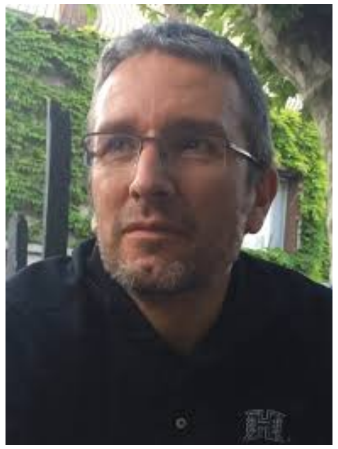
Where Fog/Edge global services should be deployed?

Farah AIT SALAHT
Ens Lyon, Inria, LIP, LYON, France
Equipe Avalon

(joint work with *Frédéric Desprez and Adrien Lebre*)



The Discovery Initiative Project

Agenda

I Part I. Overview of service placement problem in Fog environment

- Context
- Service Placement Problem (SPP) in Fog Computing
- Classification of reviewed works

II Part II. Service Placement Problem using Constraint programming and Choco solver

- System model and problem formulation
- Evaluation

III Conclusion

Agenda

Part I. Overview of service placement problem in Fog environment

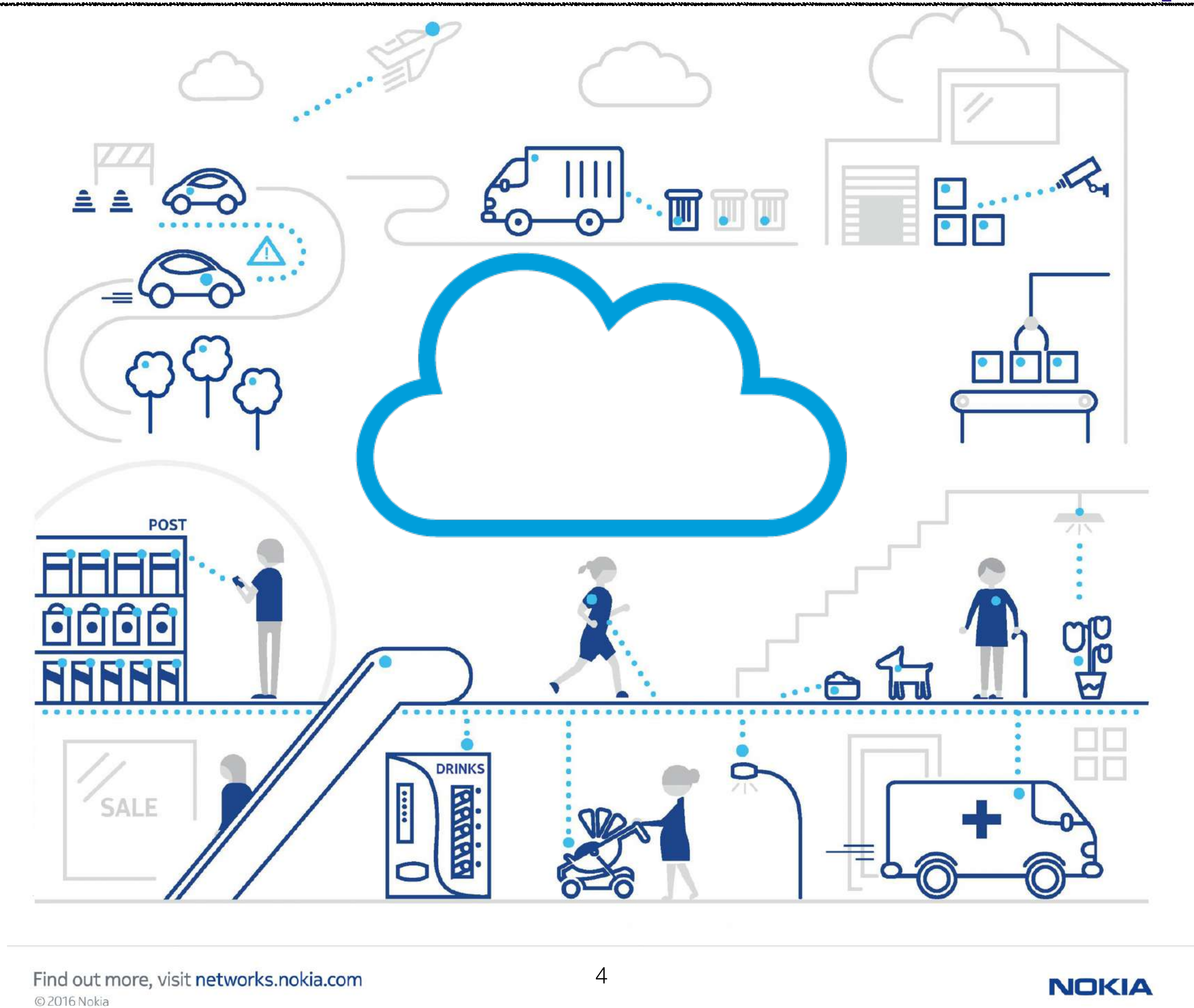
- Context
- Service Placement Problem (SPP) in Fog Computing
- Classification of reviewed works

Part II. Service Placement Problem using Constraint programming and Choco solver

- System model and problem formulation
- Evaluation

Conclusion & current + future work

Context: The lastest wave of Cloud & IoT adoption



The New Reality - Dynamic, Data Driven!

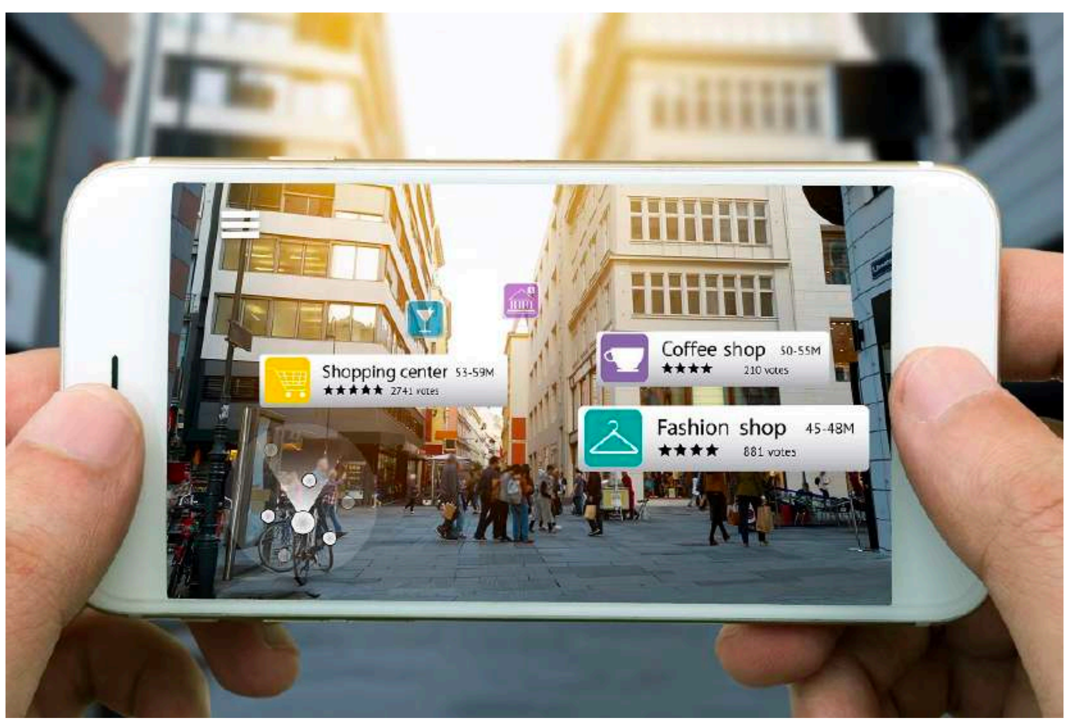
Traditional Cloud Computing architectures do not meet all of IoT requirements.

The New Reality - Dynamic, Data Driven!

New types of mobile applications

➡ Interactive applications require ultra-low network latencies e.g., augmented reality require end-to-end delays under 20 ms

➡ But latencies to the closest data center are 20-30 ms using wired networks, up to 50-150 ms on 4G mobile networks!!!



➡ Throughput-oriented applications require local computations E.g., distributed video-surveillance is relevant only close to the cameras

In a new distributed environment

➡ Reduce latency, network traffic, power consumption and increase scalability and availability

Exploit distributed and near-edge computation

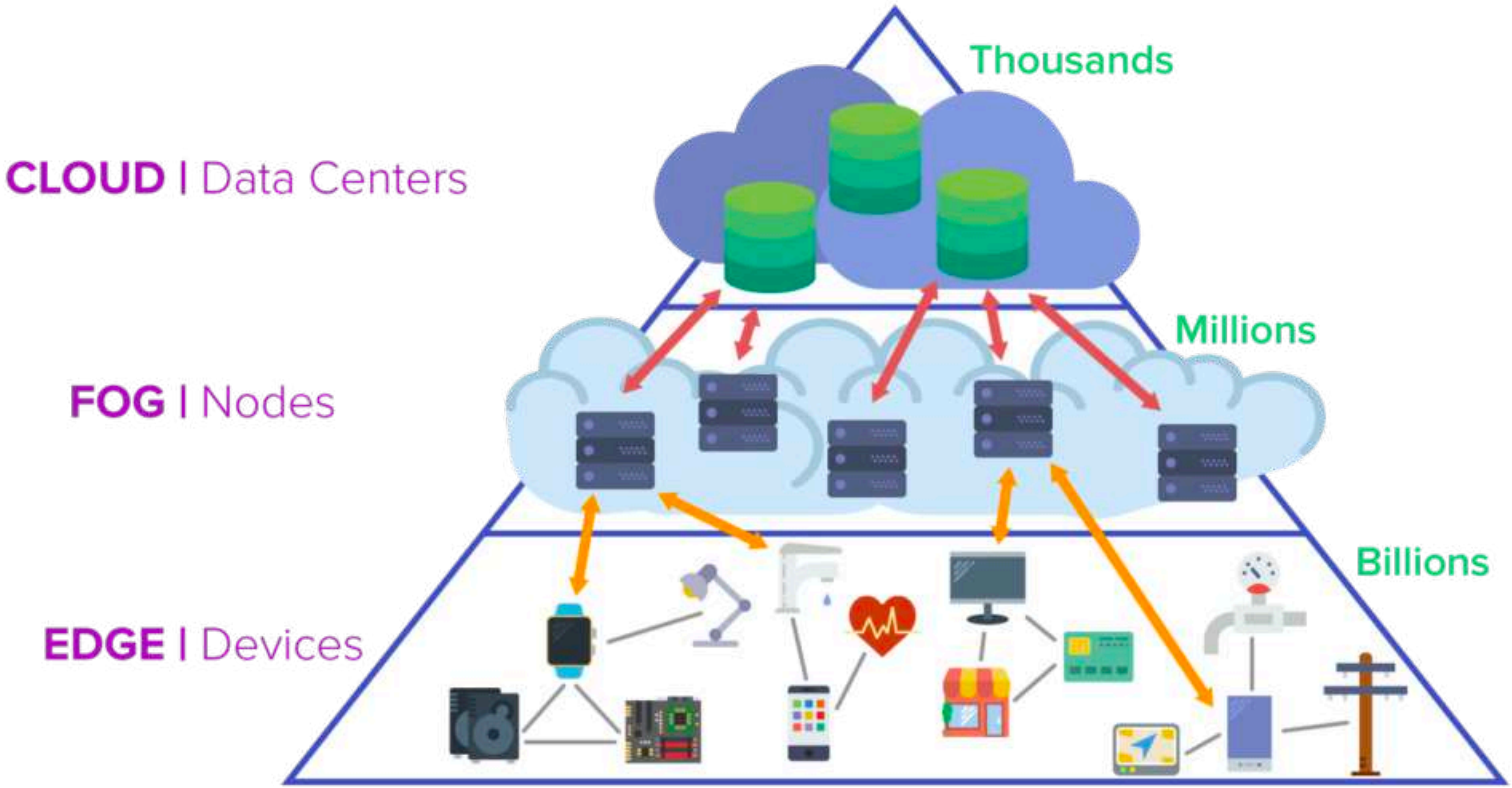
Fog Computing

("the cloud close to the ground")

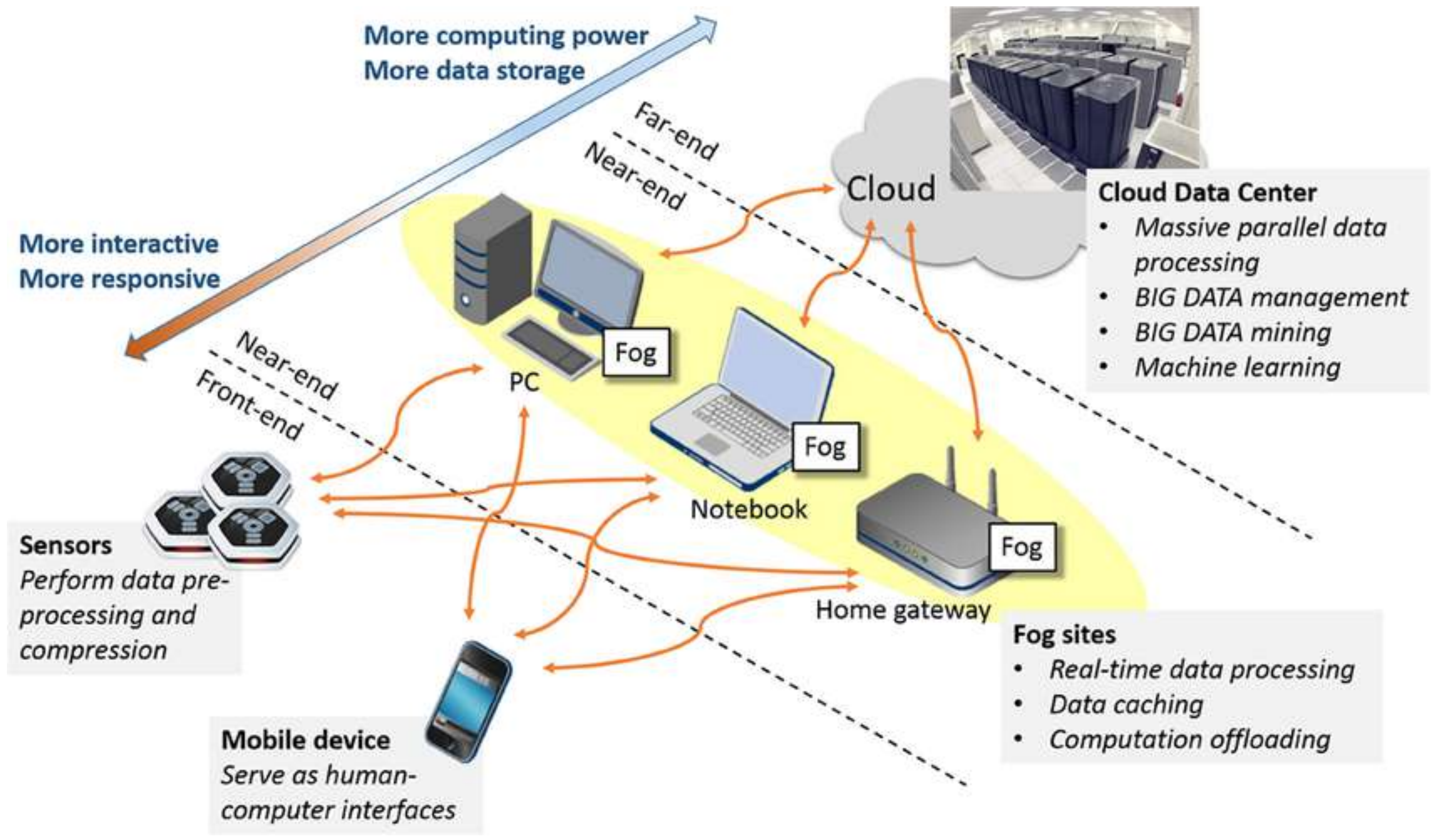
Analyze most IoT data near the devices that produce and act on that data

Fog Computing

A paradigm that extends Cloud computing and services to the edge of the network. Similar to Cloud, Fog provides data, compute, storage, and application services to end-users.



