SMT Spatio-Temporal Planning

Lamia Belouaer  
GREYC, Université de Caen Basse Normandie  
Boulevard du Maréchal Juin  
BP 5186 14032 Caen Cedex, France

Frédéric Maris  
IRIT, Université Paul Sabatier  
118 route de Narbonne  
31062 Toulouse Cedex 9, France

Abstract
Solving real planning problems requires to consider spatial and temporal information. Indeed, to be solved more efficiently many real world problems need to take the action duration, the instants of effects occurrences, the instants of requiring preconditions... and the space in which the mission is accomplished by defining the different actions zones and know the path between these zones into account. In this paper we are interested in planning problems which consider spatial and temporal dimensions. To our knowledge there is no model expressing both spatial and temporal dimensions in planning problems. The main contribution of this paper is to present an approach allowing the resolution of spatio-temporal planning problems. For this purpose we define a new SMT (Sat Modulo Theory) encoding rules for spatio-temporal planning. This new code can compile and solve spatio-temporal planning problems for which all solutions require simultaneous actions in a 2D space.

Introduction
The classical planning framework allows only limited representation of real world aspects. We aim to expand this framework with spatial and temporal aspects while maintaining good performance. In planning applications in real-world these two dimensions are useful. Indeed, to be solved more efficiently many of these problems need to take the action durations, the instants of effects occurrences, the instants of requiring preconditions... and the space in which the mission is accomplished by defining the different actions zones and the path between these zones. For instance, manipulating objects in a nuclear plant, the building evacuation in case of fire, the service in a restaurant... are some examples of such problems. To solve this kind of problem we aim at developing an approach based on spatial and temporal reasoning. However, to our knowledge there is no model expressing planning problem by considering both spatial and temporal dimensions at the same time and there is no planner solving such a problem.

The main contribution of this paper is to present an approach allowing the resolution of spatio-temporal planning problem. The idea is to allow synchronization of non-instantaneous actions in space. To this aim, we consider the encoding rules of TLP-GP-2 (Maris and Régnier 2008a) (Maris and Régnier 2008b) and we integrate new rules based on SpaceOntology (Belouaer, Bouzid, and Mouaddib 2010), and which can encode the spatial dimension of a planning problem.

The temporal planner TLP-GP-2 is based on the use of a simplified planning graph and a SAT Modulo Theory (SMT) solver. The problem representation language that TLP-GP can process allows a much greater flexibility than PDDL2.1 (Fox and Long 2003) for defining temporal domains and problems. Although its expressive power is identical to that of PDDL2.1, the extensions implemented in TLP-GP allow the user to express real-world problems much more easily (Cooper, Maris, and Regnier 2010a). To provide a rich temporal representation of actions, time points within actions can be used, other than start and end, along with sets of simple linear binary (in)equality constraints between time points. On the other hand, high-level modalities allow the user to define complex relationships between a condition or effect and an interval, including being valid over the whole interval, at some point in the interval, within a sub-interval or being subject to a continuous transition over the interval. TLP-GP can also take into account, in a very natural way, exogenous events as well as temporal goals. It is complete for the temporally expressive sublanguages of PDDL2.1 using the transformation method of (Cooper, Maris, and Regnier 2010b) to restore its completeness when handling temporally-cyclic problems.

SpaceOntology considers different space’s representations: qualitative (topological relation and fuzzy distance), quantitative (numeric distance) and hierarchical. Also, SpaceOntology defines a set of rules to deduce new information to complete the description of the environment. This allows a complex spatial description and an easy management of different spatial aspects. In this paper, we consider four concepts defined in SpaceOntology. Namely (Belouaer, Bouzid, and Mouaddib 2010): spatial entity, numeric distance, fuzzy distance, hierarchical representation. This paper presents an extension of the temporal planner TLP-GP-2, called ST-SMTPLAN, to take into account temporal dimension with some spatial aspects of SpaceOntology (numeric distance, fuzzy distance and hierarchical link).

This paper is organized as follows. First, we present vari-
uous related works on spatial and/or temporal planning. Then, the preliminaries part presents a set of required notations and definitions. Next, we define a study case in order to illustrate our claim. After, we present some of the spatio-temporal encoding rules needed to solve the problem described relative to our study case. The discussion section shows theoretically the soundness and the completeness of this two-dimensional encoding. Finally, the conclusion presents the lines of research opened by this work.

Related Work

To solve real planning problems, one of the major challenges is to consider spatial and temporal dimensions. However, to our knowledge there is no model allowing us to represent a planning problem by considering both spatial and temporal knowledge at the same time and no planner allowing us to solve such a problem. This part is a literature review of various works on temporal planners and spatial planners.

