Temporal goal reasoning for predictive performance of a tourist application

Eliseo Marzal, Jesus Ibañez, Laura Sebastia, Eva Onaindia [1]

Abstract. Many real-work smart environments make use of IoT to be provided with context-aware services. Additionally, these environments require making decisions based on predictions about future actions. This involves the use of goal-directed behaviour which may need reasoning about new goals. This paper is devoted to analyze when a new goal can be formulated. Once a plan has been computed for a given problem, exogenous events can change the environment so that a failure in the plan is caused or an opportunity arises. This paper present a goal reasoning framework where context information acquired from several external sources determines a change that may affect the execution of the current plan. This change may cause a failure or an opportunity. We show how the planning system, namely TempLM, is able to predict both failures and opportunities thanks to the analysis of the Temporal Landmarks Graph and the Temporal Propositions Graph built for the given problem.

1 INTRODUCTION

One key feature in the application of innovative technologies like IoT in smart environments is the capability of providing context-aware services. Besides real-world information, this requires anticipatory behaviour through reasoning; that is, making decisions based on predictions and expectations about future actions [1]. Particularly, many real-world applications involve unanticipated changes that may bring an alteration of the current process or a future impact on the application or even an opportunity to include some new functionality.

The purpose of a planning application is to achieve a goal through the execution of a plan or course of actions. The arrival of an unexpected environmental event introduces a new piece of information that was not taken into account during the construction of the plan and that may affect the active plan in several ways. Typically, the first reaction is to check if the plan is no longer executable and, if so, apply a repair mechanism or replanning to fix the failure that prevents the active plan from being normally executed. A second consequence is that the unanticipated event provokes a future anomaly in the plan execution. A third and more interesting derivation is whether the new data brings an opportunity to achieve a goal that is not currently contemplated in the active plan.

Goal-directed behaviour is a hallmark of intelligence aimed at discovering the changes that can be applied in the goal of an application in view of the information collected by some unanticipated events. A dynamic formulation of new goals is very helpful in situations where: a) the agent's interests are threaten and a rational anomaly response must be provided; b) goals are no longer achievable and a graceful degradation in the goal achievement is a convenient action; or c) goal achievement in the future is jeopardized, what affects future performance [14]. Thereby, goal-directed reasoning can be regarded as a context-aware responsiveness technique.

This paper is particularly devoted to analyze the first step of a goal formulation process; that is, to answer the question '*when a new goal can be formulated*?'. In some applications, opportunistic behaviour is applied when the sensory input triggers some enabling conditions to accomplish a task, and reactive plans are adopted to detect problems and recover from them automatically as well as to recognize and exploit opportunities [2]. In dynamic and complex environments, like robotics, opportunities are predicted and executed immediately in order to provide quick responsiveness but there is no usually anticipation of future situations.

In less dynamic and reactive environments, typically, goal formulation is considered when an anomaly is detected and/or the system is self-motivated to explore its actions in the world [14]. One approach that has been used to predict or anticipate future plan failures is Case-Based Planning (CBP). In CBP, when a plan fails, it is usually stored with the justification of its failure. This information is then retrieved from the case memory when looking for a similar situation which produced a failure in the past. CBP may be applied before the generation of a plan to anticipate possible problems and avoid situations that failed in the past, or after a plan has been produced to eliminate plans which may fail [13]. CBP presents though several limitations in its application to smart environments: a) predicting future failures is subject to finding an identical case in the case memory; and b) CBP allows only for anticipating a failure but not for detecting opportunities of pursuing a better goal or a new goal.

In this paper, we present a goal reasoning framework that traces the execution of a temporal plan and identifies if the context information acquired from several sources determines a change in the plan goals. Particularly, the reasoner detects situations of future failures and opportunities in the plan execution in the context of temporal planning with deadlines. The framework draws upon TempLM, an approach based on temporal landmarks to handle temporal planning problems with deadlines [11, 12], and we will show how the reasoner works on a temporal plan of a tourist application.

[1] Universitat Politècnica de València, Valencia, Spain, Email: {emarzal,jeibrui,lstarin,onaindia}@dsic.upv.es

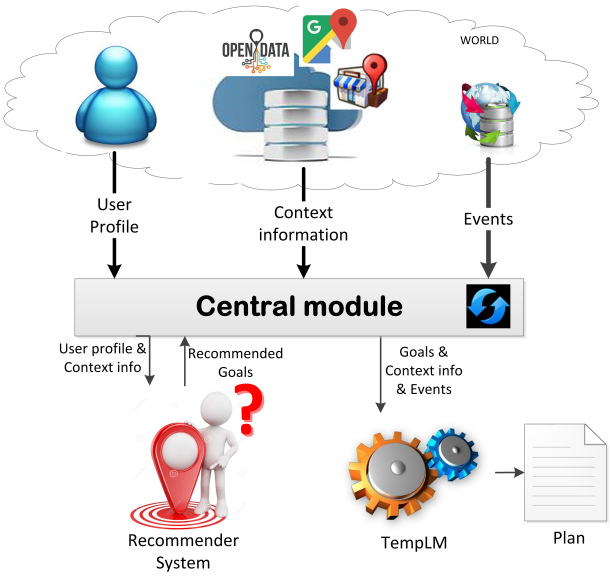

Figure 1. Architecture

This paper is organized as follows. First, a motivating example is introduced, as well as the architecture of our system. Then, some basic definitions referring to automated planning and the main characteristics of our planner TempLM are given. Section 4 introduces new definitions about exogenous events and section 5 explains the analysis that TempLM performs in order to detect future failures or opportunities caused by exogenous events. Finally, section 6 concludes and outlines some future work.