Fog Computing



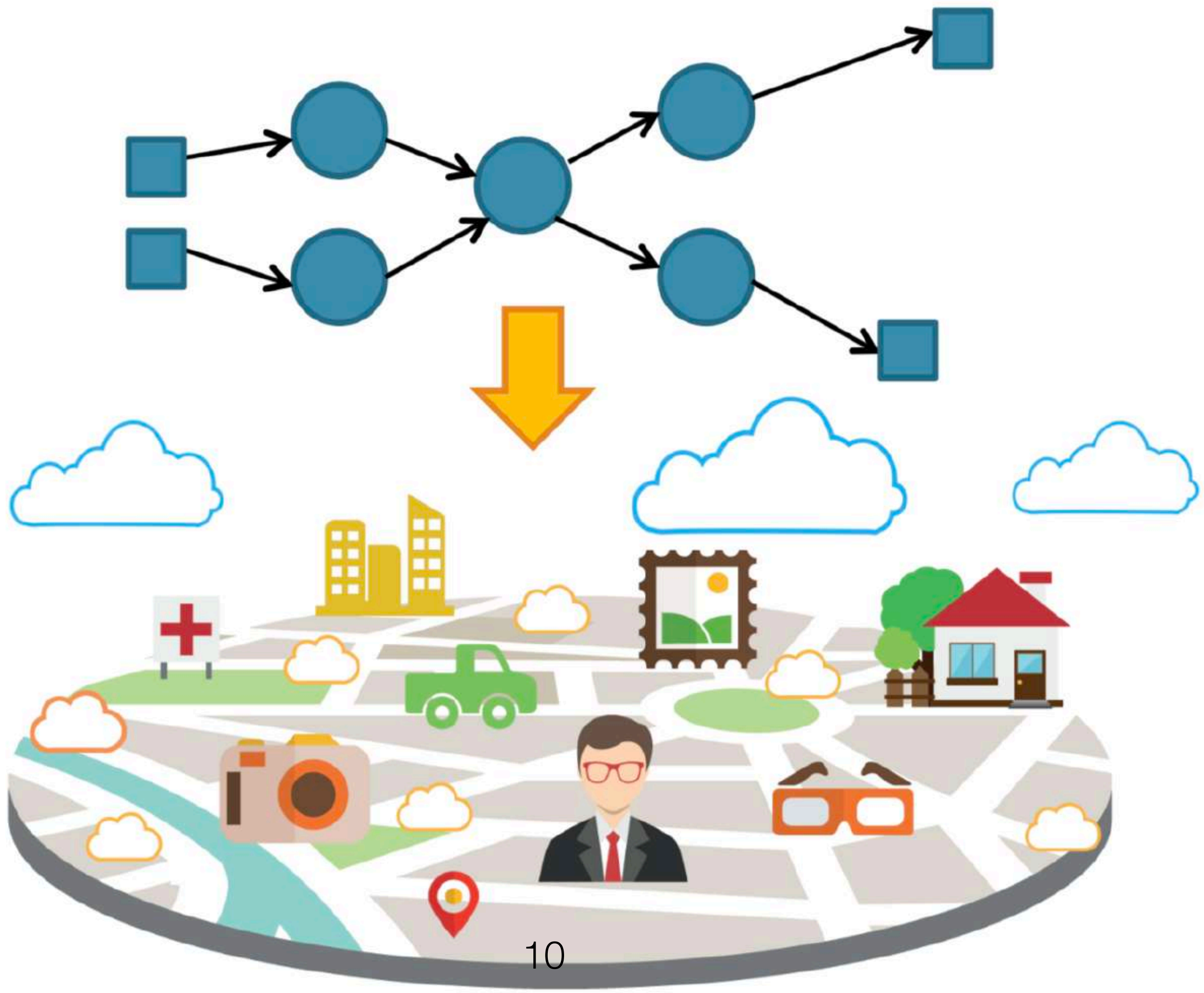
Conceptual architecture of Fog /Cloud infrastructure

Fog computing poses old and new challenges

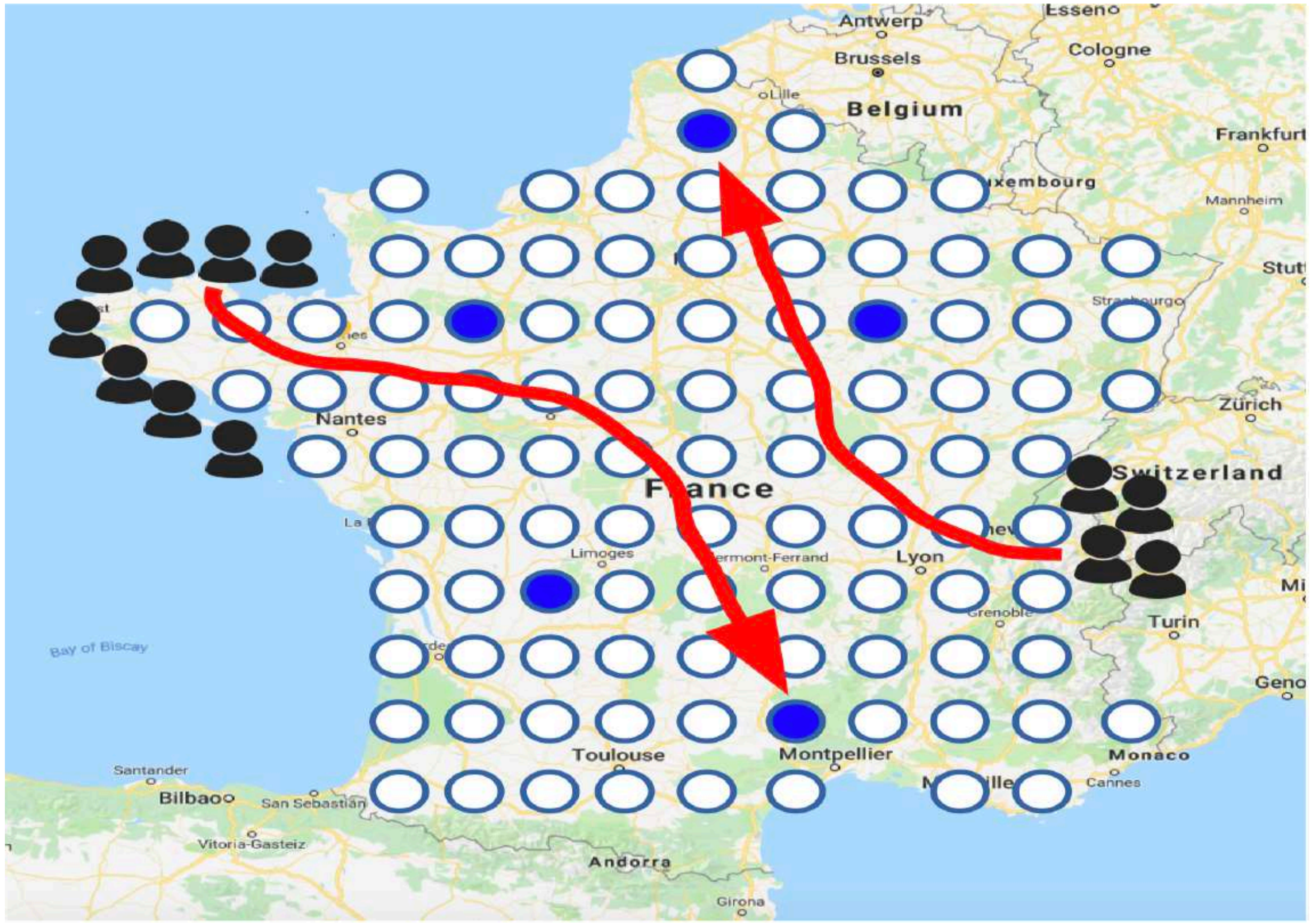
One of the main challenges in Fog computing is:

Service Placement Problem

How to assign the IoT applications to computing nodes (fog nodes) which are distributed in a Fog environment



Fog computing poses old and new challenges

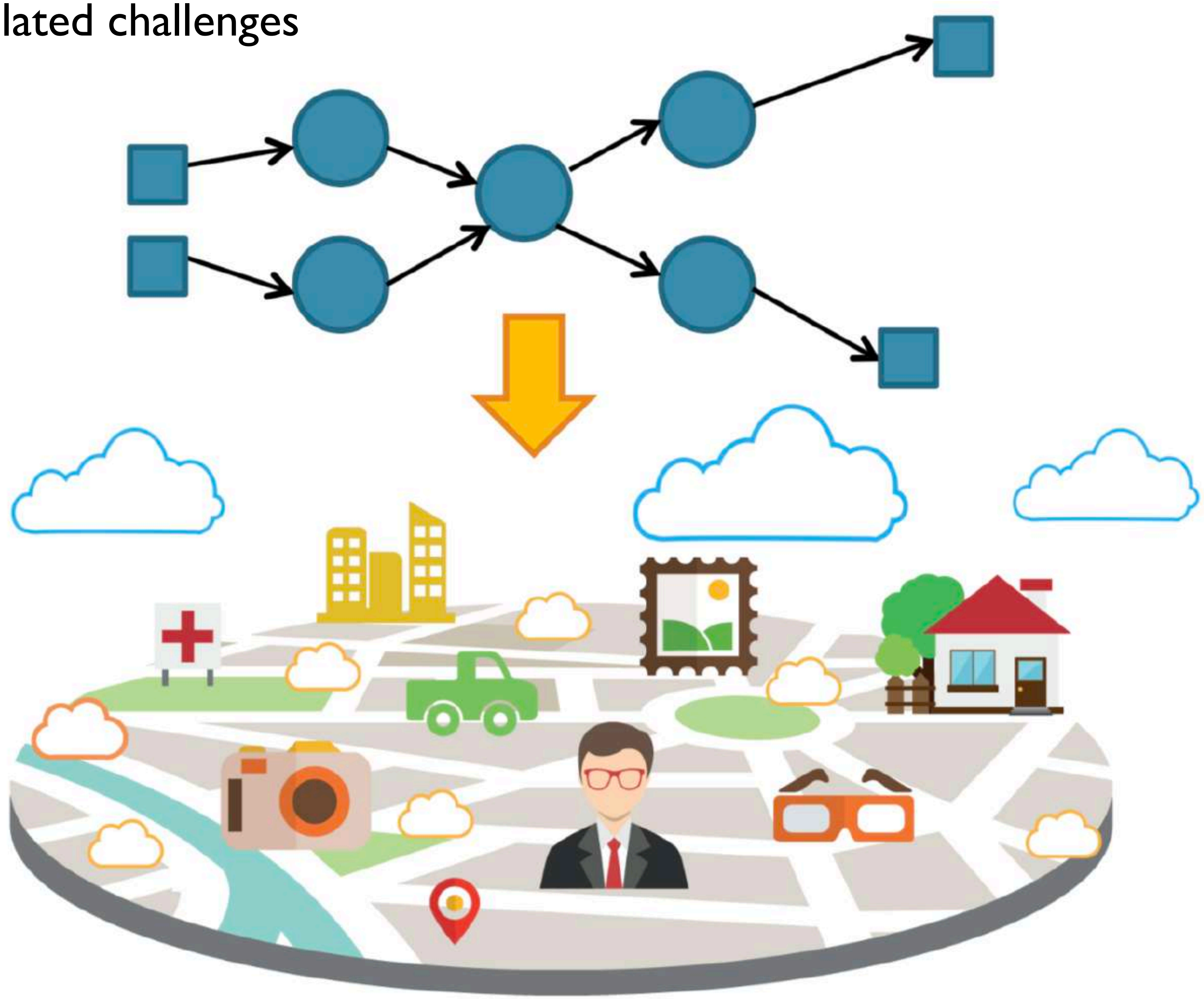


- Computing and networking resources are heterogeneous (e.g., business constraints, capacity limits, …)
- Nature of the system
- Computing and network resources are not always available
- Service cannot be processed everywhere
- …

First goal of our work

- Identify the different aspects considered in the literature regarding: problem statement, problem formulation, optimization strategies, and evaluation platforms

- Propose a classification of the surveyed works in order to identify more easily the placement-related challenges



Agenda

Part I. Overview of service placement problem in Fog environment

- ~~Context~~
- **Service Placement Problem (SPP) in Fog Computing**
- Classification of reviewed works

Part II. Service Placement Problem using Constraint programming and Choco solver

- System model and problem formulation
- Evaluation

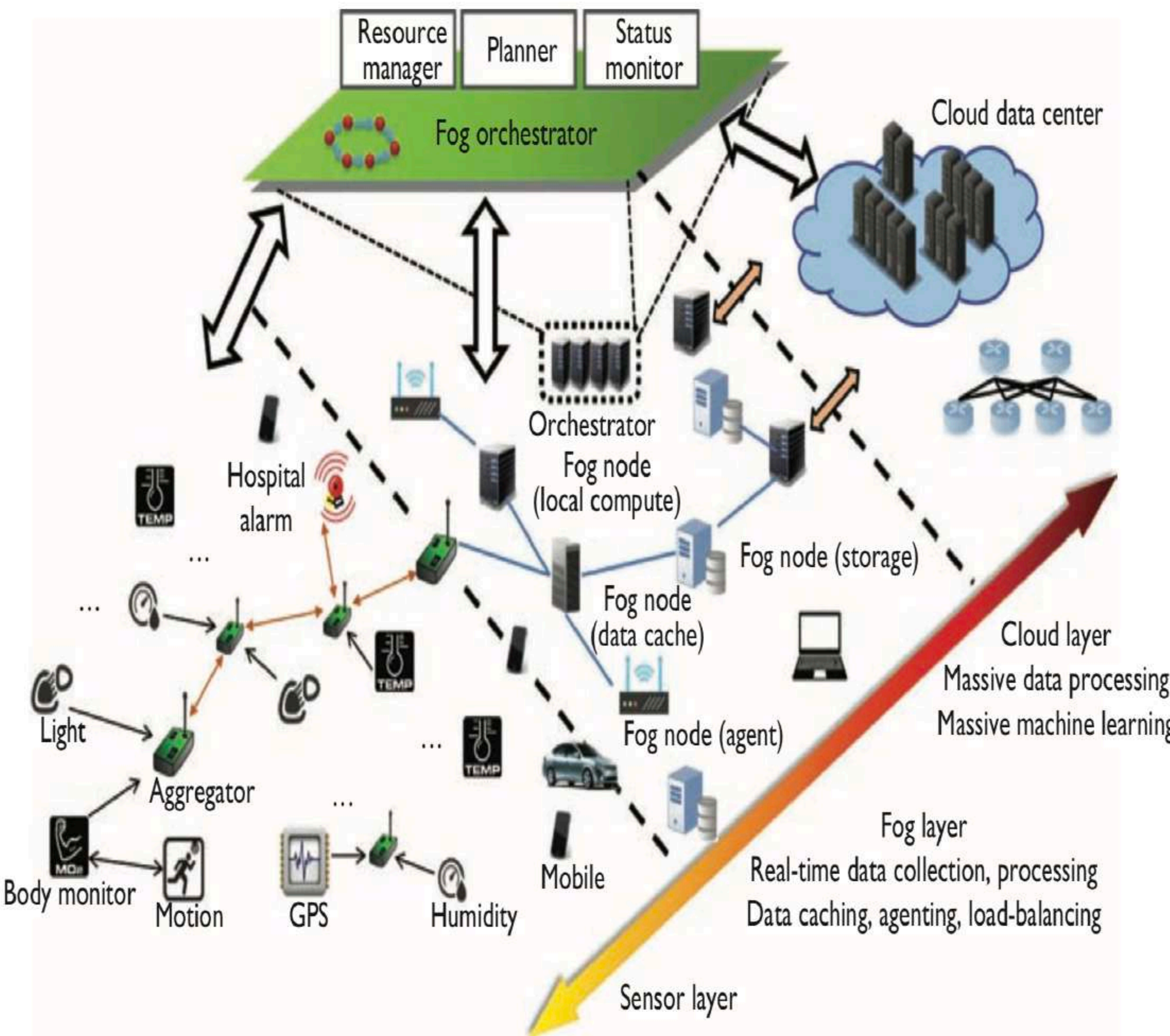
Conclusion & current + future work

Service Placement Problem in Fog Computing

- **To Address the SPP**
  - ☑ Problem statement
  - ☑ Optimization strategies
  - ☑ Evaluation environments

I.a) Infrastructure model

- **Computing and network resources**



➡ **Resources type**

- *Computing:* servers, PCs, etc
- *Networking:* gateways, routers, switches,
- base stations, etc
- *Storage:* every node that can provide storage.
- *Mobile:* vehicles, smartphones, etc
- *Endpoint abstraction:* sensor agent, actuators…

