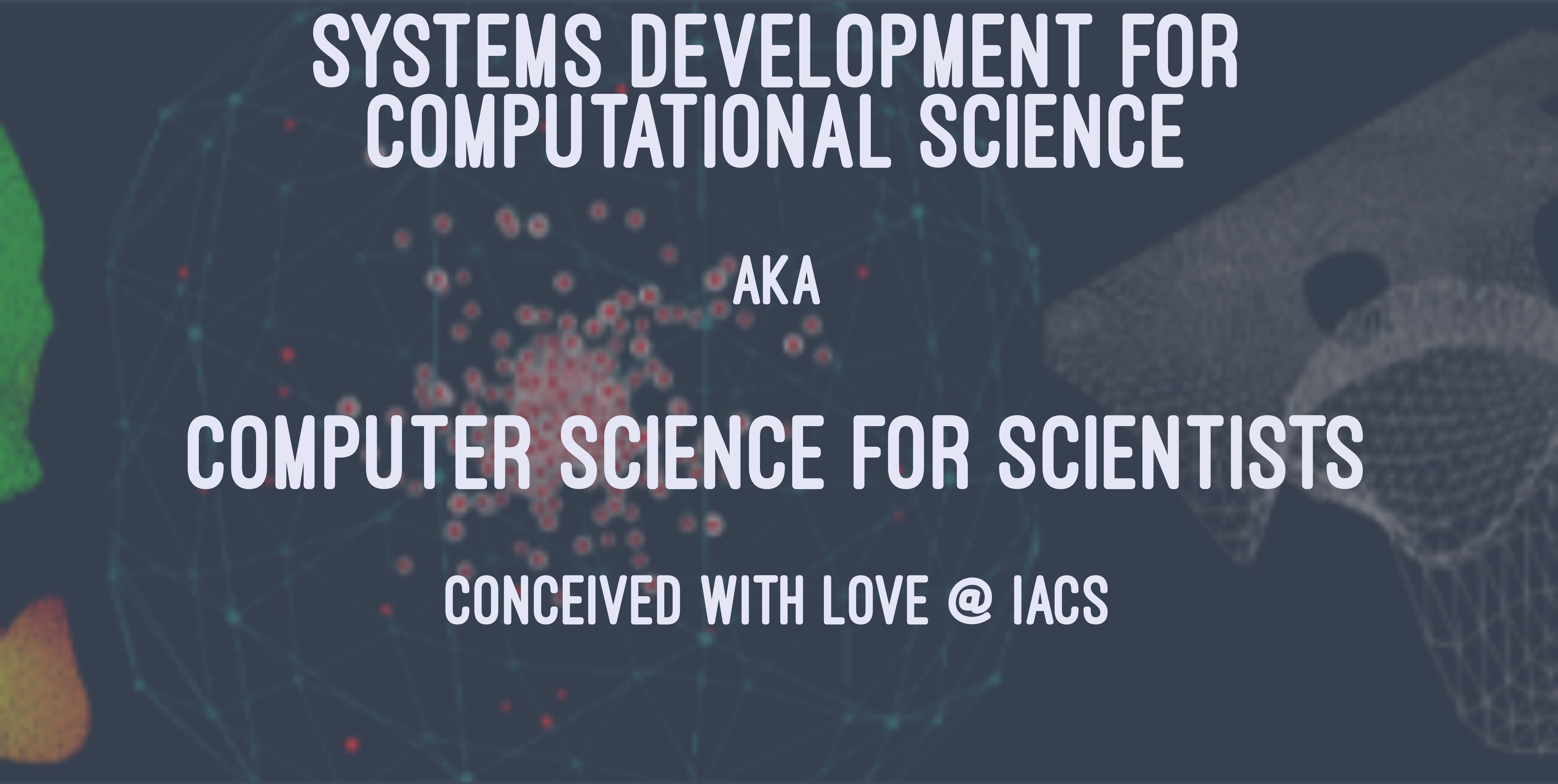
IACS

HARVARD

School of Engineering and Applied Sciences

CS207



SYSTEMS DEVELOPMENT FOR COMPUTATIONAL SCIENCE

AKA

COMPUTER SCIENCE FOR SCIENTISTS

CONCEIVED WITH LOVE @ IACS

THIS TALK:

1. WHY DO THIS COURSE
2. WHAT'S IN THE COURSE
3. LOOK AT SCHEDULE AND SOME PRACTICALITIES

WHY DO IT?

