CSE 550: Systems for all

Au 2021

Ratul Mahajan

Distributed programming challenges

Suppose you have a program that takes 100 hours to run on a computer.

You want to run it faster by distributing work across multiple computers.

What challenges would you need to solve?

Parallelize the program

- Decompose into “units” of work and interfaces between them

Transfer data between application units

Balance load

- Hard even with homogeneous nodes; harder with heterogenous ones

Handle node and network failures

The whole enterprise becomes even more complicated if the infrastructure is shared by multiple programs

Goals of distributed programming frameworks

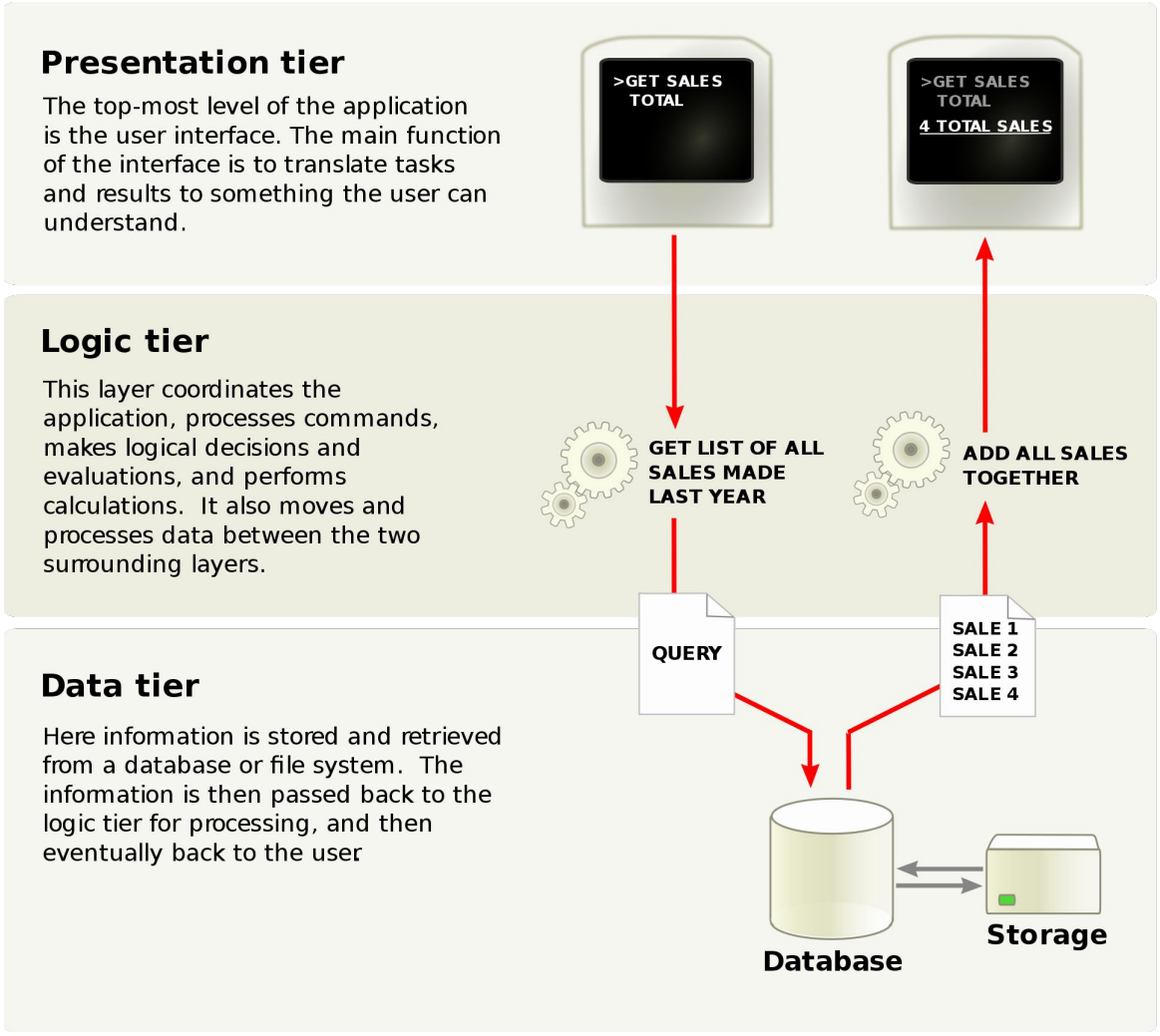
Solve (some of) these challenges in a re-usable manner