Temporal Planners

Most of temporal planning systems use an additional constraints solver for task scheduling. In the past, many planners have used a hierarchical plan-space (HTN). They used a temporal logic based on intervals and intervals, together with a Time Map Manager which manages the temporal constraints. This is the case for planners such as IXTeT (Ghallab and Alaoui 1989), Laborie and Ghallab 1995 or HSTS (Muscazzola 1993). When (Cushing, Kambhampatti, and Mausam 2007) published their important work on temporally-expressive planning, the majority of efficient temporal planners developed between 1994 and 2007 were incapable of solving temporally-expressive problems, although some planners have since been improved to solve such problems. Apart from HTN-type planners which present a relatively poor performance, only few of them can solve this type of problem. Some of them such as CRIKEY3 (Coles et al. 2008), VHPOP (Younes and Simmons 2003), LPGP (Long and Fox 2003), TLP-GP-1 (Maris and Régnier 2008a) (Maris and Régnier 2008b), and the most recent version of LPG (Gerevini, Saetti, and Serina 2010), perform a search algorithm coupled with a Simple Temporal Network (STN) solver. Others such as TMLPSAT (Shin and Davis 2004), the planner of (Hu 2007), TLP-GP-2 (Maris and Régnier 2008a) (Maris and Régnier 2008b), STEP (Huang, Chen, and Zhang 2009), use a similar method to that used by the family of BLACKBOX (Kautz and Selman 1999) classical planners. They simultaneously code a planning graph (Blum and Furst 1995) and temporal constraints, then call the solver (LP, CSP, SMT or SAT) to find a solution.

Spatial Planners

Spatial knowledge is essential in planning. For instance, in the case of building evacuation, to achieve this mission the agents have to move through the environment between the different areas. In literature, there are some planners exploiting spatial knowledge. For example: ASYMOV (Gravot, Cambon, and Alami 2005), one presented in (Guitton et al. 2008) and SPOON (Belouaer, Bouzid, and Mouaddib 2011).

The planner ASYMMOV computes plans for handling problems in which the execution of an action has an important effect on the spatial representation of the problem. For example, when an agent has to carry an object, the shape of the whole agent with the object is different from that of the agent load. The planner described in (Guitton et al. 2008) consists of two modules reasoning: a symbolic reasoning module supported by a planner task and a reasoning module supported by a path planner.

These two planners manage only the quantitative spatial information (numeric distance, coordinates...). This is not enough. For instance, to move through an environment requires a spatial knowledge for describing it. In this kind of application, it is also necessary to know the adjacency relations, the distances between the different regions to compute a path. Spatial knowledge can be quantitative (numerical), qualitative (topological or fuzzy) and hierarchical (simplifies the description of the global environment). Also, These two planners do not express planning problems taking into account the spatial dimension.

SPOON is an hybrid planner combining two reasoning modules. Symbolic reasoning module supported by task planner. Spatial reasoning module supported by path planner and SpaceOntology. This ontology provides a structured knowledge of the explored environment in the planning process. SPOON uses Space-PDDL as a language to express a planning problem taking into account the spatial dimension.

Our main contribution is to integrate some concepts of SpaceOntology in TLP-GP-2 in order to consider both spatial and temporal dimensions in planning problem.

Preliminaries

This section presents all required notations for the definition of spatio-temporal rules.

A fluent is a positive or negative atomic proposition. We define a set of spatial predicates leading to spatial fluents when instantiated. We consider conditions on the value of fluents and changes of this value may either instantaneous or be imposed over an interval. An action \( a \) is a quadruple \(< \text{Cond}(a), \text{Eff}(a), \text{Constr}(a), \text{Mov}(a) >\), where:

- \( \text{Cond}(a) \) is a set of fluents which are required to be true for \( a \) to be executed,
- \( \text{Eff}(a) \) is the set of fluents which are established or destroyed by the execution of \( a \),
- \( \text{Constr}(a) \) is a set of constraints between the relative times of events which occur during the execution of \( a \),
- \( \text{Mov}(a) \) is a set of vectors of real valued functions associated with each of spatial fluent \( \text{Move}(e) \) in \( \text{Eff}(a) \) corresponding with the movement of spatial entity \( e \) produced by \( a \).

An event corresponds to one of nine possibilities: the instant when, or the beginning or end of an interval over which a fluent is required or produced or destroyed by an action \( a \). We use respectively the notation \( a \rightarrow f \) to denote the event that action \( a \) establishes fluent \( f \), \( a \rightarrow \neg f \) to denote the event
that action \( a \) destroys fluent \( f \), and \( f \rightarrow a \) to denote the event that \( a \) requires the fluent \( f \). When these events occur over an interval, we use respectively \( a \rightarrow f \), \( a \rightarrow \neg f \) and \( f \rightarrow a \) to denote the beginning of this interval, and respectively \( a \rightarrow f \), \( a \rightarrow \neg f \), \( f \rightarrow a \) to denote its end. We use the notation \( \tau(E) \) to represent the time in a plan at which an event \( E \) occurs. For a given action \( a \), let \( \text{Events}(a) \) represents a different events which constitute its definition. If \( A \) is a set of actions, then \( \text{Events}(A) \) is the union of the \( \text{Events}(a) \) for all \( a \in A \).

For any spatial entity \( e \) we define \( T\delta(e) \) as a set of temporal variables corresponding to a spatial event on \( e \) in the actions definition. These temporal variables correspond to a reading of the spatial state of \( e \) for the conditions (numeric position, numeric distance, fuzzy distance) or an update of this state for effects (movement, hierarchical link). The set of all temporal variables corresponding to the spatial events on all of the entities of \( SpE \) is denoted \( T\delta \).

**Case Study**

To illustrate our claim let us consider the following example. A waiter is involved in the environment shown in Figure 1(a). His mission is to serve each customer with the ordered drinks: a white coffee, a black coffee and a hot chocolate (Figure 1(c)).

