# Towards Continuous Activity Monitoring with Temporal Constraints

**Jonas Ullberg, Amy Loutfi** and **Federico Pecora**
Center for Applied Autonomous Sensor Systems
Örebro University, SE-70182 Sweden
<name>.<surname>@oru.se

**Abstract**

This paper focuses on a temporal reasoning approach for human activity recognition. Specifically, we show how search and temporal propagation are used to enable long term and continuous activity recognition. Two specific issues are addressed, namely maintaining performance over long monitoring horizons and ensuring future temporal consistency of recognized activities. We propose a complete search algorithm for activity recognition which addresses these issues, in which an admissible pruning technique allows improved performance. We show a sufficient condition for guaranteeing future admissibility, and experimental results which test the limits and practical applicability of the system are presented.

## Introduction

The demand for intelligent services in home environments can be expected to grow in the near future as ubiquitous sensors become more widely available and mature. However, sophisticated intelligent home services cannot be provided in a purely reactive fashion, since they require contextual knowledge about the environment such as the activities the residents are engaged in at any given time. Obtaining such knowledge poses a series of interesting problems since it is often the case that information about human behavior cannot be sensed directly. The key to providing context about a human user in a sensor-rich environment lies in aggregating percepts from multiple sensors.

Due to its importance for realizing intelligent environments, activity recognition has received much attention in the literature and the term has been employed to indicate a variety of capabilities. In this paper we take activity recognition to mean the ability of the intelligent system to deduce temporally contextualized knowledge regarding the state of the user on the basis of a set of heterogeneous sensor readings. Prior approaches to the problem of recognizing human behavior can roughly be categorized by how the input data is processed. A common solution is the *data-driven* approach, in which models of human behavior are learned from large volumes of data over time. Notable examples include techniques employing Hidden Markov Models (Sanchez, Tentori, and Favela 2007) or neural networks (Györbíró, Fábián, and Hományi 2009). *Knowledge-driven* approaches on the other hand follow a complementary strategy in which patterns of observations are modeled from first principles rather than learned. In these approaches, sensor data is explained by hypothesizing the occurrence of human activities based on rich representations modeling typical conditions of the environment under which these activities occur. A direction currently emerging from the literature consists in representing these conditions as temporal constraints, as done for instance in (Cirillo et al. 2009; Zouba, Bremond, and Thonnat 2009; Jakkula, Cook, and Crandall 2007).

In this paper we focus on a knowledge-driven approach based on temporal constraint reasoning, in which knowledge about the environment is represented as symbolic values which are constrained with relations in Allen's Interval Algebra (Allen 1984). As in other knowledge-based approaches for activity recognition, we employ an iterated abductive process in which sensor data is explained by hypothesizing the occurrence of specific human activities. Data accumulated over days or weeks of monitoring must be analyzed so as to infer states of the monitored human being that depend on events that are separated by long temporal intervals, e.g., recognizing activities that occur every Monday. Also, we require the system to be capable of recognizing activities as soon as they occur. This requirement arises in contexts where the inferred activities should lead to the timely enactment of appropriate procedures. These requirements entail two problems.

First, the on-line nature of the application combined with the long temporal horizon over which inference is performed entails strong scalability requirements. This is achieved in our system through an admissible heuristic which markedly decreases the computational load of inference. This admissible pruning of the search space is obtained by maintaining a history of previously attempted inferences. This article analyzes the computational bounds of the resulting iterative inference process, and an experimental evaluation of the system in a number of worse case and realistic conditions is given.

Secondly, the consistency of inferred states must be considered, as any inference on the state of the user should be guaranteed to be valid under all possible future evolutions of sensor readings. In this paper we solve this issue by building into the inference procedure a set of criteria which disallows abductive inference when support cannot be guaranteed to

persist in future evolutions of sensor readings.

This paper is organized as follows. We first briefly summarize the temporal constraint based approach to activity recognition employed in our architecture. We then analyze the computational bounds of the continuous inference process and describe how to obtain an admissible pruning of the search space. We address the problem of maintaining future consistency, and conclude with an experimental evaluation of the system in several worse case, and realistic conditions.

## Constraint-Based Activity Recognition

The activity recognition system which is the object of this paper is part of an activity management system described in (Pecora and Cirillo 2009). This system deals with both activity recognition and contextual plan synthesis and actuation. In this paper we focus on the activity recognition capabilities of the system, and specifically on how they are adapted to operate continuously in scenarios where activity recognition must occur on-line, as soon as possible, and over long horizons. In this section we briefly describe the underlying activity recognition system employed, introducing the notation used in the rest of this article.

The activity recognition system is realized within the OMPS temporal reasoning framework. OMPS (Fratini, Pecora, and Cesta 2008) is a constraint-based planning and scheduling software API for developing temporal planning and scheduling applications, and has been used to develop a variety of decision support tools, ranging from highly-specialized space mission planning software to classical planning frameworks[1]. Our system leverages the domain description language provided by OMPS to model the dependencies that exist between sensor readings and the state of the human user. Specifically, we employ the state variable type to model one or more aspects of the user's activities of daily living. For instance, in the examples that follow we will use a state variable whose values are $\{\mathbf{Sleeping}, \mathbf{Cooking}, \mathbf{Eating}, \mathbf{InBed}, \mathbf{WatchingTV}, \mathbf{Out}\}$ to model the human user's domestic activities. We also employ the state variable metaphor for modeling the possible states of sensors. For instance, if the intelligent environment is equipped with a person localization sensor which can determine the position of the user in various rooms, we model a state variable whose values are $\{\mathbf{Kitchen}, \mathbf{Livingroom}, \mathbf{Bathroom}, \ldots\}$.

