Container-based Support for Autonomic Data Stream Processing through the Fog

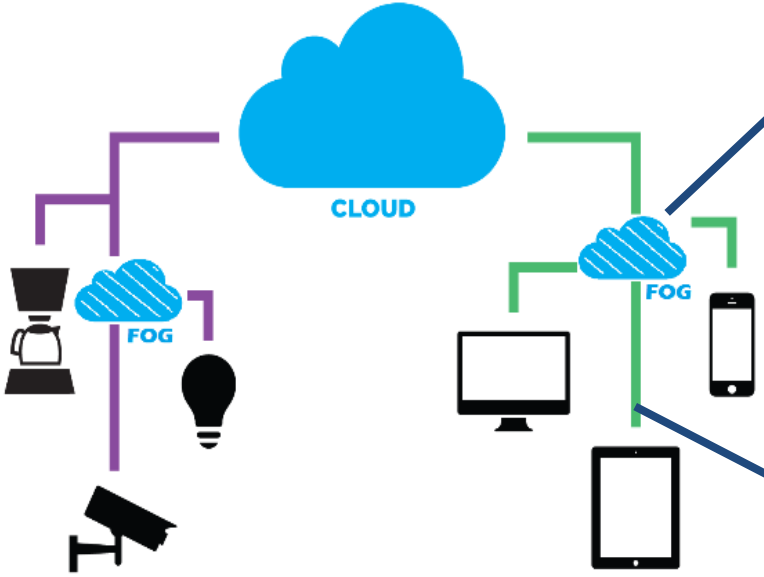
Antonio Brogi, Gabriele Mencagli, **Davide Neri,** Jacopo Soldani and Massimo Torquati
Department of Computer Science, University of Pisa, Italy
{brogi, mencagli, davide.neri, soldani, torquati}@di.unipi.it

Santiago De Compostela, 29 August 2017

Outline

- Problem & goal
- Motivating examples
- Container-based architecture
- Preliminary results
- Conclusions

Problem



Autonomic data stream parallel applications on Fog infrastructure.

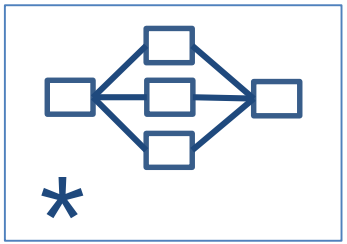
Solutions for **dynamic management** of resources within and across Fog nodes.

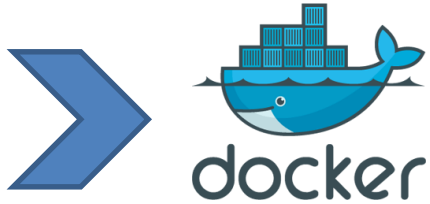
Intensive data flows with latency and/or bandwidth requirements.

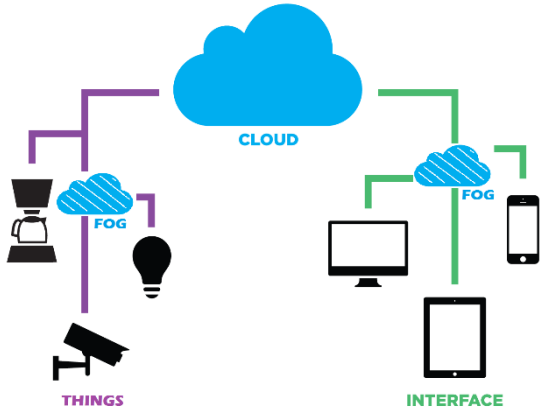
Goal

Container-based architecture for supporting autonomic data stream processing applications on Fog infrastructure.

Autonomic Data Stream Parallel Applications.







