LOGIC OF TIME

Seminar Report

Mohsin Ahmed
Guide: Dr. G.Venkatesh

Department of Computer Science and Engineering
Indian Institute of Technology, Bombay

May 15, 2018

Abstract

In this report we look at various theories of time. In particular, we present our work on the use of the rational number line to represent time.

The theory of dense linear order without end–points (DLO) can be used to represent time. This theory along with a theory of space can be used to represent trajectories of objects using an unary predicate.

To take a simpler probelm we looked at the issue of decidability of DLO with an unary predicate – the theory of DLOP – by means of quantifier elimination (QE). This path encountered several difficulties, and the problem is still open.

The problems encountered in the QE of DLOP led us to filter out the concepts necessary to represent time – which we developed in the theory of ordinal trees (OT). These trees are used to represent temporal events. Periodic as well as events that occur an unbounded number of times in an interval can be represented by our tree notation. Events (expressible in proposition logic) take place at time points and in open intervals of time.

Words in an ordinal based language can be used as an alternate representation for the trees. We then develop mechanisms to work on these trees, and investigate the expressive power of our logic. Some extensions to trees are also discussed, if feasible they may be used to design the logic. We aim for a logic that can be embedded in a conventional logic programming language, so that computation with temporal data is efficient.

Contents

1 Introduction **6**

2 Nature of Temporal Informaton **8**

2.1 Discrete Time Temporal Logics 8

2.2 Interval Time Temporal Logics 10

2.3 Requirements of a Theory of Time 11

2.4 Problems with existing theories 12

3 Dense Linear Order **14**

3.1 Axioms for DLO 14

3.2 Quantifier Elimination in DLO 14

3.3 Use of DLO as a theory of Time 15

4 Spatial Information **16**

4.1 Finite Domains 16

4.2 Plane Geometry 17

5 Motion **19**

5.1 Properties of Motion 19

5.2 A Theory for 1-Dimension Motion 19

5.3 Product Theory Construction 20

5.4 Motion in Product Theory 20

5.5 Decidability of DLOP 21

5.6 Expressibility of DLOP 23

5.7 Discussion 24

6 Ordinal Trees **26**

6.1 Representing temporal information on OTs 29

6.2 Extension to Oridnal Trees 30

6.3 Expressive power of trees 34

7 Normal Forms **35**

7.1 Ordinals 37

7.2 Reduction of Strings 38

7.3 Normalization Algorithm 39

8 Conclusion **42**

8.1 Future work 43

A Bibliography **44**

B Axioms for Plane Geometry **47**

C Classical and Nonclassical Logics **48**

C.1 Terminology 48

C.2 Classical Logics 49

C.2.1 Proposition Logic 49

C.2.2 First Order Logic 50

C.3 Nonclassical Logics 50

C.3.1 Propositional Modal Logic 51

C.3.2 Propositional Temporal Logic 51

C.3.3 First Order Temporal Logic 51

D Temporal Logic and PCP **53**

D.1 Temporal Logics 53

D.2 Two Dimension TL 53

D.3 Post Correspondence Problem 54

D.4 Coding PCP in 2D-TL 55

E Acknowledgement **58**

List of Figures

1.1 Two temporally related events 7

2.1 Possible temporal relationships between two temporal intervals 10

6.1 SPLIT node splits an interval 27

6.2 LOOP Node divides an infinite interval into an infinite sequence of sub-intervals 27

6.3 Back–loops and cross–arcs disallowed 27

6.4 Valid back-arc 28

6.5 Loop shortening 28

6.6 Bouncing Ball 29

6.7 Nested loops 29

6.8 The clock ticked till one day it stopped. 30

6.9 The ball thrown lands in the field or outside it. 31

6.10 The bouncing ball on the ledge: falls and bounces on the ground. 31

6.11 Jim ran to catch the falling ball. 32

6.12 Cause and Effect Intervals 32

6.13 Cause and Effect OT 32

6.14 Rain causes the ground to get Wet 33

6.15 Collision 33

6.16 No Collision 34

7.1 Equivalent trees: there are two ways of splitting a single interval into three parts. 36

7.2 Equivalent trees having LOOPs 36

7.3 Normal Finite Trees 38

Chapter 1

Introduction

We are looking at ways and means to represent and compute information containing temporal references. A typical example of such information is:

Example 1.1. "The bouncing ball came to rest."
There is no elegant method by which such information can be coded in the existing logic programming languages.

A query involving temporal information may require explicit mention of time in the computation.

Example 1.2. "Which train left at 1pm on 22 June 1991?"
The reference to time point is explicit as compared to: "Who was at the station when the train left?", where the reference "when the train left" is to a specific time point given through an event which occured at that time.

So it might be considered natural to simply let time be a domain whose values are the various explicit time points. Thus time is treated no differently from other domains such as the domain of individuals.

Time is a domain that has many interesting properties, and events which are temporal referents may be temporally related in many ways.

Example 1.3. "The train left the platform just as he reached the station."
This example has two events: e_1 = "the train left the platform", and e_2 = "he reached the station". The events are temporally related as shown in figure **??**.

If we treat the time domain similar to other domains such as – say – the domain of individuals then all this information has to be represented in the same way as any other information, which may lead to awkward representation and inefficiency in computations. It is far better to look for specific representation schemes for time.

Figure 1.1: Two temporally related events

Chapter 2

Nature of Temporal Informaton

Several models of time have been proposed [**?**, **?**]. The most important of them are *discrete time temporal logics* and *interval time temporal logics.*

2.1 Discrete Time Temporal Logics

These logics define time as a sequence of states, discretely ordered on the integer line. One state is taken as the current state, and other states relative to its position can be referred by temporal operators.

To express the fact that some formula holds for all future states, we qualify it by the *always* ($\Box$) temporal operator. Other temporal operators are: *sometime* ($\Diamond$), *next* ($\bigcirc$) and *until.*

We can expand propositional logic by these temporal operators to get *propostional temporal logic* (PTL). This logic is given in appendix C. Using this mechanism we can express a variety of things as seen by the examples below:

Example 2.1. "Sooner or later he will study, and then he will clear the course."

$$\Diamond(S \wedge \Diamond(C))$$

Where S stands for 'he studies' and C for 'clears the course'.

Example 2.2. "If he doesn't study now, he will never clear the course."
With the notation of example 1, the formula is:

$$\neg S \rightarrow \Box\neg C$$

Example 2.3. "Fermat's problem will either remain open forever, or eventually it will be solved."

$$\Box\neg F \vee \Diamond F$$

where F means "Fermat's problem is solved":

The time domain is not axiomatised but stated apriori. Here the $\Diamond$ operator is able to capture the notion of 'in finite time', which would not have been possible had we tried to axiomatize PTL as a theory of first order logic (FOL).

However if we had started with first order logic, and added temporal operators, we get *first order temporal logic* (FOTL) (see appendix C). In FOTL every state is described in terms of a first order domain of individuals which remains fixed over time.

First order variables are held by the usual quantifiers of FOL, moreover there are state variables ranging over individuals as a function of time-state. Some examples will clarify these concepts:

Example 2.4. "Everyone will eventually pay his fees."
Let $p(x)$ denote the fact that "x has paid his fees".

$$\forall x \Diamond p(x)$$

Here the domain of discourse is 'people' and x is a first order variable. This differs from the formula:

$$\Diamond \forall x p(x)$$

which means, eventually (someday) everyone will have had paid his/her fees.

Example 2.5. "Anyone who solves Fermat's problem will be thereafter honoured."
Let $f(x)$ and $h(y)$ denote "x has solved fermat's problem" and "y is honoured" respectively.

$$\forall x(\Box(f(x) \rightarrow \Diamond h(x)))$$

Note that $\Diamond$ is inside the scope of $\Box$, meaning $h(x)$ will occur sometime *after* $f(x)$.

Example 2.6. "The integer random variable x oscillates around zero, though its oscillations are arbitarily large."
We represent this by adding temporal operators to the theory of natural numbers:

$$\Box\Diamond(x = 0) \wedge \forall z \Diamond(x = z)$$

Though both x and z range over the integers, z is a FOL variable and x a *state–variable*. The operators $\Box\Diamond$ together mean 'infinitely often'.

Figure 2.1: Possible temporal relationships between two temporal intervals

2.2 Interval Time Temporal Logics

Many events are better described over intervals than over time points. In interval time temporal logics facts about the world are asserted over time intervals. There are thirteen possible temporal relations between two intervals (Figure **??**).

The interval operators $<$ and $\sqsubseteq$ are used to describe the relationships between periods, and the quantifiers range over periods instead of points.

$$\begin{aligned} I_1 &\overset{\text{def}}{=} (a_1, b_1) \\ I_2 &\overset{\text{def}}{=} (a_2, b_2) \\ I_1 < I_2 &\overset{\text{def}}{=} b_1 < a_2 \\ I_1 \sqsubseteq I_2 &\overset{\text{def}}{=} a_2 < a_1 \wedge b_1 < b_2 \end{aligned}$$

We look at some examples of this logic:

Example 2.7. "Jim has his music class today."
The event 'class' takes place over a period rather than a specific time point.

Example 2.8. "Jim travels to school by bus." We can describe Jim's day by the formula:

$$T < S < C$$

where T stands for 'Jim is in a bus going to school', S for 'Jim is at school', and C for 'Jim is on the bus coming from school'.

Example 2.9. "Jim played during lunch break at school."

$$P \sqsubseteq L \sqsubseteq S$$

where P denotes Jim playing, L the duration of his lunch break, and S the duration of school.

2.3 Requirements of a Theory of Time

Before we go on to develop the theory of time, let us look at some important properties of time that our theory must support:

Density This property is required especially in the theory of motion. It is frequently necessary to talk about time points between two events.

The handicap of discrete time logics is seen in the following example (we will look at it in detail later).

Example 2.10. In a finite time period, a ball made an arbitary number (unbounded) of bounces and before coming to rest.
Even if we assign a very large discrete interval to express this property, we cannot talk about another ball that was bouncing (asynchronously) in the same period.

We could treat the period of bouncing as an interval, but then we would have to express 'bouncing' extra–logically.

Homogeneity The properties of time are the same at all points, locally and globally. There are no absolute reference points. Most points are mentioned with respect to the *present* point or an event in the world by which time is measured.

Self Reflexivity Any period of time reflects the properties of the whole time line. Logically: Any open interval is isomorphic to the whole time line.

