CTL and ACTL patterns

Robert Meolic, Tatjana Kapus, Zmago Brezočnik
Faculty of Electrical Engineering and Computer Science
University of Maribor
Smetanova ul. 17, SI-2000 Maribor, Slovenia
{meolic,kapus,brezocnik}@uni-mb.si

Abstract

Model checking has become a widely used technique for verification of concurrent systems. However, its use is still much restricted to scientists with high mathematical education because temporal logic formulae are difficult to understand and even more difficult to create. Therefore, many projects have been started to find out how computers can help engineers in specifying system properties. This paper reports on using patterns of temporal logic formulae, which facilitate the usage of model checking.

1 Introduction

Model checking is a method for formal verification of concurrent systems, such as communication protocols, where properties are specified using a temporal logic. Its main advantages over equivalence and preorder checking are that it allows easier verification of incompletely specified systems, separate verification of each property, and generation of witnesses and counterexamples. Many temporal logics have been suggested to serve as a formal language for property specification. Among them, LTL and CTL are two most discussed. ACTL is another one, which is very suitable for reasoning about properties of concurrent systems described with process algebras. These three temporal logics are all a subset of μ-calculus. CTL and ACTL are especially popular because the algorithms for CTL and ACTL model checking are more efficient than those for LTL.

Unfortunately, model checking is a complex formal method which is not usable for an average user without a special knowledge about temporal logics. A solution which has been proposed by many authors is using well formed patterns [1] or building a tool for automatic translation of properties expressed in a natural language to temporal logic formulae [2]. Forming patterns is a more basic approach and can also serve as a core for translation tools.

This paper is organised as follows. The next section gives an overview of CTL and ACTL. Section 3 is the most important. It presents patterns for many common properties, where each property is described as an english sentence, CTL, and ACTL formula. In a short conclusion we evaluate our work.

2 CTL and ACTL

CTL and ACTL are both propositional branching-time temporal logics. However, the structure on which CTL and ACTL are defined, is different. CTL is interpreted over Kripke structures and ACTL is interpreted over labelled transition systems (LTSs). In a Kripke structure, each state is labelled with atomic propositions true in that state. On the other hand, in an LTS each transition is labelled with an action executed during that transition. For simplicity, we will use the term *path* in both cases, which in the case of Kripke structure means a sequence of states reached during the time and in the case of LTS means a sequence of actions executed during the time. The relationship between CTL and ACTL is described in more detail in [3]. An interesting paper about CTL which may be helpful in researching patterns is [4].

The syntax of CTL and ACTL formulae includes standard Boolean operators $\neg, \wedge, \vee, \Longrightarrow$, *path quantifiers* **E** ("there exists a path") and **A** ("for all paths"), and *temporal operators* **U** ("until"), $\underline{\mathbf{U}}$ or **UU** ("unless"), **X** ("for the next state/transition"), **F** ("for some state/transition in the future"), and **G** ("for all states/transitions in the future").

In CTL and ACTL formulae, each temporal operator is immediately preceded by a path quantifier. When path quantifier **E** is used, we require that the property expressed by the formula is valid for at least one path starting in the given state. Otherwise, when path quantifier **A** is used, we require that the property expressed by the formula is valid for all paths starting in the given state.

Let us first describe semantics of CTL formulae. A state where CTL formula φ is valid is called φ-state. The meaning of temporal operators can be explained as follows:

- Formula $\mathbf{X}\,\varphi$ is valid on path π iff the next state on the path is a φ-state.
- Formula $\mathbf{F}\,\varphi$ is valid on path π iff there exists a φ-state on the path.
- Formula $\mathbf{G}\,\varphi$ is valid on path π iff all states on the path are φ-states.
- Formula $[\varphi\ \mathbf{U}\ \varphi']$ is valid on path π iff the path begins with a finite sequence of φ-states followed by a φ'-state.

PREPRINT for EUROCON'2001, July 5-7, Bratislava, Slovak Republic

- Formula $[\varphi \underline{\mathbf{U}} \varphi']$ is valid on path π iff formula $[\varphi \mathbf{U} \varphi']$ is valid on this path or formula $\mathbf{G}\ \varphi$ is valid on this path.

