Distributed Controllers for Provably Life and Safe Car Manoeuvres on Freeways and in Urban Traffic – Also explainable?

Maike Schwammberger

University of Oldenburg

7th January 2019

GI-Dagstuhl Seminar ES4CPS, January 6-11, 2019

Personal Introduction

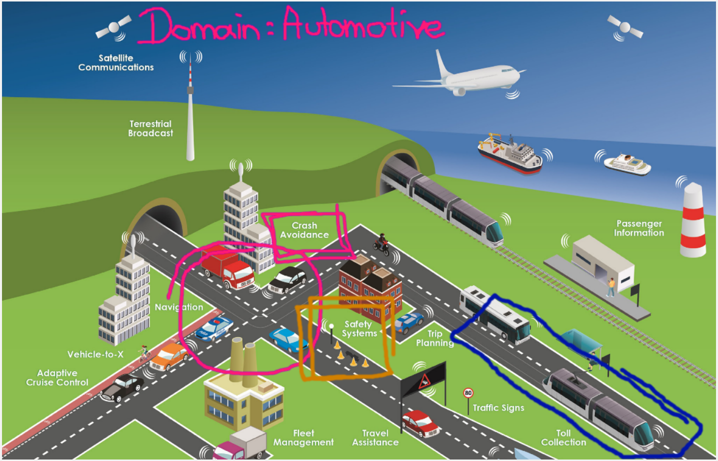
Who am I Research and teaching assistant at group *Correct System Design* at *University of Oldenburg* since March 2014

Current status Finishing phase of my PhD

Doctoral Supervisors Prof. Dr. Ernst-Rüdiger Olderog, Prof. Dr. Martin Fränzle

Subject of my PhD studies Distributed Controllers for Provably Life and Safe Car Manoeuvres on Freeways and in Urban Traffic

Projects Automatic Verification And Analysis of Complex Systems (AVACS, until Sept. 2015), Collegiate of DfG Research Training Group SCARE (since Sept. 2015)



Source: https://www.etsi.org/images/files/ETSITechnologyLeaflets/IntelligentTransportSystems.pdf

- Safety and Liveness of Autonomous Urban Traffic Manoeuvres (Intersections)
- Timely sending of Hazard Warning Messages
- Lane change (highway) and overtaking (country roads) protocols

Explainability for CPS – Talk Outline

1. An ES4CPS Problem
2. My expertise
3. Explainability of CPS in my approach
4. Outlook for this Seminar

ES4CPS Problem

Autonomous cars are...

- Distributed systems
- Mobile systems
- Systems of systems
- ...

Summarized:

Large complexity of autonomously acting cars!

Why does the car do what it does?

My Expertise – Overview

My area of expertise:
Formal specification of correct systems

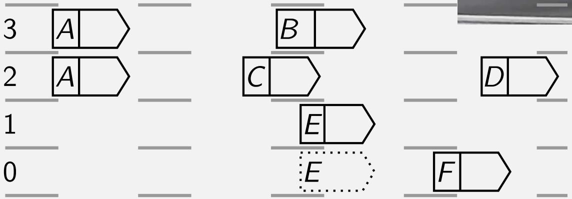
- Area of CPS: Discrete Control, formal abstraction from real-world

Overview:

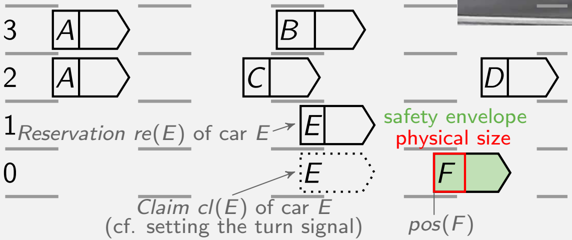
1. Formal approach for autonomous cars in (Urban) Traffic Scenarios
   - Abstract model: Urban road graph networks
   - Spatial logic UMLSL: Formalise traffic situations
   - Controllers: Formal semantics and protocols
2. Provably correct functional Controller Properties
   - Proof of safety and liveness
     - Mathematical proof and implementation
3. Case Study: A Hazard Warning Communication Protocol
   - Adapt MLSL to cope with hazards and prove timely warning

Basic Case: Highway Traffic [HLOR11]

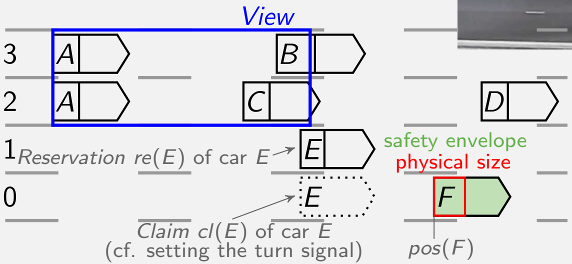
Abstract model for Highway traffic:



Abstract model for Highway traffic:

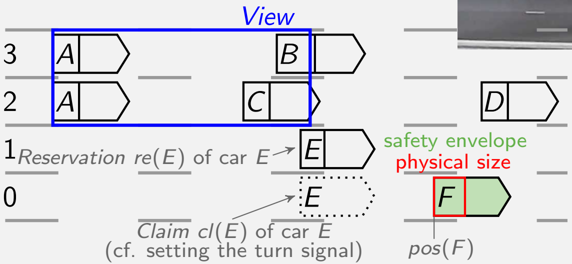


Abstract model for Highway traffic:



Traffic Snapshot: Contains positions, claims, ... of all cars in *one moment*

Abstract model for Highway traffic:

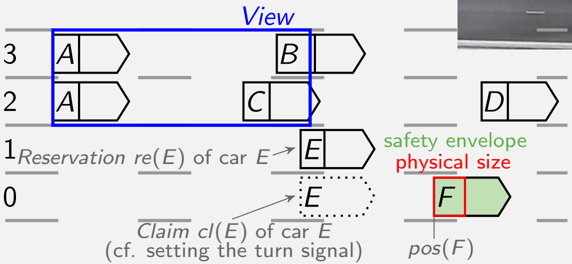


Traffic Snapshot: Contains positions, claims, ... of all cars in *one moment*

View: Consider only part of the road (locality)

Basic Case: Highway Traffic [HLOR11]

Abstract model for Highway traffic:



Traffic Snapshot: Contains positions, claims, ... of all cars in *one moment*

View: Consider only part of the road (locality)

Example MLSL formula: $\phi \equiv \begin{array}{l} re(A) \frown free \frown re(B) \\ re(A) \frown free \frown re(C) \end{array}$

Urban Traffic Manoeuvres [HS16, S18b]

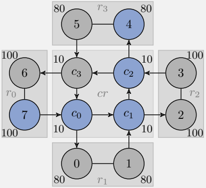
Source: china-mike.com

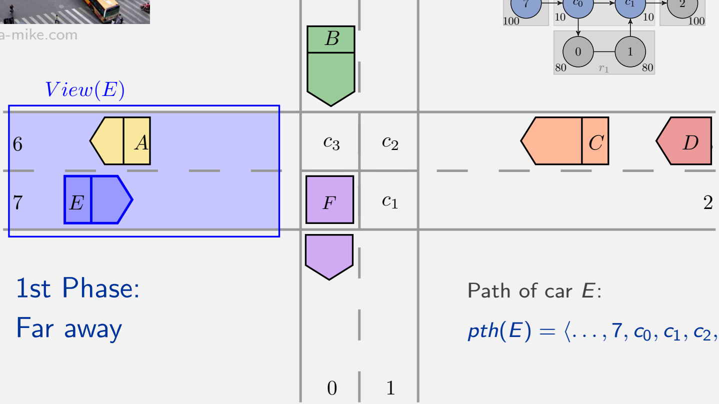
[HS16:] Hilscher, M., S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic (ICTAC 2016)
[S18b:] S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic, extension of [HS16] (TCS 2018)

Source: china-mike.com

Urban road network:

$\mathcal{N}$:





1st Phase:
Far away

Path of car E:

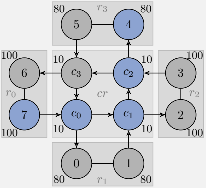
$pth(E) = \langle \dots, 7, c_0, c_1, c_2, 4, \dots \rangle$

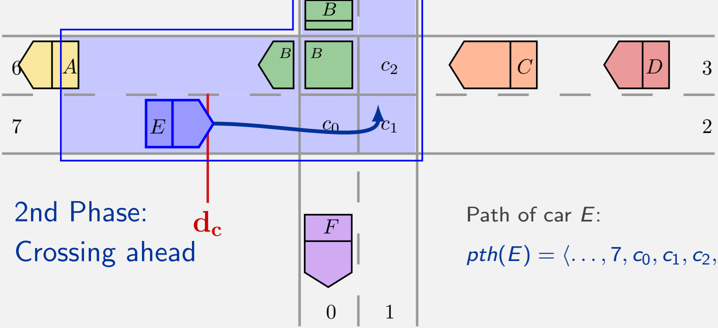
[HS16:] Hilscher, M., S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic (ICTAC 2016)
[S18b:] S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic, extension of [HS16] (TCS 2018)

Source: china-mike.com

Urban road network:

$\mathcal{N}$:





2nd Phase:
Crossing ahead

Path of car E:

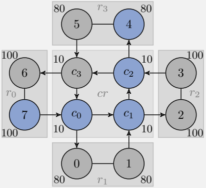
$pth(E) = \langle \ldots, 7, c_0, c_1, c_2, 4, \ldots \rangle$

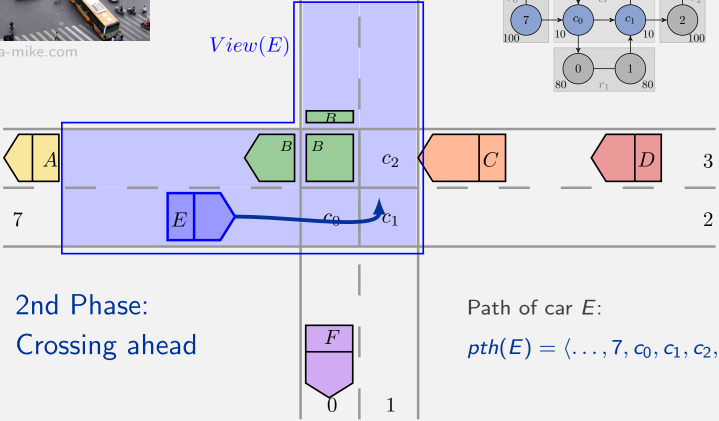
[HS16:] Hilscher, M., S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic (ICTAC 2016)
[S18b:] S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic, extension of [HS16] (TCS 2018)

Source: china-mike.com

Urban road network:

$\mathcal{N}$:





2nd Phase:
Crossing ahead

Path of car E:

$pth(E) = \langle \ldots, 7, c_0, c_1, c_2, 4, \ldots \rangle$

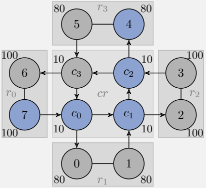
[HS16:] Hilscher, M., S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic (ICTAC 2016)
[S18b:] S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic, extension of [HS16] (TCS 2018)

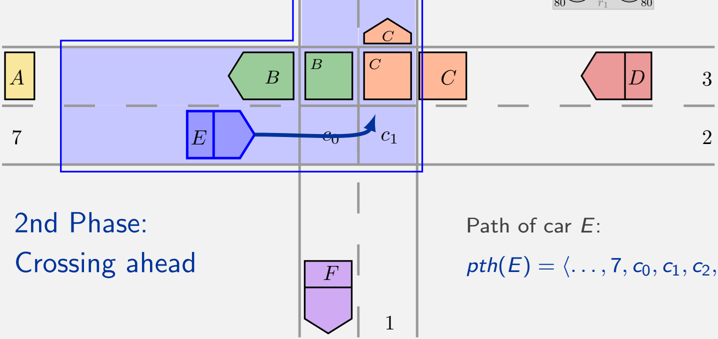
Urban Traffic Manoeuvres [HS16, S18b]

Source: china-mike.com

Urban road network:

$\mathcal{N}$:





2nd Phase:

Crossing ahead

Path of car E:

$pth(E) = \langle \ldots, 7, c_0, c_1, c_2, 4, \ldots \rangle$

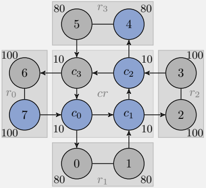
[HS16:] Hilscher, M., S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic (ICTAC 2016)
[S18b:] S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic, extension of [HS16] (TCS 2018)

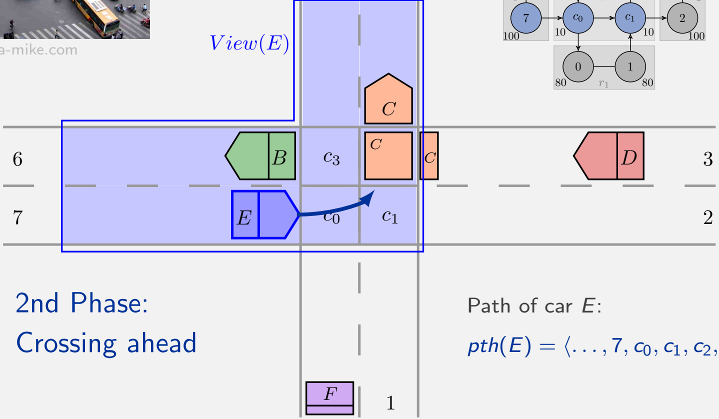
Urban Traffic Manoeuvres [HS16, S18b]

Source: china-mike.com

Urban road network:

$\mathcal{N}$:





2nd Phase:
Crossing ahead

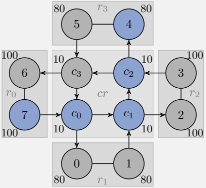
Path of car E:

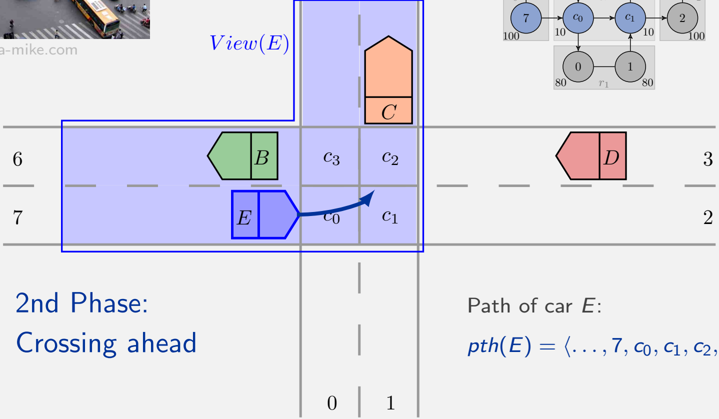
$pth(E) = \langle \dots, 7, c_0, c_1, c_2, 4, \dots \rangle$

[HS16:] Hilscher, M., S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic (ICTAC 2016)
[S18b:] S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic, extension of [HS16] (TCS 2018)