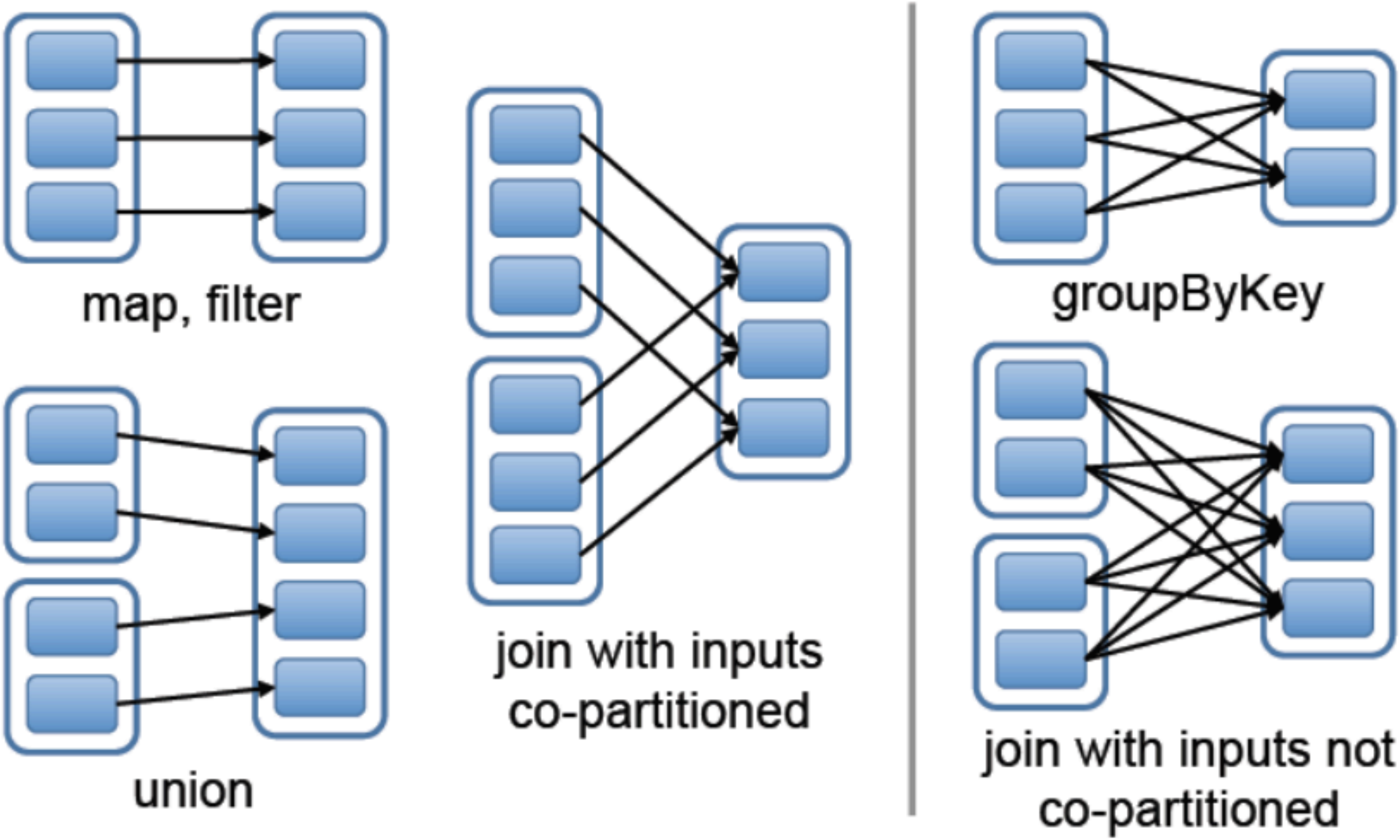
SPARK[1]

- "BIG DATA" ANALYSIS
- IMMUTABLE DATA TABLES, CALLED RDDEES
- RDDS PARTITIONED BETWEEN PROCESSES
- LAZY EVALUATION OF TRANSFORMATIONS BETWEEN RDDEES
- BCOZ GRAPH BASED REPRESENTATION OF RDD TRANSFORMATIONS

[1] DIAGS FROM HTTPS://WWW.CS.CMU.EDU/~PAVLO/COURSES/FALL2013/STATIC/SLIDES/SPARK.PDF

Narrow Dependencies:

Wide Dependencies:



map, filter

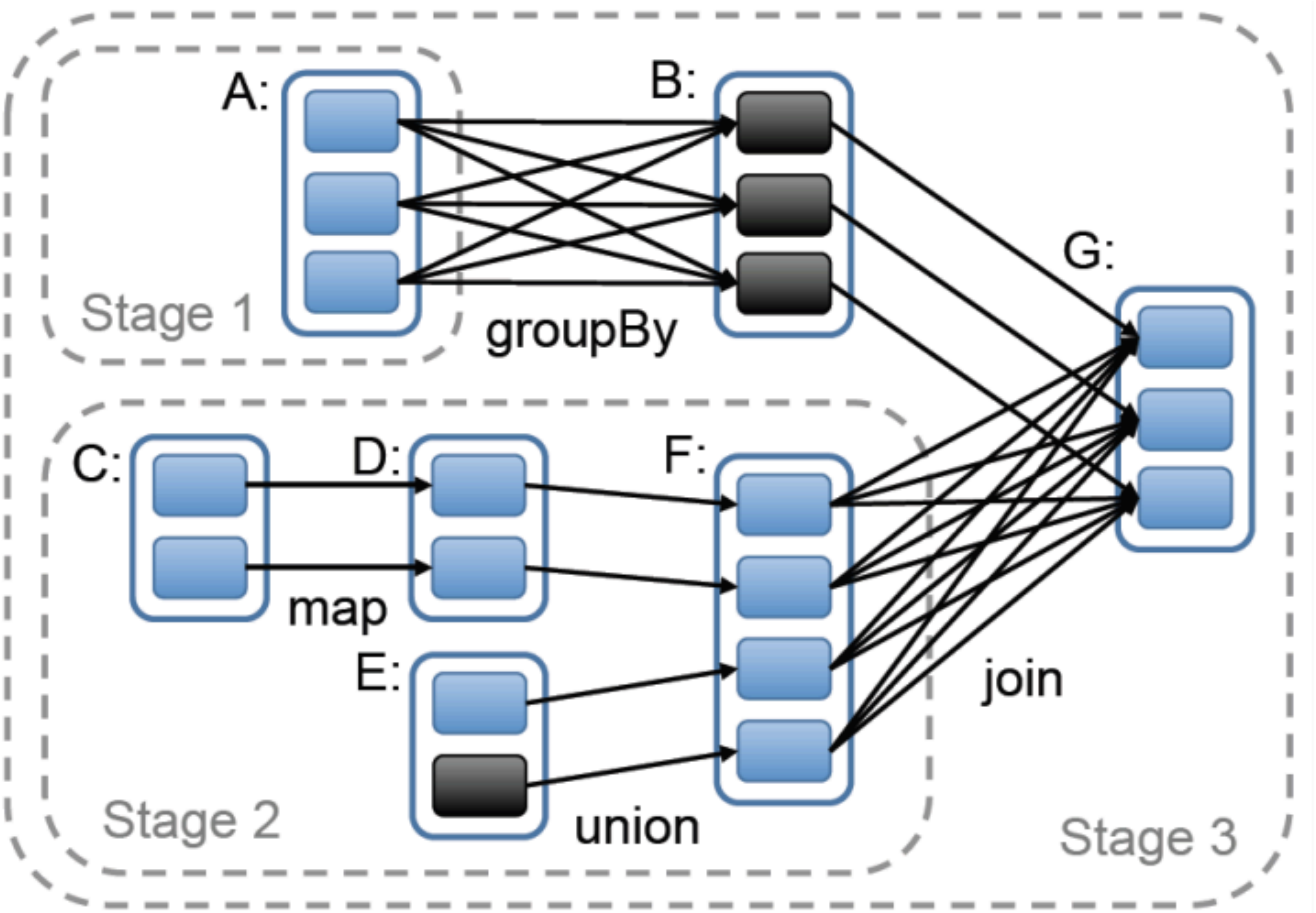
union

join with inputs co-partitioned

groupByKey

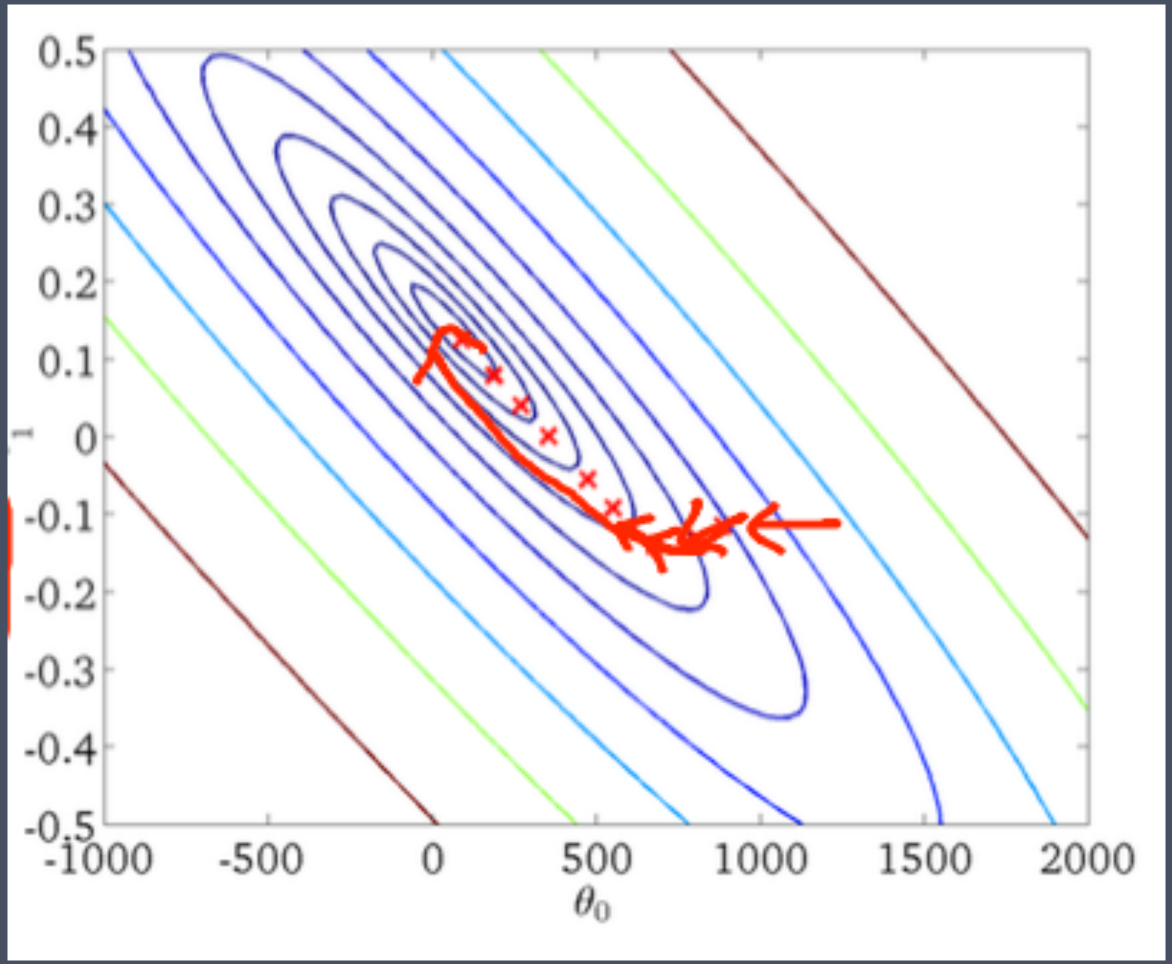
join with inputs not co-partitioned

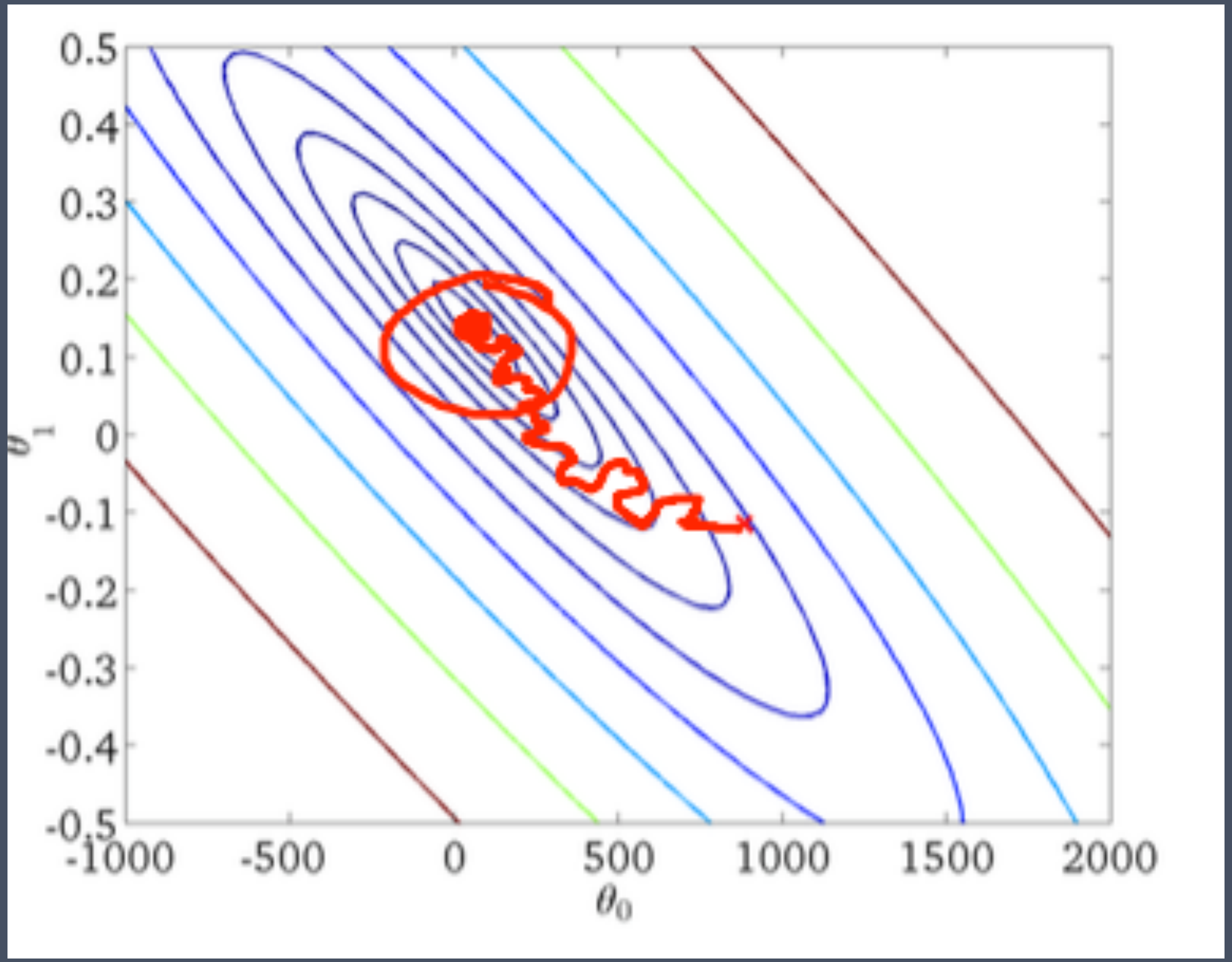
Job Scheduling



- FAULT RECOVERY USING LINEAGE FROM GRAPH
- FOR NARROWS, ONLY PARTIAL FAULT RECOVERY NEEDED (SEE GRAPH)
- KEY IDEAS: IMMUTABILITY GUARANTEES PARTICULAR DATASET AT ANY STAGE
