Quantum Optimization of Worldwide LHC Computing Grid data placement

Mircea-Marian Popa[1], F. Carminati[2], S. Vallercosa[2], F. Fracas[2], C. Grigoraş[2], L. Betev
M. Carabaş[1], G. Pantelimon Popescu[1]

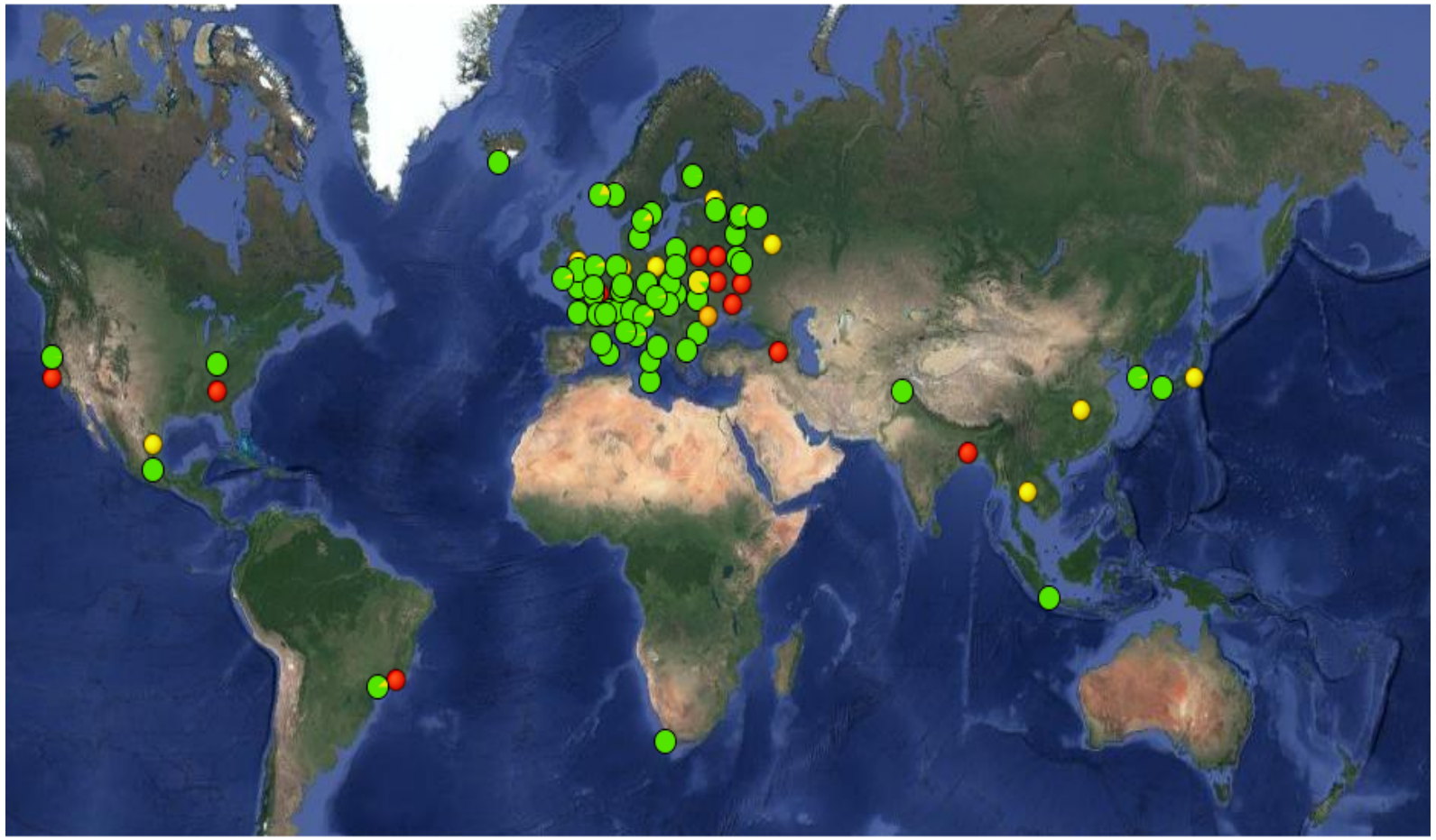
[1]University Politehnica of Bucharest, [2]CERN

