Lecture Notes for

Logic and Computability

Course Number: IND04033UF

Contact

Bettina Könighofer
Institute for Applied Information Processing and Communications (IAIK)
Graz University of Technology, Austria
bettina.koenighofer@iaik.tugraz.at

Table of Contents

10 Temporal Logic **3**
10.1 Kripke Structures 3
10.2 Computation Tree Logic $CTL^\star$ 5
10.2.1 Temporal Operators 5
10.2.2 Path Quantifiers 6
10.2.3 Syntax of $CTL^\star$ 6
10.2.4 Checking Properties of Kripke Structures 7

10 Temporal Logic

This chapter gives a brief introduction to temporal logic. Temporal logic is used to specify the dynamic behavior of systems. A formula in temporal logic might, for example, specify that a property holds in the next time step or in 3 time steps, it can express that eventually some goal state is reached, or it can specify that certain error states shall never be visited, or that some good event is happening infinitely often.

In this chapter, we will look at the intuitive semantics and syntax of the temporal logic $CTL^\star$. We will model systems via Kripke structures (Transition systems with additional labels in the states), and the temporal logic formulas will express properties of these Kripke structures.

We will briefly discuss the application of temporal logic in formal verification, where it is used to state temporal properties of systems. For instance, one property might express that whenever a request is made, access to a resource is eventually granted, but two request will never be granted simultaneously. Such statements can easily be expressed in $CTL^\star$.

10.1 Kripke Structures

In order to make observations about particular states of a system and to reason about these states, we model systems such that each state is labeled with a set of atomic propositions (Boolean variables that are true in that state). A transition system enriched with such a state-labeling is called a Kripke structure. The set of atomic prepositions used in the labels of the states are denoted as AP.

Definition - Kripke structure. *Given a set of atomic propositions AP,a Kripke structure $\mathcal{K}$ is a tuple $\mathcal{K} = (S, S_0, R, L)$ where*

1. *S, S_0, and R are defined as for transition systems,*
2. *$L : S \to 2^{AP}$ is a function that labels each state with the set of those atomic propositions that are true in that state.*

The labeling function L defines for each state $s \in S$ the set $L(s)$ of all atomic propositions that are valid in s.

Example. Draw the graph of a Kripke structure $\mathcal{K}_1$ with: $S = \{s_1, s_2, s_3\}$, $S_0 = \{s_1\}$, $R = \{(s_1, s_2), (s_2, s_1), (s_3, s_2)\}$, $AP = \{p, q\}$ and $L = \{s1, \{p\}, s2, \{p, q\}, s3, \emptyset\}$.

Solution.

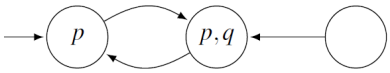

Figure 10.1: Graph for Kripke structure $\mathcal{K}_1$.

Example. Consider the example of an light switch. Initially, the light is off. Once a button is pressed, the light is turned on. To turn the light off, the button has to be released and pressed again. Model the light switch as Kripke structure.

Solution. A state is labeled with the label 1 if the light is on, and with label 0 if the light is off. Similarly, we use the label r for states in which the button is released, and p for those states in which the button is pressed.

In $\mathcal{K}_2$, initially, the button may either be pressed or released. This is modeled by means of two initial states. As example for a transition, consider the state labeled with $(0, r)$: to model the case the button is pressed, there is a transition to the state labeled $(1, p)$. Otherwise, the system remains in state $(0, r)$ by means of a self-transition.

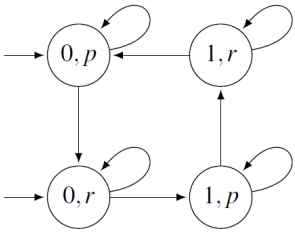

Figure 10.2: Graph for Kripke structure $\mathcal{K}_2$.

Definition - Paths and words of Kripke structures. *A **path** of the Kripke structure $\mathcal{K}$ is a sequence of states $\rho = s_1, s_2, s_3, \ldots$ such that for each $i > 0$, $R(s_i, s_{i+1})$ holds. The **word** on the path ρ is a sequence of sets of the atomic propositions $w = L(s1), L(s2), L(s3), \ldots$ over alphabet 2^{AP}.*

Example. Given the Kripke structure $\mathcal{K}_1$ of Figure 10.1. Give the word for the infinite path $\rho = s_1, s_2, s_1, s_2, s_2, \cdots = s_1, s_2, s_1, (s_2)^\omega$.

