Automatic Generation of Test-cases of Increasing Complexity for Autonomous Vehicles at Intersections

(ICCPS 2022)

Abolfazl Karimi
Parasara Sridhar Duggirala

THE UNIVERSITY
of NORTH CAROLINA
at CHAPEL HILL

Have we tested enough yet?

Waymo's autonomous vehicles have clocked 20 million miles on public roads

Waymo's driverless cars were involved in 18 accidents over 20 months

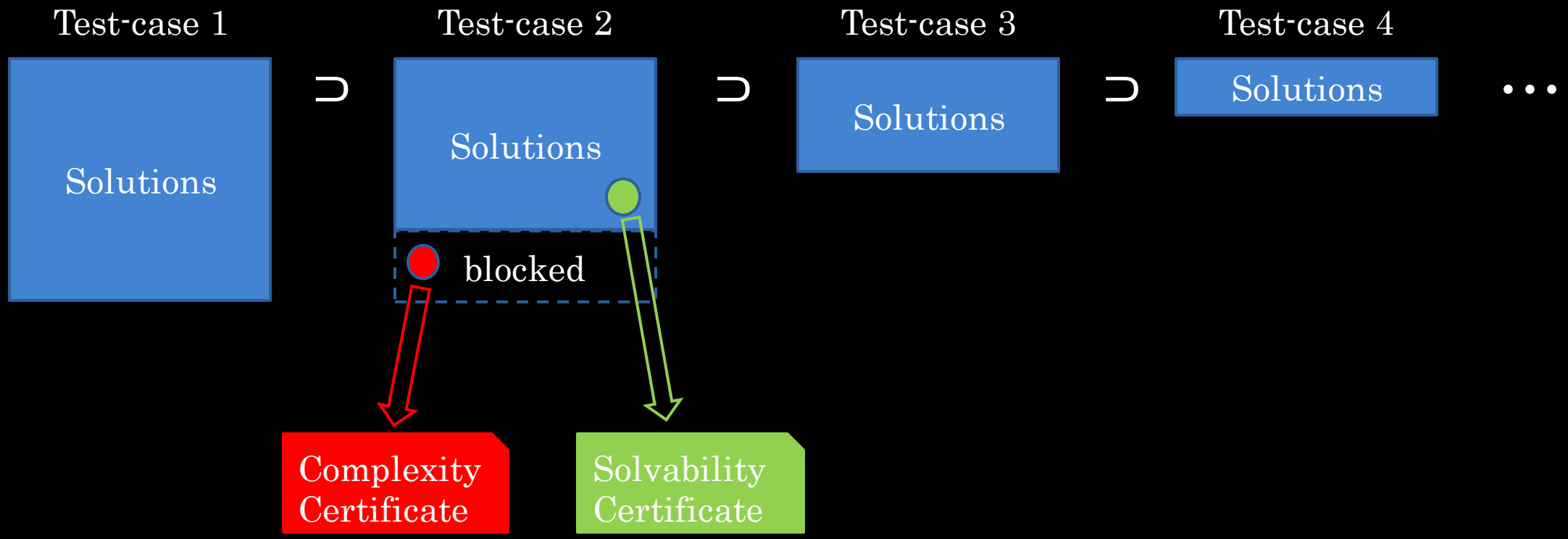
Autonomous vehicles would have to be driven hundreds of millions of miles and sometimes hundreds of billions of miles to demonstrate their reliability in terms of fatalities and injuries.

Tesla cars register one crash for every 4.31 million miles driven with Autopilot

Test-case complexity

- Shrink the set of solutions incrementally

⊃ : proper superset



Contributions

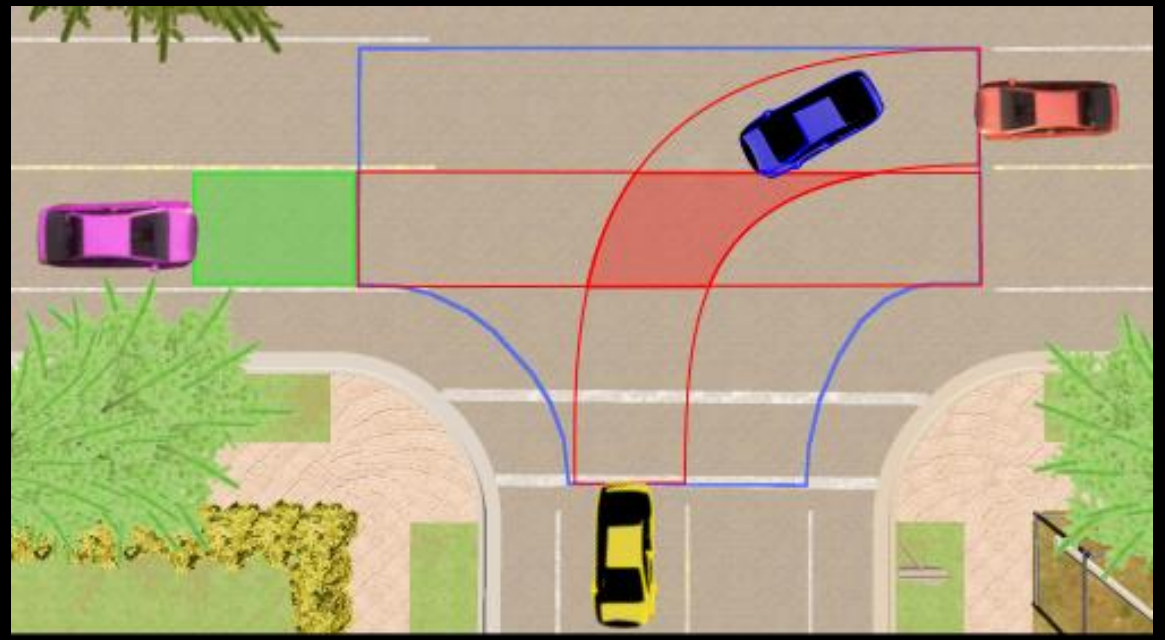
1. Formalized Test-case Complexity
   - More right-of-way constraints → more-complex
2. Generate test-cases
   - Traffic rules → concrete trajectories
3. Generate certificates
   - Complexity
   - Solvability

Outline

- Test-case scenarios
  - Predicate-level abstraction of a scenario
  - Traffic rules as PASS/FAIL criteria
- Generation algorithm
  1. Ordering of events using ASP
  2. Concretize timing of events and speed profiles using SMT
  3. If collisions, try next ASP solution
- Results