22/01/2020

The Computing Grid

Connecting computer farms across the Globe

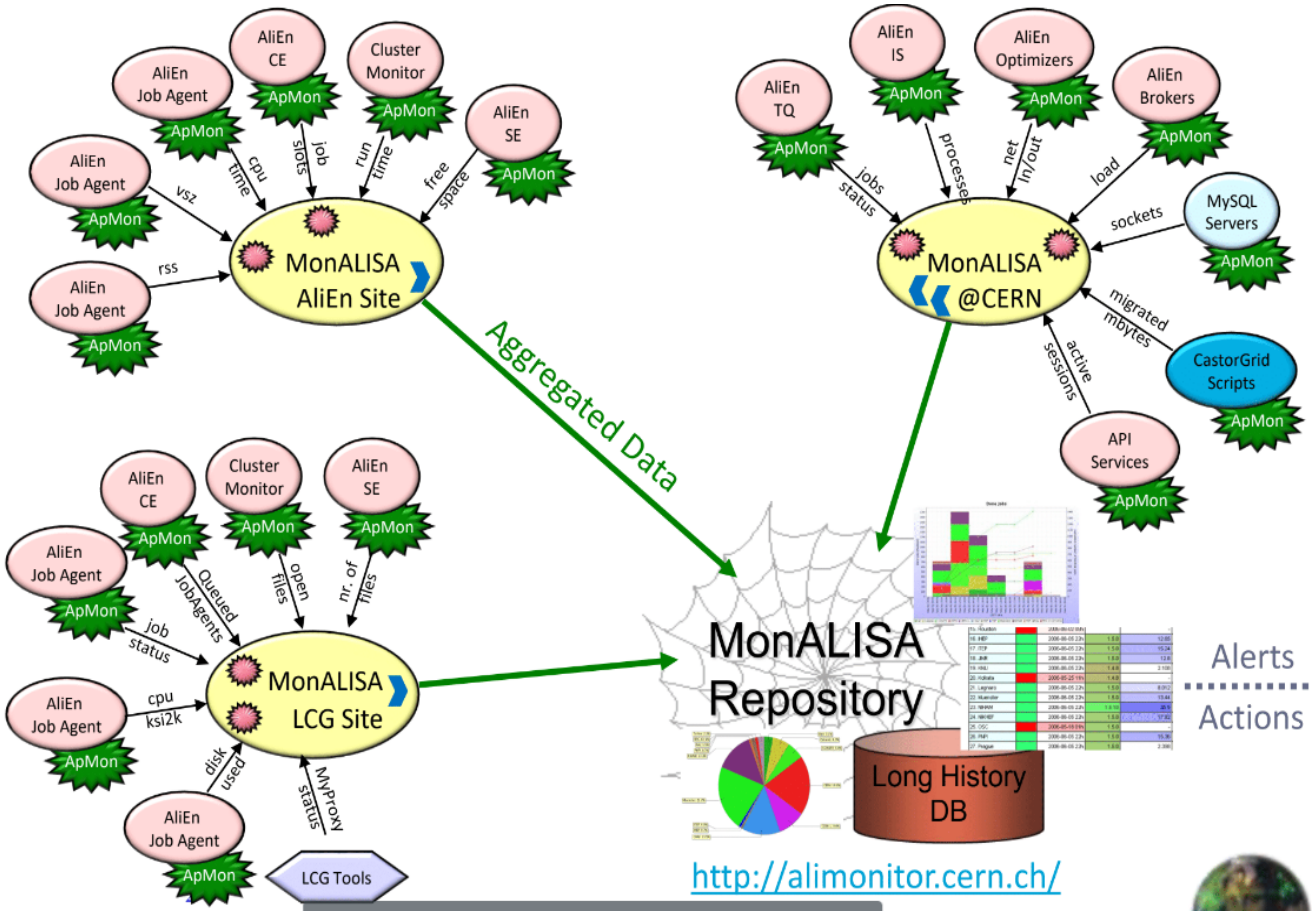
- **Clients** launch **jobs**
- **Jobs** run inside a **computing farm** and they **access files**
- **Files** may be read from a remote **storage element** (**SE**) or they may be available locally
- A file may have more than one **copy** spread across **different storage elements**



