Progress Report - SR Challenge

Patrik Schneider[1,2]

Danh Le Phuoc[3]

(1) Institute of Information Systems, Vienna University of Technology, Austria

(2) Siemens CT, Austria

(3), TU Berlin, Germany

SR Workshop 2019, Linköping

The Story so far…

- SR Workshop 2018:
  - **Initial idea developed**: First day to collect ideas, benchmarks, evaluation platforms
  - Talks given by:
    Pavel Smirnov (Hobbit), Daniel de Leng (Robotics), Thu Le Pam (CityBench benchmark), Riccardo (Evaluation of stream processing systems), Danh (Social Network Stream benchmark
- Suggestion by Boris/Jacopo:
  - **System competition does not (yet) make sense**
  - Better conduct a modelling challenge as a hackathon aka challenge
  - Choosing one problem (e.g., C-ITS) and let teams **model & solve** it
- Development after SR 2018 until now:
  - Collection of problem description, scenario, tasks
  - Evaluation and rules
  - Data, platforms, systems
  - Long term goal: Write a journal paper on challenge

What do we need for Model & Solve Challenge?

- Identified the following points:
  - Two (or more) well-defined **domains**, where the skills can be shown…
  - Model & solve **tasks**, either given (later) or collected
  - **Platform** with stream generators and (possibly) a background model (KB)
  - **Procedure** on how to conduct the challenge
- Report on the progress so far:
  - Two domains, C-ITS and Social Network streams (with tasks and stream generator)
  - Suggestion of rules for the competition and an evaluation process
  - Platform candidates
  - Possible systems/teams that could participate
- Documented on:
  https://github.com/patrik999/stream-reasoning-challenge/blob/master/challenge-description.md

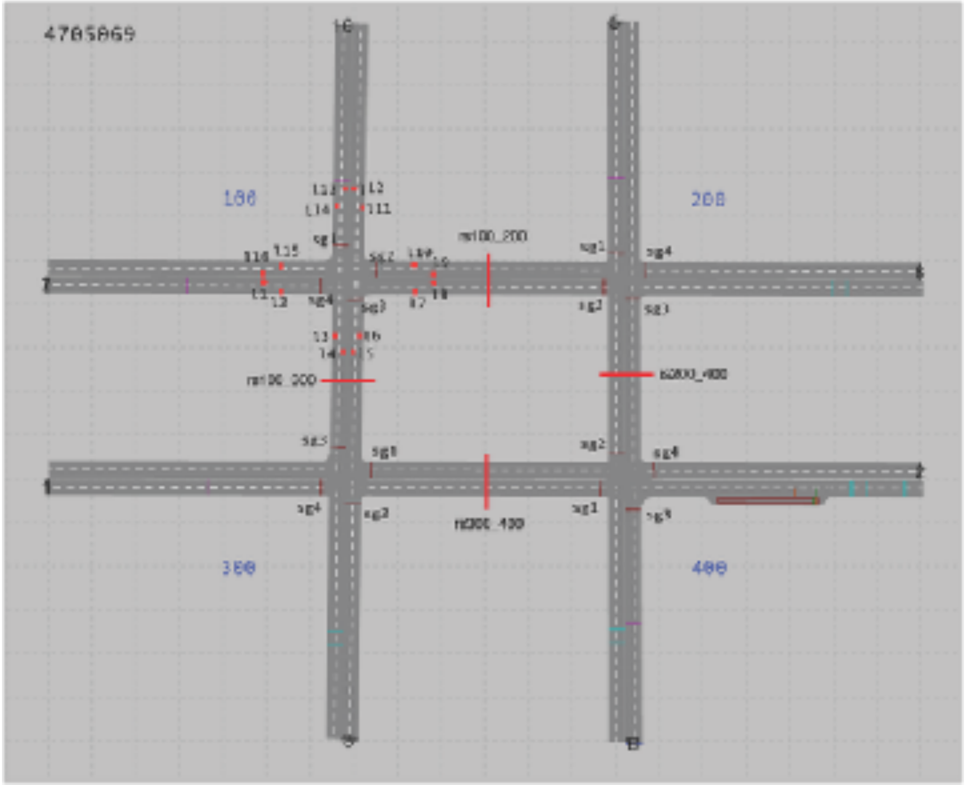
Scenario A - C-ITS

- Cooperative intelligent transportation systems (C-ITS)
- Overview:
  - One single (collective) sensor: V2X messages produced by each traffic participant
  - Spatio-Temporal and fast-streaming data
  - Complex (static) domain model
  - Autonomous actors (e.g., cars, buses, etc)
- Challenges:
  - **Intersection** topologies and complex road network of intersections
  - Signal plans can be complicated
  - Wide variety of task from **fast** detecting unexpected events (e.g. , accidents) to **slow** changing effects (e.g., traffic jam)