![Figure 1: Environment and mission.](image)

To achieve this mission the waiter considers the spatial description of the environment in which he operates in order to find a plan by which he prepares and serves drinks (Figure 1(b)). Also, the success of this mission requires the satisfaction of the following temporal constraint: when drink is served it is still warm enough. Let us consider that the waiter’s goal is to serve the white coffee. In the following, we represent the description of its mission in PDDL extended with spatial fluents (Figure 2, Figure 3). Table 1, Table 2, Table 4 and Table 5 define some elements which are useful for understanding this paper and defined in our study case.

Table 1 defines types of used objects in our problem. Table 2 presents used objects. Table 4 presents the different symbolic or spatial fluents necessary. Table 5 presents the various actions that the waiter can execute.

![Figure 2: Domain definition in PDDL.](image)

**Definition 1 (Spatio-temporal planning problem)** A spatio-temporal planning problem \( < I, A, G > \) consists of a set of actions \( A \), an initial state \( I \) and a goal state \( G \), where \( I \) and \( G \) are sets of fluents.

We introduce three basic constraints that all spatio-temporal plans must satisfy:

- **Inherent constraints** on the set of actions \( A \): for all \( a \in A \), \( a \) satisfies \( \text{Constr}(a) \).
- **Contradictory-effects constraints** on the set of actions \( A \): for all \( a_i, a_j \in A \), for all fluents \( f \) such that \( -f \in E\text{ff}(a_i) \) and \( f \in E\text{ff}(a_j) \), \( \tau(a_i \rightarrow \neg f) \leq \tau(a_j \rightarrow f) < \tau(a_j \rightarrow f) \vee \tau(a_j \rightarrow f) < \tau(a_i \rightarrow -f) \).
- **Contradictory-movements constraints** on the set of actions \( A \): for all \( a_i, a_j \in A \), for all \( e \in SpE \), for all spatial fluents \( \text{Move}(e) \) such that \( \text{Move}(e) \in E\text{ff}(a_i) \) and \( \text{Move}(e) \in E\text{ff}(a_j) \), \( \tau(a_i \rightarrow \text{Move}(e)) < \tau(a_j \rightarrow \text{Move}(e)) \vee \tau(a_j \rightarrow \text{Move}(e)) < \tau(a_i \rightarrow \text{Move}(e)) \).
Definition 2 (Spatio-temporal plan) \( P = < A, t > \) where \( A \) is a set of action-instances \( \{a_1, \ldots, a_n\} \) and \( t \) is a real-valued function on Events(\( A \)), is a spatio-temporal plan for the problem \( < I, A', G > \) if (1) \( A \subset A' \) and (2) \( P \) satisfies the inherent and contradictory-effect constraints on \( A \); when \( p \) is executed (i.e. fluents are established or destroyed at the times given by \( t \)) starting from the initial state \( I \) and (3) for all \( a_i \in A \), each \( f \in \text{Cond}(a_i) \) is true when it is required and (4) all goal fluents \( g \in G \) are true at the end of execution of \( P \).

To encode a planning problem we use a simplified planning graph \( GP \) (Blum and Furst 1995). \( \text{Node}A \) denotes the set of actions nodes of \( GP \). \( \text{ArcsCond} \) and \( \text{ArcsEff} \) respectively denote the set of conditions arcs and the effects arcs of \( GP \). \( \text{SpE} \) denotes the set of spatial entities defined in the planning problem. In the following section we present the set of rules defining the ST-SMTPLAN.

**ST-SMTPLAN: Encoding Rules for Spatio-Temporal Planning**

To encode a spatio-temporal planning problem, we define three types of rules: (1) rules (R) encode the plan solution structure of the problem based on a temporarily extended planning graph, (2) rules (T) manage temporal knowledge and (3) rules (S) manage spatial and temporal knowledge.

Formally, \( \mathcal{R} \) denotes the set of ST-SMTPLAN rules \( \mathcal{R} = \mathcal{R} \cup \mathcal{T} \cup \mathcal{S} \). In this paper, we present a subset of \( \mathcal{R} \) which are used to solve the problem defined in the study case. The rules presented here are classified by group. Those concerning the definition: initial and goal states, the graph temporally extended, the spatial and temporal dimensions simultaneously and temporally extended mutexes.

**Initial State and Goal**

We define two dummy actions \( \text{Init} \) and \( \text{Goal} \). \( \text{Init} \) produces the initial state and \( \text{Goal} \) produces the goal state. This two dummy actions are both true.

\[ \text{Init} \land \text{Goal} \quad (R1) \]

This part presents some of S1 rules for encoding the description of the initial state and goal. From a spatial perspective, the initial state considers the description of positions (rule S1.1) and hierarchical definition (rule S1.2). Intuitively, the instant at which the goal state is true is later than the instant at which the initial state is true. The rule (T1) encodes this intuition.
(S1.1): Numeric position of entities in the initial state
In SpaceOntology: a spatial entity e is a localized entity in a given space. It can be static; any element fixed in space which position changes only by an action performed by an agent. Or it can be dynamic; any entity which spatial position changes over time in space. A spatial entity is defined by its center.

\[ \bigwedge_{e \in \text{SpEnt}} \left( \begin{array}{l} x[\tau(\text{Init})](e) = e.\text{center}.x \\ y[\tau(\text{Init})](e) = e.\text{center}.y \end{array} \right) \]  

(S1.1)

Example 1 Let us consider the space defined in Figure 1(c). We focus on the table which is in the lounge.