(* is the autonomic control logic)

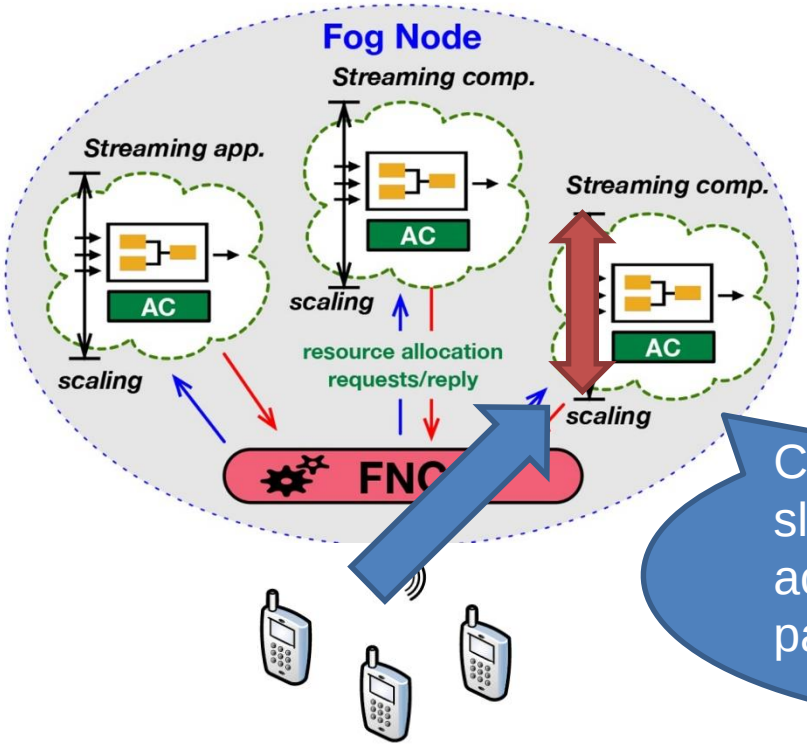
Motivating examples

Two examples motivating the development of our infrastructure:

1. **Intra-fog node scenario**: management of the resources of a Fog node (e.g. memory, CPU) assigned to applications.

2. **Inter-fog node scenario**: management of applications among different Fog nodes (e.g. migration of an application).

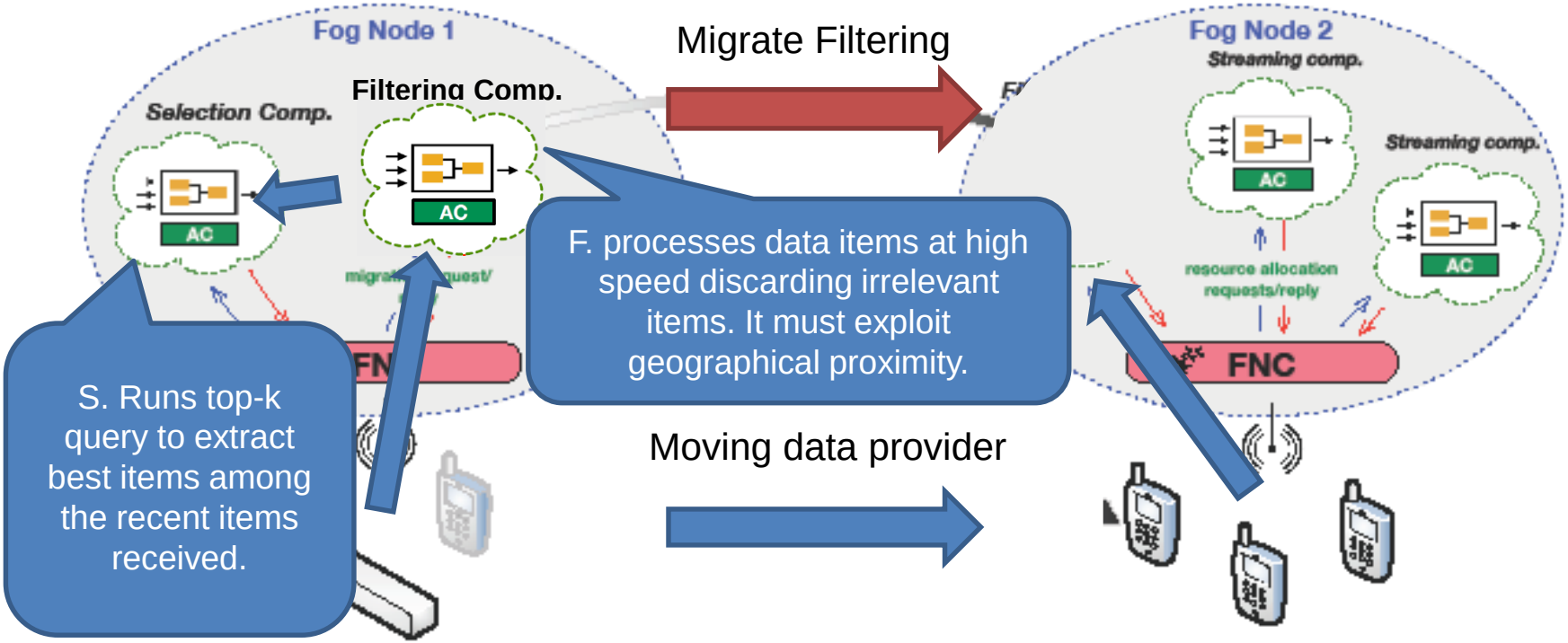
Intra-fog scenario: resources of a fog node



