New Horizons in IoT Workflows Provisioning in Edge and Cloud Datacentres for Fast Data Analytics: OSMOTIC COMPUTING APPROACH

Prof. Rajiv Ranjan, PhD SMIEEE ECS
Professor of Computing Science
Chair of Internet of Things
Newcastle University, United Kingdom
E raj.ranjan@ncl.ac.uk web: http://rajivranjan.net/
Outline

- IoT landscape
- IoT application context
- Big Data context
- Osmotic computing approach
- Research Challenges
Motivation
Internet of Things (IoT) Landscape

- Environmental Sensors
- Smart Meters
- Water Management
- Smart Grids
- Smart Agriculture

+ Software as a Service
+ Cloud computing
+ Software Defined Networking
+ Edge Computing
+ Ambient services
**IoT and Big Data Space**

### Annual Size of the Global Datasphere

<table>
<thead>
<tr>
<th>Measurement</th>
<th>Definition</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kilobyte (kB)</td>
<td>$10^3 \text{ Byte}$</td>
<td>$1.000 \text{ Byte}$</td>
</tr>
<tr>
<td>Megabyte (MB)</td>
<td>$10^6 \text{ Byte}$</td>
<td>$1.000.000 \text{ Byte}$</td>
</tr>
<tr>
<td>Gigabyte (GB)</td>
<td>$10^9 \text{ Byte}$</td>
<td>$1.000.000.000 \text{ Byte}$</td>
</tr>
<tr>
<td>Terabyte (TB)</td>
<td>$10^{12} \text{ Byte}$</td>
<td>$1.000.000.000.000 \text{ Byte}$</td>
</tr>
<tr>
<td>Petabyte (PB)</td>
<td>$10^{15} \text{ Byte}$</td>
<td>$1.000.000.000.000.000 \text{ Byte}$</td>
</tr>
<tr>
<td>Exabyte (EB)</td>
<td>$10^{18} \text{ Byte}$</td>
<td>$1.000.000.000.000.000.000 \text{ Byte}$</td>
</tr>
<tr>
<td>Zettabyte (ZB)</td>
<td>$10^{21} \text{ Byte}$</td>
<td>$1.000.000.000.000.000.000.000 \text{ Byte}$</td>
</tr>
<tr>
<td>Yottabyte (YB)</td>
<td>$10^{24} \text{ Byte}$</td>
<td>$1.000.000.000.000.000.000.000.000 \text{ Byte}$</td>
</tr>
</tbody>
</table>

Source: IDC's Data Age 2025 study, sponsored by Seagate, April 2017

Data created
£4.3m total capital investment to 2021
£25m+ total research revenue
3500+ sensors deployed to date
Largest sensor deployment in the UK
1,000,000,000 observations > 2000 per minute
Largest set of open environment monitoring data in the world
(?)
64+ variables, 24 platforms
- Weather, traffic, water flow, water quality, bees, traffic, people flows, air quality, energy consumption, waste and clean water, noise
Monitored infrastructure and buildings
CCTV – 240+ cameras > 100,000,000 images
Relationships with city stakeholders
• 1 billion + data points
• 3600 deployed sensors
• Scaleable data platform, APIs and downloads
• 62+ Variables
• 300+ CCTV feeds
• 100,000 images daily
• 5,000,000 observations daily

National Urban Observatory Facility Newcastle  
http://newcastle.urbanobservatory.ac.uk
Cloud + IoT + Big Data Approaches

- High latency
- Wasting of bandwidth

Cloud + IoT sensors + IoT sensors + IoT sensors

AWS IoT

Azure IoT

Google Cloud IoT
New IoT Programming Paradigm

Cloud Datacenter

- Abundant storage
- High availability
- High compute capacity

Cloud Layer
- Massive data processing
- Massive machine learning

Edge Layer(s)
- Real time data collection, processing
- Data caching, agenting, load-balancing

IoT Layer
- body monitor
- humidity
- motion
- light
- aggregator
- hospital alarm
- GPS
- mobile

Edge Node (Agent)
- Edge Node (Local Compute)
- Edge Node (Data Cache)
- Edge Node (Storage)

Restricted Storage
- Extremely low latency

Limited storage
- High latency
- Limited compute capacity

Massive Data Processing
- Real time data collection, processing
- Data caching, agenting, load-balancing

New IoT Programming Paradigm
Motivation

**Osmotic Computing**

Osmotic computing is a new paradigm to support the efficient execution of Internet of Things (IoT) services (microservices) and applications between cloud datacenter and the network edge by providing increased resource and data management capabilities at the edge of the network.

Osmosis Process

- In chemistry, “osmosis” represents the seamless diffusion of molecules from a higher to a lower concentration solution.