SATELLITE COMMUNICATIONS
TERRESTRIAL BROADCAST
Intermodal Communications
MOBILE
Navigation
MAN
Vehicle-to-Vehicle
Safety Systems
Passenger Information
Travel Assistance
WLAN
Traffic Signs
Pets on road warnin
Adaptive Cruise Control
Trip Planning
Bicycle
approaching
© ETSI 2008
Fleet Management
Toll Collection

Scenario A - Suggested Use Case

- Use of traffic simulation tools to generate data:
  - PTV Vissim (commercial)
  - DLR SUMO (open-source)
  - Connectors to generate log data for both, output JSON
- #-shaped street layout:
  - 4 intersections and 4 roads with 2 in/outgoing lanes
  - Road segments between intersections
  - Each intersection with 4 TLs and static signal plans
  - All geometries are defined (polygons)
  - Consistent naming
- Traffic flow:
  - Different types of vehicles (by colour)
  - Vehicles take several possible routes
  - Generated for light/medium/heavy traffic



Scenario A - Static Model (KB)

- Abstractly encoding of street layout as Datalog facts / RDF triples
- Idea: directly used by solvers
- Manually extracted from Vissim simulation
- Model:
  - Classes
  - Properties
  - Relations
  - Class hierarchies
- Geometries: Encoding as WKT (OGC standard)
- Output: JSON/LD, Datalog, CSV

```
mapIntersection(i100).
hasGeo(i100,"POLYGON((430.5 140, 520 140,
520 220, 430 220, 430 140))").

mapLaneIn(i100_l1).
mapLaneIn(i100_l2).
hasGeo(i100_l1,"POLYGON((441 168.5, 465
168.5, 465 172, 441 172, 441 168.5))").

connected(i100_l1,i100_l3).
connected(i100_l1,i100_l4).
isPartOf(i100_l8,i100).

mapSignalGroup(i100_sg1).
hasSignalGroup(i100_l1,i100_sg4).

speed(car_1, 20, 1001).
speed(car_1, 25, 1002). …
pos(car_1, “POINT(0 0)", 1001).
pos(car_1, “POINT(0 5)", 1002). …
```

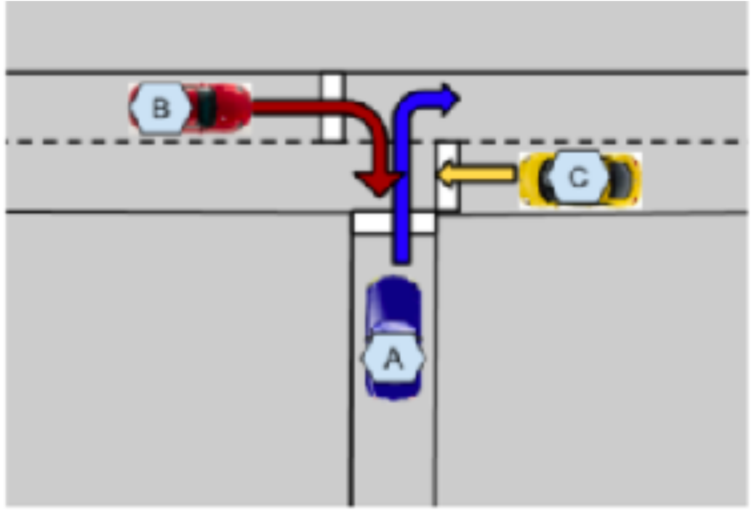
Scenario A - Dynamic Model (Streams)