Arrival rate increase => the autonomic control logic may ask to the architecture to **increase** the concurrency level of the component to process input data faster.

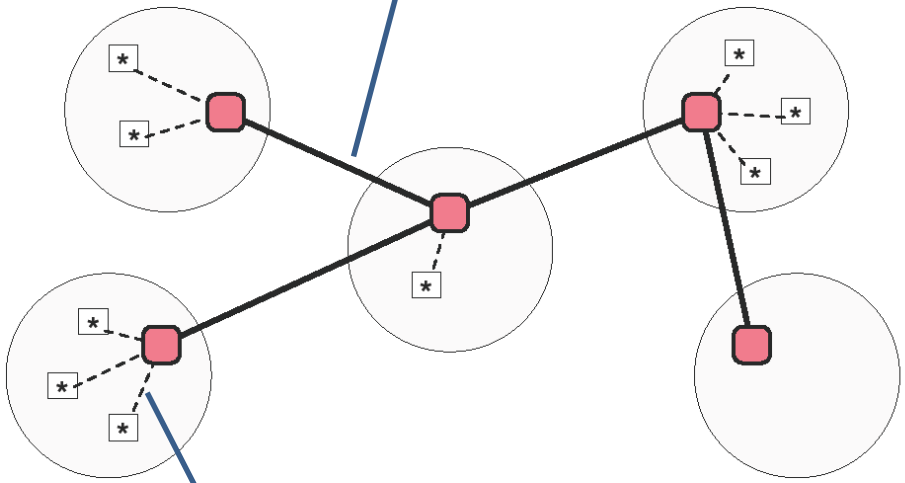
Component runs sliding-window model according to a feasible parallel pattern.

Inter-fog scenario: migration



System architecture

FNC-FNC inter-node communication (e.g., overlay network)



FNC-AC intra-node communication (e.g., socket)

Fog Node (**FN**)

FN Controller (**FNC**)

Application (**App**)

App. Controller (**AC**)

FNC-FNC connections

FNC-AC connections

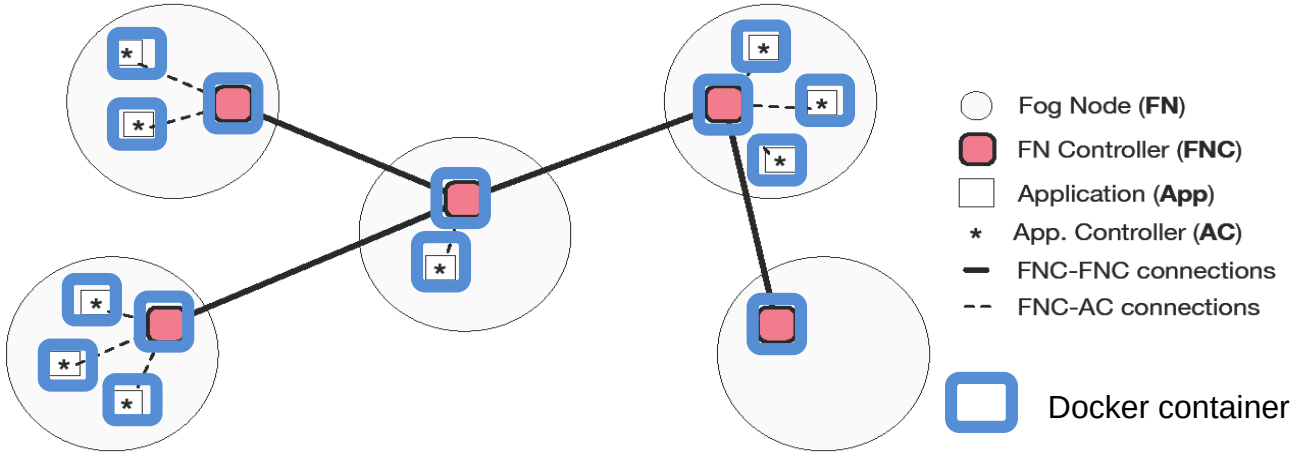
FN: Devices with limited resources running Parallel Apps.