➡ **Characteristics**

- *Computing:* CPU power, RAM, etc
- *Networking:*
  - *Type:* wireless, wired
  - *Capabilities:* latency, bandwidth, error rate…
- *Storage:* disk, etc

I.b) Application model

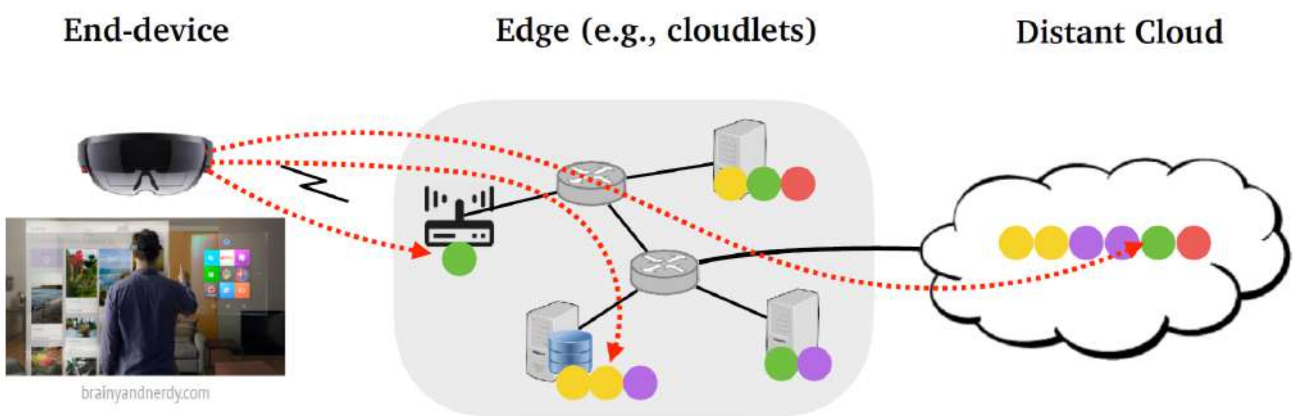
- **Application types**

    a) A data pipeline

    b) A monolithic service

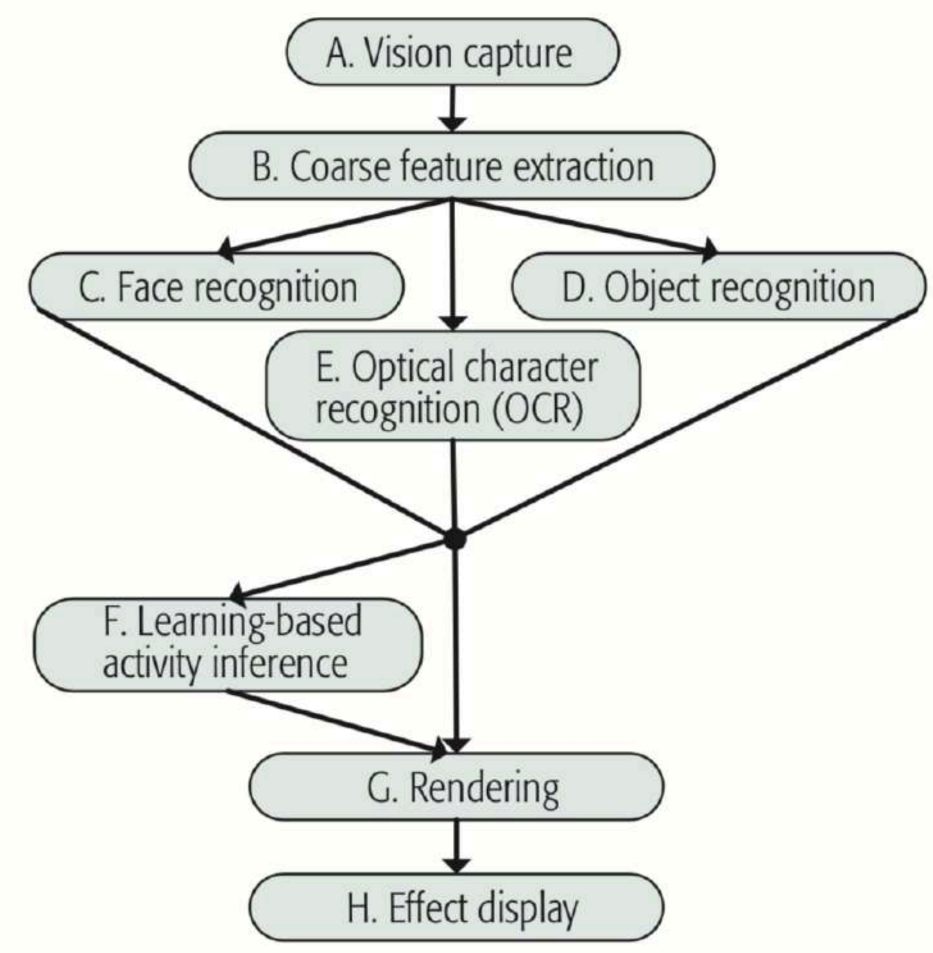
    c) A set of inter-dependent components

    d) A Directed Acyclic Graph (DAG)



- **Application requirements**

    - Computing: CPU power, number of cores, RAM, etc
    - Network-oriented:
        - Bandwidth: Per link, End-to-end
        - Latency: Per link, End-to-end
        - Error-rate: Per link, End-to-end
        - Jitter: Per link, End-to-end
    - Task-oriented: Deadline Location-oriented:
        - Location: app must run in Paris
        - Application-specific: this app can run only at some node



Example of a DAG application

-Cognitive assistance

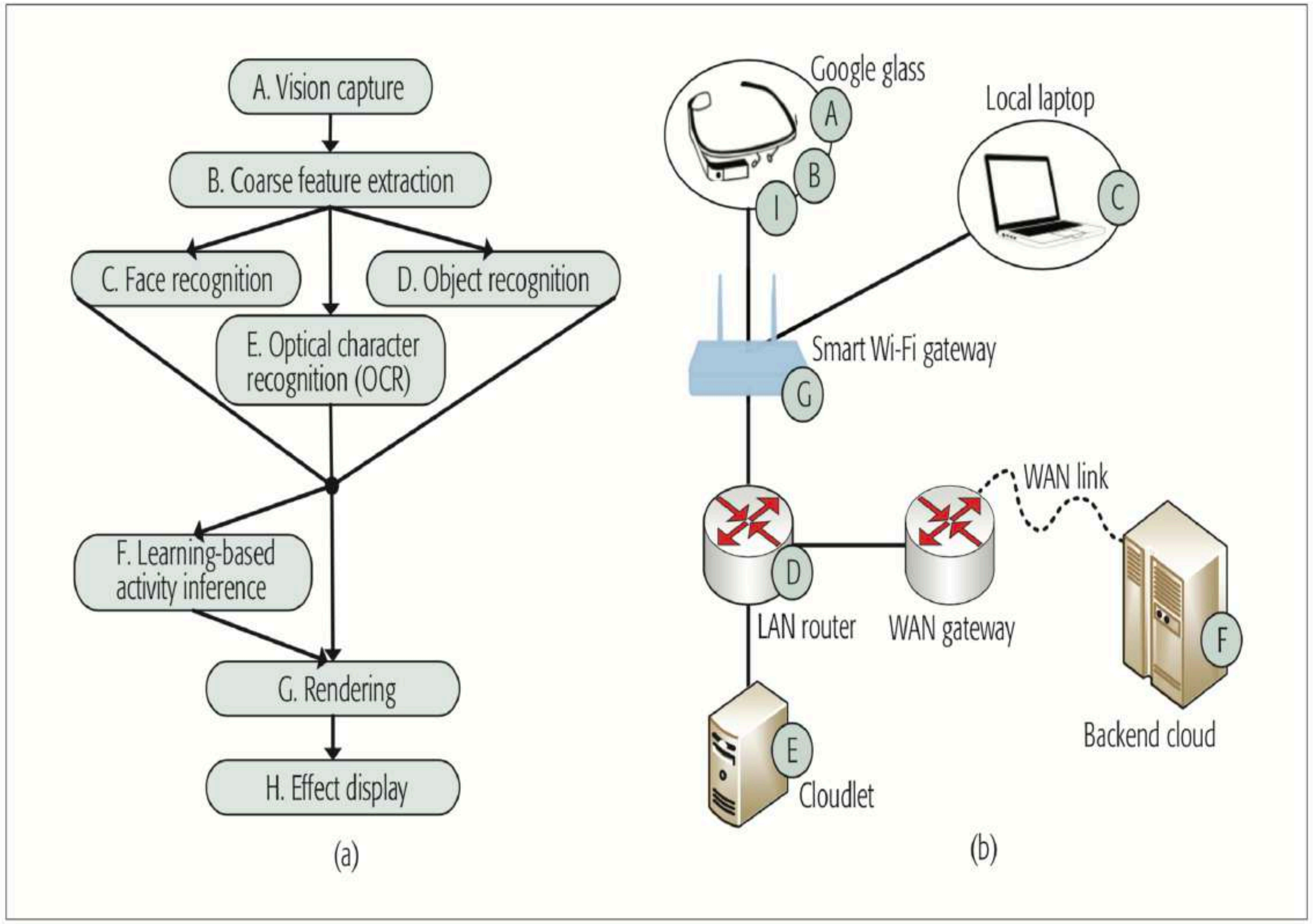
I.c) Placement pattern

▸ Defines a mapping pattern by which applications (components+communication among components) are mapped onto an infrastructure graph

Application placement involves finding the host nodes (resp. links) that satisfy the given restrictions

➡ **Example:**

- Capacity limit.
  - Physical Node limitations (CPU, RAM, Storage. . .), and
  - Physical link limitations (Bandwidth, delay. . .)
- Locality requirement
- Delay sensitivity



▸ Placement taxonomy

i) Control plan design: Centralized vs. Distributed

ii) Offline vs. Online placement

iii) Static vs. Dynamic placement

iv) Mobility support

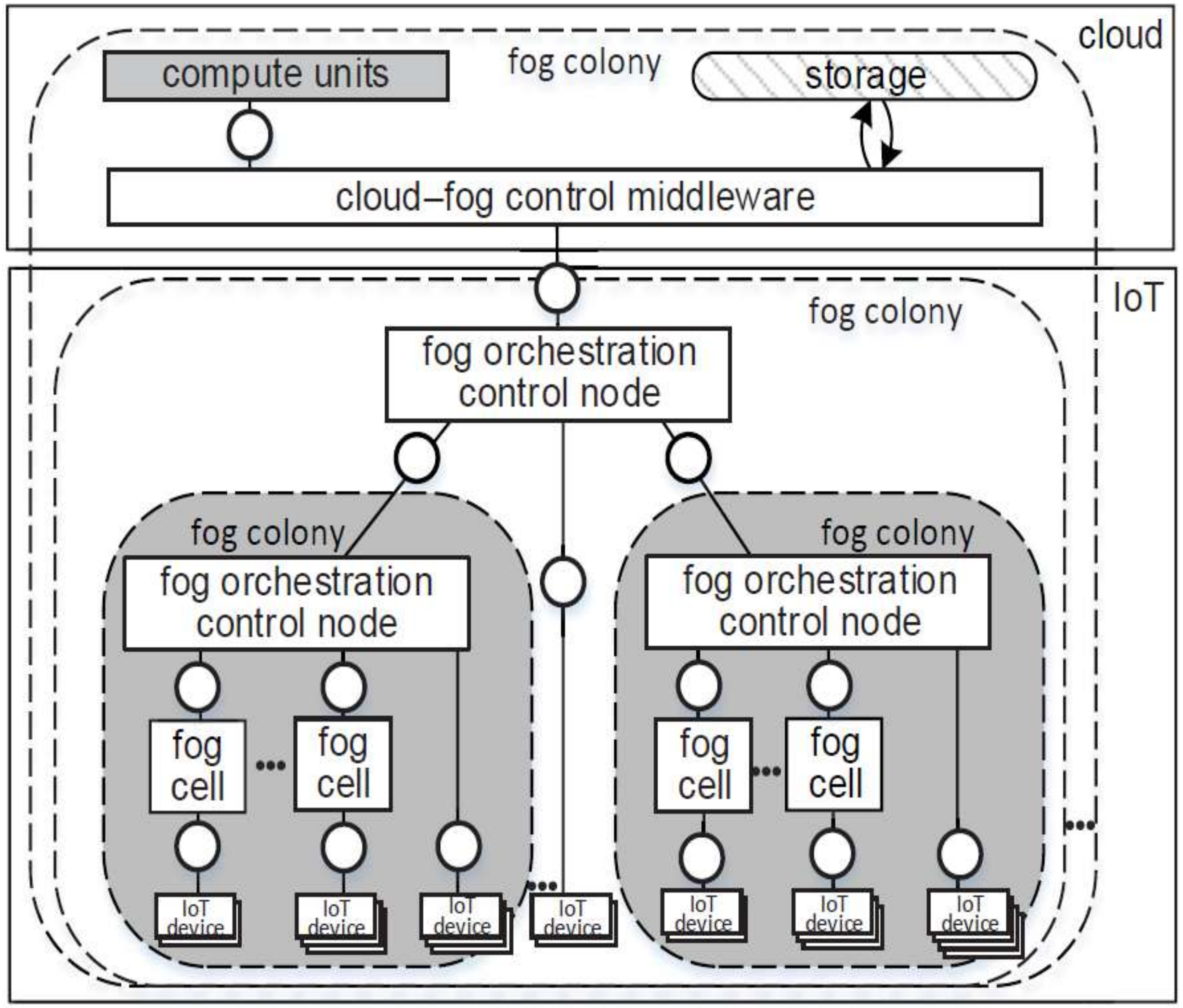
i) Control plane: Centralized vs. Distributed

- **Centralized policy**

  - Access to the entire resource and network state, application state, workload information (global view)
    - Pro: Capable of determining optimal global solutions
    - Cons: Scalability

- **Distributed policy**

  - Take decision based on local information
    - Pro: Scalability, better suited for runtime adaptation
    - Cons: Optimality is not guaranteed



ii) Offline vs. Online Placement

- **Offline Placement**
  - Takes a deployment decision at the compile-time, where all required information are available
- **Online Placement**
  - A proactive scenario of service placement. The placement of services is calculated and applied periodically

iii) Static vs. Dynamic Placement

- **Static**
  - No changes in infrastructure and/or application(s) topologies or characteristics
- **Dynamic**
  - Dynamic numbers of devices appearing and disappearing, dynamic workload distribution
  - Deploy new service, moving an operator from one host to another, or releasing service

iv) Mobility support

- **Mobility**
  - Handle the mobility of terminal nodes (resp. fog node) which can frequently change locations from one subnetwork to another
  - Ensure that associated users always receive the desired performance in terms of delay, capacity, etc

Service Placement Problem in Fog Computing

- **To Address the SPP**
  - ☑ ~~Problem statement~~
  - ☑ **Optimization strategies**
  - ☑ Evaluation environments

Optimization strategies

- **Optimization metrics**
  - *Most often considered:* Latency ; Utilization ; Cost ; Energy consumption
  - *Others:* Quality-of-experience; Blocking probability; Failed requests ; Number of computationally active Fog nodes:

- **Problem Formulation**
  - *Linear programming:* Integer Linear Programming (ILP), Integer Nonlinear Programming (INLP), Mixed Integer Linear Programming (MILP), Mixed-integer non-linear programming (MINLP), Mixed Integer Quadratic Programming (MIQP)
  - *Constraint programming*
  - *Others:* Markov decision process, stochastic optimization, Potential games, …

- **Resolution approaches**
  - Service placement: NP-hard problem
  - Exact optimization method —> scaling problem (fail to solve the problem on the Big Data scale)
  - The main focus of work within the research community is based on providing an effective approximation, heuristic or meta-heuristic approaches

Service Placement Problem in Fog Computing

- **To Address the SPP**
  - ☑ ~~Problem statement~~
  - ☑ ~~Optimization strategies~~
  - ☑ **Evaluation environments**

Evaluation environments

- **Analytic tools**
  - *Most often used:* Java ; C++ ; Matlab ; Optimization engine (IBM CPLEX, Gurobi…)
- **Simulators**
  - *Most often used:* CloudSim, SimGrid, iFogSim, OMNet++
- **Testbeds**
  - *Most often used:* FIT/IoT-LAB, Grid5000, OpenStack

How can we evaluate/compare the different proposals?

