From Scenarios To Optimally Allocated Timed Automata

Sandeep Vuppula

May 25, 2017

University of Minnesota Duluth

Table of contents

Objectives Of The Research

Our main focus of the research is,

1. To synthesize a timed automaton from a set of scenarios.
2. To optimally allocate clocks in the constructed timed automaton.

Introduction

Introduction

- Model-based design is a very effective method for designing real-time systems.
- Building formal models for systems is challenging because of the lack of good formal requirements specifications.
- In many real-time systems where safety is critical.
- Modeling a system formally can help us to understand the desired and undesired behaviours of the system.

Introduction

- To construct a formal model, the following questions are to be answered first:
  1. How the requirements should be expressed formally, and
  2. How the formal model of a real-time system can be constructed from requirements.

Introduction

- To construct a formal model, the following questions are to be answered first:
  1. How the requirements should be expressed formally, and
  2. How the formal model of a real-time system can be constructed from requirements.
- The formal model that we build is timed automata [1].

Introduction

- We use scenarios to build a formal model. A scenario is a partial description of the behaviour of a system.
- We introduce Timed Event Sequences (TES) to formally represent the scenarios formaly.
- We use mode graphs to specify the legal events that can in the system.

Introduction

- We use scenarios to build a formal model. A scenario is a partial description of the behaviour of a system.
- We introduce Timed Event Sequences (TES) to formally represent the scenarios formaly.
- We use mode graphs to specify the legal events that can in the system.

We synthesize a *minimal*, *acyclic* and *deterministic* timed automaton using TES and mode graph.

Introduction

Our timed automaton belongs to a class of timed automata that satisfies the following properties:

- A clock t_j can be reset only on the transitions emanating from a state labelled j and,
- A clock in a clock constraint on a transition r from a state q can only refer to a clock that has been reset on a transition leaving a state that dominates q. We call this *dominance assumption*.

Introduction

- Given a timed automaton $\mathcal{A}$, the problem of deciding whether there exists another timed automaton $\mathcal{B}$ that accepts the same language as that of $\mathcal{A}$ but with fewer number of clocks is undecidable.
- Moreover, the number of clocks in a given timed automaton has a direct impact on verification of the system.

Introduction

- Given a timed automaton $\mathcal{A}$, the problem of deciding whether there exists another timed automaton $\mathcal{B}$ that accepts the same language as that of $\mathcal{A}$ but with fewer number of clocks is undecidable.
- Moreover, the number of clocks in a given timed automaton has a direct impact on verification of the system.

Given a timed automaton that belongs to our class, we use *liveness analysis* of clocks to optimally allocate clocks.

Background

Real-time systems

- A real-time system takes input from its surrounding environment and produces results within a stipulated amount of time.
- In the real world, the behaviour of almost every system changes according to time.
- We can model such real time systems with the help of timed automata.

Modeling Time

There are three approaches for modeling time [1].

- Discrete time model: Time is considered as discrete and monotonically increasing sequence of integers. Limits the preciseness: in real-time systems, the events do not occur at integer times.

Modeling Time

There are three approaches for modeling time [1].

- Discrete time model: Time is considered as discrete and monotonically increasing sequence of integers. Limits the preciseness: in real-time systems, the events do not occur at integer times.
- Fictitious-clock model: It is similar to that of discrete time model except that it assumes sequence of times to be non decreasing integers. Limits accuracy, as the exact time values at which the events occur are not considered.

Modeling Time

There are three approaches for modeling time [1].

- Discrete time model: Time is considered as discrete and monotonically increasing sequence of integers. Limits the preciseness: in real-time systems, the events do not occur at integer times.
- Fictitious-clock model: It is similar to that of discrete time model except that it assumes sequence of times to be non decreasing integers. Limits accuracy, as the exact time values at which the events occur are not considered.
- Dense time model: In this model, the domain of time is considered as a dense set and the time of occurrences of events as real numbers, which increase monotonically without any limit. Difficulty in transforming dense time traces into formal languages.

Finite State Automata

A finite state automaton (FSA) or a finite state machine (FSM) is an abstract machine which has a finite number of states. On an input, the machine changes from one state to another state: this is called a transition.

Timed Automata

- A timed automaton [1] is a finite state automaton extended with a finite set of real-valued clocks.
- Upon an input, the selection of next state is based not only on the input symbol but also on the time of the current symbol with respect to the formerly read symbols.

Example: Consider a simple timed automaton in Figure 2. This automaton accepts an input sequence 'a' followed by 'b' such that, there is 2 units of time difference between any two consecutive a's and b's.

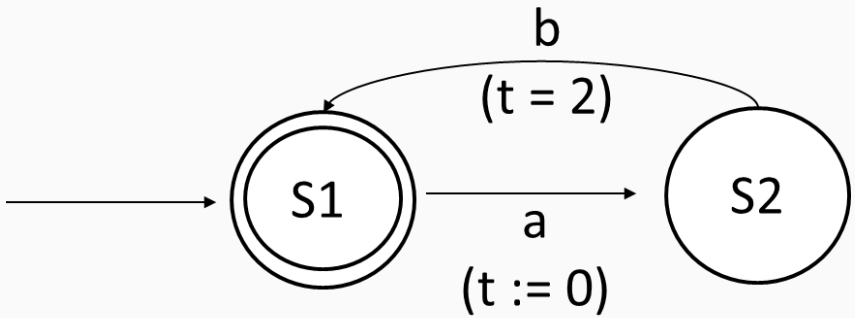


Figure 2: Simple Timed Automaton

Synthesis Of Timed Automata From Scenarios

Synthesis Of Timed Automata From Scenarios

- Constructing a time annotated graph from scenarios, and
- Constructing a timed automaton from time annotated graph.

Constructing A Timed Automaton From Time Annotated Graph

1. Determining the required number of clocks,
2. Adding clock resets,
3. Replacing the time annotations with the clock constraints

Constructing A Timed Automaton From Time Annotated Graph

content...