- Dynamic management of (micro)services across cloud and edge datacenters
  - deployment, networking, and security, …
  - providing reliable IoT support with specified levels of QoS.
Osmotic Computing

Osmotic Computing paradigm

- Restricted Storage
  - Extremely low latency
- Limited storage
  - High latency
  - Limited compute capacity
- Abundant storage
  - High latency
  - High availability
  - High compute capacity

Cloud Datacenter

- Cloud Layer
  - Massive data processing
  - Massive machine learning

- Edge Layer(s)
  - Real time data collection, processing
  - Data caching, agenting, load-balancing

- IoT Layer
  - Body monitor
  - Light
  - Humidity
  - Motion
  - GPS
  - Hospital alarm
  - Mobile

Institutions: Newcastle University, TU Wien, Fakultät für Informatik, Cardiff University, UFRN, University of Swinburne
Osmotic Computing: Federated View
Building Blocks: Containerized Microservices

- Division of functionality into multiple components
- Easy to manage and upgrade
- Lightweight
- Improved performance
Mixing Up

- IoT Sensors
- MICROservices
- Edge Computing
- Cloud Computing

Osmotic Computing
Abstract View of Osmotic IoT Application

- The dependency among various microservices is represented by a topologically ordered directed acyclic graph (DAG).
- Each microservice belongs to either a set of pre-defined IoT functions $\Psi$ or user-defined functions $\Omega$.
- Each microservice has specific hardware and software requirements $RH$ and $RS$ respectively. Some QoS constraints are also associated with each microservice.
Osmotic Computing Concept

A tunable configuration of the resource involvement
- Adapt to the available resources
- Meet application requirements (latency in this paper)
Osmotic Computing Concept

IoT application

Partitioning

Distribution

Larger search space

Verification is hard
Osmotic Computing Challenges

How to abstract IoT Data Analysis and Flow Patterns?

IoT analysis patterns
- Data capture
- Data Store
- Data inference
- Data filter
- Data aggregate
- Data visualization
- Data translate
- Actuation

Data flow patterns
- IoT connect
- Data transfer
  - Edge to Edge
  - Edge to Cloud
  - Cloud to Cloud
  - Cloud to Edge

Air Quality IoT Data Analysis Workflow

Flood Prediction IoT Data Analysis Workflow

Osmotic Computing Challenges

Microservices’ Performance Characterisation:

- How to systematically undertake performance characterisation of data analysis activities (e.g., microservices) across different parts of the infrastructure (Cloud, Edge, and IoT)?
- How to reason about heterogeneous QoS implications across Cloud, Edge and IoT?
  - The performance of a data analysis activity mapped to a Cloud layer is quite different from a gateway and/or device in the Edge and IoT layer
  - Network stability, throughput optimality, routing delays, fairness in resource sharing, available bandwidth, and sensor battery state. [Edge and IoT]
  - End-to-end response times, platform scalability and reliability, virtual server utilizations, and the costs of moving data to and from the Cloud. [Cloud]
Microservices’ Performance Characterisation

- Current benchmark kernel implementation can only test performance of specific type of IoT data analysis activities
  - TPCx-IoT can benchmark Edge layer (data aggregation, real-time analytics and persistent storage)
  - Google ROADEF & Linear Road kernels for benchmarking stream processing data analysis activity at the Edge layer.
- None, by themselves, can reveal the true bottleneck of whole IoT application graph,
- Possible research directions
  - To identify/build different suitable benchmarks from each type of the data analysis activities and hierarchically/logically combine them to draw accurate conclusions across an IoT graph in a holistic way.
Dockerizing Benchmark Kernel: Our Approach

Linear Road Kernel

Producer/Consumer/Processing services

- Position report
  - Car.dat
  - Raw data

- Accident detection
  - Detect which car has a speed of 0 from the last 30 seconds

- Segment reports
  - Finds any cars near where the accident took place

- Accident Notification
  - Broadcast to the drivers that are near the accident

Dockerizing Linear Road Kernel

Raw data
- car.dat
- car.dat.tolls.dat

Kafka producers
- PositionReportStream “POS”
- AccountBalanceRequestStream “BALANCE”
- DailyExpenditureRequestStream “DAILYEXP”
- TollHistoryRequestStream “TOLL_HIST_TABLE”

Kafka functions
- AccidentDetectionStream
- LatestAverageVelocityStream
- NumberOfVehiclesStream
- SegmentCrossingPositionReportStream
- AccidentNotificationStream
- TollNotificationStream
- CurrentTollStream
- CurrentExpenditurePerVehicleTable
- ACCOUNT_BALANCE_RESP.csv
- ACCOUNT_BALANCE_RESP.csv
- DAILY_EXP_RESP.csv
- TOLL_NOT.csv
- DAILY_EXP_RESP.csv

ACC_NOT.csv

ACCOUNT_BALANCE_RESP.csv
Testing results – Throughput
Microservices’ Performance Characterisation

- Challenges with setting up real-world benchmarking experiments in context of Osmotic Computing
  - Complexity and heterogeneity of end-point networks (e.g. WiFi, 4G, Bluetooth)
  - Heterogeneity of cloud, edge and IoT hardware resources and software stack
  - Mobility of IoT and edge devices
  - Complex interactions between the IoT and edge layers
  - Hard (if not) impossible to scale
Microservices’ Performance Characterisation: Osmotic Simulator

