Big Data Stream Mining Tutorial

Gianmarco De Francisci Morales, Joao Gama, Albert Bifet, Wei Fan

IEEE BigData 2014

Organizers (1/2)

Gianmarco De Francisci Morales

is a Research Scientist at Yahoo Labs Barcelona. His research focuses on large scale data mining and big data, with a particular emphasis on web mining and Data Intensive Scalable Computing systems. He is an active member of the open source community of the Apache Software Foundation working on the Hadoop ecosystem, and a committer for the Apache Pig project. He is the co-leader of the SAMOA project, an open-source platform for mining big data streams.

http://gdfm.me

João Gama

is Associate professor at the University of Porto and a senior researcher at LIAAD Inesc Tec. He received his Ph.D. degree in Computer Science from the University of Porto, Portugal. His main interests are machine learning, and data mining, mainly in the context of time-evolving data streams. He authored a recent book in Knowledge Discovery from Data Streams.

http://www.liaad.up.pt/~jgama

MAESTRA
LEARNING FROM MASSIVE, INCOMPLETELY ANNOTATED, AND STRUCTURED DATA

Organizers (2/2)

Albert Bifet

is a Research Scientist at Huawei. He is the author of a book on Adaptive Stream Mining and Pattern Learning and Mining from Evolving Data Streams. He is one of the leaders of MOA and SAMOA software environments for implementing algorithms and running experiments for online learning from evolving data streams.

http://albertbifet.com

Wei Fan

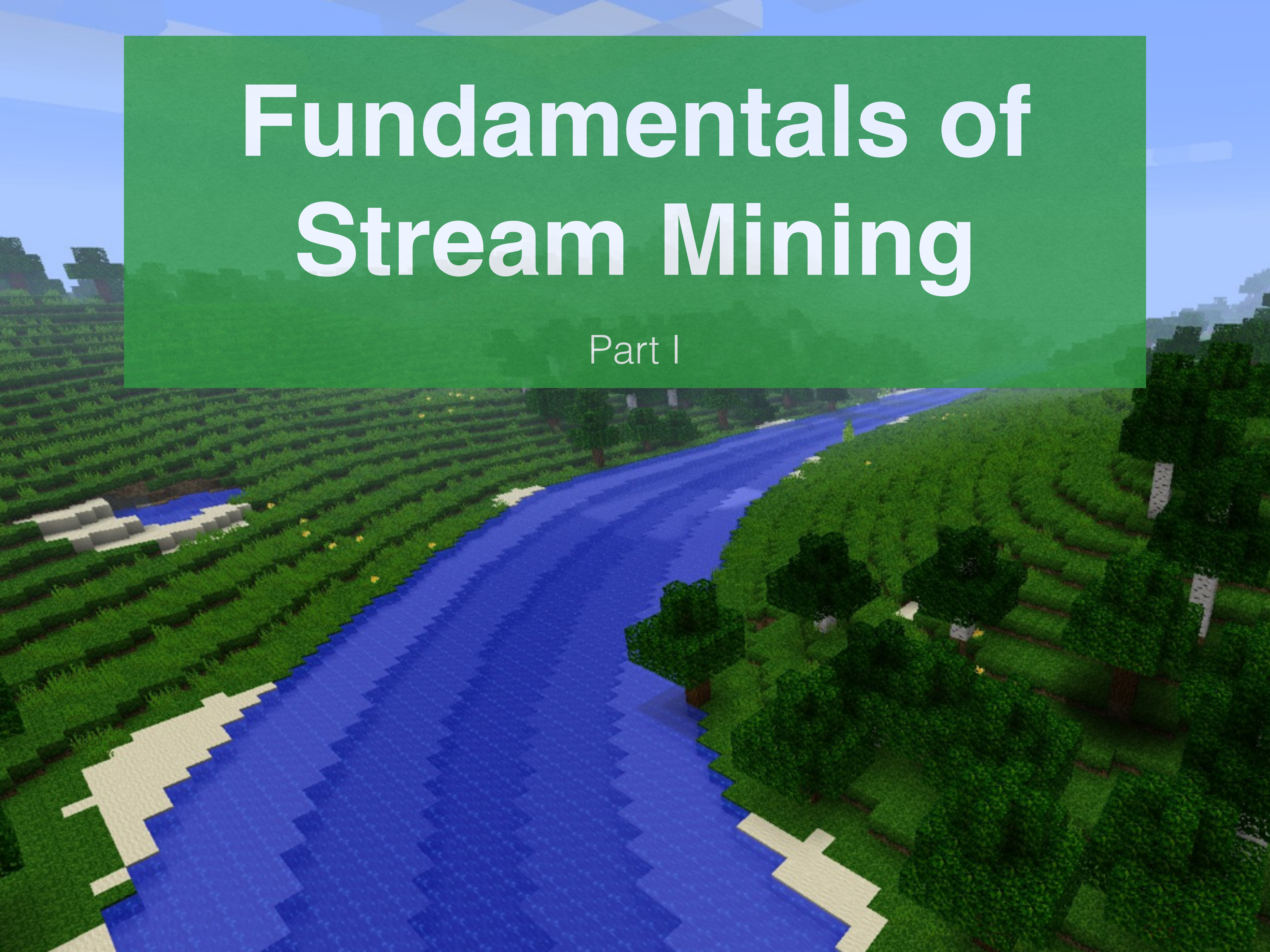
is the associate director of Huawei Noah's Ark Lab. His co-authored paper received ICDM '06 Best Application Paper Award, he led the team that used his Random Decision Tree method to win 2008 ICDM Data Mining Cup Championship. He received 2010 IBM Outstanding Technical Achievement Award for his contribution to IBM Infosphere Streams. Since he joined Huawei in August 2012, he has led his colleagues to develop Huawei Stream. SMART – a streaming platform for online and real-time processing.

http://www.weifan.info

https://sites.google.com/site/bigdatastreamminingtutorial

Outline

- **Fundamentals of Stream Mining**
  - Setting
  - Classification
  - Concept Drift
  - Regression
  - Clustering
  - Frequent Itemset Mining
- **Distributed Stream Mining**
  - Distributed Stream Processing Engines
  - Classification
  - Regression
  - Conclusions



Fundamentals of Stream Mining

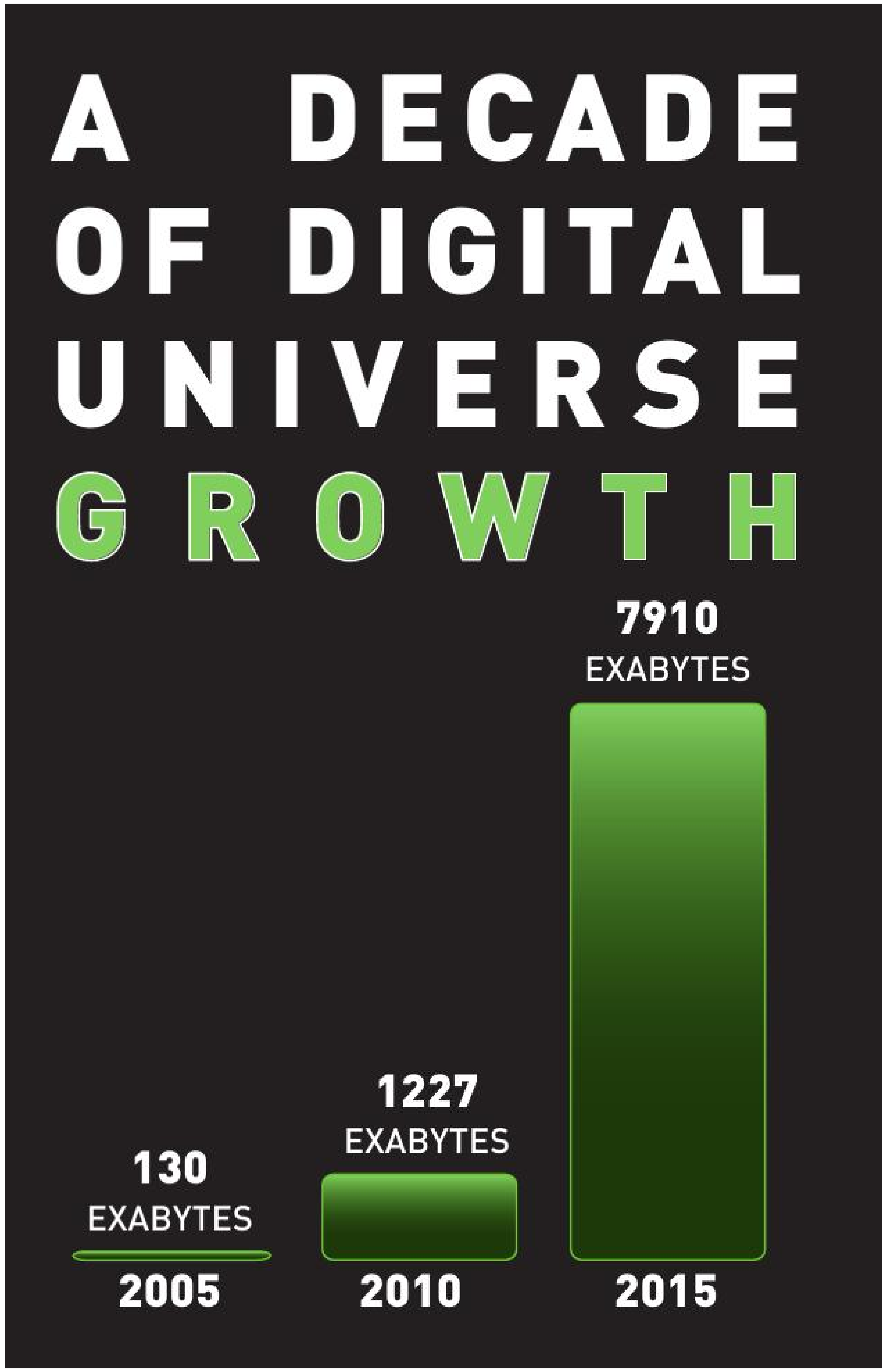
Part I

Setting

Motivation

Data is growing

Source: IDC's Digital Universe Study (EMC), 2011

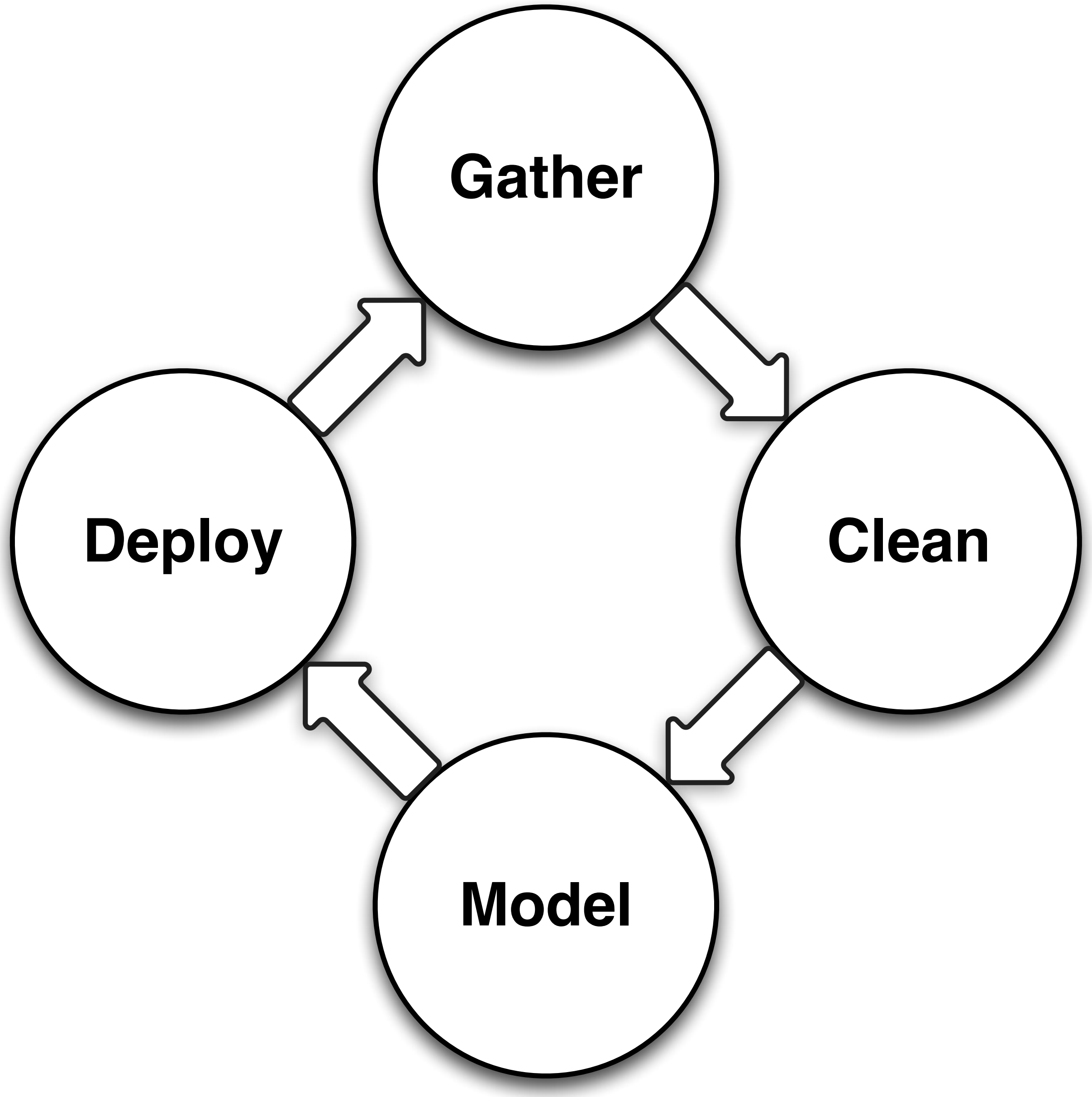


Present of Big Data

Too big to handle

“Big Data is data whose characteristics force us to look beyond the tried-and-true methods that are prevalent at that time”

– Adam Jacobs, CACM 2009 (paraphrased)

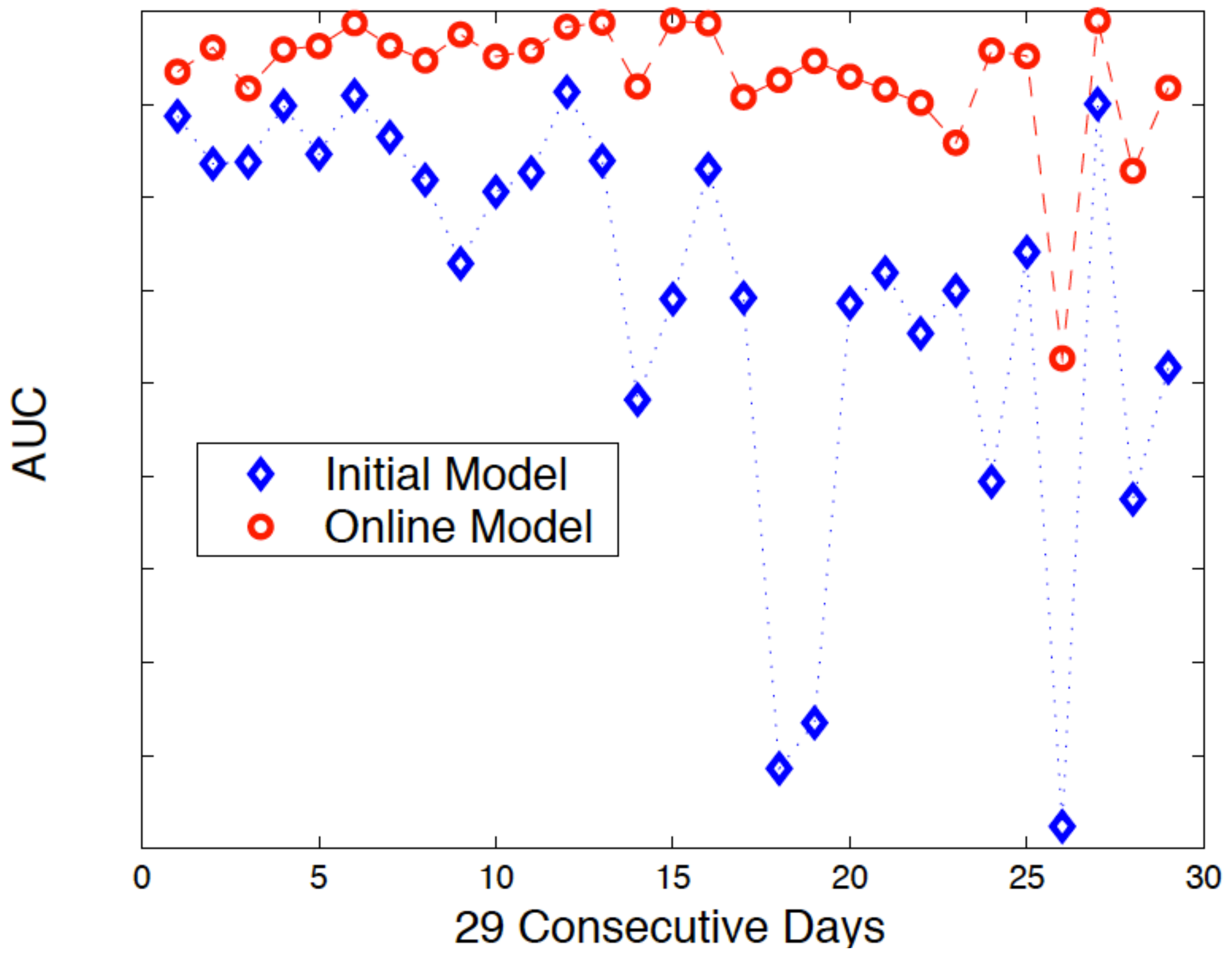


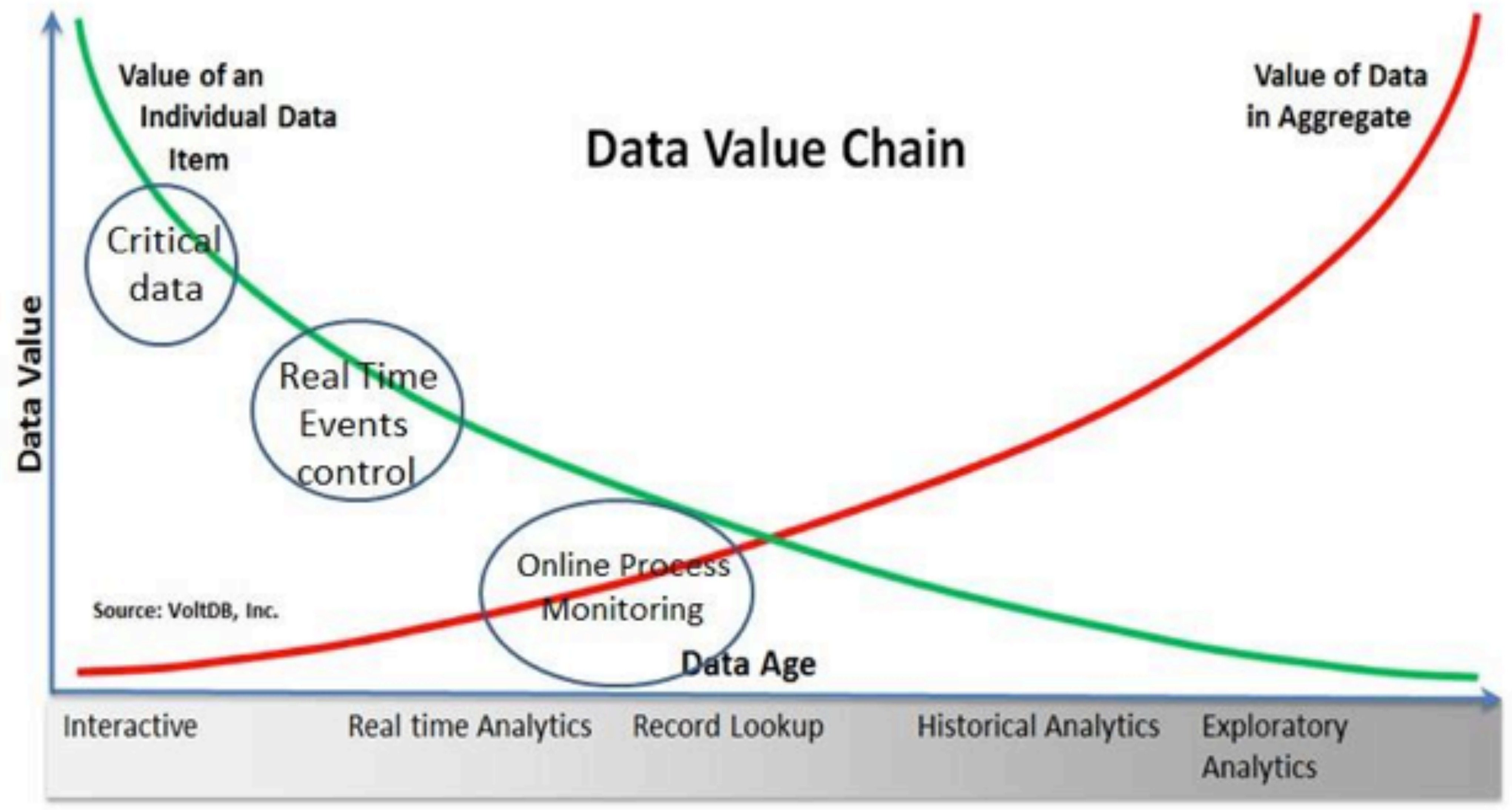
Standard Approach

Finite training sets
Static models

Pain Points

- Need to **retrain!**
  - Things change over time
  - How often?
- Data unused until next update!
  - Value of data wasted





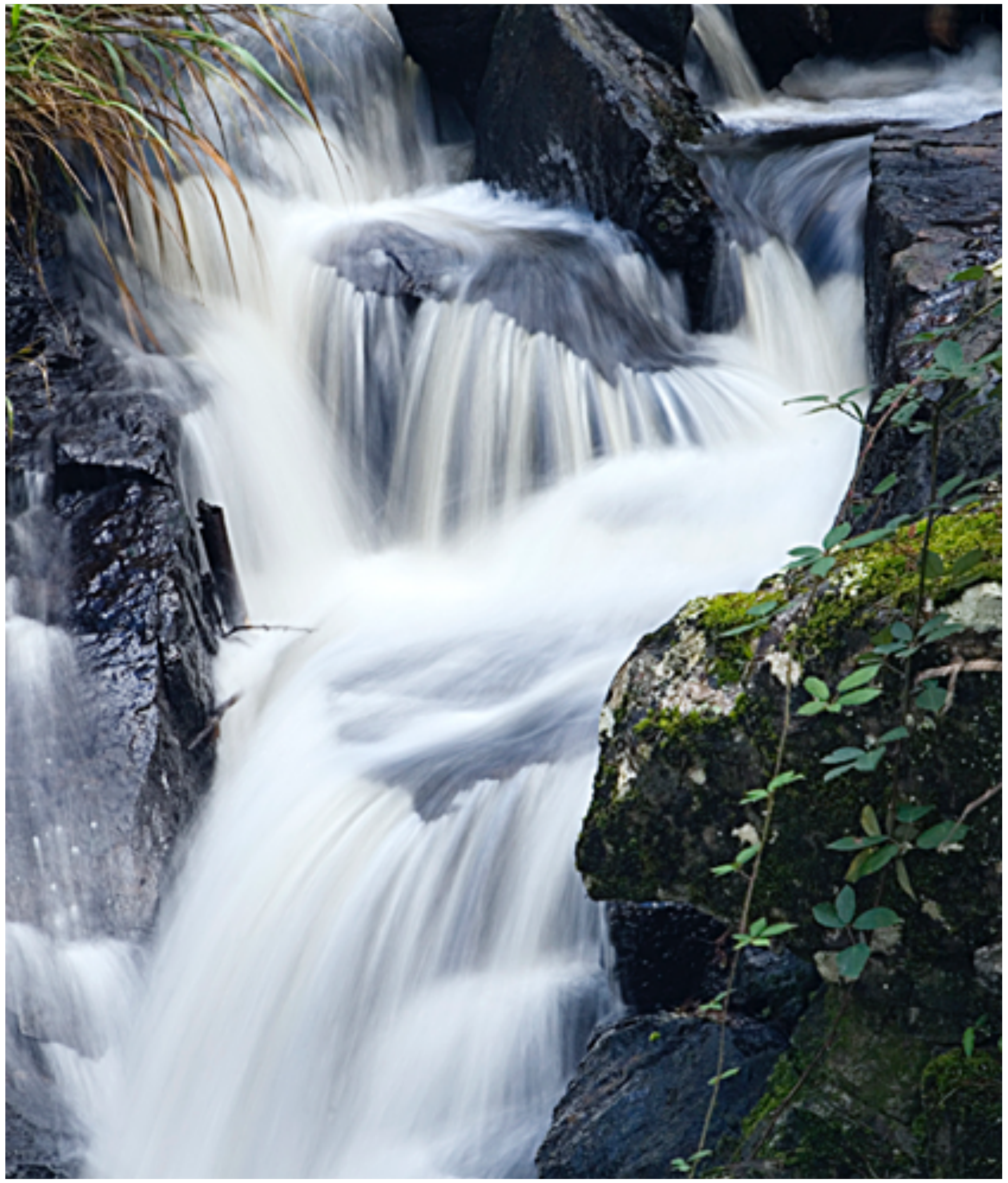
Value of Data

Online Analytics

What is happening **now**?