Source: china-mike.com

Urban road network:





2nd Phase:
Crossing ahead

Path of car E:

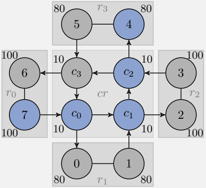
$pth(E) = \langle \dots, 7, c_0, c_1, c_2, 4, \dots \rangle$

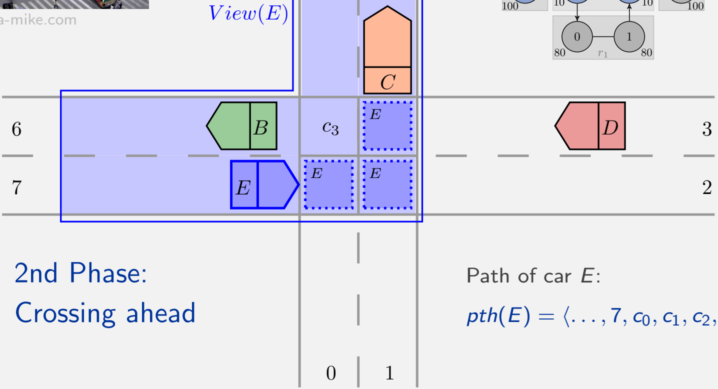
[HS16:] Hilscher, M., S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic (ICTAC 2016)
[S18b:] S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic, extension of [HS16] (TCS 2018)

Urban Traffic Manoeuvres [HS16, S18b]

Source: china-mike.com

Urban road network:





2nd Phase:
Crossing ahead

Path of car E:

$pth(E) = \langle \ldots, 7, c_0, c_1, c_2, 4, \ldots \rangle$

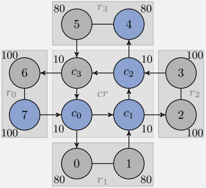
[HS16:] Hilscher, M., S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic (ICTAC 2016)
[S18b:] S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic, extension of [HS16] (TCS 2018)

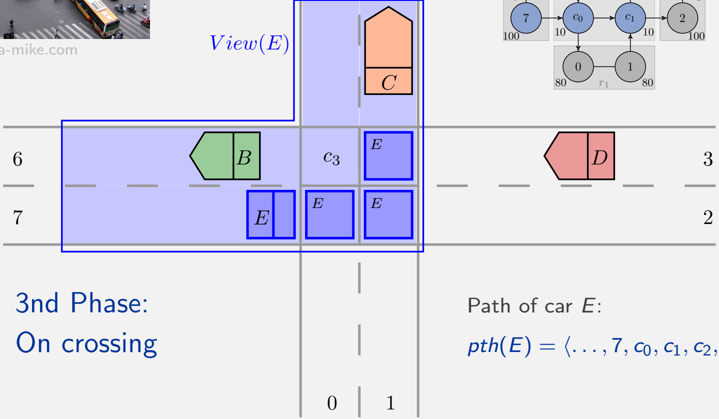
Urban Traffic Manoeuvres [HS16, S18b]

Source: china-mike.com

Urban road network:

$\mathcal{N}$:





3nd Phase:
On crossing

Path of car E:

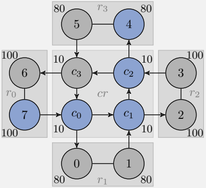
$pth(E) = \langle \ldots, 7, c_0, c_1, c_2, 4, \ldots \rangle$

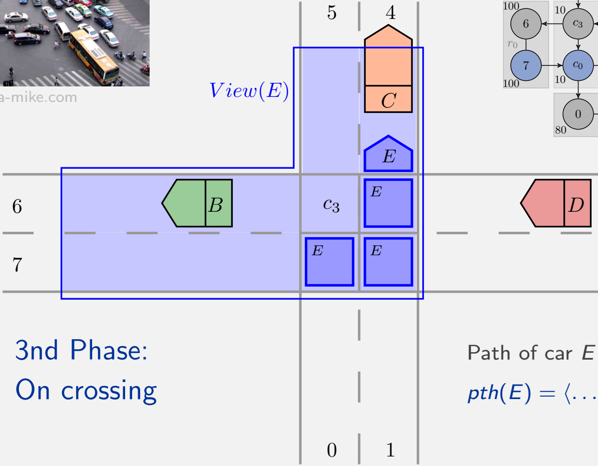
[HS16:] Hilscher, M., S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic (ICTAC 2016)
[S18b:] S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic, extension of [HS16] (TCS 2018)

Source: china-mike.com

Urban road network:

$\mathcal{N}$:





3nd Phase:
On crossing

Path of car E:

$pth(E) = \langle \dots, 7, c_0, c_1, c_2, 4, \dots \rangle$

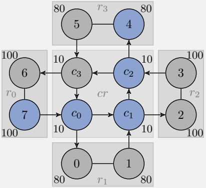
[HS16:] Hilscher, M., S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic (ICTAC 2016)
[S18b:] S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic, extension of [HS16] (TCS 2018)

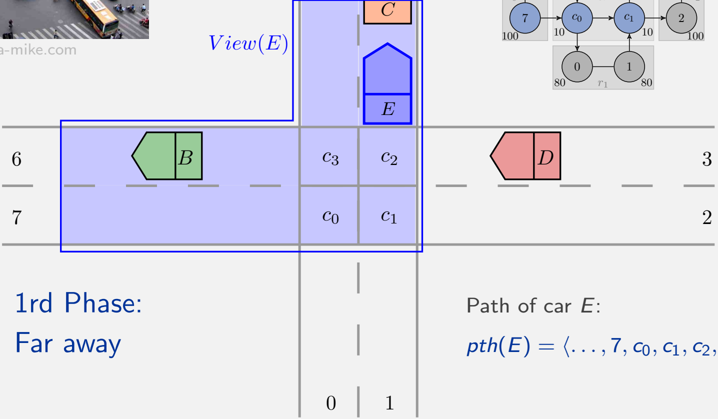
Urban Traffic Manoeuvres [HS16, S18b]

Source: china-mike.com

Urban road network:

$\mathcal{N}$:





1rd Phase:

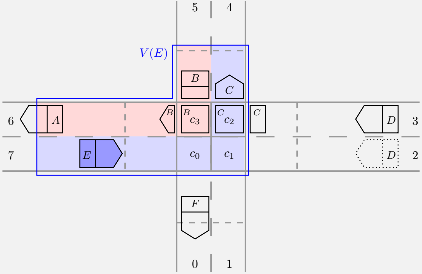
Far away

Path of car E:

$pth(E) = \langle \ldots, 7, c_0, c_1, c_2, 4, \ldots \rangle$

[HS16:] Hilscher, M., S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic (ICTAC 2016)
[S18b:] S., M.: An Abstract Model for Proving Safety of Autonomous Urban Traffic, extension of [HS16] (TCS 2018)

Logical Reasoning [HS16, S18b]

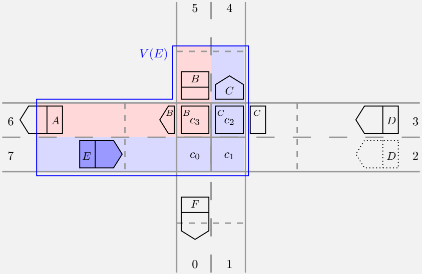


General Idea for reasoning:

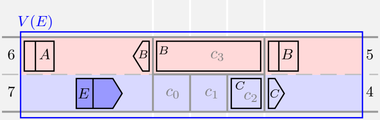
- Detach car dynamics from spatial and real-time view [MRY02]
- Use (extended version of) logic MLSL developed for highway traffic [HLOR11] and country roads [HLO13]
- Cannot reason around the corner with spatial logic MLSL
- Need to deal with bended view

[MRY02]: Moor, T., Raisch, J., O'Young, S.: *Discrete Supervisory Control of Hybrid Systems Based on l-Complete Approximations* (2002)

Logical Reasoning [HS16, S18b]

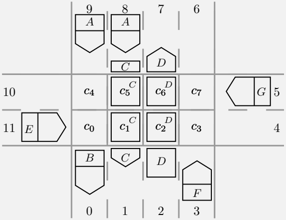


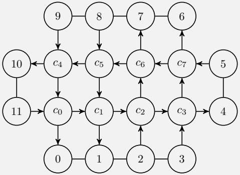
Unbend view to virtual lane:

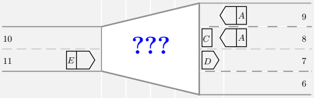


⇒ Reasoning with extended version of MLSL on a straight lane!

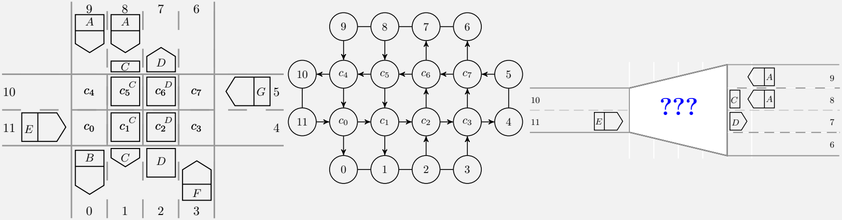
- Any type of intersection possible
    - Generic virtual lane construction
    - One virtual lane for each possible path through the intersection
    - Combine virtual lanes to parallelised virtual views





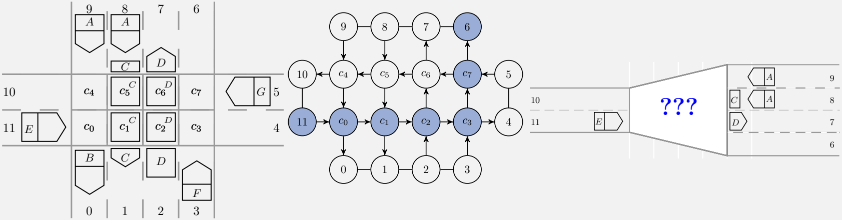


- Any type of intersection possible
  - Generic virtual lane construction
  - One virtual lane for each possible path through the intersection
  - Combine virtual lanes to parallelised virtual views



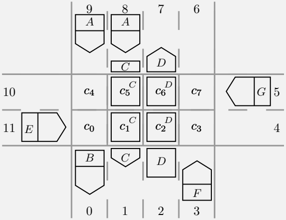
Generic Urban Road Network [S18b]

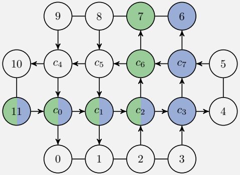
- Any type of intersection possible
  - Generic virtual lane construction
  - One virtual lane for each possible path through the intersection
  - Combine virtual lanes to parallelised virtual views

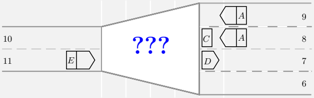


Generic Urban Road Network [S18b]

- Any type of intersection possible
  - Generic virtual lane construction
  - One virtual lane for each possible path through the intersection
  - Combine virtual lanes to parallelised virtual views

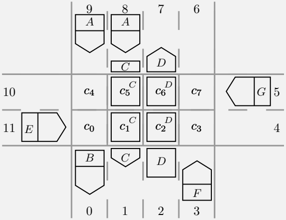


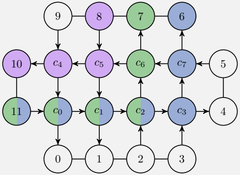


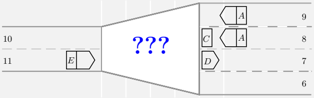


Generic Urban Road Network [S18b]

- Any type of intersection possible
  - Generic virtual lane construction
  - One virtual lane for each possible path through the intersection
  - Combine virtual lanes to parallelised virtual views

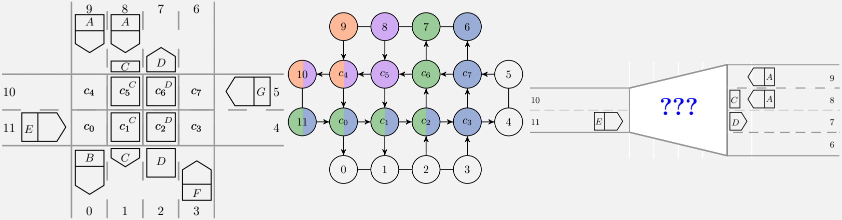




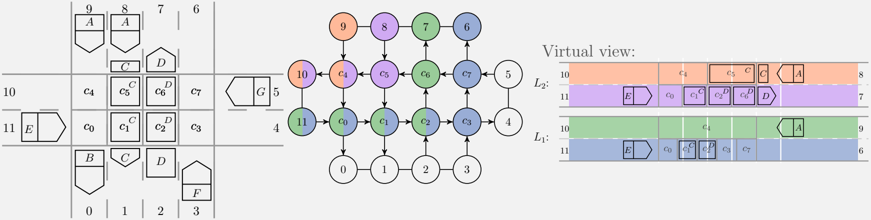


Generic Urban Road Network [S18b]

- Any type of intersection possible
  - Generic virtual lane construction
  - One virtual lane for each possible path through the intersection
  - Combine virtual lanes to parallelised virtual views

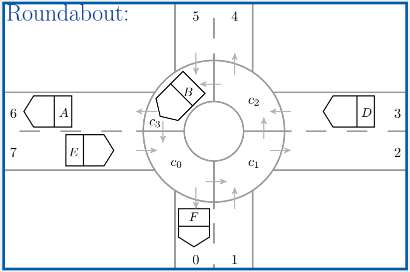


- Any type of intersection possible
  - Generic virtual lane construction
  - One virtual lane for each possible path through the intersection
  - Combine virtual lanes to parallelised virtual views

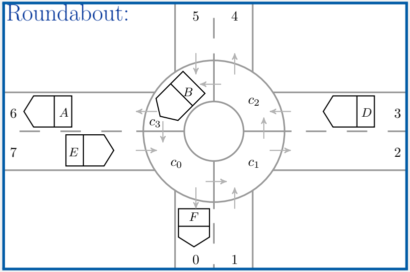


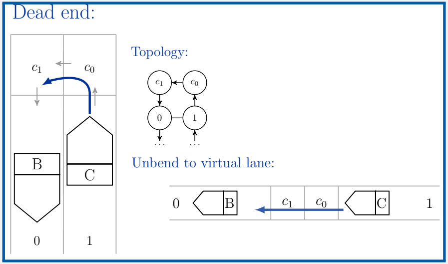
Possible Urban Structures – Examples

Roundabout:



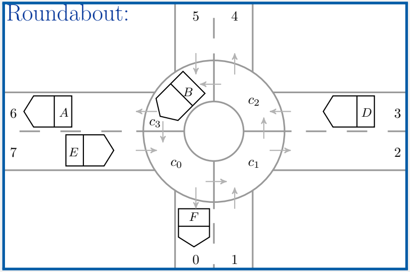
Possible Urban Structures – Examples



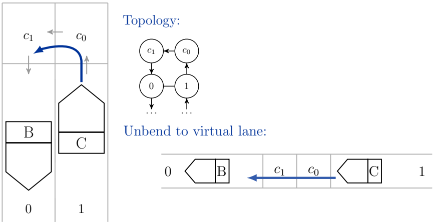


Possible Urban Structures – Examples

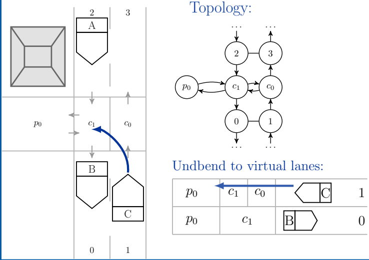
Roundabout:



Dead end:

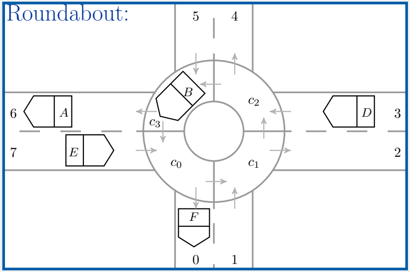


Parking spot:

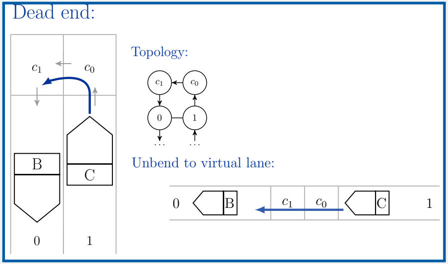


Possible Urban Structures – Examples

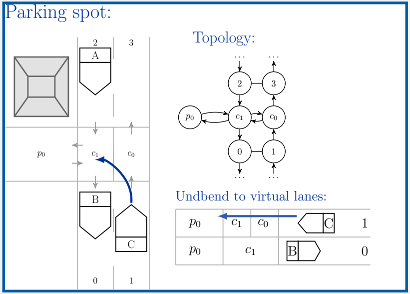
Roundabout:



Dead end:



Parking spot:



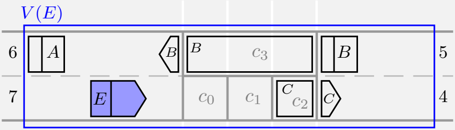
• • •

Urban Multi-lane Spatial Logic [HS16, S18b]

Syntax:

$$\phi ::= \mathit{true} \mid u = v \mid \mathit{free} \mid \mathit{cs} \mid \mathit{re}(c) \mid \mathit{cl}(c) \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \exists c : \phi_1 \mid \phi_1 \frown \phi_2 \mid \begin{array}{c}\phi_2\\ \phi_1\end{array}$$

Example:



Valuation: $\nu(\mathit{ego}) = E$, $\nu(a) = A$, $\nu(b) = B$, $\nu(c) = C$.

Formula 1: Crossing is ahead of E:
$\mathit{ca}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \frown \mathit{free}^{<d_c} \wedge \neg\langle \mathit{cs} \rangle \frown \mathit{cs} \rangle$

Formula 2: No collision with E exists:
$\neg \mathit{col}(\mathrm{ego}) \equiv \neg \exists d : d \neq \mathrm{ego} \wedge \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{re}(d) \rangle$

Formula 3: Position of E is on crossing segment:
$\mathit{oc}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{cs} \rangle$

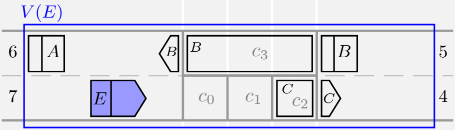
Formula 4: Position of B is on crossing segment:
$\mathit{oc}(b) \equiv \langle \mathit{re}(b) \wedge \mathit{cs} \rangle$

Urban Multi-lane Spatial Logic [HS16, S18b]

Syntax:

$$\phi ::= \mathit{true} \mid u = v \mid \mathit{free} \mid \mathit{cs} \mid \mathit{re}(c) \mid \mathit{cl}(c) \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \exists c : \phi_1 \mid \phi_1 \frown \phi_2 \mid \begin{array}{c}\phi_2\\ \phi_1\end{array}$$

Example:



Valuation: $\nu(\mathit{ego}) = E$, $\nu(a) = A$, $\nu(b) = B$, $\nu(c) = C$.

Formula 1: Crossing is ahead of E:
$\mathit{ca}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \frown \mathit{free}^{<d_c} \wedge \neg\langle \mathit{cs}\rangle \frown \mathit{cs}\rangle$

Formula 2: No collision with E exists:
$\neg\mathit{col}(\mathrm{ego}) \equiv \neg\exists d : d \neq \mathrm{ego} \wedge \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{re}(d)\rangle$

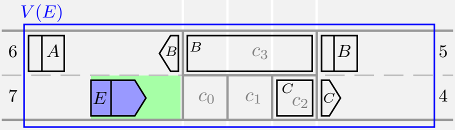
Formula 3: Position of E is on crossing segment:
$\mathit{oc}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{cs}\rangle$

Formula 4: Position of B is on crossing segment:
$\mathit{oc}(b) \equiv \langle \mathit{re}(b) \wedge \mathit{cs}\rangle$

Syntax:

$$\phi ::= \mathit{true} \mid u = v \mid \mathit{free} \mid \mathit{cs} \mid \mathit{re}(c) \mid \mathit{cl}(c) \mid \neg\phi \mid \phi_1 \land \phi_2 \mid \exists c : \phi_1 \mid \phi_1 \frown \phi_2 \mid \begin{matrix}\phi_2\\ \phi_1\end{matrix}$$

Example:



Valuation: $\nu(\mathit{ego}) = E$, $\nu(a) = A$, $\nu(b) = B$, $\nu(c) = C$.

Formula 1: Crossing is ahead of E:
$\mathit{ca}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \frown \mathit{free}^{<d_c} \land \neg\langle \mathit{cs}\rangle \frown \mathit{cs}\rangle$ ✔

Formula 2: No collision with E exists:
$\neg \mathit{col}(\mathrm{ego}) \equiv \neg\exists d : d \neq \mathrm{ego} \land \langle \mathit{re}(\mathrm{ego}) \land \mathit{re}(d)\rangle$

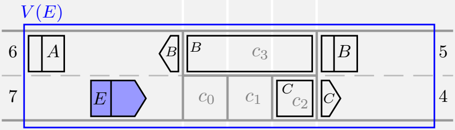
Formula 3: Position of E is on crossing segment:
$\mathit{oc}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \land \mathit{cs}\rangle$

Formula 4: Position of B is on crossing segment:
$\mathit{oc}(b) \equiv \langle \mathit{re}(b) \land \mathit{cs}\rangle$

Syntax:

$$\phi ::= \mathit{true} \mid u = v \mid \mathit{free} \mid \mathit{cs} \mid \mathit{re}(c) \mid \mathit{cl}(c) \mid \neg\phi \mid \phi_1 \land \phi_2 \mid \exists c : \phi_1 \mid \phi_1 \frown \phi_2 \mid \begin{array}{c}\phi_2\\ \phi_1\end{array}$$

Example:



Valuation: $\nu(\mathit{ego}) = E$, $\nu(a) = A$, $\nu(b) = B$, $\nu(c) = C$.

Formula 1: Crossing is ahead of E:
$\mathit{ca}(\mathrm{ego}) \;\equiv\; \langle \mathit{re}(\mathrm{ego}) \frown \mathit{free}^{<d_c} \land \neg\langle \mathit{cs}\rangle \frown \mathit{cs}\rangle$ ✔

Formula 2: No collision with E exists:
$\neg \mathit{col}(\mathrm{ego}) \;\equiv\; \neg\exists d : d \neq \mathrm{ego} \land \langle \mathit{re}(\mathrm{ego}) \land \mathit{re}(d)\rangle$

Formula 3: Position of E is on crossing segment:
$\mathit{oc}(\mathrm{ego}) \;\equiv\; \langle \mathit{re}(\mathrm{ego}) \land \mathit{cs}\rangle$

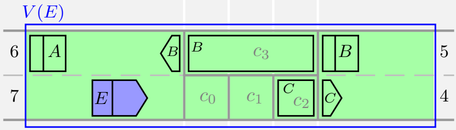
Formula 4: Position of B is on crossing segment:
$\mathit{oc}(b) \;\equiv\; \langle \mathit{re}(b) \land \mathit{cs}\rangle$

Urban Multi-lane Spatial Logic [HS16, S18b]

Syntax:

$$\phi ::= \mathit{true} \mid u = v \mid \mathit{free} \mid \mathit{cs} \mid \mathit{re}(c) \mid \mathit{cl}(c) \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \exists c : \phi_1 \mid \phi_1 \frown \phi_2 \mid \begin{array}{c}\phi_2\\ \phi_1\end{array}$$

Example:



Valuation: $\nu(\mathit{ego}) = E$, $\nu(a) = A$, $\nu(b) = B$, $\nu(c) = C$.

Formula 1: Crossing is ahead of E:
$\mathit{ca}(\mathrm{ego}) \;\equiv\; \langle \mathit{re}(\mathrm{ego}) \frown \mathit{free}^{<d_c} \wedge \neg\langle \mathit{cs}\rangle \frown \mathit{cs}\rangle$ ✔

Formula 2: No collision with E exists:
$\neg\mathit{col}(\mathrm{ego}) \;\equiv\; \neg\exists d : d \neq \mathrm{ego} \wedge \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{re}(d)\rangle$ ✔

Formula 3: Position of E is on crossing segment:
$\mathit{oc}(\mathrm{ego}) \;\equiv\; \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{cs}\rangle$

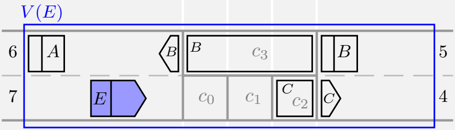
Formula 4: Position of B is on crossing segment:
$\mathit{oc}(b) \;\equiv\; \langle \mathit{re}(b) \wedge \mathit{cs}\rangle$

Urban Multi-lane Spatial Logic [HS16, S18b]

Syntax:

$$\phi ::= \mathit{true} \mid u = v \mid \mathit{free} \mid \mathit{cs} \mid \mathit{re}(c) \mid \mathit{cl}(c) \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \exists c : \phi_1 \mid \phi_1 \frown \phi_2 \mid \frac{\phi_2}{\phi_1}$$

Example:



Valuation: $\nu(\mathit{ego}) = E$, $\nu(a) = A$, $\nu(b) = B$, $\nu(c) = C$.

Formula 1: Crossing is ahead of E:
$\mathit{ca}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \frown \mathit{free}^{<d_c} \wedge \neg\langle \mathit{cs} \rangle \frown \mathit{cs} \rangle$ ✔

Formula 2: No collision with E exists:
$\neg \mathit{col}(\mathrm{ego}) \equiv \neg\exists d : d \neq \mathrm{ego} \wedge \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{re}(d) \rangle$ ✔

Formula 3: Position of E is on crossing segment:
$\mathit{oc}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{cs} \rangle$

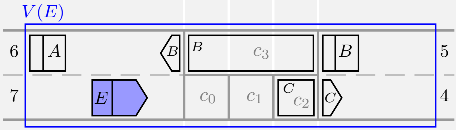
Formula 4: Position of B is on crossing segment:
$\mathit{oc}(b) \equiv \langle \mathit{re}(b) \wedge \mathit{cs} \rangle$

Urban Multi-lane Spatial Logic [HS16, S18b]

Syntax:

$$\phi ::= \mathit{true} \mid u = v \mid \mathit{free} \mid \mathit{cs} \mid \mathit{re}(c) \mid \mathit{cl}(c) \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \exists c : \phi_1 \mid \phi_1 \frown \phi_2 \mid \begin{array}{c}\phi_2\\ \phi_1\end{array}$$

Example:



Valuation: $\nu(ego) = E$, $\nu(a) = A$, $\nu(b) = B$, $\nu(c) = C$.

Formula 1: Crossing is ahead of E:
$\mathit{ca}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \frown \mathit{free}^{<d_c} \wedge \neg\langle \mathit{cs}\rangle \frown \mathit{cs}\rangle$ ✔

Formula 2: No collision with E exists:
$\neg \mathit{col}(\mathrm{ego}) \equiv \neg\exists d : d \neq \mathrm{ego} \wedge \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{re}(d)\rangle$ ✔

Formula 3: Position of E is on crossing segment:
$\mathit{oc}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{cs}\rangle$ ✘

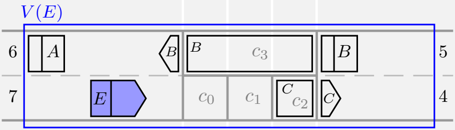
Formula 4: Position of B is on crossing segment:
$\mathit{oc}(b) \equiv \langle \mathit{re}(b) \wedge \mathit{cs}\rangle$

Urban Multi-lane Spatial Logic [HS16, S18b]

Syntax:

$$\phi ::= \mathit{true} \mid u = v \mid \mathit{free} \mid \mathit{cs} \mid \mathit{re}(c) \mid \mathit{cl}(c) \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \exists c : \phi_1 \mid \phi_1 \frown \phi_2 \mid \begin{matrix}\phi_2\\ \phi_1\end{matrix}$$

Example:



Valuation: $\nu(\mathit{ego}) = E$, $\nu(a) = A$, $\nu(b) = B$, $\nu(c) = C$.

Formula 1: Crossing is ahead of E:
$\mathit{ca}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \frown \mathit{free}^{<d_c} \wedge \neg\langle \mathit{cs}\rangle \frown \mathit{cs}\rangle$ ✔

Formula 2: No collision with E exists:
$\neg \mathit{col}(\mathrm{ego}) \equiv \neg\exists d : d \neq \mathrm{ego} \wedge \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{re}(d)\rangle$ ✔

Formula 3: Position of E is on crossing segment:
$\mathit{oc}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{cs}\rangle$ ✘

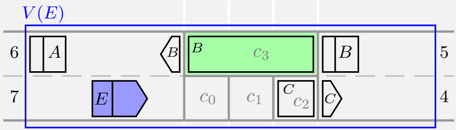
Formula 4: Position of B is on crossing segment:
$\mathit{oc}(b) \equiv \langle \mathit{re}(b) \wedge \mathit{cs}\rangle$

Urban Multi-lane Spatial Logic [HS16, S18b]

Syntax:

$$\phi ::= \mathit{true} \mid u = v \mid \mathit{free} \mid \mathit{cs} \mid \mathit{re}(c) \mid \mathit{cl}(c) \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \exists c : \phi_1 \mid \phi_1 \frown \phi_2 \mid \begin{array}{c}\phi_2\\ \phi_1\end{array}$$

Example:



Valuation: $\nu(\mathit{ego}) = E$, $\nu(a) = A$, $\nu(b) = B$, $\nu(c) = C$.

