# Temporal Representation of Ecological Knowledge

Edjard de Souza Mota

MSc Information Technology: Knowledge Based Systems

Department of Artificial Intelligence
University of Edinburgh

September-1994

## Abstract

This MSc thesis proposes a temporal logic to represent knowledge about seasonal cycles in ecosystems. The logic is mainly based on what we call *modular temporal classes*, and a simple temporal logic interpreter system is also defined and implemented to reason with ecological sentences expressed in a temporal language we call **NatureTime**.

The **NatureTime** system was tested with some ecological examples from the Indigenous Knowledge project. This project concerns with the construction of a knowledge based toolkit to represent and reason with ecological and agroforestry knowledge in developing countries. This thesis shows how useful can be a temporal reasoner system like the **NatureTime**, to be integrated in the Indigenous Knowledge project for making predictions, and detecting temporal conflicts.

## Acknowledgements

I would like to thank my supervisors Mandy Haggith for her confidence and guidance, mainly through the analysis of the ecological examples, and Alan Smaill for his gentle support in help me with formal definitions, and Dave Robertson for his support and helpful criticisms in the system construction.

Thanks to Wamberto Vasconcelos that has been supporting me since I arrived in Edinburgh, and now for his help this project.

Thanks to my beloved wife Cirlene for her unmeasurable support and presence.

Thanks to my parents and brothers for their support, even so far, but so close.

Thanks to the Brazilians that live in Edinburgh that has been supporting us, to not miss so much the distance of our country, mainly to Alberto and Goretti.

Thanks to fellow MSc students for many moments of fun, specially to Min-in Chen and Hugo Terashima.

Finally, thanks to CAPES(the Brazilian Ministry of Education Agency) who provided financial support during my year of study via studentship n ọ 01723/93-8.

# Table of Contents

# List of Figures

# Chapter 1

# Introduction

This thesis describes a temporal logic to represent knowledge about seasonal cycles in ecosystems. A simple temporal logic interpreter is also defined and implemented to reason with ecological sentences expressed in a language that we call **NatureTime**(NT), because the theory in which it is based is an attempting to model aspects of temporal behaviour which are typically expressed by ecologists in natural language.

## 1.1 Related Works

To our knowledge, the only attempt to model temporal aspects of ecological phenomena in terms of temporal logic, was proposed in [Gray 93], using a linear time representation. Although this work provides a useful analysis of the problems related with temporal representation of ecological knowledge, the idea behind the traditional linear point structure of time that he used is to represent the day-by-day flow of time as the axis of the positive integers, as shown in figure 1–1, where each point correspond to exactly one day. In this framework of time, the information "March, 10th of 1990" is represented by the number 69. He reduced the set of integers for a set of 3650 points to represent 10 years, from 1990 up to 2000. As pointed out in his conclusions, the linear time representation can lead to some problems in representing the circularity of the months and seasons. We add that there is a lack of isomorphism between Gray's representation, $\{1, 2, \ldots\}$ as flow of time, and the way in which time is commonly described in the ecological domain. As ecologists normally make reference to seasons, using natural language expressions that have a cyclical meaning, we decided not to base our work on that of Gray.

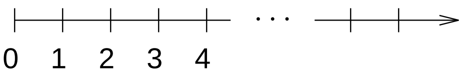


**Figure 1–1:** Traditional linear point structure representation of time.

This MSc project had to evaluate the usefulness of including a specialist temporal reasoner for making predictions, and detecting temporal conflicts, to be used as a tool in the Indigenous Knowledge (IK) project. The IK project is concerned with the construction of a knowledge based toolkit to represent and reason with ecological and agroforestry knowledge in developing countries [Haggith *et al.* 92]. As we will see, this thesis shows some of the temporal contents of ecological knowledge using some examples from the IK project, and we also propose a simple temporal language that seems to be very useful not only for the ecological domain investigated.

## 1.2 Time Representation: Perspective

There are three categories of time conception found in philosophical and psychological literature as pointed out in [Koomen 89]:

1. *natural* - is based on natural phenomena that occur in the natural world. For instance, days, lunar month, and seasons.

2. *conventional* - is based on a time structure of social events. For instance, seconds, minutes, that can also use some natural conventions, as months and years.

3. *logical* - is based on a logico-mathematical structure with operations of ordering and transitivity.

While no computational theory or practical A.I. system has been developed based on the first, the third has been the most researched and many theories have been proposed to create a suitable framework of time, based on a linear point structure, with the emphasis on applications in problem solving and planning.

Once we have chosen which category of time we want to work on, we have to choose how to present our logic, and the way of reasoning about time and changes through time, in our logic.

## 1.3 Temporal Logic: Perspectives

According to [Gabbay 92], temporal logics can be presented in the following ways

1. by predicate calculus, adding one parameter for time
2. by special temporal logics, where two methods can be used
   (a) semantic presentation
   (b) axiomatic presentation or other deductive system for the connectives
3. by direct reference to events.

The present work proposes one temporal logic that falls in category 2.(a), but without a deep mathematical investigation of how to relate our logic with others formalized, mainly because the time we have for this project did not permit this kind of investigation.

In relation to the view of reasoning about time and changes through time that our logic will be concerned with, [Gabbay 92] points out that it can be, in terms of modal and temporal logic,

1. Declarative, where basically we have a database $\Delta$, and for any query $Q$ we are interested to know whether $\Delta$ and $Q$ satisfies the relation $\Delta \vdash Q$, in the appropriate logic, *i.e.* can $Q$ be deduced from $\Delta$?
2. Procedural (or Modal and Temporal Logic Programming), is concerned with a procedural reading of the syntatical structure of the database to get an answer

3. Imperative, where the idea is that we are "inside" the flow of time in which the future has not yet happened. In this way, a sentence is read imperatively, that is create a future state in which the statement is true.

According to the context of the present work, we should develop a system in which we we are much more interested in looking at the time as if the *past, present* and *future* are all "frozen", in the sense that we are externally observing the flow of time, and any point can be the present point So the declarative view seems the most suitable for our purposes.

## 1.4 Which Logic To Use?

In the early phase of this project we initially intended to use Gabbay's temporal logic [Gabbay 87], an extension of temporal logic programming to allow modal and temporal connectives, that is based on a linear point structure of time. The linear point structure could lead us to the same problems as in the Grays's attempt.

Another possibility could be the temporal logic with *Since* and *Until* and fixed point operators [Gabbay 89], [Barringer *et al.* 91]. However, as pointed by the authors "the logic is based on the natural numbers as flow of time" (linear structure), and is suitable for controlling process behaviour. Consequently, such logic is not suitable for our purposes as mentioned in section 1.3.

The logic based on intervals proposed in [Allen 84], and axiomatized in [Allen & Hayes 85] could be used. The problem related with cyclical or recurrent events could be partially solved using the results of [Koomen 89], that proposed a theory of how to represent recurrences in a time logic of intervals by using complicated concepts of classes of intervals such as *downward and upward hereditary*, *liquid, concatenable and solid*. This treatment creates a hierarchical view of linear time intervals similar to that shown in figure 1–2, where each $e^i_j$ is a representation for one interval of the level $i-1$. In this way we could model *week* as an interval of seven *days*, and so on. For instance, if we treat the points

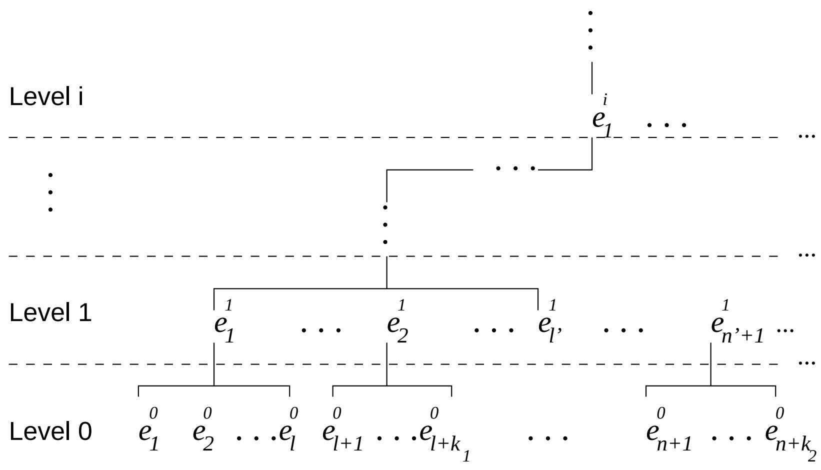


**Figure 1–2:** Linear hierarchy of time representation.

of the level 0 as day, than each point of the level 1 is a reference for an interval of days, that can be weeks, or months.

Despite the fact that this theory seems to allow a considerable number of properties for this kind of interval, and provides mechanisms to infer relational constraints between intervals, it is not clear how to use such mechanisms in practice to model cyclical events in a "natural" way and obtain appropriate inferences.

Recently, [Knight & MA 94] pointed that for many artificial intelligence applications, the linear point structure representation is inadequate to express temporal facts and the storage and organisation of incomplete temporal knowledge. Also, Allen's system lacks comprehensive mechanisms to deal appropriately with duration reasoning.

As none of the time representations looked at would satisfy our needs, we needed to develop a different approach.

## 1.5 A Natural Time Logic Approach

From the beginning of this project, we had thought of a model of time in which we could define cyclical temporal classes in terms of natural events such as *day* that represents one sunrise to another, *month* that represents some features of lunar cycle in a given number of days, and so on. These examples are inherently cyclical temporal classes, and one may be *circularly inside* another forming a hierarchy which we call a *hierarchy of cycles*. Of course there are some natural events that are not cyclical. For example, life is one phenomenon that starts with one event (birth) and finishes with another (death), and there is no such concept as "two lives" for the same being, unless we extend such concepts for *spiritual* life using meta-physical concepts like *metempsychosis*, or better still *reincarnation*, that are outside the scope of this work.

Which mathematical framework could be used to model temporal classes like those above? The framework of time we suggest is to model such a *hierarchy of cycles* using modular arithmetic, which is also called clock arithmetic [Biggs 87]. In this view, each cyclical temporal class is defined by one modular set, and so we permit the succession of time in a cyclical way, where the last element of the set is followed by the first.

One question still remains. Given that we need to propose and experiment with a new temporal logic, what are the features our logic must have to be considered a temporal logic? [Gabbay 92] says that a logical system is considered temporal if it is a structure with a well defined framework of time, $T$, an irreflexive and transitive ordering relation $<$ (notion of "later than") and a successor operator $+$. Our logic possesses all of these with the difference that as we will have modular temporal classes, to represent temporal entities that are inherently cyclical, then instead of a structure like $(T, <, +)$, we will extend it to support circular ordering in the levels of time hierarchy, and the flow of time will be defined only in the one level that embodies the flow of temporal classes in the others, *i.e.* the top level.

The main content of this thesis is in two parts: first is the definition of a very simple logico-mathematical model of natural time; second is an experiment to use such a logico-mathematical approach, and implement a simple temporal reasoner interpreter based on it. This is a standard Prolog meta-interpreter with a restricted form of unification on temporal labels, or types. The interpreter was tested on several ecological examples which demonstrate some of the temporal reasoning required in the Indigenous Knowledge project.

## 1.6 Thesis Organization

The rest of this thesis is organized in the following way.

In chapter 2 we will analyse some ecological examples, to show the necessity of a temporal reasoner system. The analysis of which will help us to find out the main features desired for such a system. In chapter 3 we shall define our temporal ontology of time to allow the concept of cyclical temporal classes. In chapter 4, we shall define the syntax of our temporal logic, and as we do not intend to formally present its semantics, we also shall present the specification of the **NatureTime** system that was implemented (its Prolog implementation is in the appendix of this work) to test our approach. In chapter 5 we will show how to use the language defined in chapter 3 to represent the examples of chapter 2, and obtain temporal reasoning with the **NatureTime**. We also show some analysis of the results obtained. Finally in chapter 6 we will present the conclusions of the work, and present some possible future extensions.

# Chapter 2

# The Temporal Content of Ecological Knowledge

## 2.1 Guidelines of Investigation

In this chapter we shall investigate the nature of temporal information in the scope of our ecological domain, in order to guide our search for a suitable representation. The expected conclusions will help us to find a formalism to represent the temporal statements. The main goal is to have an idea of the behaviour that a temporal reasoner should use to answer the types of temporal queries that can be made about this subject.

To achieve the purpose of this chapter, we shall discuss some simple ecological examples, their presentation will be in the order of difficulty of representation and inference. Although the examples to be shown are not the most complex in terms of temporal information, they have the advantage of being computationally tractable, and the temporal features of their contents are found in a large number of other ecological examples.

In the investigation of temporal knowledge, we will simplify everything related to processes by describing them as the pair of events that start and finish one process, and the period of time between them.

## 2.2 Ecological Examples

### 2.2.1 Planting & Harvesting

This example was extracted from [Gray 93]. It originates from knowledge acquired from famers in Nepal, in the IK project.

> **Example 2.1**: Maize takes about 3 1/2, four months to mature. It is planted around June, July. It depends on the variety of maize. In the high lands it takes about 6 months, because of the temperature. There are varieties which are grown in the lower areas, where temperature is quite high, so in the lower areas just two months is enough. It depends on altitude and crop variety. But in the mid hills it normally takes 3 1/2 to 4 months from planting to harvesting. When maize is about 2 1/2 or 3 months after sowing, then at this time the farmers start planting millet. Millet takes about 3 months to mature.

In this example the term *mature* refers to a process, so we shall not use it, but the events that mark its beginning (plant), and we can assume its end (harvest), although one crop can be harvested without stop the process of maturation. Furthermore, we shall use the terms *planted* and *harvested*, instead of planting and harvesting.

#### Analysis

The kind of temporal questions that one could ask about this example are related to:

1. the time between planting and harvesting (the duration of the maturation of) maize in general;

2. the time between planting and harvesting (the duration of the maturation of) maize in a particular area;

3. the months of maize harvesting, in general ;

4. the months of maize harvesting, in a particular area;

5. the months in which we should start planting millet;

6. the months in which we should harvest millet.

The first two questions can be easily solved by a simple query, but the others clearly need some temporal reasoning. Lets see now what kind of temporal reasoning would be necessary to answer those questions.

- The information available for 3 is the time $t$ (the interval from June to July) in which maize is planted, and the period of time $\Delta$ (3 months and half, 4 months in general). The rule is quite simple: given $t$ and $\Delta$ then infer $t + \Delta$. In this example, since we have two possible initial times we may conclude

  for $t$ = June then maize can be harvested, in general, from the middle of September till the middle of October, or during the whole month of October

  for $t$ = July then maize can be harvested, in general, from the middle of October till the middle of November, or during the whole month of November.

  optionally we could conclude that maize can be harvested from the middle of September up to November.