Constructing A Timed Automaton From Time Annotated Graph

$m^{initial}$: card-not-inserted

(insert-card, $\{\}$)
(enter-pin, $\{W - t_0 \geq 5,\ W - t_0 \leq 60\}$)
(incorrect-pin, $\{\}$)
(re-enter-pin, $\{W - t_0 \geq 5,\ W - t_0 \leq 60\}$)
(correct-pin, $\{\}$)
(request-data-from-bank, $\{\}$)
(display-menu, $\{W - t_4 \leq 5\}$)

m^{final}: menu-displayed

TES of Scenario 1

$m^{initial}$: card-not-inserted

(insert-card, $\{\}$)
(enter-pin, $\{W - t_0 \geq 5,\ W - t_0 \leq 60\}$)
(correct-pin, $\{\}$)
(request-data-from-bank, $\{\}$)
(display-menu, $\{W - t_4 \leq 5\}$)

m^{final}: menu-displayed

TES of Scenario 2

Figure 3: Timed Event Sequences of the ATM

Constructing A Timed Automaton From Time Annotated Graph

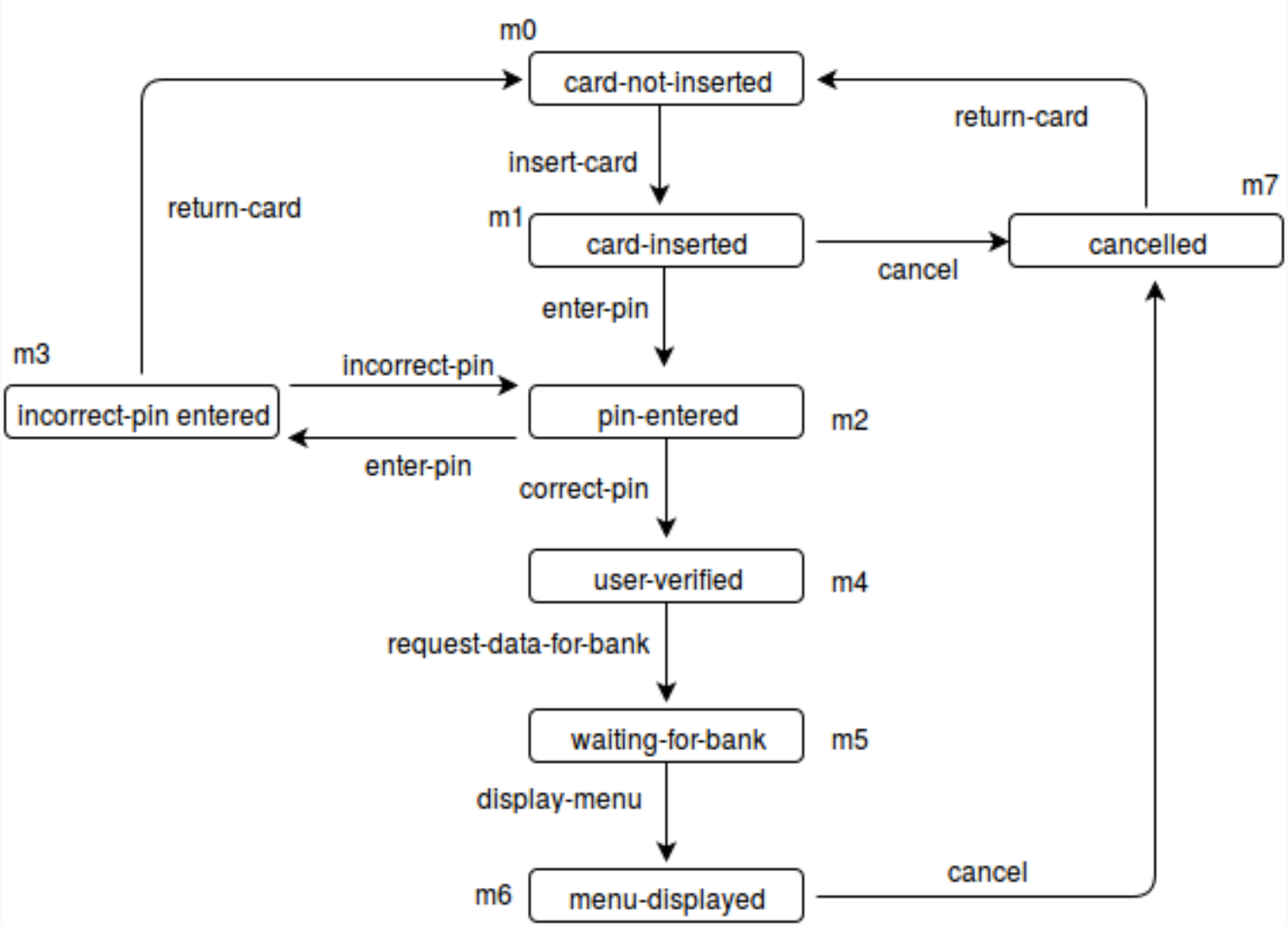


Figure 4: Mode Graph for ATM

Constructing A Timed Automaton From Time Annotated Graph

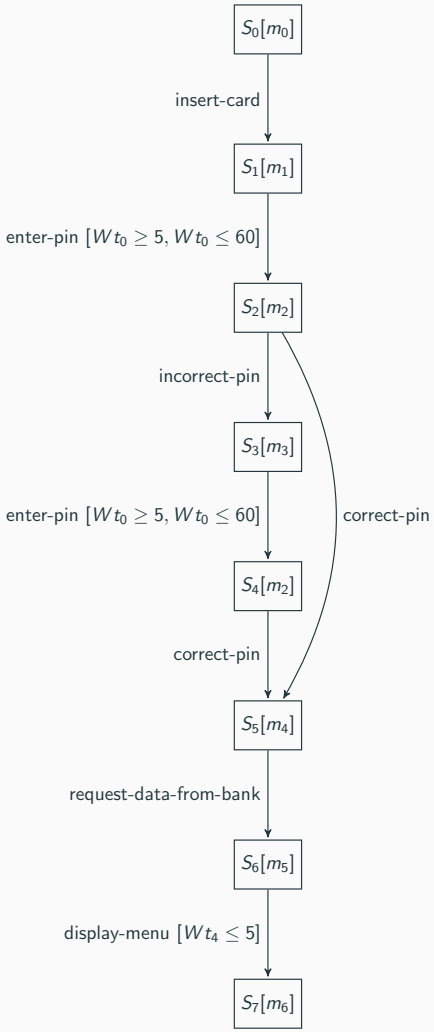


Figure 5: Time annotated graph synthesized from two TES in Figure **??**

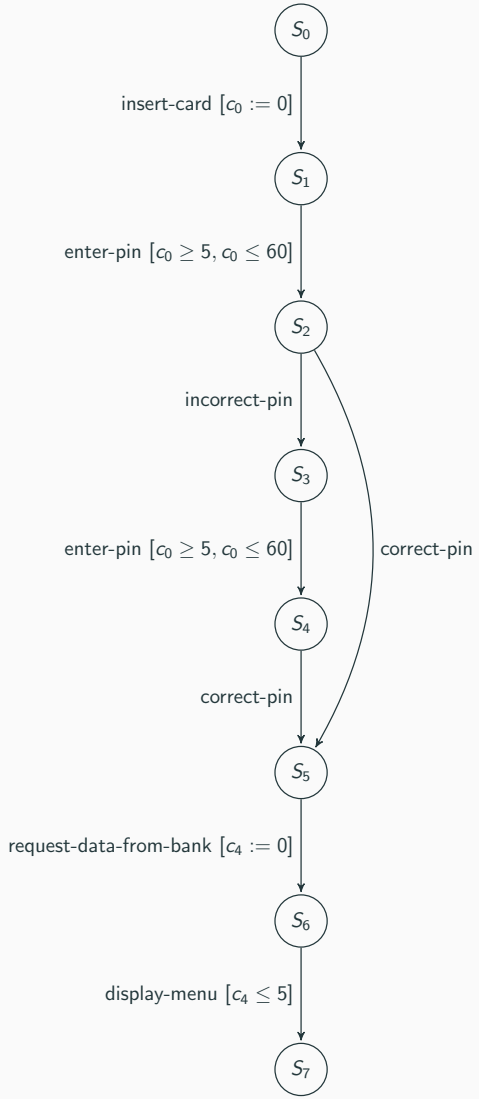


Figure 6: Timed automaton constructed from time annotated graph

Optimal Clock Allocation of Timed Automata

Optimal Clock Allocation of Timed Automata

- Liveness analysis
- Clock allocation

Liveness Range Analysis

- **clock_ref**: $clock\_ref(r)$ is the set of clocks which are referred to in the clock constraints on r.
- **born**: $born(r)$ identifies a clock that is reset on r whose value can be used on some transition reachable from r.
- **active**: $active(r)$ identifies clocks that are "alive" on r (i.e., their values may be subsequently used). Notice that $born(r) \subseteq active(r)$.
- **needed**: Maps transition r to $active(r) \cup clock\_ref(r)$.

Liveness Range Analysis Example

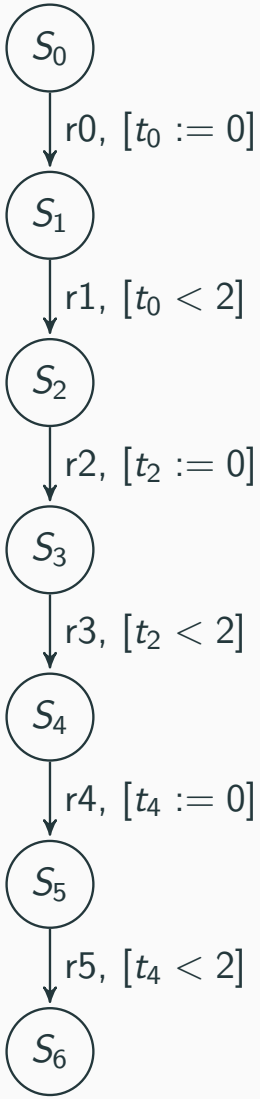


Figure 7: A simple timed automaton

Table 1: *born* and *active* values

Transition	Born	Active
r_0	$\{0\}$	$\{0\}$
r_1	ϕ	ϕ
r_2	$\{2\}$	$\{2\}$
r_3	ϕ	ϕ
r_4	$\{4\}$	$\{4\}$
r_5	ϕ	ϕ

