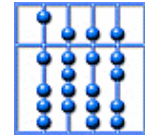


Vorlesung Specification of Distributed Systems

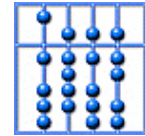
Dr. Bernhard Schätz
Leopold-Franzens Universität Innsbruck
Sommersemester 2005



Overview

1. Introduction
2. Basics: Behavior, Interaction, Concurrency
3. Coroutines
4. Communicating Processes
5. Data Flow Models
6. Coordination
7. Executions
8. State-Based Model

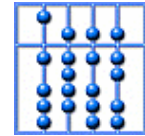
9. Property Descriptions



Overview

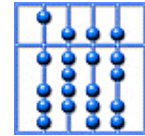
1. Introduction
2. Basics: Behavior, Interaction, Concurrency
3. Coroutines
4. Communicating Processes
5. Data Flow Models
6. Coordination
7. Executions
8. State-Based Model

9. Property Descriptions
 1. Temporal Properties
 2. Safety and Liveness Properties



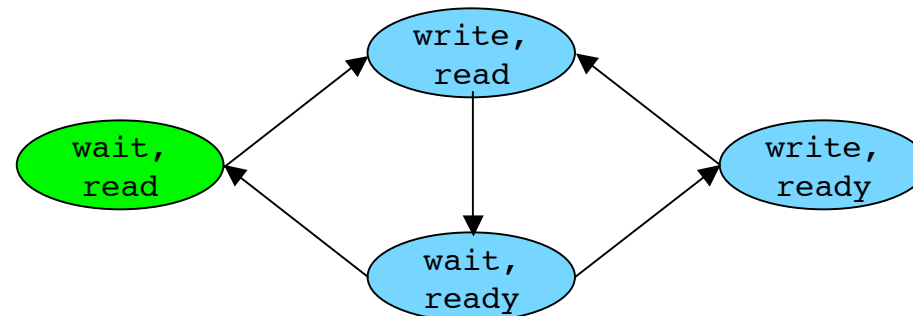
9 Questions

1. Exercise: Define a Kripke structure for the producer/consumer system describing which component is currently busy consuming or producing.
2. Exercise: Extend the Kripke structure to describe that either producer or consumer might get stuck during production or consumption.
3. What are corresponding logical forms of the following properties:
 - In every execution, the producer is always either busy or ready
 - In every execution, if producer and consumer are ready they will be busy in the next step
 - In every execution, the consumer will not eventually always be ready
 - There is an execution with the producer eventually always being busy
4. Which of the above properties hold
 - In the system of Exercise 1?
 - In the system of Exercise 2?
5. Express the following property in CTL: “It is possible that the producer eventually never becomes ready”. Is it equivalently expressible in LTL?
6. Express the following property in LTL: “In every execution, the consumer will eventually always be ready”. Is it equivalently expressible in CTL?
7. Express the following property in CTL*: “It is possible that the consumer is always finally ready”. Is it equivalently expressible in LTL? In CTL?



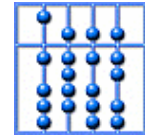
Exercise 1

Define a Kripke structure for the producer/consumer system, describing which component is currently busy producing/consuming or ready to exchange.



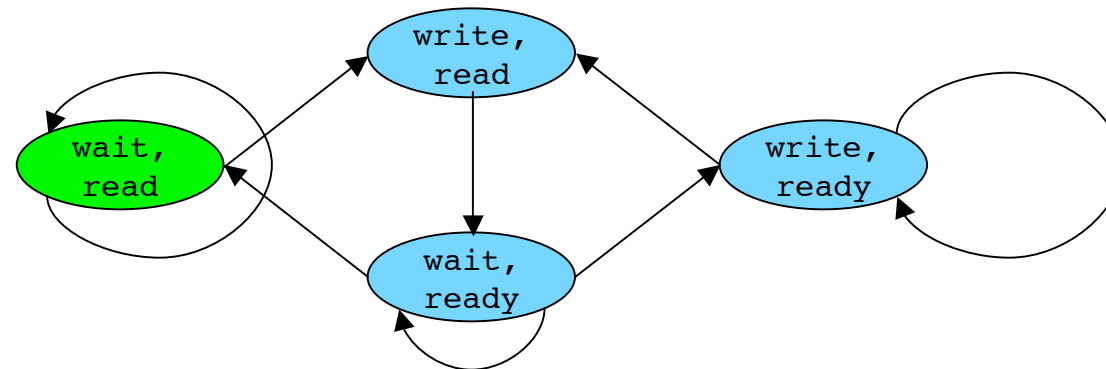
Kripke Structure: (S, S_0, T, P, O)

- $S = \{wR, WR, wr, Wr\}$
- $S_0 = \{wR\}$
- $T = \{(wR, WR), (WR, wr), (wr, wR), (wr, Wr)\}$
- $P = \{\text{"producer ready"}, \text{"producer busy"}, \text{"consumer ready"}, \text{"consumer busy"}\}$
- $O = \{wR \rightarrow \{\text{"producer busy"}, \text{consumer ready}\}, WR \rightarrow \{\text{"producer ready"}, \text{consumer ready}\}, wr \rightarrow \{\text{"producer busy"}, \text{consumer busy}\}, Wr \rightarrow \{\text{"producer ready"}, \text{consumer busy}\},$



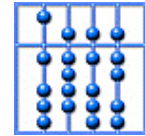
Exercise 2

How can the model from Exercise 1 be extended to describe that either producer or consumer might get stuck during production or consumption?



