Temporal Logic with Fixed Points

Behnam Banieqbal and Howard Barringer*

Department of Computer Science
University of Manchester
Oxford Road
Manchester, M13 9PL

1 Introduction

The application of propositional temporal logic (PTL) in Computer Science has been extensively studied over the past decade. In particular, it has been shown to be a most useful tool for specifying and verifying safety and liveness properties of concurrent systems, see for example [MP81, Lam83, CES83]. Unfortunately, for several applications PTL is not expressive enough. In [SCFG82], for example, it is shown that PTL can not specify the behaviour of an unbounded FIFO buffer with indistinct messages. Although PTL is in this sense weak, the language is still attractive from a specification and verification standpoint; it is, after all, decidable. Notwithstanding this, it is natural to search for "decidable extensions". Several such "decidable extensions" for PTL have been examined and here we mention three in particular.

1. Pierre Wolper, [Wol83], noted that it was not possible to express in PTL the fact that a proposition had to be true on every even moment. He gave a simple way of extending PTL, by basing logics on regular grammars, and obtained a family of temporal logics, referred to as ETL. He showed that ETL languages are decidable and that the family has the expressiveness of ωRE.

2. In [BKP84] where a technique for achieving compositional temporal semantics and proof systems for concurrent languages based on shared variables is presented, PTL was modified to handle finite sequences as models. In particular a *chop* operator, $\mathcal{C}$, was introduced to compose sequentially two finite sequences.[1] Furthermore, an iterated version of chop, $\mathcal{C}^*$ was also used to simplify the temporal description of while loops. Issues of decidability and an axiomatisation of PTL + $\mathcal{C}$ have been considered in the thesis [Ros85].

3. Also, in [BKP84, BKP85, BKP86] PTL is further extended with fixed point notation to name directly maximal and minimal solutions to temporal equations. The requirement for such solutions arises naturally when we look for temporal semantics to recursive procedures. For example, given procedure P defined by body $B(P)$, let x stand for the temporal semantics of P and then $f(x)$ be a temporal formula, dependent on x, which represents the semantics of the body $B(P)$, then clearly $x \Leftrightarrow f(x)$. We use notation $\nu x.f(x)$ ($\mu x.f(x)$) to refer to the maximal (minimal) solutions of $x \Leftrightarrow f(x)$. We now commonly refer to the extension of PTL with just ν and μ-forms as νTL.

Although it was believed that νTL and the ETL family were expressively equivalent, the ETL formalism is not well-suited for the description of arbitrary recursive procedures. Generally, it would be necessary

*The work reported here has been supported under Alvey PRJ/SE/054 (SERC grant GR/D/57942)

[1]The use of $\mathcal{C}$ was not necessary for obtaining compositionality, it just sweetened the notation!

to define a grammar operator that corresponded to each recursive procedure; thus a semantics could not be given in a compositional and syntax-directed fashion. However, ETL had a decision procedure and axiomatisation which was lacking in the νTL case. In this paper we first give a rigorous definition of νTL and follow with a decision procedure for the language. This shows that νTL is decidable and expressively complete with respect to ωRE. Our algorithm involves a novel graph algorithm which we believe to be of interest in its own right. Since presentation of our result at this colloquium, Vardi [Var88] has constructed a decision procedure for an extension of νTL which includes past time operators. His methods are based on the construction of 2-way Büchi automata. For νTL the two methods are equivalent in time complexity. However Vardi's construction is polynomial space whereas our method runs in exponential space.

2 The Language νTL

2.1 Syntax and Semantics

A formula of the fixed point temporal language νTL is constructed from a set of propositions and variables by means of three types of constructors:

- The usual boolean connectives $\vee, \wedge, \neg, \Rightarrow$, etc.
- The unary next operator $\bigcirc$.
- The maximal and the minimal fixed point constructs $\nu x.f, \mu x.f$ where f is a νTL formula.

We shall focus on the fixed point construction. This notion is in fact a generalisation of the grammar operators of Wolper's ETL language. In $\nu x.f$ or $\mu x.f$, the operator ν or μ binds all the occurrences of x in f which are not already bound. Thus for instance in $\nu x.\{(a \wedge \bigcirc x) \vee \mu \mathbf{x}.(b \wedge \bigcirc \mathbf{x})\}$ the x is bound by ν and the $\mathbf{x}$ is bound by μ. A bound variable plays the role of a place holder and therefore can be freely renamed. This implies that by renaming the variables if necessary, we can make the distinct fixed point operators bind distinct variables. We assume from now on that every formula satisfies this property. For instance, the equivalent of the above formula would be $\nu x.\{(a \wedge \bigcirc x) \vee \mu y.(b \wedge \bigcirc y)\}$. A non-bound variable is called free. A formula without free variables is termed closed otherwise it is open.

As with other propositional temporal languages, a model σ is an infinite sequence of states $\sigma = \sigma_0\sigma_1\cdots$ where each state assigns truth values to all the propositions. As usual $\sigma^i = \sigma_i\sigma_{i+1}\cdots$ denotes the i-tail of the sequence. Our approach to the semantics of the fixed points is to work with the set of models of a formula rather than an individual model. Let S be the total set of sequences and let $P(S)$ be the power set of S, that is all the subsets of S. We will show how to pick from $P(S)$ the models M_f of a closed formula f. Then $\sigma \models f$ iff $\sigma \in M_f$. Now the satisfaction relation is defined only for closed formulae and only a subset of them. In fact some closed fixed points with negations on the bound variables do not exist. Examples are given below. It will be shown that, in the absence of essential negation on the fixed points, they do exist. Now a nested formula of a closed formula has free variables when regarded on its own. So our semantics must be able to deal with open formulae. Our approach is to regard an open formula as a map where the free variables can each be assigned a set from $P(S)$ and the result is again in $P(S)$. To be precise if f is free on n variables, then we construct a map $[f] : P(S)^n \to P(S)$. Closed formulae can be included in this scheme by setting $[f]() = M_f$ i.e. $[f]$ is the constant map. The essence of our definition is the recursive construction of [.]:

1. $[p]() = \{\sigma | \sigma(p) = \text{ true}\}$.
2. $[x]$ is the identity map.
3. For the connectives, we show how $\wedge$ is done. The others are similar. Suppose f and g are free on m and n variables, respectively, of which r are in common. Then $f \wedge g$ has $m+n-r$ free variables. Let us order the variables such that f shares its last r free variables with the first r of g. Then

$$[f \wedge g](M_1, \cdots, M_{m+n-r}) = [f](M_1, \cdots, M_m) \cap [g](M_{m-r+1}, \cdots, M_{m+n-r})$$

In the right hand side if either m or n is zero then the function application returns the model set as before.

