In-Network Decision Making Intelligence for Task Allocation in Edge Computing

Kostas Kolomvatsos, Christos Anagnostopoulos

School of Computing Science

University of Glasgow

30th International Conference on Tools with Artificial Intelligence

November 5-7, 2018

Volos, Greece

Outline

- Introduction
- Challenges
- Tasks Allocation
- Data Aware Mechanism
- Experimental Evaluation
- Conclusions and Future Work

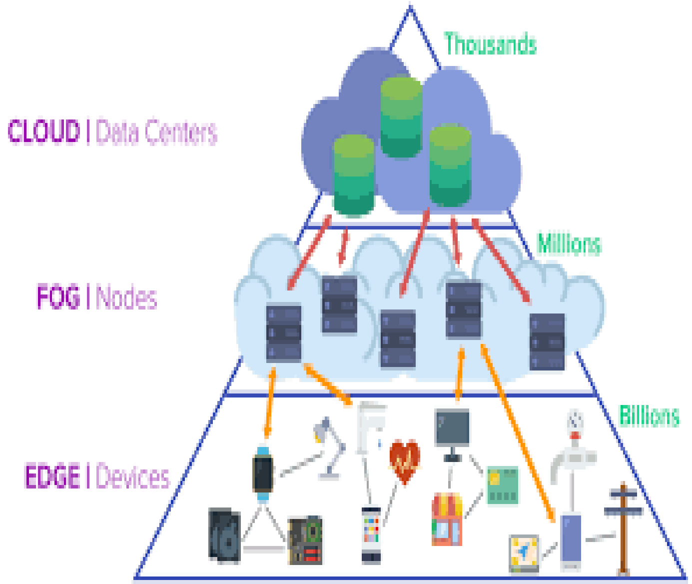
INTRODUCTION

- Internet of Things (IoT) offers a vast infrastructure of devices
- Intelligent analytics are offered on top of data collected by IoT nodes, i.e., sensing and computing devices
- Nodes can become knowledge producers through local processing

Introduction

- Legacy techniques involve data processing at the Cloud
- Cloud supports centralized processing
- Problem: Increased latency
- Need for support time sensitive applications

- Solution: Edge Computing
- It applies local processing at the edge nodes



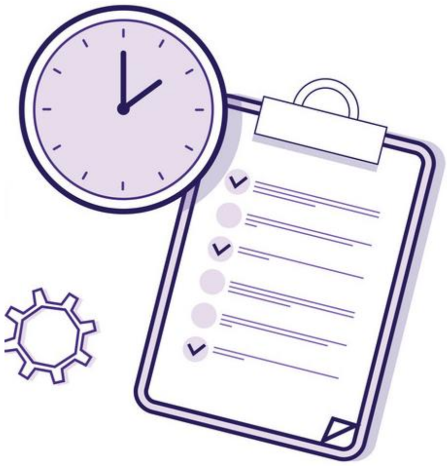
Challenges

- Keep analytics processing close to nodes
- We try to limit the latency in providing responses
- Avoid data migration (increases the communication overhead)

- To provide analytics, nodes should execute a set of tasks

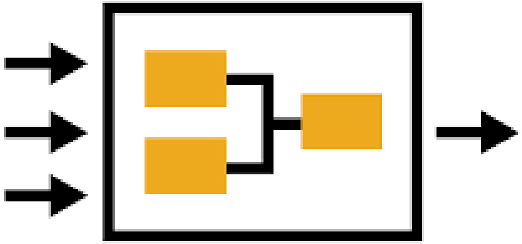
Tasks Allocation at the Edge

- Task management is used for *distributing tasks* among Edge Devices
- It should be done in an automated manner
- It is not necessary to explicitly define the capabilities or location of edge nodes
- Data are distributed as they are generated at different geographical places



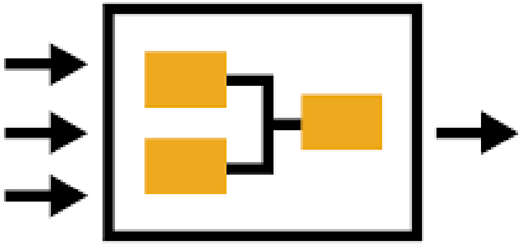
Autonomous Tasks Processing

- We focus on the behavior/status of each node (nodes' context)
- Nodes may act autonomously and decide about the allocation of tasks (*local execution or not*)
- Our technique takes into consideration:
  - Tasks characteristics
  - Nodes' characteristics
  - The data present in every node



