Universidade do Minho

Universidade do Minho

Licenciatura em Ciências da Computação

Interacção e Concorrência 2023-24 - Practical assignment

Miguel Ângelo Alves de Freitas (A91635)

April 22, 2024

Index

1 Introduction

Simulation and analysis of traffic light control systems using the MCRL2 toolset, a powerful suite for modeling, validating, and verifying concurrent systems.

2 Problem 1

Consider the following junction where traffic is controlled by three traffic lights (processes A1, A2, and A3):

1. The traffic controller
$$T \triangleq C \mid A1 \mid A2 \mid A3$$
consists of three copies A1, A2, and A3 of a process traffic light L, in parallel with a control process C. Process L enforces the usual infinite loop behaviour of a traffic light showing green, followed by yellow, and then red, in cycle. Process C ensures that the green light is activated first in A1, then in A2, and finally in A3, in a loop, avoiding any clashes. Specify processes L and C.

2. Use MCRL2 to draw the transition system of process T. What more can you do with MCRL2 to analyse the behaviour of process T?

3. Apply once the expansion theorem to process T. Comment the result you obtained.

Solution to Problem 1

2.1 System Model

The system is defined in MCRL2 as follows:

```
% Action declarations
act
  startA1, startA2, startA3,  % Actions to start each traffic light
  doneA1, doneA2, doneA3,     % Actions to indicate completion of a cycle
  green, yellow, red;         % Traffic light color change actions

% Traffic Light Process Definition
proc
  L = green . yellow . red . L;

% Individual Traffic Lights
proc
  A1 = startA1 . L . doneA1 . A1;
  A2 = startA2 . L . doneA2 . A2;
  A3 = startA3 . L . doneA3 . A3;

% Controller Process
proc
  C = startA1 . doneA1 . startA2 . doneA2 . startA3 . doneA3 . C;

% System Initialization with Synchronization
init
  allow({startA1, startA2, startA3, doneA1, doneA2, doneA3, green, yellow, red},
comm({
   doneA1 | startA2 -> startA2,   % Synchronize on doneA1 and startA2
   doneA2 | startA3 -> startA3,   % Synchronize on doneA2 and startA3
   doneA3 | startA1 -> startA1},
         C || A1 || A2 || A3));
```

2.1.1 Traffic Light Process

Each traffic light follows a standard cycle (green, yellow, red) implemented in the process L as follows:

```
proc L = green . yellow . red . L;
```

This process signifies an infinite loop where a traffic light changes from green, to yellow, and finally to red before starting the cycle again.

2.1.2 Individual Traffic Lights

Three instances of the traffic light process L are created, each controlled by specific start and done actions:

```
proc
  A1 = startA1 . L . doneA1 . A1;
  A2 = startA2 . L . doneA2 . A2;
  A3 = startA3 . L . doneA3 . A3;
```

Each traffic light process starts with a unique start action, processes through L, signals the completion of its cycle with a done action, and then repeats.

2.1.3 Controller Process

The controller C orchestrates the sequence of the traffic lights ensuring that each light starts only after the previous one has completed its cycle:

```
proc C = startA1 . doneA1 . startA2 . doneA2 . startA3 . doneA3 . C;
```

This ensures the non-overlapping operation of the traffic lights, hence avoiding any potential clashes at the junction.

2.2 System Initialization and Synchronization

The system is initialized to allow the synchronization between the completion of one traffic light's cycle and the start of the next:

```
init
  allow({startA1, startA2, startA3, doneA1, doneA2, doneA3, green, yellow, red},
    comm({
      doneA1 | startA2 -> startA2,
      doneA2 | startA3 -> startA3,
      doneA3 | startA1 -> startA1},
      C || A1 || A2 || A3));
```

The 'allow' function restricts the set of actions visible externally, while 'comm' specifies how actions from different processes synchronize to perform new actions. This setup effectively chains the operation of the three traffic lights in a circular sequence, ensuring a continuous, collision-free operation at the junction.