Formula 1: Crossing is ahead of E:
$\mathit{ca}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \frown \mathit{free}^{<d_c} \wedge \neg\langle \mathit{cs} \rangle \frown \mathit{cs} \rangle$ ✔

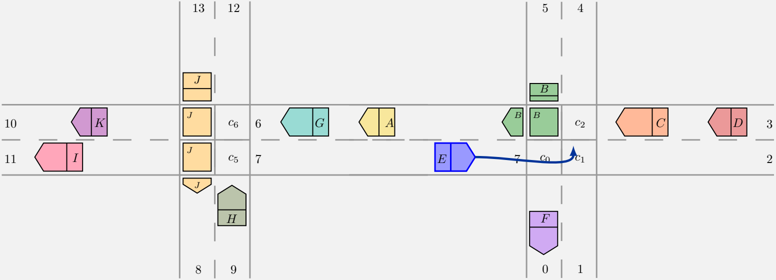
Formula 2: No collision with E exists:
$\neg \mathit{col}(\mathrm{ego}) \equiv \neg \exists d : d \neq \mathrm{ego} \wedge \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{re}(d) \rangle$ ✔

Formula 3: Position of E is on crossing segment:
$\mathit{oc}(\mathrm{ego}) \equiv \langle \mathit{re}(\mathrm{ego}) \wedge \mathit{cs} \rangle$ ✘

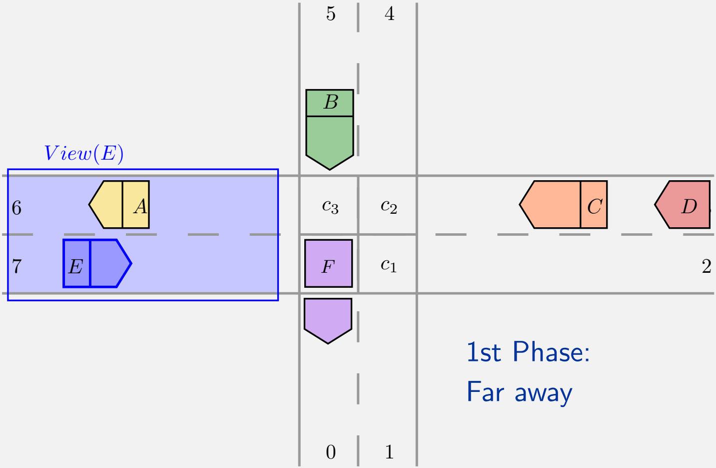
Formula 4: Position of B is on crossing segment:
$\mathit{oc}(b) \equiv \langle \mathit{re}(b) \wedge \mathit{cs} \rangle$ ✔

Controller – Assumptions I

- Controller type: Automotive-Controlling Timed Automata
- For urban traffic, equip every car with the following controllers:
  - Distance Controller [DHO06]
    - Keep safety distance to car in front or to intersection
  - Road Controller [HLO13]
    - Handles parts between intersections (≈ country roads)
    - Lane change manoeuvres with opposing traffic
  - Crossing Controller [HS16, S18$_2$]
    - Safely cross an intersection

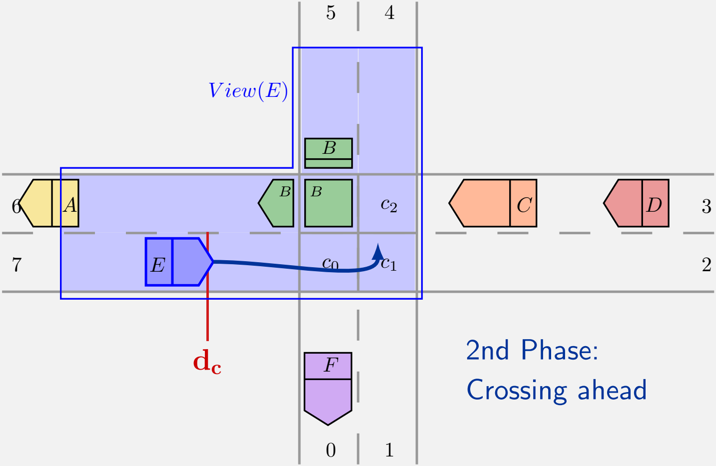


Crossing Controller – Phases (Reminder)



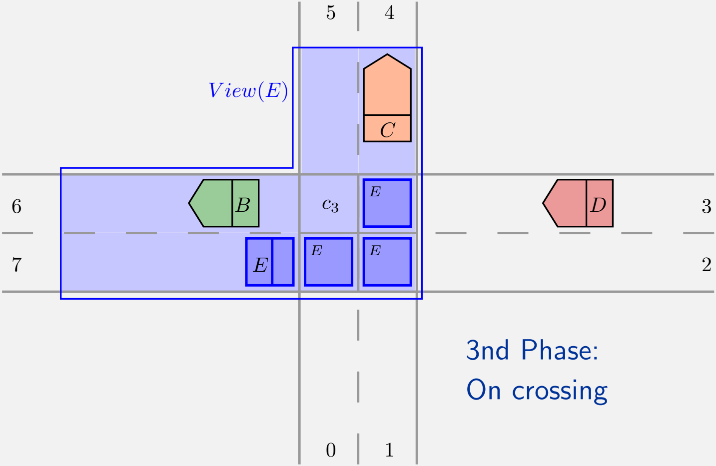
1st Phase:
Far away

Crossing Controller – Phases (Reminder)



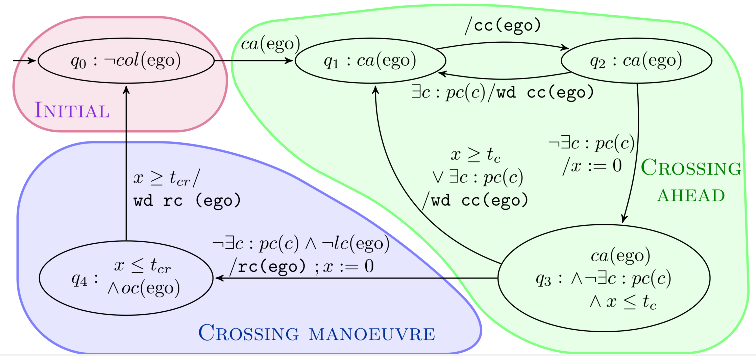
2nd Phase:
Crossing ahead

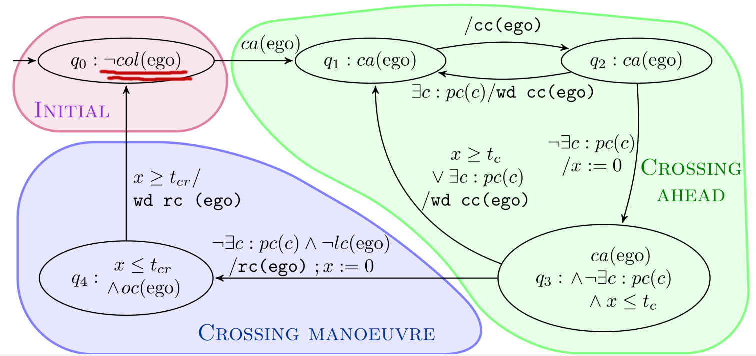
Crossing Controller – Phases (Reminder)



3nd Phase:
On crossing

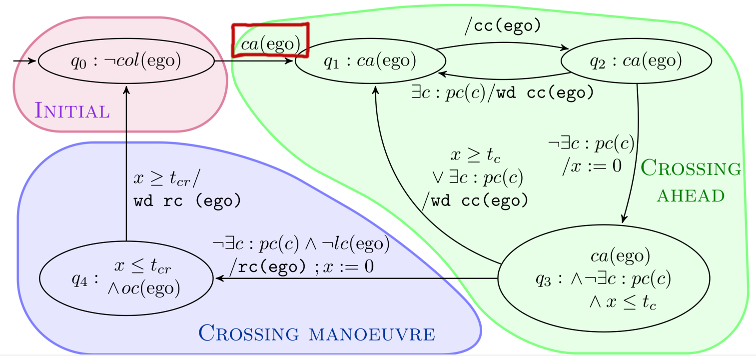
Crossing Controller [HS16, S18b]





Initial: Initial state guarantees collision freedom

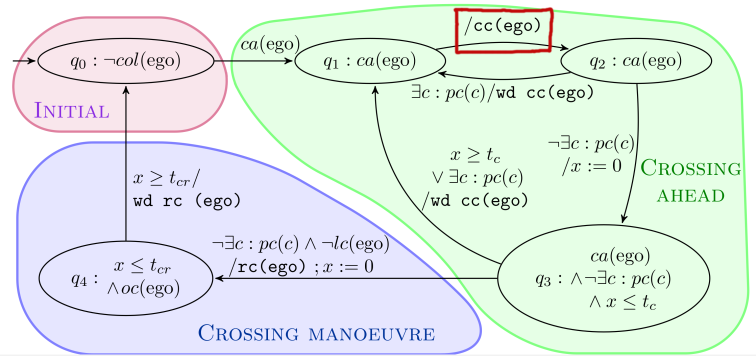
Crossing Controller [HS16, S18b]



Initial: Initial state guarantees collision freedom

Crossing ahead: If a crossing is ahead, *claim* crossing segments and check for potential collisions

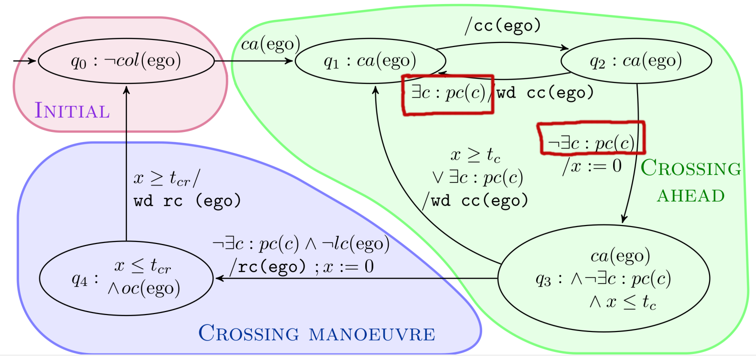
Crossing Controller [HS16, S18b]



Initial: Initial state guarantees collision freedom

Crossing ahead: If a crossing is ahead, *claim* crossing segments and check for potential collisions

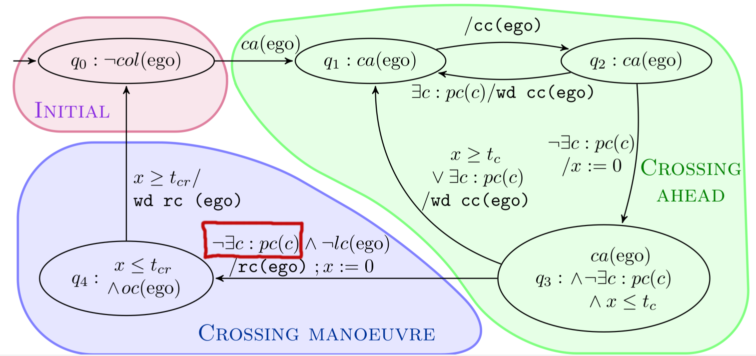
Crossing Controller [HS16, S18b]



Initial: Initial state guarantees collision freedom

Crossing ahead: If a crossing is ahead, *claim* crossing segments and check for potential collisions

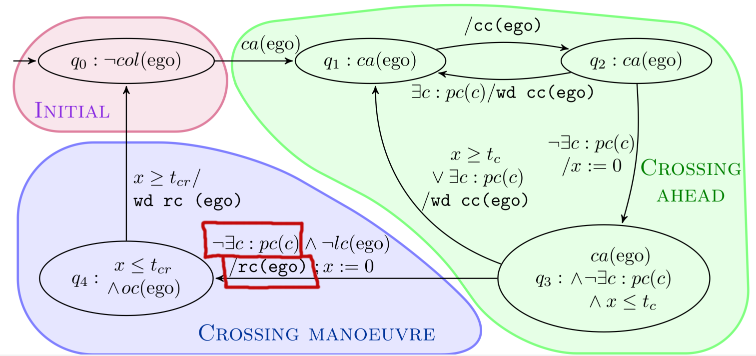
Crossing Controller [HS16, S18b]



Initial: Initial state guarantees collision freedom

Crossing ahead: If a crossing is ahead, *claim* crossing segments and check for potential collisions

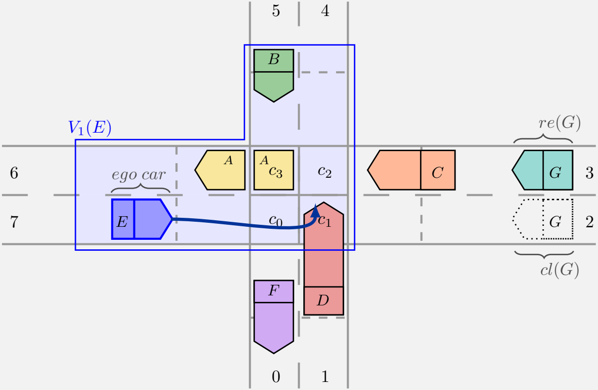
Crossing manoeuvre: If no potential collision detected, *reserve* crossing segments and enter intersection



Initial: Initial state guarantees collision freedom

Crossing ahead: If a crossing is ahead, *claim* crossing segments and check for potential collisions

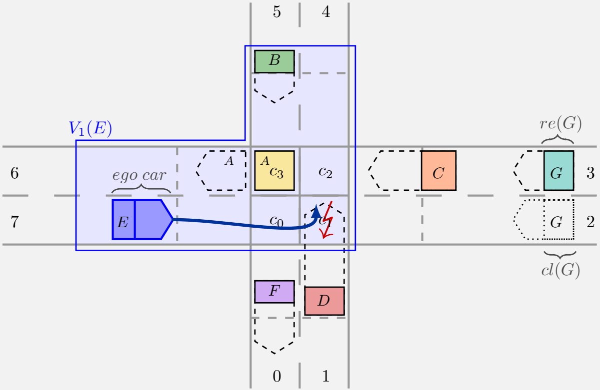
Crossing manoeuvre: If no potential collision detected, *reserve* crossing segments and enter intersection



Sor far:

All cars know physical size and braking distance of all cars

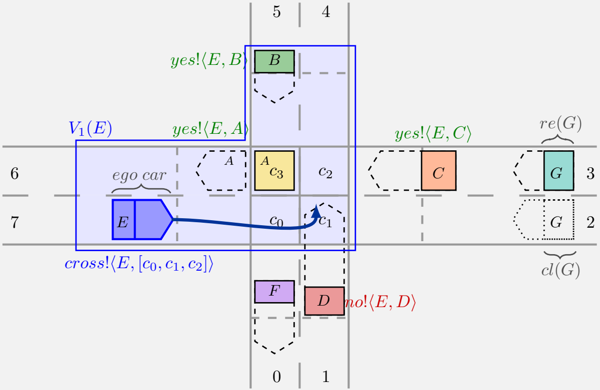
[S17:] Schwammberger, M.: Imperfect Knowledge in Autonomous Urban Traffic Manoeuvres (FVAV, 2017)



Now:

Imperfect Knowledge:
- Cars do not know braking distance of other cars
- Potential collision of *E* and *D* not visible!