- KEY IDEAS: COARSE GRAINED COMPUTATION REPRESENTED AS GRAPH ALLOWS SCHEDULING AND RECOVERY
- GRAPH DEPENDENCY SORTING: MAKEFILES, THEANO, SPARK, DASK, ETC, ETC

ANOTHER EXAMPLE: ASYNCHRONOUS SGD





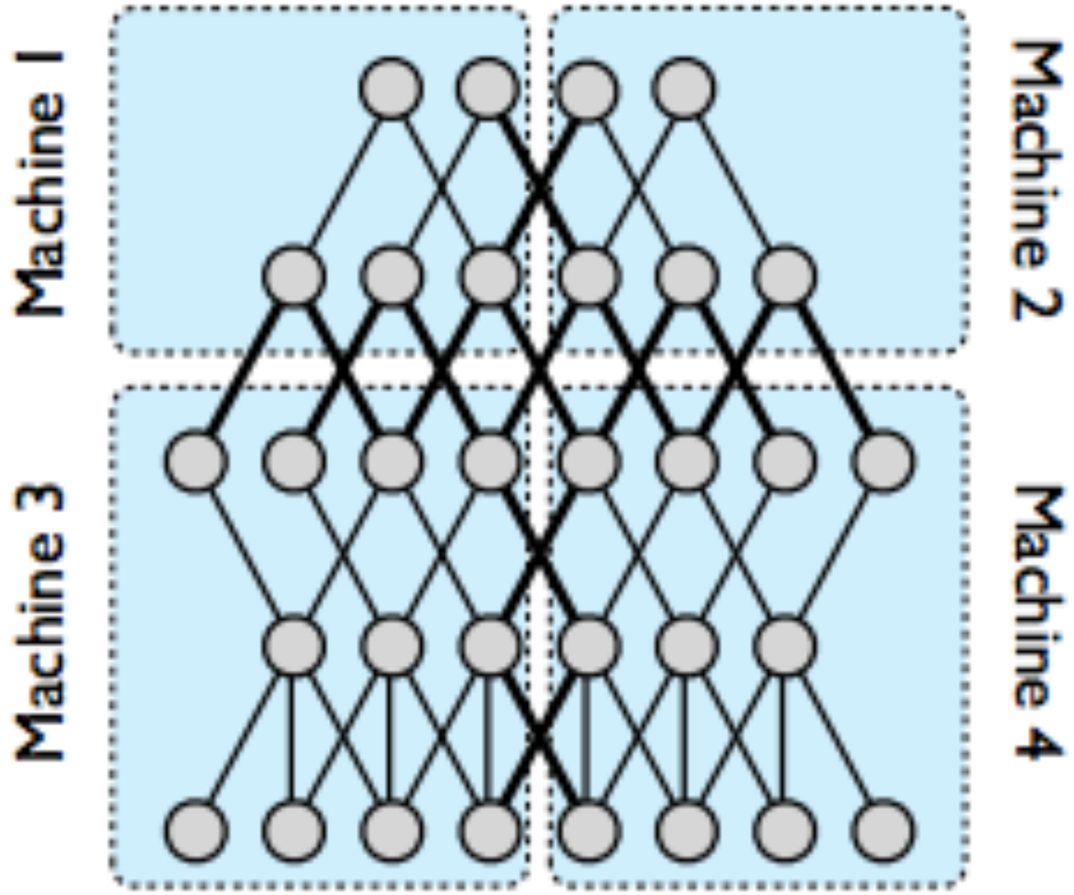
(FROM ANDREW NG'S COURSERA COURSE)

TO MINIMIZE $Q(w) = \sum_{i=1}^{n} Q_i(w)$ (FROM WIKIPEDIA)

- CHOOSE AN INITIAL VECTOR OF PARAMETERS AND LEARNING RATE .
- REPEAT UNTIL AN APPROXIMATE MINIMUM IS OBTAINED:

