COL733: Projects

The projects can be done in a group of up to 3 students. If you have an idea already, you may post your project idea on Piazza to recruit teammates.

Project categories

There are three types of projects that can be done:

Benchmarking study

This looks similar to the "Evaluation" section of the papers we have been discussing. Here you take two competing cloud systems of your choice and run benchmarks on them to compare their latency/consistency/availability/fault-tolerance/throughput, etc. You can also pick one system and evaluate it under different conditions.

In the project proposal, you need to describe:

- Which systems are you picking? Why are they competitors?
- What difference do you expect to see between the two systems? Why does this difference matters in the real world?
- What benchmarks will you run? How will these benchmarks quantify these differences?

Good projects may be posted as blogs on the course page (with students' permission). You can take inspiration from other blogs such as

1. Benchmarking Apache Kafka, Apache Pulsar, and RabbitMQ: Which is the Fastest? https://www.confluent.io/blog/kafka-fastest-messaging-system/
2. Apache Spark vs Apache Flink. https://www.ververica.com/blog/curious-case-broken-benchmark-revisiting-apache-flink-vs-databricks-runtime

Half-baked ideas:

- Implement some benchmarks like the WordCount labs on Flink and Ray and compare performance in presence of stragglers.
- Pick two key-value stores like Memcached vs Redis and compare performance.
- Study the behaviour of parameters through experimentation. Is there a trade-off? Can you develop guidelines on when to use what parameter values? You can come up with scenarios and try them: what happens if I vary discretization interval in D-Streams; what happens if I modify gossip protocol version update behaviour in Cassandra, how effective are Flink's unaligned checkpoints in straggler mitigation, etc.

Your research idea

This looks similar to the Benchmarking study but instead of comparing two commercial systems, you compare an existing system with your research prototype. Your research prototype can modify an existing system by modifying its source code or you can implement

it from scratch. You can choose to reimplement a published paper instead of a research idea of your own.

In the project proposal, you need to describe:

- Which baseline system are you picking?
- What does your research prototype hope to achieve?
- What difference do you expect to see between the two systems? Why does this difference matters in the real world?
- What benchmarks will you run? How will these benchmarks quantify these differences?

Compared to the benchmarking study, here the number of benchmarks you are expected to run is lesser. You can just run benchmarks that will evaluate the changes made by your research prototype.

Half-baked ideas:

- Add support for persistent memory in Spark to speed up checkpointing done for fault-tolerance.
- Build a testing methodology to verify that a given task is idempotent, determinstic, and stateless.
- Try to reimplement any of the studied systems from scratch. Discuss why they were hard to build.
- Survey build time optimizations done in Flink. Try to add new build time optimizations.
- Try to reimplement parts of existing papers of your choice. Some examples: Ray [OSDI'18], Pregel [Sigmod'11], Naiad [SOSP'13], Scanner [SIGGRAPH'18].
- Explore why fcall and stream commands are hard to add in redis raft. Try to add them and get them merged into redis.
- Survey reasons for non determinism in tensorflow. Try to use techniques such as model checking and equivalence checking to flag non deterministic programs.
- In serverless applications, idle time cost = 0. Explore implications of this on the design and analysis of parallel algorithms.

Your startup idea

Here the focus is to build an end-to-end non-trivial product of your choice. This product may support your startup idea. Or you may reimplement an existing product (like we roughly implemented trending topics on Twitter in Labs). This will likely use a number of cloud systems: a computing system, a storage system, a deployment/management approach, etc. In the project proposal, you need to describe:

- What product are you building? Why is it important/useful?
- Why is building this product challenging?
- What systems will you use?
- How will you evaluate your product?
    - The evaluation should be technical. How many concurrent users can you support, what is the turnaround time, etc.

Half-baked ideas:

- Build a peer-to-peer Instagram. Instead of sending a user's photo to a central server, the photos are added to a Dynamo-like peer-to-peer token ring. For security, a photo can only be viewed by knowing the user's encryption key.
- Build a collaborative editor like Google Doc using CRDTs.
- Build a scalable, responsive credit card payment processing system that internally manages a non-trivial state machine for every transaction. Example state machine transitions: basic authentication of credit card, check balance, ML based fraud detection, send OTP, block card, send transaction summary email, etc.
- Build a few Google earth engine workflows on top of Spark/Flink. Or try to unpack how workflows run in Google Earth Engine. Can it be improved in any way?

Deliverables- Proposal stage

After the minors week, the project group must hand in a project proposal. The proposal should contain the names and entry numbers of all the students in the project group. The proposal document shall be less than a page long; you can just succinctly answer the questions asked under project categories.

You can declare your group here. You can start working on your project right away after submitting your proposals. In most cases, the proposals will be accepted as is. If there are some comments, they will be communicated to you and you can course-correct your project execution.

Deliverables- Final

Towards the end of the semester, each group will give a project presentation to describe what they set out to do / what they ended up doing / major roadblocks / and final learnings. You can find your presentation slot here. There will be 8 minutes for the talk + 2 minutes for questions and answers. Each project member should talk about the work they've done. Some tips on communication.

The slides along with a brief project report should be combined together as a PDF and uploaded to Moodle. The combined PDF should contain the names and entry numbers of all the students in the project group and a link to your code (such as a GitHub repo). It should also report the work done individually by each group member.

Late policy

There is no strict deadline for proposals. The proposal submission deadline is only a suggestion. We advise you to submit as soon as possible so you can start working on your projects. This will also give you enough time to course-correct if required.

There will be no make-up presentations and no late submissions.