A multi-agent approach towards cooperative overtaking in vehicular networks

Adrian Groza, Bogdan Iancu, Anca Marginean

Intelligent Systems Group
Department of Computer Science
Technical University of Cluj-Napoca, Romania
anca.marginean@cs.utcluj.ro

June 3, 2014

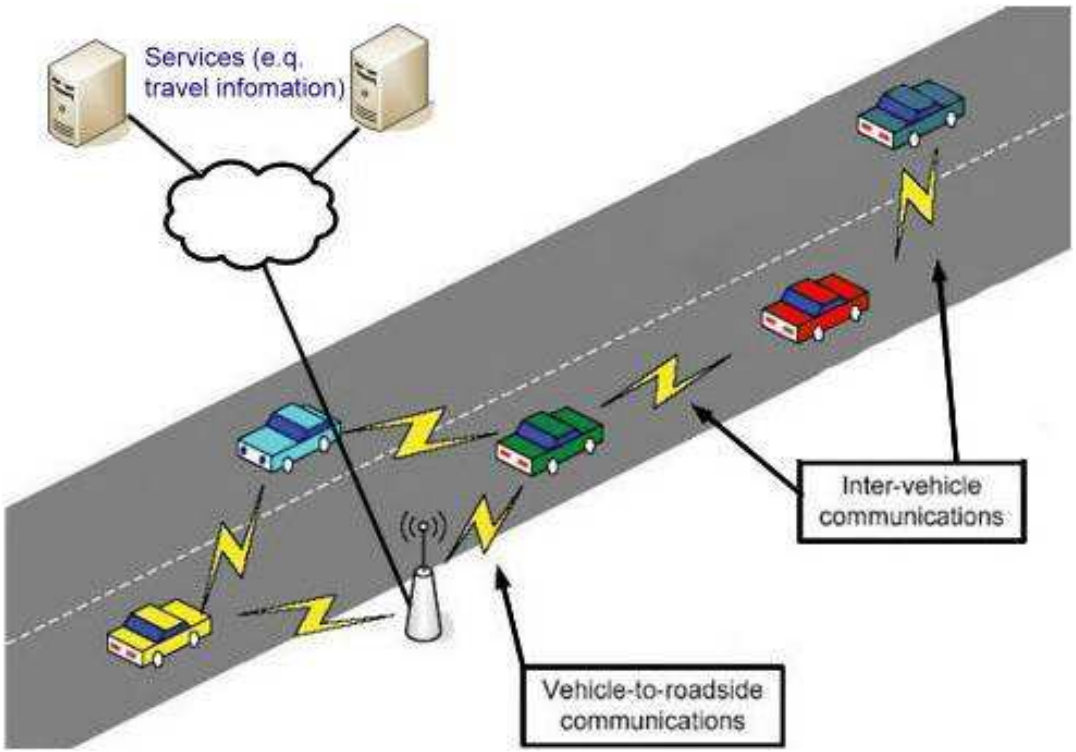
Outline

VANETs - Vehicular Ad-hoc Networks (VANETs)

- Application of mobile ad-hoc networks.
- Vehicles as mobile nodes able to self-configure
  - aims: creation of a communication network via wireless links.
- Communication Vehicle-2-X: vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I)
  - aims: cooperation among different vehicles on the road.

Vehicle-2-X communication

Vehicular communication standard: Wireless Access in Vehicular Environments (WAVE) or IEEE 802.11p



Geocast ad hoc routing protocol

- beaconing: continuously and periodically broadcast information to all neighbours in the reception range
- location service
- forwarding: GeoUnicast, GeoBroadcast