1. RANDOMLY SHUFFLE EXAMPLES IN THE TRAINING SET.
2. FOR I=1..N, DO:

- $w_{new} = w - \eta \nabla Q_i(w)$



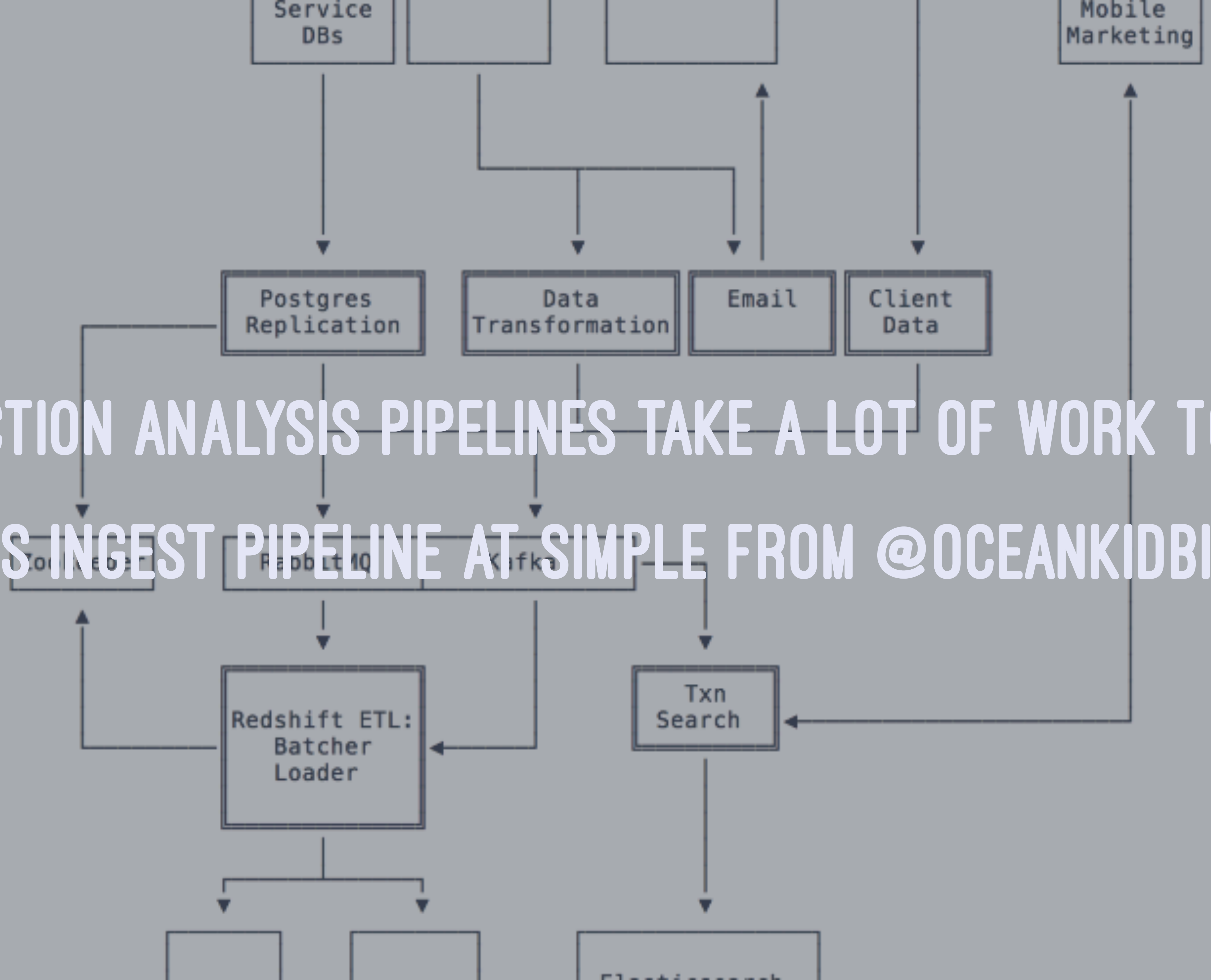
- SEE DEAN ET.AL 2012
- RUN COPY OF MODEL ON EACH BATCH (MULTIPLE BATCHES)
- COMMUNICATE PARAMETERS WITH SHARDED PARAMETER SERVER ASYNCHRONOUSLY (MULTIPLE PARAMETER SHARDS)

- ROBUST TO MACHINE FAILURE
- BUT MORE STOCHASTIC
- NOT GUARANTEED UPDATED PARAMETERS
- IN PRACTICE RELAXING CONSISTENCY WORKS!
- BETTER FOR MORE LOCAL NEURAL NETS

SCALING FROM RESEARCH TO PRODUCTION CAN BE COMPLEX:

READ HTTPS://OPENAI.COM/BLOG/INFRASTRUCTURE-FOR-DEEP-LEARNING/

NEED TO PROVISION CLUSTERS OF GPUS/CPUS STARTING FROM 1 MACHINE TO MANY...



PRODUCTION ANALYSIS PIPELINES TAKE A LOT OF WORK TOO!

ANALYSIS INGEST PIPELINE AT SIMPLE FROM @OCEANKIDBILLY

YOU MAY, ADDITIONALLY, HAVE TO CREATE:

- INTERNAL DASHBOARDS AND APPS FOR BUSINESS DECISIONS
- THUS WILL HAVE TO BE FAMILIAR WITH WEB TECHNOLOGIES
- KNOW HOW TO USE VIRTUALIZED AND CONTAINERIZED INFRASTRUCTURE
- PRODUCE AND CONSUME INTERNAL AND EXTERNAL API ENDPOINTS

HOW TO START?