Stream Mining

- Maintain models online
    - Incorporate data on the fly
    - Unbounded training sets
    - Detect changes and adapts
    - Dynamic models



Big Data Streams

- Volume + Velocity (+ Variety)
- Too large for single commodity server main memory
- Too fast for single commodity server CPU
- A solution needs to be:
  - Distributed
  - Scalable

Data Sources

- User clicks
- Search queries
- News
- Emails
- Tumblr posts
- Flickr photos
- Finance stocks
- Credit card transactions
- Wikipedia edit logs
- Facebook statuses
- Twitter updates
- Name your own…

Future of Big Data

Drinking from a firehose

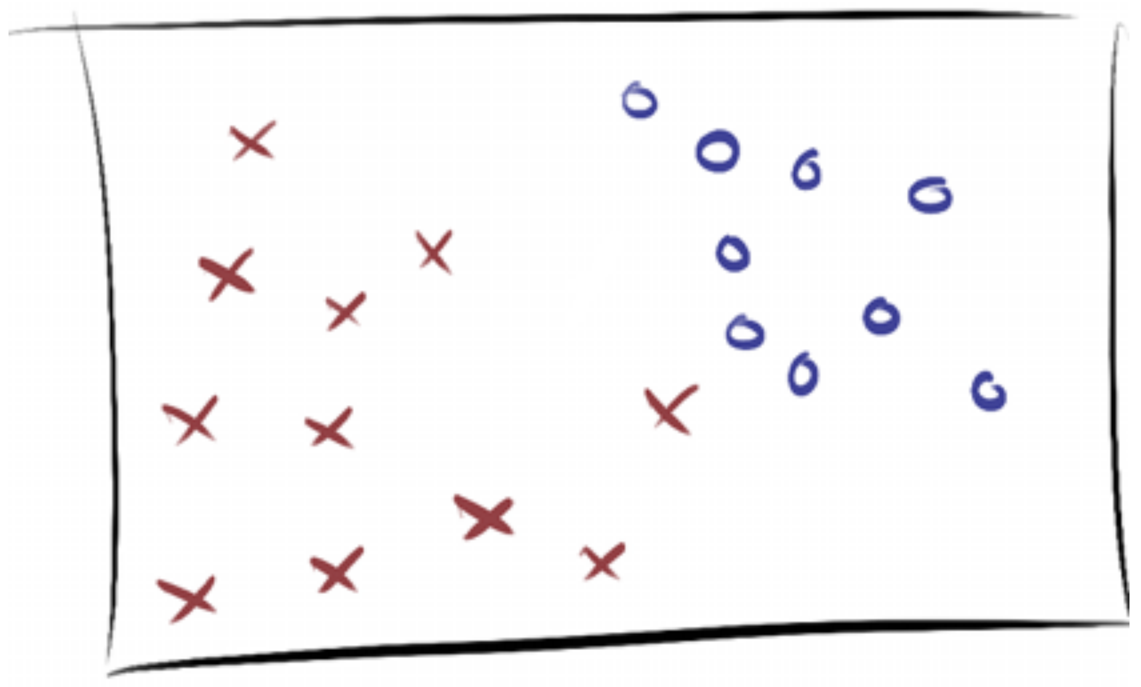
Approximation Algorithms

- General idea, good for streaming algorithms
- Small error ε with high probability $1-\delta$
    - True hypothesis H, and learned hypothesis $\hat{H}$
    - $\Pr[\ |H - \hat{H}| < \varepsilon|H|\ ] > 1-\delta$

Classification

Definition

Given a set of training examples belonging to n_C different classes, a classifier algorithm builds a model that predicts for every unlabeled instance x the class C to which it belongs



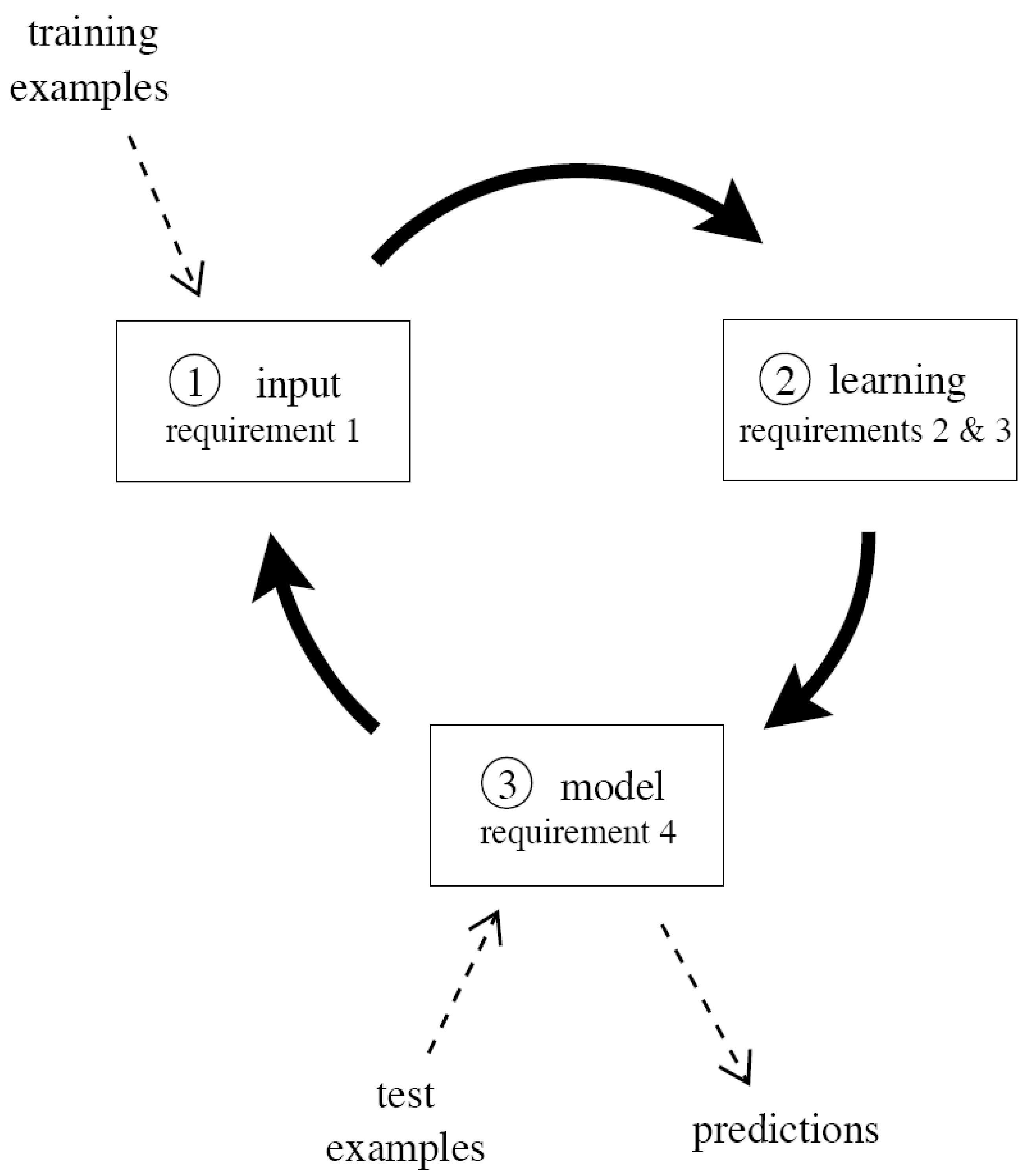
Examples

- Email spam filter
- Twitter sentiment analyzer

Photo: Stephen Merity http://smerity.com

Process

- One example at at time, used at most once
- Limited memory
- Limited time
- Anytime prediction



Naïve Bayes

- Based on Bayes' theorem

$$P(C|x) = \frac{P(x|C)P(C)}{P(x)}$$

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}$$

- Probability of observing feature x_i given class C

$$P(C|x) \propto \prod_{x_i \in x} P(x_i|C)P(C)$$

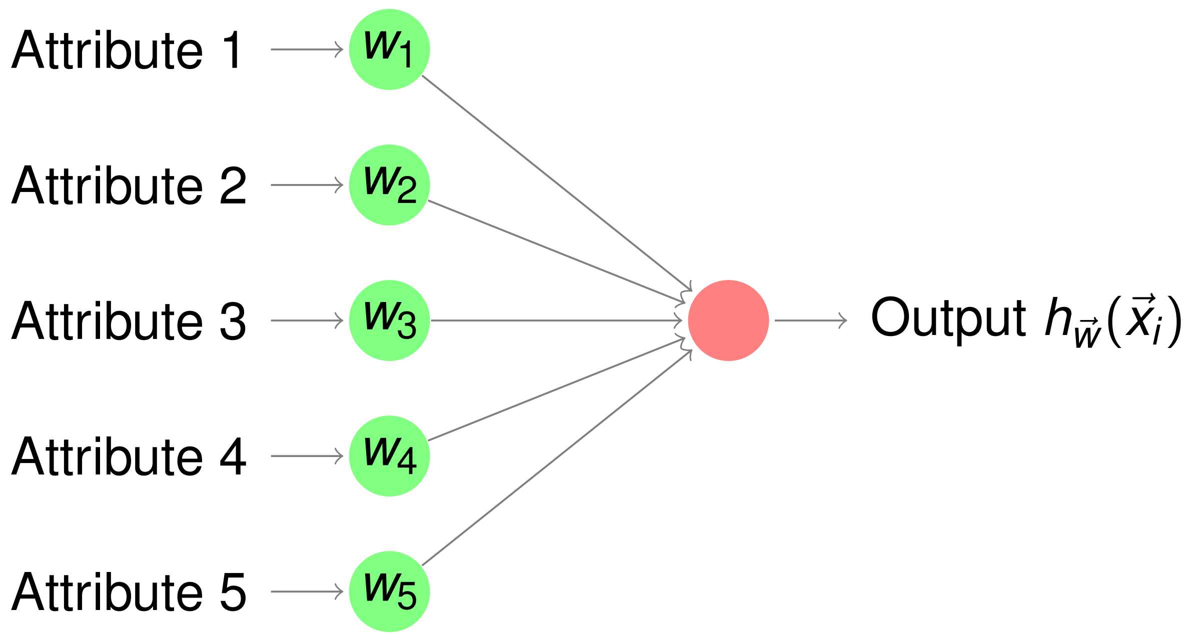
- Prior class probability P(C)

- Just counting!

$$C = \arg\max_C P(C|x)$$

Perceptron

- Linear classifier
- Data stream: $\langle \vec{\mathbf{x}}_i, y_i \rangle$
- $\tilde{y}_i = h_{\vec{w}}(\vec{\mathbf{x}}_i) = \sigma(\vec{\mathbf{w}}_i^T \vec{\mathbf{x}}_i)$
- $\sigma(x) = 1/(1+e^{-x})$ $\sigma' = \sigma(x)(1-\sigma(x))$
- Minimize MSE $J(\vec{\mathbf{w}}) = ½\sum(y_i - \tilde{y}_i)^2$
- SGD $\vec{\mathbf{w}}_{i+1} = \vec{\mathbf{w}}_i - \eta \nabla J \, \vec{\mathbf{x}}_i$
  - $\nabla J = -(y_i - \tilde{y}_i)\tilde{y}_i(1-\tilde{y}_i)$
  - $\vec{\mathbf{w}}_{i+1} = \vec{\mathbf{w}}_i + \eta(y_i - \tilde{y}_i)\tilde{y}_i(1-\tilde{y}_i)\vec{\mathbf{x}}_i$



Perceptron Learning

```
PERCEPTRON LEARNING($Stream, \eta$)
1  for each class
2      do PERCEPTRON LEARNING($Stream, class, \eta$)
```

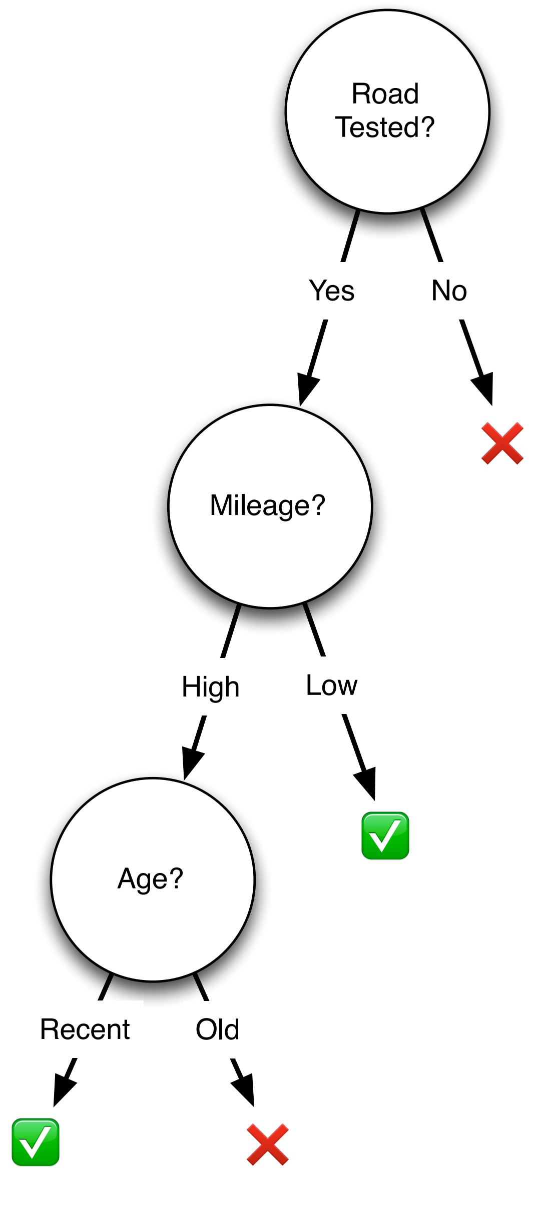
```
PERCEPTRON LEARNING($Stream, class, \eta$)
1  ▷ Let $w_0$ and $\vec{w}$ be randomly initialized
2  for each example $(\vec{x}, y)$ in Stream
3      do if $class = y$
4            then $\delta = (1 - h_{\vec{w}}(\vec{x})) \cdot h_{\vec{w}}(\vec{x}) \cdot (1 - h_{\vec{w}}(\vec{x}))$
5            else $\delta = (0 - h_{\vec{w}}(\vec{x})) \cdot h_{\vec{w}}(\vec{x}) \cdot (1 - h_{\vec{w}}(\vec{x}))$
6          $\vec{w} = \vec{w} + \eta \cdot \delta \cdot \vec{x}$
```

```
PERCEPTRON PREDICTION($\vec{x}$)
1  return $\arg\max_{class} h_{\vec{w}_{class}}(\vec{x})$
```

Decision Tree

- Each node tests a features
- Each branch represents a value
- Each leaf assigns a class
- Greedy recursive induction
  - Sort all examples through tree
  - x_i = most discriminative attribute
  - New node for x_i, new branch for each value, leaf assigns majority class
  - Stop if no error | limit on #instances

Car deal?



Very Fast Decision Tree

Pedro Domingos, Geoff Hulten: “Mining high-speed data streams”. KDD ’00

- AKA, Hoeffding Tree
- A small sample can often be enough to choose a near optimal decision
- Collect sufficient statistics from a small set of examples
- Estimate the merit of each alternative attribute
- Choose the sample size that allows to differentiate between the alternatives

Leaf Expansion

- When should we expand a leaf?
- Let x_1 be the most informative attribute, x_2 the second most informative one
- Is x_1 a stable option?
- Hoeffding bound
  - Split if $G(x_1) - G(x_2) > \varepsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}}$

HT Induction

HT Induction

```
HT($Stream, \delta$)
1   $\triangleright$ Let HT be a tree with a single leaf(root)
2   $\triangleright$ Init counts $n_{ijk}$ at root
3   for each example $(x, y)$ in Stream
4         do HTGROW($(x, y), HT, \delta$)
```

HT Induction

```
HT($Stream, \delta$)
1  $\triangleright$ Let HT be a tree with a single leaf(root)
2  $\triangleright$ Init counts $n_{ijk}$ at root
3  for each example $(x, y)$ in Stream
4        do HTGROW($(x, y), HT, \delta$)
```

```
HTGROW($(x, y), HT, \delta$)
1  $\triangleright$ Sort $(x, y)$ to leaf $l$ using $HT$
2  $\triangleright$ Update counts $n_{ijk}$ at leaf $l$
3  if examples seen so far at $l$ are not all of the same class
4      then $\triangleright$ Compute $G$ for each attribute
5             if $G$(Best Attr.)$-G$(2nd best) $> \sqrt{\frac{R^2 \ln 1/\delta}{2n}}$
6               then $\triangleright$ Split leaf on best attribute
7                    for each branch
8                        do $\triangleright$ Start new leaf and initiliatize counts
```

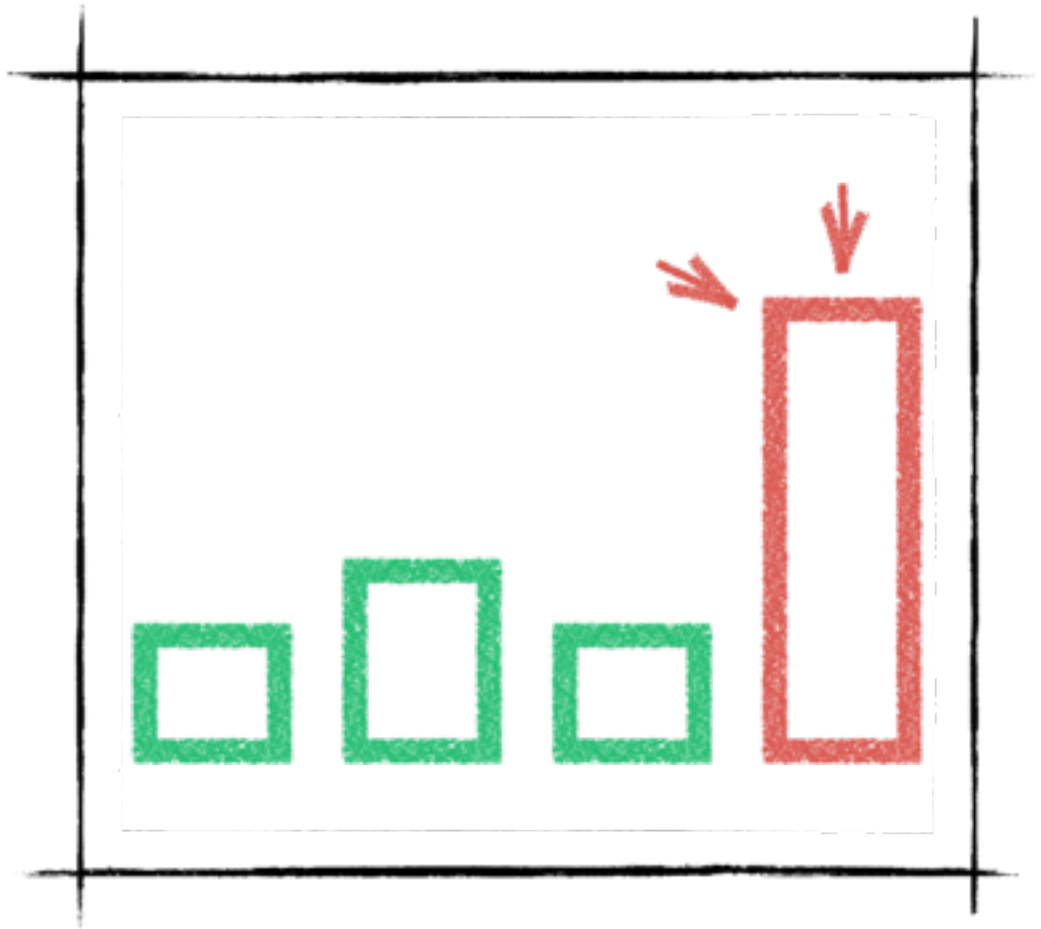
Properties

- Number of examples to expand node depends only on Hoeffding bound (ε decreases with $\sqrt{n}$)
- Low variance model (stable decisions with statistical support)
- Low overfitting (examples processed only once, no need for pruning)
- Theoretical guarantees on error rate with high probability
    - Hoeffding algorithms asymptotically close to batch learner. Expected disagreement δ/p (p = probability instance falls into a leaf)
- Ties: broken when $\varepsilon < \tau$ even if $\Delta G < \varepsilon$

Concept Drift

Definition

Given an input sequence $\langle x_1, x_2, \ldots, x_t \rangle$, output at instant *t* an alarm signal if there is a distribution change, and a prediction $\hat{x}_{t+1}$ minimizing the error $|\hat{x}_{t+1} - x_{t+1}|$



