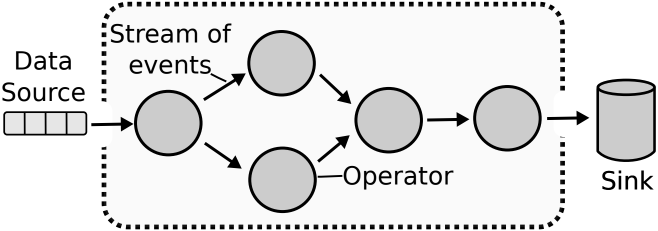
Latency-Aware Placement of Data Stream Analytics on Edge Computing

Alexandre da S. Veith, Marcos D. de Assunção, Laurent Lefèvre

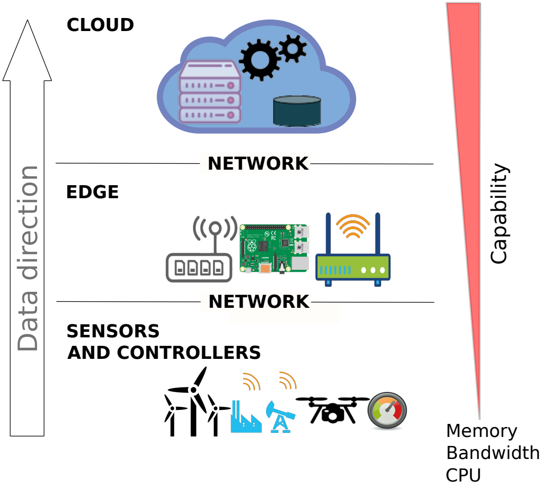
alexandre.veith@ens-lyon.fr

13th November 2018

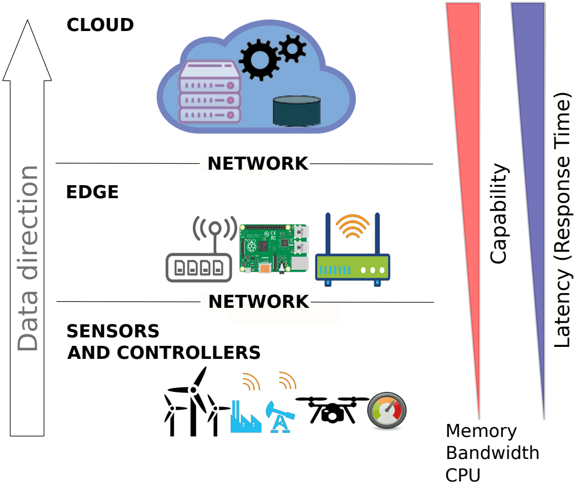
Data Stream Analytics



Infrastructure for Deploying Data Stream Analytics

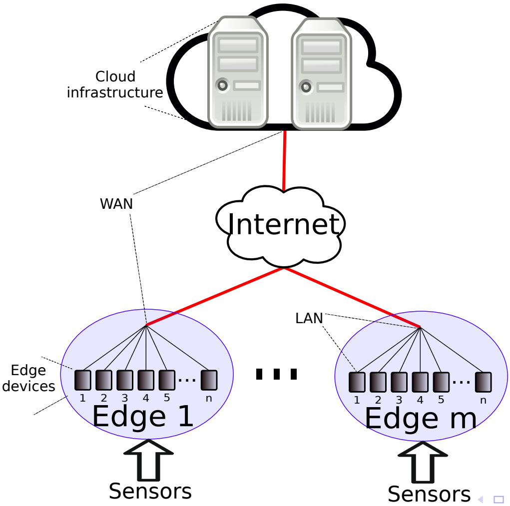


Infrastructure for Deploying Data Stream Analytics

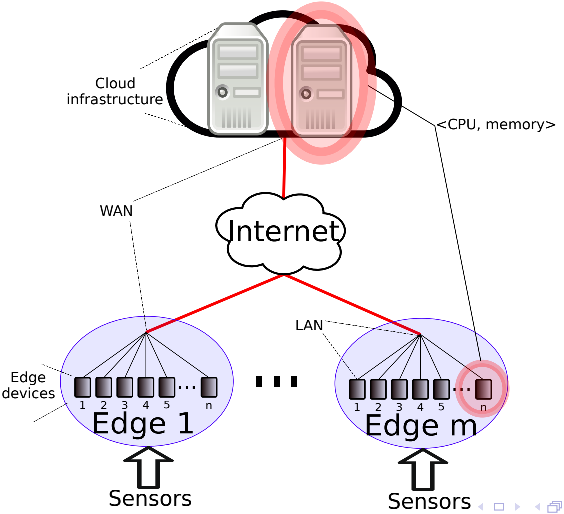


How to **split** the data stream operators **dynamically** onto edge and cloud reducing the response time and respecting the environment constraints?