Kripke Structure: (S, S_0, T, P, O)

- $S = \{wR, WR, wr, Wr\}$
- $S_0 = \{wR\}$
- $T = \{(wR, WR), (WR, wr), (wr, wR), (wr, Wr), (wr, wr), (wR, wR), (Wr, Wr)\}$
- $P = \{\text{"producer ready"}, \text{"producer busy"}, \text{"consumer ready"}, \text{"consumer busy"}\}$
- $O = \{wR \rightarrow \{\text{"producer busy"}, \text{"consumer ready"}\}, WR \rightarrow \{\text{"producer ready"}, \text{"consumer ready"}\}, wr \rightarrow \{\text{"producer busy"}, \text{"consumer busy"}\}, Wr \rightarrow \{\text{"producer ready"}, \text{"consumer busy"}\},$



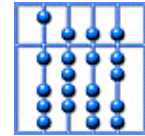
Question 3

What are corresponding logical forms of the following properties:

1. In every execution, the producer is always either busy or ready
2. In every execution, if producer and consumer are ready they will be busy in the next step
3. In every execution, the consumer will not eventually always be ready
4. There is an execution with the producer eventually always being busy

Formalizations:

1. $AG ("p \text{ busy}" \vee "p \text{ ready}")) \text{ or } G ("p \text{ busy}" \vee "p \text{ ready}"))$
2. $AG (("p \text{ ready}" \wedge "c \text{ ready}") \Rightarrow AX ("p \text{ busy}" \wedge "c \text{ busy}")) \text{ or } G (("p \text{ ready}" \wedge "c \text{ ready}") \Rightarrow X ("p \text{ busy}" \wedge "c \text{ busy}"))$
3. $A(\neg FG "c \text{ is ready}")) \text{ or } \neg FG "c \text{ is ready}"))$
4. $EFG "p \text{ is busy}"))$

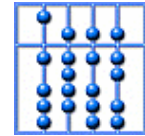


Question 4

Which of the above properties hold

- In the system of Exercise 1?
- In the system of Exercise 2?

1. $G ("p \text{ busy}" \vee "p \text{ ready}")$
 - Exercise 1: True
 - Exercise 2: True
2. $G (("p \text{ ready}" \wedge "c \text{ ready}") \Rightarrow X ("p \text{ busy}" \wedge "c \text{ busy}"))$
 - Exercise 1: True
 - Exercise 2: True
3. $\neg FG "c \text{ ready}"$
 - Exercise 1: True
 - Exercise 2: False
4. $EFG "p \text{ busy}"$
 - Exercise 1: False
 - Exercise 2: True



Question 5

Express the following property in CTL: “It is possible that the producer eventually never is ready”. Is it equivalently expressible in LTL?

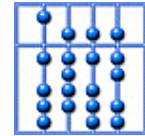
This property talks about the existence of a specific execution path:

CTL: $EF \neg(\text{“p is ready”})$

LTL: LTL uses implicit quantification over all execution paths of a system.

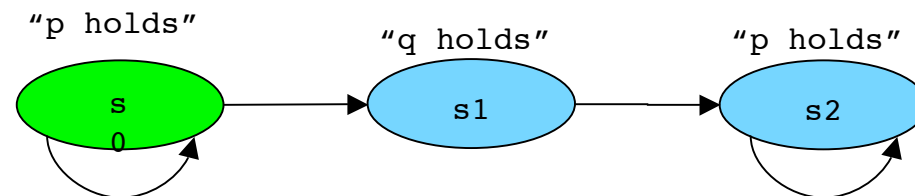
Therefore the existence of a path with a certain property is not expressible in LTL

Note: Consequently, CTL is not comparable with LTL



Question 6

Express the following property in LTL: “In every execution, the consumer will eventually always be ready”. Is it equivalently expressible in CTL?

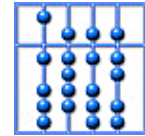


LTL: FG “c is ready”

CTL: FG “c is ready” is equivalent to the CTL* formula AFG “c is ready”. However, AFG “p” is not $AF AG$ “p” (see above example). Thus, FG “c is ready” is not equivalently expressible in CTL.

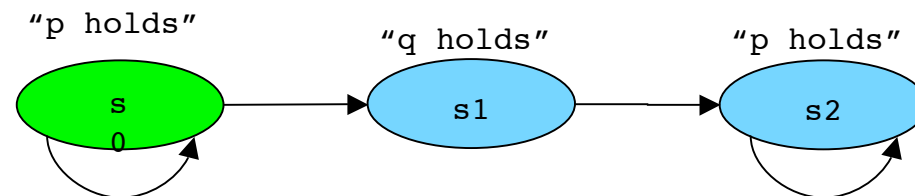
Note: Consequently, LTL is not comparable with CTL.

Note: This furthermore indicates that fairness properties are not expressible in CTL



Question 7

Express the following property in CTL*: “It is possible that the consumer is always eventually ready”. Is it equivalently expressible in LTL? In CTL?



CTL*: $E(GF \text{ "c is ready"})$

LTL: Not expressible, since only propositions over all execution paths are possible

CTL: Not expressible, since only $EG \text{ EF "c is ready"}$ corresponds to the syntactic restriction of CTL. However, as shown in the above example, $E(GF \text{ "q"}) \leftrightarrow EG (EF (\text{"q"}))$ (as well as $EG (AF (\text{"p"}))$)

Note: Consequently, CTL* is a proper superset of both LTL and CTL.