Provide acceptable performance (throughput, latency)

The catch: Need to express your program in a specific model

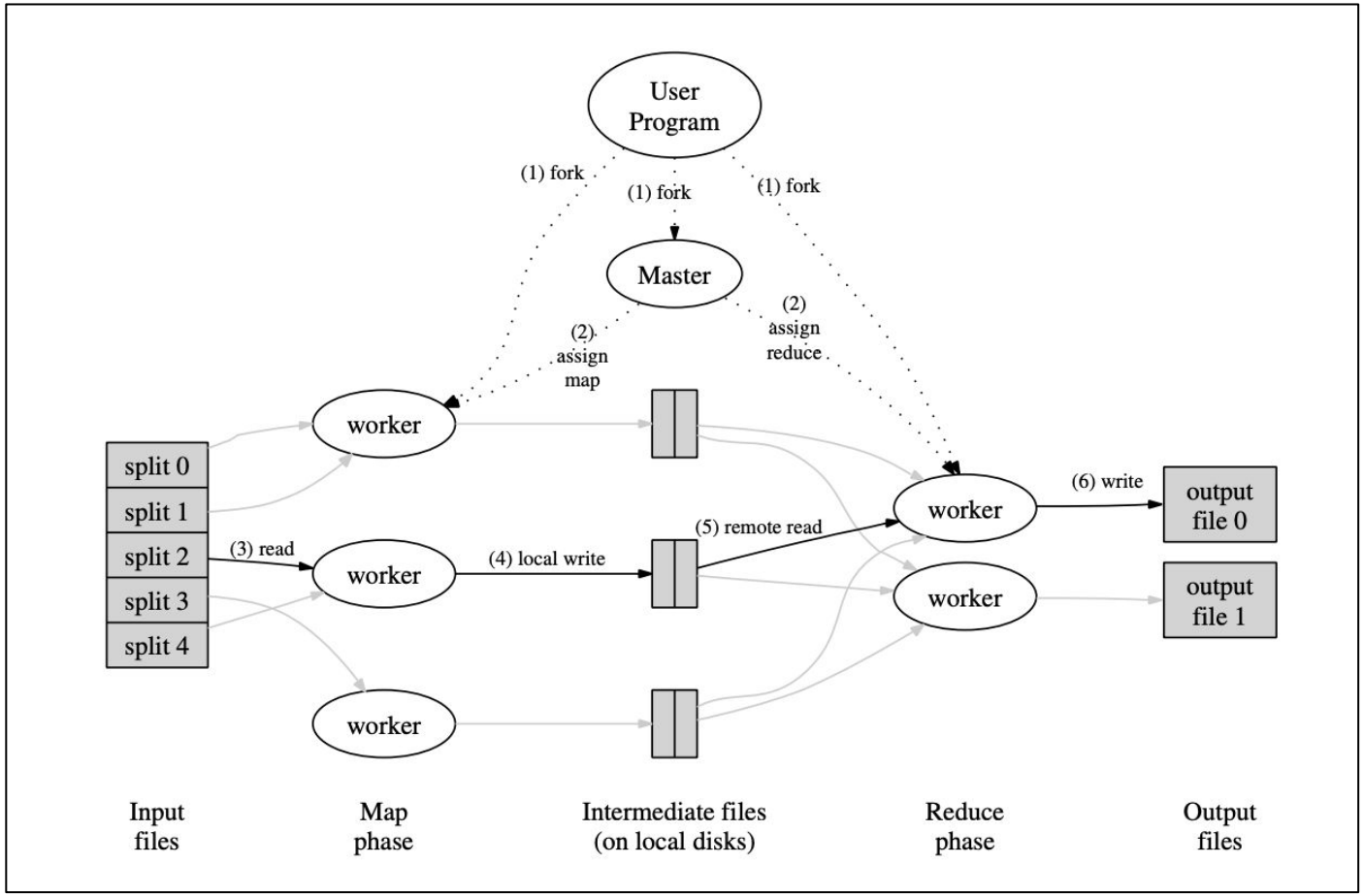
- A blessing and a curse