Semantics of ACTL is a little more complex. An action for which ACTL formula χ is valid is called χ-action. A state where ACTL formula φ is valid is called φ-state. The transition from state p to state q where action formula χ is valid for the action executed during this transition and ACTL formula φ is valid in state q is called (χ, φ)-*transition*.

The meaning of temporal operators can be explained as follows:

- Formula $\mathbf{X}\{\chi\}\ \varphi$ is valid on path π iff the first transition on the path is a (χ, φ)-transition.
- Formula $\mathbf{F}\{\chi\}\ \varphi$ is valid on path π iff there exists a (χ, φ)-transition on the path.
- Formula $\mathbf{G}\ \varphi\{\chi\}$ is valid on path π iff ACTL formula φ is valid in the first state of this path and all transitions on the path are (χ, φ)-transitions.
- Formula $[\varphi\ \{\chi\}\ \mathbf{U}\ \{\chi'\}\ \varphi']$ is valid on path π iff ACTL formula φ is valid in the first state of this path and the path begins with a finite sequence of (χ, φ)-transitions followed by a (χ', φ')-transition.
- Formula $[\varphi\ \{\chi\}\ \underline{\mathbf{U}}\ \{\chi'\}\ \varphi']$ is valid on path π iff formula $[\varphi\ \{\chi\}\ \mathbf{U}\ \{\chi'\}\ \varphi']$ is valid on this path or formula $\mathbf{G}\ \varphi\{\chi\}$ is valid on this path.

In ACTL formulae the constant *true* can be omitted in many cases, for example:

$$\begin{aligned} \mathbf{E}[true\ \{\chi\}\ \mathbf{U}\ \{\chi'\}\ \varphi'] &= \mathbf{E}\ [\{\chi\}\ \mathbf{U}\ \{\chi'\}\ \varphi'] \\ \mathbf{A}[\varphi\ \{true\}\ \mathbf{U}\ \{true\}\ \varphi'] &= \mathbf{A}\ [\varphi\ \mathbf{U}\ \varphi'] \end{aligned}$$

There are also two widely accepted abbreviations of ACTL operators:

$$\begin{aligned} <\chi>\varphi &= \mathbf{EX}\{\chi\}\ \varphi \\ [\chi]\,\varphi &= \neg\mathbf{EX}\{\chi\}\ \neg\varphi \end{aligned}$$

ACTL formula $<\alpha>\varphi$ is valid in the given state iff there exists a transition with action α from that state to a state where ACTL formula φ is valid. ACTL formula $[\alpha]\varphi$ is valid in the given state iff all transitions with action α from that state lead to a state where ACTL formula φ is valid.

3 CTL and ACTL patterns

In this section, CTL and ACTL formulae for different properties are presented. To express properties with natural language, we introduce a *Property Oriented Notation*. Each given property determines a property in Kripke structure and a property in LTS. The core of properties are symbols $p, q, r, s, ..$, which represent atomic propositions. In a Kripke stucture, proposition p is valid in p-states, whereas in an LTS, proposition p is valid for p-actions.

3.1 State patterns

If we choose a state in the future, then we can specify what will happen before and/or after that state is reached. We call such patterns *state patterns*.

(S1) p is valid all the time in the future:

CTL: $\mathbf{AG}\ p$
ACTL: $\mathbf{AG}\ \{p\}$

(S2) Always in the future, if q is valid, then p is valid all the time before it:

CTL: $\mathbf{AG}\ (\neg p \Longrightarrow \mathbf{AX}\ \mathbf{AG}\ \neg q)$
ACTL: $\mathbf{AG}\ [\neg p]\ \mathbf{AG}\ \{\neg q\}$

(S3) If q is valid in the future for the first time, then p is valid all the time before it:

CTL: $\neg\mathbf{E}[\neg q\ \mathbf{U}\ (\neg p \wedge \neg q \wedge \mathbf{EF}\ q)]$
ACTL: $\neg\mathbf{E}[\{\neg q\}\ \mathbf{U}\ \{\neg p \wedge \neg q\}\ \mathbf{EF}\ \{q\}]$

(S4) Always in the future, if q is valid, then p is valid all the time after it:

CTL: $\mathbf{AG}\ (q \Longrightarrow \mathbf{AG}\ p)$
ACTL: $\mathbf{AG}\ [q]\ \mathbf{AG}\ \{p\}$