Isotropy The properties of time are independent of its direction. It is often argued that the laws of physics are reversible, and there is no proof over support one direction over the other. A partial or total order is used to relate time points. We will assign the positive direction to *future* time points.

No end–points There is neither a starting–point nor an end–point of time.

Atomicity of Time–points It is frequent to talk about “the next time point”, for example take the following utterance: “the spark (immediately) caused an explosion”. We can claim that if the spark occured now, then at the next time point there will be an explosion. What is meant by the next time point? It is usually the time point at which the *next event* occurs, and between these two time points hardly any other event occurs. Even if an event occured, to a casual observer all three events would seem simultaneous.

If time is to be represented as a finite chain of points, we must ensure there are sufficient number of points to correctly represent the temporal order of events.

Continuity It is continous in the sense that all limits of sequences of time points are also time points.

Measure If time is to be measured quantitatively, some metric must be defined on it. Usually a periodic event is used to measure time. For example: pendulum clock, atomic vibrations, and the rotation of the earth are all used to measure time.

2.4 Problems with existing theories

We tried first order temporal logic (FOTL) to represent motion. If we take any finite FOL theory along with temporal operators, we only get propositional temporal logic (PTL). However, if we take first order logic with even the simplest of temporal languages, we get undecidable theories. For example: the theory of natural numbers with the successor function was combined with temporal logic, this FOTL theory was found to capture the set of natural numbers and also finiteness. A consequence of the *compactness theorem* for any first- order-logic theory is that if it has models of arbitarily large finite size then it must have infinite models, but as the next example shows the compactness theorem does not hold here.

Example 2.11. The models of the following sentence are the class of finite models:

$$(a = 0) \wedge \forall x \Diamond (a = x) \wedge \Diamond \Box (a = 0)$$

It roughly says that the state variable a is initially zero $(a = 0)$, and for every individual x in the domain a visits it in some finite time $\forall x \Diamond (a = x)$. Even though a visits all individuals in finite time, it does not imply there are a finite number of them. Take the case of natural numbers, though every natural number is finite, there are an infinite number of them. So we add the condition, that after some finite time a finishes visiting all individuals and comes to rest at zero $(\Diamond \Box (a = 0))$.

Example 2.12. The following sentence captures the notion of natural number:

$$(a = 0) \wedge \forall x \Diamond (a = x) \wedge \Box (a = x \rightarrow \Diamond (a = s(x)))$$

Here the state variable a starts at zero ($a = 0$), for every number x, it eventually reaches that number. Not only that, after visiting any x it also visits the successor of that number ($s(x)$).

Using an unary function (succesor) we can axiomatize arithmetic. This logic is therefore undecidable and incomplete [**?**]. We next look at the theory of dense linear order without endpoints as the next candiate for the theory of time.

Chapter 3

Dense Linear Order

In this chapter we look at the first order theory of *dense linear order without endpoints* (DLO), defined with the binary predicate '$<$'.

3.1 Axioms for DLO

The theory of dense-linear-order without endpoints can be axiomatized by the following axioms:

$$\begin{aligned}
&\text{Order} && \forall xy(x < y \lor y < x \lor x = y) && (3.1)\\
&\text{Irreflexivity} && \forall x \not< x && (3.2)\\
&\text{Transitivity} && \forall xyz((x < y \land y < z) \to (x < z)) && (3.3)\\
&\text{Density} && \forall xy(x < y \to \exists z(x < z \land z < y)) && (3.4)\\
&\text{No endpoints} && \forall x \exists uv(u < x \land x < v) && (3.5)
\end{aligned}$$

If the truth of any formula in a theory can be decided mechanically in finite number of steps, the theory is called decidable [**?**, **?**, **?**]. The theory determined by the above axioms is decidable by the method of quantifier elimination (QE).

3.2 Quantifier Elimination in DLO

Quantifier Elimination decides the truth of a given closed formula of DLO by systematically eliminating quantifiers in the formula, until the formula becomes simple enough to decide by means of propositional logic.

First the given formula is put in a prenex normal form: that is all quantifiers are pulled out and the quantifier free part is in disjunctive normal form. Then the quantifiers are eliminated one at a time, starting with the innermost.

The quantifier $\forall$ is written as $\neg\exists\neg$, and the right $\neg$ pushed into the quantifier free formula. Since $\exists$ commutes with $\vee$, we need only remove the $\exists x$ quantifier from $\exists x \bigwedge_i \phi_i$.

We can also get rid of negation by noting the following consequnces of *Order axiom*:

$$\neg(x < y) \leftrightarrow (y < x \vee y = x)$$

$$\neg(x = y) \leftrightarrow (x < y \vee y < x)$$

Using the equality axioms of FOL we can remove all = from the formula. If any ϕ_i contains equality we can simplify further. We can replace $(x = x)$ by $\top$, If $x = y$ is present in the subformula $((x = y) \wedge \psi)$, we can use y as the value of x in the rest of the subformula (ψ) and remove $x = y$. then x in ψ is replaced by y.

We then get the formula in the form:

$$\exists x(\bigwedge_i (l_i < x) \wedge \bigwedge_j (x < k_j))$$

Now such an x exists if there is a non–empty interval between the lower bounds and the upper bounds, so we replace it by the equivalent quantifier free formula:

$$\bigwedge_i \bigwedge_j (l_i < k_j)$$

3.3 Use of DLO as a theory of Time

Example 3.1. "The ball makes an unbounded number of bounces in finite time"
This represents a converging sequence of points on Q. The sequence of points denote: 'the first bounce', 'the second bounce', ... The limit point denotes the time-point when the ball comes to rest.

This sequence can be used to identify the next event, but only locally. Suppose now another ball is simultaneously bouncing at a faster rate. Our old sequence can no longer globally identify the next event. Two bounces of the first ball may be separated by any number (arbitarily large) of bounces of the second ball. This requires time to be dense.

Chapter 4

Spatial Information

To discuss motion, we need a robust theory of space also. We describe the important properties of space that our theory must support:

Isotropy The properites of space are independent of the direction

Homogenous Its properties are postion invariant.

Continous and Connected between any two points there exists a continous path.

Unbounded There are no boundaries.

Measurable We can measure distance, assuming object lengths are translation invariant.

Three Dimension Space has three dimensions.

A theory of space must take into account all these properties. Various geometries have tried to define and reason about spatial properties. The most important ones are finite geometry, Euclid's axiomatic geometry, and Tarski's axiomatic geometry. The simplest approach is based on coordinates and numerical methods using trignometry.

4.1 Finite Domains

In these geometries, space has only a finite number of points. A line is formed by connecting two points, hence the number of lines is also finite. They are decidable and are useful in combinatorial problems, but are not very useful for expressing motion.

4.2 Plane Geometry

This is the usual two dimesion geometry. Various researches from Euclid to Tarski have given axioms that define the geometry of space accurately and rules of inference which can be used to obtain theorems of geometry.

To model plane geometry, we should be able to talk about points, lines, curves, distances, angles, polygons and areas. Coordinate geometry is easily modelled in the first order theory of real numbers also known as the theory of real closed fields (RCF). Though only plane geometry is discussed here, generalization to higher dimensions is possible.

A point is the basic undefined unit. The set of points in space (here $\mathcal{R}\times\mathcal{R}$) is considered to be the domain of the model. A point p will be represented by it rectangular coordinates (p_x, p_y).

Let p, q, r, s be points in $\mathcal{R}^2$, we define the following:

$$\text{distance} \quad |\overline{p\,q}| \quad \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2} \tag{4.1}$$

$$\text{equidistant} \quad \delta(p, q, r, s) \quad \overline{p\,q} = \overline{r\,s} \tag{4.2}$$

$$\text{between} \quad \beta(p, q, r) \quad \begin{array}{l} ((p_x \le q_x \le r_x) \vee (r_x \le q_x \le p_x)) \wedge \\ ((p_y \le q_y \le r_y) \vee (r_y \le q_y \le p_y)) \end{array} \tag{4.3}$$

$$\text{line} \quad l(p) \quad ap_x + bp_y + c = 0 \tag{4.4}$$

$$\text{curve} \quad c(p) \quad \text{polynomial}(p_y, p_x) = 0 \tag{4.5}$$

$$\text{region} \quad r(p) \quad \text{polynomial}(p_y, p_x) \le 0 \tag{4.6}$$

Tarski [**?**] has given a complete axiom system using the predicates δ and β, this system is given in appendix B.

This theory is axiomatizable, complete and decidable; though not finitely axiomatizable. This is because an infinite number of axioms are required to capture the completeness[1] of real numbers.

Arbitrary polygons cannot be expressed in this theory because we need to talk about sets of points. That is possible if we introduce existential monadic predicates[2] to represent arbitary finite sets of points. However this theory can interpret arithmetic, and hence is essentially undecidable.

The $Th(RCF)$ plays an important role in the axiomatization of euclidean geometry [**?**]. Most of elementary geometry is expressible in RCF. Tarski has shown this theory to be complete and decidable by the method of quantifier elimination [**?**, **?**].

[1]In the topological sense: every closed and bounded set of real numbers has a real least upper bound.

[2]Monadic second order logic.

Quantifier elimination has nonelementary time complexity (w.r.t. size of formula), as elimination of each quantifier increases the length of the formula exponentially. However the recent method of Cylinderical Algebraic Decomposition (CAD) by Collins decides RCF in $2^{2^{|\phi|}}$ time [**?**].

The L_{RCF} is $(\mathcal{R}, 0, 1, +, <, -, *)$, where $\mathcal{R}$ is the set of real numbers. Note that $\{1, 0, <, -\}$ are all definable in $\mathcal{R}$ using $\{+, *\}$. The predicate *integer*(x) is not definable in RCF, otherwise $Th(\mathcal{N}, +, *)$ would be interpretable in RCF, making skolem arithmetic decidable.[3]

A term t in RCF is a multinomial in some variables $(x_1, \ldots, x_m)$, written as

$$t(x_1, \ldots, x_m) = \sum_i (c_i \prod_j x_{i,j}) \tag{4.7}$$

Atomic formulæ can be written as $(\mathit{term} = 0)$ or $(\mathit{term} > 0)$. To apply QE [**?**], we need to consider only formulæ of type:

$$\exists x((\bigwedge_i p_i(x) = 0) \wedge (\bigwedge_j q_j(x) > 0)) \tag{4.8}$$

RCF can be axiomatized as follows:

1. Axioms for ordered commutative field.
2. Axioms for existence of square roots and roots of polynomials of odd degree:

$$\forall x \exists y(x = y^2 \vee x = -y^2) \tag{4.9}$$

$$\forall x_0 \ldots x_{2n} \exists y(x_0 + x_1 y + \cdots + x_{2n} y^{2n} + y^{2n+1} = 0) \tag{4.10}$$

Since all positive numbers have square roots in RCF, we can define $(y = \sqrt{x})$ as $(x \geq 0 \rightarrow y^2 = x)$.

