CS 744: GANDIVA

Shivaram Venkataraman

Fall 2019

ADMINISTRIVIA

- Course project proposal
- Midterm

Machine Learning

Bismarck

Supervised learning, Unified Interface
Shared memory, Model fits in memory

Parameter Server

Large datasets, large models (PB scale)
Consistency model, Fault tolerance

Tensorflow

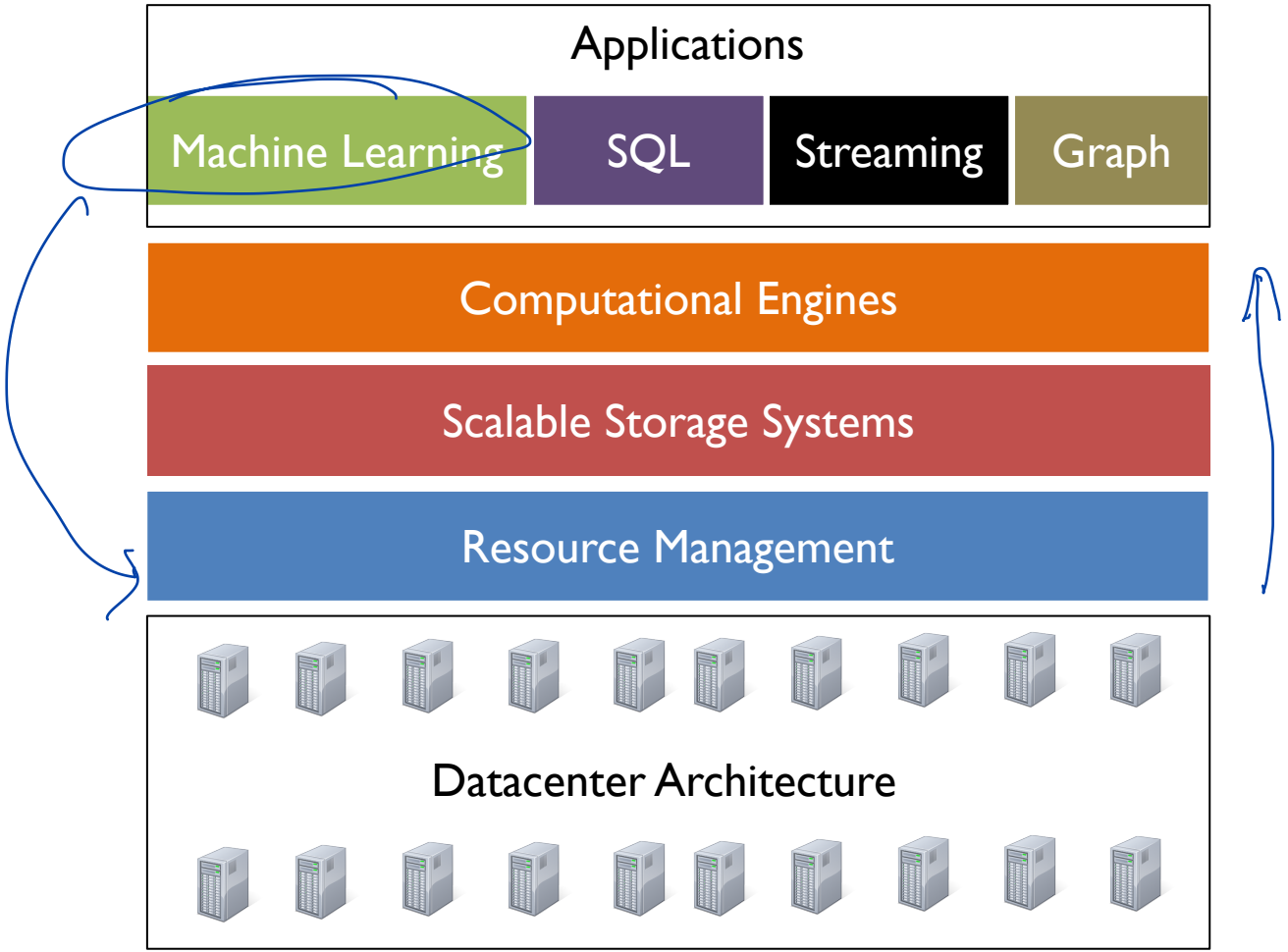
Need for flexible programming model
Dataflow graph, Heterogeneous accelerators