  The rule needed involves the temporal concept of after, which could be used to informally describe the rule as follows

  *maize can be harvested 3 1/2 or 4 months* after *the time it was planted.*

  The chain of reasoning should be

  *Since maize can be planted around June,July, and 3 1/2 or 4 months after that it can be harvested, then maize can be harvested from the middle of September up to November.*

- The information available for 4 is almost the same as for 3. Now, the difference is that we must name a specific place, which affects and the period between the time it can be planted and harvested. So, the conclusion should be, for example, *Since maize is planted in the high lands in June/July, and 6 months after that it can be harvested, then maize can be harvested in the high lands from December up to January.*

- The information available for 5 is quite complex because there is no direct information about the time of planting millet. However, this information is hidden in one sentence that could be expressed as

  *Millet can be planted by the farmers 2 1/2 or 3 months* after *the time maize was planted.*

  Now, we can use the knowledge about the time of planting maize to infer the time of planting millet.

  *Since millet can be planted 2 1/2 or 3 months* after *the maize was planted, and maize was planted in June/July, then millet can be planted from middle of August up to September.*

- The information available for 6 needs to be derived from 5, with the information about the period of time between when millet can be planted and harvested.

  *Millet can be harvested 3 months* after *the time it was planted.*

  Now, the process of temporal reasoning should first deduce the time of planting millet (by the reasoning used in 5), and then apply the same process of reasoning used for 1.

  *Since millet can be planted from the middle of August up to September, then millet can be harvested from the middle of November up to December.*

This example is about the seasons that one crop (maize and millet), are planted and harvested, and any seasonal event is related with intervals of time (*e.g.* from September up to October). One interpretation of season is that the event related with the season (in this case plant and harvest) happens throughout the temporal interval of that season. Then, *maize can be planted from June up to July* can be expressed as *maize can be planted* throughout *the interval from June to July.*

Now, $i$) what do we mean by *maize can be planted* **throughout** *the interval from June to July*? $ii$) What does the deduction *maize can be harvested 6 months* **after** *the interval from June to July* (that is **throughout** the interval from December up to January) mean? The interpretation of **throughout** deserves very careful attention because it may imply the notion of a continuous occurrence of one event, but maybe not. For example, the answer for the question "Can maize be planted throughout June?", is yes because June is included in the interval from June to July. If we ask "Can maize be planted on June 24th?", the answer should not necessarily be yes, although a month is formed by a sequence of days. Also, the question "Should maize be planted from 0 clock to 24 clock of any day of June?" should not be true since the activity of planting is not performed during the night (at least for this case).

In the present work, if we say that one event $e$ happens throughout one interval $I$ of a given measure of time, then the event should happen in every subinterval $I'$ of another temporal measure that directly defines the interval of $I$. In this way, the **throughout** has the same meaning of the modal connective "always' (usually represented as $\Box$). So $e$ **throughout** $I$ means *for every subinterval $I'$ in $I$ $e$* **throughout** $I'$.

Considering it, in $i$) we mean that maize can be harvested **throughout** any sub-interval $T_1$ of June up to July, *e.g.* June, July or 28th of July. $ii$) we mean that maize is harvested **throughout** any sub-interval $T_2$ of December up to January, *e.g.* 20th December, or from 3rd up to 12th of January. The relation between $T_1$ and $T_2$ is that if one particular crop of maize is planted **throughout** $T_1$, and it is harvested **throughout** $T_2$, then the period of time between $T_1$ and $T_2$ must be exactly 6 months, for any $T_1$ inside June up to July.

Another way to interpret *maize can be planted from June up to July* is *maize can be planted* **sometime** *the interval from June up to July*. Here, instead of "always", the analogy with modal logic is the connective "sometime" ($\Diamond$), where the meaning for $e$ **sometime** $I$ is *there is one sub-interval $I'$ of $I$, then $e$ occur at $I'$*.

For reasons of time, we will work out just with the *throughout* connective.

Another aspect of this example that seems to be essential in making temporal inferences is a period of time. By period of time we mean the length of of time that can be related with any interval. Such period is to be used when we want to refer to one process by the pair of events that start and finish it, with this type of period between these events. In this way, the **after** relationship between the season in which maize is planted and a period of time will produce another interval of time, that is the season in which maize is harvested.

There is one more interesting concept in this example that deserves our attention. In the sentence *It is planted around June, July*, which we had assumed to be the two months, the term *around* expresses vague information about one interval. It could be that we had **around** *June*. How should our temporal reasoner represent and interpret such a sentence? Despite the fact that in the original sentence *June,July* is an explicit interval, and in the other we have only June, we can assume that the closest months of June are May and July, and so **around** *June* means the interval from May up to July. But we could also have another period different of one month. For instance, *3 months around June*, that is equivalent to *from March up to September*.

### Conclusion

We have identified the following temporal elements

- the temporal units **interval** of time to represent the time in which events can happen, and **period** of time, the length of time between any pair of events that represent some process;
- the temporal connective **throughout**, and **sometime**, not to be used in this work;;
- the temporal unit operators
  - **after** which associates one period of time and one interval (or moment, if wanted) with another interval in the future, where the period is the time between the two intervals; and
  - **around** which associates one period of time and one interval (or moment, if wanted) with another interval of time that include the first.

Along with such identification we noted that we need a model of time that provides a hierarchical way to talk about temporal units. We mean that we need some way to say that one measure of time is included in other in a cyclical way. For instance, one year is a complete cycle of 12 months, a month is (generally) a complete cycle of at least 28 days, and so on. It suggests another view of time rather than point structure, or interval structure, only.

### 2.2.2 Fruit Pie

> **Example 2.2**: Blackberries are ready to eat from the beginning of August till the end of October. Apples are harvested on September 1st and will keep for 6 months. For how long can I eat blackberry and apple pie? (*i.e.* For what period are they both available?).

#### Analysis

In the example, the phase that *blackberries are ready* can be assumed to mean that they can be eaten, and *the beginning of August till the end of October* assumed to be the whole months. So, using the temporal elements identified in section 2.2.1 we have

*blackberries can be eaten* throughout *the interval from August up to October*

The sentence *on September 1st and will keep for 6 months* can be assumed the whole month of September and 5 months more. It means that the end of the interval is one month of the next year, but nothing was said about year nor even about specific September of a given year, as in the last sentence and in the farmer example as well! The reason is because any season is inherently cyclical, *i.e.* happens every year. As the discussion about the time representation will be done in chapter 3, let's assume for a while that such cyclical feature of season is captured by our imagined temporal representation. Then, we have that sentence expressed

*apples can be eaten* throughout *the interval from September up to February*

The question asked can be divided in two others, expressed in the following way

1. what is the interval in which blackberries and apples can be eaten?

2. for how long can blackberries and apples be eaten?

The correct correct steps should be to solve the first question, by searching for an interval I that is the intersection between August up to October and September up to February. The matching between these two intervals should be I, and the conclusion is

*blackberries and apples can be eaten throughout September up to October*

Now, using the interval deduced from the first question, and given that throughout means the inclusion of the ends, the inference to be performed is to find the period of time between the two endings of I. Then we have

*blackberries and apples can be harvested for a* period *of 2 months.*

#### Conclusion

We have identified the following important requirements for our temporal reasoner:

- the time representation must deal with cyclical feature of seasons;

- the relation between intervals and periods needs to be also defined in terms of the length of time between the ends of one interval.

- the necessity of matching definitions (unification) of intervals

### 2.2.3 Vegetable Salad

**Example 2.3**: Radishes take 1 month from planting to harvest, lettuces take 2 months, tomatoes take 6 months (if planted after 1st December up to 1st May) cucumbers take 4 months (if planted after 1st February up to 1st June) and carrots take three months at any time of year. If I want to eat salad containing all of the above, when can I eat it and when will I need to plant each vegetable to get it?.

**Analysis**

The information in this example can expressed as follows

- *radishes are harvested 1 month* after *the time they are planted*
- *lettuces are harvested 2 months* after *the time they are planted*
- *carrots are harvested 3 months* after *the time they are planted*
- *tomatoes are harvested 6 months* after *the time from 1st December up to 1st May, if they are planted* throughout *that interval of time*
- *cucumbers are harvested 4 months* after *the time from 1st February up to 1st June, if they are planted* throughout *that interval of time*

As no definite sentence was said about the season in which all vegetables can be planted, we can assume that they can be planted throughout any interval of time, except for tomatoes and cucumbers. However, the deductions for some queries will not be conclusive at all. For example, one might ask "when can radishes be harvested?" Given the informations above, the reasonable answer for such query should be *1 month after the interval of time they were planted.* Even in this way, we will assume they can be planted throughout any interval of time.

In the case of tomatoes and cucumbers the answer for a query like the one above will only be conclusive if the time they were assumed to be planted is throughout the interval specified in their sentences. However, the question asked does not have such feature, as we have seen before, it is much more complicated than that, and we need to consider some further steps to better understand it.

1. It is necessary to infer during which interval of time all the different vegetables can be harvested. But only with tomatoes and cucumbers we can obtain conclusive deductions.

2. suppose that such interval $I$ is successfully deduced. What were the intervals of time in which each one should be planted to provide they are all harvested throughout $I$?

Since radishes, lettuces and carrots were assumed to be planted throughout any interval $I_1$, it is reasonable to assume they can be harvested throughout any other interval $I_2$, if $I_1$ and $I_2$ obey the rule of inference that relate both events (harvested and planted). In this sense, we can deduce the interval in which both tomatoes and cucumbers can be harvested together, and assume that the others are harvested throughout the same interval, as desired.

The question can be treated in two ways. The first is to use some concept of cause-effect relationship, and we could use a relation (ECAUSE) proposed by [Allen 84]. However, the theories on causation consider it a stronger relation than implication. For instance, "every time Fred came to Edinburgh, it rained" might be true, but "Fred come to Edinburgh caused rain" is not.

The second way is to use the Knowledge Base (KB) not only to work out when a standard query Q follows from the KB

KB ⊢ Q

(where Q is logically a single statement), but also to reason about when an implication hold, given the KB

KB ⊢ P → Q

(where P and Q are single statements).

We will use one relation to deal with it, called IMPLY(P,Q). This relation will be defined on chapter 3, and basically it is true if we can conclude, from the KB, that P → Q. However, the chain of reasoning in which we intend to use for the definition of this relation, can not have rules with more than one single statement in the precondition. For example, suppose *foo* is some property, then we could have *foo rice* **throughout** *I if foo tea and foo coffee* **throughout** *I*, and in this case the cause-effect is more complicated because *foo rice* requires that two crops hold the property *foo*.

As the example involves rules of the form $A$ *if* $B$, both $A$ and $B$ are atoms, then the correct steps of deduction to solve our problem are

- *tomatoes can be harvested* throughout *June up to 1st November*
- *cucumbers can be harvested* throughout *June up to 1st October*
- *tomatoes and cucumbers can be harvested* throughout *June up to 1st October*(assume the same interval for radishes, lettuces and carrots being harvested)
- *The* causes *of tomatoes, cucumbers, radishes, lettuces and carrots can be harvested* throughout *June up to 1st October if*

  *tomatoes are planted* throughout *December up to March*

  *cucumbers are planted* throughout *February up to May*

  *radishes are planted* throughout *May up to August*

  *lettuces are planted* throughout *April up to July*

  *carrots are planted* throughout *March up to September*

**Conclusion**

The following points were uncovered from this example

- in order to obtain answer for queries like *when should some event happens in the past such that other happens in the future*, we need to use some form of reasoning about conditional temporal relationships, such as the relation IMPLY as mentioned before

### 2.2.4 Christmas Tree

**Example 2.5**: If a tree is 1m tall at 1 year old, adds 25 % to its height every year, is pruned by 1m per year when its height is higher than 10m, and is planted in 1990, which year will I have a 20m Christmas Tree on December 25th?

### Analysis

In this example we have not just temporal information related to one state of affairs, but the continuous changing of the height of the tree over time. Furthermore, such process is not exactly continuous since it is stopped every once time, every year due the pruning.

The ideal representation should be one that could deduce the height of the tree before and after the time of pruning. To this this end it would be necessary to represent the process of growing for every moment, in other words simulate it. If we call $Y$ the age of the tree, the height is a function of the age of the tree $h(Y)$, $gr$ is the growing rate per year (1.25), and $pr$ (1m) the pruning factor, then we have the following recursive formula

$$h(Y) = \begin{cases} 1 & \text{if } Y = 1 \\ h(Y-1) \times 1.25 & \text{if } h(Y-1) \leq 10 \\ h(Y-1) \times 1.25 - 1 & \text{if } h(Y-1) > 10 \end{cases}$$

The chain of reasoning beyond this representation is *if the age of the tree is 1 year, then its height is 1m. If its height of last year was greater or equal 1m and less or equal to 10m, then its height in the current year is the last plus % 25 more. If its height of last year was greater than 10m, then its height in the current year is the last plus % 25 minus 1m.* For the query done, this recursivity stops for a tree with 23 years old, with a height of 20.80, approximately, when year is equal to 2013.

### Conclusion

The main conclusion we outline from this example is to show that our logic needs to support some simple form of ecological modelling. Temporal cycles in such models are usually handled by imperative looping structures, but we want to avoid any imperative structure. Then, what we need is that

- the temporal language must provide some way to deal with simulation, such that temporal cycles can be represented in a declarative way, where the looping will be done by recursion, with some event as a base case.

### 2.2.5 Grass for Hay

> **Example 2.4**: Between April and the end of August, it is possible to make hay. Grass needs to grow for two months before it can be harvested for hay. If my sheep use the meadow for lambing until the end of May, and are then moved up to the hills, when will I be able to bring my first crop of hay in? Can I get a second crop of hay? If I want two crops of hay, when is the latest that I should move my sheep off the meadow? The sheep won't come down from the hills until the end of September.

#### Analysis

In this example we have the following knowledge to be represented

- *Between April and the end of August, it is possible to make hay.* - Here we have the season of making hay that could be expressed in the following way

  *hay can be made* throughout *April up to August.*

- *Grass needs to grow for two months before it can be harvested for hay.* - This can expressed as

  *Grass can be harvested 2 months* after *it starts growing.*

- *If my sheep use the meadow for lambing until the end of May, and are then moved up to the hills, when will I be able to bring my first crop of hay in?* - This sentence alone is not complete, because we need to know when the sheep go back to the meadow. So we also need the sentence *The sheep won't come down from the hills until the end of September.* Now we can represent it as *sheep can use the meadow* throughout *October up to May.*

Note, that we used the verb "can" because is reasonable to assume that the farmer can move sheep off the meadow at any time during the interval in which they can be there. Now, we need one more sentence, that is implicit. This is to say that the grass start to grow long enough for hay 1 month after the sheep have been moved. Then we have

*grass start to grow for hay 1 month after the last month in which sheeps have used the meadow.*

Finally, our chain of reasoning must consider all these information, and it could be, for example, in the following step.

*Since my sheep used the meadow until the and of May, and one month after the grass start to grow up, then the grass start to grow in the meadow* throughout *June.*

*Since the grass start to grow in the meadow* throughout *June and 2 months after that we can harvest grass for hay, then we can harvest grass for hay* throughout *August.*

- *Can I get a second crop of hay?* - If we plant only throughout June then the answer should be "no", because we have only September that is out of the hay season.

