } (\tau(t_1), \tau(t_2)) \in \mathcal{R} \\
\langle \mathcal{M}, \nu \rangle & \models & t_1 \preccurlyeq t_2 \text{ iff } (\tau(t_1), \tau(t_2)) \in \mathcal{R} \text{ or } \tau(t_1) = \tau(t_2) \\
\langle \mathcal{M}, \nu \rangle & \models & \neg\varphi \text{ iff } \langle \mathcal{M}, \nu \rangle \models \varphi \text{ is false} \\
\langle \mathcal{M}, \nu \rangle & \models & (\varphi \wedge \psi) \text{ iff } \langle \mathcal{M}, \nu \rangle \models \varphi \text{ and } \langle \mathcal{M}, \nu \rangle \models \psi \\
\langle \mathcal{M}, \nu \rangle & \models & \forall v.\varphi \text{ iff } \langle \mathcal{M}, \nu' \rangle \models \varphi \text{ for all } \nu' \text{ which} \\
 & & \text{agree with } \nu \text{ except possibly on } v.
\end{array}
$$

An interpretation $\mathcal{M}$ is a *model* for a wff φ (written as $\mathcal{M} \models \varphi$) if $\langle \mathcal{M}, \nu \rangle \models \varphi$ for all variable assignments ν. A wff is a *closed formula* if it contains no free variables. If a closed formula φ is satisfied by a interpretation $\mathcal{M}$ and a variable assignment, then $\mathcal{M}$ is a model of φ. A wff φ is said to be *satisfiable* if it has a model. A wff φ is said to be *valid* (written as $\models \varphi$) if its negation is not satisfiable.