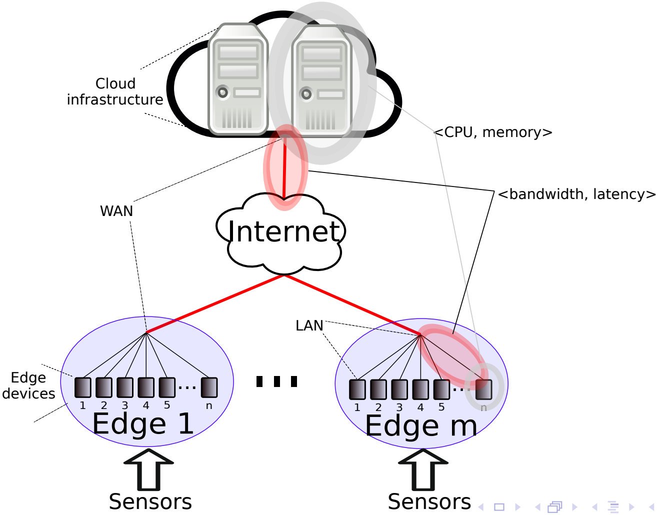
Infrastructure Topology



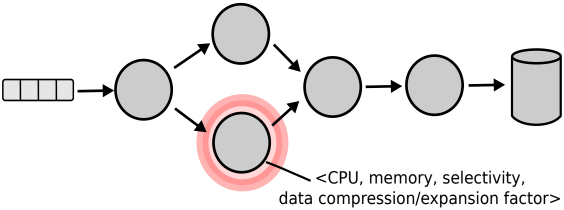
Infrastructure Topology



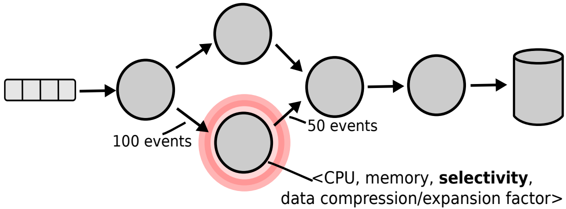
Infrastructure Topology

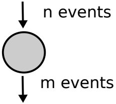


Application Topology



Application Topology

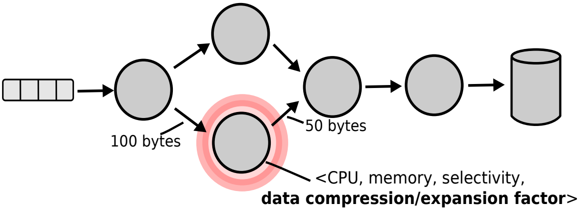


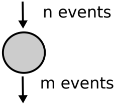


Selectivity

The ratio of number of
input events to output events

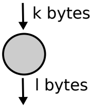
Application Topology





Selectivity

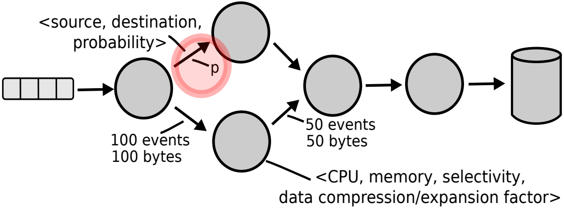
The ratio of number of input events to output events