2 TOURIST APPLICATION EXAMPLE

In order to illustrate the foundations and contributions of this paper, a problem in the context of smart tourist will be used. **Smart tourism** involves several components that are supported by information and communication technologies (ICT) [9]. On one hand, it refers to *Smart Destinations*, which are cases of smart cities that apply smart city principles to support mobility, resource availability and allocation, sustainability and quality of life/visits. Second, the *Smart resident/visitor experience* focuses on technology-mediated tourism experiences and their enhancement through personalization, context-awareness and real-time monitoring [4]. Finally, *Smart business* refers to the complex business ecosystem that creates and supports the exchange of touristic resources and the co-creation of the tourism experience. These smart systems include a wide range of technologies in direct support of tourism such as decision support systems and the more recent recommender systems, context-aware systems, autonomous agents searching and mining Web sources, ambient intelligence, as well as systems that create augmented realities.

In this sense, our system aims to improve the resident/visitor experience by reacting in advance to changes in the environment that may cause failures or opportunities in the previously computed agenda. The architecture of our system, shown in Figure 1, is composed of the following modules:

- The **Central module** is the core of the system. It is in charge of generating the initial plan, considering the user profile and the context information. Additionally, it listens to new events that may require to update this plan.
- The **Recommender System** selects the recommended visits for a tourist, given her user profile and the context information.
- The TempLM **planner** develops two main tasks: in first place, it receives the goals computed by the recommender system and the context information and it builds the initial plan for the tourist; then, every time a new event is received by the central module, TempLM analyses it to determine whether the plan needs to be updated.

In this paper, we will focus on the second task of TempLM, that is, on the analysis of events to detect failures or opportunities in the plan.

An example of the problem that we are facing is the following. A tourist arrives to Valencia (Spain) and she is staying at Caro Hotel. She uses our system to obtain a personalized agenda for her visit. First, the recommender system analyses her user profile to select a set of recommended visits with a recommended duration. Let us assume that the user is recommended to visit the Lonja for 1.5 hours, the Cathedral and the Central Market for 2 hours, respectively. These recommended goals, some other user preferences related to the time windows when she prefers to perform the activities along with information about the context, such as the opening hours of the places to visit and the geographical distances between places, are compiled into a planning problem that is formulated as an *Automated Planning Problem* [8], with durative actions and deadlines. This problem is solved by our planner TempLM.

Figure 2 shows the plan obtained for this tourist. The segments at the bottom represent the opening hours of places (i.e. the Lonja is open from 10h until 19h) or the time windows of preferences indicated by the user (i.e. the time for having lunch ranges from 14h to 16h). The green boxes represent the actions in the plan. Specifically, in this domain, three types of actions can be performed: visiting an attraction, having lunch and moving from one place to another. The duration of these actions is determined by the corresponding parameters, that is, the attraction to visit or the origin and destination of the movement action, respectively. For example, the action `(visit T Lonja)` takes from 10:09h to 11:29h. In addition, the visit action must consider the opening hours/preference time window; for example, the action `(visit T Centralmarket)` can only be performed from 15:30h until 19h. The whole plan must fit into the available time of the user indicated in Figure 2 as the timeline, that is, from 10h until 19h. A more detailed description of the compilation of this problem and domain can be found in [10].

If we consider this context, there are some events that may occur during the visit of our tourist. For example:

- The Lonja may close earlier or open later, that is, its available time window may change; this may imply that the visit action has to finish before than expected or it has to be delayed, respectively.
- The Ricard Camarena restaurant may be fully-booked, which implies that another restaurant in the area must be selected.

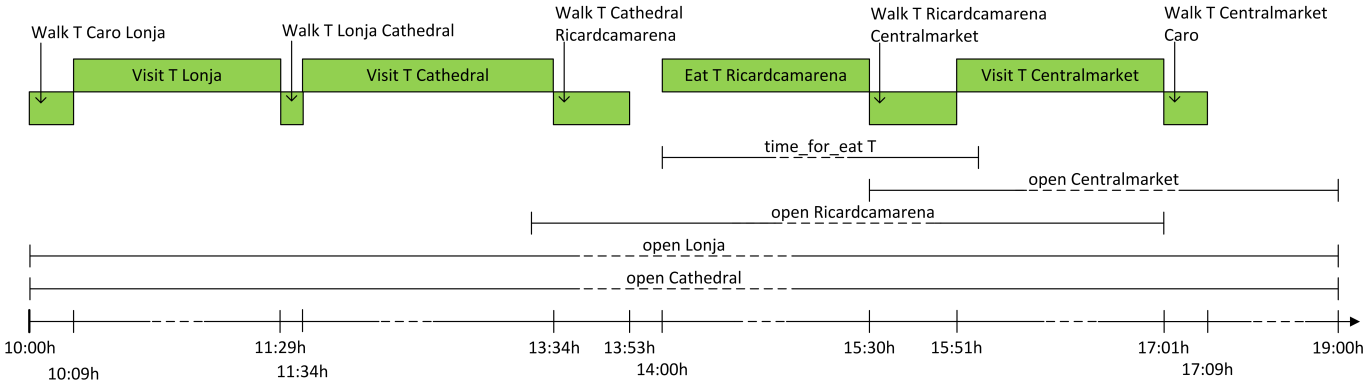

Figure 2. Plan in execution

- The user may take longer to walk from his hotel to the Lonja, so the visit action must be delayed.

These exogenous events are received by the central module from the external data sources and are analyzed by TempLM in order to react in consequence. There are some events that may cause a plan failure (i.e. the attraction is closed) or a plan modification (i.e. the user gets later that expected to an attraction), whereas others may cause the appearance of a free time slot that can be used to include an affordance/opportunity. In this current work, we will focus on the detection of both failures and time slots for opportunities, and we will give some hints about how they can be solved or exploited.

