The Train Benchmark: Cross-Technology Performance Evaluation of Continuous Model Queries

Gábor Szárnyas

MTA-BME Lendület Research Group on Cyber-Physical Systems
Fault Tolerant Systems Research Group
McGill University, Department of Electrical and Computer Engineering

Budapest University of Technology and Economics
Department of Measurement and Information Systems

LDBC benchmarks

- Graphalytics
- Semantics Publishing Benchmark
- Social Network Benchmark
  - Interactive
  - Business Intelligence

LDBC benchmarks

- Graphalytics
- Semantics Publishing Benchmark
- Social Network Benchmark
  - Interactive
  - Business Intelligence

Small updates

Local patterns

LDBC benchmarks

- Graphalytics
- Semantics Publishing Benchmark
- Social Network Benchmark

○ Interactive	Small updates	Local patterns
○ Business Intelligence	Large updates	Global patterns

LDBC benchmarks

- Graphalytics
- Semantics Publishing Benchmark
- Social Network Benchmark
  - Interactive
  - Business Intelligence

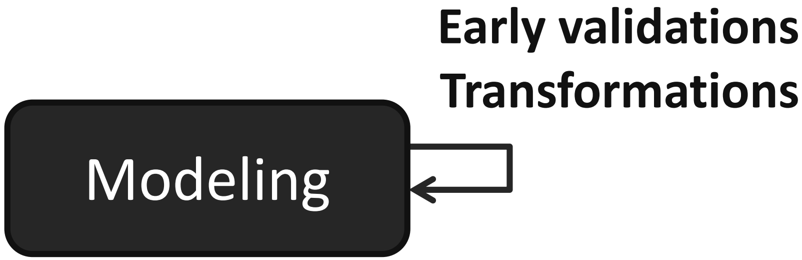
Small updates

Global patterns

Model-Driven Engineering

Modeling

Model-Driven Engineering

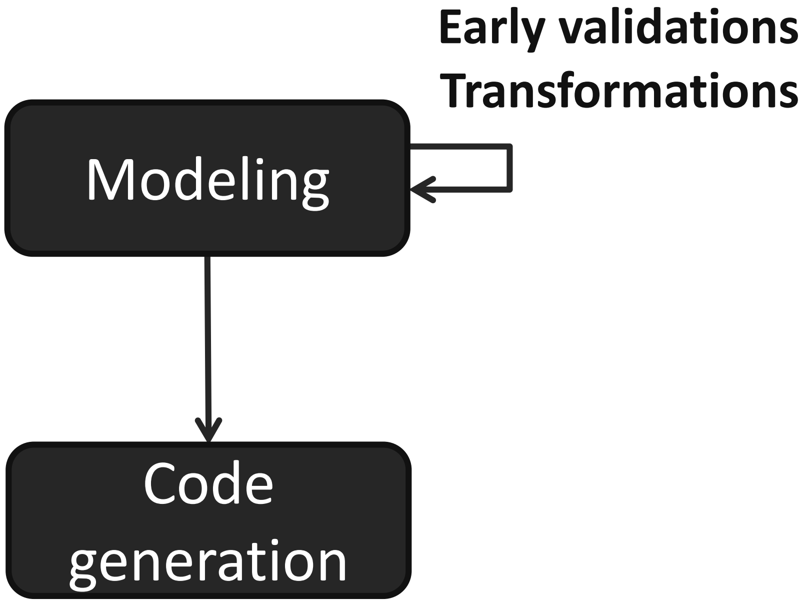


Early validations

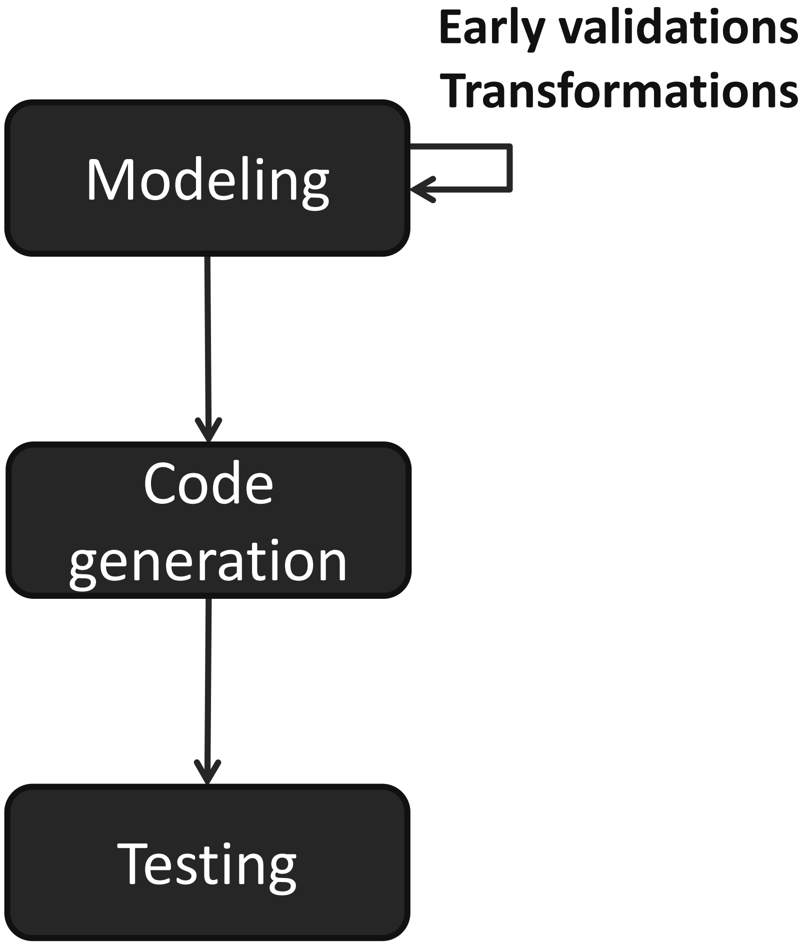
Transformations

Modeling

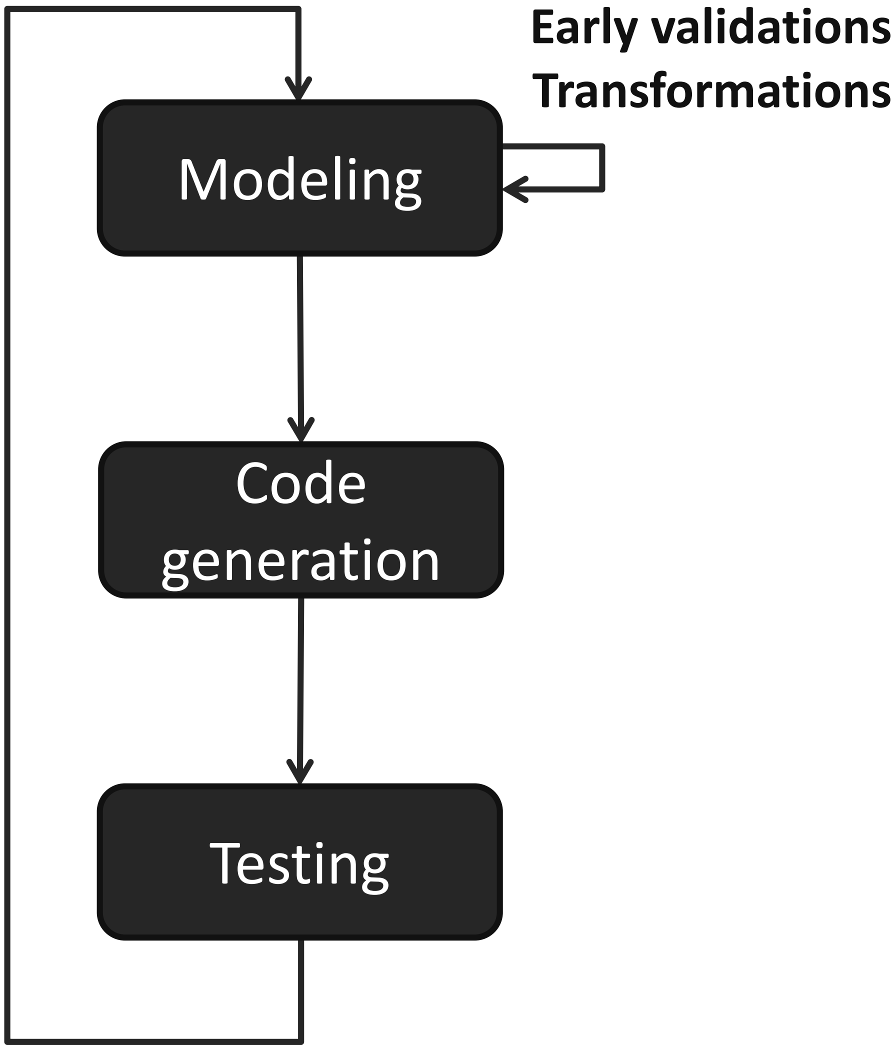
Model-Driven Engineering



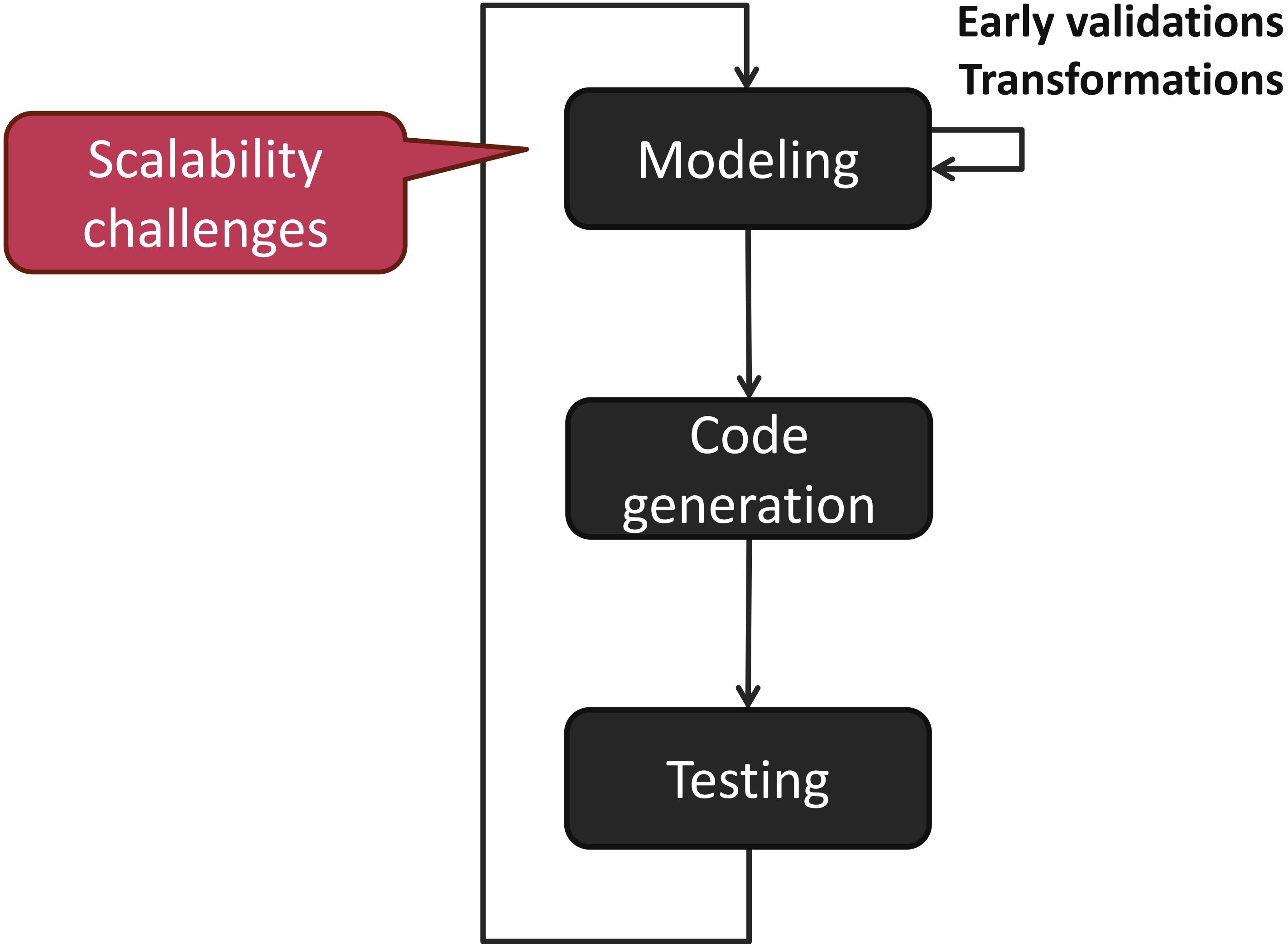
Model-Driven Engineering



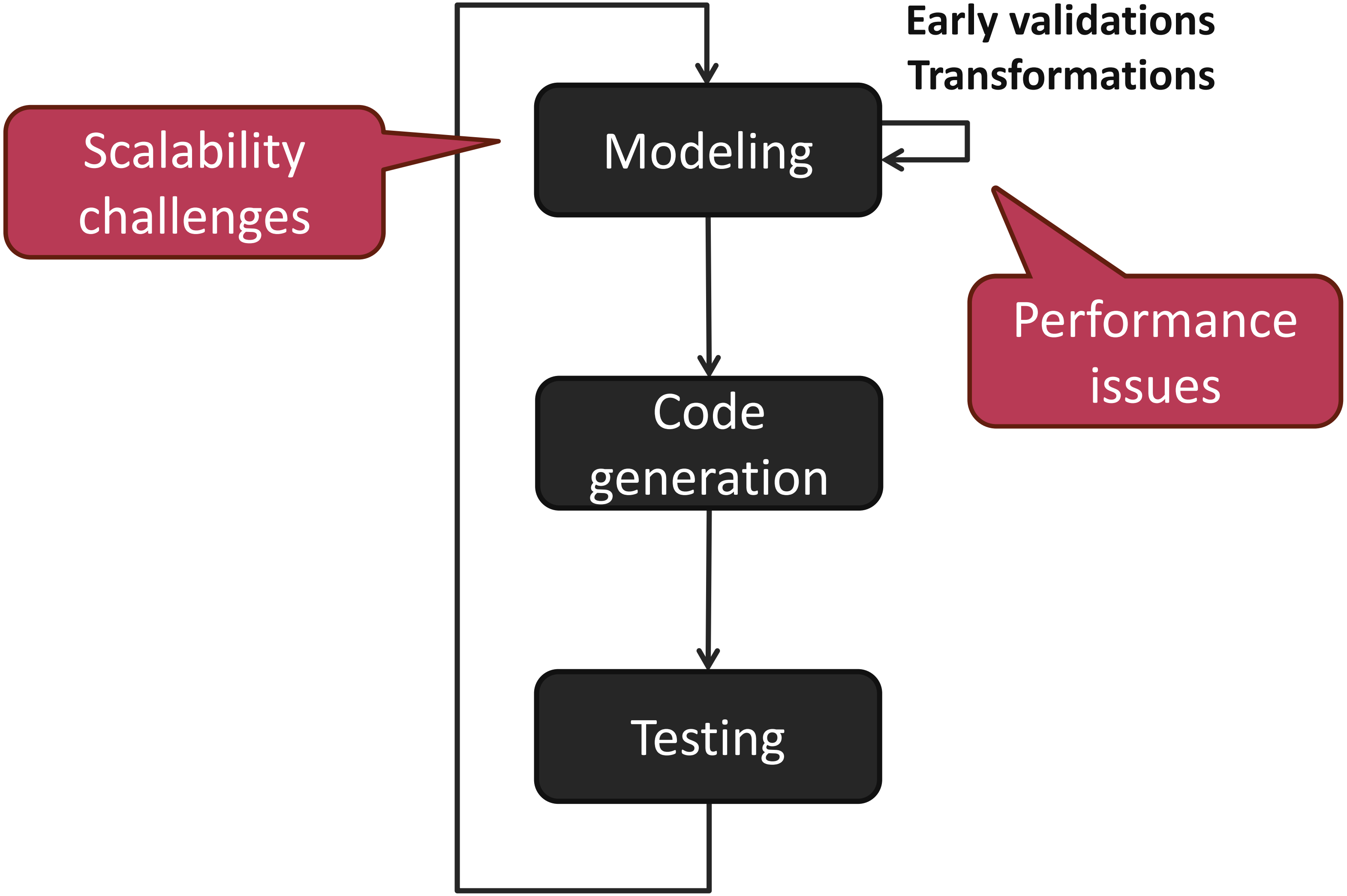
Model-Driven Engineering



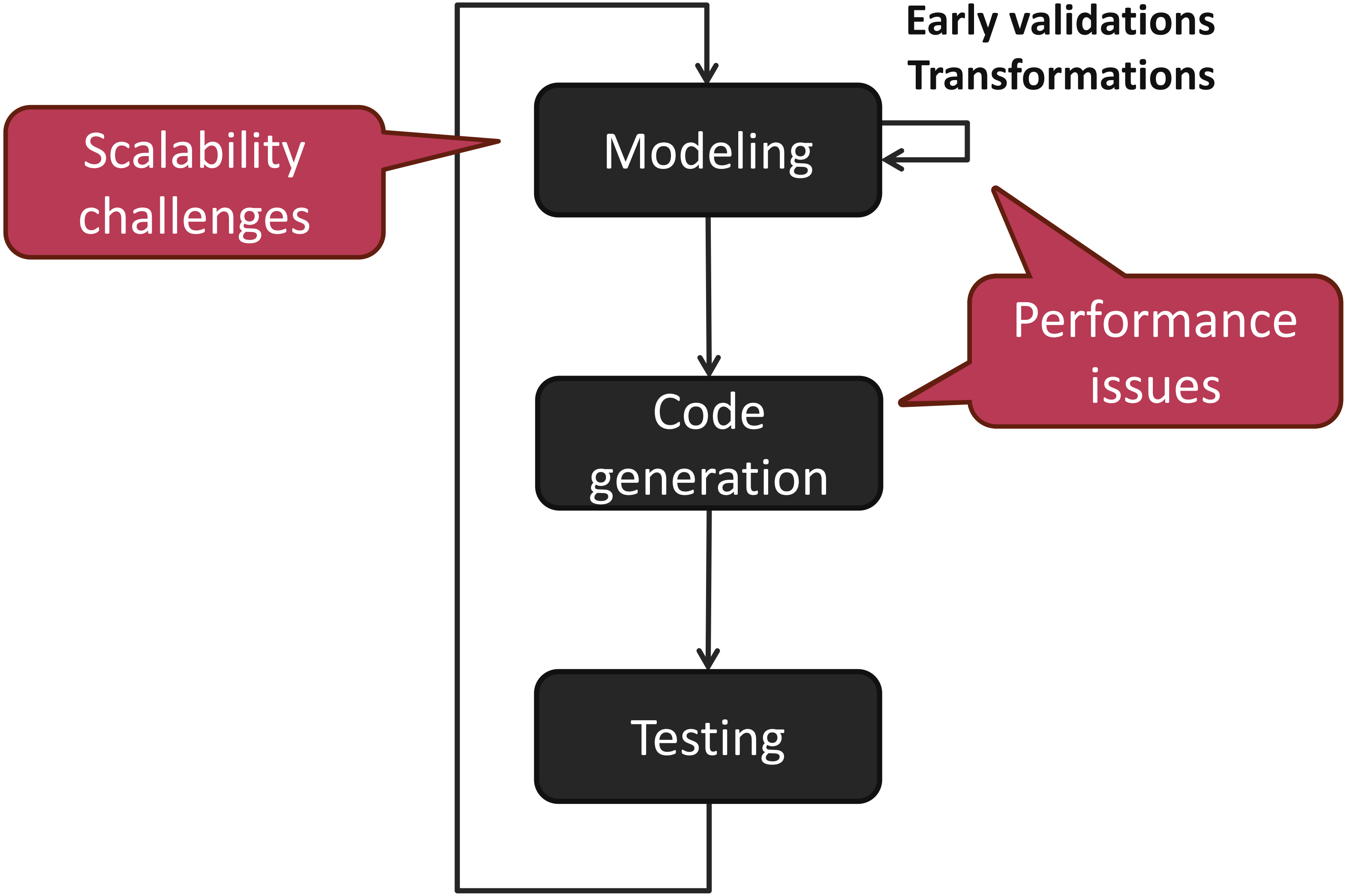
Model-Driven Engineering



Model-Driven Engineering



Model-Driven Engineering



Model Sizes

- **Models = graphs** with 100M–1B elements
  - Car industry
  - Avionics
  - Software analysis
  - Cyber-physical systems

application	model size
software models	10^8
sensor data	10^9
geo-spatial models	10^{12}

Source: Markus Scheidgen, *Automated and Transparent Model Fragmentation for Persisting Large Models*, 2012

Model Sizes

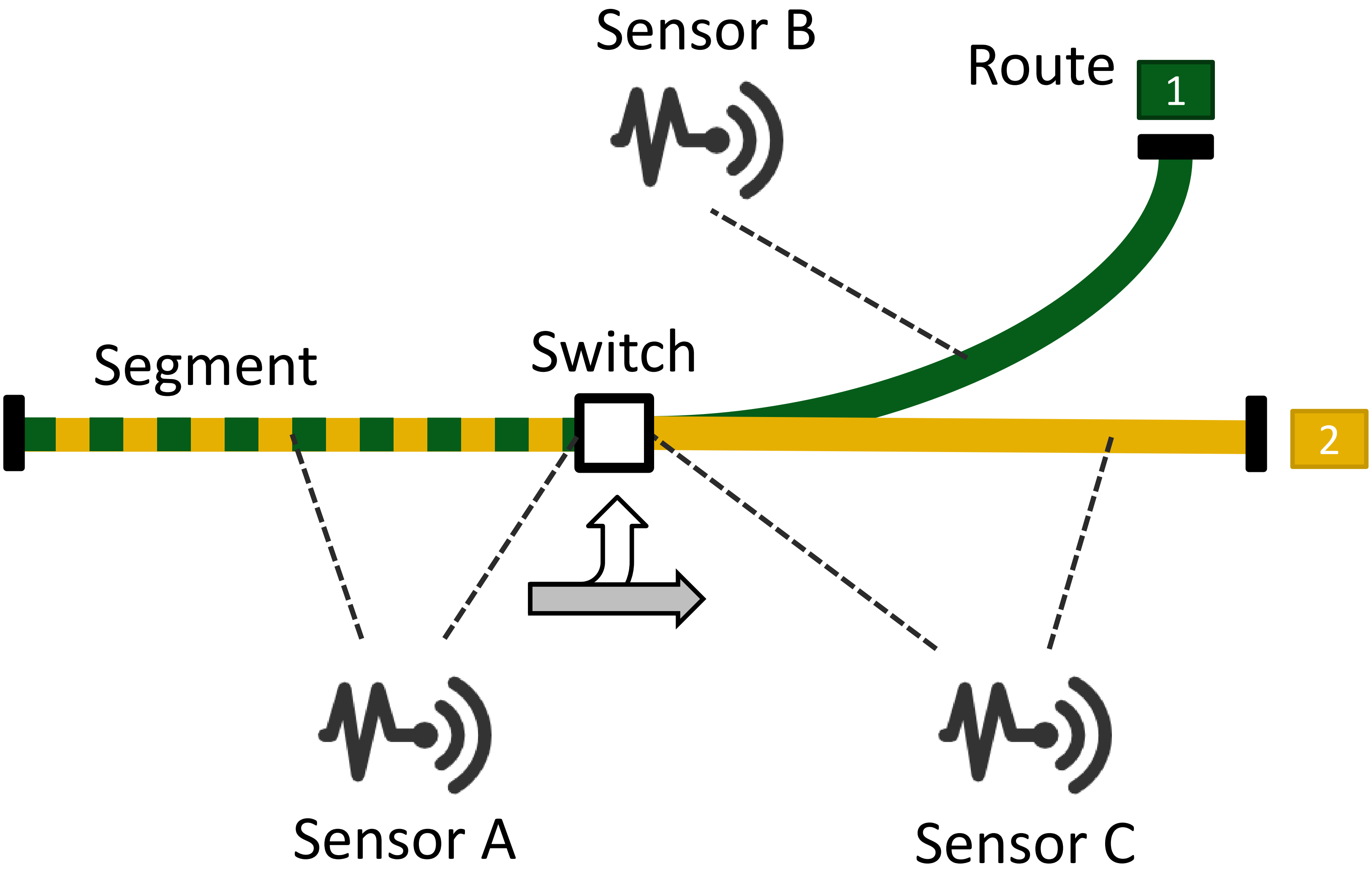
- **Models = graphs** with 100M–1B elements
  - Car industry
  - Avionics
  - Software analysis
  - Cyber-physical systems

Validation may take hours

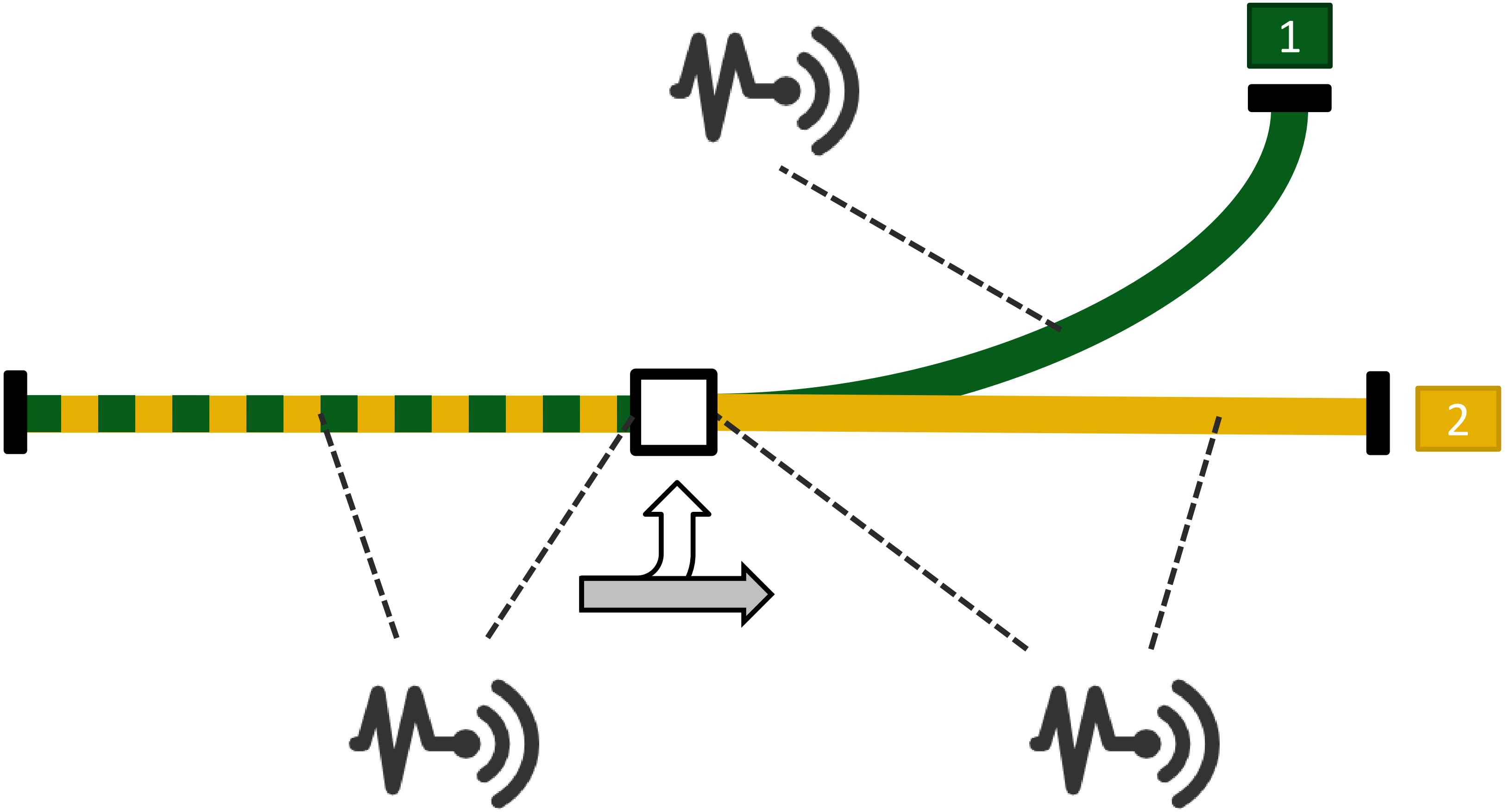
application	model s
software models	10^8
sensor data	10^9
geo-spatial models	10^{12}

Source: Markus Scheidgen, *Automated and Transparent Model Fragmentation for Persisting Large Models*, 2012

