IEEE International Workshop on Computer Aided Modeling and Design of Communication Links and Networks
September 14-16, 2020 // Virtual Conference

Data Stream Processing in Software Defined Networks: Perspectives and Challenges

Alessandra Fais, Gregorio Procissi, Stefano Giordano, Francesco Oppedisano

Presenter: Alessandra Fais
PhD Student in Information Engineering
Università di Pisa
Email: alessandra.fais@phd.unipi.it

Outline

- Introduction: the current scenario
  - Modern (*softwarized*) networks
  - Network *flexibility* and *programmability*
- Network *scalability* and *performance*
  - Motivations: why these aspects are so crucial
  - Proposed strategies for network applications design
    - Two-layer processing model
    - Acceleration through Data Stream Processing
- Conclusions

Introduction: the current scenario

Softwarized network context:

- Evolution of **S**oftware **D**efined **N**etworking (**SDN**) and **N**etwork **F**unction **V**irtualization (**NFV**)
- Fully programmable general-purpose network nodes
- Separation of control and data planes
- Software network applications
  - Instantiated through the controller
  - Computation burden depends on the use case

Network flexibility and programmability

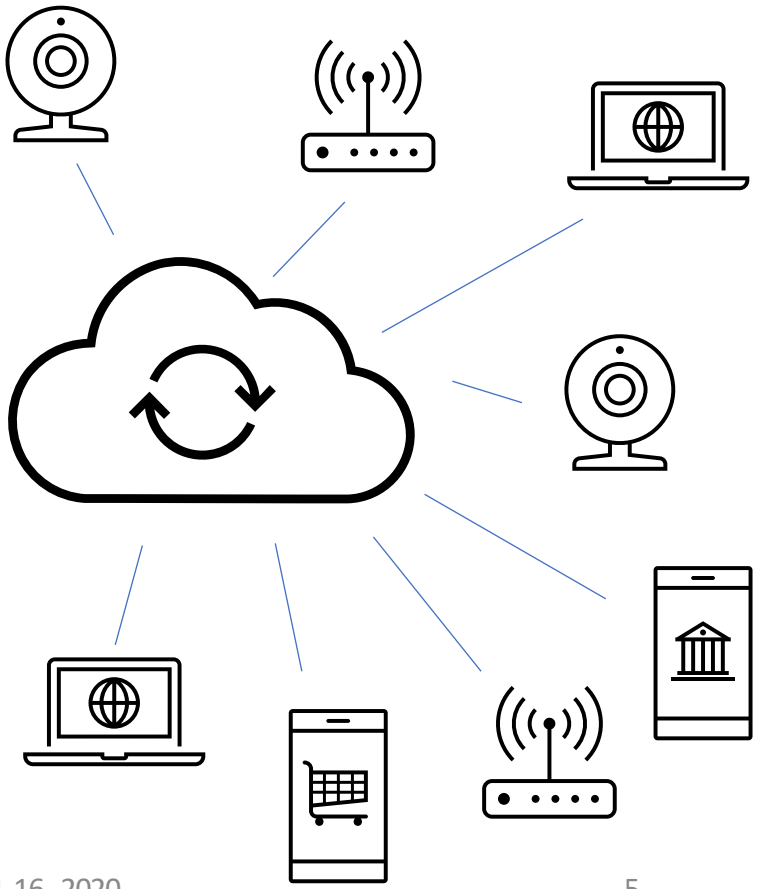
- **Programming abstractions** to allow the controller to instruct the data plane processing nodes
- Early **data processing at network line rate** in the data plane
  - Efficient capturing engines for fast packet handling
- Perform possibly complex computations in the control plane (e.g., DPI, SIEM)
  - Efficient design of network applications

DPI = Deep Packet Inspection
SIEM = Security Information and Event Management

Network applications scalability and performance (I)

WHY?

- Increasing number of connected devices
- Enormous amount of data collected from the network to be processed (*Big-Data*)
  - Network monitoring
  - Security



Network applications scalability and performance (II)

HOW?