ModeGraph

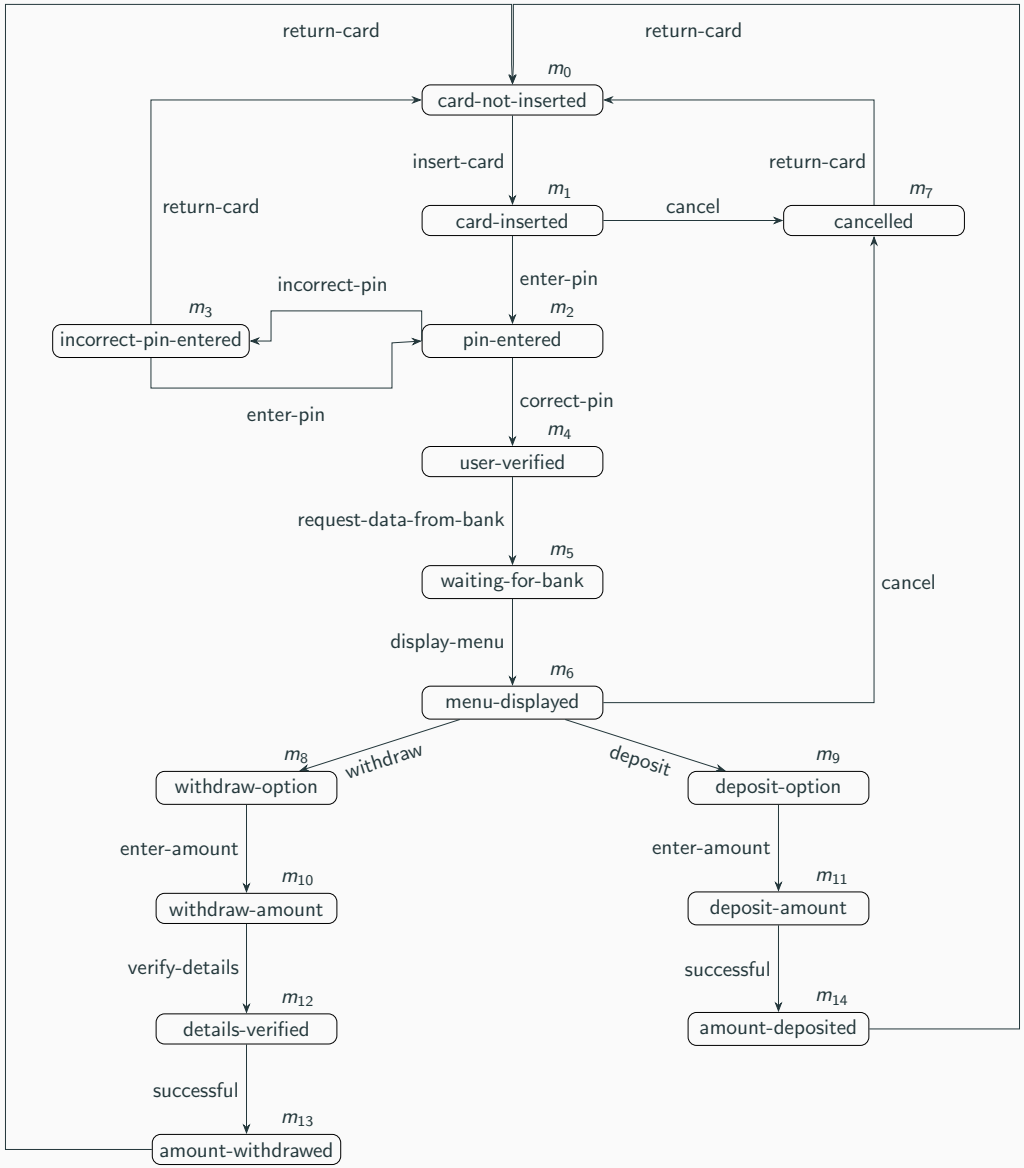


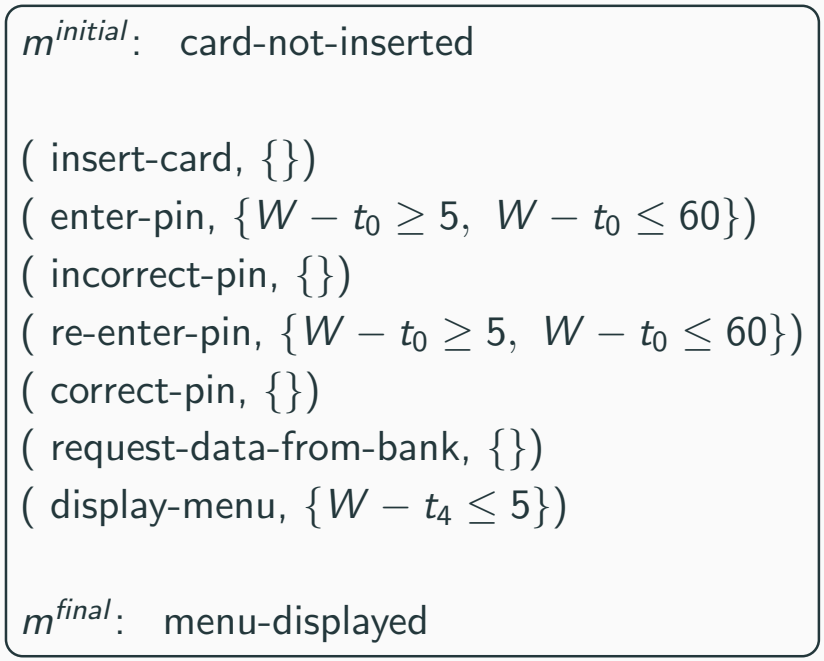
Figure 8: Mode graph of the ATM

Case Studies

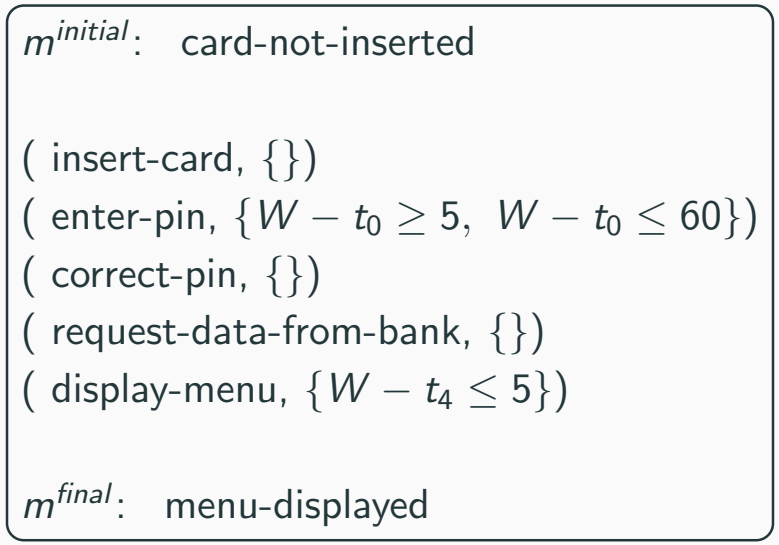
Automated Teller Machine (ATM)

Explain original scenario

