Multi-Objective Reinforcement Learning for Reconfiguring Data Stream Analytics on Edge Computing

Alexandre Veith, Felipe Souza, Marcos Assuncao, Laurent Lefevre, Julio Anjos

alexandre.veith@ens-lyon.fr [ICPP'2019]

9th August 2019

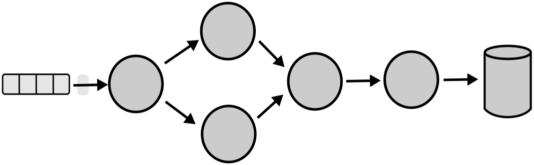
Agenda

Data Stream Processing (DSP) Application

- **Long-running** application
- Data is processed as it is generated
- **(Near) real-time** data analytics
- DSP application must handle **continuous and unbounded** data streams with **overwhelming rates**

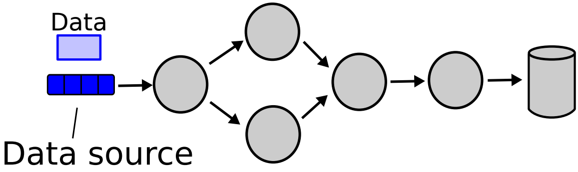
Data Stream Processing (DSP) Application

- Under most frameworks, a DSP application is structured a dataflow



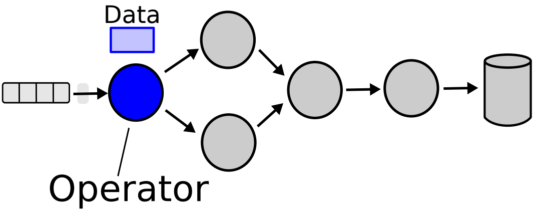
Data Stream Processing (DSP) Application

- Under most frameworks, a DSP application is structured a dataflow
  - Data sources



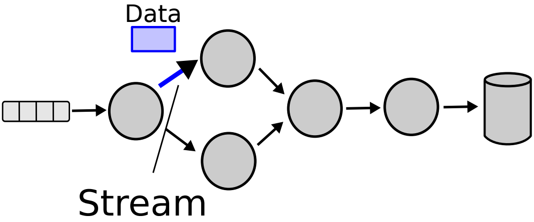
Data Stream Processing (DSP) Application

- Under most frameworks, a DSP application is structured a dataflow
  - Data sources
  - Operators (stateless and stateful)



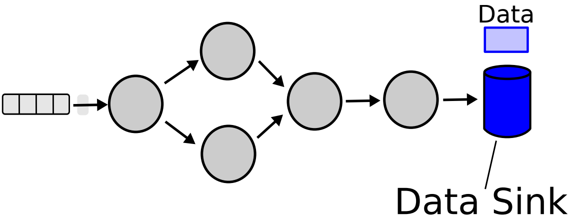
Data Stream Processing (DSP) Application

- Under most frameworks, a DSP application is structured a dataflow
  - Data sources
  - Operators
  - Streams

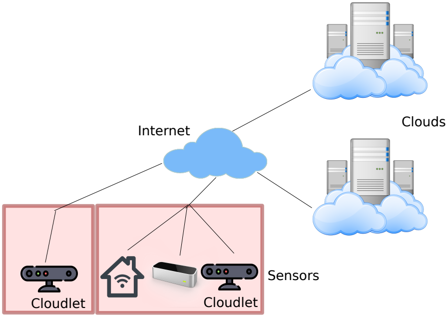


Data Stream Processing (DSP) Application

- Most of frameworks view the DSP application as a dataflow
  - Data sources
  - Operators
  - Streams
  - Data sinks

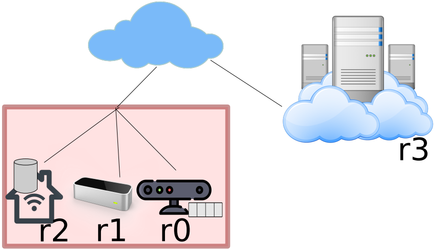


Infrastructure Topology



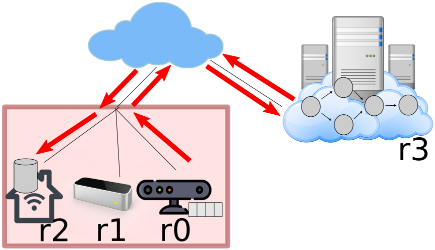
Infrastructure Topology

- Initial operator placement (application configuration)



