Toward a Smart Platform for Data Stream Processing in Smart Cities: Distributed Qos-aware Scheduling in Storm

Valeria Cardellini, Vincenzo Grassi, Francesco Lo Presti, Matteo Nardelli
Data Stream Processing (DSP)

continuously process streams of data generated by multiple, distributed sources, to extract valuable information
DSP Applications

Possible (Centralised) Solutions

Introducing Amazon Kinesis
Managed Service for Real-Time Processing of Big Data

1. Flexible data input
2. Data failover snapshots
3. Real-time, fault-tolerant, scalable, in-stream processing
4. Guaranteed data delivery
DSP Applications over Fog Computing
Cloud Computing/Data Center Approach

- Centralized computation at large-scale data centers
- “Distant” to be accessed by mobile devices or urban sensors
- Assumption: enough bandwidth to collect the data
The Fog Computing Approach

“Distant” Cloud extended with small data centers at the edges of the network, “near” to the customers
Our Vision

- Provide an integrated platform that facilitates development, deploying and the run time management of smart data-stream services for urban environments that are valuable for their stakeholders
  - designers/developers
    - development and (pre-deployment) validation tools
  - providers
    - QoS assurances, effective provisioning, runtime adaptation
  - users/citizens
    - trustable services
Synopsis Middleware

- **Execution Environment**
  - Resource Abstraction/Virtualization
  - Stream Oriented Communication
  - Service Composition
    - Discovery
    - Selection
    - Composition

- **Qos Aware Run Time Management**
  - QoS Monitoring
  - Run time Adaptation
    - Service addition, removal, substitution, rewiring, ...
Storm Architecture

spout → bolt → bolt → bolt → bolt

worker → worker → worker → worker → worker → worker → worker → worker → worker

Supervisor → Supervisor → Supervisor → Supervisor

LAN

Nimbus sched

ZooKeeper
Distributed Scheduling in Storm

Urban Environment

CINI Annual Workshop on ICT for Smart Cities & Communities
Palermo 30 ottobre 2015
Storm Extension

**Storm**

Centralized scheduler (Nimbus)

Round-robin scheduling policy

Agnostic to network delays and other nodes' attributes

Not aware of environmental and stream rate changes

**Developed extension**

Distributed scheduler (Supervisor)

QoS-aware scheduling policy

Enhance the system with adaptation capabilities, introducing the MAPE model

- Poor performance in geographically distributed and highly dynamic environment
QoS-aware Scheduling in (distributed) Storm

Centralized Solution (Benchmark)

\[ \max_x w_r \left( \frac{R_{\text{max}} - R(\bar{x})}{R_{\text{max}} - R_{\text{min}}} \right) + w_a \left( \frac{\log A(\bar{x}) - \log A_{\text{min}}}{\log A_{\text{max}} - \log A_{\text{min}}} \right) \]

\[
\begin{align*}
C_i x_{i,u} & \quad C_u & \quad u & \quad V_{\text{res}} \\
\left. \begin{array}{c}
\forall i \in V_{\text{dsp}} \\
\forall u \in V_{\text{res}}
\end{array} \right\} \quad \begin{array}{c}
x_{i,u} = 1 \\
\end{array} \quad \left. \begin{array}{c}
i \in V_{\text{dsp}}
\end{array} \right\}
\end{align*}
\]

\[
\begin{align*}
x_{i,u} & = y_{(i,j),(u,v)} & \forall (i,j) \in E_{\text{dsp}}(u,v) & \forall V_{\text{res}}^i, V_{\text{res}}^j & \\
x_{i,v} & = y_{(i,j),(u,v)} & \forall (i,j) \in E_{\text{dsp}}(u,v) & \forall V_{\text{res}}^i, V_{\text{res}}^j & \\
x_{i,u} & \in \{0,1\} & \forall i \in V_{\text{dsp}}, u \in V_{\text{res}} & \\
y_{(i,u),(j,v)} & \in \{0,1\} & \forall (i,j) \in E_{\text{dsp}}(u,v) & \forall V_{\text{res}}^i, V_{\text{res}}^j & \\
\end{align*}
\]

Distributed Algorithm (Pietzuch et al.)

Operator placement algorithm

- represents the application as a system of springs
- minimizes the elastic energy of the system

CINI Annual Workshop on
ICT for Smart Cities & Communities
Palermo 30 ottobre 2015
Conclusions & Future Works

- Integrated platform for DSP applications development, deployment and run time management

- Middleware Approach: Extending existing frameworks/platform
  - Compatible with existing frameworks/applications
  - Distributed implementation
    - Heterogeneous environments
    - Include location, privacy concerns
    - QoS scheduling
    - Integrate SDN-capabilities

- Storm
  - Distributed scheduling
  - Adding stateful migration

- Mesos
  - Distributed implementation