Automated Teller Machine (ATM)



TES of Scenario 1



TES of Scenario 2

Figure 9: Timed Event Sequences of the ATM

Automated Teller Machine (ATM)

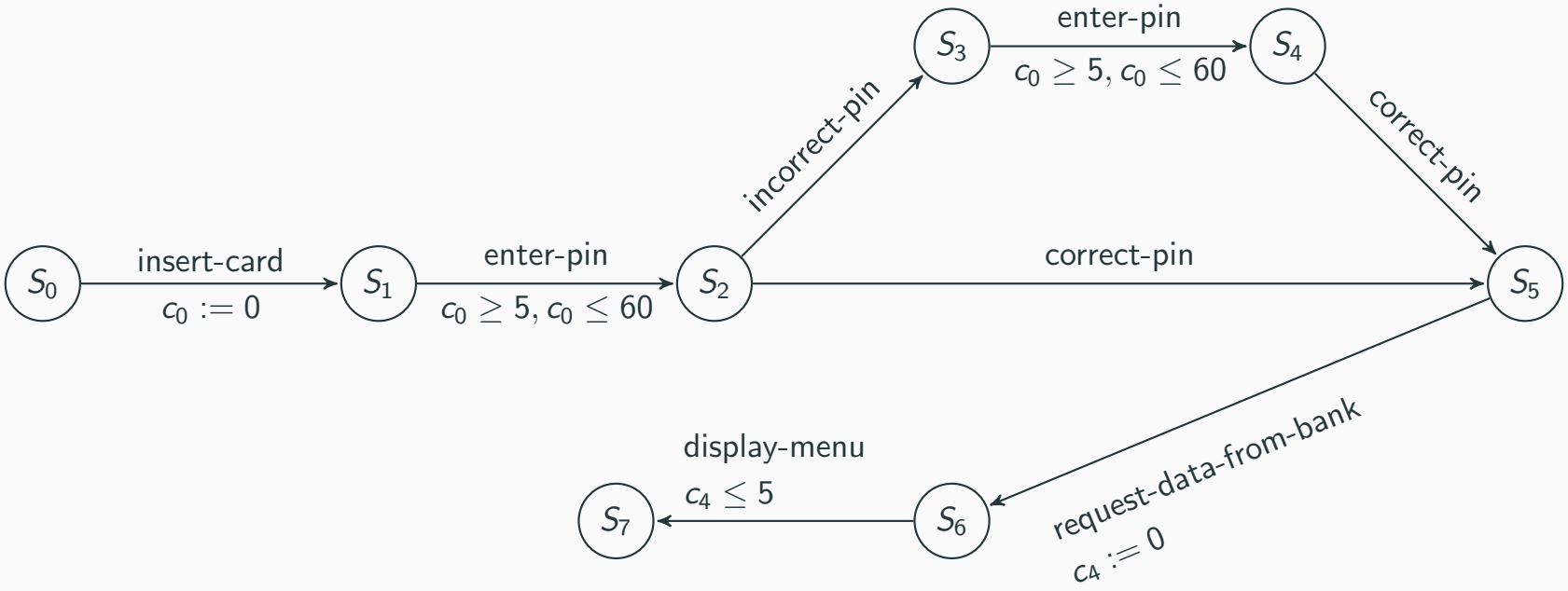
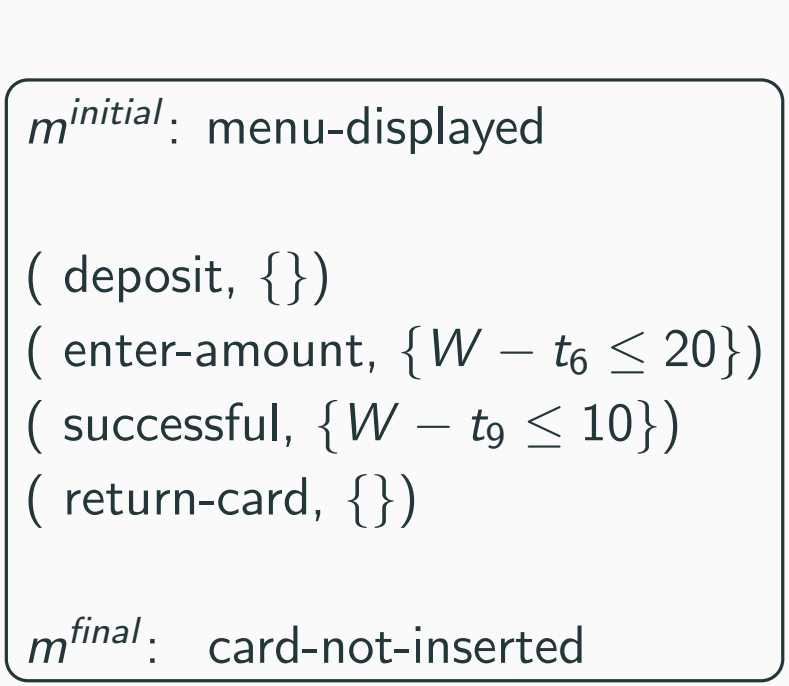