4. $[\bigcirc f] = T \circ [f]$ where $T : P(S) \to P(S)$ is defined by $T(M) = \{\sigma | \sigma^1 \in M\}$.

5. Now we deal with the fixed points. If x is not free in f - that is it does not occur in f - then $\nu x.f$ and $\mu x.f$ are identical with f. Otherwise suppose that f has $n \geq 1$ free variables. Order them such that x is the first one. Let $M_i \in P(S)$, $2 \leq i \leq n$ be $n-1$ arbitrary model sets. Now $[f](.,M_2,\cdots,M_n)$ is a map on $P(S)$. Whenever this map has a maximum fixed point M or a minimum fixed point M' for all $n-1$ tuples from $P(S)$, then $\nu x.f$ or $\mu x.f$ is defined and $[\nu x.f](M_2,\cdots,M_n) = M$ or $[\mu x.f](M_2,\cdots,M_n) = M'$.

We can immediately derive some basic properties of νTL from the construction just given. In the following theorem and beyond $f(x/y)$ is used to denote the formula obtained from f by replacing all the occurrences of y with x.

Theorem 1 *Given that the fixed points exist, we have*

1. $\sigma \models f \wedge g$ *iff* $\sigma \models f$ *and* $\sigma \models g$. *Similarly for other connectives*

2. $\mu x.f = \neg \nu x.\neg f(\neg x/x)$ *as maps. In particular if the fixed point formula is closed then* $\mu x.f \Leftrightarrow \neg \nu x.\neg f(\neg x/x)$

3. $\nu x.f \Leftrightarrow f(\nu x.f/x)$, $\mu x.f \Leftrightarrow f(\mu x.f/x)$. *The process of replacing the left hand side by the right hand side is known as "unwinding".*

4. *Suppose* f, t *are* νTL *formulae such that* $t \Leftrightarrow f(t/x)$ *where* f *is free on* x. *Then* $t \Rightarrow \nu x.f$, $\mu x.f \Rightarrow t$. *This is the so called fixed point induction.*

Proof The first part is easy. For the second part, employing the notation above, we observe that the fixed points of $[f](.,M_2,\cdots,M_n)$ and $[\neg f(\neg x/x)](.,M_2,\cdots,M_n)$ are complementary in S. This implies the minimum fixed point of one is the maximum fixed point of the other. For the third equation, we have $f(M,M_2,\cdots,M_n) = M$ and $[f(\nu x.f/x)](M_2,\cdots,M_n) = f([\nu x.f](M_2,\cdots,M_n),M_2,\cdots,M_n) = f(M,M_2,\cdots,M_n) = M$ as required. The fourth part is left to the reader.

2.2 Existence Theorem

The previous section left the question of the existence of fixed points open. We will deal with this problem here. The curried map $[f](.,M_2,\cdots,M_n)$ operates on the complete lattice $P(S)$ under the set inclusion. It has been shown by Tarski [Tar55] that, for a monotonic function on complete lattices, the minimum and maximum fixed points exist.

Theorem 2 *A monotonic function* f *on a complete lattice* L *has unique maximum and minimum fixed points.*

Monotonic means that $x \leq y$ implies $f(x) \leq f(y)$. Actually we can show how to construct the fixed points (abstractly). Construct two sublattices $\{f_\wedge^\alpha(1)\}$ and $\{f_\vee^\alpha(0)\}$ of L where 0 and 1 are the least and the greatest element of L and the maps are inductively defined for all ordinals α as follows

$$f_\wedge^\alpha(x) = \begin{cases} f(f_\wedge^{\alpha-1}(x)) & \text{if } \alpha \text{ is not limiting} \\ \bigcap_{\beta<\alpha} f_\wedge^\beta(x) & \text{if } \alpha \text{ is limiting} \end{cases}$$

$$f_\vee^\alpha(x) = \begin{cases} f(f_\vee^{\alpha-1}(x)) & \text{if } \alpha \text{ is not limiting} \\ \bigcup_{\beta<\alpha} f_\vee^\beta(x) & \text{if } \alpha \text{ is limiting} \end{cases}$$

Now we can apply the above theorem to these sublattices to get

Corollary 1 *Let* f *be a monotonic function on a complete lattice. Then there exists two ordinals* α *and* α' *and the maximum and the minimum fixed points are* $f_\wedge^\alpha(1)$ *and* $f_\vee^{\alpha'}(0)$

Corollary 2 *Let f and g be two monotonic functions on a complete lattice and $f(x) \leq g(x)$ for all x. Then their maximum and minimum fixed points are related similarly.*

To ensure that the νTL formula f exists, we have to make sure that all the maps which arise in the construction of [f] are monotonic. It is clear that non-monotonicity stems from the negation symbol preceding a variable or a fixed point construction. But by Theorem 1 we can push the latter negations in. So we can assume that this has been done. The required condition says then that a negation symbol should occur only before a proposition. This guarantees monotonicity because all the constructs other than negation preserve monotonic behaviour. For instance let us suppose that $[f]$ on $P(S)^n$ is montonic on all the variables. Then so is the curried function $[f](.,M_2,\cdots)$. So $[\nu x.f]$ and $[\mu x.f]$ both exist. Also they are monotonic on all their variables. This follows from Corollary 2. From now on we will restrict νTL to consist of positive formulae (that is the negation symbol occurs only before propositions) and those which reduce to positive form when the negation symbols are pushed through. For the rest of the paper our aim is to develop a decision procedure for this logic.