Ray

Reinforcement learning applications
Actors and tasks, Local and global scheduler

Inference

Clipper


Applications
Machine Learning
SQL
Streaming
Graph
Computational Engines
Scalable Storage Systems
Resource Management
Datacenter Architecture

MACHINE LEARNING WORKFLOW?

PyTorch

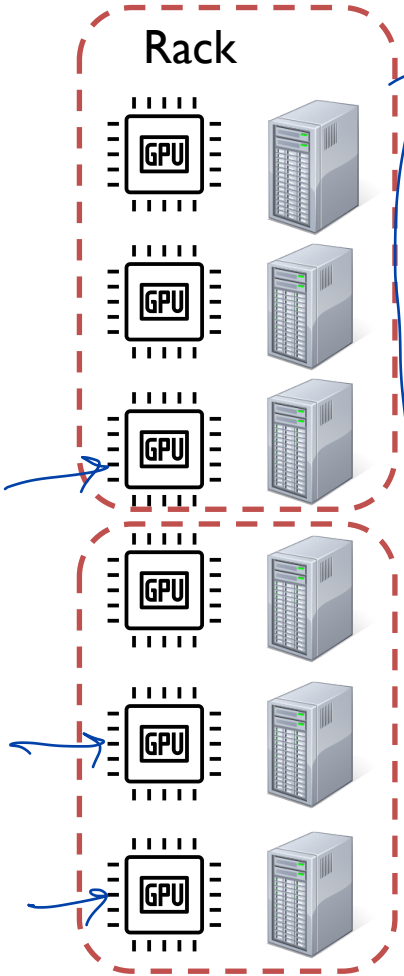
TensorFlow

mxnet

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 10, kernel_size=5)
        self.conv2 = nn.Conv2d(10, 20, kernel_size=5)
        self.conv2_drop = nn.Dropout2d()
        self.fc1 = nn.Linear(320, 50)
        self.fc2 = nn.Linear(50, 10)
```

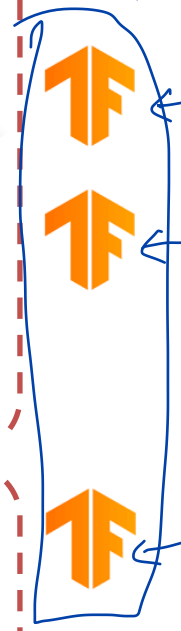
SHARED ML CLUSTERS

↳ Multi tenant