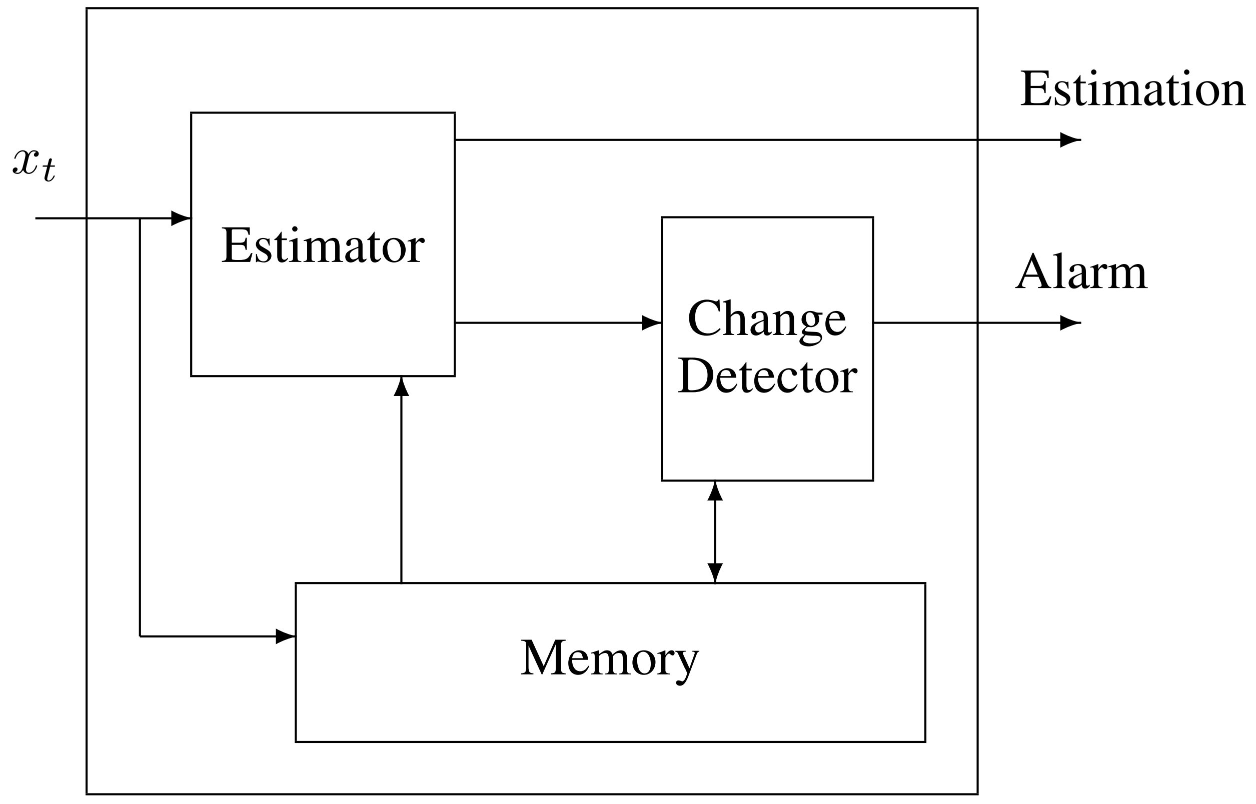
Outputs

- Alarm indicating change
- Estimate of parameter

Photo: http://www.logsearch.io

Application

- Change detection on evaluation of model
- Training error should decrease with more examples
    - Change in distribution of training error
    - Input = stream of real/binary numbers
- Trade-off between detecting true changes and avoiding false alarms



Cumulative Sum

- Alarm when mean of input data differs from zero
- Memoryless heuristic (no statistical guarantee)
- Parameters: threshold h, drift speed v
- $g_0 = 0, \quad g_t = \max(0, g_{t-1} + \varepsilon_t - v)$
- if $g_t > h$ then alarm; $g_t = 0$

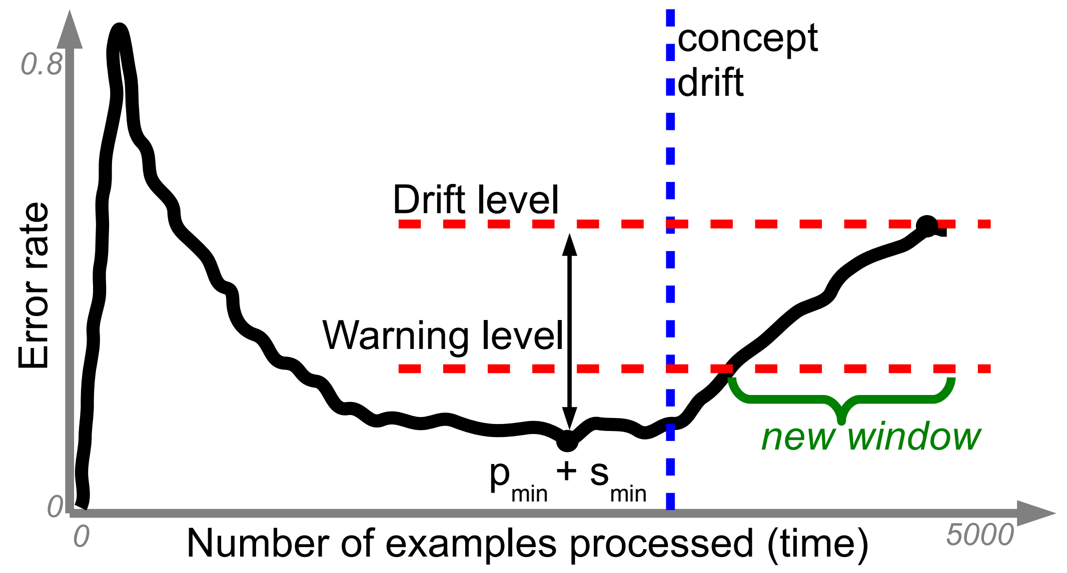
Page-Hinckley Test

- Similar structure to Cumulative Sum
- $g_0 = 0, \quad g_t = g_{t-1} + (\varepsilon_t - v)$
- $G_t = \min_t(g_t)$
- if $g_t - G_t > h$ then alarm; $g_t = 0$

Statistical Process Control

J Gama, P. Medas, G. Castillo, P. Rodrigues: “Learning with Drift Detection”. SBIA '04

- Monitor error in sliding window
- Null hypothesis:
  no change between windows
- If error > warning level
  learn in parallel new model
  on the current window
- if error > drift level
  substitute new model for old



Concept-adapting VFDT

G. Hulten, L. Spencer, P. Domingos: “Mining Time-Changing Data Streams”. KDD ‘01

- Model consistent with sliding window on stream
- Keep sufficient statistics also at internal nodes
  - Recheck periodically if splits pass Hoeffding test
  - If test fails, grow alternate subtree and swap-in when accuracy of alternate is better
- Processing updates O(1) time, +O(W) memory
  - Increase counters for incoming instance, decrease counters for instance going out window

VFDTc: Adapting to Change

J. Gama, R. Fernandes, R. Rocha: “Decision Trees for Mining Data Streams”. IDA (2006)

- Monitor error rate
- When drift is detected
    - Start learning alternative subtree in parallel
- When accuracy of alternative is better
    - Swap subtree
- No need for window of instances

Hoeffding Adaptive Tree

A. Bifet, R. Gavaldà: “Adaptive Parameter-free Learning from Evolving Data Streams” IDA (2009)

- Replace frequency counters by estimators
    - No need for window of instances
    - Sufficient statistics kept by estimators separately
- Parameter-free change detector + estimator with theoretical guarantees for subtree swap (ADWIN)
    - Keeps sliding window consistent with “no-change hypothesis”

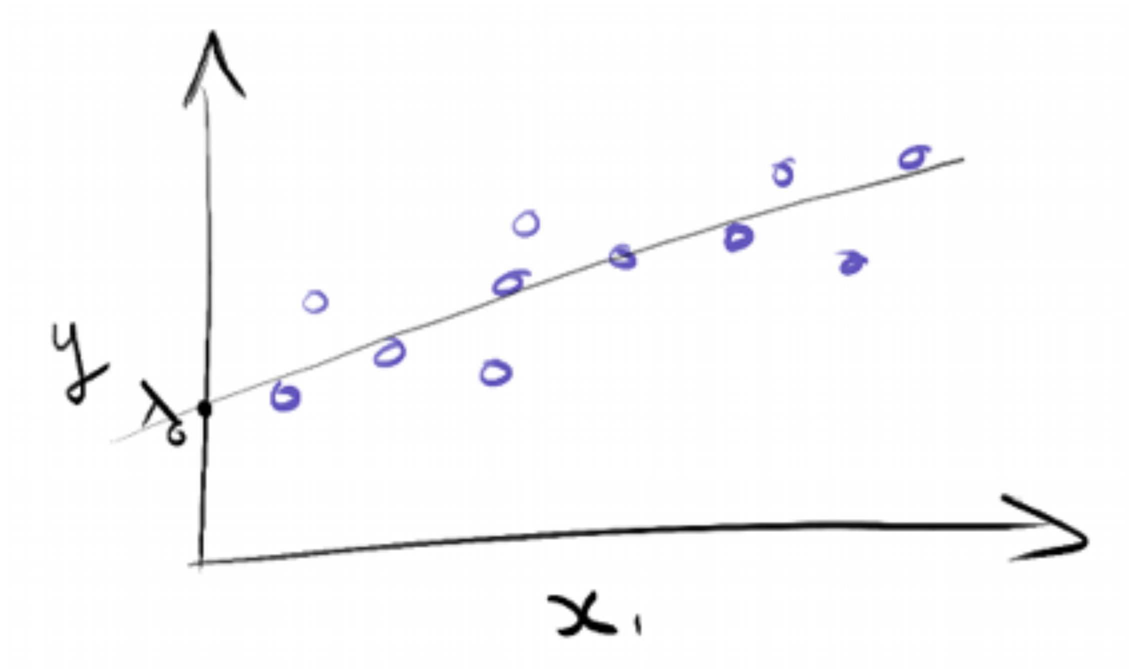
A. Bifet, R. Gavaldà: “Learning from Time-Changing Data with Adaptive Windowing”. SDM ‘07

Regression

Definition

Given a set of training examples with a numeric label, a regression algorithm builds a model that predicts for every unlabeled instance x the value with high accuracy

$$y = f(x)$$



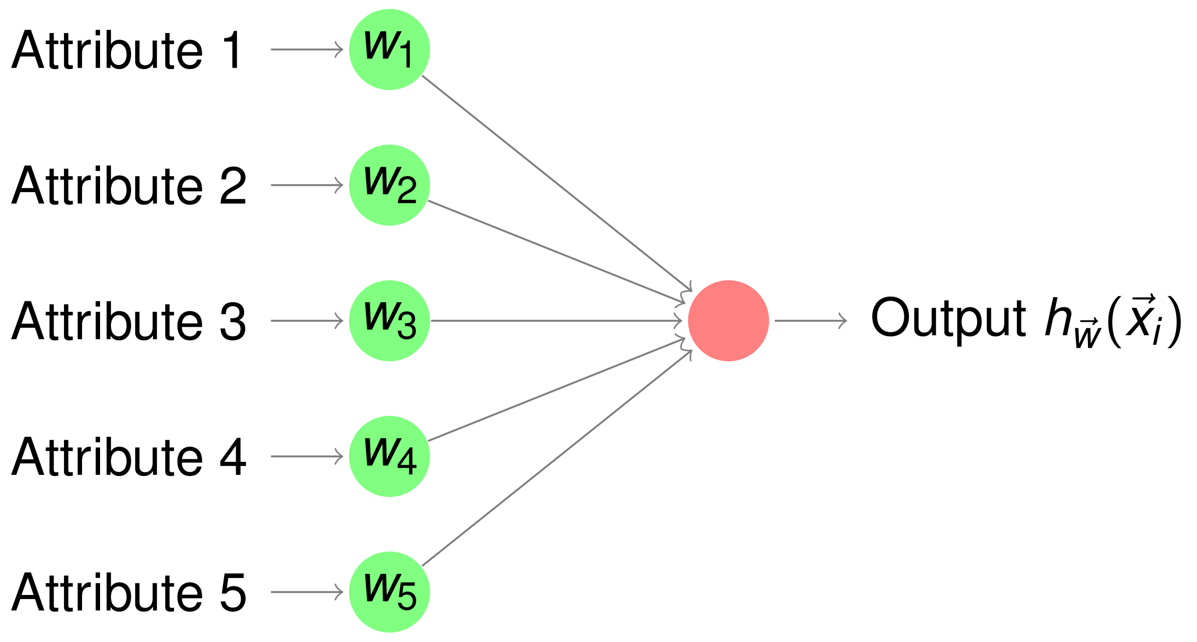
Examples

- Stock price
- Airplane delay

Photo: Stephen Merity http://smerity.com

Perceptron

- Linear regressor
- Data stream: $\langle \vec{\mathbf{x}}_i, y_i \rangle$
- $\tilde{y}_i = h_{\vec{w}}(\vec{\mathbf{x}}_i) = \vec{\mathbf{w}}^T \vec{\mathbf{x}}_i$
- Minimize MSE $J(\vec{\mathbf{w}})=\frac{1}{2}\sum(y_i-\tilde{y}_i)^2$
- SGD $\vec{\mathbf{w}}' = \vec{\mathbf{w}} - \eta \nabla J \, \vec{\mathbf{x}}_i$
    - $\nabla J = -(y_i-\tilde{y}_i)$
    - $\vec{\mathbf{w}}' = \vec{\mathbf{w}} + \eta(y_i-\tilde{y}_i)\vec{\mathbf{x}}_i$



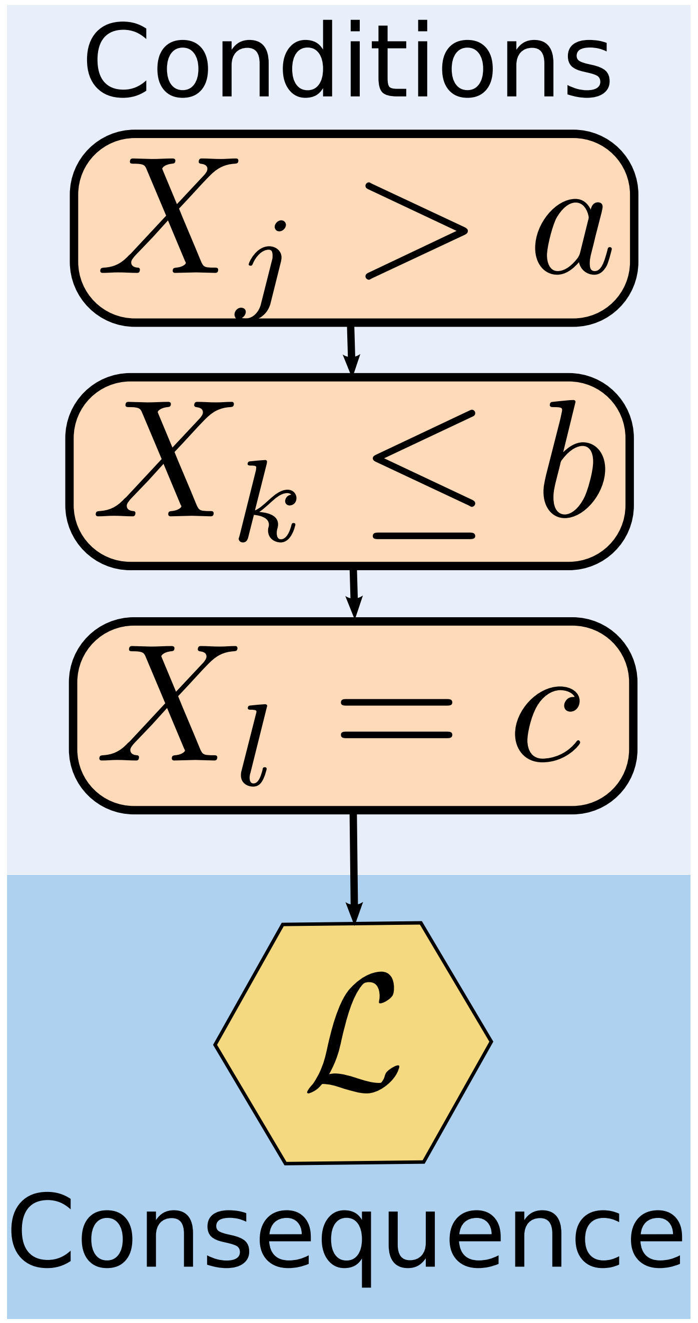
Regression Tree

- Same structure as decision tree
- Predict = average target value or linear model at leaf (vs majority)
- Gain = reduction in standard deviation (vs entropy)

$$\sigma = \sqrt{\sum (\tilde{y}_i - y_i)^2/(N-1)}$$

Rules

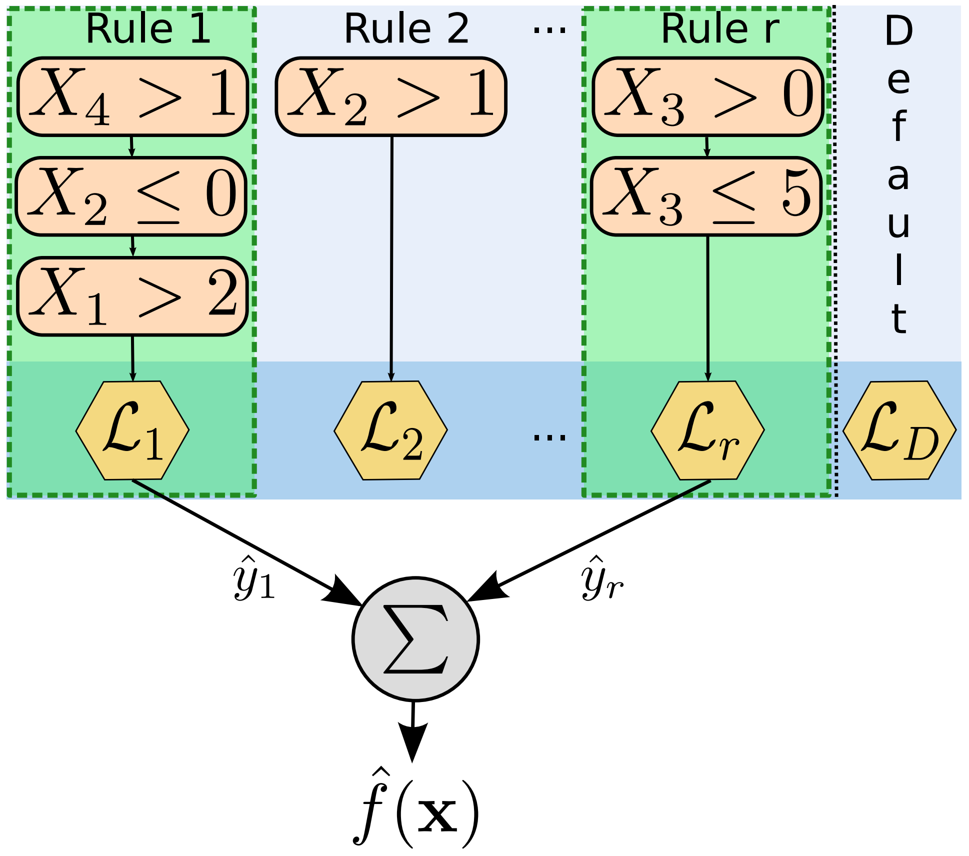
- Problem: very large decision trees have context that is complex and hard to understand
- Rules: self-contained, modular, easier to interpret, no need to cover universe
- $\mathcal{L}$ keeps sufficient statistics to:
    - make predictions
    - expand the rule
    - detect changes and anomalies



Adaptive Model Rules

E. Almeida, C. Ferreira, J. Gama. "Adaptive Model Rules from Data Streams." ECML-PKDD ‘13

- Ruleset: ensemble of rules
- Rule prediction: mean, linear model
- Ruleset prediction
    - Weighted avg. of predictions of rules covering instance x
    - Weights inversely proportional to error
    - Default rule covers uncovered instances



E.g: $\mathbf{x} = [4, -1, 1, 2]$

$$\hat{f}(\mathbf{x}) = \sum_{R_l \in S(\mathbf{x}_i)} \theta_l \hat{y}_l,$$

AMRules Induction

- Rule creation: default rule expansion
- Rule expansion: split on attribute maximizing σ reduction
    - Hoeffding bound $\varepsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}}$
    - Expand when $\sigma_{1st}/\sigma_{2nd} < 1 - \varepsilon$
- Evict rule when P-H test error large
- Detect and explain local anomalies

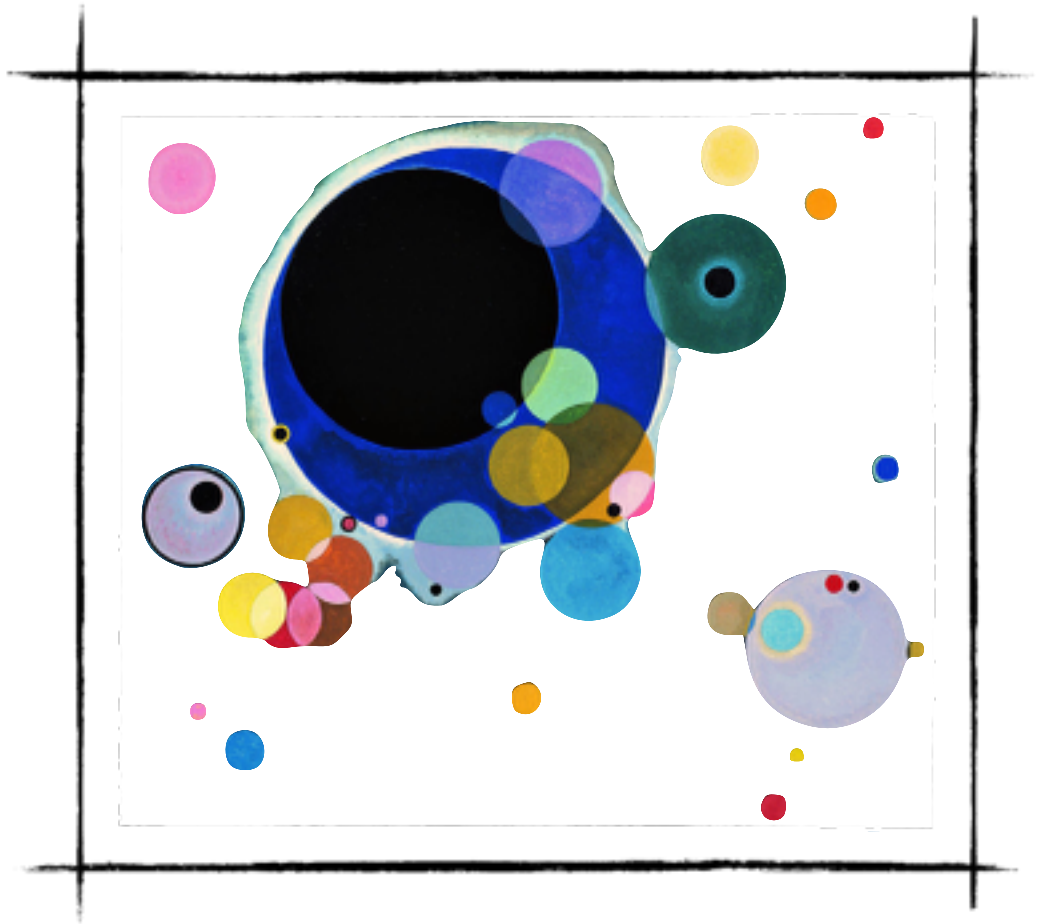
Algorithm 1: Training AMRules

```
Input:  S: Stream of examples
begin
    $\mathcal{R} \leftarrow \{\}$, $D \leftarrow 0$
    foreach $(\mathbf{x}, y) \in S$ do
        foreach Rule $r \in S(\mathbf{x})$ do
            if $\neg$IsAnomaly($\mathbf{x}, r$) then
                if PHTest($error_r$, $\lambda$) then
                    Remove the rule from $\mathcal{R}$
                else
                    Update sufficient statistics $\mathcal{L}_r$
                    ExpandRule($r$)
        if $S(\mathbf{x}) = \emptyset$ then
            Update $\mathcal{L}_D$
            ExpandRule($D$)
            if $D$ expanded then
                $\mathcal{R} \leftarrow \mathcal{R} \cup D$
                $D \leftarrow 0$
    return ($\mathcal{R}$, $\mathcal{L}_D$)
```

Clustering

Definition

Given a set of unlabeled instances, distribute them into homogeneous groups according to some common relations or affinities.