Multi-agent and Car-2-X

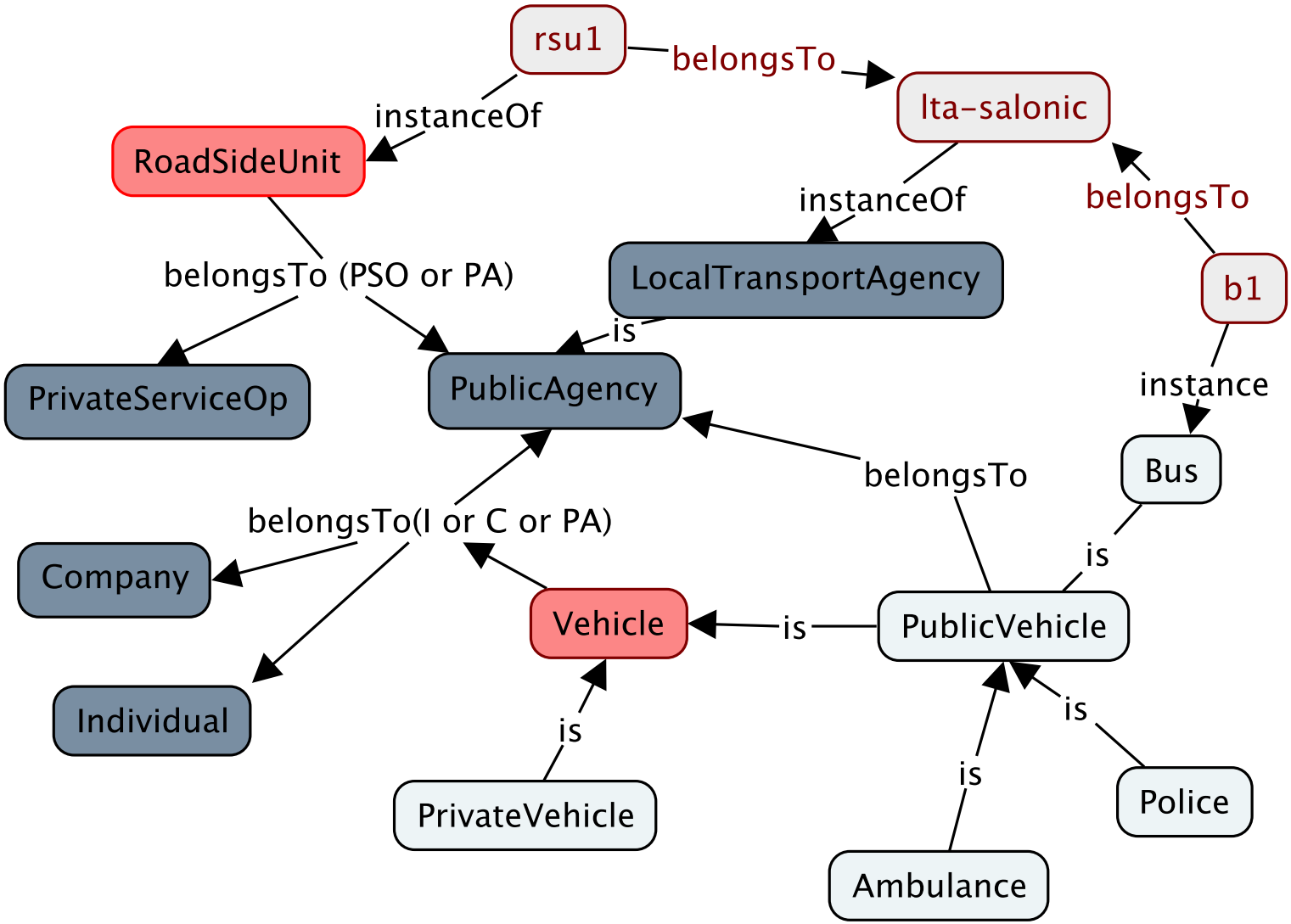
- Aim: integration of agent technology in the emerging field of vehicular networks.
- Technology: Car-2-X communication, multi-agent systems, reasoning on ontologies

Vehicle overtaking scenario

Our system makes use of:

- the AllegroGraph system for geospatial reasoning
- RacerPro server for semantic and temporal reasoning.

Modeling VANETs terminology in DL



Types of data - PrimitiveDataElement. DataFrame

Data elements are formalized by Society of Automative Engineers (SAE)

(*implies* *Latitude* *PrimitiveDataElement*)

(*implies* *Longitude* *PrimitiveDataElement*)

(*implies* *Velocity* *PrimitiveDataElement*)

(*implies* *VehicleLength* *PrimitiveDataElement*)

Types of data - PrimitiveDataElement. DataFrame

Data elements are formalized by Society of Automative Engineers (SAE)

$$(implies \quad Latitude \quad PrimitiveDataElement)$$
$$(implies \quad Longitude \quad PrimitiveDataElement)$$
$$(implies \quad Velocity \quad PrimitiveDataElement)$$
$$(implies \quad VehicleLength \quad PrimitiveDataElement)$$
$$(implies \quad Latitude \quad (and(some \quad hasValue \quad Real)$$
$$(all \quad measures \quad UnitOfMeasure)$$
$$(some \quad hasAcc \quad Real)))$$