(S5) If q is valid in the future for the last time, then p is valid all the time after it:

CTL: $\mathbf{AG}\ (q \Longrightarrow (\mathbf{EX}\ \mathbf{EF}\ q \vee \mathbf{AG}\ p))$
ACTL: $\mathbf{AG}\ [q]\ (\mathbf{EF}\ \{q\} \vee \mathbf{AG}\ \{p\})$

(S6) p is valid for at least one time in the future:

CTL: $\mathbf{AF}\ p$
ACTL: $\mathbf{AF}\ \{p\}$

(S7) Always in the future, if q is valid, then p is valid for at least one time before it:

CTL: $\neg\mathbf{E}[\neg p\ \mathbf{U}\ q]$
ACTL: $\neg\mathbf{E}[\{\neg p\}\ \mathbf{U}\ \{q\}]$

(S8) If q is valid in the future for the last time, then p is valid for at least one time before it:

CTL: $\neg\mathbf{E}[\neg p\ \mathbf{U}\ (q \wedge \mathbf{EX}\ \mathbf{EG}\ \neg q)]$
ACTL: $\neg\mathbf{E}[\{\neg p\}\ \mathbf{U}\ \{q\}\ \mathbf{EG}\ \{\neg q\}]$

(S9) Always in the future, if q is valid, then p is valid for at least one time after it:

CTL: $\mathbf{AG}\ (q \Longrightarrow \mathbf{AF}\ p)$
ACTL: $\mathbf{AG}\ [q]\ \mathbf{AF}\ \{p\}$

(S10) If q is valid in the future for the first time, then p is valid for at least one time after it:

CTL: $\neg\mathbf{E}[\neg q\ \mathbf{U}\ (q \wedge \mathbf{EG}\ \neg p)]$
ACTL: $\neg\mathbf{E}[\{\neg q\}\ \mathbf{U}\ \{q\}\ \mathbf{EG}\ \{\neg p\}]$

3.2 Path patterns

Let p and q be two properties valid in the given state. With regard to p and q, there exist four different types of path beginning in this state:

Type 1: p and q are valid until a state is reached where p is still valid and q is not valid.

Type 2: p and q are valid until a state is reached where p is not valid and q is still valid.

Type 3: p and q are valid until a state is reached where p is not valid and q is not valid.

Type 4: p and q are valid on the whole path.

Let quadruple ($type_1$, $type_2$, $type_3$, $type_4$), $type_i \in \{0, 1\}$, denote states, where for all i such that $type_i = 0$, there is no path of $type_i$ with regard to properties p and q beginning in it. We call patterns, which define ($type_1$, $type_2$, $type_3$, $type_4$)-states *path patterns*.

(0,0,0,0)-state with regard to properties p and q

CTL: $false$
ACTL: $false$

(0,0,0,1)-state with regard to properties p and q

CTL: $\mathbf{AG}(p \wedge q)$
ACTL: $\mathbf{AG}\{p \wedge q\}$

(0,0,1,0)-state with regard to properties p and q

CTL: $\mathbf{A}[(p \wedge q)\ \mathbf{U}\ (\neg p \wedge \neg q)]$
ACTL: $\mathbf{A}[\{p \wedge q\}\ \mathbf{U}\ \{\neg p \wedge \neg q\}]$

(0,0,1,1)-state with regard to properties p and q

CTL: $\neg\mathbf{E}[(p \wedge q)\ \mathbf{U}\ ((p \wedge \neg q) \vee (\neg p \wedge q))]$
ACTL: $\neg\mathbf{E}[\{p \wedge q\}\ \mathbf{U}\ \{(p \wedge \neg q) \vee (\neg p \wedge q)\}]$

(0,1,0,0)-state with regard to properties p and q

CTL: $\mathbf{A}[q\ \mathbf{U}\ (q \wedge \neg p)]$
ACTL: $\mathbf{A}[\{q\}\ \mathbf{U}\ \{q \wedge \neg p\}]$

(0,1,0,1)-state with regard to properties p and q

CTL: $\neg\mathbf{E}[p\ \mathbf{U}\ \neg q]$
ACTL: $\neg\mathbf{E}[\{p\}\ \mathbf{U}\ \{\neg q\}]$