![Figure 4: Spatial entities positions.](image)

To locate this table, we consider its center and project it on both axis (Figure 4). This allows us to instantiate the rule (S1.1) as follows:

\[ \begin{align*} x[\tau(\text{Init})](\text{table}) &= 4 \\ y[\tau(\text{Init})](\text{table}) &= 3.2 \end{align*} \]

Remark 1 We assume that we are in 2D space. x denotes x-axis and y denotes y-axis. All rules are defined for x-axis and y-axis. In this paper, we detail only rules for x-axis. Rules can be extend to 3D.

(S1.2): Inclusion relation of entities in the initial state
SpaceOntology permits the hierarchical representation in order to simplify the description of the global environment. To represent the hierarchical relation, we define the spatial fluent Inside. This spatial fluent expresses the inclusion link between two spatial entities. Formally, for each pair of spatial entities \((e_1, e_2)\) where the relation \(e_1 \subset e_2\) is true, then we add the following rule:

\[ \bigwedge_{(e_1, e_2) \in \text{SpEnt}^2, e_1 \subset e_2} \text{Inside}[\tau(\text{Init})](e_1, e_2) \]  

(S1.2)

Example 2 The initial state of our problem is depicted in Figure 1(a). \(\tau(\text{Init})\) is the instant of the initial time. \(\text{Inside}[\tau(\text{Init})](\text{waiter}, \text{kitchen})\) expresses that the waiter is in the kitchen in the initial state. Thus, to express spatial inclusion in the initial state we instantiate the rule S1.2 as follow:

\[ \begin{align*} &\text{Inside}[\tau(\text{Init})](\text{shelf}, \text{kitchen}) \\ &\text{Inside}[\tau(\text{Init})](\text{cooker}, \text{kitchen}) \\ &\text{Inside}[\tau(\text{Init})](\text{waiter}, \text{kitchen}) \end{align*} \]

(T1): Lower and upper bounds
The initial instant (when propositions of the initial state are true) precedes every instant of the beginning of the preconditions of the other actions. The final instant (when the goal propositions are true) follows all instants of the end of the effects of the other actions.

\[ (\tau(\text{Init}) \leq \tau(\text{Goal})) \land \bigwedge_{a \in \text{NodeA}} \left( \begin{array}{l} (\tau(\text{Init}) \leq \text{Min}_{\text{Cond}(a)} \{\tau(p \rightarrow a)\}) \\ \land (\text{Max}_{q \in \text{Eff}(a)} \{\tau(a \rightarrow q)\} \leq \tau(\text{Goal})) \end{array} \right) \]  

(T1)

Encoding of Temporally Extended Planning Graph
The rules (R) encode the problem as temporally extended planning graph. They encode the causal links between conditions (R2) and effects and actions selections (R3).

(R2): Conditions production by causal links
If an action \(b\) is active in the plan, then for each of its preconditions \(p\), it exists at least one causal link (noted \(\text{Link}(a, p, b)\)) from the action \(a\) (which produces this precondition) to the action \(b\).

\[ \bigwedge_{(p, b) \in \text{ArcsCond}} b \Rightarrow \bigvee_{(a, p) \in \text{ArcsEff}} \text{Link}(a, p, b) \]  

(R2)

(R3): Actions activation and partial order
If a causal link exists between an action \(a\) which produces a precondition \(p\) for an action \(b\), then \(a\) and \(b\) are actives in the plan. Moreover, the instant when \(a\) certainly produces \(p\) precedes or is the same than the instant when \(b\) begins to need \(p\).

\[ \bigwedge_{(a, p) \in \text{ArcsEff}} \bigwedge_{(p, b) \in \text{ArcsCond}} \left( \text{Link}(a, p, b) \Rightarrow \left( \tau(a \rightarrow p) \leq \tau(p \rightarrow b) \right) \right) \]  

(R3)

Encoding Spatial Knowledge
An action may require the satisfaction of spatial relations between two spatial entities for execution. An action can change the spatial definitions between two spatial entities. Spatial information is integrated into the sets \(\text{Cond}(a)\) and \(\text{Eff}(a)\) when the action \(a\) is defined in the planning domain.

A spatial entity may be dynamic or static. The spatial entity \(e\) is dynamic, the use of the predicate \(\text{Move}(e)\) in the effects of the action \(a\) corresponds to the movement of the spatial entity \(e\) by the action \(a\) on a given temporal interval \([t_1, t_2]\). With the fluent \(\text{Move}(e)\) we associate a moving function. We define on the interval \([t_1, t_2]\) a linear function corresponding to the movement of spatial entity \(e\) for each axis. For horizontal axis (\(ox\)) the linear function is denoted by \(\Delta_{\tau}^a[e](\theta) = v_{\tau}^a(e) \times t\). The global movement of the entity \(e\) produced by an action \(a\) on the \(x\)-axis is denoted by \(\Delta_{\tau}^a(e)\) (\(\Delta_{\tau}^a(e) = \Delta_{\tau}^a(\tau(a \rightarrow \text{Move}(e)) - \tau(a \rightarrow \text{Move}(e))(e))\)).
**Spatial Conditions** The rules S2 encode the spatial conditions. Indeed, it is necessary to compute distances (S2.1) and to know numeric distances or approximative distances (S2.2) in order to execute actions or not.