Traditional 3-tier model

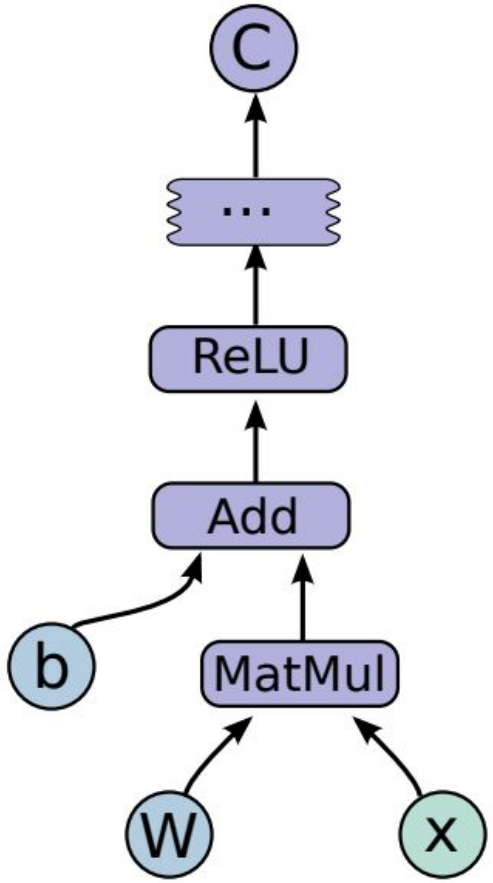


https://en.wikipedia.org/wiki/Multitier_architecture

Map reduce model



TensorFlow model (example)



Timely dataflow (Naiad)