Examples

- Market segmentation
- Social network communities

Photo: W. Kandinsky - Several Circles (edited)

Approaches

- Distance based (CluStream)
- Density based (DenStream)
- Kernel based, Coreset based, much more…
- Most approaches combine online + offline phase
- Formally: minimize cost function
  over a partitioning of the data

Static Evaluation

- Internal (validation)
  - Sum of squared distance (point to centroid)
  - Dunn index (on distance d)
    D = min(inter-cluster d) / max(intra-cluster d)
- External (ground truth)
  - Rand = #agreements / #choices = 2(TP+TN)/(N(N-1))
  - Purity = #majority class per cluster / N

Streaming Evaluation

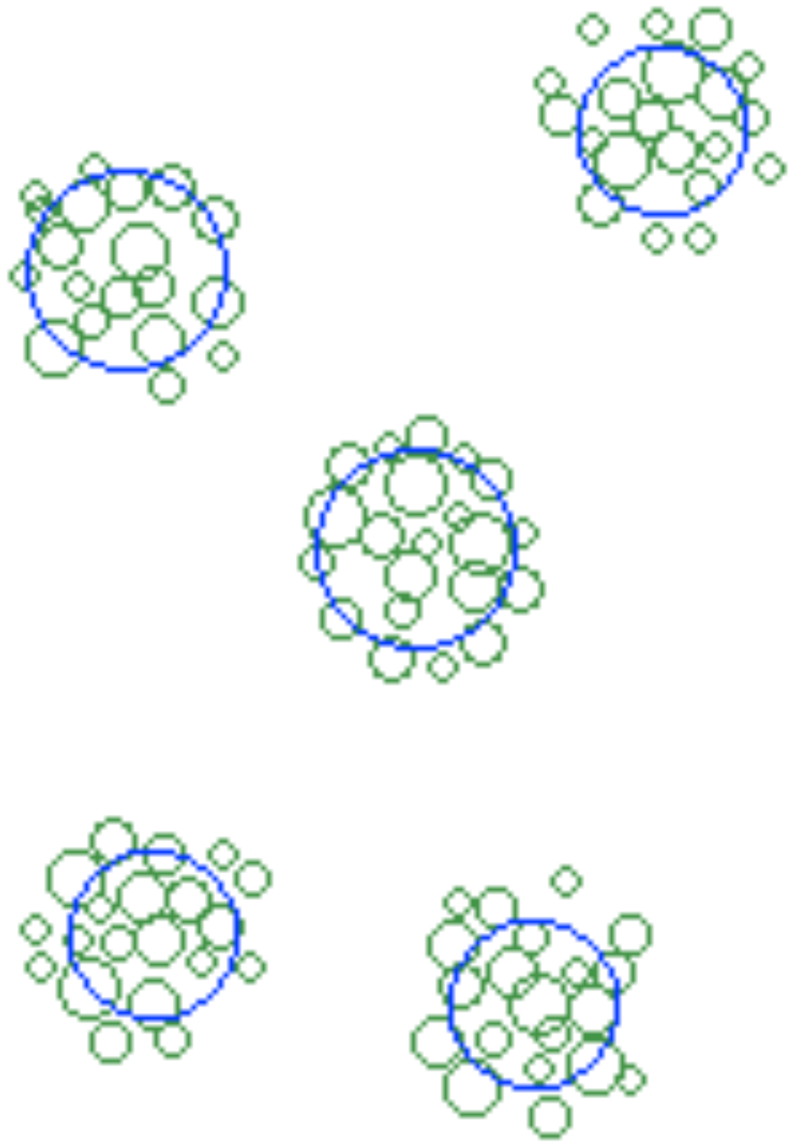
H. Kremer, P. Kranen, T. Jansen, T. Seidl, A. Bifet, G. Holmes, B. Pfahringer:
"An effective evaluation measure for clustering on evolving data streams". KDD '11

- Clusters may: appear, fade, move, merge
  - Missed points (unassigned)
  - Misplaced points (assigned to different cluster)
  - Noise
- Cluster Mapping Measure CMM
  - External (ground truth)
  - Normalized sum of penalties of these errors

Micro-Clusters

Tian Zhang, Raghu Ramakrishnan, Miron Livny: “BIRCH: An Efficient Data Clustering Method for Very Large Databases”. SIGMOD ’96

- AKA, Cluster Features CF
  Statistical summary structure
- Maintained in online phase,
  input for offline phase
- Data stream $\langle \vec{x}_i \rangle$, d dimensions
- Cluster feature vector
  N: number of points
  LS_j: sum of values (for dim. j)
  SS_j: sum of squared values (for dim. j)
- Easy to update, easy to merge
- # of micro-clusters ≫ # of clusters



CluStream

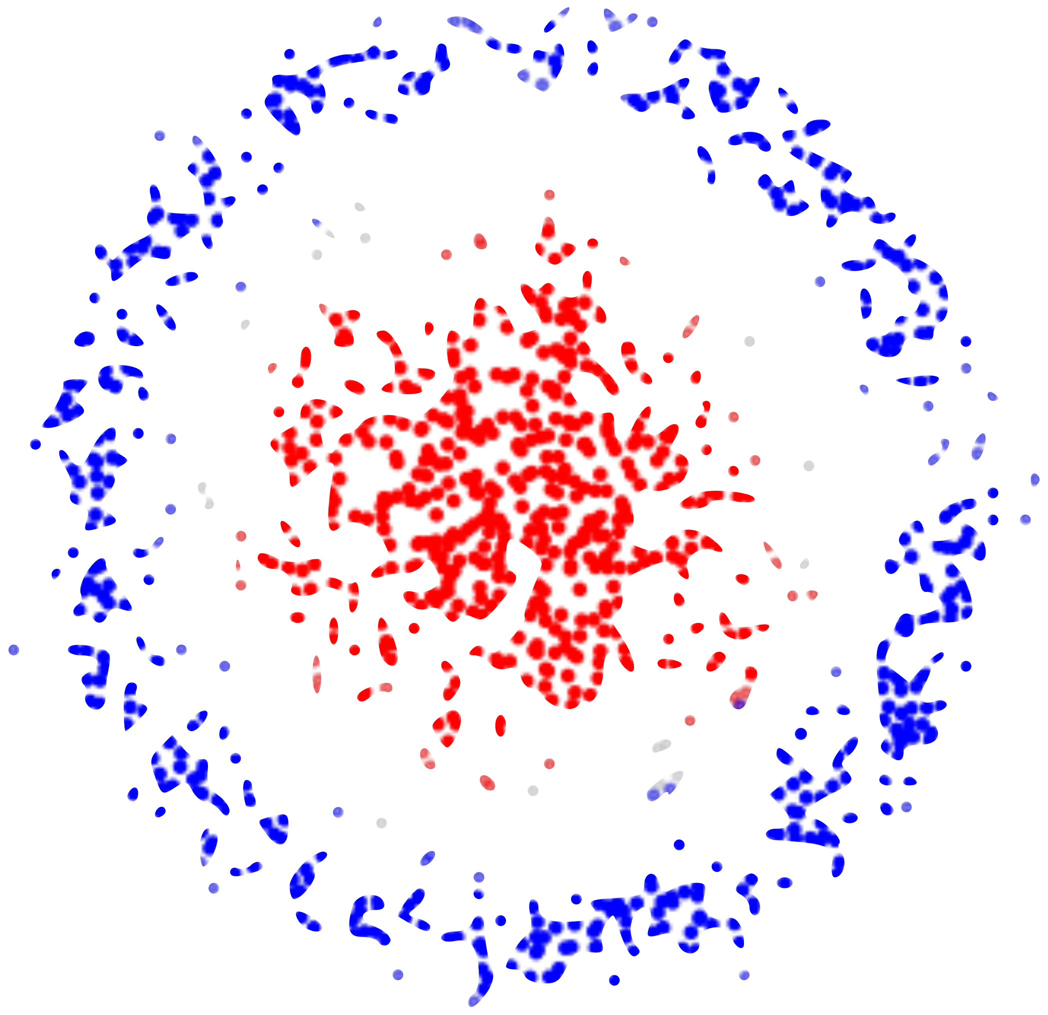
Charu C. Aggarwal, Jiawei Han, Jianyong Wang, Philip S. Yu: “A Framework for Clustering Evolving Data Streams”. VLDB ‘03

- Timestamped data stream $\langle t_i, \vec{x}_i \rangle$, represented in d+1 dimensions
- Seed algorithm with q micro-clusters (k-means on initial data)
- Online phase. For each new point, either:
    - Update one micro-cluster (point within maximum boundary)
    - Create a new micro-cluster (delete/merge other micro-clusters)
- Offline phase. Determine k macroclusters on demand:
    - K-means on micro-clusters (weighted pseudo-points)
    - Time-horizon queries via pyramidal snapshot mechanism

DBSCAN

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu: “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”. KDD ‘96

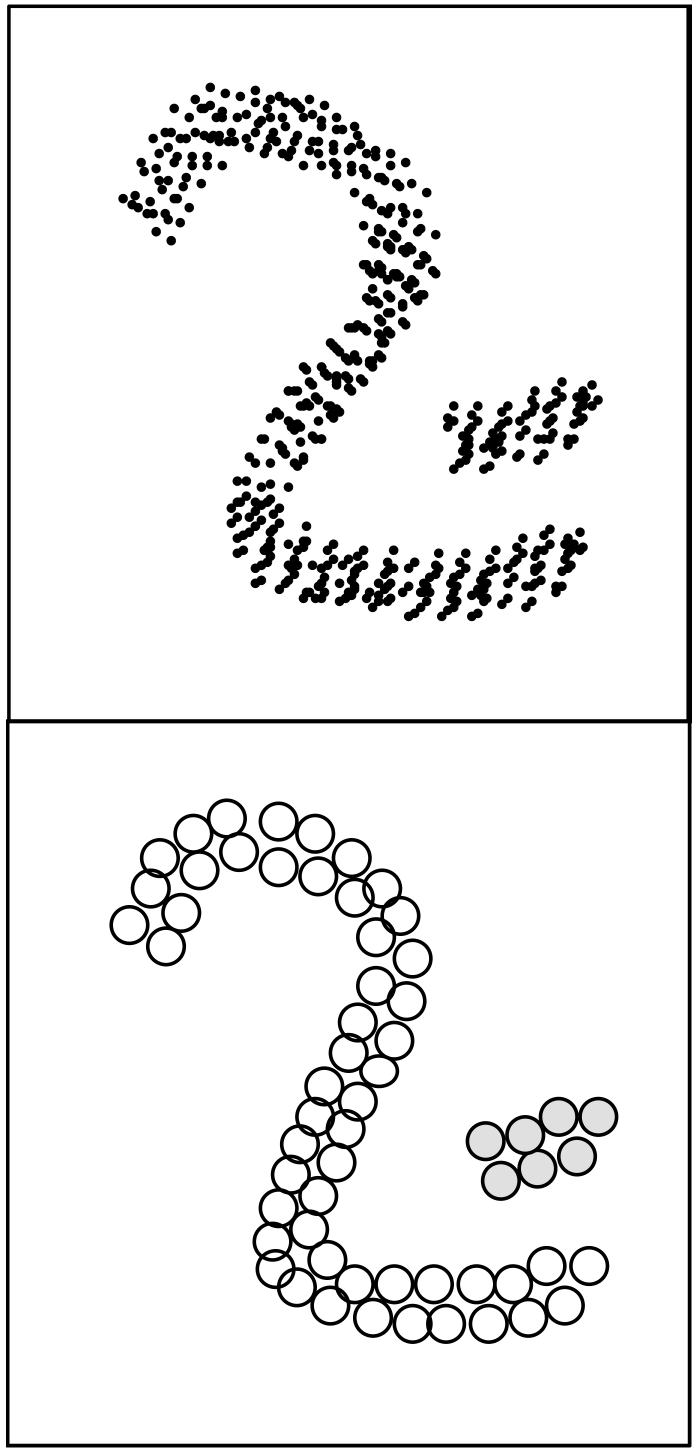
- ε-n(p) = set of points at distance $\leq \varepsilon$
- Core object q = ε-n(q) has weight $\geq \mu$
- p is directly density-reachable from q
  - p $\in$ ε-n(q) $\wedge$ q is a core object
- p_n is density-reachable from p_1
  - chain of points $p_1,\ldots,p_n$ such that p_{i+1} is directly d-r from p_i
- Cluster = set of points that are mutually density-connected



DenStream

Feng Cao, Martin Ester, Weining Qian, Aoying Zhou: “Density-Based Clustering over an Evolving Data Stream with Noise”. SDM ‘06

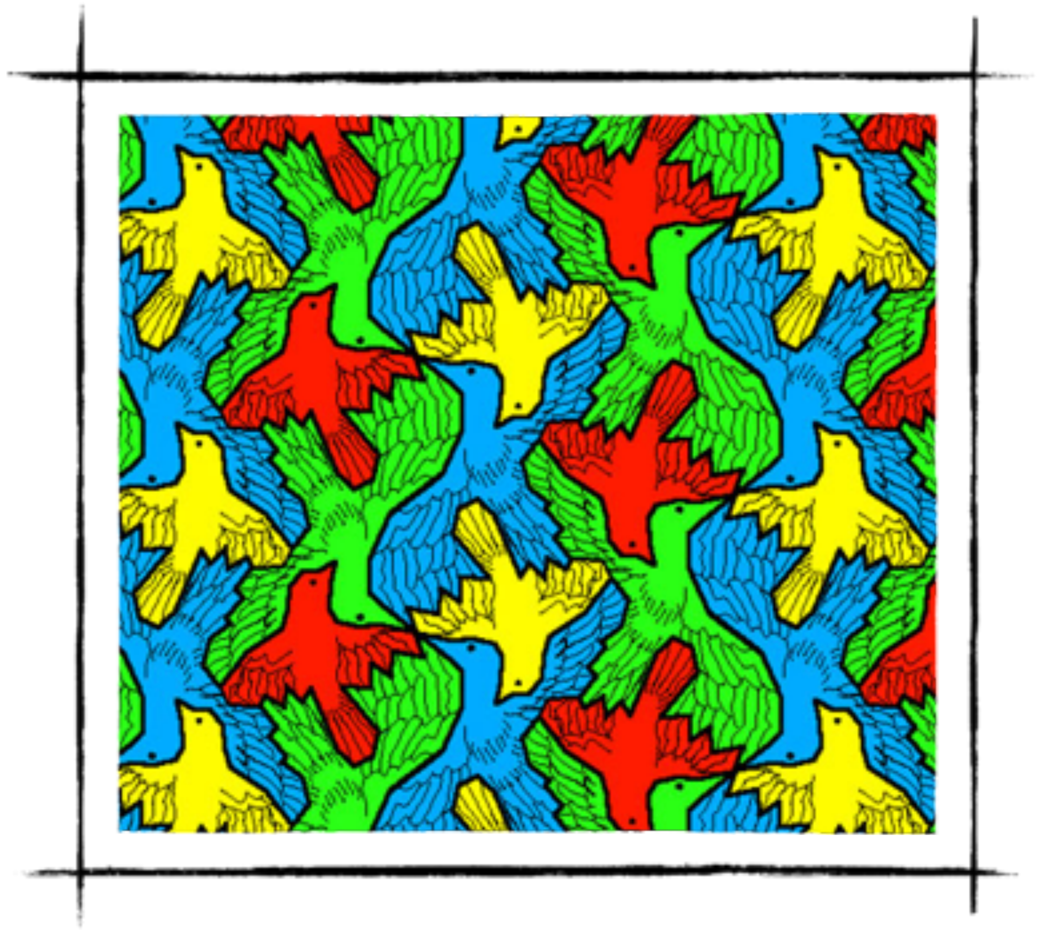
- Based on DBSCAN
- Core-micro-cluster: CMC(w,c,r)
  weight w > μ, center c, radius r < ε
- Potential/outlier micro-clusters
- Online: merge point into p (or o)
  micro-cluster if new radius r'< ε
  - Promote outlier to potential if w > βμ
- Else create new o-micro-cluster
- Offline: DBSCAN



Frequent Itemset Mining

Definition

Given a collection of sets of items, find all the subsets that occur frequently, i.e., more than a minimum *support* of times

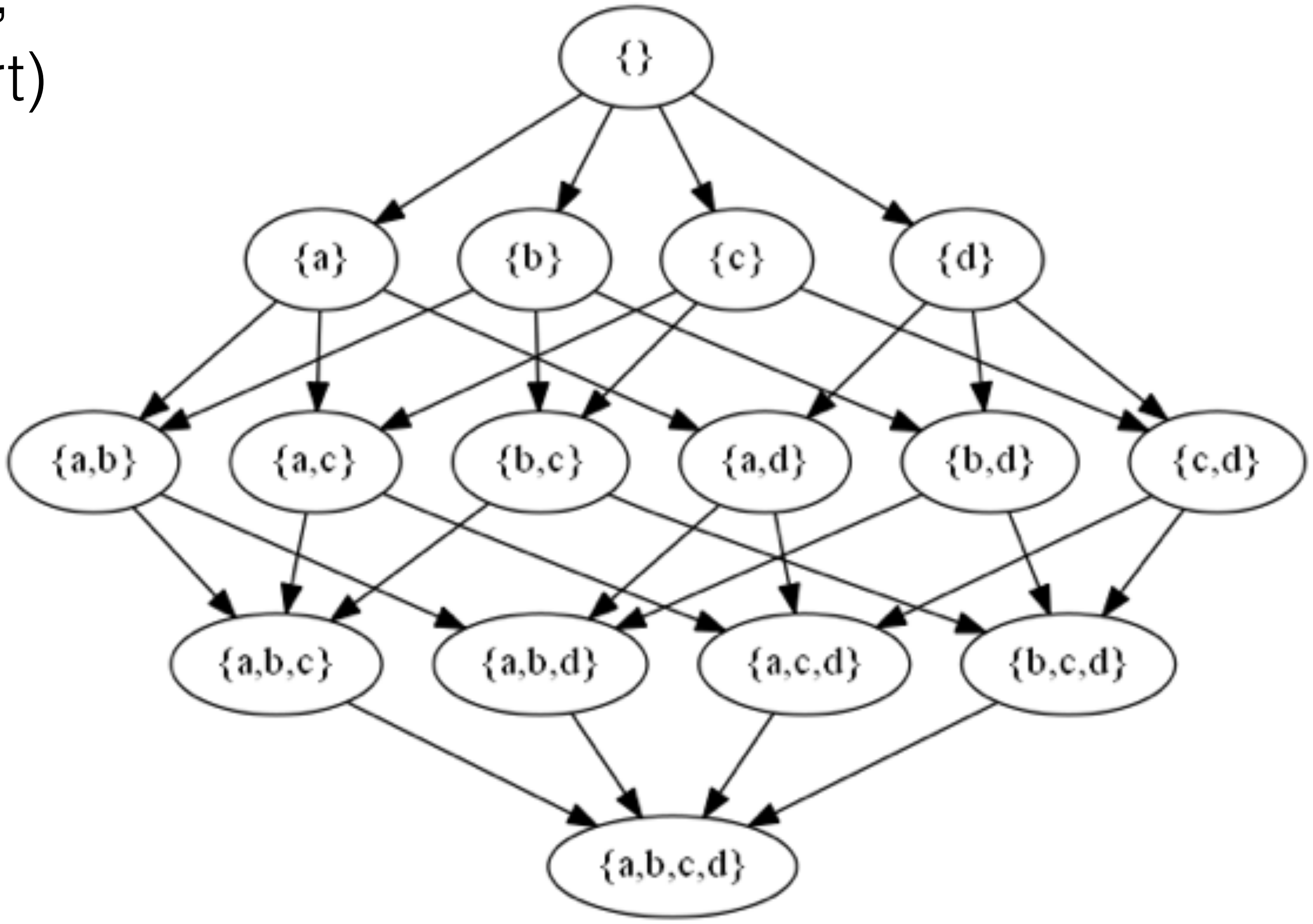


Examples

- Market basket mining
- Item recommendation

Fundamentals

- Dataset D, set of items t ∈ D, constant *s* (minimum support)
- Support(t) = number of sets in D that contain t
- Itemset t is frequent if support(t) ≥ s
- Frequent Itemset problem:
  - Given D and s, find all frequent itemsets



Example

Document	Patterns
d1	abce
d2	cde
d3	abce
d4	acde
d5	abcde
d6	bcd

Support	**Frequent**
d1,d2,d3,d4,d5,d6	c
d1,d2,d3,d4,d5	e,ce
d1,d3,d4,d5	a,ac,ae,ace
d1,d3,d5,d6	b,bc
d2,d4,d5,d6	d,cd
d1,d3,d5	ab,abc,abe be,bce,abce
d2,d4,d5	de,cde

minimal support = 3

Example

Document	Patterns
d1	abce
d2	cde
d3	abce
d4	acde
d5	abcde
d6	bcd

Support	**Frequent**
6	c
5	e,ce
4	a,ac,ae,ace
4	b,bc
4	d,cd
3	ab,abc,abe be,bce,abce
3	de,cde

Variations

- *A priori* property: t ⊆ t' → support(t) ≥ support(t')
- Closed: none of its supersets has the same support
  - Can generate all freq. itemsets and their support
- Maximal: none of its supersets is frequent
  - Can generate all freq. itemsets (without support)
- Maximal ⊆ Closed ⊆ Frequent ⊆ D

Itemset Streams

- Support as fraction of stream length
- Exact vs approximate
- Incremental, sliding window, adaptive
- Frequent, closed, maximal

Lossy Counting

G. S. Manku, R. Motwani: “Approximate frequency counts over data streams”. VLDB '02

- Keep data structure D with tuples (x, freq(x), error(x))
- Imagine to divide the stream in buckets of size $\lceil 1/\varepsilon \rceil$
- Foreach itemset x in the stream,
  B_{id} = current sequential bucket id starting from 1
  - if $x \in D$, freq(x)++
  - else $D \leftarrow D \cup (x, 1, B_{id} - 1)$
- Prune D at bucket boundaries: evict x if $freq(x) + error(x) \leq B_{id}$