(S2.1): Computing numeric distances between entities

For each couple of spatial entities \( (e_1, e_2) \) we denote by \( d_2[t] \{e_1, e_2\} \) a numeric distance between \( e_1 \) and \( e_2 \) according to x-axis.

\[
\bigwedge_{(e_1, e_2) \in \text{SpE}^2/|e_1 \neq e_2|} \bigwedge_{t \in T \delta_{e_1}} \bigwedge_{t \in T \delta_{e_2}}
\begin{cases}
(x[t](e_1) \leq x[t](e_2) \Rightarrow d_2[t] \{e_1, e_2\} = x[t](e_2) - x[t](e_1) \\
(x[t](e_1) < x[t](e_2) \Rightarrow d_2[t] \{e_1, e_2\} = x[t](e_1) - x[t](e_2)
\end{cases}
\]

(S2.2)

(S2.2): Numeric distance between entities

The spatial fluent \( \text{Distance}(e_1, e_2) = \text{value} \) where value is a rational number can be used as a constraint in the definition of \( \text{Cond}(a) \). It represents a condition of fixed numeric distance between two spatial entities \( e_1 \) and \( e_2 \). A constraint \( d_2[t] \{e_1, e_2\} = \text{value} \) is then added for each action \( a \) of the graph at \( t = \tau(\text{Distance}(e_1, e_2) = \text{value} \rightarrow a) \). We can generalize by replacing equality by inequality.

\[
\bigwedge_{(e_1, e_2) \in \text{SpE}^2/|e_1 \neq e_2|} \bigwedge_{a \in \text{NodeA}/(\text{Distance}(e_1, e_2) = \text{value})} \bigwedge_{t \in \text{Cond}(a)}
\begin{cases}
d_2[t] \{e_1, e_2\} = \text{value} \rightarrow a \{e_1, e_2\} = \text{value}
\end{cases}
\]

(S2.3)

(S2.3): Fuzzy distance between entities

SpaceOntology defines four concepts \{Near, NearEnough, FarEnough, Far\} in order to express fuzzy distance. The special fluent \( \text{FD}_{label}(e_1, e_2) \) represents a fuzzy distance between two entities \( e_1 \) and \( e_2 \) such as \( label = \{\text{Near, NearEnough, Far, FarEnough}\} \). Each label is associate to an interval \([\alpha_{label}, \beta_{label}]\).

For example, let us consider the fuzzy distance near enough, so \( label = \text{NearEnough} \) and the associated interval is \([\alpha_{\text{NearEnough}}, \beta_{\text{NearEnough}}]\) (Figure 5).

\[
\bigwedge_{(e_1, e_2) \in \text{SpE}^2/|e_1 \neq e_2|} \bigwedge_{a \in \text{NodeA}/\text{FD}_{label}(e_1, e_2) \in \text{Cond}(a)}
\begin{cases}
\alpha_{\text{label}} \leq d_2[t] \{\text{FD}_{label}(e_1, e_2) \rightarrow a\} \{e_1, e_2\} < \beta_{\text{label}}
\end{cases}
\]

(S2.3)

**Example 3** Let us consider our study case. We define the following variables:

\[
\begin{align*}
a &= \text{Pick(milk,shelf)} \\
t &= \tau(\text{FD}_{\text{Near}}(\text{waiter,shelf}) \rightarrow \text{Pick(milk,shelf)}) \\
e_1 &= \text{waiter} \\
e_2 &= \text{shelf}
\end{align*}
\]

The application of the rule (S2.1) computes the distance between the waiter and the shelf:

\[
\begin{align*}
\{x[t](\text{waiter}) \leq x[t](\text{shelf}) \Rightarrow \\
d_2[t] \{\text{waiter,shelf}\} = x[t](\text{shelf}) - x[t](\text{waiter}) \\
\{x[t](\text{shelf}) < x[t](\text{waiter}) \Rightarrow \\
d_2[t] \{\text{waiter,shelf}\} = x[t](\text{waiter}) - x[t](\text{shelf})
\end{align*}
\]

Let us consider that the execution of the action \( \text{Pick} \) requires that the distance between the waiter and the shelf is 0.3. The application of the rule (S2.2) checks the distance between the waiter and the shelf:

\[
\begin{align*}
\{\text{Pick(milk,shelf)} \Rightarrow \\
(d_2[t] \{\text{FD}_{\text{Near}}(\text{waiter,shelf}) = 0.3 \rightarrow \text{Pick(milk,shelf)}\}\{\text{waiter,shelf}\} = 0.3
\end{align*}
\]

The application of the rule (S2.3) checks the fuzzy distance between the waiter and the shelf:

\[
\begin{align*}
\{\text{Pick(milk,shelf)} \Rightarrow \\
\alpha_{\text{Near}} \leq \\
d_2[t] \{\text{FD}_{\text{Near}}(\text{waiter,shelf}) \rightarrow \text{Pick(milk,shelf)}\}\{\text{waiter,shelf}\} < \beta_{\text{Near}}
\end{align*}
\]

**Spatial effects** Computing distances between entities requires to know their numerical position. It is therefore necessary to know these positions in the initial state, but also to recalculate when the effect of an action changes.