Solution. The path ρ produces the infinite word
$w = \{p\}, \{p, q\}, \{p\}, \{p, q\}, \{p, q\}, \{p, q\} \cdots = \{p\}, \{p, q\}, \{p\}, (\{p, q\})^\omega$.

10.2 Computation Tree Logic $CTL^\star$

The intuitive semantics of $CTL^\star$ is based on the notion of **computation trees**. Given a Kripke structure $\mathcal{K}$ with initial state s_0, its computation tree is formed by unwinding the structure into a tree with the root being s_0, as shown in Figure 10.3.

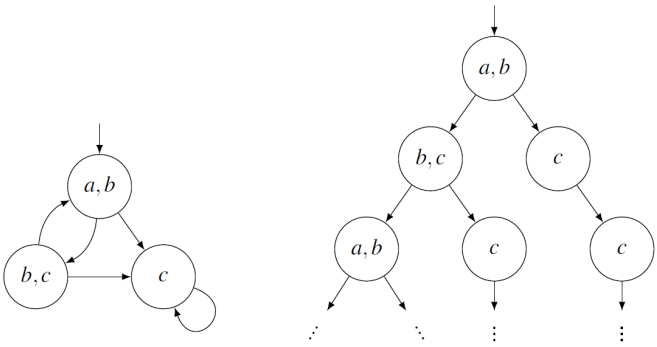

Figure 10.3: Left: Kripke structure, Right: Corresponding computation tree

The computation tree shows all of the possible paths and words (executions of the system) starting from s_0. In general, we require that every state of the Kripke structure has at least one outgoing edge. Therefore, all branches of the tree are infinite.

The temporal logic $CTL^\star$ has three kinds of operators: logical operators $(\neg, \vee, \wedge \dots)$, temporal operators, and path quantifiers, to specify properties of Kripke structures. While path quantifiers describe properties of a state, temporal operators describe properties of paths. The corresponding formulas are called **state formulas** and **path formulas**.

10.2.1 Temporal Operators

The temporal operators describe properties that *hold along a given infinite path* through the computation tree. We discuss the intuitive meaning of four basic operators below, assuming φ and ψ are formulas describing state properties.

- **neXt -** $X\varphi$: φ has to hold at the next state.

- **Future (Finally, Eventually) -** $F\varphi$: φ eventually has to hold somewhere on the subsequent path.
- **Globally (Always) -** $G\varphi$: φ has to hold on the entire subsequent path.
- **Until -** $\varphi U\psi$: ψ holds at the current or a future position, and φ has to hold until that position. At that position φ does not have to hold any more.

Their intuitive meaning of the operators is also presented in Figure 10.4. In the figure, p and q are formulas describing state properties.

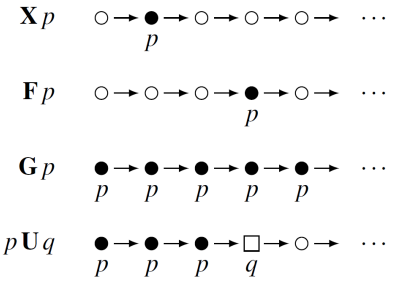

Figure 10.4: Illustration of temporal operators on an infinite path. Circles and squares represent states along the path and arrows represent transitions between states. The states are labeled with their atomic propositions.

10.2.2 Path Quantifiers

$CTL^\star$ defines properties of the Kripke structure that models a system by defining properties of the corresponding computation tree. To reason about the branching structure of the tree, $CTL^\star$ has the following two path quantifiers:

- **All -** $A\varphi$: φ has to hold on all paths starting from the current state.
- **Exists -** $E\varphi$: There exists at least one path starting from the current state where φ holds.

Figure 10.5 gives a few examples of computation trees satisfying simple $CTL^\star$ formulas, consisting of a path quantifier and a temporal operator.

10.2.3 Syntax of $CTL^\star$

The syntax of propositional $CTL^\star$ extends propositional logic by temporal operators and path quantifiers. We distinguish between two types of formulas in $CTL^\star$: state formulas (which are true in a specific state) and path formulas (which are true along a specific path).