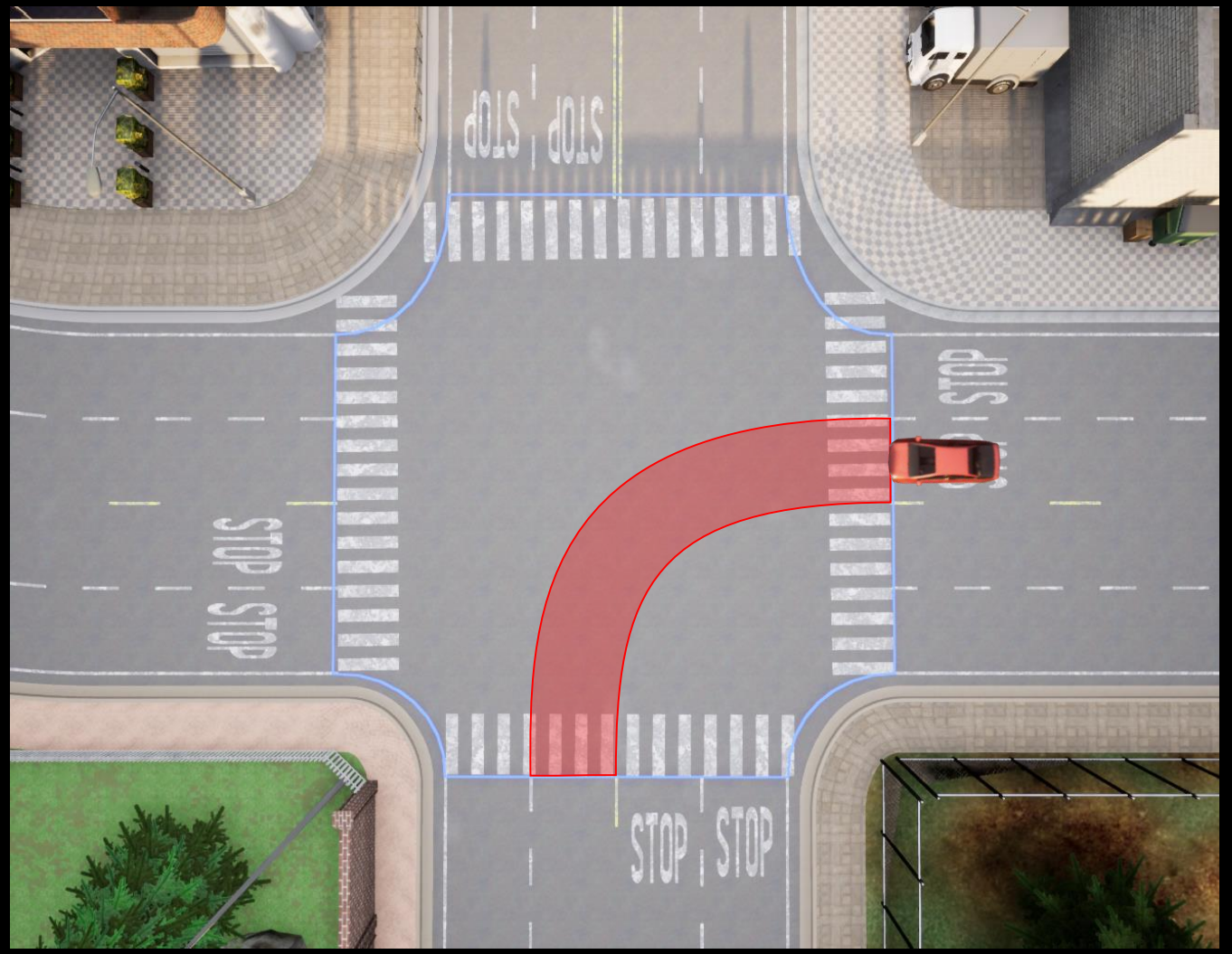
Predicate abstraction of a scenario

- Regions
  - Lanes
  - Lane sections
- Events
  - Entering
  - Exiting
  - Velocity reaching a threshold
- Temporal relations
  - Earlier
  - Same time