3 BACKGROUND

This section introduces some planning concepts and then it summarizes the main characteristics of our planner TempLM.

3.1 Planning concepts

Definition 3.1 *A* **temporal planning problem with deadline constraints** *is a tuple* $\mathcal{P} = \langle P, O, I, G, D \rangle$*, where* P *is the set of propositions,* O *is the set of ground actions,* I *and* G *are sets of propositions that represent the initial state and the goal description and* D *is a set of deadline constraints of the form* (p, t)*, denoting that proposition* p *must be achieved within* t *time units.*

For example, a proposition in I can be (be T Caro), indicating that initially the tourist is at the Caro hotel. It is important to note that, apart from the propositions and functions that are initially true, the initial state I may also contain *timed initial literals (TILs)*. TILs, which were first introduced in PDDL2.2[6], are a very simple way of expressing that a proposition becomes true or false at a certain time point. A TIL can be represented as a pair (p, t), where p is a (positive or negative) proposition and t is a time point. Specifically, if p is a positive proposition, then t indicates the time point at which p becomes true and if it is a negative proposition, then t indicates the time point at which p becomes false. For example, ((not (open Lonja)),19h) is a TIL in I that indicates that the Lonja will close at 19h.

Definition 3.2 *A* **simple durative action** $a \in O$ *is defined as a tuple* $(pre_{\vdash}, eff_{\vdash}, pre_{\leftrightarrow}, pre_{\dashv}, eff_{\dashv}, dur)$[5]*:*

- $pre_{\vdash}$ *(*$pre_{\dashv}$*) are the start (end) conditions of* a*: at the state in which* a *starts (ends), these conditions must hold.*
- $eff_{\vdash}$ *(*$eff_{\dashv}$*) are the start (end) effects of* a*: starting (ending)* a *updates the world state according to these effects. A given collection of effects* $eff_x, x \in \{\vdash, \dashv\}$ *consists of:*
  - eff_x^-*, propositions to be deleted from the world state;*
  - eff_x^+*, propositions to be added to the world state*
- $pre_{\leftrightarrow}$ *are the invariant conditions of* a*: these must hold at every point in the open interval between the start and end of* a*.*
- dur *is the duration of* a*.*

An example of an action is shown here:

Action: `Eat(?t: tourist, ?r: restaurant)`
$pre_{\vdash}$ = { (free_table ?r) }, $pre_{\dashv} = \emptyset$
$pre_{\leftrightarrow}$ = { (open ?r), (time_for_eat ?t), (be ?t ?r) }
$eff_{\vdash} = \emptyset, eff_{\dashv}$ = { (eaten ?t) }
dur = (eat_time ?t ?r)

Definition 3.3 *A* **temporal plan** Π *is a set of pairs* (a, t)*, where* $a \in O$ *and* t *is the start execution time of* a*.*

The temporal plan for the running example is shown in Figure 2.

Definition 3.4 *Given a temporal plan* Π*, the* **induced plan** Π^* *for* Π *is the set of pairs defined as* [7]*:*

$$\Pi^* = \{(a_{\vdash}, t), (a_{\dashv}, t + dur(a))\}, \quad \forall (a, t) \in \Pi$$

For simplicity, we will refer to any pair in Π^* as an "action".

For example, Π^* would include $(\texttt{(walk T Caro Lonja)}_{\vdash}$, 10:00) and $(\texttt{(walk T Caro Lonja)}_{\dashv}$, 10:09). In the induced plan, we only consider the start and the end time points of the actions in the original temporal plan because these are the time points interesting for building the state resulting from the execution at a certain time point t, as shown below.

Definition 3.5 *Given a state S_t defined as a set of propositions that are true at time t and a pair $(a_x, t) \in \Pi^*$, an action a_x is* **applicable** *in S_t if $pre_x(a) \cup pre_{\leftrightarrow}(a) \subseteq S_t$.*

With this definition, only the start and the end time points of the actions are considered. In order to mitigate the fact that the overall conditions are not checked during the execution of the action, we check them at the start and the end of the action, although this is not consistent with the definition of a durative action in PDDL2.1 (where the *overall* conditions of an action a starting at t have to be fulfilled during the interval $[t + \varepsilon, t + dur(a) - \varepsilon]$).

Definition 3.6 *Given a time point t, let Π_t^* be the subset of actions $(a, t') \in \Pi^*$ such that $t' < t$. A* **temporal state** *S_t is composed by a set of propositions $p \in P$ and a set of TILs (denoted by Γ_t) resulting from applying the actions in Π_t^* from I (denoted by $I \rightarrow_{\Pi_t^*} S_t$), that is, it is assumed that the actions in Π_t^* are applicable in the corresponding state. Formally, we define S_t recursively, where $S_0 = I$:*

$$S_t = S_{t-1} - \left(\bigcup_{\forall (a_x, t') \in \Pi_t^*} eff_x^-(a) \right) \cup \left(\bigcup_{\forall (a_x, t') \in \Pi_t} eff_x^+(a) \right)$$

For example, the state $S_{10:09}$ reached after the execution of the action ((walk T Caro Lonja)$_\dashv$,10:09h) will be the same as the initial state but $S_{10:09}$ will contain the new location of the tourist (be T Lonja). It is important to notice that if we compute the state $S_{10:05}$, then the location of the user is unknown because the proposition (be T Caro) is deleted at the start time of the execution of the action and the proposition (be T Lonja) is not added until the end time of the action.