(0,1,1,0)-state with regard to properties p and q

CTL: $\mathbf{A}[q\ \mathbf{U}\ \neg p]$
ACTL: $\mathbf{A}[\{q\}\ \mathbf{U}\ \{\neg p\}]$

(0,1,1,1)-state with regard to properties p and q

CTL: $\neg\mathbf{E}[p\ \mathbf{U}\ (p \wedge \neg q)]$
ACTL: $\neg\mathbf{E}[\{p\}\ \mathbf{U}\ \{p \wedge \neg q\}]$

(1,0,0,0)-state with regard to properties p and q

CTL: $\mathbf{A}[p\ \mathbf{U}\ (p \wedge \neg q)]$
ACTL: $\mathbf{A}[\{p\}\ \mathbf{U}\ \{p \wedge \neg q\}]$

(1,0,0,1)-state with regard to properties p and q

CTL: $\neg\mathbf{E}[q\ \mathbf{U}\ \neg p]$
ACTL: $\neg\mathbf{E}[\{q\}\ \mathbf{U}\ \{\neg p\}]$

(1,0,1,0)-state with regard to properties p and q

CTL: $\mathbf{A}[p\ \mathbf{U}\ \neg q]$
ACTL: $\mathbf{A}[\{p\}\ \mathbf{U}\ \{\neg q\}]$

(1,0,1,1)-state with regard to properties p and q

CTL: $\neg\mathbf{E}[q\ \mathbf{U}\ (q \wedge \neg p)]$
ACTL: $\neg\mathbf{E}[\{q\}\ \mathbf{U}\ \{q \wedge \neg p\}]$

(1,1,0,0)-state with regard to properties p and q

CTL: $\mathbf{A}[(p \wedge q)\ \mathbf{U}\ ((p \wedge \neg q) \vee (\neg p \wedge q))]$
ACTL: $\mathbf{A}[\{p \wedge q\}\ \mathbf{U}\ \{(p \wedge \neg q) \vee (\neg p \wedge q)\}]$

(1,1,0,1)-state with regard to properties p and q

CTL: $\neg\mathbf{E}[(p \wedge q)\ \mathbf{U}\ (\neg p \wedge \neg q)]$
ACTL: $\neg\mathbf{E}[\{p \wedge q\}\ \mathbf{U}\ \{\neg p \wedge \neg q\}]$

(1,1,1,0)-state with regard to properties p and q

CTL: $\neg\mathbf{EG}\ (p \wedge q)$
ACTL: $\neg\mathbf{EG}\ \{p \wedge q\}$

(1,1,1,1)-state with regard to properties p and q

CTL: $true$
ACTL: $true$

3.3 Occurrence and order patterns

Another classification of patterns is presented on A Specification Pattern System page on the Internet maintained by M. Dwyer [1]. Patterns are classified into occurrence patterns and order patterns (Figure 1). *Occurrence patterns* are used to express requirements related to the existence or lack of existence of certain states/events during well-defined interval of time. *Order patterns* are used to express requirements related to pairs of states/events during well-defined interval of time. Presented CTL patterns are taken from the Internet (some of them are simplified) while ACTL patterns were constructed by us.

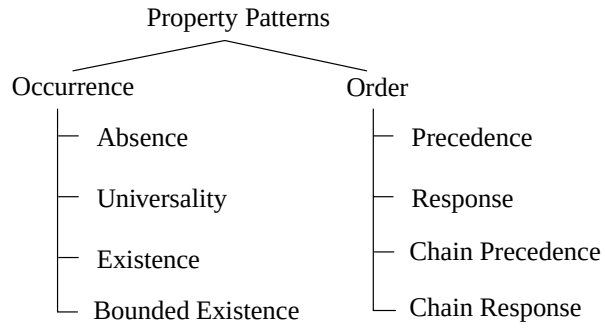

Figure 1: The patterns as classified by M. Dwyer

To understand formulae, here is a short comment. Something must happen *before/after* q iff it must happen before/after first q. None is required if q does not exist. Something must happen *between* q *and* r if it must happen between any q and first r after it. None is required if q does not exist or it is not followed by r.

Absence: To describe an interval of time that is free of certain states/events.