FNC: assigns resources to Apps and schedule Apps among FNs.

App: stream data applications

AC: Autonomic control loop of App that interacts with the FNC to scale up/down resources (e.g. memory, CPU).

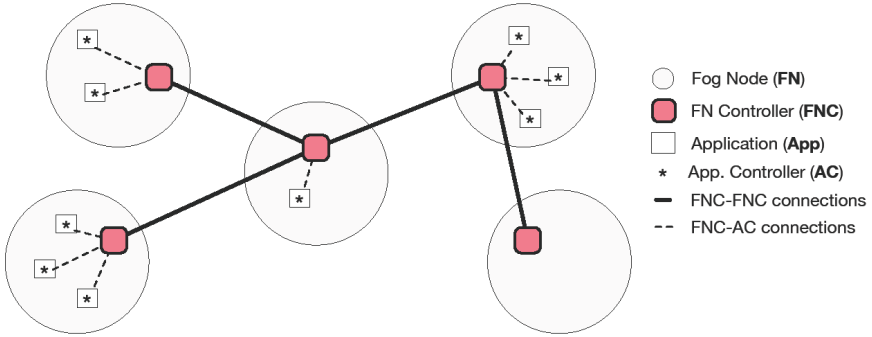
Container-based architecture



Containers and Docker benefits (some):

- **Isolates** parallel *Applications* running in a Fog node.
- **Meters** the resources (e.g., memory, CPUs) assigned to a container (`$ docker update --cpuset-cpus 0,1 ubuntu`)
- *Provides* ***Checkpoint*** and ***Restore*** mechanisms of a running container.

System architecture: fog node join/detach



Fog node **joins** the architecture

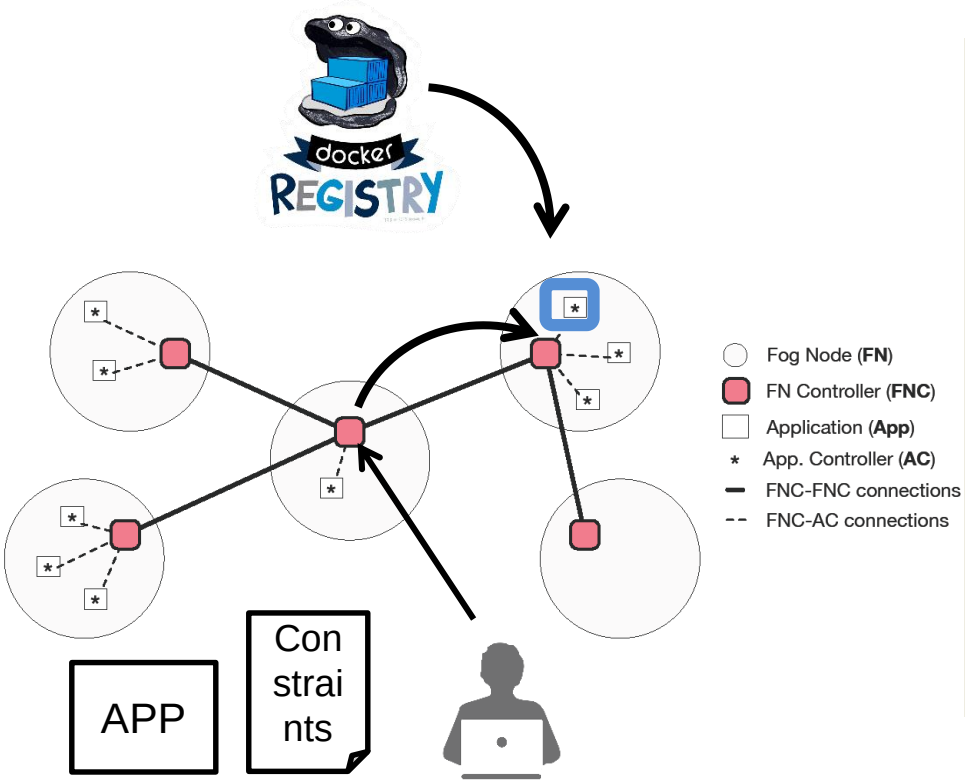
1. The FNC of the new FN connects to one or more existing FNCs.
2. FNC communicates the resources of the new FN.
3. The resources information of the new FN are sent to the other FNCs.

