

# Verification and Validation of Cyber-Physical Systems

*Model and Verify CPS using "Büchi Automata"*

Enkeledi Mema

[enkeledi.mema@stud.hshl.de](mailto:enkeledi.mema@stud.hshl.de)

Electronic Engineering

# Motivation for verification and validation of CPS

On January 1990, AT&T  
Telephone Shortage



Costs: 100 Million US\$  
Source: Software flaw(Bug)

On June 1996 Ariane 5 crash



Costs: 500 Million US\$  
Source: Software flaw(Bug)

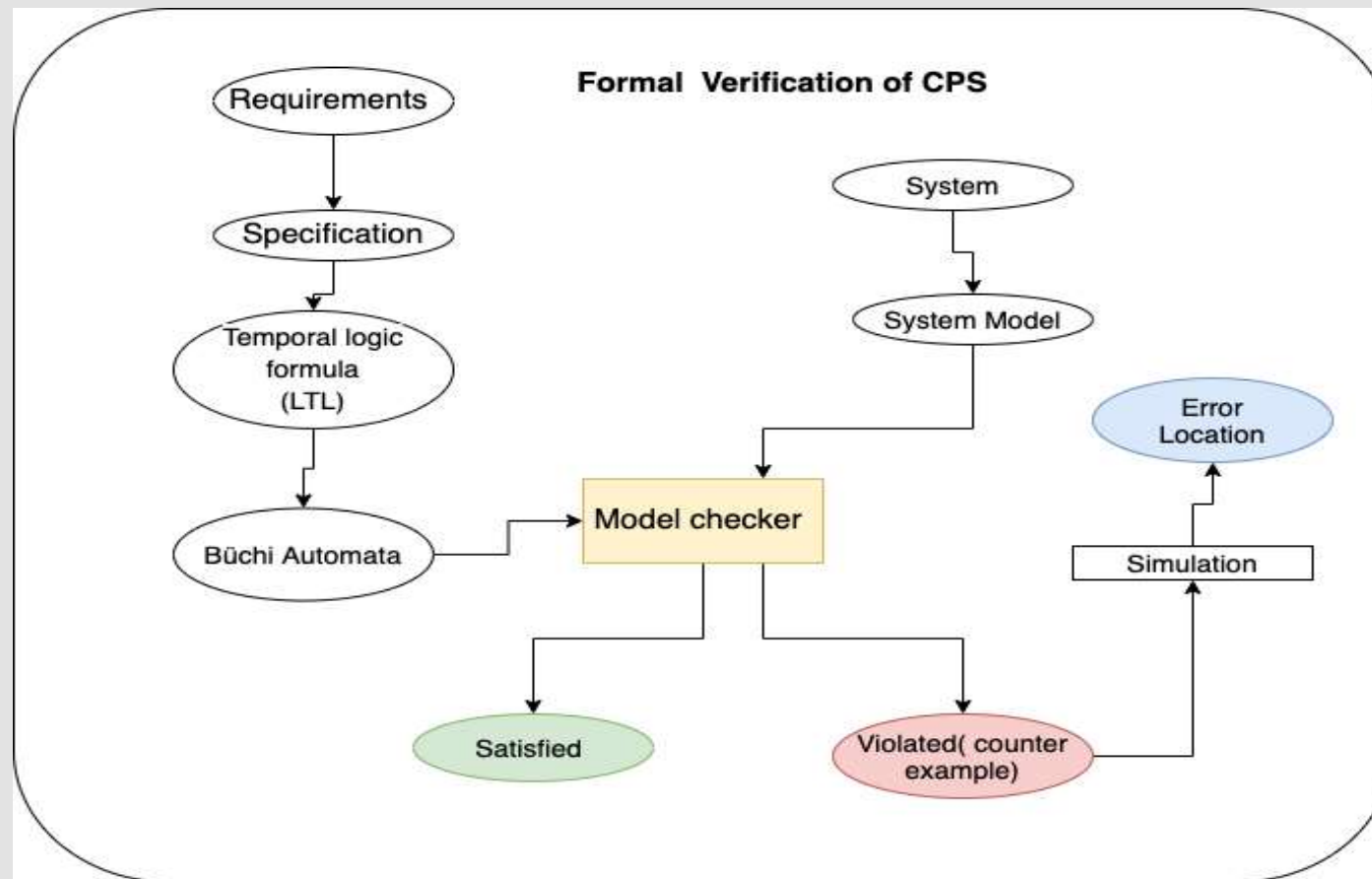
# Motivation for verification and validation of CPS