- Design a **two-layer processing model**
  - Controller workload reduction
- **Accelerate data processing** performed within network applications
  - Data stream processing computational model

Important assumption: programmable nature of network devices (multicore CPUs, FPGA and/or GPU co-processors)

Multi-layer Processing Model

Network applications
- Software implementations of network functionalities
- Run on top of the controller

- Scalability is strongly related to the controller architecture
  - Logically centralized entity
  - May become a scalability bottleneck
- Reduction of the controller workload
  - Distributed architecture
  - Parallelization of each controller instance

Multi-layer Processing Model

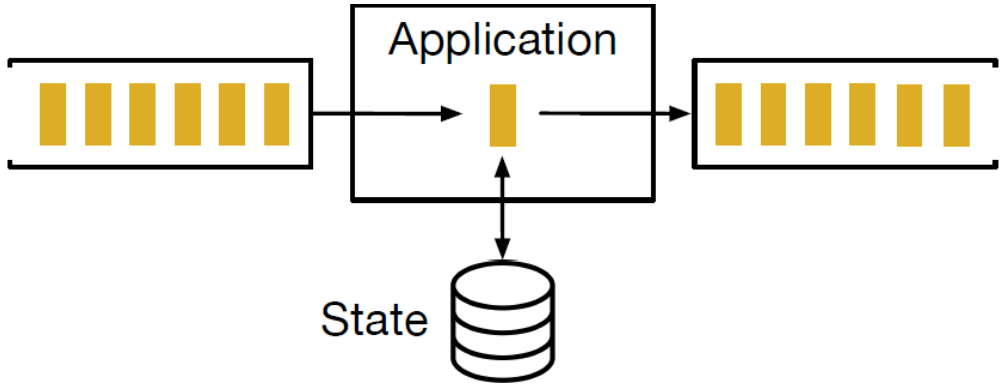
Distribute part of the computation in the network nodes

- Data plane level
  - First stage of coarse-grained processing
  - Data: directly collected from the network in the form of streams (e.g., packets)
- Control plane level
  - Second stage of fine-grained processing
  - Data: partial results processed in the first stage (e.g., statistics)

Data Stream Processing (DaSP)

Network applications
- Collect data in the form of **streams**
- Strict performance requirements (real-time processing)

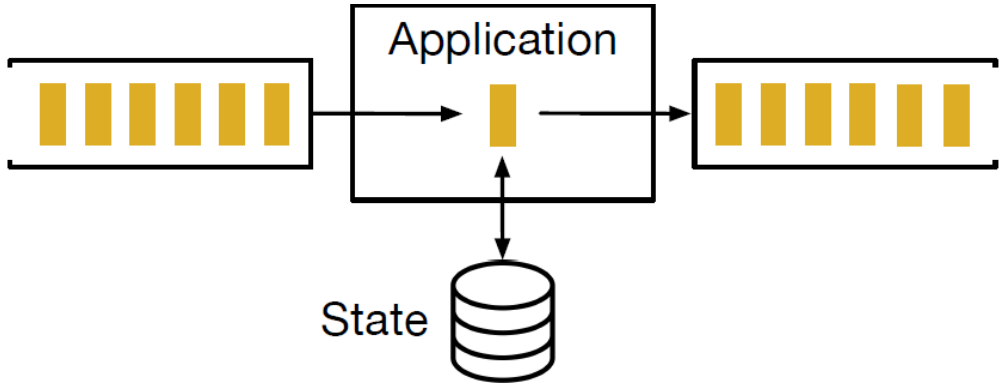
- Adopt **Data Stream Processing**
  - Accelerate both layers
    - DaSP at both data and control planes
  - Exploit parallelism
    - Achieve *high throughput* and *low latency*



Data Stream Processing (DaSP)