Fog Node **detaches** from the architecture[2]:

1. FNC communicates that the FN is going to detach to the others FNCs.
2. FNCs remove the FN from their view.
3. The Apps of the detached node are migrated to another FN.

[2] The availability of FN must be monitored for detecting unexpected detach (e.g., heartbeat)

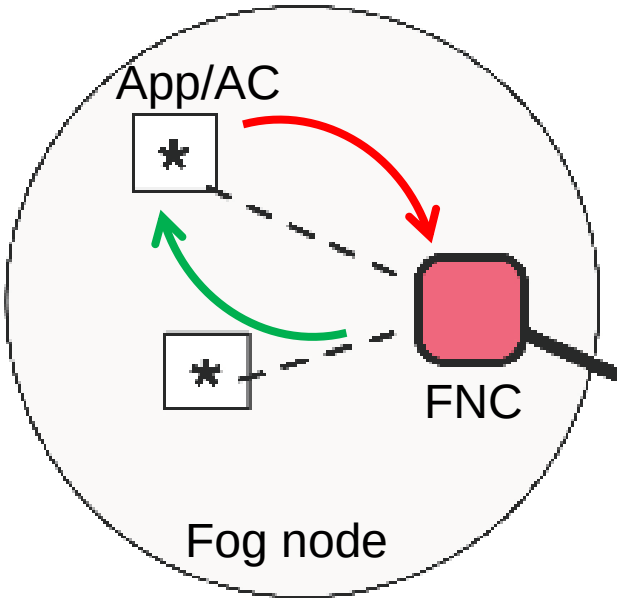
System architecture: deploy an App



Step for **deploying** an App:

1. User connects to one FNC indicating the App to be deployed and the deployment constraints.
2. FNC identifies the FN that satisfy the deployment constraints.
3. FNC of the selected FN (i) downloads the Docker images, (ii) assigns the initial resources, and (iii) starts the App.
4. FNC interacts with the AC to scale the resources (when necessary)