- *If I want two crops of hay, when is the latest that I should move my sheep off the meadow?* - This question suggests some action in which we could change some assertions in our knowledge base. However, the connective throughout is only used to say that some event happens somewhere in the whole of one interval, and does not say that we cannot consider only some sub-interval. We expect that our temporal reasoner system can also consider all sub-intervals that exist in the interval from October up to May.

  Assuming that, what we need is to $i$) find one interval $I_h^1$ of harvest grass for hay. $ii$) Assuming such $I_h^1$, we use the implication temporal relationship IMPLY, to find the interval $I_p^1$ for grass start to grow to allow that time of harvesting. Assume that in the same $I_p^1$ we can have another harvesting, and use the same relation to find the time necessary grass start to grow $I_p^2$. Finally, uses IMPLY relation to find the time $I_s$ that sheep should be moved to allow the planting in $I_p^2$.

Using this chain of reasoning, and using the first deduction harvesting grass we have

*Grass can be harvested for hay* throughout *August, if it started to grow* throughout *June. Grass can be harvested for hay* throughout *June, if it started to grow* throughout *April. Grass start to grow* throughout *April, if sheep was moved off the meadow* throughout *an interval of time that finish in March.*

#### Conclusion

This example has shown us a very interesting aspect of temporal statements. It is reasonable to to make any definitive statement, but specify a broad interval within which some event may possibly occur throughout that interval. If we had stated that *sheep use the meadow until the end of May*, we could never find another possible time of harvesting during the hay season. Along with this observation, we also identify the necessity that a temporal reasoner system should allow many alternative time intervals, that is, sub-intervals of the intervals expressed in the temporal sentences of our knowledge base.

## 2.3 Temporal Features Desirable

The objective of this section is, based on the analysis and conclusions of the section 2.2, to figure out the main concepts that a temporal language should have to represent the temporal information we want, and which features of temporal reasoning are needed to deduce the correct temporal conclusions from temporal knowledge expressed in the language.

### 2.3.1 Time Representation

In terms of temporal representation, the conclusions pointed out that

- the language should deal with temporal information related with units of time, and basic temporal relations rather than the awkward temporal knowledge of expressions like "frequently", "occasionally", etc.

- the units of time should be represented in as many different levels as necessary (*e.g.* days, weeks, months). This means that the equivalence between any two levels and transformations of temporal units from one to other should be provided.

- the language should provide units of time to represent cyclical information. The cyclical representation in one level of time should be related with the periodical occurrence of temporal units at other related levels of time. For instance, June is one temporal entity [1] of every year. So, *planted maize* **throughout** *June* will mean that every June maize can be planted. A complete interval of one year is from January up to December, then in the level of months January must follow December because they are cyclical temporal entities.

- the language should provide a way to relate temporal units by using relations to represent how the levels are related to each other in terms of definition. The user should have the facility of change the units of time without affecting the semantic of the language (*e.g.* a microbiologist may wish to represent the life cycles of viruses in terms of minutes, hours ...).

Considering such requirements the temporal units to be specified are those identified before, and we just add the necessity of a cyclical interval and the concepts that appear with them. So, the units are:

- *cyclical intervals of time* to represent cyclical temporal entities, and along with it the definition of *relative past, future*, *e.g.* June is the future of January 5 months after it, and June is also the past of January 7 month before it, and also 19 months, and so on.

[1] To be defined in next chapter

- periods of time as length of intervals, *e.g.* 2 months is the period of the interval between June and August, but also the period of any other interval with equal length.

- one unit to represent the *absolute* flow of time that relates the relevant levels of time representation. This cannot refers to cyclical intervals, but includes all levels which allow periodicity.

- linear intervals of time defined by *absolute* flow of time

### 2.3.2 Temporal Reasoning

In terms of temporal reasoning, the conclusions pointed out that

- the language should allow temporal operators such as after and around, to transform complex temporal informations in simple forms of intervals, *e.g. 1 month after 2 months after June* is equivalent to *September*

- the language should allow temporal connectives analogous to "always" ($\Box$), and "sometimes" ($\Diamond$) of modal logic. As we wrote before, in this project we will specify just the first by the *throughout* connective.

- the language should allow not only standard forms of queries, but also to reason about when an implication hold.

# Chapter 3

# An Ontology for Natural Time

*What, then, is time? I know well enough what it is, provided that nobody asks me; but if I am asked what it is and try to explain, I am baffled.*

Saint Augustine (The Confission, Book XI, section 14).

In this chapter we shall present a logico-mathematical structure of natural time, where we can represent concepts of time we have seen in the examples of chapter 2. Our main challenge will be to allow the definition of inherently cyclical temporal classes in our logic, and also join it with the linear feature of the instances of them in the flow of time. To achieve this we will present our ontology of time by laying out a series of formal definitions of temporal concepts.

## 3.1 Another Ontology of Time

In this section we will define the concepts of Purely Cyclic and Linear/Cyclic structure of time. The first is a base structure that to become computationally tractable needs some restrictions, and such restrictions yield the second one. The ontology presented here is based on the following assumptions:

- *temporal entity* (TE) - is a reference to one measure of time, *e.g.* June, 24th, 1980.
- TEs with similar temporal features are grouped in *temporal classes*, *e.g.* June and January are TEs of the class month, 1st and 25th are TEs of the class day.

- TEs of some temporal classes are circularly ordered, *e.g.* in the class "month" the last element, December, is followed by the first, January.

- TEs will be represented in our logic by the number of their position in their class, *e.g.* June is 6 in the class months.

### 3.1.1 A Purely Cyclic Structure of Time

**Definition 3.1** *A* hierarchical cycle structure of time $\mathcal{T}$ *is an ordered triple* $\langle \mathcal{L}_C, \mathcal{E}, \overset{t}{\succ} \rangle$, *where* $\mathcal{L}_C$ *is the set of modular temporal classes which determine the levels of hierarchy,* $\mathcal{E}$ *is a non-empty set of temporal entities,* $\overset{t}{\succ}$ *is a binary relation of temporal ancestor.*

Each level of this hierarchy represents one temporal class as defined below.

**Definition 3.2** *Let* $t_1, \ldots, t_m$ *be a sequence of temporal entities, with a well defined set of properties which depends on a strictly circular ordering among them. We say that such sequence is a* modular temporal class *(MTC),* $C^m$, *where each* $t_i$ *corresponds to exactly one element of a modular set* $Z_m$. *If each* $t_i$ *is an instance of another MTC* $C^k$, *then we also say that* $C^m$ *is a MTC defined by* $C^k$.

**Example 3.1.1** - The temporal concept of "hour" can be modelled as one MTC defined by 60 instances of the MTC minute, and minute is another MTC defined by 60 instances of the MTC second, and so on.

**Definition 3.3** *Let* $i$, $j$ *be integers where* $j > i$, $C^m$ *one MTC defined by the set of temporal entities* $\{t_1, \ldots, t_m\}$, *where* $t_1$ *is one instance of* $C^{m1'}, \ldots, t_m$ *is an instance of* $C^{mk'}$ *and* $C^{m1'}, \ldots, C^{mk'}$ *are members of the same modular temporal class* $C^{m'}$. *We write* $C^m \overset{t}{\succ} C^{m'}$ *to say that* $C^m$ *is the* temporal ancestor *of* $C^{m'}$. *We also write* $C^m_j$ *to say that the level of* $C^m$ *is* $j$, *and* $C^{m'}_i$ *to say that the level of* $C^{m'}$ *is* $i$.

**Example 3.1.2** - The temporal ancestor of second is minute, and of minute is hour, and of hour day, and so on.

Note that in such definition if $m'_1 = m'_2 \ldots = m'_k = m'$ then all $t_i$s are direct elements of $\mathcal{C}_i^{m'}$. In fact the definition of one MTC $C_i^m$ is directly dependent on the existence of another MTC that sub-divide it. It determines what is called *density of time*, that is for every MTC $C_j^{m_j}$ there exists a different MTC $C_i^{m_i}$ such that $C_j^{m_j} \stackrel{t}{\succ} C_i^{m_i}$.

**Properties of** $\stackrel{t}{\succ}$ - The $\stackrel{t}{\succ}$ relation has the following properties.

- *transitive* - if $\mathcal{C}_i^{mi}$, $\mathcal{C}_j^{mj}$ and $\mathcal{C}_k^{mk}$ are MTCs and $\mathcal{C}_k^{mk} \stackrel{t}{\succ} \mathcal{C}_j^{mj}$ and $\mathcal{C}_j^{mj} \stackrel{t}{\succ} \mathcal{C}_i^{mi}$, then $\mathcal{C}_k^{mk} \stackrel{t}{\succ} \mathcal{C}_i^{mi}$.

- *reflexive* - if $\mathcal{C}_i^{mi}$ is a MTC then $\mathcal{C}_i^{mi} \stackrel{t}{\succ} \mathcal{C}_i^{mi}$ (every MTC can be subdivided in itself)

- *anti-symetric* - if $\mathcal{C}_i^{mi}$, $\mathcal{C}_j^{mj}$ are MTCs, and $i \neq j$, and $\mathcal{C}_j^{mj} \stackrel{t}{\succ} \mathcal{C}_i^{mi}$, then $\mathcal{C}_i^{mi} \stackrel{t}{\not\succ} \mathcal{C}_j^{mj}$.

Such concept can be understood in the following way: for each complete cycle of TEs of one class (a path of length equal to m), we associate such a cycle with other temporal class.

**Example 3.1.3** - $C_2^3$ is one MTC defined by the MTC $C_1^4$, and $C_1^4$ is defined by integers. As each instance of $C_2^3$ is represented by its position, then $C_2^3 = \{1, 2, 3\}$, and $C_1^4 = \{1, 2, 3, 4\}$. In this way, each complete cycle of $C_1^4$ corresponds to exactly one element of $C_2^3$. Figure 3–1 shows graphically what we mean in this example.

This example can also be seen, if we forget the level above, as a collection of modular sets in a bag like $\{\{1, 2, 3, 4\}, \{1, 2, 3, 4\}, \{1, 2, 3, 4\}\}$. In other words, if we have in one

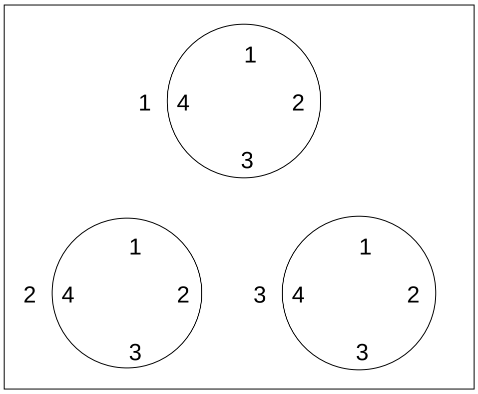


**Figure 3–1:** Simple example of a circular hierarchy of time.

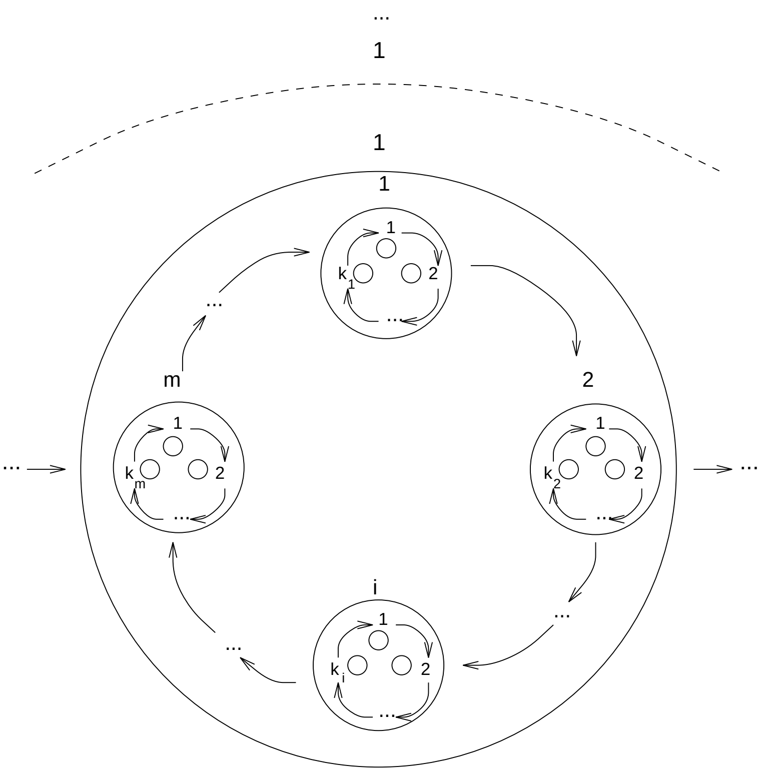


**Figure 3–2:** A hierarchical view of cyclical time representation.

level $i$, one MTC with $m = k_i$, of TEs of one class in the level $i-1$, then every instance of the class in the level $i$ refers to exactly one sequence of TEs of the level $i-1$ with length $k_i$, or one complete cycle.

This hierarchical view of time representation can be seen in figure 3–2.

Given such time structure definition we should introduce how to represent the units of time, temporal connectives and the truth conditions for the connectives. However, due the complex feature of such approach in which the flow of time is embodied in a infinite number of levels of hierarchy, we need to impose some restrictions in order to avoid such complexity and makes the approach more computationally tractable.

### 3.1.2 A Linear-Cyclic Structure of Time

As we pointed out in section 3.1.1, the approach for time representation as a purely cyclic structure is not very useful because we do not know which level of temporal class we can use to talk about temporal relations between events in terms of past and future, since we always need one level above, and so on. Now, we introduce some restrictions and formulate a new definition of time representation that inherits the properties associated with MTC and the levels of hierarchy.

The first natural restriction is to impose highest and lowest bound limits, and consider the lowest as a kind of temporal entity that does not need to be sub-divided, but is not necessarily a point. The highest level is then the level to be used for temporal deductions about past and future given a certain "instant of time". However, this lead to a problem: the time succession in the highest level goes around in circles for ever.

One solution to this problem would be to consider that all kinds of events can be captured in terms of temporal aspects, in only one big cycle, with the difficulty that we need to know the value of $m$ for the highest level. Furthermore, we would not have 'flow of time', unless we could note other macro-phenomena that needed the creation of another class to represent the flow of time, and so on.

This strongly suggests our second restriction, that is that the highest level must be a linear structure in order to refer the flow of the TEs of type MTCs. We have two alternatives. First is to define another type of temporal class to be used only as the highest level of hierarchy, and the second is always to define the highest level with a very big value for m to represent the infinite. The second approach seems to be reasonable since we will do not have problems of *eternal* cycle and the big value of $m$ will not be reached in practical applications. However, even in this case we need to formally define this new view. The second alternative will be used in this work.

**Definition 3.4** *Let $C_n^k$ the highest MTC, in a $\mathcal{T}$ structure with k levels. We call the* flow of time *to an special MTC defined by an infinity sequence of temporal entities of $C_n^k$, and then for each instance of $C_n^k$ there is one $x \in Z^+$ associated with it.*

Now, we can define the mixed hierarchical structure of time as follows.