MonALISA

Coordinating and analyzing file transfer

- Answers **2 types of questions (queries)** from clients:
  - which is the optimal SE to **READ** a input file from ?
  - where to **WRITE** output data ?
- Gathers data on **network topology** and **SE availability**
- A set of **central nodes** answer queries using a **heuristic**
- Currently focus on **READ queries**
  - **READ/WRITE** ratio is **15/1** queries



Project Workflow

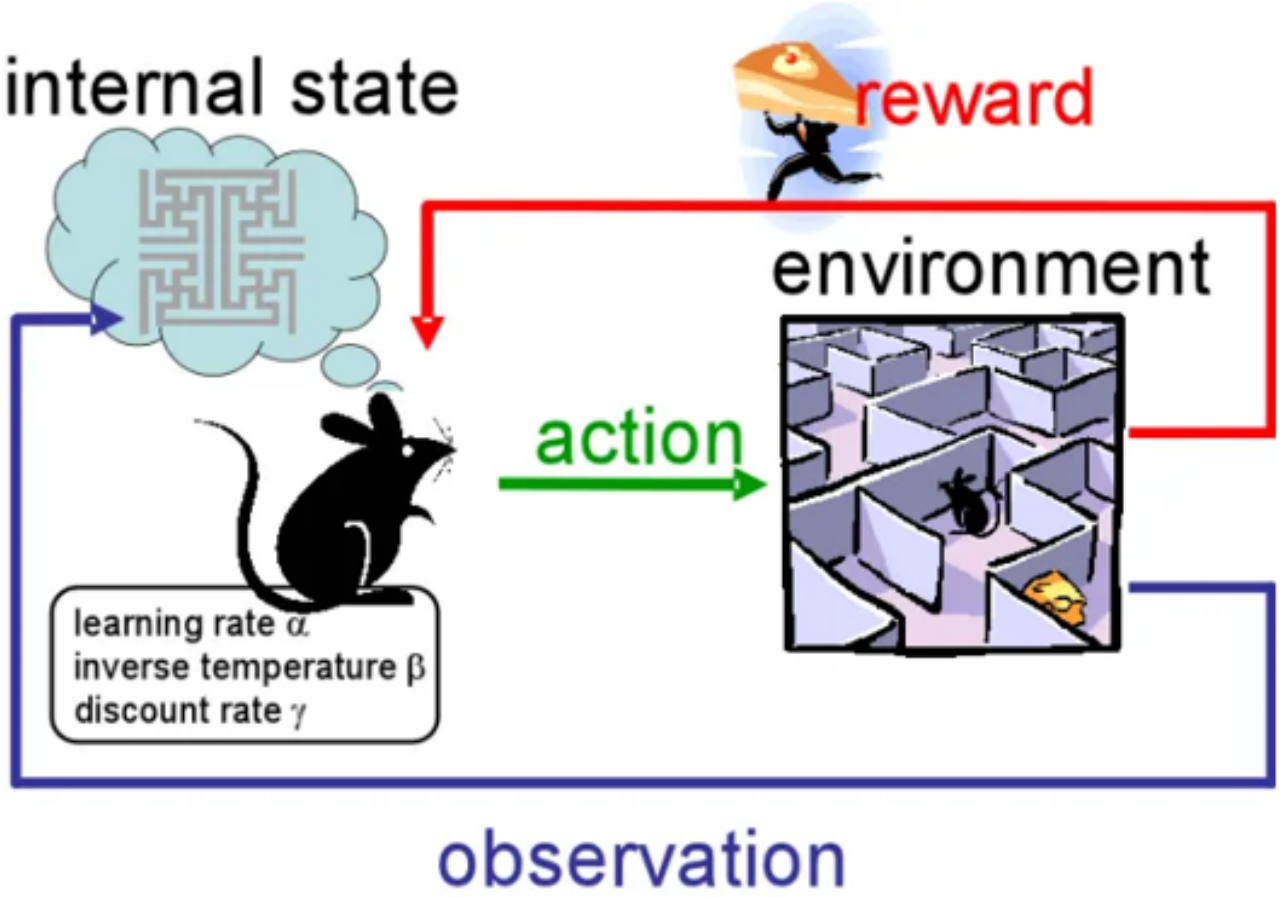
From Idea to Implementation

- **Goal**: use a **hybrid classical-quantum reinforcement learning algorithms** to optimize global data access **efficiency**