2.3 Verification of Safety Properties

Safety properties ensure that conflicting events do not occur simultaneously. For example, no two traffic lights should be green at the same time. The property can be specified in modal mu-calculus as follows:

```
[true*] (
    <startA1>true => [(!startA2) && (!startA3)] true &&
    <startA2>true => [(!startA1) && (!startA3)] true &&
    <startA3>true => [(!startA1) && (!startA2)] true
)
```

Verification commands:

```
mcrl22lps -v traffic_system.mcrl2 traffic_system.lps
lps2pbes -v --formula=property.mcf traffic_system.lps traffic_system.pbes
pbes2bool -v traffic_system.pbes
```

The modal formula used for verifying the safety property is given below:

$$\Box\Big(\langle \text{startA1}\rangle \text{true} \implies [\neg\text{startA2} \wedge \neg\text{startA3}]\text{true} \wedge$$
$$\langle \text{startA2}\rangle \text{true} \implies [\neg\text{startA1} \wedge \neg\text{startA3}]\text{true} \wedge$$
$$\langle \text{startA3}\rangle \text{true} \implies [\neg\text{startA1} \wedge \neg\text{startA2}]\text{true}\Big),$$

where:

- $\Box$ ("always") operator asserts that the property must hold in all reachable states.
- $\langle$startAX$\rangle$true checks if there is a possibility that the action 'startAX' can occur.
- $\implies$ represents implication.
- $[\neg$startAY $\wedge$ $\neg$startAZ$]$true ensures that immediately after 'startAX', neither 'startAY' nor 'startAZ*canoccur*.

2.4 Liveness Properties

Liveness properties check that each component will eventually perform its expected function, such as each traffic light turning green:

```
[true*] <startA1>true && [true*] <startA2>true && [true*] <startA3>true
```

2.5 Simulation and Trace Analysis

Simulation can be done interactively using 'mcrl2i' or by generating traces with 'lps2lts':

```
mcrl2i traffic_system.mcrl2
lps2lts --trace-length=10 --generate-trace=example.trace traffic_system.lps
```

2.6 Performance Analysis

Performance analysis involves measuring how quickly and efficiently the system completes cycles under various configurations, which is crucial for optimizing throughput and response times.

2.7 Interpretation of Results

The MCRL2 model checker returned a result of *true* for the safety property, indicating that the system adheres to the specified safety constraints under all circumstances. This confirms that the traffic light controller is robust against the risk of simultaneous green signals, thereby preventing potential conflicts at the traffic junction.

Applying the Expansion Theorem

The expansion theorem in process algebra allows for the simplification of expressions involving parallel compositions by expanding them into a set of possible transitions. We apply this theorem to the process:

$$T = C \mid A1 \mid A2 \mid A3$$

2.8 Expansion Theorem

The expansion theorem for a process P composed of sub-processes $P1, \ldots, Pn$ is given by:

$$P = P_1 \mid \cdots \mid P_n = \sum_{a \in Act} \left(\bigoplus_{i:a \in I_i} a \cdot (P_1 \mid \cdots \mid P_{i-1} \mid P_i' \mid P_{i+1} \mid \cdots \mid P_n) \right)$$

where $P_i \xrightarrow{a} P_i'$ and $\bigoplus$ indicates a nondeterministic choice involving action a.

2.9 Expansion Steps

1. The system initiates with the controller C starting $A1$ with startA1, leading to:
$$T \to \text{startA1} \cdot (C' \mid A1' \mid A2 \mid A3)$$
where $A1'$ is the sequence of traffic light $A1$ proceeding through its cycle.

2. $A1$ processes the traffic light cycle L, which is a series of color changes green $\cdot$ yellow $\cdot$ red repeated indefinitely. C' is now in a waiting state for doneA1 to trigger startA2.

3. After completion of L by $A1$, resulting in doneA1, C synchronizes this with startA2 to activate $A2$:
$$\text{doneA1} \mid \text{startA2} \to \text{startA2} \cdot (C'' \mid A1 \mid A2' \mid A3)$$
where $A2'$ is the initiation of $A2$'s cycle.

2.10 Comments on the Results

The expansion theorem allows us to explicitly trace the sequence of events and state transitions in the traffic light control system, ensuring proper synchronization and validating safety requirements. The detailed expansion helps in analyzing potential deadlock scenarios and understanding the dependency and causality among actions.