**Definition 3.5** *A* hierarchical mixed structure of time $\mathcal{TM}$ *is a 5-tuple $\langle \mathcal{L}_C, \mathcal{E}, \overset{t}{\succ}, T, > \rangle$, where $\mathcal{L}_C$ is a finite set of sets of MTC, $\mathcal{E}$ is a non-empty set of temporal entities, $\overset{t}{\succ}$ is a*

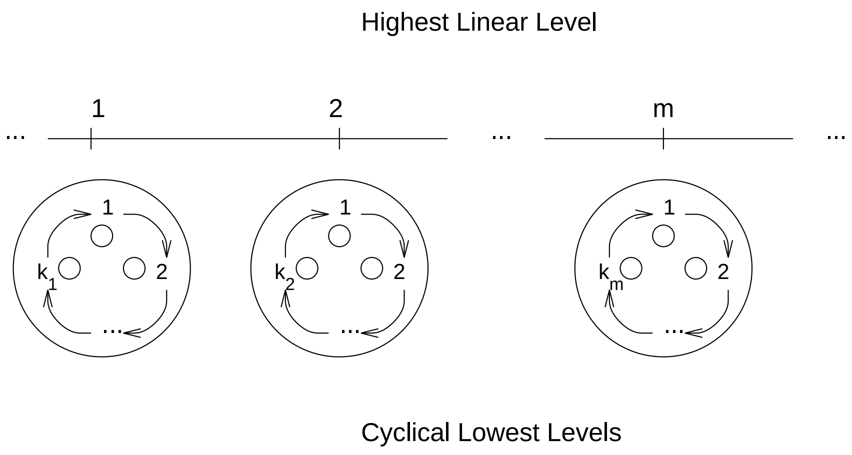


**Figure 3–3:** A mixed view of time representation.

*binary relation of temporal successor, $T$ is a finite set of positive integers ordered by the binary relation $>$, and each $x \in T$ corresponds to one instance of the MTC of the highest level.*

Note that we introduce a finite set $T$ of integers, because we are not interested in the infiniteness feature of the integers for computational reasons. This new view of time representation is shown in figure 3–3.

### 3.1.3 Units of Time

Using this new mixed hierarchy of time representation, we can define the units of time needed to reason about temporal statements. The basic elements normally considered as units of time are instants (or points) and intervals (or periods) of time, and length of time is a measure of the distance in time. In this work, we will use the word "period" not associated with "interval", as usual, but it will be treated as the length of time. The objective of this section is to present such elements.

#### Range Notation

Before defining intervals we have to define a operator that will be used for intervals of time definition, that is the *range* notation to express a numerical interval with no temporal class associated.

**Definition 3.6** *Let $x_1, x_2 \in Z^+$, then we write $x_1...x_2$, to mean that we consider all values from $x_1$ to $x_2$.*

**Example 3.1.4** - Valid ranges are 2...3, 1...1, 10...2 (for modular intervals). Non valid ranges are -1...0, 2...-1.

### Cyclic Interval of Time

In a point structure of time, an interval is represented by a pair $[s, t]$, where $s$ and $t$ are the starting and ending points, respectively. However, as mentioned in section 1.4, we need complex concepts to represent temporal phenomena that are inherently cyclical temporal classes.

**Definition 3.7** *Let s, t be instances of the same MTC $C_i^m$, in a $\mathcal{TM}$ model of time with k levels, and $i < k$. We call interval(s...t,$C_i^m$) a* modular interval (MI) *of time between s and t in the level i. We also say that such an interval represents one* cyclical interval.

The interval defined is considered to be cyclical, or intermittent, because $s$ and $t$ are TEs of one MTC, and by definition all levels higher than $i$ are defined by TEs of the MTC below. This means that, every TE of the levels above $i$ must have the interval defined by $s$ and $t$. For instance, $interval$(3...4, month) refers to every instance of the interval between March and April, or the occurrence of such interval every year, since every year has one instance of it.

Note that in this definition we did not make any difference between $s$ and $t$. It means that we can have intervals that express only one temporal entity in one level. This is another type of "instant" of time, since $interval(s...s, C_i^m)$ expresses an interval with only one temporal entity.

**Example 3.1.5** - what follows are examples of cyclic intervals in a calendar model of time, where *month* and *day* are MTC classes defined.

$interval(1...9, month)$ - interval between every January and September;

$interval(10...3, month)$ - interval between every October and March;

$interval(5...5, month)$ - every month of May;

$interval(20...5, day)$ - every interval between the 20th day of one month and the 5th day of the next month.

### Matching Relations Between Cyclical Intervals

The matching relations are basically those defined by [Allen & Hayes 85], with the difference that we have more cases of matching than they. When using cyclic intervals, the matching can be done only between intervals of the same level of hierarchy, mainly those in the levels of modular temporal classes because of their circular ordering. What we shall specify now are the three basic types of matching related with intervals (with their all sub-cases), which are EMPTY, INCLUDED and OVERLAPPING. We do not need to specify the symetric case of each one because they are obvious.

In the following analysis of cases we are considering two things. First, that the ordering of the circular set is like the clock, and that $s_1...t_1$ is the range of interval $I_1$ and $s_2...t_2$ is the range of interval $I_2$. Second, we will assume linearity between the elements, by picking the first without the circular link with the last.

1. EMPTY match - there are two different cases with the empty result, or one fail result, as shown in figure 3–4. For each case we have the following conditions, and the result of the match for all of them is the same, that is the empty set.

   (a) $t_2 > s_2$ and $s_2 > t_1$ and $t_1 > s_1$

   (b) $s_2 > t_1$ and $t_1 > s_1$ and $s_1 > t_2$

   **Example 3.1.6** - Suppose we are considering the MTC month, then the following are empty matches represented by $\phi$.

   (a) $interval(2...3, month)$ and $interval(5...5,\text{month})$, where $I_1 = 2...3$ and $I_2 = 5...5$, result is $\phi$.

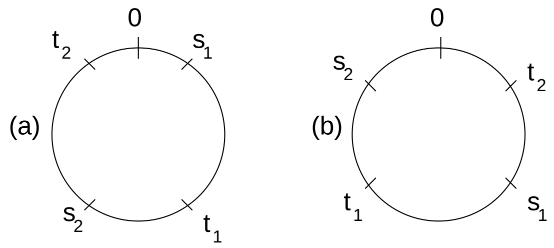


**Figure 3–4:** Empty matching of modular intervals.

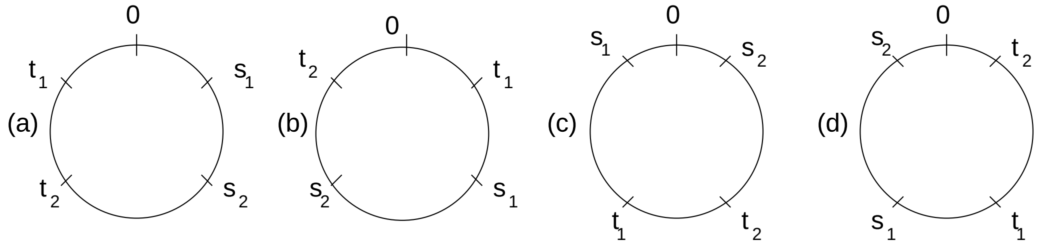


**Figure 3–5:** Matching of one interval including other.

(b) $interval(4...6, month)$ and $interval(10...2, month)$, where $I_1$ =4...6 and $I_2$ = 10...2, result is $\phi$.

2. INCLUDED Match - there are four different cases of one interval is included in th other as showed in figure 3–5. For each case we have the following conditions, and the result of the match for all of them is the same, that is $s_2...t_2$.

   (a) $t_1 \geq t_2$ and $t_2 > s_2$ and $s_2 \geq s_1$

   (b) $t_2 > s_2$ and $s_2 \geq s_1$ and $s_1 > t_1$

   (c) $s_1 > t_1$ and $t_1 \geq t_2$ and $t_2 > s_2$

   (d) $s_2 \geq s_1$ and $s_1 > t_1$ and $t_1 \geq t_2$

**Example 3.1.7** - The following are matches where the first includes the second.

(a) $interval(2...10,\text{month})$ and $interval(4...6, month)$, result is $interval(4...6, month)$.

(b) $interval(4...1, month)$ and $interval(6...10, month)$, result is $interval(6...10, month)$.

(c) $interval(10...8, month)$ and $interval(1...4, month)$, result is $interval(1...4, month)$.

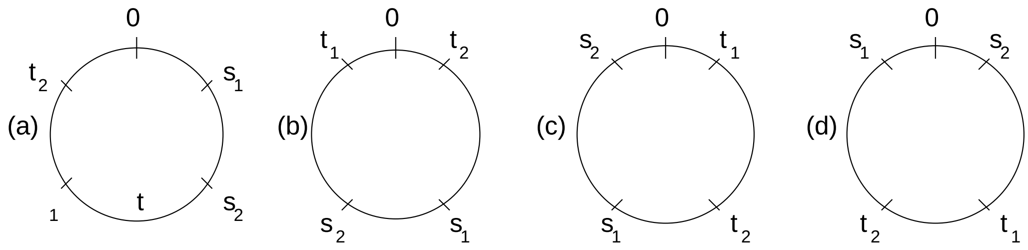


**Figure 3–6:** First Four cases of overlapping between intervals.

(d) $interval(10...4,month)$ and $interval(11...4,month)$, result is $interval(11...4,month)$

3. OVERLAPPING Match - there are 6 different cases of overlap between two modular intervals. They are diveded in two groups

   i the cases showed in figure 3–6. For each case we have the following conditions, and the result of the match for all of them is the same, that is $s_2...t_1$

   (a) $t_2 > t_1$ and $t_1 \geq s_2$ and $s_2 > s_1$

   (b) $t_1 \geq s_2$ and $s_2 \geq s_1$ and $s_1 > t_2$

   (c) $s_2 > s_1$ and $s_1 > t_2$ and $t_2 > t_1$

   (d) $s_1 > t_2$ and $t_2 \geq t_1$ and $t_1 \geq s_2$

   **Example 3.1.8** - The following are overlapped matches of the first three cases between intervals.

   (a) $interval(2...8,month)$ and $interval(4...10,month)$, result is $interval(4...8,month)$.

   (b) $interval(4...11,month)$ and $interval(6...3,month)$, result is $interval(6...11,month)$.

   (c) $interval(7...2,month)$ and $interval(11...4,month)$, result is $interval(11...2,month)$.

   (d) $interval(10...4,month)$ and $interval(2...8,month)$ , result is $interval(2...4,month)$.

   ii the cases shown in figure 3–7. For each case we have the following conditions, and the result are two possible intervals $s_1...t_2$ and $s_2...t_1$

   (a) $t_2 \geq s_1$ and $s_1 > t_1$ and $t_1 \geq s_2$

   (b) $s_2 > t_2$ and $t_2 \geq s_1$ and $s_1 > t_1$

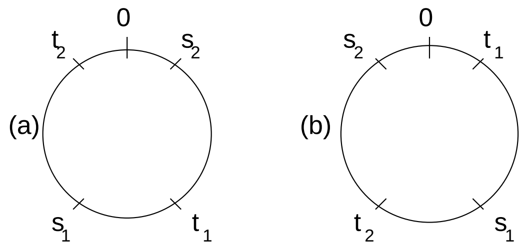


**Figure 3–7:** The Last two cases of overlapping between intervals.

**Example 3.1.9** - The following are overlaps between intervals where two results are possible.

(a) $interval(8...4,month)$ and $interval(2...11,month)$, results are $interval(2...4,month)$ and $interval(8...11,month)$.

(b) $interval(4...2,month)$ and $interval(11...6,month)$, results are $interval(4...6,month)$ and $interval(11...2,month)$.

In this case, when matching the intervals we expect to obtain only one answer, and the other can be computable as an alternative matching. Computationally speaking it is normally done by backtracking.

### Instant of time

In a point structure representation of time, one instant of time is considered as one single point. In our approach, we consider that even the instances of the MTC of the lowest level can be sub-divided in one cycle of TEs of another MTC. So the relation between the two approaches cannot be done directly. However, if we interpret [1] the TEs of the lowest level as points, we have no loss of the properties as in the original consideration, it is rather a simplification than a complication. Considering such simplification lets see what is an instant of time in our approach and compare it with the point structure.

In a $\mathcal{TM}$ model of time, one instance of one MTC is useful to represent the flow of time only if it is referred to one instance of the class above, and so on until the highest

[1] Note that such interpretation is only for the purpose of comparison

level that can be associated with a linear sequence of numbers. In other words, we need a way to say that one particular instance of one MTC in a level $i$ is a member of one particular instance of the MTC in the level $i+1$. This chain of relation until the highest level is the instant of time for such model of time, since the highest level is rather a big interval than one point of time. Then we can state the following relation.

If we have $k$ levels of hierarchy in a $\mathcal{TM}$ model of time, then the instant of time $n$ in a linear point structure can be represented as a function of the instances of each level, *i.e.* $n = t^T(t_1, \ldots, t_k)$, where its meaning is that $t_1$ belongs to the cycle defined by $t_2$, belongs to $\ldots$ and $t_{k-1}$ belongs to the cycle defined by $t_k$. The range of this function is the positive integers $Z^+$.

Another way to better understand is to remember the definition 3.2 of MTC and its association with the bag, in example 3.1.3.

**Example 3.1.10** - In the calendar model of time $t$(Day, Month, Year) means the order of the element of the big bag with Year number of bags of complete month cycles and days (365 days and 366 days for leap years), Month - 1 complete bags of days, and Day number of elements. Suppose we have $t$(10,March, 1990), then its correspondent point structure instant of time is 1990 $\times$ 365 + 1990 div 4 [2] + days-in(January) + days-in(February) + 10 = 726916.

Now, we can define a linear interval time by using the range of instants of time.

**Definition 3.8** *Let s=t($s_1, \ldots, s_k$), t=t($t_1, \ldots, t_k$) two instants of time in a in a k-level $\mathcal{TM}$ model. Then s...t is the interval of time between s and t. We also say that this interval is a linear interval.*

**Example 3.1.11** - The expression $t$(10,3,1990) ... $t$(5,1,1994) is a linear interval of time.

[2]This is the leap year factor, where div is the integer division.

**Period of Time**

The third element that we consider as a unit of time to be defined is the period of time, that has a different meaning and application to an interval, that is, *period of time is the length or portion of time* [3].

In our $\mathcal{TM}$ model of time this unit is clearly identified as the length of any interval. This is to express sentences that say "how long" some statement $S$ is true or not. Then we have the following definition.

**Definition 3.9** *Let $I$ be one interval in a $\mathcal{TM}$ model of time, $\Delta \in Z^+$. Then* time between *is function $T_B$ that maps $I$ to $\Delta$. Delta is called $\Delta$ the length of $I$.*

**Example 3.1.12** - $T_B(interval(9...10,month)$ = 2 months, $T_B(t(10,3,1990)...t(5,1,1994))$ = 3 years plus 9 months plus 26 days, or 1397 days.

**Relations Between Periods** - The relations between periods are those that relateone period of one level of hierarchy with another levels.