Figure 10: Timed automaton synthesized from Scenario 1 and Scenario 2

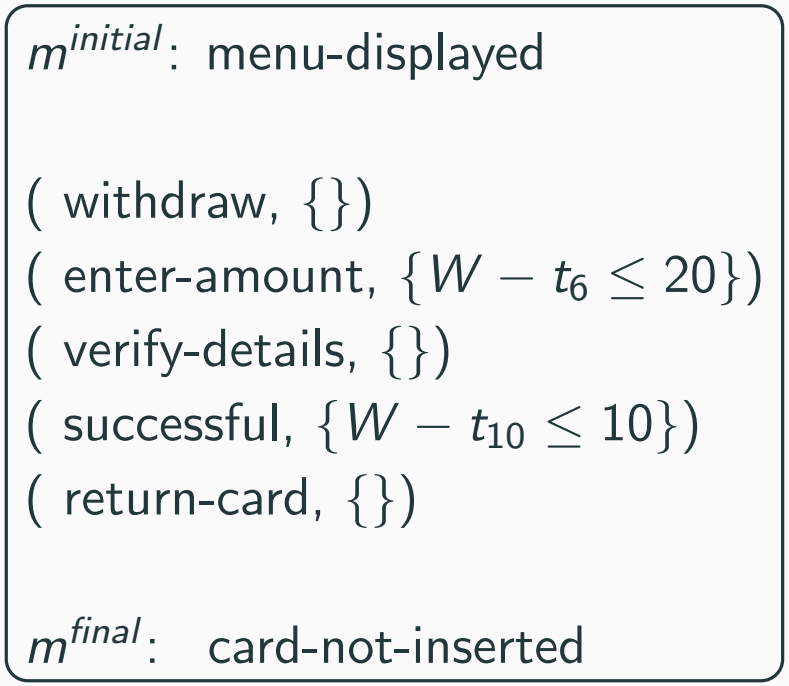
Automated Teller Machine (ATM)

Explain extended scenario

Automated Teller Machine (ATM)



TES of Scenario 3



TES of Scenario 4

Figure 11: Timed Event Sequences of the ATM with withdraw and deposit option

Automated Teller Machine (ATM)