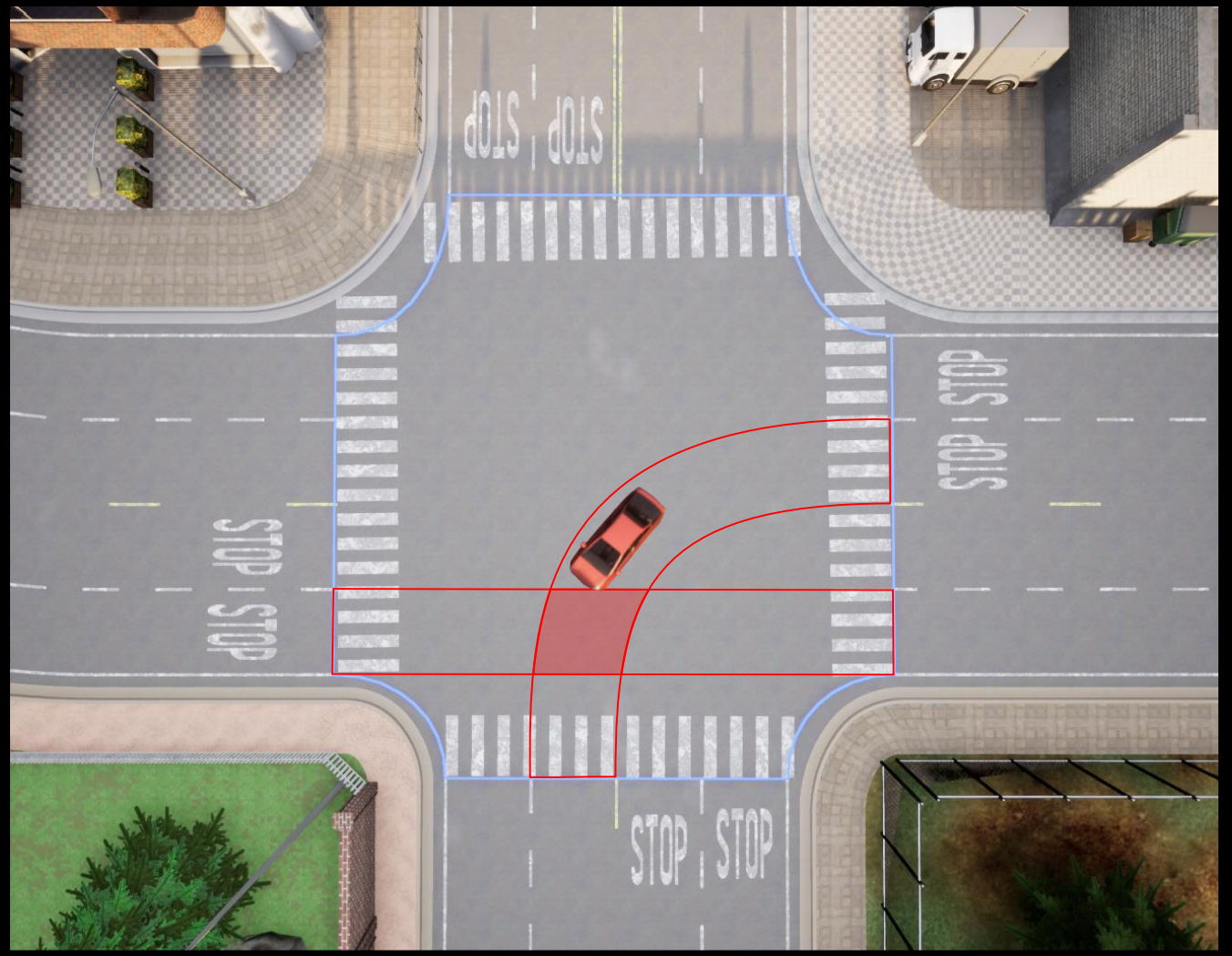


Event: arrival at intersection

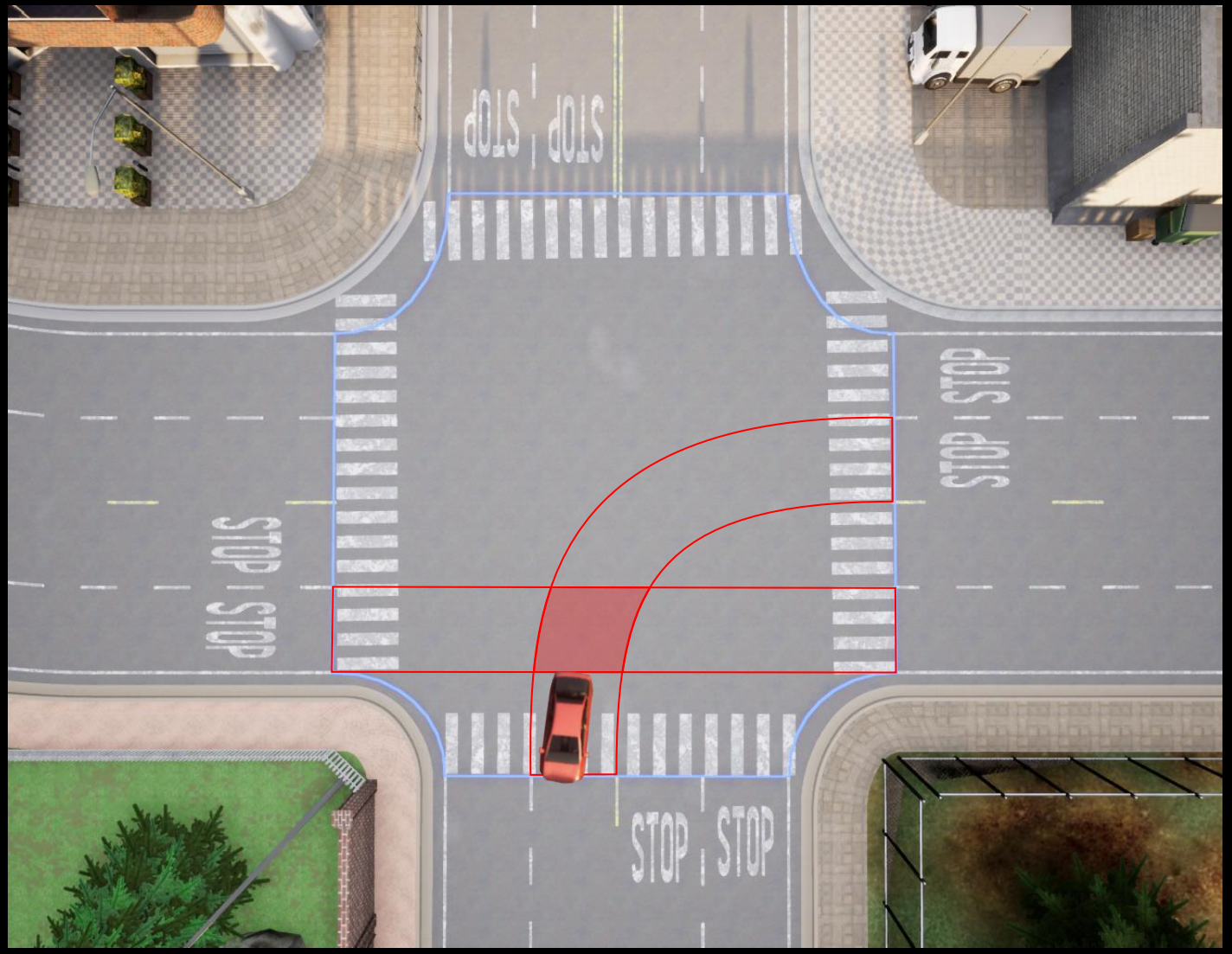
Event: entering a lane



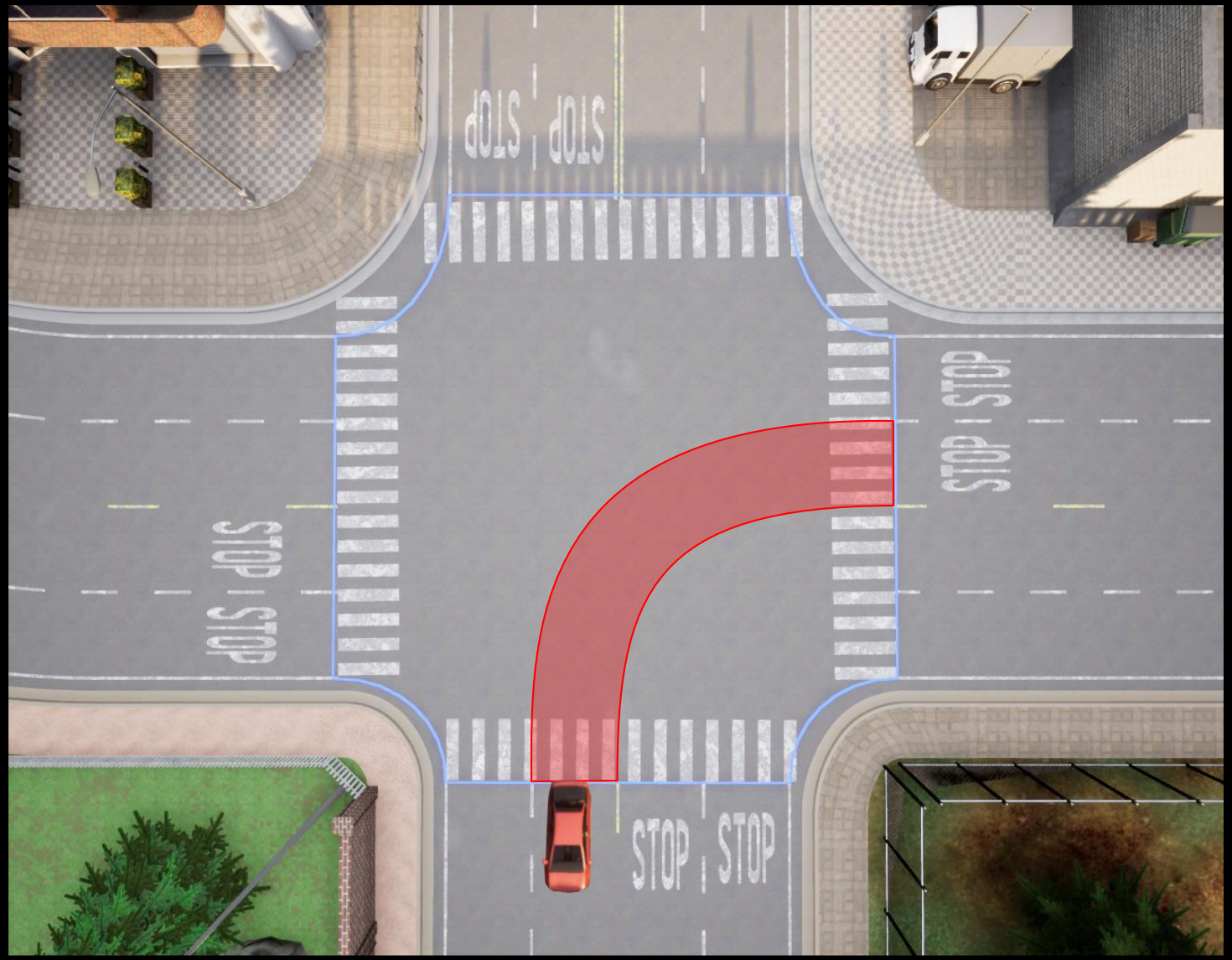
Event: entering a shared section



Event: exiting a shared section



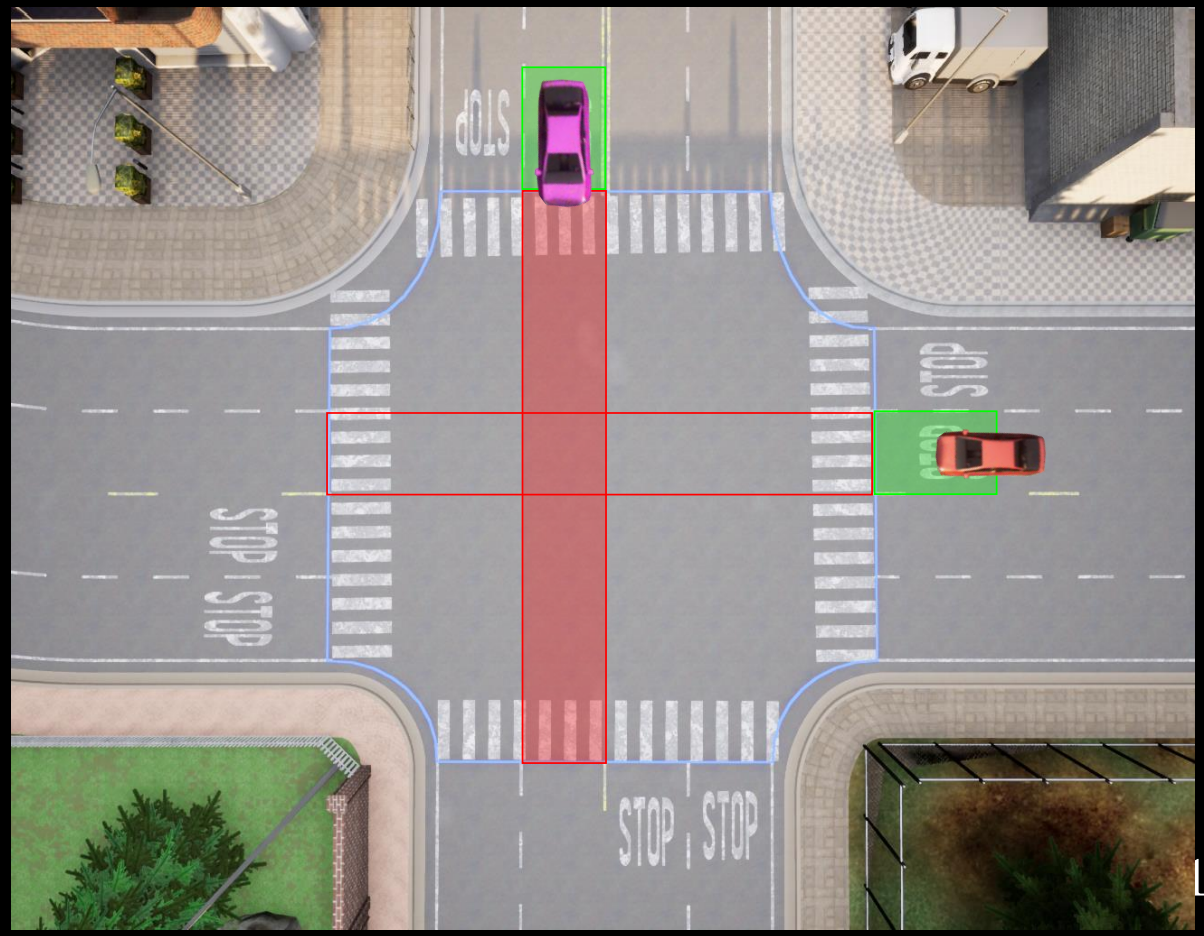
Events: exiting a lane



Relative order of events

Pink arrives first.