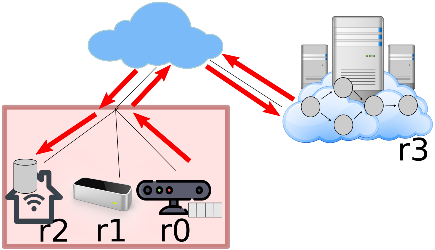
Infrastructure Topology

- Initial operator placement (application configuration)
  - The whole application dataflow is placed on the cloud (traditional)



Infrastructure Topology

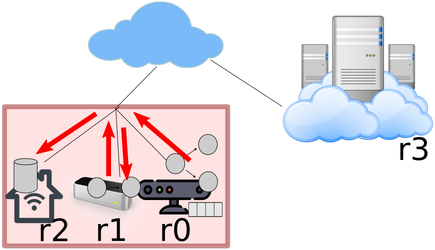
- Initial operator placement (application configuration)
  - The whole application dataflow is placed on the cloud (traditional)



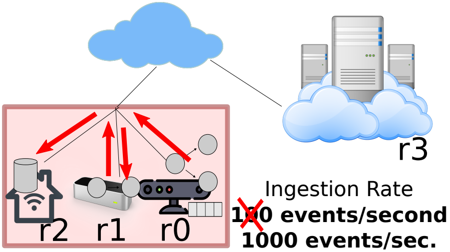
High communication overhead [Hu+16] - hard to achieve (near) real-time data analytics

Infrastructure Topology

- Initial operator placement (application configuration)
  - The whole application dataflow is placed on the cloud (traditional)
  - Exploring edge devices
    - Devices have limited CPU, memory and bandwidth capabilities



Application Reconfiguration



Application Reconfiguration

The process for reorganising or migrating DSP application operators across computing resources.

Application Reconfiguration

Problem

How to reconfigure the DSP application by considering multiple QoS metrics and by meeting infrastructure and application requirements?

Application Reconfiguration

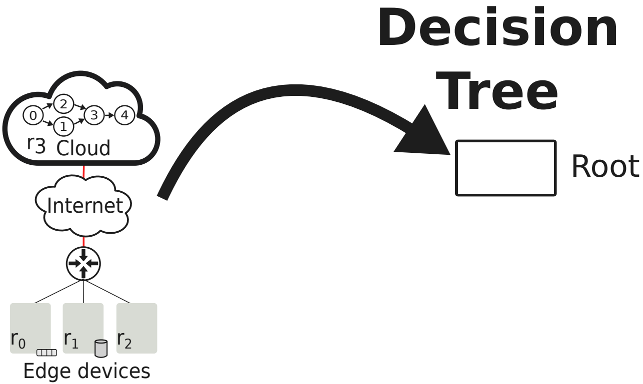
Problem

How to reconfigure the DSP application by considering multiple QoS metrics and by meeting infrastructure and application requirements?

Involved challenges:

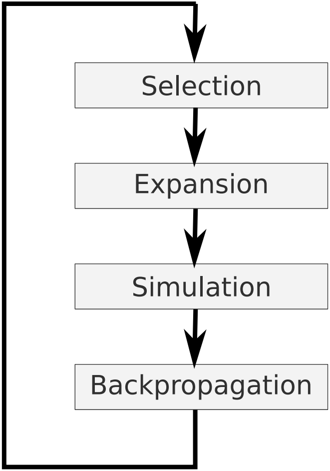
- Reconfiguration overhead (mainly because of stateful operators);
- Large search space to determine new operator placement plans; and
- Limitations of edge devices.

Monte-Carlo Tree Search (MCTS)



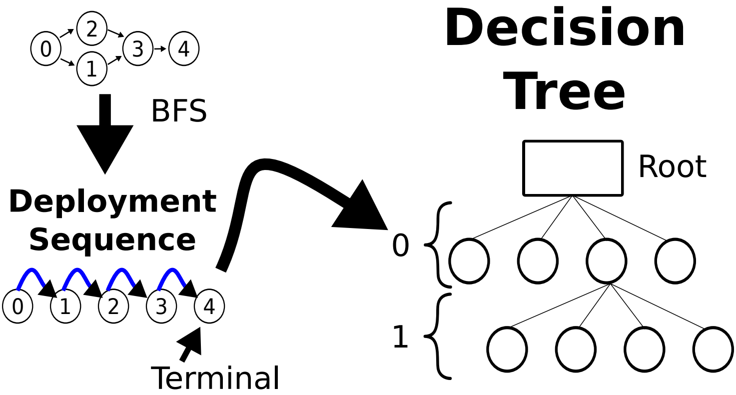
- The **root** of the tree refers to the current deployment