- Dependable
- Reliable
- Safe
- Secure

# Where are CPS used ?

- Precision farming
- Intelligent transportation
- Environmental control
- Avionics
- Traffic control and safety
- Process control
- Telemedicine
- Manufacturing
- Smart city

# Model checking overview



# Introduction to Büchi Automata

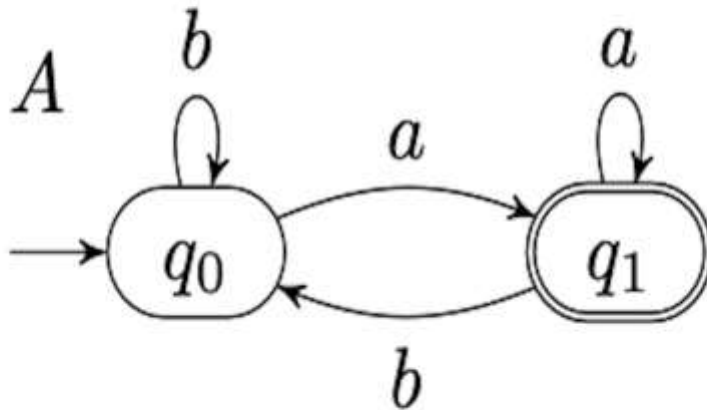
Let  $A(Q, \Sigma, \delta, q_0, F)$  a tuple be a deterministic Büchi automaton

- $Q$  is a set of states;
- $\Sigma$  is a finite set called the alphabet of  $A$ ;
- $\delta: Q \times \Sigma \rightarrow Q$  is a transition function of  $A$ ;
- $q_0$  is initial state of  $A$  ;
- $F$  is the acceptance condition ;

# Operations for BA application

- Determinization
- Emptiness checking
- Minimization

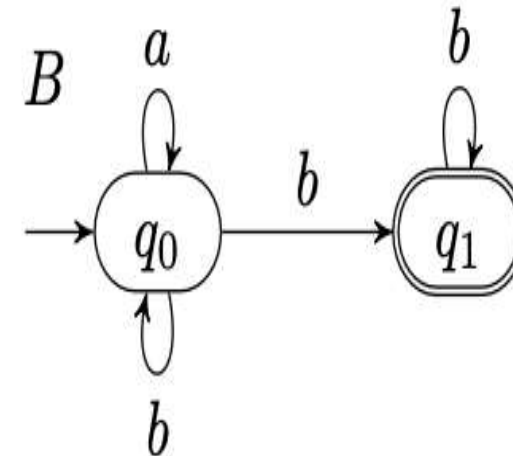
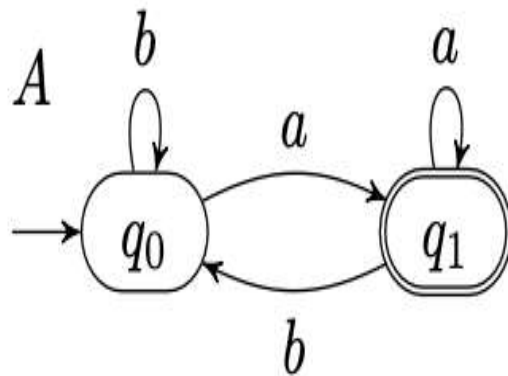
# Büchi automata example



- $\Sigma = \{a, b\}$  which is the alphabet.
- $\delta = \{(q_0, a, q_1), (q_0, b, q_0), (q_1, a, q_1), (q_1, b, q_0)\}$  the set of the transition.
- $F = \{q_1\}$  the acceptance condition.
- $q_0 = \{q_0\}$  the initial state.