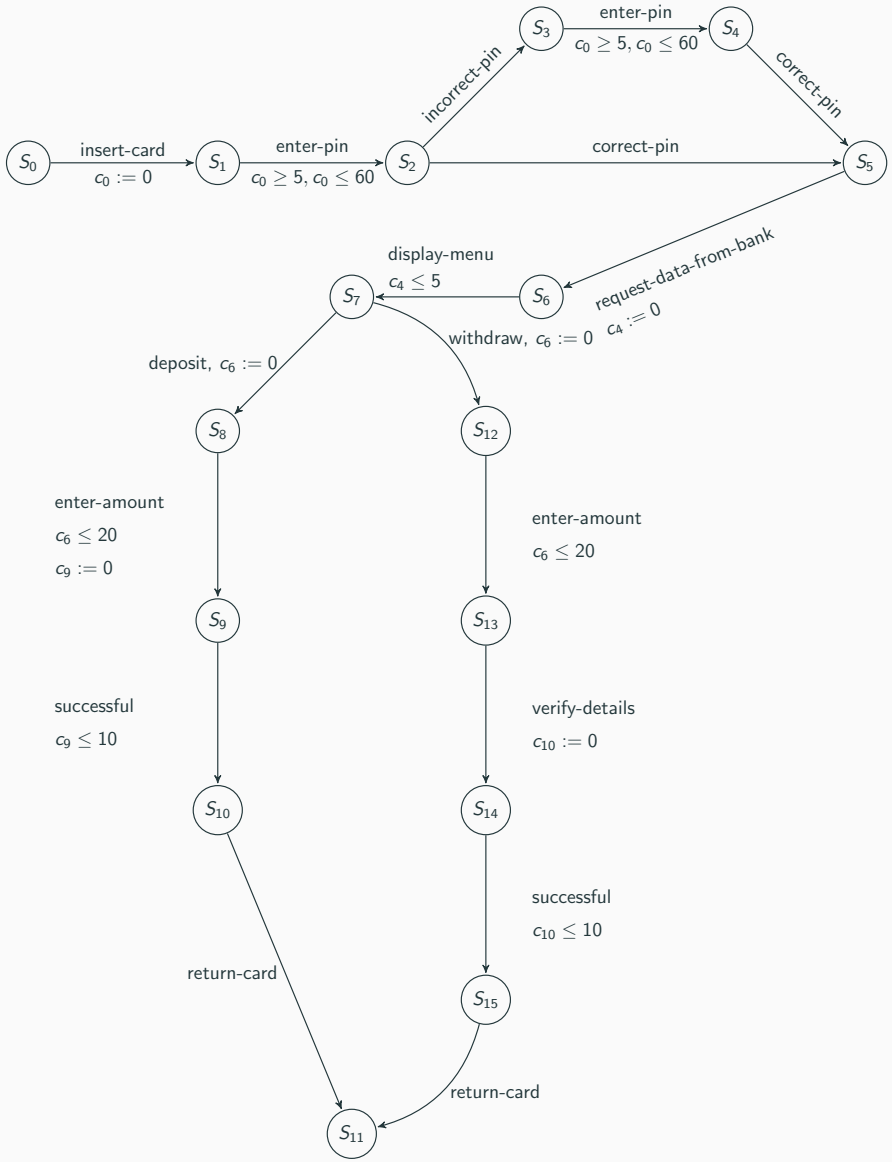


Figure 12: The synthesized timed automaton of the ATM

Light Control System

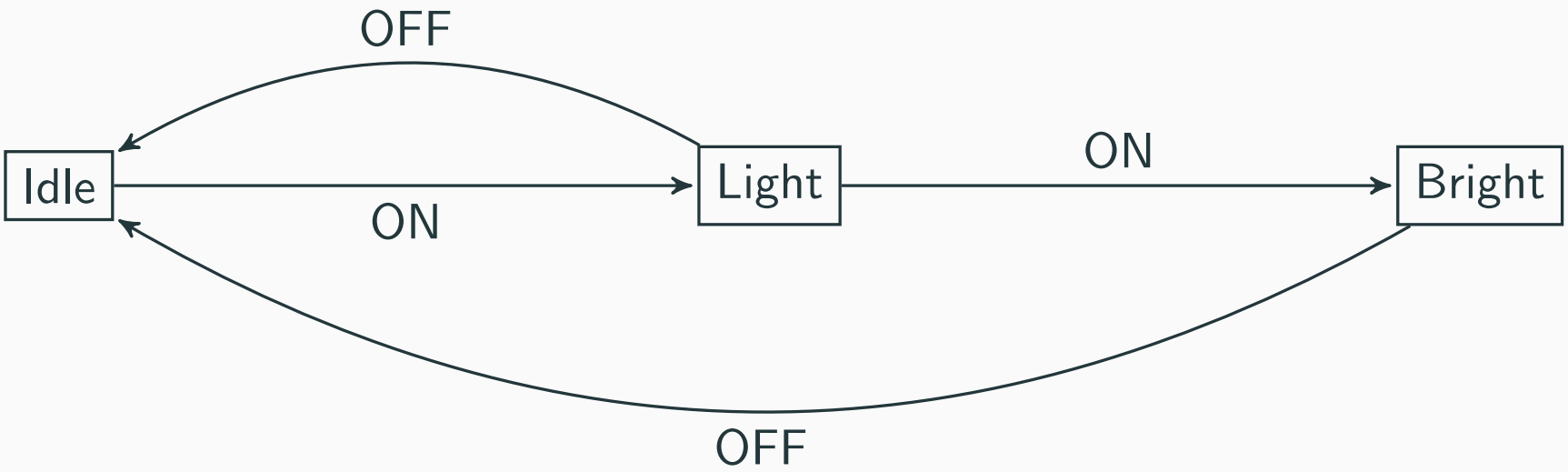


Figure 13: Mode graph of the Light Control System

Light Control System

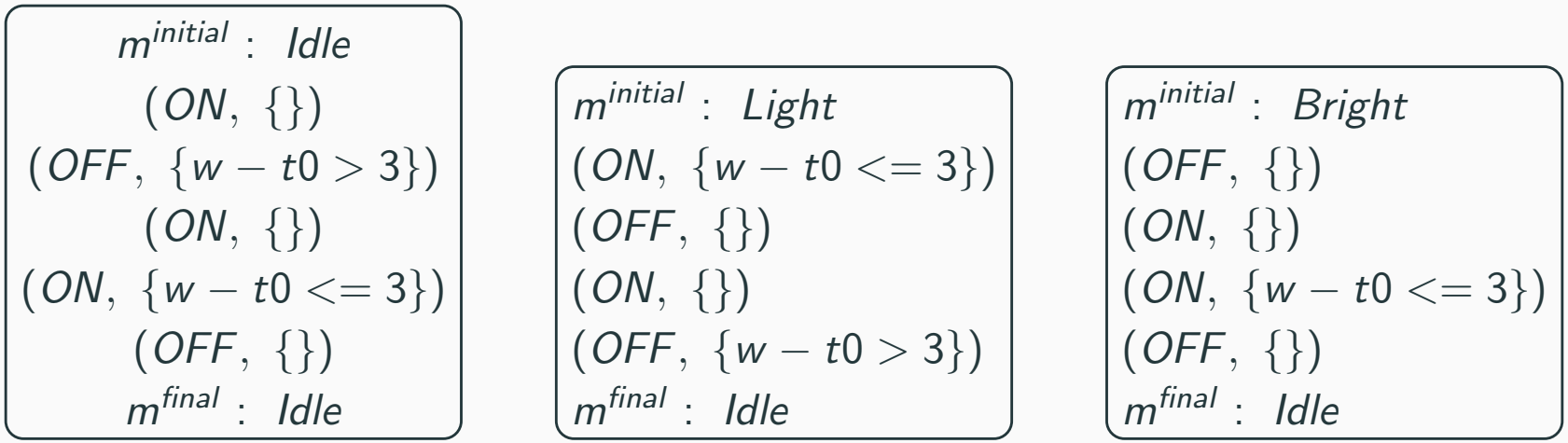


Figure 14: Timed Event Sequences of the Light Control System

Light Control System

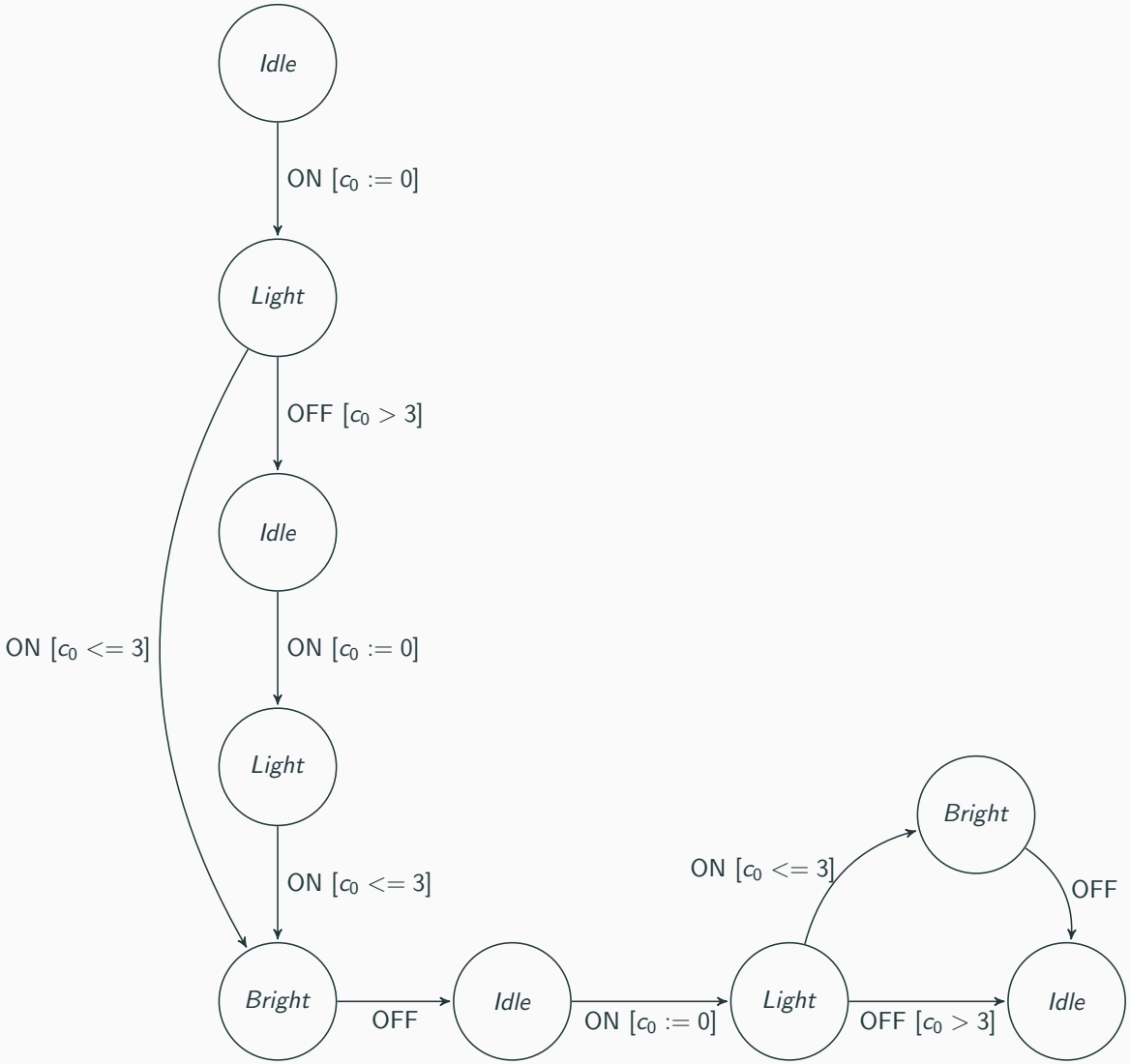


Figure 15: Timed automaton of the Light Control System

Traffic Light

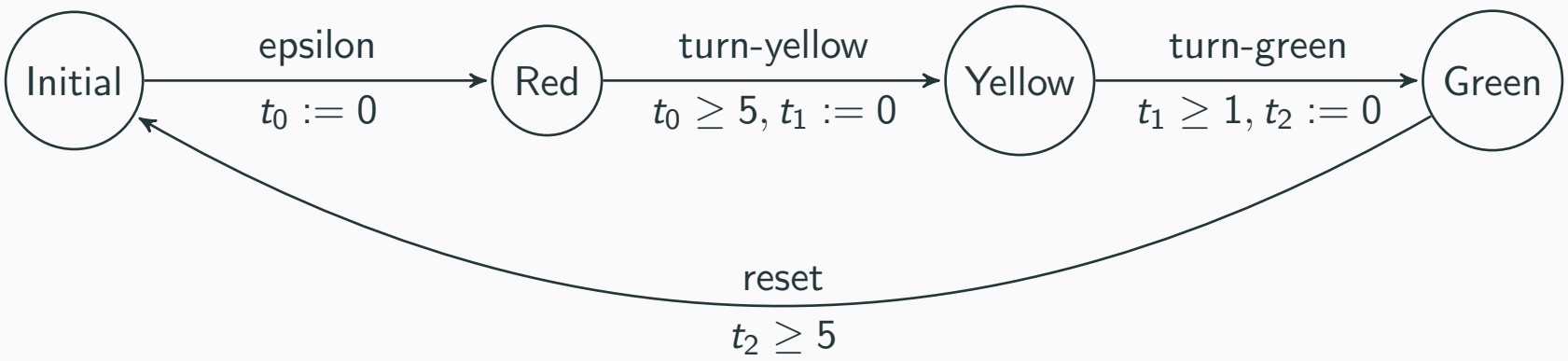