[3]Which is known to be undecidable.

Chapter 5

Motion

We take a brief look at the properties of motion before trying to describe it in our theory.

5.1 Properties of Motion

1. An object must be at exactly one point at any given time. Its trajectory may be treated as a total function from time–points to space–points.
2. Only one object may occupy a spatial point at any one time point. Two objects trying to occupy the same space–point at the same time result in a ‘collision’, so trajectories would be non-intersecting curves in space–time.
3. The trajectory of the object is a continous path in space–time. An object cannot jump between space–points in zero time interval.
4. A body in motion continues moving unless otherwise stated, this is the frame assumption.

5.2 A Theory for 1-Dimension Motion

Let us take DLO as the underlying theory of time (T_t) and also as the underlying theory (T_s) of space. If we can somehow combine these theories, a unary predicate can describe describe trajectories in the combined theory T_{st}. If both the theories are individually decidable, and the process of combining simple enough, it is reasonable to hope that the combined theory is also decidable.

5.3 Product Theory Construction

Given two theories T_1 and T_2 languages L_1 and L_2 respectively, and with models M_1 and M_2 repectively, $T_1 \otimes T_2$ and T_{12} both will denote the product theory.

The combined language L_{12} is the disjoint union of the two languages, along with new unary predicates ($\{P_a(x) : a \in A\}$ where A is a set of objects under consideration), and two equality symbols ($=_1, =_2$). If a predicate name is common to both the theories, we will suffix the predicate name with the theory name. Define the domain of the product model M_{12} to be the cartesian product of the domains of M_1 and M_2.

The new variables range over points of M_{12}, and the two equality symbols allow extraction of the individual coordinates of the points. A formula of T_1, which is true of all points in M_1 is now a formula of T_{12} which is true of all points of M_{12}.

$$\models_{M_1} \forall x \phi(x) \Rightarrow \models_{M_{12}} \forall xy(y =_1 x \rightarrow \phi'(y))$$

A formula of T_1, which is true of some point c in M_1 is now a formula of T_{12} which is true of some point of M_{12} whose first coordinate equals c.

$$\models_{M_1} \exists x \phi(x) \Rightarrow \models_{M_{12}} \exists xy(y =_1 x \wedge \phi'(y))$$

Atomic predicate from T_1 (or T_2) is satisfied in M_{12} iff the projection of its variables and constants satisfy the predicate in M_1 (or M_2).

$$\models_{M_1} p(c) \Rightarrow \models_{M_{12}} \forall x(x =_1 c \rightarrow P_1(x))$$

Hence we have a truth preserving translation of formulas from $T_1$and T_2 to $T_1 \otimes T_2$. For example, the product theory of two dense linear orders ($(Q, <) \otimes (Q, <)$) would be the theory of $(Q \times Q, <_1, <_2, =_1, =_2)$

5.4 Motion in Product Theory

Let us use T_1 to denote the theory of time and T_2 to denote the theory of space, where both T_1 and T_2 are theories of dense linear order without endpoints (DLO). Using the new unary predicate $p(x)$ to mark out an object path consisting of space-time points, we can express a variety of motion:
Object at rest (Fig. 1):

$$\forall xy((p(x) \wedge p(y)) \rightarrow x =_2 y)$$

Object monotonically moving in the forward direction (Fig. 2):

$$\forall xy((p(x) \wedge p(y) \wedge x <_1 y) \rightarrow x <_2 y)$$

Object constrained to move between an lower bound l and upper bound u (Fig. 3):

$$\forall x(p(x) \rightarrow (l <_2 x <_2 u))$$

Continuity of motion (Fig. 4):

$$\forall xyz \exists w((p(x) \wedge p(y) \wedge x <_2 z <_2 y) \rightarrow (p(w) \wedge x <_1 w <_1 y \wedge w =_2 z))$$

Functionality of trajectory:

$$\forall xy((p(x) \wedge p(y) \wedge x =_1 y) \rightarrow x =_2 y)$$

To represent the motion of a bouncing ball we define a series of subformulas to define the formula for *bouncing* (Figs. 5 to 12):

$$\begin{array}{rcl}
\mathit{up}(x,y) & \overset{\text{def}}{=} & p(x) \wedge p(y) \\
& \wedge & \forall uv((x <_1 u <_1 v <_1 y \wedge p(u) \wedge p(v)) \rightarrow (u <_2 v)) \\
\mathit{down}(x,y) & \overset{\text{def}}{=} & p(x) \wedge p(y) \\
& \wedge & \forall uv((x <_1 u <_1 v <_1 y \wedge p(u) \wedge p(v)) \rightarrow (v <_2 u)) \\
\mathit{top}(x,y) & \overset{\text{def}}{=} & \mathit{up}(x,y) \\
& \wedge & \forall u((u <_1 x <_1 y \wedge p(u)) \rightarrow \neg \mathit{up}(u,y)) \\
& \wedge & \forall v((x <_1 y <_1 v \wedge p(v)) \rightarrow \neg \mathit{up}(x,v)) \\
\mathit{bot}(x,y) & \overset{\text{def}}{=} & \mathit{down}(x,y) \\
& \wedge & \forall u((u <_1 x <_1 y \wedge p(u)) \rightarrow \neg \mathit{down}(u,y)) \\
& \wedge & \forall v((x <_1 y <_1 v \wedge p(v)) \rightarrow \neg \mathit{down}(x,v)) \\
b(x,y,z) & \overset{\text{def}}{=} & \mathit{bot}(x,y) \wedge \mathit{top}(y,z) \wedge z <_2 x \\
\mathit{bouncing} & \overset{\text{def}}{=} & \forall xyz \exists uv((b(x,y,z) \rightarrow b(z,u,v)) \\
& \wedge & \forall xyzuv \left(\begin{array}{c} b(x,y,z) \\ \wedge \\ b(z,u,v) \end{array} \right) \rightarrow \left(\begin{array}{c} (y =_2 u) \\ \wedge \\ (v <_2 z <_2 x) \end{array} \right)
\end{array}$$

5.5 Decidability of DLOP

The next objective is is to look for a fast decision procedure to decide the truth of sentences in the language. In the product theory, a formula will express a path of an object. A model for the formula implies the existence of such a path. If this theory admits quantifier elimination, then we can extract the path explicitly from the quantifier free formula. A theory T admits quantifier elimination (QE), if for every formula ϕ in language of T, there is a quantifier free formula ψ that is T-equivalent to it:

$$T \vdash (\phi \leftrightarrow \psi)$$

DLO admits QE [**?**], it works by showing $\exists x \phi(x)$ is true iff the set represented by $\phi(x)$ is an union of nonempty line segments on the rational line. Quantifiers can be eliminated in DLO $\otimes$ DLO, by considering a formula $\phi(x)$ to be representing unions of rectangles on the rational plane. If we can show QE for DLO $\otimes$ DLO with a unary predicate (DLOP), then we have a decision procedure for our theory of space-time with space and time both represented by rationals.

We first considered the simpler problem of showing the decidability of DLO with a single unary predicate (DLOP). If we can show the decidability of this theory then we may be able to extend it to $DLO \otimes DLO$ with a unary predicate.

Given any formula in the language of DLOP, $(Q, <, p)$. We put the formula in prenex normal form, all quantifiers are pulled out, negations pushed in, and the quantifier free part is put in disjunctive normal form. We can start as in the case of DLO, except that non-empty intervals where x exists can be discarded in DLO, but in DLOP the truth values to p over the interval must be kept until all references to p are processed.

We use square brackets to denote closed intervals, and parenthesis to denote open intervals and the following abbreviations:

$$\begin{array}{rcl} (a < x < b) & for & (a < x) \wedge (x < b) \\ \forall x \in (a,b) \ldots & for & \forall x((a < x < b) \to \ldots) \\ \exists x \in (a,b) \ldots & for & \exists x((a < x < b) \wedge \ldots) \end{array}$$

To keep track of the truth assignments to p over the DLO-line, we expand the language as follows:

$$\begin{array}{rcl} p^{\forall 1}_{(a,b)} & \stackrel{\text{def}}{=} & \forall x(x \in (a,b) \to p(x)) \\ p^{\forall 0}_{(a,b)} & \stackrel{\text{def}}{=} & \forall x(x \in (a,b) \to \neg p(x)) \\ p^{\exists 1}_{(a,b)} & \stackrel{\text{def}}{=} & \exists x(x \in (a,b) \wedge p(x)) \\ p^{\exists 0}_{(a,b)} & \stackrel{\text{def}}{=} & \exists x(x \in (a,b) \wedge \neg p(x)) \end{array}$$

Negation of (fixed) regions is as follows:

$$\begin{array}{rcl} \neg p^{Q\,k}_{(a,b)} & \Rightarrow & p^{Q'\,(1-k)}_{(a,b)} \\ & & \text{where: } \forall' = \exists,\ \exists' = \forall \end{array}$$

Example 5.1. QE in DLOP:
Given:

$$\phi \quad = \quad \phi_1 \wedge \phi_2$$

$$\begin{array}{rcl}\phi_1 & = & \forall xy((x < y) \to \exists uv((x < u < y) \wedge (x < v < y) \wedge p(u) \wedge \neg p(v))) \\ \phi_2 & = & \exists ab \forall z(a < z < b \to p(z))\end{array}$$

Convert as follows:

$$\begin{array}{rcl}\phi_1 & \Rightarrow & \forall xy \left(p^{\exists 1}_{(x,y)} \wedge p^{\exists 0}_{(x,y)}\right) \\ \phi_2 & \Rightarrow & \exists ab(p^{\forall 1}_{(a,b)})\end{array}$$

During QE, $\forall xy \ldots$ would be written as $\neg \exists xy \neg \ldots$.

$$\begin{array}{rcl}\phi_1 & \Rightarrow & \neg \exists xy((p^{\forall 0}_{(x,y)} \vee p^{\forall 1}_{(x,y)}) \wedge (x < y)) \\ \phi_1 & \to & \neg \exists xy(p^{\forall 1}_{(x,y)} \wedge (x < y)) \\ \phi_1 \wedge \phi_2 & \to & \perp\end{array}$$

We must note that in negation of $p^{Q,k}_{(a,b)}$ works only as long as constant intervals are concerned. The problem is if we store values of p over variable intervals, on negation existential variables become free. In ϕ_1, x and y are free variables and can be instantiated to any value. The leading $\exists$ is misleading, we must take note of the leading negation sign.