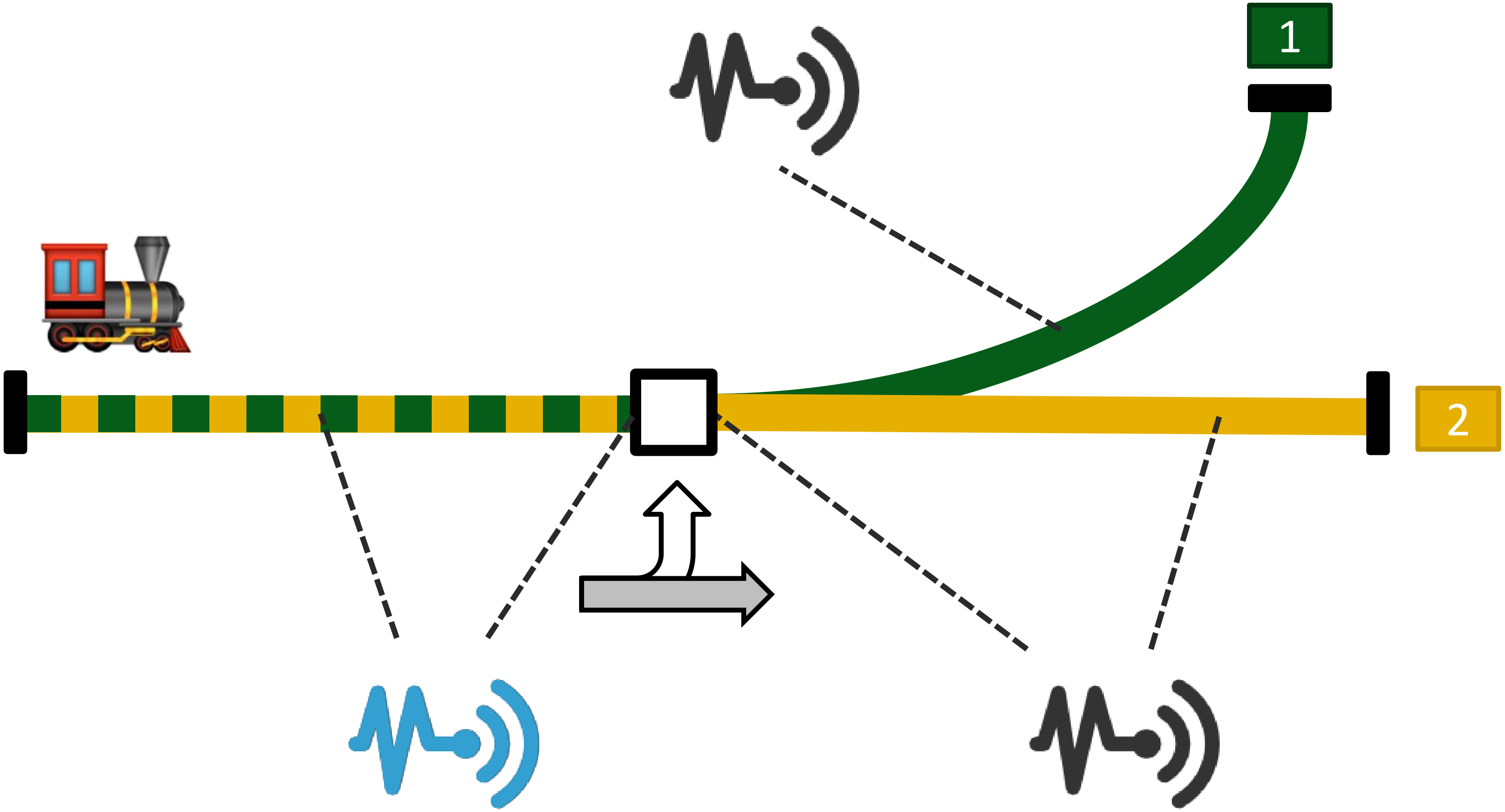
Railway example



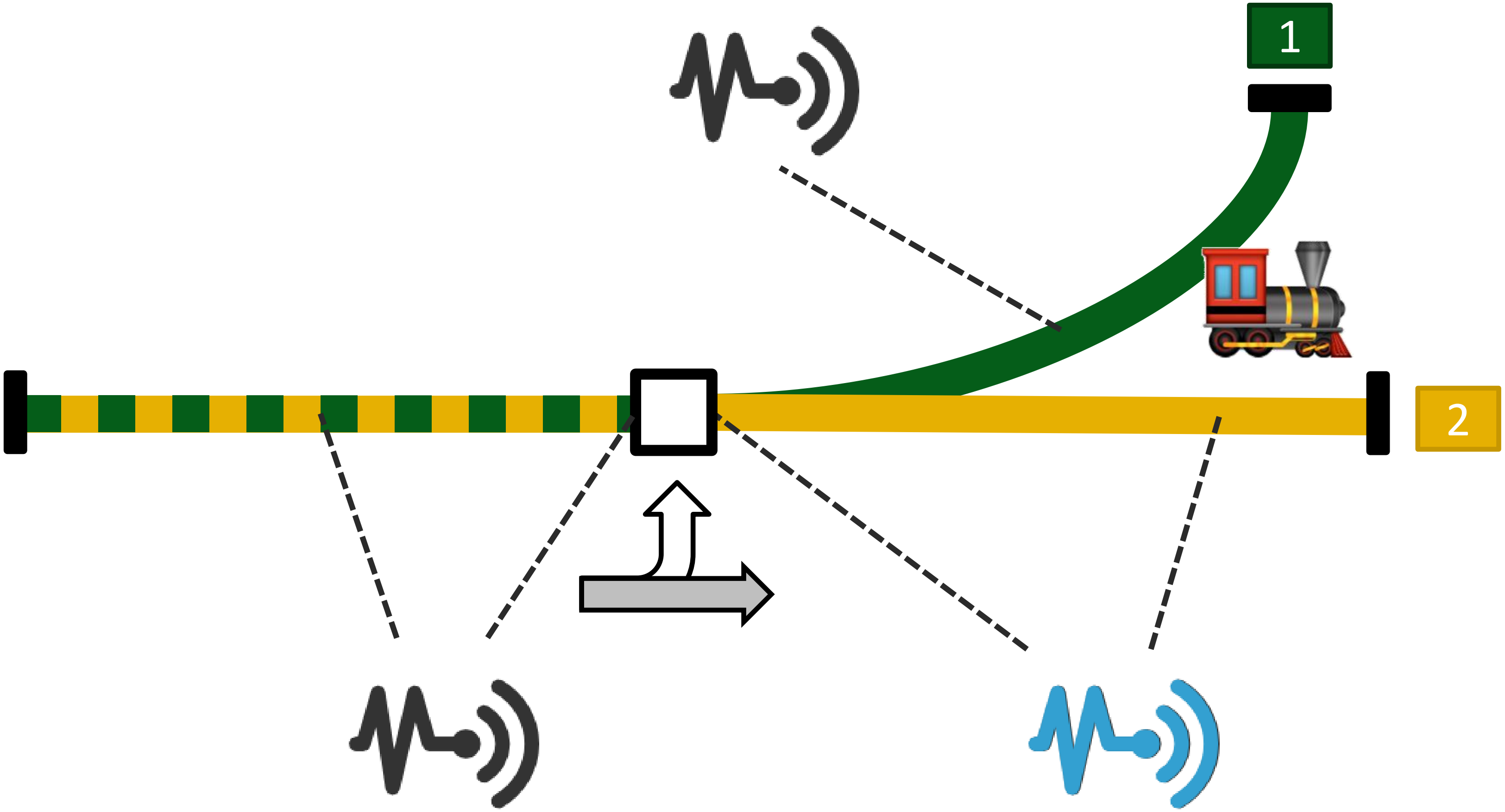
Railway example



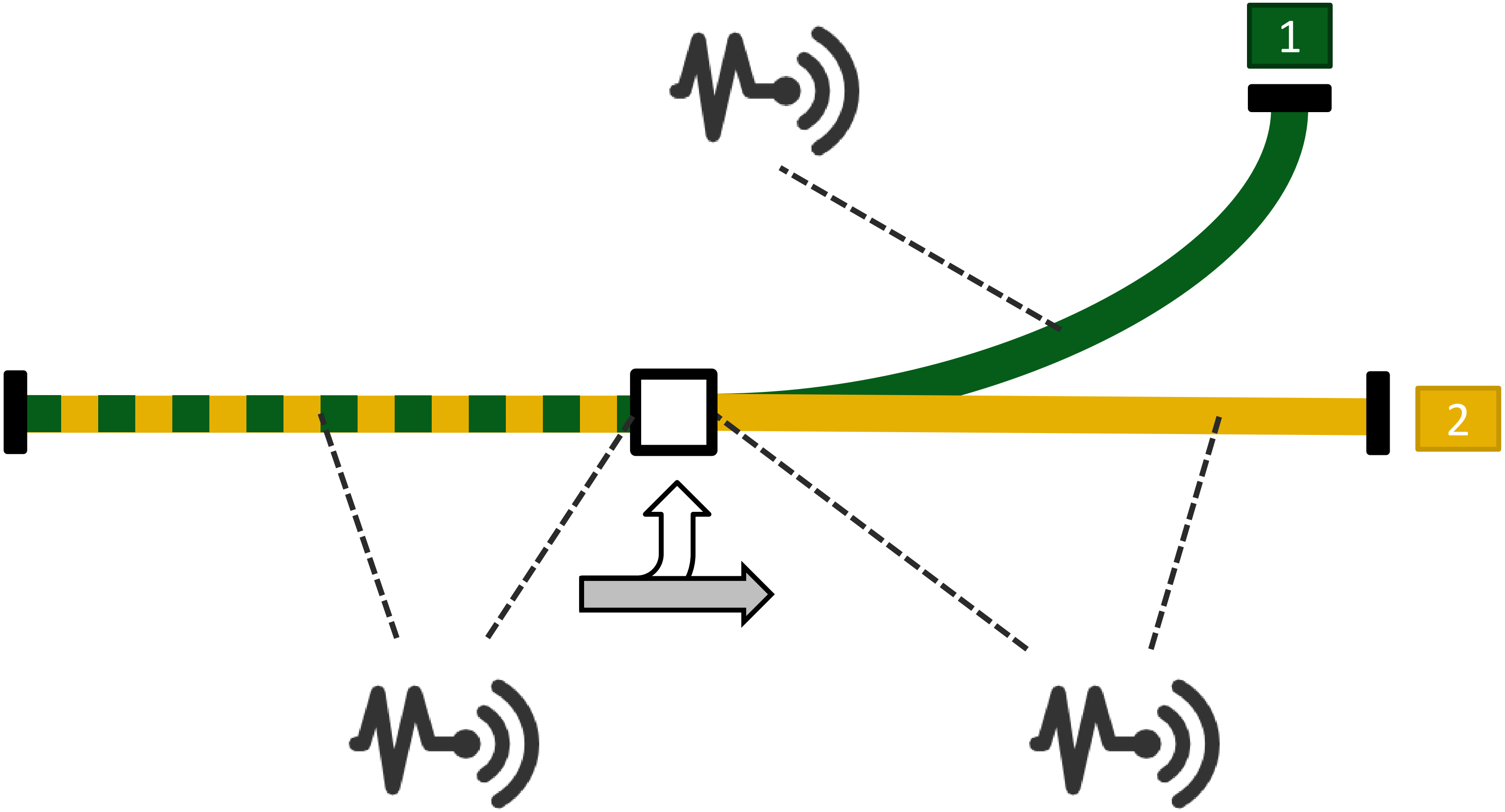
Railway example



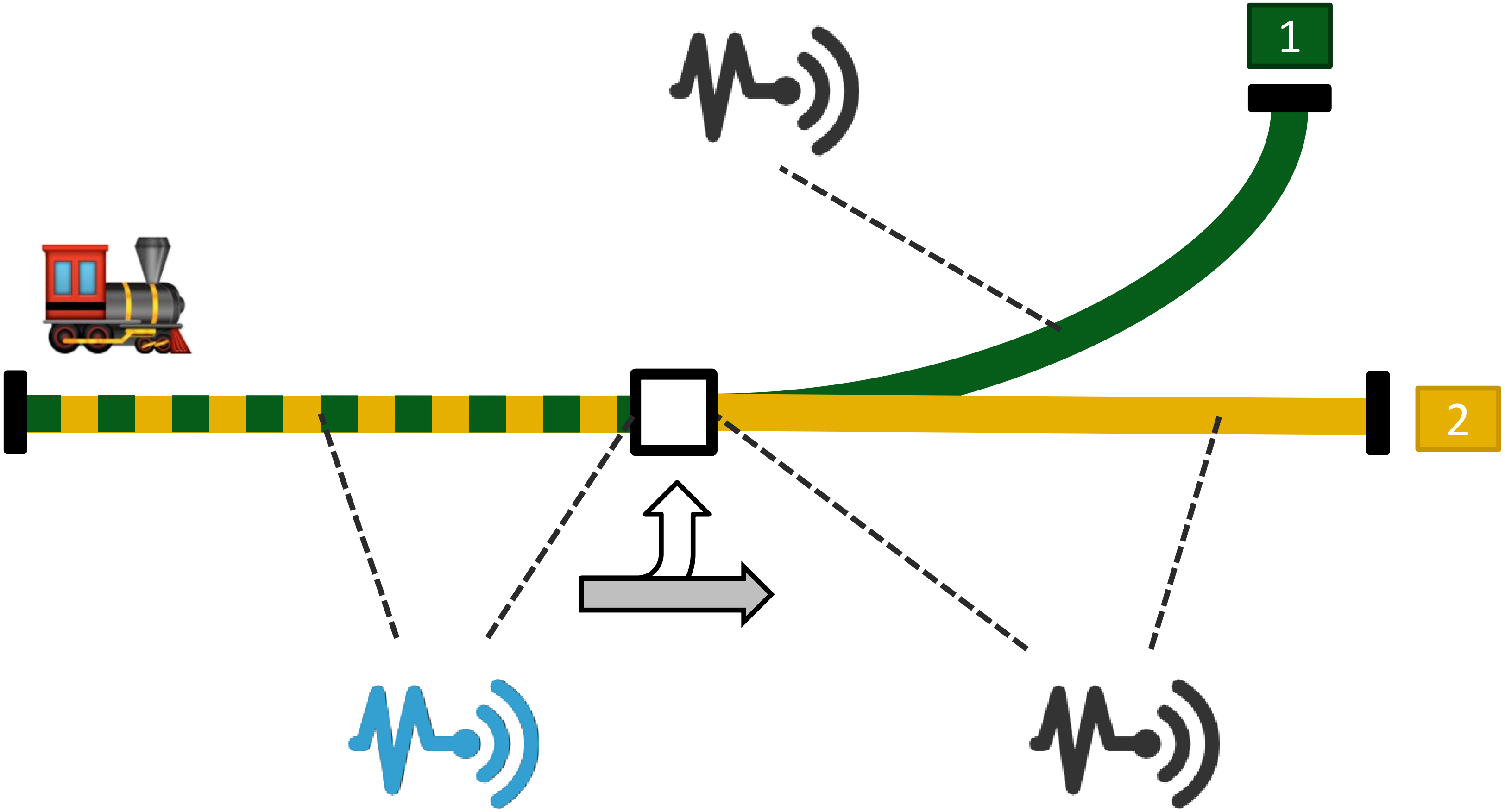
Railway example



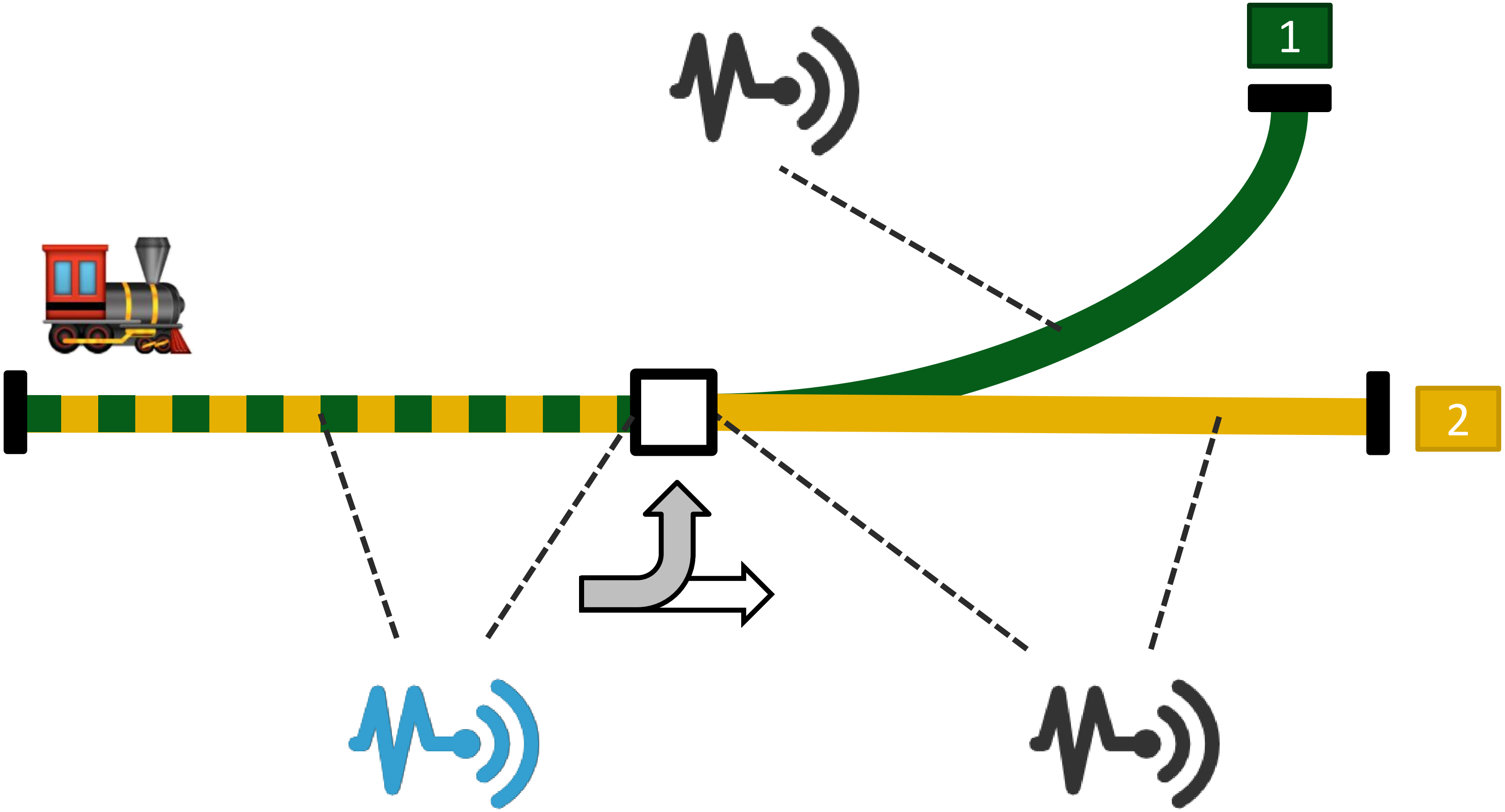
Railway example



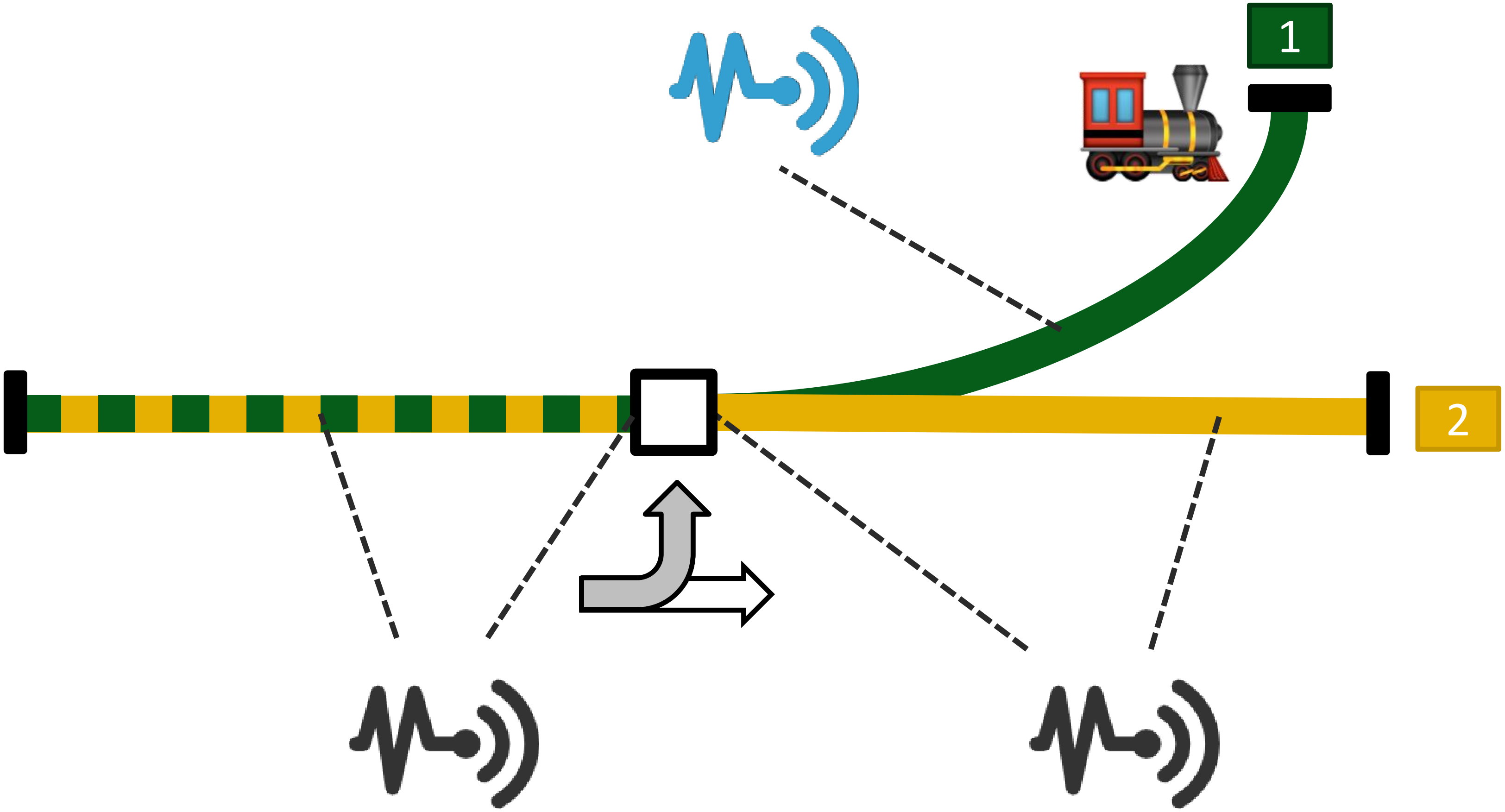
Railway example



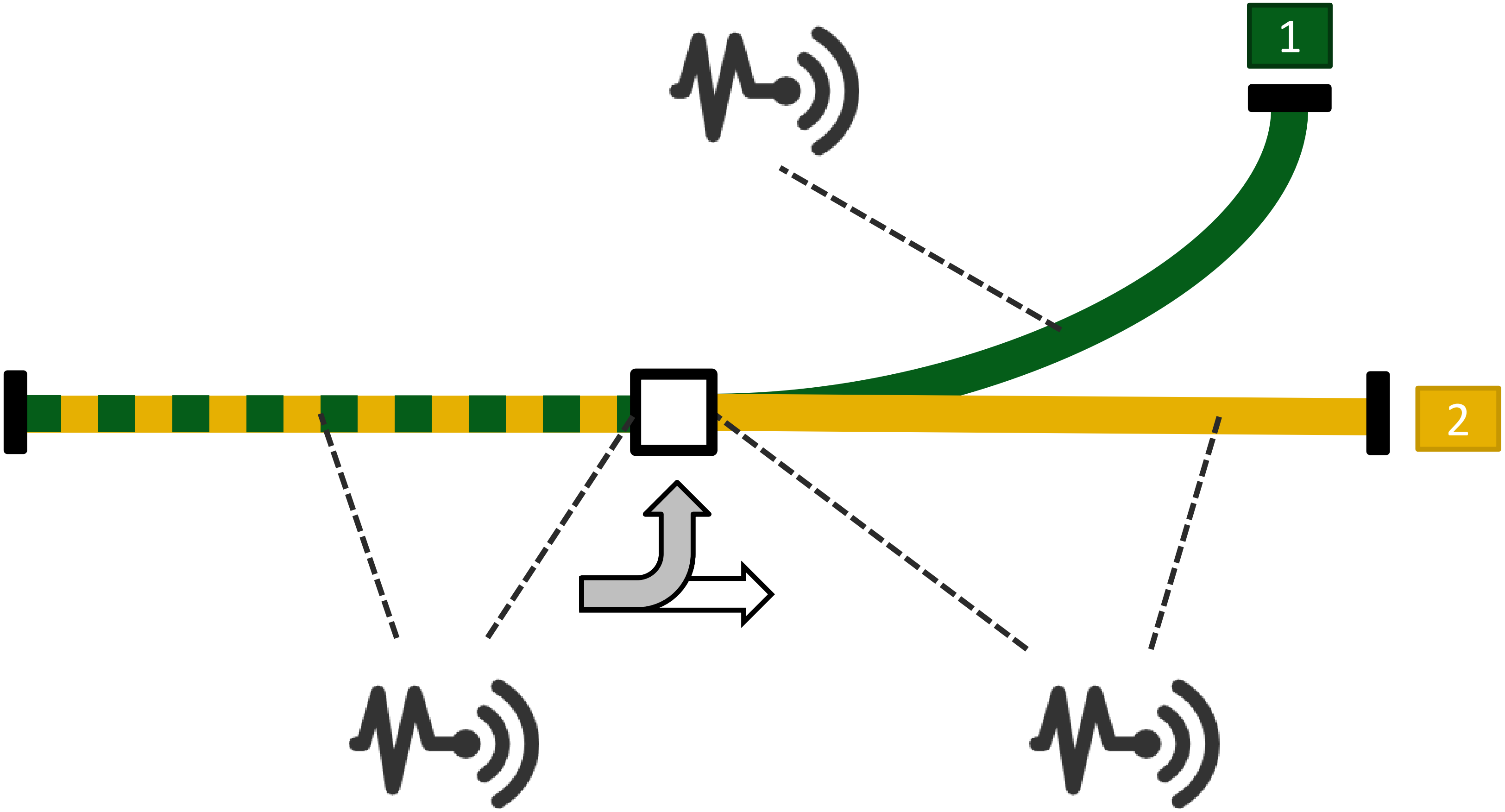
Railway example



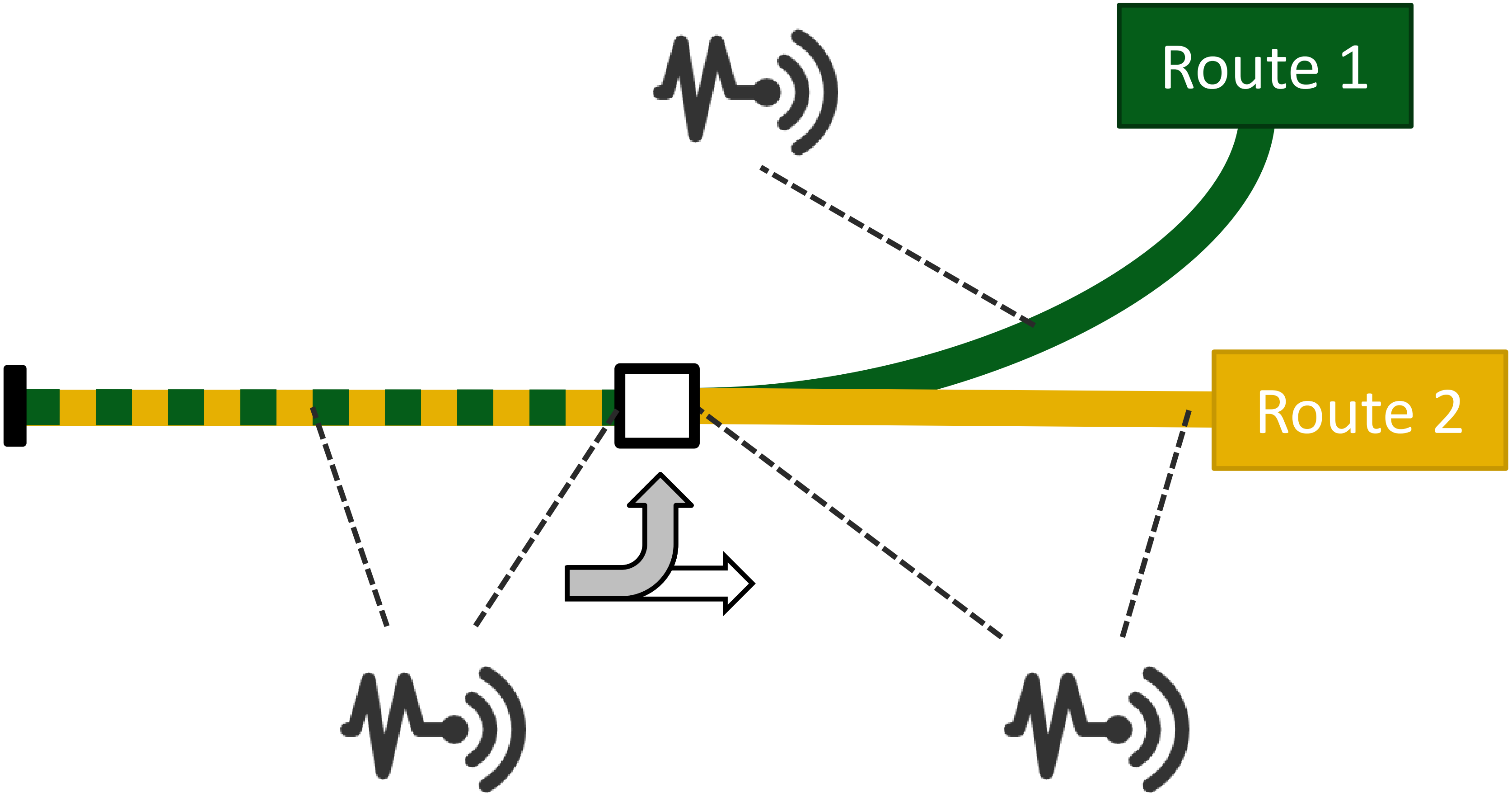
Railway example



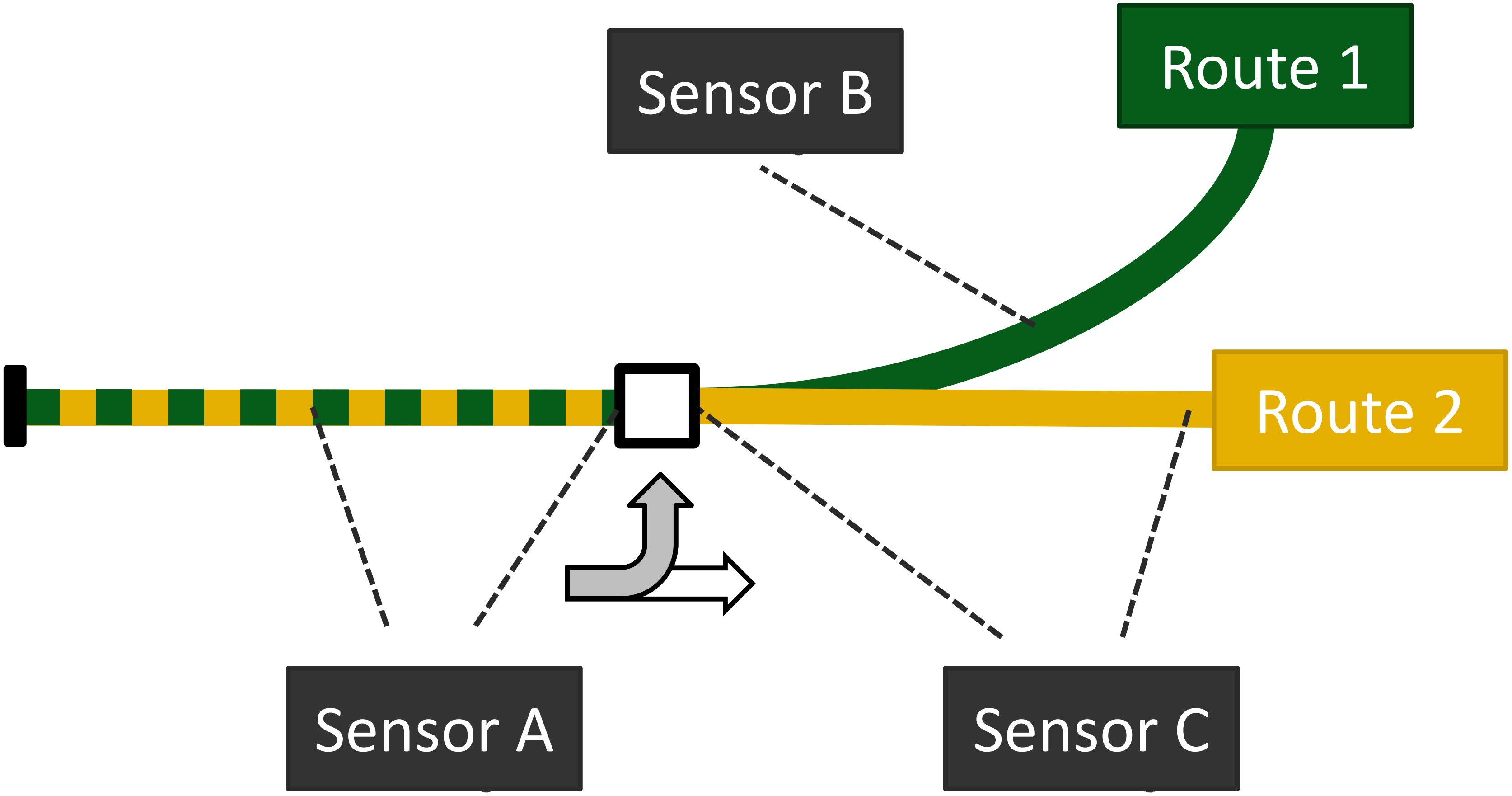
Railway example



Railway example



Railway example



Sensor B

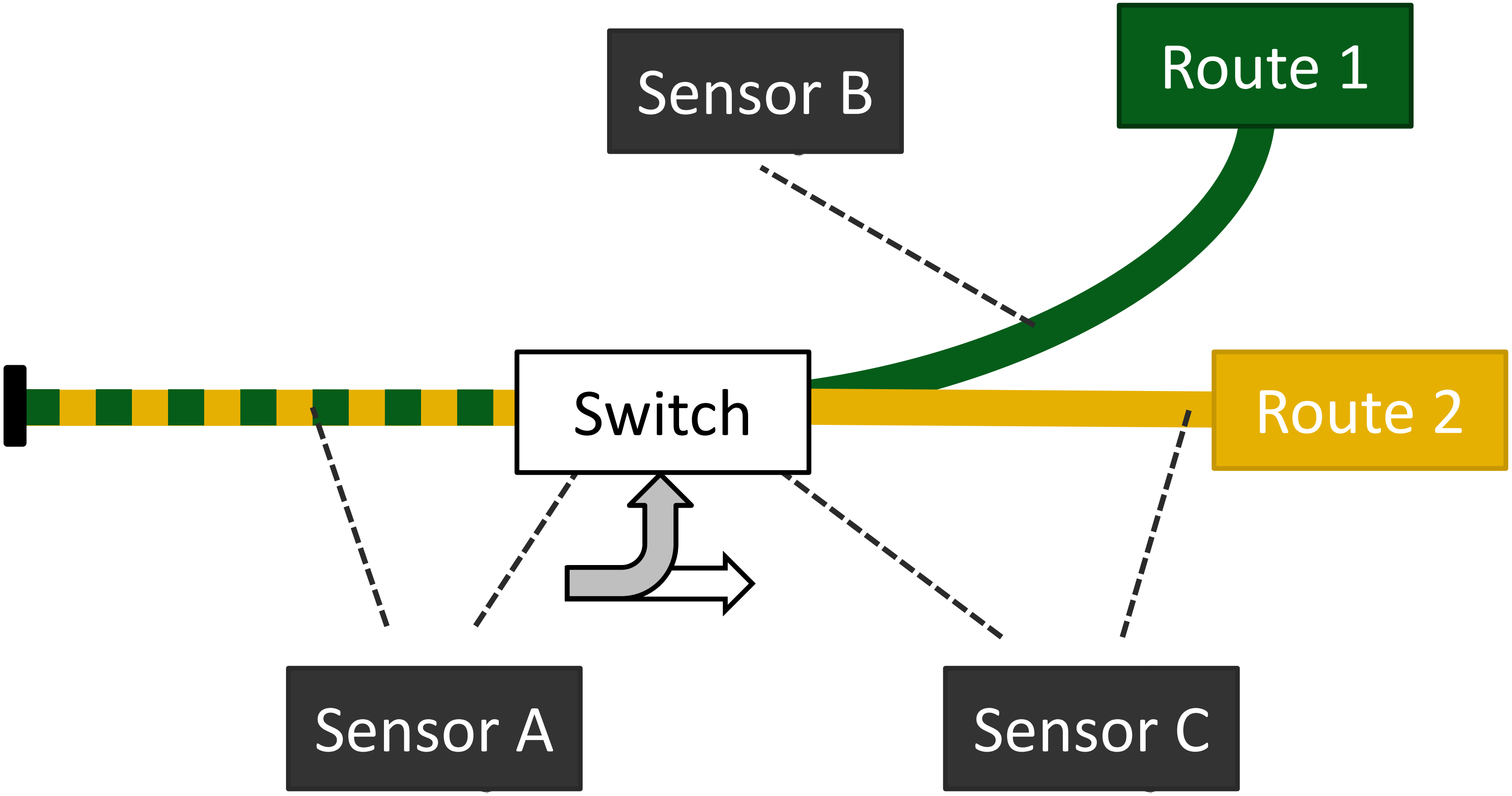
Route 1

Route 2

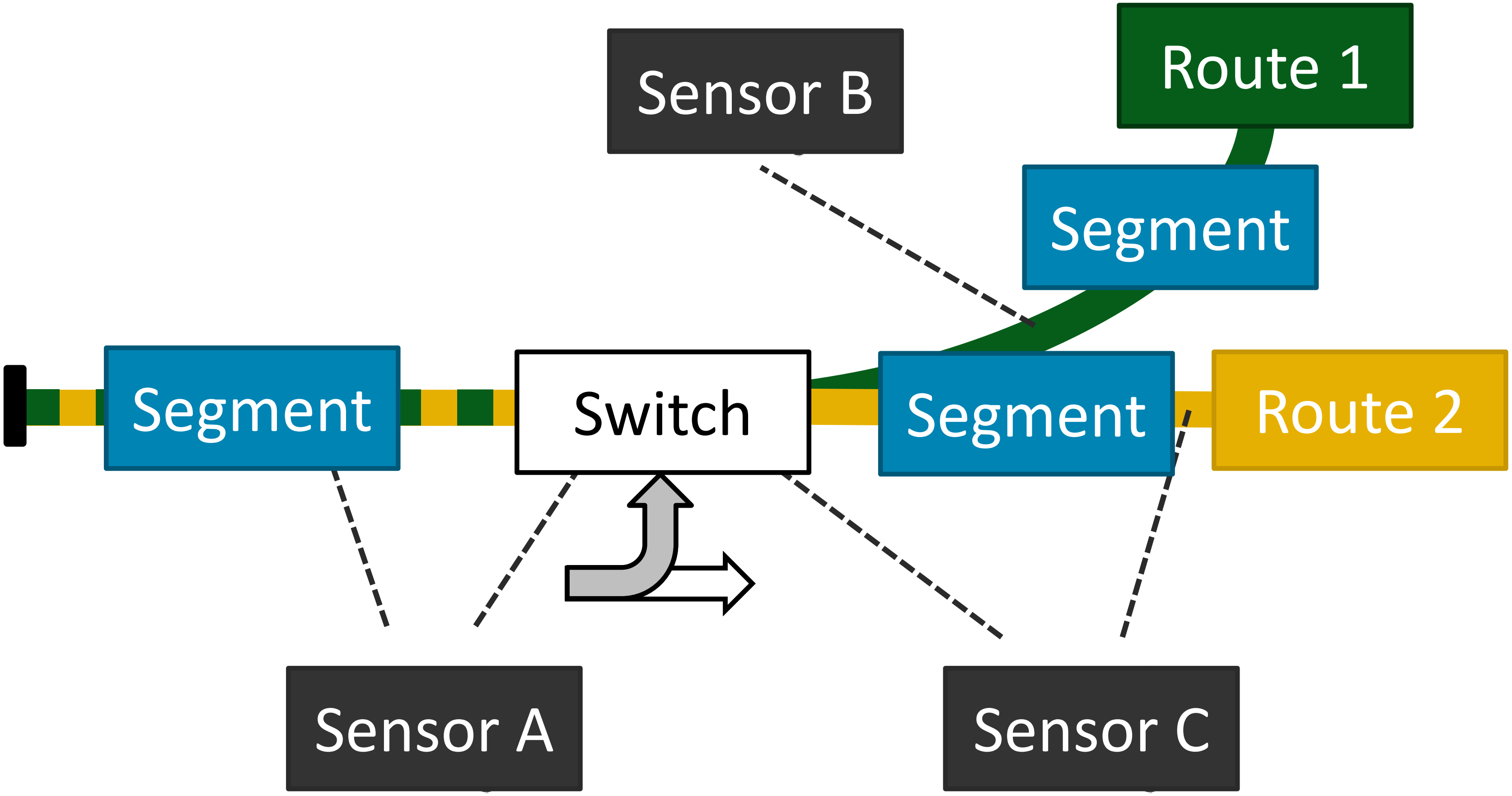
Sensor A

Sensor C

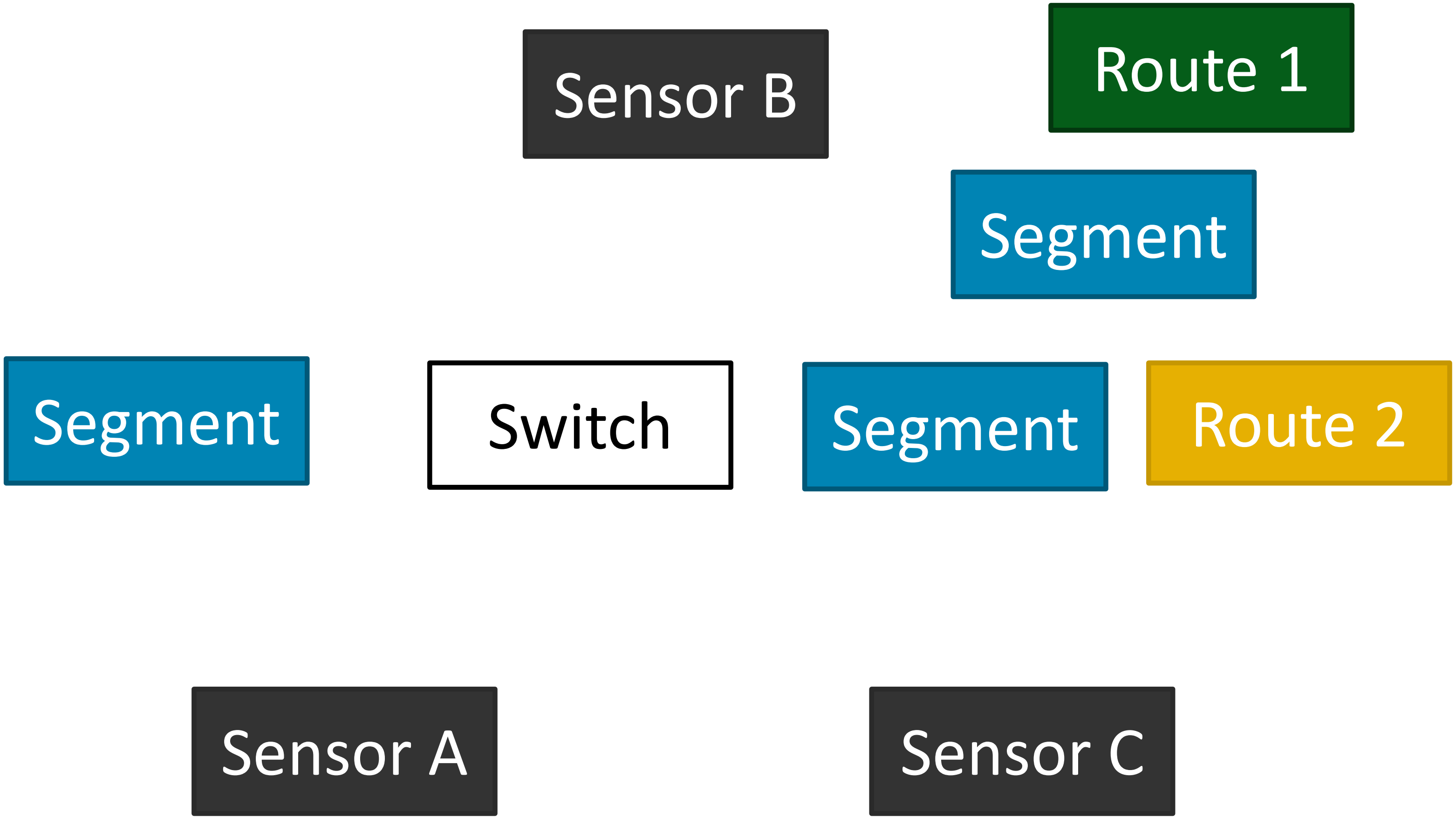
Railway example



Railway example



Railway instance model



Sensor B

Route 1

Segment

Segment

Switch

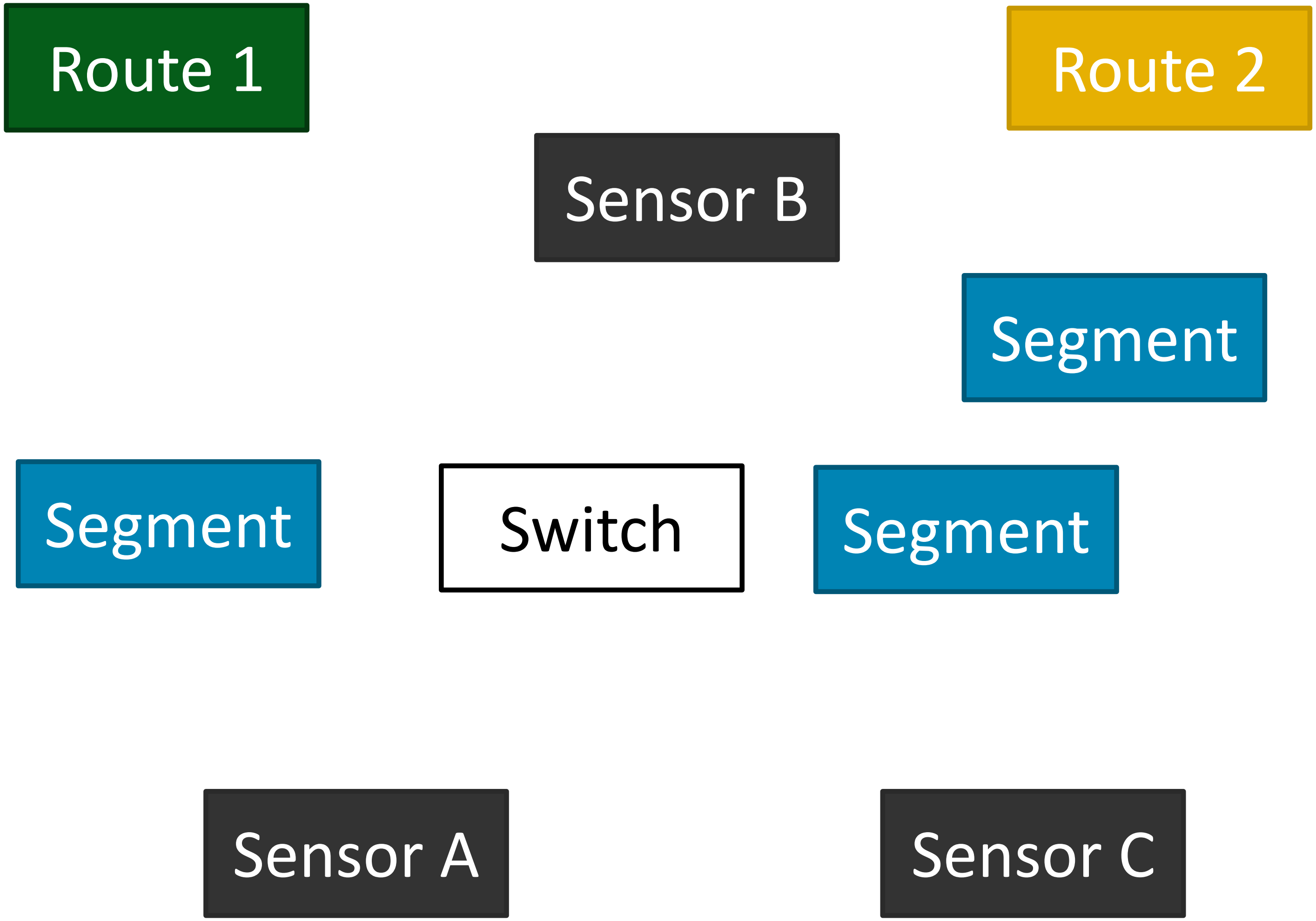