Monte-Carlo Tree Search (MCTS)



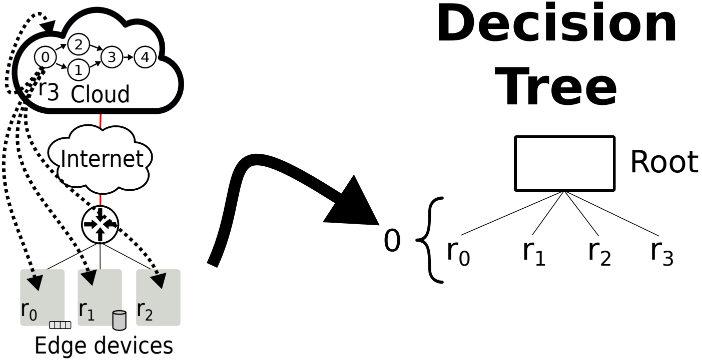
- A budget within a number of iterations is used to execute the MCTS loop
- The algorithm builds a decision tree with possible reconfiguration deployments

Monte-Carlo Tree Search (MCTS)



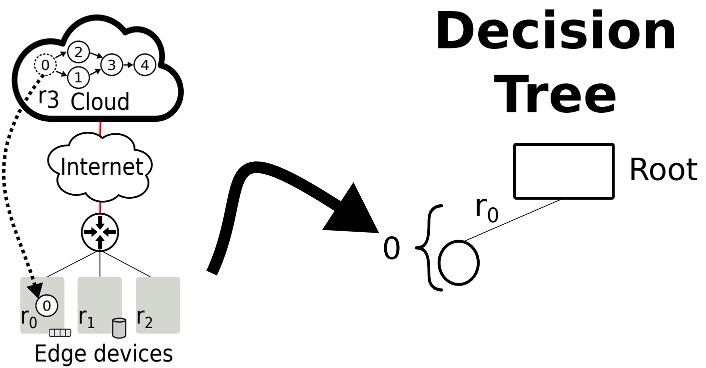
- The **deployment sequence** is used to create the tree levels

Monte-Carlo Tree Search (MCTS)



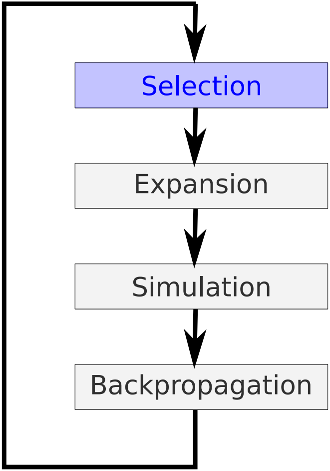
- **Edges** are the available actions to move or not an operator from one resource to another

Monte-Carlo Tree Search (MCTS)



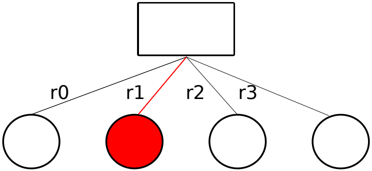
- **Node** comprises the state of the new deployment and variables for supporting future decisions

Monte-Carlo Tree Search (MCTS)

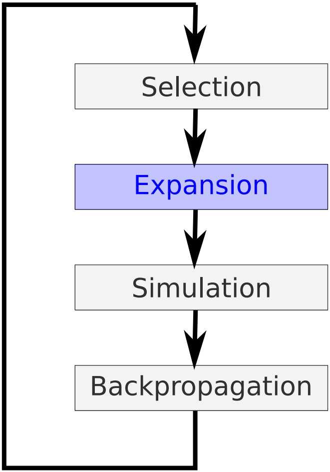



```
while node is not terminal do
    if node is fully expanded then
        Get the most promising node^a;
    else
        return node;
    end
end
```

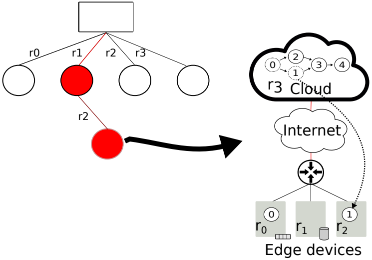
[a] Upper Confidence Bound for Trees - UCB1 algorithm [SB18]



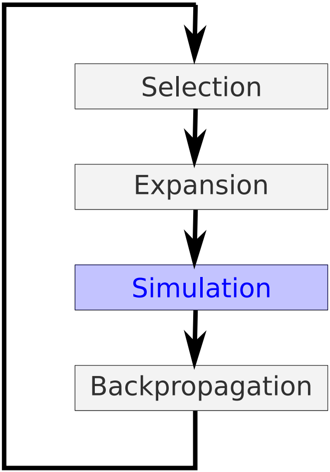
Monte-Carlo Tree Search (MCTS)