# Büchi automata and Generalized Büchi automata



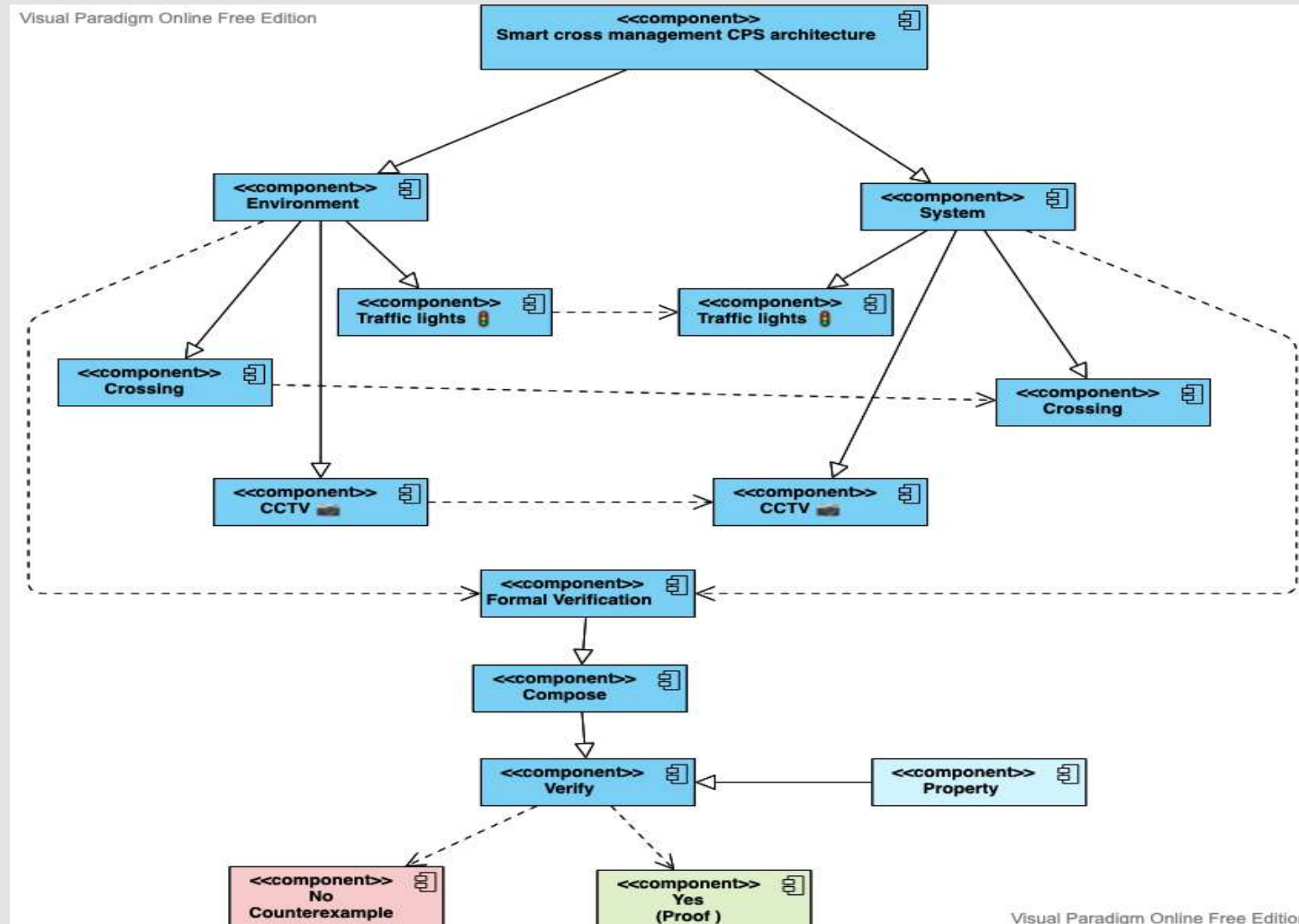
# CPS Use case (Smart cross-section)