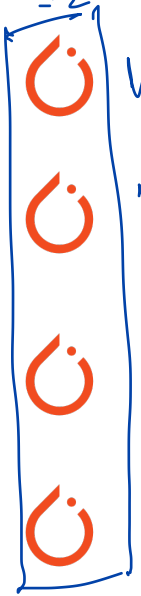


Maintainence
Power / Cooling

"exclusive"



J_1



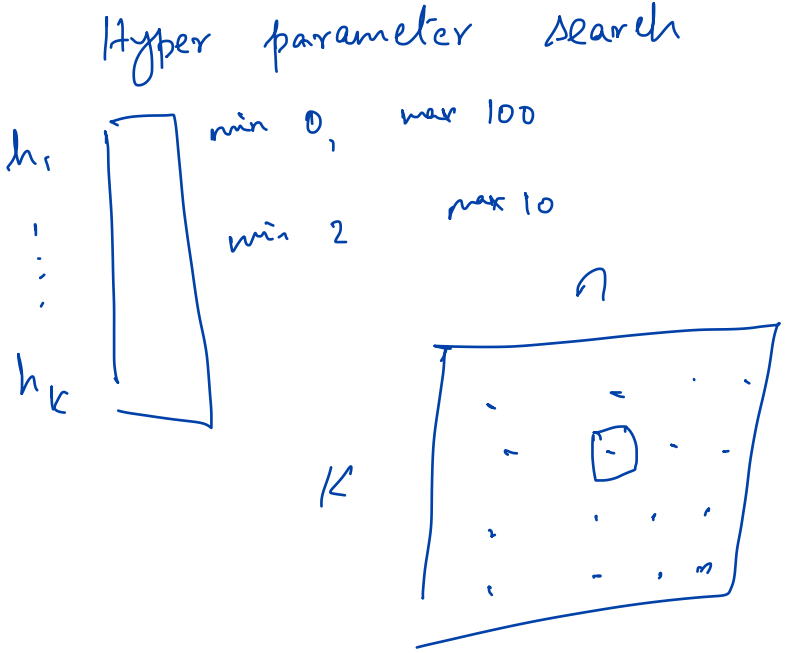
J_2 → enqueue

if J_1 takes 1 day to run
and J_2 waits for 1 day

→ Efficiency of cluster hardware is low

WORKLOAD

Feedback-driven exploration



tune

$\min f(x) + \frac{1}{2}\lambda \|x\|_2$

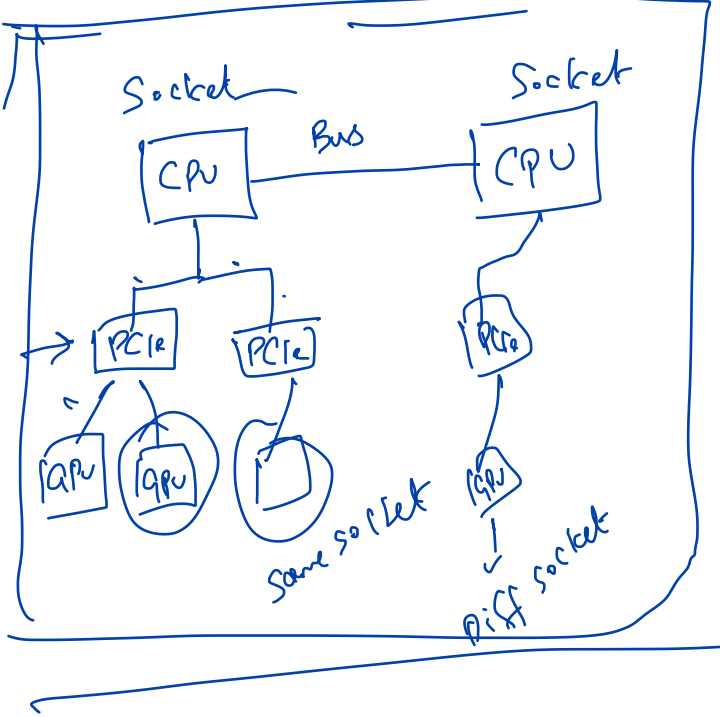
value? regularization

→ learning rate

Early stopping

truncate execution

of suboptimal hyper parameters



~~AFFINITY~~ Locality

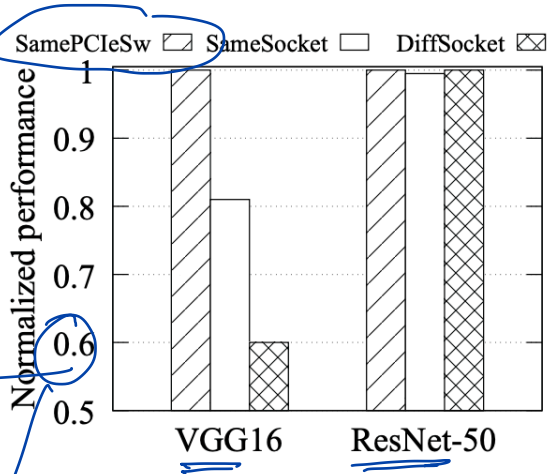


Figure 1: Intra-server locality.

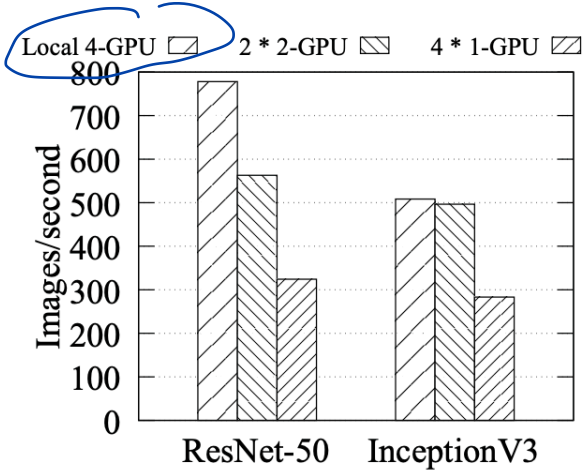
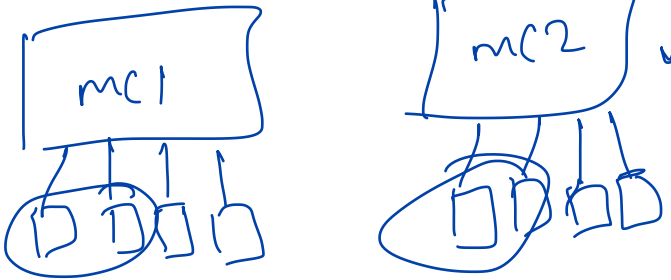


Figure 2: Inter-server locality.