The problem of negating partial values of p over the DLO is yet to be solved and the satisfiablity problem of DLOP is still open.

Notes:

1. If we combine two propositional temporal logics, the product theory is undecidable. This undecidability proof is based on the coding of the Post correspondence problem in DLOP. This proof is given in appendix, the result was derived by an alternate method in [**?**].
2. While PTL captures the notion of 'finite' time, which is the essense of computability – of program termination, 'finiteness' is not definable in DLOP.
3. Tarski's theory of geometry extended with an unary predicate gives rise to undecidability [**?**].

5.6 Expressibility of DLOP

We now look at some examples in DLOP to see its expressive power and limitations:

Example 5.2. p holds for all rationals:

$$\forall x p(x)$$

Example 5.3. p holds only in $(-\infty, x)$ for some x:

$$\exists x \forall y (p(y) \leftrightarrow y < x)$$

Example 5.4. p holds only at n distinct points:

$$\exists x_1 \ldots x_n (\bigwedge_{ij} (x_i \neq x_j) \wedge \forall y (p(y) \rightarrow \bigvee_i (y = x_i)))$$

Example 5.5. The formula ϕ defines an infinte sequence of points starting at x, and ending before y:

$$\begin{aligned}
\phi &= \exists xy (p(x) \wedge x < y \wedge \phi_1(x,y) \wedge \phi_2(x,y)) \\
\phi_1(x,y) &= \forall u ((u < x \rightarrow \neg p(u)) \wedge (u > y \rightarrow \neg p(u))) \\
\phi_2(x,y) &= \forall u (((x \leq u < y) \wedge p(u)) \rightarrow (\phi_3(x,y,u))) \\
\phi_3(x,y,u) &= \exists v (p(v) \wedge u < v < y \wedge \forall t (u < t < v \rightarrow \neg p(t)))
\end{aligned}$$

All the models of ϕ are not isomorphic.

Where does the sequence of points converge ? The limit point can be:

1. A rational number $= y$.
2. A rational number $< y$.
3. A irrational number $< y$.
4. It converges to some point before y and the sequence starts again.

There is no way to refer to the limit point of a sequence in DLOP. Choice four above is particularly devastating, in the sense it does not allow us to even capture the notion 'converging sequence'.

Example 5.6. Dense Mix:

$$\forall xy ((x < y) \rightarrow \exists uv (p(u) \wedge \neg p(v)))$$

The only models for this formula, are those that assign p true and false in every open interval. All its models are isomorphic, the isomorphism is the same as that between any two DLOs.

5.7 Discussion

There may be subsets of formulas of the product theory which could be shown to be decidable more easily and this needs to be explored. In particular sufficiently restricting the trajectories of objects by axioms may result in decidable theories.

In the next chapter, we try to get hold of the useful expressive power of DLOP and discard the unused power so as to simplify representation and computations involving time.

Chapter 6

Ordinal Trees

In this chater we try to capture the useful expressive power of DLOP for reasoning about time. We have only the concepts of density, periodicity, inifinte sequences with limit points, and have no end–points. This lead us to develop the 'ordinal–tree'(OT) notation to represent temporal events. Other notions are later introduced to express spatial dislocaltion and temporal choice.

DLOP allows us to talk about sequences converging both in the positive as well as the negative direction of the rational line. However, in modelling of peroidic events (e.g. bouncing ball) time points are required only as a sequence of points converging in the positive direction. Hence this property will be shared by OTs.

We make trees out of intervals, with the 'root' as the 'present' time–point. Whenever more time points are required, the tree is split by adding a SPLIT node on the interval (Figure **??**).

An interval may be split by a periodic event, say the ticks of a clock. To represent this we use the LOOP node (Figure **??**). Even a finite interval may be split into an infinite sequence of subintervals. The tree can be viewed as repeatedly (ω times) splitting the right hand side of the interval.

We disallow trees with back–loops and cross–arcs (Figure **??**). Rules for creating back-arcs can be stated as follows: An arc may only begin from the right-hand-side of a SPLIT node and may not crossover to its left neighbours. With this rule, the arc may go back to some ancestor of the node of its origin. (Figure **??**). However one more simplification is possible, an arc that goes to the ancestor, need only loop back to itself (Figure **??**).

Figure 6.1: SPLIT node splits an interval

Figure 6.2: LOOP Node divides an infinite interval into an infinite sequence of sub-intervals

Figure 6.3: Back–loops and cross–arcs disallowed

Figure 6.4: Valid back-arc

Figure 6.5: Loop shortening

Figure 6.6: Bouncing Ball

Figure 6.7: Nested loops

6.1 Representing temporal information on OTs

The arcs of the OT can be labelled with formula of the object language – to denote facts about events in that interval in the world. We will use propositional symbols in this chapter as labels, though we may as well be using modal or first order logic formulae as labels.

Example 6.1. Bouncing Ball:
Let: D denote that the ball is moving downwards, and $\neg D$ stand for 'the ball is not moving downwards' (Figure **??**).

Example 6.2. Nested loops:
Nested loops represent a sequence of converging sequences, they are different from a single sequence (Figure **??**).

Figure 6.8: The clock ticked till one day it stopped.

Example 6.3. The clock:
Let the proposition P denote that the pendulum of the clock is at the center position. Then the figure **??** shows the working of the clock for the whole of its future.

6.2 Extension to Oridnal Trees

Time and temporal events can now be represented by labelled OTs. We now introduce three kinds of nodes and give examples of their usage:

OR These nodes have multiple arcs going out of it. They are allowed only on the outgoing right branches only. The arcs denote the different events than can occur, in some sense they denote choice.

AND This node denotes multiple copies of the time–intervals spatially dislocated, they can be used to represent two or more events that occur simultaneously — in the same time interval but at different locations,

CAUSES Some events are always followed by other events. To explicitly represent the cause and effect temporally we use these nodes. Note that while the first two are used to build models (as facts), the CAUSES nodes are used to build constraints on the models (as rules) and will be useful during computations.

Example 6.4. "A ball thrown will either land inside the field or go out of it"
The ball flying through the air (A), will either land in the field (F) or fall outside the filed (O). Figure **??** shows the temporal relation between the events. (Figure **??**).

Example 6.5. "Ball bouncing on a ledge falls off"
The ball is moving up (P) and down ($\neg P$) on the ledge, until it falls off and moves up (Q)

Figure 6.9: The ball thrown lands in the field or outside it.

Figure 6.10: The bouncing ball on the ledge: falls and bounces on the ground.

and down ($\neg Q$) on the ground (Figure **??**).

Example 6.6. "Jim ran to catch the ball:"
Here we want to show that jim was running (J), as the ball was falling (F), and then either Jim caught the ball (C) or the ball fell to the ground (G) (Figure **??**).

The cause always precedes the effect though the time difference between the two varies from almost zero to any large amount. The five cases of how the intervals of cause and effect are temporally related are shown in figure **??**. The CAUSES node is infact made from two SPLIT nodes and an AND node. This is represent by OT as: The special relationship (besides temporal) between cause and effect are represented by the CAUSES node figure **??**.

Figure 6.11: Jim ran to catch the falling ball.

Figure 6.12: Cause and Effect Intervals

Figure 6.13: Cause and Effect OT

Figure 6.14: Rain causes the ground to get Wet

Figure 6.15: Collision

Example 6.7. "When it rains the ground is wet"
Rain (R) causes the ground to get wet (W), see figure **??** [1].

Example 6.8. Collision:
Object x moves towards point z (P), while object y is also moving to z (Q), at time t they collide and x moves away from z (R), and y also moves away from z (S), see figure **??**.

Example 6.9. No Collision:
Object x moves towards point z (P) and reaches it at time A before y does, while object y is also moving to z (Q) and reaches it at time B, x moves away from z (R), and y also moves away from z (S), see figure **??**.

[1]When we infer from the wet ground that it is/was raining we are actually doing abduction, or looking for possible causes. Abduction is not always accurate.

Figure 6.16: No Collision

6.3 Expressive power of trees

There is no measure of time, any one time interval is isomorphic to any other time interval. However a clock ticking over the interval (x, ∞) simultaneously with other events can be used to mark out time points.

Trees can define sequences converging to points, sequence of sequences converging to a point, etc. But the sequence can only converge in the positive direction. It cannot represent a dense–mix of P and $\neg P$, as the DLOP can:

$$\forall xy \exists uv (x < y \rightarrow ((x < u < v < y) \wedge p(u) \wedge p(v)))$$

Some kind of resolution can be used to merge and match these trees. Trees can be labelled with event variables, that get instantiated on matching.

We look at methods to find normal forms for ordinal trees in the next chapter.

Chapter 7

Normal Forms

Suppose the tree is labelled by P and $\neg P$, then we can breakup time into the following types of closed-open intervals [1]:

1. P at both a and in (a, b).
2. $\neg P$ at both a and in (a, b).
3. P at a and $\neg P$ in (a, b).
4. $\neg P$ at a and P in (a, b).

These four items can be represented by four alphabets $\Sigma = \{a, b, c, d\}$, and a labelled tree can be represented by a ω–string on the alphabet Σ. Let the following be rules to make strings (in the set Σ^ω) out of it:

$$\begin{array}{rcll} x & \in & \Sigma^\omega & x \in \Sigma \\ (t\,s) & \in & \Sigma^\omega & t, s \in \Sigma^\omega \\ t^\omega & \in & \Sigma^\omega & t \in \Sigma^\omega \end{array}$$

Note that $\Sigma^* \subset \Sigma^\omega$.

Do different trees represent non-isomorphic interval breakups ? The answer is unfortunately no. There are two ways of splitting a single interval into three parts (figure **??**). The problem does not end with finite equivalences, even LOOPed trees can be equivalent, as the next example shows.

Example 7.1. Equivalent trees having LOOPs (Figure **??**).

[1]Let [a,b) be the clopen-interval under consideration

Figure 7.1: Equivalent trees: there are two ways of splitting a single interval into three parts.