Agenda

I Part I. Overview of service placement problem in Fog environment

- ☑ ~~Context~~
- ☑ ~~Service Placement Problem (SPP) in Fog Computing~~
- ☑ **Classification of reviewed works**

II Part II. Service Placement Problem using Constraint programming and Choco solver

- ☑ System model and problem formulation
- ☑ Evaluation

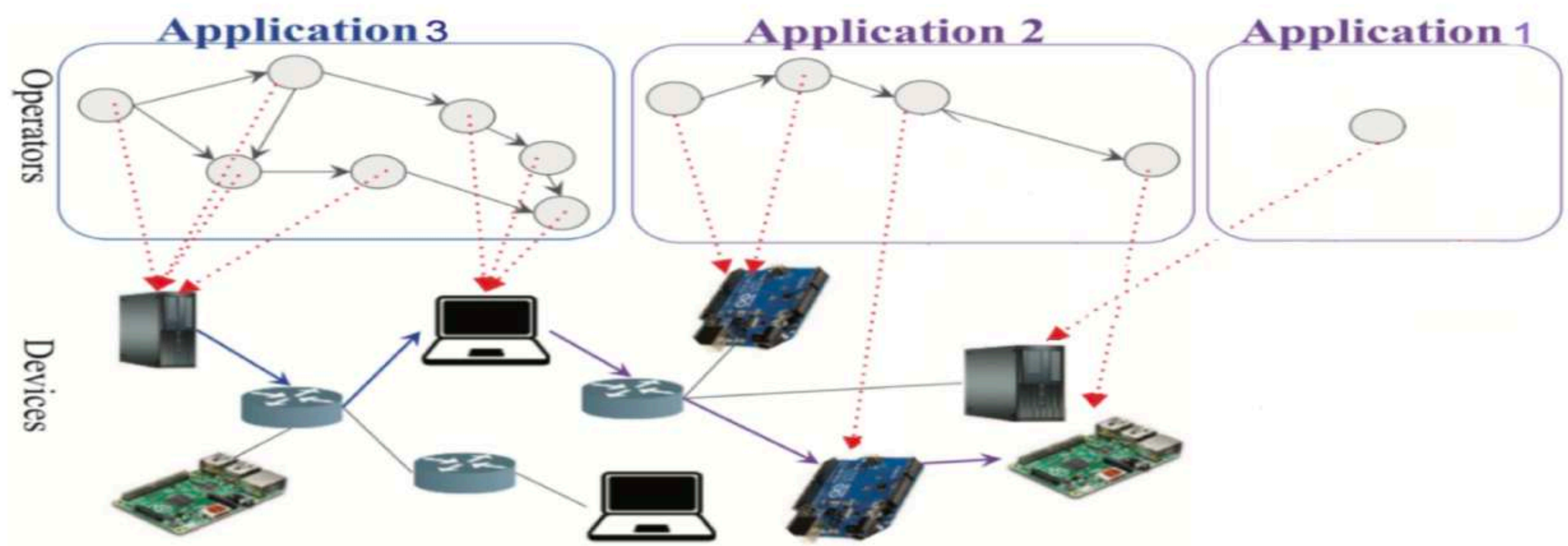
Conclusion & current + future work

Classification of reviewed works

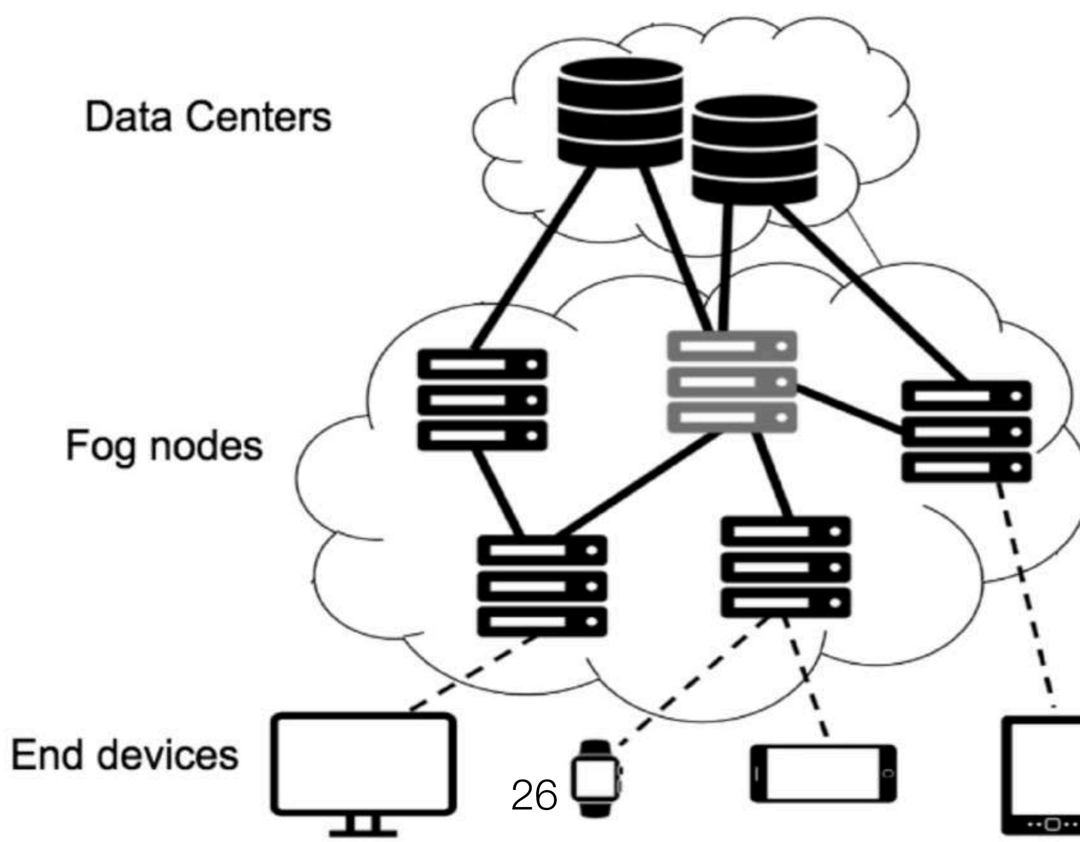
- **Classification**
  - ☑ Identified Scenarios (use-cases)
  - ☑ Classification according to Placement taxonomy
  - ☑ Classification according to resolution approaches

Identified Scenarios (use-cases)

- **Scenario 1: Assign application(s) according to QoS requirements**

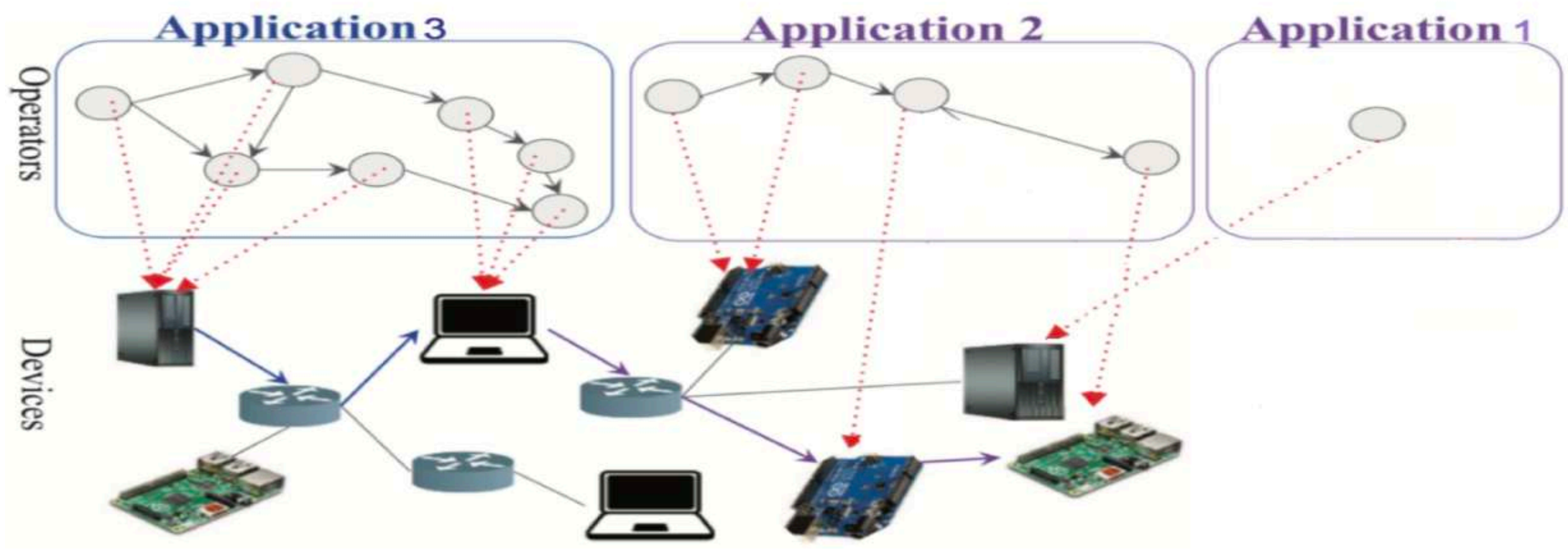


- **Scenario 2: Ensure latency and QoS for a service by service assignment**



Identified Scenarios (use-cases)

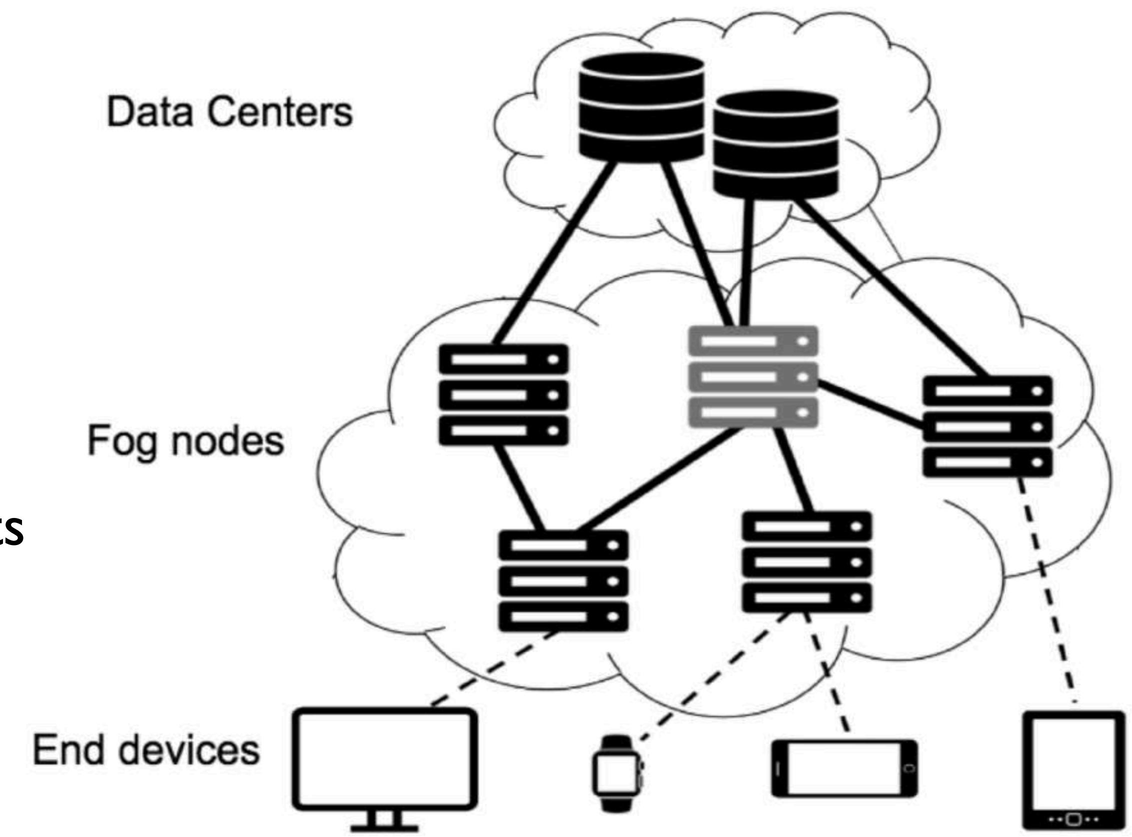
- **Scenario 1: Assign application(s) according to QoS requirements**



- **According to application definition we identify:**
  - ***Scenario 1.1:*** *Deploy applications that receives continuous data from one or more data sources*
  - ***Scenario 1.2:*** *Deploy a set of monolithic applications*
  - ***Scenario 1.3:*** *Deploy a set of applications each abstracted as a set of interdependent components*
  - ***Scenario 1.4:*** *Deploy a set of services each abstracted as a Directed Acyclic Graph (DAG).*
  - ***Scenario 1.5:*** *Deploy a set of services each abstracted as a Data Stream Processing.*

Identified Scenarios (use-cases)

- **Scenario 2: Ensure latency and QoS for a service by service assignment**



- Dissemination of data in a Fog environment
- Replication can exploit geographical locality of requests in addition to temporal locality

- ***Challenges***: Decision on replication and placement of data originating from a central server towards the end devices in Cloud network
- Find answers to the following questions: *Which data objects to replicate? When to create or destroy a replica? How many replicas for each object to create? Where to store each replica? How to redirect requests to the closest replica?*

Classification of reviewed works

- **Classification**
  - ☑ **Identified Scenarios (use-cases)**
  - ☑ Classification according to Placement taxonomy
  - ☑ Classification according to resolution approaches

Service model		Papers
Scenario 1	Scenario 1.1	[5, 8, 12, 14, 21, 38–40, 45, 48, 50, 57, 70, 71, 82, 84, 97, 104, 105, 125, 139, 140, 144, 145, 157? –160]
	Scenario 1.2	[29, 30, 49, 62, 74, 92, 111, 123, 127, 130, 131, 135, 137]
	Scenario 1.3	[66, 87, 98, 118, 122, 128, 129, 142, 143]
	Scenario 1.4	[13, 15, 17, 24, 25, 41, 51, 60, 65, 69, 72, 89, 93, 94, 99, 126, 136, 144, 150, 153]
	Scenario 1.5	[10, 16, 28, 52]
Scenario 2		[9, 11, 96, 124, 162, 163]

Classification of reviewed works

- **Classification**
  - ☑ ~~Identified Scenarios (use-cases)~~
  - ☑ **Classification according to Placement taxonomy**
  - ☑ Classification according to resolution approaches

Scenario	Reference	Service placement taxonomy			
		Control plane	Online	Dynamic	Mobility
Scenario 1.1	[12, 21, 38–40, 45, 47, 50, 57, 70, 84, 97, 104, 105, 125, 160]	C	✓		
	[14, 159]	C	✓	✓	
	[140, 144]	C	✓	✓	✓
	[5, 8, 71, 81, 82, 139, 157, 158]	Di	✓		
	[67, 145]	Di	✓	✓	✓
Scenario 1.2	[62, 137]	C			
	[74, 92, 123, 135]	C	✓		
	[29]	C	✓	✓	
	[30, 111]	C	✓	✓	✓
	[131]	Di			
	[49, 127, 130]	Di	✓		
Scenario 1.3	[66]	C			
	[87, 118]	C	✓		
	[122, 142]	C	✓	✓	
	[98, 128, 129, 143]	Di	✓		
Scenario 1.4	[17, 24, 25, 51, 65, 69, 72, 94, 99, 114, 136, 150]	C			
	[41, 60, 89, 146, 153]	C	✓		
	[13, 93]	C	✓	✓	
	[15]	C	✓	✓	✓
	[126]	Di	✓		
Scenario 1.5	[10, 16, 28]	C	✓		
	[52]	C	✓	✓	
Scenario 2	[9, 163]	C			
	[96, 162]	C	✓		
	[11, 124]	Di	✓	✓	

Classification of reviewed works

- **Classification**
  - ~~Identified Scenarios (use-cases)~~
  - ~~Classification according to Placement taxonomy~~
  - **Classification according to resolution approaches**

