Adding a Zoom to Linear Temporal Logic*

Ulrich Endriss

Department of Computer Science, King's College London
Strand, London WC2R 2LS, United Kingdom
Email: endriss@dcs.kcl.ac.uk

1 Introduction

Temporal logic. One of the great success stories of non-classical logics in mainstream computer science is that of linear temporal logic and its applications to systems specification and verification [2]. The simple formalism of a sequence of states and a valuation function mapping atomic propositions to sets of states, combined with propositional and modal operators to construct complex formulas, is surprisingly powerful when describing the behaviour of a given system over time.

However, this logic does *not* support the concept of modularisation in a natural manner. Each time we want to add additional detail about a subsystem associated with one of the states, we have to revise the specification of the entire system.

Zooming in. We propose to remedy this shortcoming *by adding a zoom to linear temporal logic:* If we (recursively) relate every state with another time line to describe the behaviour of the subsystem associated with that state we obtain a tree-like structure. In fact, we obtain an *ordered tree*: the relation connecting a state with all the states in the time line beneath it is a tree (if we also introduce a root) and the children of each node (the states of a single time line) are ordered. In this paper, we discuss a modal logic with frames that are ordered trees. It provides modal operators working both along the branches of a tree (level-of-detail dimension) and along the order declared over the children of a node (temporal dimension).

We define syntax and semantics of this new logic in Section 2 and briefly sketch a decidability proof in Section 3. A full paper is currently in preparation [1].

2 Ordered Tree Logic

We present syntax and semantics of the modal logic of (discretely) ordered trees.

Syntax. The set of *well-formed formulas* A of our logic is formed according to the following BNF production rule (p stands for propositional letters):

$$A \quad ::= \quad p \mid \neg A \mid A \wedge A \mid \ominus A \mid \ominus A \mid \circlearrowleft A \mid \diamondleftarrow A \mid \diamondrightarrow A \mid \circlearrowleft\!\!\!\!\uparrow A \mid \diamondsuit\!\!\!\!\downarrow A \mid \diamondsuit\!\!\!\!\downarrow^{+} A$$

Additional propositional connectives and box-operators may be introduced as defined operators in the usual way (like, for example, $\boxminus\varphi = \neg\diamondrightarrow\neg\varphi$).

Semantics. A *discretely ordered tree* $\mathcal{T}$ is a tree where the children of each node form a *discrete order* (that is, between any two sibling nodes there can only be finitely many other nodes). A *model* is a pair $\mathcal{M} = (\mathcal{T}, V)$, where $\mathcal{T}$ is such an ordered tree and V is a valuation function mapping propositional letters to sets of nodes of $\mathcal{T}$. Truth of a formula φ in $\mathcal{M}$ at a node $t \in \mathcal{T}$ is defined as follows:

1. $\mathcal{M}, t \models p$ iff $t \in V(p)$ for propositional letters p
2. $\mathcal{M}, t \models \neg\varphi$ iff not $\mathcal{M}, t \models \varphi$
3. $\mathcal{M}, t \models \varphi \wedge \psi$ iff $\mathcal{M}, t \models \varphi$ and $\mathcal{M}, t \models \psi$
4. $\mathcal{M}, t \models \circlearrowleft\varphi$ iff t has a parent t' and $\mathcal{M}, t' \models \varphi$
5. $\mathcal{M}, t \models \ominus\varphi$ iff t has a left neighbour t' and $\mathcal{M}, t' \models \varphi$
6. $\mathcal{M}, t \models \ominus\varphi$ iff t has a right neighb. t' and $\mathcal{M}, t' \models \varphi$
7. $\mathcal{M}, t \models \diamondleftarrow\!\!\!\!\uparrow\varphi$ iff t has an ancestor t' with $\mathcal{M}, t' \models \varphi$
8. $\mathcal{M}, t \models \diamondleftarrow\varphi$ iff t has a left sibling t' with $\mathcal{M}, t' \models \varphi$
9. $\mathcal{M}, t \models \diamondrightarrow\varphi$ iff t has a right sibling t' with $\mathcal{M}, t' \models \varphi$
10. $\mathcal{M}, t \models \diamondsuit\!\!\!\!\downarrow\varphi$ iff t has a child t' with $\mathcal{M}, t' \models \varphi$
11. $\mathcal{M}, t \models \diamondsuit\!\!\!\!\downarrow^{+}\varphi$ iff t has a descendant t' with $\mathcal{M}, t' \models \varphi$

