A Discrete Particle Swarm Optimization for IoT services placement over Fog infrastructures

PhD Student:
- Tanissia DJEMAI

Directed by:
- Patricia STOLF
- Thierry MONTEIL
- Jean-Marc PIERSON
Outline

Introduction
- Smart cities
- Internet of Things applications
- Large scale computing infrastructures

Problem formulation
- Fog hierarchical Infrastructures
- IoT applications graphs
- Objective function

Strategies
- CloudOnly
- FogOnly
- FogCloud
- IoTCloud
- DCT
- DPSO

Experimental approach & results
- Methodology
- Results

Conclusion
- Current Work
- Future prospects
Heterogeneity
Dynamicity
Users number
Energy greedy

25%
28%
47%
“Fog computing is a horizontal, physical or virtual resource paradigm that resides between smart end-devices and traditional cloud or data centers.” [NIST 2017]

Toplogy
We consider a Hierarchical Three-Layered Fog infrastructure represented with a non oriented graph $G_M = (\mathcal{M}, \mathcal{E})$. 
## IoT applications

<table>
<thead>
<tr>
<th>Class \ QoS</th>
<th>Delay-sensitivity</th>
<th>Bandwidth demand</th>
<th>Communication frequency</th>
<th>CPU demand</th>
<th>Data Location</th>
<th>Mobility</th>
<th>Priority</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interactive-Real Time</td>
<td>High &lt;50ms</td>
<td>High</td>
<td>High</td>
<td>High-Medium</td>
<td>Local-Vicinity</td>
<td>High-medium-low</td>
<td>1</td>
<td>Augmented reality games</td>
</tr>
<tr>
<td>Mission-critical</td>
<td>High &lt;20ms</td>
<td>High</td>
<td>High</td>
<td>High-Medium</td>
<td>Local-Vicinity</td>
<td>High-medium-low</td>
<td>0</td>
<td>EEG</td>
</tr>
<tr>
<td>Streaming</td>
<td>Medium &lt;150</td>
<td>High</td>
<td>Medium</td>
<td>Medium-Low</td>
<td>Local-Vicinity-Remote</td>
<td>High-medium-low</td>
<td>2</td>
<td>Videoconferencing, Camera surveillance</td>
</tr>
<tr>
<td>Best effort</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>Low</td>
<td>Remote</td>
<td>High-medium-Low</td>
<td>3</td>
<td>File sharing etc.</td>
</tr>
</tbody>
</table>

![Diagram of IoT applications]

**Introduction**

**Problem formulation**

**Strategies**

**Experimental approach & results**

**Conclusion**
Energy & delay violation

**Objective function**

\[
\forall k, t \in \mathbb{N} : \quad f = \min_{i \in [0, N-1], j \in [0, M-1]} [(1 + \lambda)E_f]
\]

\[
r_{am_i} \leq r_{am_j} \forall i \in [0, N-1], \forall j \in [0, M-1] \quad (i)
\]

\[
r_{cpu_i} \leq r_{cpu_j} \forall i \in [0, N-1], \forall j \in [0, M-1] \quad (ii)
\]

\[
\sum_{j \in [0, M-1]} y_{k}^{f}(i, j) = 1, \forall i \in [0, N-1] \quad (iii)
\]

- (i) and (ii) are respectively memory and computing constraints for placing service \(i\) on machine \(j\).
- (iii) means that a service \(s_i\) should be placed only in one device.
Discrete Particle Swarm Optimization approach

- Semi-stochastic population-based approach.
- Inspired by the collective behavior of social animals (Birds flocking, Fish schooling).
- A set of particles with a position, velocity and a set of neighbors exploring the multidimensional search space through iterations.
- The particle's movement (direction and speed) between each iteration is a consequence of its own experience (local search method) and its neighboring one (semi-global or global search).
1. Initialize all particles uniformly
2. Initialize velocities to 1
3. Evaluate fitness for each particle Xk
4. Update Personal best (pb)
5. Update ring neighbor best (nb)
6. Update velocity
7. Update particle position

Is max iterations reached?