Segment

Route 2

Sensor A

Sensor C

Railway instance model



Route 1

Route 2

Sensor B

Segment

Segment

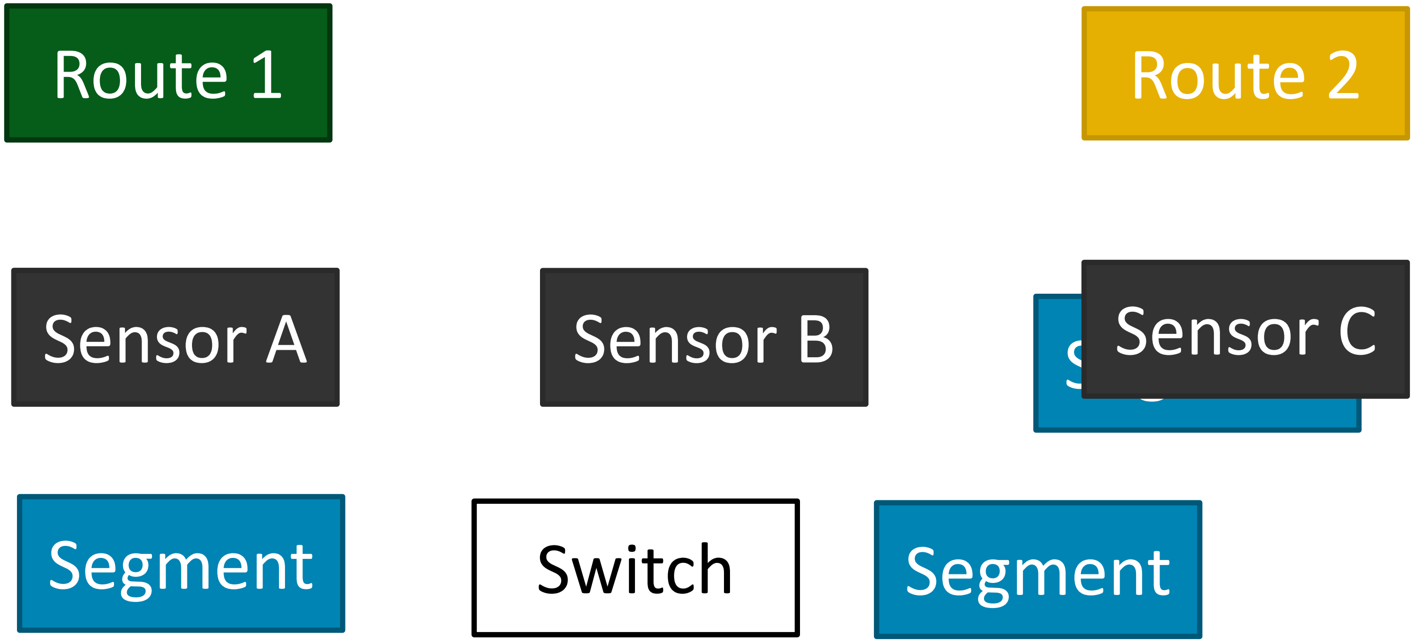
Switch

Segment

Sensor A

Sensor C

Railway instance model



Route 1

Route 2

Sensor A

Sensor B

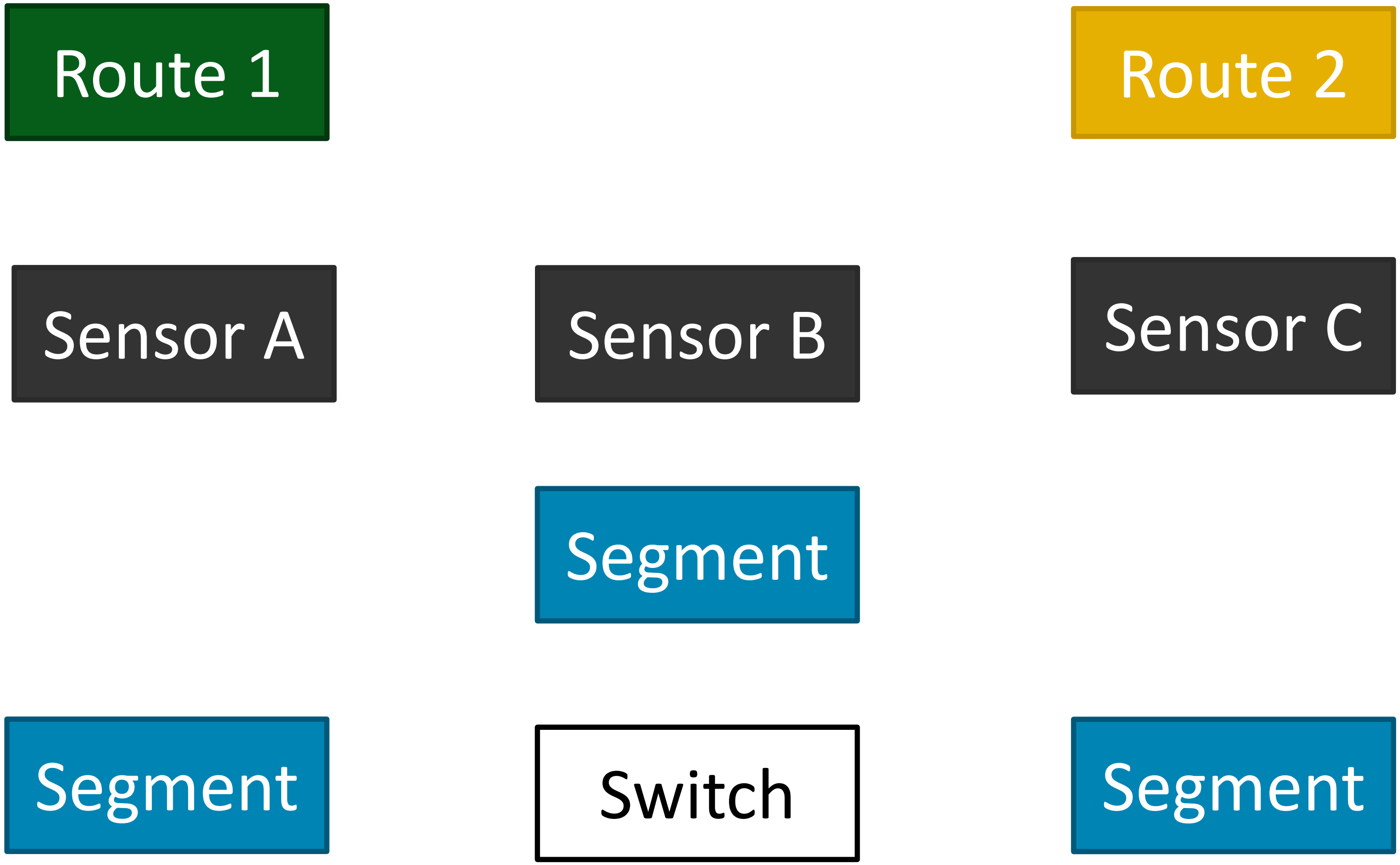
Sensor C

Segment

Switch

Segment

Railway instance model



Route 1

Route 2

Sensor A

Sensor B

Sensor C

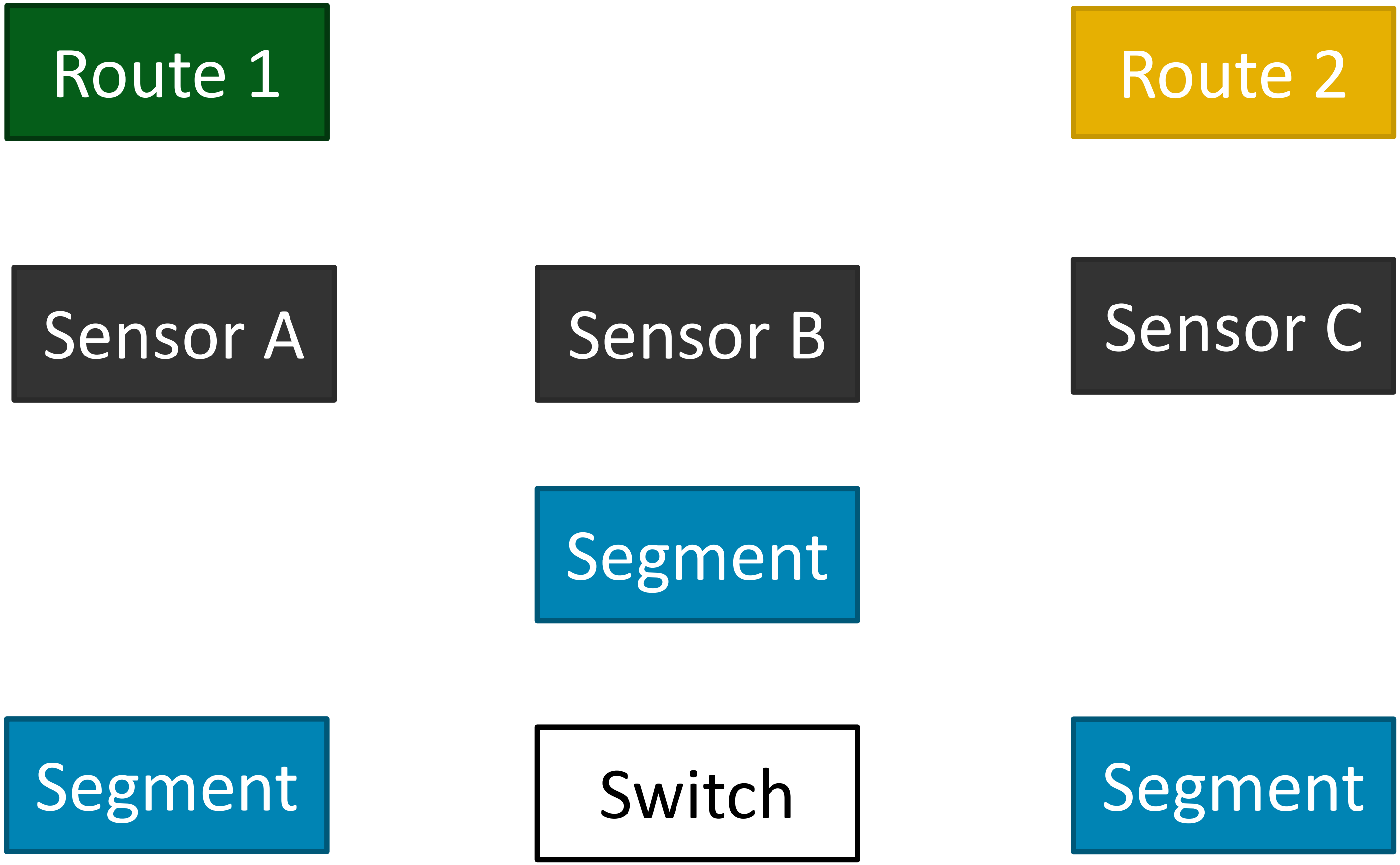
Segment

Segment

Switch

Segment

Railway instance model



Route 1

Route 2

Sensor A

Sensor B

Sensor C

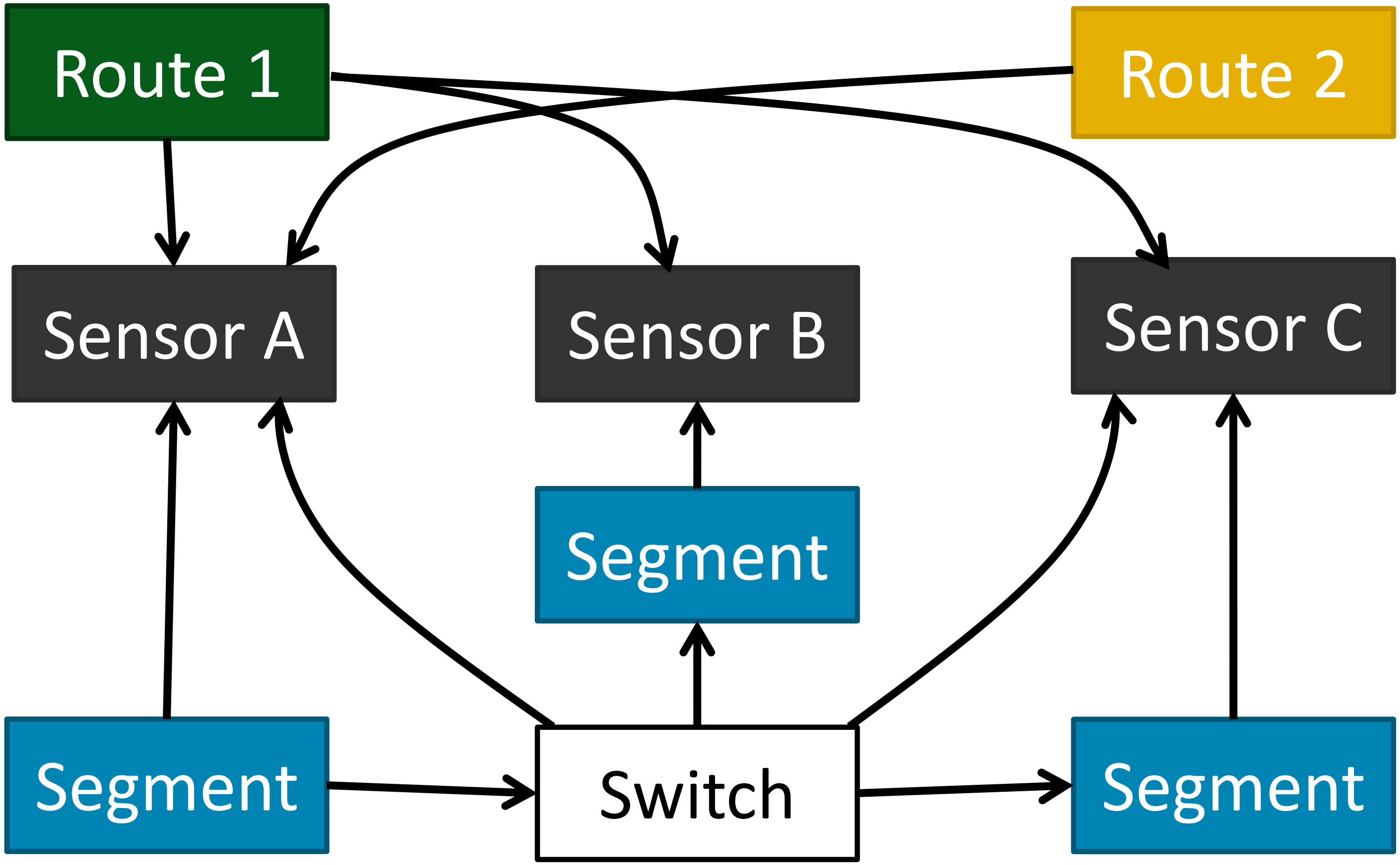
Segment

Segment

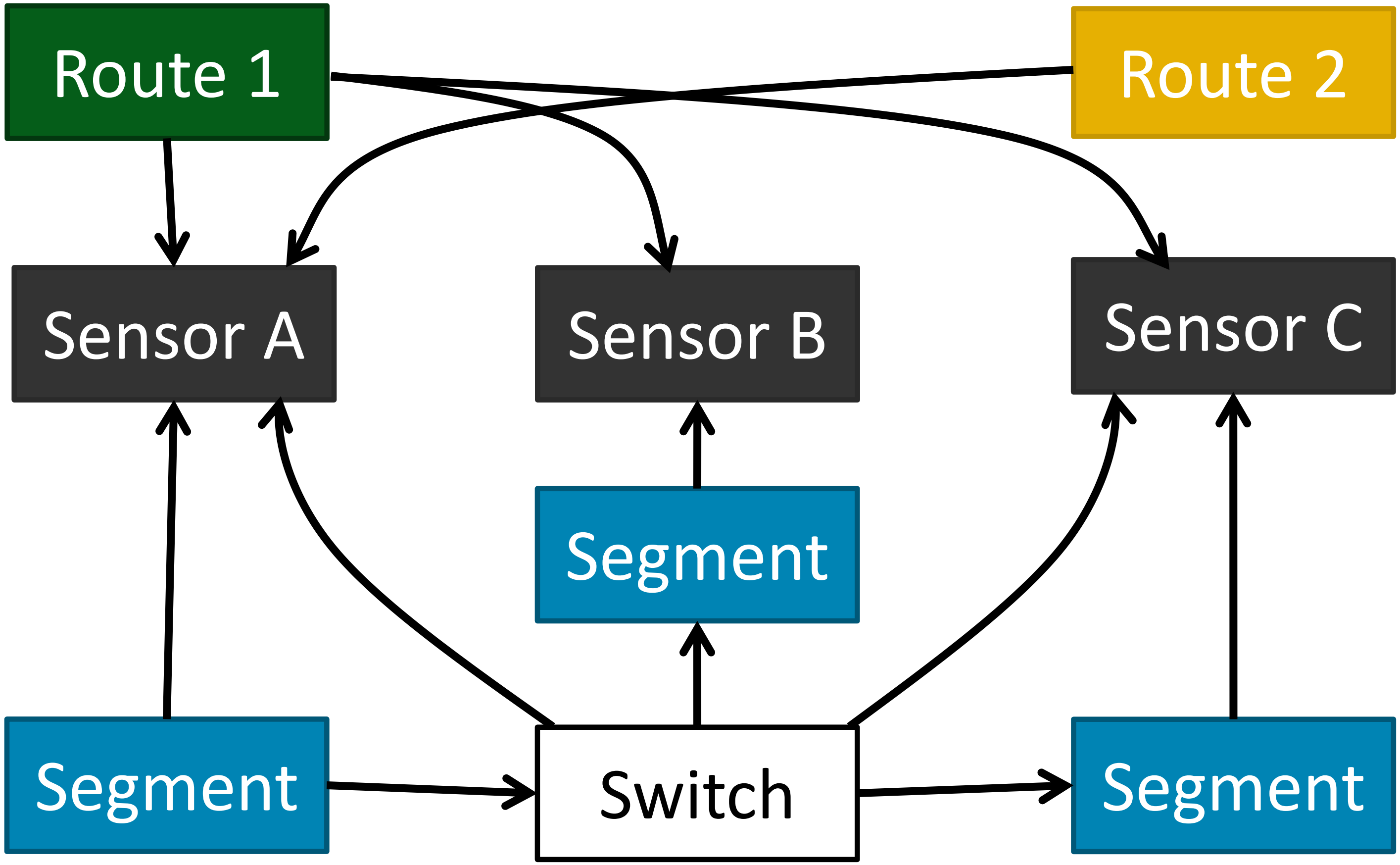
Switch

Segment

Railway instance model



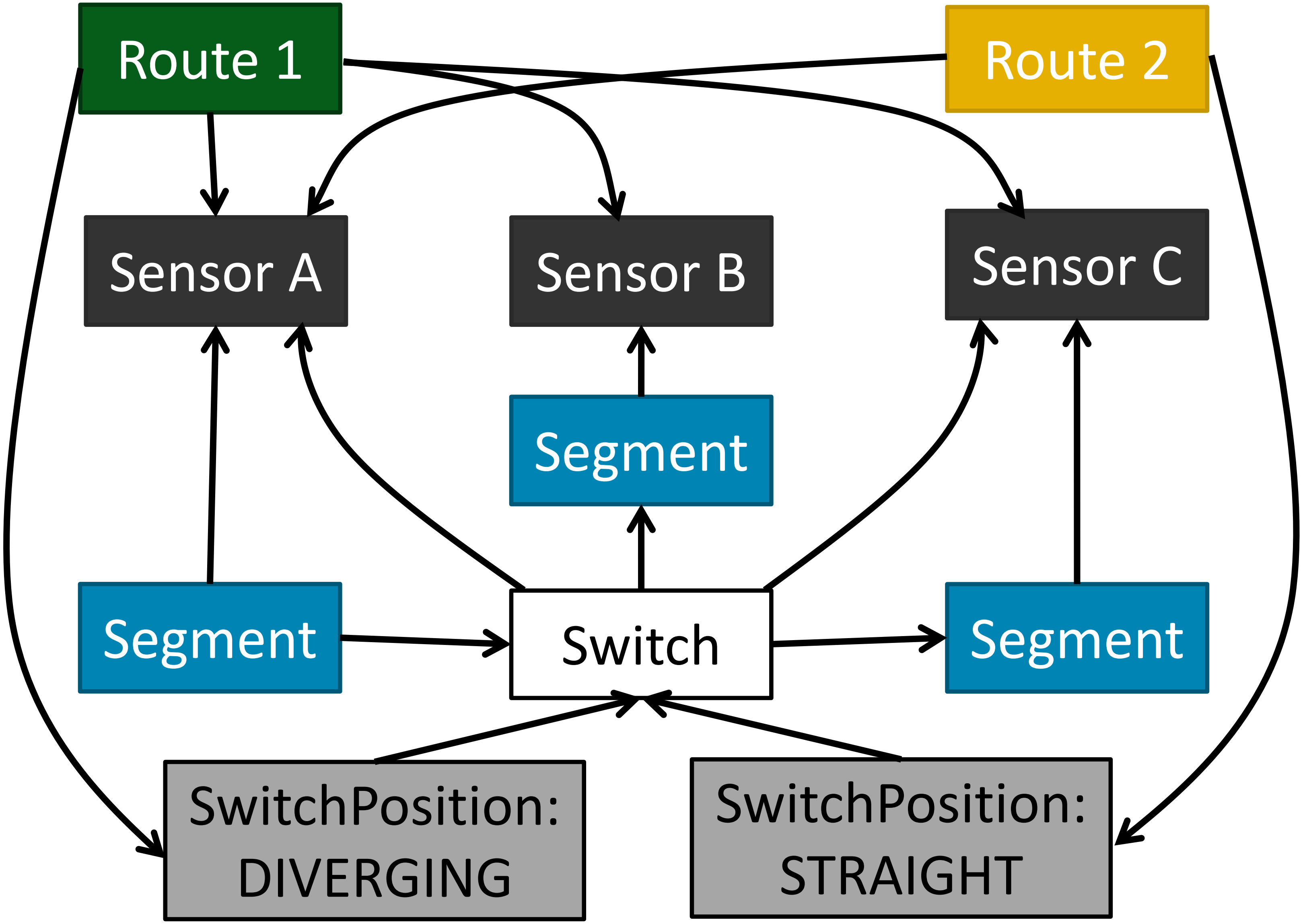
Railway instance model



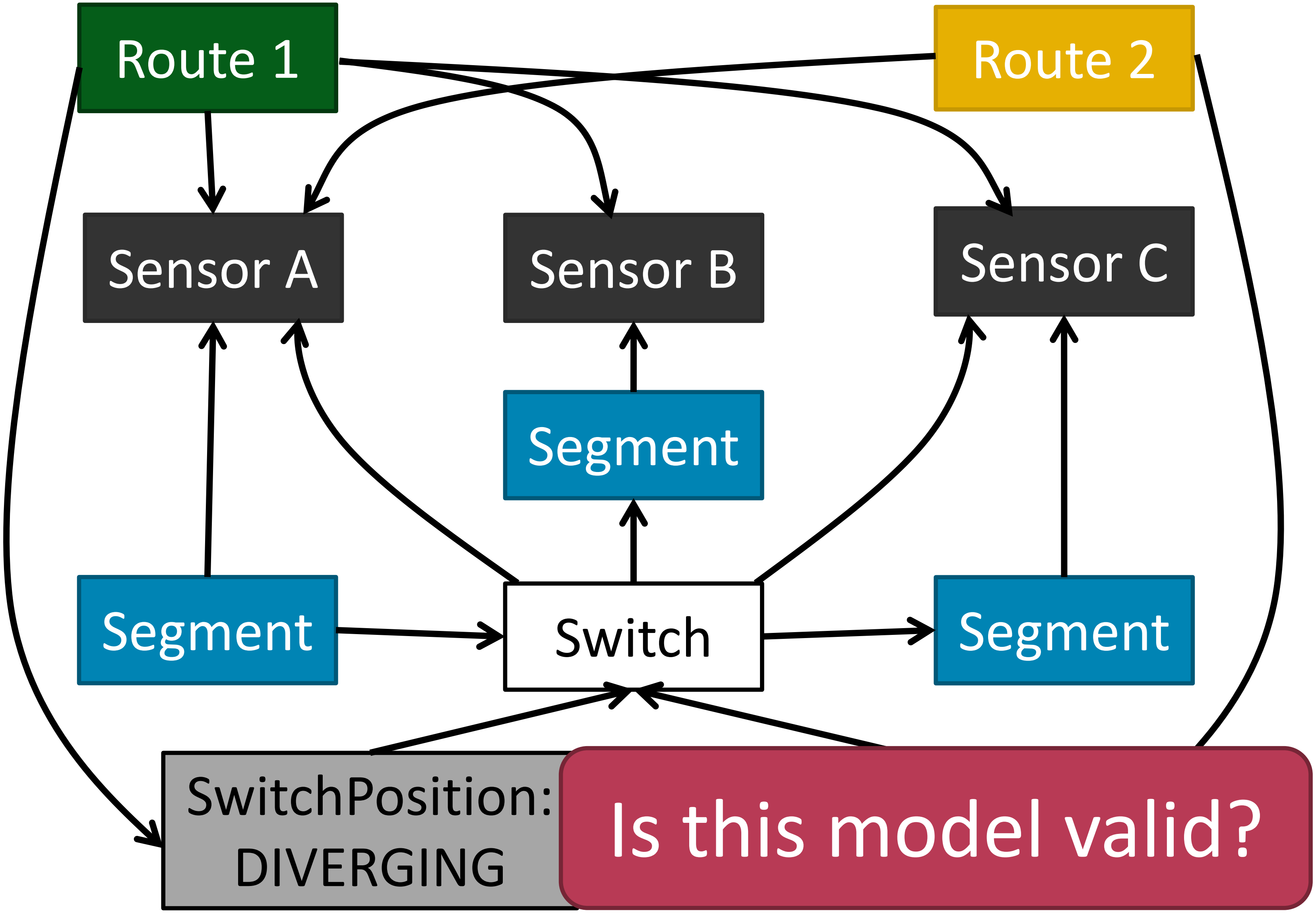
SwitchPosition: DIVERGING

SwitchPosition: STRAIGHT