[S17:] Schwammberger, M.: Imperfect Knowledge in Autonomous Urban Traffic Manoeuvres (FVAV, 2017)



Now:

Solution:
- Communicate with all cars on crossing or approaching crossing
- These cars are Helper Cars

[S17:] Schwammberger, M.: Imperfect Knowledge in Autonomous Urban Traffic Manoeuvres (FVAV, 2017)

Explainability of our Traffic Controllers

What should be explained:

- *Why* and *how* do our controllers do what they do?
- *What* can happen (good and bad things)?

On the way to explainability:

Analysability, *perhaps also understandability*

System analysis techniques:

- Testing or simulation
- Monitoring of system processes
- Model checking (Verify wether a model meets a specification)
- Verification (Assurance of correct behaviour)
- ...

Explainability of our Traffic Controllers

What should be explained:

- *Why* and *how* do our controllers do what they do?
- *What* can happen (good and bad things)?

On the way to explainability:

Analysability, *perhaps also understandability*

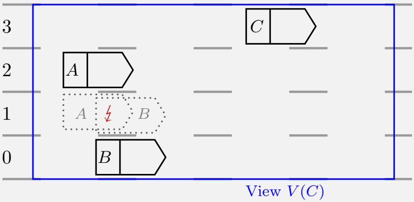
System analysis techniques:

- Testing or simulation
- Monitoring of system processes
- Model checking (Verify wether a model meets a specification)
- Verification (Assurance of correct behaviour)
- ...

Analysis of System Properties [S18a]

- Show for characteristic Abstract Model:
  - Spatial property: No Collision may ever occur (Safety)
  - Temporal property: All cars change lanes from time to time (Liveness)

1. Safety: Proof by hand (Induction over number of reachable traffic snapshots, via semantics of logic and controller)
2. Liveness: UPPAAL implementation of lane change controller
   - UPPAAL: Model Checking for timed automata
   - Also checked other properties



One input model for UPPAAL

[BDL04:] Behrmann, G., David, A. and Larsen, K.G.: A Tutorial on UPPAAL (2004)

Safety:

Any two cars may never collide.

Safety property (collision check) as UMLSL formula:

$$cc \equiv \neg\exists c\colon c \neq ego \wedge \langle re(ego) \wedge re(c) \rangle$$

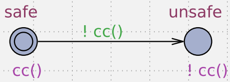
UPPAAL implementation:

- Collision check in UPPAAL code:

```
bool cc () {
   return not exists(c:carid_t)
         c != ego and intersect(res[ego],res[c]);
}
```

- Verification query: `A[] not Observer1.unsafe`

  ('On all paths holds globally that Observer1 is not in state 'unsafe'.')



Global Safety Observer automaton `Observer1`.

System properties: Liveness

Liveness:

Something good (e.g. a lane change) eventually happens.

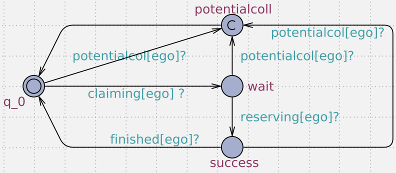
Liveness property as temporal logic style formula:

$$Live \equiv \forall c : \lozenge \left\langle \begin{array}{c} re(c) \\ re(c) \end{array} \right\rangle$$

(lane change)

UPPAAL implementation:

- Verification query: `A<> Observer(i).success`
  ('On all paths holds <u>finally</u> that Observer(i) is in state 'success'.')



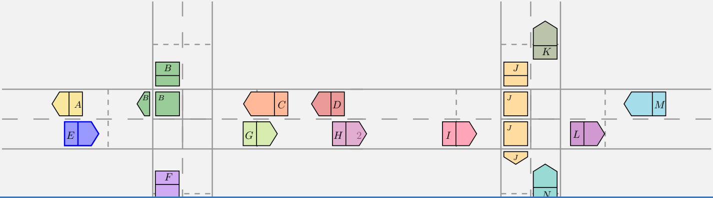
One Observer automaton `Observer(i)` *for each car i.*

Content Conclusion:

1. Urban Traffic:
    - Abstract Model, Logic UMLSL and Crossing Controller
    - Different concepts of knowledge
2. Proof of properties (safety and liveness)
3. Hazard Warning Case Study (not shown here)

Explainability:

- Analysability of system: Concise syntax and semantics
- Temporal properties: UPPAAL Model checking
- One component (controller) for each concern eases up explainability



Seminar expectations/ interests

Term 'Explainability' (area of autonomous traffic/ other):

- What should actually be explainable to whom?
- State of the art
- Standards? Guidelines?

Possible external expertise for me:

- Where else does explainability already exist in my approach?
- How can it be improved?
- What do other researchers explain in automotive domain?
- How are my abstraction and explainability combinable? How/ where do they profit from each other?

Literatur – Own work

- [HS16] Hilscher, M. and Schwammberger, M.: *An Abstract Model for Proving Safety of Autonomous Urban Traffic*. In A. Sampaio and F. Wang, editors: *Theoretical Aspects of Computing (ICTAC)*, volume 9965 of *LNCS* (October 2016).
- [OS17] Olderog, E.-R. and Schwammberger, M.: *Formalising a Hazard Warning Communication Protocol with Timed Automata*. *Models, Algorithms, Logics and Tools – Essays Dedicated to Kim Guldstrand Larsen on the Occasion of His 60th Birthday*, volume 10460 of *LNCS* (August 2017).
- [S17] Schwammberger, M.: *Imperfect Knowledge in Autonomous Urban Traffic Manoeuvres*. In Bulwahn, L. and Kamali, M. and Linker, S., editors: *Proceedings of First Workshop on Formal Verification of Autonomous Vehicles*, volume 257 of *EPTCS* (September 2017).
- [$S18_1$] Schwammberger, M.: *Introducing Liveness into Multi-lane Spatial Logic lane change controllers using UPPAAL*. In Gleirscher, M., Kugele, S. and Linker, S., editors: *Proceedings 2nd International Workshop on Safe Control of Autonomous Vehicles*, volume 269 of *EPTCS* (April 2018).
- [$S18_2$] Schwammberger, M.: *An abstract model for proving safety of autonomous urban traffic*. In volume 744 of *Theoretical Computing Science* (August 2018).

Literatur – Further MLSL Papers

- [FHO15] Fränzle, M. and Hansen, M.R. and Ody, H.: *No Need Knowing Numerous Neighbours*. In Meyer, R. and Platzer, A. and Wehrheim, H., editors: *Correct System Design Symposium*, volume 9360 of *LNCS* (2015).
- [HLOR11] Hilscher, M., Linker, S., Olderog, E.-R. and Ravn, A.P.: *An Abstract Model for Proving Safety of Multi-Lane Traffic Manoeuvres*. In Qin, S. and Qiu, Z., editors: *Proceedings of the 13th international conference on Formal methods and software engineering*, volume 6991 of *LNCS* (2011).
- [HLO13] Hilscher, M., Linker, S. and Olderog, E.-R.: *Proving Safety of Traffic Manoeuvres on Country Roads*. In Liu, Z., Woodcock J., and Zhu, H., editors: *Theories of Programming and Formal Methods – Essays Dedicated to Jifeng He on the Occasion of His 70th Birthday*, volume 8051 of *LNCS* (2013).
- [L15] Linker, S.: *Proofs for Traffic Safety – Combining Diagrams and Logic*. PhD thesis (2015).
- [$L17_1$] Linker, S.: *Spatial Reasoning About Motorway Traffic Safety with Isabelle/HOL*. In Polikarpova, N. and Schneider, S., editors: *Integrated Formal Methods*, volume 10510 of *LNCS* (2017).
- [$L17_2$] Linker, S.: *Hybrid Multi-Lane Spatial Logic*. In: *Archive of Formal Proofs* (2017).
- [O15] Ody, H.: *Undecidability Results for Multi-Lane Spatial Logic*. In Leucker, M., Rueda, C. and Valencia, F.D., editors: *International Conference on Theoretical Aspects of Computing – ICTAC*, volume 9399 of *LNCS* (2015).
- [O17] Ody, H.: *Monitoring of Traffic Manoeuvres with Imprecise Information*. In Bulwahn, L. and Kamali, M. and Linker, S., editors: *Proceedings of First Workshop on Formal Verification of Autonomous Vehicles*, volume 257 of *EPTCS* (2017).
- [O18] Olderog, E.-R.: *Space for Traffic Manoeuvres: An Overview*. In Jones, C. and Wang, J. and Zhan, N., editors: *Symposium on Real-Time and Hybrid Systems*, Springer (2018).

Literatur – Related work

- [AD94] Alur, R. and Dill, D.L.: *A Theory of Timed Automata*. In Nivat, M., editor: *Theoretical Computer Science*, volume 126 (1994).
- [BDL04] Behrmann, G., David, A. and Larsen, K.G.: *A Tutorial on UPPAAL*. In Bernardo, M., Corradini, F., editors: *Formal Methods for the Design of Real-Time Systems: 4th International School on Formal Methods for the Design of Computer, Communication and Software Systems: Real Time*, volume 3185 of *LNCS* (2004).
- [DHO06] Damm, W., Hungar, H. and Olderog, E.R.: *Verification of Cooperating Traffic Agents*. In: *International Journal of Control*, Taylor and Francis (2006).
- [M85] Moszkowski, B.: *A Temporal Logic for Multilevel Reasoning About Hardware*. In: *Computer*, IEEE (1985).
- [MRY02] Moor, T., Raisch, J., O'Young, S.: *Discrete Supervisory Control of Hybrid Systems Based on l-Complete Approximations*. In: *Discrete Event Dynamic Systems* (2002).
- [TWR10] Toben, T., Westphal, B. and Rakow, J.H.: *Spotlight Abstraction of Agents and Areas*. In: *Quantitative and Qualitative Analysis of Network Protocols*, Dagstuhl (2010).

Literatur – Related ITS Approaches

- [CEJ+12] Chan, E. et al.: *SAfe Road TRains for the Environment (SARTRE): Project final report* (2012).
- [DH01] Damm, W. and Harel, D.: *LSCs: Breathing Life into Message Sequence Charts*. In: *Formal Methods in System Design* (2001).
- [DMPR18] Damm W., Möhlmann, M., Peikenkamp, T. and Rakow, A.: *A Formal Semantics for Traffic Sequence Charts*. In: *Principles of Modeling - Essays Dedicated to Edward A. Lee on the Occasion of His 60th Birthday* (2018).
- [LAB+11] Montemerlo, M. et al.: *Junior: The Stanford Entry in the Urban Challenge*. In: *J. Field Robot.* (2011).
- [LGS98] Lygeros, J. and Godbole, D.N. and Sastry, S.S.: *Verified hybrid controllers for automated vehicles*. In: *IEEE Transactions on Automatic Control* (1998).
- [LP11] Loos, S.M. and Platzer, A.: *Safe Intersections: At the Crossing of Hybrid Systems and Verification*. In: *Intelligent Transportation Systems (ITSC)* (2011).
- [WGJG08] Werling, M., Gindele, T., Jagszent, D. and Groll, L.: *A robust algorithm for handling moving traffic in urban scenarios*. In: *2008 IEEE Intelligent Vehicles Symposium* (2008).
- [XL16] Xu, B. and Li, Q.: *A Spatial Logic for Modeling and Verification of Collision-Free Control of Vehicles*. In: *21st International Conference on Engineering of Complex Computer Systems (ICECCS)* (2016).
- [XL17] Xu, B. and Li, Q.: *A bounded multi-dimensional modal logic for autonomous cars based on local traffic and estimation*. In: *International Symposium on Theoretical Aspects of Software Engineering (TASE)* (2017).

Related Work – Intelligent Transportation Systems

Urban Traffic:

- DARPA Grand Challenge 2007 Candidates
  - Junior (2nd place, Stanford) [LAB+11]automatic verification whether
  - AnnyWAY (finalist, Berlin) [WGJG08]
  - Algorithms only apply for specific DARPA roadmap
- Loos, Platzer [LP11]:
  - Centralised scheduling at intersections of single lanes, one car per lane
  - Verification with KEYmaera
- Xu, Li [XL16, XL17]:
  - Space-grid model for reasoning about urban traffic

Other traffic scenarios:

- Damm et al: Traffic Sequence Charts [DMPR18]:
  - Visual specification language based on LSCs [DH01]
  - Specification of dynamic evolution of traffic
- Platooning approaches
  - California Path Project [LGS98]
  - European SARTRE Project [CEJ+12]

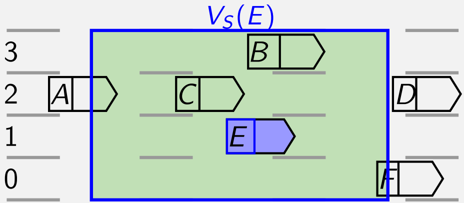
Related Work – The MLSL Approach

- Overall Goal:
  - Autonomous car manoeuvres
  - Use formal methods to certify safety of these manoeuvres
- The MLSL Approach:
  - Spatial logic *Multi-lane Spatial Logic* (MLSL) to reason about traffic situations
  - Controllers using MLSL to undertake safe traffic manoeuvres (e.g. lane-change)
- Existing works (Overview: [O18]):

	Basic Cases	Extensions	Implementations
Highway Traffic	[HLOR11] [L15], [O15]	[FHO15], [O17] [OS17]	[L17a,L17b] [S18a]
Country Roads	[HLO13]		
Urban Traffic	[HS16, S18b]	[S17]	

Basics: Spotlight principle

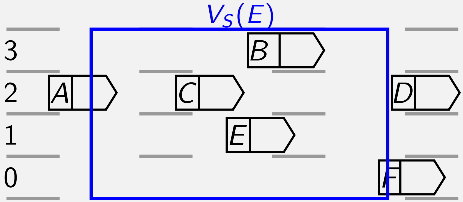
- Semantics of MLSL formulas:
  Evaluated only in view, not in complete traffic snapshot
  - Consider only surroundings of view owner (here: E)
- Standard View $V_S(E)$:
  Look ahead and back up to a horizon h, include all lanes
- E.g. collision check formula:
  $\neg col(ego) \equiv \neg\exists c\colon c \neq \mathrm{ego} \wedge \langle re(c) \wedge re(ego)\rangle$
- Formula satisfied: $\mathcal{TS}, V_S(E), \nu \models \neg col(ego)$



[TWR10]: Toben T., Westphal B., Rakow, Jan-Hendrik: *Spotlight Abstraction of Agents and Areas* (2010)

Lane-change controller protocol ($\nu(ego) = E$):

