ТЕХНОСФЕРА

# Лекция 12
# Large scale machine learning

Владимир Гулин

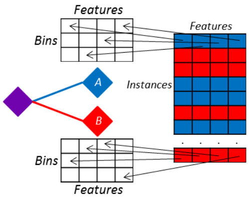
19 мая 2018 г.

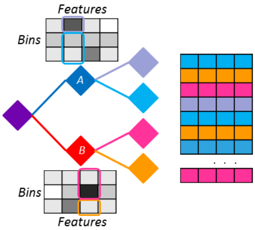
# План лекции

Large scale decision trees ensembles

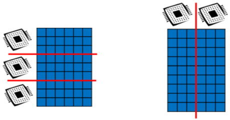
Large scale neural networks

# Distributed tree construction



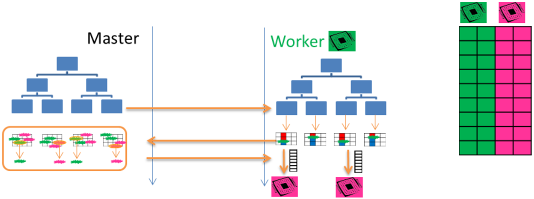




- Наблюдение 1: Одного прохода по данным достаточно на каждый уровень дерева
- Наблюдение 2: Итерироваться можно либо по точкам, либо по фичам



# Distributed tree construction

## Feature Distributed




Master

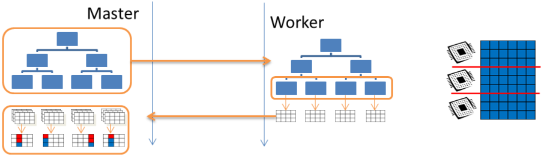
- Request workers to expand a set of nodes
- Wait to receive best per-feature splits from workers
- Select best feature-split for every node
- Request best splits' workers to broadcast per-instance assignments and residuals

Worker

- Pass through all instances for local features, aggregating split histograms for each node
- Select local features' best splits for each node, send to master

# Distributed tree construction

## Data Distributed




Master

- Send workers current model and set of nodes to expand
- Wait to receive local split histograms from workers
- Aggregate local split histograms, select best split for every node

Worker

- Pass through local data, aggregating split histograms
- Send completed local historograms to master

# Distributed tree construction

## Data Distributed for sparse features

**Algorithm 2** FindBestSplit

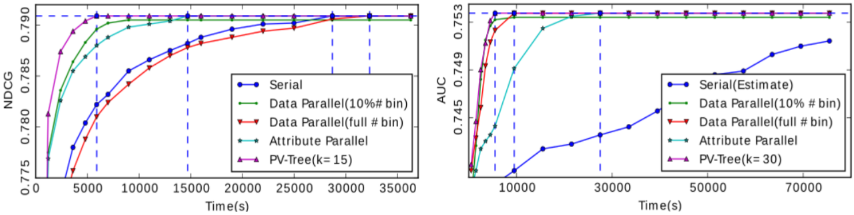
```
Input: DataSet D
for all X in D.Attribute do
    ▷ Construct Histogram
    H = new Histogram()
    for all x in X do
        H.binAt(x.bin).Put(x.label)
    end for
    ▷ Find Best Split
    leftSum = new HistogramSum()
    for all bin in H do
        leftSum = leftSum + H.binAt(bin)
        rightSum = H.AllSum - leftSum
        split.gain = CalSplitGain(leftSum, rightSum)
        bestSplit = ChoiceBetterOne(split,bestSplit)
    end for
end for
return bestSplit
```

**Algorithm 3** PV-Tree_FindBestSplit

```
Input: Dataset D
localHistograms = ConstructHistograms(D)
▷ Local Voting
splits = []
for all H in localHistograms do
    splits.Push(H.FindBestSplit())
end for
localTop = splits.TopKByGain(K)
▷ Gather all candidates
allCandidates = AllGather(localTop)
▷ Global Voting
globalTop = allCandidates.TopKByMajority(2*K)
▷ Merge global histograms
globalHistograms = Gather(globalTop, localHistograms)
bestSplit = globalHistograms.FindBestSplit()
return bestSplit
```

# Distributed tree construction

## Results




(a) LTR, 8 machines

(b) CTR, 32 machines

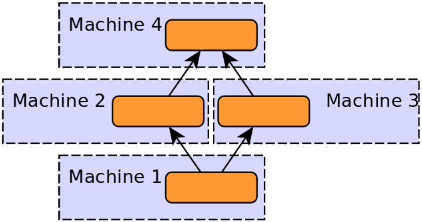
Figure 1: Performances of different algorithms