Pink enters first.

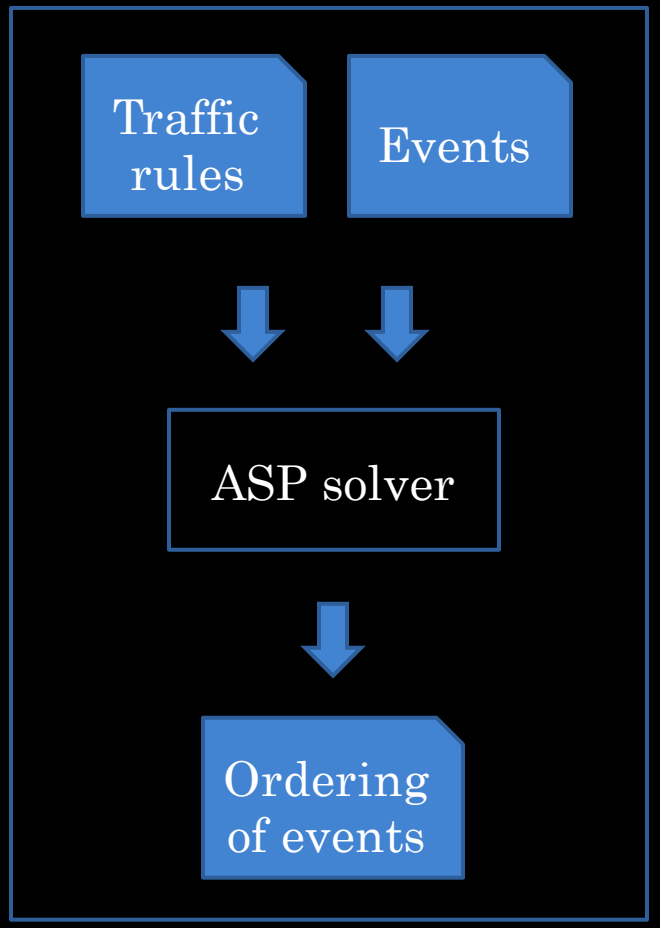


Traffic rules and events' order

- *"whoever arrives first, should enter first."*
- *First-Order-Logic formulation:*

```
violatesRightOfForRule(V1, V2, fcfs) :-
 arrivedAtTime(V1, Ta1),
 arrivedAtTime(V2, Ta2),
 lessThan(Ta1, Ta2),
 enteredAtTime(V1, Te1),
 enteredAtTime(V2, Te2),
 lessThan(Te2, Te1).
```

Order events using ASP

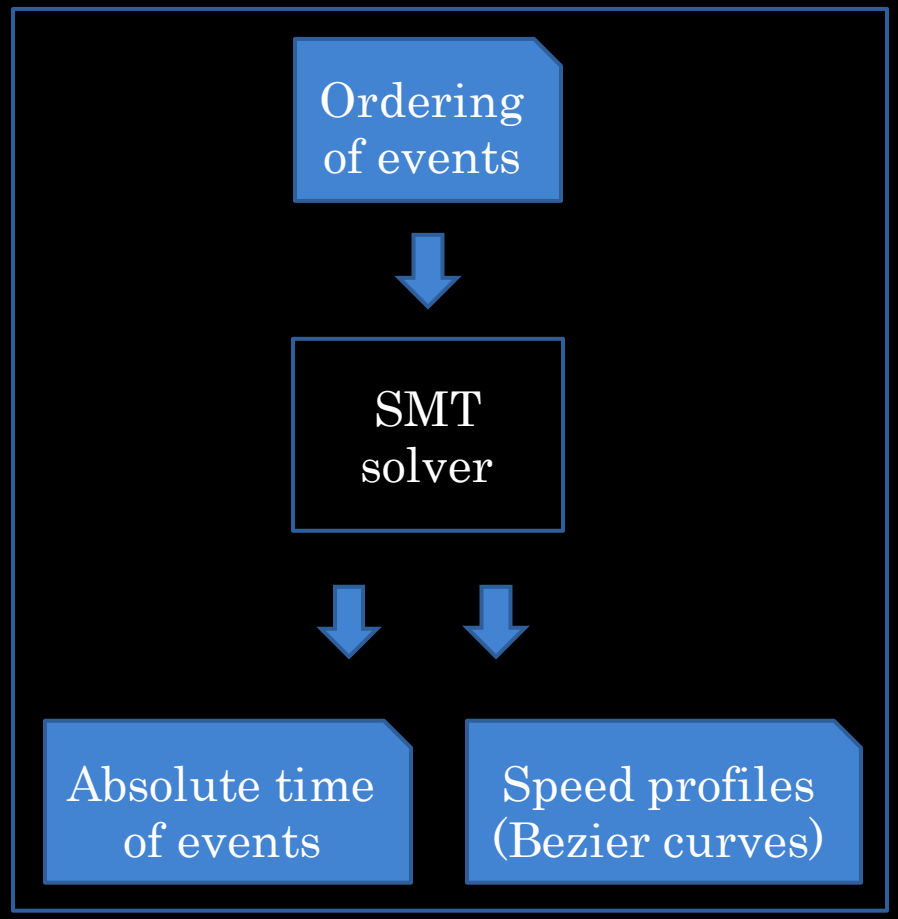


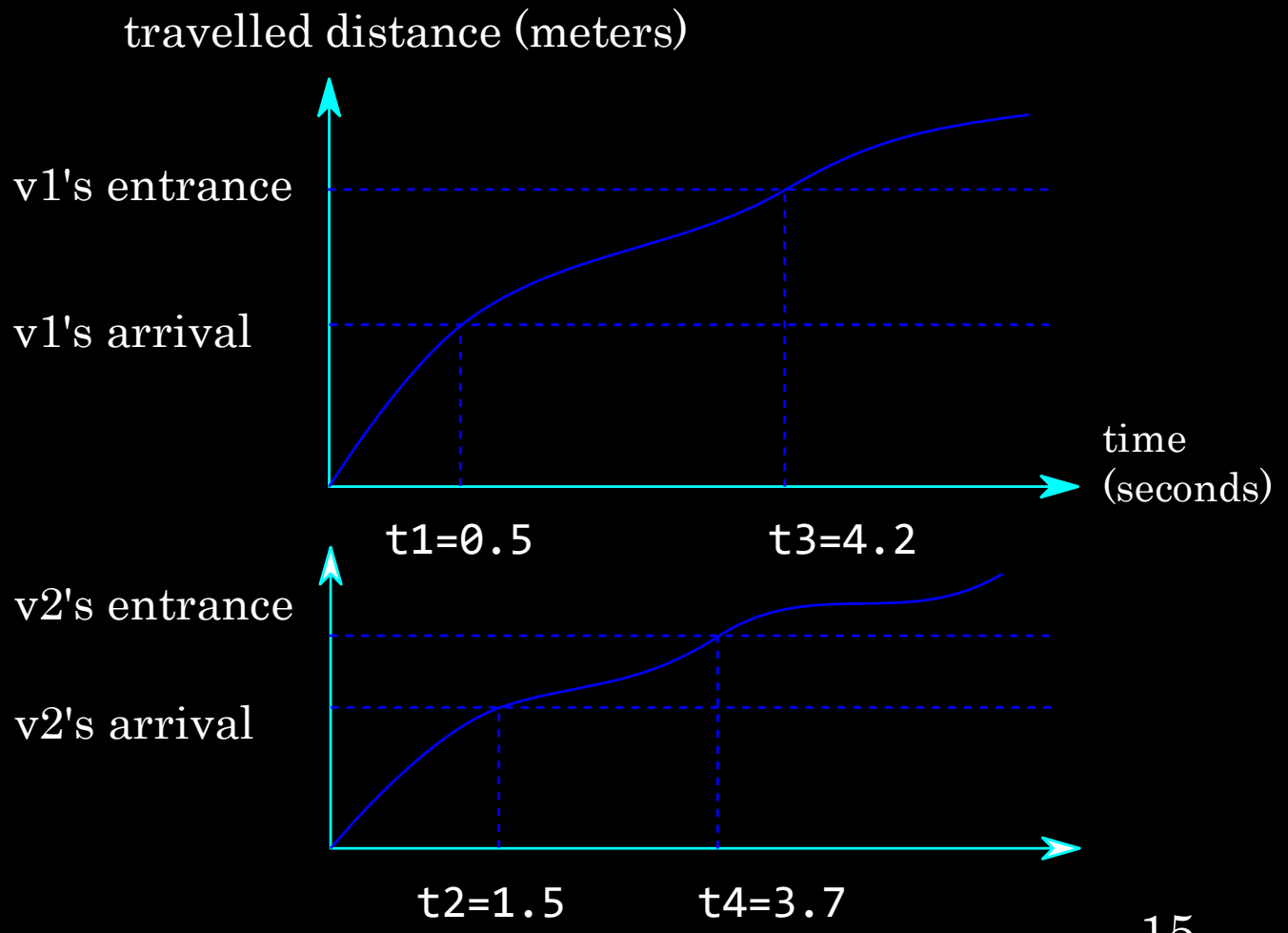

```
violatesRightOf(V1, V2) :-
 arrivedAtTime(V1, Ta1),
 arrivedAtTime(V2, Ta2),
 lessThan(Ta1, Ta2),
 enteredAtTime(V1, Te1),
 enteredAtTime(V2, Te2),
 lessThan(Te2, Te1).
```

```
arrivedAtTime(v1, t1).
arrivedAtTime(v2, t2).
enteredAtTime(v1, t3).
enteredAtTime(v2, t4).
```