Definition 3.7 *The* **duration** *(makespan) of an induced plan Π^* executed from the initial state of the problem is $dur(\Pi^*) = max_{\forall (a_\dashv, t) \in \Pi^*} \big(t\big) - T_i$; i.e., the end time of the action that finishes last minus the time point at which the plan starts T_i, assuming that all the actions in Π^* are applicable in the corresponding state.*

For instance, $dur(\Pi^*)$ in Figure 2 is 7 h. and 9 min., because the last action ends at 17:09h and the plan starts at 10h.

Definition 3.8 *An induced plan Π^* is a* **solution for a temporal planning problem with deadline constraints** *$\mathcal{P} = \big\langle P, O, I, G, D \big\rangle$ if the following conditions hold:*

1. *$G \subseteq S_g$, where $I \rightarrow_{\Pi_t^*} S_g$, where $t = dur(\Pi^*)$*
2. *$\forall (p, t) \in D : \exists t' \leq t : p \in S_{t'}$, where $I \rightarrow_{\Pi_{t'}^*} S_{t'}$*

This definition indicates that it is not only necessary that a plan Π^* reaches the goals, but also that all the propositions present in D are achieved before the corresponding deadline.

3.2 TempLM

TempLM [11] is a temporal planning approach for solving planning problems with deadlines that has demonstrated an excellent behaviour in the detection of unsolvable problems and the resolution of overconstrained problems. It draws upon the concept of temporal landmark, which is a proposition that is found to necessarily happen in a solution plan in order to satisfy the deadlines of the problem goals. The set of temporal landmarks extracted from a problem along with their relationships form a temporal landmarks graph (TLG) that is conveniently used to take decisions during the construction of the solution plan and for guiding the search process.

Definition 3.9 *A* **Temporal Landmarks Graph (TLG)** *is a directed graph $G = (V, E)$ where the set of nodes V are landmarks and an edge in E is a tuple of the form $(l_i, l_j, \prec_{\{n,d\}})$ which represents a necessary or dependency ordering constraint between the landmarks l_i and l_j, denoting that l_i must happen before l_j in a solution plan.*

Landmarks are also annotated with various *temporal intervals* that represent the validity of the corresponding temporal proposition ([11]):

- The **generation interval** of a landmark is denoted by $[min_g(l), max_g(l)]$. $min_g(l)$ represents the earliest time point when landmark l can start in the plan. $max_g(l)$ represents the latest time point when l must start in order to satisfy D.
- The **validity interval** of a landmark l is denoted by $([min_v(l), max_v(l)])$ and it represents the longest time that l can be in the plan.
- The **necessity interval** of a landmark l is denoted by $([min_n(l), max_n(l)])$ and it represents the set of time points when l is required as a condition for an action to achieve other landmarks.

These intervals are given some initial values that are then updated by means of a propagation method, as explained in [11]. In order to be consistent, both the generation interval and the necessity interval must fall into the validity interval. Figure 3 shows a part of the initial TLG built for this problem. The validity interval of a landmark is represented by a segment, the max_g is indicated by a small bold vertical line (min_g is always equal to min_v) and the necessity interval is represented by a green box inside the segment. For example, the validity interval of the landmark (visited T Cathedral) is [12:05h, 19h] and the $max_g = 19h$; the necessity interval for this landmark is empty.

TempLM applies a search process in the space of partial plans in order to find a solution plan for a problem $\mathcal{P}$. A node in the search tree is defined as $n = (\Pi, S_t, TLG_\Pi)$, where Π is the conflict-free partial plan of n, S_t is the state reached at time $t = dur(\Pi)$ after the execution of Π from I and TLG_Π is the refined TLG after taking into account the information in Π. Given a node n of the search tree, a successor of n results from adding an executable action a to the partial plan of n, provided that a does not cause conflicts with the actions of n. Hence, the plan of the successor node will contain a newly added action, information which can be exploited to find new temporal landmarks in the TLG [12].

4 CHANGES IN THE ENVIRONMENT

This section introduces some definitions related to the management of changes in the environment. We assume that in our system changes happen due to exogenous events.

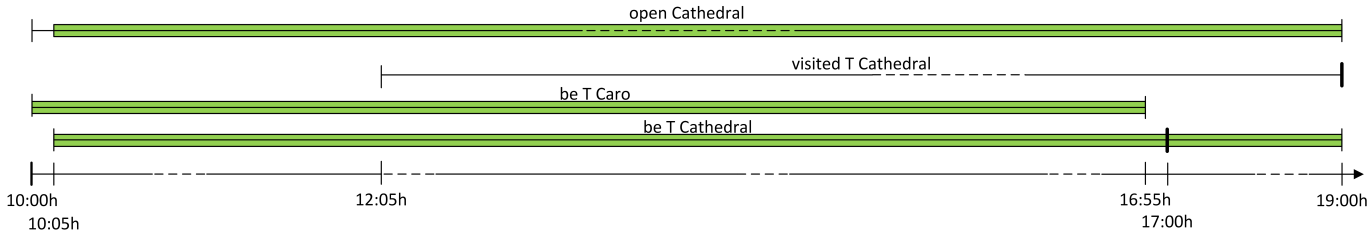

Figure 3. Initial partial TLG for this problem

Definition 4.1 *We define an* **exogenous event** θ *as a tuple* (t_a, p, t_c)*, where* t_a *is the* arrival time point*, that is, the time point when the exogenous event is received in the system, and* (p, t_c) *denotes a TIL.*

For example, let ((not (open Centralmarket)),17h) be a TIL in I that indicates that the Central Market will close at 17h. If at 14h it is known that the Central Market will close one hour earlier, the exogenous event that will be received by the system is the following: (14h, (not (open Centralmarket)), 16h) and it will invalidate the previous TIL.

