Christian-Albrechts-Universität zu Kiel
Institut für Informatik

Priv.-Doz. Dr. Frank Huch, Marius Rasch

Institut für Informatik

6. Übung zur Vorlesung „Concurrent and Distributed Programming“

Abgabe am Monday, 20. May 2019 - 18:00

Aufgabe 1 - Using LTL — 1 Punkt

In this exercise you shall use LTL for specification and testing.

1. The first implementation of `Chan` using `MVars` is incorrect for `isEmpty`. Formulate a LTL property broken by the incorrect behaviour of the implementation. Additionally you can test an implementation of `Chan` using this property.
2. The first implementation of the distibuted chat has an error when there are two logins at the same time at different nodes. Formulate a LTL property broken by this behaviour. Additionally you can test the distibuted chat using this property.

Aufgabe 2 - Missing LTL operations and global propositions — 1 Punkt

1. Extend the LTL-Implementation from the lectur by *Until*, *Release* and *weak Until*. Therefore, use the following equivalences:

$$\phi U \psi \simeq \neg(\neg\phi R \neg\psi)$$

$$\phi R \psi \simeq \psi \wedge (\phi \vee X \phi R \psi)$$

2. By testing it can't be disproved, that a preposition will be valid at some time. But it can be proven, that it will be valid after at most n steps. Implement a LTL operation $Fn\phi$ meaning that ϕ must be valid within the next n steps.

Aufgabe 3 - Verbose LTL tests — 1 Punkt

Using the existing LTL implementation Erlang programs can be tested. But you can't say how good a program was tested and failed tests don't allow to draw a conclusion about the error.

To fix this you shall

1. create a posibility to count, how often a LTL operation like *Finally* or *Globally* gets unwind.
2. print the path to an invalid state, if such an invalid state occures in a test.

Aufgabe 4 - a$^n b^n c^n$ 1 Punkt

Use the implemenation shown in the lecture to implement a Turing machine for the language $\{a^n b^n c^n | n \geq 0\}$.