(Ab1) p is false:

CTL: $\mathbf{AG}\,\neg p$

ACTL: $\mathbf{AG}\,\{\neg p\}$

(Ab2) p is false before q:

CTL: $\neg\mathbf{E}[\neg q\,\mathbf{U}\,(p \wedge \neg q \wedge\ \mathbf{EF}\,q)]$

ACTL: $\neg\mathbf{E}[\{\neg q\}\,\mathbf{U}\,\{p \wedge \neg q\}\,\mathbf{EF}\,\{q\}]$

(Ab3) p is false after q:

CTL: $\mathbf{AG}\,(q \Longrightarrow \mathbf{AG}\,\neg p)$

ACTL: $\mathbf{AG}\,[q]\,\mathbf{AG}\,\{\neg p\}$

(Ab4) p is false between q and r:

CTL: $\mathbf{AG}\,((q \wedge \neg r) \Longrightarrow \neg\mathbf{E}[\neg r\,\mathbf{U}\,(p \wedge \neg r \wedge\ \mathbf{EF}\,r)])$

ACTL: $\mathbf{AG}\,[q \wedge \neg r]\,\neg\mathbf{E}[\{\neg r\}\,\mathbf{U}\,\{p \wedge \neg r\}\,\mathbf{EF}\,\{r\}]$

Universality: To describe an interval of time which contains only states/events with a desired property.

(Un1) p is true:

CTL: $\mathbf{AG}\,p$

ACTL: $\mathbf{AG}\,\{p\}$

(Un2) p is true before q:

CTL: $\neg\mathbf{E}[\neg q\,\mathbf{U}\,(\neg p \wedge \neg q \wedge\ \mathbf{EF}\,q)]$

ACTL: $\neg\mathbf{E}[\{\neg q\}\,\mathbf{U}\,\{\neg p \wedge \neg q\}\,\mathbf{EF}\,\{q\}]$

(Un3) p is true after q:

CTL: $\mathbf{AG}\,(q \Longrightarrow \mathbf{AG}\,p)$

ACTL: $\mathbf{AG}\,[q]\,\mathbf{AG}\,\{p\}$

(Un4) p is true between q and r:

CTL: $\mathbf{AG}\,((q \wedge \neg r) \Longrightarrow \neg\mathbf{E}[\neg r\,\mathbf{U}\,(\neg p \wedge \neg r \wedge\ \mathbf{EF}\,r)])$

ACTL: $\mathbf{AG}\,[q \wedge \neg r]\,\neg\mathbf{E}[\{\neg r\}\,\mathbf{U}\,\{\neg p \wedge \neg r\}\,\mathbf{EF}\,\{r\}]$

Existence: To describe an interval of time that contains an instance of certain states/events.

(Ex1) p becomes true:

CTL: $\mathbf{AF}\,p$

ACTL: $\mathbf{AF}\,\{p\}$

(Ex2) p becomes true before q:

CTL: $\neg\mathbf{E}[\neg p\,\mathbf{U}\,q]$

ACTL: $\neg\mathbf{E}[\{\neg p\}\,\mathbf{U}\,\{q\}]$

(Ex3) p becomes true after q:

CTL: $\neg\mathbf{E}[\neg q\,\mathbf{U}\,(q \wedge \mathbf{EG}\,\neg p)]$

ACTL: $\neg\mathbf{E}[\{\neg q\}\,\mathbf{U}\,\{q\}\,\mathbf{EG}\,\{\neg p\}]$

(Ex4) p becomes true between q and r:

CTL: $\mathbf{AG}\,((q \wedge \neg r) \Longrightarrow \neg\mathbf{E}[\neg p\,\mathbf{U}\,r])$

ACTL: $\mathbf{AG}\,[q \wedge \neg r]\,\neg\mathbf{E}[\{\neg p\}\,\mathbf{U}\,\{r\}]$

Precedence: To describe relationships between a pair of states/events where the occurrence of the first is a necessary precondition for an occurrence of the second.