**Definition 3.10** *Let $P$ = period($\Delta, C_i^m$) one length of time in level $i$. We call* down-equivalence ($Eq \downarrow$) *the relation between $P$ and other lower levels, is defined as*

$Eq \downarrow (P) = P$; *or*

$Eq \downarrow (P) = Eq \downarrow (period(\ \Delta \times m, C_{i-1}^{m'})$

**Example 3.1.13** - $Eq \downarrow$(*period* (2,year)) are *period*(2,year), or *period*(24,month) or *period*(365,day).

**Definition 3.11** *Let $P$ = period(D,$C_i^m$) one length of time in level $i$. We call* up-equivalence($Eq \uparrow$) *the relation between $P$ and other higher levels, is defined as*

[3]Oxford Advanced Learner's Dictionary - Oxford University Press, 1989.

$Eq \uparrow (P) = P$, *if* $D < m$

$Eq \uparrow (P) =$ *period(D',$C_i^m$) plus* $Eq \uparrow$ *(period(R,$C_{i+1}^{m'}$)), where D' = D* **mod** *m, and* **mod** *is the reminder of D / m.*

**Example 3.1.13** - *Eq* $\uparrow$(*period*(1000,day) is period(10,day) plus period(9,month) plus period(2,year).

### 3.1.4 Future and Past relations

Now we can define the concepts of *future* and *past* in a $\mathcal{TM}$ model of time. However, as the flow of time can only be captured by time instants, or linear intervals, we will first define these concepts as usual, and then we will show another vision of *future* and *past* for cyclic intervals.

**Definition 3.12** *Let* $s = t(s_1, \ldots, s_k)$ *and* $t = t(t_1, \ldots, t_k)$. *We call* is_future *a binary relation between t and s iff*

$s_k < t_k$, *or*

$s_i = t_i$ *and* $s_{i-1} < t_{i-1}$, *for* $i = k \ldots 2$

*We say that t is an* absolute future *of s, and s is an* absolute past *of t.*

We use such a relation to define all matching relations between linear intervals as defined by [Allen & Hayes 85].

However, as we also want to make inferences in terms of cyclic intervals, we need to define what we call *relative past* and *future*. Such relations can only be defined if they use some relation that involves intervals and periods.

**Definition 3.13** *Let s,t be two cyclic intervals of a MTC* $C_i^m$, *in a* $\mathcal{TM}$ *model with k levels, and* $i < k$, *and p is a period of time of the level i. We say that s is the* relative past *of t, where the period of time between them is P, noted by future(s,p,t) iff* $s \oplus p = t$. *We say that p is the relative future of s.*

This relation is called relative because it is also true if we change the positions of $s$ and $t$, *i.e.* $future(t, p', s)$, where $p' = m \ominus p$.

**Example 3.1.13** - Examples of this relation

$future(interval($2...2,month$), period($3,month$), interval($5...5,month$))$ and is also true

$future(interval($5...5,month$), period($9,month$), interval($2...2,month$))$.

Such relation can also be used for time instants, but only in one way, *i.e.* if $s$ and $t$ are time instants and $future(s, p, t)$ is true then $future(t, p', s)$ is not true.

### 3.1.5 Temporal Operators and Connectives

Before to give the definition of the temporal operators to be used with temporal units, we need to define two operations to be used. These operations are *up-wave modular sum,* and *subtraction.*

#### Up-Wave Modular Sum and Subtraction

**Definition 3.14** *Let $P$ = period($\Delta, C_i^m$), and $I = t(s_1, \ldots, s_k)$. We call the* up-wave modular sum *between $P$ and $I$, $I\, \omega_\oplus\, P$, defined as*

$I\, \omega_\oplus\, P = t(s_1, \ldots, s_i + \Delta, \ldots, s_k)$, *iff* $s_i + \Delta < m$

$I\, \omega_\oplus\, P = t(s_1, \ldots, s_i \oplus \Delta, \ldots, s_k)\ \omega_\oplus\ P'$, *iff* $s_i + \Delta \geq m$, *and* $P' = period(\Delta', C_{i+1}^{m'})$, *where* $\Delta' = s_i + \Delta$ *div* $m$.

**Definition 3.15** *Let $P$ = period($\Delta, C_i^m$), and $I = t(s_1, \ldots, s_k)$. We call the* up-wave modular subtraction *between $P$ and $I$, $I\, \omega_\ominus\, P$, defined as*

$I\, \omega_\ominus\, P = t(s_1, \ldots, s_i - \Delta, \ldots, s_k)$, *iff* $\Delta < s_i$

$I\, \omega_\ominus\, P = t(s_1, \ldots, s_i \ominus Delta, \ldots, s_k)\ \omega_\ominus\ P'$ , *iff* $\Delta >= s_i$ *and* $P' = period(\Delta', C_{i+1}^{m'})$, *where* $\Delta' = \Delta$ *div* $m$.

**The Operators "around" and "after"**

This type of operators are to be used only with temporal units.

The "around" operator is inteded to express the vagueness of some temporal knowledge embodied in some complex chaining of temporal information. Its semantic interpretation is one interval of time associated with the vague information it is intend to represent.

**Definition 3.16** *Let $P = period(\Delta, C_i^m)$ a period of time, $T$ one temporal interval in a $\mathcal{TM}$ model of time. Then* P around T *is a vague interval of time that can be reduced to*

- *$interval(s \ominus \Delta ... t \oplus \Delta, C_i^m)$-*

  *iff $T$ is a cyclical interval $interval(s...t, C_i^m)$*

- *$t(s'_1, \ldots, s'_n)\ \omega_\ominus\ P\ ...\ t(t_1, \ldots, t'_n)\ \omega_\oplus\ P$,*

  *iff $T$ is a linear interval $t(s_1, \ldots, s_n)...\ t(t_1, \ldots, t_n)$*

**Example 3.1.14** - Examples of this operator are

$period$(1,month) $around$ $interval$(6...6,month) = $interval$(5...7,month)

$period$(1,month) $around$ $t$(1,12,1990)...$t$(1,12,1990) = $t$(1,11,1990)...$t$(1,1,1991)

The "after" operator is defined as follow.

**Definition 3.17** *Let $P = period(\Delta, C_i^m)$ one period of time, $T$ one temporal interval in a $\mathcal{TM}$ model of time. Then* P after T *is one unit $T'$ of time in the future of $T$, defined as*

- *$interval(s_1 \oplus \Delta ... s_2 \oplus \Delta, C^m)$,*

  *iff $T$ is a cyclical interval $interval(s_1 ...\ s_2, C^m)$*

- *$t(s'_1, \ldots, s'_n)\ \omega_\oplus\ P\ ...\ t(t_1, \ldots, t'_n)\ \omega_\oplus\ P$.*

  *iff $T$ is a linear interval $t(s_1, \ldots, \ldots, s_n)...\ t(t_1, \ldots, \ldots, t_n)$.*

**Example 3.1.15** - Examples of this operator are:

*period*(3,month) *after interval*(3...4,month) is equivalent to *interval*(6...7,month)

*period*(2,month) *after* $t$(1,10,1990)...$t$(12,11,1990) = $t$(1,12,1990)...$t$(12,1,1991).

**Temporal Connectives**

The only temporal connective we are interested in the moment is to express sentences like *maize can be planted* **throughout** *June*. This operator is equivalent to the "always" modal operator. Although we can not express things like "sometimes", we intend to give a well defined the behaviour for **throughout**, and any other connective can be introduced in the extension of this work.

**Definition 3.18** *Let E be an event, and T a temporal interval. Then E @ T is true iff E @ T' is true, and T' is a sub-interval of T, for every T' of T.*

### 3.1.6 The IMPLY relation

Finally, the last thing we want to define is the IMPLY relation.

**Definition 3.19** *Let* $E_i$, $E_j$ *be events, and* $T_i$, $T_j$ *be time intervals. We say that* imply$(E_i@T_i, E_j@T_j)$ *if either*

- $E_i@T_i \rightarrow E_j@T_j$ *or*

- $E_i@T_i \rightarrow E_k@T_k$ *and* imply$(E_k@T_k, E_j@T_j)$*, where* $E_k$ *and* $T_k$ *are events and time interval, respectively.*

# Chapter 4

# NatureTime: A Temporal Reasoning System

In this chapter we will specify the syntax of our logic, and also a simple temporal reasoner called **NatureTime**, based on the logic defined in chapter 3. The syntax of the logic and the meta-interpreter are included in the same chapter because we intend to show the semantics of the logic in a informal way, and so we will use the meta-interpreter to do it. Basically, it will be done in the following way: for each sentence in our language, we will give its semantics by showing the meta-interpreter clause to solve the sentence, and explain the semantic by using the declarative interpretation of Prolog.

In section we extend a standard Prolog meta interpreter with a restricted form of unification on temporal labels, or types. This unification is the main extension of the meta interpreter, and its behaviour must satisfy the relations between units of time specified in chapter 3.

## 4.1 Syntax

The main goal of this section is to formalize the syntax of our logic based on the ontology presented in section 3.1. However, we will not formally describe the semantic of the language, because we will leave this for section 4, where it will be done in a informal way when we show the specification of the **NatureTime** system, a temporal reasoning system based on our logic. Time limitations prevent detailed development of a formal semantics and the project plan was not to prove the properties of soundness of the logic, just to implement a very simple meta-interpreter, to demonstrate how the logic could be used.

### 4.1.1 Vocabulary

The vocabulary is formed by

- a finite set $\mathcal{T}_v$ of variable symbols; we shall denote members of this set with capital letters. For instance, $X$,$Y$ are examples of variables.

- a finite set $\mathcal{T}_f$ of function symbols of the form $f_i^n, i \geq 1$, where $n$ is the arity of the function. When $n = 0$ we call the function a constant symbol $c_i$, and lowest and flow_time is a special constant, while period, interval, t are special functions, and "...",plus are special infix binary functions of this set. For instance, period, plus, interval, t in period(2,hour) plus period(37, minute), interval(2...3,month), t(17,6,1994) are function symbols

- a finite set $\mathcal{T}_p$ of predicate symbols $p_i^n$, $i \geq 1$, where $n > 0$ is the arity of $p_i$, where temp_mod_class$^3$ is special predicate symbol of arity 3.

- a set $\mathcal{V}_M$ of modal values defined as

  $\mathcal{V}_M = \{$infinity, smallest$\} \cup \mathbb{N}^+$

- a finite ordered set $\mathcal{T}_h$ of modular temporal classes $\{mc_1, \ldots, mc_n\}$, called the set of *time levels*, where

  - $mc_n =$ temp_mod_class(flow_time,$c_n$, infinity)
  - $mc_i =$ temp_mod_class($c_i, c_{i-1}, m_v$), $1 < i < n$, where $c_i$ and $c_{i-1}$ are constant symbols of $\mathcal{T}_f$, $c_{i-1} \neq$ lowest, and $m_v \in \mathcal{V}_M, m_v \neq$ infinity
  - $mc_1 =$ temp_mod_class($c_1$, lowest, smallest), and $c_1$ is one constant symbol of $\mathcal{T}_f$

each $c_i$ is called the name of the class $mc_i$. For instance,

{ temp_mod_class(flow_time, hour, infinity)

temp_mod_class(hour, minute, 60),

temp_mod_class(minute, second, 60),

temp_mod_class(second,lowest, smallest)} is one set of *time levels*.

- a set of propositional connectives ∼(negation), & (conjunction), or(disjunction), $\Longleftarrow$ (implication)

- the symbols of truth values T,F

- the temporal connective @

- a set $\mathcal{T}_{pt}$ of pure temporal operators consisting of { after, around }

### 4.1.2 Classes of Expressions

Using these symbols we define the following classes of expressions

- a *logical term* is recursively defined as

  - a variable $X$ of $\mathcal{T}_v$
  - any function symbol $f^0$,or a constant.
  - if $t_1, \ldots, t_n$ are terms then $f_i^n(t_1, \ldots, t_n)$ is a term

  **Example 4.1.1** - *maize, grass, height(tree)* , are logical terms.

- a *temporal term* is

  - a *period term* that is
    - a *single period* period$(d, c)$ where $c$ is the name of one element of $\mathcal{T}_h$, and $d \in \mathbb{N}^+$
    - a *composed period* $p$ plus $p_c$, where $p$ is *single period*, and $p_c$ is *period* term

- interval($s...t, c$), where $c$ is one element of $\mathcal{T}_h$, $s, t \in \mathbb{N}^+$ and range on the $m_v$ of $c$
- $s_1...s_2$, where $s_1$ and $s_2$ are in the form t$(t_1, \ldots, t_k)$, $k$ is the number of *time levels* or the length of $\mathcal{T}_h$, and each $t_i \in \mathbb{N}^+$ and range on the $m_v$ of its correspondent class

**Example 4.1.2** - Temporal terms are

period(3,month) plus period(2,day)

interval(10...3,month)

t(15,2,1994)...t(3,4,1994)

- a pure temporal expression is
  - $t$, if $t$ is a temporal term
  - $p$ *op* $t$, where $p$ is a *period* term, $op \in \mathcal{T}_{pt}$, and $t$ is a pure temporal expression

**Example 4.1.3** - Pure temporal expressions are

period(3,month) after interval(10 ...11,month)

period(1,month) after (period(2, month) around interval(5...5,month))

period(24,year) after t(17,7,1970) ... t(17,7,1970).

- an atomic formula (AF) is
  - T and F are atomic formulae
  - if $t_1, \ldots, t_n$ are logical terms and $p_i^n \in \mathcal{T}_p$, then $p_i^n(t_1, \ldots, t_n)$ is an atomic formula
- a single event description (SED) is an AF $p$, if $p \neq$ T and $p \neq$ F
- a composite event (CE) is recursively defined as
  - a SED

- $p$ & $q$, where $p$ is a SED and $q$ a CE

- an event description (ED) is either a SED or a CE

  **Example 4.1.4** - planted(grass, meadow) is a SED, while harvested(maize, high_lands) & harvested (maize,mid_hills) is a CE, and both are ED.

- an atomic temporal formula (ATF) is

  - $p$, if $p$ is an ED
  - $p$ @ $t$, where $p \in \mathcal{T}_p$, and $t$ is a pure temporal expression

  **Example 4.1.5** - time between(interval (2...4,month),period(3,month)), and harvested(maize, high_lands) @ interval(12...1,month) are ATF.

- *body* is

  - $p$ & $q$
  - $p$ or $q$
  - $r$

  where $p, q$ are bodies, and $r$ an ATF

  **Example 4.1.6** - A typical example of a body is
  time between(interval(2...4,month),period(3,month)) & harvested(maize, high_lands) @ interval(12...1,month) are ATF.

- a well formed temporal formula (WFTF) is

  - if $p$ is an ATF, then $p$ is a WFTF with an *empty body*
  - if $q$ is an ATF and $p \neq$ T,F, and $p$ is a body then

    $q \Longleftarrow p$ is a WFTF, and $q$ is called the *head*

  Note that a WFTF where its *head* is an AF, and *body* is formed only by atomic formulae corresponds exactly to a Horn clause with not temporal contents.