2.3 Examples

To illustrate the discussions so far, we will consider some examples. Some fixed points will be monotonic and some not. Among the non-monotonic ones, some do not exist, while the others do and are equivalent to monotonic formulae.

1. Consider $f = p \wedge \bigcirc x$. To see what $\nu x.f$ is, we look at [f]. Given $M \in P(S)$ we have $[f](M) = \{\sigma | \sigma_0(p) = \text{ true and } \sigma^1 \in M\}$. Suppose $M = [f](M)$. Then $\sigma \in M$ implies $\sigma_0(p) =$ true and $\sigma^1 \in M$. Thus $\sigma_i(p) =$ true for all i. Let $N = \{\sigma | \sigma_i(p) =$ true for all $i\}$. Then $M \subseteq N$ and $[f](N) = N$. This proves that N is exactly the model set of $\nu x.f$. In TL the equivalent formula is $\Box p$.

2. $\mu x.(p \vee \bigcirc x) \Leftrightarrow \neg \nu x.\neg(p \vee \bigcirc \neg x) \Leftrightarrow \neg \nu x.(\neg p \wedge \bigcirc x)$ This is $\neg \Box \neg p \Leftrightarrow \Diamond p$.

3. $\mu x.(p \wedge \bigcirc x) \Leftrightarrow$ false.

4. $\nu x.(p \wedge \bigcirc x \wedge \bigcirc^2 \neg x) \Leftrightarrow$ false.
   For writing t for the fixed point, we have $t \Leftrightarrow p \wedge \bigcirc t \wedge \bigcirc^2 \neg t$. Substitute for $\bigcirc t$ in the right hand side gives $t \Leftrightarrow p \wedge \bigcirc p \wedge \bigcirc^2 t \wedge \bigcirc^3 \neg t \wedge \bigcirc^2 \neg t \Leftrightarrow$ false.

5. $t = \nu x.(p \wedge \bigcirc \neg x)$ does not exist.
   For let σ be a model such that $\sigma \models p$ and $\sigma^1 = \sigma$. Then $\sigma \models t$ iff $\sigma \models \neg t$, a contradiction.

6. $\nu x.(p \wedge \bigcirc \neg p \wedge \bigcirc \neg x \wedge \bigcirc^2 x) \Leftrightarrow \nu x.(p \wedge \bigcirc \neg p \wedge \bigcirc^2 x)$.
   The reader should convince himself that the two formulae on the left and the right hand side (without the νx.) have the same fixed point sets. In other words, the $\bigcirc \neg x$ term is redundant. The fixed point on the right hand side is the set of models where p is true on exactly the even states.

7. $\nu x.(p \wedge x \wedge \bigcirc \neg x)$ does not exist. For it can be shown that $[p \wedge x \wedge \bigcirc \neg x]$ has fixed point sets (e.g. the empty set), but it does not have a unique maximum fixed point set.

2.4 Reduction to Guarded form

From now on we confine ourselves to the monotone case. Thus the bound variables are positive and the fixed points are either maximal or minimal.

We will call the occurrence of a variable guarded if a $\bigcirc$ operator precedes it. We show that unguarded occurrences of bound variables can be eliminated without affecting the semantics. There are two rules for ν,

1. $\nu x.(f \vee (x \wedge g)) \Leftrightarrow \nu x.(f \vee g)$. The implication from left to right follows from Corollary 2. For the reverse implication let t denote the right hand fixed point. Then $t \Leftrightarrow f(t/x) \vee g(t/x)$. So $t \Leftrightarrow f(t/x) \vee (t \wedge g(t/x))$. Thus $t \Rightarrow \nu x.(f \vee (x \wedge g))$.

2. $\nu x.(x \vee f) = \text{true}$.

Similar rules hold for the minimal fixed point. By unwinding and applying these rules we can eliminate any unguarded variables, e.g.:

$$\nu x.(a \wedge x \wedge \mu y.(b \vee y \vee (x \wedge \bigcirc y))) = \nu x.(a \wedge \mu y.(b \vee (x \wedge \bigcirc y))) =$$

$$\nu x.(a \wedge (b \vee (x \wedge \bigcirc \mu y.(b \vee (x \wedge \bigcirc y))))) = \nu x.((a \wedge b) \vee (a \wedge \bigcirc \mu y(b \vee (x \wedge \bigcirc y)))).$$

3 Decision Procedure

The key to the νTL decision procedure is the Tarski-Knaster fixed point theorem which allows us, in principle, to decide if a fixed point formula is valid on a model. One snag is that more than a finite number of iterations may be required to conclude the procedure. Another problem is that all the models have to be checked for satisfiability. To get round the first problem, we show that if a formula is satisfiable, then it has a special, so called eventually periodic, model for which the testing of validity is finitary. Other logics such as extended tmporal logic ETL, quantified propositional temporal logic QPTL also share this property [Wol82, SC85]. The solution of the second problem involves a marking algorithm which, in our opinion, is interesting in its own right.

3.1 The Approximation Method

Let σ be a model. We consider the question of checking if a formula holds over σ. We start with an informal discussion. It is clear that the cases of interest are the fixed point formulae. Let $\nu x.f(x)$ be a maximal fixed point which we abbreviate to x. We abbreviate $f^{\alpha}_{\wedge}(\text{true})$ to x^{α}. Thus $\sigma \models x$ iff $\sigma \models x^{\alpha}$ for all α. So we must check the validity of x^{α} on σ. If α is not limiting, then $x^{\alpha} = f(x^{\alpha-1})$. Since f in general involves temporal operators, it is necessary to know the validity of $x^{\alpha-1}$ on all suffices of σ. The starting value is $x^0 = \text{true}$ which holds everywhere. For a nonlimiting ordinal α we can determine the suffices where $x^{\alpha} = f(x^{\alpha-1})$ holds by recursion on the depth of the fixed point nesting. Thus in the case when $f(x)$ involves further fixed points, this step requires a transfinite amount of iterations. If α is limiting, then $x^{\alpha} = \wedge_{\beta<\alpha} x^{\beta}$, so that x^{α} holds on σ^i if σ^i receives all x^{β} for $\beta < \alpha$. The Tarski-Knaster theorem guarantees that there is an ordinal α such that exactly the same states receive x^{α} as $x^{\alpha+1}$ and then x holds at those points.