Definition 4.2 *We say that two exogenous events* (t_a, p, t_c) *and* (t'_a, p', t'_c) *are* linked[2] *if they refer to the same proposition* $(p = p')$ *and their arrival time is the same* $(t_a = t'_a)$*.*

We can indicate a modification in the actual duration of an action with two linked exogenous events that refer to the proposition(s) that will be achieved as a result of the execution of the action. The first one deletes a proposition p at the time it was expected and the second event adds that proposition at the new time. For example, if at 13h it is known that the action `(eat T RicardCamarena)` to be executed from 14h to 15:30h will take 30 minutes more than expected, these two linked exogenous events are received by the system: (13h, (not (eaten T)), 15:30h) and (13h, (eaten T), 16h). The first one deletes the expected event of the tourist having finished lunch at 15:30h and the second adds the proposition at the time when the tourist will actually finish having lunch.

We denote by t_{cur} the current execution time when an event θ is received and by $S^o_{t_{cur}}$ the current (observed) state. Therefore, $S^o_{t_{cur}}$ will contain the propositions that are true in the current world and the functions with their actual value in the current world. Moreover, let $\Gamma_{t_{cur}-\varepsilon}$ be the set of TILs in the state at the time immediately prior to t_{cur}. $S^o_{t_{cur}}$ will contain the TIL in θ plus the TILs in $\Gamma_{t_{cur}-\varepsilon}$ that are consistent with θ. For example, as Figure 2 shows, $\Gamma_{14h-\varepsilon}$ contains, among others, the TILs in Table 1 (top). If the event θ =(14h, (not (open Centralmarket)), 17:30h) is received, then Γ_{14h} in $S^o_{t_{cur}}$ will be updated as shown in Table 1 (bottom). That is, the time window in which the Central Market remains open is updated from [15:30h, 19h] to [15:30h, 17:30h].

Definition 4.3 *We define the* **executed partial plan at** t_{cur}*, denoted by* Π^*_{ex}*, as:*

$$\Pi^*_{ex} = \{(a, t) \in \Pi^* : t < t_{cur}\}$$

[2] For simplicity, we denote by θ a single event or two linked events.

Table 1. TILs at 14h

$\Gamma_{14h-\varepsilon}$
((open Centralmarket), 15:30h),
((not (open Centralmarket)), 19h),
((not (open Ricardcamarena)), 17h), ...
Γ_{14h}
((open Centralmarket), 15:30h),
((not (open Centralmarket)), 17:30h),
((not (open Ricardcamarena)), 17h), ...

Definition 4.4 *We define the* **remaining partial plan at** t_{cur}*, denoted by* Π^*_r*, as:*

$$\Pi^*_r = \{(a, t) \in \Pi^* : t \geq t_{cur}\}$$

Definition 4.5 *We define the* **resulting state of** Π^*_{ex} *(*$S_{t_{cur}}$*) as the state reached after executing the actions in* Π^*_{ex}*, that is:* $I \rightarrow_{\Pi^*_{ex}} S_{t_{cur}}$

Definition 4.6 *We define the* **difference** *between two states* S_i *and* S_j *as:*

$$Diff(S_i, S_j) = (S_i - S_j) \cup (S_j - S_i)$$

Definition 4.7 *A* **discrepancy** *is a non-empty difference between the state that should have been reached with the executed part of the plan* $S_{t_{cur}}$ *and the current observed state* $S^o_{t_{cur}}$*, that is:* $Diff(S_{t_{cur}}, S^o_{t_{cur}}) \neq \emptyset$*.*

Obviously, the discrepancy between these two sets will at least contain the TIL from the event θ. For example, if we consider the example in Table 1, the event that arrives at 14h is reflected in the resulting Γ. Discrepancies may cause failures or opportunities in the plan.

Definition 4.8 *We say that there is a* **failure in the plan** *if:*

1. *There is a discrepancy at* t_{cur}*, and*
2. $S^o_{t_{cur}} \rightarrow_{\Pi^*_r} S_g : G \not\subseteq S_g$*, that is, the plan does not reach the goal due to this discrepancy.*

Note that Π^* was a solution for the initial planning problem, that is, it was executable from I. This definition indicates that there has been a change in the environment at a certain time point between I and t_{cur} which causes a failure in the execution of action a and, consequently, a failure in the execution of the plan.

Definition 4.9 *We say that there is an* **opportunity in the plan** *if:*

1. *There is a discrepancy at t_{cur}, and*
2. *$S^o_{t_{cur}} \rightarrow_{\Pi^*_r} S_g : G \subseteq S_g$, that is, the plan reaches the goal in spite of this discrepancy, and*
3. *this discrepancy causes a different execution of the remaining plan, that is, there is a time point t where the state reached from $S_{t_{cur}}$ is different from the state reached from $S^o_{t_{cur}}$; formally, given $t \in [t_{cur}, dur(\Pi^*)]$, let Π' be the set of actions in Π^*_r scheduled to start between t_{cur} and t; we define S^o_t and S_t as $S^o_{t_{cur}} \rightarrow_{\Pi'} S^o_t$ and $S_{t_{cur}} \rightarrow_{\Pi'} S_t$, respectively; the execution of the remaining plan is different if $Diff(S_t, S^o_t) \neq Diff(S_{t_{cur}}, S^o_{t_{cur}})$.*

An opportunity in this case is defined as a discrepancy that still allows to reach the goals but that causes a difference in the execution of the original plan.

It could be the case that a discrepancy does not cause a failure or an opportunity, because it affects an object that is not considered in the current plan.