Autonomous Tasks Processing

- Tasks may be delivered through streams
- They have specific characteristics, e.g *size, complexity, deadline, priority, software requirements*



- Nodes also exhibit specific characteristics, e.g., *load, throughput*
- Nodes 'own' a multidimensional dataset
- We should decide on the local execution of a task

Data Aware mechanism

- We can support an adaptive scheme to be *fully aligned with nodes' internal status, tasks requirements and the collected data*

- Target:
  - Develop a relevant decision mechanism
  - Decisions should be taken in a distributed, autonomous manner

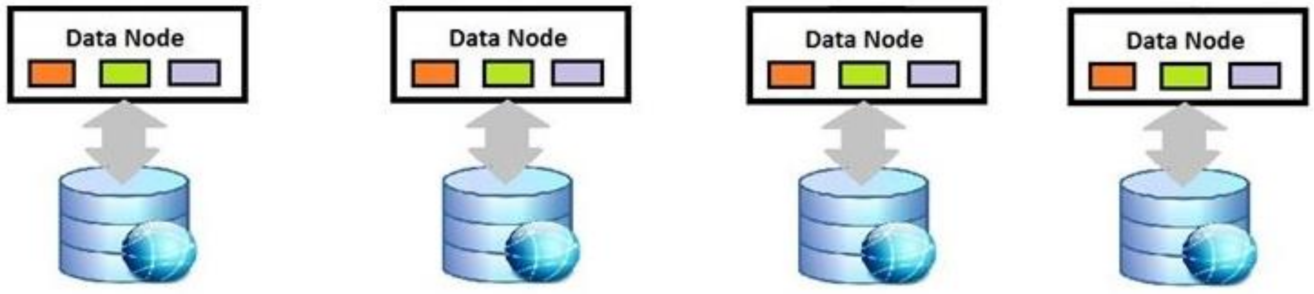
Data Aware mechanism

- Upon a task reception, nodes create the *context vector*
  - Nodes load
  - Tasks priority
  - Available resources

Data Aware mechanism

- The mechanism takes into consideration the data present at the nodes

- Nodes decide:
  - Local execution
  - Execution in the group
  - Execution at the Cloud



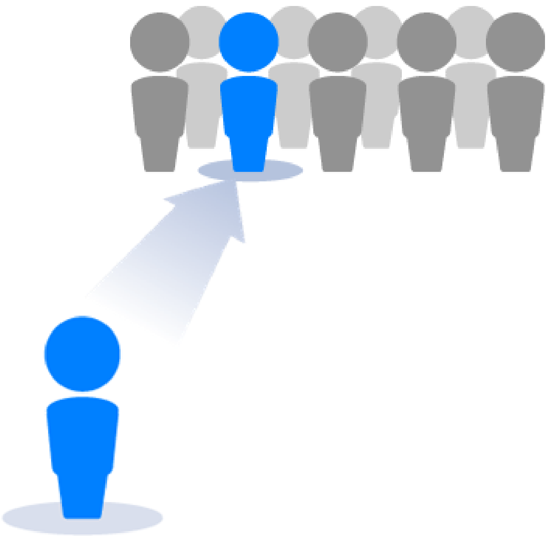
Data Aware mechanism

- Nodes exchange contextual information
- Such information will affect the decision making

- Every node calculates an *information vector* for every peer
  - Data statistical difference
  - The load
  - The communication cost

Data Aware mechanism

- If a task will not be executed locally, it will be sent to a peer with:
  - Similar data
  - Low load
  - Low communication cost



- If no peer is appropriate for executing the task, then send it to Cloud

Data Aware mechanism

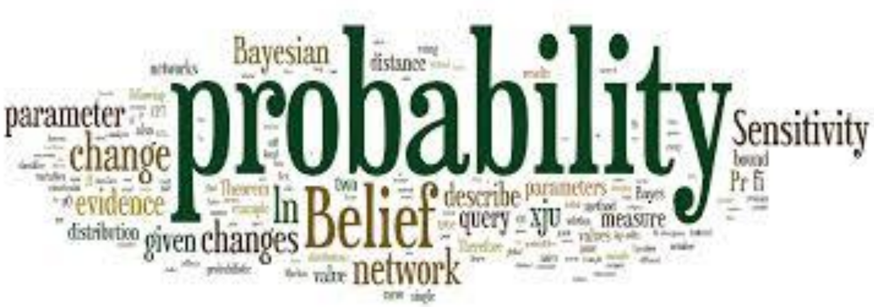
- The decision making:
  - Modeling
    - *the contextual vectors (for tasks)*
    - *the information vectors (for peers)*