The current state of the state variables representing the user and the sensors is maintained in a *decision network* (DN), a network whose nodes represent values of state variables and whose edges represent temporal constraints among these values. More specifically, a node of the network is called a *decision*, and represents an assertion on the value of a state variable in a given flexible time interval:

**Definition 1.** *A decision $d^x_{\mathbf{v}}$ is a pair $\langle \mathbf{v}, [I_s, I_e] \rangle$, where $x$ is a state variable, $\mathbf{v}$ is a possible value of the state variable $x$, and $I_s, I_e$ represent, respectively, an interval of admissibility of the start and end times of the decision.*

[1] A complete description of OMPS (Open Multi-component Planner and Scheduler) is outside the scope of this paper.

For instance, the decision $d^{\text{Light}}_{\text{Off}} = \langle \mathbf{Off}, [[10, 10], [40, 40]] \rangle$ on a state variable representing a luminosity sensor could model the fact that the the environment was dark from time instant 10 to time instant 50. We express the lack of knowledge on the temporal evolution of the light sensor with flexible bounds on the end time of the decision. For instance, $d^{\text{Light}}_{\text{Off}} = \langle \mathbf{Off}, [[10, 10], [40, \infty)] \rangle$ expresses the fact that the sensor has ceased to perceive light at time 10 and that this perception has not thus far (time instant 40) changed. An example evolution of this sensor's readings is given in Figure 1, where the unbounded end time of the light's and bed's states are represented with a dashed extension. As the flexible time intervals of decisions are used to model the evolution of sensor readings in time, the same principle is used to model temporal uncertainty on the state of human activities. For instance, the decision $d^{\text{Human}}_{\text{Sleeping}} = \langle \mathbf{Sleeping}, [[20, 20], [40, \infty)] \rangle$ on a state variable modeling the activity of a human being represents the fact that he or she has begun to sleep at time instant 20, and that this activity does not end before time instant 40.

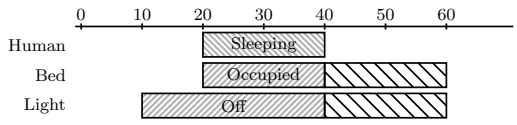


Figure 1: Timelines of three state variables: the first representing a human user, the other two representing sensors in the environment.

Decisions are bound by *temporal constraints*. Such constraints are bounded variants of the relations in the restricted Allen's Interval Algebra (Allen 1984; Vilain, Kautz, and van Beek 1989). Temporal constraints enrich Allen's relations with bounds through which it is possible to fine-tune the relative temporal placement of constrained decisions. For instance, suppose the two decisions $d^{\text{Human}}_{\text{Sleeping}}$ and $d^{\text{Light}}_{\text{Off}}$ model, respectively, the state of the human and the state perceived by the luminosity sensor. The constraint $d^{\text{Human}}_{\text{Sleeping}}$ DURING $[3, 5][0, \infty)$ $d^{\text{Light}}_{\text{Off}}$ then states that **Sleeping** should be temporally contained in **Off**, that the start time of **Sleeping** must occur between 3 and 5 units of time after the beginning of **Off**, and that the end time of **Sleeping** should occur some time before the end of **Off**.

The DN is at all times kept consistent through *temporal constraint propagation*. This ensures that the temporal intervals underlying the decisions are kept consistent with respect to the temporal constraints, while decisions are anchored flexibly in time. In other words, adding a temporal constraint to the DN will either result in the calculation of updated *bounds* for the intervals $I_s, I_e$ for all decisions, or in a propagation failure, indicating that the added constraint or decision is not admissible. For instance, given the constraints depicted in Figure 2, namely $d^{\text{Human}}_{\text{Sleeping}}$ DURING $[0, \infty)[0, \infty)$ $d^{\text{Light}}_{\text{Off}}$ and $d^{\text{Human}}_{\text{Sleeping}}$ EQUALS $[0, \infty)[0, \infty)$ $d^{\text{Bed}}_{\text{Occupied}}$, temporal propagation would update the bounds of $d^{\text{Human}}_{\text{Sleeping}}$ to $[[20, 20], [40, \infty)]$. In OMPS, temporal constraint propagation is a polynomial time operation, as it is based on a Simple Temporal Network (Dechter, Meiri, and Pearl 1991) in

which each decision is represented by two time points (start and end times of the decision). Allen's temporal relations are represented as simple distance constraints between time points, according to simple transformations.

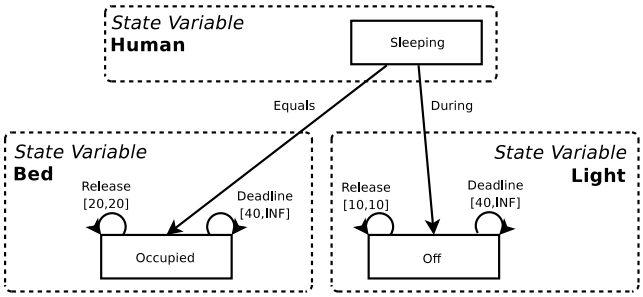


Figure 2: Example decision network involving three decisions, each on a different state variable. The timelines extracted from this DN are shown in Figure 1.

Given the DN, a *timeline* of a state variable shows the values of the state variable in time as they are determined by the decisions and constraints imposed on this state variable in the DN. Figure 1 shows a possible timeline for the three state variables described above. This timeline is obtained from the DN shown in Figure 2. Notice that, in general, it is possible to extract many timelines for a state variable, as constraints bound decisions' start and end times flexibly.