5 DETECTION OF FAILURES AND OPPORTUNITIES IN TempLM

The TLG that TempLM builds to solve a planning problem gives an schema of which propositions must be achieved in the plan, and when they must be achieved in order to reach the goals on time. Additionally, a Temporal Proposition Graph, which gives an exact picture of the propositions that are achieved in the plan and when they are achieved, is defined:

Definition 5.1 *A* **Temporal Proposition Graph (TPG)** *for a given induced plan Π^* is a representation of the time interval in which a proposition $p \in P$ is true during the execution of Π^*. This is denoted as the validity interval of the proposition p. A TPG also defines the generation and necessity intervals of a proposition, with the same meaning as for a landmark in the TLG.*

Figure 4 shows the TPG for the plan in Figure 2. In this case, all the propositions that appear during the execution of the plan are represented. It can be observed that the validity interval of the propositions in the TPG is much smaller than in the TLG (see Figure 3), given that the TLG is just an estimation, whereas the TPG is an accurate representation of the solution plan. For example, in Figure 3, proposition (be T Cathedral) ranges from 10:05h until 19h, whereas in Figure 4 it is shrunk to the interval [11:34h, 13:34h].

The aim of this section is to explain how both the TLG and the TPG can be used in order to detect failures and opportunities due to exogenous events that may arrive during the execution of a plan, and also to give some hints about how they can be solved or exploited.

As explained in section 2, when a new exogenous event is detected by the central module, TempLM receives the current execution time t_{cur}, the current observed state $S^o_{t_{cur}}$ and the exogenous event θ as input. Then, TempLM computes the state that should have been reached as a result of the execution of the plan until time t_{cur}, that is, it computes $S_{t_{cur}}$. The next step is to analyze whether the event θ has caused a discrepancy between $S_{t_{cur}}$ and $S^o_{t_{cur}}$, as indicated in Definition 4.7. In case of a discrepancy between both states, several cases can be found.

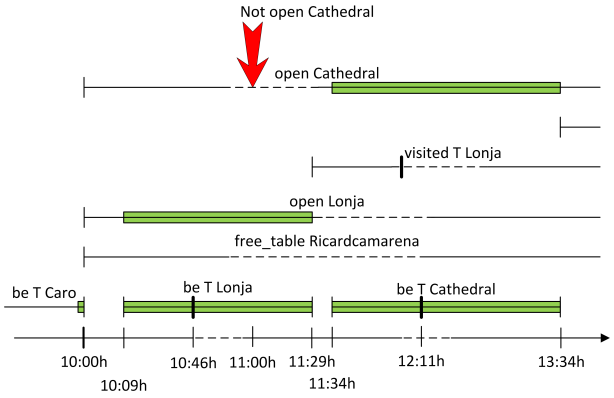

Figure 5. Example of future failure

Case 1. The discrepancy does not affect the plan
This case corresponds to a situation where the exogenous event $\theta = (t_a, p, t_c)$ affects a proposition that has not any influence in the plan. This means that the proposition p that causes the discrepancy is not present in the TPG of the plan and it is not mutex [3] with any other proposition in the TPG. Formally, TempLM does not detect any failure or opportunity if:

$$p \notin TPG_{\Pi^*} \wedge \neg\exists p' \in TPG_{\Pi^*} : mutex(p, p')$$

For example, let us assume that this exogenous event (10, (not (free_table Cantinella)), 13h) is received, indicating that the Cantinella restaurant will not have free tables at 13h. This does not affect the current plan, given that the restaurant where the tourist is going to have lunch is a different one.

Case 2. A failure is detected
The second analysis is aimed at detecting a failure caused by an exogenous event $\theta = (t_a, p, t_c)$. We distinguish two situations: when the event causes a present failure, that is, $t_a = t_c = t_{cur}$ or when it causes a future failure, that is, $t_a = t_{cur} < t_c$. In both cases, a proposition $p' : mutex(p, p')$ which belongs to the TPG is deleted before expected when it is still needed to reach the goals. Formally, TempLM detects a failure if:

$$p' \in TPG_{\Pi^*} \wedge t_c < max_n(p')$$

For example, if at $t_{cur} = 10h$ an event indicating that the Cathedral will be closed at 11h arrives, expressed as θ =(10, (not (open Cathedral)), 11h), a failure is caused because t_c =11h, $mutex$((not(openCathedral)), (openCathedral)) and $t_c < max_n$(open Cathedral) =13:34h, as it can be observed in Figure 5. In this case, TempLM is able to easily predict a future failure due to an exogenous event.

Once the failure is detected, two situations can occur: (1) the problem is unsolvable or (2) the problem can be solved using other elements defined in the context. In order to distinguish these situations, TempLM finds the node $(\Pi, S, TLG_{\Pi^*_{ex}})$