Railway instance model

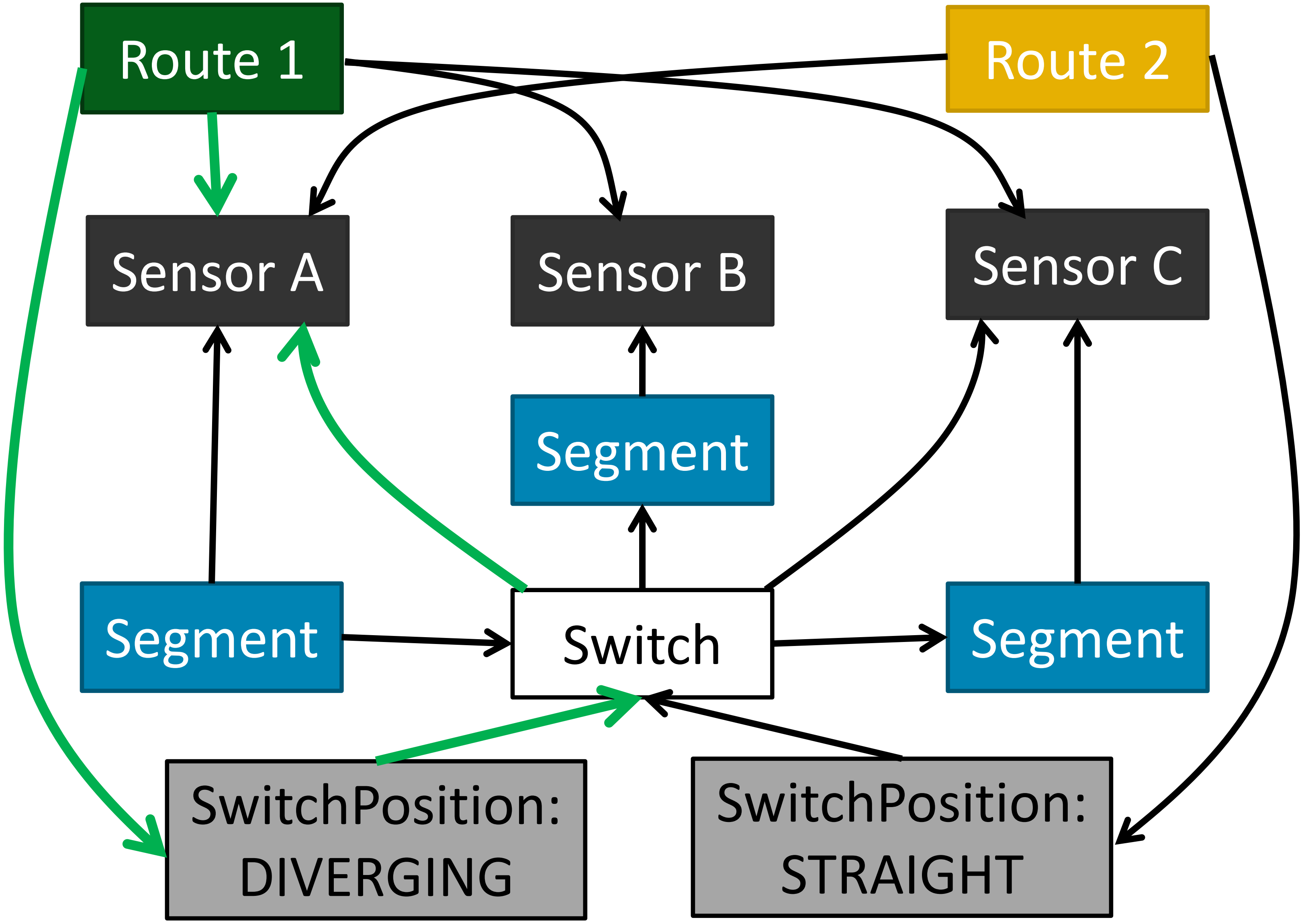


Railway instance model

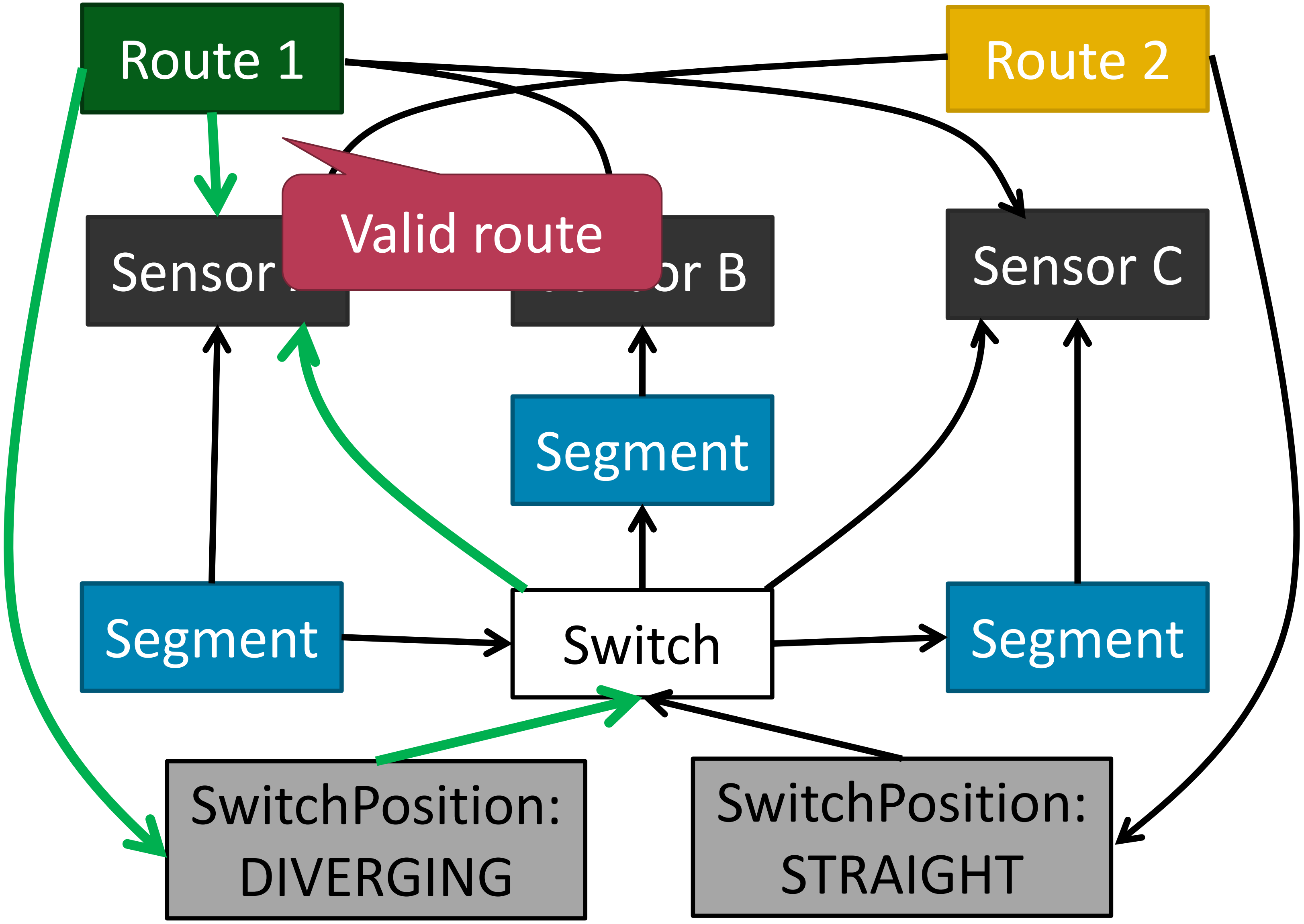


Is this model valid?

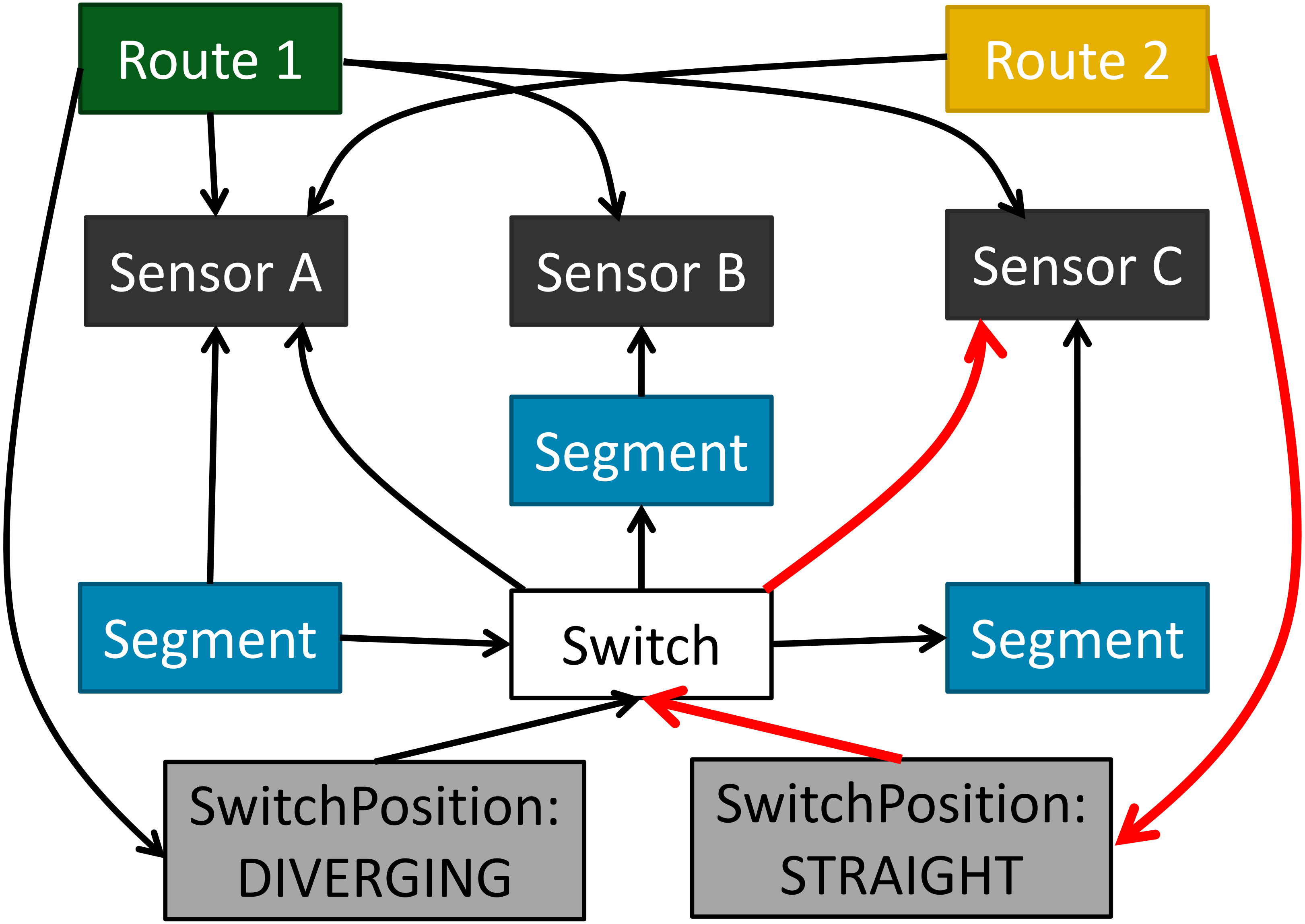
Railway instance model



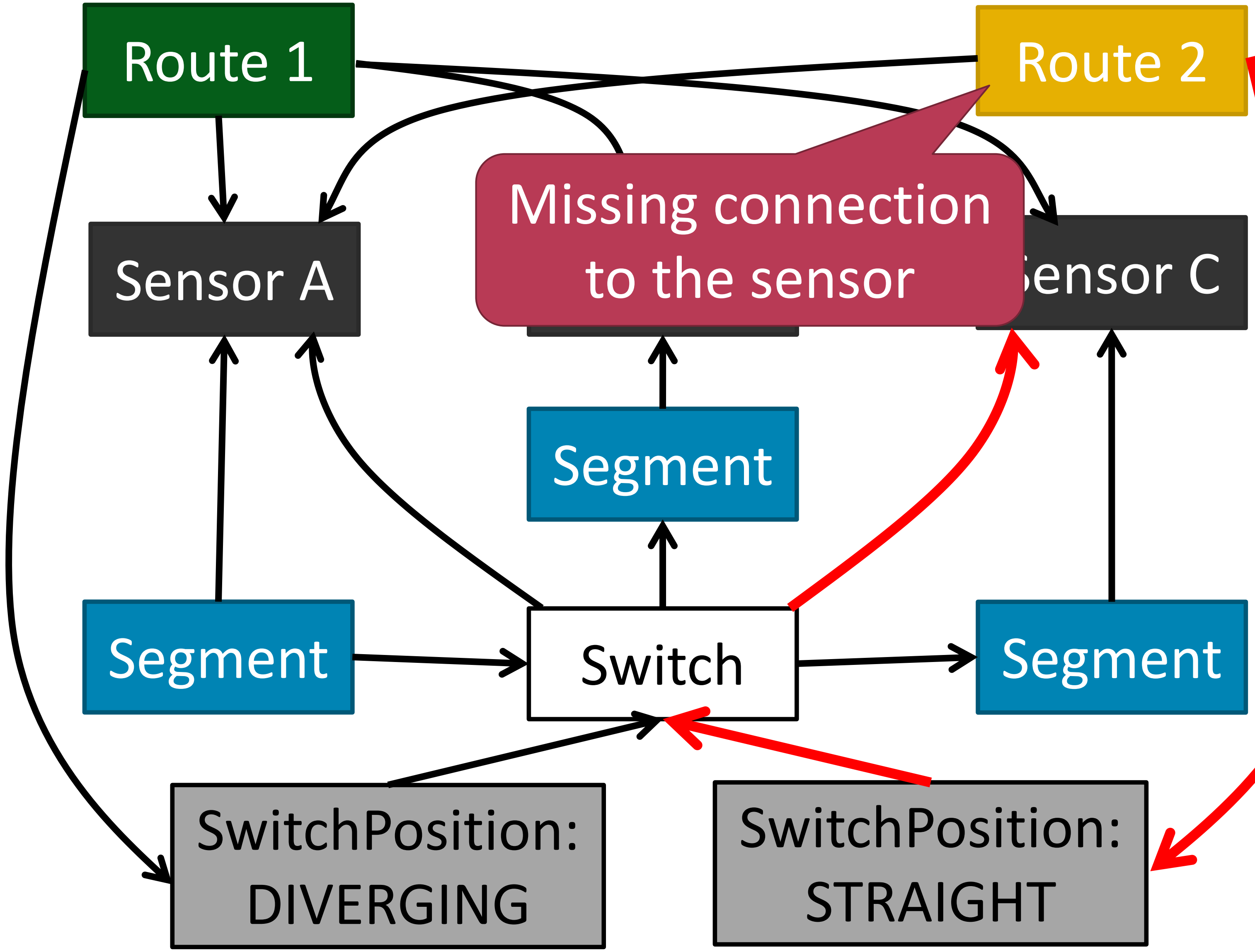
Railway instance model



Railway instance model



Railway instance model



Missing connection to the sensor

Benchmark criteria

Benchmark criteria

Early 1990s:

Benchmark criteria

Early 1990s:

before the „browser wars”

Benchmark criteria

Early 1990s:

Benchmark criteria

Early 1990s:

„benchmark wars”

Benchmark criteria

Early 1990s:

„benchmark wars”

Jim Gray, *Benchmark Handbook*, 1993

Criteria for domain-specific benchmarks:

Benchmark criteria

Early 1990s:

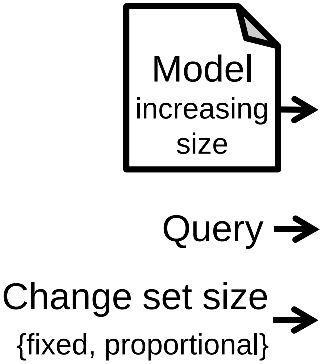
„benchmark wars”

Jim Gray, *Benchmark Handbook*, 1993

Criteria for domain-specific benchmarks:

- Relevant
- Scalable
- Portable
- Simple

Benchmark phases

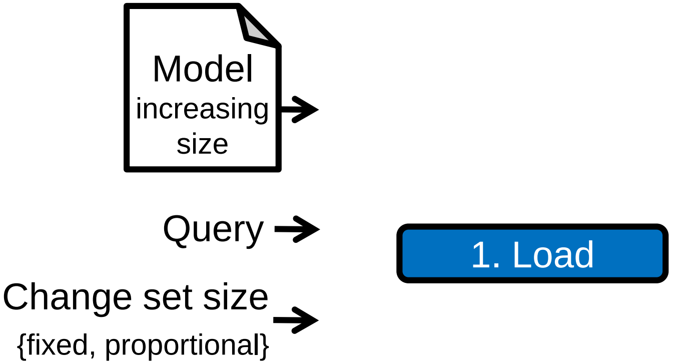


Model
increasing size

Query

Change set size
{fixed, proportional}

Benchmark phases



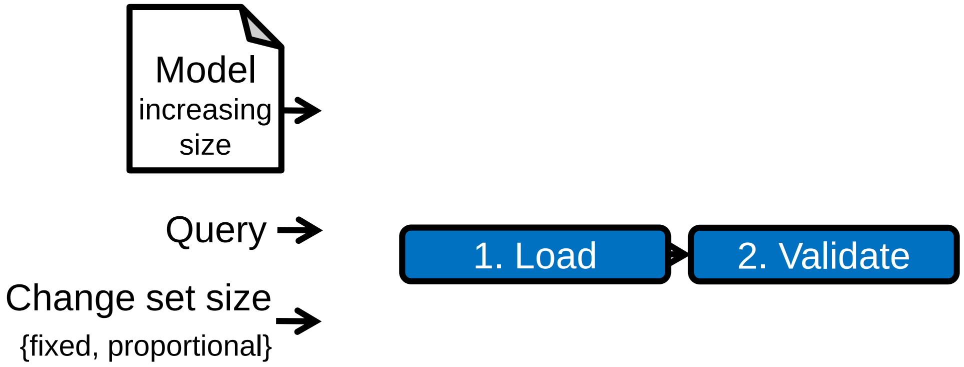
Model
increasing size

Query

Change set size
{fixed, proportional}

1. Load

Benchmark phases



Model

increasing size

Query

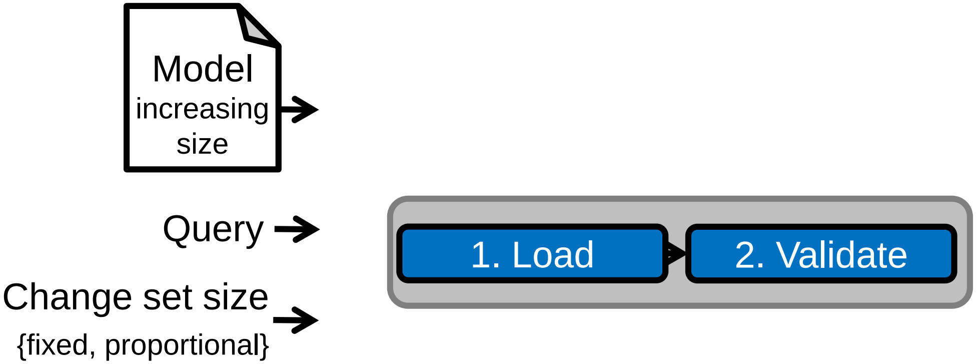
Change set size

{fixed, proportional}

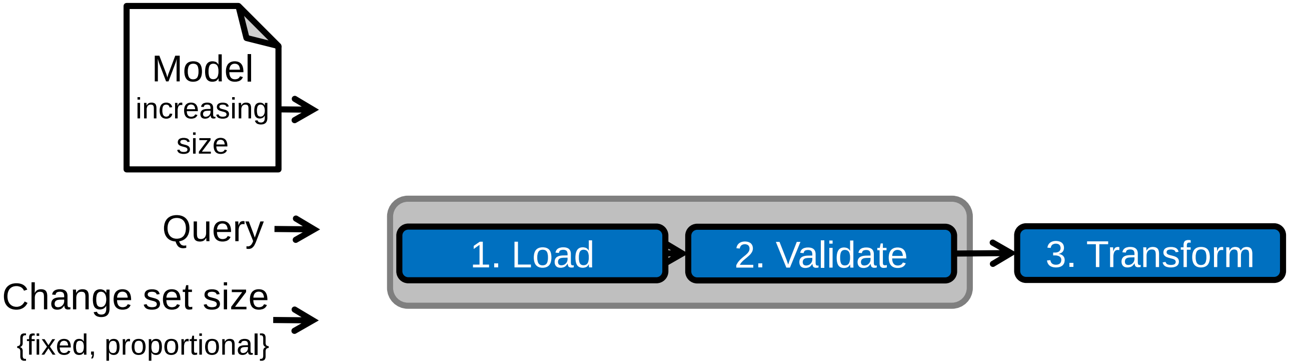
1. Load

2. Validate

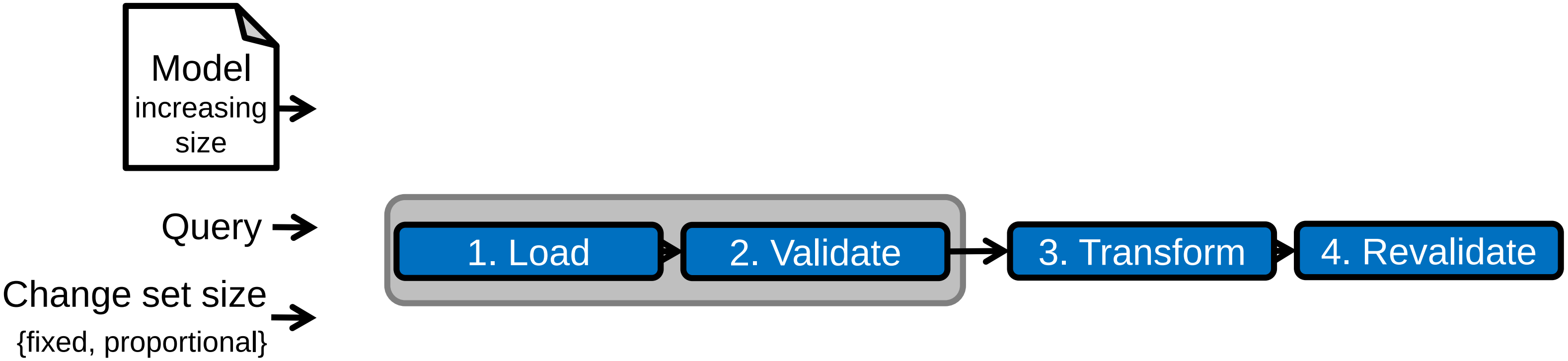
Benchmark phases



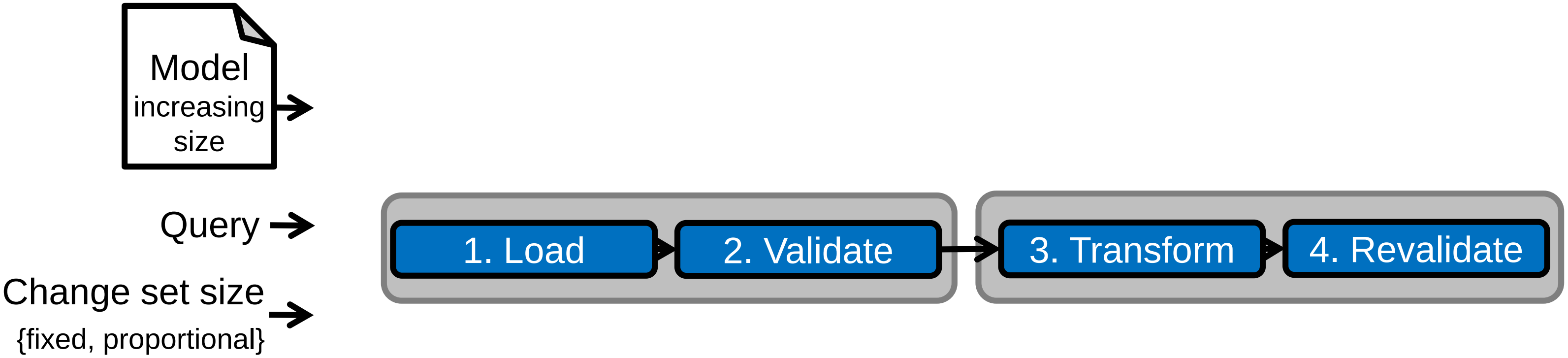
Benchmark phases



Benchmark phases



Benchmark phases



Model increasing size

Query

Change set size {fixed, proportional}

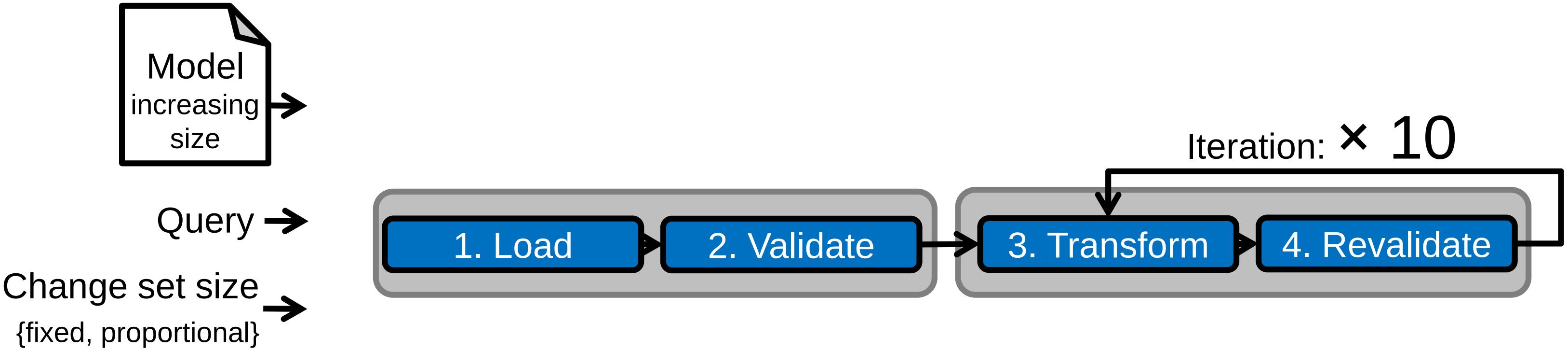
1. Load

2. Validate

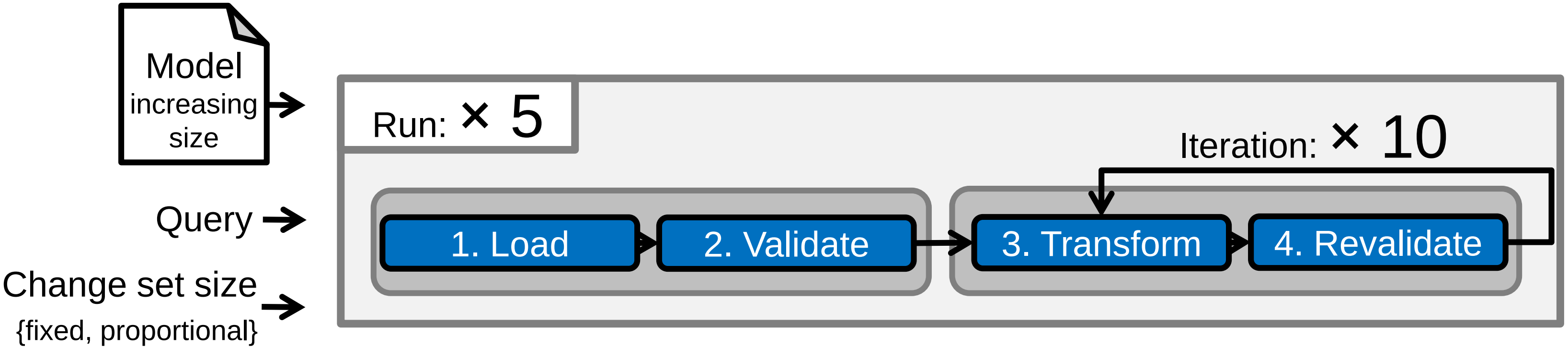
3. Transform

4. Revalidate

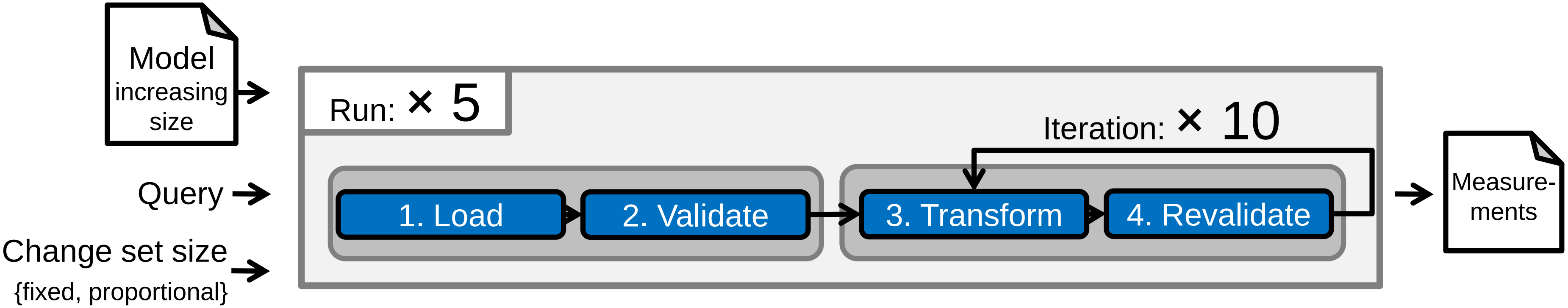
Benchmark phases



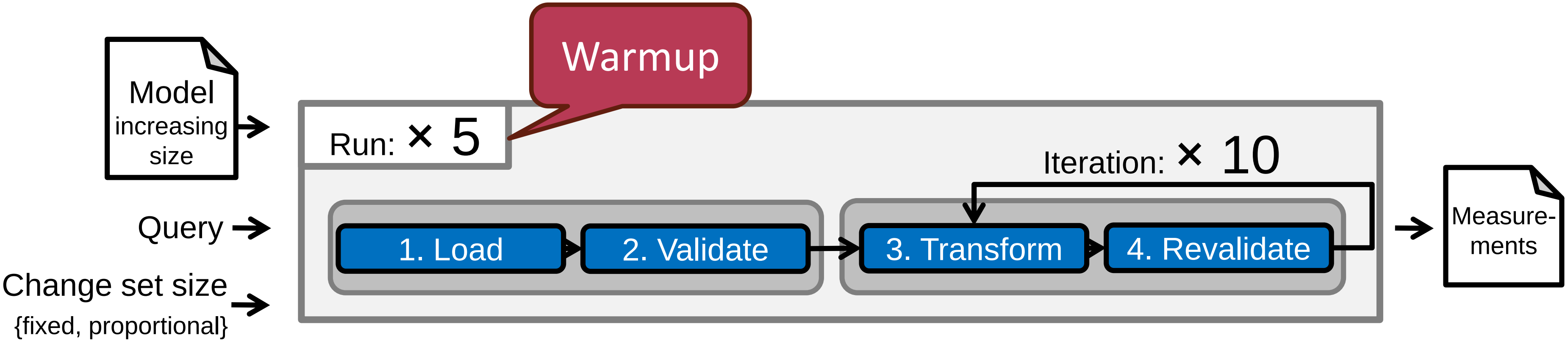
Benchmark phases



Benchmark phases

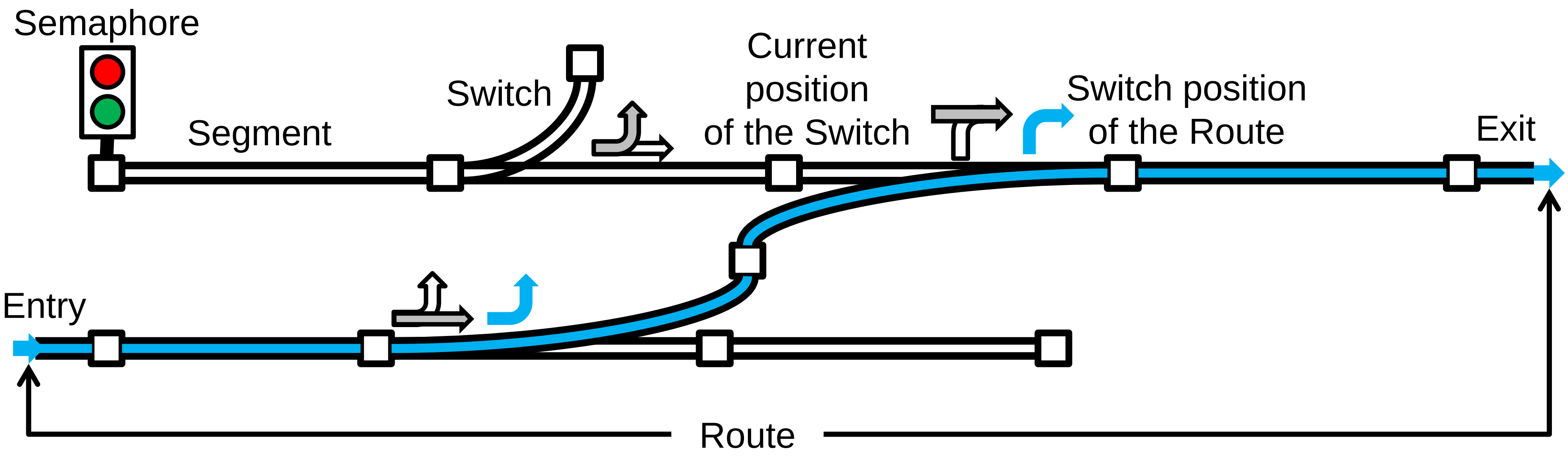


Benchmark phases

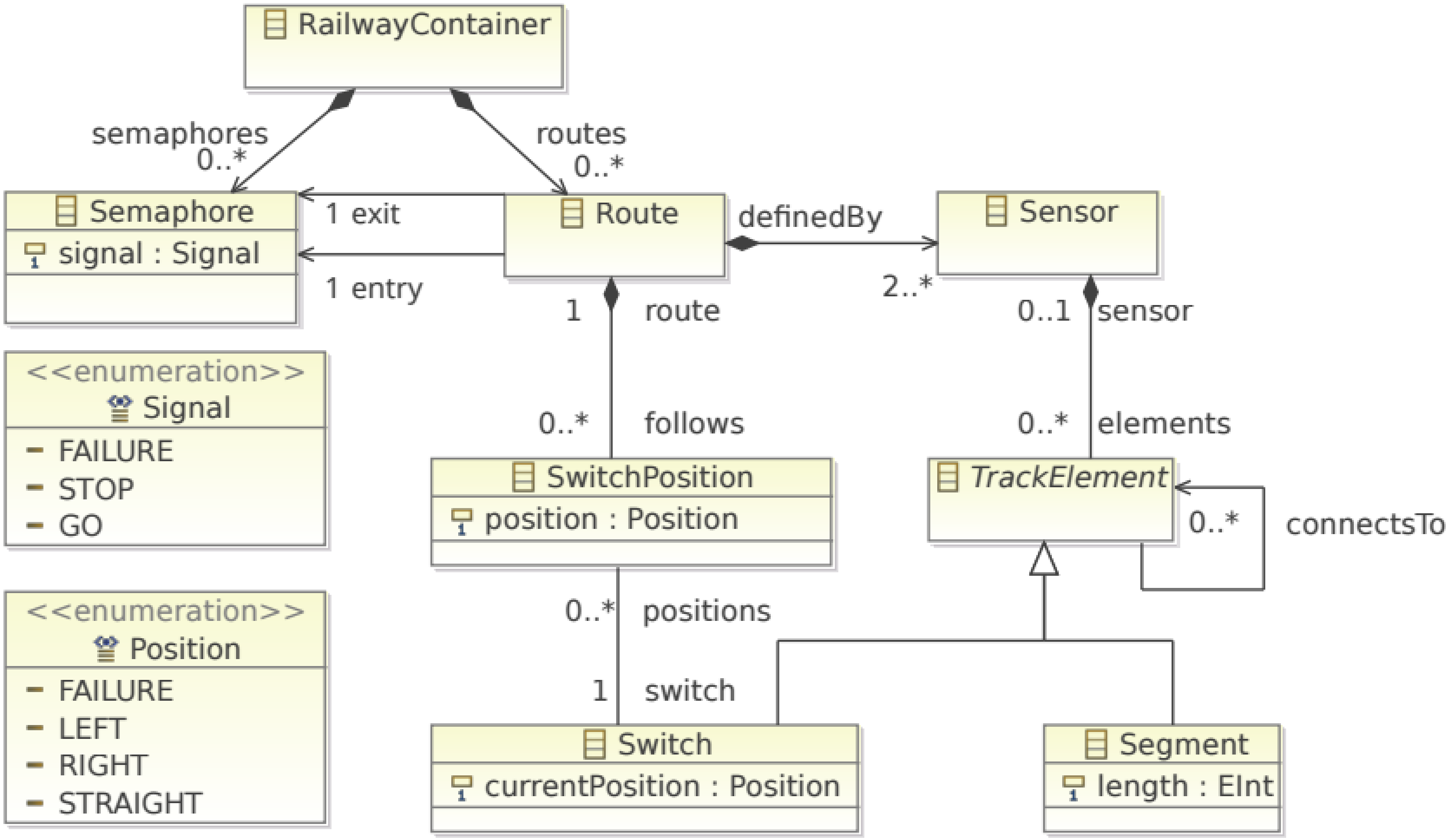


Railway model

- Synthetic model
- Customizable model generator

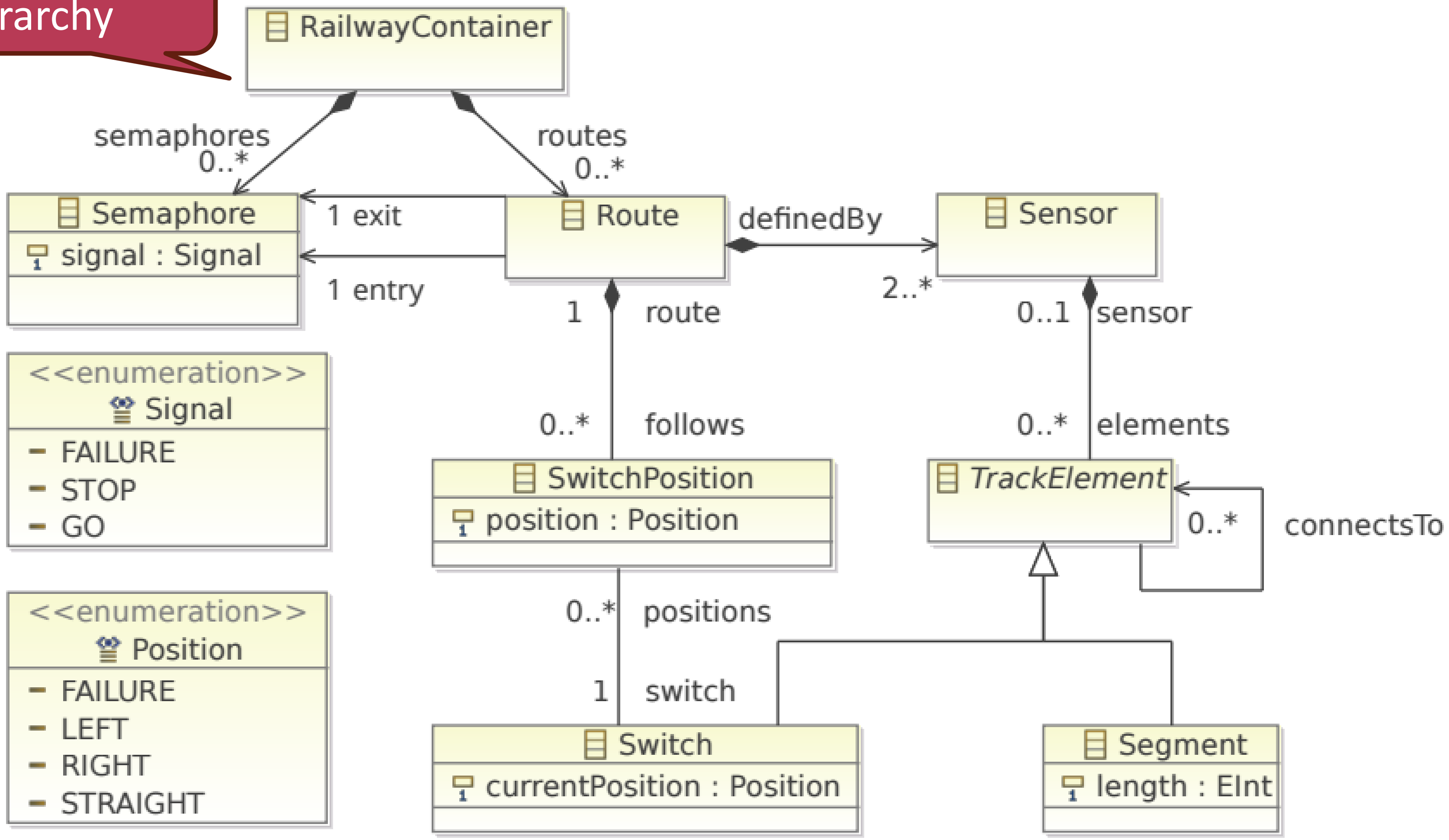


EMF metamodel

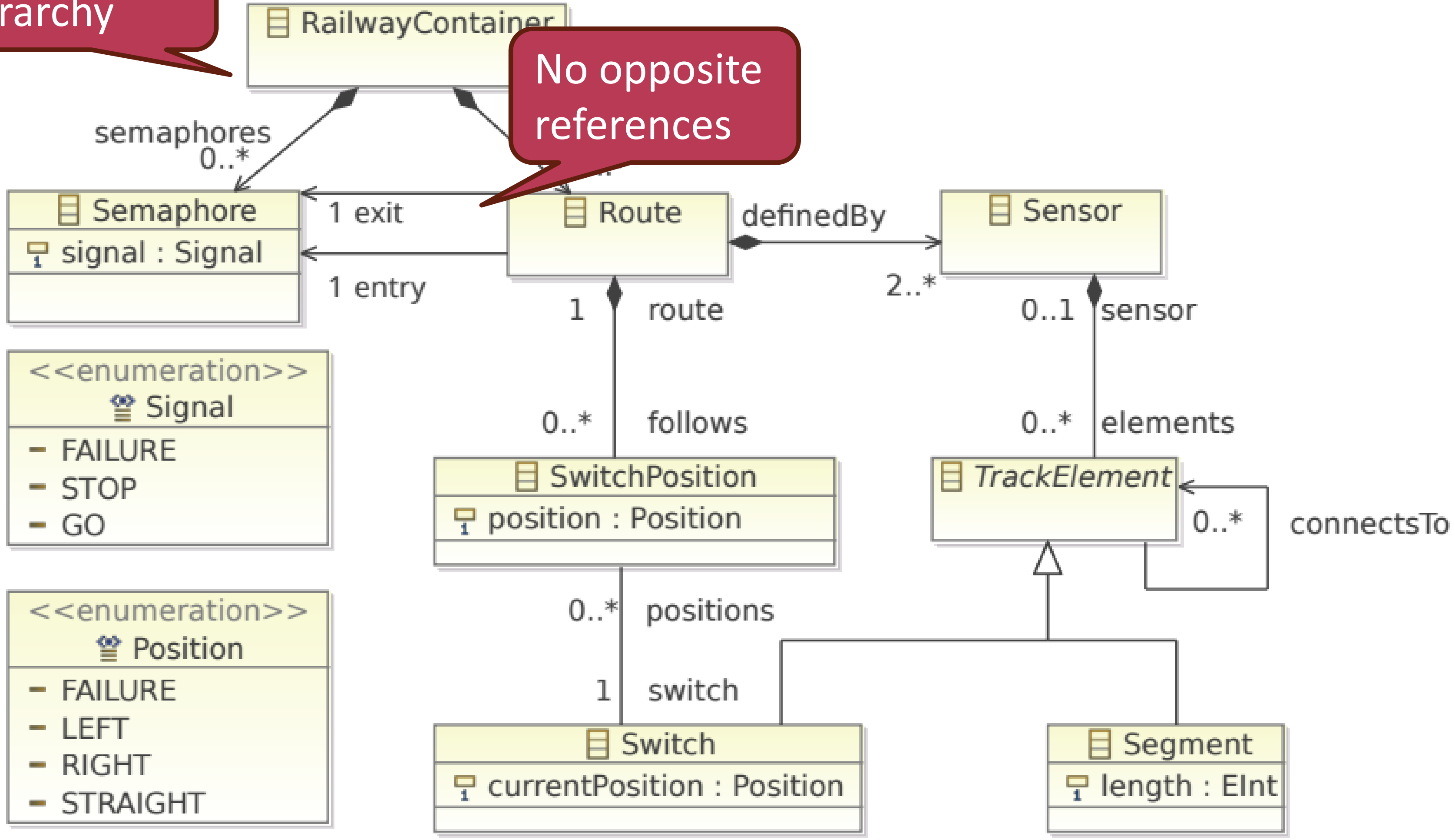


EMF metamodel

Containment hierarchy



EMF metamodel



Containment hierarchy

No opposite references

RailwayContainer

semaphores 0..*

Semaphore
signal : Signal

1 exit

1 entry

Route

definedBy

2..*

Sensor

1 route

0..* follows

0..1 sensor

0..* elements

<<enumeration>>
Signal
- FAILURE
- STOP
- GO

SwitchPosition
position : Position

TrackElement

0..* connectsTo

<<enumeration>>
Position
- FAILURE
- LEFT
- RIGHT
- STRAIGHT

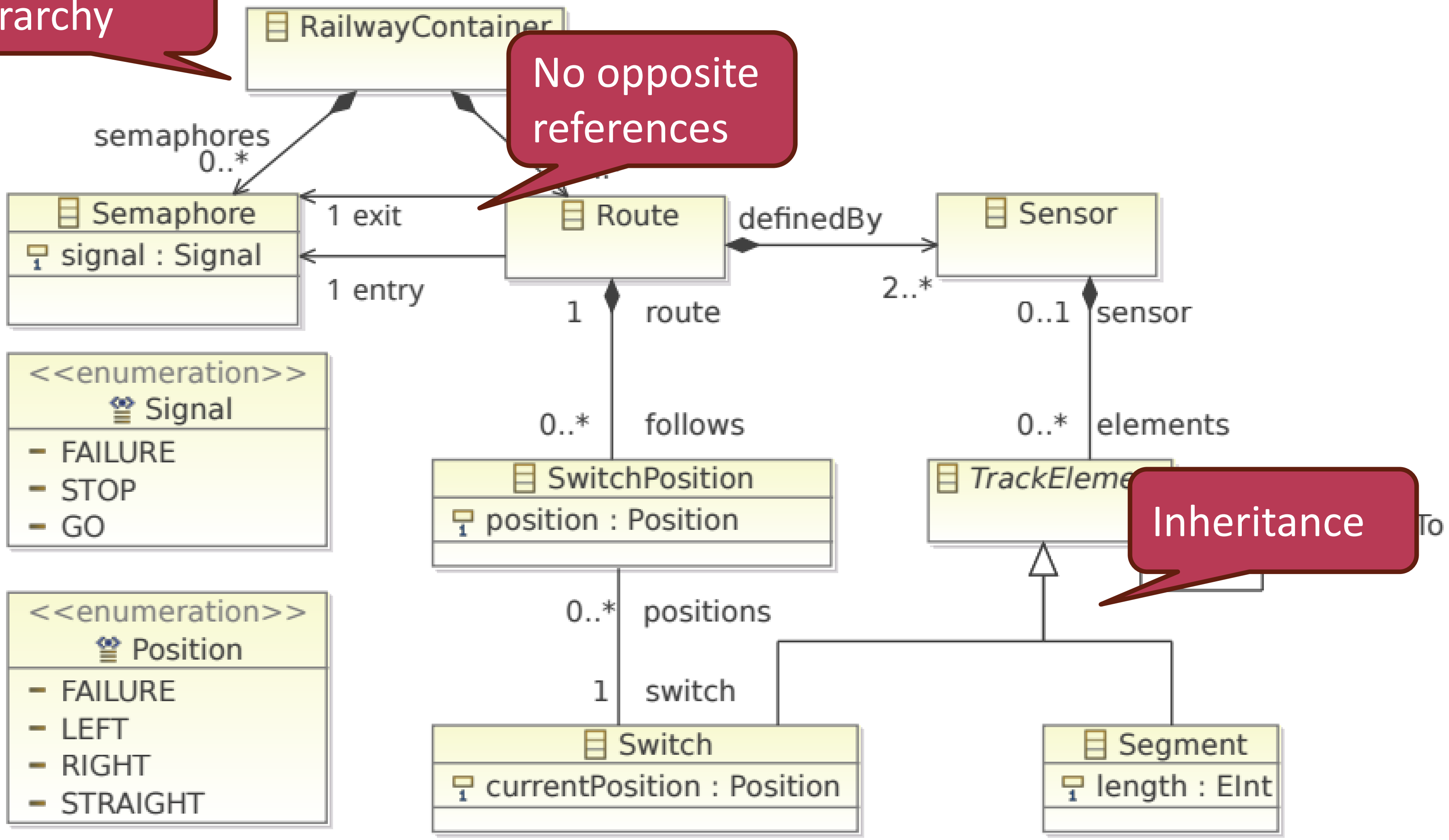
0..* positions

1 switch

Switch
currentPosition : Position

Segment
length : EInt

EMF metamodel

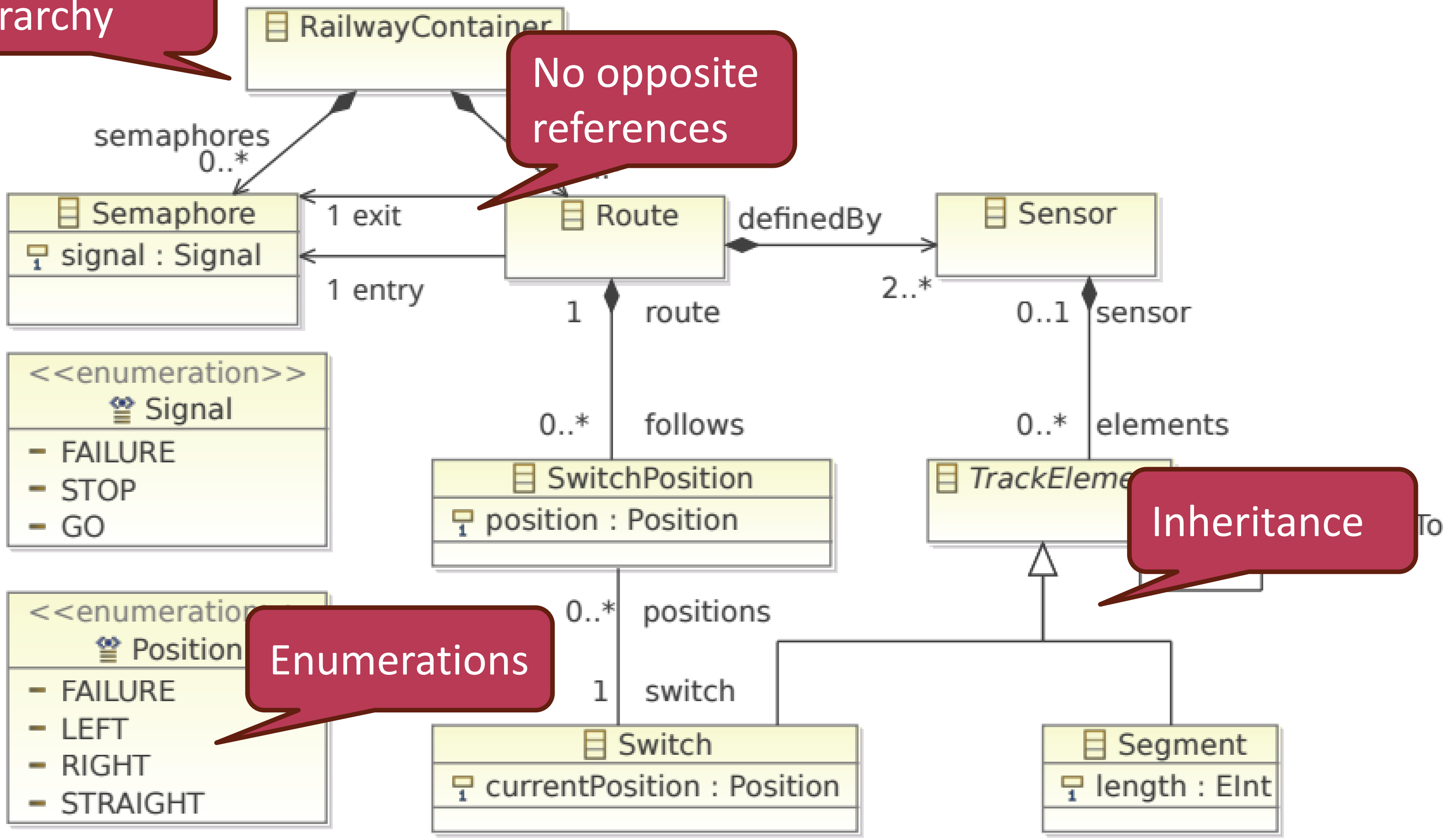


Containment hierarchy

No opposite references

Inheritance

EMF metamodel



Containment hierarchy

No opposite references

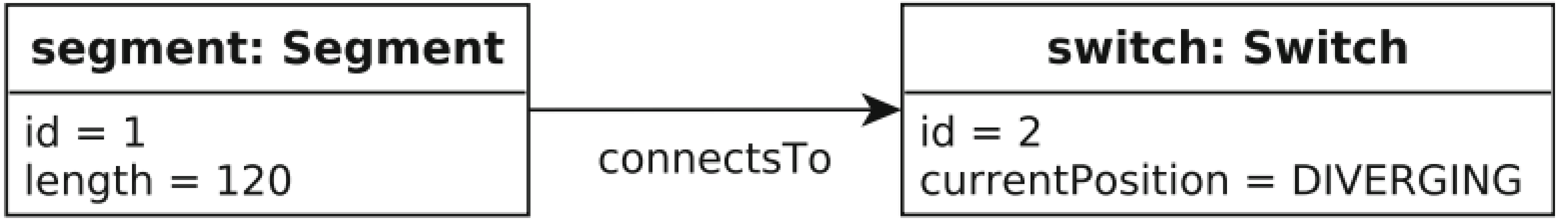
Inheritance