## Representing Sensor Readings in the Decision Network

Every sensor in the environment is represented by a state variable whose values model its possible sensor readings. The activity recognition system implements a sensing process for each of these state variables. If $DN_t$ represents the decision network at time $t$, running the sensing process at time $t' > t$ for the state variable $x$ will yield a new decision network $DN_{t'} = Sense_x(DN_t, t')$. This function, which is run iteratively at a given rate $f$, reads the interface of the sensor and models the current sensor reading in the decision network, yielding a DN in which the state variable is constrained to take on the sensed values in the appropriate time intervals. Specifically, if a new reading $\mathbf{v_s}$ is sensed at time $t_0$ by sensor $x$, a decision $d^x_{v_s} = \langle \mathbf{v_s}, [I_s, I_e] \rangle$ is added to the DN with a fixed interval of admissibility $I_s = [t_0, t_0]$ for its start time, and a flexible interval $I_e = [t_0, \infty)$ for its end time which reflects the uncertainty about the reading's temporal evolution. The lower bound on the end time interval is then periodically updated to reflect the incoming sensor readings: if at time $t_i > t_0$ the sensor provides the same reading, then the $Sense_x$ procedure will constrain $I_e$ to $[t_i, \infty)$, reflecting the fact that the sensor value persists at least until the current time $t_i$; conversely, if the sensor reading changes at time $t_i$, the interval of admissibility for the end time is fixed to $[t_i, t_i]$, reflecting the knowledge that the sensed value $\mathbf{v_s}$ persisted in the interval $[t_0, t_i]$. Thus, as time progresses, the end time intervals of sensed decisions are progressively constrained until their flexible intervals are in effect fixed in time.

## Continuous Inference Process

Along with a sensing procedure for each sensor, our system also runs an iterative inference process. This inference process operates at the same rate $f$ as the sensing procedures, thus realizing an on-line sensing-deduction loop.

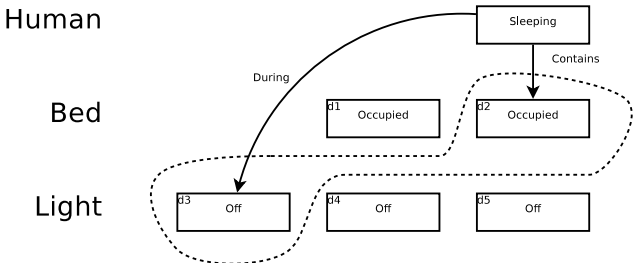


Figure 3: A hypothesis asserting that the human being is sleeping is supported by a possible combination of decisions on sensory state variables.

The inference procedure employs a *domain theory* which describes the conditions under which patterns of sensor readings indicate a certain human activity (i.e., a particular value of the state variable representing the user). These prerequisites are specified as *synchronizations*. A domain theory thus models a collection of criteria for recognizing activities from sensor readings. More specifically,

**Definition 2.** *A* synchronization *is a tuple* $\langle \langle \mathbf{v_{ref}}, x \rangle, \mathcal{R} \rangle$, *where*

- $\mathbf{v_{ref}}$ *is a* reference value *of a* reference state variable $x$*;*
- $\mathcal{R}$ *is a set of* requirements*, each in the form* $\langle \langle \mathbf{v_i}, j \rangle, R_i \rangle$ *where*
  - $\mathbf{v_i}$ *is a value of a state variable* $j$ *(called a* target value*)*
  - $R_i$ *is a bounded temporal constraint between the reference value* $\mathbf{v_{ref}}$ *of state variable* $x$ *and the target value* $\mathbf{v_i}$ *of state variable* $j$

For instance, a synchronization with reference value **Sleeping** on state variable $x = \text{Human}$ and requirements

$$\mathcal{R} = \{\langle \langle \mathbf{Occupied}, \text{Bed} \rangle, \text{CONTAINS} \rangle, \\ \langle \langle \mathbf{Off}, \text{Light} \rangle, \text{DURING} \rangle\}$$

models that the human being is sleeping during an interval of time when the light is off and temporally contains a sensor reading indicating that the bed sensor is activated.

The inference procedure implemented in our system continuously attempts to assess the applicability of a set of synchronizations in the DN. Specifically, this is done by adding a decision that represents a hypothesis on the current state of the monitored human being, selecting a synchronization whose reference value matches the current hypothesis, and *expanding* the synchronization's requirements. More specifically, expanding a set of requirements $\mathcal{R}$ equates to (1) adding a new decision to the DN with value $d^x_{v_{ref}} = \langle \mathbf{v_{ref}}, [[0, \infty), [0, \infty)] \rangle$; (2) finding a suitable set of decisions in the DN whose values are equal to the values $\mathbf{v_i}$; and (3) imposing the constraints $R_i$ between the reference decision $d^x_{ref}$ and these decisions. If the imposition of these

constraints does not lead to a propagation failure, then we say that the hypothesis has been *supported*. The "candidate" decisions that are used to represent the hypotheses in the DN are in the form $d_{hyp}^{\text{Human}} = \langle \mathbf{v}, [[0,\infty),[0,\infty)] \rangle$, representing the fact that the system will try to support a hypothesized state $\mathbf{v}$ in some interval of time for the monitored state variable Human. The procedure can be seen as an operator $Support(DN_t, d_{hyp}^x, \alpha)$ that returns true if the decision $d_{hyp}^x$ on monitored state variable $x$ can be supported in a decision network $DN_t$ using a set of target decisions $\alpha$, and false otherwise.