(S3): Computing new positions Let us consider the temporal interval \([t_1, t_2]\). The spatial entity \( e_1 \) position at time \( t_2 \) is calculated from its position at time \( t_1 \) and from the movements \( \delta^{\text{Ind}}_{e_1}[t_1, t_2]\{e_1, a, e_2\} \) induced by all entities \( e_2 \) that can be moved by an action \( a \).
Example 4 Let us consider the following action

\[ Go(\text{shelf, table}) \]. Note by:

- \( t_1 = \tau(\text{Go(\text{shelf, table})} \rightarrow \text{Move}_1(\text{waiter})) \)
- \( t_2 = \tau(\text{Go(\text{shelf, table})} \rightarrow \text{Move}_2(\text{waiter})) \)

Let us apply rule S3.

\[
x[2](\text{cup}) = \begin{cases} 
\delta_{\text{cup}}^\text{Ind}[1,2](\text{cup, Go(\text{shelf, cooker}), waiter}) + \\
\delta_{\text{cup}}^\text{Ind}[1,2](\text{cup, Go(cooker, shelf), waiter}) + \\
\delta_{\text{cup}}^\text{Ind}[1,2](\text{cup, Go(\text{shelf, table}), waiter}) + \\
... 
\end{cases}
\]

To compute the induced movement of an entity \( e \) on other entities or on entity \( e \), it is necessary to know the hierarchical relation between \( e \) and all other entities at the instant when \( e \) starts or ends its movement. We can now describe the rules considering hierarchical links.

(S4.1): Production and removal of hierarchical link If an action \( a \) which produces the inclusion of a spatial entity \( e_1 \) in a spatial entity \( e_2 \) at time \( t = \tau(a \rightarrow \text{Inside}(e_1, e_2)) \) is active in the plan then the predicate \( \text{Inside}[t](e_1, e_2) \) corresponding to the inclusion \( e_1 \subset e_2 \) at time \( t \) is true.

\[
(\forall (e_1, e_2) \in \text{SpE}^2 \exists \tau \exists a \exists \text{NodeA/Inside}(e_1, e_2) \in \text{Eff}(a) \Rightarrow \text{Inside}[t](e_1, e_2)) 
\]

(S4.1.1)

If an action \( a \) which removes the inclusion \( e_1 \subset e_2 \) at time \( t = \tau(a \rightarrow \neg\text{Inside}(e_1, e_2)) \) is active in the plan then the predicate \( \neg\text{Inside}[t](e_1, e_2) \) is false.

\[
(\forall (e_1, e_2) \in \text{SpE}^2 \exists \tau \exists a \exists \text{NodeA/\neg\text{Inside}(e_1, e_2)} \in \text{Eff}(a) \Rightarrow \neg\text{Inside}[t](e_1, e_2)) 
\]

(S4.1.2)

Example 5 The waiter’s movement between the shelf and the table \( (\text{Go}(\text{shelf, table})) \) has two spatial effects on hierarchy. The first is the production of the hierarchical link “the waiter is inside the lounge” (applying the rule (S4.1.1)):

\[
\begin{cases} 
\text{Go(\text{shelf, table})} \Rightarrow \\
\text{Inside}[\tau(\text{Go(\text{shelf, table})} \rightarrow \text{Inside(\text{waiter, lounge})})](\text{waiter, lounge}) 
\end{cases}
\]

The second is the destruction of the hierarchical link “the waiter is inside the kitchen” (applying the rule (S4.1.2)):

\[
\begin{cases} 
\text{Go(\text{shelf, table})} \Rightarrow \\
\neg\text{Inside}[\tau(\text{Go(\text{shelf, table})} \rightarrow \neg\text{Inside(\text{waiter, kitchen})})](\text{waiter, kitchen}) 
\end{cases}
\]

(S4.2): Propagation of hierarchical links in time If a hierarchical link \( e_1 \subset e_2 \) is active in the plan at time \( t \in \tau \), when a spatial event occurs, then there exists at least one positive hierarchical link protection interval (denoted \( \text{LinkInside}(e_1, e_2, a, t) \)) of an action \( a \), which produces \( \text{Inside}(e_1, e_2) \), up to time \( t \).

\[
(\forall (e_1, e_2) \in \text{SpE}^2 \exists \tau \exists a \exists \text{NodeA/Inside}(e_1, e_2) \in \text{Eff}(a) \Rightarrow \text{Inside}[t](e_1, e_2)) 
\]

(S4.2.1)

Similarly, if a hierarchical link \( e_1 \subset e_2 \) is not active in the plan at time \( t \in \tau \), when a spatial event occurs, then there exists at least one negative hierarchical link protection interval (denoted \( \text{LinkNotInside}(e_1, e_2, a, t) \)) of an action \( a \), which produces \( \neg\text{Inside}(e_1, e_2) \), up to time \( t \).

Example 6 We assume that \( t \) is the instant when the action serve \( (\text{Serve(\text{table, cup})}) \) requires that the waiter has the cup \( t = \tau(\text{Inside(\text{cup, waiter})} \rightarrow \text{Serve(\text{table, cup})}) \). If at the instant \( t \), the waiter has the cup then it exists a link to propagate this property from the action \( \text{Make}_W(C)(\text{milk, coffee, cup}) \) which produces it.

\[
\begin{cases} 
\text{Inside}[t](\text{cup, waiter}) \\
\Rightarrow \text{LinkInside}(\text{cup, waiter, Make}_W(C)(\text{milk, coffee, cup}), t) 
\end{cases}
\]

If a link propagates the fact that the waiter has the cup from the action \( \text{Make}_W(C)(\text{milk, coffee, cup}) \), up to the instant \( t \), then this action must be active in the plan and produces the hierarchical link before the instant \( t \).
Example 7 Let us consider the two actions: prepare the coffee \( \text{Make}_W\_C(\text{milk, coffee, cup}) \) and move between the cooker and the shelf \( \text{Go}(\text{cooker, shelf}) \). In order to execute these two actions, we must respect one of this situation (applying the rule (S5.1.1)):

