Data Stream Processing and Analytics

Spring Semester 2019

Course Info

Vasiliki (Vasia) Kalavri
kalavriv@inf.ethz.ch
Announcements and course updates

• Course Website and Moodle
  • https://www.systems.ethz.ch/courses/spring2019/dspa/
  • Slides and all other material will be available there
  • Assignments will be submitted in Moodle: https://moodle-app2.let.ethz.ch/course/view.php?id=10535

• Contact
  • Vasiliki Kalavri kalavriv@inf.ethz.ch
  • Zaheer Chothia (TA): zchothia@inf.ethz.ch
  • Please include DSPA-2019 in the e-mail subject

• Announcements will be sent to you by e-mail and published at the website
What is this course about?

**Low-latency** analysis of **continuous**, **distributed**, **rapid** data events.

Stream processor:
- in-memory storage
- Standing queries
- ad-hoc queries
- Output streams
- external stable storage
Example streams and applications

Sensor measurements
- anomaly detection, incident risk calculation

Financial transactions
- fraud detection, stock trading

Location and traffic data
- report train system status, find optimal routes

Web logs
- online recommendations, personalization

Network packets
- intrusion detection, load balancing

Online social interactions
- trending topics, sentiment analysis
Topics

• Systems
  • Architecture and design
  • Scalability and elasticity
  • Fault-tolerance and processing guarantees
  • State management

• Algorithms
  • Windowing semantics and optimizations
  • Basic data stream mining
  • Complex event processing

• Streaming applications and use-cases
Tools

Apache Flink: flink.apache.org

Apache Kafka: kafka.apache.org

Apache Beam: beam.apache.org

Google Cloud Platform: cloud.google.com

Timely Dataflow: github.com/TimelyDataflow/timely-dataflow
Outcomes

At the end of the course, you will hopefully:

• know **when** to use stream processing and **how** to use the available tools, their features and guarantees

• have a solid understanding of how **stream processing systems work** and what affects their performance

• be aware of **problems** you might face when building streaming pipelines, **trade-offs** you need to consider, and **tricky bits** of streaming apps
Scope

• **Theory and practice, old & new**
  • the first stream processors were invented almost 15+ years ago
  • virtually none of those systems is used today
  • many of their concepts and ideas are, often *rebranded*
  • modern stream processors have to deal with new challenges
Course Structure

1. Lectures
   - Mondays 10-12
   - ~15’-20’ discussion on topics of previous week
   - Introduction of new topics
   - Guest lectures from stream processing experts

2. Exercise Sessions
   - Mondays 13-15
   - Seminar-style: review and discuss research papers
   - Hands-on: analyze streaming data, use and compare streaming tools

3. Semester Project
   - In teams of 2 students
   - Implement, test, and evaluate an end-to-end stream processing application
   - Choose tools and design architecture as you like
Semester Project

1. Choose your teammate (or let them choose you!)
   - let me know if you cannot find a teammate and I will match you

2. Choose your system
   1. Apache Flink: Java, high-level API, component-heavy, fast
      (Exercise Session #1 - Feb 18)
   2. Timely Dataflow: Rust, low-level API, lightweight, super-fast
      (Exercise Session #2 - Feb 25)

3. Send me an e-mail with subject “DSPA Semester Project 2019” and the above information to register your project by 01-Mar-2019
Semester Project

- Development on gitlab (https://gitlab.inf.ethz.ch)

- Deliverables
  - code with comments
  - documentation
  - tests
  - written reports

- Milestones
  - Design document [18-March-2019]
  - Midterm progress report [29-Apr-2019]
  - Final report [1-June-2019]
Milestone 1: Design document

One per team, max 2 pages, 18-Mar-2019

Briefly describe what you are planning to build and how you are planning to build it.

Briefly argue why this design solves the problem and describe its advantages.

Answer the following:

- Where does your application read input from? How does this choice affect latency and semantics (order)?

- Where does your application output results? How are these results shown to the user?

- What is your stream processor of choice? What features guided your choice?

- Will you use other auxiliary systems / tools, e.g. for intermediate data?
Milestone 2: Midterm report

One per team, max 2 pages, 29-April-2019

Briefly describe:

- your progress so far

- any divergence from your original plan

- challenges and issues faced and how you solved them or planning to solve them

- outstanding tasks and timeline for completion
Milestone 3: Final report

One per student, max 4 pages, 01-June-2019

Briefly describe:

- high-level architecture and functionality of your application

- usage instructions, i.e. how can I use what you built? What result shall I expect? How can I run tests to verify correctness?

- your individual contribution, i.e. how did you and your partner split the work?

- reflect on what you would do differently if you could build this all over again

- mention anything you found fun, interesting, surprising while working on this project
Grading Scheme

- No Exam
- Participation in class: 10%
- Weekly assignments (reviews and hands-on): 50%
- Semester project (code and reports): 40%

Look out for this symbol!
You are here to learn, I am here to help you.

Ask questions!

Speak up if you don’t understand. I can explain it again :-)

Make suggestions, send me feedback.