Enumerations

Instance model generation

- Randomly generated with increasing sizes
  - 1, 2, 4, …
- Violations are inserted during generation
- Multiple formats:
  - EMF
  - property graph
  - RDF
  - SQL

EMF model



segment: Segment

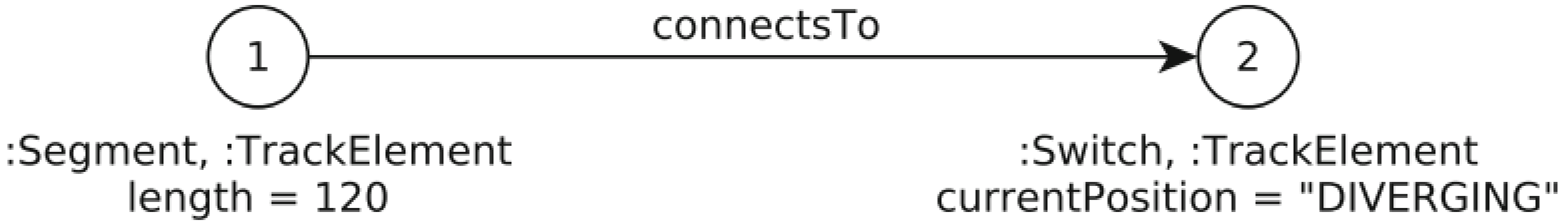
id = 1
length = 120

connectsTo

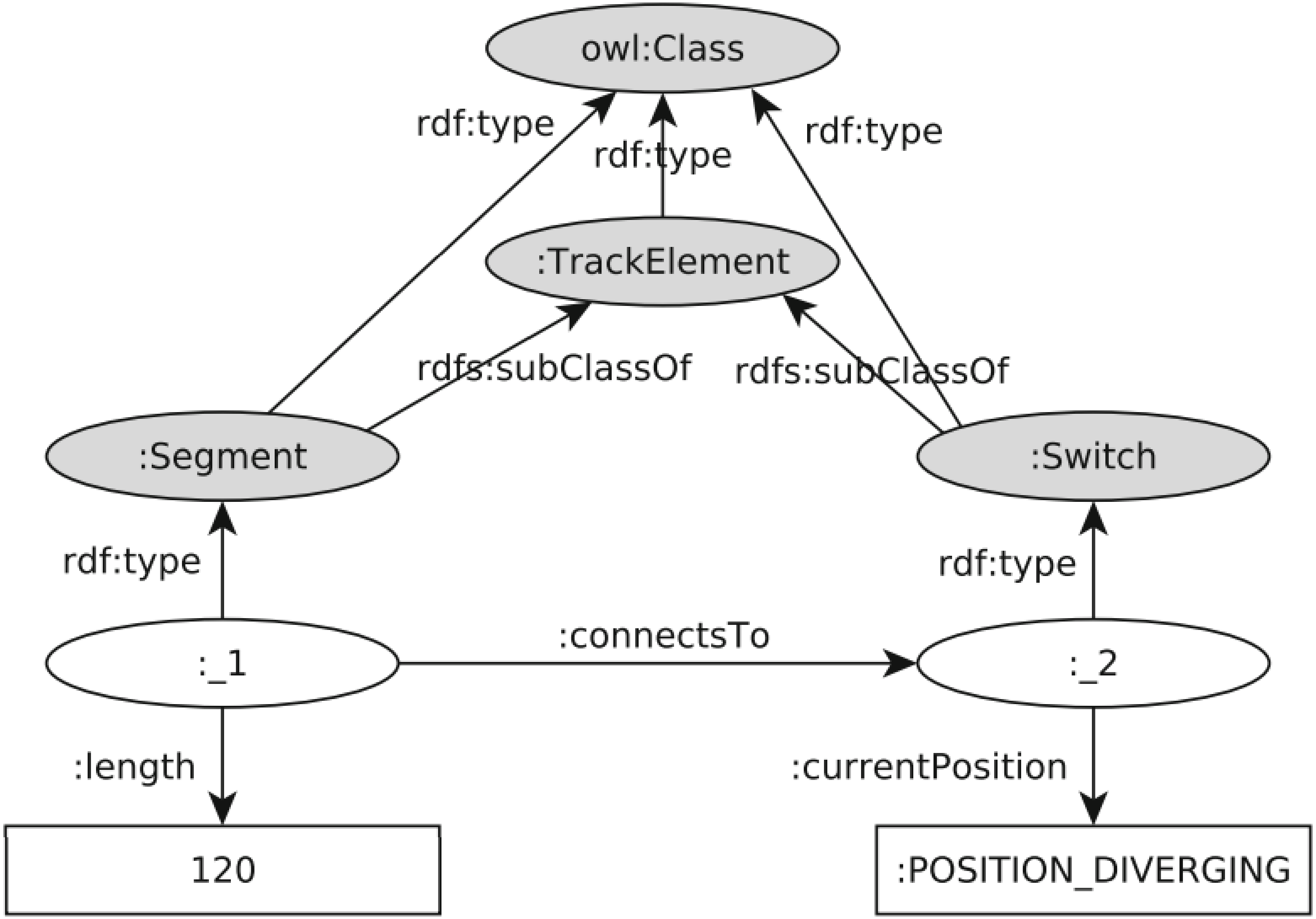
switch: Switch

id = 2
currentPosition = DIVERGING

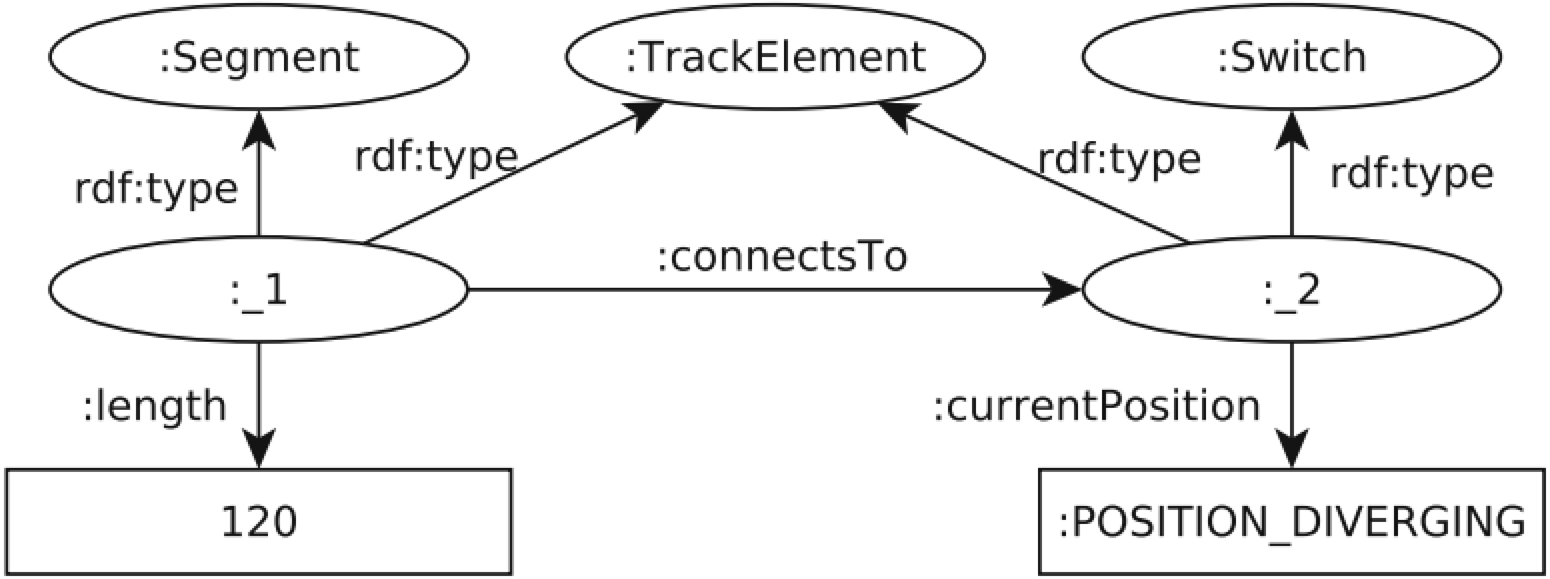
Property graph



RDF with metamodel



RDF with inferred tuples



Validation and transformation

- Well-formedness constraints

Validation and transformation

- Well-formedness constraints
- Queries are looking for **error patterns**

Validation and transformation

- Well-formedness constraints
- Queries are looking for **error patterns**
  - Lots of filtering, joins, etc.

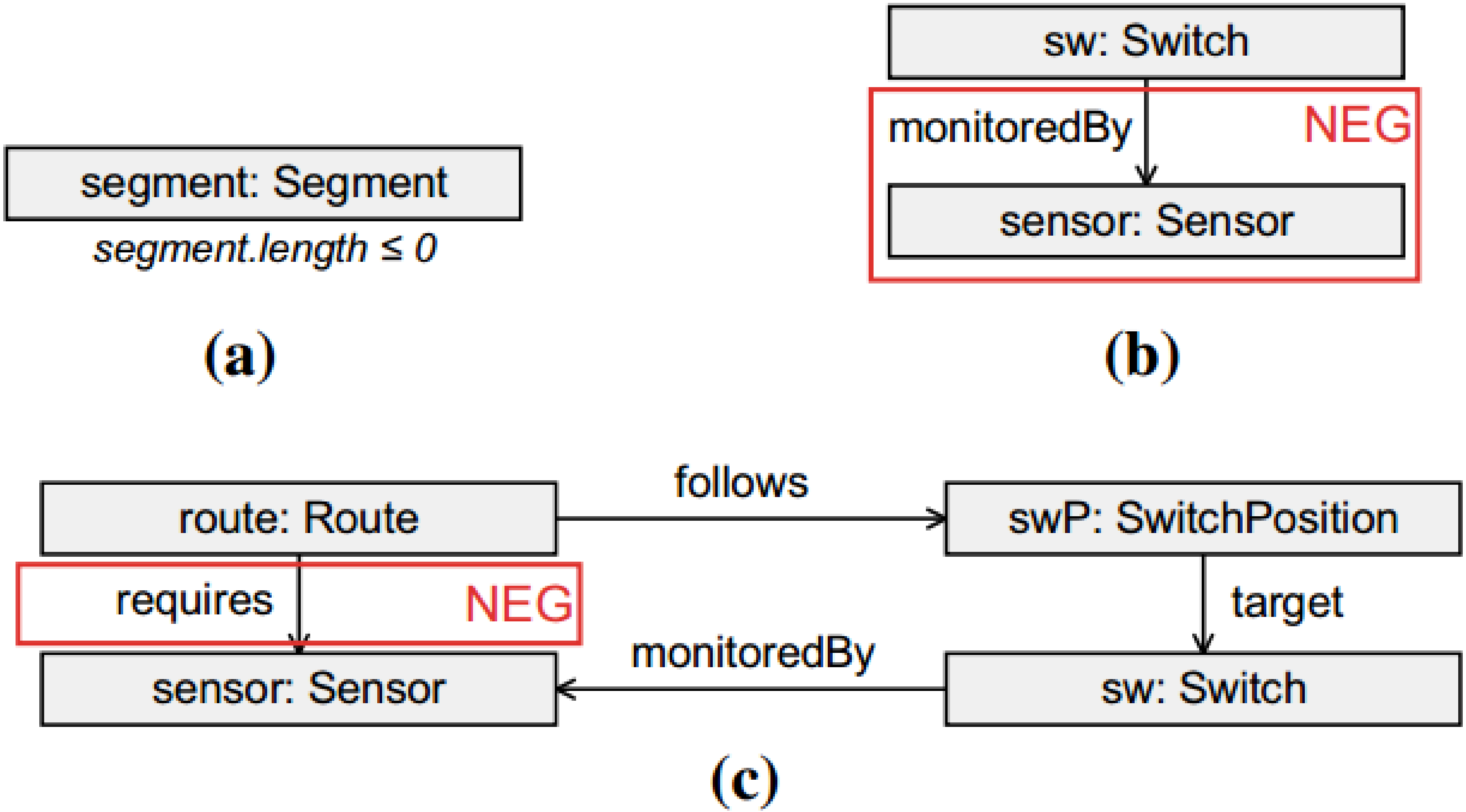
Validation and transformation

- Well-formedness constraints
- Queries are looking for **error patterns**
  - Lots of filtering, joins, etc.
- Transformations

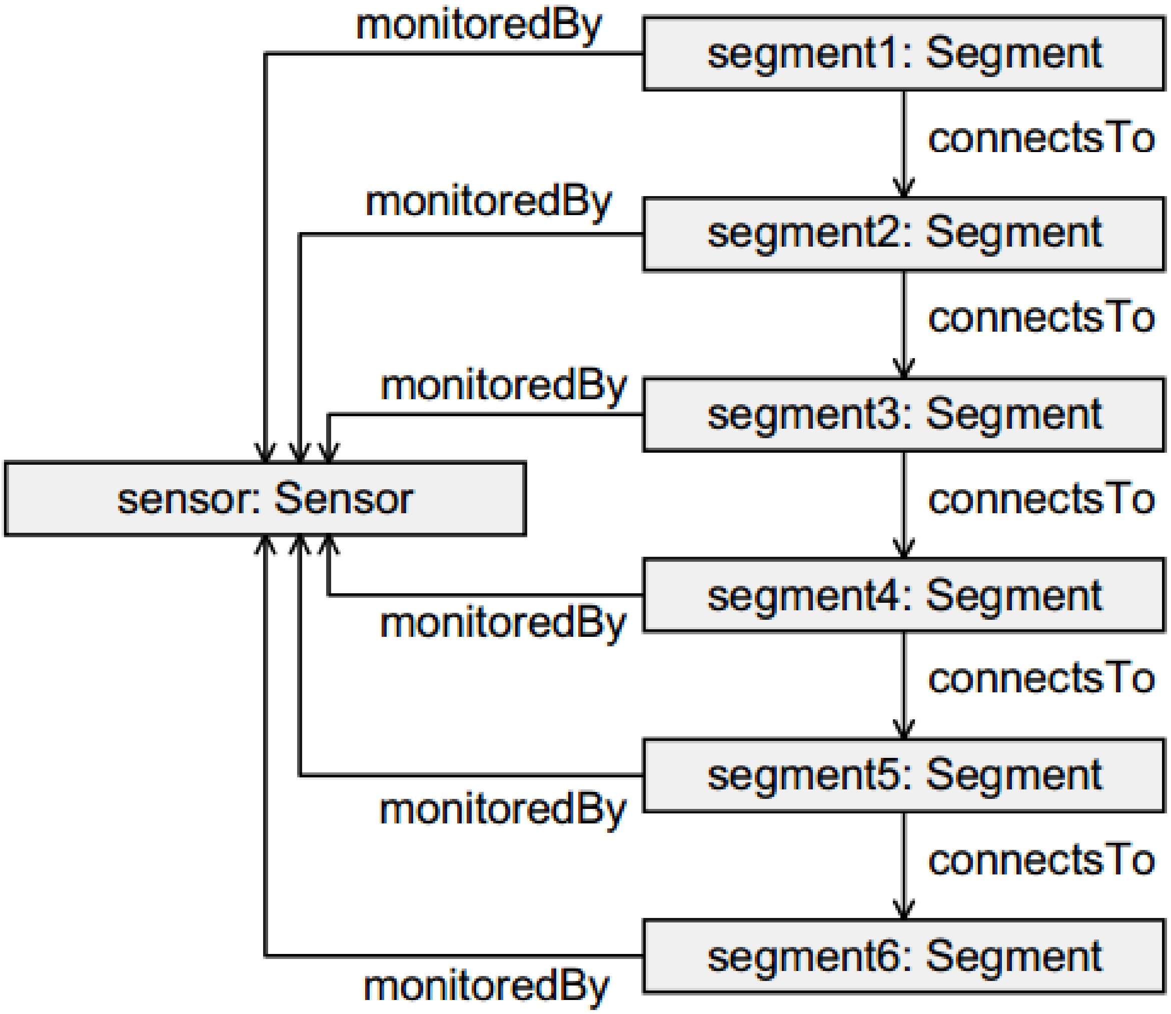
Validation and transformation

- Well-formedness constraints
- Queries are looking for **error patterns**
  - Lots of filtering, joins, etc.
- Transformations
  - Fault injections
  - Quick fix-like repair operations

Queries

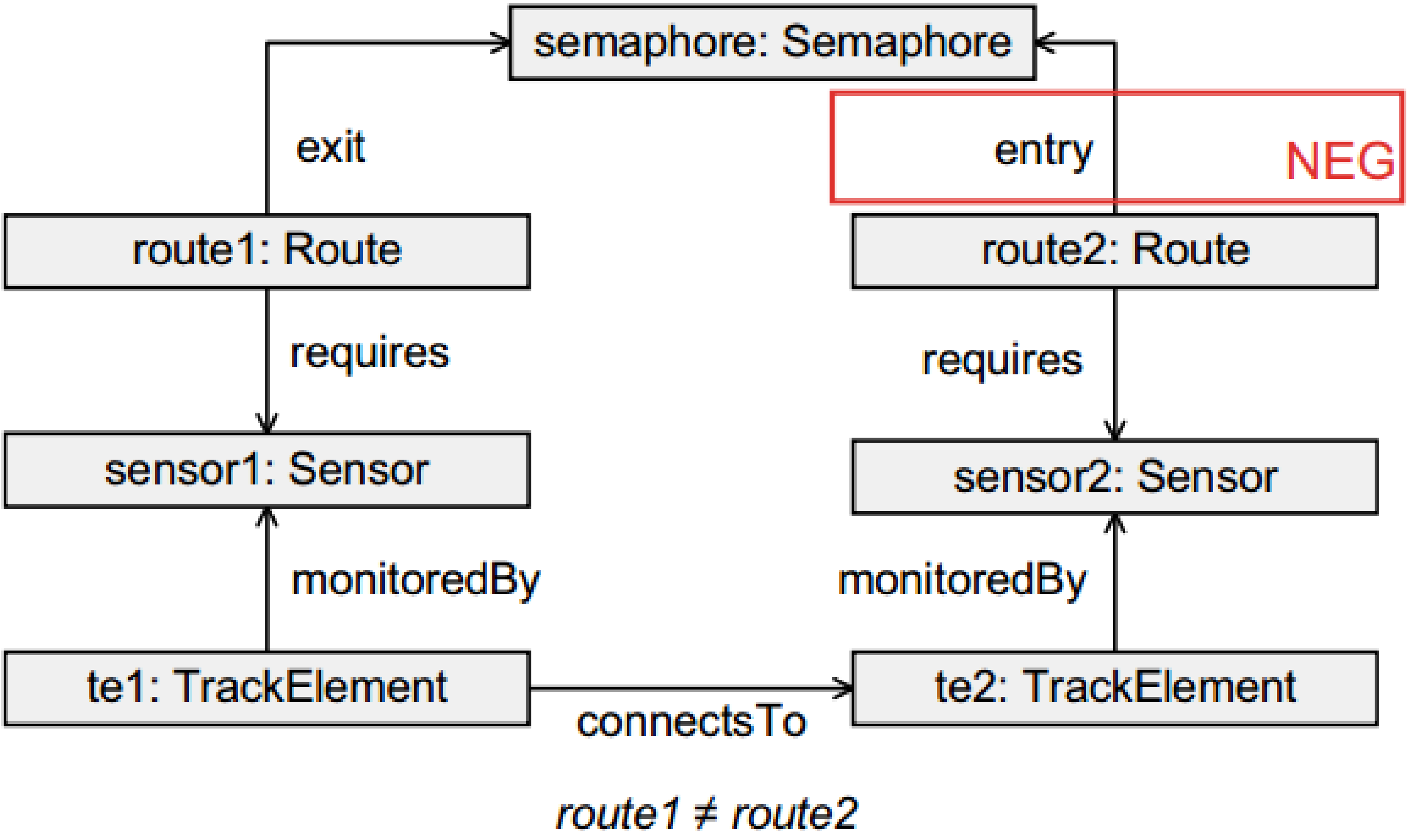

segment: Segment
segment.length ≤ 0
(a)
sw: Switch
monitoredBy
NEG
sensor: Sensor
(b)
route: Route
follows
swP: SwitchPosition
requires
NEG
target
monitoredBy
sensor: Sensor
sw: Switch
(c)

Queries


monitoredBy
segment1: Segment
connectsTo
monitoredBy
segment2: Segment
connectsTo
monitoredBy
segment3: Segment
sensor: Sensor
connectsTo
segment4: Segment
monitoredBy
connectsTo
segment5: Segment
monitoredBy
connectsTo
segment6: Segment
monitoredBy

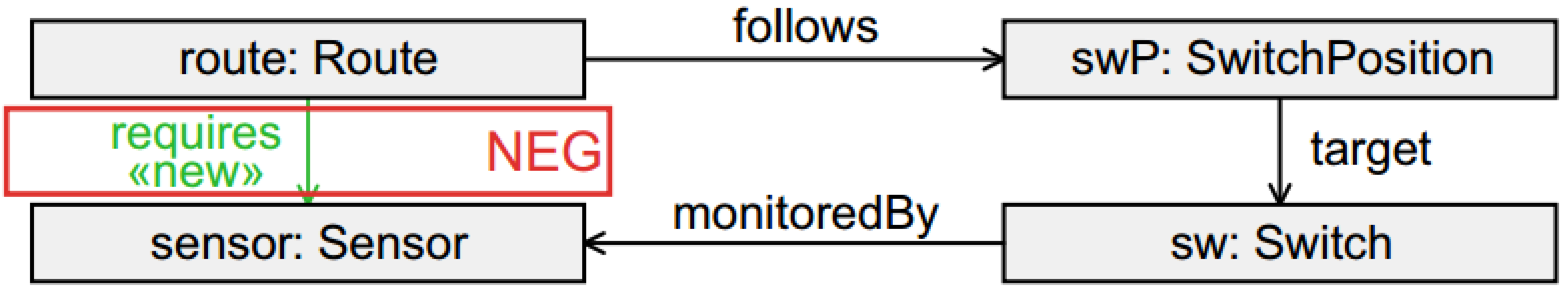
(e)

Queries

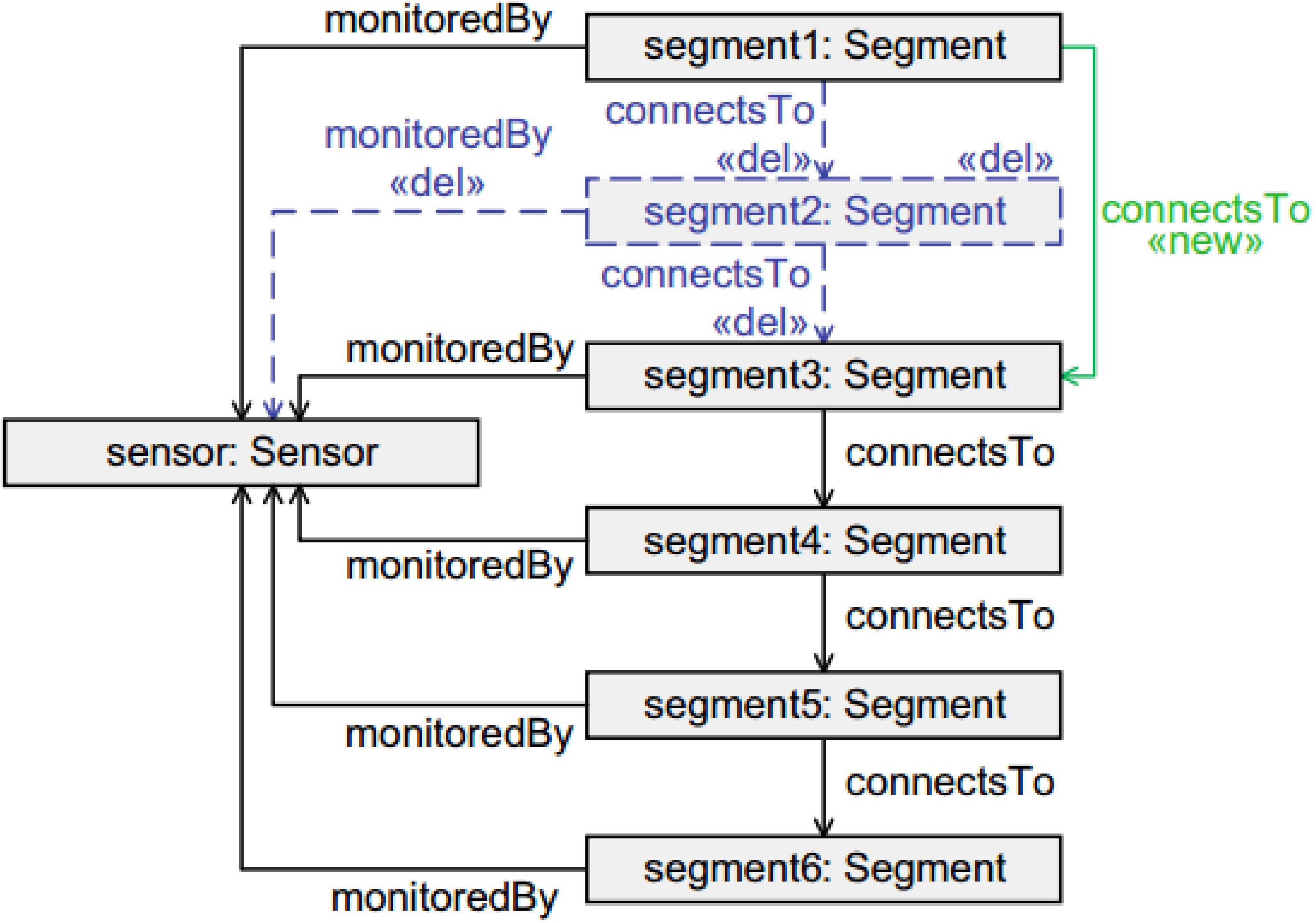

semaphore: Semaphore
exit
entry
NEG
route1: Route
route2: Route
requires
requires
sensor1: Sensor
sensor2: Sensor
monitoredBy
monitoredBy
te1: TrackElement
te2: TrackElement
connectsTo
route1 ≠ route2