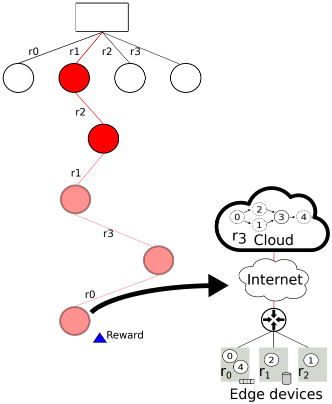
1 choose an untried action;
2 add a new child to the selected node;
3 initiate the decision variables;
4 return new node;



Monte-Carlo Tree Search (MCTS)



while *node is not terminal* **do**
  Choose an action randomly;
end
Simulate the new placement and determine its reward;



Quality of Service Metrics

The reward considers the following QoS metrics:

- **Response time**
  end-to-end latency from the time events are generated to the time they reach the sinks.

Quality of Service Metrics

The reward considers the following QoS metrics:

- **Response time**
  end-to-end latency from the time events are generated to the time they reach the sinks.
- **WAN traffic**
  data volume crossing WAN network links.

Quality of Service Metrics

The reward considers the following QoS metrics:

- **Response time**
  end-to-end latency from the time events are generated to the time they reach the sinks.
- **WAN traffic**
  data volume crossing WAN network links.
- **Monetary cost**
  calculates the monetary cost using the number of connections and exchanged messages across the cloud and the edge, and vice-versa (*i.e.* IoT Hub Based).

Quality of Service Metrics

The reward considers the following QoS metrics:

- **Response time**
  end-to-end latency from the time events are generated to the time they reach the sinks.
- **WAN traffic**
  data volume crossing WAN network links.
- **Monetary cost**
  calculates the monetary cost using the number of connections and exchanged messages across the cloud and the edge, and vice-versa (*i.e.* IoT Hub Based).
- **Reconfiguration Overhead**
  The total downtime incurred by migrating operator code and operator states.

Single aggregate cost based on **Simple Additive Weighting method** [Yoo+95] covers the four QoS metrics:

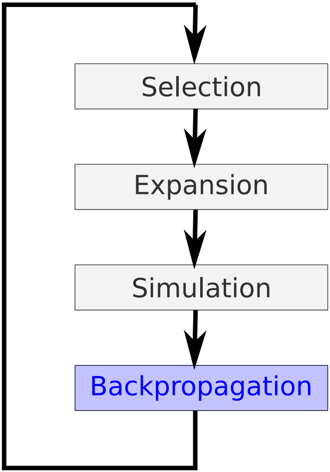
$w_0 \times$ **Response time** $+ w_1 \times$ **WAN traffic**$+$
$w_2 \times$ **Monetary cost** $+ w_3 \times$ **Reconfiguration Overhead**

where w corresponds to weight assigned to the metric.

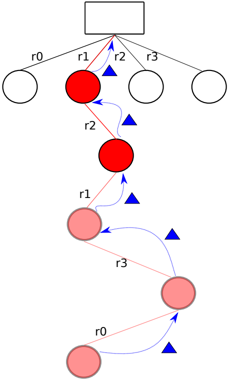
Reward

The Reward is the difference between the **single aggregate cost** of the current deployment and the **single aggregate cost** of new deployment.

Monte-Carlo Tree Search (MCTS)

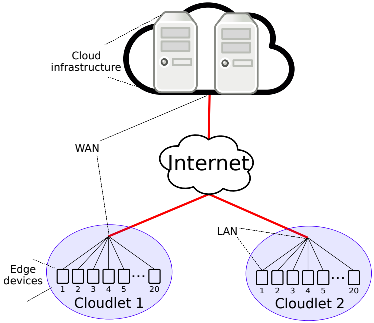



```
node ← current node;
while node is not the root do
    update node decision variables;
    node ← parent of node;
end
```



- Simulation tool developed atop **OMNET++**.
- One iteration of a Monitoring, Analysis, Planning and Execution (**MAPE**) loop is considered for operator reconfiguration:
  - Analysed when the execution reaches **300** seconds or all application paths have processed **500** messages; whichever comes last.
  - RL algorithms with a computational budget of **10,000** iterations.

Experimental Setup



- 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 [Hu+16]: Latency **U(0.015-0.8)ms** and bandwidth equal to 100 Mbps;
- WAN [Hu+16]: Latency **U(65-85)ms** and bandwidth equal to 1 Gbps.