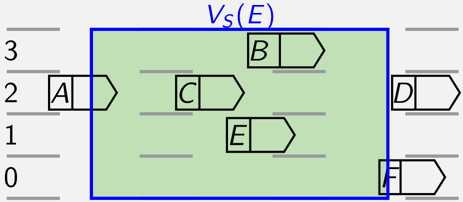
1. Initial traffic situation is safe (i.e.: no collisions):
   $\neg col(ego) \equiv \neg\exists c : c \neq \mathrm{ego} \wedge \langle re(c) \wedge re(ego) \rangle$
2. Set turn signal (set claim on a neighbouring lane)
3. Check for potential collisions:
   $pc(c) \equiv c \neq \mathrm{ego} \wedge \langle cl(\mathrm{ego}) \wedge (re(c) \vee cl(c)) \rangle$
4. If no potential collisions, change lane (change claim into reservation)
5. Finished (drive on new lane)



Basic Case: Highway Traffic [HLOR11]

Lane-change controller protocol ($\nu(ego) = E$):

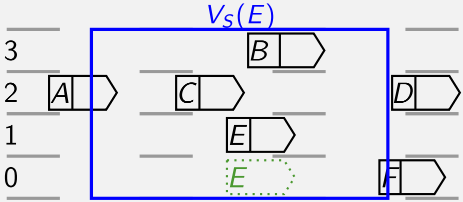
1. Initial traffic situation is safe (i.e.: no collisions):
   $\neg col(ego) \equiv \neg\exists c : c \neq \mathrm{ego} \wedge \langle re(c) \wedge re(ego) \rangle$
2. Set turn signal (set claim on a neighbouring lane)
3. Check for potential collisions:
   $pc(c) \equiv c \neq \mathrm{ego} \wedge \langle cl(\mathrm{ego}) \wedge (re(c) \vee cl(c)) \rangle$
4. If no potential collisions, change lane (change claim into reservation)
5. Finished (drive on new lane)



Basic Case: Highway Traffic [HLOR11]

Lane-change controller protocol ($\nu(ego) = E$):

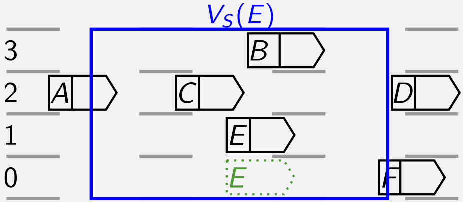
1. Initial traffic situation is safe (i.e.: no collisions):
   $\neg col(ego) \equiv \neg\exists c : c \neq \mathrm{ego} \wedge \langle re(c) \wedge re(ego)\rangle$
2. Set turn signal (set claim on new lane)
3. Check for potential collisions:
   $pc(c) \equiv c \neq \mathrm{ego} \wedge \langle cl(\mathrm{ego}) \wedge (re(c) \vee cl(c))\rangle$
4. If no potential collisions, change lane (change claim into reservation)
5. Finished (drive on new lane)



Basic Case: Highway Traffic [HLOR11]

Lane-change controller protocol ($\nu(ego) = E$):

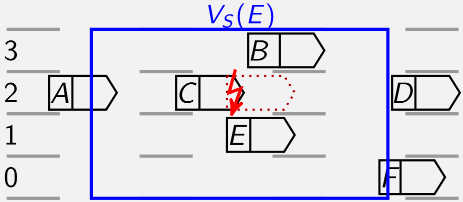
1. Initial traffic situation is safe (i.e.: no collisions):
   $\neg col(ego) \equiv \neg\exists c : c \neq \text{ego} \land \langle re(c) \land re(ego)\rangle$
2. Set turn signal (set claim on a neighbouring lane)
3. Check for potential collisions:
   $pc(c) \equiv c \neq \text{ego} \land \langle cl(\text{ego}) \land (re(c) \lor cl(c))\rangle$
4. If no potential collisions, change lane (change claim into reservation)
5. Finished (drive on new lane)



Basic Case: Highway Traffic [HLOR11]

Lane-change controller protocol ($\nu(ego) = E$):

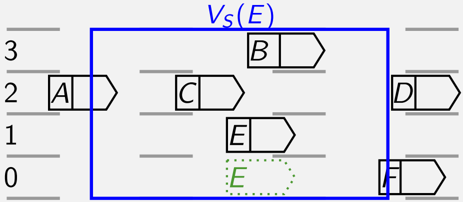
1. Initial traffic situation is safe (i.e.: no collisions):
   $\neg col(ego) \equiv \neg\exists c : c \neq \mathrm{ego} \wedge \langle re(c) \wedge re(ego)\rangle$
2. Set turn signal (set claim on a neighbouring lane)
3. Check for potential collisions:
   $pc(c) \equiv c \neq \mathrm{ego} \wedge \langle cl(\mathrm{ego}) \wedge (re(c) \vee cl(c))\rangle$
4. If no potential collisions, change lane (change claim into reservation)
5. Finished (drive on new lane)



Basic Case: Highway Traffic [HLOR11]

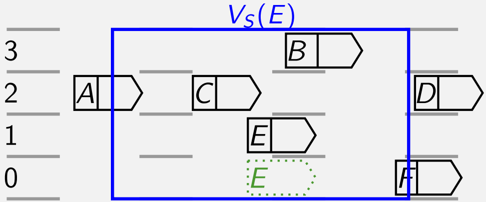
Lane-change controller protocol ($\nu(ego) = E$):

1. Initial traffic situation is safe (i.e.: no collisions):
   $\neg col(ego) \equiv \neg\exists c : c \neq \mathrm{ego} \land \langle re(c) \land re(ego) \rangle$
2. Set turn signal (set claim on a neighbouring lane)
3. Check for potential collisions:
   $pc(c) \equiv c \neq \mathrm{ego} \land \langle cl(\mathrm{ego}) \land (re(c) \lor cl(c)) \rangle$
4. If no potential collisions, change lane (change claim into reservation)
5. Finished (drive on new lane)



Lane-change controller protocol ($\nu(ego) = E$):

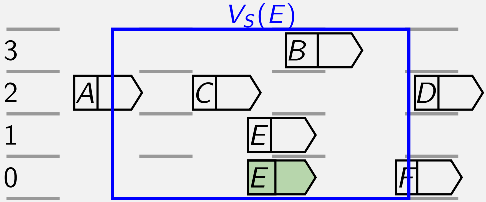
1. Initial traffic situation is safe (i.e.: no collisions):
   $\neg col(ego) \equiv \neg\exists c : c \neq \mathrm{ego} \wedge \langle re(c) \wedge re(ego) \rangle$
2. Set turn signal (set claim on a neighbouring lane)
3. Check for potential collisions:
   $pc(c) \equiv c \neq \mathrm{ego} \wedge \langle cl(\mathrm{ego}) \wedge (re(c) \vee cl(c)) \rangle$
4. If no potential collisions, change lane (change claim into reservation)
5. Finished (drive on new lane)



Basic Case: Highway Traffic [HLOR11]

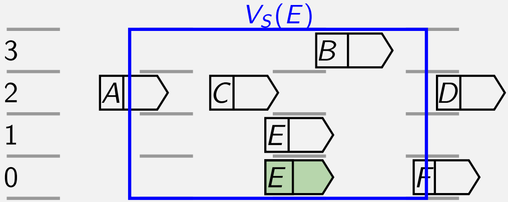
Lane-change controller protocol ($\nu(ego) = E$):

1. Initial traffic situation is safe (i.e.: no collisions):
   $\neg col(ego) \equiv \neg\exists c : c \neq \mathrm{ego} \wedge \langle re(c) \wedge re(ego) \rangle$
2. Set turn signal (set claim on a neighbouring lane)
3. Check for potential collisions:
   $pc(c) \equiv c \neq \mathrm{ego} \wedge \langle cl(\mathrm{ego}) \wedge (re(c) \vee cl(c)) \rangle$
4. If no potential collisions, change lane (change claim into reservation)
5. Finished (drive on new lane)



Lane-change controller protocol ($\nu(ego) = E$):

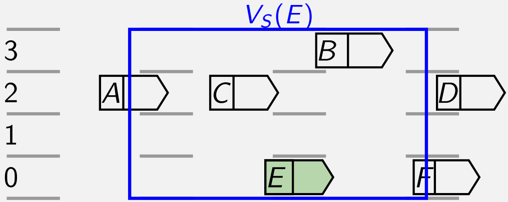
1. Initial traffic situation is safe (i.e.: no collisions):
   $\neg col(ego) \equiv \neg\exists c\colon c \neq \mathrm{ego} \wedge \langle re(c) \wedge re(ego) \rangle$
2. Set turn signal (set claim on a neighbouring lane)
3. Check for potential collisions:
   $pc(c) \equiv c \neq \mathrm{ego} \wedge \langle cl(\mathrm{ego}) \wedge (re(c) \vee cl(c)) \rangle$
4. If no potential collisions, change lane (change claim into reservation)
5. Finished (drive on new lane)



Basic Case: Highway Traffic [HLOR11]

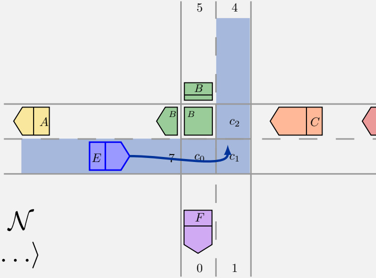
Lane-change controller protocol ($\nu(ego) = E$):

1. Initial traffic situation is safe (i.e.: no collisions):
   $\neg col(ego) \equiv \neg\exists c : c \neq \mathrm{ego} \wedge \langle re(c) \wedge re(ego) \rangle$
2. Set turn signal (set claim on a neighbouring lane)
3. Check for potential collisions:
   $pc(c) \equiv c \neq \mathrm{ego} \wedge \langle cl(\mathrm{ego}) \wedge (re(c) \vee cl(c)) \rangle$
4. If no potential collisions, change lane (change claim into reservation)
5. Finished (drive on new lane)

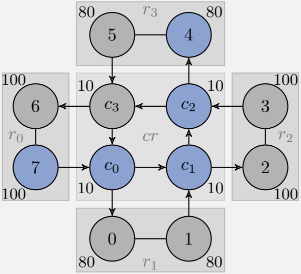


Urban Road Network [HS16, S18b]

- Urban road network $\mathcal{N} = (\mathcal{V}, E_u, E_d, \omega)$:
  - Nodes from set $\mathcal{V} = \mathbb{CS} \cup \mathbb{L}$
  - Directed edges $E_d \in (\mathcal{V} \times \mathcal{V}) \backslash (\mathbb{L} \times \mathbb{L})$
  - Undirected edges $E_u \in (\mathbb{L} \times \mathbb{L})$
  - Real weight $\omega(\nu)$ of nodes $\nu \in \mathcal{V}$
- Path of cars *pth*
  - Cars follow infinite path $pth\colon \mathbb{Z} \to \mathcal{V}$ in $\mathcal{N}$
  - Example: $pth(E) = \langle \ldots, 7, c_0, c_1, c_2, 4, \ldots \rangle$

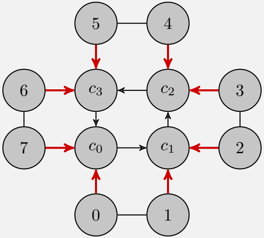


$\mathcal{N}$:



Topological Sanity Conditions [S18b]

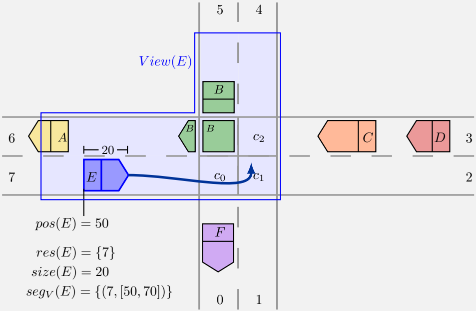
- Idea:
  Exclude road networks that are pointless
  - E.g. intersections without outgoing edges (avoid dead- or lifelocks)



- Sanity Condition 1:
  Each node $\nu : \mathcal{V}$ has a predecessor and a successor

- Sanity Condition 2:
  Before and after an intersection, there is a road

Abstract Representation of Cars [S18b]



Positions and size of car B from viewpoint of car E:

- Highway traffic [HLOR11]: One position on one lane
- Now: Collection of segments a car occupies
- Abstract sensor function: $size(B) = \Omega_E(B)$
- E.g. segments of car B:

$$seg_V(B) = \{(5, [70, 80]), (c_3, [0, 10]), (6, [0, 5])\}$$

[E01:] W. Elmenreich: An Introduction to sensor fusion (2001)
[K99:] L.A. Klein: Sensor and Data Fusion Concepts and Applications (1999)

- Syntax:

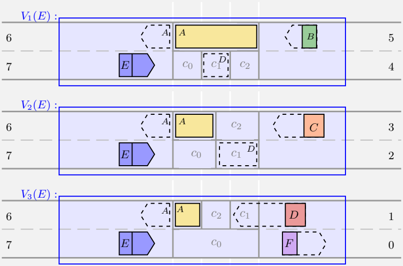
$$\phi ::= true \mid u = v \mid free \mid cs \mid re(c) \mid cl(c) \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \exists c : \phi_1 \mid \phi_1 \frown \phi_2 \mid \begin{array}{c}\phi_2\\ \phi_1\end{array}$$

  - c: Car variable or special constant *ego* for ego car E under consideration
  - u, v: car variables or real variables
  - Cars claim ($cl(c)$) or reserve ($re(c)$) space
  - Special atoms: *free* (free space) and *cs* (crossing segment)
  - Horizontal chop operator $\frown$ from interval temporal logic [M85]

- Semantics: Satisfaction of UMLSL formulae is defined wrt ...
  - ... a Traffic Snapshot $\mathcal{TS}$,
  - ... a View $V(E) = (L, X, E)$ and
  - ... a valuation ν of variables.