(f)

Transformation



Transformation



Threats to validity

Threats to validity

- **Operating system:** caching, scheduled jobs

Threats to validity

- **Operating system:** caching, scheduled jobs
- **Cloud noise** from the environment

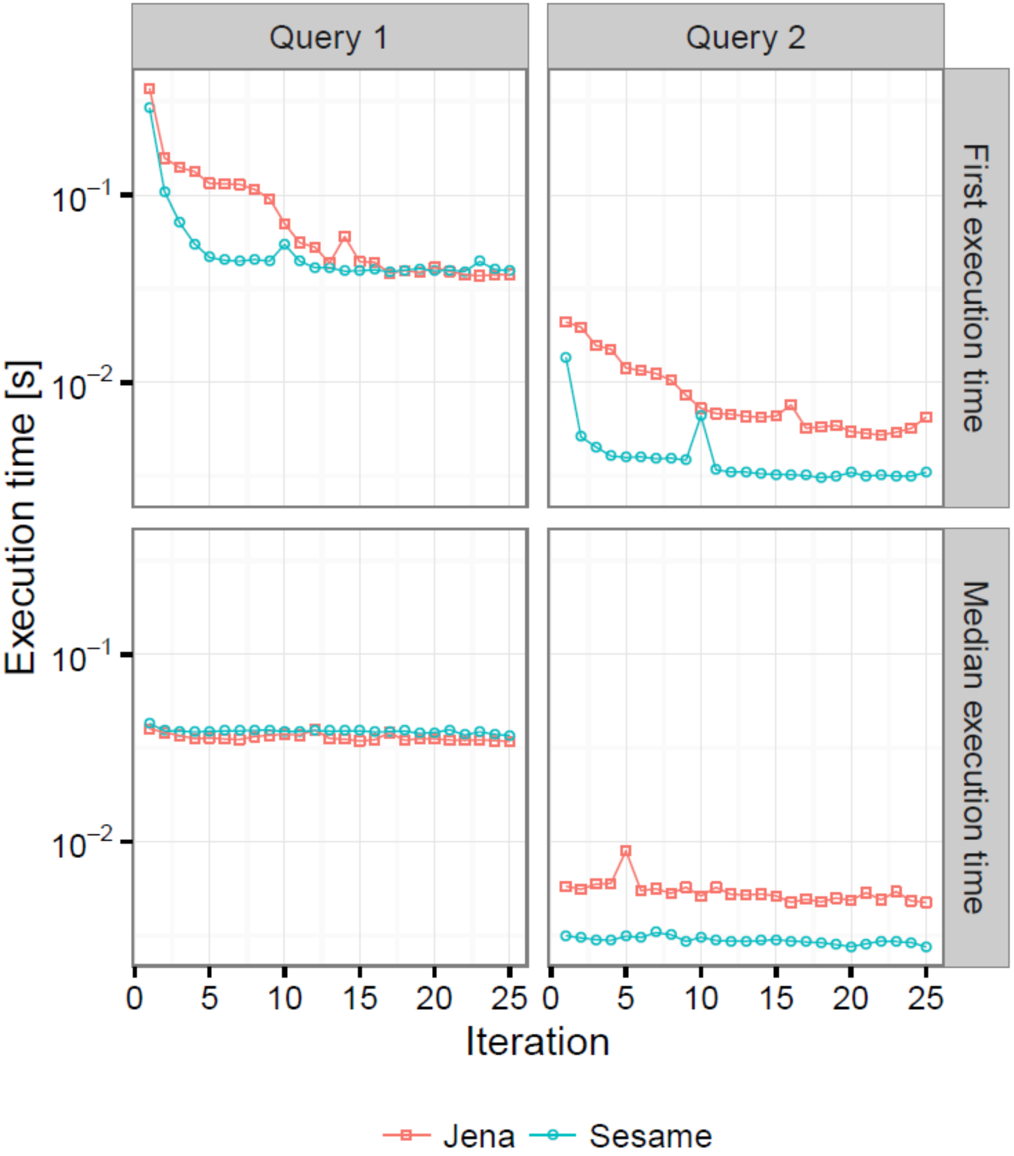
Threats to validity

- **Operating system:** caching, scheduled jobs
- **Cloud noise** from the environment
- **Parallel processes & multithreaded execution**

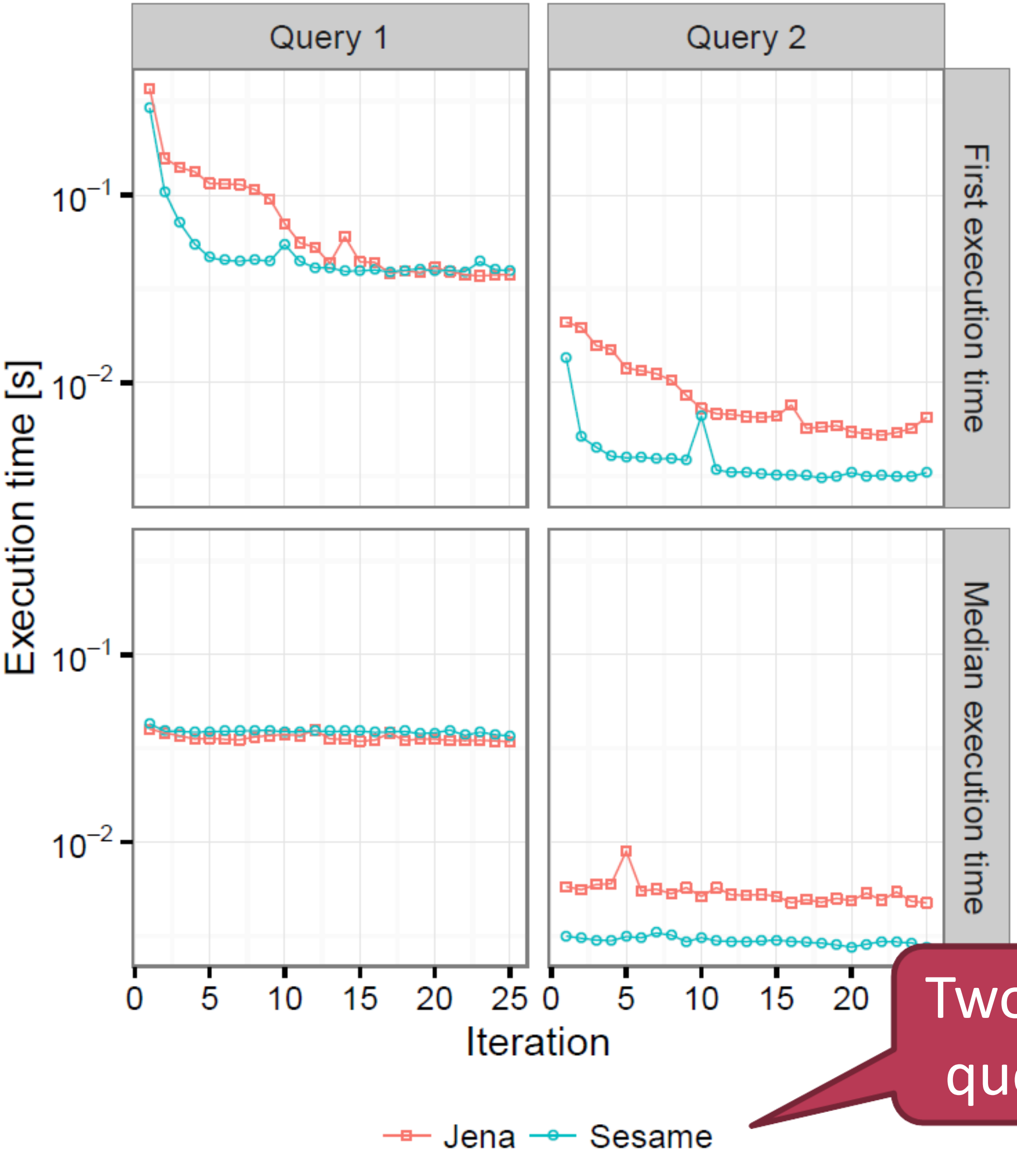
Threats to validity

- **Operating system:** caching, scheduled jobs
- **Cloud noise** from the environment
- **Parallel processes & multithreaded execution**
- **Managed runtime environments**
  - Java Virtual Machine
  - .NET CLR