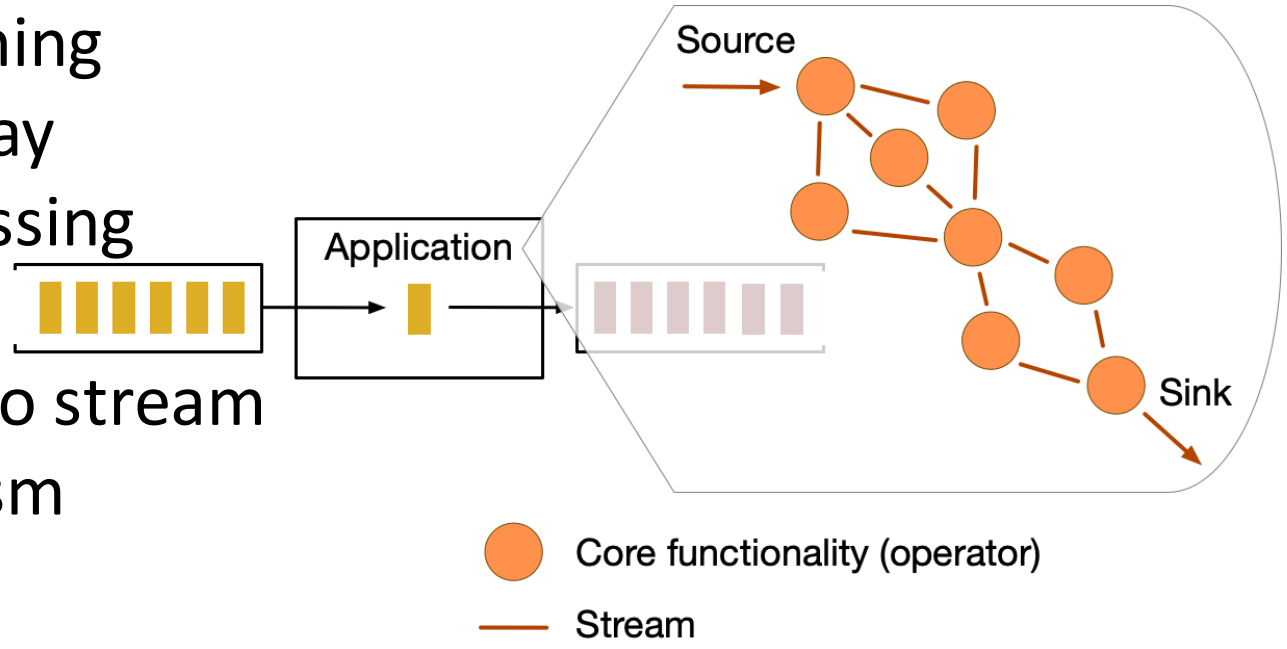
- Several complexities related to stream processing
  - **Infinite sequences** of data items (impossible to store all the data!)
  - Continuous arrivals at very **high speed**
  - **Irregular** and time-varying input **rates**
  - **Significance** of data decays over time

Computation must be performed *on-the-fly*



Data Stream Processing Frameworks

- **Proper abstractions**
  - Implement efficient streaming applications in an easier way
  - Guarantee real-time processing through parallelism
  - Hide complexities related to stream management and parallelism
- **Data-flow graph model**



Data Stream Processing Frameworks

- **State-of-the-art frameworks**
  - Java-based solutions
  - Target distributed systems (clusters of homogeneous machines)

Apache Flink™

→ Control plane layer of network applications

- **Other solutions**

WINDFLOW

  - C++-17 library
  - Target single shared memory systems (multicore CPUs + GPUs)

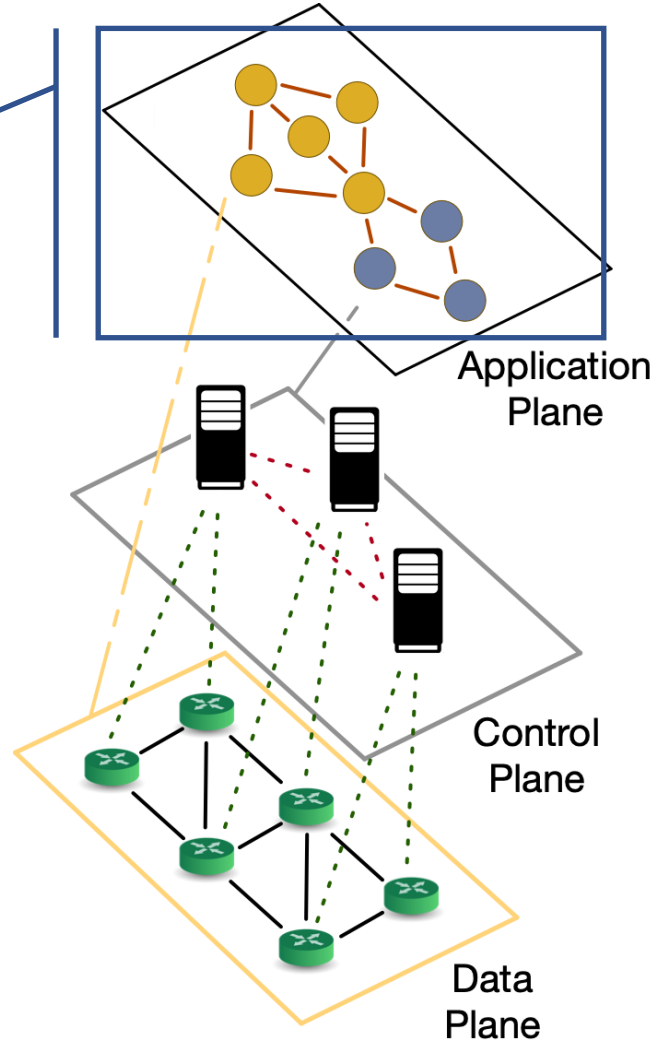
→ Both data and control plane layer of network applications