System architecture: resources management

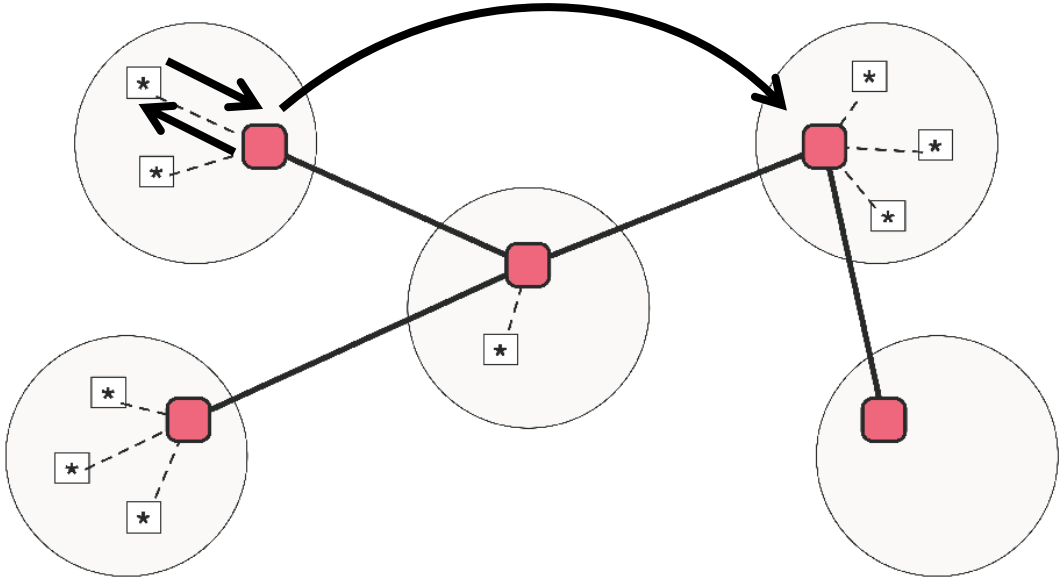


FNC scale up/down the resources assigned to an App (e.g., CPU, memory) by limiting the Docker container resources.

Two policies:

- **Reactive:** AC realizes that app needs more resources and sends a request to the FNC.
- **Predictive**: FNC needs to remove some of the resources assigned to an app.

System architecture: migrate an App



Steps for **migrating** an App:

1. FNC sends a migration request to the AC of the App to be migrated.
2. AC stores the current state of the App (if any) and sends *migration reply*
3. FNC migrate the app and the state on the new FN.

Preliminary results

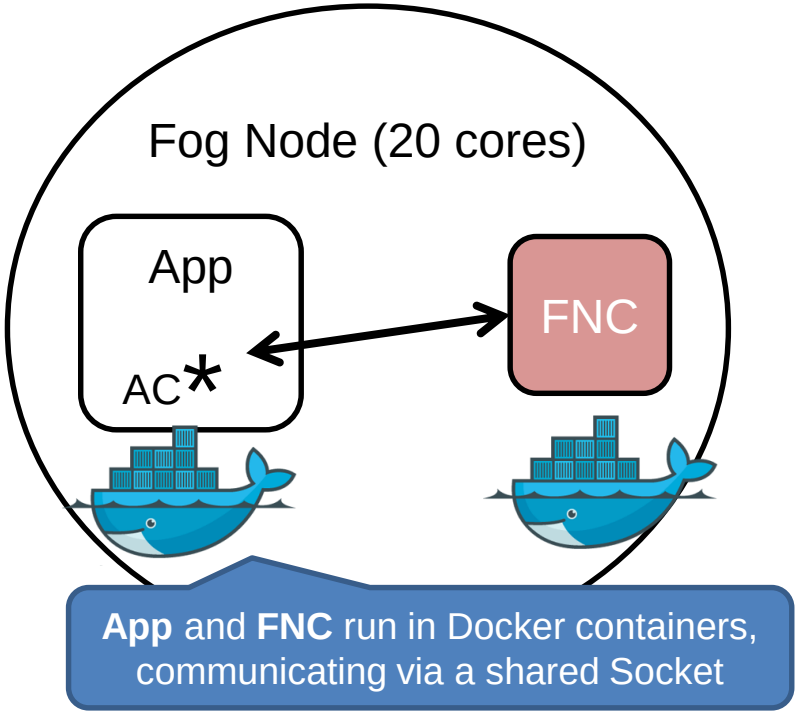
Feasibility of using Docker:

1. **Intra-fog:** time to increase/decrease the CPUs to a container.
2. **Inter-fog:** time to migrate a container.

Intra-fog scenario

The experiment:

- **Fog Node:** machine with 20 cores.
- **App:** consumes the cores of the fog node using the *cpuburn*[1] tool:
  - Every 5 secs AC asks to the FNC to increase/decrease the number of cores assigned to App.
- **FNC**: waits for incoming requests from AC and increases or decreases the assigned cores to the App.