  - replace **MonALISA** heuristic with an intelligent agent
- Steps:
  1. **Gather raw data** logged from the live system
  2. **Preprocess** raw data into a format suitable for machine learning
  3. **Train** classical deep/machine learning algorithms
  4. Develop **quantum circuits** to accelerate specific components

Reinforcement Learning (RL)

How software agents learn

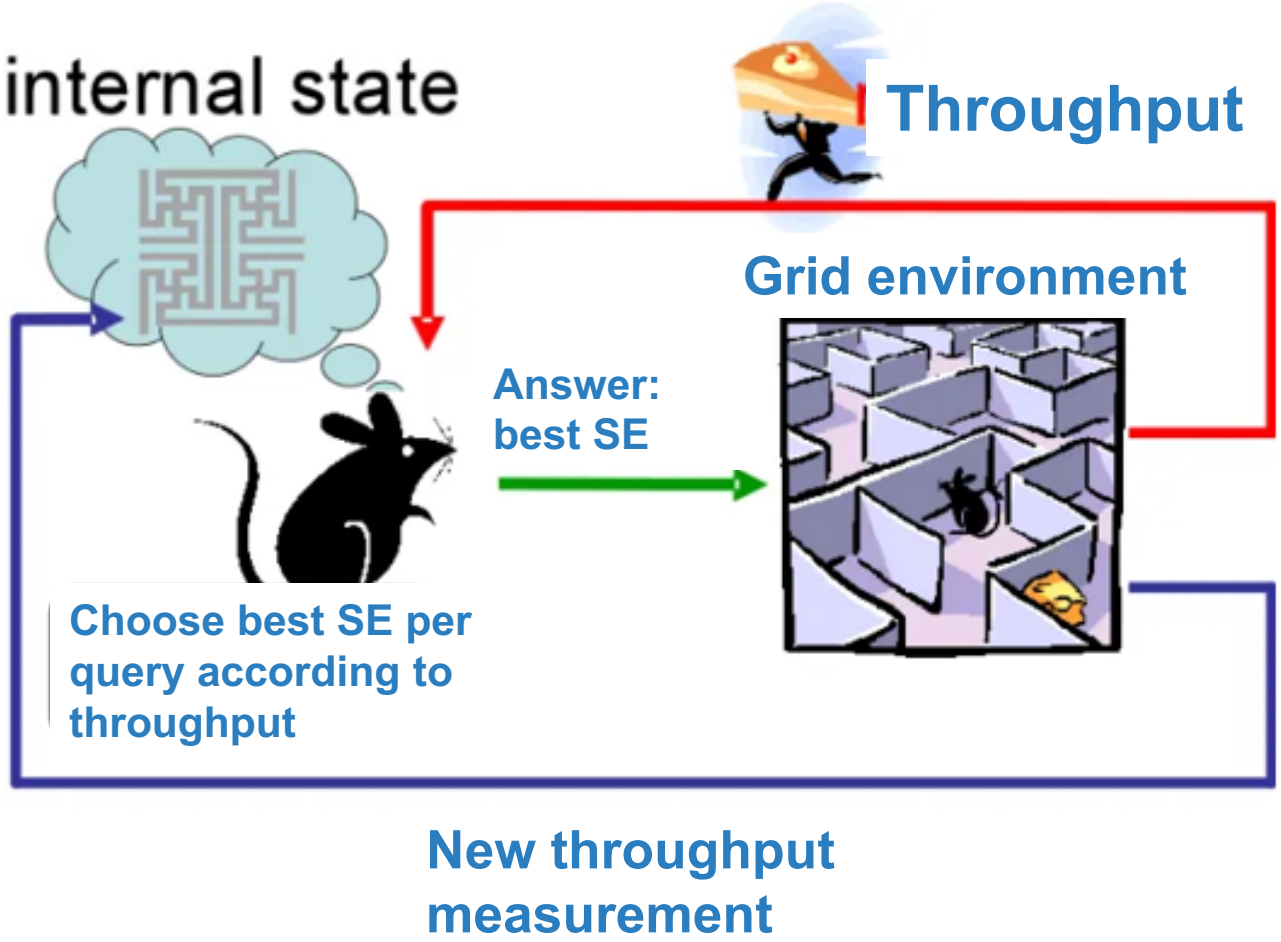
- An **agent** executes **actions** based on an input **state** in order to maximize a **reward** it will receive.
- **Agent-Environment interaction** is key.
- The Computing Grid is a **dynamic** environment.