  - Probabilistic local task allocation
  - Multi-criteria local task allocation

Data Aware mechanism

- Probabilistic approach
  - We can adopt Bayesian inference
  - Two classes: *Local execution (C1) or not (C2)*
  - We build on a training dataset for classification

  - Based on context vector for a task the classifier delivers the result



DATA AWARE MECHANISM

- Multi-criteria decision making
  - We build an ordered list of information vectors (data for peers)
  - We provide rankings for peers
  - Ratings are calculated based on the information vectors
  - The candidate with the highest score is selected to host the task

Experimental Evaluation

- We assess
  - The *correct selection of tasks* that will be locally executed (Aspect A)
  - The *correct identification of the appropriate peer* when tasks is offloaded (Aspect B)
  - The *'closeness' of the result* to the optimal solution (Aspect C)

- Metrics
  - For Aspects A & B: Precision (P), Recall (R), F-Measure (F)
  - For Aspect C: We 'create' the ideal node and its information vector

    [min_load, min_comm_cost, min_data_distance]
  - Closeness is represented by ω_i, i.e., the Euclidean distance with the ideal node

Experimental Evaluation

- Datasets
  - Real dataset related to companies bankruptcy*
  - Real dataset related to indoor environmental data**
- Training dataset
  - We create 300 context vectors and best actions
  - 65% of vectors indicate local processing
  - 35% of vectors indicate tasks offloading
- We construct networking topology of 5,000 nodes

* https://archive.ics.uci.edu/ml/datasets/qualitative bankruptcy
** http://db.csail.mit.edu/labdata/labdata.html