- the instant when the action \( \text{Make}_W\_C(\text{milk, coffee, cup}) \) finish to produce that the waiter has the cup is before the instant when the action \( \text{Go}(\text{cooker, shelf}) \) begin to move the waiter.
- the instant when the action \( \text{Go}(\text{cooker, shelf}) \) finish to move the waiter between the cooker and the shelf is before the instant when the action \( \text{Make}_W\_C(\text{milk, coffee, cup}) \) begins to produce that the waiter has the cup.

\[
\begin{align*}
\text{Make}_W\_C(\text{milk, coffee, cup}) & \land \text{Go}(\text{cooker, shelf}) \\
\Rightarrow & \begin{cases} 
\tau(\text{Make}_W\_C(\text{milk, coffee, cup}) \rightarrow \text{Inside}(\text{cup, waiter})) \\
< \tau(\text{Go}(\text{cooker, shelf}) \rightarrow \text{Move}(\text{waiter})) 
\end{cases} \\
\lor & \tau(\text{Go}(\text{cooker, shelf}) \rightarrow \text{Move}(\text{waiter})) \\
\lor & \tau(\text{Make}_W\_C(\text{milk, coffee, cup}) \rightarrow \text{Inside}(\text{cup, waiter}))
\end{align*}
\]

We can now calculate the induced movements by one entity over another.

(S6.1): Induced movement with hierarchical link We assume an action that \( a \) which produces a movement of a spatial entity \( e_2 \) (\( \Delta^a_t(e_2) \neq 0 \))is active in the plan. We consider a spatial entity \( e_1 \) such as \( e_1 \subset e_2 \). Also, we consider the temporal interval \( I = [t_1, t_2] \). Let \( [t, t'] \) the intersection of \( I \) and the interval on which \( e_2 \) movement is produced by \( a \). If this intersection is not empty then the movement of \( e_1 \) induced by \( e_2 \) is \( \delta^\text{ind}_{a}[t_1, t_2](e_1, a, e_2) \) is equal to \( \Delta^a_t[t' - t](e_2) \), else the induced movement is zero.

\[
\begin{align*}
\text{(S6.1.1)} & \quad \text{Move}(e_2) & \quad \text{Move}(e_2) & \quad \text{Move}(e_2) \\
\text{(S6.1.2)} & \quad \text{Move}(e_2) & \quad \text{Move}(e_2) & \quad \text{Move}(e_2) \\
\text{(S6.1.3)} & \quad \text{Move}(e_2) & \quad \text{Move}(e_2) & \quad \text{Move}(e_2) \\
\text{(S6.1.4)} & \quad \text{Move}(e_2) & \quad \text{Move}(e_2) & \quad \text{Move}(e_2) \\
\text{(S6.1.5)} & \quad \text{Move}(e_2) & \quad \text{Move}(e_2) & \quad \text{Move}(e_2) \\
\text{(S6.1.6)} & \quad \text{Move}(e_2) & \quad \text{Move}(e_2) & \quad \text{Move}(e_2)
\end{align*}
\]

Figure 6: Possible configurations between an interval and spatial entity’s movement.

The different configurations and the associated rules are illustrated by Figure 6. Here we detail only the rule (S6.1.1).

Example 8 Let us denote by:

- \( t_1 = \tau(\text{Go}(\text{shelf, table}) \rightarrow \text{Move}(\text{waiter})) \)
- \( t_2 = \tau(\text{Go}(\text{shelf, table}) \rightarrow \text{Move}(\text{waiter})) \)

We are in the case of rule (S6.1.4). On the global movement interval \([t_1, t_2]\) over which the waiter moves from the shelf to the table. If the waiter has the cup, each of the movement (\( \text{Move}_1(\text{waiter}) \) and \( \text{Move}_2(\text{waiter}) \)) induces the movement on the cup.

\[
\begin{align*}
\text{Go}(\text{shelf, table}) & \land \\
\text{Inside}[\tau(\text{Go}(\text{shelf, table}) \rightarrow \text{Move}(\text{waiter}))](\text{cup, waiter}) & \land \\
\tau(\text{Go}(\text{shelf, table}) \rightarrow \text{Move}(\text{waiter})) & \land \\
\tau(\text{Go}(\text{shelf, table}) \rightarrow \text{Move}(\text{waiter})) & \land \\
\Rightarrow (\delta^\text{ind}_{a}[t_1, t_2](\text{cup, Go}(\text{shelf, table}), \text{waiter}) = \\
\Delta^a_t[t_1, t_2](e_2 - \tau(\text{Go}(\text{shelf, table}) \rightarrow \text{Move}(\text{waiter}))), (\text{cup, waiter}))}
\end{align*}
\]

(S6.2): Induced movement without hierarchical links If an action \( a \) which moves an entity \( e_2 \) with displacement \( \Delta^a_t(e_2) \) is active in the plan and we have a spatial entity \( e_1 \) distinct from \( e_2 \) \((e_1 \neq e_2)\) and \( e_1 \) is not inside \( e_2 \) \((e_1 \not\subset e_2)\) when \( a \) starts to move \( e_2 \) then the displacement of \( e_1 \) on the interval \([t_1, t_2]\) induced by the movement of \( e_2 \) produced by the action \( a \) is \( \delta^\text{ind}_{a}[t_1, t_2](e_1, a, e_2) = 0 \).