MonALISA in a RL paradigm

Linking to RL elements

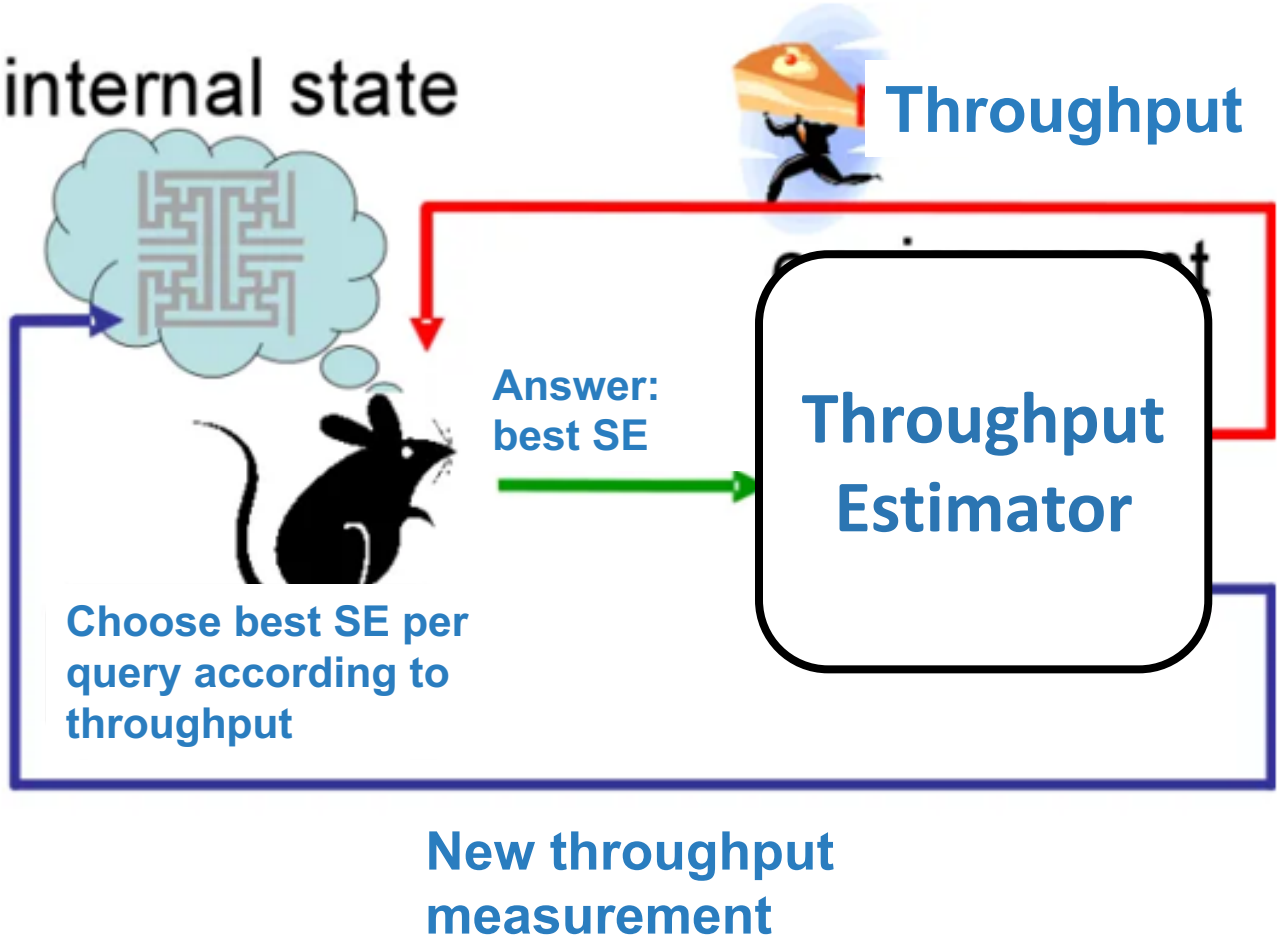
- **Agent**: a machine learning algorithm
- **State**:
  - a **Query**
    - a computing farm
    - a list of SEs presented in no particular order
    - a read size
- **Action**:
  - an **Answer**: a list of SEs optimally sorted
- **Reward**:
  - total **throughput** of the system
    - Simulate using a Throughput Estimator (TE) (network)