In the remainder of this paper, we indicate that there is a one-to-one matching between the values of decisions in a set $\alpha$ and the values of a set of requirements $\mathcal{R}$ with the notation $Unifies(\alpha, \mathcal{R})$.

Notice that for each synchronization, the framework must attempt to impose the required constraints between the candidate decision and a number of possible decisions whose values unify with the target values. An example of this is shown in Figure 3, where a candidate decision asserting that the human being is sleeping is supported by two of the six possible combinations of decisions modeling the state of the bed and light sensors in the environment. In the worst case, all possible selections of target decisions need to be explored: given a synchronization with requirements $\mathcal{R} = \{R_1 \dots R_{|\mathcal{R}|}\}$ and assuming there are $n_i$ decisions in the DN which unify with the requirement $R_i$, then it is necessary to perform $\prod_{i=1}^{|\mathcal{R}|} n_i$ tests. Under the simplified assumption that there are an equal number $m$ of applicable decisions for each requirement, the number of tests to be performed is $O(m^{|\mathcal{R}|})$. Notice also that every attempt to employ a combination of sensed decisions to provide support for a hypothesis will require temporal constraint propagation, which is polynomial in the number of decisions in the DN. Overall, it is clear that this will not scale well during long-term monitoring since the number of decision in the DN grows as time progresses.

The technique sketched above realizes an iterated abductive process, whereby sensor data is periodically explained by hypothesizing the occurrence of specific human activities. Note that the synchronizations provide a representation scheme similar to chronicles as described in (Dousson and Maigat 2007) and previous papers. This work is similar in that temporal propagation is used to determine when sensory events provide support for given chronicle descriptions. However, the work of Dousson et al. is event-driven, and new constraint networks are instantiated when sensory events occur. Conversely, in our system sensing and inference occur at a given rate, and all sensory events and deduced activities are maintained in one constraint network (the DN). One of the differences this entails is related to how the two approaches ensure scalability. In (Dousson and Maigat 2007), efficiency is obtained by curtailing the number of chronicle instances; in our system, assessing whether a synchronization applies requires non-trivial search in the space of possible combinations of sensor readings. This search space is characterized in the following section, and an admissible heuristic is described. Further comparison with the work by Dousson et at. is the topic of future work.

## Pruning the Search Space

In order to reduce the cost of supporting decisions, we introduce the concept of *inactive* decisions.

**Definition 3.** *A decision* $d = \langle \mathbf{v}, [[l_s, u_s], [l_e, u_e]] \rangle$ *is* inactive *when* $l_s = u_s$ *and* $l_e = u_e$.

A decision on a state variable modeling a sensor becomes inactive when the state variable's sensing procedure has bounded its end time (i.e., when the sensor reading is no longer being sensed). An important property of such decisions in our approach is that they can no longer provide new information to the inference process. In fact, since constraints on sensor decisions are only tightened, is easy to see that if $Support(DN_t, d_{hyp}^x, \alpha)$ is false, where $\alpha$ is a set of inactive decisions, then $Support(DN_{t'}, d_{hyp}^x, \alpha)$ will be false for all $t' > t$. The opposite also holds, i.e., if $Support(DN_t, d_{hyp}^x, \alpha)$ is true, then $Support$ holds true for the same set of target decisions for all $t' > t$. It should also be noted that if a new decision appears in the DN (such as a new percept) and it is taken into account for inference, i.e., one of the decisions in $\alpha$ is active, then the DN may indeed be found to support the hypothesis $d_{hyp}^x$. Overall, sets of inactive decisions represent support that will never affect the inference process in a new way. All decisions on sensory state variables are bound to become inactive, as the $Sense_y(DN_t, t')$ operator for sensor $y$ will eventually set the upper bound of any sensed decision to a fixed time.

Inactive decisions can be leveraged for pruning as shown in procedure `ActivityRecognition`. Specifically, the

**Procedure** `ActivityRecognition(`$x$,$DN_t$`)`

```
1  foreach v ∈ PossibleValues(x) do
2      d^x_hyp ← ⟨v, [[0,∞), [0,∞)]⟩
3      foreach Synch. S = ⟨⟨v_ref, x⟩, R⟩ : v_ref = v do
4          Ω ← ∅
5          foreach α ⊆ DN_t : Unifies(α, R) do
6              if ∃ d ∈ α : d is active then
7                  Ω ← Ω ∪ α
8          foreach α ∈ Ω do
9              if Support(DN_t, d^x_hyp, α) then
10                 DN_t ← DN_t ∪ d^x_hyp
11                 return success
12     return failure
```

procedure is applied to a monitored component $x$ given a decision network $DN_t$, and attempts to find support for one of its possible values. Support for a possible value is attempted through all applicable synchronizations in the domain theory (line 3), and only sets of supporting decisions in the DN that contain at least one active decision are considered (lines 5–7). The procedure terminates either when support is found or all synchronizations have been attempted. Note that the `ActivityRecognition` procedure is greedy, in that the first acceptable hypothesis is selected in support of current sensor readings. Also note that in line 10 the DN is incrementally updated when a hypothesis is confirmed.