Figure 7.2: Equivalent trees having LOOPs

$$\begin{array}{rcl} a\,a^\omega & = & a^\omega \\ a^\omega\,(a^\omega)^\omega & = & (a^\omega)^\omega \\ a^\omega\,a & \neq & a^\omega \\ (a^\omega)^\omega\,a^\omega & \neq & (a^\omega)^\omega \end{array}$$

This happens because of ordinal arithmetic and prompts us to take a brief look at the ordinals in the next section.

We later give an algorithm to compute an unique equivalent string for any string in Σ^ω, thus providing a method for normalizing strings.

7.1 Ordinals

Ordinals are generalizations of the natural numbers [**?**]. The first limit ordinal is denoted by ω, which can be reached at the 'end' of the sequence of natural numbers. After ω, we start a fresh sequence $\omega+1, \omega+2, \ldots$, until we reach the next limit ordinal $\omega + \omega = \omega 2$. We continue in this fashion until we pass by every ωn, and finally reach $\omega\omega = \omega^2$. We can moreover reach every finite power of omega, that is ordinals less than ω^ω. Ordinals larger than ω^ω are not useful for our theory of time.

Ordinal arithmetic is the generalization of addition of numbers with the following properties:

$$1 + \omega = \omega \neq \omega + 1$$

$$\omega + \omega = \omega 2 \neq 2\omega = \omega$$

Similarly for higher powers:

$$\omega^n + \omega^m = \omega^m \neq \omega^m + \omega^n$$

where $0 < n < m < \omega$.

$$\omega \cdot n = \underbrace{\omega + \cdots + \omega}_{\text{n times}} \neq n \cdot \omega = \omega$$

In general any ordinal $< \omega^\omega$ can be put in the form:

$$\omega^n a_n + \cdots + \omega^1 a_1 + a_0$$

where $n, a_0, \ldots, a_n$ are all natural numbers, and $a_n > 0$.

Trees without LOOPs can be identified with finite natural numbers, and each LOOP is identified with an ω.

For finite intervals n, we define normal OTs as follows: The first interval is identified with one, and the right sub interval is split to accomodate the other $(n-1)$ intervals recursively (Figure **??**).

Figure 7.3: Normal Finite Trees

7.2 Reduction of Strings

Example 7.2.

$$\begin{array}{rc} & (ba)^{\omega}(ab)^{\omega^2} \\ \Rightarrow & b(ab)^{\omega}(ab)^{\omega^2} \\ \Rightarrow & b(ab)^{\omega^2} \end{array}$$

Example 7.3.

$$\begin{array}{rc} & (ba)^{\omega 2}(ab)^{\omega^2} \\ \Rightarrow & (ba)^{\omega}(ba)^{\omega}(ab)^{\omega^2} \\ \Rightarrow & (ba)^{\omega}b(ab)^{\omega}(ab)^{\omega^2} \\ \Rightarrow & (ba)^{\omega}b(ab)^{\omega^2} \end{array}$$

Example 7.4.

$$(ab)^{\omega}(ba)^{\omega}(ab)^{\omega}$$

$$\begin{array}{cc} \Rightarrow & a(ba)^{\omega 2}(ab)^{\omega} \\ & OR \\ \Rightarrow & (ab)^{\omega}b(ab)^{\omega 2} \end{array}$$

7.3 Normalization Algorithm

Given any string in Σ^{ω} we will compute an equivalent string in Σ^{ω}, moreover the equivalent string will be unique.

Let us call the phenomena: '$a \cdot a^{\omega} \Rightarrow a^{\omega}$' as 'gobbling', and the phenomena:

$$(a\,b\,c)^{\omega} = a(bca)^{\omega} = ab(cba)^{\omega} = abc(abc)^{\omega} = \dots$$

as 'rotation' and let $|v|$ be the (ordinal) length of string v.

We will treat all strings in Σ^* as single elements, as they cannot be processed.

Let $\alpha(k)$ denote some ordinal of the form:

$$\alpha(k) = \omega^k a_{ik} + \dots + \omega^1 a_{i1} + a_{i0} = \sum_{j=k}^{0} \omega^j \cdot a_{ij}$$

where $a_{ik} > 0$ and $\omega^0 = 1$ and i will be used to index different α and k.

If possible we let a string of power higher order ordinal gobble v^{ω} rather than v^{ω} gobble strings on its left. Hence we start comparing adjacent strings right to left.

Since strings of order ω^2 and larger cannot be rotated, we have to look only up to power ω^2. Two adjacent strings can have powers ω^0 (finite), ω and $\geq \omega^2$, we have to consider nine cases.

Algorithm 1. Normalization
Given

$$s = v_1^{\alpha_1(k_1)} \cdots v_n^{\alpha_n(k_n)} = t_1 \cdots t_n$$

with $n \geq 0$ and $k_1, \dots, k_n$ natural numbers, and each α_i denoting some ordinal $< \omega^{\omega}$.

1. For every t_i, check if there exists x and $j > 1$ such that $x^j = t_i$, if yes, replace $v_i^{\alpha(k_i)}$ by $x^{j \cdot \alpha(k_i)}$. where the multiplication is ordinal multiplication.
2. For every $v_i^{\alpha(k_i)}$, apply this algorithm recursively on v_i to get x_i. Replace $v_i^{\alpha(k_i)}$ by $x_i^{\alpha(k_i)}$.

3. Compare adjacent $t_i t_{i+1}$ starting at the rightmost end of s, that is $i = n-1, n-2, \ldots, 1$.

   if $v_i = v_{i+1}$ **then** replace $v_i^{\alpha(k_i)}\, v_{i+1}^{\alpha(k_{i+1})}$, by $v_i^{\alpha(k_i)+\alpha(k_{i+1})}$, where the + denotes ordinal addition.

   else let t_i be $x^{\alpha(l)}$ and let t_{i+1} be $y^{\beta(m)}$, l and m are natural numbers, $\alpha(l), \beta(m)$ ordinals, x, y strings in Σ^ω. There are nine cases as the value of $\alpha(l), \beta(m)$ ranges in finite, from ω to ω^2, and beyond ω^2:

   1,2,3. $l > 1 \wedge m \geq 0$ No change.

   4,5. $l = 1 \wedge m \geq 1$ If rotation of x^ω can be gobbled up by $y^{\beta(m)}$ then
   Reduce: $x^{\omega n} \cdot y^{\beta(m)} \Rightarrow x^{\omega(n-1)} x^\omega y^{\beta(m)}$
   else, no change

   6. $l = 1 \wedge m = 0$ No change.

   7,8. $l = 0 \wedge m \geq 1$ We have $x\, y^{\beta(m)}$. If any suffix of x can be gobbled up by $y^{\beta(m)}$, gobble it.

   9. $l = 0 \wedge m = 0$ Adjacent finite strings should have been merged earlier, and will not occur.

The function $NF()$ will denote application of the algorithm. Let us look at some examples:

Example 7.5. Revisited:

$$\begin{aligned} & NF((ba)^\omega (ab)^{\omega^2}) \\ \Rightarrow\ & NF(NF(ba)^\omega\, NF(ab)^{\omega^2}) \\ \Rightarrow\ & NF((ba)^\omega (ab)^{\omega^2}) \\ & rotation((ba)^\omega \Rightarrow b(ab)^\omega \\ \Rightarrow\ & NF(b(ab)^\omega (ab)^{\omega^2}) \\ \Rightarrow\ & NF(b(ab)^{\omega^2}) \end{aligned}$$

Example 7.6. Revisited:

$$\begin{aligned} & NF((ba)^{\omega 2} (ab)^{\omega^2}) \\ \Rightarrow\ & (ba)^\omega (ba)^\omega (ab)^{\omega^2} \\ \Rightarrow\ & (ba)^\omega b(ab)^{\omega^2} \end{aligned}$$

As our algorithm avoids splitting up strings of higher powers ($\geq \omega^2$) to gobble finite strings:

$$\begin{aligned} & b(ab)^{\omega^2} \\ \not\Rightarrow\ & b(ab)^\omega (ab)^{\omega^2} \\ \not\Rightarrow\ & (ba)^\omega (ab)^{\omega^2} \end{aligned}$$

Example 7.7. Revisited:
The middle $(ab)^\omega$ rotates to $a(ba)^\omega$ and then $(ba)^\omega$ gets gobbled by the term on its right. We get this as normal form, because we go from right to left.

$$\begin{aligned} & NF((ab)^\omega(ba)^\omega(ab)^\omega) \\ \Rightarrow \quad & (ab)^\omega a(ab)^{\omega 2} \end{aligned}$$

Chapter 8

Conclusion

In the last one year we studied the nature of temporal information, and looked at one case study, namely 'motion'. Motion seems to involve most of the intricacies of temporal information. Our main objective in the study was to find a decidable theory which could be used to represent and reason about motion. Since both space (i.e. geometry) and time (DLO) have interesting and expressive decidable theories, we thought it reasonable to expect that they could be combined to get a decidable theory. The task of combining the theories can be worked out, but the task of showing it to be decidable is open and appears difficult.

Our first attempt was to see if the theory could be undecidable. If we could show that some interesting properties could be expressed then the combined theory becomes undecidable. This was achieved by coding the PCP in 2-dimension temporal-logic (appendix D). However it is not clear if such properties can be coded in DLOP.

Our second attempt was to try QE to show that the theory is decidable. We first tried this for the theory of DLO, augmented with a single unary predicate p, (the theory DLOP), because we felt we could extend this to the combined case. Partial models of DLOP appear during QE, and on encountering a $\forall$ quantifier, we need to negate these partial models, however we could not find a way of handling information about negated partial models uniformly. It is easy to handle individual cases of negation, but for QE, we need to give a general method.

From our study of QE for DLOP, we found that the kinds of useful intervals that were being defined implicitly by DLOP formulas can essentially be coded by structures which we call ordinal trees.

Ordinal trees are easier to manipulate, and can express interesting information such as 'converging to' which was earlier not possible in DLOP. To make ordinal trees really useful, we made rules on what kind of arcs are valid and invalid; we also introduced a special kind of node to talk about spatially dislocated events occuring in the same time interval, and nodes for temporal choice in the future.

8.1 Future work

Our next step would be to look at the following issues:

1. Decidablity of DLOP
2. Decidability of DLO $\times$ DLO with an unary predicate.
3. Define operations on ordinal trees, which would represent temporal operations.
4. Extend conventional logic programming languages by embedding the concepts of time inplicitly in it. This would make temporal reasoning elegant and efficient.
5. See if ordinal trees are useful for representing temporal information in natural languages.