- Scenarios 1.1 -

Category	Solutions	References	Objective
C/Off/S/nM	Approximation	[160]	*Finds the minimum congestion ratio.*
		[125]	*Each user aims to maximize its own quality of experience.*
	Heuristic	[38, 39]	*Minimizes the power consumption of the Cloud-Fog Computing.*
		[45]	*Minimizes the total task computing latency under reliability constraints.*
		[50]	*Minimizes service time and minimize expensive resource over provisioning.*
		[56]	*Minimizes the overall unit cost for deploying Fog Computing supported Medical Cyber-Physical System.*
		[70]	*Minimizes the blocking probability.*
		[84]	*Optimizes the network usage.*
		[104, 105]	*Minimizes the overall latency of storing and retrieving data in a Fog.*
		[12, 21, 40]	
	Meta-heuristic	[97]	*Minimizes the total energy consumption of the mobile.*
C/On/Dy/nM	Heuristic	[14]	*Minimizes failed requests.*
		[159]	*Minimizes overall cost (processing, storage, communication).*
C/On/Dy/M	Heuristic	[140]	*Minimizes the cost of execution and accounting for delays and location in constraints.*
		[144]	*Minimizes the average cost over time.*
Di/On/S/nM	Heuristic	[5]	*Minimizes the response time and maximize the throughput.*
		[8]	*Meets service level agreement and quality of services.*
		[139]	*Minimizes the cumulative delay of executing mobile services.*
		[158]	*Reduces the service delay for IoT applications.*
		[71, 81, 82, 157]	
Di/On/Dy/M	Heuristic	[145]	*Reduces the application delay of IoT applications in the Smart Grid.*

Classification of reviewed works

- **Classification**
  - ~~Identified Scenarios (use-cases)~~
  - ~~Classification according to Placement taxonomy~~
  - **Classification according to resolution approaches**

- Scenarios 1.2 -

Category	Solutions	References	Objective
C/Off/S/nM	Exact	[137]	*Minimizes the number of Fog nodes leads to an overall power-consumption minimization.*
		[62]	*Minimizes the overall communication cost.*
C/On/S/nM	Approximation	[135]	*Minimizes the total weighted response time over all the deployed jobs.*
	Heuristic	[74]	*Maximizes the utilization of the residual computation capacities of the end devices.*
		[123]	*Maximizes the service quality experienced by end users.*
	Meta-heuristic	[92]	*Ensures the best convergence between user expectations and scope within the Fog environment that eventually maximizes the QoE.*
C/On/Dy/nM	Heuristic	[29]	*Equally satisfies all applications.*
C/On/Dy/M	Heuristic	[111]	*Minimizes the long-term time-average service latency under the constraints of long-term cost budget.*
	Machine Learning	[30]	*Minimizes the service costs, and in the meantime, improve the QoE by providing different service resolutions based on user demands, user mobility and dynamic network resources.*
Di/Off/S/nM	Exact	[131]	*Minimizes the service latencies in a Fog Computing environment, while the fulfillment of capacity requirements were guaranteed.*
Di/On/S/nM	Heuristic	[127]	*Completing the user job at a minimum cost.*
		[130]	*Maximizes utilization of fog resources.*
		[49]	*Maximizes the revenue obtained in the provision of the Fog infrastructure to application tenants.*

Classification of reviewed works

- **Classification**
  - ~~Identified Scenarios (use-cases)~~
  - ~~Classification according to Placement taxonomy~~
  - **Classification according to resolution approaches**

- Scenarios 1.3 -

Category	Solutions	References	Objective, Algorithms, and tool
C/Off/S/nM	Heuristic	[66]	*Maximizes the number of satisfied requests.*
C/On/S/nM	Heuristic	[87]	*Minimizes the total inter-cloudlet communication traffic in the cloudlet mesh.*
		[118]	*Optimizes multiple objectives: maximize number of accepted IoT application requests, maximize service bandwidth, minimize service migrations between iterations, minimize number of active computational nodes, minimize the number of active gateways, minimize hop count between computational nodes and end devices, and minimize of path loss.*
C/On/Dy/M	Heuristic	[122]	*Optimizes task deployment over distributed edges in terms of saving bandwidth and reducing latency.*
		[142]	*Minimizes the hop count between users location and serving nodes, the hop count between communication nodes, and the number of service migrations.*
Di/On/S/nM	Exact	[98]	*Maximizes the number of tasks deployed on the Fog landscape.*
		[129]	*Maximizes the number of tasks deployed on the Fog landscape.*
	Heuristic	[129]	*Maximizes the number of service placements to Fog landscape.*
	Meta-heuristic	[128]	*Maximizes the number of service placements to Fog landscape (rather than to Cloud ones).*

Classification of reviewed works

- **Classification**
  - ~~Identified Scenarios (use-cases)~~
  - ~~Classification according to Placement taxonomy~~
  - **Classification according to resolution approaches**

- Scenarios 1.4 -

Category	Solutions	References	Objective
C/Off/S/nM	Exact	[17]	*Minimize the overall inter-node latency and ensure the QoS of the applications.*
	Heuristic	[24]	*Determines eligible deployments of composite applications to Fog infrastructures.*
		[25]	*Determines eligible deployments of composite applications by estimating the cost of deploying over Fog infrastructures.*
		[94]	*Finds a valid deployment that optimizes the different objectives: the minimum runtime, the minimum user cost and the maximum battery lifetime.*
		[51]	*Minimizes the overall cost (placement and link costs).*
		[65]	*Maximizes the number of satisfied IoT analytics.*
		[69]	*Minimizes end-to-end delay.*
		[72]	*Minimizes maximum cost service node.*
		[99]	*Minimizes network cost.*
		[136]	*Efficient utilization of network resources and minimize application latency.*
		[150]	*Minimises the average response time of deployed IoT applications.*
C/On/S/nM	Approximation	[146]	*Minimizes the maximum weighted cost on each physical node and link.*
	Heuristic	[41]	*Minimizes the provisioning cost.*
		[60]	*Determines an eligible application placement.*
		[89]	*Minimizes increment of energy consumption.*
		[153]	*Minimizes the response time.*
C/on/Dy/nM	Heuristic	[93]	*Minimizes network cost.*
	Meta-heuristic	[13]	*Multi-objective: Minimize cost, maximize user support, minimize latency, maximize user footprint.*
C/On/Dy/M	Heuristic	[15]	*Minimizes the cost to run the application.*
Di/On/S/nM	Heuristic	[126]	*Minimizes the cost to run the application.*

Classification of reviewed works

- **Classification**
  - ~~Identified Scenarios (use-cases)~~
  - ~~Classification according to Placement taxonomy~~
  - **Classification according to resolution approaches**

- Scenarios 1.5 -

Category	Solutions	References	Objective
C/On/S/nM	Exact	[16]	*Minimizes overall operational cost.*
		[28]	*Minimizes the application end-to-end latency.*
	Heuristic	[10]	*Minimises end-to-end latency.*
C/On/Dy/nM	Heuristic	[52]	*Minimizes the total makespan for all event analytics, while meeting energy and compute constraints of the resources.*
	Meta-heuristic	[52]	*Minimize the total makespan.*

- Scenarios 2 -

Category	Solutions	References	Objective
C/Off/S/nM	Exact	[163]	*Minimizes the average data traffic in the edge network.*
	Heuristic	[163]	*Minimizes the overall latency of storing and retrieving data in a Fog.*
		[9]	*Maximizes the energy efficiency while maintaining the successful delivery.*
C/On/S/nM	Heuristic	[162]	*Minimizes the maximum average task completion time.*
		[96]	*Fog-aware replica placement and context-sensitive differential consistency.*
Di/On/Dy/nM	Heuristic	[11]	*Minimizes storage cost.*
		[124]	*Allocates services taking into account the bandwidth of the network and the node availability.*

Classification of reviewed works

- **Classification**
  - ☑ ~~Identified Scenarios (use-cases)~~
  - ☑ ~~Classification according to Placement taxonomy~~
  - ☑ ~~Classification according to resolution approaches~~

- **Goal of this classification**
  - This classification aims to simplify the user's access to references in a particular category
  - Identify a flavor of some challenges that arise when deploying IoT applications in a Fog environment

Agenda

Part I. Overview of service placement problem in Fog environment

- ~~Context~~
- ~~Service Placement Problem (SPP) in Fog Computing~~
- ~~Classification of reviewed works~~

Part II. Service Placement Problem using Constraint programming and Choco solver

- System model and problem formulation
- Evaluation

Conclusion & current + future work

Service Placement Problem using Constraint programming and Choco solver

- **Goal**

  - Provide a generic and easy upgraded model able to handle the identified scenarios

  - As first study, we propose to consider the sub-cases of scenario 1, i.e., scenarios: 1.1, 1.2, 1.3, 1.4 and 1.5

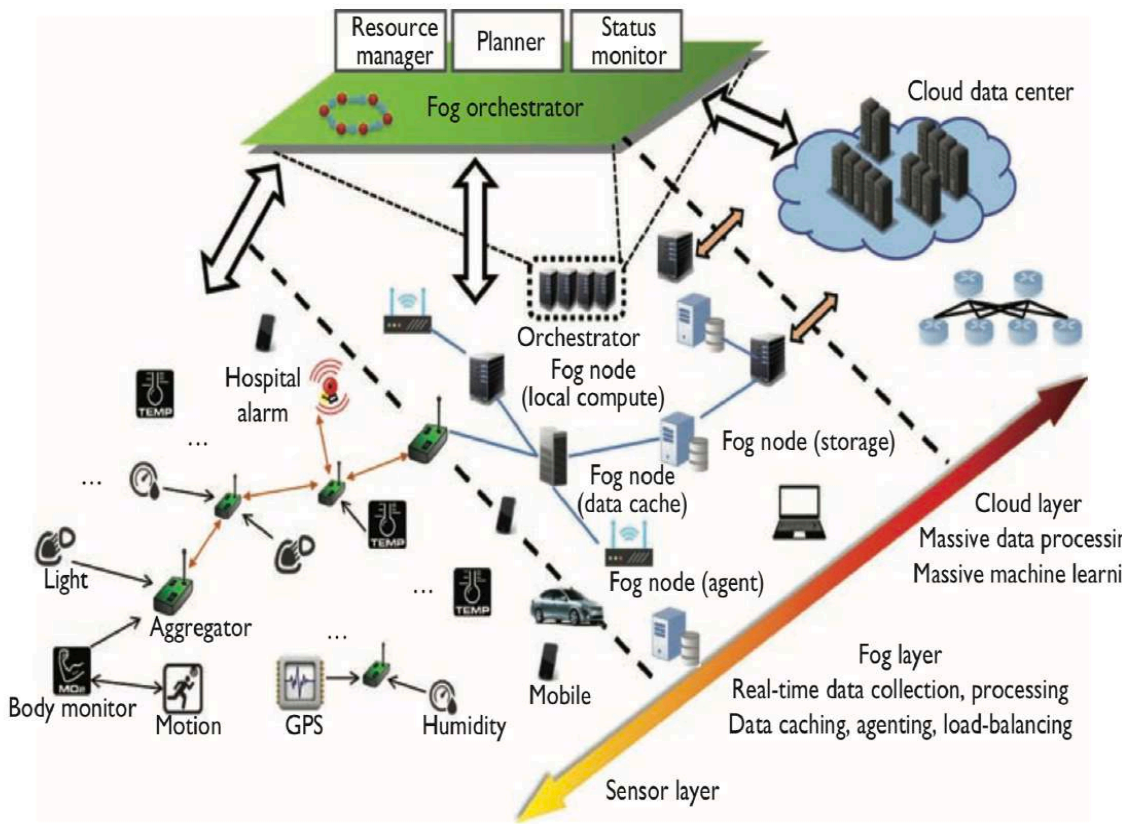
  - Provide a new formulation of the placement problem considering a general definition of service and infrastructure network through graphs using constraint programming

Published in:

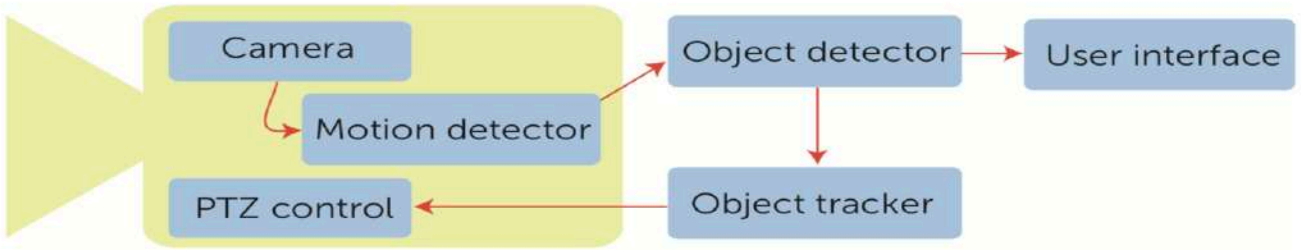
- *Service Placement in Fog Computing Using Constraint Programming*. F. Ait-Salaht, F. Desprez, A. Lebre, C. Prud'homme and M. Abderrahim. *IEEE* ***SCC 2019***, 2019.

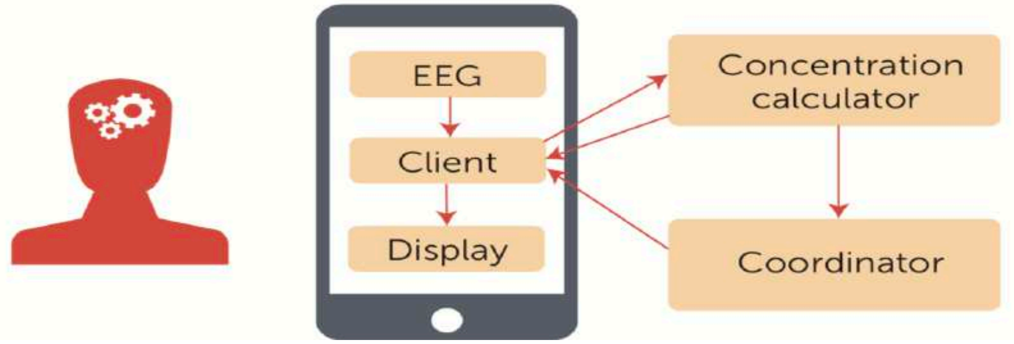
System model and problem formulation

- **Infrastructure**
  - A directed graph **G = <V,E>** represents the Network
  - **V**: set of vertices or nodes (server)
  - **E**: set of edges or arcs (connections)
  - Each node defines **CPU** and **RAM** capacities
  - Each arc defines a **latency** and a **bandwidth** capacity



- **Application**
  - An application is an ordered set of components
  - A component requires CPU/RAM to work
  - A component can send data (bandwidth, latency)
  - Some components are fixed (f-ex., cameras)





System model and problem formulation

- **Placement (Mapping)**

Assign services (each component and each edge) to network infrastructure (node and link) such that:

- **CPU capacity** of each node is respected
- Same goes with **RAM capacity**
- **Bandwidth capacity** is respected on arcs too
- **Latencies** are satisfied

Constraint Programming model (CP)

- **What is CP?**

  - **CP** stands for **Constraint Programming**
  - **CP** is a general purpose implementation of **Mathematical Programming**

    Like LP = Linear Programming or SAT = Clauses
  - MP theoretically studies optimization problems and resolution techniques
  - It aims at describing real combinatorial problems in the form of Constraint Satisfaction Problems and solving them with Constraint Programming techniques
  - The problem is solved by alternating constraint filtering algorithms with a search mechanism

Constraint Programming model (CP)