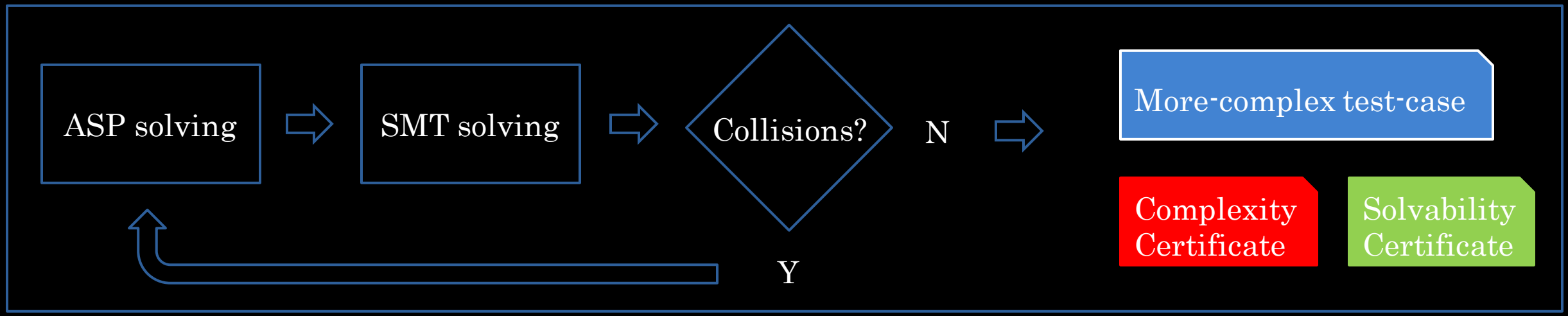
```
lessThan(t1, t2).
lessThan(t4, t3).
```

Absolute timing of events using SMT

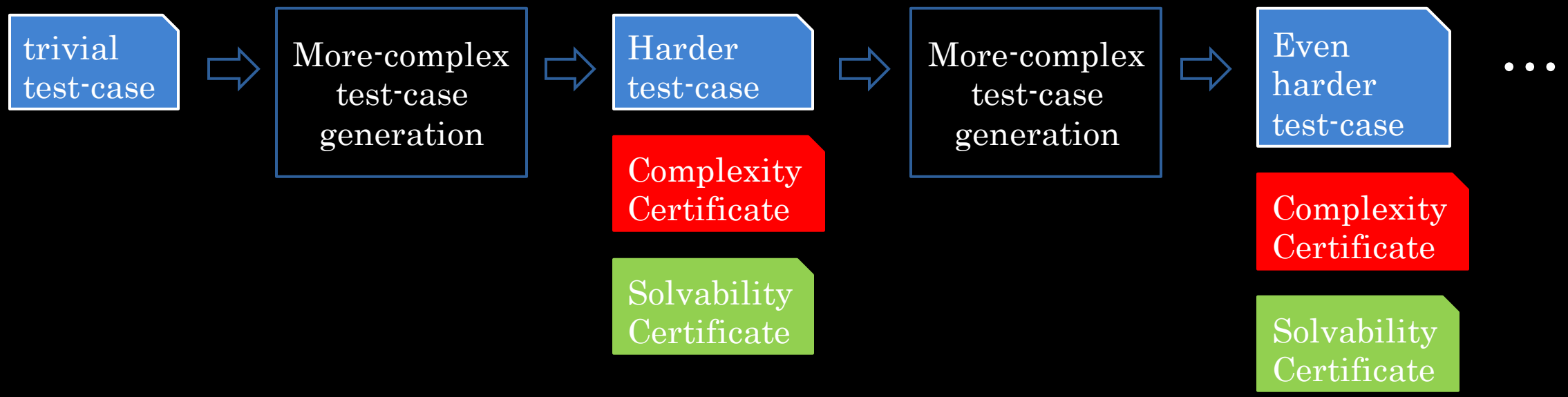




ASP + SMT + collisions



Sequence of increasingly more complex test-cases



Results

- Generate test-cases
- Test autopilot
- Test autopilot + RSS
- Show certificates

Autopilot passes Test-case 1

Autopilot fails Test-case 2 !

Autopilot+RSS passes Test-case 2 !

Autopilot+RSS fails Test-case 3 !

Test-case 2 certificates

- Complexity Certificate:
  - yields to Test-case 1 non-egos
  - violates a Test-case 2 non-ego
- Solvability Certificate:
  - yields to Test-case 1&2 non-egos

Future work

1. Generating reactive scenarios:
   - a non-ego behavior is a function of ego
2. Numerical approximation of complexity
   - Finitization of trajectory space

     e.g. motion primitives and lattices
3. Better collision-checking (or enforcing)