The situation for $\mu x.f(x)$ is similar. One starts with $x^0 = \text{false}$ and builds up x^{α} inductively. Then x is valid on σ^i if it receives some x^{α}.

The approximation procedure may require transfinite ordinals to complete as illustrated by the following example. Consider the formula

$$f = \nu x.\{(p \wedge (q \vee \bigcirc x)) \vee \bigcirc \mu y.(\nu z.(x \wedge \bigcirc z) \vee \bigcirc y)\}$$

There are two propositions p and q. The model is σ with the following assignments. p is true on every state except σ_0. q is true on σ_{i^2} and false everywhere else. Now we work out where x^i is valid. As mentioned x^0 is valid nowhere. It is easy to see that x^1 is valid on all indices i^2 except 0. Then x^2 is valid on indices $i^2, i^2 - 1$ except 0, x^3 is valid on indices $i^2, i^2 - 1, i^2 - 2$ except 0, etc. The point is that in each step y^{α} is invalid everywhere because it requires x^{i-1} to be valid on every state of some tail of σ. This does not happen because the distances between the squares goes to infinity. Now what about x^{ω}, the first infinite ordinal? As every state but σ_0 will receive some x^i, so x^{ω} is valid on all the indices except 0. In the next iteration, the y fixed point comes to life for the first time, making $x^{\omega+1}$ valid on every suffix. In particular, $\sigma \models f$.

The authors are grateful to Orna Lichtenstein for bringing, via Amir Pnueli, this fascinating example to their attention. The first version of this paper had to undergo substantial revision as a result of this fact as a finitary decision procedure has to get round this problem. We are thankful to Amir Pnueli for pointing out the necessary idea. The key ingredient is that σ has been specifically chosen to show this complicated behaviour. In fact we will show that, amongst the models of any formula, there always occur models with a repeating structure. The approximation method behaves finitary for such models.

3.2 Hintikka Structures

To elaborate the approximation procedure precisely, we define the concept of a Hintikka structure.

Let σ be a model and let f be a νTL formula. We will assume throughout that the bound variables of f are named distinctly, by renaming them if necessary. We extract a system Σ of equations, one for each fixed point, of the form

$$x_i = f_i(x_1, \ldots, x_n)$$

where f_i is a formula defining the body of the fixed point x_i and $x_1 \ldots x_n$ are all the fixed points involved in f. The formula f_i involves only logical connectives, propositions, x_i and $\bigcirc$.

The other information from f is the scope partial order. We say x is in the scope of y if the definition of x in f is syntactically nested in the definition of y. This order is essential in the characterization of good structures later on.

We now define a Hintikka structure (H-structure) on σ generated by f (or an f Hintikka structure). It is a labelling $L(\sigma_i)$ of the states σ_i of σ by subformulae of f satisfying the following properties.

1. $f \in L(\sigma_0)$.
2. If $f_1 \wedge f_2 \in L(\sigma_i)$ then $f_1, f_2 \in L(\sigma_i)$.
3. If $f_1 \vee f_2 \in L(\sigma_i)$ then either f_1 or f_2 is in $L(\sigma_i)$.
4. If $p \in L(\sigma_i)$ then p is true on σ_i.
5. If $x \in L(\sigma_i)$ then $g(\mathbf{x}) \in L(\sigma_i)$ where $x = g(\mathbf{x})$ is the equation in Σ defining x.
6. If $\bigcirc g \in L(\sigma_i)$ then $g \in L(\sigma_{i+1})$.

We will call a formula of the type $\bigcirc^i x$ a (fixed point) variable. Item 5 above provides the expansion of a fixed point formula. This expansion together with the other items provide a parenthood relation on the variables of successive states. Namely $v_1 \in L(\sigma_i)$ is a parent of $v_2 \in L(\sigma_{i+1})$ when either $v_1 = \bigcirc v_2$ or v_1 is not preceded by $\bigcirc$ and $\bigcirc v_2$ occurs in the defining equation of v_1 (or rather in a disjunct thereof occuring in $L(\sigma_i)$).

Lemma 1 *If $\sigma \models f$ then there is an f Hintikka structure on σ.*

Proof. By structural induction. The cases of interest are the fixed points. We illustrate a maximal fixed point. Let $\sigma \models \nu x.f(x)$. Applying the ideas in section 2, let α be the ordinal for which the labelling by x^α and $x^{\alpha+1}$ of σ are identical (so that $\nu x.f(x) = f^\alpha_\wedge(\text{true})$). Set $I = \{i \ ; \ \sigma_i \text{ receives } x^\alpha\}$. To construct the H-structure let σ' be obtained from σ by extending the set of propositions to include x and putting $\sigma'_i(x) = \text{true}$ precisely for $i \in I$. Then by the approximation procedure $\sigma'^i \models f(x)$ precisely for $i \in I$. By induction we get a Hintikka structure generated by $f(x)$ on σ'^i. The union of all these structures gives the required Hintikka structure. A similar procedure works for the minimal fixed point.

The converse of above does not hold. It will be valid under restrictions on the class of H-structures to be discussed later.

3.3 νTL to QPTL Translation

QPTL is the extension of linear temporal logic with quantification over state variables. The reader can refer to [Wol82] for further information. Here we will establish a syntactic translation. The basis for it is the observation that a fixed point formula on a model can be encoded by a proposition taking true values at the points where the formula is true. Let us make this idea precise.