**Theorem 1 (Completeness).** *The* `ActivityRecognition` *procedure is complete under the assumption that* `ActivityRecognition`$(x, DN_{t'})$ *always follows* $Sense_Y(DN_t, t')$*, where* $Y$ *is a set of sensors that are updated at time* $t'$ *. In other words, recognition is carried out every time new sensory information is obtained.*

*Proof.* We are guaranteed that if a set $\alpha$ of decisions cannot support a hypothesis $d^x_{hyp}$ at time $t$, this set need only be attempted in subsequent calls to `ActivityRecognition` as a subset of a support set $\alpha \cup \alpha'$ where $\alpha'$ contains at least one active decision. A decision $d^y$ on a sensory component $y$ can become inactive only as a consequence of the sensing procedure $Sense_y(DN_t, t')$. Also, when a sensor signals a new sensed value, this is modeled as a decision with unbounded end-time. If the `ActivityRecognition` procedure is always applied after a sensor update, we are thus guaranteed that no sensory decision will become inactive without having previously been employed in an attempt to support a hypothesis. In other words, every set of decisions containing at least one active decision will be considered for support, thus proving completeness. □

The added requirement that states that at least one of the target decisions of a unification should be active increases the performance during long term monitoring since the bulk of the decisions will be inactive (i.e., the number of active decisions is bounded by the number of sensors). For example, consider a synchronization that requires two decisions $\mathbf{A}$ and $\mathbf{B}$, and that there are seven inactive and three active decisions with the value $\mathbf{A}$, and five inactive and two active decisions with the value $\mathbf{B}$. The number of sets of decisions that contain at least one active decision are therefore $(3 \times 2) + (3 \times 5) + (7 \times 2) = 35$, while a naïve approach would in this case have to attempt $(7 + 3) * (5 + 2) = 70$ combinations.

More in general, we can quantify the amount of pruning that is achieved as follows.

**Theorem 2.** *Given a synchronization* $\langle\langle \mathbf{v_{ref}}, x\rangle, \mathcal{R}\rangle$*, let* $DN_t$ *be a decision network containing* $m$ *decisions that unify with each target value* $\mathbf{v_i}$ *in* $\mathcal{R}$*. The cost of searching for a set* $\alpha$ *such that* $Unifies(\alpha, \mathcal{R})$ *and* $Support(DN_t, d^x_{hyp}, \alpha)$ *holds (lines 4–11 in the* `ActivityRecognition` *procedure) is* $O(m^{|\mathcal{R}|-1})$.

*Proof.* If we do not prune sets of decisions that are all inactive, a synchronization will require

$$\prod_{i=1}^{|\mathcal{R}|}(I_i + A_i)$$

combinations to be tried, where $|\mathcal{R}|$ is the number of requirements for the synchronization, and $I_i$ and $A_i$ are, respectively, the number of inactive and active decisions that the $i$-th target value in $\mathcal{R}$ can unify against. Conversely, if sets of all inactive decisions are discarded, it is only necessary to attempt

$$\prod_{i=1}^{|\mathcal{R}|}(I_i + A_i) - \prod_{i=1}^{|\mathcal{R}|}(I_i)$$

different sets of decisions. Due to the binomial theorem, and assuming without loss of generality that each sensory state variable has an equal number of active and inactive decisions that unify with each target value, i.e., $I_i = I$ and $A_i = A$, we obtain:

$$\begin{aligned}(I + A)^{|\mathcal{R}|} - I^{|\mathcal{R}|} &= \sum_{i=0}^{|\mathcal{R}|}\binom{|\mathcal{R}|}{i} I^i A^{|\mathcal{R}|-i} - I^{|\mathcal{R}|} \\ &= \sum_{i=0}^{|\mathcal{R}|-1}\binom{|\mathcal{R}|}{i} I^i A^{|\mathcal{R}|-i}\end{aligned}$$

where the maximum power of $I$ is $|\mathcal{R}| - 1$, thus decreasing the cost per synchronization by one order of magnitude. □

## Maintaining Future Consistency

Consistency problems arise when a set of decisions $\alpha$ can act as support for a decision $d$ at time $t$ but not in all future temporal evolutions of the DN. This is clearly never the case if $\alpha$ contains only inactive decisions. However, as shown earlier, the inference procedure will support a hypothesis with a set that includes at least one active decision. This implies that support may cease to exist in future evolutions of the DN, i.e., there may exist $\alpha$ such that $Support(DN_t, d^x_{hyp}, \alpha)$ is true and $Support(Sense_y(DN_t, t'), d^x_{hyp}, \alpha)$ is false for some sensor $y$ at time $t' > t$. An example of such a situation can be seen in Figure 4 (top), which represents the situation at $t = 40$. Here, the decision $d^{\text{Human}}$ with value **Sleeping** is supported by $\alpha = \{\langle \mathbf{Occupied}, [[20, 20], [40, \infty)]\rangle, \langle \mathbf{Off}, [[10, 10], [40, \infty)]\rangle\}$ so that **Sleeping** is required to occur during a time span which EQUALS the one of **Occupied** and DURING **Off**. If the temporal evolution of **Off** and **Occupied** are as indicated by their dashed extensions in the figure, these requirements cease to hold when $t > 50$. On the other hand, if the evolution of sensor readings proceeds as shown in Figure 4 (bottom), then the hypothesis of **Sleeping** deduced by the the inference procedure will hold. In general, the inference procedure as we have described it so far is "optimistic", as it recognizes activities as soon as their requirements are found to hold, and ignores the possibility that supported hypotheses will not hold in the future due to "unexpected" evolutions of sensor readings.