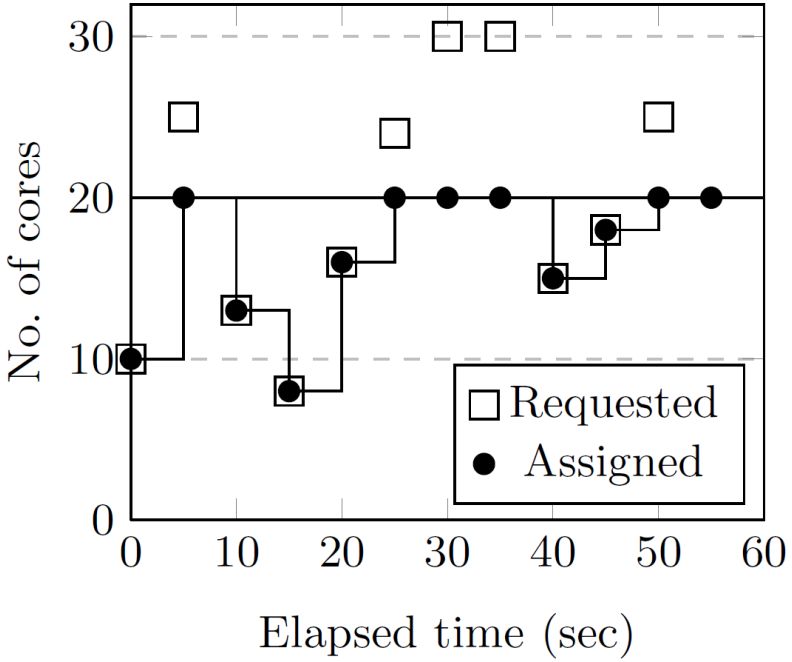


[1] https://patrickmn.com/projects/cpuburn/

Intra-fog scenario (cont.)

Results:

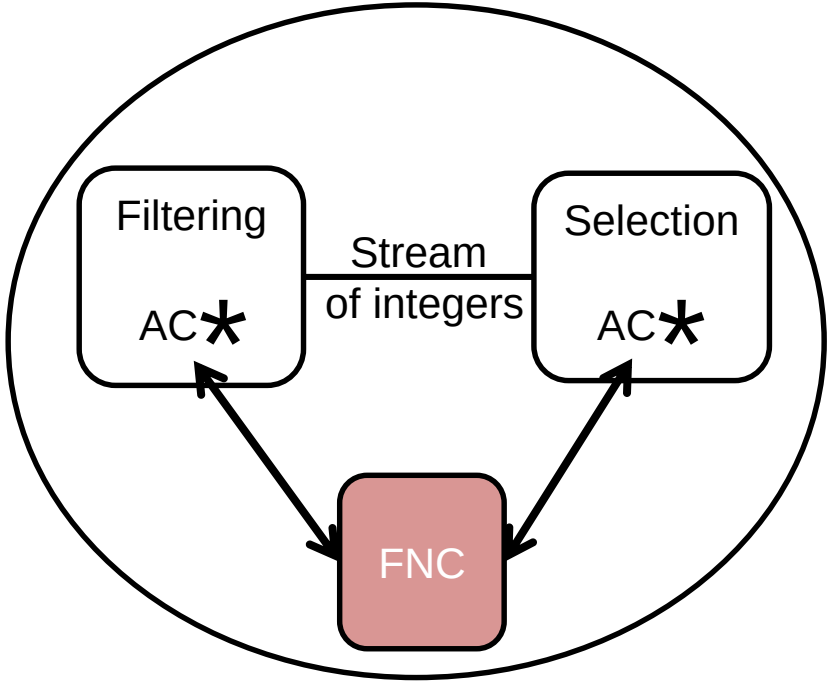
- ~ 80 **µs** (std 16 **µs**): time required by the FNC to assign the requested cores to App.
- *Socket* file communication on the same Fog node is feasible.
- Docker permits limiting the resources assigned to parallel applications.