(Pr1) s precedes p:

CTL: $\neg\mathbf{E}[\neg s\,\mathbf{U}\,(p \wedge \neg s)]$

ACTL: $\neg\mathbf{E}[\{\neg s\}\,\mathbf{U}\,\{p \wedge \neg s\}]$

(Pr2) s precedes p before q:

CTL: $\neg\mathbf{E}[(\neg s \wedge \neg q)\,\mathbf{U}\,(p \wedge \neg s \wedge \neg q \wedge\ \mathbf{EF}\,q)]$

ACTL: $\neg\mathbf{E}[\{\neg s \wedge \neg q\}\,\mathbf{U}\,\{p \wedge \neg s \wedge \neg q\}\,\mathbf{EF}\,\{q\}]$

(Pr3) s precedes p after q:

CTL: $\neg\mathbf{E}[(\neg q\,\mathbf{U}\,(q \wedge \mathbf{E}[\neg s\,\mathbf{U}\,(p \wedge \neg s)])]$

ACTL: $\neg\mathbf{E}[\{\neg q\}\,\mathbf{U}\,\{q\}\,\mathbf{E}[\{\neg s\}\,\mathbf{U}\,\{p \wedge \neg s\}]]$

(Pr4) s precedes p between q and r:

CTL: $\mathbf{AG}((q \wedge \neg r) \Longrightarrow \neg\mathbf{E}[(\neg s \wedge \neg r)\,\mathbf{U}\,(p \wedge \neg s \wedge \neg r \wedge \mathbf{EF}r)])$

ACTL: $\mathbf{AG}\,[q \wedge \neg r]\,\neg\mathbf{E}[\{\neg s \wedge \neg r\}\,\mathbf{U}\,\{p \wedge \neg s \wedge \neg r\}\,\mathbf{EF}\,\{r\}]$

Response: Cause-effect. An occurrence of the cause must be followed by an occurrence of the effect.

(Re1) s responds to p:

CTL: $\mathbf{AG}\,(p \Longrightarrow\ \mathbf{AF}\,s)$

ACTL: $\mathbf{AG}\,[p]\,\mathbf{AF}\,\{s\}$

(Re2) s responds to p before q:

CTL: $\neg\mathbf{E}[\neg q\,\mathbf{U}\,(p \wedge \neg q \wedge \mathbf{E}[\neg s\,\mathbf{U}\,q])]$

ACTL: $\neg\mathbf{E}[\{\neg q\}\,\mathbf{U}\,\{p \wedge \neg q\}\,\mathbf{E}[\{\neg s\}\,\mathbf{U}\,\{q\}]]$

(Re3) s responds to p after q:

CTL: $\neg\mathbf{E}[\neg q\,\mathbf{U}\,(q \wedge \neg\,\mathbf{AG}\,(p \Longrightarrow\ \mathbf{AF}\,s))]$

ACTL: $\neg\mathbf{E}[\{\neg q\}\,\mathbf{U}\,\{q\}\neg\,\mathbf{AG}\,[p]\,\mathbf{AF}\,\{s\}]$

(Re4) s responds to p between q and r:

CTL: $\mathbf{AG}\,((q \wedge \neg r) \Longrightarrow \neg\mathbf{E}[\neg r\,\mathbf{U}\,(p \wedge \neg r \wedge \mathbf{E}[\neg s\,\mathbf{U}\,r])])$

ACTL: $\mathbf{AG}\,[q \wedge \neg r]\,\neg\mathbf{E}[\{\neg r\}\,\mathbf{U}\,\{p \wedge \neg r\}\,\mathbf{E}[\{\neg s\}\,\mathbf{U}\,\{r\}]]$

4 Conclusion

This paper presents an interesting study in the field of formal verification with model checking. Patterns of CTL and ACTL formulae were discussed. Using the patterns, complex temporal logic formulae can be constructed more easily. We hope that this research will help us in the process of building a tool for automatic translation of properties expressed in natural language to temporal logic formulae.

References

[1] M. Dwyer et. al. Patterns in property specifications for finite-state verification. In *Proceedings of the 21st International Conference on Software Engineering*, May, 1999. `http://www.cis.ksu.edu/santos/spec-patterns/`

[2] A. Holt. Formal verification with natural language specifications: guidelines, experiments and lessons so far. *South African Computer Journal*, (24):253–257, November 1999.

[3] R. De Nicola et. al. An action based framework for verifying logical and behavioural properties of concurrent systems. *Computer Networks and ISDN Systems*, 25:761–778, 1993.

[4] Klaus Schneider. CTL and Equivalent Sublanguages of CTL*. In *IFIP 1996*, 1996.