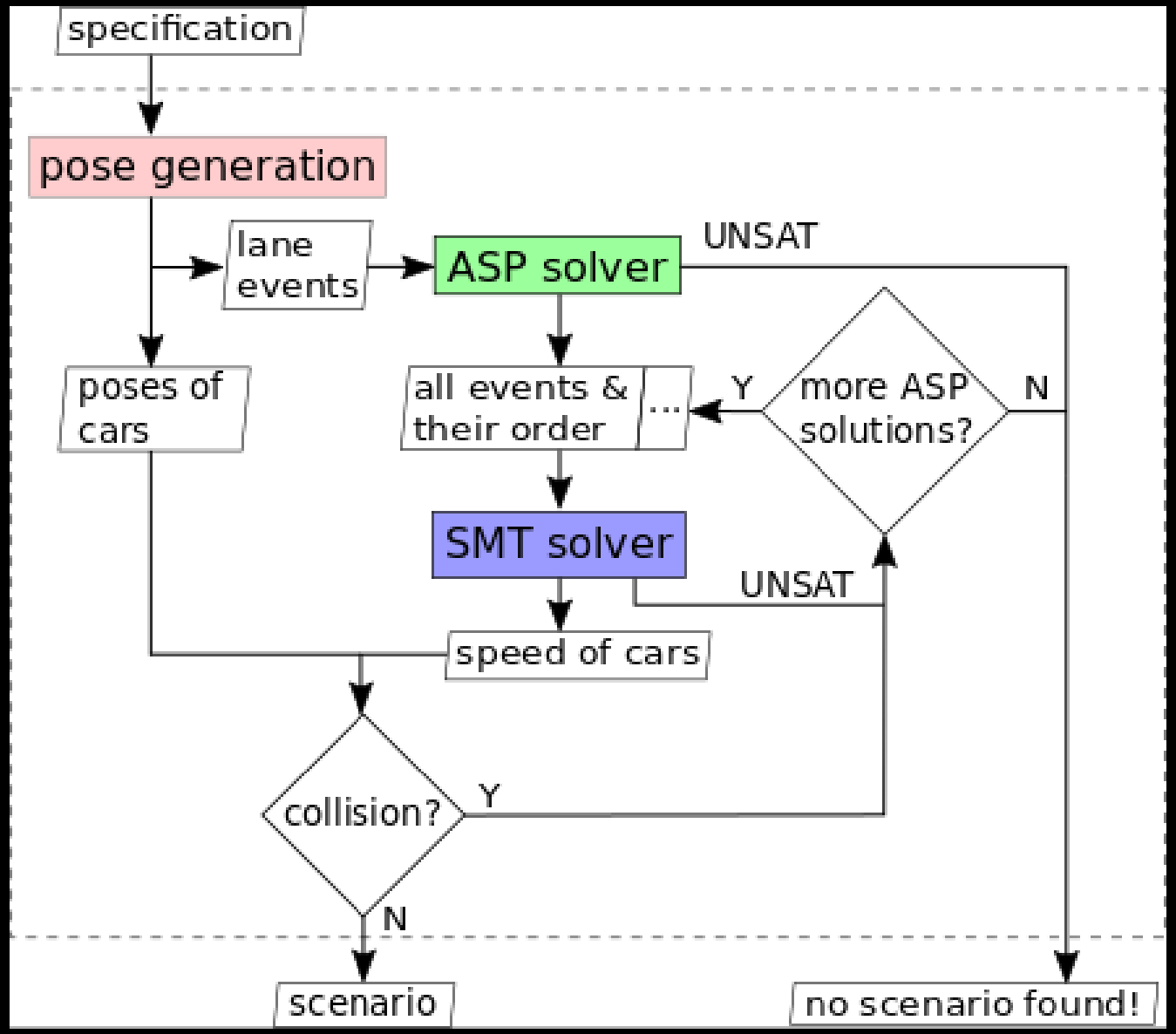
Extra slides

Complexity of SMT constraints

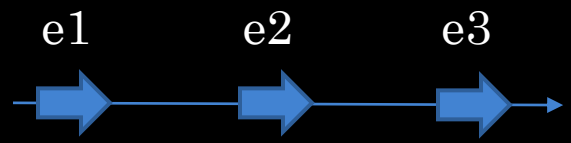
- Linear constraints
  - Temporal order of events
  - Bounds on instantaneous speed at an event
    - Slope between control points
- Quadratic constraints
  - Continuity of speed
    - left slope = right slope
  - Bounds on acceleration

Autopilot+RSS passes Test-case 1

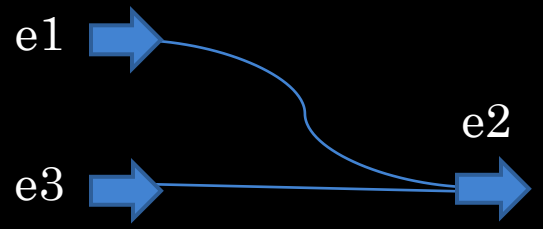
Scenario Generation algorithm



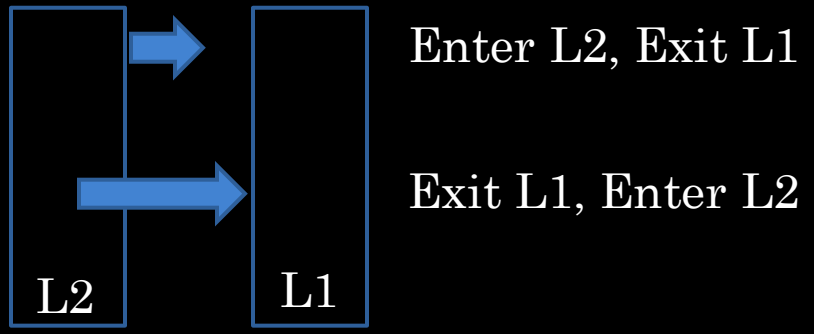
- No teleportation!



- Nonholonomic steering



- Size of vehicles



Traffic rules and order of events

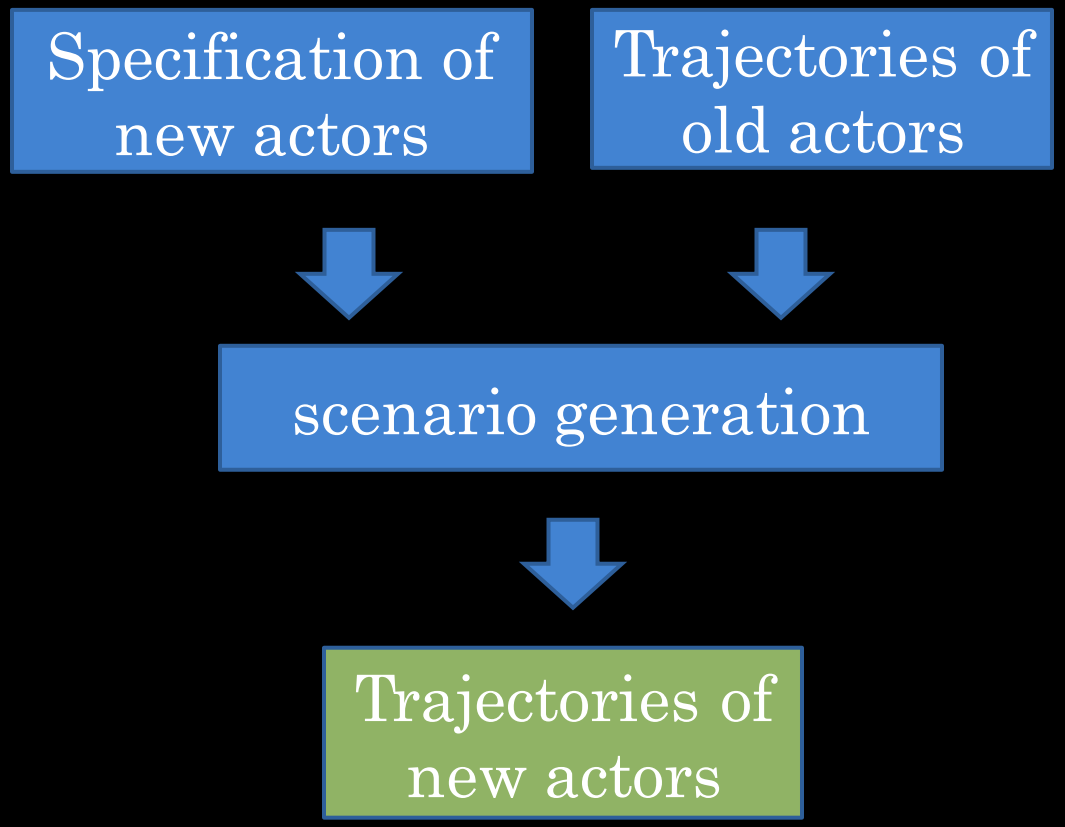
- Whoever arrives first, should enters first.
- If A and B arrive simultaneously and A is on the right of B, then A should enter first.
- …

Automatic test-case generation

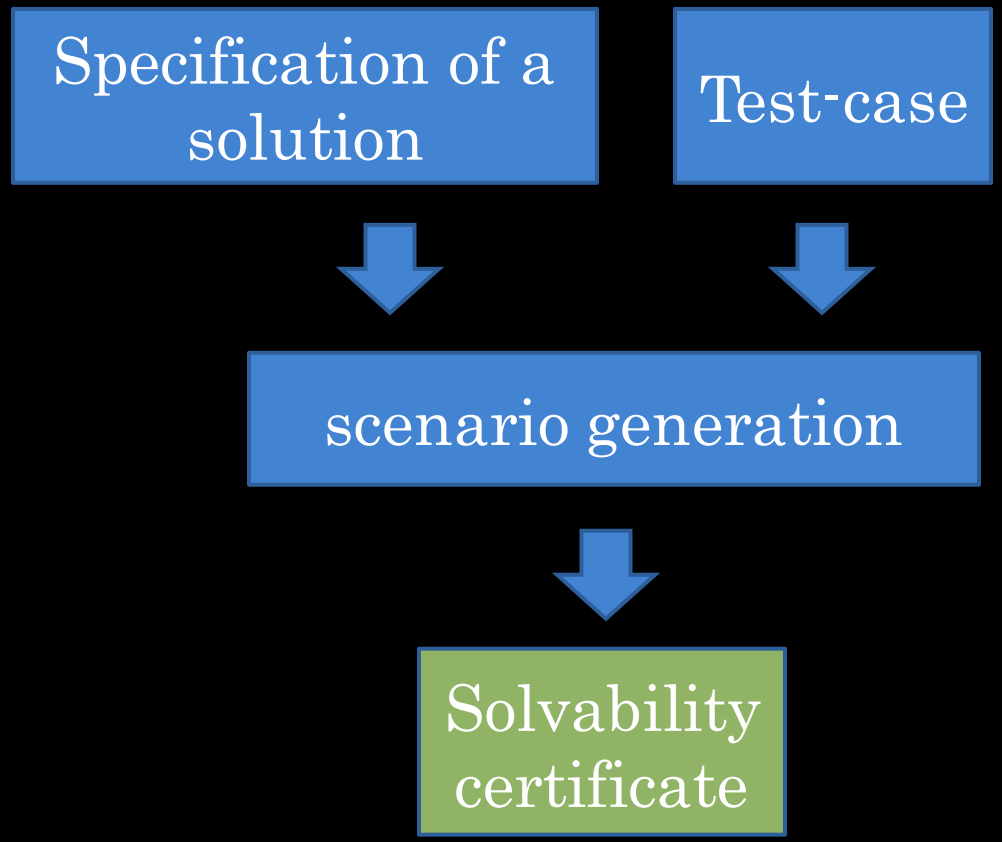
1. Goal:
   - Combinatorial coverage of sequence of events,