Definition - Syntax of $CTL^\star$. *Let AP be the set of atomic propositions. The syntax of* ***state formulas*** *is given by the following rules:*

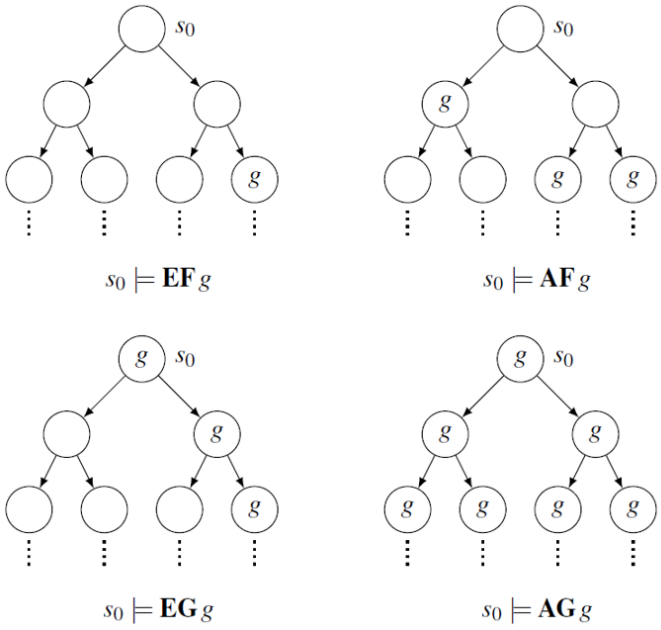

Figure 10.5: Example $CTL^\star$ properties

- *If $p \in AP$, then p is a state formula.*
- *If φ and ψ are state formulas, then $\neg\varphi$, $\varphi \vee \psi$, and $\varphi \wedge \psi$ are state formulas.*
- *If f is a **path formula**, then Ef and Af are state formulas.*

*The syntax of **path formulas** is given by the following rules:*

- *If f is a state formula, then f is also a path formula.*
- *If f and g are path formulas, then $\neg f$, $f \vee g$, $f \wedge g$, Xf, Ff, Gf, and fUg are path formulas.*

$CTL^\star$ is the set of state formulas defined by the rules above. Therefore, $CTL^\star$ formulas are Boolean propositions, temporal formulas with a leading path quantifier and Boolean combinations thereof.

10.2.4 Checking Properties of Kripke Structures

Given a Kripke structure $\mathcal{K}$.

- If φ is a *state formula*, $s \vDash \varphi$ means that φ holds at state s in $\mathcal{K}$.
- If φ is a *path formula*, $\rho \vDash \varphi$ means that φ holds along path ρ in $\mathcal{K}$.

Example. Consider the Kirpke structure and its computation tree from Figure 10.3. Let ρ_1 and ρ_2 be the leftmost and rightmost paths, respectively.

Answer the following questions:

- Does ρ_1 satisfy the formula $\varphi_1 \coloneqq Gb$?
- Does ρ_2 satisfy the formula $\varphi_1 \coloneqq Gb$?
- Is the formula $\varphi_2 \coloneqq EGb$ true in the initial state s_0?
- Is the formula $\varphi_3 \coloneqq AGb$ true in the initial state s_0?
- Is the formula $\varphi_4 \coloneqq EXX(a \wedge b)$ true in the initial state s_0?

Solution.

- Yes, $\rho_1 \vDash Gb$ holds since the property b holds in every state.
- No, $\rho_2 \nvDash Gb$ since b only holds in the first state and not on all states along ρ_2.
- Yes, $s_0 \vDash EGb$ holds since there is a path (ρ_1) that satisfies Gb.
- No, $s_0 \nvDash AGb$ since there is at least one path (ρ_2) that does not satisfy Gb.
- Yes, $s_0 \vDash EXX(a \wedge b)$ holds since the third state of ρ_1 is labeled with $\{a, b\}$.

Chapter 10 was based on the following book.

- Edmund M. Clarke, Orna Grumberg, Daniel Kroening, Doron A. Peled, Helmut Veith: Model checking, 2nd Edition. MIT Press 2018, ISBN 978-0-262-03883-6