Appendix A

Bibliography

Bibliography

[Arn 88] Arnon, D.S., Geometric reasoning with logic and algebra, *Artificial Intelligence* **37** (1988) 37-60.

[Bar 77] Barwise, J., Introduction to first order logic, in: J. Barwise (Ed.),*Handbook of Mathematical Logic*, (North Holland, Amsterdam, 1977) 5-46.

[Ben 88] van Benthem, J. F. A. K., *Time, Logic and Computation*, (LNCS 354) Springer Verlag 1988.

[Ben 83] van Benthem, J. F. A. K., *The Logic of Time*, (D. Reidel, Dordrecht, Holland, 1983).

[Boo 79] Boolos, G., *The Unprovability of Consistency*, (Cambridge University Press, 1979).

[Che 80] Chellas, B., *Introduction to Modal Logic*, (Cambridge University Press, 1980).

[End 72] Enderton, H., *A Mathematical Introduction to Logic*, (Academic press 1972).

[Hal 86] Halpern, J., Reasoning about knowledge: an overview, in: J. Halpern (ed.), *Reasoning about Knowledge*, (Morgan Kaufman 1986) 1-17.

[Kap 88] Kapur, D. and Mundy J.L., (Eds.), *Special Volume on Geometric Reasoning*, *Artificial Intelligence* **37** journal, (North Holland, Amsterdam, 1988).

[Kri 67] Kriesel, G. and Kirvine, J.L., *Introduction to Mathematical Logic*, (North Holland, Amsterdam, 1967) 60-64.

[Lad 86] Ladner, R. and Reif, J., The logic of distributed protocols, preliminary report, in: J. Halpern (ed.), *Reasoning about Knowledge*, (Morgan Kaufman 1986) 207-222.

[Leh 84] Lehman, D., Knowledge, common knowledge and related puzzles, *Proceedings of Third ACM Conference on Principles of Distributed Computing*, 1984, 62-67.

[Mon 69] Monk, D. J., Introduction to set theory, McGraw Hill, NY, 1969.

[Par 87] Parikh, R., Knowledge and problem of logical omniscience, *ISMIS-87*, North Holland, 432-439.

[Rac 79] Rackoff, J. and Ferrante C.W., *The computational complexity of logical theories*, Lecture Notes in Mathematics **718**, (Springer Verlag, 1979).

[Rei 47] Reichenback, Hans, *Elements of Symbolic Logic* The Macmillan Co., NY, 1947.

[Rog 71] Rogers, R., *Mathematical Logic and Formalized Theories*, (North Holland, 1971).

[Sho 88] Shoham, Y., *Reasoning about Change*, (MIT Press, 1988).

[Smu 86A] Smullyan, R., *Forever Undecided*, Oxford University Press 1986.

[Smu 86B] Smullyan, R., Logicians who reason about themselves, in: J. Halpern (ed.), *Reasoning about Knowledge*, (Morgan Kaufman 1986) 341-352.

[Spa 90] Spaan, Edith, Nexttime is not necessary, in: R. Parikh (ed.), *Reasoning about Knowledge*, (Morgan Kaufman 1990) 241-256.

[Sza 87A] Szalas, A., A complete axiomatic characterization of first order temporal logic of linear time, *Theoretical Computer Science* **54**, 1987.

[Sza 87B] Szalas, A., Arithmetical axiomatization of first order temporal logic, *Information Processing Letters*, **26**, 1987.

[Sza 88] Szalas, A. and Holenderski, L., Incompleteness of first order temporal logic with until, *Theoretical Computer Science* **57**, 1988.

[Tar 51] Tarski, A., *A Decision Method for Elementary Algebra and Geometry* (University of California Press, Berkeley, CA, 2nd ed., 1951).

[Tar 53] Tarski, A., *Undecidable Theories*, (North Holland, Amsterdam, 1953).

[Tar 69] Tarski, A., What is elementary geometry ?, in J. Hintikka (Ed.), *The Philosophy of Mathematics*, (Oxford University Press, 1969) 164 - 177.

[Tar 86] Tarski, A., Truth and proof, in: S.R. Givant and R.N. Mckenzie (Eds.), *Tarski: Collected Papers*, Volume **4**, (Birkauser, Basel, 1986).

[Wol 83] Wolper, P., Temporal logic can be more expressive, *Information and Control* **56**, 1983.

Appendix B

Axioms for Plane Geometry

Ax1	$\forall xy\,[\beta(xyx) \rightarrow (x = y)]$
Ax2	$\forall xyzu\,[(\beta(xyu) \wedge \beta(yzu)) \rightarrow \beta(xyz)]$
Ax3	$\forall xyzu\,[(\beta(xyz) \wedge \beta(xyu) \wedge (x \neq y)) \rightarrow (\beta(xzu) \vee \beta(xuz))]$
Ax4	$\forall xy\,[\delta(xyyx)]$
Ax5	$\forall xyz\,[\delta(xyzz) \rightarrow (x = y)]$
Ax6	$\forall xyzuvw\,[(\delta(xyzu) \wedge \delta(xyvw)) \rightarrow \delta(zuvw)]$
Ax7	$\forall txyzu\exists v\,[(\beta(xtu) \wedge \beta(yuz)) \rightarrow (\beta(xvy) \wedge \beta(ztv))]$
Ax8	$\forall txyzu\exists vw\,[(\beta(xut) \wedge \beta(yut) \wedge (x \neq u))$
	$\rightarrow (\beta(xzv) \wedge \beta(xyw) \wedge \beta(vtw))]$
Ax9	$\forall xx'yy'zz'uu'\,[(\delta(xyx'y') \wedge \delta(yzy'z') \wedge \delta(xux'u') \wedge \delta(yuy'u') \wedge$
	$\beta(xyz) \wedge \beta(x'y'z') \wedge (x \neq y)) \rightarrow \delta(zuz'u')]$
Ax10	$\forall xyuv\exists z\,[\beta(xyz) \wedge \delta(yzuv)]$
Ax11	$\forall xyz\,[\neg\beta(xyz) \wedge \neg\beta(yzx) \wedge \neg\beta(zxy)]$
Ax12	$\forall xyzuv\,[(\delta(xuxv) \wedge \delta(yuyv) \wedge \delta(zuzv) \wedge (u \neq v))$
	$\rightarrow (\beta(xyz) \vee \beta(yzx) \vee \beta(zxy))]$
Ax13*	$\forall p, \ldots, q\,[\forall xy\exists z((\phi(x) \wedge \psi(y)) \rightarrow \beta(zxy))$
	$\longrightarrow \exists u\forall xy((\phi(x) \wedge \psi(y)) \rightarrow \beta(xuy))]$

Appendix C

Classical and Nonclassical Logics

We define the syntax and semantics of proposional, first order, modal, propositional temporal and first order temporal logics.

C.1 Terminology

The *Language* ($\mathcal{L}$) of a logic consists of logical and nonlogical constants together with a set of rules to generate a set of well formed formulæ ($wff(\mathcal{L})$). The logical constants have fixed meaning in a logic, while the nonlogical constants can have any interpretation.

$\mathcal{M}$ is called a *model* for a set Γ of formulæ ($\Gamma \subseteq wff(\mathcal{L})$) , if every $\phi \in \Gamma$ is interpreted as true in $\mathcal{M}$; written as $\models_{\mathcal{M}} \Gamma$. A set Γ *logically implies* ϕ (ϕ is a *logical consequence* of Γ), if ϕ is true in every model of Γ; written as $\Gamma \models \phi$. The set of logical consequences of Γ is denoted by $Cn(\Gamma) = \{\phi : \Gamma \models \phi\}$. A formula is *valid* if it is true in all models ($\models \phi$), and *unsatisfiable* if it has no models. It follows that ϕ is valid iff $\neg\phi$ is unsatisfiable.

A *theory* $\mathcal{T}$ is any set of sentences closed under logical consequences, that is $Cn(\mathcal{T}) = \mathcal{T}$. A *theory of a model* $\mathcal{M}$ is the set of sentences true in a model, denoted by $Th(\mathcal{M}) = \{\phi :\models_{\mathcal{M}} \phi\}$.

A theory $\mathcal{T}$ is *inconsistent* if it contains all formulæ in $wff(\mathcal{L})$, otherwise it is *consistent.* An inconsistent theory is unsatisfiable. It is *complete* if for every $\phi \in L$, ϕ or $\neg\phi$ is in $\mathcal{T}$. Clearly for any model $\mathcal{M}$, $Th(\mathcal{M})$ is complete.

A *rule of inference* (ROI), is a rule to deduce true formulæ from formulæ already known to be true. A *proof* of ϕ from Γ is a finite deduction of ϕ from Γ using the ROI, and is denoted by $\Gamma \vdash \phi$. The set of formulæ provable from Γ is denoted by $Pr(\Gamma) = \{\phi : \Gamma \vdash \phi\}$.

Theory $\mathcal{T}$ is (finitely) *axiomatizable* if there is a (finite) recursive set of axioms Ax, such that $Cn(Ax) = \mathcal{T}$. A theory $\mathcal{T}$ with axioms Ax is *sound* if $Pr(Ax) \subseteq \mathcal{T}$. $\mathcal{T}$ is *partially decidable* (a recursively enumerable set), iff there is a recursive set of axioms $Ax : Cn(Ax) = \mathcal{T}$. It is *decidable* if $\mathcal{T}$ is a recursive set[1]. Clearly, a partially decidable complete theory is decidable.

A logic is *compact*, if whenever $\Gamma \models \phi$ then there is $\Gamma_{finite} \subseteq \Gamma : \Gamma_{finite} \models \phi$. A theory T has *finite model property* (fmp), when every sentence not in $\mathcal{T}$ can be falsified in a finite model. If a complete theory has finite model property, then we can decide if a sentence is not in it by enumerating finite models till we find a model falsifying the sentence. Thus a complete theory having fmp is decidable, and so a partially decidable theory having fmp.

C.2 Classical Logics

In this section we present [?, ?, ?] propositional logic (PL) and first order logic (FOL).

C.2.1 Proposition Logic

The language of PL (L_{PL}) has the following logical symbols: logical operators $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ standing for (not, and, or, implies, iff) respectively, and the truth constants $\top$ (true) and $\bot$ (false). The nonlogical symbols are a set $Prop = \{p_i\}$ of propositions, where the subscripts will denote any convenient enumeration. The set of formulae is the closure of *Prop* and truth constants under the logical operations.