Types of data - PrimitiveDataElement. DataFrame

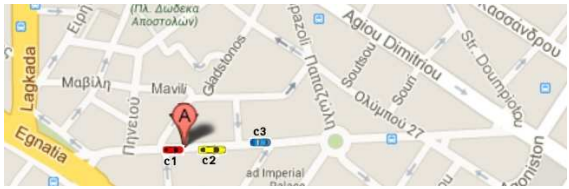
Data elements are formalized by Society of Automative Engineers (SAE)

(*implies* *Latitude* *PrimitiveDataElement*)
(*implies* *Longitude* *PrimitiveDataElement*)
(*implies* *Velocity* *PrimitiveDataElement*)
(*implies* *VehicleLength* *PrimitiveDataElement*)
(*implies* *Latitude* (*and*(*some* *hasValue* *Real*)
(*all* *measures* *UnitOfMeasure*)
(*some* *hasAcc* *Real*)))
(*implies* *DataFrame* (*and* (*some* *hasID* *ID*)
(*some* *hasDescription* *String*)
(*some* *hasContent* *PrimitiveDataElement*)))

PositionDataFrame. SenderDataFrame

(implies PositionDataFrame (and DataFrame (equal hasID 21)
(some hasLat Latitude) (some hasLong Longitude)))
(implies SenderDataFrame (and DataFrame (equal hasID 15)
(some hasLength Real) (some hasWidth Real)
(some hasModel Vehicle)))

Abox for data elements



(instance lat1 (and Latitude (= hasValue 40.6393)
(= hasAcc .9)))
(instance long1(and Longitude (= hasValue 22.9446)
(= hasAcc .9)))

(instance p1 (and PositionDataFrame
(= hasLatitude lat1) (= hasLongitude long1)))
(instance daciaLogan Vehicle)
(instance s1 (and SenderDataFrame
(= hasLength 4.288) (= hasWidth 1.989)))
(equals hasModel daciaLogan)))

Types of Messages

Definition

A vehicular message is a tuple $\langle CT, TM, Data, Range \rangle$, where CT is a shortcut for the concept *CommunicationType*, TM transmission mode, Data represents the content of the message, and Range is maximum communication range.

- CommunicationType: V2V or V2I
- TransmissionMode: Periodic EventDriven

Message - *n-ary relationship* ontology design pattern

Types of Messages

```
51. (implies Message (and (some hasComm CommunicationType)
52.                       (some hasTransmission TransmissionMode)
53.                       (some hasContent Data)
54.                       (some hasRange Integer)))
```

Types of Messages

```
51. (implies Message (and (some hasComm CommunicationType)
52.                       (some hasTransmission TransmissionMode)
53.                       (some hasContent Data)
54.                       (some hasRange Integer)))
57. (equiv CommunicationType (or V2V V2I))
58. (disjoint V2V V2I)
```

Types of Messages

```
51. (implies Message (and (some hasComm CommunicationType)
52.                       (some hasTransmission TransmissionMode)
53.                       (some hasContent Data)
54.                       (some hasRange Integer)))
57. (equiv CommunicationType (or V2V V2I))
58. (disjoint V2V V2I)
59. (equiv TransmissionMode (or Periodic EventDriven))
60. (disjoint Periodic EventDriven)
```

Types of Messages

```
51. (implies Message (and (some hasComm CommunicationType)
52.           (some hasTransmission TransmissionMode)
53.           (some hasContent Data)
54.           (some hasRange Integer)))
57. (equiv CommunicationType (or V2V V2I))
58. (disjoint V2V V2I)
59. (equiv TransmissionMode (or Periodic EventDriven))
60. (disjoint Periodic EventDriven)
61. (implies PeriodicMessage (and Message
62.           (some hasTransmission Periodic)
63.           (some hasfrequency Time)))
64. (implies EventDrivenMessage (and Message
65.           (some hasTransmission Event-Driven)
66.           (some isTriggeredby Event)))
```

Types of Messages

```
51. (implies Message (and (some hasComm CommunicationType)
52.          (some hasTransmission TransmissionMode)
53.          (some hasContent Data)
54.          (some hasRange Integer)))
57. (equiv CommunicationType (or V2V V2I))
58. (disjoint V2V V2I)
59. (equiv TransmissionMode (or Periodic EventDriven))
60. (disjoint Periodic EventDriven)
61. (implies PeriodicMessage (and Message
62.          (some hasTransmission Periodic)
63.          (some hasfrequency Time)))
64. (implies EventDrivenMessage (and Message
65.          (some hasTransmission Event-Driven)
66.          (some isTriggeredby Event)))
67. (implies Accident Event)
68. (implies TrafficJam Event)
69. (implies Overtaking Event)
```