A COURSE WITH 3 SIMULTANEOUS THEMES…

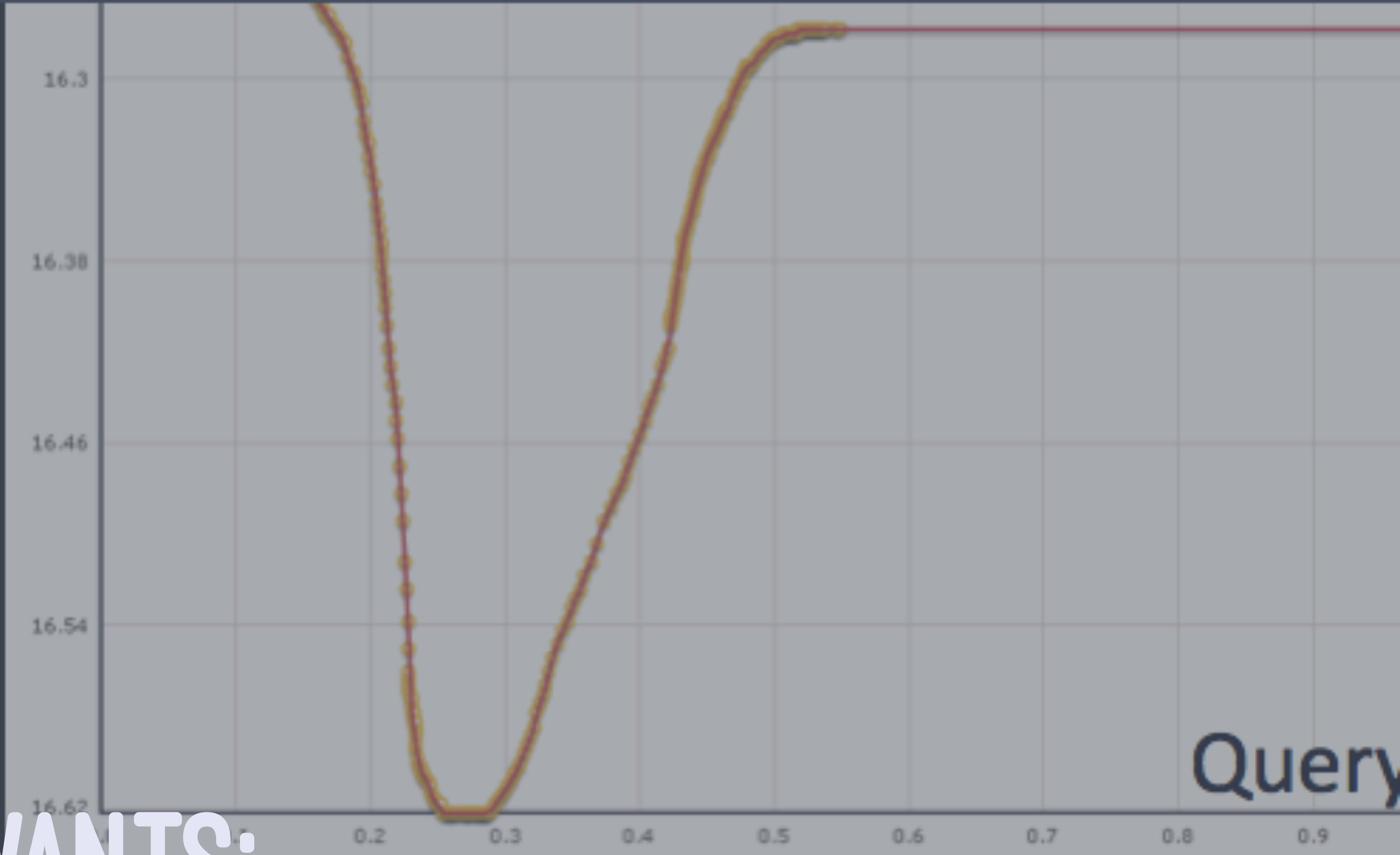
1. ENGINEERING A TIGHT SHIP

2. WHAT'S IN THE LANGUAGE

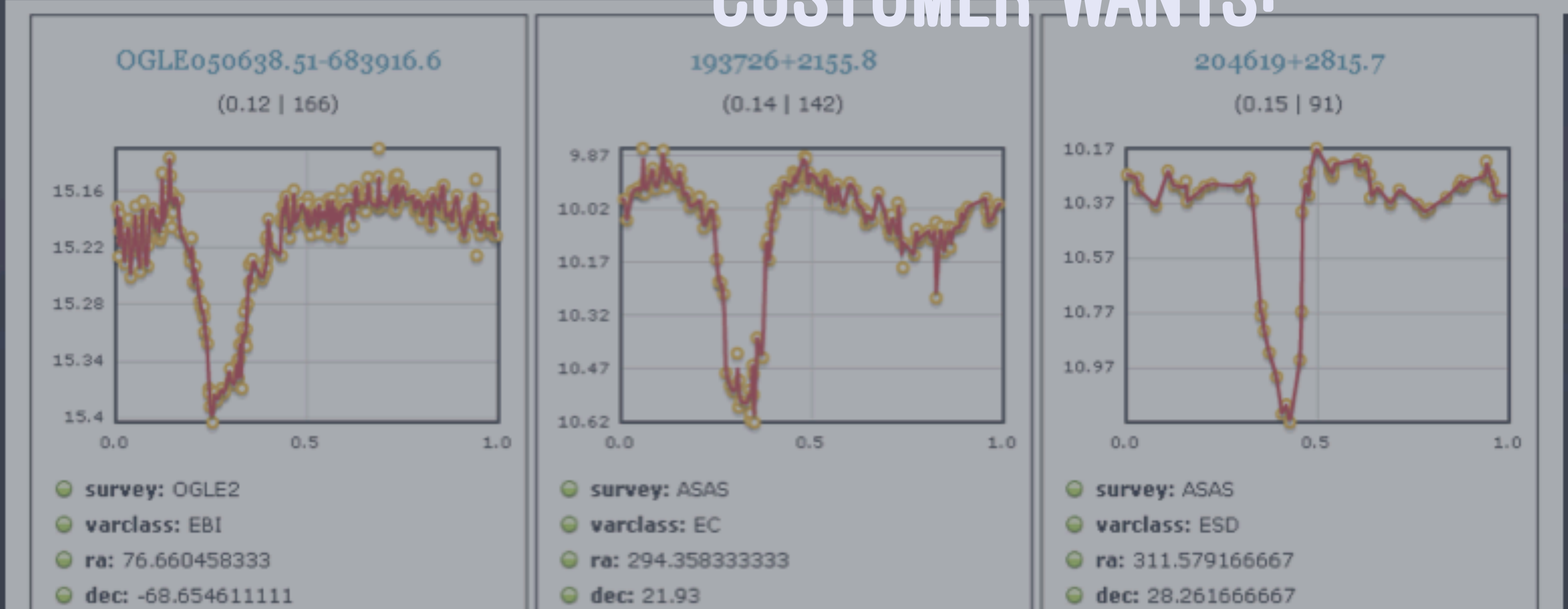
3. ALGORITHMS FIND DATA.