A structure $\mathcal{M}$ is a function assigning truth values to $\phi \in wff(L_{PL})$, as given below.

$$\models_{\mathcal{M}} (\top) \tag{C.1}$$
$$\models_{\mathcal{M}} p_i \quad \text{iff} \quad \mathcal{M} : p_i \mapsto \top \tag{C.2}$$
$$\models_{\mathcal{M}} \neg\phi \quad \text{iff} \quad \not\models_{\mathcal{M}} \phi \tag{C.3}$$
$$\models_{\mathcal{M}} \phi \rightarrow \psi \quad \text{iff} \quad \not\models_{\mathcal{M}} \phi \text{ or } \models_{\mathcal{M}} \psi \tag{C.4}$$

Other operators and truth constants are defined as follows:

$$\bot \overset{\text{def}}{=} \neg\top \tag{C.5}$$
$$\phi \vee \psi \overset{\text{def}}{=} \neg\phi \rightarrow \psi \tag{C.6}$$
$$\phi \wedge \psi \overset{\text{def}}{=} \neg(\phi \rightarrow \neg\psi) \tag{C.7}$$
$$\phi \leftrightarrow \psi \overset{\text{def}}{=} (\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi). \tag{C.8}$$

[1]We assume Church's thesis

A PL formula is a *tautology* if it is true in every model. The ROI used is *detachment*: from ϕ and $\phi \to \psi$, derive ψ. PL is compact, finitely axiomatizable and decidable in polynomial space and exponential time.

C.2.2 First Order Logic

The language of FOL (L_{FOL}) has the following logical symbols: logical operators $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$, quantifiers $\forall$ (for all) and $\exists$ (there exists), equality $=$, a set $X = \{x_i\}$ of variables, truth constants $\{\top, \bot\}$, The nonlogical symbols are sets $Const = \{c_i\}$ of constants, $Pred = \{p_i^n\}$ of predicates and $Func = \{f_i^n\}$ of functions, where the superscript will denote the arity. *Terms* are generated by closing $Const, X$ under $Func$. *Atomic formulæ* are $\top, \bot, p_i^n(t_1, \ldots, t_n), t_i = t_j$, where t_i is a term. The set of formulae is the closure of atomic formulæ under the logical operations and quantification by $\exists x_i$ or $\forall x_i$. A variable x occuring in ϕ is *bound*, if it is in the scope of a of $\exists x_i$ or $\forall x_i$ quantifier, otherwise it is called *free*. The set of free variables of ϕ is denoted by $FV(\phi)$. A *sentence* (or closed formula) is a formula without free variables.

A structure $\mathcal{M}$ consist of a non empty set $\mathcal{U}$ (called the *domain* of interpretation), where elements of $Const$, $Pred$ and $Func$ are interpreted as elements, relations and functions (of appropriate arity) on $\mathcal{U}$. The meaning of quantifiers is as follows:

$$\models_{\mathcal{M}} \forall x \phi(x) \quad \text{iff} \quad \forall b \in \mathcal{U}, \models_{\mathcal{M}} \phi(b) \tag{C.9}$$

$\models_{\mathcal{M}} (x = y)$ iff x and y represent the same element in $\mathcal{U}$. The dual of $\forall$ is $\exists$, $\exists x \phi(x)$ is defined as $\neg \forall x \neg \phi(x)$.

The rules of of inference used in FOL are modus ponens and generalization:

$$\text{MP} \quad \tfrac{\phi, \phi \to \psi}{\psi} \tag{C.10}$$

$$\text{Gen} \quad \tfrac{\phi(x)}{\forall x \phi(x)} \tag{C.11}$$

FOL is compact, finitely axiomatizable and undecidable.

C.3 Nonclassical Logics

In this section we present propositional modal logic (PML) [?], propositional temporal logic (PTL) [?] and first order temporal logic (FOTL) [?].

C.3.1 Propositional Modal Logic

PML is an extension of PL by the modalities $\{\Box, \Diamond\}$. The $\Box\phi$ (*box*) can read as ϕ is necessarily true, and its dual $\Diamond$ (*diamond*) for possibily true.

We will consider *normal* modal logics only, these logics satisfy the Kripke's axiom K : $\Box(\phi \to \psi) \to \Box\phi \to \Box\psi$. A *kripke structure* $\mathcal{M}$ consists of a set of worlds $\mathcal{W} = \{w_i\}$ and a binary accessibily relation R on the worlds. The worlds accessible from w are called the *possible worlds* for w. Each world w is a PL structure with truth as defined for PL, $\Box\phi$ is true at w in $\mathcal{M}$ iff ϕ is true in all possible worlds of w. Its dual, $\Diamond\phi$ is defined as $\neg\Box\neg\phi$.

C.3.2 Propositional Temporal Logic

PTL is an extension of PL by the following logical operation symbols $\{\Box, \Diamond, \bigcirc, \sqcup, \mathit{Atnext}\}$ called henceforth, eventually, next, until, atnext operations respectively. A model for PTL is an infinite linear sequence of worlds $< s_0, s_1, \ldots >$, where each world is a PL structure. The new logical operators are interpreted as follows:

$$\models_{s_i} \Box\phi \quad \text{iff} \quad \forall k \geq i :\models_{s_k} \phi \tag{C.12}$$

$$\models_{s_i} \bigcirc\phi \quad \text{iff} \quad \models_{s_{i+1}} \phi \tag{C.13}$$

$$\models_{s_i} \phi \sqcup \psi \quad \text{iff} \quad \exists k \geq i : \forall m \in \{i, \ldots, k-1\} \models_{s_m} \phi \text{ and } \models_{s_k} \psi \tag{C.14}$$

$$\models_{s_i} \phi\, \mathit{Atnext}\, \psi \quad \text{iff} \quad \exists k \geq i : \forall m \in \{i, \ldots, k-1\} \models_{s_m} \neg\phi \text{ and } \models_{s_k} (\phi \wedge \psi) \tag{C.15}$$

$$(\bigcirc z) = t \quad \overset{\text{def}}{=} \quad \forall x(t = x \to \bigcirc(z = x)) \tag{C.16}$$

We can use terms of the form $(\bigcirc z)$, because of the last definition.

C.3.3 First Order Temporal Logic

FOTL is the first order version of PTL. The language includes constants, functions, variables, equality, predicates, PL operators, FO quantifiers, operators and modalities of PTL. There are two types of variables: global variables $X = \{x_i\}$ which can be quantified, and state variables $Z = \{z_i\}$ which cannot be quantified.

A model for FOTL is an infinite linear sequence of states: $< s_0, s_1, \ldots >$, where each state is a FO structure. The domain of interpretation ($\mathcal{U}$) of the FO structure, is the same in each state. The constants, functions and predicates have the same interpretation in each state. Every state can independently assign values to the state variables from the domain. The meaning of logical constants are those from PL, PTL and FOL, (except that of the quantifiers). The meaning quantifiers depend on the temporal context:

$$\models_{\mathcal{M},<s_0,\ldots>} \Box\exists x\phi(x) \quad \text{iff} \quad \forall i \geq 0,\, \exists b_i \in \mathcal{U}\,, \models_{\mathcal{M},<s_i,\ldots>} \phi(b_i) \tag{C.17}$$

$$\models_{\mathcal{M},<s_0,\ldots>} \exists x\Box\phi(x) \quad \text{iff} \quad \exists b \in \mathcal{U},\, \forall i \geq 0\,, \models_{\mathcal{M},<s_i,\ldots>} \phi(b) \tag{C.18}$$

The looping problem of a turing machine can be expressed as a formula in FOTL, this formula is valid iff the turing machine loops. Hence FOTL is undecidable [**?**]. FOTL with addition and multiplication is strongly incomplete, since the theory of numbers is finitely axiomatizable in it [**?**].

Appendix D

Temporal Logic and PCP

D.1 Temporal Logics

PTL is an extension of Propositional logic by adding the following logical operation symbols $\{\Box, \Diamond, \bigcirc\}$ called *henceforth, eventually* and *next*, operations respectively. A model for PTL is an infinite linear sequence of worlds $< s_0, s_1, \ldots >$, where each world is a propositonal logic structure. The temporal operators are interpreted as follows:

$$\models_{s_i} \Box\phi \quad \text{iff} \quad \forall k \geq i : \models_{s_k} \phi \tag{D.1}$$

$$\models_{s_i} \bigcirc\phi \quad \text{iff} \quad \models_{s_{i+1}} \phi \tag{D.2}$$

$$\Diamond\phi \quad \overset{\text{def}}{=} \quad \neg\Box\neg\phi \tag{D.3}$$

D.2 Two Dimension TL

A model for 2D-TL is an infinite array of worlds: $\{s_{i,j} : 0 \leq i, j\}$, where each world is a propositional logic structure. The following temporal operators : $\{\Box_X, \Box_Y, \bigcirc_X, \bigcirc_Y\}$ relate the worlds. These temporal operators are interpreted as follows:

$$\models_{s_{i,j}} \Box_X\phi \quad \text{iff} \quad \forall k \geq i : \models_{s_{k,j}} \phi \tag{D.4}$$

$$\models_{s_{i,j}} \Box_Y\phi \quad \text{iff} \quad \forall l \geq j : \models_{s_{i,l}} \phi \tag{D.5}$$

$$\models_{s_{i,j}} \bigcirc_X\phi \quad \text{iff} \quad \models_{s_{i+1,j}} \phi \tag{D.6}$$

$$\models_{s_{i,j}} \bigcirc_Y\phi \quad \text{iff} \quad \models_{s_{i,j+1}} \phi \tag{D.7}$$

The symbols: $\{\Box, \Diamond, \bigcirc, \Diamond_X, \Diamond_Y\}$ are defined as follows:

$$\Box\phi \quad \overset{\text{def}}{=} \quad \Box_X\Box_Y\phi \tag{D.8}$$

$$\Diamond\phi \overset{\text{def}}{=} \neg\Box\neg\phi \tag{D.9}$$
$$\bigcirc\phi \overset{\text{def}}{=} \bigcirc_X \bigcirc_Y \phi \tag{D.10}$$
$$\Diamond_X\phi \overset{\text{def}}{=} \neg\Box_X\neg\phi \tag{D.11}$$
$$\Diamond_Y\phi \overset{\text{def}}{=} \neg\Box_Y\neg\phi \tag{D.12}$$