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CROSSLAB
Innovation for industry 4.0

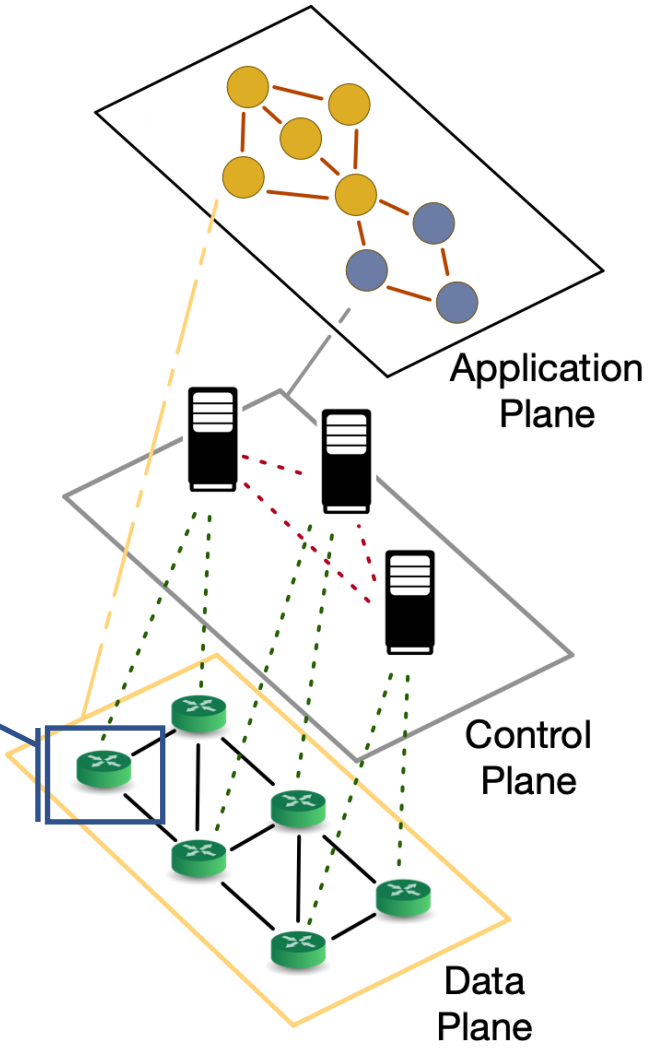
Multi-layer DaSP-based network applications design

- Application modeled as *data flow graph*
- Optimal deployment of *core functionalities*
  - On a given network device
  - On a given controller instance
  - On multiple distributed controller instances



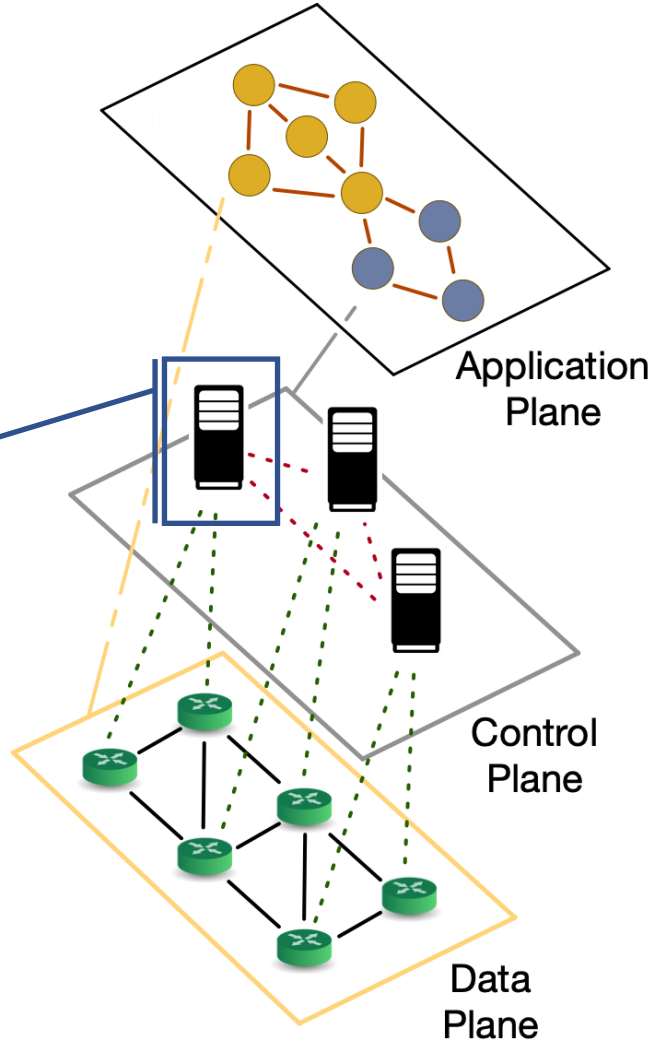
Multi-layer DaSP-based network applications design