The root of the problem lies in the fact that the temporal constraints imposed to support a hypothesis may introduce indirect constraints between decisions representing sensor readings. As shown in Figure 2, the DN resulting from the application of $Support(DN_{40}, \langle \mathbf{Sleeping}, [[0, \infty), [0, \infty)]\rangle, \alpha)$ where $\alpha$ contains $\langle \mathbf{Occupied}, [[20, 20], [40, \infty)]\rangle$ and $\langle \mathbf{Off}, [[10, 10], [40, \infty)]\rangle$, introduces a dependency between the two sensed decisions involving the reference decision on the Human state variable. This can be appreciated by noting that if the end time of the **Occupied** decision were constrained to $[50, \infty)$, then by temporal propagation the end time of the **Off** decision would be also constrained to $[50, \infty)$. This is clearly unacceptable, as start and end times

of decision representing sensor readings should only be affected by the variations of the physical sensor readings. In other words, it should never be the case that updating one sensor reading affects another sensor reading.

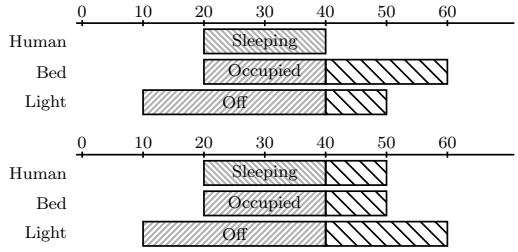


Figure 4: Recognition of an activity at time 40 resulting in an inconsistent (top) and consistent (bottom) future scenario.

Identifying such indirect dependencies among sensor readings constitutes a sufficient condition for determining whether support for a hypothesis is guaranteed to persist in the future. In other terms, assuming $Support$ is correct, the overall iterative process shown in procedure `ActivityRecognition` is correct as long as the following conditions for future consistency of recognized hypotheses are applied.

**Theorem 3 (Correctness).** *Let $d^x_{hyp}$ be a hypothesis such that $Support(DN_t, d^x_{hyp}, \alpha)$ holds. The following conditions are sufficient for guaranteeing that $Support(DN_{t'}, d^x_{hyp}, \alpha)$ will hold for every $t' > t$:*

- *for every pair $d_i, d_j \in \alpha$, imposing the constraint $d_i$ DEADLINE $[l_i, l_i]$ does not change the bounds of the end time of $d_j$, where $l_i$ is the lower bound of the end time of $d_i$;*
- *for every pair $d_i, d_j \in \alpha$, imposing the constraint $d_i$ DEADLINE $[u_i, u_i]$ does not change the bounds of the end time of $d_j$, where $u_i$ is the upper bound of the end time of $d_i$.*

*Proof.* Given that the underlying temporal problem is a Simple Temporal Problem (as all constraints represent simple intervals defining the distance between decisions' start and end times), the temporal relations induced by the network on any two time points can be modeled as a simple distance constraint between the two time points (Dechter, Meiri, and Pearl 1991). Assume that at time $t$ we consider two decisions $d_i$ and $d_j$ of the support set $\alpha$, and that the constraint induced by the temporal network on the end times $t^i_e, t^j_e$ of these decisions imposes $t^j_e - t^i_e \geq l$ and $t^j_e - t^i_e \leq u$. We at this point attempt two tests. First, we constrain the interval of admissibility of $t^i_e$ to its upper bound. The imposition of this further constraint may or may not affect the lower bound of $t^j_e$. Whether it does so depends on the positive slack $l$ allowed by the induced constraint. Second, we constrain the interval of admissibility of $t^i_e$ to its lower bound, in which case the slack allowance $u$ will determine whether this further constraint affects the upper bound of $t^j_e$. It is easy to see that the forward and backward slack allowed by network, i.e., $l$ and $u$, is maximally exhausted by imposing one of the two constraints above on the interval of admissibility of $t^i_e$, and that no other constraint on $t^i_e$ has more power to reduce the interval of admissibility of $t^j_e$.

The above observation, together with the fact that the $Sense$ function will only constrain the end times of the decisions in $\alpha$ more as time goes by, proves that the above tests are sufficient for guaranteeing that sensor decisions in a support set $\alpha$ can end at any time in the future without introducing inconsistencies. In essence, testing the impact of constraining end times of decisions in $\alpha$ on the end times of each other decision in $\alpha$ as described above will expose any indirect dependencies introduced by activity recognition. □

The previous theorem is leveraged in our framework by following each successful application of the $Support$ procedure (line 11 in `ActivityRecognition`) with a test ascertaining whether the above sufficient condition holds. If the added hypothesis together with its supporting set fails the sufficient condition, the results of activity recognition are discarded, thus guaranteeing that states that may not be supported in the future are never committed to.

## Evaluation

The `ActivityRecognition` procedure provides a means to achieve the required performance as it reduces the cost of supporting a hypothesis with a synchronization from $O(m^{|\mathcal{R}|})$ to $O(m^{|\mathcal{R}|-1})$. This effectively means that finding support for a decision through a synchronization with two requirements can be done in linear time with respect to the number of applicable target decisions in the DN.

In order to assess whether the performance increase obtained as a result of the admissible pruning can support long-term monitoring scenarios, we compare the performance of two implementations of our system, one employing no optimization, and one which prunes the search space as shown earlier. All tests described in this section were carried out on an Intel Core2 Duo processor @ 2.33 GHz.