Throughput Estimator

Why do we need one ?

- MonALISA is a **live** system.
  - The **agent is** pre-trained **offline**.
- Need a **benchmark function** to asses the **quality** of the agent's **answers**.
- **Throughput** values are logged every 2 minutes.
  - It is possible to get throughput values more frequently ?
  - The **Agent** needs to modify its **policy** faster.
- TE Input: (**query, answer)**
- TE Output: **Throughput**



Building the training data set

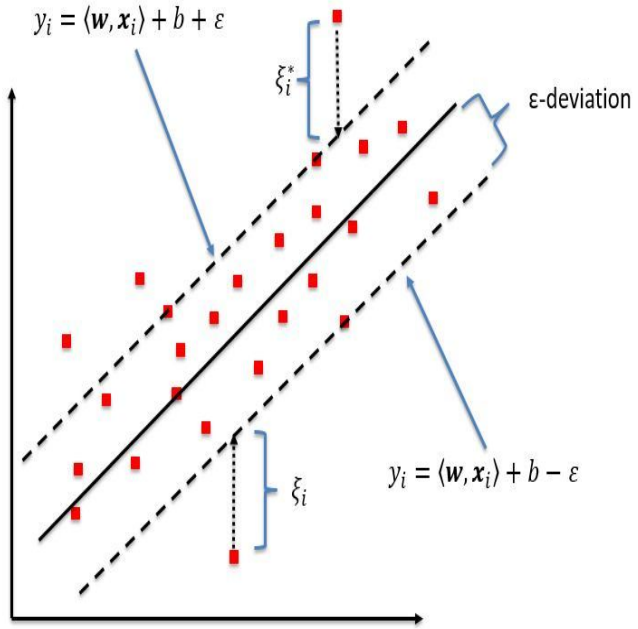
- MonALISA is a very **complex** system
  - ~120000 active jobs simulatneously
  - ~150 million queries per 24 hours
- Start by **reducing problem size**: focus on CERN(~ 40% of resources)
- Log **24h raw data** from the live system
- **Aggregate** information from different **sources/formats/time granularity**
- **Correlate queries, answers** and **throughput** values using the **time tags**