Moment

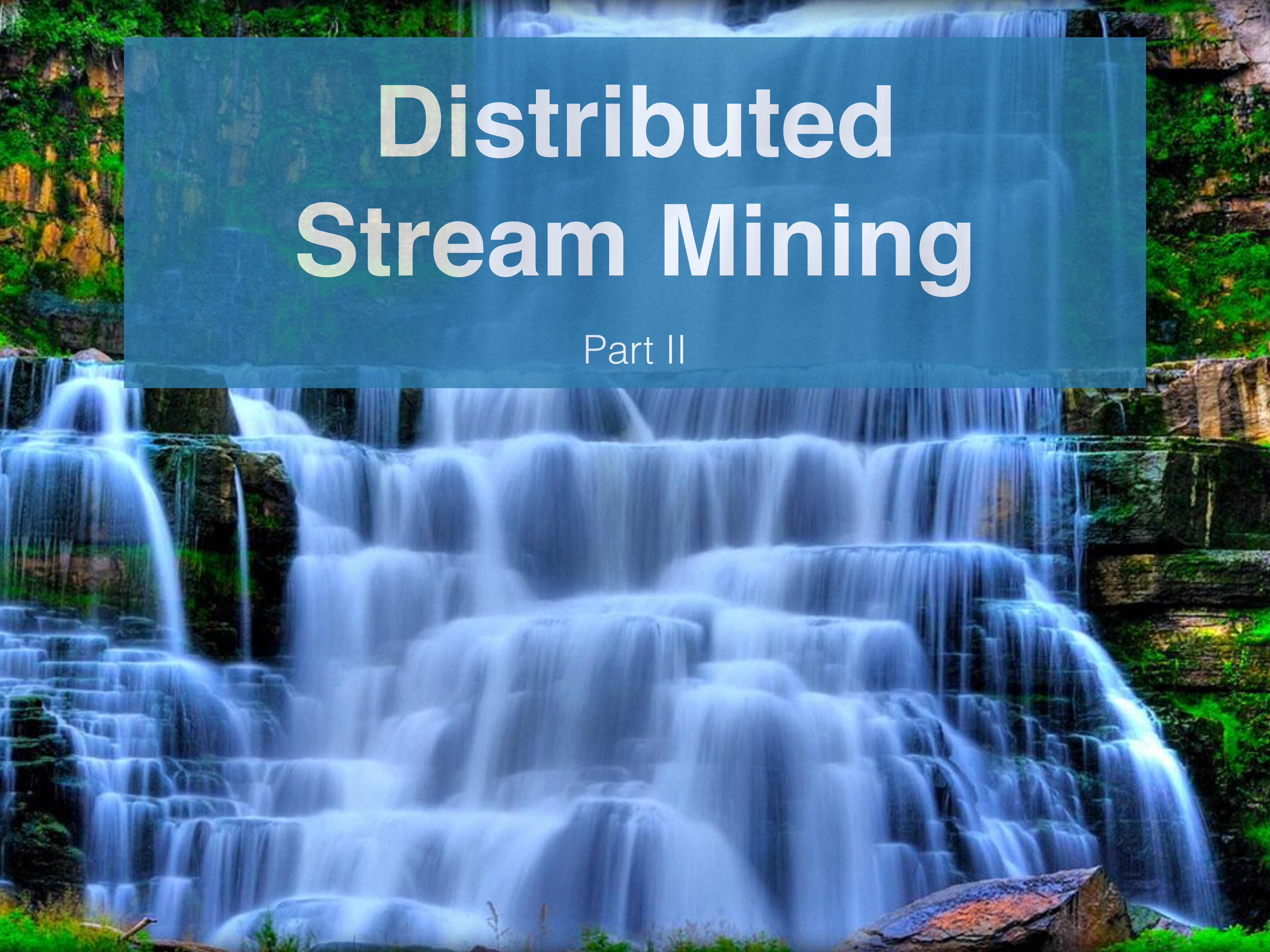
Y. Chi , H. Wang, P. Yu , R. Muntz: “Moment: Maintaining Closed Frequent Itemsets over a Stream Sliding Window”. ICDM ‘04

- Keeps track of boundary below frequent itemsets in a window
- Closed Enumeration Tree (CET) (~ prefix tree)
    - Infrequent gateway nodes (infrequent)
    - Unpromising gateway nodes (infrequent, dominated)
    - Intermediate nodes (frequent, dominated)
    - Closed nodes (frequent)
- By adding/removing transactions closed/infreq. do not change

FP-Stream

C. Giannella, J. Han, J. Pei, X. Yan, P. S. Yu: “Mining frequent patterns in data streams at multiple time granularities”. NGDM (2003)

- Multiple time granularities
- Based on FP-Growth (depth-first search over itemset lattice)
- Pattern-tree + Tilted-time window
- Time sensitive queries, emphasis on recent history
- High time and memory complexity



Distributed Stream Mining

Part II

Outline

- **Fundamentals of Stream Mining**
  - Setting
  - Classification
  - Concept Drift
  - Regression
  - Clustering
  - Frequent Itemset Mining
- **Distributed Stream Mining**
  - Distributed Stream Processing Engines
  - Classification
  - Regression
  - Conclusions

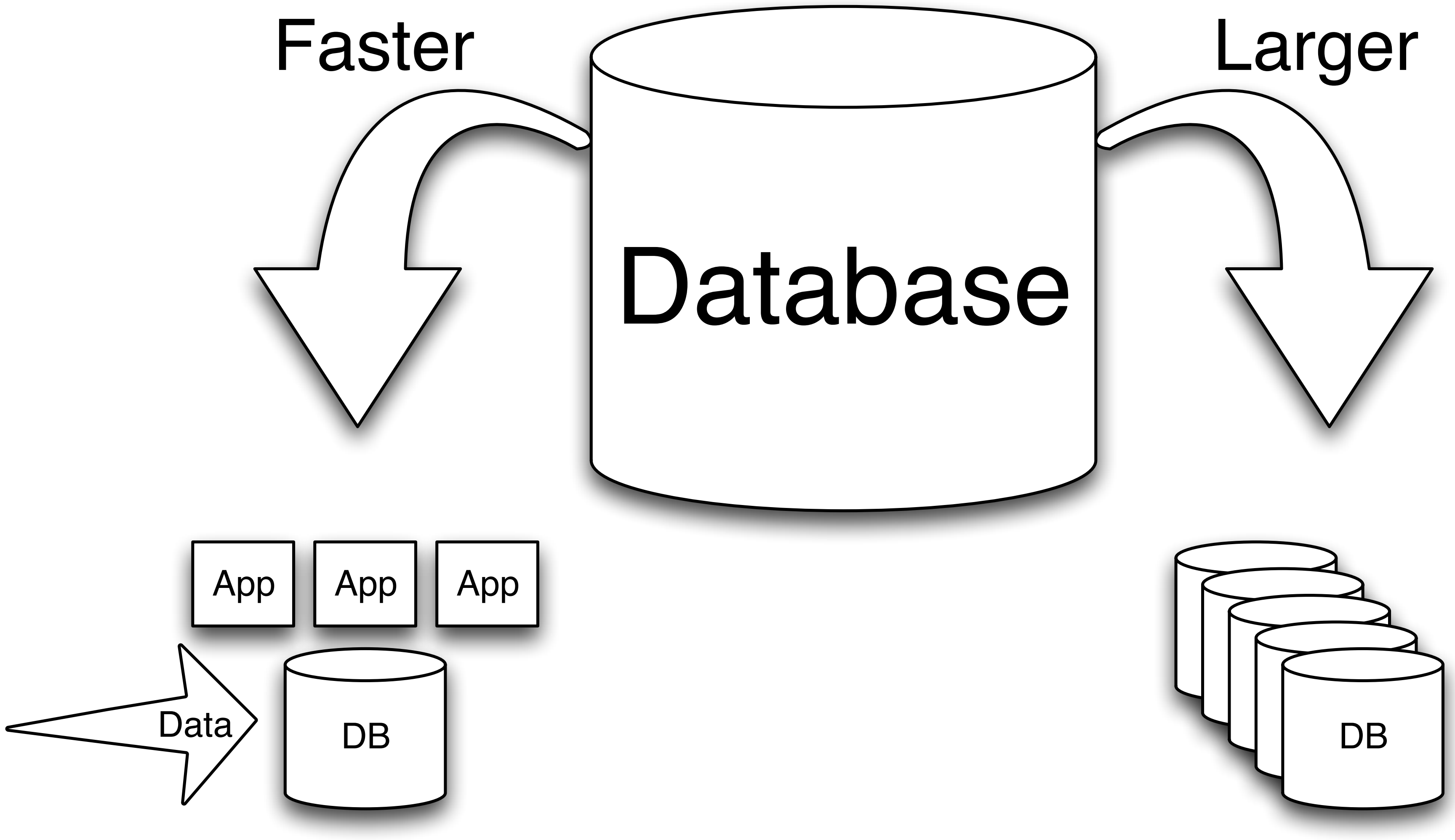
Motivation

- Datasets already stored on clusters
    - Don’t want to move everything to single powerful machine
- Clusters ubiquitous and cheap (e.g., see TOP500), supercomputers expensive and monolithic
    - Clusters easily shared, leverage economy of scale
- Largest problem solvable by single machine constrained by hardware
    - How fast can you read from disk or network

Distributed Stream Processing Engines

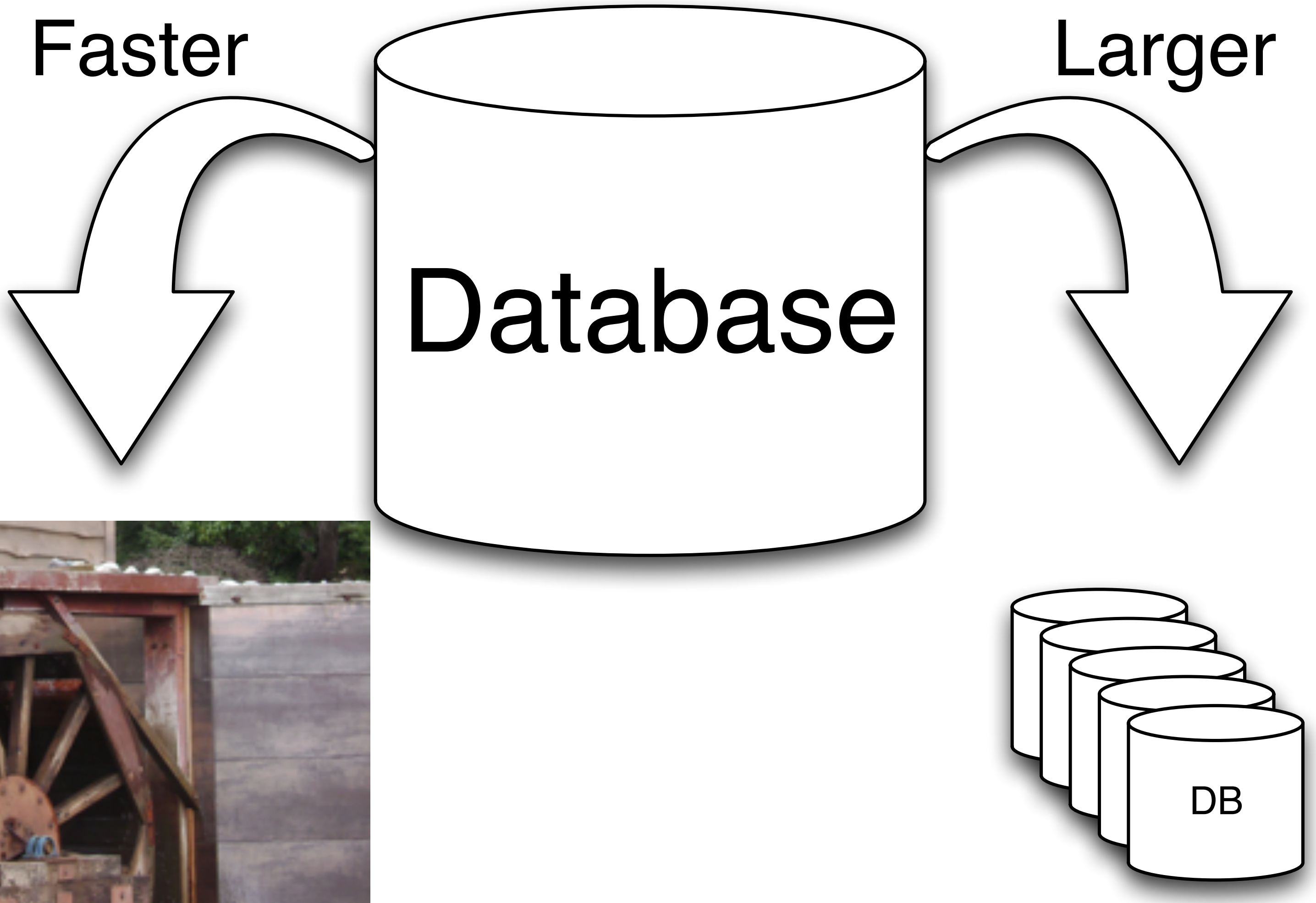
A Tale of two Tribes

M. Stonebraker U. Çetintemel: "'One Size Fits All': An Idea Whose Time Has Come and Gone". ICDE '05



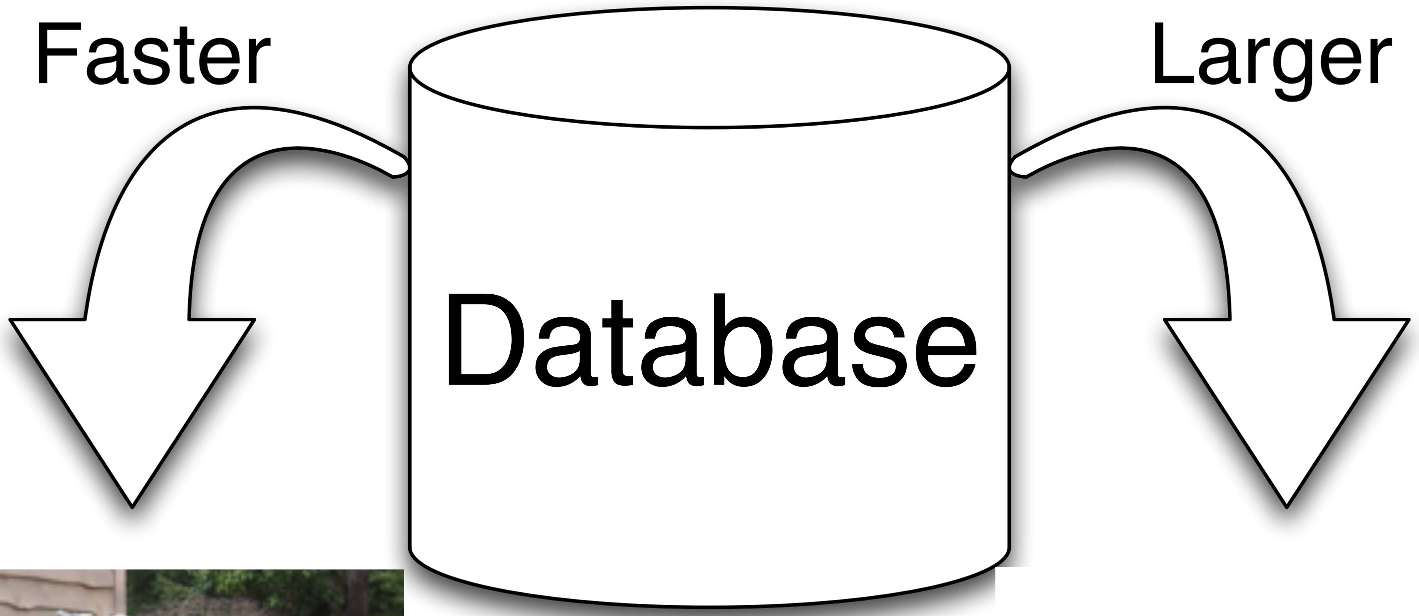
A Tale of two Tribes

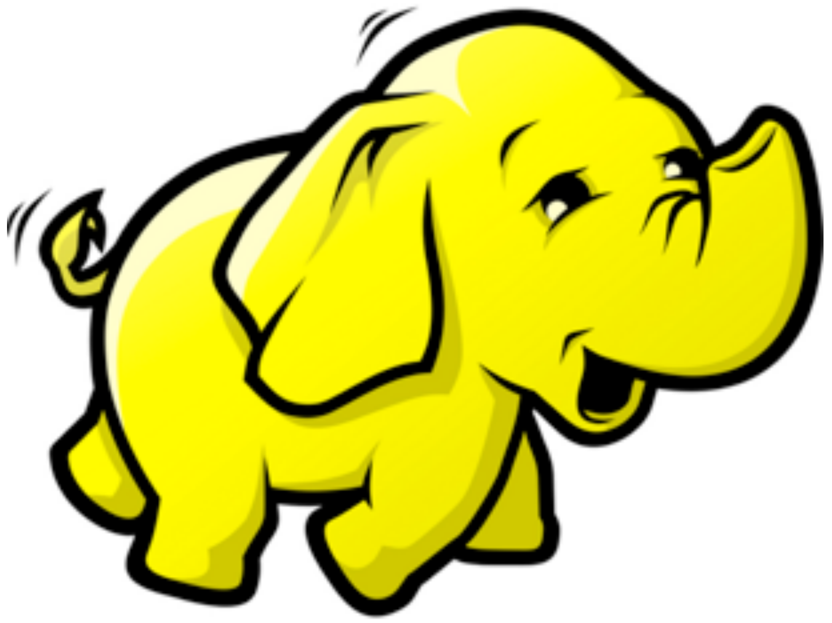
M. Stonebraker U. Çetintemel: "'One Size Fits All': An Idea Whose Time Has Come and Gone". ICDE '05



A Tale of two Tribes

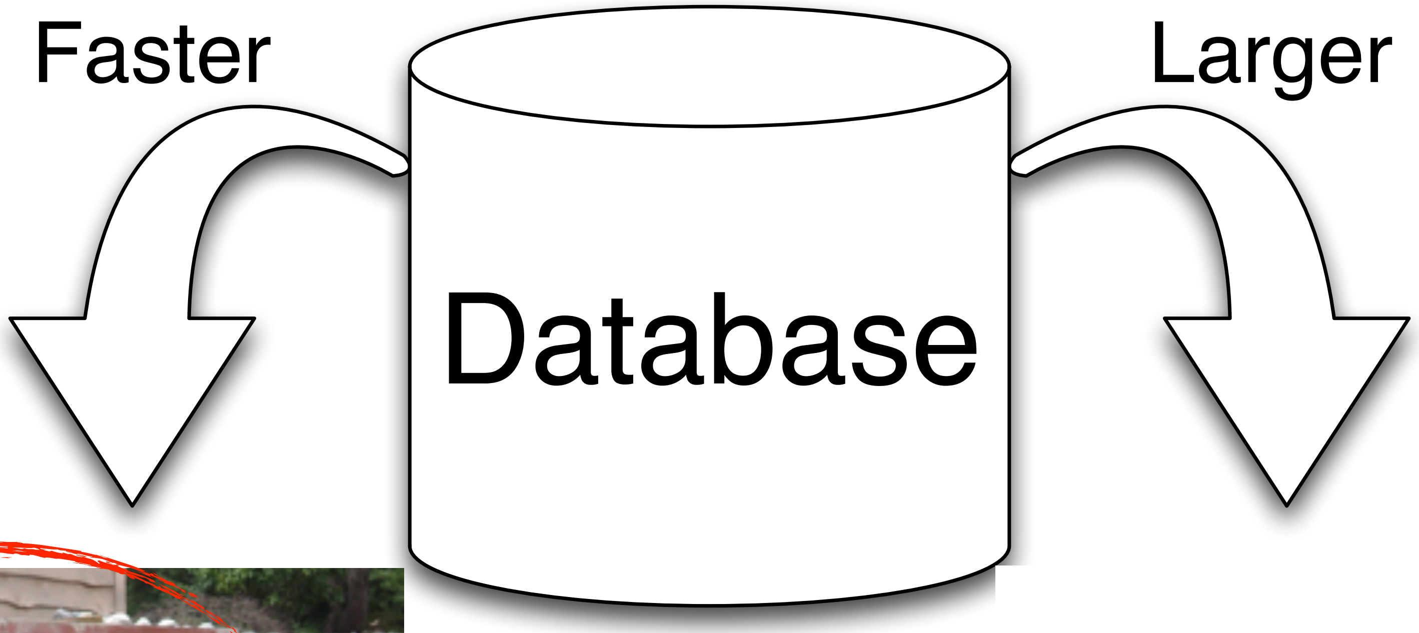
M. Stonebraker U. Çetintemel: "'One Size Fits All': An Idea Whose Time Has Come and Gone". ICDE '05