     VS probabilistic coverage (random sampling)
2. Constraints:
   - Kinematics (nonholonomic steering, smooth velocities, …)
   - Collisions (vehicles cannot pass through each other)

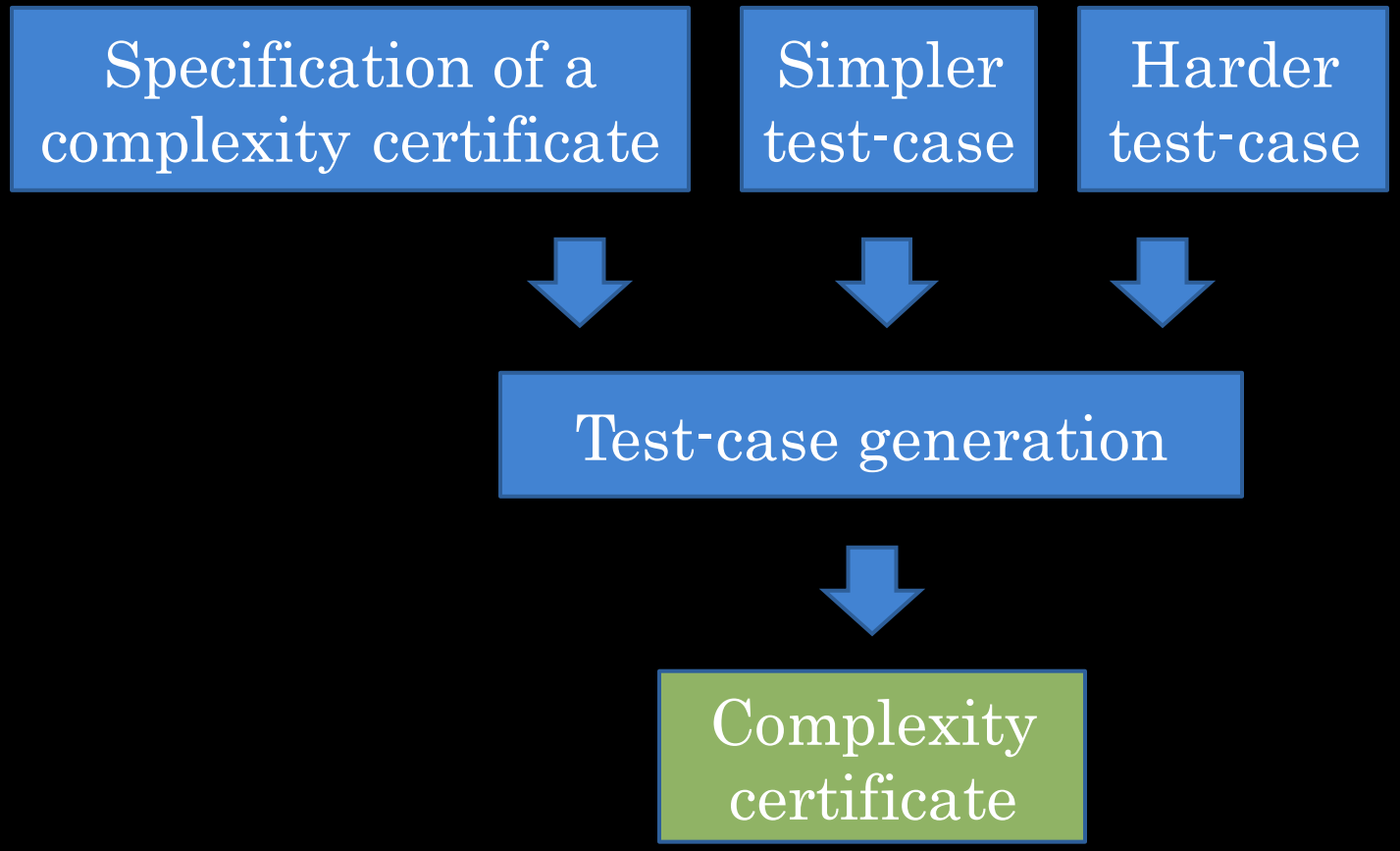
Adding new actors to a scenario



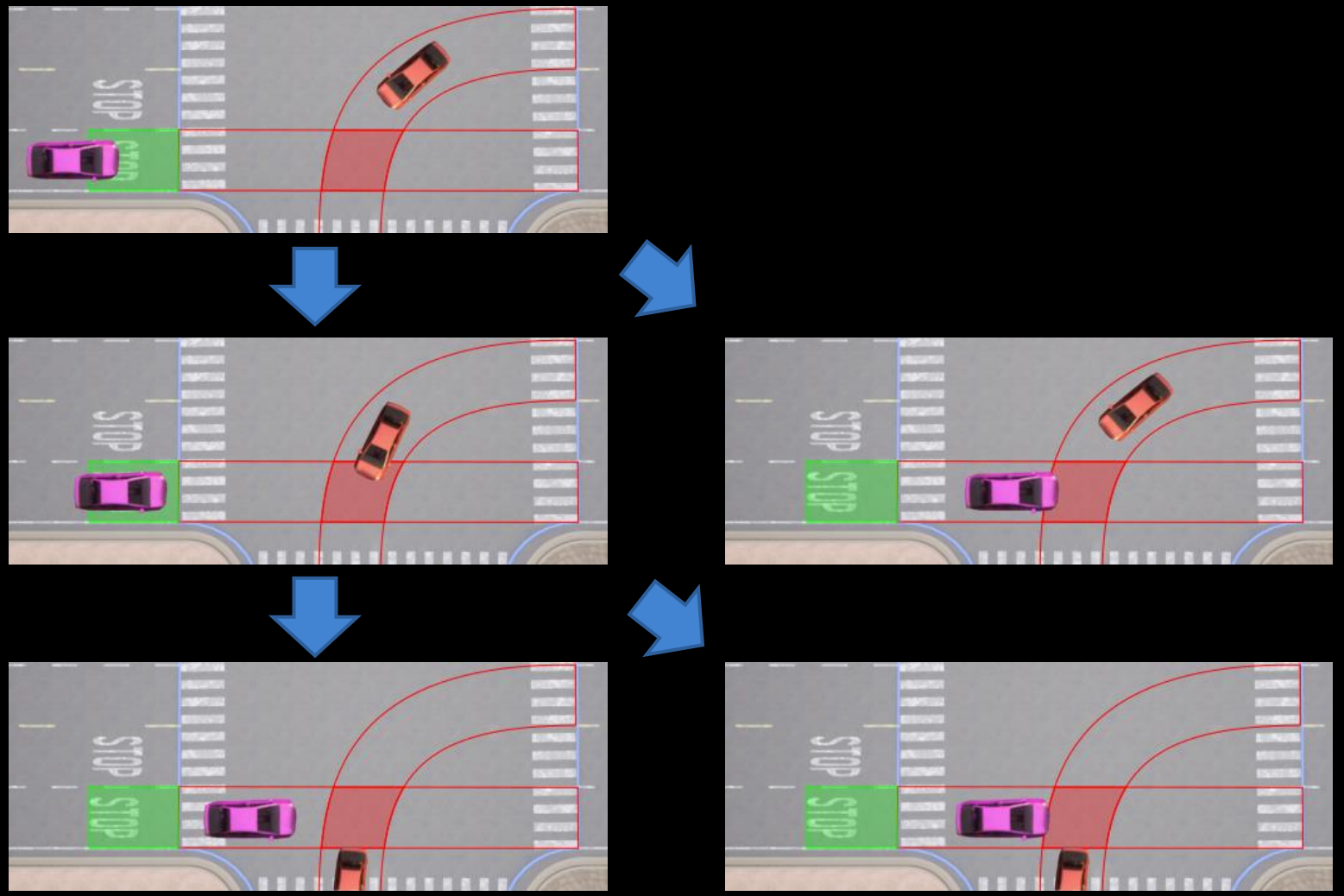
Synthesizing solvability certificate



Synthesizing complexity certificate



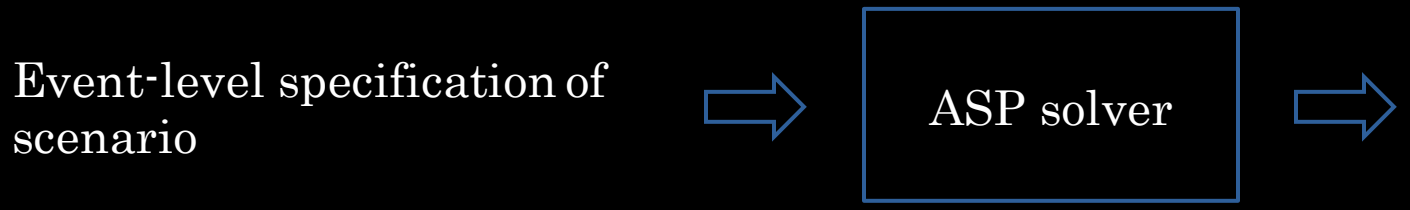
Event ordering examples



Test-case complexity

- Increases probability of failure
  - e.g. difficulty levels of video games
- Fair & efficient comparison of AVs
  - How many levels each AV passes?
- Interpretability
  - Event-level specification
  - Trajectory-level certificate (a blocked solution)

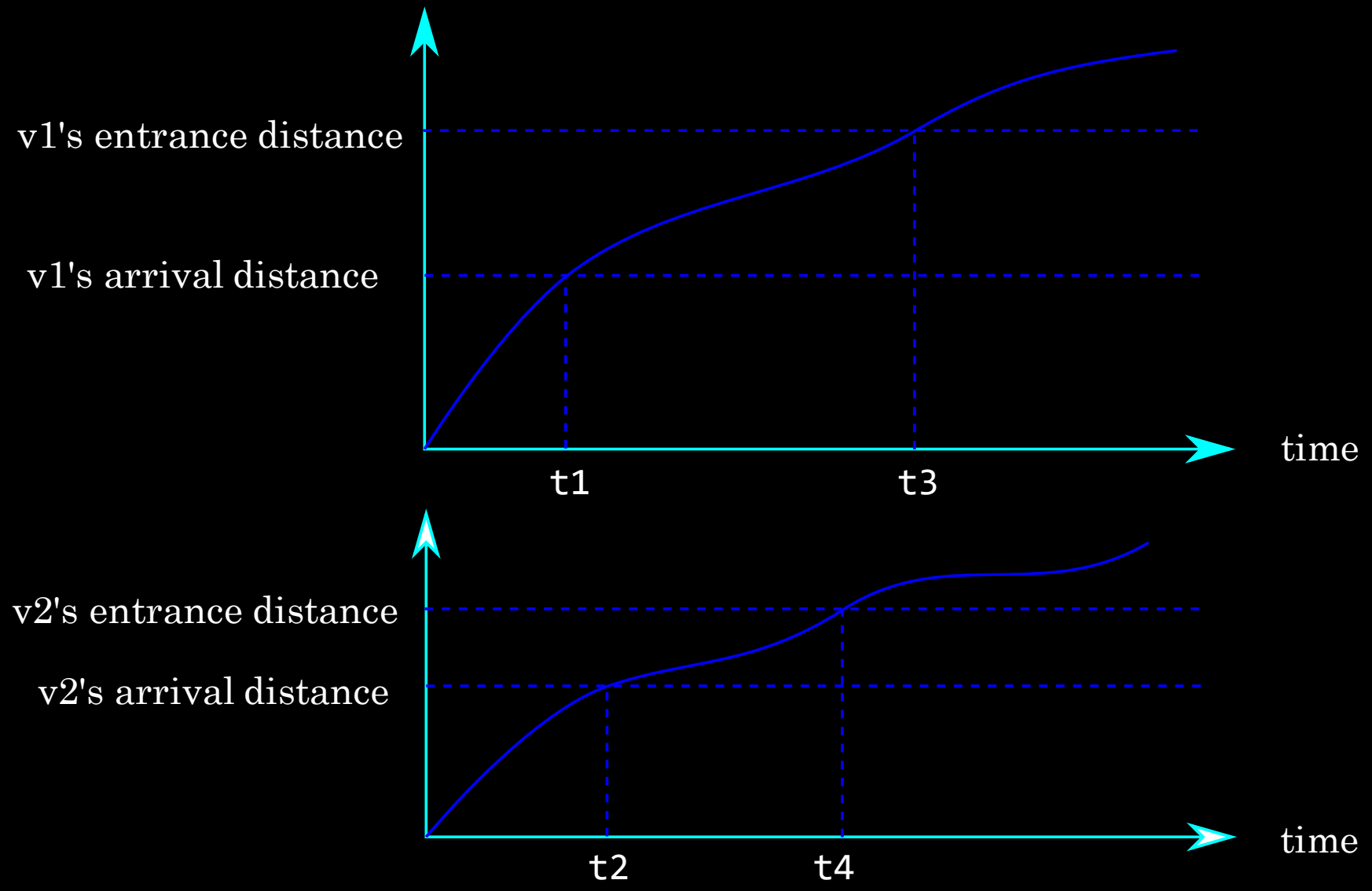
Test-case generation: events




```