Types of Messages

```
51. (implies Message (and (some hasComm CommunicationType)
52.           (some hasTransmission TransmissionMode)
53.           (some hasContent Data)
54.           (some hasRange Integer)))
57. (equiv CommunicationType (or V2V V2I))
58. (disjoint V2V V2I)
59. (equiv TransmissionMode (or Periodic EventDriven))
60. (disjoint Periodic EventDriven)
61. (implies PeriodicMessage (and Message
62.           (some hasTransmission Periodic)
63.           (some hasfrequency Time)))
64. (implies EventDrivenMessage (and Message
65.           (some hasTransmission Event-Driven)
66.           (some isTriggeredby Event)))
67. (implies Accident Event)
68. (implies TrafficJam Event)
69. (implies Overtaking Event)
70. (ShortRangeMessage (and Message (< hasRange 1000)))
```

Lane Changing Warning Message

```
LaneChangeWarningMessage = ⟨ V2V, EventDriven
  Data[
    DataFrames [SenderDataFrame, PositionDataFrame],
    PrimitiveDataElements[Velocity, Acceleration,
                          TurnSignalStatus]
    ], 400⟩
```

Types of agents

According to communication type:

(implies PassiveAg (and Agent (some sendMsg PeriodicMsg)))
(implies ActiveAg (and Agent (some sendMsg EventDrivenMsg)))

Types of agents

According to communication type:

(implies PassiveAg (and Agent (some sendMsg PeriodicMsg)))
(implies ActiveAg (and Agent (some sendMsg EventDrivenMsg)))

According to overtaking type:

(implies NormalAg (and Agent (some hasEvent NormalOvertaking)))
(implies FlyingAg (and Agent (some hasEvent FlyingOvertaking)))
(implies PiggyAg (and Agent (some hasEvent PiggyOvertaking)))
(implies Two+Ag (and Agent (some hasEvent TwoPlusOvertaking)))

Types of agents

According to communication type:

(implies PassiveAg (and Agent (some sendMsg PeriodicMsg)))
(implies ActiveAg (and Agent (some sendMsg EventDrivenMsg)))

According to overtaking type:

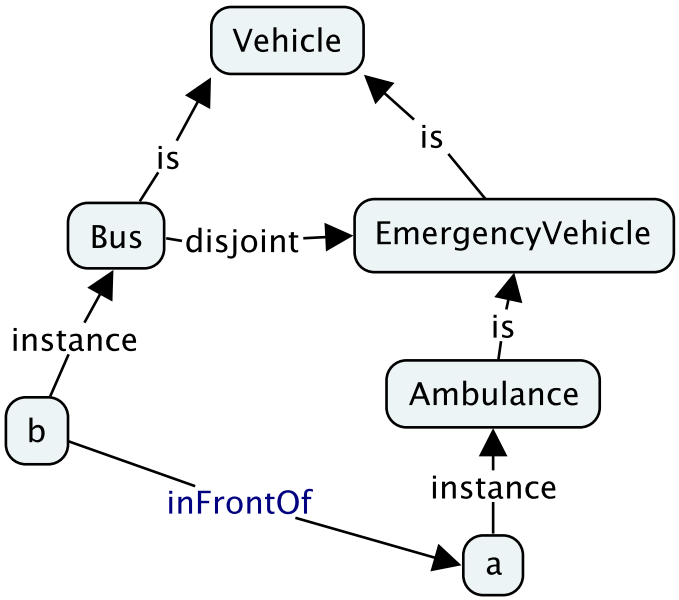
(implies NormalAg (and Agent (some hasEvent NormalOvertaking)))
(implies FlyingAg (and Agent (some hasEvent FlyingOvertaking)))
(implies PiggyAg (and Agent (some hasEvent PiggyOvertaking)))
(implies Two+Ag (and Agent (some hasEvent TwoPlusOvertaking)))

According to politeness:

```
(implies PoliteAg (and Agent
        (or (some hasEvent DecreasingSpeedDuringOvertaking))
         (some hasEvent SignalsRightBeforeOvertaking)
         (some hasEvent ThankMsgAfterLaneChanging))))
```

Domain knowledge

- Vehicular ontol



- Open Street Maps to Allegro Graph Server

Geospatial continuous reasoning

1. based on location service and beaconed data
2. Allegro GeoSpatial Reasoning:
   - get the cars inside a polygon that defines the street
   - get the cars on the same street ordered by their distance to the current car:

SELECT ?car ?p {?car ex:location ?p.
?car onStreet ex:way1.}
ORDER BY $<$ *http* : //*franz.com*/*ns*/*allegrograph*/3.0/*geospatial*/*fn*/
haversineKilometers $>$ (?*o*, *POINT*(22.939007, 40.640392))

Temporal predicates in vehicular streams

Temporal predicate	Informal semantics
$((move\ ?o)\ t_{start}\ t_{end})$	object $?o$ is known to be moving between time t_{start} and time t_{end}
$((approach\ ?o1\ ?o2)\ t_{start}\ t_{end})$	$?o1$ is approaching object $?o2$ during the time interval $[t_{start}, t_{end}]$
$((behind\ ?o1\ ?o2)\ t_{start}\ t_{end})$	$?o1$ is behind object $?o2$ during the time interval $[t_{start}, t_{end}]$
$((beside\ ?o1\ ?o2)\ t_{start}\ t_{end})$	$?o1$ is beside object $?o2$ during the time interval $[t_{start}, t_{end}]$
$((in\text{-}front\text{-}of\ ?o1\ ?o2)\ t_{start}\ t_{end})$	$?o1$ is in front of object $?o2$ during the time interval $[t_{start}, t_{end}]$

Assertions of Primitive Events

11. (define-event-assertion ((move a1) 5 60))
12. (define-event-assertion ((move b1) 1 50))
13. (define-event-assertion ((approach a1 b1) 10 20))
14. (define-event-assertion ((behind a1 b1) 10 20))
15. (define-event-assertion ((beside a1 b1) 20 30))
16. (define-event-assertion ((in-front-of a1 b1) 30 60))

a1 b1 : approach, behind 10-20

Assertions of Primitive Events

11. (define-event-assertion ((move a1) 5 60))
12. (define-event-assertion ((move b1) 1 50))
13. (define-event-assertion ((approach a1 b1) 10 20))
14. (define-event-assertion ((behind a1 b1) 10 20))
15. (define-event-assertion ((beside a1 b1) 20 30))
16. (define-event-assertion ((in-front-of a1 b1) 30 60))

a1 b1 : approach, behind 10-20

Assertions of Primitive Events

11. (define-event-assertion ((move a1) 5 60))
12. (define-event-assertion ((move b1) 1 50))
13. (define-event-assertion ((approach a1 b1) 10 20))
14. (define-event-assertion ((behind a1 b1) 10 20))
15. (define-event-assertion ((beside a1 b1) 20 30))
16. (define-event-assertion ((in-front-of a1 b1) 30 60))

b1: beside 20-30: a1

a1

Assertions of Primitive Events

11. (define-event-assertion ((move a1) 5 60))
12. (define-event-assertion ((move b1) 1 50))
13. (define-event-assertion ((approach a1 b1) 10 20))
14. (define-event-assertion ((behind a1 b1) 10 20))
15. (define-event-assertion ((beside a1 b1) 20 30))
16. (define-event-assertion ((in-front-of a1 b1) 30 60))

b1 a1: in-front-of 30-60

Complex Events Recognition: Racer Reasoning

```
(define-event-rule ((overtake ?i ?j) ?t1 ?t2)
        ((?i vehicle) ?t0 ?tn)
        ((?i ?j on-same-line) ?t0 ?tn)
        ((move ?i) ?t0 ?t2)
        ((move ?j) ?t1 ?t2)
        ((approach ?i ?j) ?t1 ?t3)
        ((behind ?i ?j) ?t1 ?t3)
        ((beside ?i ?j) ?t3 ?t4)
        ((in-front-of ?i ?j) ?t4 ?t2)
        ((on-same-line ?i ?j) ?t4 ?t2))
```

Overtaking scenario

Before overtaking, c1 should check that:

1. it is allowed to overtake;
2. c2 does not signal left;
3. there is sufficient distance to return to the same lane without endangering vehicle c3 coming from the opposite direction or breaching the norms (i.e. continuous line);
4. no other vehicle is overtaking c1, by checking the road behind;
5. signal intention to overtake for long enough to warn all other road users.

`(timenet-retrieve ((overtake c1 c2) ?t1 ?t2))` - to check if the vehicle c1 successfully overtook c2

Conclusions

Integration of agent technologies with network communication has huge potential in the context of vehicular networks.
Contributions:

- a model to encapsulate vehicular data as assertion in an ontology
- formalisation of messages for vehicular communication in Description Logic
- formalisation of agent types for the overtaking scenario.

The agents were empowered with temporal and spatial reasoning capabilities in order to reason on volatile vehicular knowledge.