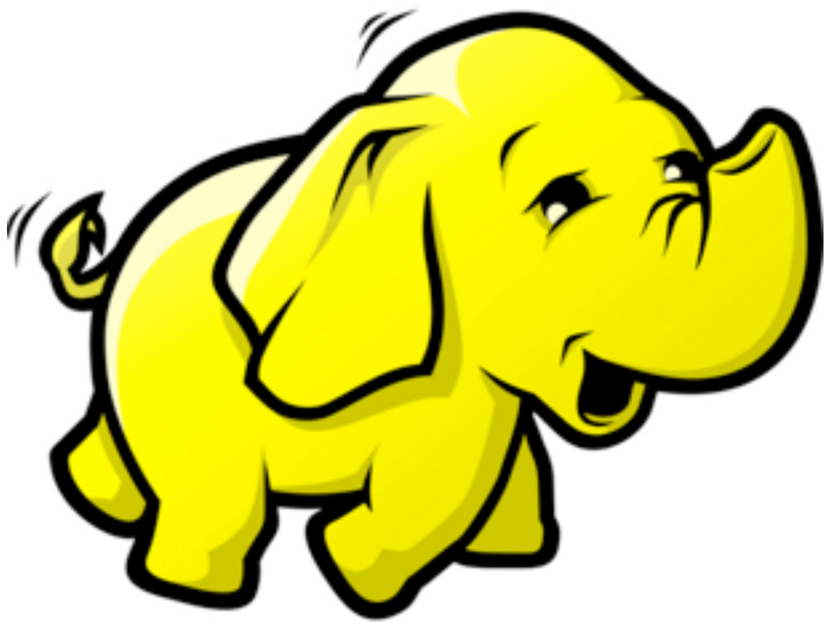




A Tale of two Tribes

M. Stonebraker U. Çetintemel: "'One Size Fits All': An Idea Whose Time Has Come and Gone". ICDE '05

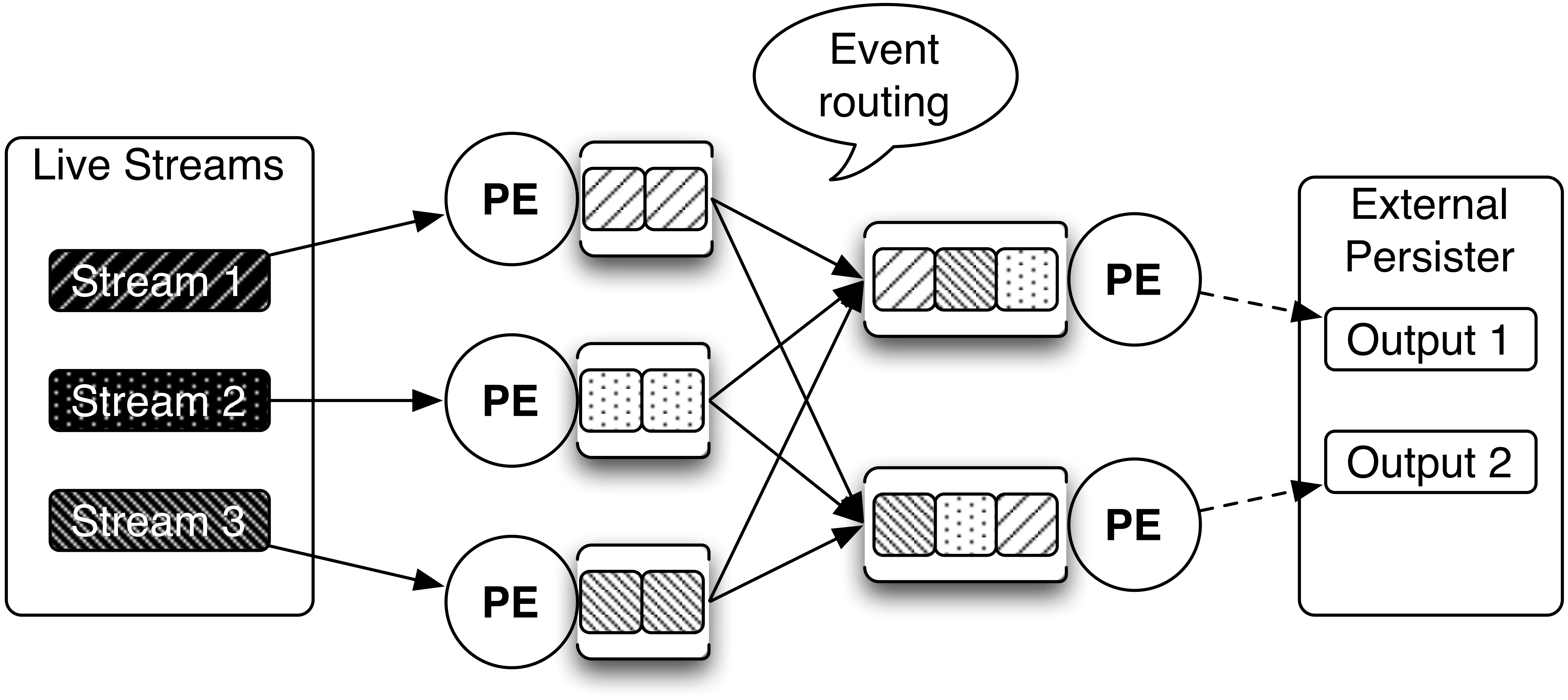




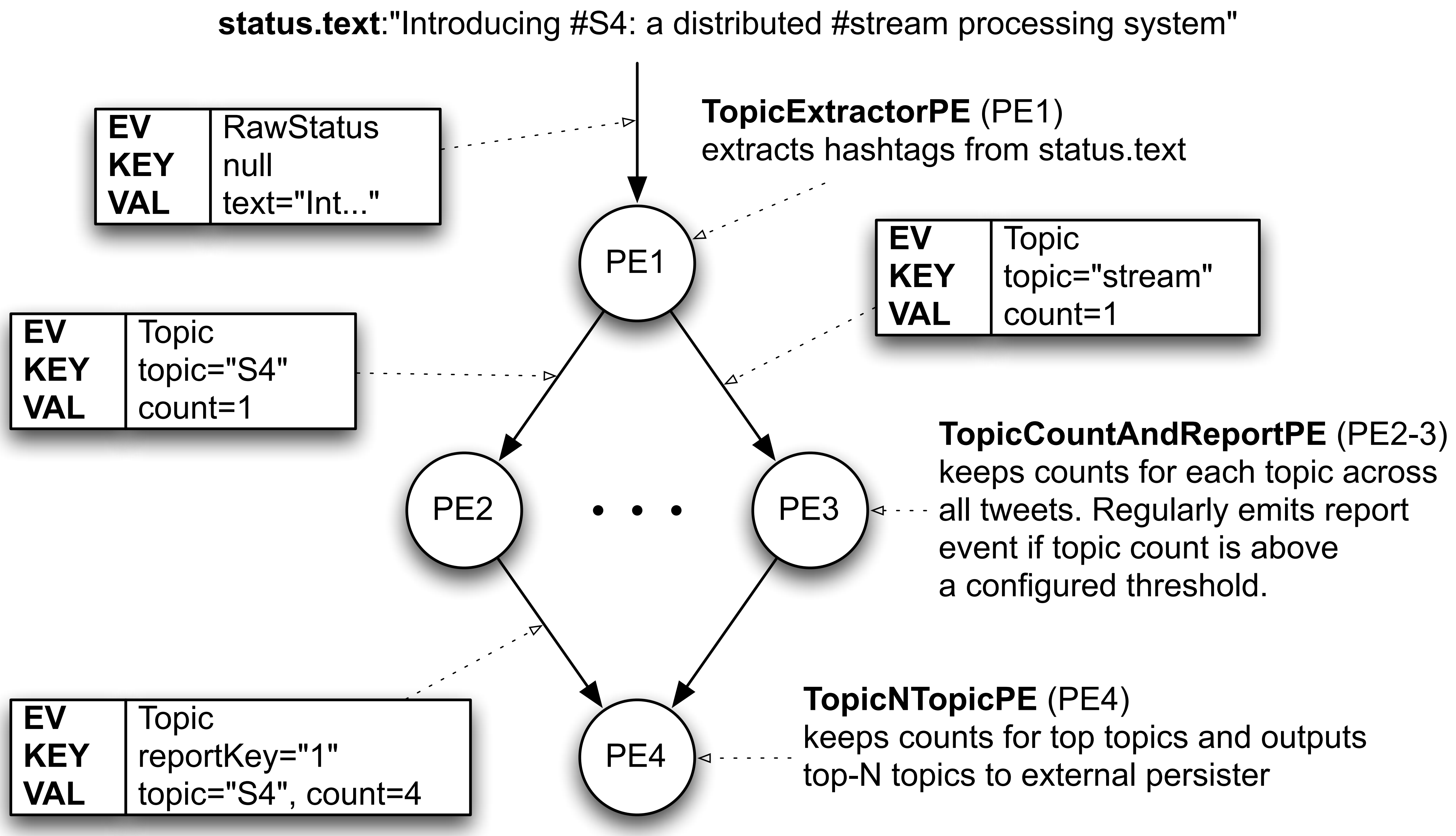
SPE Evolution

1st generation	2003	Aurora	Abadi et al., “Aurora: a new model and architecture for data stream management,” VLDB Journal, 2003
	2004	STREAM	Arasu et al., “STREAM: The Stanford Data Stream Management System,” Stanford InfoLab, 2004.
2nd generation	2005	Borealis	Abadi et al., “The Design of the Borealis Stream Processing Engine,” in CIDR ’05
	2006	SPC	Amini et al., “SPC: A Distributed, Scalable Platform for Data Mining,” in DMSSP ’06
	2008	SPADE	Gedik et al., “SPADE: The System S Declarative Stream Processing Engine,” in SIGMOD ’08
3rd generation	2010	S4	Neumeyer et al., “S4: Distributed Stream Computing Platform,” in ICDMW ’10
	2011	Storm	http://storm.apache.org
	2013	Samza	http://samza.incubator.apache.org

Actors Model



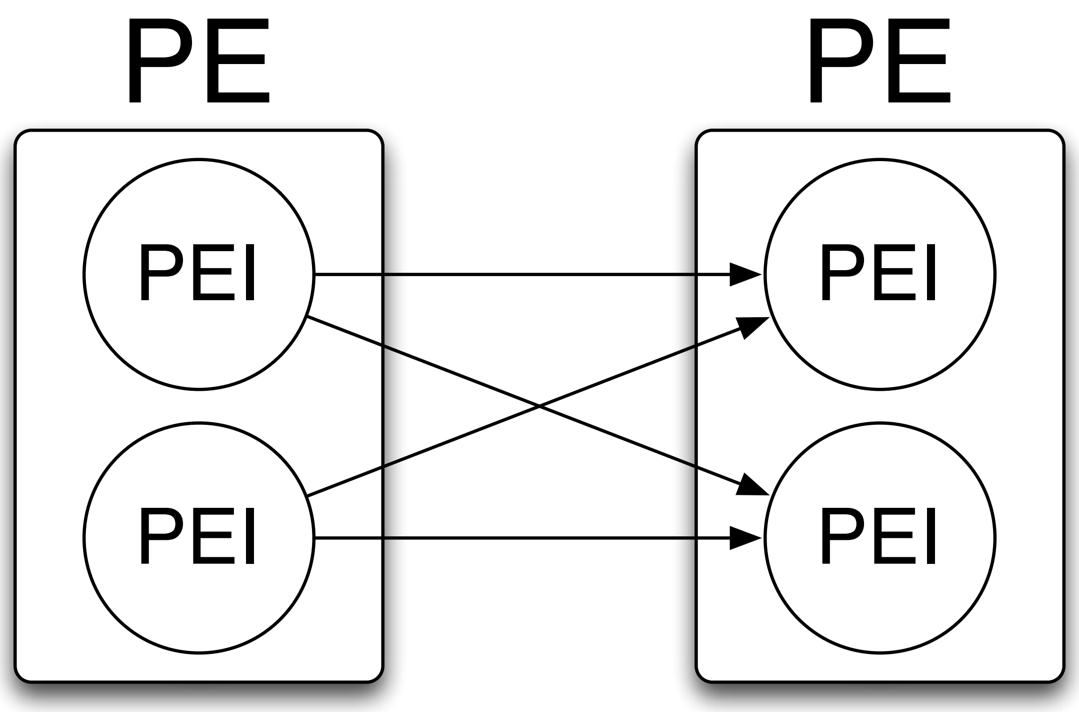
S4 Example



status.text:"Introducing #S4: a distributed #stream processing system"
TopicExtractorPE (PE1)
extracts hashtags from status.text
EV RawStatus
KEY null
VAL text="Int..."
PE1
EV Topic
KEY topic="stream"
VAL count=1
EV Topic
KEY topic="S4"
VAL count=1
PE2
• • •
PE3
TopicCountAndReportPE (PE2-3)
keeps counts for each topic across all tweets. Regularly emits report event if topic count is above a configured threshold.
EV Topic
KEY reportKey="1"
VAL topic="S4", count=4
PE4
TopicNTopicPE (PE4)
keeps counts for top topics and outputs top-N topics to external persister

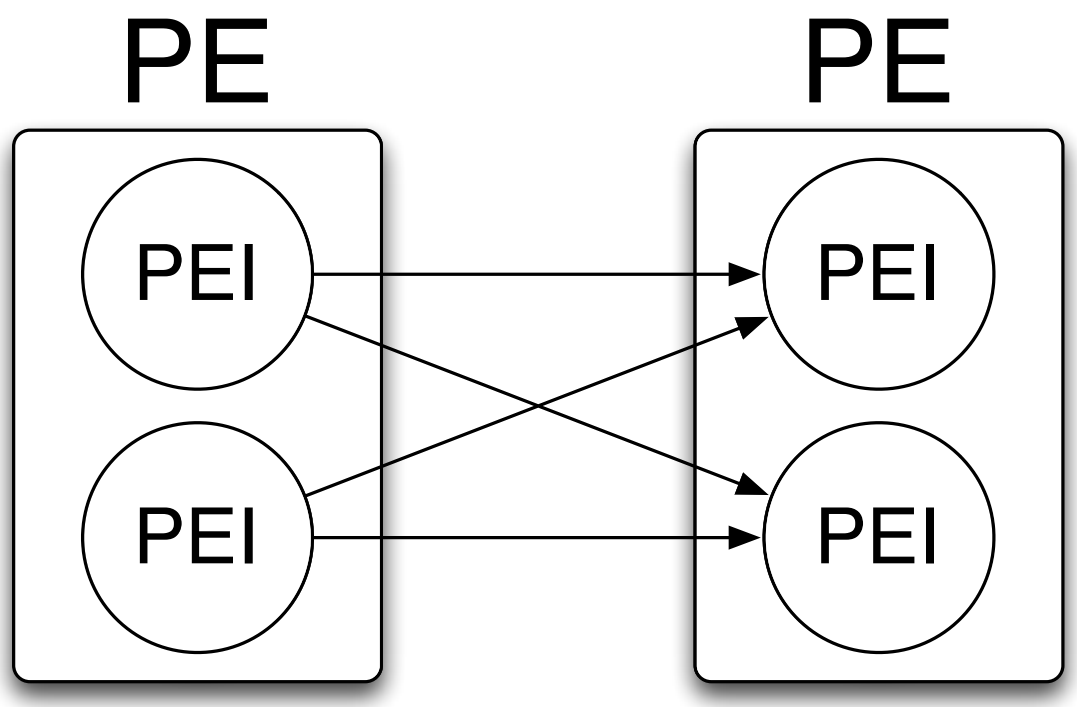
Groupings

- Key Grouping (hashing)
- Shuffle Grouping (round-robin)
- All Grouping (broadcast)



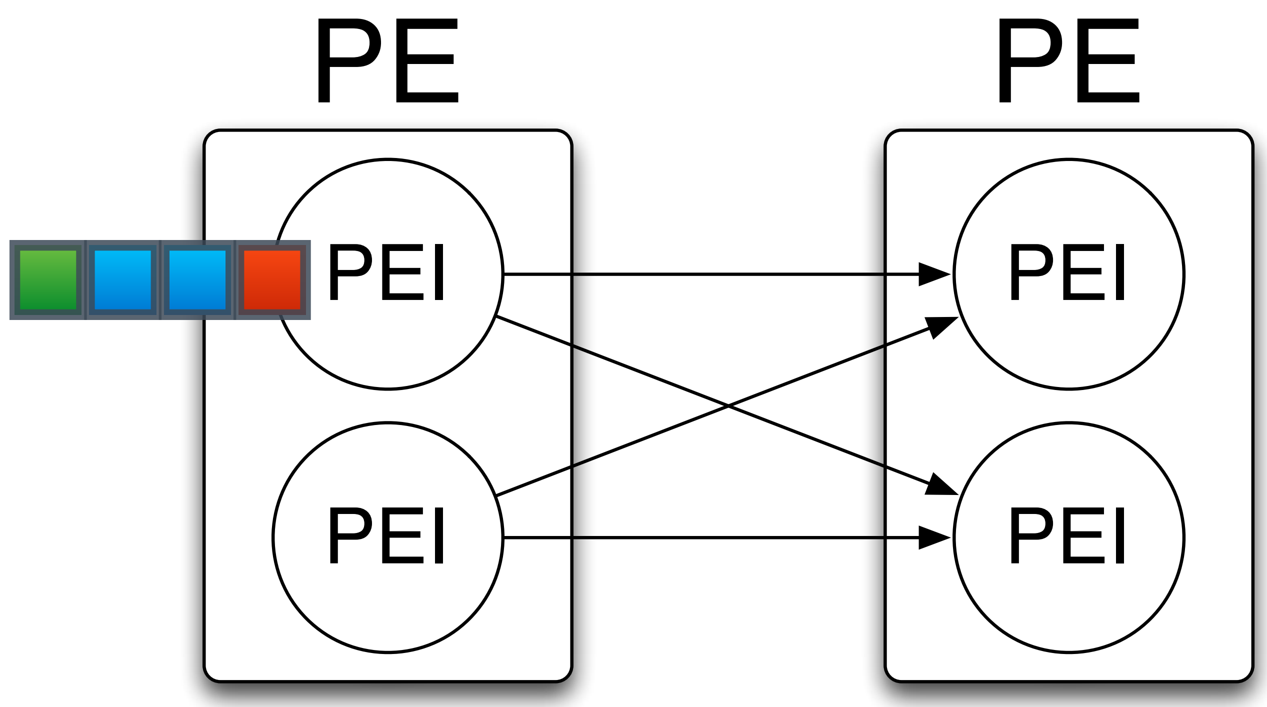
Groupings

- **Key Grouping (hashing)**
- Shuffle Grouping (round-robin)
- All Grouping (broadcast)



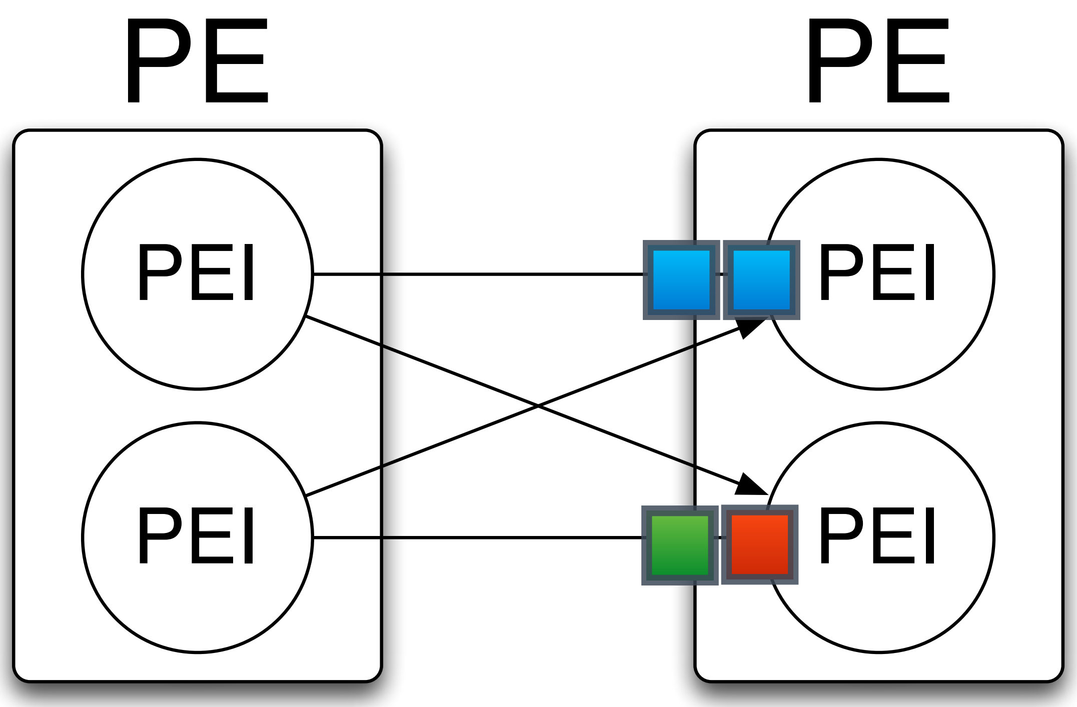
Groupings

- **Key Grouping (hashing)**
- Shuffle Grouping (round-robin)
- All Grouping (broadcast)



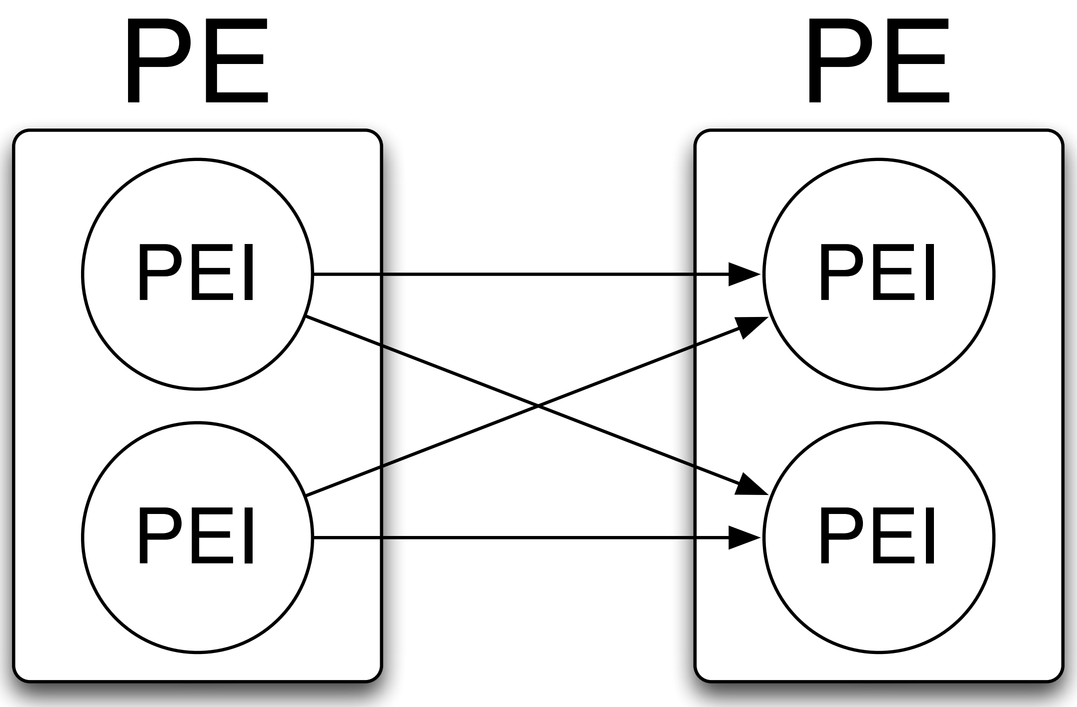
Groupings

- **Key Grouping (hashing)**
- Shuffle Grouping (round-robin)
- All Grouping (broadcast)



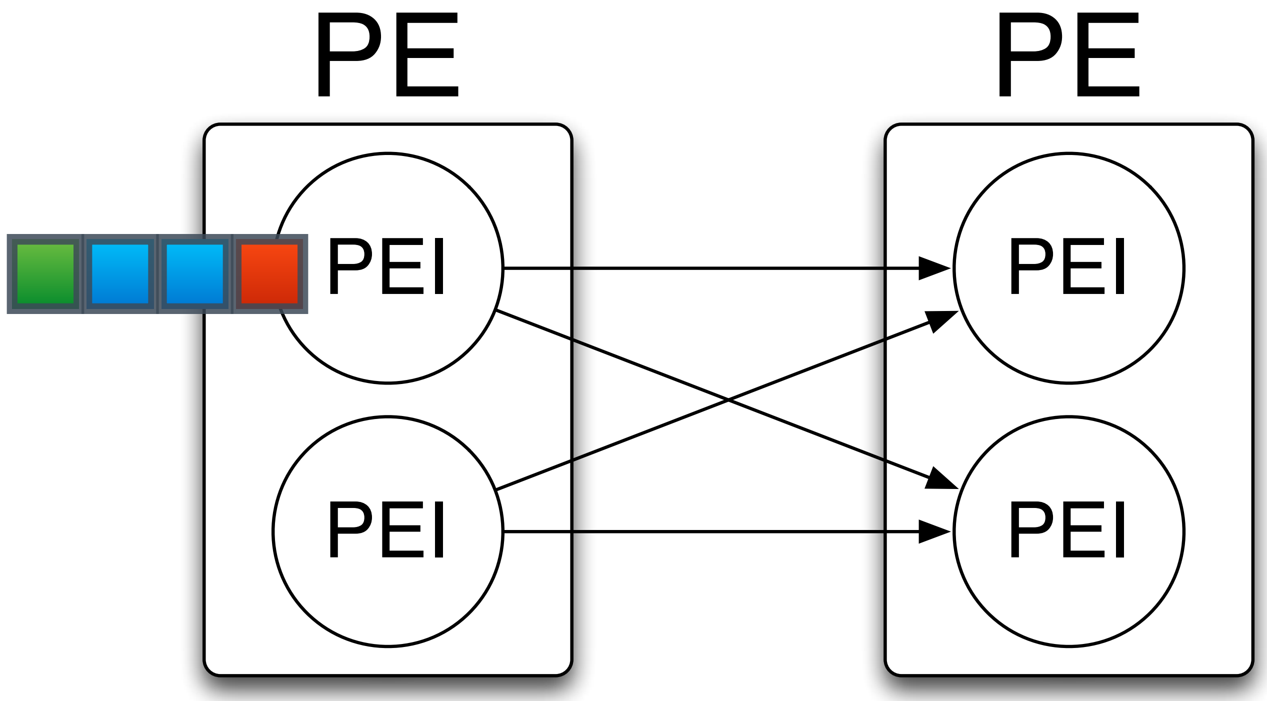
Groupings

- Key Grouping (hashing)
- **Shuffle Grouping (round-robin)**
- All Grouping (broadcast)