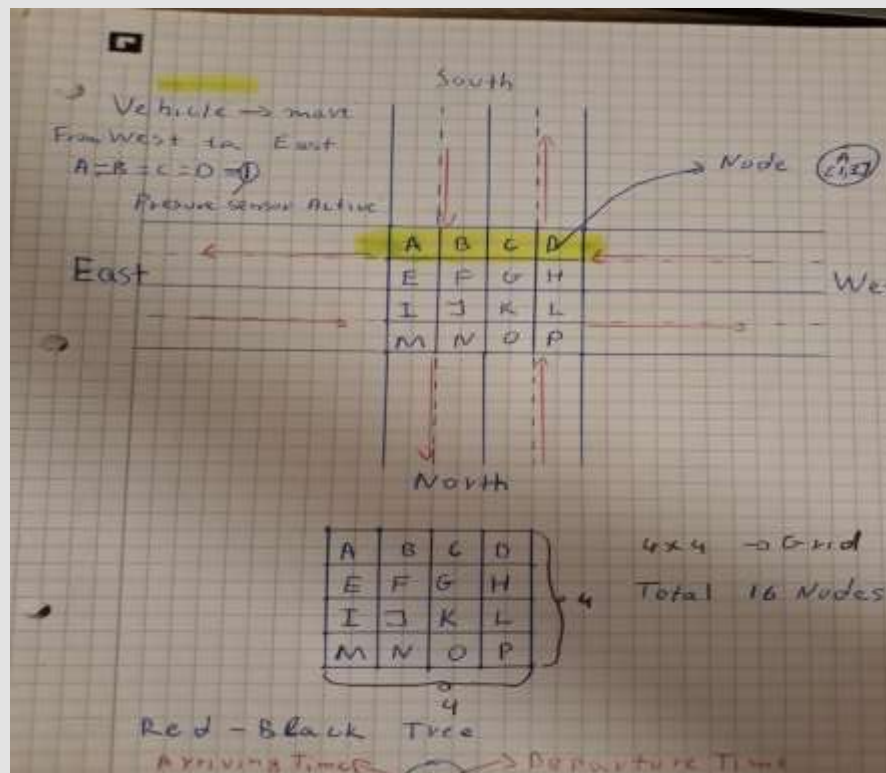
## Benefits:

- Dealing with safety critical systems.
- Facilitation in construction of smart cross-section.
- Time efficiency.