A formula φ is called *satisfiable* iff it has a model (i.e. iff there are $\mathcal{M} = (\mathcal{T}, V)$ and $t \in \mathcal{T}$ with $\mathcal{M}, t \models \varphi$).

3 Bounded Finite Models

Ordered tree logics can be shown to be decidable by establishing a bounded finite model property (fmp), that is, by showing that any formula φ that is satisfiable in *some* model is also satisfiable in a model of limited size (where the maximal size is a function of the length of φ). Our techniques are similar to those used by Sistla and Clarke [3] to establish upper complexity bounds for propositional linear temporal logics.

Theorem 1 (Bounded FMP) *The modal logic of discretely ordered trees has the bounded finite model property.*

Theorem 1 is a corollary to Lemmas 2 and 5, proofs for which we are going to informally sketch below. The following observation will play a central role: To check whether a given formula φ is true at some node in a given model we have to check whether certain subformulas of φ are true at certain (other) nodes in the model; formulas that are not subformulas of φ are not relevant. So instead of models, we can work with *type models*, that is ordered trees where each node is associated with a certain *type* (a set of subformulas of the input formula φ).

Bounded branching. We first work towards controlling the horizontal dimension and show how we can reduce the branching factor of a given model.

Lemma 1 (Sibling pruning) *The stretch between two siblings of the same type can be removed (including one of the end nodes) without affecting satisfiability, provided we keep the node of evaluation as well as a witness for each formula of the form $\diamondsuit\!\!\!\!\downarrow\varphi$ or $\diamondsuit\!\!\!\!\downarrow^{+}\varphi$ in the parent.*

*This work has been supported by the EPSRC under grant reference numbers GR/R45369 and GR/N23028.

We can show this by structural induction. For atoms and propositional connectives, pruning elsewhere in the tree does not affect satisfiability. Formulas of the form $\Diamond\varphi$ or $\Diamond^+\varphi$ are not affected by definition. Neither are formulas of the form $\bigcirc\varphi$ or $\Diamond\!\!\!\!-\varphi$ as they are either removed from the tree or refer to nodes unaffected by the pruning operation. The horizontal modalities provide the interesting cases. We exemplify the general idea for formulas of the form $\Diamond\!\!\!\!\rightarrow\varphi$. Suppose $\Diamond\!\!\!\!\rightarrow\varphi$ is true at a node somewhere to the left of the stretch to be removed (we need to check that at least one witness survives). If $\Diamond\!\!\!\!\rightarrow\varphi$ is also true at the left one of the 'pruning nodes', then it must equally hold at the right one (because they have the same type), i.e. φ is true somewhere to the right of the stretch (and we are done). Otherwise, φ must already hold somewhere before the stretch (and we are done as well).

We can use the Sibling Pruning Lemma to prove the following result:

Lemma 2 (Bounded branching) *A formula φ of length n is satisfiable iff it is satisfiable in a (periodic) type model with a maximal branching factor of $(n+1)\cdot 2^n$.*

Observe that for any set of children there are at most n nodes the Sibling Pruning Lemma has to respect (the node of evaluation and up to $n-1$ witnesses), but it can be freely applied in between those n nodes. There are up to n subformulas of φ and, hence, up to 2^n consistent types, so we can reduce the $n-1$ finite stretches in between to at most 2^n-2 nodes each.