40%.

ML models are sensitivity to locality of placement

INTRA JOB PREDICTABILITY

while

Sample

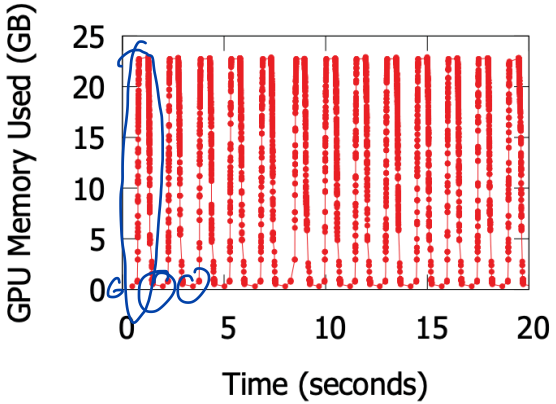
↓ Compute ∇f

Aggregate Sync / Async

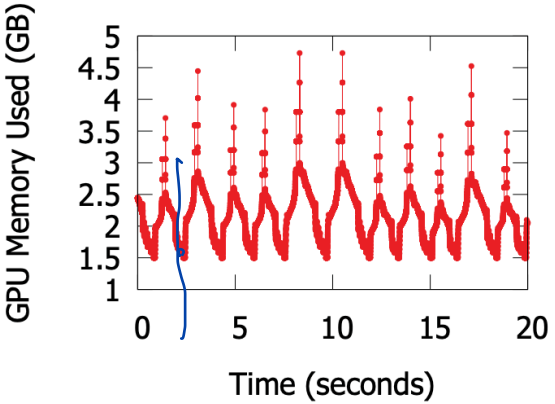
↔ end

Predictability → useful

Heterogeneity

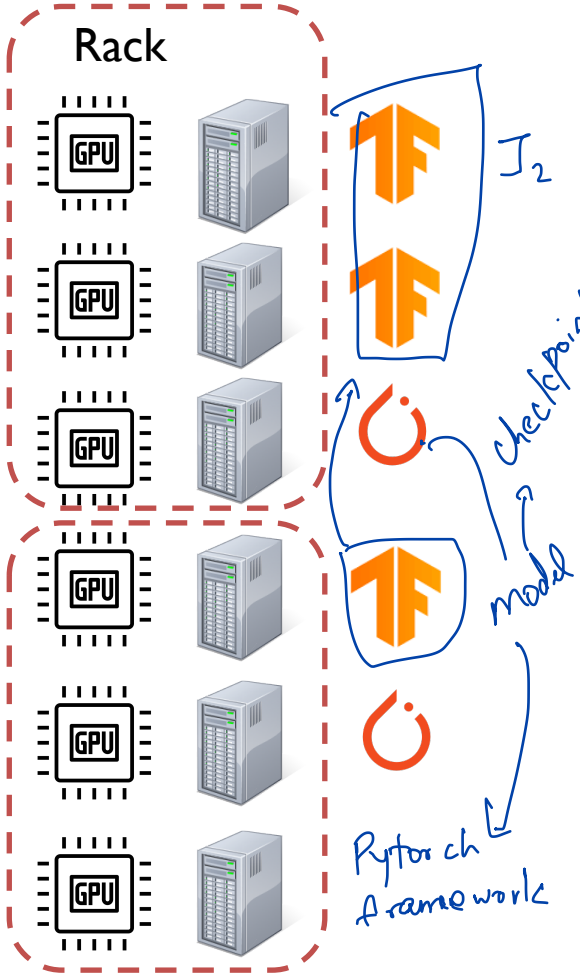


(a) ResNet50/Imagenet



(b) GNMT/WMT'14 En-De

MECHANISMS (1)



1. Suspend-Resume

→ suspend job 1, place job 2 on GPUs for time quantum → 1 min

→ suspend job 2, resume Job 1

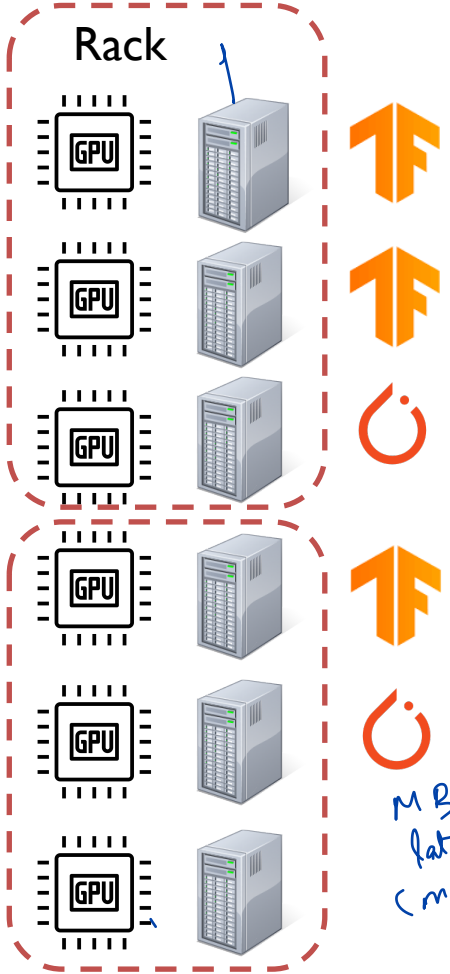
→ Use Intra job predictability to suspend MB finishes

2. Migration

→ More tasks to improve locality or improve utilization

→ Copy state old mc and new mc