The effect of warmup

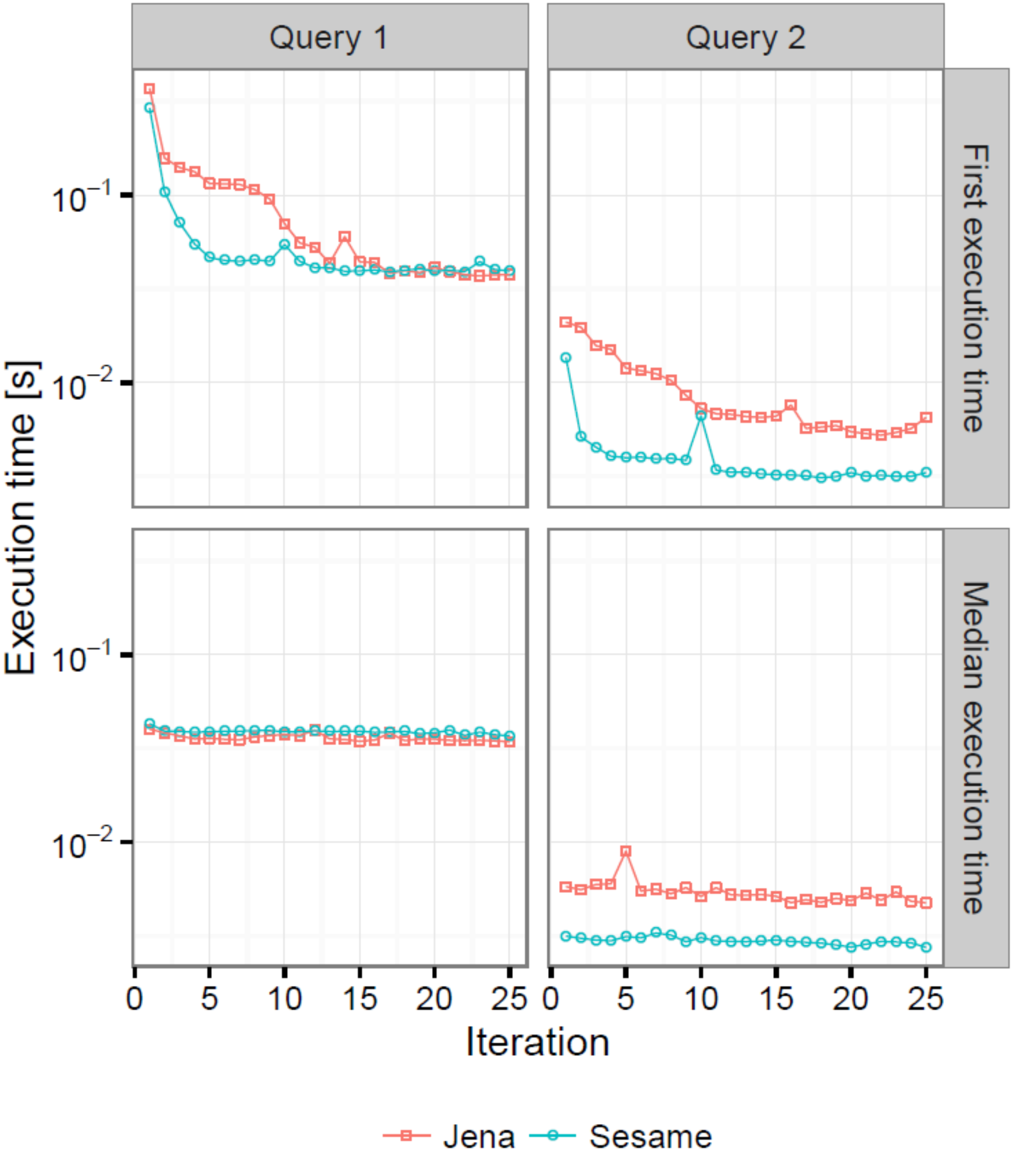


The effect of warmup

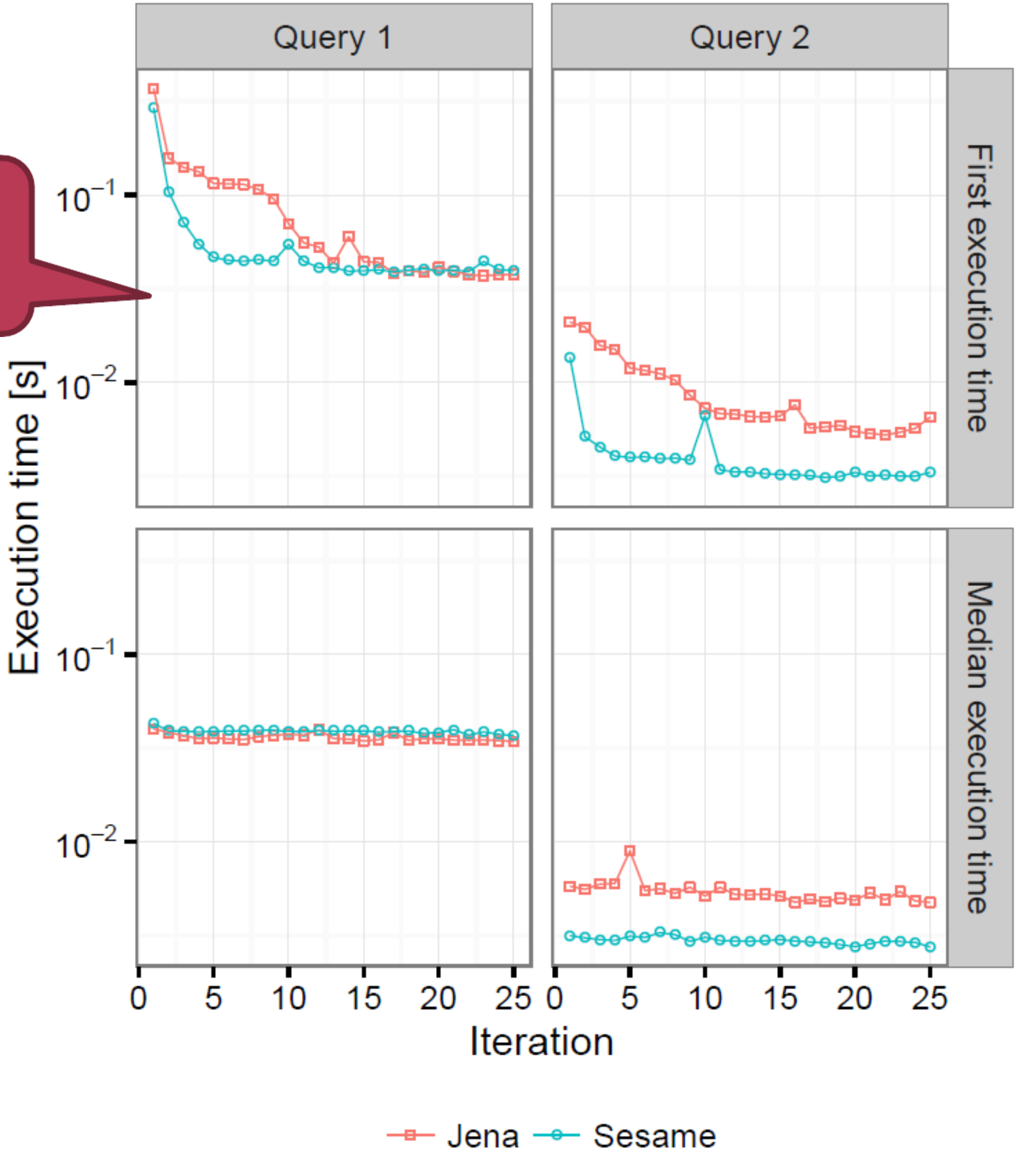


Two Java-based query engines

The effect of warmup



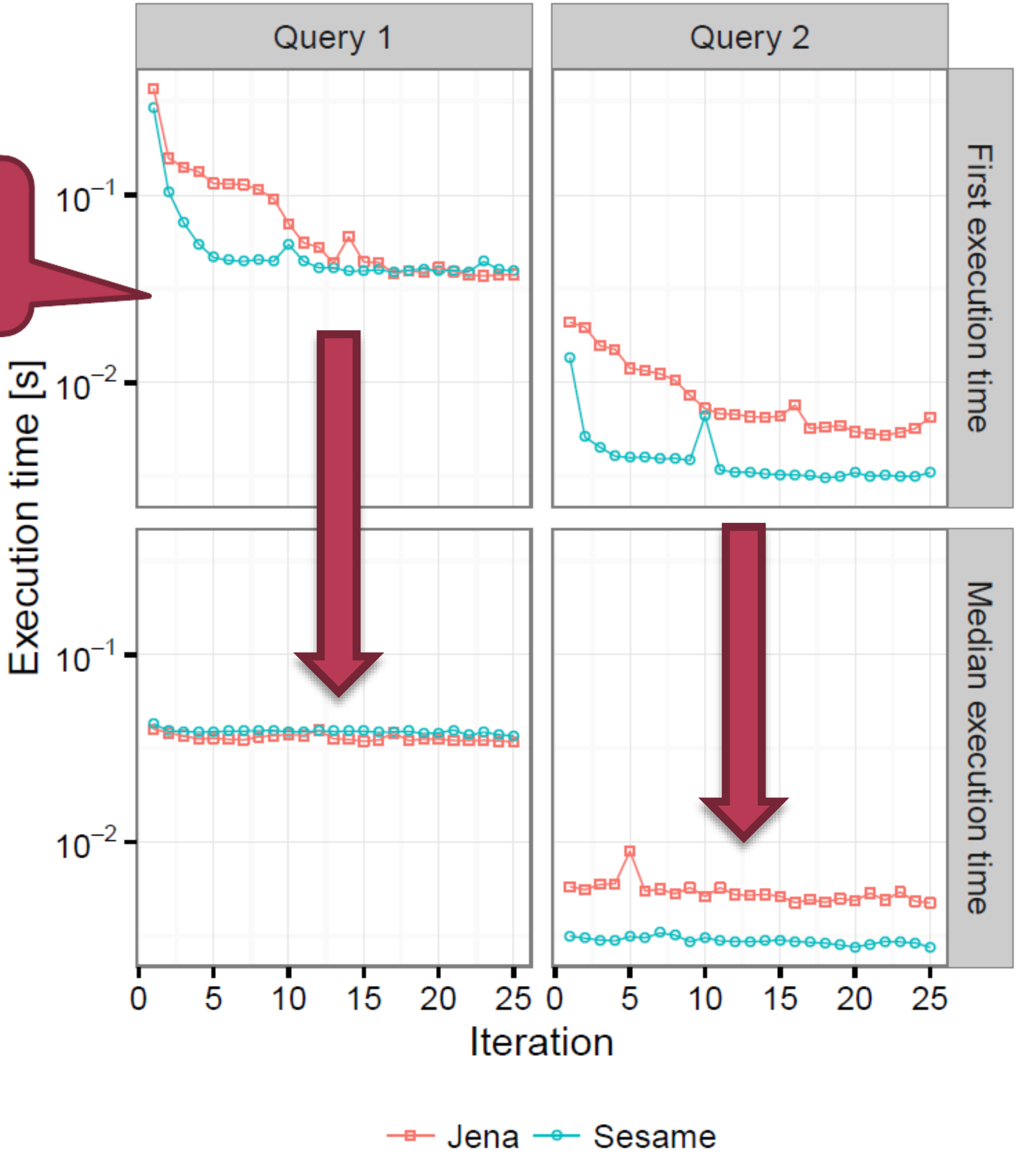
The effect of warmup



First execution time

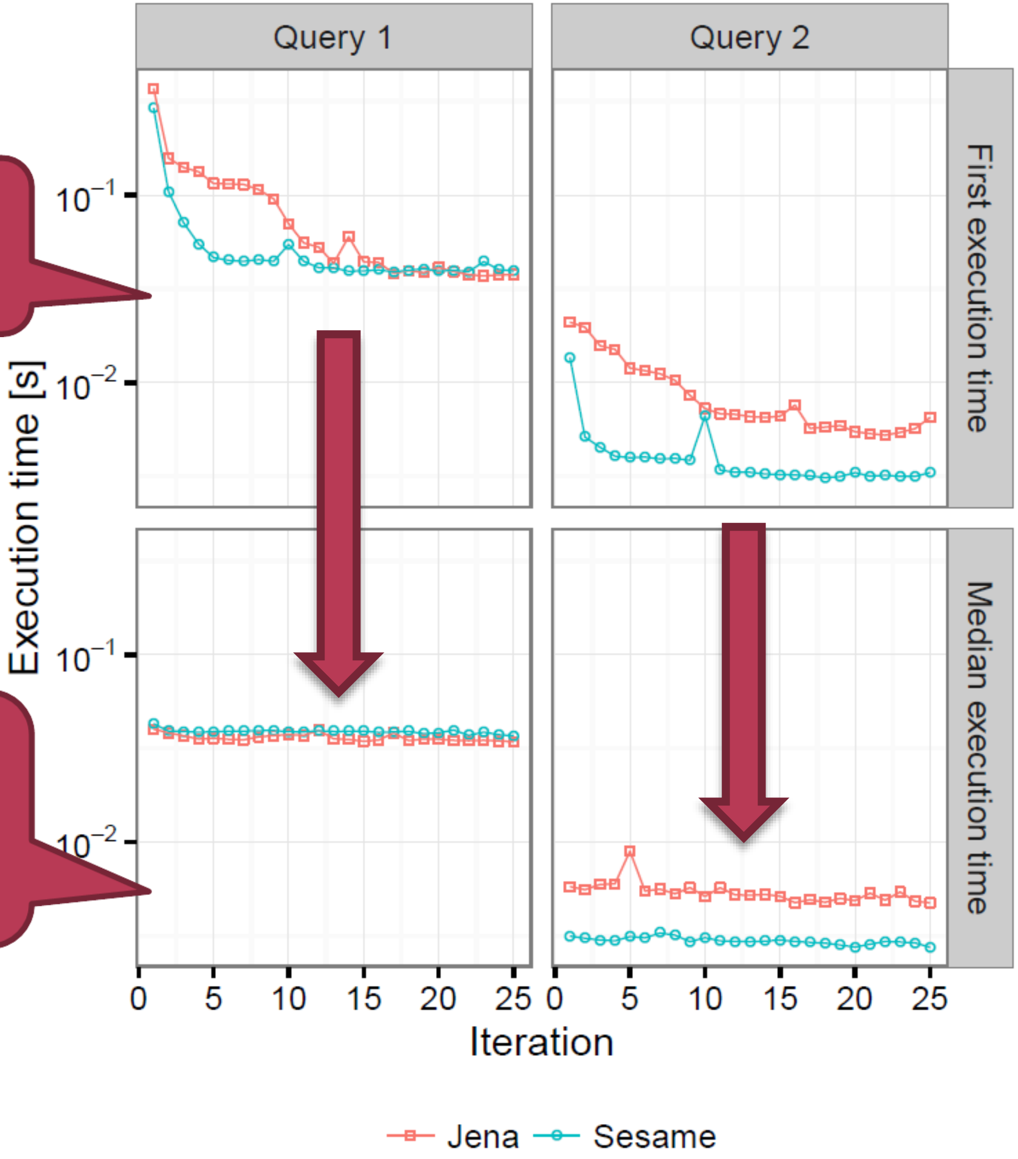
The effect of warmup

First execution time



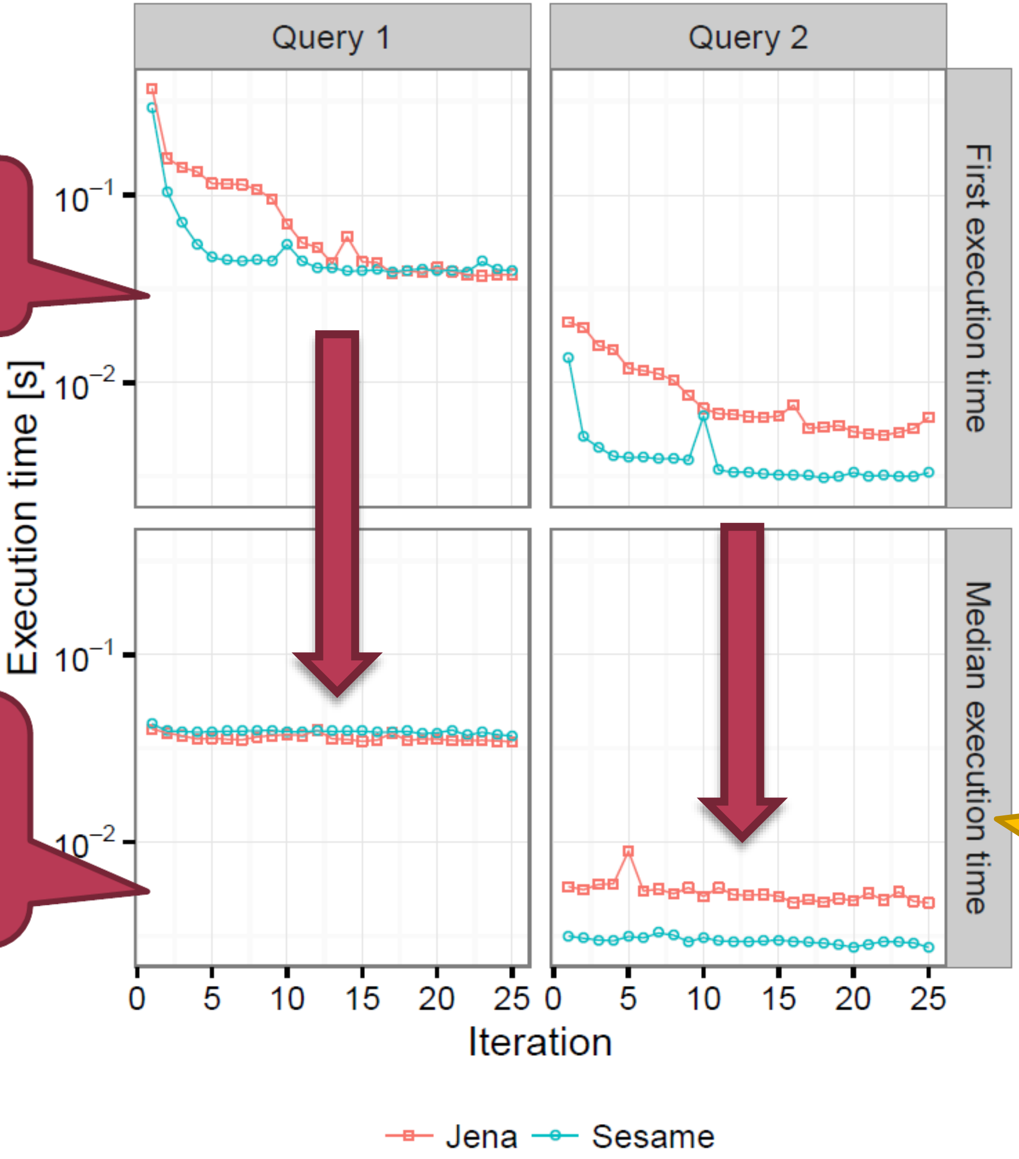
The effect of warmup

First execution time



Median execution time of 5 executions

The effect of warmup



First execution time

Median execution time of 5 executions

Perform multiple executions

Benchmark environment

- Quad-core Xeon CPU
- SSD
- 16 GB RAM
- Ubuntu 14.04 with Oracle Java 8

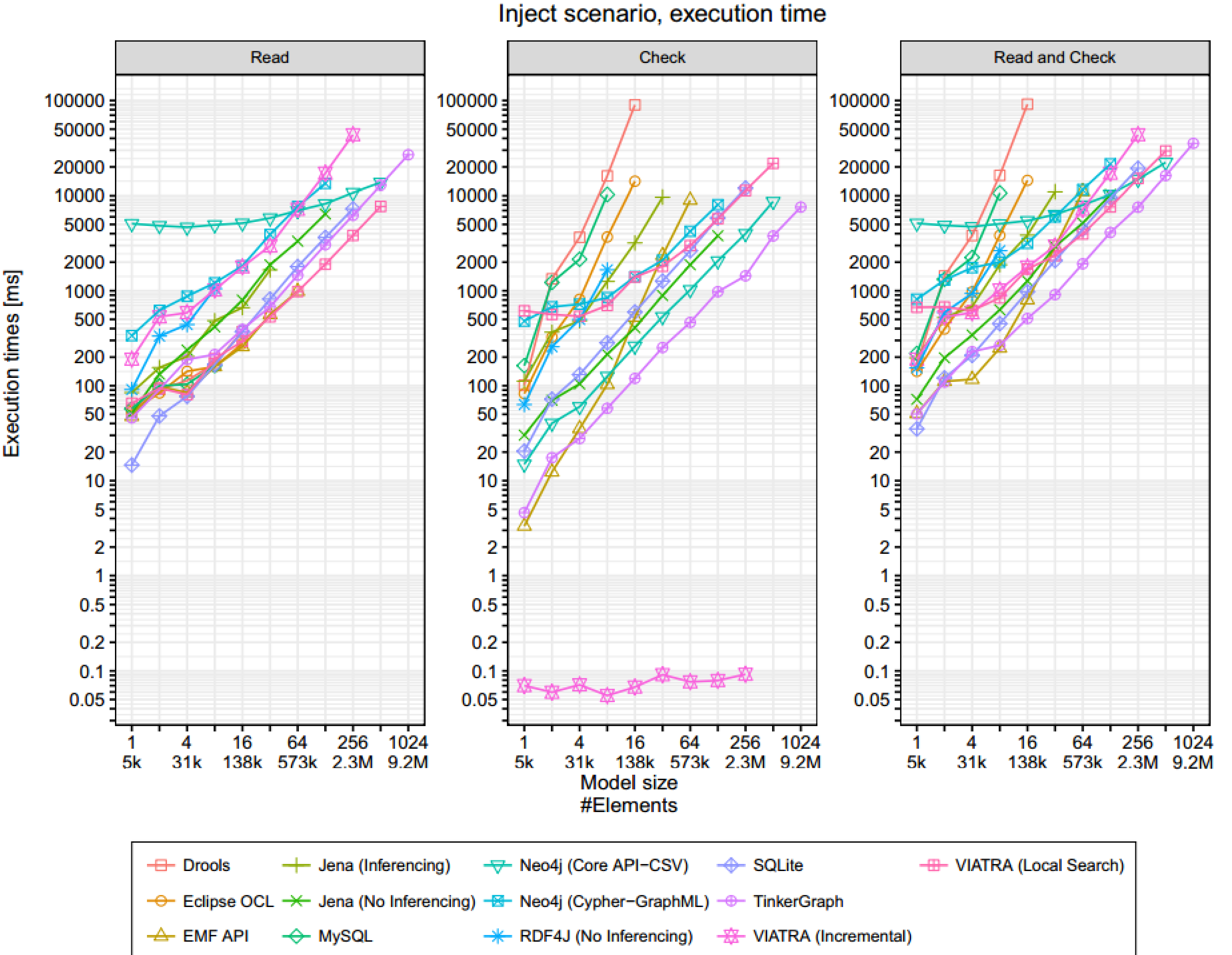
DigitalOcean

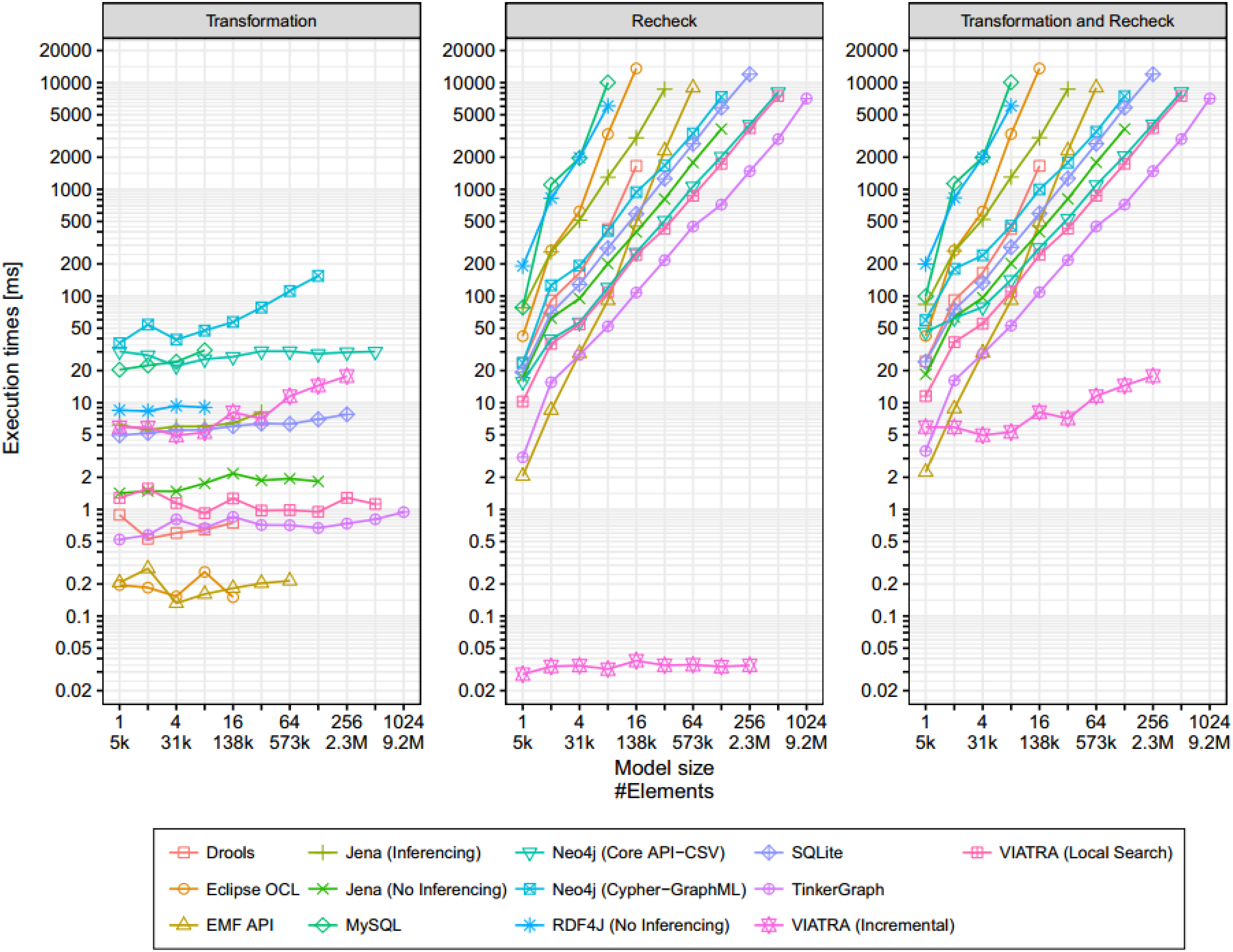
Benchmark environment

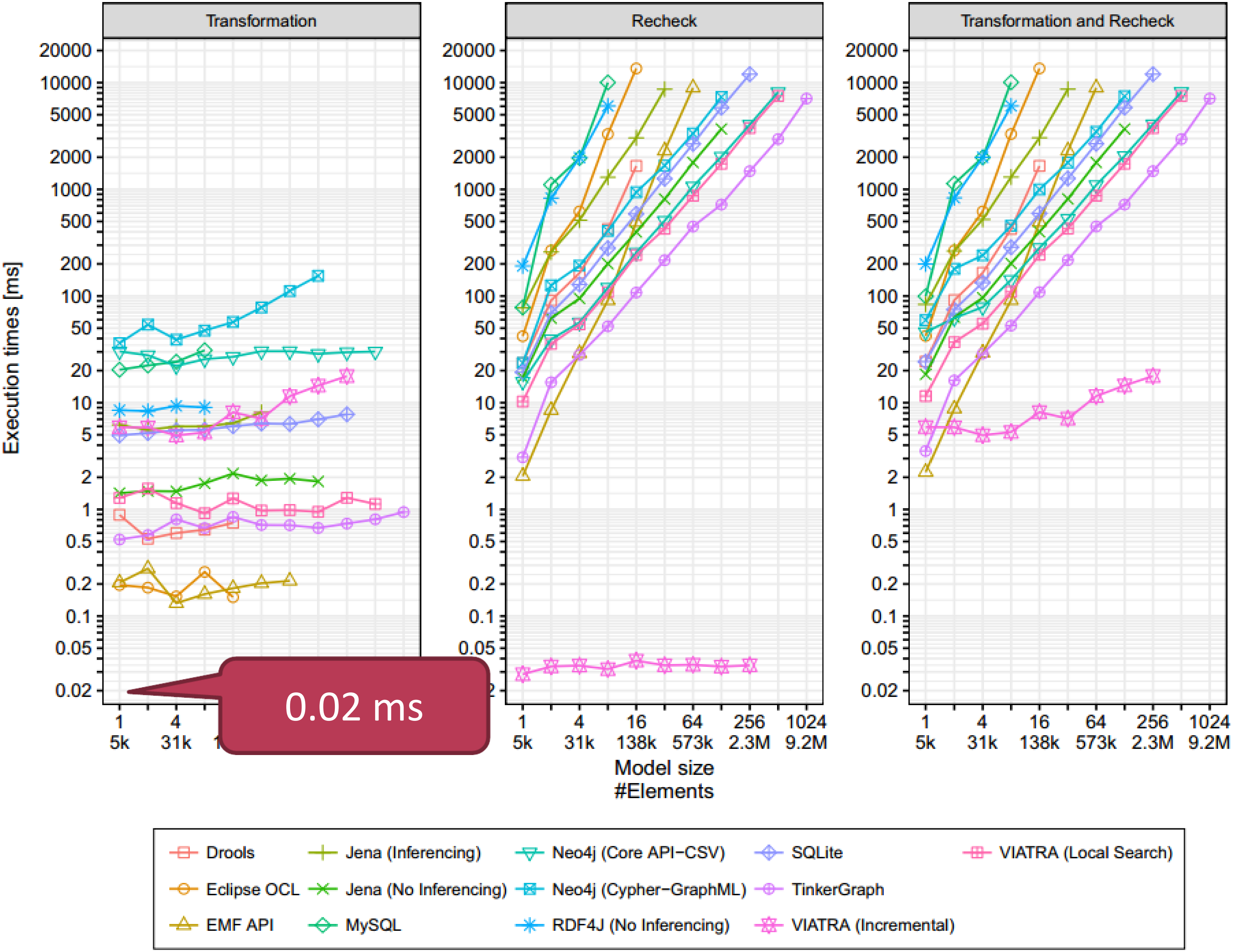
- Quad-core Xeon CPU
- SSD
- 16 GB RAM
- Ubuntu 14.04 with Oracle Java 8

Support for educational users

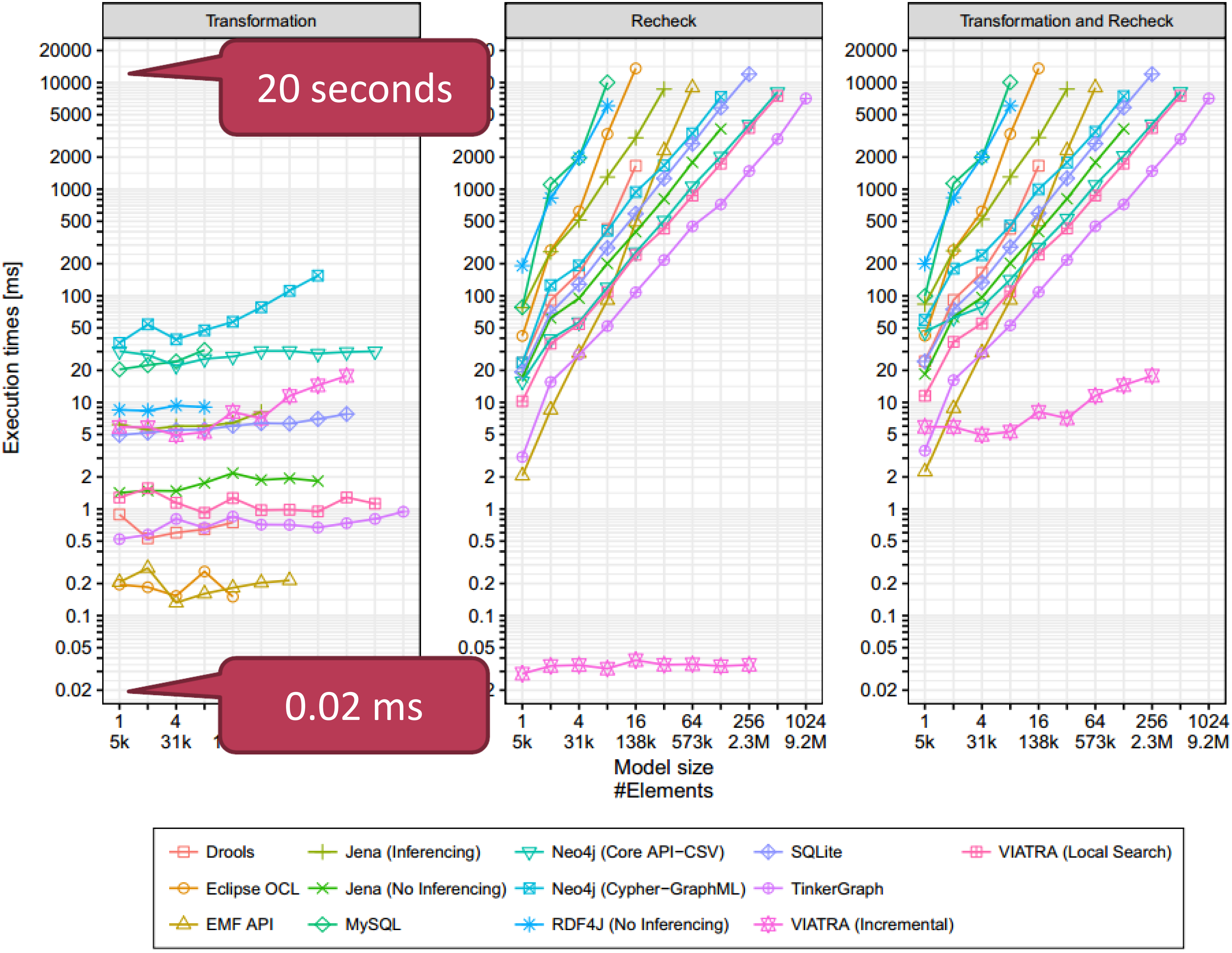
DigitalOcean


Inject scenario, execution time
Read
Check
Read and Check
Execution times [ms]
100000
50000
20000
10000
5000
2000
1000
500
200
100
50
20
10
5
2
1
0.5
0.2
0.1
0.05
1
4
16
64
256
1024
5k
31k
138k
573k
2.3M
9.2M
Model size
#Elements
Drools
Eclipse OCL
EMF API
Jena (Inferencing)
Jena (No Inferencing)
MySQL
Neo4j (Core API−CSV)
Neo4j (Cypher−GraphML)
RDF4J (No Inferencing)
SQLite
TinkerGraph
VIATRA (Incremental)
VIATRA (Local Search)


Transformation
Recheck
Transformation and Recheck
Execution times [ms]
20000
10000
5000
2000
1000
500
200
100
50
20
10
5
2
1
0.5
0.2
0.1
0.05
0.02
1
4
16
64
256
1024
5k
31k
138k
573k
2.3M
9.2M
Model size
#Elements
Drools
Eclipse OCL
EMF API
Jena (Inferencing)
Jena (No Inferencing)
MySQL
Neo4j (Core API−CSV)
Neo4j (Cypher−GraphML)
RDF4J (No Inferencing)
SQLite
TinkerGraph
VIATRA (Incremental)
VIATRA (Local Search)

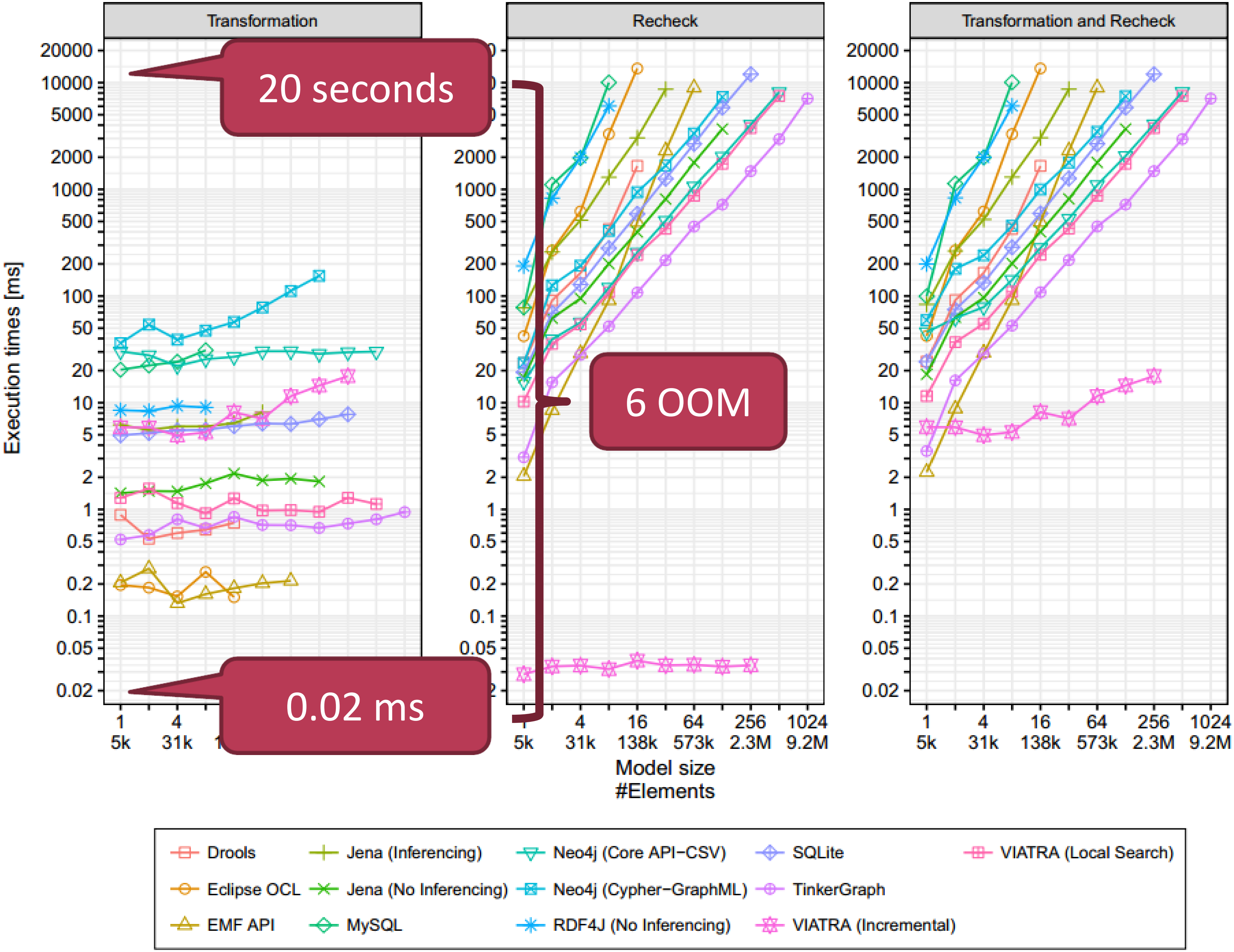

Transformation
Recheck
Transformation and Recheck
Execution times [ms]
20000
10000
5000
2000
1000
500
200
100
50
20
10
5
2
1
0.5
0.2
0.1
0.05
0.02
1
5k
4
31k
16
138k
64
573k
256
2.3M
1024
9.2M
Model size
#Elements
Drools
Eclipse OCL
EMF API
Jena (Inferencing)
Jena (No Inferencing)
MySQL
Neo4j (Core API−CSV)
Neo4j (Cypher−GraphML)
RDF4J (No Inferencing)
SQLite
TinkerGraph
VIATRA (Incremental)
VIATRA (Local Search)

0.02 ms



20 seconds

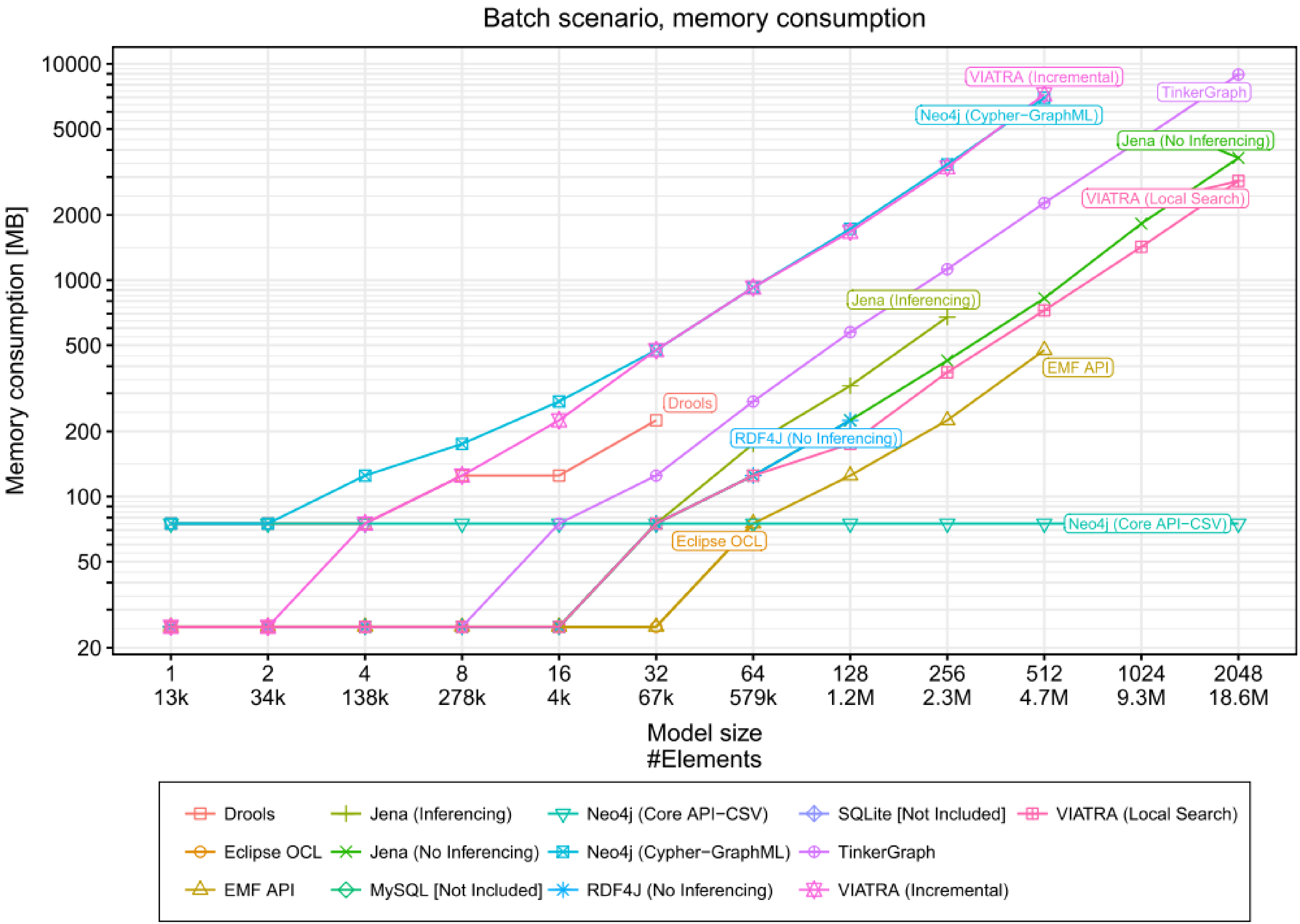
0.02 ms



20 seconds

6 OOM

0.02 ms


Batch scenario, memory consumption
Memory consumption [MB]
10000
5000
2000
1000
500
200
100
50
20
VIATRA (Incremental)
TinkerGraph
Neo4j (Cypher−GraphML)
Jena (No Inferencing)
VIATRA (Local Search)
Jena (Inferencing)
EMF API
Drools
RDF4J (No Inferencing)
Neo4j (Core API−CSV)
Eclipse OCL
1
13k
2
34k
4
138k
8
278k
16
4k
32
67k
64
579k
128
1.2M
256
2.3M
512
4.7M
1024
9.3M
2048
18.6M
Model size
#Elements
Drools
Eclipse OCL
EMF API
Jena (Inferencing)
Jena (No Inferencing)
MySQL [Not Included]
Neo4j (Core API−CSV)
Neo4j (Cypher−GraphML)
RDF4J (No Inferencing)
SQLite [Not Included]
TinkerGraph
VIATRA (Incremental)
VIATRA (Local Search)

Lessons learnt

Lessons learnt

- Contact tool authors for assistance

Lessons learnt

- Contact tool authors for assistance
- Very hard to tell the **actual** memory consumption
  - Manual calls to the garbage collector – not enough
  - Profiler – still not enough
  - Setting a hard limit is the best approach

Lessons learnt

- Contact tool authors for assistance
- Very hard to tell the **actual** memory consumption
  - Manual calls to the garbage collector – not enough
  - Profiler – still not enough
  - Setting a hard limit is the best approach
- Difficult to get **relevant results**

Lessons learnt

- Contact tool authors for assistance
- Very hard to tell the **actual** memory consumption
  - Manual calls to the garbage collector – not enough
  - Profiler – still not enough
  - Setting a hard limit is the best approach
- Difficult to get **relevant results**
  - Lots of factors can add noise to the results

Lessons learnt

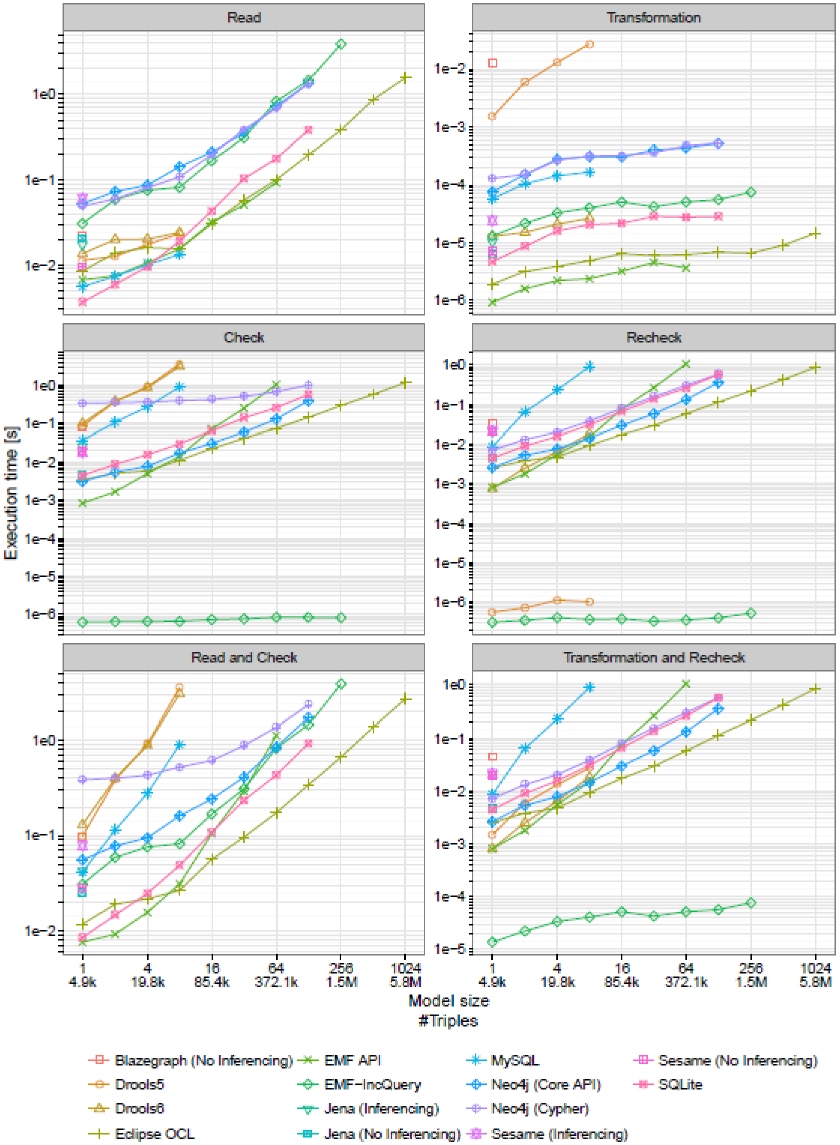
- Contact tool authors for assistance
- Very hard to tell the **actual** memory consumption
  - Manual calls to the garbage collector – not enough
  - Profiler – still not enough
  - Setting a hard limit is the best approach
- Difficult to get **relevant results**
  - Lots of factors can add noise to the results
  - Nothing works on the first try

Lessons learnt

- Contact tool authors for assistance
- Very hard to tell the **actual** memory consumption
  - Manual calls to the garbage collector – not enough
  - Profiler – still not enough
  - Setting a hard limit is the best approach
- Difficult to get **relevant results**
  - Lots of factors can add noise to the results
  - Nothing works on the first try
- Visualization helps **a lot**
  - Automate with R/Python script

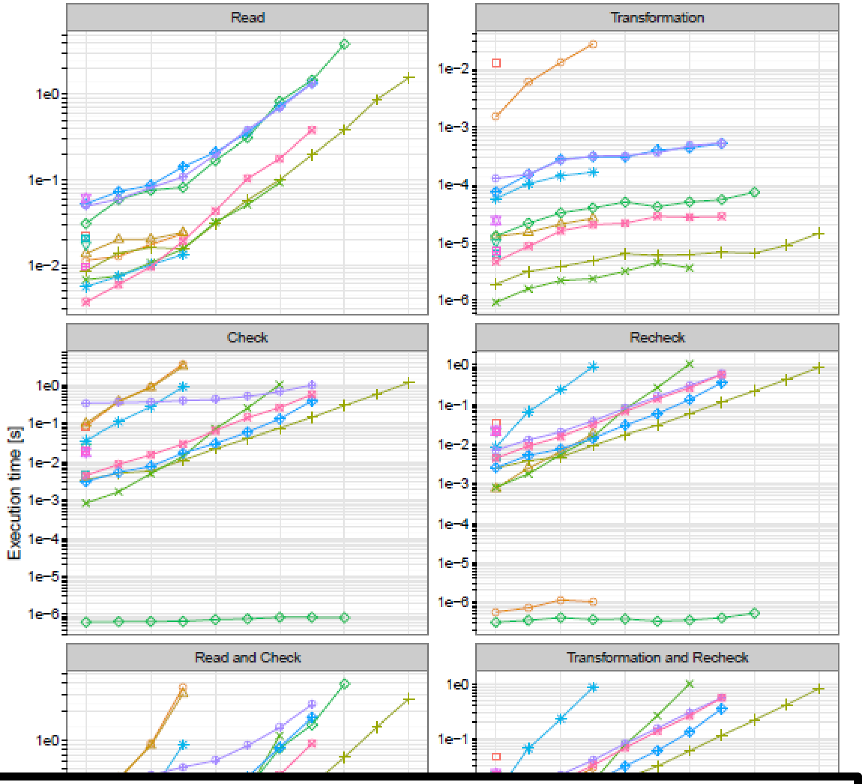
Train Benchmark summary

- Scalable model generator
  - 4 representation formats
- Realistic workload
  - 6 validation queries
  - 12 transformations
  - Implemented for 12+ tools
- Visualization and reporting



Train Benchmark summary

- Scalable model generator
  - 4 representation formats
- Realistic workload
  - 6 validation queries
  - 12 transformations
  - Implemented for 12+ tools
- Visualization and reporting



Gábor Szárnyas, Benedek Izsó, István Ráth, Dániel Varró: *The Train Benchmark: cross-technology performance evaluation of continuous model queries*, SOSYM 2017, http://link.springer.com/article/10.1007/s10270-016-0571-8