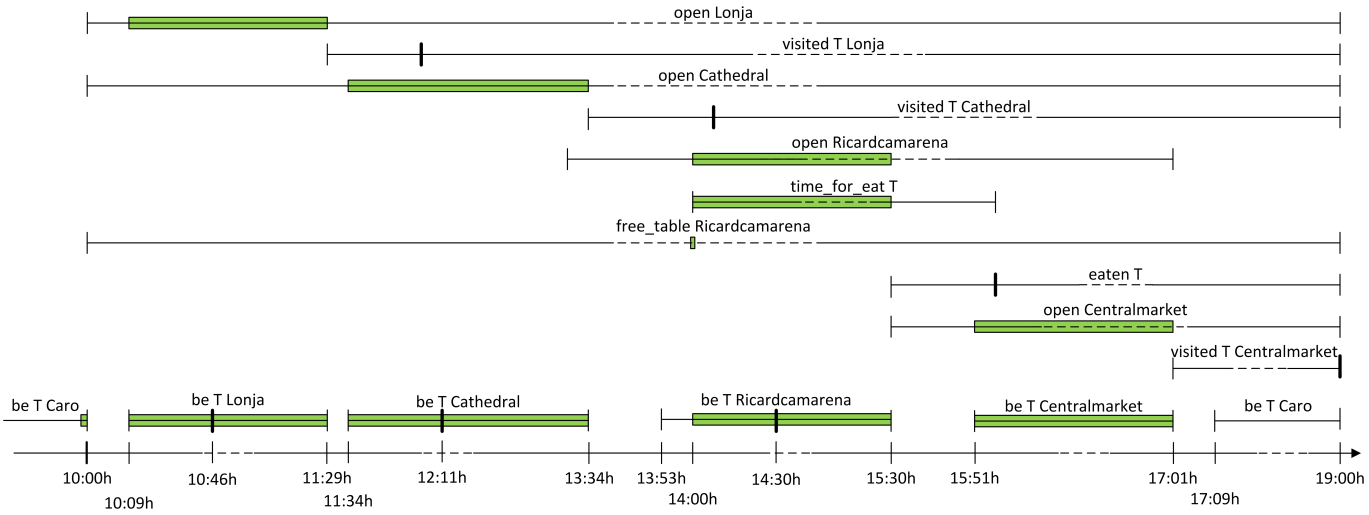

Figure 4. Temporal Proposition Graph of the plan

in the search space built when solving the original problem, where $S = S_{t_{cur}}$ computed from Π^*_{ex}. If the TLG in this node is updated with the information about the event θ, this new information is propagated across the TLG (obtaining $TLG^{\theta}_{\Pi^*}$) and an inconsistency [11] between two intervals of a given proposition is found, then the problem is unsolvable. It is important to remark that the propositions in the TLG are necessary to solve the problem (they are landmarks); therefore, if an inconsistency is found in the intervals associated to a landmark, then the problem is unsolvable. Formally:

$$\exists p \in TLG^{\theta}_{\Pi^*} : max_g(p) \notin [min_v(p), max_v(p)]$$

For example, if the event (10, (not (open Cathedral)), 11h) arrives, then the problem is unsolvable because, due to θ, the value of the validity interval of (open Cathedral) changes, i.e. max_v(open Cathedral) $= 11h$. This information is propagated to the proposition (visited T Cathedral) in the TLG^*_{Π}, which updates max_g(visited T Cathedral) $= 11h$, indicating that, in order to reach the goals, the Cathedral must be visited before 11h. Given that this value does not belong to the validity interval of (visited T Cathedral), as it can be observed in Figure 3, there is an inconsistency in these intervals. Therefore, we can affirm that the problem, after the arrival of the event, is unsolvable. In fact, there is not a plan that satisfies the goals, because even in the case that the Cathedral is the first visit, it takes 2 hours and then, it would last until 12:05h, which is later than the time of closing.

In any other case, TempLM performs the search of a new plan starting from Π^*_{ex}, that is, Π^*_r is discarded and substituted by the eventually new plan found. If, for example, an event indicating that the Ricard Camarena restaurant will be fully-booked at 13:45h (θ =(13, (not (free_table Ricardcamarena)),13:45h) arrives, then TempLM detects a failure because t_c =13:45h $<$ max_n(free_table Ricardcamarena) =14:00h. In this situation, TempLM would be able to find a different solution plan, because having lunch at Ricard Camarena restaurant is not a hard goal; the hard goal is (eaten T), which can be achieved by having lunch in any other restaurant. Therefore, a new plan from Π^*_{ex} in which the tourist has lunch in a different restaurant could be found by TempLM.

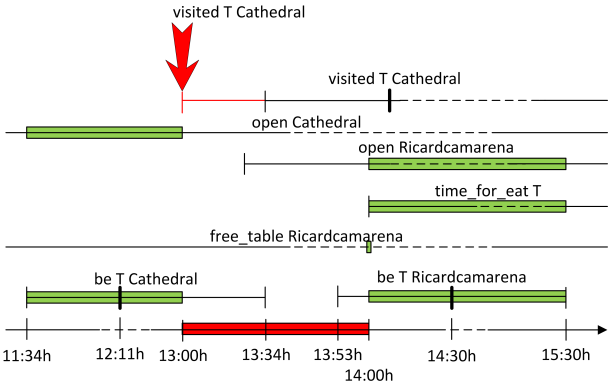

Figure 6. First example of a future opportunity

Case 3. An opportunity is detected

The last analysis is devoted to detect an opportunity when a new exogenous event $\theta = (t_a, p, t_c)$ arrives. In this case, p is a proposition that is achieved before than expected, which permits to consider an action as finished before its complete execution. Formally, TempLM detects an opportunity if:

$$p \in TPG_{\Pi^*} \wedge t_c < min_v(p)$$

For example, the arrival of an event in which the visit to the Cathedral finishes at 13h, i.e. θ =(13, (visited Cathedral), 13h), causes the detection of an opportunity. The red arrow in Figure 6 indicates the new time point at which the proposition (visited Cathedral) is achieved and the red rectangle indicates