- Streamed traffic data generated by the simulation tools Vissim or Sumo
- Two ways to feed the solvers…
- Replay from logs:
  - Area of cooperative intelligent transportation systems (C-ITS)
  - Recorded and replay by Python script,
  - Simple spatial relations (overlap, contains,…) materialised in script
- Output: JSON/LD, Datalog, CSV
- Direct from running Vissim/Sumo:
  - Use of interface/connector
  - Dynamic integration into tool
- Data model are annotated facts or triples:
  - `speed(car_1, 20, 1001).`
  - `pos(car_1, “POINT(0 0)", 1001).`

Scenario A - Modelling Task

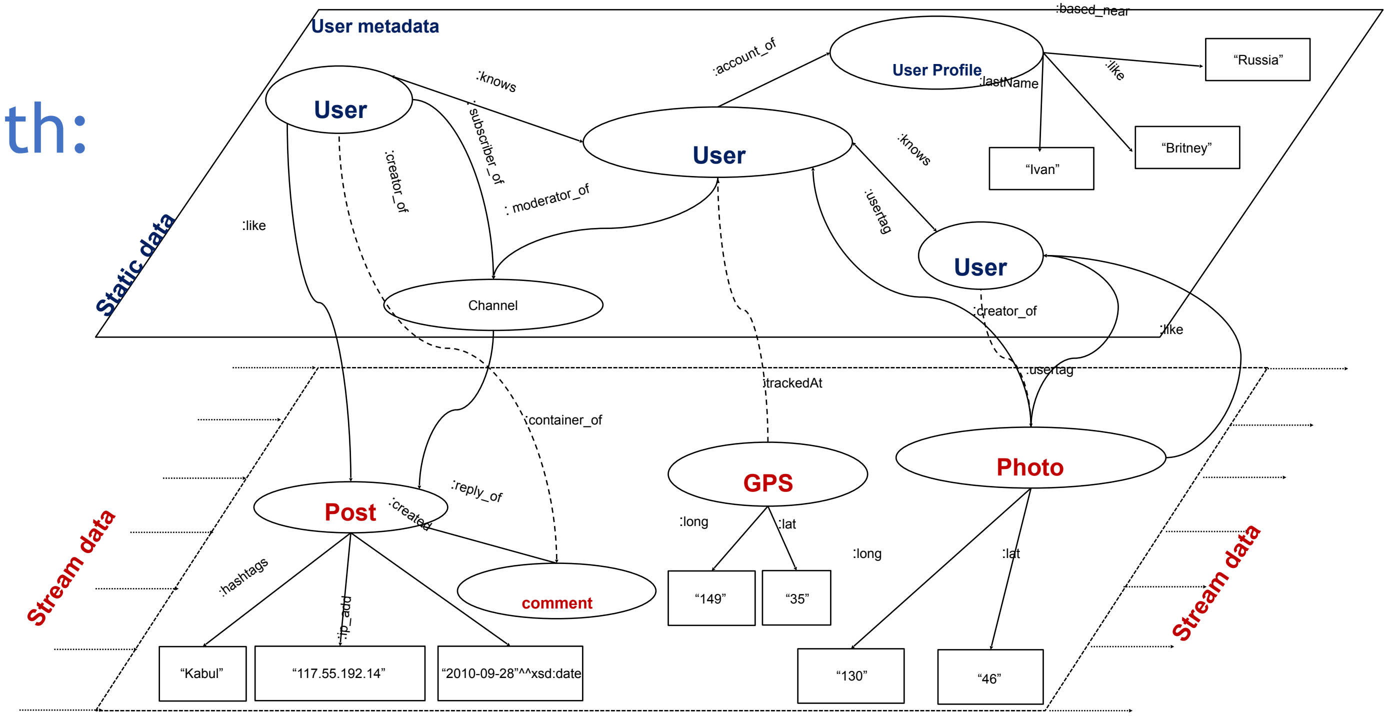
- C-ITS model & solve tasks, increase in difficulty:

- Task 1 (Traffic Statistics):
  - Calculating the number of vehicles and average speed on each intersection
  - One time or continues collect
  - Split by vehicle type, destination, …
- Task 2 (Detection wrong vehicle behaviour):
  - Event detecting can be formulated by different wrongdoings
  - Speeding on specific section, red light violation, U-turn, accident
- Additional Task 3 (Traffic Jam/Waves):
  - Detecting a traffic jam on an intersection
  - Need to take (valid) stops due to red lights into account
  - Detecting traffic waves (phantom traffic jams) more challenging
- Make the tasks harder by:
  - Addining noise, to simulate faulty sensors/measurements
  - Delete values to make streams sparser
  - Transient properties, e.g., road closure