- Abbreviation: $\langle\phi\rangle$ for „ϕ *holds somewhere in* $V(E)$“

Logical reasoning with Urban Multi-lane Spatial Logic



- Potential collision: $pc(c) \equiv c \neq \mathrm{ego} \wedge \langle cl(\mathrm{ego}) \wedge (re(c) \vee cl(c))\rangle$
- Crossing ahead: $ca(\mathrm{ego}) \equiv \langle re(\mathrm{ego}) \frown free^{<d_c} \wedge \neg\langle cs\rangle \frown cs\rangle$
- Potential helper: $ph(c) \equiv c \neq \mathrm{ego} \wedge \langle (oc(c) \vee ocac(c)) \wedge \neg lc(c)\rangle$
- Further formulas:
  - Collision check $col(\mathrm{ego})$,
  - On crossing $oc(c)$
  - Crossing ahead for opposing car $ocac(c)$
  - Active lane change manoeuvre $lc(c)$

Automotive-Controlling Timed Automata [S14, HS16]

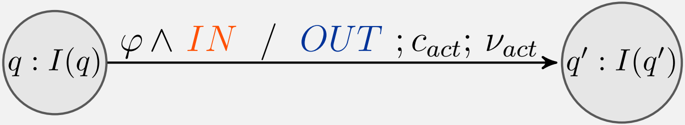
- Extended timed automata [AD94]
- UMLSL-formulae as guards φ and invariants $I(q)$
  - Potential collision check: $\exists c : pc(c)$
  - Crossing ahead: $ca(\mathrm{ego})$
- Controller actions c_{act} for lane change and crossing manoeuvres
  - **c**laim **c**rossing segments: $\mathtt{cc}(\mathrm{ego})$
  - **r**eserve **c**rossing segments: `rc()`
- Clock and data updates ν_{act} (cf. $x := 0$)

$$q : I(q) \xrightarrow{\varphi \;/\; c_{act} \;;\; \nu_{act}} q' : I(q')$$

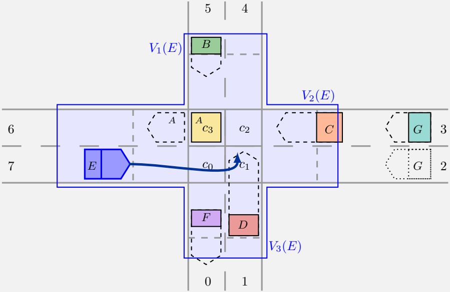
[AD94]: Alur, R., Dill, D.L.: *A Theory of Timed Automata*, TCS (1994)
[S14]: Schwammberger M.: *Semantik von Controllern für sicheren Fahrspurwechsel*, masters' thesis (2014)

ACTA with Communication [S14, S17, OS17]

- Broadcast communication with data constraints:
  - Output action OUT: $cross!\langle E, [c_0, c_1, c_2]\rangle$
  - Input action IN: $cross?\langle c, cs\rangle : c \neq h$ (for helper with id variable h)



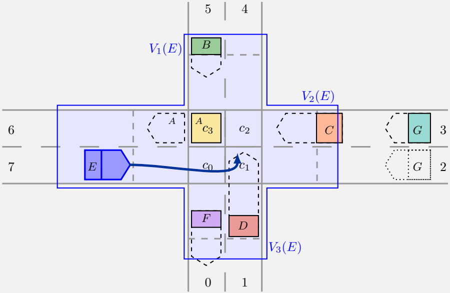
[HS17]: Olderog, E.R., S.M.: *A Hazard Warning Communication Protocol with Timed Automata* (2017)
[S14]: Schwammberger M.: *Semantik von Controllern für sicheren Fahrspurwechsel*, masters' thesis (2014)



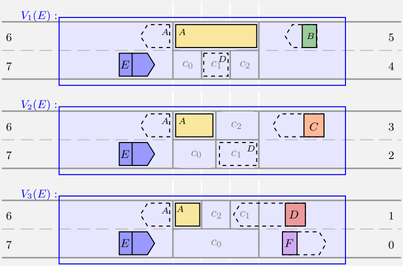
Virtual Communication Multi-View:

- Communicate with all cars on crossing or approaching crossing
- Problem: Cross-shaped view does not allow for reasoning with MLSL
- Solution: Build three straight virtual views $V_i(E)$
  - $V_1(E)$: Look left,
  - $V_2(E)$: Look ahead,
  - $V_3(E)$: Look right.

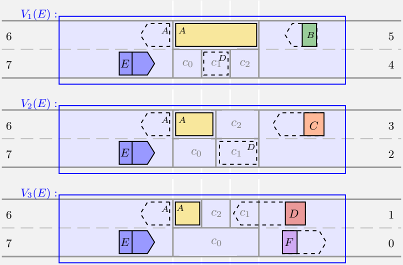
[S17:] Schwammberger, M.: Imperfect Knowledge in Autonomous Urban Traffic Manoeuvres (FVAV, 2017)



Straight virtual views $V_i(E)$:



Potential Helper Cars



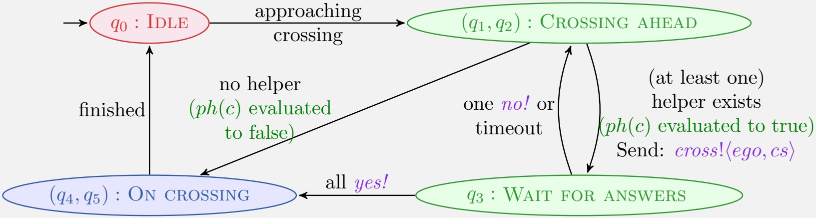
Abbreviations:

- Cars **on** **c**rossing: $oc(c) \equiv \langle re(c) \wedge cs \rangle$
- **O**pposing **c**ar **a**pproaching the **c**rossing: $ocac(c)$
  (More or less a reversed crossing ahead check)
- Summary: Potential helper check:

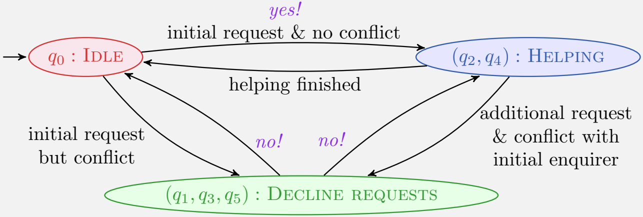
$$ph(c) \equiv c \neq \mathrm{ego} \wedge \langle (oc(c) \vee ocac(c)) \wedge \neg lc(c) \rangle$$

Crossing and Helper Controller

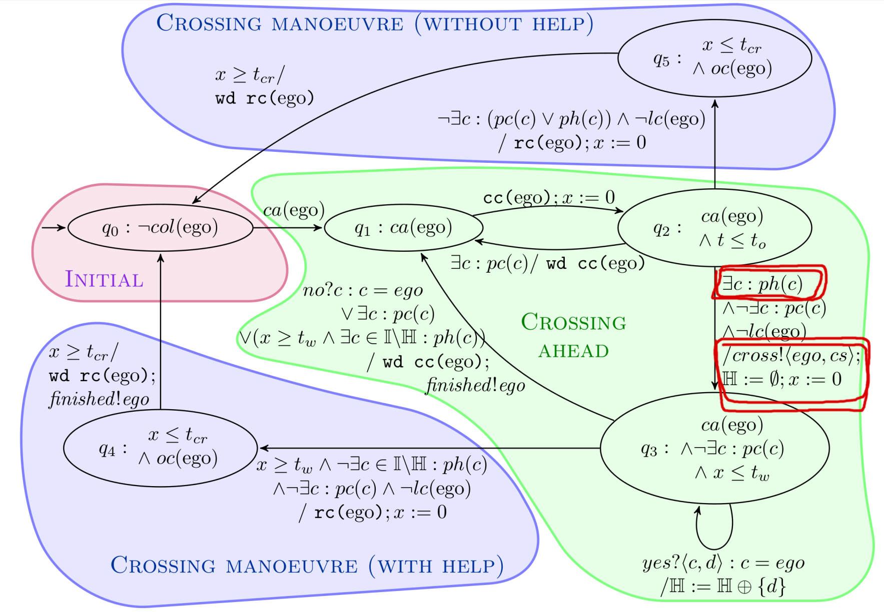
Crossing controller protocol:



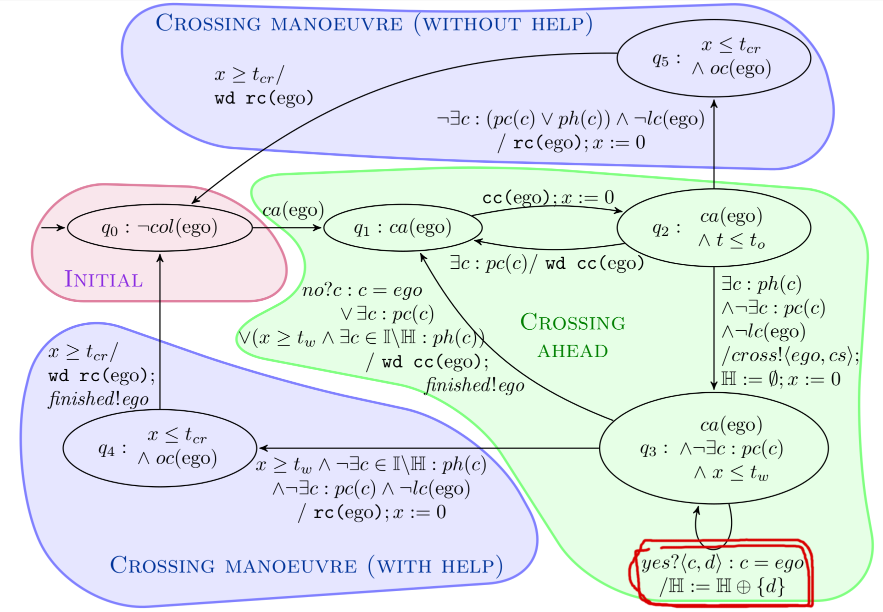
Helper controller protocol:



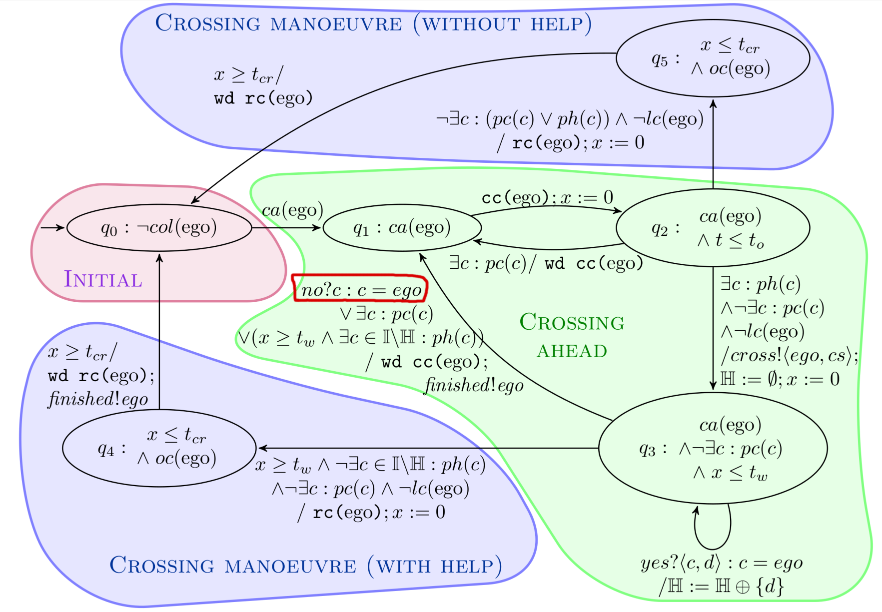
Crossing Controller with Communication



Crossing Controller with Communication



Crossing Controller with Communication



Safety proof for Crossing Controller [HS16, S18b]

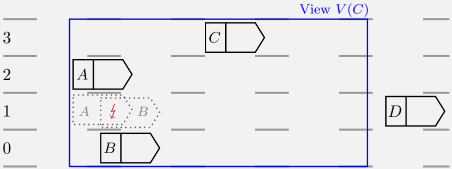
- Safety property:

$$Safe \equiv \forall c, d : c \neq d \rightarrow \neg\langle re(c) \wedge re(d)\rangle$$

- Assumptions:
  1. The initial traffic snapshot $\mathcal{TS}_0$ is safe.
  2. Every car is equipped with each a Distance Controller, Road Controller and Crossing Controller

- Proof Outline:
  - Prove safety from perspective of an arbitrary car E (all cars behave similarly, spotlight principle)
  - Prove that all traffic snapshots $\mathcal{TS}$ reachable from $\mathcal{TS}_0$ are safe
  - Proof over semantics of logic and controller
  - Proof by induction over number of traffic snapshots needed to reach a traffic snapshot from $\mathcal{TS}_0$

UPPAAL Implementation: Abstract Model and Logic



- Data Structure for Abstract Model:

```
pos_t res[carid_t]={ { {0,0,1,0}, 10, 5},
                     { {1,0,0,0}, 12, 5},
                     { {0,0,0,1}, 40, 5}};
```

- Formulas of Multi-lane Spatial Logic (potential collision check):

```
bool pc (carid_t c) {
   return c != ego
      and (intersect(clm[ego],res[c])
         or intersect(clm[ego],clm[c]));
}
```

Implementation: Setting Reservations and Claims

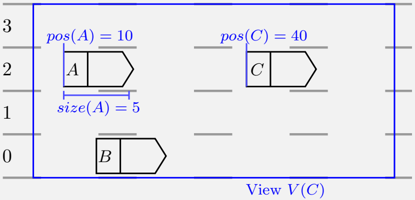
- Setting a claim:

```
void claim(laneid_t lane) {
    clm[ego].lane[lane] = true;
}
```

- Transform existing claim into a reservation:

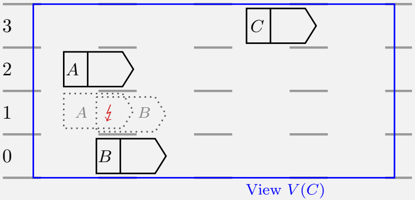
```
void reservation() {
    for (i:laneid_t)
    {
        if (clm[ego].lane[i]) {
            res[ego].lane[i] = true;
            clm[ego].lane[i] = false;
        }
    }
}
```

Implementation: MLSL formulas



- Function `intersect` to detect intersections of safety envelopes:

```
bool intersect(const pos_t p1, const pos_t p2) {
   return exists(lane:  laneid_t)
      p1.lane[lane] and p2.lane[lane]
         and not (p1.pos > p2.pos+p2.size
                 or p2.pos > p1.pos+p1.size);
}
```



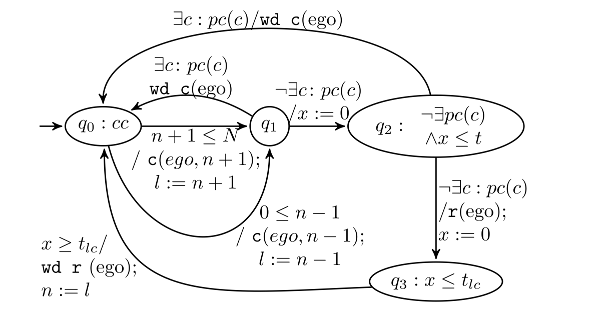
- MLSL formula potential collision:
  $pc(c) \equiv c \neq ego \wedge \langle cl(ego) \wedge (re(c) \vee cl(c)) \rangle$

- Formula in UPPAAL:

```
bool pc (carid_t c) {
   return c != ego
      and (intersect(clm[ego],res[c])
         or intersect(clm[ego],clm[c]));
}
```

Liveness issues with controller from [HLOR11]

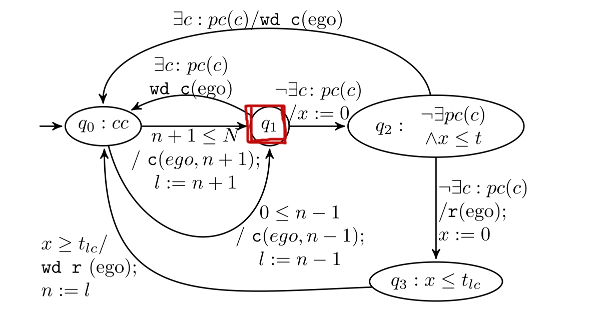
- Liveness issue 1: No clock invariant at state q_1:
  - System is allowed to stay in q_1 forever
- Liveness issue 2: No clock guards on outgoing edges of q_1:
  - Each system is allowed to alternately claim and withdraw claims infinitely often in 0 time (livelock)
  - No other system can act in between



Lane change controller for highway traffic from [HLOR11].