First, we experimentally verify the complexity bounds shown earlier by performing two tests with a domain theory containing only one synchronization. In the first test, the domain contains one synchronization stating that a value $\mathbf{v_{ref}}$ should be recognized if it occurs DURING value $\mathbf{A}$ (on one sensory state variable) and should CONTAIN value $\mathbf{B}$ (on another sensory state variable). The sensory input shown in Figure 6 (top) was fed to the systems over a period of 200 seconds. Notice that the number of combinations of support decisions that need to be explored – i.e., the number of sets $\alpha$ used to attempt $Support(DN_t, \langle \mathbf{v_{ref}}, [0,\infty)[0,\infty) \rangle, \alpha)$ – constantly increases over time. Also, notice that the sensor readings occur in a repeating pattern such that no combination of targets can act as support for $\mathbf{v_{ref}}$. This situation represents the worst case, as all combinations of supporting decisions must be attempted in order to conclude that $\mathbf{v_{ref}}$ cannot be supported. Figure 5 (top) compares the CPU time required by the `ActivityRecognition` procedure in the two systems. As shown, when pruning is employed, the performance of the system grows linearly with the number of sensory events, while in the absence of pruning we obtain a quadratic increase in CPU time.

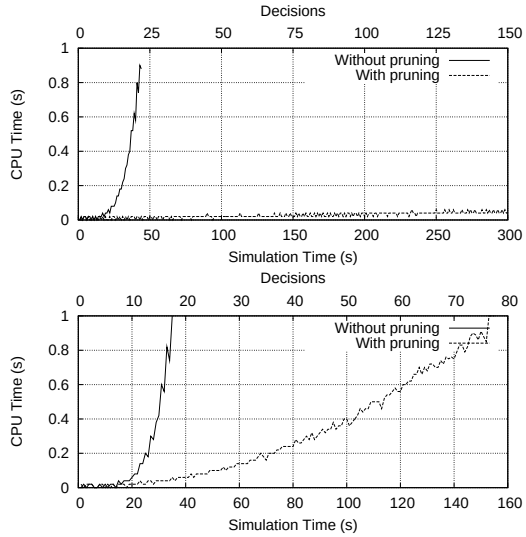


Figure 5: CPU time required to prove lack of support for a hypothesis using a synchronization with two (top), and three (bottom) requirements.

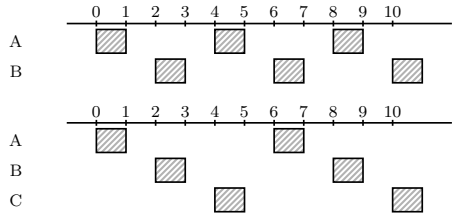


Figure 6: Recurring patterns used in the performance tests in Figure 5 (top and bottom, respectively).

Figure 5 (bottom) shows the second test, where a similar synchronization that has three requirements was used. As for the first test, the input for the second test (Figure 6, bottom) feeds sensor readings that never support the reference value $\mathbf{v_{ref}}$, thus yielding a quadratic increase in CPU time with pruning, as opposed to cubic complexity without pruning.

Our last experiment aims to assess how the system performs in a more realistic scenario, where a domain theory containing ten synchronizations models meaningful activities of daily living. A similar scenario is described more extensively in (Cirillo et al. 2009), where an experimental run in a real sensor-rich environment with a human test subject is described. Suffice it to say here that these activities, each depending on at most $|\mathcal{R}| = 3$ sensor readings, include the previously described **Sleeping** activity, as well as several other activities such as **Cooking**, **WatchingTV** and **HavingLunch**. Six sensor state variables were employed, and the sensory input was modeled in such a way that it would constitute feasible input from a real world scenario (e.g., the location state variable providing the position of the human being was fed realistic movements of a human being in a topologically correct model of a small apartment).

Figure 7 shows the performance of the system obtained over a monitoring horizon of one week. The test serves as

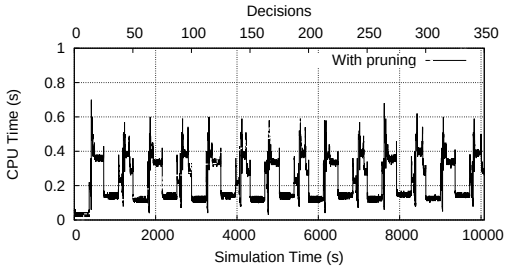


Figure 7: CPU time required to recognize human activities once each minute during a one-week long scenario.

an experimental proof of the fact that it is possible to use the current system to recognize activities over periods as long as one week. Activity recognition and sensing occurs once per minute (0.02 Hz). The criteria for choosing this rate is related to the resolution that is necessary in the specific monitoring scenario. Clearly, this rate has to be low enough to allow for the `ActivityRecognition` process to terminate, but also high enough to guarantee that meaningful variations of sensor readings are detected. In the present test, we assume that variations of sensor readings within one minute is sufficient for detecting all meaningful activities. However, notice that it would have been possible to continuously recognize activities at a frequency of approximately 1.4 Hz throughout the entire scenario, as the maximum CPU time of the `ActivityRecognition` procedure is about 0.7 seconds.

At the end of the week, the DN contained close to 350 decisions (a combination of sensor readings and the recognized activities), with an average production of approximately 50 decisions per day. As can be seen, the system scales well with the growing number of decisions, which indicates that recognizing activities at a greater level of detail (e.g., activities with a shorter time-span that occur more often, such as relocating etc.) is not believed to cause any performance problems as long as the number of decisions are reasonable (e.g., one can assume that the current system can detect activities with a similar level of detail during ten weeks while maintaining an adequate performance for most domestic applications, but not necessarily for an entire year).