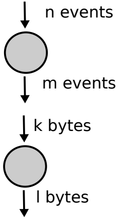


Data compression/ expansion factor

The ratio of the size of input events to the size of output events

Application Topology





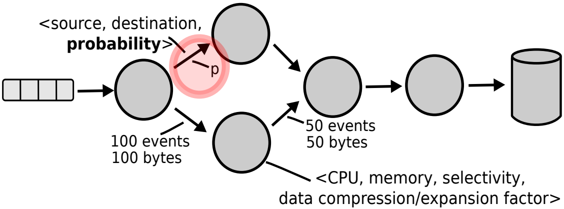
Selectivity

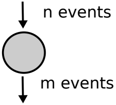
The ratio of number of
input events to output events

Data compression/ expansion factor

The ratio of the size of input events
to the size of output events

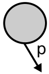
Application Topology





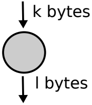
Selectivity

The ratio of number of input events to output events



Probability

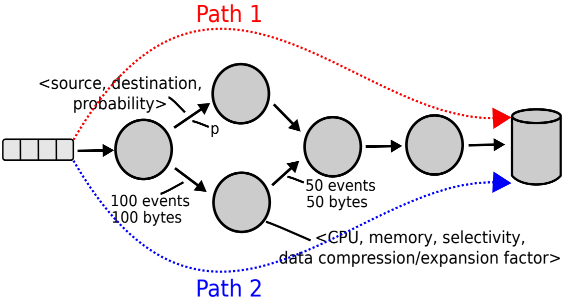
Probability of an event crosses the stream

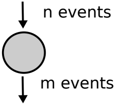


Data compression/ expansion factor

The ratio of the size of input events to the size of output events

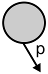
Application Topology





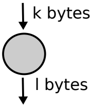
Selectivity

The ratio of number of
input events to output events



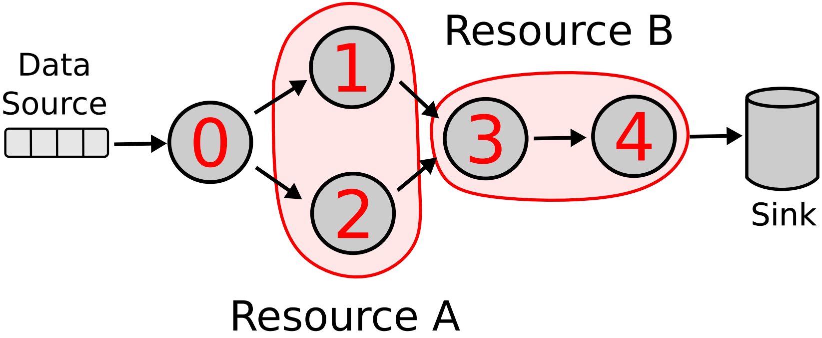
Probability

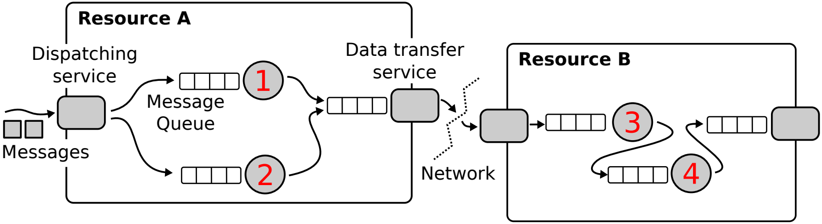
Probability of an event crosses the stream



Data compression/ expansion factor

The ratio of the size of input events
to the size of output events


Data
Source
0
1
2
3
4
Resource B
Resource A
Sink



- Queues for **Computation**(operator) and **Communication**(data transfer service)
- Model is based on **Queueing Theory** - **M/M/1**
- **Memory constraint** is based on the queues sizes
- **Response time** is equal to the sum of computation and communication into a path