3 Problem 2

Recall the observable transition relation $x \Rightarrow \subseteq P \times P$, where $x \in L = (\text{Act} - \{\tau\}) \cup \{\epsilon\}$. As discussed in the lectures, a $\epsilon \Rightarrow$-transition corresponds to zero or more transitions through an unobservable action τ.

Consider two new modal operators that express, respectively, the possibility and the need for a property to be valid after performing an arbitrary amount of unobservable behaviour.

$$E \models \langle\langle\rangle\rangle\phi \iff \exists F \in \{E' | E\epsilon \Rightarrow E'\}.F \models \phi$$

$$E \models \mathcal{J}\mathcal{K}\phi \iff \forall F \in \{E' | E\epsilon \Rightarrow E'\}.F \models \phi$$

By abbreviation we can now define the "observable versions" of $\langle K \rangle$ and $[K]$, for $K \subseteq L$. Thus,

$$\langle\langle K\rangle\rangle\phi \text{ abv } = \langle\langle\rangle\rangle\langle K\rangle\langle\langle\rangle\rangle\phi$$

$$\mathcal{J}\mathcal{K}\mathcal{K}\phi \text{ abv } = \mathcal{J}K[K]\mathcal{J}K\phi$$

1. Explain the meaning of formulas $\langle\langle\text{fdee}\rangle\rangle\text{true}$ and $\mathcal{J}$-Kfalse. Illustrate their use through the specification of four different, non-bisimilar processes such that $\langle\langle\text{fdee}\rangle\rangle\text{true}$ holds in two of them and $\mathcal{J}$-Kfalse in the other two.

2. In the logic you have studied in the lectures, formula $\langle - \rangle\text{true} \wedge [-a]\text{false}$ expresses inevitability, i.e., the occurrence of action a is inevitable. Which of the formulas

   (a) $\langle\langle - \rangle\rangle\text{true} \wedge \mathcal{J}$-aKfalse

   (b) $\mathcal{J}K\langle\langle - \rangle\rangle\text{true} \wedge \mathcal{J}$-aKfalse

   if any, would express a similar property in the observational setting? Justify your answer. If none seems suitable, provide an alternative specification.

3. In the lectures, you have studied a close relationship between bisimilarity and modal equivalence for the logic then introduced. Discuss in some detail if and how a similar result holds relating observational equivalence and modal equivalence for the extended logic.

Solution to Problem 2

3.1 Part 1

Formula: $\langle\langle \text{fdee} \rangle\rangle \text{true}$

This formula represents the possibility of achieving a state where *true* (which is always valid) can be reached through an arbitrary sequence of unobservable τ actions, followed by the action sequence "fdee", and followed by another arbitrary sequence of unobservable τ actions. Specifically:

- $\langle\langle\rangle\rangle$ allows any number of τ transitions.
- $\langle K \rangle$ for $K =$ "fdee" implies that the sequence of actions "fdee" is possible.
- Another $\langle\langle\rangle\rangle$ allows more τ transitions after "fdee".

This formula will be true in any process where there is a reachable state via "fdee" potentially flanked by τ transitions.

Formula: $\mathcal{J}$-Kfalse

This formula asserts that after an arbitrary amount of unobservable behavior (represented by $\epsilon \Rightarrow$), and the specific action sequence "K" (which might be misnoted), it is not possible to reach a state where the proposition is false (which is always false). However, if corrected as $\mathcal{J}$Kfalse, it would mean:

- $\mathcal{J}K$ implies for all sequences where "K" happens.
- $[K]$false means that after "K", the result must always be false, which is a contradiction unless "K" leads to no state.

Illustration with Processes:
Consider four different processes:

1. **Process P1**: Always performs "fdee" flanked by τ actions.
2. **Process P2**: Can perform "fdee" optionally after some τ actions.
3. **Process P3**: Never performs "fdee", only τ.
4. **Process P4**: Terminates immediately, no actions.