No

START

Yes

STOP

1. Initialize all particles uniformly
2. Initialize velocities to 1
3. Evaluate fitness for each particle Xk
4. Update Personal best (pb)
5. Update ring neighbor best (nb)
6. Update velocity
7. Update particle position

Is max iterations reached?

No

START

Yes

STOP

1. Initialize all particles uniformly
2. Initialize velocities to 1
3. Evaluate fitness for each particle Xk
4. Update Personal best (pb)
5. Update ring neighbor best (nb)
6. Update velocity
7. Update particle position

Is max iterations reached?

No

START

Yes

STOP

1. Initialize all particles uniformly
2. Initialize velocities to 1
3. Evaluate fitness for each particle Xk
4. Update Personal best (pb)
5. Update ring neighbor best (nb)
6. Update velocity
7. Update particle position

Is max iterations reached?

No

START

Yes

STOP

1. Initialize all particles uniformly
2. Initialize velocities to 1
3. Evaluate fitness for each particle Xk
4. Update Personal best (pb)
5. Update ring neighbor best (nb)
6. Update velocity
7. Update particle position

Is max iterations reached?

No

START

Yes

STOP

1. Initialize all particles uniformly
2. Initialize velocities to 1
3. Evaluate fitness for each particle Xk
4. Update Personal best (pb)
5. Update ring neighbor best (nb)
6. Update velocity
7. Update particle position

Is max iterations reached?

No

START

Yes

STOP

1. Initialize all particles uniformly
2. Initialize velocities to 1
3. Evaluate fitness for each particle Xk
4. Update Personal best (pb)
5. Update ring neighbor best (nb)
6. Update velocity
7. Update particle position

Is max iterations reached?

No

START

Yes

STOP

1. Initialize all particles uniformly
2. Initialize velocities to 1
3. Evaluate fitness for each particle Xk
4. Update Personal best (pb)
5. Update ring neighbor best (nb)
6. Update velocity
7. Update particle position

Is max iterations reached?

No

START

Yes

STOP

1. Initialize all particles uniformly
2. Initialize velocities to 1
3. Evaluate fitness for each particle Xk
4. Update Personal best (pb)
5. Update ring neighbor best (nb)
6. Update velocity
7. Update particle position

Is max iterations reached?
CloudOnly
IoTFogOnly
IoTCloud(IC)
FogCloud(FC)
Dicothomous (DCT)
Discret Particle Swarm Optimization (DPSO)
Experimental approach & results

Introduction

Problem formulation

Strategies

(1) Real Time (RT)

(2) Mission Critical (MC)

(3) Streamin (ST)

(4) Best Effort (BE)

Conclusion
Experimental approach & results
<table>
<thead>
<tr>
<th>Layer</th>
<th>DPSO</th>
<th>BPSO</th>
<th>CloudOnly</th>
<th>FogOnly</th>
<th>DCT</th>
<th>FC</th>
<th>IC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud Layer</td>
<td>23.1</td>
<td>13.7</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0</td>
<td>50</td>
</tr>
<tr>
<td>Fog Layer</td>
<td>43.7</td>
<td>61.3</td>
<td>0.0</td>
<td>85.2</td>
<td>50</td>
<td>100.0</td>
<td>0</td>
</tr>
<tr>
<td>IoT Layer</td>
<td>33.2</td>
<td>25.0</td>
<td>0.0</td>
<td>14.8</td>
<td>50</td>
<td>0</td>
<td>50</td>
</tr>
</tbody>
</table>
Conclusion (1)

1. Evolutionary approach and basic placement strategies.
2. DPSO gives a good tradeoff between energy and delay values.
3. Execution time.
5. Hierarchical topology.
6. Linear energy consumption profile.
7. Static infrastructure and VMs.
Conclusion (2)

Experimental approach & results

Introduction
Problem formulation
Strategies
Conclusion (3)

Solution quality impacted by time.

Users mobility estimation

Efficient handover and migrations approches

Evaluation

Services availability.
BIBLIOGRAPHY