Problem.

- Searching Light Curves
- 100-1000 Data-points



CUSTOMER WANTS:



OGLE050638.51-683916.6

(0.12 | 166)

- survey: OGLE2
- varclass: EBI
- ra: 76.660458333
- dec: -68.654611111

193726+2155.8

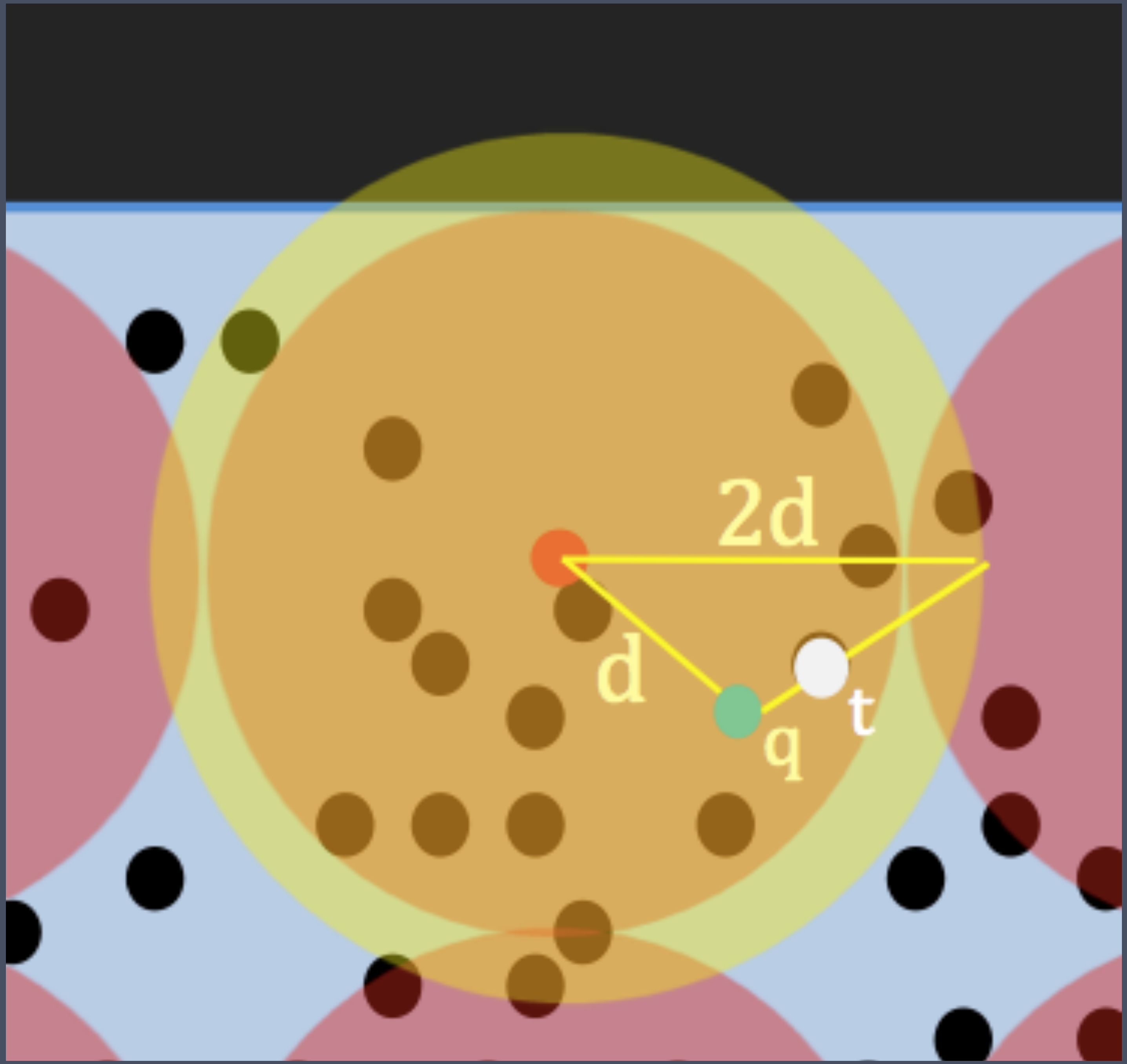
(0.14 | 142)