↳ 1GB ≃ longer time period

MECHANISMS (2)



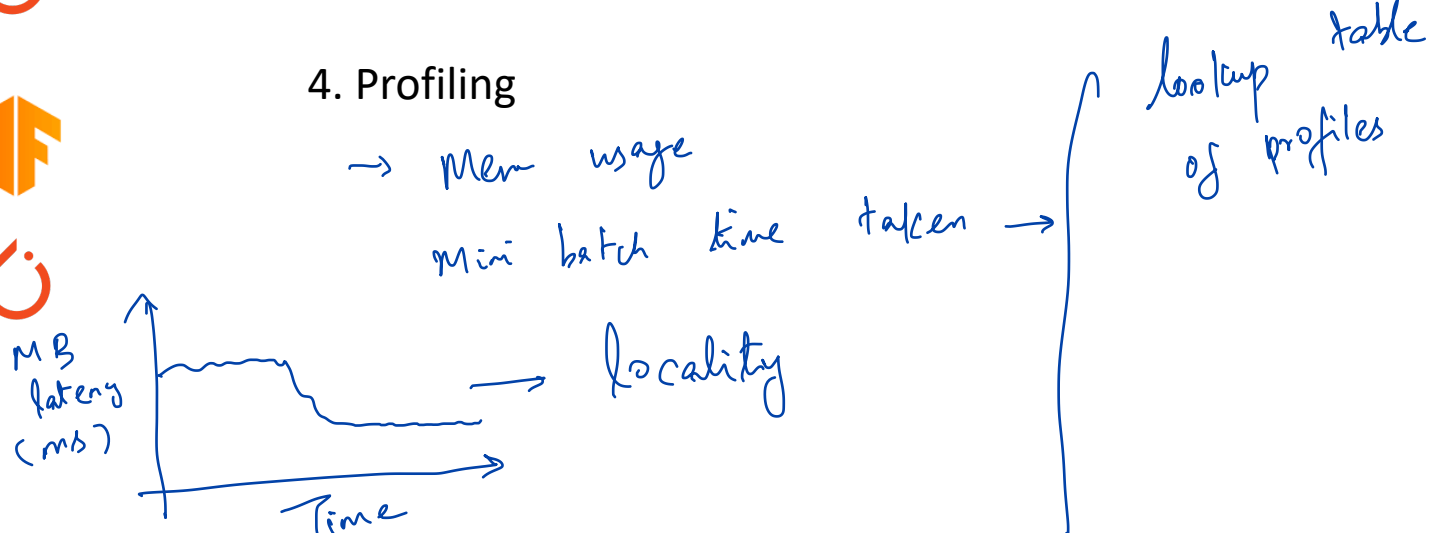
3. Grow-shrink

→ if GPUs idle, can we give them to existing tasks

b = 256, 2 GPUs
+
2 GPUs when they become free

4. Profiling

→ Mem usage
mini batch time taken →



SCHEDULING POLICY

Goals

- *early feedback*
- *cluster efficiency*
- *cluster-level fairness?* → Non-goal

Two modes

- Reactive
- Introspective

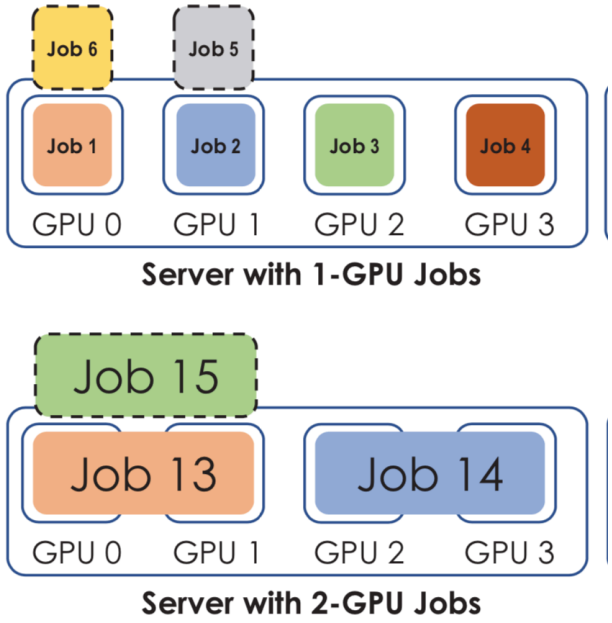
REACTIVE MODE

React to events

- Job arrivals, departures, failures

Hierarchical Preference

- Nodes with same "*affinity*"
- Nodes with "*different affinity*"
- Nodes with "*no affinity*"
- Suspend-resume …



INTROSPECTIVE MODE