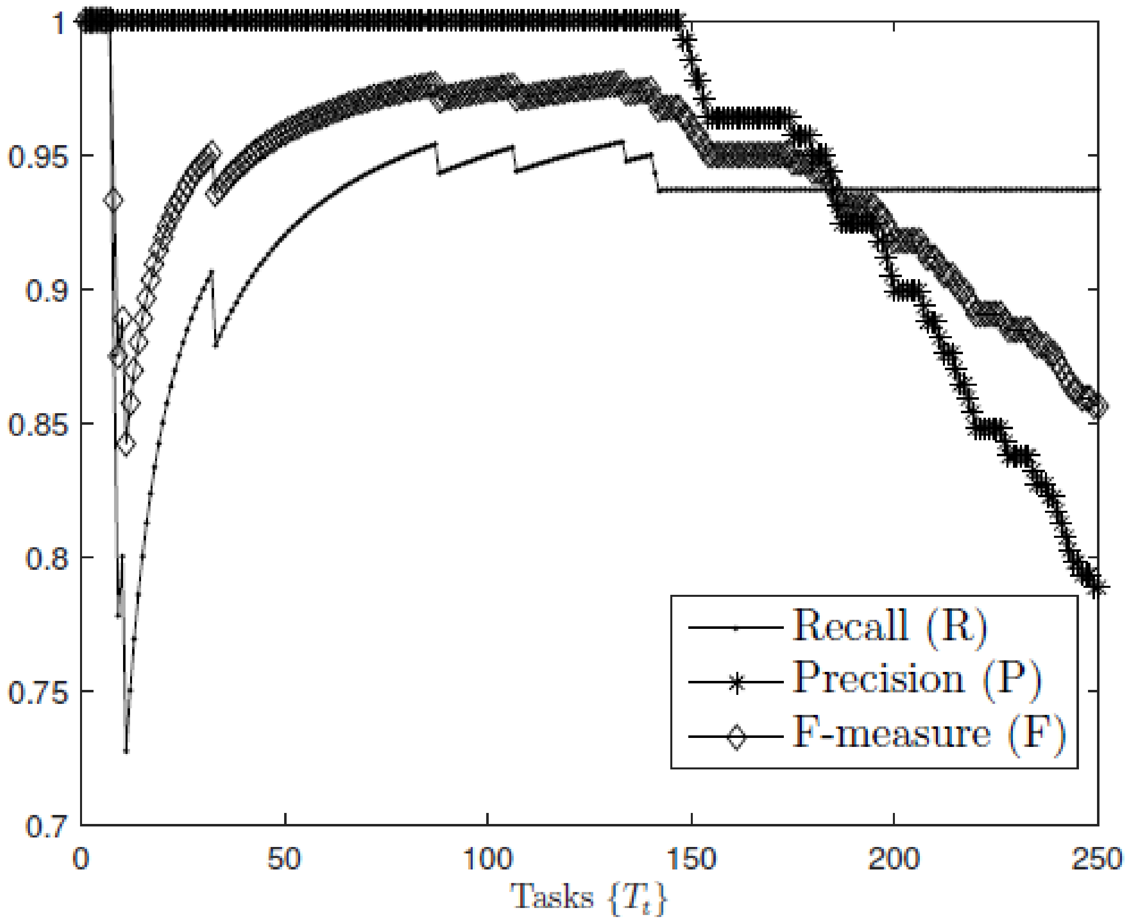
Experimental Evaluation

- In multi-criteria optimized tasks allocation, we focus on the following scenarios (different weights for each criterion)

Scenario	load (λ)	comm. (κ)	resources (ρ)	distance (δ)
Scenario A	0.25	0.25	0.25	0.25
Scenario B	0.70	0.10	0.10	0.10
Scenario C	0.10	0.10	0.40	0.40
Scenario D	0.10	0.10	0.10	0.70