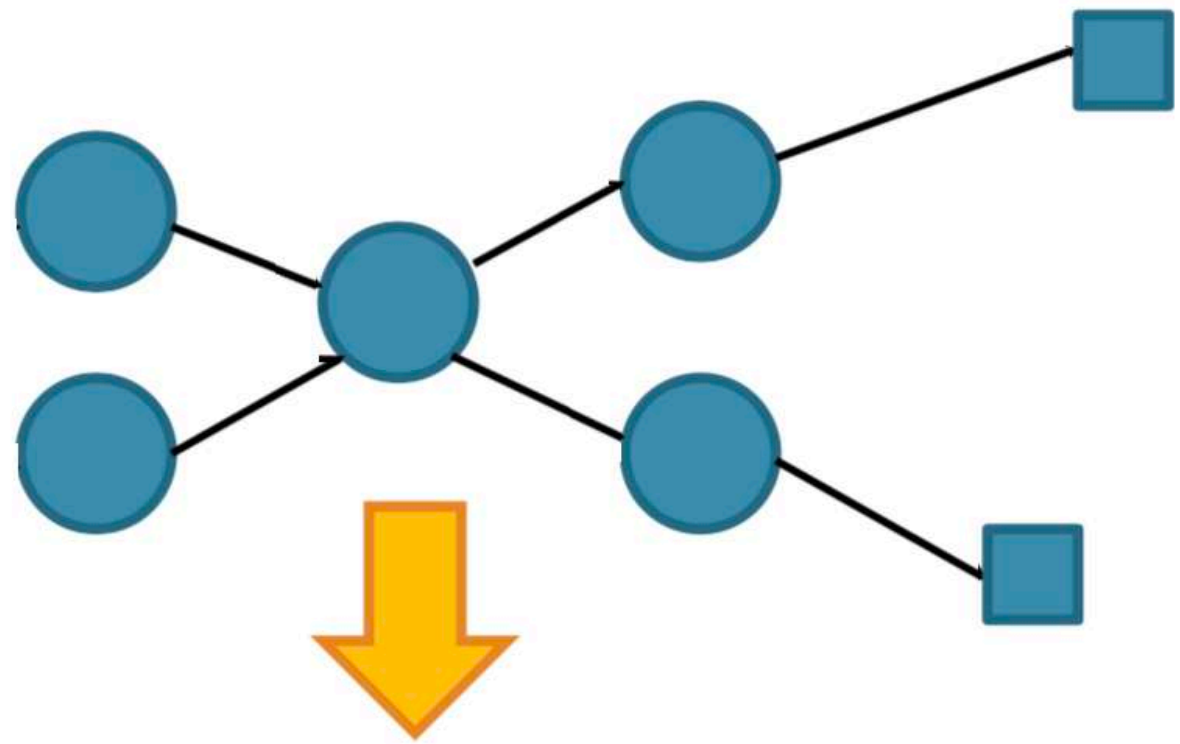
- ## Modeling steps

There are **3** main steps to follow:

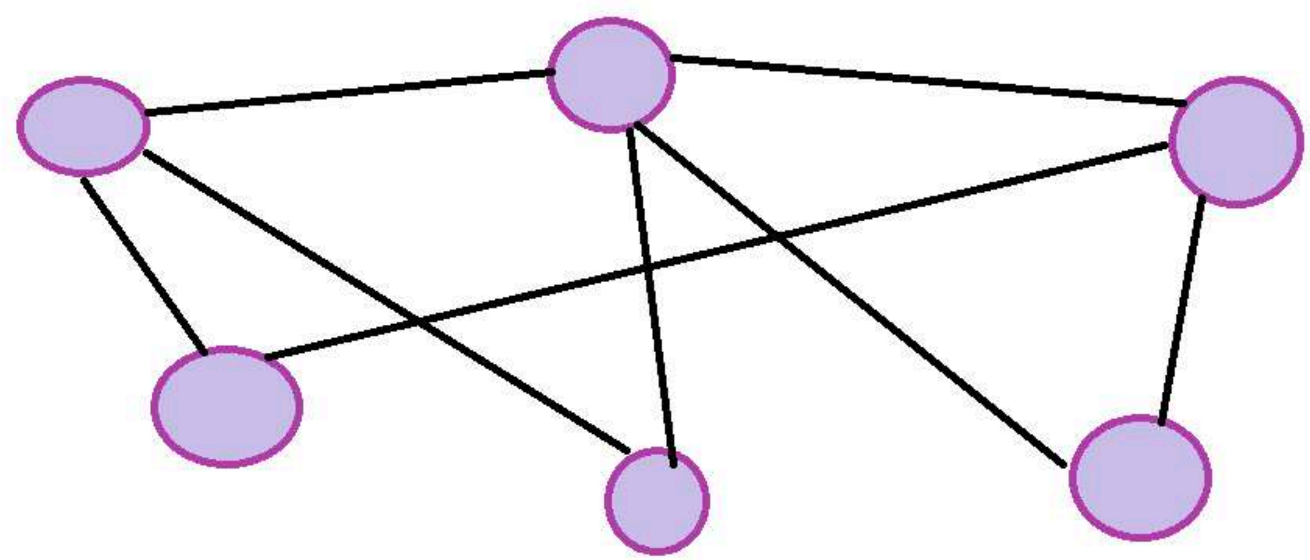
- Declare variables and their domain
- Find relation between them
- Declare a objective function, if any

Hint

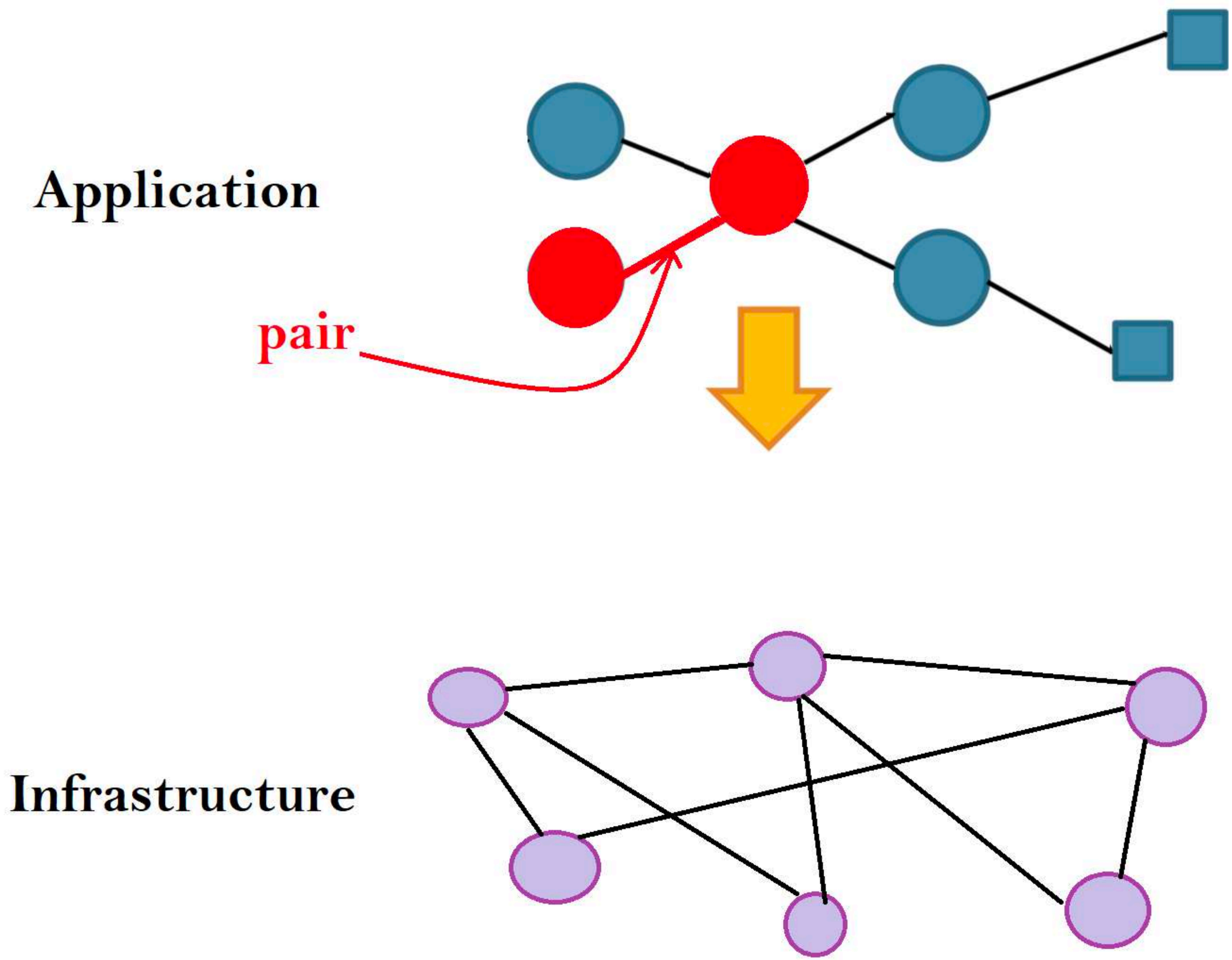
Application



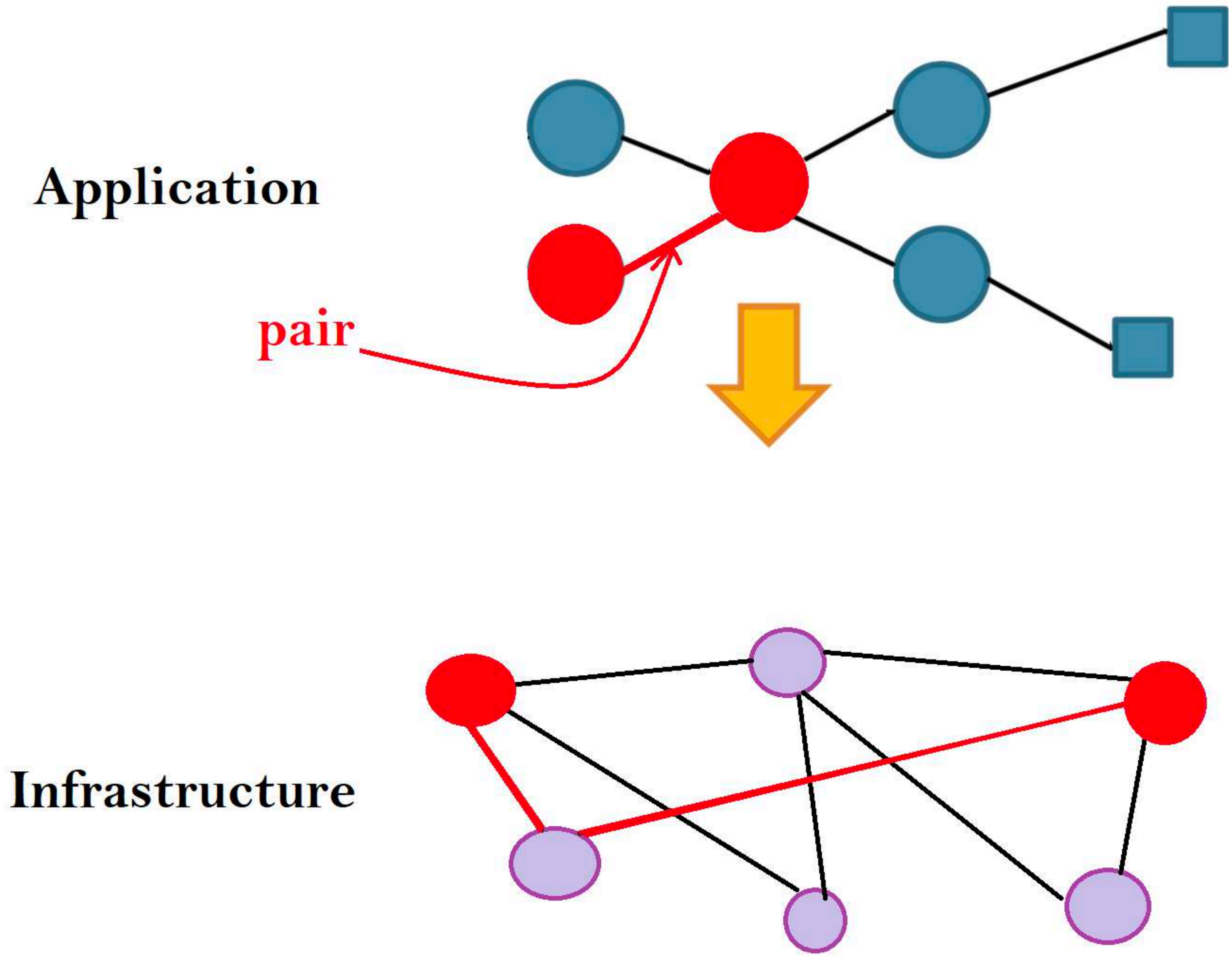
Infrastructure



Hint



Hint



Constraint Programming model (CP)

- **Variables and domains**

The **variables** related to **nodes** are:

- s_k: node that hosts the component source of pair k
- t_k: node that hosts the component sink of pair k
- h_i: node that hosts component i
- $n_{k,j}$: node at position j in the path of pair k
- p_k: position of t_k in n_k (position of $s_k = 0$)

k denotes a pair in service graph.

Constraint Programming model (CP)

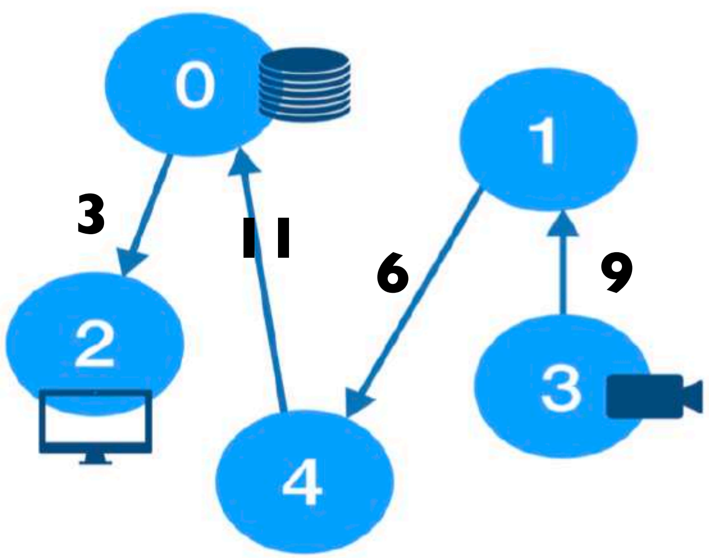
- **Variables and domains**

The **variables** related to **arcs** are:

- $a_{k,j}$: arc between $n_{k,j}$ and $n_{k,j+1}$ in path of pair k
- $b_{k,j}$: bandwidth on arc $a_{k,j}$
- $l_{k,j}$: latency on arc $a_{k,j}$

k denotes a pair in service graph.

Example of application mapping



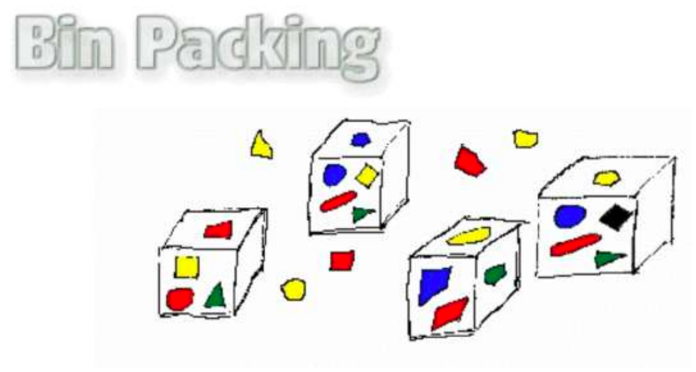
h_i	3	0	2	
p_k	3	1		

$k = 0$

n_k	3	1	4	0
a_k	9	6	11	
l_k	12	18	11	
b_k	700	700	700	

Constraints

- **Constraints on nodes**
  - **Straightforward ones, satisfying capacities:**
    - Respect CPU capacity of each node
      - **bin_packing(<h_k, cpu_comp>, CPU)**
    - Respect RAM capacity of each node
      - **bin_packing(h_k, ram_comp>, RAM)**



Constraints

- **Constraints on nodes**
  - **Links between h_k, $n_{k,j}$ and p_k:**
    - The item at position p_k in path n_k is equal to t_k:
      - **element**(t_k, **<**n_k**>**, p_k) $\Longleftrightarrow$ t[k]=n[k][p[k]]
    - Position of t_k if source and sink of a pair are identical:
      - $s_k = t_k \iff p_k = 1$
    - Avoid cycles (all items of n_k are different except $s_k = t_k$):
      - **allDifferent**($n_{k,[1:]}$)
    - Contiguous pairs:
      - $t_k = s_j$, **for two adjacent pairs** k **and** j
    - Positions are lexicographically ordered:
      - $n_{k,0} = s_k$

Constraints

- **Constraints on arcs**

  - **Straightforward ones, satisfying capacities:**

    - Respect bandwidth limit of each arcs:
      - **bin_packing(<$b_{k,j}$, bdw_pair>, BDW)**
    - Satisfy latencies, per pair :
      - **element($l_{k,j}$, <LAT>, $a_{k,j}$)** $\iff l[k][j] = LAT[a[k][j]]$
      - **sum(l_k, "<=", lat_pair$_k$)**

Constraints

- **Constraints between nodes and arcs**

- **Connect everything together**
  - Represent paths extensively, per service:
    - $table(n_{k,j}, a_{k,j}, n_{k,j+1}, T)$

  where T is a set of tuples extract from infrastructure graph G, that lists all links.