MCTS Methods

Another version of our Reinforcement Learning approach:

- **TDTS-Sarsa(λ)** creates intermediary scores for each operator movement and employs them when estimating the reward.

Initial placement policies:

- **Traditional** method deploys the whole application dataflow on the cloud.
- **LB** [TD17] method evaluates edge devices capacity to place operators on edge (memory and CPU).

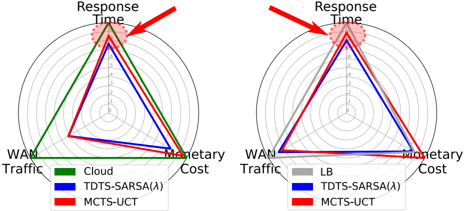
Evaluated Applications

Parameter	Value	Unit
cpu	1000-10000	Instructions per second
Data compression rate	10-100	%
mem	100-7500	bytes
Input event size	100-2500	bits/second
Selectivity	10-100	%
Input event rate	1000-10000	Number of messages
ws (window size)	1-100	Number of messages

- **Eleven application graphs** with single and multiple data paths are considered.
- The number of **stateful** operators corresponds to **20%** of the whole application.

Traditional and State-of-the-art

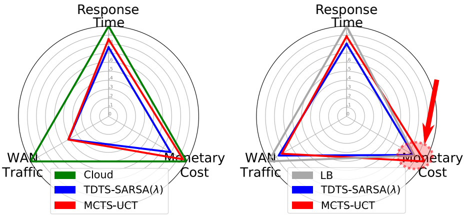
Response time with weight equal to 1.



RL algorithms achieved over **20% better response time**, and reduce the WAN traffic by over **50%** and the monetary cost by **15%** when comparing to Cloud approach.

Traditional and State-of-the-art

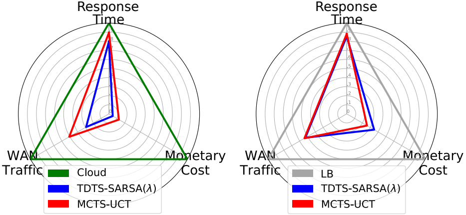
Response time with weight equal to 1.



Single-criterion optimisation fails in bringing **monetary cost and WAN traffic guarantees**. Monetary cost increased by over **15%**.

Traditional and State-of-the-art

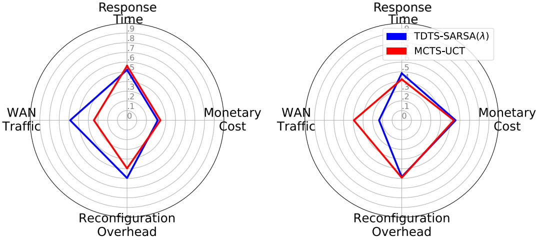
Cloud and LB with weights for response time, monetary cost and WAN traffic equal to 0.33.



The **multi-objective** approach provides a **holistic view** of the environment and allows for optimising **multiple metrics simultaneously** while avoiding unwanted spikes of monetary cost and WAN traffic.

Multiple Weights to QoS Metrics

Cloud and LB with weights 0.4, 0.2, 0.2, and 0.2 for end-to-end latency, monetary cost, WAN traffic and reconfiguration overhead, respectively.



The set of metric weights as well as the initial placement (Cloud or LB) dictate the performance of the RL algorithms.

Conclusions

- By considering only response time as a QoS metric, it is not possible to guarantee WAN traffic, monetary cost, and reconfiguration overhead.
- Multiple QoS metric evaluation provides a holistic view of the environment (infrastructure + application).
- Reinforcement learning allows the system to converging to good placements.

Future Work

- Investigation of machine learning mechanisms.
- Real tesbed.

Questions?

Hu, Wenlu et al. “Quantifying the Impact of Edge Computing on Mobile Applications”. In: *7th ACM SIGOPS Asia-Pacific Wksp on Systems*. APSys '16. Hong Kong, Hong Kong: ACM, 2016, 5:1–5:8. ISBN: 978-1-4503-4265-0.

Sutton, Richard S. and Andrew G. Barto. *Reinforcement Learning: An introduction*. MIT press, 2018.

Taneja, M. and A. Davy. “Resource aware placement of IoT application modules in Fog-Cloud Computing Paradigm”. In: *IFIP/IEEE Symp. on Integrated Net. and Service Management (IM)*. May 2017, pp. 1222–1228.

Yoon, K.P. et al. *Multiple Attribute Decision Making: An Introduction*. Multiple Attribute Decision Making: An Introduction. SAGE Publications, 1995.