**Example 4.1.7** - An example of a WFTF is

harvested(Crop) @ period(D,C) after T
← planted(Crop) @ T.

## 4.2 Enhanced Meta-interpreter for Temporal Reasoning

In this section we will show the specification of the simple meta-interpreter of the **NatureTime** system. The language we will use to specify the **NatureTime** is Prolog. Our methodology will be to show clauses of a standard meta-interpreter and beside that the extended temporal version, with an explanation of each extension. Before showing the specification we will put some restriction on the syntax presented in section 4.1 for reasons of implementation.

### 4.2.1 Language Restrictions

For reasons of time we could not develop sophisticated meta-interpreter and unification algorithms that could control the binding of temporal variables in temporal atomic formulae. The problem we found was in sentences were a variable to a pure temporal expression appears in more than one ATF, in the body of a WFTF.

**Example 4.2.1** - Suppose we had in our knowledge base

```
foo(tea) @ interval(10...4).
foo(coffee) @ interval(8...2,month).
foo(rice) @ T <== foo(tea) @ T & foo(coffee) @ T.
```

In this case we want to find one interval `T` in which both `tea` and `coffee` hold some property. To do that in the way the expression is written, we would need to generate all sub-intervals for `foo(tea) foo(coffee)`, and then look for those sub-intervals that are the same [1]. However, if the cyclic interval is big, or for linear intervals, the time consumption is so high that just to find one solution, we may need a half of a day.

In the version of **NatureTime** described in this work, if one wants to express that a group of events happen throughout an interval of time, then it can only be expressed by using composite event description as specified in the syntax. Then, the example should be written as

```
foo(rice) @ T<== (foo(tea) & foo(coffee)) @ T.
```

What follows from this observation is another restriction: a variable of a pure temporal expression can appear just once in the body of a WFTF. This means that if it appears again, this will only be in standard Prolog goal.

### 4.2.2 Meta-interpreter Specification

Our meta-interpreter is basically an extension of the one presented by [Sterling & Shapiro 86]. The extensions are the introduction of the logical connective to represent conjunction and disjunction; as we have temporal Horn clause, we will change the predicate `clause` for a convenient representation for temporal Horn clause; and finally the introduction of a unification predicate to unify the pure temporal expressions. This result is shown in figure 4–1

In figure 4–1, shows the standard Prolog meta-interpreter and the enhanced meta-interpreter, which we call *temp_solver*. The first clause consults in the Knowledge Base (KB) to check whether we can directly deduce `A @ T1` or not. The second clause is to deal with composite events, where we want to know if there is some time interval throughout which they can all happen together. Clauses 3 and 4 are just another way to re-write the standard clauses for logical conjunction and disjunction, and so do not need any detailed explanation. The last clause is the clause for logical implication, which gives an alternative

[1]We implemented such simple version but we decide do not put in this work.

| (a) | (b) |
|---|---|
| `solve(true).` | `solve(A @ T1,A @ T3):-` |
| | `    \+ A = (_ & _),` |
| | `    A @ T2,` |
| | `    temp_unify(T1,T2,T3).` |
| | `solve(A @ T,A @ T):-` |
| | `    A = (A1 & A2),` |
| | `    solve(A1 @ _,A1 @ T1),` |
| | `    solve(A2 @ _,A2 @ T2),` |
| | `    temp_unify(T1,T2,T).` |
| `solve((A,B)):-` | `solve(A & B, A1 & B1):-` |
| `    solve(A),` | `    solve(A,A1),` |
| `    solve(B).` | `    solve(B,B1).` |
| `solve((A;B)):-` | `solve(A or B, R):-` |
| `    solve(A);` | `    solve(A,R);` |
| `    solve(B).` | `    solve(B,R).` |
| `solve(A):-` | `solve(A @ T1, A @ T3):-` |
| `    clause(A,B),` | `    A @ T2 <== Precondition,` |
| `    solve(B).` | `    solve(Precondition,_),` |
| | `    temp_unify(T1,T2,T3).` |

**Figure 4–1:** (a) - Simple meta-interpreter for pure Prolog; (b) - Enhanced meta-interpreter for **NatureTime.**

for the first two clauses if they fail. To understand their semantics we need to understand the declarative interpretation of each one, and they should be interpreted as follows.

1 - `A` *is true throughout* `T1`, *if* `A` *is not a composite event, and* `A` *is true throughout* `T2`, *and* `T3` *is the temporal unification of* `T1` *and* `T2`, *given by the procedure* `temp_unify`.

2 - `A` *is true throughout* `T` *if, A is a composite event consisting of* `A1 & A2`, *and A1 is true sometime in* `T1`, *and A2 is true sometime in* `T2`, *and* `T` *is the temporal unification of* `T1` *and* `T2`, *given by the procedure* `temp_unify`.

5 - `A` *is true throughout* `T1` *if, the temporal* `A @ T2` *is the head of a temporal Horn clause with body* `Precondition`*, and the body can be solved, and* `T3` *is the temporal unification of* `T1` *and* `T2`*, given by the procedure* `temp_unify`.

**Example 4.2.1** - Suppose we have the following sentences in one KB

```
foo(corn) @ interval(5...6,month) .
foo(tea) @ interval(2...10, month).
foo(coffee) @ interval(8...4,month).
foo(rice) @ I <== (foo(tea) & foo(coffee)) @ I.
```

Then the answer for the following queries are (from now on the symbols ":|" and ">>" represent the query and answer prompts, respectively, of the **NatureTime** system).

```
:| foo(corn) @ interval(6...9,month).
>> foo(corn) @ interval(6...6,month)
:| foo(corn) @ interval(3...5,month).
>> foo(corn) @ interval(5...5,month)
:| more.
>> Sorry, no more solution is possible.
|: foo(rice) @ T.
>> foo(rice)@interval(8...10,month)
|: more.
>> foo(rice)@interval(2...4,month)
|: more.
A:  Sorry, no more solution is possible!
```

The `more` command is just a special command of **NatureTime** to allow all possible solutions by backtracking(analogous to the ";" command in standard Prolog).

Note, in clause 5, that we are not interested in the solution for the body `Precondition`. The reason is because if T2 is also in the body, then it must be instantiated when solving the body.

To complete what we will call *time-solver*, we introduce the solve clauses for negation, Prolog goals. This addition can be seen in figure 4–2.

```
solve(~ A,_):- !,                  % ~ A is true if it
   \+ solve(A,_).                  % cannot be solved
solve(A,A):-                       % A is true if it is
  \+ temp_formula(A),              % not a temp_formula
   A.                              % and it succeds
```

**Figure 4–2:** The additional clause to *time-solver*

Only the last clause needs a simple explanation. Its interpretation is

*A is true if it is not a temporal formula and it is a Prolog goal that is true.*

## 4.3 Temporal Unification

In this section we will show the main parts of the unification process performed by **NatureTime**. This unification is basically divided in two parts, *reduction* and *unification of units*. Deduction transforms complex pure temporal expressions in to canonical forms of temporal units, cyclical and linear intervals. Unification performs the matching between canonical forms according to the matching relations between intervals. Before showing them, we will specify the unification process.

### 4.3.1 Basic Unification Algorithm

The basic unification algorithm is very simple as shown in figure 4–3, because it just call the reduction and the unification of temporal units.

The semantics of such clauses are given as follow.

1. *Two terms unify if they are the same the term* `Term`*, and* `UTerm` *can be reduced to* `RUTerm`*, and* `Term` *can be reduced to RUTerm*

```
temp_unify(Term,Term,UTerm):-
   reduce(UTerm,RUterm),
   reduce(Term,RUTerm).
temp_unify(Term1,Term2,UTerm):-
   \+ Term1 = Term2,
   reduce(Term1,RTerm1),
   reduce(Term2,RTerm2),
   reduce(UTerm,RUTerm),
   unify_units(RTerm1,RTerm2,RUTerm).
```

**Figure 4–3:** Clauses for the `temp_unify` procedure.

2. `Term1` *unifies with* `Term2` *if,* `Term1` *is not equal to* `Term2`, *and* `Term1` *can be reduced to* `RTerm1`, *and* `Term2` *can be reduced to* `RTerm2`, *and* `UTerm` *can be reduced to* `RUTerm`, *and the unification of* `RTerm1` *and* `RTerm2` *is* `RUTerm`.

Note that in both clauses we reduce the `RUTerm`. It is done in this way to allow the possibility to have complex temporal expressions in the `UTerm` as well. This facility allow the unification of expressions like `period(1,month) after interval(2...4)` with `period(1,month) after (period(1,month) around interval(3...3,month)`.

### 4.3.2 Reduction of Pure temporal Expressions

We will divide the reduction clauses in to two groups. The first consists of the clauses to deal with canonical forms of temporal expression. The second, deals with complex forms.

```
reduce(X,X):-
   time_var(X), !.
reduce(T1...T2,T1...T2).
reduce(interval(S1...T1,Class),interval(S1...T1,Class)).
reduce(period(Delta,Class),period(Delta,Class)).
```

**Figure 4–4:** Clauses of `reduce` for canonical forms of temporal expressions.

```
reduce(around(Step,T1),Result):-
   reduce(T1,T2),
   expand_unit(T2,Step,Result).
reduce(after(Delta,T1),T2):-
   reduce(T1,TExp1),
   reduce(T2,TExp2),
   future(TExp1,Delta,TExp2).
```

**Figure 4–5:** Clauses of `reduce` for complex forms of temporal expressions.

In figure 4–4 we have the clauses to deal with canonical forms of temporal expressions. For each one we have the following interpretation:

1. `X` *is reduced to itself if it is a temporal variable, given by* `time_var`. The cut (`!`) is to guarantee that a variable can only be reduce to a variable. The `time_var` interpretation is *a term is a temporal variable if it s a Prolog variable; or a conical form where all its elements are Prolog variables*;

2. *a linear interval* `T1...T2` *is reduced to itself*;

3. *a cyclical interval of time is reduced to itself.*

4. *a period of time is reduced to itself*

Note that we do not worry whether in an expression `T1...T2`, both terms are instants of time or not. However as specified in the syntax of the language, section 4.1, in a expression like `T1...T2`, `T1` and `T2` must be instants of time. and we will assume that a type checking to verify it was already performed by some program of time dialog that reads one sentence, and verifies whether it is a well formed temporal formula, or not. (This was implemented as part of the **NatureTime**, see appendix for details).

Figure 4–5 shows the clauses for complex temporal expressions. For the scope of this project we identify just two temporal operators, and their semantics are given as follow.

```
unify_units(T1...T2,T3...T4,TR):-
   linear_unify(T1,T2,T3,T4,TR).
unify_units(interval(I1,C),interval(I2,C),interval(IR,C)):-
   mod_temp_class(_,C,M),
   integer(M),
   modular_unify(I1,I2,IR).
```

**Figure 4–6:** The first two clauses for the `unify_units` procedure.

1. *is true if* `T1` *can be reduced to* `T2`*, and* `T2` *can be expanded by* `Step` *according to the definition 3.12 resulting in* `Result`. Here, we do not need to specify the `expand_unit` because it just implements that definition.(Code is in the appendix)

2. *is true if* `T1` *can be reduced to* `TExp1`*, and* `T2` *reduced to* `TExp2`*, and* `TExp2` *is the future of* `TExp1` *after* `Delta`*, according to definition 3.13.* Once again the detailed implementation is in the appendix.

### 4.3.3 Unification of Units

The unification of units is based on the matching relations between intervals, and it is done by the `unify_units` procedure that also perform the matching relation between cyclical intervals and linear intervals. We will describe the set of clauses for this procedure in two groups.

The first group treats the different types of unification for linear and cyclical intervals. Figure 4–6 shows its specification.

The semantics of these clauses are given as follow.

1. *the linear intervals* `T1...T2` *and* `T3...T4` *unify, if their linear unification succeeds, producing the new linear interval* `TR`

2. *the cyclic intervals of the MTC* `C` *with ranges* `I1` *and* `I2` *unify if,* `C` *defines other MTC by the modular value* `M`*, and* `M` *is integer, and the modular unification of* `I1` *and* `I2` *succeeds, producing the range* `IR` *of a new cyclic interval of C*

The second group deals with the relation between cyclical interval with linear intervals. For this reason, certainly this is the most important part of this specification, without it, the concept of cyclical interval would be useless. This is shown in figure 4–7.

```
unify_units(interval(I,C),T1...T2,TR):-
   time_instance(I,C,T3...T4),
   linear_unify(T1,T2,T3,T4,TR).
unify_units(T1...T2,interval(I,C),TR):-
   unify_units(interval(I,C),T1...T2,TR).
```

**Figure 4–7:** The Last two clauses for the `unify_units` procedure.

The semantic interpretation of this clauses are

1. *a cyclical interval, with range* `I` *of a MTC C, unifies with a linear interval* `T1..T2` *if, the time_instance of the cyclical interval is* `T3...T4`*, and the linear unification between* `T1..T2` *and* `T3...T4` *succeeds, resulting in a new linear interval* `TR`.

2. *a linear interval* `T1...T2` *unifies with a cyclical interval, with range* `I` *of a MTC* `C`*, if the cyclical interval unifies with the linear.*

**Example 4.3.1** - `interval(3...3,month)` matches with `t(1,3,1990)...t(1,3,1990)`, `t(1,3,1980)... t(1,3,1980)`, and so on.

# Chapter 5

# Results

The objective of this chapter is to show how we can use the **NatureTime** language to represent temporal knowledge from the ecological domain, how to express a query in the language, showing the results of deduction performed for each query. Finally, we shall point out the strengths and weaknesses of the language and the meta-interpreter.

After defining the temporal classes for the set of examples of chapter 2, we will show for each example the original sentence in English followed by the corresponding expression in our temporal language. Then the original queries, and the result expected will be followed by the query in the language and the result of the deduction performed. If more than one representation is possible, due to ambiguities of representation, we will show the alternatives and compare both results with the expected results.

## 5.1 The Calendar Model of Time

In this section we will define a calendar model of time in our logic, in order to represent the units of time in terms of the temporal classes defined.

### 5.1.1 Time Structure

The time structure to be considered is defined considering only three levels of time hierarchy: day, month and year. The flow of time is determined by the highest level (year) and the lowest level (day) is considered to be the least interval. In this way we have

```
mod_temp_class(flow_time,year,infinity).
mod_temp_class(year,month,12).
mod_temp_class(month,day,30).
mod_temp_class(day,lowest_level,smallest).
```

The interpretation of the definitions are

- year is the flow of time
- year is modularly defined by 12 months
- month is modularly defined by 30 days
- day is not considered to be modularly defined by any other temporal class

There are two things to be noted. First, the modular definition of month does not match the real calendar since each month has the same number of days. The reason is because we decided to not consider such details, but to concentrate our effort on the temporal inference definitions. Any further extension in order to represent a model of time in which we can have many sorts of the same modular temporal class (MTC), as is the case for the normal calendar, must change the definition of MTC to allow different modular values. Second, the fact that day could be defined as 24 hours, and so on, does not represent any advantage for the domain investigated because none of the knowledge described refers to time units smaller than days.