\[
\begin{align*}
\Delta^a_t[e_1, e_2] & \land \\
\land \neg\text{Inside}[\tau(\text{Go}(\text{shelf, table}) \rightarrow \text{Move}(\text{waiter}))](\text{cup, waiter}) & \land \\
\Rightarrow (\delta^\text{ind}_{a}[t_1, t_2](e_1, a, e_2) = 0)
\end{align*}
\]

Example 9 In our study case; the waiter moves from shelf to table without taking the cup \((\text{cup} \not\subset \text{waiter})\). The application of (S6.2) shows that the displacement of the cup induced by waiter’s movement is zero.

\[
\begin{align*}
\text{Go}(\text{shelf, table}) & \land \\
\neg\text{Inside}[\tau(\text{Go}(\text{shelf, table}) \rightarrow \text{Move}(\text{waiter}))](\text{cup, waiter}) & \land \\
\Rightarrow (\delta^\text{ind}_{a}[t_1, t_2](\text{cup, Go}(\text{shelf, table}), \text{waiter}) = 0)
\end{align*}
\]

(S6.3): Induced movement by an inactive action If an action \( a \) which produces a movement of a spatial entity \( e_2 \), noted \( \Delta^a_t(e_2) \), is not active in the plan and given a spatial entity \( e_1 \) (not necessarily distinct from \( e_2 \)) then the movement of \( e_2 \), noted \( \delta^\text{ind}_{a}[t_1, t_2](e_1, a, e_2) \), on the temporal interval \([t_1, t_2]\) induced by the movement of \( e_2 \) produced by \( a \) is zero.
Temporally extended mutexes

Two propositions are mutually exclusive when:

- the propositions are antagonists, for instance $p$ and $\neg p$;
- or the propositions represent the movement of the same spatial entity $e$ (the special predicate $Move(e)$ is used by two actions).

In these cases, the propositions can not occur on the same temporal interval.

(R4): Temporally extended mutexes If a causal link protects a proposition $p$ and an action produces its is active in the plan, then the temporal interval corresponding to the causal link and the temporal interval corresponding to activation of $\neg p$ by the action are disjunctive.

\[
\begin{align*}
&\land (a, p) \in ArcsEff \land (b, \neg p) \in ArcsEff \land \land \\
&\land (Link(a, p, b) \land c) \land \land \\
&\Rightarrow (\tau(c \rightarrow \neg p) < \tau(a \rightarrow p)) \lor (\tau(p \rightarrow \neg b) < \tau(c \rightarrow \neg p))
\end{align*}
\]  
(R4.1)

If two actions respectively producing a proposition $p$ and its negation are active in the plan, then the temporal intervals corresponding to the activation of $p$ and activation of $\neg p$ are disjunctive.

\[
\begin{align*}
&\land (a, p) \in ArcsEff \land (b, \neg p) \in ArcsEff \land \\
&a \land b \land \land \\
&\Rightarrow (\tau(b \rightarrow \neg p) < \tau(a \rightarrow p)) \lor (\tau(a \rightarrow \neg b) < \tau(b \rightarrow \neg p))
\end{align*}
\]  
(R4.2)

Mutexes of actions that move the same entity If a causal link protects the proposition $Move(e)$ which represents a spatial entity $e$ moving and also an action produces the proposition $Move(e)$ in the plan then the temporal interval corresponding to the causal link and the temporal interval corresponding to the activation of $Move(e)$ by the action are disjunctive. If two actions produce $Move(e)$ are active in the plan, then the temporal intervals associated with activation of $Move(e)$ by these actions are disjunctive.

To obtain respectively the rules (S7) corresponding to these kind of mutual exclusion, it suffices to replace $p$ and $\neg p$ by $Move(e)$ in rules (R4.1) and (R4.2).

(S8): Mutexes on hierarchical links protection interval Similarly as causal link protection (R4.1), we add two rules, respectively (S8.1) and (S8.2), providing a hierarchical link protection for $LinkInside(e_1, e_2, a, t)$ and $LinkNotInside(e_1, e_2, a, t)$.
initial state and manipulate fuzzy distances between spatial entities. Moreover, the principle of TLP-GP allows us to obtain a temporal planning system capable of solving problems that require concurrency of actions. This system uses a SMT solver and benefit directly from improvements in this type of solver in terms of performance.

This paper is a theoretical work. In order to prove the efficiency of this encoding, we need to define some spatio-temporal planning benchmarks. To simplify the representation space we considered any spatial entity as a point (this point defines the center). This reduces the expressiveness. In our future work we will focus on the size and shape of each spatial entity. An avenue for future research is to use the principle of TLP-GP-I which performs the backward search on the planning graph. This will allows us to query SpaceOntology and code the extracted spatial knowledge at each action selection. This allows a great spatial expressivity added to the temporal expressivity.

References


Cooper, M.; Maris, F.; and Regnier, P. 2010a. Compilation of a high-level temporal planning language into pddl 2.1. In Tools with Artificial Intelligence (ICTAI), volume 2, 181–188. IEEE.

Cooper, M.; Maris, F.; and Regnier, P. 2010b. Solving temporally-cyclic planning problems. In Temporal Representation and Reasoning (TIME), 113–120. IEEE.