We aim to **minimize the sum of the response times** (all paths)

Response Time Rate (RTR) Strategy

Function *EstimateResponseTimes*$(\mathcal{N} = (\mathcal{R}, \mathcal{L}), \mathcal{G} = (\mathcal{O}, \mathcal{S}), o)$

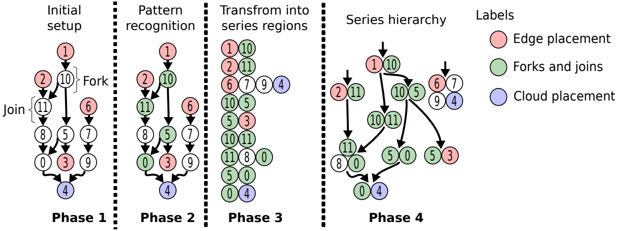
- **for** $child \in \langle o, * \rangle \subset \mathcal{S}$ **do**
  - $upstreams \leftarrow \langle child, r \rangle, \forall r \in \mathcal{R}$ ***and*** $mo_{\langle child, r \rangle} = 1$
- **for** $r \in \mathcal{R}$ **do**
  - $comm \leftarrow 0$
  - **for** $mapping \in upstreams$ **do**
    - **if** `GetHost`$(mapping) \neq r$ **then**
      - $com \leftarrow comm + ctime_{\langle mapping \rangle \langle o, r \rangle}$
  - **if** `MeetConstraints` **then**
    - $rt \leftarrow rt \cup \langle r, stime_{\langle o, r \rangle} + comm \rangle$
- **return** rt

- The strategy organizes the operator deployment sequence using BFS-Traversal algorithm
- For each operator in the operator deployment sequence
  - Computation and communication estimation for all resources
  - Evaluate memory, CPU, and bandwidth constraints
  - Resource with shortest required time (computation + communication) is elected to host the operator

RTR with Region Patterns (RTR+RP) Strategy

RTR+RP reduces the complexity of RTR by giving priority to the location of the sink placement (**edge has higher priority**)

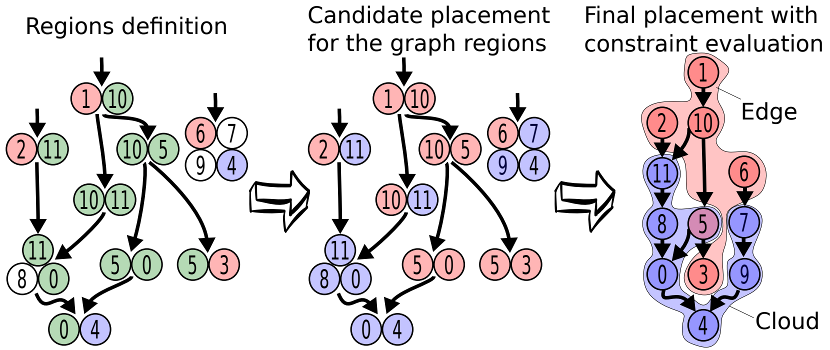
Patterns recognition based on the application behavior (forks and joins), and the location of data sources and sinks

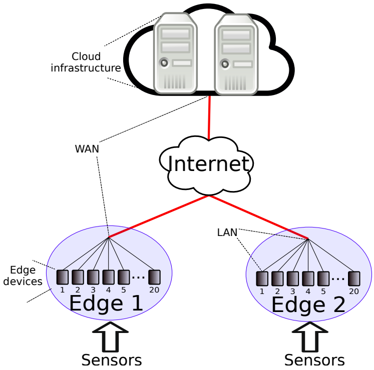


Decompose the application graph following the parallel regions (series-parallel decomposition graphs)

RTR with Region Patterns (RTR+RP) Strategy

- Split the application graph following the pathways
- Calculate the Response Time Rate only to the edge side