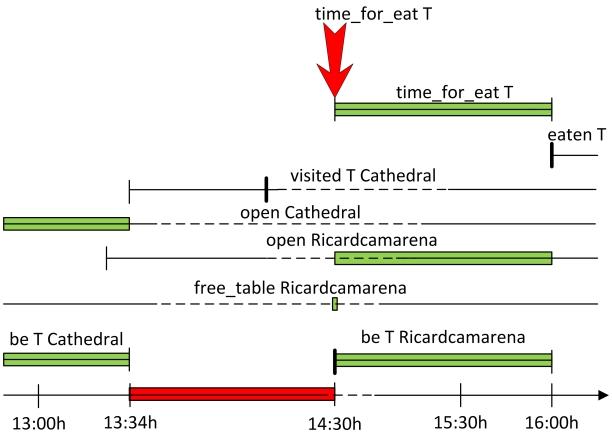

Figure 7. Second example of a future opportunity

the available interval of time thanks to this event (which can be considered as an opportunity). It can be observed that, in this example, t_{cur} =13h $< min_v$((visited Cathedral)) =13:34h and the necessity interval of (open Cathedral) and (be T Cathedral) are shrunk because (visited Cathedral) has been achieved before than expected. At this moment, the recommender system could be invoked to obtain a new visit (a new goal) to be performed during the time interval between 13h and 14h, before (be T Ricardcamarena). TempLM would obtain a plan to reach this new goal and then the remaining original plan would be executed.

There is another case where TempLM can detect an opportunity:

$$p \in TPG_{\Pi^*} \wedge t_c > min_v(p) \wedge$$

$$\neg \exists q \in TPG^{\theta}_{\Pi^*} : max_g(q) \notin [min_v(q), max_v(q)]$$

This case represents the situation when a proposition is achieved later than expected, but it does not affect the achievement of the remaining goals. That is, there is not any proposition q whose intervals are inconsistent after updating the TPG_{Π^*} with the exogenous event and propagating the information across the TPG_{Π^*} to obtain $TPG^{\theta}_{\Pi^*}$.

For example, let us assume that the tourist now prefers to have lunch between 14:30h and 16h, instead of between 14h and 16h, but the action still takes 90 minutes. This situation triggers the following exogenous event: θ =(13, (time_for_eat T), 14:30h). This event causes that TempLM detects an opportunity because min_v(time_for_eatT) =14:00h and t_c =14:30h; this provokes that min_v(time_for_eat T) is updated and this information is propagated to min_v(eatenT) (max_g(eaten T) still belongs to the corresponding validity interval). Additionally, the necessity intervals of (time_for_eat T), (open Ricardcamarena) and (be T Ricardcamarena) are also updated, as it is shown in Figure 7. The available time for the opportunity, also shown in Figure 7, can be used to reach a new goal recommended by the recommender system, as explained above.

Thanks to the analysis presented in this section, we have been able to show how TempLM is able to predict future failures or opportunities.

6 CONCLUSIONS AND FURTHER WORK

This paper has introduced a goal reasoning framework where context information acquired from several external sources determines a change that may affect the execution of the current plan. This change may cause a failure or an opportunity. We have shown how the planning system, namely TempLM, is able to predict both failures and opportunities thanks to the analysis of the Temporal Landmarks Graph and the Temporal Propositions Graph built for the given problem.

As for further work, our aim is to implement in TempLM the analysis described in this paper. Then, we will be able to test the system in a real environment. This will imply an analysis to decide which events should be considered among the huge amount of context information that is supplied by external sources.

ACKNOWLEDGMENTS

This work has been partially supported by Spanish Government Project MINECO TIN2014-55637-C2-2-R.

REFERENCES

[1] E. Aarts and J.L. Encarnaao, *True visions: the emergence of ambient intelligence*, New York. Springer, 200.
[2] M. Beetz and H. Grosskreutz, 'Probabilistic hybrid action models for predicting concurrent percept-driven robot behavior', *J. Artif. Intell. Res. (JAIR)*, **24**, 799–849, (2005).
[3] A. Blum and M. Furst, 'Fast planning through planning graph analysis', *Artificial Intelligence*, **90**(1-2), 281–300, (1997).
[4] D. Buhalis and A. Amaranggana, 'Smart tourism destinations enhancing tourism experience through personalisation of services', in *Information and Communication Technologies in Tourism 2015*, 377–389, Springer, (2015).
[5] A. J. Coles, A. I. Coles, M. Fox, and D. Long, 'Colin: Planning with continuous linear numeric change', *Journal of Artificial Intelligence Research*, **44**, 1–96, (2012).
[6] S. Edelkamp and J. Hoffmann, 'Pddl2. 2: The language for the classical part of the 4th international planning competition', *4th International Planning Competition (IPC'04)*, (2004).
[7] M. Fox and D. Long, 'Pddl2. 1: An extension to pddl for expressing temporal planning domains.', *J. Artif. Intell. Res.(JAIR)*, **20**, 61–124, (2003).
[8] M. Ghallab, D. Nau and P. Traverso, *Automated Planning. Theory and Practice.*, Morgan Kaufmann, 2004.
[9] U. Gretzel, M. Sigala, Z. Xiang, and C. Koo, 'Smart tourism: foundations and developments', *Electronic Markets*, **25**(3), 179–188, (2015).
[10] J. Ibañez, L. Sebastia, and E. Onaindia, 'Planning tourist agendas for different travel styles', in *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, p. in press, (2016).
[11] E. Marzal, L. Sebastia, and E. Onaindia, 'On the use of temporal landmarks for planning with deadlines', in *Proc. of the 24th International Conference on Automated Planning and Scheduling*, pp. 172–180. AAAI Press, (2014).
[12] E. Marzal, L. Sebastia, and E. Onaindia, 'Temporal landmark graphs for solving overconstrained planning problems', *Knowledge-Based Systems*, (2016).
[13] L. Spalazzi, 'A survey on case-based planning', *Artificial Intelligence Review*, **16**, 3–36, (2001).
[14] S. Vattam, M. Klenk, M. Molineaux, and D. W. Aha, 'Breadth of approaches to goal reasoning: A research survey', Technical report, DTIC Document, (2013).