3.2 Part 2: Expressing Inevitability

Analysis of Given Formulas

- (a) $\langle\langle - \rangle\rangle \text{true} \wedge J_{-\text{a}} K \text{false}$ states that it is possible to reach a state where "true" holds after any number of unobservable actions, and inevitably, any action other than "a" leads to a state where "false" holds, thereby making action "a" inevitable under observability.
- (b) $JK\langle\langle - \rangle\rangle \text{true} \wedge J_{-\text{a}} K \text{false}$ similarly articulates that for all states reachable via unobservable transitions, reaching "true" is inevitable and actions not "a" invariably result in "false."

Analysis: Neither formula perfectly matches the inevitability expressed by $\langle - \rangle \text{true} \wedge [-\text{a}]\text{false}$ due to the complexities introduced by unobservable transitions. An alternative might involve explicitly modeling these transitions or adjusting the logic to handle unobservables directly.

Alternative Specification:

$$\langle\langle a \rangle\rangle \text{true}$$

This states that after arbitrary τ transitions, action "a" must occur, aligning more directly with the inevitability of "a".

3.3 Part 3: Relationship Between Observational Equivalence and Modal Equivalence

In classical bisimilarity, modal equivalence is guaranteed by the Hennessy-Milner theorem, under the assumption of image-finiteness. However, observational equivalence introduces a more lenient form of equivalence that allows sequences of unobservable actions (τ-transitions) to be ignored when considering the behavior of processes.

Traditional relationship between bisimilarity and modal equivalence states that two processes are bisimilar if and only if they satisfy the same modal formulas. In the context of observational equivalence:

- **Observational Equivalence:** Two processes are observationally equivalent if they behave the same under observation, ignoring τ transitions.
- **Modal Equivalence in Extended Logic:** Two processes that are observationally equivalent would satisfy the same set of modal formulas involving $\langle\langle K \rangle\rangle$ and $\mathcal{J}\mathcal{K}K$, given that these formulas account for τ transitions.

Thus, a similar result would hold: if two processes are observationally equivalent, they should be modally equivalent under the extended logic with $\epsilon \Rightarrow$-based modalities.

The extension of the logic with operators that deal specifically with $\epsilon \Rightarrow$-transitions suggests an intention to capture the nuances introduced by these unobservable behaviors.This complicates the relationship because:

1. **Observational Equivalence and Bisimilarity**: Two processes that are observationally equivalent may differ in their internal (unobservable) structure. Therefore, when extended with modal operators that account for $\epsilon \Rightarrow$-transitions, the modal logic must be able to express properties that are preserved across these internal structures.
2. **Extended Modal Operators**: The introduction of $\langle\langle\rangle\rangle$ and $\mathcal{J}\mathcal{K}$ implies that the modal logic can specify properties of processes after any number of τ-transitions. Thus, for modal equivalence to coincide with observational equivalence, these modal operators must be sensitive to the τ-transition structure in a way that mirrors the leniency of observational equivalence.

3. **Alternative Characterizations of Equivalence**: One may need to consider whether alternative characterizations of equivalence are more appropriate in the extended logic. For instance, it may be beneficial to look at weak bisimulation or some form of branching bisimulation as a basis for establishing modal equivalence in the presence of τ-transitions.

4. **Proof Adjustments**: Proofs of equivalence in this extended logic will likely require different techniques, potentially involving new induction principles or coinductive arguments that are suited to the richer structure of processes with τ-transitions.

5. **The Extended Hennessy-Milner Theorem**: It might be necessary to establish an extended version of the Hennessy-Milner theorem that specifically addresses the extended logic. The theorem would need to define the conditions under which modal equivalence with respect to the extended modal operators implies observational equivalence, and vice versa.

To summarize, while bisimilarity and modal equivalence are intimately connected in classical process calculi, the introduction of unobservable actions and their corresponding modal operators in the extended logic necessitates a careful re-examination of their relationship. This might involve creating new proof techniques, extending existing theorems, or possibly finding new forms of equivalence that are more apt for the extended setting.

References

[1] , *Validating communication of a dynamic traffic management system*, 2023 27th International Conference on Engineering of Complex Computer Systems (ICECCS), Available: IEEE Xplore, `https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10321863`.