Inter-fog scenario

Steps of the test:

- *Filtering* produces a stream of data (100 integers every sec)
- *Selection* receives and prints the data.
- After 5 sec *FNC* sends to the *Filtering* a *migration request*
- *Filtering* received the request, performs a clean-up phase, sends *migration reply* to FNC.
- FNC receives *migration reply* and perform a checkpoint of the *Filtering*
- Immediately after, FNC restores the *Filtering component*

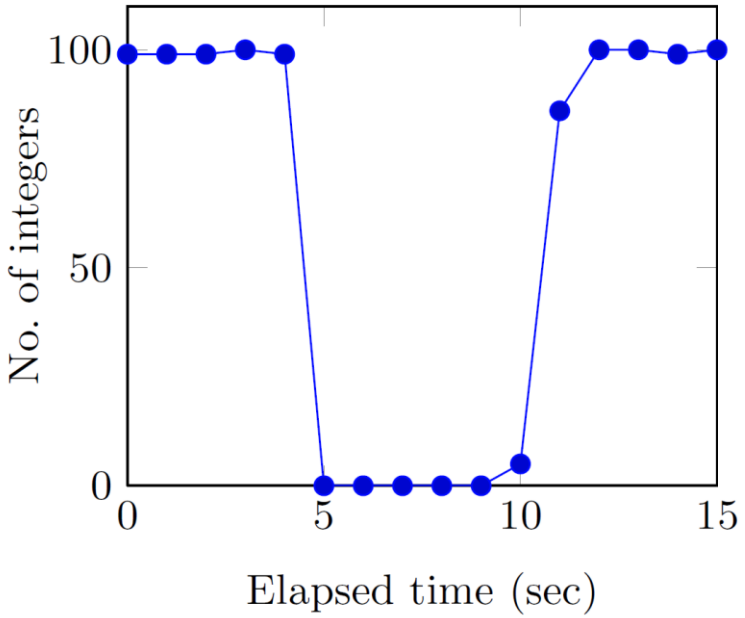


Inter-fog scenario (cont.)

Results:

- ~5 **secs** downtime experienced by the *Selection* component.

- Time to *checkpoint* and *restore* a container is quite high[1].



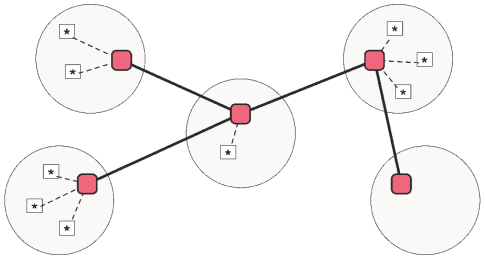
[1]Docker checkpoint and restore are under development and we expect to see further optimization in next releases

Conclusions

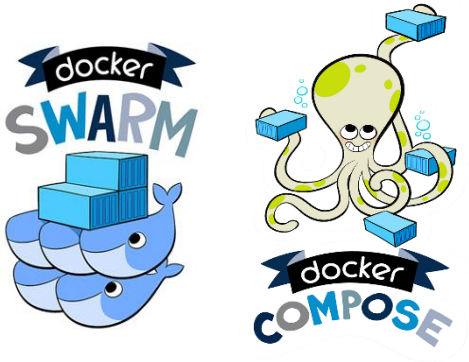
- Data Stream processing and Fog computing should still be explored and analysed.
- Container-based technology can be exploited for deploying parallel streaming applications on Fog.

- We propose a Docker-based architecture for deploying autonomic application in Fog infrastructure.
- Preliminary results show that Docker is a viable approach for fog-oriented framework.

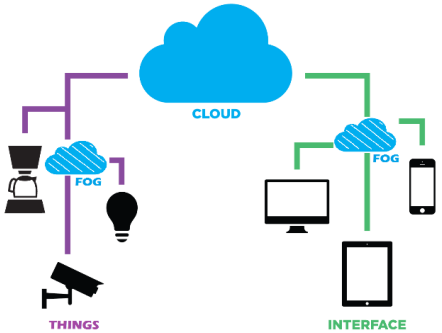
Future work



Implement the architecture



Exploit Docker platform



Run real parallel apps

Thank You

davide.neri@di.unipi.it

Q&A

GitHub - https://github.com/di-unipi-socc/ffdocker