Scenario B - Social Network Stream

- Social stream data of people
- Data generated by localised users connected to a social network
- Using data generator of LSBench
- Data generator emulates users with:
  - Their social media connections
  - Their posts with comments
  - Their locations



- Keep as possible extension, but ignore for now

Scenario B - Modelling Task

- Suggestion by Danh based on Social Network Stream

- Task 1 (Photo tagged by friends):
  - Notify if a user has been tagged in a photo
  - Within a day that his/her friend has liked the photo
- Task 2 (Comments liked):
  - Notify a person that all comments on a post of a channel that he/she is subscribed have been liked by friends
- Task 3 (Photo tagged close by):
  - Task 1, but the photo has to be tagged nearby
- Task 4 (Photo tagged by non-friends)
  - Task 1, but tagged by people that are not friends
- Task 5 (All posts and photos liked)
  - Notify a user of all the posts and photos liked by friends of his/her friends

Scenarios C - Combined

- Combination of Scenario A and B:
  - People in vehicle tweeting about traffic, events, etc.
  - Combine social media analysis with traffic information
  - Possible new tasks
- Combination of Scenario A with traffic video streams:
  - Feeds recorded by traffic cameras
  - Use Machine Learning (ML) directly
  - Build on top of ML results
- Aim beyond existing community by include other technologies:
  - Machine Learning tasks
  - Database (relational and graph) tasks
  - Robotics tasks
  - Cyber-physical Systems tasks

How to run a Challenge

- Plan to run it as a hackathon, but what is it really?
  - Hacking is creative problem solving
  - An event of limited duration where people come together to solve problems
- Important considerations, from https://hackathon.guide:
  1. Venue & date!
  2. Build anticipation
  3. Welcoming newcomers
  4. Cultivating good projects
     Clear, attainable, newcomers, well organized
  5. Can be hacking and training!
  6. Proper registration (e.g., Eventbrite)
  7. Tasks 10 days, 3 days, 1 day before, clear schedule

Realisation, Rules, and Evaluation

- For hackathon rules needed (adapted from the ASP challenge)
- Possible rules for a challenge::
  1. Organisers given a set of task (preselected or voted)
  2. Organizers set up and provide the evaluation platform
  3. Teams are allowed to use any solver (or solving script)
  4. Teams have to work out their own problem encoding
  5. Solutions should be presented at the end of the competition
  6. Evaluation of their solution (either by jury or voting)
- Any other rules/ideas?
  - One idea, was that teams have to use the other teams solvers (Boris/Jacopo)

Realisation, Rules, and Evaluation (cont.)

- What is evaluated?
  - Processing time
  - Completeness/problem coverage
  - Easiness of use
  - Elegancy/ingenuity of modelling
- How is evaluating?
  - Jury, and/or
  - Participants
- Some Reward?
  - Do we like to have awinner and price?
  - Invite the participants to join the journal publication?
- How do we build the teams?
  1. Decided beforehand (on sign-up), based on systems
  2. Teams build on the competition day
  3. Teams with swapping members

Evaluation Platform

- Agree on an evaluation platform (not yet)
- We have the following options:

  (1) Custom:
  - We provide our own (simple) platform
  - Use of own python scripts (exit) sending websocket messages

  (2) RSPLab / TripleWave
  - Tailored for SR evaluations

  (3) Hobbit
  - General purpose platform

- How is the platform hosted:
  - Online as web service
  - Offline, we provide either container or data files + scripts
  - Both