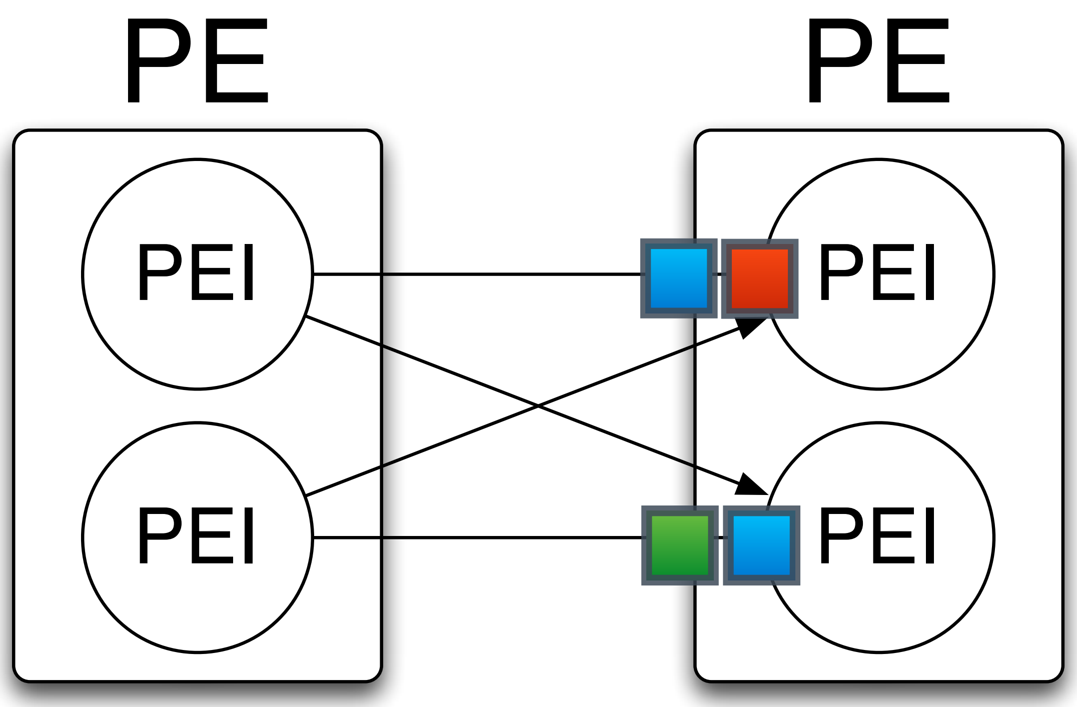
Groupings

- Key Grouping (hashing)
- **Shuffle Grouping (round-robin)**
- All Grouping (broadcast)



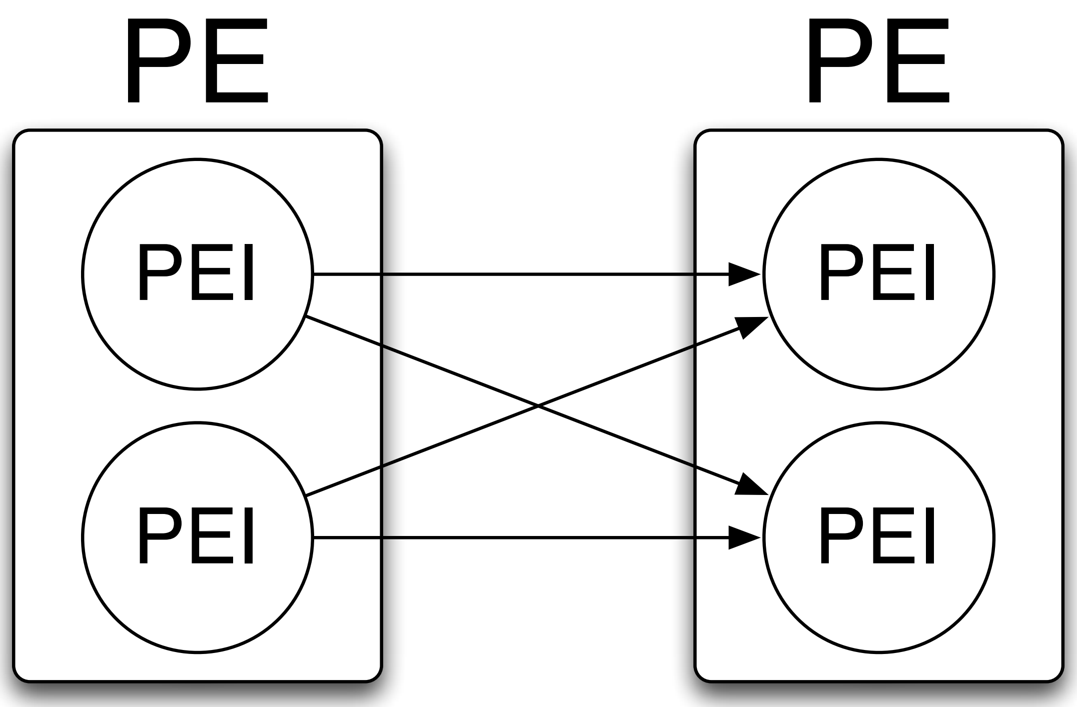
Groupings

- Key Grouping (hashing)
- **Shuffle Grouping (round-robin)**
- All Grouping (broadcast)



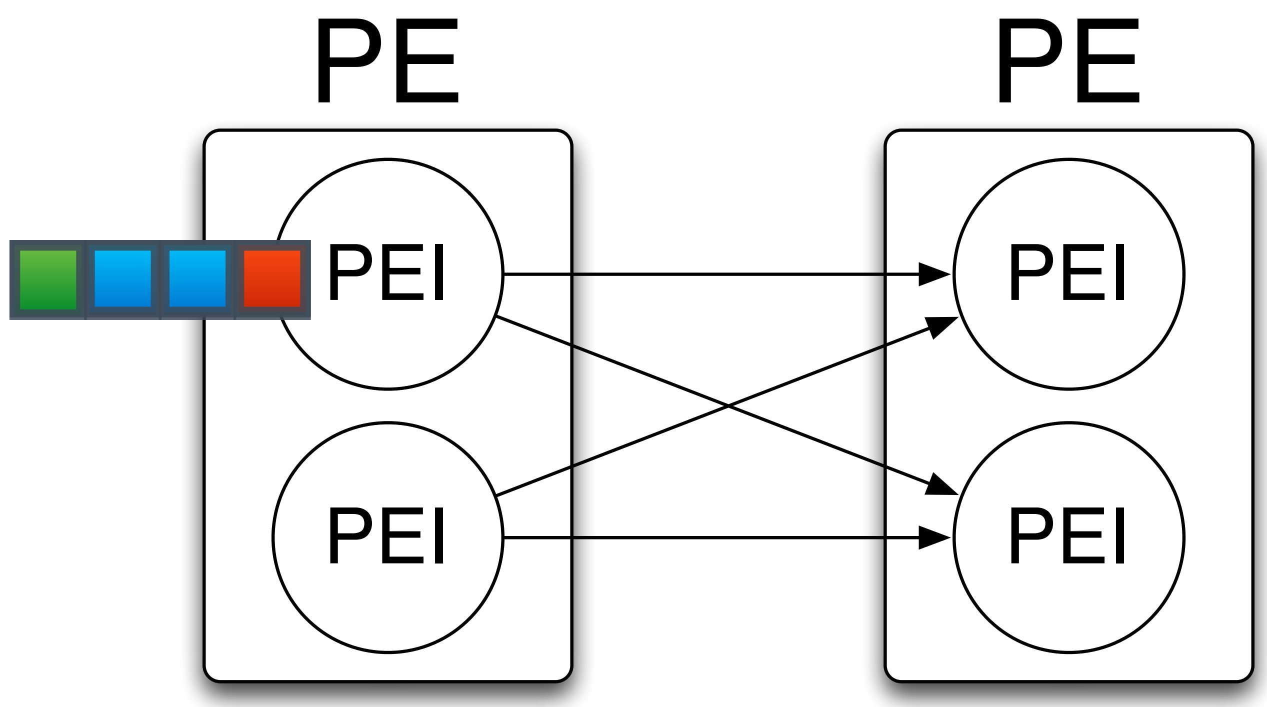
Groupings

- Key Grouping (hashing)
- Shuffle Grouping (round-robin)
- **All Grouping (broadcast)**



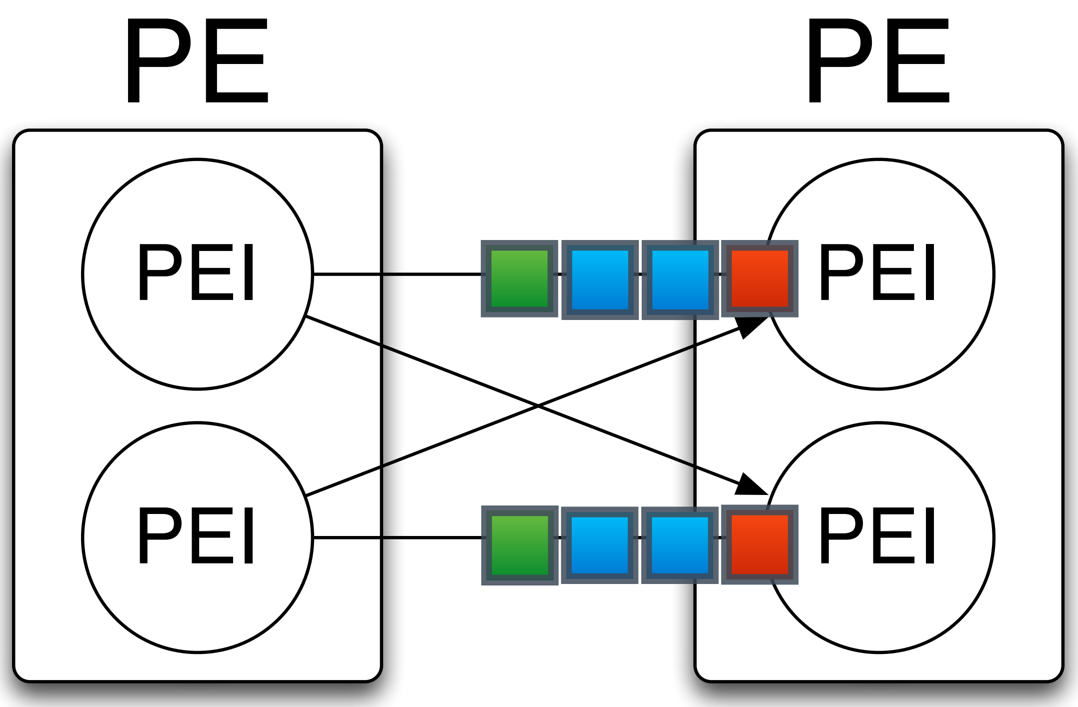
Groupings

- Key Grouping (hashing)
- Shuffle Grouping (round-robin)
- **All Grouping (broadcast)**



Groupings

- Key Grouping (hashing)
- Shuffle Grouping (round-robin)
- **All Grouping (broadcast)**

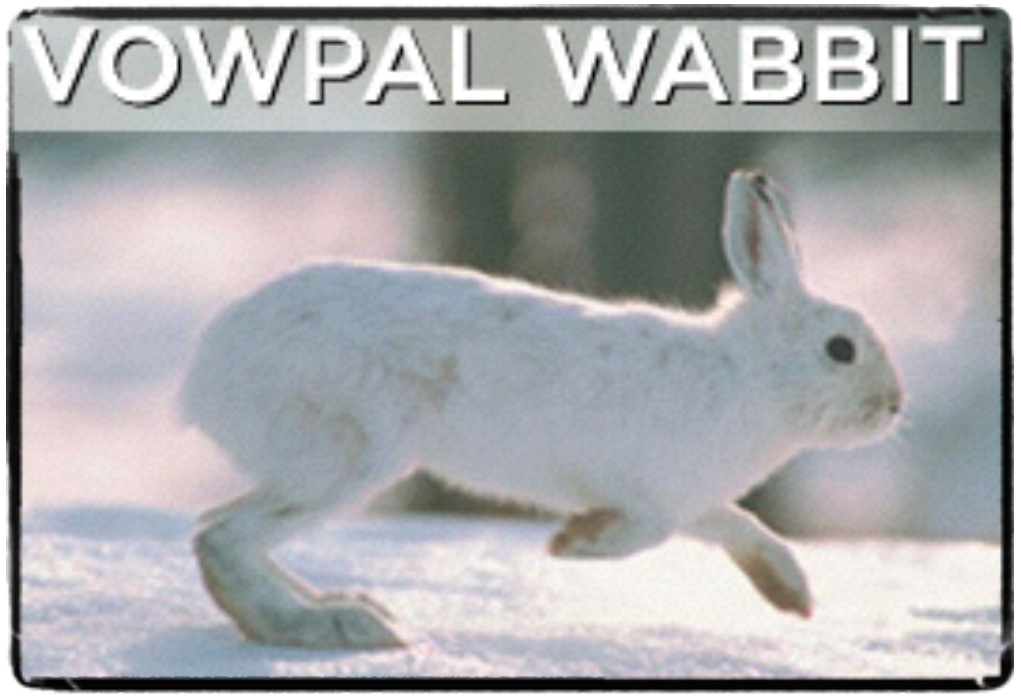


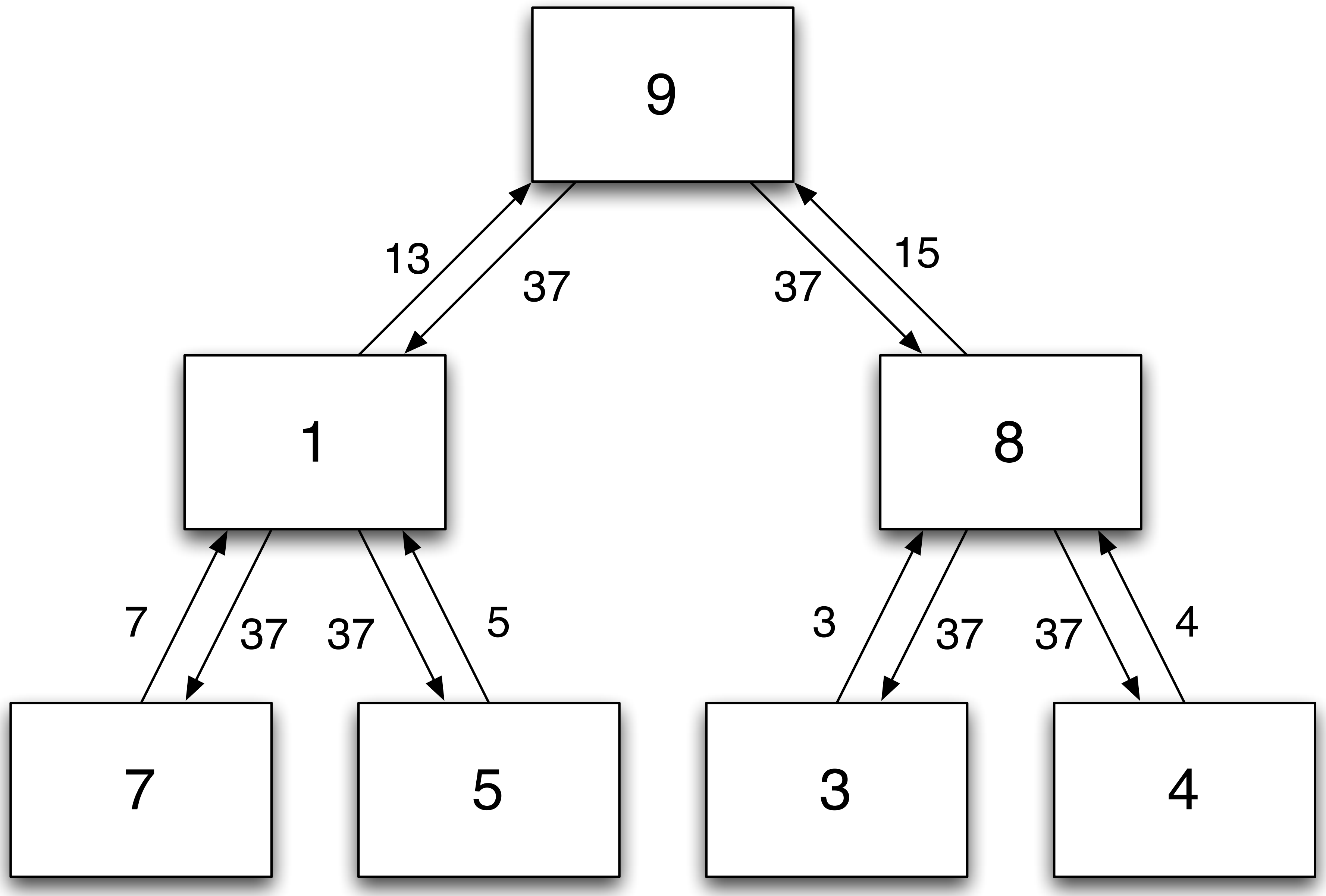
Classification

Hadoop AllReduce

A. Agarwal, O. Chapelle, M. Dudík, J. Langford: “A Reliable Effective Terascale Linear Learning System”. JMLR (2014)

- MPI AllReduce on MapReduce
- Parallel SGD + L-BFGS
- Aggregate + Redistribute
    - Each node computes partial gradient
    - Aggregate (sum) complete gradient
    - Each node gets updated model
- Hadoop for data locality (map-only job)





AllReduce

Reduction Tree

Upward = Reduce   Downward = Broadcast (All)

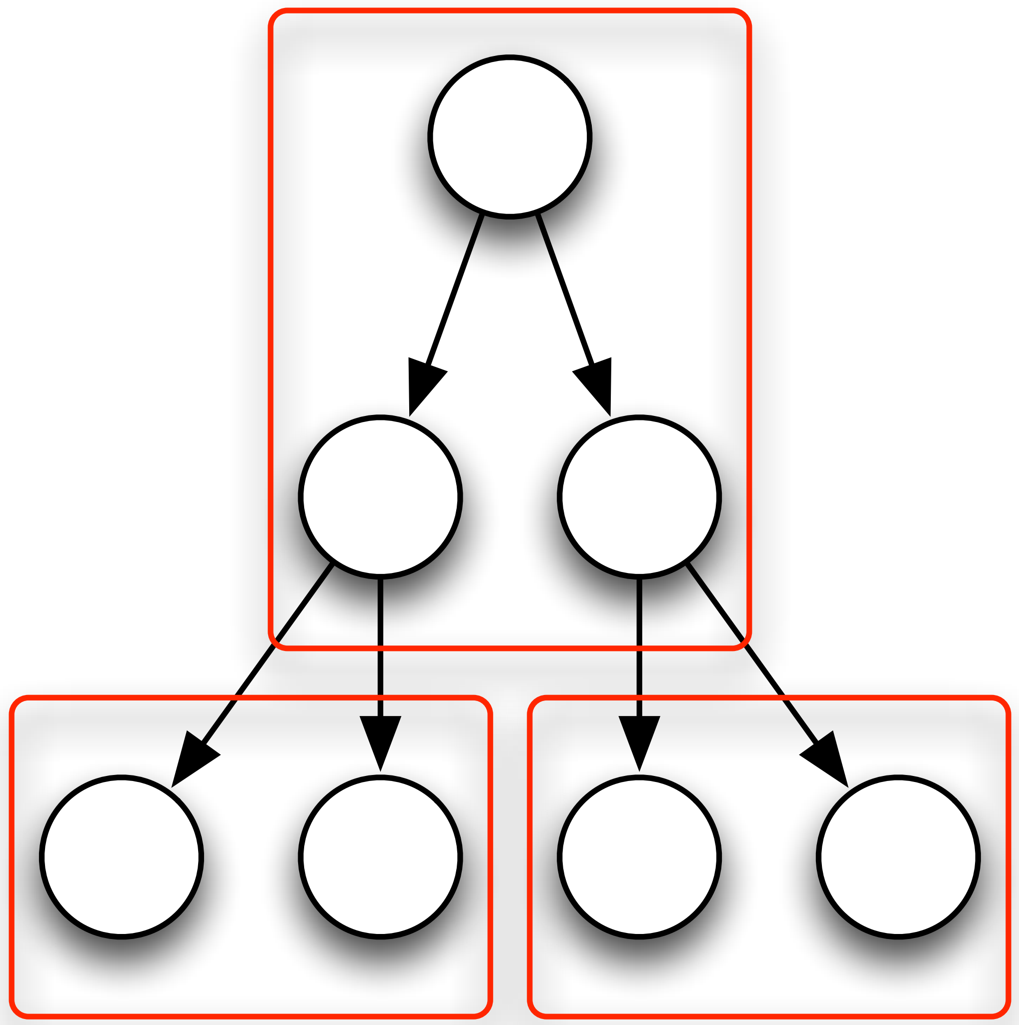
Parallel Decision Trees

Parallel Decision Trees

- Which kind of parallelism?