The left- and rightmost stretch, however, may be infinite. This is where periodicity comes into play. We consider the rightmost stretch: In case it is finite, we can reduce it to 2^n-1 nodes. Otherwise, there is a point from which onward all types that *do* come up come up infinitely often. Each one of them must have another type that occurs infinitely often as its right-hand neighbour. Hence, we can apply the Sibling Pruning Lemma in such a way that we obtain a periodic stretch of types (by always pruning between a node and the next occurrence of a node of the same type together with its 'designated neighbour'). The rightmost stretch as a whole (including the period) can then be pruned down to 2^n-1 nodes.

Altogether, we get a maximum of $n+(n-1)\cdot(2^n-2)+2\cdot(2^n-1) < (n+1)\cdot 2^n$ nodes.

Bounded depth. Now we turn to the vertical dimension and show how to reduce the length of branches in a given tree, again, without affecting satisfiability.

Lemma 3 (Branch pruning) *If a node t_1 and its descendant t_2 have the same type, we can replace the subtree beneath t_1 with the tree beneath t_2 without affecting satisfiability, provided we do not remove the node of evaluation.*

We omit the proof, which is similar to that of the Sibling Pruning Lemma.

To allow for the finite representation of recursive trees we introduce the notion of a *link*: a tree with a link from a node t_1 to another node t_2 represents the tree we get by (recursively) replaceing t_1 with t_2 (together with the subtrees beneath them).

Lemma 4 (Link introduction) *If two nodes t_1 and t_2 have the same type, we can introduce a link from t_1 to t_2 without affecting satisfiability, provided we keep the node of evaluation and a witness for each formula of the form $\Diamond^+\varphi$ in t_2.*

Again, we omit the proof and only point out that the ideas are essentially the same as before.

Lemma 5 (Bounded depth) *A formula φ of length n is satisfiable iff it is satisfiable in a type model (with links) with a maximal branch length of 2^{n+1}.*

Let us first observe that we can use the Branch Pruning Lemma to ensure that every type has a first occurrence at a node of depth $\leq 2^n$ (there are up to 2^n types and we can shorten any branch that has more than one node of the same type). We pick a minimal 'upper part' of the tree that features each type at least once.

Then we turn all branches into finite branches by applying the Link Introduction Lemma in such a way that links point from a node in the 'lower part' to one in the 'upper part'. We can always find a node far enough down the tree so that the lemma becomes applicable (i.e. that we keep all the required witnesses).

Finally, we use again the Branch Pruning Lemma, this time to reduce the 'lower part' to a maximal height of 2^n. Observe that, as we restrict pruning to the 'lower part' alone, no 'link-heads' will be cut off.

Decidability. Decidability is a direct consequence of the bounded fmp: a naïve decision procedure could simply enumerate all potential models up to the known maximal size and check each one of them. Hence, we obtain the following main result:

Theorem 2 (Decidability) *The satisfiability problem for the modal logic of discretely ordered trees is decidable.*

4 Conclusion

We conclude with a brief outlook on possible directions for future research in this area.

Complexity analysis. A bounded fmp provides a first step towards a complexity analysis of the logic under investigation. However, given that our bounds are exponential in *both* dimensions of a tree, at this stage, we can only establish a NEXPTIME upper bound and it is not clear if (and how) this could be improved upon.

Extensions. A number of interesting extensions to our logic, both to the language and to the underlying semantics, are possible. One such extension would be to investigate the addition of the temporal operators *since* and *until* (certainly to the horizontal dimension, but possibly also along branches). On the semantical level, an interesting extension would be to drop the condition of discreteness for the order declared over sibling nodes and to consider dense orders or general linear orders.

References

[1] U. Endriss. A Modal Logic of Ordered Trees. Manuscript, 2002.

[2] A. Pnueli. The Temporal Logic of Programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, pages 46–57. IEEE, 1977.

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