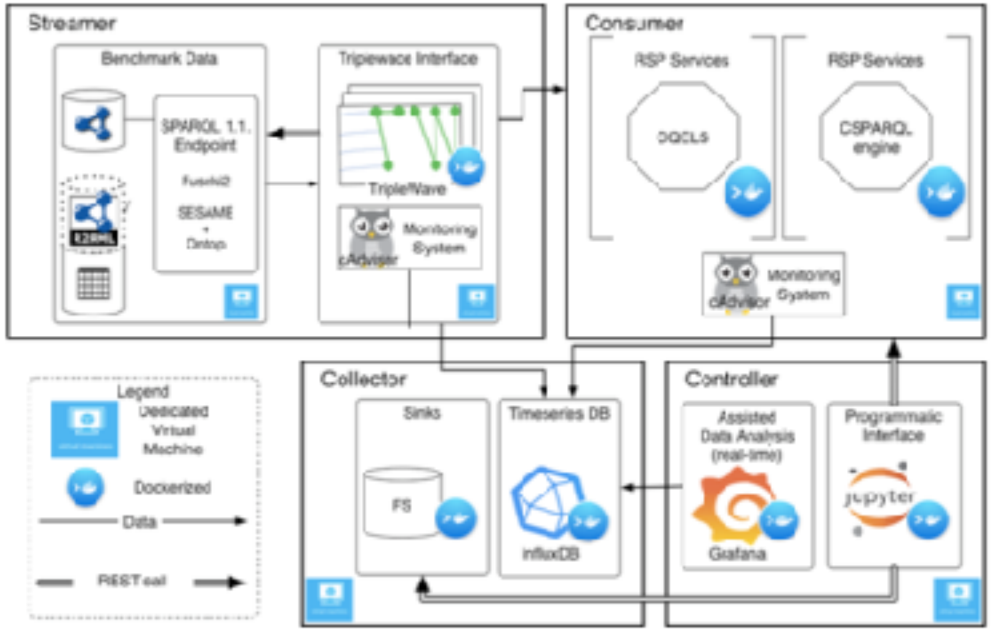


Fig. 1: RSPLab Architecture and Implementation

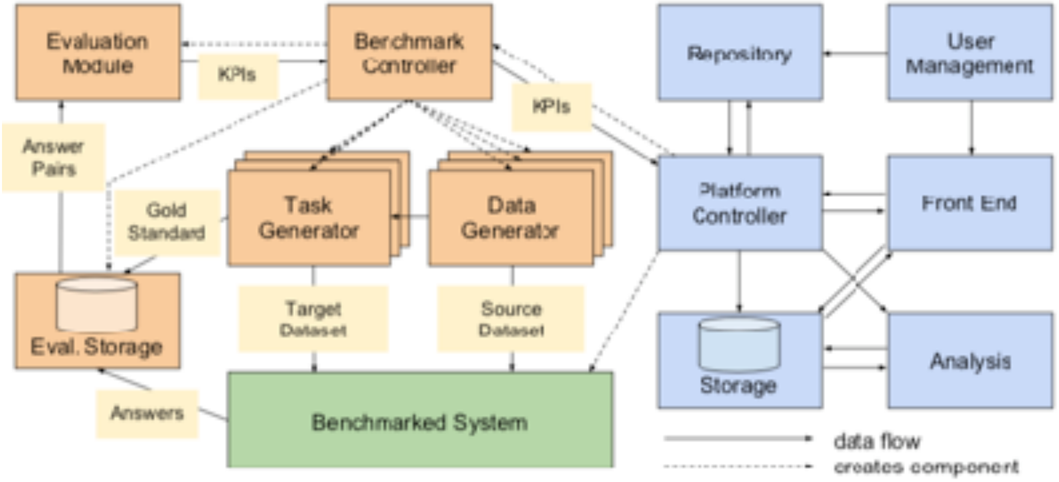
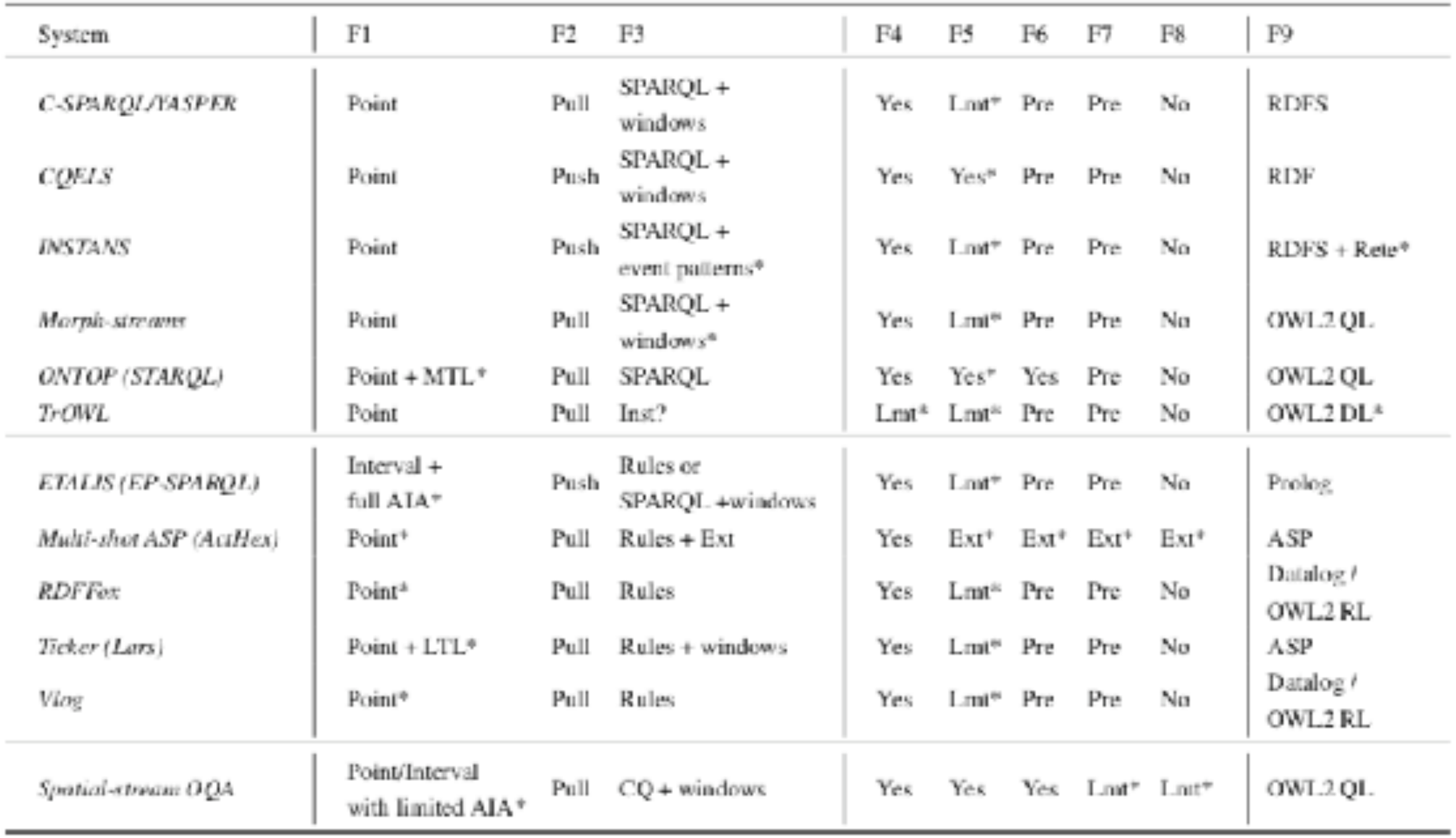


Fig. 1. Interaction of the components of HOBBIT Platform

Possible Teams/Systems

- We could provide a set of systems, or teams bring their own
- Wide variety of existing systems:

- Systems to consider:
  - CQELS (TU Berlin)
  - C-SPARQL/YASPER (Poly Milano)
  - Hexlite (TU Wien)
  - RDFox (Oxford)
  - Laser (VU Amsterdam)
  - Others ideas?