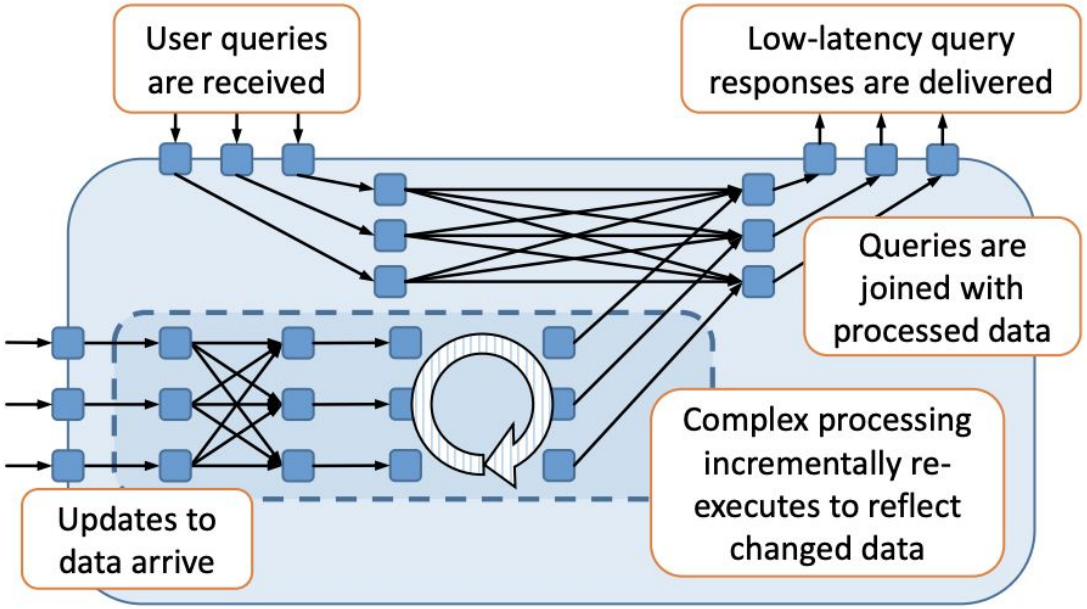
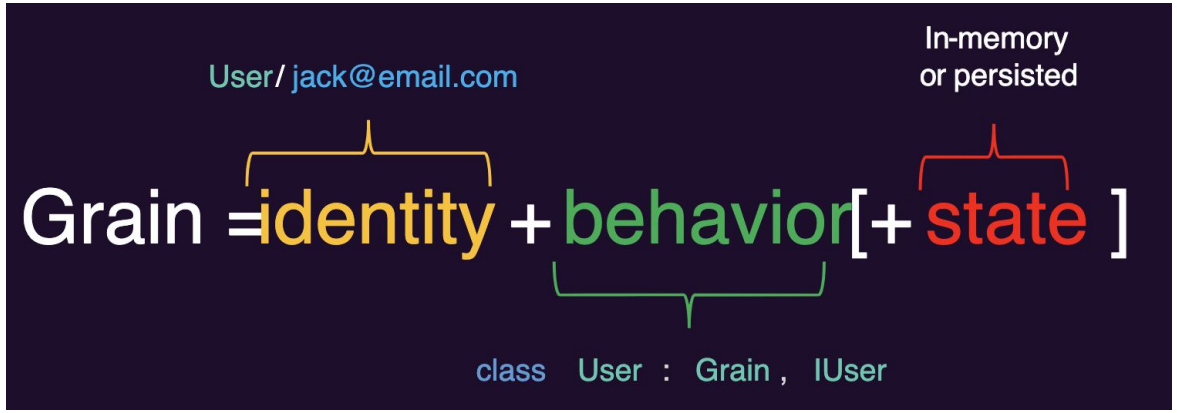
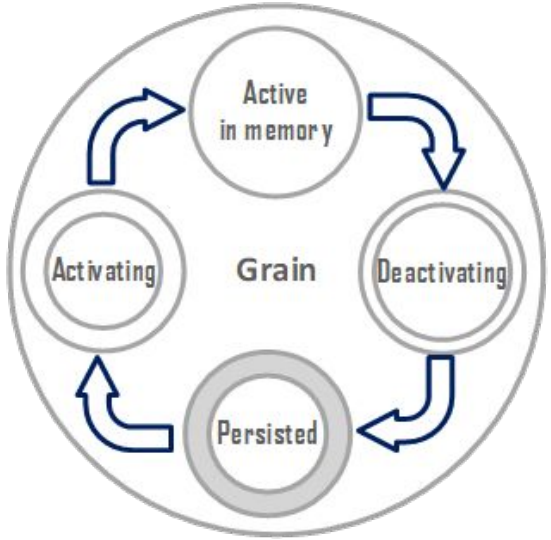


Figure 1: A Naiad application that supports real-time queries on continually updated data. The dashed rectangle represents iterative processing that incrementally updates as new data arrive.

https://www.microsoft.com/en-us/research/wp-content/uploads/2013/11/naiad_sosp2013.pdf

Virtual actor (Orleans)





https://github.com/dotnet//orleans

Model selection considerations

Application type: Batch, interactive, streaming

- Often dictates performance metric

Computational graph: Stages, cycles, ..

Transparency: Black box, gray box, white box

Over to Emmanuel