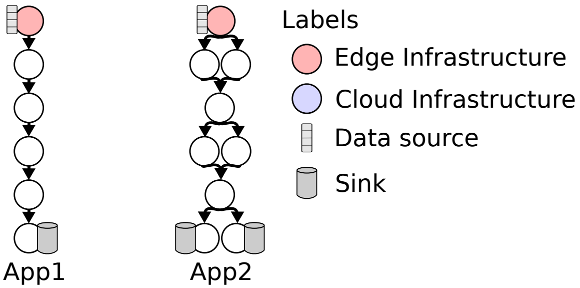
- Discrete event simulation;
- Edge: Two sites with 20 Raspberry PI 2 (4,74 MIPS at 1GHz and 1GB of RAM);
- Cloud: Two AMD RYZEN 7 1800x (304,51 MIPS at 3.6GHz and 1TB of RAM);
- LAN: Latency U(0.015-0.8)ms and bandwidth equal to 100 Mbps;(*)
- WAN: Latency: U(65-85)ms and bandwidth equal to 1 Gbps.(*)

* Hu, W., Gao, Y., Ha, K., Wang, J., Amos, B., Chen, Z., Pillai, P., Satyanarayanan, M.: *Quantifying the impact of edge computing on mobile applications.* In: 7th ACM SIGOPS Asia-Pacific Wksp on Systems. pp. 5:1–5:8. APSys '16, ACM, New York, NY, USA (2016)

Response time: end-to-end latency from the time events are generated to the time they reach the sinks.
Comparison: To demonstrate the gains obtained by our approach, we compared the proposed strategies against:

- Traditional approach (**cloud-only**) which deploys all operators in the cloud, apart from operators provided in the initial placement;
- **Taneja et. al. (LB)** which iterates a vector containing the application operators, gets the middle host of the computational vector and evaluates CPU, memory, and bandwidth constraints to obtain the operator placement.

Taneja, M., Davy, A. "*Resource aware placement of iot application modules in fog-cloud computing paradigm*". In: IFIP/IEEE Symp. on Integrated Net. and Service Management (IM). pp. 1222–1228 (May 2017)



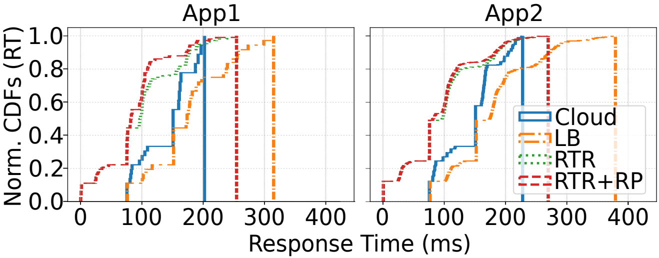
Events sizes: text - 10 bytes, pictures/objects - 50KB, and voice records - 200KB

Input event rates: Each event size has three input event rates

CPU requirements: 10 bytes - 3.7952 IPS, 50 KB-18976 IPS, and 200 KB - 75904 IPS

Selectivity: 100, 75, 50 and 25%

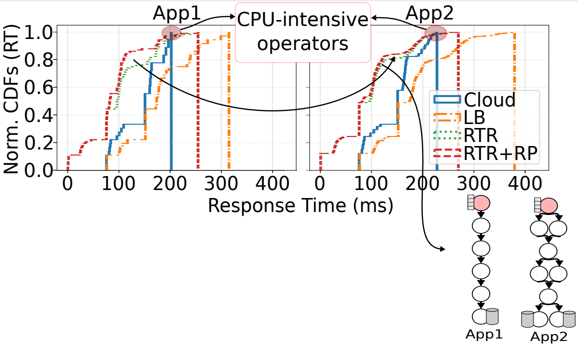
Data compression factor: 25, 50 and 75%



432 experiments (4 selectivities, 4 data compression rates, 3 input event rates, 3 sink locations and 3 input event sizes)

RTR and RTR+RP have shown to be over 95% more efficient than cloud-only approach and LB