Simulator release site:
https://rajivranjan.net/iotsim/iotsim-release/
IoTSim-Edge Case Study
- Capacity Planning for Road Side Units (RSUs)
IoTSim-Edge Case Study  
- Capacity Planning for Road Side Units (RSUs)

<table>
<thead>
<tr>
<th><strong>IoT device</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Current location</td>
<td>0, 0, 0</td>
</tr>
<tr>
<td>IoT type</td>
<td>car</td>
</tr>
<tr>
<td>Movable</td>
<td>true</td>
</tr>
<tr>
<td>Data frequency</td>
<td>1</td>
</tr>
<tr>
<td>Data generation time</td>
<td>1</td>
</tr>
<tr>
<td>Network protocol</td>
<td>bluetooth</td>
</tr>
<tr>
<td>IoT Protocol</td>
<td>XMPP</td>
</tr>
<tr>
<td>Max battery capacity</td>
<td>70</td>
</tr>
<tr>
<td>Battery drainage rate</td>
<td>1</td>
</tr>
<tr>
<td>Number of entity</td>
<td>‘variable’</td>
</tr>
<tr>
<td>Velocity</td>
<td>0.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Edge device 1</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Raspberry Pi</td>
</tr>
<tr>
<td>Location</td>
<td>0, 0, 0</td>
</tr>
<tr>
<td>Movable</td>
<td>false</td>
</tr>
<tr>
<td>Signal range</td>
<td>25+25</td>
</tr>
<tr>
<td>Max IoT device capacity</td>
<td>10000</td>
</tr>
<tr>
<td>Max battery capacity</td>
<td>200000 units</td>
</tr>
<tr>
<td>MIPS</td>
<td>10000</td>
</tr>
<tr>
<td>RAM</td>
<td>10000</td>
</tr>
<tr>
<td>Bandwidth</td>
<td>10000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Microservices</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>1</td>
</tr>
<tr>
<td>MIPS</td>
<td>10000</td>
</tr>
<tr>
<td>RAM</td>
<td>10000</td>
</tr>
<tr>
<td>BW</td>
<td>10000</td>
</tr>
<tr>
<td>Shrinking factor</td>
<td>‘variable’</td>
</tr>
<tr>
<td>Network protocol</td>
<td>bluetooth</td>
</tr>
<tr>
<td>Uplink</td>
<td>-</td>
</tr>
<tr>
<td>Downlink</td>
<td>2</td>
</tr>
</tbody>
</table>

Simulation Configuration for IoT device, Edge device and Microservices
IoTSim-Edge Case Study
- Capacity Planning for Road Side Units (RSUs)

(a) Average execution time for each edgelet

(b) Average energy consumption by the edge devices
Osmotic Computing Challenges

**Microservice deployment:**

- How to map microservices to Edge, IoT, and Cloud layers based on non-functional requirements?
- How to solve the NP-hard deployment problem?
- To this end, novel decision-making techniques based on multi-criteria optimization and multi-criteria decision making techniques should be investigated.

![Diagram of Non-functional Requirements](Image)

*Fig: Non-functional Requirements*
Osmotic Computing Challenges

**Microservice Networking**
- Limitation of Software Defined Networking (SDN) in context of IoT includes:
  - having a centralized controller
  - connecting millions of IoT devices to a centralised controller is not scalable
- One of the important research direction will be to
  - first subdivide the controlling layer to create hierarchy of controllers
  - develop algorithms for coordinating bandwidth allocation across controllers

![Diagram showing SDN architecture with controllers and routers](image-url)
Microservice Contention

- The co-deployed microservices on Cloud or Edge datacenters can lead to contention problems which will affect QoS.
- Research in Osmotic Computing should be focus on novel microservice consolidation techniques that can dynamically detect and resolve resource contention.
Osmotic Computing Challenges

Microservice Monitoring

- How to monitor & debug IoT applications in real-time?
- How to effectively analyze collected monitoring data to detect root causes of QoS violations and failures?
Osmotic Computing Challenges

**Microservice orchestration and elasticity control**

- How to holistically autoscale?
- How to replicate data to avoid data loss?
- How to load-balance based on resource and data flow?
Publications Related to Osmotic Computing


Building Blocks: IoT Sensors and Actuators

**SENSORS & ACTUATORS**

We are giving our world a digital nervous system. Location data using GPS sensors. Eyes and ears using cameras and microphones, along with sensory organs that can measure everything from temperature to pressure changes.
Building Blocks: Connectivity

2 CONNECTIVITY

These inputs are digitized and placed onto networks.
Building Blocks: Edge Computing

- Edge Gateway
- Raspberry Pi
- Arduino
- Dell Edge Gateway
- Parking Kiosks
- Weather Sensors
- EV Charging

Communication Protocols:
- ZigBee
- Bluetooth low energy
- Bluetooth
Smart World of Things in Connected “Cities”

Source: http://images.libelium.es/content/applications/libelium_smart_world_infographic_big.png