## 5.2 Planting & Harvesting Example

In the Example 2.2.1 of section 2.2, the farmer example is about seasons of planting and harvesting maize and millet. Given the nature of some sentences, that express information about periods of time in different levels of temporal class (*e.g. 3 1/2, 4 months*), this example can be represented in two ways. The first is a simplified version and second is a more representative version because much less of the original information is missed due to the language limitations.

### 5.2.1 Simplified Version of Representation

A simplified version of the farmer example does not consider information that relates different levels of temporal class, as we will see in this section.

#### From English to NatureTime

- *It (maize) is planted around June, July.* - as the interval is explicitly mentioned we do not need the around operator, and so this sentence is represented as

  ```
  planted(maize) @ interval(6...7,month).
  ```

  As the original sentence refers to the months as a whole, the representation captures the interpretation that in every day during both months it is true that the event happens. Furthermore, since June and July are cyclical temporal entities, the general interval also captures such cyclicity, because there are instances of both months in every year.

- *Maize takes about 3 and half, four months to mature.* Due the context, we can assume that this sentence refers to the average time of maturation, and in the sentence

  *But in the mid hills it normally takes 3 1/2 to 4 months from planting to harvesting.*

  we have the same information. Then, without considering the "1/2" factor we have

  ```
  harvested(maize,mid_hills) @ period(N,month) after T
            <== planted(maize) @ T & between(3,4,N).
  ```

- *In the high lands it takes about 6 months, because of the temperature.* - Here, the additional information about temperature is not relevant for the temporal representation. It could be attached in some way to the event of planting maize in the high lands. Then we have

  ```
  harvested(maize,high_lands) @ period(6,month) after T
            <== planted(maize) @ T.
  ```

- *There are varieties which are grown in the lower ares, where temperature is quite high, so in the lower areas even two months is enough.* - Once again the temperature information is not relevant for temporal representation purposes. Then we have

  ```
  harvested(maize,low_areas) @ period(2,month) after T
            <== planted(maize) @ T.
  ```

- *When maize is about 2 1/2 or 3 months after sowing, then at this time the farmers start planting millet.* - As before, we assume 2 instead of 2 1/2 months, and sowing can be changed to planting.

  ```
  planted(millet) @ period(N,month) after T
            <== planted(maize) @ T & between(2,3,N).
  ```

- *Millet takes about 3 months to mature.* - It is represented as

  ```
  harvested(millet) @ period(3,month) after T
  ```

```
<== planted(millet) @ T.
```

The types of queries related with this example that were pointed out can be expressed as follows.

- *what are the months of maize harvesting, in general (mid hills)?* The answers should be *from September to November*. In our language such a query can be made as follows.

```
|: harvested(maize,mid_hills) @ T.
>> harvested(maize,mid_hills)@interval(9...10,month)
|: more.
>> harvested(maize,mid_hills)@interval(10...11,month)
|: more.
A:  Sorry, no more solutions are possible!
```

  The complete answer appears if we ask for more solutions because of the variable N in the rule, which can assume two value. This also the reason why there are answers which appear twice. Note that the query has been solved by inferring the harvesting date from the plating date.

- *In which months can maize be harvested in the high lands?.*

```
|: harvested(maize,high_lands) @ T.
>> harvested(maize,high_lands)@interval(12...1,month)
A:  Sorry, no more solutions are possible!
```

- *In which months can maize be harvested in the low areas?.*

```
|: harvested(maize,low_areas) @ T.
>> harvested(maize,low_areas)@interval(8...9,month)
|: more.
A:  Sorry, no more solutions are possible!
```

- *In which months can millet be planted?*

```
|: planted(millet) @ T.
>> planted(millet)@interval(8...9,month)
|: more.
>> planted(millet)@interval(9...10,month)
A:  Sorry, no more solutions are possible!
```

- *In which months can millet be harvested?*

```
|: harvested(millet) @ T.
>> harvested(millet)@interval(11...12,month)
|: more.
>> harvested(millet)@interval(12...1,month)
A:  Sorry, no more solutions are possible!
```

  Note that here two inference steps have been required

### 5.2.2 More Representative Version

A more "realistic" representation should capture all the relevant information available in as much detail as possible. In this example, if we want to achieve it we need to represent the informations like *3 1/2 to 4 months*. In the current example, there are two sentences in this case, and they could be represented as follows.

#### From English to NatureTime

- `planted(millet) @ period(15,day) plus period(2,month) after T`

  `          <== planted(maize) @ T.`

  Note that any inference to deduce the time in which millet can be harvested, will produce a time representation $t(t_1, \ldots, t_k)$ because of the composition of periods in different levels of the time hierarchy.

- Now, we need to use two clauses because the original knowledge expresses a variable period of time.

```
harvested(maize,mid_hills)) @ period(15,day) and
          period(3,month) after T <== planted(maize) @ T.
harvested(maize,mid_hills)) @ period(4,month) after T
          <== planted(maize) @ T.
```

Now the queries are the same, but due to the different representation for maize (in the mid hills) and millet we have more expressive answers for the queries related with them.

- ```
  |: harvested(maize,mid_hills) @ T.
  >> harvested(maize,mid_hills)@t(16,9,_3020)...t(15,11,_3020)
  |: more.
  >> harvested(maize,mid_hills)@interval(10...11,month)
  |: more.
  A:  Sorry, no more solutions are possible!
  ```

  The first answer we have a different time unit because of the composite period of time. The meaning of this answer is *in the mid hills, maize can be harvested from 16th September up to 15th November, every year.*

- In which time can millet be planted?

  ```
  |: planted(millet) @ T.
  >> planted(millet)@t(16,8,_3187)...t(15,10,_3187)
  |: more.
  A:  Sorry, no more solutions are possible!
  ```

  The meaning for this answer is, for example, is *millet can be planted from 16th August up to 15th October, every year.*

- *In which months can millet be harvested?.*
  ```

```
|: harvested(millet) @ T.
>> harvested(millet)@t(16,11,_4929)...t(15,1,_4929+1)
|: more.
A:  Sorry, no more solutions are possible!
```

The meaning for such answer is *millet can be harvested from 16th November of one year up to 15th January of the next year.*

## 5.3 Fruit Pie Example

### 5.3.1 Representation

#### From English to NatureTime

The example 2.2.2, has a very simple representation, as we will see in the following sentences.

- *Blackberries are ready to eat from the beginning of August till the end of October.*

  Here, since it mentions from the beginning of one month to the end of other it is expected that both are included, and so we need to represent the time interval just in one level of hierarchy. Then we have

  ```
  eaten(blackberries) @ interval(8...10,month).
  ```

- *Apples are harvested on September 1st and will keep for 6 months.*

  Now, we must take care because the period mentioned cannot be captured by the time unit period of the language. The reason is because we would need a temporal operator to represent the process of. However, we can simply transform such information into a time interval. Including September, and the last month as variable, with the constraint that the length of such interval is 6 months. In this way we have

  ```
  eaten(apples) @ interval(9...Y,month)
  <== time_between(interval(9...Y,month),period(6,month)).
  ```

As we saw in section 2.2.2 we need two steps to answer our question. However, as we can also make Prolog queries in the language, we can use the relation between interval and period as a query to be answered. Then we have

*For how long can I eat blackberry and apple pie?*

Can be expressed in our language as

```
|: (eaten(blackberries) & eaten(apples)) @ T & time_between(T,P).
>> (eaten(blackberries)&eaten(apples))@interval(9...10,month) &
   time_between(interval(9...10,month),period(2,month))
|: more.
A:  Sorry, no more solutions are possible!
```

The meaning for such answer is *blackberries and apples can be eaten throughout the in the interval from September up to October, and the length of time between September and October is the period of two months* Note here that we simplified the knowledge, since what we have is one specific time of picking the apple and the period in which it allow our logic to deal with both points and intervals it is not possible yet, even with the simplification the answer obtained was quite satisfactory.

## 5.4 Vegetables Salad Example

The example 2.2.3 is a more complicated example not just because of the possibility of two versions of representation, but because of the type of query asked as we will see. Now, the problem of representation is related to the information about intervals that start and finish with specific days of months, and so we cannot use the general interval as defined for only one level, but the representation of collection of intervals.

### 5.4.1 Simplified Representation

In the simplified version of this example we simply ignore the information about days and model every thing as months.

**From English to NatureTime**

- *Radishes take 1 month from planting to harvest* - Here we have one rule that relates the time of planting with the time of harvesting by an after time operation and the period of time between both events. Then we have

  ```
  harvested(radishes) @ period(1,month) after T

            <== planted(radishes) @ T.
  ```

  As we do not have any time information of when radishes can be planted, we can assume that they can be planted at any time. So, we have

  ```
  planted(radishes) @ T.
  ```

- *lettuces take 2 months* - The knowledge about lettuces are similar to that about radishes, and so we have

  ```
  harvest(lettuces) @ period(2,month) after T

            <== planted(lettuces) @ T.
  ```

Analogously, we need to introduce the sentence

```
planted(lettuces) @ T.
```

- *tomatoes take 6 months (if planted after 1st December up to 1st May)* - Here we have two possible simplifications. First, given that the day of December is the first, we can assume the whole month, and that May is not in the interval since only its 1st day is mentioned. Second, instead of including the time of planting in the rule that relates harvesting and planting,as in the original sentence we assume that the time of planting is just one factual sentence, *i.e.* not one rule as follow:

```
planted(tomatoes) @ interval(12...4,month).
```

  Now, we need one rule to express the relation between plant and harvest

```
harvested(tomatoes) @ period(6,month) after T
          <== planted(tomatoes) @ T.
```

- *cucumbers take 4 months (if planted after 1st February up to 1st June)*

  Here the situation is the same as in the case of tomatoes, and can be represent as follows:

```
planted(cucumbers) @ interval(2...5,month).
```

  Analogously, we need the sentence

```
harvested(cucumbers) @ period(4,month) after T
          <== planted(cucumbers) @ T.
```

- *and carrots take three months at any time of year.*

  Here is explicitly said that we can plant carrots at any time. Then we have the following sentences

```
planted(carrots) @ T.

harvest(carrots) @ period(3,month) after T
          <== planted(carrots) @ T.
```

As explained in 2.2.3 we need to solve each query as follow

- *when can tomatoes and cucumbers be harvested together?*
- *when should tomatoes, cucumber, radishes, lettuces and carrots be planted in order they can be harvested all together?*

As we saw before, the second should be expressed as a sequence of implication relations by using the IMPLY relation (see definition 3.19) in such way that we can find the time of planting in the chain of reasoning. The correct query should be.

```
|: harvested(tomatoes) @ T & harvested(cucumbers) @ T &
     imply(planted(carrots) @ T1, harvested(carrots) @ T) &
     imply(planted(lettuces) @ T2, harvested(lettuces) @ T) &
     imply(planted(radishes) @ T3,harvested(radishes) @ T) &
     imply(planted(tomatoes) @ T4,harvested(tomatoes) @ T) &
     imply(planted(cucumbers) @ T5, harvested(cucumbers) @ T).
>> (harvested(tomatoes)&harvested(cucumbers))@interval(6...9,month) &
   harvested(carrots)@interval(6...9,month)
   CAN HAPPEN IN THAT TIME IF
   planted(carrots)@interval(3...6,month)
 &
   harvested(lettuces)@interval(6...9,month)
   CAN HAPPEN IN THAT TIME IF
   planted(lettuces)@interval(4...7,month)
 &
   harvested(radishes)@interval(6...9,month)
   CAN HAPPEN IN THAT TIME IF
   planted(radishes)@interval(5...8,month)
 &
   harvested(tomatoes)@interval(6...9,month)
   CAN HAPPEN IN THAT TIME IF
```

```
   planted(tomatoes)@interval(12...3,month)
 &
   harvested(cucumbers)@interval(6...9,month)
   CAN HAPPEN IN THAT TIME IF
   planted(cucumbers)@interval(2...5,month)
:| more.
A: Sorry, no more solutions are possible.
```

Its meaning is *tomatoes and cucumbers can be harvested together from June up to September. Assuming that carrots, lettuces radishes and are also harvested throughout that same interval, then all can be harvested throughout that interval if,*

- *carrots are planted from March up to June*
- *lettuces are planted from April up to July*
- *radishes are planted from May up to August*
- *tomatoes are planted from December up to March*
- *cucumbers are planted from February up to May*

### 5.4.2 More Representative Version

There are only two sentences that need to be changed if we want a more "realistic" representation of the temporal knowledge of this example. They are related with the interval of planting tomatoes and cucumbers that can be expressed as.

#### From English to NatureTime

- `harvest(tomatoes) @ period(6,month) after t(2,12,X)...t(1,5,X+1) <== planted(tomatoes) @ t(2,12,X)...t(1,5,X+1).`

- `harvest(cucumbers) @ period(4,month) after t(2,12,X)...t(1,5,X+1)`
  `<== planted(cucumbers) @ t(2,2,X)...t(1,6,X).`

The query is the same as before, but now the answer is more accurate

```
>> (harvested(tomatoes)&harvested(cucumbers))@t(2,6,_15896+1)...t(1,10,_15896+1) &
   harvested(carrots)@t(2,6,_15896+1)...t(1,10,_15896+1)
   CAN HAPPEN IN THAT TIME IF
   planted(carrots)@t(2,3,_15896+1)...t(1,7,_15896+1)
 &
   harvested(lettuces)@t(2,6,_15896+1)...t(1,10,_15896+1)
   CAN HAPPEN IN THAT TIME IF
   planted(lettuces)@t(2,4,_15896+1)...t(1,8,_15896+1)
 &
   harvested(radishes)@t(2,6,_15896+1)...t(1,10,_15896+1)
   CAN HAPPEN IN THAT TIME IF
   planted(radishes)@t(2,5,_15896+1)...t(1,9,_15896+1)
 &
   harvested(tomatoes)@t(2,6,_15896+1)...t(1,10,_15896+1)
   CAN HAPPEN IN THAT TIME IF
   planted(tomatoes)@t(2,12,_15896)...t(1,4,_15896+1)
 &
   harvested(cucumbers)@t(2,6,_15896+1)...t(1,10,_15896+1)
   CAN HAPPEN IN THAT TIME IF
   planted(cucumbers)@t(2,2,_15896+1)...t(1,6,_15896+1)
```

Its meaning is *tomatoes and cucumbers can be harvested together from 2nd June up to 1st October. Assuming that carrots, lettuces radishes and are also harvested throughout that same interval, then all can be harvested throughout that interval if,*

- *carrots are planted from 2nd March up to 1st July*
- *lettuces are planted from 2nd April up to 1st August*

- *radishes are planted from 2nd May up to 1st September*
- *tomatoes are planted from 2nd December of last year up to 1st April of the next year*
- *cucumbers are planted 2nd February up to 1st June*

## 5.5 The Christmas Tree Example

### 5.5.1 Representation

#### From English to NatureTime

The 2.2.4 requires a representation of the growing of a tree. As pointed in the exposition of this example, this needs to be represented by recursive formulas rather than imperative loop structure. If we break the knowledge included in that question, then we have the following representation.

- *a tree planted in 1990* - is represented as

  ```
  planted(tree) @ t(D,M,1990)...t(D,M,1990).
  ```

- *If a tree is 1m tall at 1 year old, adds 25 % to its height, is pruned by 1m per year when its height is higher than 10m...* - is represented by the following clauses

  ```
  height(tree,1) @ period(1,year) after T <== planted(tree) @ T.
  height(tree,H) @ period(1,year) after T
        <== height(tree,H1) @ T &
            (H1 >= 1)           &
            (H1 =< 10)          &
            (H is H1*1.25).
  height(tree,H) @ period(1,year) after T
        <== height(tree,H1) @ T  &
  ```

```
            (H1 > 10)            &
            (H is H1*1.25 - 1).
```

To show how the system simulate the growing of the tree we have the following query.