## Temporal Propagation

While Theorem 2 shows that pruning reduces the cost of the `ActivityRecognition` procedure by one order of magnitude, the existence of inactive decisions in the DN brings with it another advantage which can be leveraged to further increase performance. Specifically, temporal propagation occurs every time $Support(DN_t, d^x_{hyp}, \alpha)$ is evaluated. This incurs a cost which is polynomial in the number $n$ of decisions in $DN_t$. Our specific implementation of the underlying temporal propagation is based on the Floyd-Warshall algorithm, whose computational cost is $O(n^3)$ (Dechter, Meiri, and Pearl 1991). Notice, however, that inactive decisions are not flexible in time, and thus do not need to partake in temporal constraint propagation. This is achieved in our implementation by periodically removing

pairs of timepoints corresponding to inactive decisions from the underlying temporal network. Any effect these fixed timepoints have on other (flexible) timepoints is modeled as constraints on these timepoints, thus effectively reducing the number of timepoints over which the Floyd-Warshall propagation procedure iterates (two for every active decision).

Although the exclusion of inactive decisions from temporal constraint propagation alone cannot provide a significant performance increase, it can be leveraged to increase the performance of the system when pruning is done. In fact, notice that in the worst case, one synchronization requires $m^{|\mathcal{R}|-1}$ propagations, each with a cost of $n^3$ (due to Floyd-Warshall's temporal propagation). We would therefore incur in cost $m^{|\mathcal{R}|-1} \cdot n^3$ for each synchronization. Again under the assumption that there are an equal number $m$ of decisions that unify with each target value in $\mathcal{R}$, the DN contains $|\mathcal{R}| \cdot m$ decisions. As a consequence, the cost per synchronization would be $O(m^{|\mathcal{R}|+2})$ if inflexible timepoints were never excluded from propagation. Conversely, by excluding inactive decisions from temporal propagation our system periodically curtails this computational load, therefore guaranteeing a cost of $O(m^{|\mathcal{R}|-1})$.

## Conclusions

The ability to perform long term human activity recognition is fundamental as applications move towards real-world domains. Solutions for knowledge driven approaches that allow temporal reasoning need to be tractable over long monitoring horizons as well as ensure future temporal consistency of recognized activities. This paper has investigated these two specific aspects in relation to a temporal reasoning approach. The main contribution has been to present a technique for pruning search in the space of supporting sensor readings which leverages the flexible bounds of sensor readings in the decision network. This process was shown to improve performance over a naïve approach which disregards the temporal flexibility of decisions. The improvement was twofold: on one hand, less combinations of sensor readings need to be explored to support a hypothesis; on the other hand, inactive decision are excluded from temporal propagation, thus ensuring a constant sized temporal network.

Ensuring future consistency as we have described in this paper inevitably affects the ability to recognize activities as they happen. However, this does not impact activity recognition more than necessary, as only certain combinations of constraints are filtered. For instance, a synchronization stating DURING together with CONTAINS will be considered "safe" once the contained sensor reading has become inactive. As for other limitations of our system, further analysis is needed.

As the performance of the system depends on STP propagation cost as well as the number of support sets to be attempted, future work will consider different algorithms to solve the STP in order to fully evaluate the performance gain of the current approach (e.g., (Xu and Choueiry 2003)). Other possible directions for future investigations include integrating our approach with data-driven methods in order deal with sensor noise and possibly provide on-line model training and adaptation.

**Acknowledgments.** The Authors wish to thank Alessandro Saffiotti and Marcello Cirillo for their support and insightful comments as well as the anonymous reviewers for their helpful observations.

## References

Allen, J. 1984. Towards a general theory of action and time. *Artificial Intelligence* 23(2):123–154.

Cirillo, M.; Lanzellotto, F.; Pecora, F.; and Saffiotti, A. 2009. Monitoring Domestic Activities with Temporal Constraints and Components. In *5th Int. Conf. on Intelligent Environments*.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal constraint networks. *Artif. Intell.* 49(1-3):61–95.

Dousson, C., and Maigat, P. L. 2007. Chronicle recognition improvement using temporal focusing and hierarchization. In *Proc. IJCAI'07*, 324–329.

Fratini, S.; Pecora, F.; and Cesta, A. 2008. Unifying Planning and Scheduling as Timelines in a Component-Based Perspective. *Archives of Control Sciences* 18(2):231–271.

Györbíró, N.; Fábián, Á.; and Hományi, G. 2009. An activity recognition system for mobile phones. *Mob. Netw. Appl.* 14(1):82–91.

Jakkula, V.; Cook, D.; and Crandall, A. 2007. Temporal pattern discovery for anomaly detection in a smart home. In *Proc. 3rd IET Conf. on Intelligent Environments*, 339–345.

Pecora, F., and Cirillo, M. 2009. A Constraint-Based Approach for Plan Management in Intelligent Environments. In *Proc. of the Scheduling and Planning Applications Workshop at ICAPS09*.

Sanchez, D.; Tentori, M.; and Favela, J. 2007. Hidden markov models for activity recognition in ambient intelligence environments. In *Proc. of the 8th Mexican Int. Conf. on Current Trends in Computer Science*, 33–40. Washington, DC, USA: IEEE Computer Society.

Vilain, M.; Kautz, H.; and van Beek, P. 1989. Constraint propagation algorithms for temporal reasoning: A revised report. In Weld, D., and de Kleer, J., eds., *Readings in Qualitative Reasoning about Physical Systems*, 373–381. Morgan Kaufmann.

Xu, L., and Choueiry, B. 2003. A New Efficient Algorithm for Solving the Simple Temporal Problem. In *Proc. of Int. Syp. on Temporal Representation and Reasoning*.

Zouba, N.; Bremond, F.; and Thonnat, M. 2009. Multisensor fusion for monitoring elderly activities at home. In *IEEE Int. Conf. on Advanced Video and Signal based Surveillance*.