Lemma 2 *Let f be a νTL formula and let g be a closed subformula of f in the sense that all the variables in g are defined within g. Let σ be a model and let σ' be an extension of σ with a proposition p, not occuring in f, which is given the interpretation of g (i.e. $\sigma'^i \models p$ iff $\sigma^i \models g$). Then $\sigma^i \models f$ iff $\sigma'^i \models f[p/g]$ where the last formula results from f by replacing g by p.*

Proof. This follows from the approximation procedure.

Lemma 3 *Let $\nu x.f$ ($\mu x.f$) be a fixed point formula. Let σ be a model and let σ' be an extension of it with a proposition x which takes the interpretation of $\nu x.f$ ($\mu x.f$) over σ. Then*

$$\sigma'^i \models x \text{ iff } \sigma'^i \models f$$

where x in f is to take its propositional significance.

Proof. Observe that

$$\nu x.f \Leftrightarrow f[\nu x.f/x], \quad \mu x.f \Leftrightarrow f[\mu x.f/x]$$

Now apply lemma 1 to the right-hand sides.

Lemma 4 *Let $f(x)$ be a vTL formula involving a free variable x. Let σ be a model, then extend σ to σ' by assigning values to x in such a way that whenever $\sigma'^i \models x$ then $\sigma'^i \models f(x)$. Then in fact $\sigma'^i \models x$ implies $\sigma^i \models \nu x.f(x)$.*
For the minimal case, σ' must be such that whenever $\sigma'^i \models f(x))\}$ then $\sigma'^i \models x$. Then $\sigma^i \models \mu x.f(x)$ implies $\sigma'^i \models x$.

Proof. In the approximation procedure, it is easy to prove (by transfinite induction) that $\sigma'^i \models x$ implies $\sigma'^i \models x^\alpha$ for all α (in the minimal case $\sigma'^i \models x^\alpha$ implies $\sigma'^i \models x$). The result follows from Tarski-Knaster theorem.

We can now expose the translation functor T. Remember that all the fixed point variables are to be distinct.

1. $T(f_1 \wedge f_2) = Tf_1 \wedge Tf_2$, etc.
2. $Tp = p$, $Tx = x$.
3. $T\nu x.f = \exists x.x \wedge \Box(x \Rightarrow Tf) \wedge \forall x'.\{\Box(x' \Rightarrow Tf[x'/x]) \Rightarrow \Box(x' \Rightarrow x)\}$.
4. $T\mu x.f = \exists x.x \wedge \Box(Tf \Rightarrow x) \wedge \forall x'.\{\Box(Tf[x'/x] \Rightarrow x') \Rightarrow \Box(x \Rightarrow x')\}$.

In 3 and 4 we need an extra dashed variable for the translation of each fixed point.

Theorem 3 *$\sigma \models f$ iff $\sigma \models Tf$.*

Proof The cases of interest are the fixed point formulae. We consider just the maximal case, the minimal case is similar. Let $\sigma \models \nu x.f(x)$. In $T\nu x.f(x)$ we have an existentially quantified x. We will give it the interpretation of $\nu x.f(x)$ as in Lemma 3 and denote the extended model by σ'. So $\sigma' \models x$. We have, by induction over the f structure, that for all i, $\sigma'^i \models f(x)$ iff $\sigma'^i \models Tf(x)$ where x is interpreted as a proposition in $f(x)$. By lemma 3 $\sigma'^i \models x$ iff $\sigma'^i \models f(x)$. Thus $\sigma' \models \Box(x \Rightarrow Tf(x))$

Now suppose σ'' be an extension of σ' by assignments to x' such that $\sigma'' \models \Box(x' \Rightarrow Tf(x'))$. It follows, by structural induction on f, that for all i, if $\sigma''^i \models x'$ then $\sigma''^i \models f(x')$. By lemma 4 $\sigma''^i \models x'$ implies $\sigma''^i \models \nu x'.f(x')$ for all i. So by the definition of x, $\sigma'' \models \Box(x' \Rightarrow x)$ as required. The converse statement follows directly from Lemma 4.

3.4 An Example of the Translation Procedure

Let $f(.,.)$ and $g(.,.)$ be two formulae, each with two variables, involving only logical connectives, propositions and $\bigcirc$. We have

$T\nu x.f(x, \mu y.g(x,y)) =$
$\exists x.x \wedge \Box\{x \Rightarrow [\exists y.y \wedge \Box(y \Leftarrow g(x,y)) \wedge \forall y'.(\Box(y' \Leftarrow g(x,y')) \Rightarrow \Box(y \Rightarrow y'))]\} \wedge$
$\forall x'.\{\Box\{x' \Rightarrow [\exists y.y \wedge \Box(y \Leftarrow g(x',y)) \wedge \forall y'.(\Box(y' \Leftarrow g(x',y')) \Rightarrow \Box(y \Rightarrow y'))]\} \Rightarrow \Box(x' \Rightarrow x)\}$

3.5 Eventually Periodic Models

A model σ is called eventually periodic if there exist constants $m \geq 0, n > 0$ such that $\sigma_i = \sigma_{i+n}$ for all $i \geq m$. It is clear that $\sigma^i = \sigma^{i+n}$ for all $i \geq m$.

Lemma 5 *If a vTL formula has a model then it has an eventually periodic model.*

Proof. This follows from the corresponding statement for QPTL [Wol82] and Theorem 3.