# GBM ON HADOOP

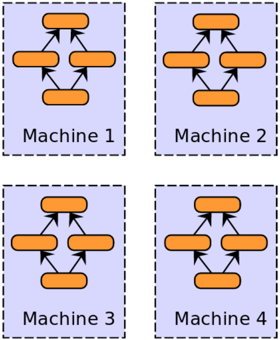
# Large scale neural networks

# Paradigms

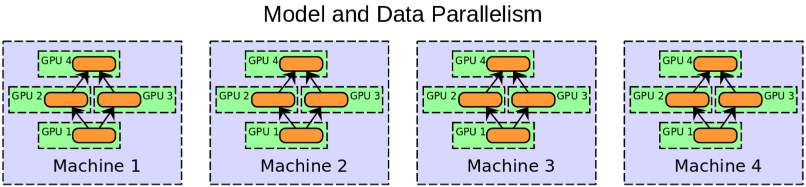
Model Parallelism




Data Parallelism



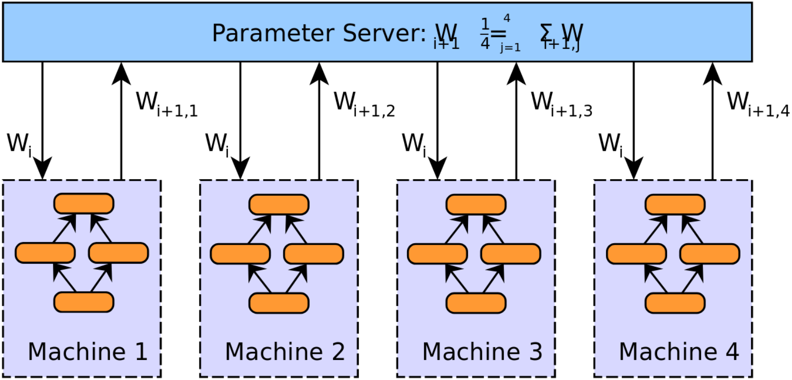
# Model and data parallelism



# Parameter averaging

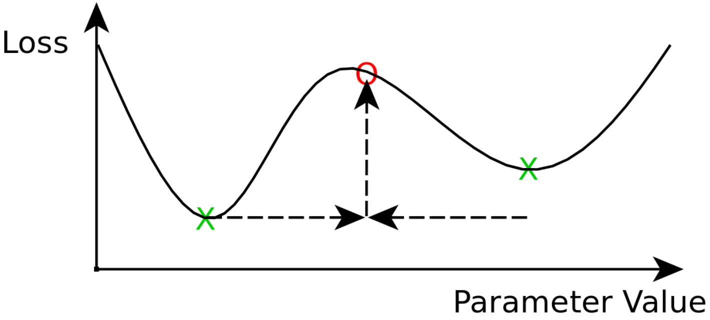
1. Initialize the network parameters randomly based on the model configuration
2. Distribute a copy of the current parameters to each worker
3. Train each worker on a subset of the data
4. Set the global parameters to the average the parameters from each worker
5. While there is more data to process, go to step 2

# Parameter averaging

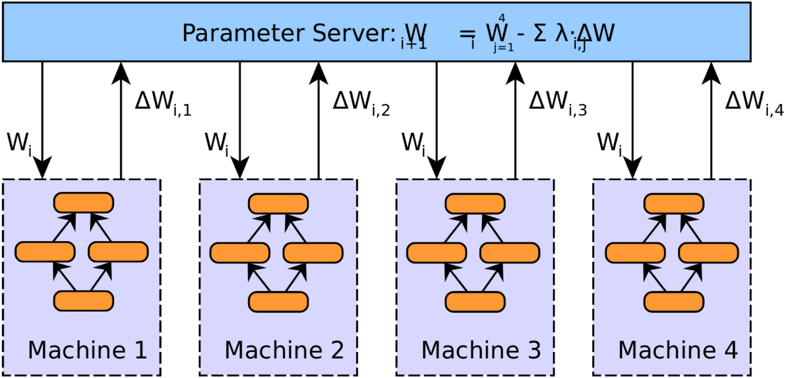


# Parameter averaging problem

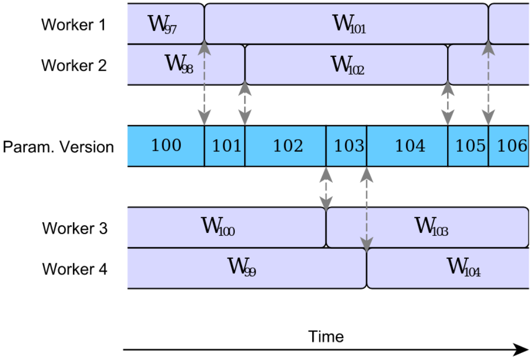
Сумма N локальных минимумов не является глобальным минимумом



# Asynchronous Stochastic Gradient Descent



# Stale Gradient Problem




$$\vdots$$

$$W_{101} = W_{100} - \lambda \Delta W_{97}$$

$$W_{102} = W_{101} - \lambda \Delta W_{98}$$

$$W_{103} = W_{102} - \lambda \Delta W_{100}$$

$$W_{104} = W_{103} - \lambda \Delta W_{99}$$

$$\vdots$$

Вопросы