Distribution and orchestration of real-time analytic workflows over heterogeneous distributed infrastructures

Elli Kartsakli - Senior Researcher
elli.kartsakli@bsc.es
Barcelona Supercomputing Center (BSC-CNS)
Challenges in edge and cloud computing ecosystems

- Distributed computing
- Heterogeneous networking
- Massive & disperse data
- Heterogeneous computing infrastructures

Cloud Continuum

Big Data Value Association (BDVA) Webinar
COMPSs: A framework for computation distribution

- Open-source framework for development and execution of applications over distributed infrastructures
  - [http://compss.bsc.es](http://compss.bsc.es)

- "Sequential programming with parallel execution"
- Agnostic of the underlying infrastructure
- Seamlessly integrated with distributed data store
Application Example – Object detection with COMPSs

[Diagram showing object detection, tracking, deduplication, and data model creation]
Application Example – COMPSs task based programming

```python
@task
def get_detected_objects(camera_socket):
    return tkDNN_detected_objects(camera_id)

@task(object_list=IN, tracked_objects=IN, returns=list)
def tracker(object_list, tracked_objects):
    return track(object_list, tracked_objects)

@task(object_list=COLLECTION_IN, returns=list)
def deduplicator(tracked_objects):
    return deduplicated_obj(tracked_objects)

@constraint(AppSoftware="aggregatorNode")
@task(deduplicated_objects=IN, dC_model = IN)
def create_data_model(deduplicated_obj):
    snapshot = dC_model.Create_snapshot(deduplicated_obj)
    return snapshot
```

- Write sequential code
- Annotate tasks to be distributed with `@task` and identify their dependencies
- Mark any HW constraints with `@constraint`

```python
## Main function ##
while True:
    for i, socket in camera_sockets:
        obj_list = get_detected_objects(socket)
        tracked_obj[i] = tracker(obj_list, tracked_obj[i])
        deduplicated_obj = deduplicator(tracked_obj)
        snapshot = create_data_model(deduplicated_obj)
```

- Deduplication
- Object detection
- Object tracking
- Data Model creation

Big Data Value Association (BDVA) Webinar
Application example - Runtime

- COMPSs creates the *task dependency graph* (TDG)

  - TDG example with 1 s of processing from a single camera
  - In CLASS we will process live video streams of up to 16 cameras
The COMPSs master **deploys** the workers across the compute continuum and **distributes** the tasks at runtime, based on the **scheduling** policy.

- COMPSs also handles the transfer of data wherever needed for the computation.
Workflow Orchestration

- Scheduling decisions consider the real-time properties of the system.
- Offline profiling to measure *task execution times* and *data transfer times*.
- Different scheduling approaches have been implemented, aiming to minimize the response time:
  - MILP: optimal but computationally intensive.
  - Heuristics: suboptimal but tractable at runtime.
- The allocation of tasks to resources is static for each scheduling iteration:
  - *Response time upper bounds* can be provided.
  - *Re-allocation of resources* is triggered when changes are detected in the compute continuum or real-time properties are not met (e.g., task deadlines are missed).
Resource reallocation and scaling at runtime

- Rescheduling based on resource availability at runtime
- Dynamic offloading of tasks to the cloud through Rotterdam CaaS

Big Data Value Association (BDVA) Webinar
CLASS software architecture for workflow distribution and orchestration

✓ Distribution of complex analytics workflows over the compute continuum
✓ Transparent to programmer and agnostic to infrastructure
✓ Provision of real-time guarantees
✓ Reallocation of resources and scaling on-the-fly to meet the real-time requirements
Thank you!

www.class-project.eu
Twitter: @EU_CLASS
LinkedIn: linkedin.com/company/classproject