The following relations hold:

$$\models_{s_{i,j}} \Box\phi \quad \text{iff} \quad \forall k \geq i, \forall l \geq j :\models_{s_{k,l}} \phi \tag{D.13}$$
$$\models_{s_{i,j}} \Diamond\phi \quad \text{iff} \quad \exists k \geq i, \exists l \geq j :\models_{s_{k,l}} \phi \tag{D.14}$$
$$\models_{s_{i,j}} \bigcirc\phi \quad \text{iff} \quad \models_{s_{i+1,j+1}} \phi \tag{D.15}$$
$$\models_{s_{i,j}} \Diamond_X\phi \quad \text{iff} \quad \exists k \geq i :\models_{s_{k,j}} \phi \tag{D.16}$$
$$\models_{s_{i,j}} \Diamond_Y\phi \quad \text{iff} \quad \exists l \geq j :\models_{s_{i,l}} \phi \tag{D.17}$$

$$\models \Box_X\Box_Y\phi \iff \Box_Y\Box_X\phi \tag{D.18}$$
$$\models \Diamond_X\Diamond_Y\phi \iff \Diamond_Y\Diamond_X\phi \tag{D.19}$$
$$\models \bigcirc_X \bigcirc_Y \phi \iff \bigcirc_Y \bigcirc_X \phi \tag{D.20}$$
$$\models \neg \bigcirc \phi \iff \bigcirc\neg\phi \tag{D.21}$$

D.3 Post Correspondence Problem

It is undecidable whether a given instance of Post Correspondence Problem (PCP) has a solution. Let Σ be a set of alphabets. Given a subset of pairs of nonempty strings on Σ:

$$\{< \alpha_1, \beta_1 >, \ldots, < \alpha_n, \beta_n >\} \subset \Sigma^+ \times \Sigma^+$$

The problem is to find a match (solution of the PCP) of a sequence of indices:

$$\ll i_1 \ldots i_k \gg: 1 \leq i_j \leq n, 1 \leq j \leq k$$

such that [1] [2]:

$$\alpha_{i_1} \cdots \alpha_{i_k} = \beta_{i_1} \cdots \beta_{i_k}$$

Example D.1. Instance of PCP: Let $\Sigma = \{c, d\}$ and

$$A = \ll \alpha_1, \alpha_2, \alpha_3 \gg = \ll cdcc, ccc, dccd \gg \tag{D.22}$$
$$B = \ll \beta_1, \beta_2, \beta_3 \gg = \ll cdc, ddd, cdccd \gg \tag{D.23}$$

[1] Note that k has to be finite, an infinite sequence is not considered to be a match. The system $A = \ll c \gg$ and $B = \ll cc \gg$ has no solution.

[2] There is a simpler decidable problem $A^+ \cap B^+ = \emptyset$, where only the alphabets are matched and not the indices.

Then $\ll 1, 3 \gg$ is a match:

$$\begin{aligned} cdcc.dccd &= \alpha_1.\alpha_3 = && \text{(D.24)} \\ cdc.cdccd &= \beta_1.\beta_3 && \text{(D.25)} \end{aligned}$$

D.4 Coding PCP in 2D-TL

Given an instance of PCP: $\Sigma = \{c, d\}$ and

$$\begin{aligned} A &= \ll \alpha_1, \ldots, \alpha_n \gg && \text{(D.26)} \\ B &= \ll \beta_1, \ldots, \beta_n \gg && \text{(D.27)} \\ &\quad \text{where } \alpha_i, \beta_i \in \Sigma^+, 1 \leq i \leq n. \end{aligned}$$

Let $k = \lceil log_2(n) \rceil$, we use propositions

$$\{P, Q, D, M, A_1 \ldots A_k, B_1 \ldots B_k\}$$

to code the given PCP as a formula *match* of this logic. The formula *match* will be satisfiable if and only if the given PCP has a solution. Since solving a PCP is undecidable, the satisfiability problem for our logic is undecidable. [3]

We define the formulae:

$$\begin{aligned} up(\phi) &\overset{\text{def}}{=} \Box_X((\phi \rightarrow \Box_Y \phi) \wedge (\neg\phi \rightarrow \Box_Y \neg\phi)) && \text{(D.28)} \\ right(\phi) &\overset{\text{def}}{=} \Box_Y((\phi \rightarrow \Box_X \phi) \wedge (\neg\phi \rightarrow \Box_X \neg\phi)) && \text{(D.29)} \\ code_A(c) &\overset{\text{def}}{=} P && \text{(D.30)} \\ code_A(d) &\overset{\text{def}}{=} \neg P && \text{(D.31)} \\ code_A(\sigma_0 \ldots \sigma_t) &\overset{\text{def}}{=} \bigwedge_{i=0}^{t} \bigcirc_X^i code_A(\sigma_i) \\ &\quad \text{where } \sigma_i \in \Sigma, 0 \leq i \leq t && \text{(D.32)} \\ code_B(c) &\overset{\text{def}}{=} Q && \text{(D.33)} \\ code_B(d) &\overset{\text{def}}{=} \neg Q && \text{(D.34)} \end{aligned}$$

[3]Notation:

$$\bigcirc^n \phi \overset{\text{def}}{=} \underbrace{\bigcirc \cdots \bigcirc}_{n\,times} \phi$$

$$\|\alpha\| \overset{\text{def}}{=} \text{number of alphabets in } \alpha.$$

$$
\begin{aligned}
code_B(\sigma_0 \ldots \sigma_t) &\overset{\text{def}}{=} \bigwedge_{i=0}^{t} \bigcirc_Y^i code_B(\sigma_i) \\
&\quad \text{where } \sigma_i \in \Sigma, 0 \leq i \leq t && \text{(D.35)}
\end{aligned}
$$

We use propositions $\{A_1 \ldots A_k\}$ to binary code the indices of $\alpha_i, 1 \leq i \leq n$. Similarly $\{B_1 \ldots B_k\}$ binary codes indices of of $\beta_i, 1 \leq i \leq n$.

$$
\begin{aligned}
bincode_A(0) &\overset{\text{def}}{=} \neg A_k \wedge \ldots \wedge \neg A_1 && \text{(D.36)} \\
bincode_A(1) &\overset{\text{def}}{=} \neg A_k \wedge \ldots \wedge \neg A_2 \wedge A_1 && \text{(D.37)} \\
\bar{A} \equiv t &\overset{\text{def}}{=} bincode_A(t) && \text{(D.38)} \\
bincode_B(0) &\overset{\text{def}}{=} \neg B_k \wedge \ldots \wedge \neg B_1 && \text{(D.39)} \\
bincode_B(1) &\overset{\text{def}}{=} \neg B_k \wedge \ldots \wedge \neg B_2 \wedge B_1 && \text{(D.40)} \\
\bar{B} \equiv t &\overset{\text{def}}{=} bincode_B(t) && \text{(D.41)}
\end{aligned}
$$

$$
\begin{aligned}
code_\alpha(\alpha_i) &\overset{\text{def}}{=} code_A(\alpha_i) \wedge \bar{A} \equiv i \wedge \bigwedge_{j=1}^{\|\alpha_i - 1\|} (\bigcirc_X^i \bar{A} \equiv 0) && \text{(D.42)} \\
code_\beta(\beta_i) &\overset{\text{def}}{=} code_B(\beta_i) \wedge \bar{B} \equiv i \wedge \bigwedge_{j=1}^{\|\beta_i - 1\|} (\bigcirc_Y^i \bar{B} \equiv 0) && \text{(D.43)}
\end{aligned}
$$

The next two formulae lay out the α horizontally, and β vertically. They assert every α_i is followed by another α_j identically in every row, and every β_i is followed by another β_j identically in every column.

$$
\begin{aligned}
arrange_A &\overset{\text{def}}{=} \Box_X \left(\bigwedge_{\alpha_i \in A} (code_\alpha(\alpha_i) \rightarrow \bigvee_{\alpha_j \in A} ((\bigcirc_X)^{\|\alpha_i\|} code_\alpha(\alpha_j))) \right) \\
&\wedge \quad (\bigvee_{\alpha_i \in A} code_\alpha(\alpha_i)) \wedge up(P) \wedge \bigwedge_{i=1}^{k} up(A_i) && \text{(D.44)} \\
arrange_B &\overset{\text{def}}{=} \Box_Y \left(\bigwedge_{\beta_i \in B} (code_\beta(\beta_i) \rightarrow \bigvee_{\beta_j \in B} ((\bigcirc_Y)^{\|\beta_i\|} code_\beta(\beta_j))) \right) \\
&\wedge \quad (\bigvee_{\beta_i \in B} code_\beta(\beta_i)) \wedge right(Q) \wedge \bigwedge_{i=1}^{k} right(B_i) && \text{(D.45)}
\end{aligned}
$$

Now we assert formulae to match them:

$$\begin{aligned} match_\Sigma &\overset{\text{def}}{=} \Box(D \to (\bigcirc D \land \bigcirc_X \Box_X \neg D \land \bigcirc_Y \Box_Y \neg D)) \\ &\land \quad \Box(D \to (P \iff Q)) \end{aligned} \tag{D.46}$$

$$\begin{aligned} match_i &\overset{\text{def}}{=} (M \land \bar{A} \equiv \bar{B}) \to (\bigcirc M \land \bigcirc_X \Box_X \neg M \land \bigcirc_Y \Box_Y \neg M) \\ &\land \quad (M \land \bar{A} \equiv 0 \not\equiv \bar{B}) \to (\bigcirc_X M \land \bigcirc_Y \Box_Y \neg M) \\ &\land \quad (M \land \bar{A} \not\equiv 0 \equiv \bar{B}) \to (\bigcirc_Y M \land \bigcirc_X \Box_X \neg M) \end{aligned} \tag{D.47}$$

$$\begin{aligned} match &\overset{\text{def}}{=} arrange_A \land arrange_B \\ &\land \quad match_\Sigma \land \Box match_i \\ &\land \quad (\bar{A} \equiv \bar{B} \not\equiv 0 \land M \land D) \\ &\land \quad \bigcirc\Diamond(\bar{A} \equiv \bar{B} \not\equiv 0 \land M \land D) \end{aligned} \tag{D.48}$$

Appendix E

Acknowledgement

I thank Dr. G. Venkatesh for his guidance, constant help in clearing my doubts, and giving direction to the problems.