arrivedAtTime(v1, t1).
arrivedAtTime(v2, t2).
:- violatedRightOf(v2, v1).
```

```
lessThan(t1, t2).
enteredAtTime(v1, t3).
enteredAtTime(v2, t4).
lessThan(t4, t3).
```

Test-case generation: velocities



Scenario-based testing

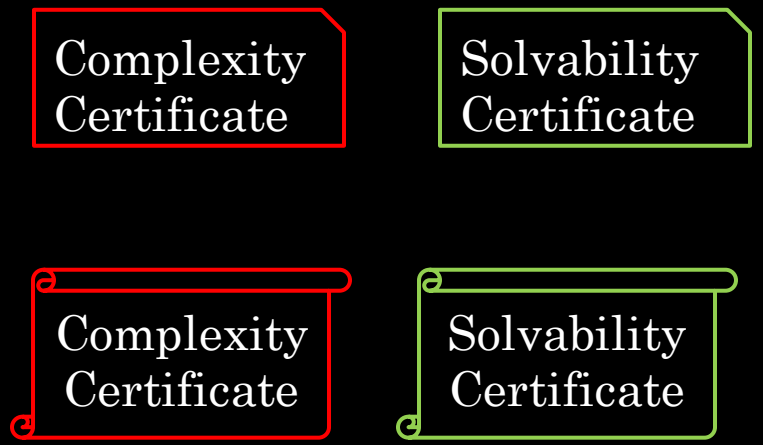
- System-level vs. Component-level
- External vs. internal behavior
  - traffic rules vs. energy consumption
- Blackbox
- Simulation-based

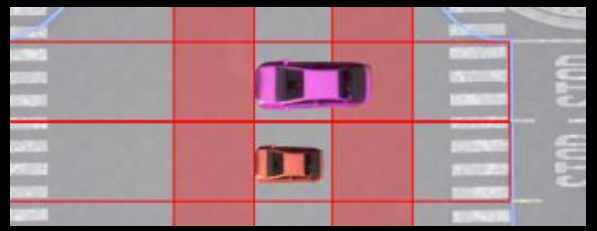
Forcing increase in complexity

Given an old test-case, generate a more-complex new test-case

Event-level specification of scenario
+
Specification of a complexity evidence

Lane events and their temporal order


Complexity Certificate
Solvability Certificate
Complexity Certificate
Solvability Certificate



L2 L1

Enter L2, Exit L1

Exit L1, Enter L2