Liveness issues with controller from [HLOR11]

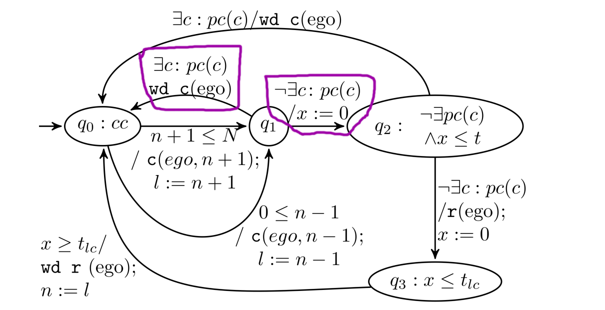
- Liveness issue 1: No clock invariant at state q_1:
  - System is allowed to stay in q_1 forever
- Liveness issue 2: No clock guards on outgoing edges of q_1:
  - Each system is allowed to alternately claim and withdraw claims infinitely often in 0 time (livelock)
  - No other system can act in between



Lane change controller for highway traffic from [HLOR11].

Liveness issues with controller from [HLOR11]

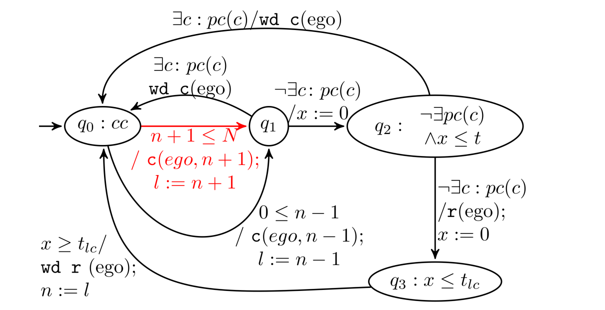
- Liveness issue 1: No clock invariant at state q_1:
  - System is allowed to stay in q_1 forever
- Liveness issue 2: No clock guards on outgoing edges of q_1:
  - Each system is allowed to alternately claim and withdraw claims infinitely often in 0 time (livelock)
  - No other system can act in between



Lane change controller for highway traffic from [HLOR11].

Liveness issues with controller from [HLOR11]

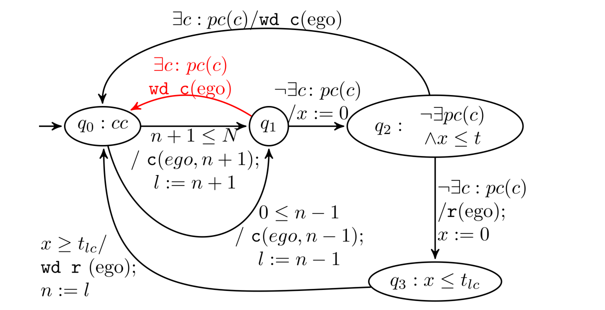
- Liveness issue 1: No clock invariant at state q_1:
  - System is allowed to stay in q_1 forever
- Liveness issue 2: No clock guards on outgoing edges of q_1:
  - Each system is allowed to alternately claim and withdraw claims infinitely often in 0 time (livelock)
  - No other system can act in between



Lane change controller for highway traffic from [HLOR11].

Liveness issues with controller from [HLOR11]

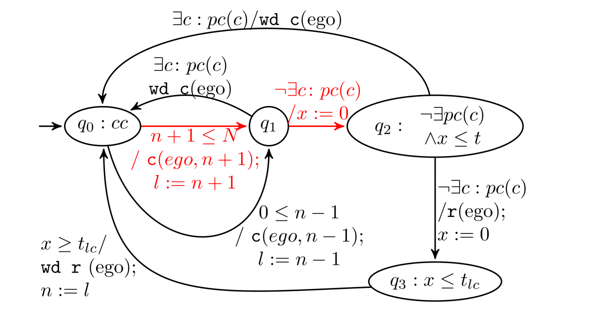
- Liveness issue 1: No clock invariant at state q_1:
  - System is allowed to stay in q_1 forever
- Liveness issue 2: No clock guards on outgoing edges of q_1:
  - Each system is allowed to alternately claim and withdraw claims infinitely often in 0 time (livelock)
  - No other system can act in between



Lane change controller for highway traffic from [HLOR11].

Liveness issues with controller from [HLOR11]

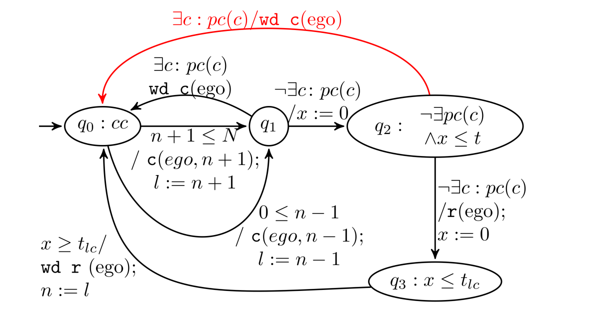
- Liveness issue 1: No clock invariant at state q_1:
  - System is allowed to stay in q_1 forever
- Liveness issue 2: No clock guards on outgoing edges of q_1:
  - Each system is allowed to alternately claim and withdraw claims infinitely often in 0 time (livelock)
  - No other system can act in between



Lane change controller for highway traffic from [HLOR11].

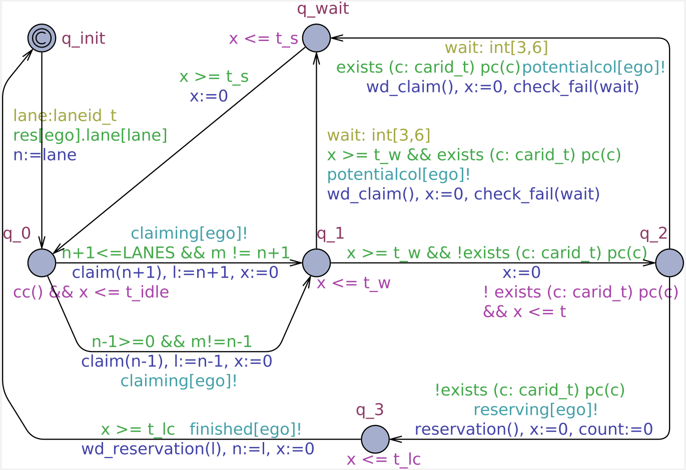
Liveness issues with controller from [HLOR11]

- Liveness issue 1: No clock invariant at state q_1:
  - System is allowed to stay in q_1 forever
- Liveness issue 2: No clock guards on outgoing edges of q_1:
  - Each system is allowed to alternately claim and withdraw claims infinitely often in 0 time (livelock)
  - No other system can act in between

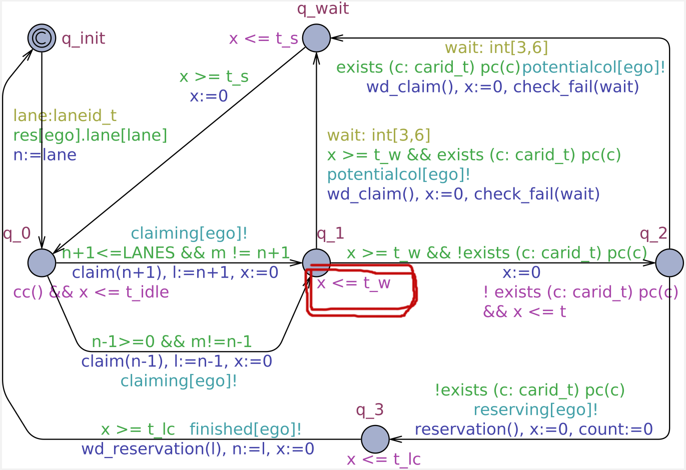


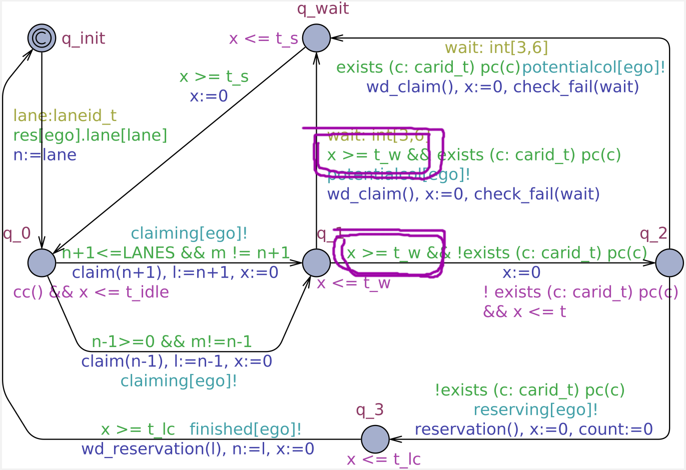
Lane change controller for highway traffic from [HLOR11].

UPPAAL Implementation: Revised Alive Controller [S18a]

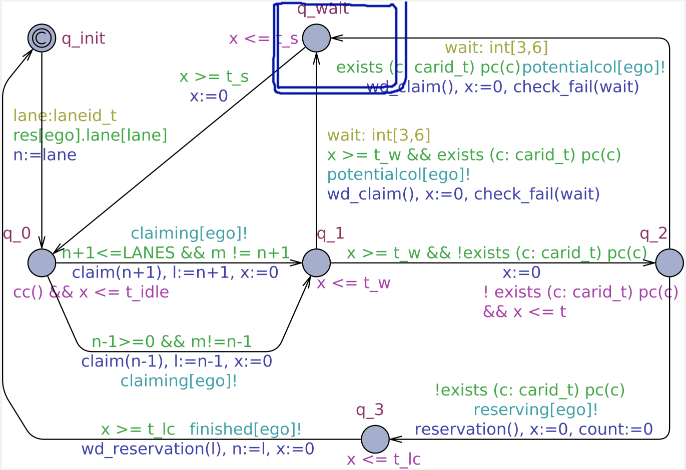


UPPAAL Implementation: Revised Alive Controller [S18a]



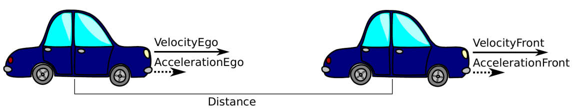

q_wait
q_init
x <= t_s
wait: int[3,6]
exists (c: carid_t) pc(c) potentialcol[ego]!
wd_claim(), x:=0, check_fail(wait)
x >= t_s
x:=0
lane:laneid_t
res[ego].lane[lane]
n:=lane
wait: int[3,6]
x >= t_w && exists (c: carid_t) pc(c)
potentialcol[ego]!
wd_claim(), x:=0, check_fail(wait)
q_0
claiming[ego]!
n+1<=LANES && m != n+1
claim(n+1), l:=n+1, x:=0
cc() && x <= t_idle
q_1
x <= t_w
x >= t_w && !exists (c: carid_t) pc(c)
x:=0
q_2
! exists (c: carid_t) pc(c)
&& x <= t
n-1>=0 && m!=n-1
claim(n-1), l:=n-1, x:=0
claiming[ego]!
!exists (c: carid_t) pc(c)
reserving[ego]!
reservation(), x:=0, count:=0
q_3
x >= t_lc finished[ego]!
wd_reservation(l), n:=l, x:=0
x <= t_lc

UPPAAL Implementation: Revised Alive Controller [S18a]



Outlook: Distance Controller in UPPAAL [S18a]

- Current assumption of constant speed
- Need Distance Controller for cars with different speed
- Existing UPPAAL Distance Controller [LMT15]:
  - From group of Kim Larsen, synthesised with UPPAAL Stratego
  - One ego car and one front car
  - Ego car always keeps sufficient distance to front car
- Problems with existing implementation:
  - Implemented only for abstract model with one single lane
  - Lane change is not considered/ possible
  - More lanes: More cars have to be considered
  - More cars: More parallel UPPAAL automata

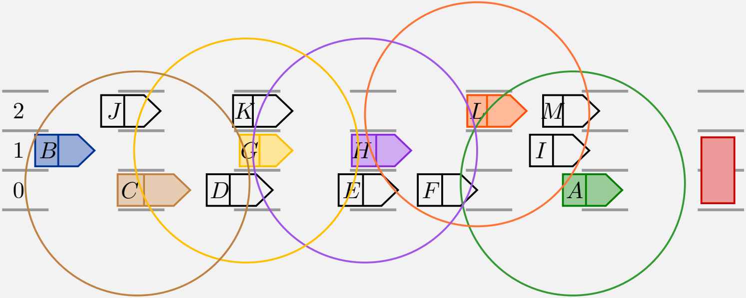


[LMT15:] Larsen K. G., Mikučionis, M. and Taankvist J. H.: Safe and Optimal Adaptive Cruise Control (CSD 2015)

Motivation: Hazard Warning Case Study [OS17]

Hazard Warning Protocol

- Correctly and timely transmit hazard warning to an approaching car
- Multi-hop communication chain
- MLSL Extension Hazard Warning Multi-lane Spatial logic (HMLSL)
- Initial hazard warning message: $\mathit{hazard}!\langle\{0,1\},\overrightarrow{c}\rangle$



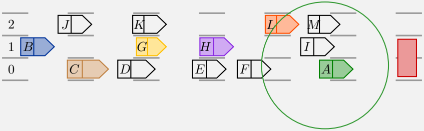
[OS17:] Olderog, E.R., Schwammberger, M.: *Formalising a Hazard Communication Protocol with Timed Automata* (Models, Algorithms, Logics and Tools, 2017)

First Controller:

- Hazard Detection Controller
  - Only active in car which detects hazard (here: car A)
  - Computes communication chain (here: $\overrightarrow{c} = [A, L, H, G, C, B]$)
  - Sends initial warning message to own forwarding controller

Second Controller:

- Forwarding Controller
  - Forwards warning
  - Forwarded parameters:
    - Affected lanes
    - Communication chain
  - Example for message sending: $hazard!\langle[0,1], \overrightarrow{c}\rangle$



Timely Warning and Spatial Hazard Safety [OS17]

Prove two aspects:

- Timing property: Whenever a hazard is detected by a car A, a distinct car B is warned within less than t time units, depending on the size of the communication chain $\overrightarrow{c}$.

  $\Rightarrow$ Proof outline: Proof by induction over number of cars in $\overrightarrow{c}$ by assistance of UPPAAL (verify properties of Observer automata)

- Spatial property: There never exists a traffic snapshpot, where the following property is violated for an arbitrary car:

  $$\mathit{Safe\text{-}hz}(\mathrm{ego}) \equiv \neg\langle \mathit{re}(\mathrm{ego}) \wedge \mathit{hz}\rangle$$

  $\Rightarrow$ Proof outline: Proof by induction over traffic snapshots