# CPS components for the use case



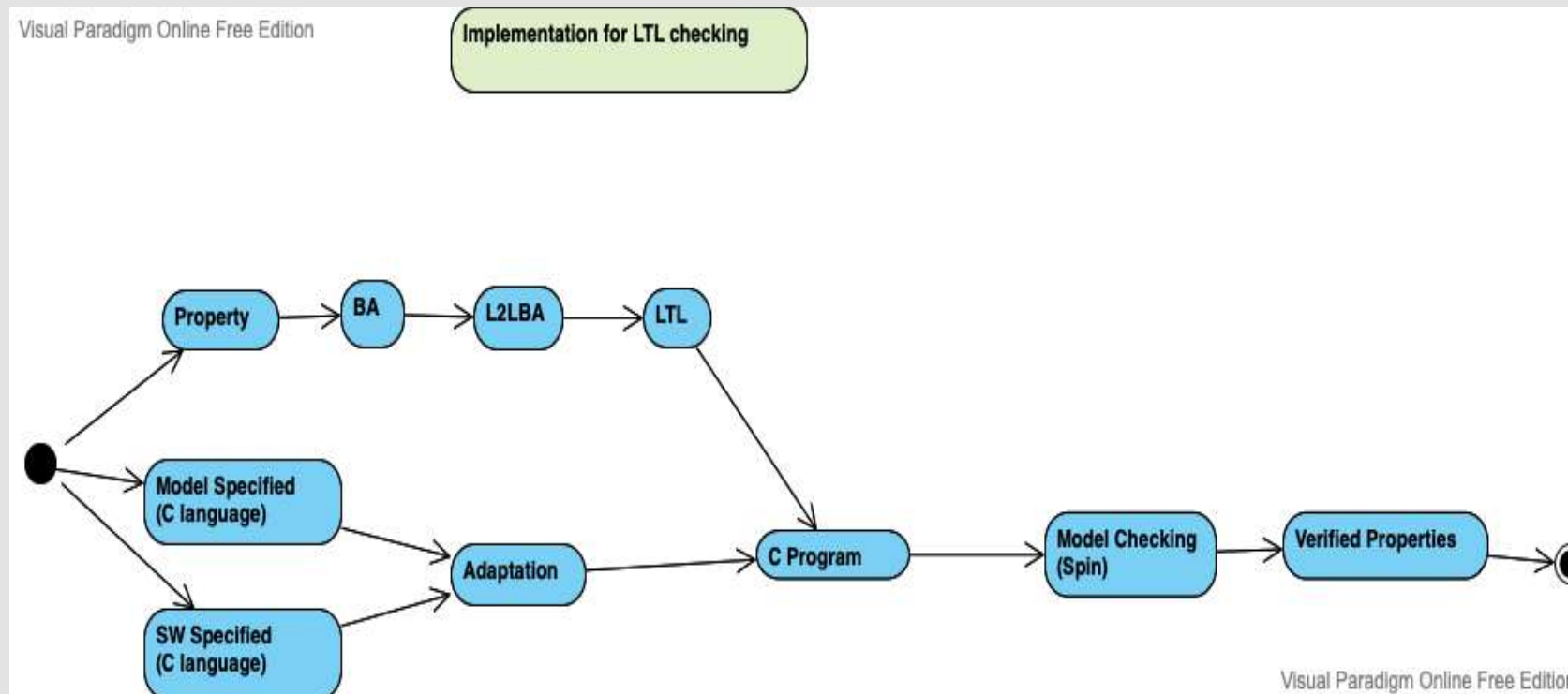
# Example for the use case



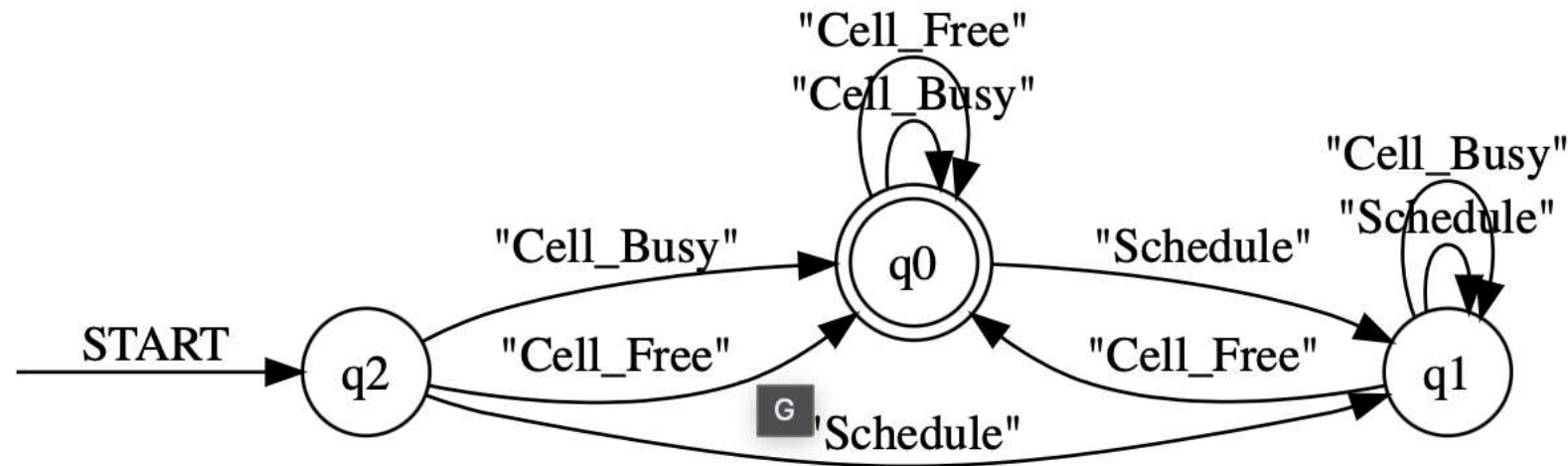
```

LTL = G ("Schedule" IMP (F "Cell_Free" AND NOT "Cell_Busy")),
ALPHABET = [Cell_Free, Schedule, Cell_Busy]
    
```