bns://tf-job-3243:8888
-worker-1

Monitor and optimize placement of jobs periodically

Actions

- Packing → Over subscribe by using resources predictability | greedy → profiling → undo mis
- Migration
- Grow-shrink

Priority = High 2 time quantum
low 1 time quantum

DISCUSSION

https://forms.gle/aHYbNcTFdGJtXefj9

What are some guarantees provided by Mesos that are not provided by Gandiva? Explain with an example

↳ DRF as its scheduling policy

Mesos - "resource guarantee"

- Strategy proof, while Gandiva. Jobs can inflate mini batch size

- Fairness → sharing incentive

Are mechanisms in Gandiva also useful in a cluster running Apache Spark jobs? Provide one example either for or against

↳ Migration is not that useful because of backup tasks?

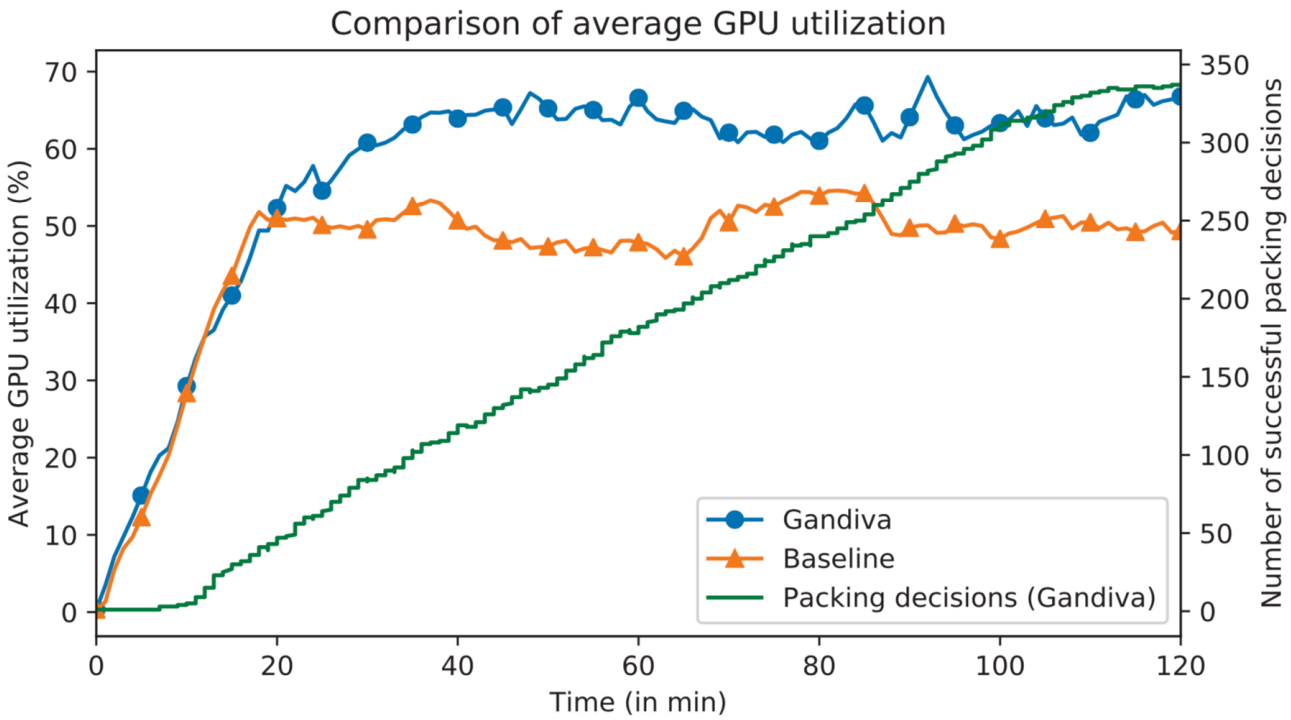
↳ need to transfer data, RDD or shuffle data

Can be useful for large shuffles

↳ Suspend/Resume may not be useful because of lack of predictability

↳ PageRank, Logistic Regression do have predictability within the DAG

↳ Naturally happening task boundaries


Comparison of average GPU utilization
Average GPU utilization (%)
Number of successful packing decisions
Time (in min)
Gandiva
Baseline
Packing decisions (Gandiva)
0
10
20
30
40
50
60
70
0
50
100
150
200
250
300
350
0
20
40
60
80
100
120

NEXT STEPS

New module on SQL!

Course project introductions

Midterm