- Application modeled as *data flow graph*
- Optimal deployment of *core functionalities*
  - On a given network device
  - On a given controller instance
  - On multiple distributed controller instances



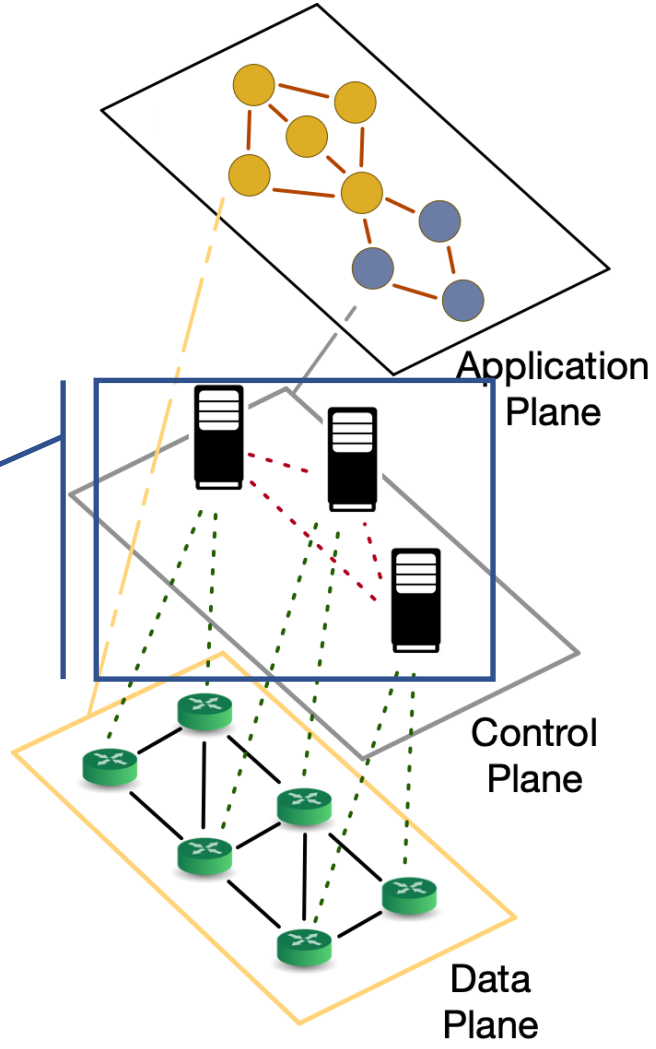
Multi-layer DaSP-based network applications design

- Application modeled as *data flow graph*
- Optimal deployment of *core functionalities*
  - On a given network device
  - On a given controller instance
  - On multiple distributed controller instances



Multi-layer DaSP-based network applications design

- Application modeled as *data flow graph*
- Optimal deployment of *core functionalities*
  - On a given network device
  - On a given controller instance
  - On multiple distributed controller instances



Conclusions

Performance issues of network applications in SDN scenarios are explored

- Improve **scalability**
  - Design network applications following a **new hierarchical scheme**
- Improve **performance**
  - Adopt **Data Stream Processing** computational model
    - Performance increased through parallelism
    - Exploit multicore architectures and possibly co-processors

Future developments: implementation and results (for both data and control planes), operators' deployment, …