Now we consider the question of checking for the validity of a fixed point formula x ($= \nu x.f(x)$ or $\mu x.f(x)$). Since σ^i and σ^{i+n}, for all $i \geq m$, receive the same x^α it is necessary to consider only the labels of σ_i for $0 \leq i \leq m+n-1$. If I_α is the set of indices i between 0 and $m+n-1$ which receive x^α, we have

$$\{0, 1, \cdots, m+n-1\} = I_0 \supseteq I_1 \supseteq \cdots \text{ in the } \nu \text{ case,}$$

$$\emptyset = I_0 \subseteq I_1 \subseteq \cdots \text{ in the } \mu \text{ case.}$$

It follows that $I_\alpha = I_{\alpha+1}$ for some $\alpha \leq m+n$. This proves

Lemma 6 *Let σ be an eventually periodic model with m, n as above. Then for any $i \geq m+n$,*

$$\sigma \models \nu x.f(x) \text{ iff } \sigma \models f^i(true)$$

$$\sigma \models \mu x.f(x) \text{ iff } \sigma \models f^i(false)$$

3.6 Good Hintikka Structures

The next step is to relate eventually periodic models to H-structures. If σ is an eventually periodic model with m, n defined as above, then it is necessary to build the labelling $L(\sigma_i)$ only for $0 \leq i \leq m+n-1$. So whenever a formula is added to $L(\sigma_{m+k})$, $k \geq n$, it is in fact added to $L(\sigma_{m+r})$ where $0 \leq r \leq n-1$ is the remainder of k by n. In other words the Hintikka structure is built step by step on $\sigma_0 \ldots \sigma_{m+n-1}$ where the successor of σ_{m+n-1} is σ_m, etc. Such an H-structure is called *compatible*.

Lemma 7 *Let σ be an eventually periodic model of f. Then there exists a compatible Hintikka structure on σ generated by f.*

Proof The proof follows similar lines as that of Lemma 1.

The converse again does not hold. It is in fact necessary to add an extra condition to ensure that minimal fixed points are satisfied. In a Hintikka structure on a model σ, let $v_1 \in L(\sigma_i)$, $v_2 \in L(\sigma_j)$ $j > i$ be two variables. We say v_1 generates v_2 if there is a chain of parents from v_1 to v_2 (along $\sigma_i \cdots \sigma_j$) such that every variable in the chain is in the scope of v_1. We define a *good* Hintikka structure to be one in which no minimal variable generates itself infinitely often. In the case of a compatible H-structure, this means that there is no loop of the form

$$v_0 \rightarrow v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_{kn}$$

where $v_0 = v_{kn}$, the variable v_i is in $L(\sigma_{m+r})$ where r is the remainder of i mod n and finally some v_j is a minimal variable with every v_i in its scope.

We will show that with the restriction of good structures, the implication of the lemma becomes an equivalence. The construction of the compatible good H-structure is more complex, since only appropriate choices regarding formulae of the type $f_1 \vee f_2$ gives rise to a good H-structure.

3.7 The Result

Theorem 4 *Let f be an vTL formula and let σ be a model.*

1. *If σ is an eventually periodic model of f then there exists a good compatible Hintikka structure on σ generated by f.*

2. *If f generates a good Hintikka structure on σ then $\sigma \models f$.*

Proof.
Part 1: Suppose that $\sigma \models f$ and we want to establish a Hintikka structure on σ. It is done by structural induction on f. The cases of interest are the fixed points.

1. $\nu x.f(x)$. Applying the approximation procedure, let x denote $f^i_\wedge(\text{true})$ for some sufficiently large i. Then 0 occurs among the indices I which receive x. Furthermore $f(x)$ holds at exactly the same indices (where x must be treated as a proposition). Adding the good Hintikka labellings of $f(x)$ and x to the states indexed by I, we obtain a Hintikka labelling generated by x. Moreover it is good. The reason is that a bad chain in it does not involve x since x is maximal and outside the scope of all minimals. Hence it already occurs in the Hintikka labelling of some $f(x)$, contradicting that the Hintikka labellings generated by $f(x)$ are all good.

2. $\mu x.f(x)$. This is the interesting case. To build up the Hintikka structure of x, we proceed according to the approximation procesure for x, starting from x^0 = false. Whenever σ_j receives x^i, then x is added to $L(\sigma_j)$. Then $f(x^i)$ is true on σ^j (x^i to be taken as a proposition) so that we can add the good Hintikka structure for $f(x)$ on σ^j. This procedure is repeated until σ_0 receives $x = x^n$ for some n. At this point we obtain a Hintikka structure H which we prove to be good. For let us have a bad loop in H. Suppose x occurs in the loop. An instance of x generated another instance of x only when they are added in two successive steps, contradicting that x generated itself. No other minimal can occur in a bad loop since as the loop is free from x so it must occur in the Hintikka structure of some $f(x)$ which is good. Done.

Part 2: Again this is done by structural induction. We consider the maximal and minimal fixed points.

1. $\nu x.f(x)$. Let I be the set of indices at which x occurs in the Hintikka structure. ($0 \in I$). We will show by induction on i that these points receive x^i for all i. Clearly they receive x^0. Suppose they have received x^i for some i. Now treating x in the Hintikka structure as x^i, the given H-structure is the union of the set of Hintikka structures generated by all $f(x^i)$ at the indices of I. Hence they are good and inductively $x^{i+1} = f(x^i)$ is valid at these points. This shows that the states of I receive all x^i and hence x. So $\sigma \models \nu x.f(x)$.

2. $\mu x.f(x)$. Again this is the interesting case. Since x does not occur in any infinite chain of parents, it follows that there are only a finite number of instances of x which do not generate any further x. Chasing parents back from these instances, after a finite number of steps k, we reach x of σ_0. Let I_1 be the indices labelled by x which does not generate any further labels of x and inductively construct $I_2, \ldots, I_k$. Taking the Hintikka structure generated by $f(x)$ on the I_1 instances, we see that they are subsets of the full Hintikka structure. Also they contain no instances of x. Hence $f(x^0) = x^1$ (x^0 = false) holds on I_1 by induction. Working backwards, we establish in the same way that $x^2 = f(x^1)$ holds on I_2 etc. Finally we conclude that x^k holds on I_k. Hence $\sigma \models \mu x.f(x)$. Done.

3.8 Graph Building

We have seen that if a formula f has a model, then it has an eventually periodic model in which case it generates a compatible good Hintikka structure. For the converse, it is required that f generates only a good Hintikka structure. So the question of satisfiability of f is equivalent to whether f can generate such a structure. It will be shown here that the f Hintikka structures correspond to paths through a labelled graph

$\mathcal{G}_f$. The graph is grown by successive decomposition of the connectives and expansion of the fixed point variables. Its nodes are labelled by a set of subformulae of f. A formula in a node may be starred. This indicates that the formula has been dealt with in an earlier step of the decomposition. The growing starts from a node N_f containing f and proceeds by successive application of one of the following steps to an unstarred formula g in a node N until no further nodes or edges are added. Note that if a node to be created already exists, then merely an edge is added (if necessary).