Figure 16: Timed automaton of the Traffic Light

Traffic Light

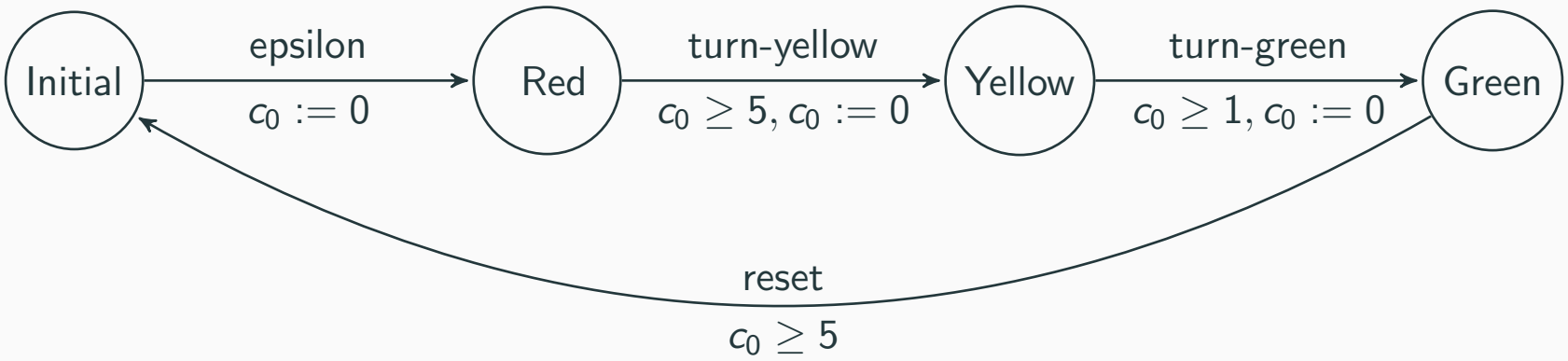


Figure 17: The optimally allocated timed automaton of the Traffic Light

CSMA/CD Protocol

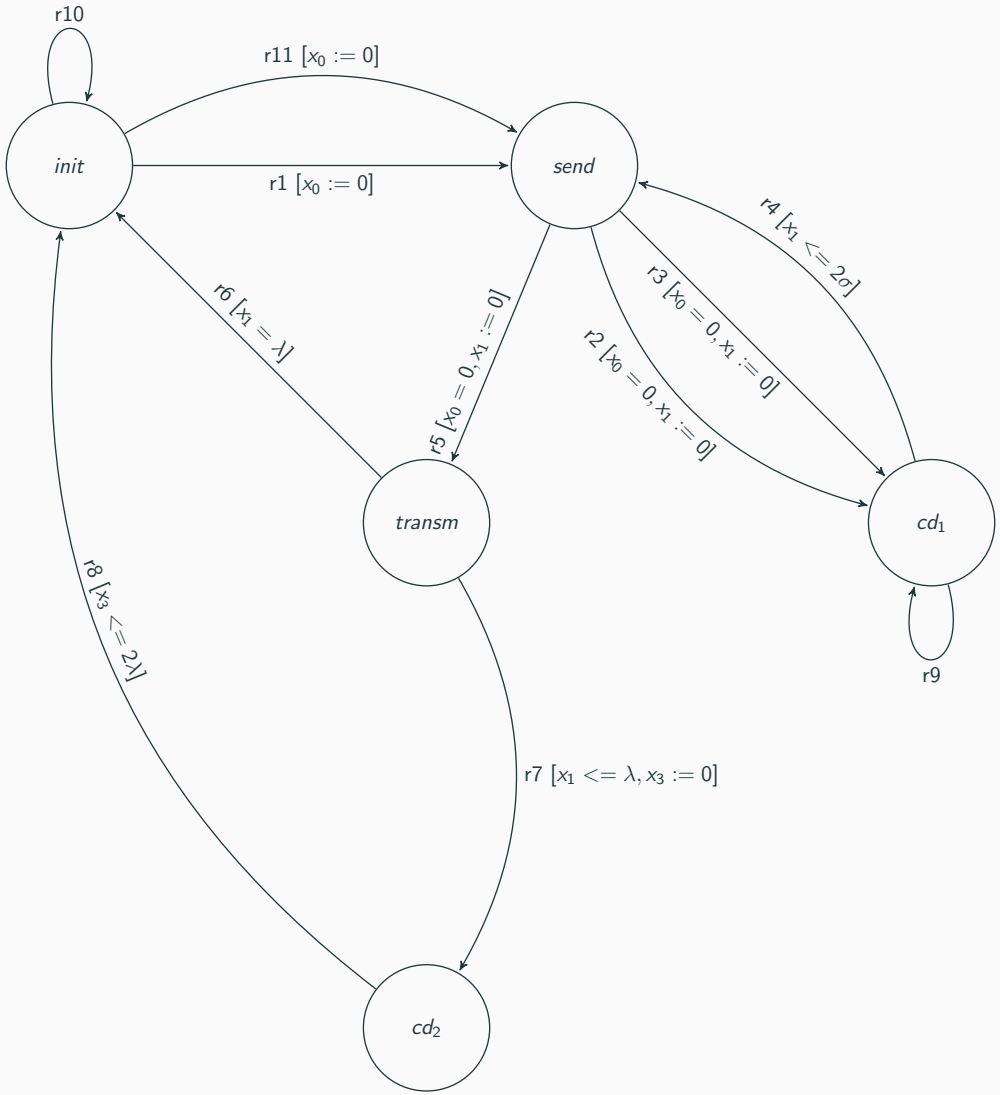


Figure 18: The timed automaton for the sender in CSMA/CD protocol

CSMA/CD Protocol

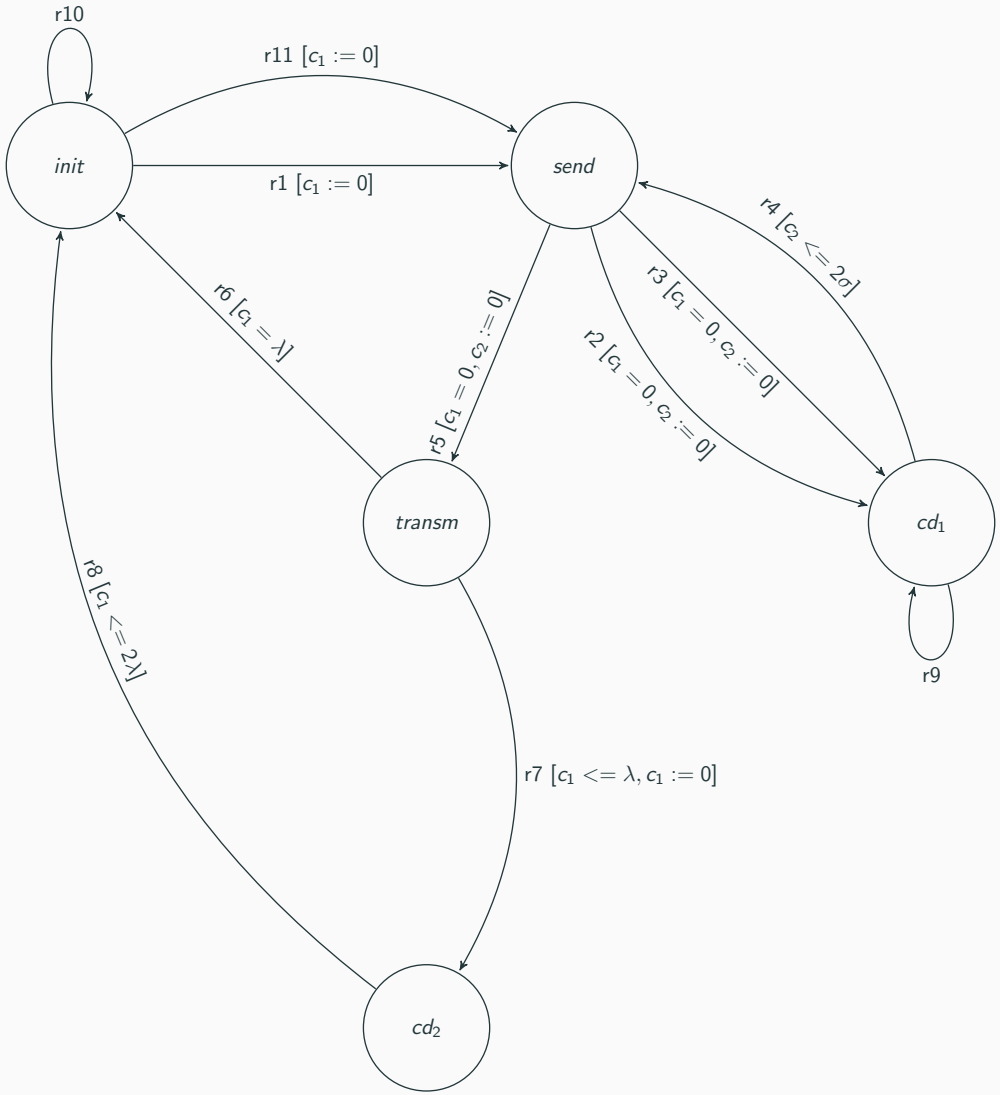


Figure 19: The optimally allocated timed automaton for the sender in CSMA/CD protocol

Conclusion

Conclusion

conclude here [2] [3]

THANK PROFESSORS??? Neda and Committee mem??

Questions?

References i


R. Alur and D. L. Dill.
A theory of timed automata.
Theor. Comput. Sci., 126(2):183–235, Apr. 1994.


N. Saeedloei.
From scenarios to timed automata.
Technical report, Department of Computer Science, University of Minnesota Duluth, Duluth, MN, jun 2016.


N. Saeedloei and F. Kluzniak.
Optimal clock allocation for a class of timed automata.
Technical report, Department of Computer Science, University of Minnesota Duluth, Duluth, MN, Sept 2016.