```
|: height(tree,H) @  t(25,12,Y)...t(25,12,Y).
>> height(tree,1)@t(25,12,1991)...t(25,12,1991)
|: more.
>> height(tree,1.25)@t(25,12,1992)...t(25,12,1992)
|: more.
>> height(tree,1.5625)@t(25,12,1993)...t(25,12,1993)
|: more.
>> height(tree,1.953125)@t(25,12,1994)...t(25,12,1994)
```

Note that every time `more` is asked the height given is that of the next year. Now, the query *which year will I have a 20m Christmas Tree on December 25th?* - is represented as

```
height(tree,H) @ t(25,12,Y)...t(25,12,Y) & (H >= 20).
```

However, due to the re-calculation of the height (which is done in two recursive sentences), this takes a long time to run. We have satisfactory time waiting until height 11, as shown in the following query

```
|: height(tree,H) @ t(25,12,Y)...t(25,12,Y) & (H >= 11).
>> height(tree,11.641532182693481)@t(25,12,2002)...t(25,12,2002) &
   11.641532182693481>=11
```

After this, we have to wait a long, long time.

This type of inefficiency in logic programs is discussed in [Robertson *et al.* 91], where a technique of recording the results of satisfying particular goals is shown. This solution was not tested due to the short time we had, but it can be introduced in future versions of **NatureTime**.

## 5.6 Grass for Hay Example

### From English to NatureTime

The example 2.2.5 can be represented as follow

- *hay can be made* throughout *April up to August.* This sentence is about the hay season, not necessarily related with the event of harvest grass for hay. It only say the best time to have hay, or to make hay, that depends of harvesting. Then we can have.

  ```
  make(hay) @ interval(4...8,month).
  ```

- *Grass can be harvested 2 months* after *it starts growing* - can be represented as, for example

  ```
  harvested(grass,hay) @ period(2,month) after T
  <== grow_free(grass,meadow) @ T.
  ```

  Note, that "free" here is just to say that there are no sheep eating the grass, and so the grass is free of its "predator".

- *sheep can use the meadow* throughout *October up to May* - is represented as

  ```
  can_use(sheep,meadow) @ interval(10...5,month).
  ```

  Now, we need to say that sheep only use the meadow when it can use, that is represented as

  ```
  use(sheep,meadow) @ T <== can_use(sheep,meadow) @ T.
  ```

- *grass start to grow for hay 1 month after the last month in which sheep have used the meadow* - is represented as

  ```
  grow_free(grass,meadow) @ period(1,month) after interval(X...X,month)
  <== use(sheep,meadow) @ interval(Z...X,month).
  ```

Finally, our first query *when will I be able to bring my first crop of hay in?*

```
|: harvested(grass,hay) @ T.
>> harvested(grass,hay)@interval(8...8,month)
|: more.
A:  Sorry, no more solutions are possible!
```

Its meaning is *grass can be harvested for hay throughout the whole month of August*

As expected, no second crop is possible if we consider that the chain of reasoning just finds the interval that sheep can use the meadow as a definite one, without the possibility to change. However, we can force the interval of harvesting to verify if the sheep can be out of the meadow, since their interval of use of it is not a definite one. Using the IMPLY relation, the queries *Can I get a second crop of hay? If I want two crops of hay, when is the latest that I should move my sheep off the meadow?*, can be represented as follow, with the answer shown after that.

```
|: (harvested(grass,hay) & make(hay)) @ T &
     imply(grow_free(grass,meadow) @ T1,harvested(grass,hay) @ T) &
     imply(grow_free(grass,meadow) @ T2,harvested(grass,hay) @ T1) &
     make(hay) @ T1 &
     imply(use(sheep,meadow) @ T3,grow_free(grass,meadow) @ T2).
>> (harvested(grass,hay)&make(hay))@interval(8...8,month) &
   harvested(grass,hay)@interval(8...8,month)
   CAN HAPPEN IN THAT TIME IF
   grow_free(grass,meadow)@interval(6...6,month)
 &
   harvested(grass,hay)@interval(6...6,month)
   CAN HAPPEN IN THAT TIME IF
   grow_free(grass,meadow)@interval(4...4,month)
 &
   make(hay)@interval(6...6,month) &
```

```
   grow_free(grass,meadow)@interval(4...4,month)
   CAN HAPPEN IN THAT TIME IF
   use(sheep,meadow)@interval(10...3,month)
|: more.
A:  Sorry, no more solutions are possible!
```

Its meaning is, *grass can be harvested for hay throughout the whole month of August, and August is an interval of time for hay season, and to be harvested in that time, grass need to be free in the meadow throughout the whole month of June, and if we also want to harvest grass for hay in June, then this time must also be one interval of hay season, and the grass need to be free in the meadow throughout the whole month of April, and if we want grass free in April, then sheep should use meadow at most until March.*

## 5.7 Analysis of the Results

In this section we shall make some analysis of the results obtained by representing the ecological problems of chapter 2, using the NatureTime system. There are two points to be analysed. First the expressive power of NatureTime to represent temporal knowledge and seconds is the temporal reasoning that can be performed by such system.

### 5.7.1 Representation

#### Strengths of the System

In terms of representation, **NatureTime** seems to well represent cyclical knowledge by using the concept of cyclical intervals. Such representation allows good representation of ecological seasons as in the sentence

```
planted (maize) @ interval (6...7,month)
```

With such facility we do not need to say that it can occurs every year, since the MTC month is included in the year definition, and year is flow of time.

Another advantage is the relation with such abstract concept with flow of time represented by t(D,M,Y) where the third element is the only that include all others in linear flow.

The temporal operator after seems to well represent the relation between past and future, the labels used by the language allows a legible reading of temporal statements, even in the case of relative past and future, or absolute past and future.

Another good thing in **NatureTime** is to allow the declarative representation of ecological modeling as in the case of the Christmas Tree.

### Weaknesses of the System

The language can not deal with specific points in time that, according to many authors, is quite clear that some statements refers to point in time. In the example of Fruit Pie, the sentence *apple is picked on 1st September, and can be eaten for 6 months* seems to be related with a specific point in time and with an interval in the future. Instead of the representation we did, where we assumed that Apple is good to be eaten in the exact moment it is picked, we could have the dual temporal operator of after, *i.e.* before, and may be the following representation would be better.

```
picked(apples) @ t(1,9,X)...t(1,9,X).
eaten(apples) @ X...Y
  <== picked(apples) @ period(1,day) before X...X &
      time_between(X...Y,period(6,month).
```

Note that even in this case we use an interval to represent one single point. A deep study may be can show that have both points and intervals does not really represent a great advantage, but makes the language much more complicated. So the weakness here is related with the absence of an operartor like before rather than the impossibility to have points.

The language does not permit to define *sorts* of MTC to allow differents modular values for the same MTC. As said in section 5.1.1, we cannot represent the real calendar model of time because of it.

Another weakness of the representation is that questions like *at least, at most* must be interpreted by the user when putting variables in the ending points of intervals. For example in the hay example, the rule

```
grow_free(grass,meadow) @ period(1,month) after interval(X...X,month)
hspace2cm <== use(sheep,meadow) @ interval(Z...X,month).
```

can interpreted as *grass is free to grow up 1 month after X if sheep use the meadow <u>at most</u> until X*. Note that the reading of the sentence does not necessarily mean at most, although we can interpret in this way.

Due the restriction of the language, section 4.2.1, we have to group all events that we want to say they happen at the same time interval.

### 5.7.2 Temporal Reasoning

#### Strengths of the System

In terms of temporal reasoning the system has shown to satisfactorily solve some temporal deductions about temporal knowledge in the domain of the examples used.

The introduction of temporal operators *after, around*, leaves the user free to write his/her inference rules without the necessity of using properties and relations of temporal units. Without a deep analysis, we have not found such kind of mechanisms in the temporal reasoning systems we could read about, as in the **TIMELOGIC** system [Koomen 87].

The system can deal with complex temporal expressions and reduce them to simple forms of interval.

The system allows the user, if he/she wants, to access predicates to reason about properties of intervals, *e.g.* `future/3`, `is_future/2` (linear intervals only).

Another good point reached is the possibility to perform temporal deduction on simulations in a declarative way, as shown in the Christmas Tree example.

The language allow the implication relation IMPLY, that provides a good way to obtain answer for queries in the form "Can we deduce A $\rightarrow$ B from a knowledge base K?"

#### Weaknesses of the System

Although we can obtain good temporal deductions, the size of pure temporal expressions for simulation problems makes the process of unification be "heavy" due the re-calculation of them. As a consequence, time consumption is still high and an improvement, as suggested in section 5.5, is really necessary.

The IMPLY relation is still too simple to be considered a "panacea" to solve all kinds of queries that it can solve. For instance, we cannot successfully use it in the Planting & Harvesting, section 5.2, because of the clause

```
planted(millet) @ period(N,month) after T

            <== planted(maize) @ T & between(2,3,N).
```

The body of the clause has a conjunction, and as the reader can can see in the IMPLY definition in section 3.1.6, this kind of expression is not handled yet.

Due to the simplicity of our unification algorithm, the system still cannot deal with queries like the following.

Suppose we have in a KB the sentence `foo @ interval(1...6,month)..`

```
|: foo @ T1 & time_between(T1,period(3,month)).
>> Sorry, no solution is possible.
```

However, we would expect the system give at least one sub-interval, the other could be obtained by backtracking, with period 3 months, since the query asks if there is a 3 month interval throughout which foo holds. A careful and sophisticated unification algorithm is really needed.

# Chapter 6

# Conclusions and Future Works

The main goal of the project has been reached, and a new theory of time representation seems to emerge, since the **NatureTime** system has shown to be quite useful to reason about seasonal cycles and temporal conflicts. Although the ecological domain investigated does not represent all kind of ecological knowledge, it shows a lot of temporal features that many other ecological and non ecological domains may possess.

The language defined provides a comparatively simple way to define a model of time, to represent temporal knowledge, and to make queries about such knowledge, although it also cannot represent some temporal statements.

The results with the ecological examples has shown that the theory can solve a quite good number of problems of temporal conflicts and cyclicity of some events, although it needs to be full developed, and is also necessary a deep analysis comparing it with other temporal logics and time ontologies.

The **NatureTime** system could be used as a specialist temporal reasoner in the context of the IK project, although some warnings of representation should also be provided. However, to be used in real applications considerably more is needed, in order to full develop the time theory in which it is based, and a careful study of the unification algorithm is also needed so that the language can be used with as few restrictions as possible.

As a consequence, we suggest the following possible extensions to this work:

1. Theoretical - concerns with the deep development of the time theory presented, comparing its real advantages and disadvantages is respect to the existent time theories. Basically, this can be, for example

   - redefine the **temp_mod_class** predicate to allow members of one class to have different modular values, according to a predetermined range. As a consequence, redefine all modular operations that uses this relation;
   - introduce the dual connective of **@**, "always", *i.e.* "sometimes";
   - investigate the necessity for more temporal operators, rather than **after**, **around**;
   - develop a formal semantics of the logic to prove some properties of soundness.

2. Computational - concerns with the development of a sophisticated unification algorithm to allow a specification/implementation of the language, direct from the theory. This could be

   - redefine the *temp_solver* to allow different instantiations on intervals and their sub-intervals; or
   - control the binding of variables as done by the concept of *environment*in a Warren Abstract Machine, for portability and efficient implementation; or
   - try another way to provide different bindings.

3. Application - concerns with the investigation of the real applications of the time theory developed, to look for new problems that the theory/language cannot solve, to supply the theory with new problems to be investigated. This could involve, for example

   - investigating other ecological and non ecological domains of applications to test the theory;
   - investigating problems that deal with time granularity, and seeing if our time theory solve some of such problems

# Bibliography


[Allen & Hayes 85] James F. Allen and P. J. Hayes. A common sense theory of time. In *Proceedings of the of the International Joint Conference on Artificial Intelligence*, pages 528–531, 1985.

[Allen 84] James F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23(1), 1984.

[Barringer *et al.* 91] Howard Barringer, Michael Fisher, Dov Gabbay, and Anthony Hunter. Meta-reasoning in executable temporal logic. In *Knowldege Representation*, 1991.

[Biggs 87] Norman L. Biggs. *Discrete Mathematics.* Oxford Science Publication, 1987.

[Gabbay 87] Dov Gabbay. *Modal and Temporal Logic Programming. In Temporal Logics and their Applications - Antony Galton Editor.* ACADEMIC PRESS, 1987.

[Gabbay 89] Dov Gabbay. The declarative past and imperative future. In H. Barringer, editor, *Coloquim on Temporal Logics and Specification*, pages 409–448. Springer Verlag, 1989. V(398).

[Gabbay 92] Dov Gabbay. Temporal logic: Mathematical foundations, Part I. Technical Report MPI-I-92-213, Max-Planck Institute fur Informatik, Saarbrucken, Germany, 1992.

[Gray 93] Babak Gray. T-prolog - a temporal logic for an ecological domain. Unpublished M.Sc. thesis, Department of Artificial Intelligence/University of Edinburgh, 1993.

[Haggith *et al.* 92] Mandy Haggith, Dave Robertson, Dan Walker, Fergus Sinclair, and Robert Muetzelfeldt. TEAK - tools for eliciting agroforestry knowledge. In *British Computer Society Symposium of IT - Enabled Charge in Developing Countries*, 1992.

[Knight & MA 94] B. Knight and J. MA. Time representation: A taxonomy of of temporal models. *Artificial Intelligence Review*, 7(6), 1994.

[Koomen 87] Johaness A.G.M. Koomen. The TIMELOGIC temporal reasoning system in common lisp. Technical Report Technical Report 231, Department of Computer Science - University of Rochester, Rochester, USA, 1987.

[Koomen 89] Johaness A.G.M. Koomen. Reasoning about recurrence. Technical Report Technical Report 307, Department of Computer Science - University of Rochester, Rochester, USA, 1989.

[Robertson *et al.* 91] David Robertson, Alan Bundy, Robert Muetzefeldt, Mandy Haggith, and Michael Uschold. *Eco-Logic Logic-Based Approaches to Ecological Modelling.* The MIT Press, 1991.

[Sterling & Shapiro 86] Leon Sterling and Ehud Shapiro. *The Art of Prolog.* The MIT Press, 1986.