1. $g = f_1 \vee f_2$. Then create two children N_1, N_2 of N and put
$$N_i = (N \setminus \{g\}) \cup \{g^*, f_i\}$$
2. $g = f_1 \wedge f_2$. Then create a child N' of N with $N' = (N \setminus \{g\}) \cup \{g^*, f_1, f_2\}$.
3. $g = x$. Let $x = f(\mathbf{x})$ be the defining equation of x. Then the child N' of N is
$$N' = (N \setminus \{x\}) \cup \{x^*, f(\mathbf{x})\}$$
4. None of above. Then every formula in N is either $\bigcirc g$ or an atomic proposition. Such a node is called a state. The child of N is $N' = \{g;\ \bigcirc g \in N\}$.

There is a close connection betweenHintikka structures and infinite consistent paths in $\mathcal{G}_f$ through N_f. Let π be such a path and let S_i denote the states along it. Consistent means that no state contains a proposition and its negation. The formulae of S_i, on removing the stars, actually build an f Hintikka structure in the following sense. Take any model σ with the property: $\sigma_i(p)$ = true for $p \in S_i$, $\sigma_i(p)$ = false for $\neg p \in S_i$. Then the S_i form an f Hintikka labelling of σ. The converse statement holds as well. If $L(\sigma_i)$ is an f Hintikka labelling on a model σ, then it can be proved, by structural induction on f, that there is a consistent path in $\mathcal{G}_f$ whose states contain exactly the labellings $L(\sigma_i)$.

Finally we come to the connection with the good paths. We define exactly as before a parenthood relation on the variables of successive nodes of $\mathcal{G}_f$. A variable $v \in N$ generates a variable $v' \in N'$ further along a path π when there is a chain of variables, starting with v and ending in v', in the segment of π from N to N' in which every variable is in the scope of v. A *good* path is one in which no minimal variable indefinitely generates itself. We define correspondingly the notions of an eventually periodic path and the concept of good paths among them.

The above discussion and theorem 4 imply

Theorem 5 *A vTL formula f has a model if and only if $\mathcal{G}_f$ has a consistent good path.*

The consistency requirement is dealt with easily. It is merely necessary to remove all states which contain inconsistent labelling (that is a proposition and its negation). We will also remove any other nodes from where all paths lead to only such states. For ease of notation we designate the resulting graph by $\mathcal{G}$. The decision problem has now been reduced to the checking of $\mathcal{G}$ for the existence of good paths. This is done with the marking algorithm of section 3.10.

3.9 Remark

In general
$$\mu x.f(x, \nu y.g(\mu z.f(z,y),y)) \text{ and } \mu x'.f(x', \nu y'.g(x',y'))$$
are not equal. Denoting the fixed points by their variables, we have:

$$\begin{aligned} x &= f(x,y) \\ y &= g(z,y) \\ z &= f(z,y) \end{aligned}$$

$$\begin{aligned} x' &= f(x',y') \\ y' &= g(x',y') \end{aligned}$$

From these equations it can be deduced that a path π of $\mathcal{G}_x$ corresponds to a path π' of $\mathcal{G}_{x'}$ and conversely. But the goodness property is not preserved. for example π can have the sequence of parents

$$x, y, z, y, z, y, z, \cdots$$

Then π is good. However π' has the generation:

$$x', y', x', y', x', y', \cdots$$

Hence it is not good. In general we can only deduce $x' \Rightarrow x$.

3.10 Marking

The graph construction procedure has furnished us with a graph $\mathcal{G}$. Here we describe a way of detecting if $\mathcal{G}$ has a good path. Since all the formulae apart from the variables are not of any interest to us, we may as well delete them. Note that each edge $e_i : M \rightarrow N$ is provided with a parenthood relation $R_i \subseteq M \times N$. Thus using infix notation xR_iy if x is the parent of y. The question can be formulated as follows. Does there exist a path $\pi : e_1 e_2 \cdots$ such that for any sequence of variables x_i, a number n and a minimal variable x, if every x_i is in the scope of x and $x_i R_{i+n} x_{i+1}$ then x occurs finitely often amongst the x_i. Stated in simpler terms, the condition is that no minimal variable should generate itself infinitely often.

The first step in our method is to dispose of the scope restriction. To do so we use subscripted variables. The subscript is a minimal variable. Thus x_y records that x has been generated by y. This leads to another graph $\mathcal{G}'$ which can be constructed in a stepwise fashion. There is a mapping of the nodes of $\mathcal{G}'$ to the nodes of $\mathcal{G}$. Let M' be a node of $\mathcal{G}'$ which maps to the node M of $\mathcal{G}$. Given an edge $e : M \rightarrow N$ with the relation R, the corresponding child N' and relation R' are defined as follows:

1. If $x_t \in M'$, xRy and y is in the scope of t then $y_t \in N'$ and $x_t R' y_t$.
2. Add x_x to N' whenever $x \in N$ is a minimal variable.

It should be clear to the reader that in $\mathcal{G}'$ the edge relations take account of the scope restrictions. Therefore a path in $\mathcal{G}'$ is good if no minimal variable, that is x_x for some minimal variable x, generates itself infinitely often. Moreover there is a bijection between the good paths in $\mathcal{G}$ and $\mathcal{G}'$. In this way we have removed the scope problem. To avoid the primes, we will use the original graph and leave out the scope considerations.