# Steps for the CPS verification



# BA for formal verification of Property



# BA converted to Spin

```
T0_init :    /* init */
  if
  :: (((! ((cell_free))) || ((schedule)))) -> goto accept_S20
  :: (! ((cell_busy))) -> goto T0_S39
  :: (((! ((cell_free)) && (cell_busy)) || ((cell_busy) && (schedule)))) -> goto accept_all
  fi;
accept_S20 :  /* 1 */
  if
  :: (((! ((cell_free))) || ((schedule)))) -> goto T0_init
  :: (! ((cell_busy))) -> goto T0_S39
  :: (((! ((cell_free)) && (cell_busy)) || ((cell_busy) && (schedule)))) -> goto accept_all
  fi;
accept_S39 :  /* 2 */
  if
  :: ((schedule)) -> goto T0_init
  :: (! ((cell_busy))) -> goto T0_S39
  :: ((cell_busy) && (schedule)) -> goto accept_all
  fi;
T0_S39 :     /* 3 */
  if
  :: ((schedule)) -> goto accept_S20
  :: (! ((cell_busy))) -> goto T0_S39
  :: (! ((cell_busy)) && (schedule)) -> goto accept_S39
  :: ((cell_busy) && (schedule)) -> goto accept_all
  fi;
accept_all : /* 4 */
  skip
```



# Spin example

Cross\_Traffic\_Enkeledi\_Mema.pml

Spin Version 6.5.1 -- 31 July 2020 :: iSpin Version 1.1.1 -- 23 February 2014

Edit/View Simulate / Replay Verification Swarm Run <Help> Save Session Restore Session <Quit>

Open... ReOpen Save Save As... Syntax Check Redundancy Check Symbol Table Find:

```

1  #define cell_free 1
2  #define cell_busy 2
3  chan communication_channel = [1] of { byte };
4  proctype check_cell_free_A()
5  {
6      communication_channel!cell_free
7
8      printf("Cell A Free!\n")
9  }
10 proctype check_cell_busy_A()
11 {
12     communication_channel!cell_busy
13     printf("Cell A busy!\n")
14 }
15 proctype check_cell_free_B()
16 {
17     communication_channel!cell_free
18
19     printf("Cell B Free!\n")
20 }
21 proctype check_cell_busy_B()
22 {
23     communication_channel!cell_busy
24     printf("Cell B busy!\n")
25 }
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Automata View zoom in zoom out

init

check\_cell\_free\_A

check\_cell\_busy\_A

S6

check\_cell\_free\_B

check\_cell\_busy\_B

S2

check\_cell\_free\_A()

S3

check\_cell\_busy\_A()

S4

check\_cell\_free\_B()

S5

check\_cell\_busy\_B()

(run C)

S7

end

spin: nothing to report

12 simulate/replay

13 verification

14 simulate/replay



# Video or live simulation

# Conclusion

- CPS verification and validation.
- Omega expression languages such as BA and GBA.
- CPS has huge potential to change the traffic behaviour in a smart city.

# Bibliography

- [AAF] Angluin, Dana; Antonopoulos, Timos; Fisman, Dana: Strongly Unambiguous Büchi Automata Are Polynomially Predictable With Membership Queries. p. 17 pages. Artwork Size: 17 pages Medium: application/pdf Publisher: Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik GmbH, Wadern/Saarbruecken, Germany Version Number: 1.0.
- [Al] Alur, Rajeev: Principles of cyber-physical systems. The MIT Press.
- [Ch] Chechik, Marsha: Automata-Theoretic LTL Model-Checking. p. 12.
- [EL] Eid, Abdulla; Lavalle, Steven M: Finite -Automata and Buchi Automata. p. 17.
- [Fi] Fisher, Corey S: A More Robust Corpus of Büchi Automata. p. 47.
- [Go] Goos, Gerhard; Hartmanis, Juris; van Leeuwen, Jan; Hutchison, David; Kanade, Takeo; Kittler, Josef; Kleinberg, Jon M; Mattern, Friedemann; Mitchell, John C; Naor, Moni; Nierstrasz, Oscar; Rangan, C Pandu; Steffen, Bernhard: Lecture Notes in Computer Science. p. 646.
- [Gr] Greer, Christopher; Burns, Martin; Wollman, David; Griffor, Edward: , Cyber-physical systems and internet of things.
- [He] Hempfling, Christina: Julius-Maximilians-Universität Würzburg Faculty of Mathematics and Computer Science Course of Studies: Computer Science. p. 120.
- [Li] Li, Yong; Turrini, Andrea; Chen, Yu-Fang; Zhang, Lijun: Learning Büchi Automata and Its Applications. 11430:38–98. Series Title: Lecture Notes in Computer Science.

# Thank you for attention!

## Questions?