~500 ((**queries**, **answers**), **throughput**) values = **training dataset**

Throughput prediction

PRELIMINARY

- Tested algorithms so far:
  - **Support Vector Regressor** (SVR)
  - **Decision Forest Regressor**
  - **Multilayer Perceptron** - MLP
  - **Recurrent Neural Networks** (RNN)
- Metric **M**: mean absolute percentage error
- Best so far
  - **Support Vector Regressor**
    - inspired by Support Vector Machine
    - idea: include as many points as possible inside the margin
  - **Train: M**= 55%, **Validation: M**= 65%



https://www.researchgate.net/figure/Schematic-of-the-one-dimensional-support-vector-regression-SVR-model-Only-the-points_fig5_320916953

New data will be available in the near future → Deep Learning algorithms could be viable

Quantum Prospect

Reduces Theoretical Temporal Complexity

Use Quantum Computing to accelerate components in the RL approach

- Reward Estimator network currently uses a SVR
- Quantum Support Vector Machines are among the best studied cases today
  - key: vector inner product is done using a quantum circuit
  - can be adapted to a regression task
- In the future: once quantum hardware interfaces faster with classical, the **Agent** could become quantum as well

Summary

- Simulation and optimisation of a complex system such as MonALISA is a big challenge
- We are re-casting the problem in terms of Reinforcement Learning
  - Currently developing the Throughput Estimator
- Initial results on a small dataset are promising
  - Further optimisation is needed
  - Alternative models (time series) will be investigated
- A hybrid-quantum system is the final goal
  - We will accelerate the Throughput Estimator by implementing it as a quantum circuit

Thank you

QUESTIONS?

mircea-marian.popa@cern.ch

CONTACTS

FEDERICO CARMINATI
CERN openlab CIO
federico.carminati@cern.ch

LATCHEZAR BETEV
ALICE Offline
Latchezar.Betev@cern.ch

SOFIA VALLERCOSA
CERN openlab
Sofia.Vallercosa@cern.ch

COSTIN GRIGORAS
ALICE Offline
Costin.Grigoras@cern.ch

FABIO FRACAS
CERN openlab
fabio.fracas@cern.ch

MIHAI CARABAS
UPB Associate Professor
mihai.carabas@gmail.com

PANTELIMON GEORGE POPESCU
UPB Associate Professor
pgpopescu@gmail.com

www.cern.ch/openlab

Building The Data Set

- Size of the system:
  - ~120000 active jobs each moment in time
  - ~150 million queries per 24 hours across the entire Grid
- Reduce problem size: use only queries for which **CERN** is the **optimal choice**
- Log 24h raw data from the live system:
  - A **set of queries**:
    - count: ~ 4 million
  - A set of measured **throughput** values:
    - count: 627
- Use **MonALISA logs** to answer queries
  - **Simulate the heuristic based onnetwork topology** and **SE availability**
- Correlate (queries, answers, throughput values) using the time tags, one can build a training set for the Reward Generator.

Building The Data Set

Part 2

- Small data set at the moment:
  - training: 501
  - validation: 126
  - why ?
    - 627 throughput values sampled at 2 minutes interval => a span of 1254 minutes = ~20 hours
    - despite huge number of logged queries
- In a 2 minute interval:
  - maximum 96224 queries
  - minimum 267 queries

Building The Data Set

Part 3

- Problem: feeding one query answer at a time to the TE is not feasible
- Solution: **grouping total read size in a temporal bin** ; example given:
  - take a 60 minute period ending at the moment in time a throughput value is received
  - split the period into 1000 equally-sized temporal bins
  - 60 minutes / 1000 = 3.6 seconds bins
  - sum up the read size of all the queries that are perceived in a temporal bin
  - why ?
    - query answers favor CERN => more queries see CERN as their optimal option => read size per temporal bin is influenced