The main result in our search for the algorithm is the following

Lemma 8 *A path $\pi : N_0 \rightarrow N_1 \rightarrow \cdots$ is good if and only if there exists a labelling of the variables in the nodes with numbers in the range 1 to $2 \times \max_i |N_i|$ with the following properties:*

1. *Every variable has a label.*
2. *Minimal variables have odd labels.*
3. *The label of a child is at most that of any parent.*
4. *All the children of a minimal variable will have a smaller label in some node further along the path.*

Proof The if part is relatively easy. For if x is a minimal variable and generates itself infinitely often, then they are labelled by odd numbers which must keep decreasing according to the fourth item. Contradiction.

Let us assume now that π is good. We show how to give the labellings inductively. Suppose that the labels upto k have been used where k is even (possibly 0). First give a label of $k+1$ to all the variables all of whose children in some further node are already labelled. Next, if all the remaining variables are maximal then label them by $k+2$ and we are done. If not, there exists a non-labeled minimal variable x all of whose children are maximal. For if this is not the case then we can extend any chain of parents involving minimals to reach yet another minimal contradicting the fact that π is good. Moreover if x occurs in the node N, then it has non-labelled children in all the nodes in the tail of π away from N. Otherwise x would

be marked by our first step. Now give all these children the label $k+2$. Observe that at the end of these two steps the properties 2 to 4 are preserved. Furthermore, either all the variables are labelled or the number of unlabelled variables in the tail of π (that is $min_i(max_{j>i} \mid \{x; x \in N_j \text{ is unlabelled}\} \mid)$) has decreased. Therefore, after at most $2 \times max_i \mid N_i \mid$ steps the first alternative must hold. Done.

Our method is based on this lemma. The idea is to construct a succession of labelled graphs, each dependent solely on the previous one, such that the stages of a good path, as it is progressivley labelled, thread through them.

Let us have the graph $\mathcal{G}_k$ in which the numbers upto k have been used (k even). The next graph is constructed in a sequence of refinements. To begin with $\mathcal{G}_{k+1}$ is identical with $\mathcal{G}_k$. The refinement step is as follows. Let $x \in N$ be a variable such that all its children in some (immediate) child N' are labelled. Then the refined graph is identical except for the following points. It contains a new node N'' which is a copy of N where all the variables with the property of x are labelled with $k+1$. Moreover there is no edge from N to N', rather an edge form N'' to N'. Finally all the edges leading to N are duplicated to N'' as well. If a copy of N'' already exists, then only the necessary edges are added. These refinements must come to an end as the total number of variables which can be labelled decreases with each step.

Now the next step. First we give a label of $k+2$ to all the nonminimal variables. Then in a sequence of refinements which are identical to the previous step, we remove the label of a variable if it maps to a unlabelled variable in a child node. New nodes may be created as in the above paragraph.

It is important to note that each refinement preserves the property 4. Hence the graph resulting from a sequence of above steps enjoys the properties 2 to 4.

If $\mathcal{G}$ has a good path, then after at most $2 \times \max \mid N \mid$ steps we arrive at a graph $\mathcal{G}_m$ in which some nodes are totally labelled. Conversely, suppose that we have $\mathcal{G}_m$. It can be seen that the totally labelled nodes constitute a subgraph $\mathcal{H}$ which is a connected component and contains the root.[2] Since $\mathcal{G}_m$ satisfies the properties 2 to 4, the same holds for $\mathcal{H}$. Therefore $\mathcal{G}$ has a good path. We have shown,

Lemma 9 *$\mathcal{G}$ has a good path if and only if the application of at most* $\max \mid N \mid$ *pairs of the above steps produces a graph with a totally labelled node.*

[2]It may be necessary to apply the first step beforehand.

References

[BKP84] H. Barringer, R. Kuiper, and A. Pnueli.
Now You May Compose Temporal Logic Specifications.
In *Proceedings of the Sixteenth ACM Symposium on the Theory of Computing*, 1984.

[BKP85] H. Barringer, R. Kuiper, and A. Pnueli.
A compositional temporal approach to a csp-like language.
In E. J. Neuhold and G. Chroust, editors, *Formal Models of Programming*, pages 207–227. IFIP, North-Holland, 1985.

[BKP86] H. Barringer, R. Kuiper, and A. Pnueli.
A Really Abstract Concurrent Model and its Temporal Logic.
In *Proceedings of the Thirteenth ACM Symposium on the Principles of Programming Languages*, St. Petersberg Beach, Florida, January 1986.

[CES83] E. M. Clarke, E. A. Emerson, and A. P. Sistla.
Automatic Verification of Finite State Concurrent Systems using Temporal Logic Specifications: A Practical Approach.
In *Proceedings of the Tenth ACM Symposium on the Principles of Programming Languages*, 1983.

[Lam83] L. Lamport.
Specifying concurrent program modules.
ACM Transactions on Programming Languages and Systems, 5(2):190–222, July 1983.

[MP81] Z. Manna and A. Pnueli.
Verification of Concurrent Programs: The Temporal Framework.
In Robert S. Boyer and J. Strother Moore, editors, *The Correctness Problem in Computer Science*. Academic Press, London, 1981.

[Ros85] R. Rosner.
A Choppy Logic.
Master's thesis, Department of Computer Science, The Weizmann Institute of Science, 1985.

[SC85] A. P. Sistla and E. M. Clarke.
The Complexity of Propositional Linear Temporal Logics.
ACM Journal, 32(3):733–749, July 1985.

[SCFG82] A. Sistla, E. M. Clarke, N. Francez, and Y. Gurevich.
Can message buffers be characterised in linear temporal logic, 1982.

[Tar55] A. Tarski.
A Lattice-theoretical Fixedpoint Theorem and its Appplications.
Pacific Journal of Mathematics, 5:285–309, 1955.

[Var88] Moshe Y. Vardi.
A Temporal Fixpoint Calculus.
In *Proceedings of the Fifteenth ACM Symposium on the Principles of Programming Languages*, pages 250–259, San Diego, California, January 1988.
(Extended Abstract).

[Wol82] P. Wolper.
Synthesis of Communicating Processes from Temporal Logic Specifications.
PhD thesis, Stanford University, 1982.

[Wol83] P. Wolper.
Temporal Logic Can Be More Expressive.
Information and Control, 56, 1983.