Summary

- We defined
  - A very basic model
  - Satisfying all rules
  - Easy to upgrade
  - Ready to be solved …
- And now ?
  - Implement the model on Choco solver
    - Choco is a Free Open-Source Java library dedicated to Constraint Programming

Agenda

I Part I. Overview of service placement problem in Fog environment

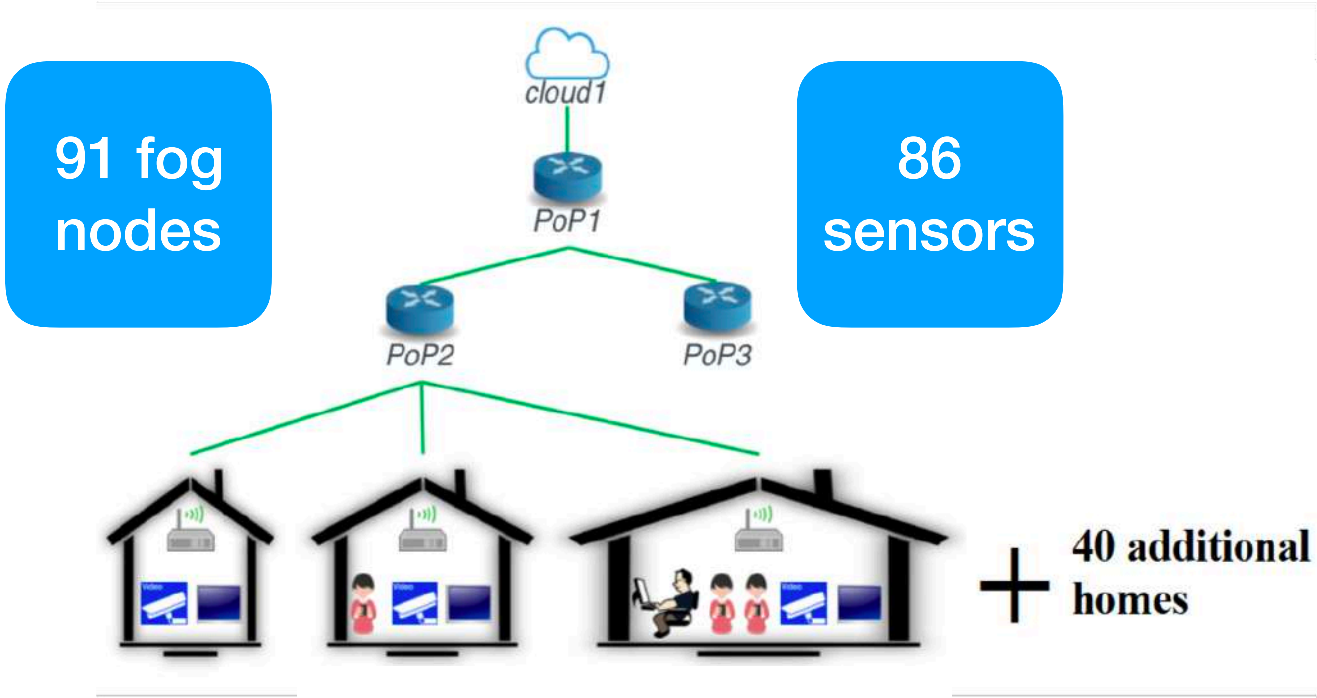
- ☑ ~~Context~~
- ☑ ~~Service Placement Problem (SPP) in Fog Computing~~
- ☑ ~~Classification of reviewed works~~

II Part II. Service Placement Problem using Constraint programming and Choco solver

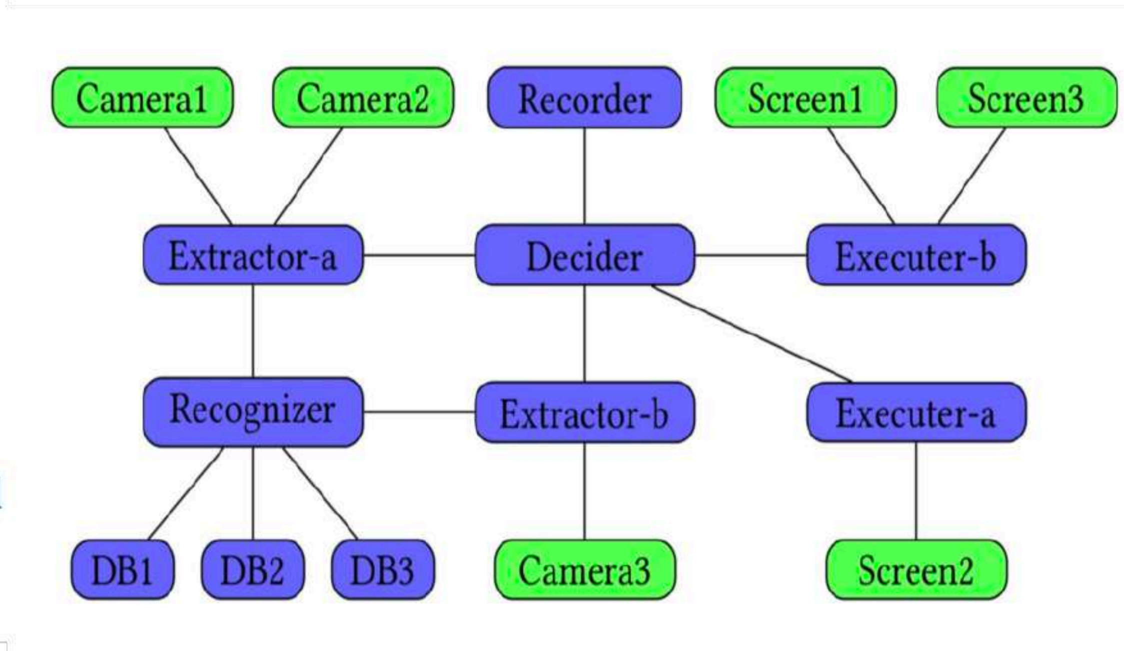
- ☑ ~~System model and problem formulation~~
- ☑ **Evaluation**

Conclusion & current + future work

Experiment I



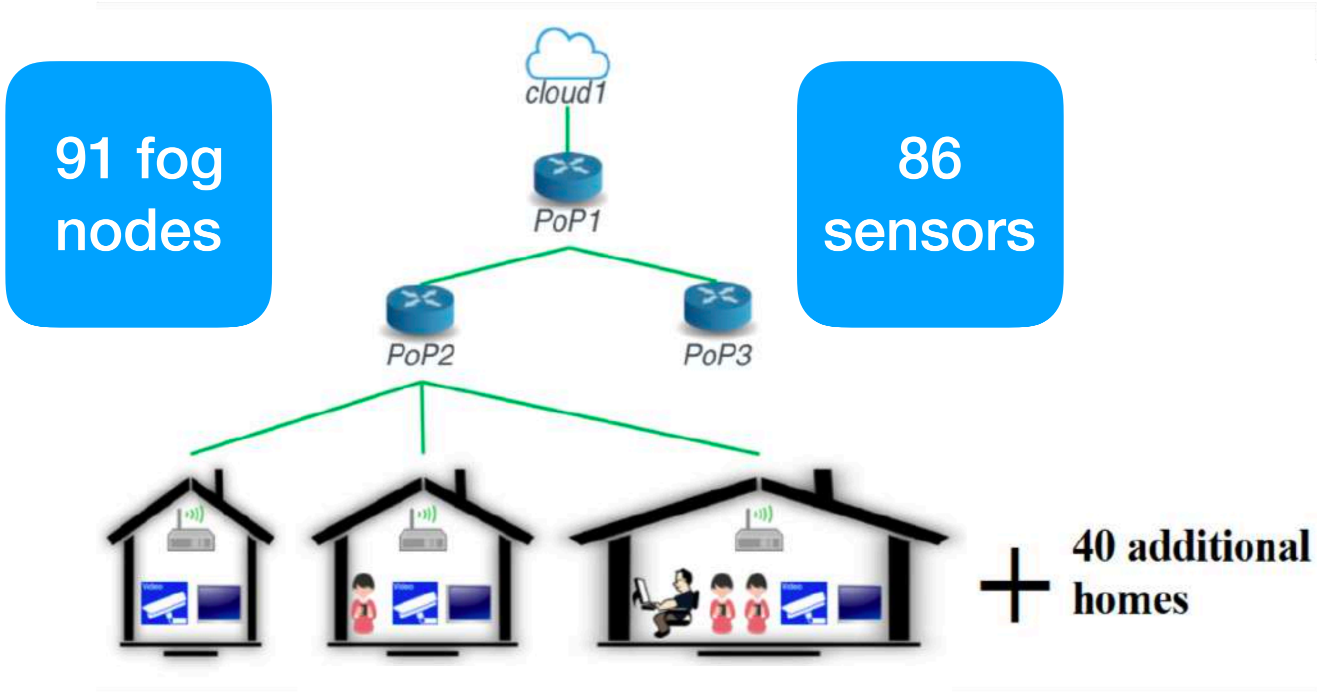
Infrastructure



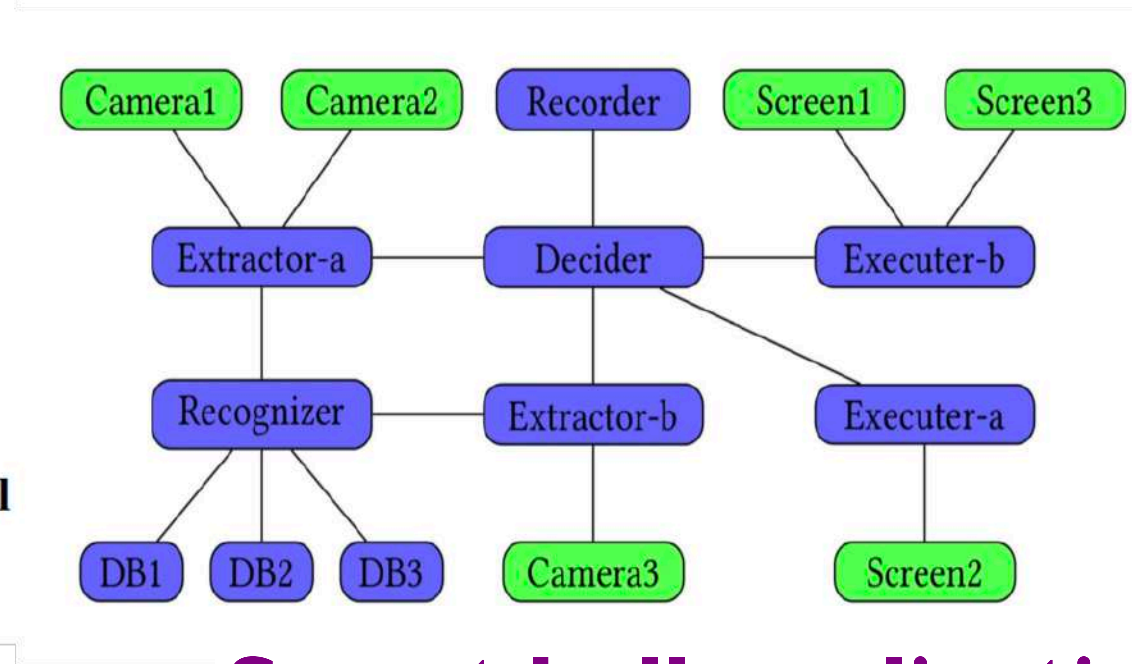
Smart bell application

- **Requirements**
  - Resources: CPU, RAM, DISK
  - Networking: Latency and Bandwidth
  - Locality
- **Objective**
  - Minimize average latency

Experiment I



Infrastructure



Smart bell application

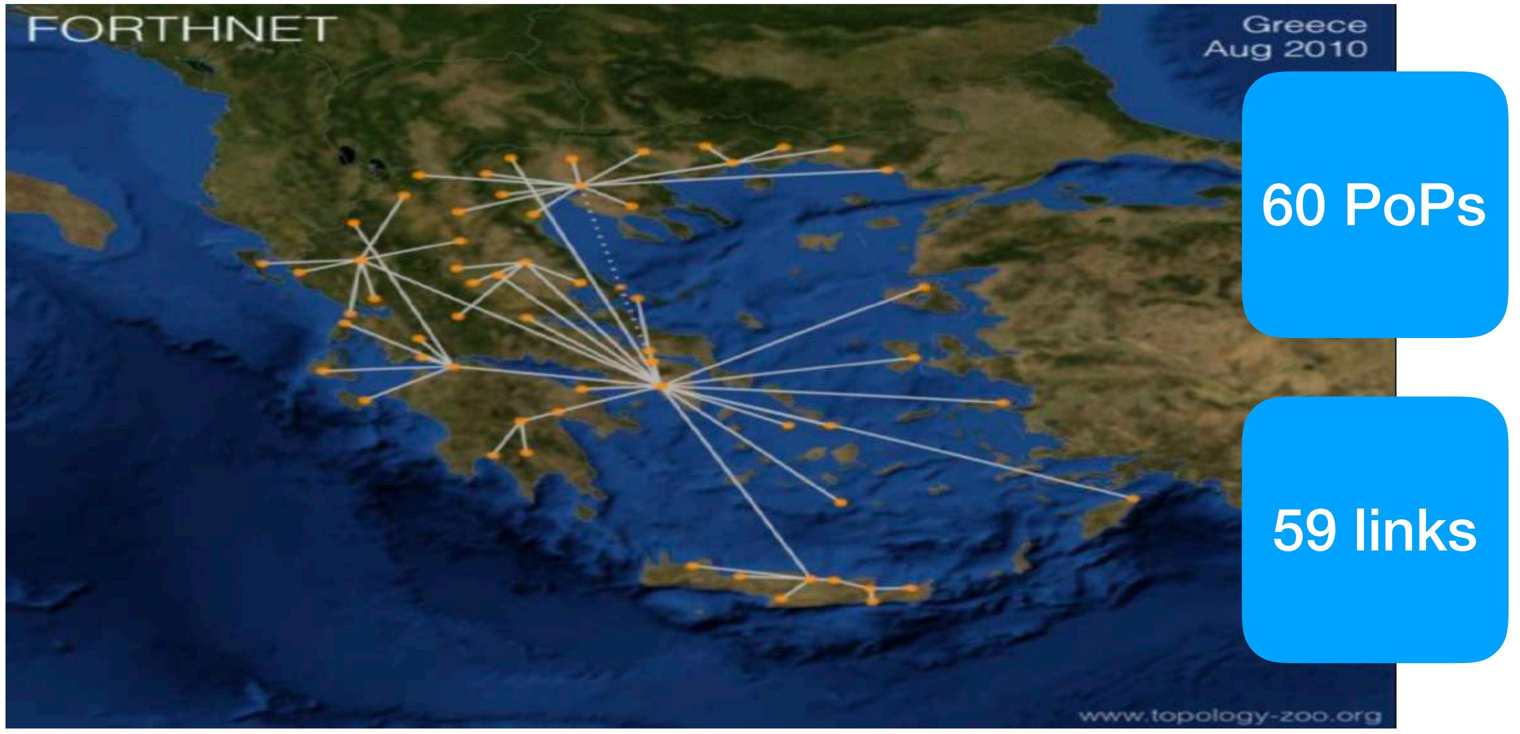
Algorithm	Resolution time (s)	Quality of the solution
ILP	343	100%
First Fit	265	186.8%
GA	28	143.9%
DAFNO-InitCO-DCO(0.3)	0.003	100.3%
CP-SPP	**0.559**	**100%**

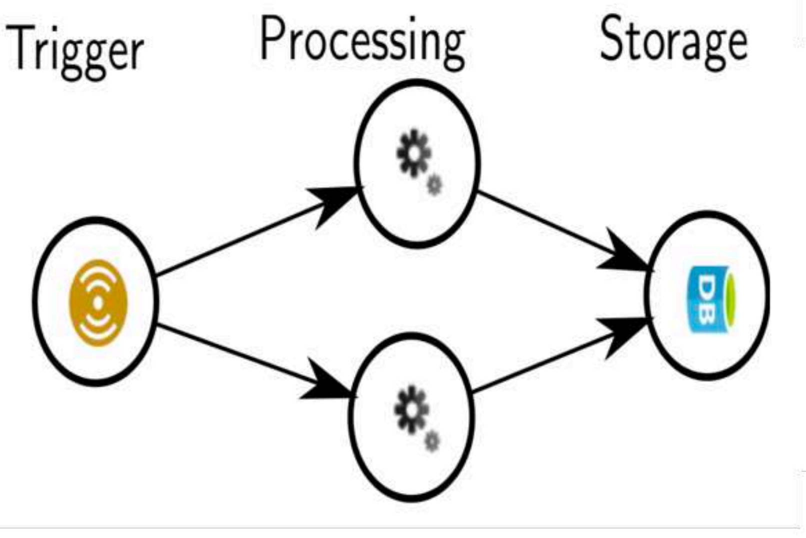
TABLE I: Evaluation Results of Different Placement Algorithms for the Smart Bell application [9].