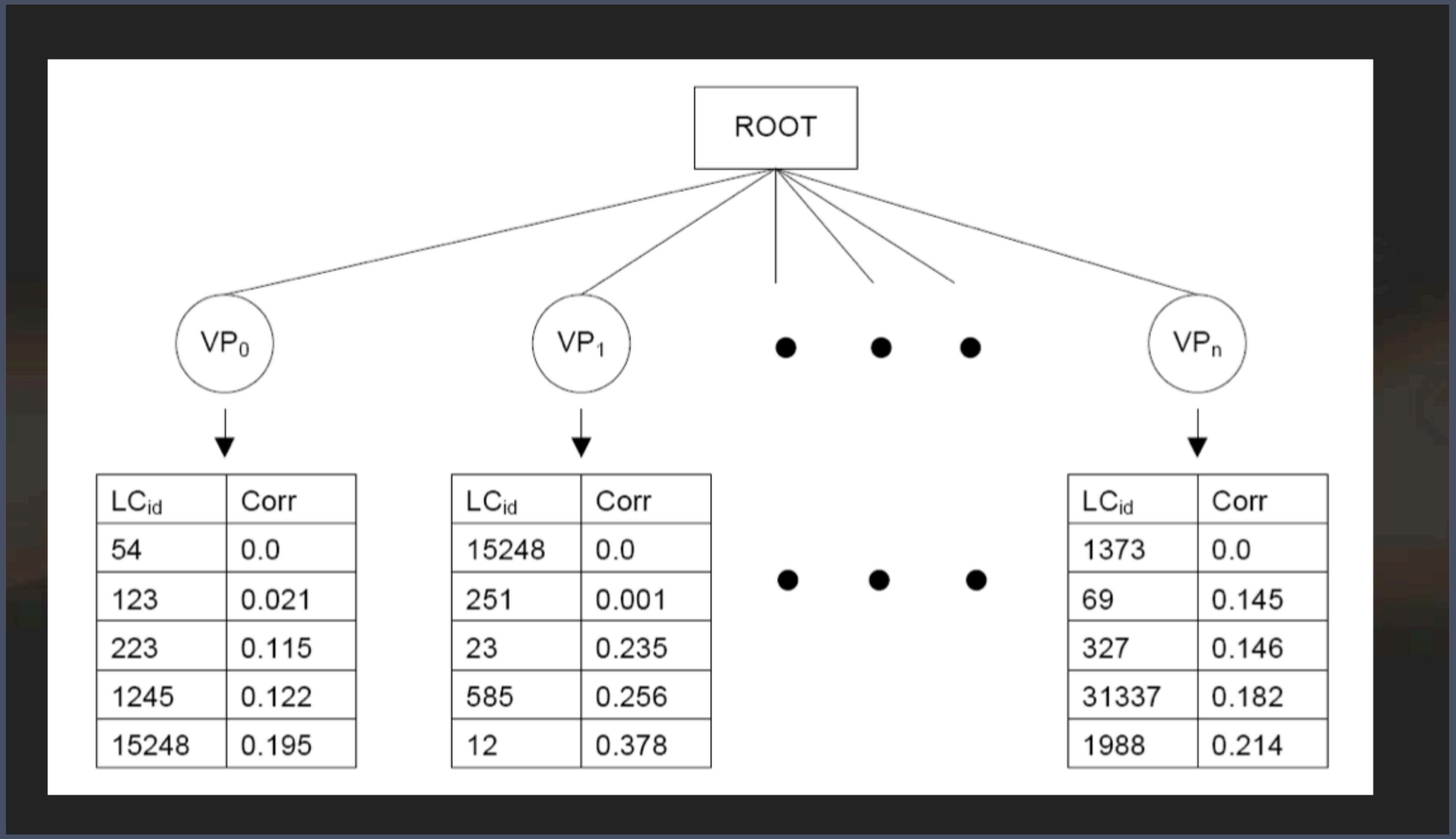
- survey: ASAS
- varclass: EC
- ra: 294.358333333
- dec: 21.93

204619+2815.7

(0.15 | 91)

- survey: ASAS
- varclass: ESD
- ra: 311.579166667
- dec: 28.261666667





LC$_{id}$	Corr
54	0.0
123	0.021
223	0.115
1245	0.122
15248	0.195

LC$_{id}$	Corr
15248	0.0
251	0.001
23	0.235
585	0.256
12	0.378

LC$_{id}$	Corr
1373	0.0
69	0.145
327	0.146
31337	0.182
1988	0.214

KNN SEARCH IN A LARGE SPACE, LEADS TO AN INDEX...

I. A TIME SERIES LIBRARY

(WITH OPERATIONS BETWEEN, AND ALGORITHMS ON TIME SERIES)

- PYTHON
- EXECUTION MODEL, PROGRAMMING PARADIGMS, OBJECT MODEL
- SEQUENCES, ITERATORS, LISTS, ARRAYS, TREES, HASHES
- TESTS, DEBUGGING, CI, AUTOMATING

II. A TIME SERIES DATABASE

(WITH INDEXING, SIMILARITY QUERIES, QUERY LANGUAGE)

- WRITING A CLI, A REPL, A DSL, WEB ACCESS
- APPROPRIATE ON-DISK DATA STRUCTURES FOR THE INDEXES AND TS
- USAGE OF A DATABASE

III. A REST INTERFACE AND UI TO THESE AND MORE

(THE MORE IS YOUR CHOICE, BUT YOU MUST DELIVER A DEMO)

- YOU WRITE AN EXTENSION OF YOUR CHOICE
- YOU PACKAGE EVERYTHING UP IN ONE OR MORE SERVERS, ACCESSED BY A WEB API AND INTERFACE, AND PROVIDE A DEMO

HOW?

- ALL WORK IS ON GITHUB.
- ONE HOMEWORK A WEEK, TO BE SUBMITTED INDIVIDUALLY
- YOU WILL WORK IN GROUPS OF 3-4. YOU CAN COLLABORATE AS MUCH AS YOU WANT WITHIN THE GROUP (INCLUDING HOMEWORK, BUT YOU MUST WRITE UP YOUR OWN HOMEWORK).
- ONE REPO PER GROUP FOR YOUR PROJECT.
- NO EXAMS

HOW (CONTD)

- THE BASIC PART OF THE PROJECT (I AND II) WILL BE DONE BY ALL GROUPS.
- ONCE A GROUP WILL IMPLEMENT A FEATURE FOR ANOTHER GROUP
- GROUPS WILL DO DIFFERENT THINGS FOR THE ELECTIVE PART AND REST API, WEB UI, DATABASE

WHEN?

- MW 2.30PM - 4PM LECTURE PIERCE 209
- FRIDAYS 3PM-4.30PM (LAB) PIERCE 301

GRADE

- HOMEWORK (40%)
- PROJECT (45%)
- PROJECT X-DEVEL (7%)
- PARTICIPATION (8%): IN DISCUSSIONS, COMMITS, PEER REVIEW AND CODE REVIEW EXERCISE IN M1 OR M2

YOU WILL PEER-REVIEW EACH OTHER.

WE'LL DO PRACTICALITIES IN LAB 1 THIS FRIDAY. BUT IF YOU WANT TO GET STARTED:

- INSTALL ANACONDA PYTHON (PYTHON 3.5)
- CREATE A GITHUB ACCOUNT, WITH A REPO CALLED `cs207work`
- INSTALL A GIT-CLIENT. ON WINDOWS INSTALL GIT-BASH.
- YOU MAY OPTIONALLY WANT TO INSTALL THE GITHUB CLIENT.
- CHOOSE A CODE EDITOR: I LIKE ATOM.

- FUN IS GUARANTEED!
- LOTS OF COOL STUFF TO LEARN
- LEARN TO LEARN, NOT TO GET A GRADE

UNICORN LEVEL 1 UNLOCKED.