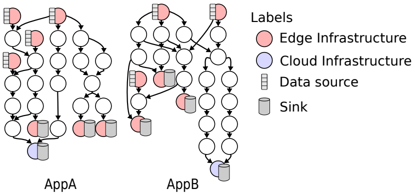
Cloud-only achieved 5% better results (when the blue line crosses the red at approx. 200ms)



432 experiments (4 selectivities, 4 data compression rates, 3 input event rates, 3 sink locations and 3 input event sizes)

RTR and RTR+RP have shown to be over 95% more efficient than cloud-only approach and LB

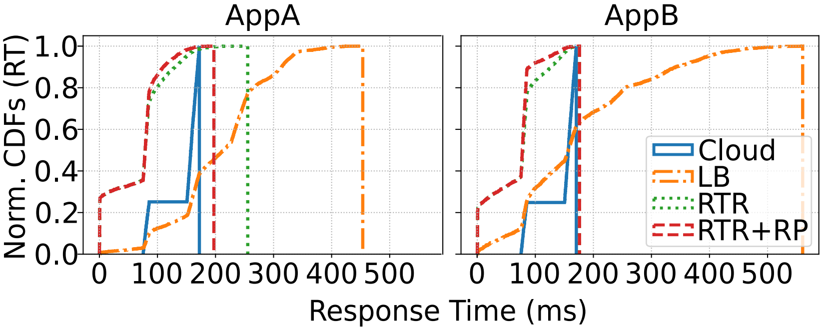
Cloud-only achieved 5% better results (when the blue line crosses the red at approx. 200ms)



Parameter	Value
cpu	1-100
Data compression rate	0%-90%
mem	100-7500
Input event size	100-2500
Selectivity	10%-100%
Input event rate	1000-10000

This scenario presents multiple operator behaviors and larger numbers of operators

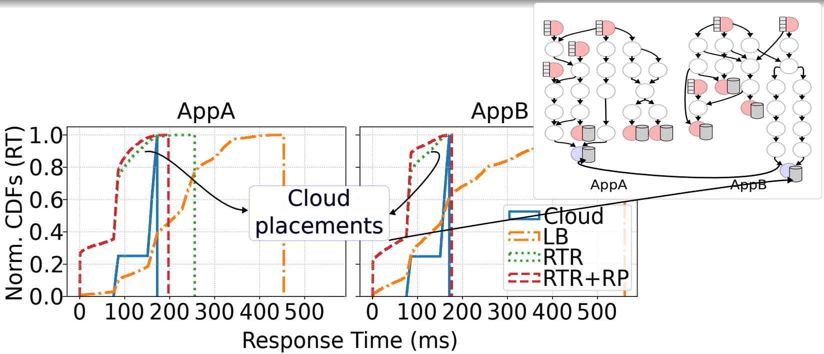
- Parameters of the operators vary using a uniform distribution with the ranges presented in the table
- The edges host the sink and source placements, except for the sink on the critical path which are hosted on the cloud



Our strategies outperformed in over **50%** and **57%** the cloud-only and the LB, respectively.

The communication overhead for sinks placed on edge at cloud-only was about **160 ms**, and RTR+RP was **76 ms**.

Our solution outperformed cloud-only in up to **52%**, but sinks on the cloud, RTR+RP had a slight performance loss of **3%**.



Our strategies outperformed in over **50%** and **57%** the cloud-only and the LB, respectively.

The communication overhead for sinks placed on edge at cloud-only was about **160 ms**, and RTR+RP was **76 ms**.

Our solution outperformed cloud-only in up to **52%**, but sinks on the cloud, RTR+RP had a slight performance loss of **3%**

Conclusions and Future Work

Summary

- A model and the DSP placement problem formalization
- Two strategies to improve the response time
- A performance comparison using a simulated environment

Conclusions

- The key behaviors (forks and joins) of the dataflows directed us to our strategies
- Our strategies using the dataflow aspects allow us to be 50% better in response time

Future Work

- An evaluation using a real environment
- Determine the optimal value and compare with our solutions
- A model to deal with the reconfiguration phase

Questions?