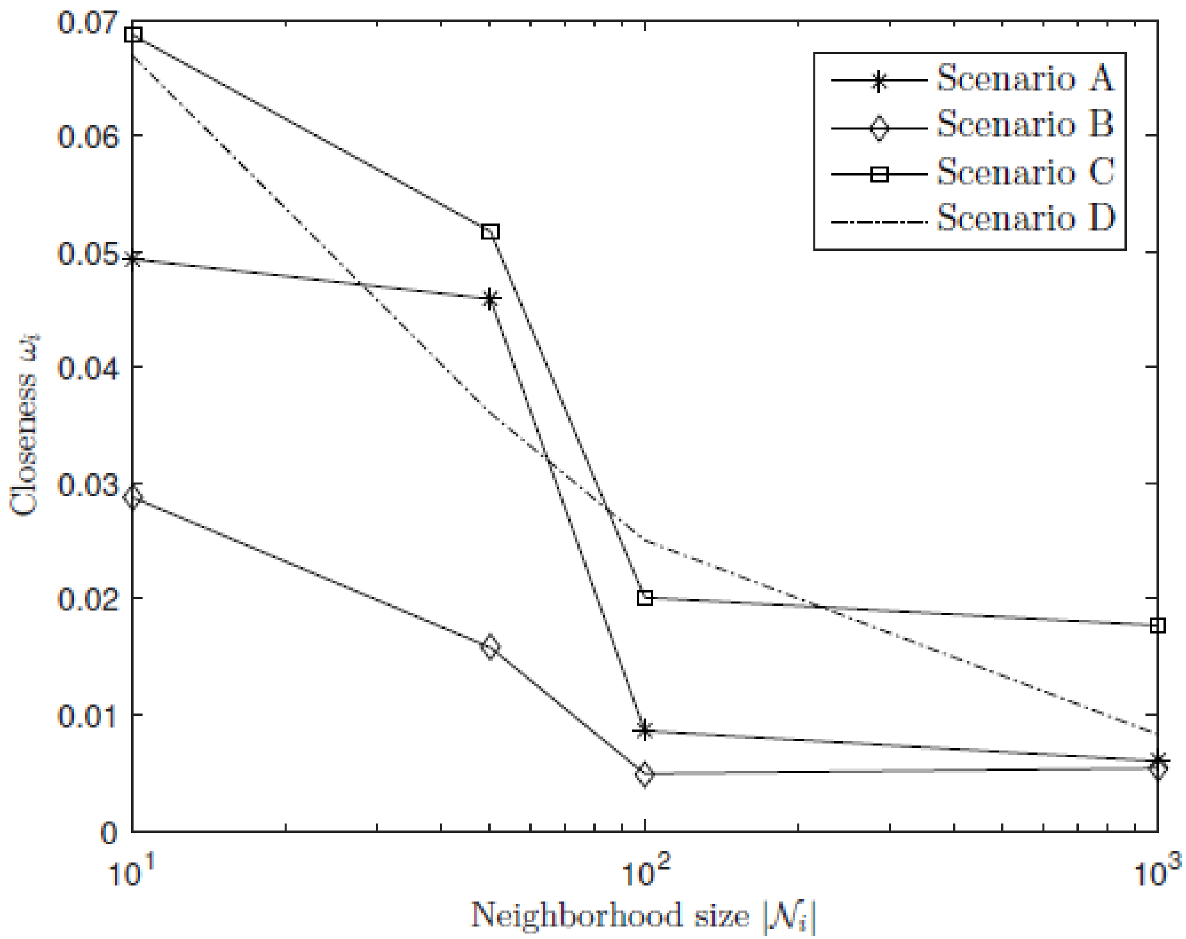
Experimental Evaluation

- Results for Precision, Recall and F-Measure



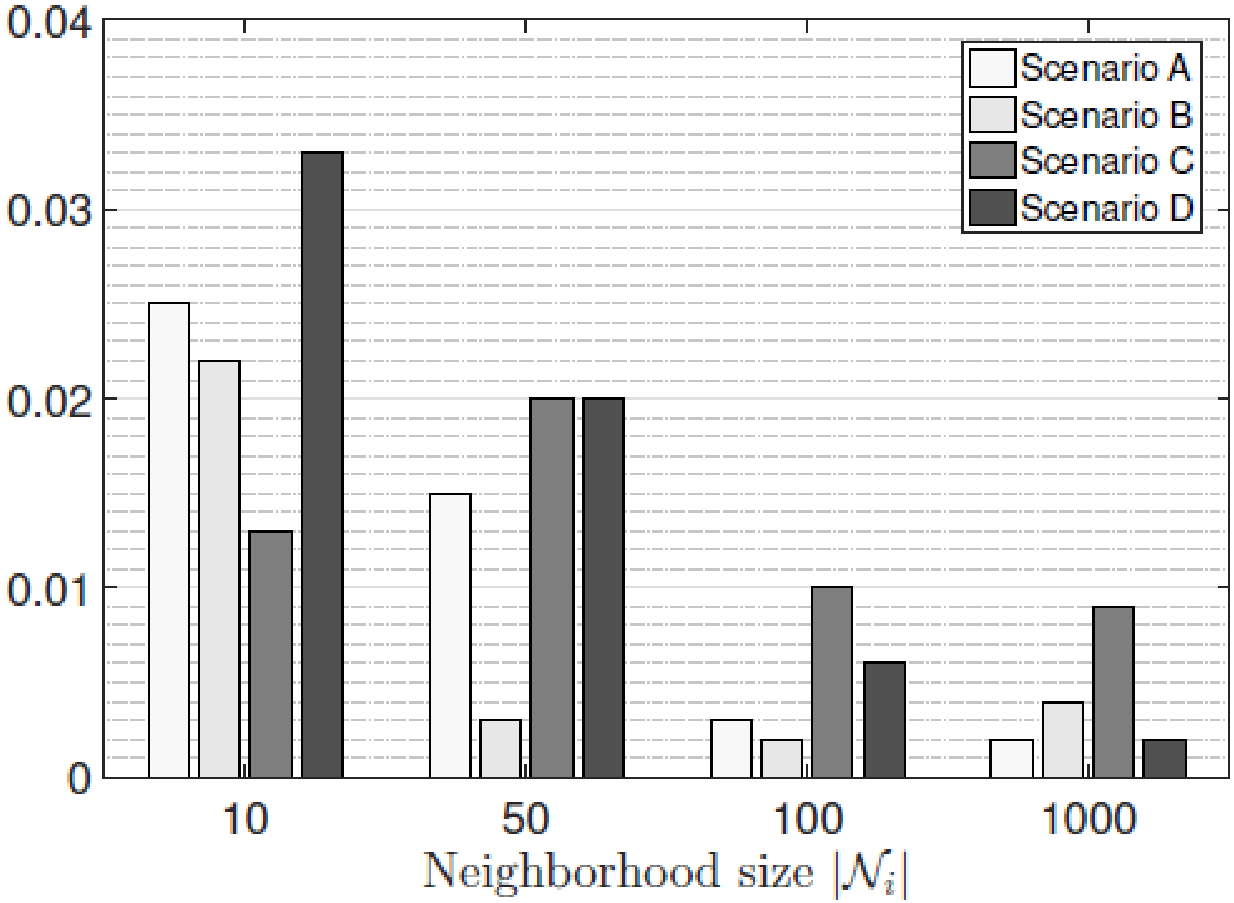
Experimental Evaluation

- Closeness with the ideal node



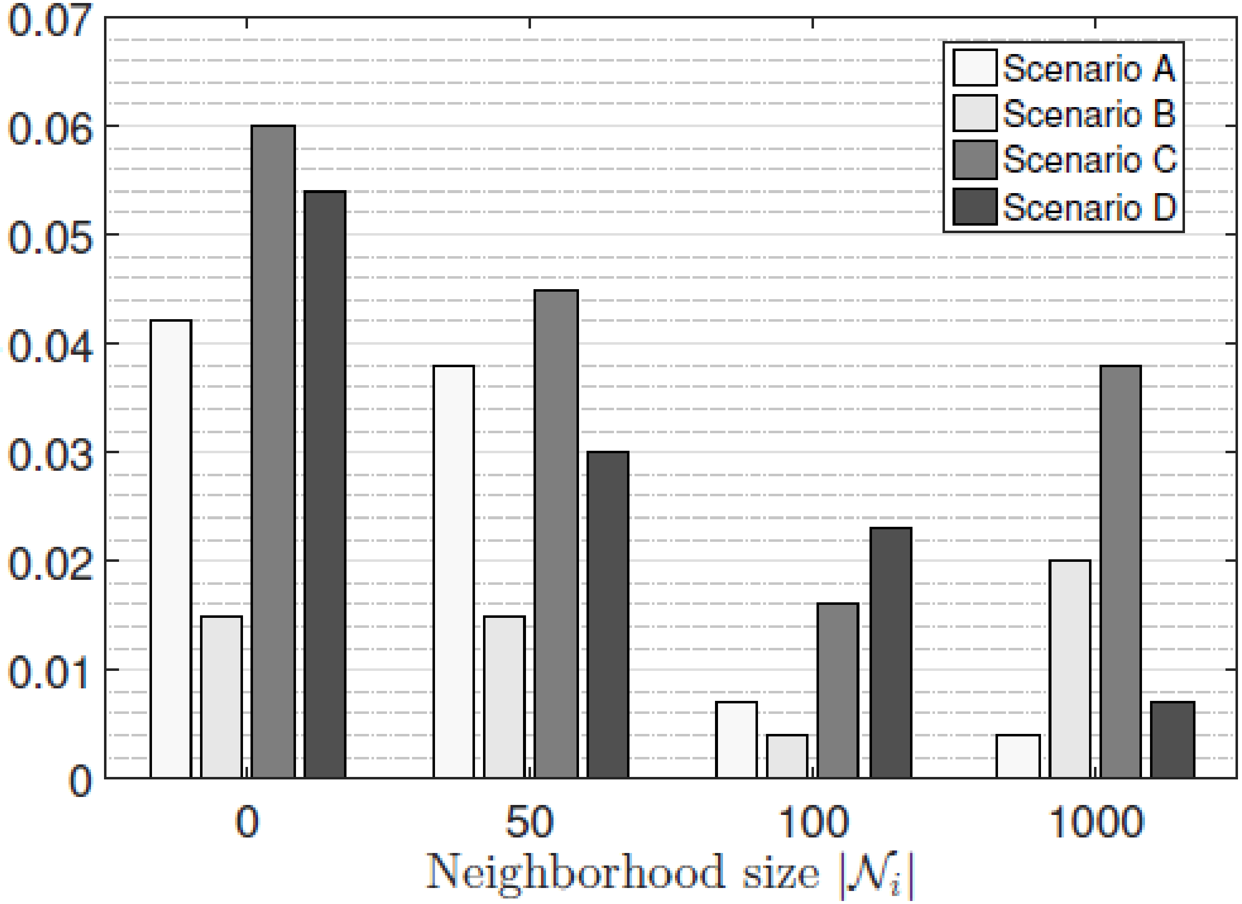
Experimental Evaluation

- Closeness for load



EXPERIMENTAL EVALUATION

- Closeness for data



Conclusions and Future Work

- Our sequential decision making manages to select the appropriate action for each task
- We manage to get efficient decisions related to the local processing
- We can select the best possible peer when tasks are offloaded

- Time-optimized decisions could increase the efficiency

Thank You!!

Questions?