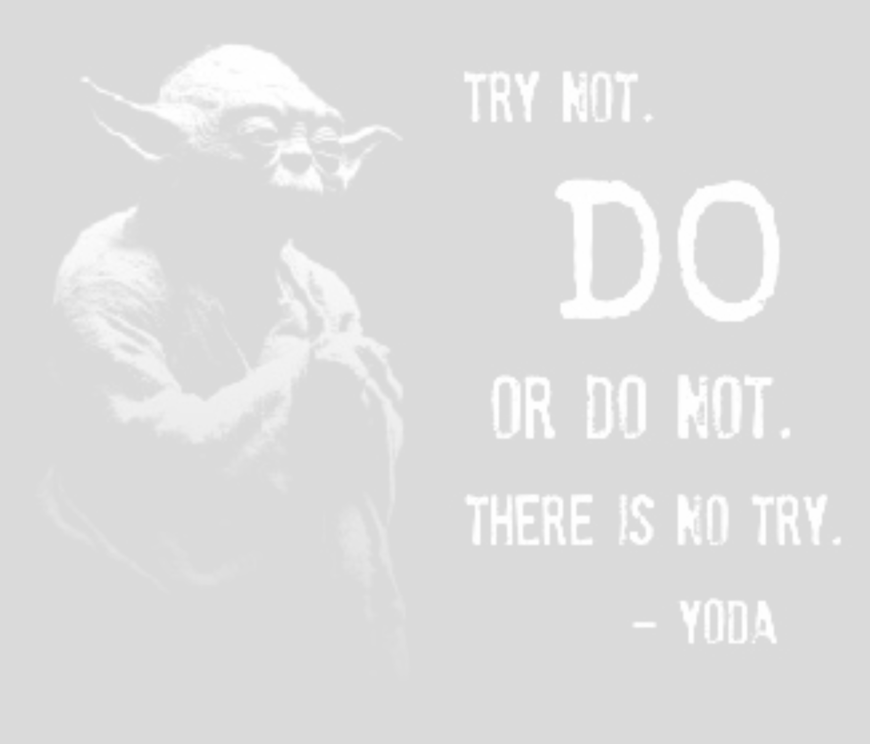
System	F1	F2	F3	F4	F5	F6	F7	F8	F9
C-SPARQL/YASPER	Point	Pull	SPARQL + windows	Yes	Lmt*	Pre	Pre	No	RDFS
CQELS	Point	Push	SPARQL + windows	Yes	Yes*	Pre	Pre	No	RDF
INSTANS	Point	Push	SPARQL + event patterns*	Yes	Lmt*	Pre	Pre	No	RDFS + Rete*
Morph-streams	Point	Pull	SPARQL + windows*	Yes	Lmt*	Pre	Pre	No	OWL2 QL
ONTOP (STARQL)	Point + MTL*	Pull	SPARQL	Yes	Yes*	Yes	Pre	No	OWL2 QL
TrOWL	Point	Pull	Inst?	Lmt*	Lmt*	Pre	Pre	No	OWL2 DL*
ETALIS (EP-SPARQL)	Interval + full AIA*	Push	Rules or SPARQL +windows	Yes	Lmt*	Pre	Pre	No	Prolog
Multi-shot ASP (ActHex)	Point*	Pull	Rules + Ext	Yes	Ext*	Ext*	Ext*	Ext*	ASP
RDFFox	Point*	Pull	Rules	Yes	Lmt*	Pre	Pre	No	Datalog / OWL2 RL
Ticker (Lars)	Point + LTL*	Pull	Rules + windows	Yes	Lmt*	Pre	Pre	No	ASP
Vlog	Point*	Pull	Rules	Yes	Lmt*	Pre	Pre	No	Datalog / OWL2 RL
Spatial-stream OQA	Point/Interval with limited AIA*	Pull	CQ + windows	Yes	Yes	Yes	Lmt*	Lmt*	OWL2 QL

Table 3

Qualitative comparison of our features on languages/systems (* indicates comments in the paragraph on systems. *Ext*, *Pre*, and *Lmt* means evaluation by external atoms, preprocessing needed, and limited coverage)

Conclusion (before)

- Need to be discussed and agreed on open questions…
- All documented on: https://github.com/patrik999/stream-reasoning-challenge/blob/master/challenge-description.md
- Need to define next steps… Fix date and location:
  - Colocated with next SR workshop
  - Colocated with ISWC/ESWC
  - Independent event in Berlin/Wien
- Team commitment important!



Conclusion (decided at final session)

1. The challenge will be collocated with SR workshop 2020
2. We will have the model & solve challenge on the C-ITS scenario with the tasks:
   - Collect traffic statistics (Task 1)
   - Detect traffic event (Task 2)
   - Detect traffic congestions (Bonus, Task 3)
3. Data generated by traffic simulation with output: JSON/LD, Datalog, CSV
4. Rules: standard hackathon with solver tutorials first, and evaluation of ingenuity/easiness of use by jury and participants
5. Commitment with teams including solvers important:
   - CQELS (TU Berlin)
   - C-SPARQL/YASPER (Poly Milano)
   - Hexlite (TU Wien)
   - RDFox (Oxford)
   - Laser (VU Amsterdam)
   - Others have to be asked (RDFox)