Experiment 2

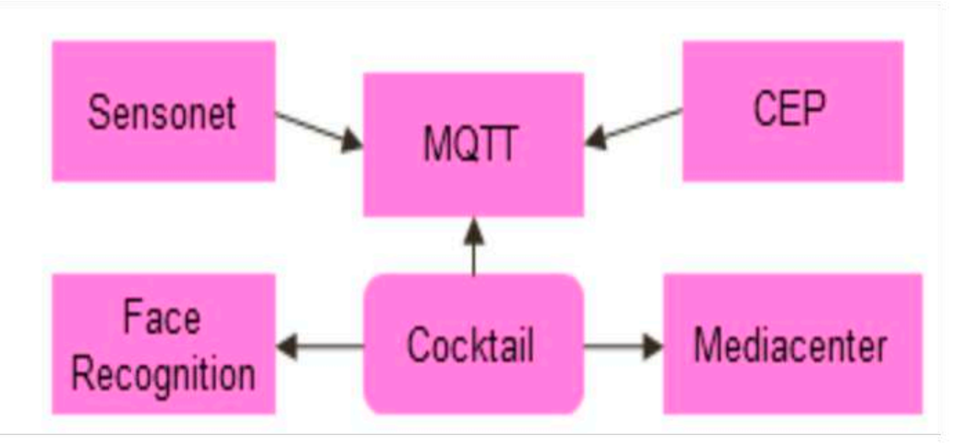
Infrastructure

Greek Forthnet topology

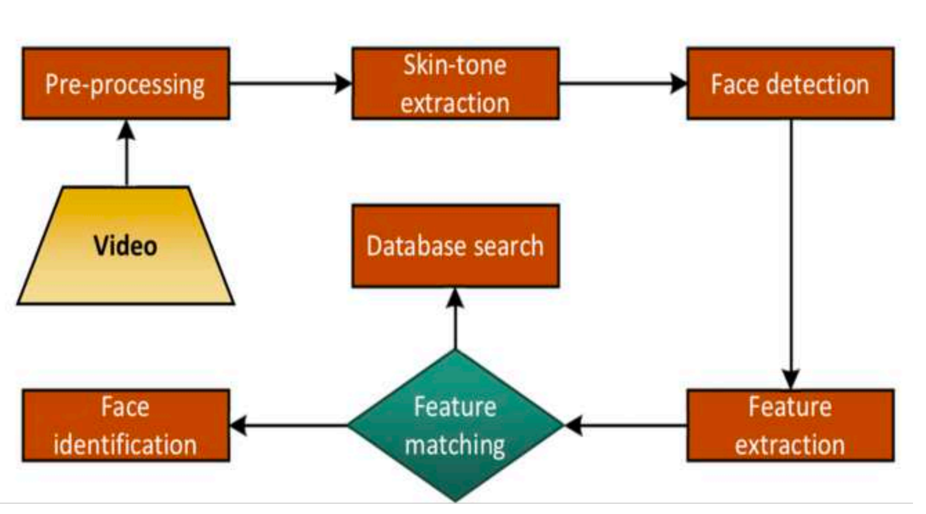




(a)



(b)

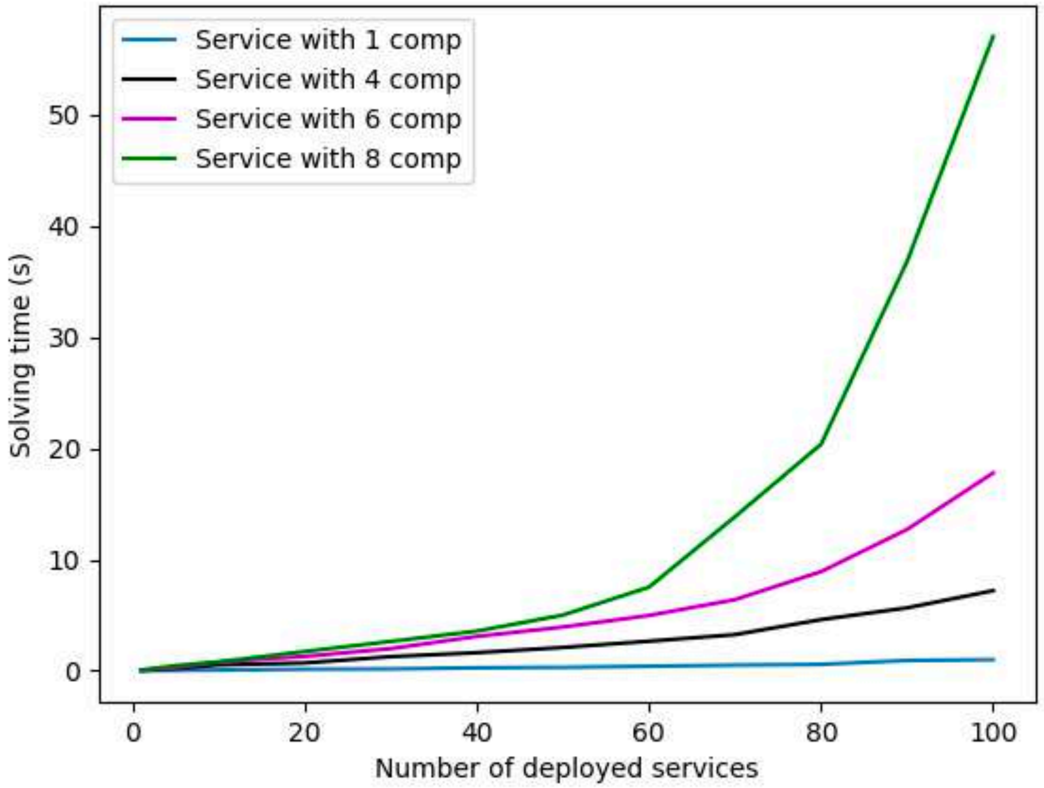


(c)

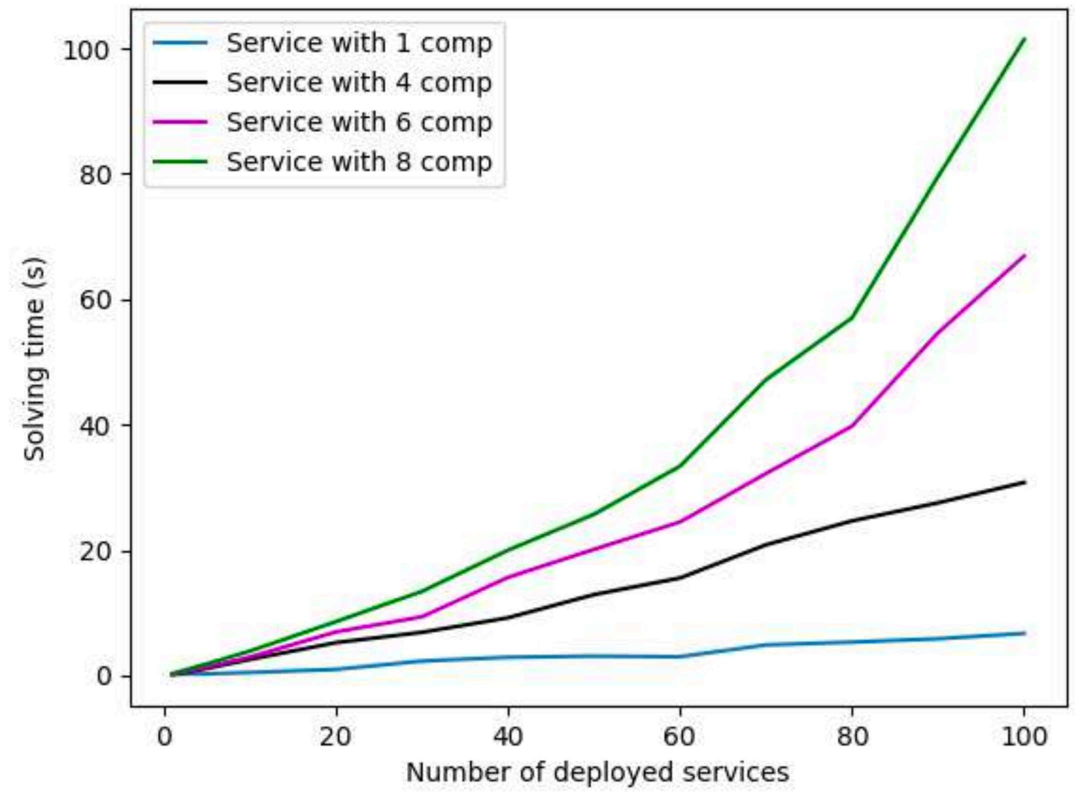
Applications

(a) Storage Application, (b) Smart Bell application, and (c) A face recognition application

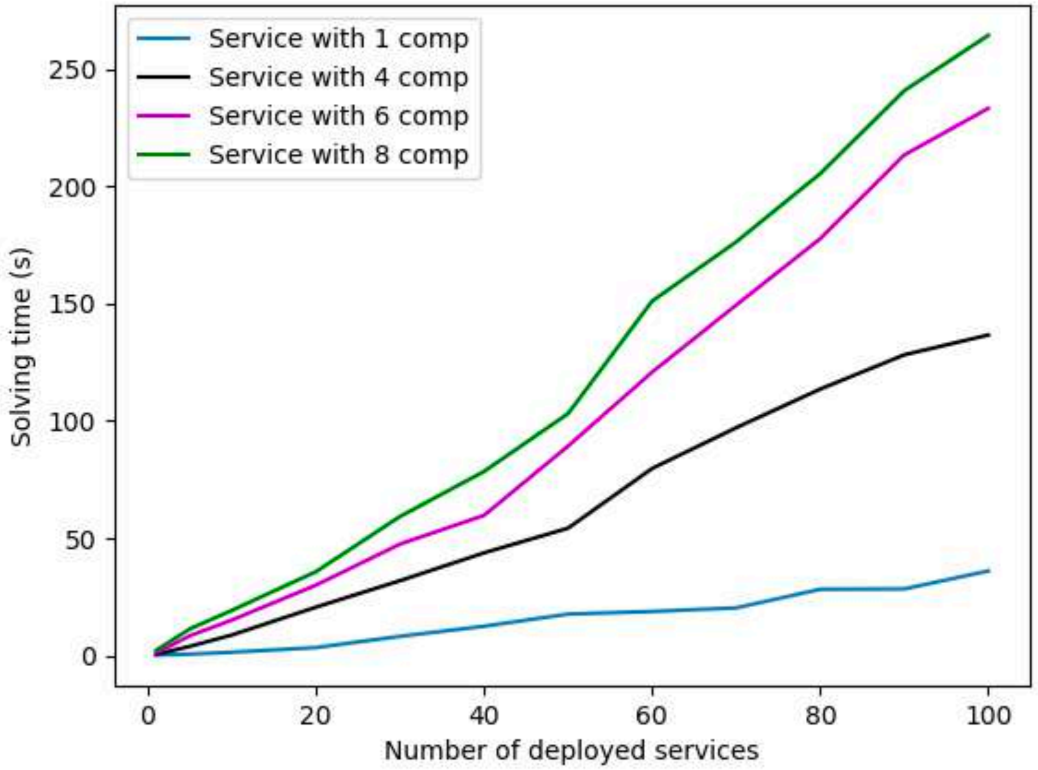
Experiment 2



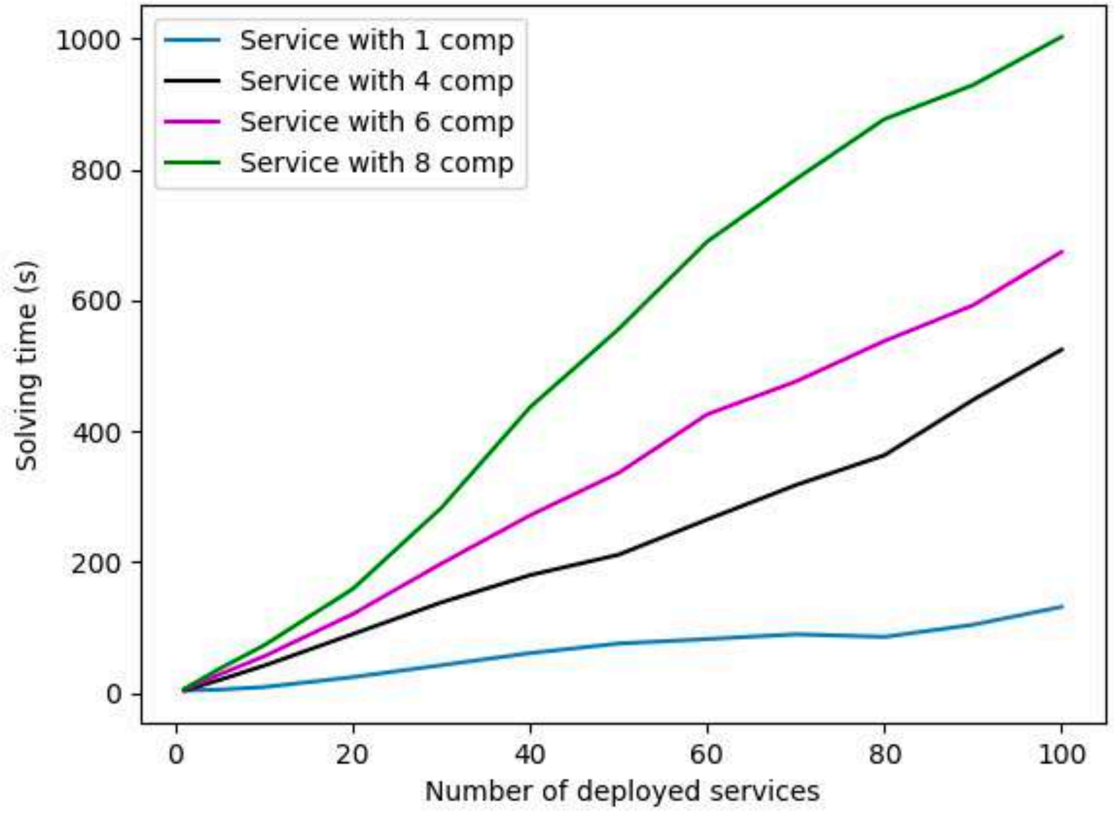
(a) For G with 120 nodes



(b) For G with 300 nodes



(c) For G with 600 nodes



(d) For G with 1200 nodes

Agenda

I Part I. Overview of service placement problem in Fog environment

- ☑ ~~Context~~
- ☑ ~~Service Placement Problem (SPP) in Fog Computing~~
- ☑ ~~Classification of reviewed works~~

II Part II. Service Placement Problem using Constraint programming and Choco solver

- ☑ ~~System model and problem formulation~~
- ☑ ~~Evaluation~~

Conclusion & current + future work

Conclusion & current + future work

- **Conclusion**
  - Present a classification of the reviewed papers in order to simplify the user's access to references in a particular category Identify a flavor of some challenges that arise when deploying IoT applications in a Fog environment
  - Provide service placement model that can not only be easily enhanced (deployment constraints/objectives), and upgraded (exploiting any resolution approach) but that also shows a competitive tradeoff between resolution times and solutions quality
- **Future work**
  - Extend our model to include the scenario 2
  - Extend our model to include the notion of the service sharing
  - Investigate the relevance of our model in the context of the reconfiguration
- **Current work** ***(join work with C. Perez, F. Desprez and L. Lefèvre)***
  - Concurrent Planning and Execution Phases in Reconfiguration Loops
  - Investigate placement strategies under Planning phases


Thank You
For Your Atten