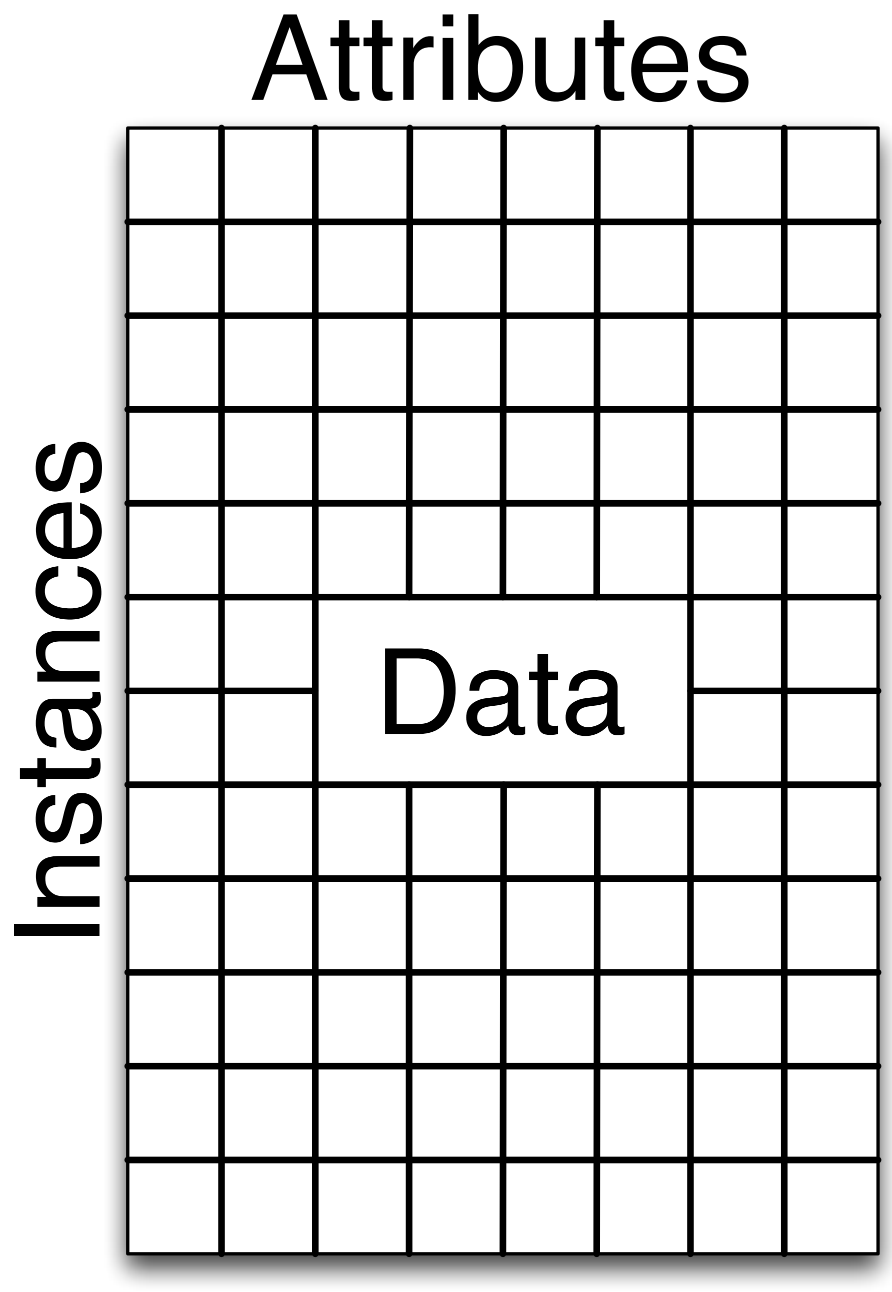
Parallel Decision Trees

- Which kind of parallelism?
  - Task



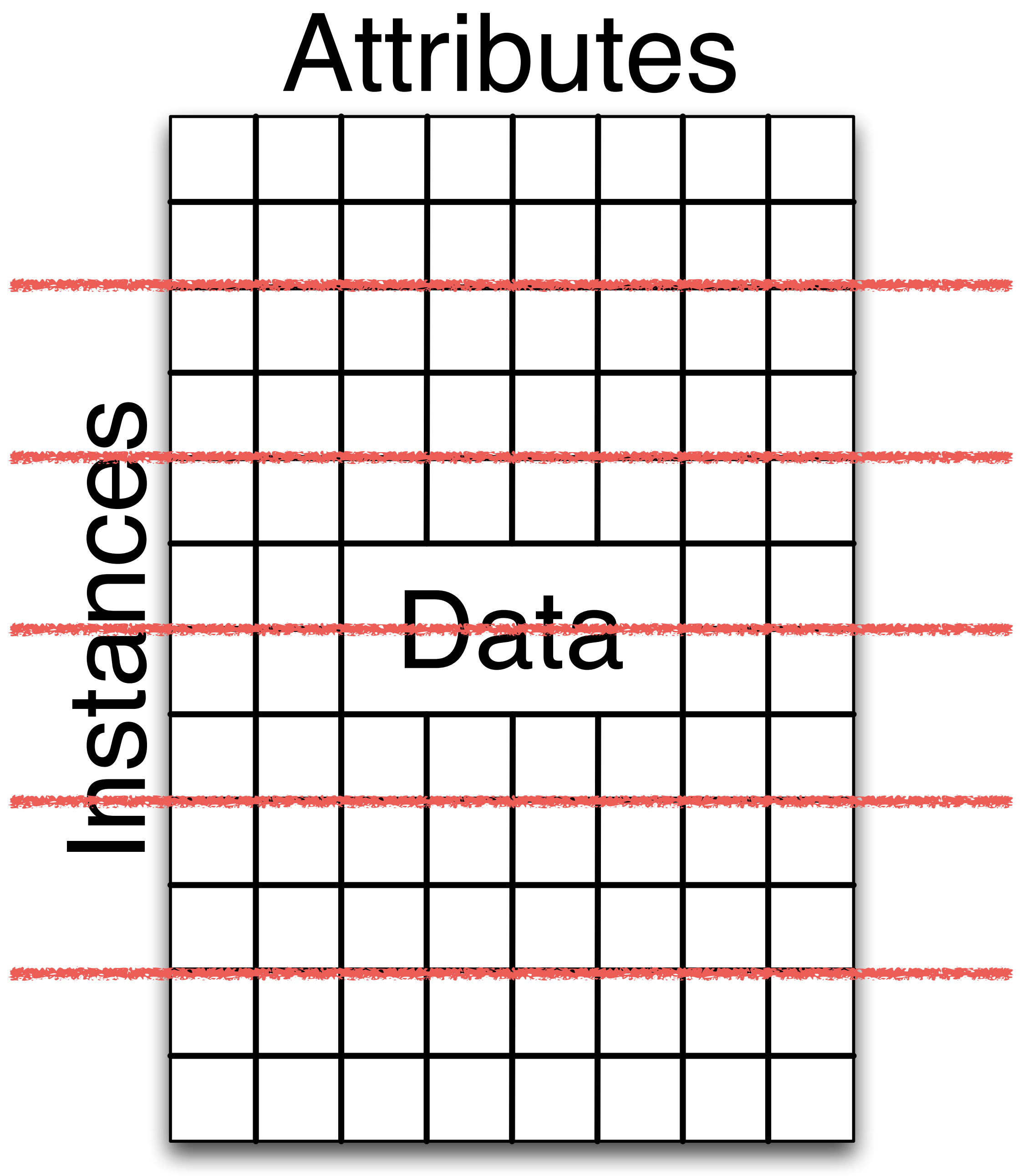
Parallel Decision Trees

- Which kind of parallelism?
  - Task
  - Data



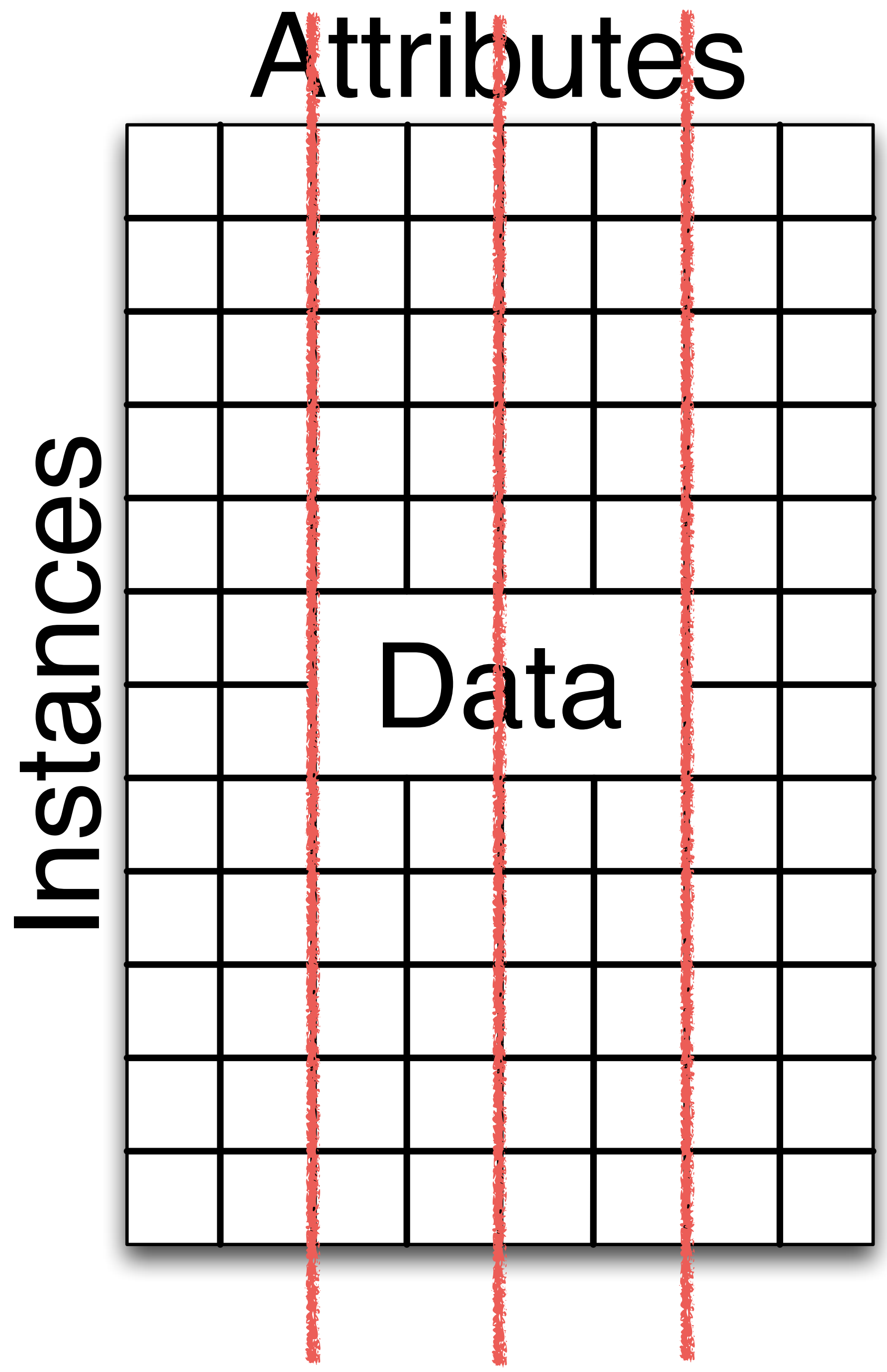
Parallel Decision Trees

- Which kind of parallelism?
    - Task
    - Data
        - Horizontal



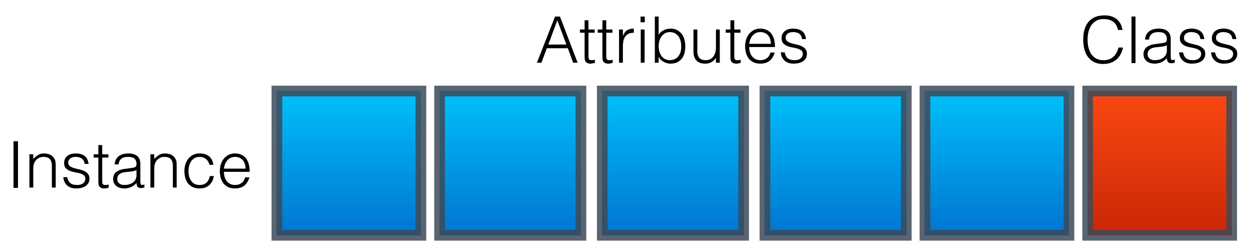
Parallel Decision Trees

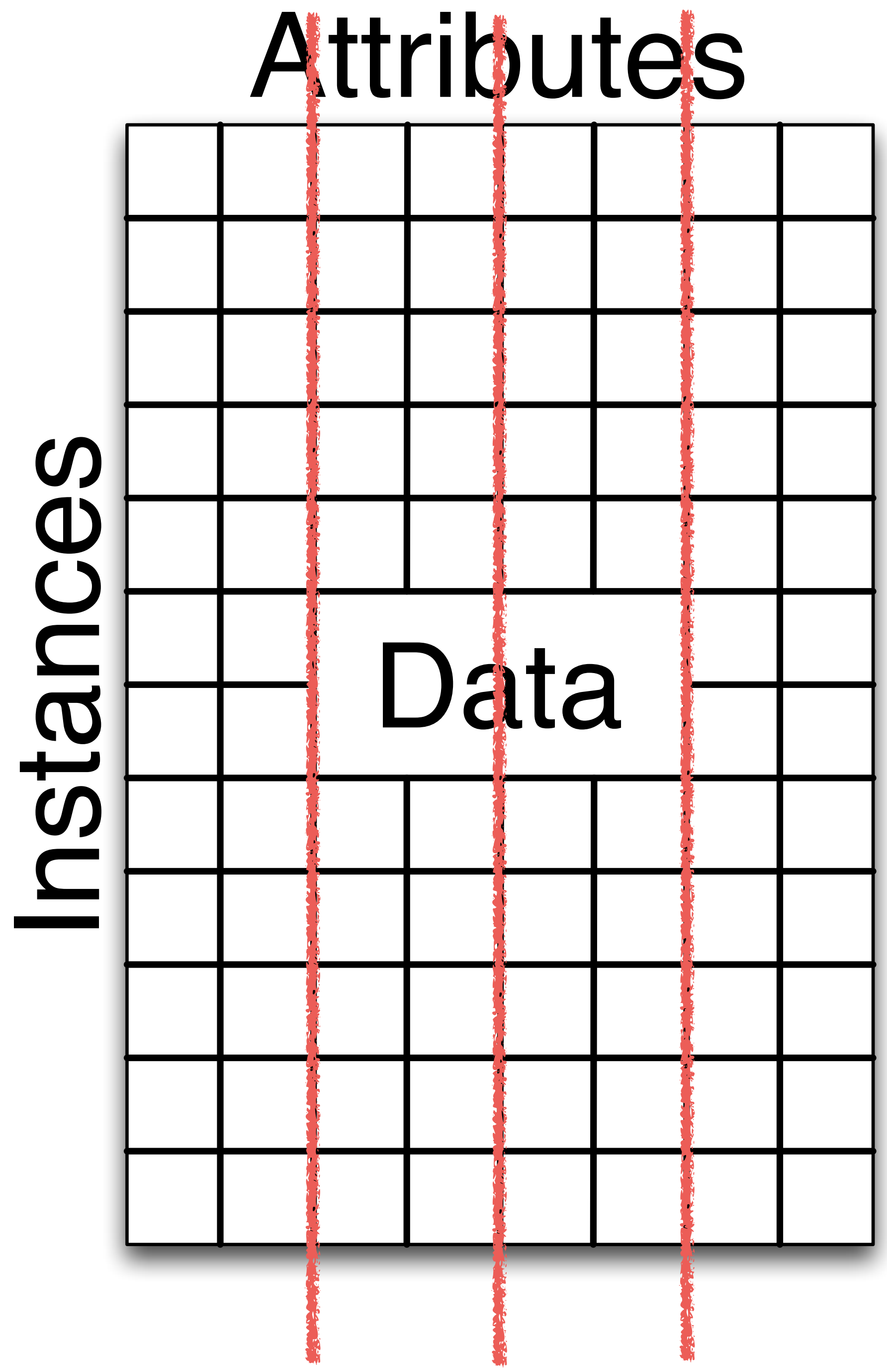
- Which kind of parallelism?
  - Task
  - Data
    - Horizontal
    - Vertical



Parallel Decision Trees

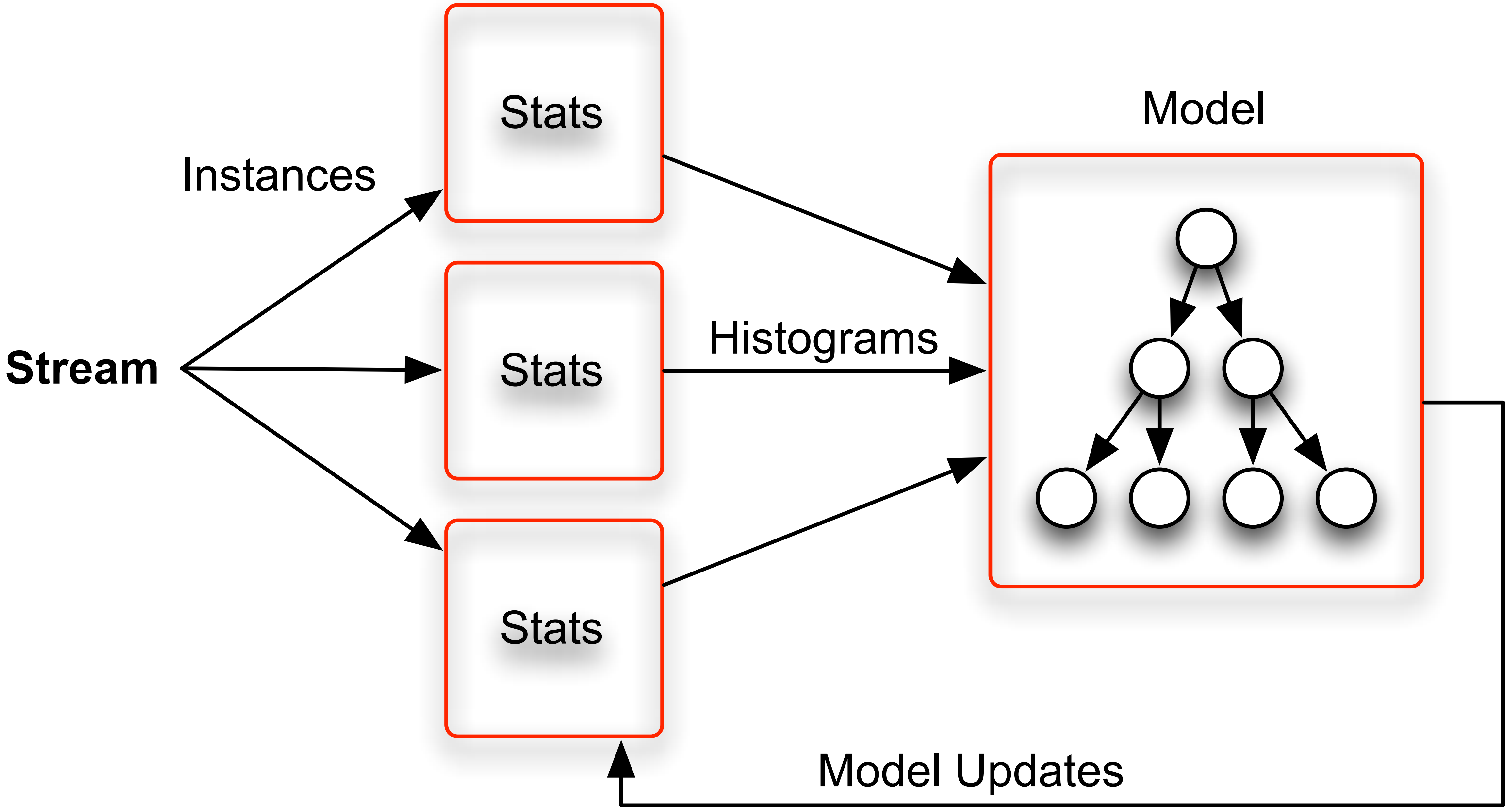
- Which kind of parallelism?
    - Task
    - Data
        - Horizontal
        - Vertical





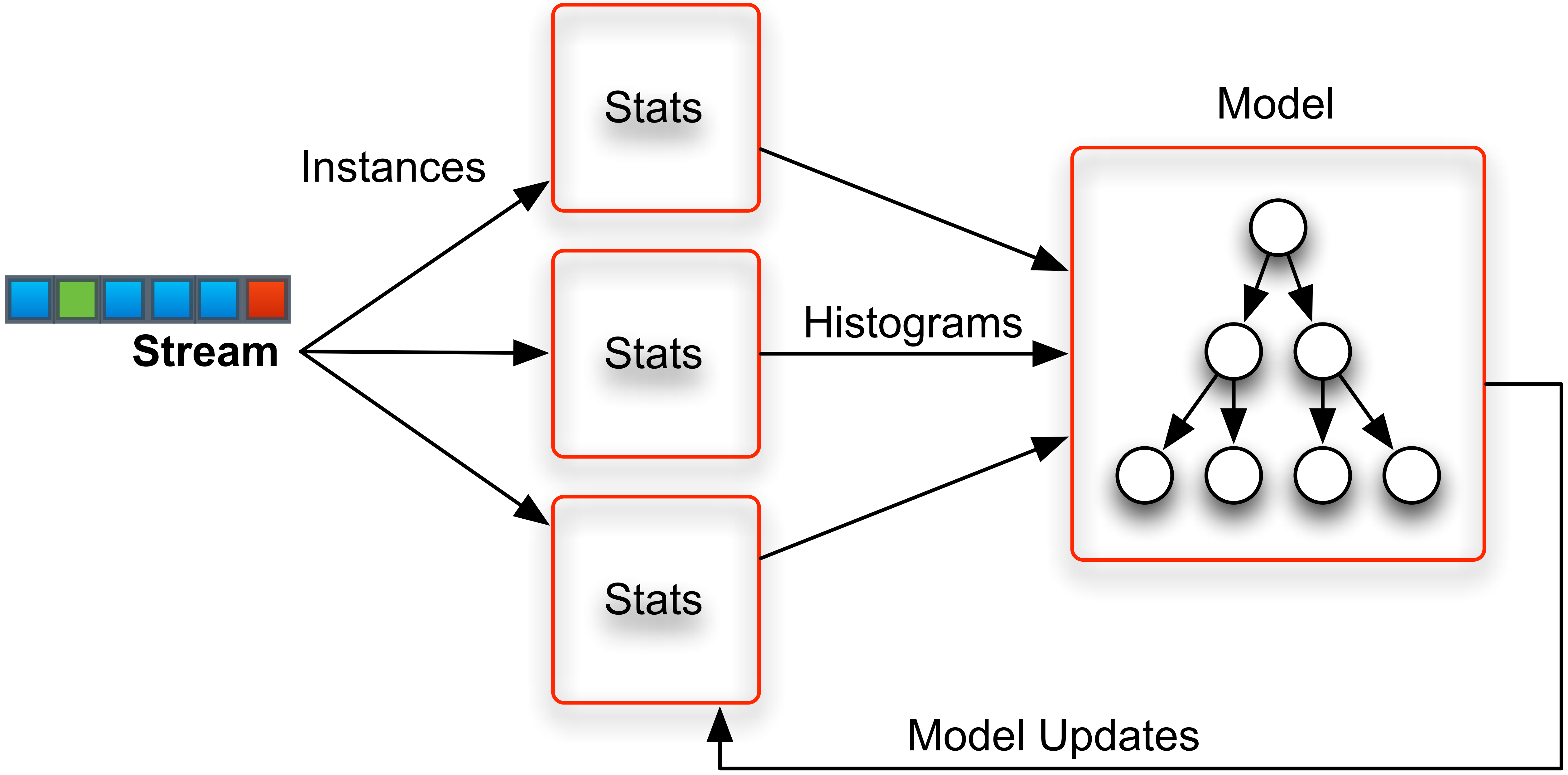
Horizontal Partitioning

Y. Ben-Haim, E. Tom-Tov: “A Streaming Parallel Decision Tree Algorithm”. JMLR (2010)



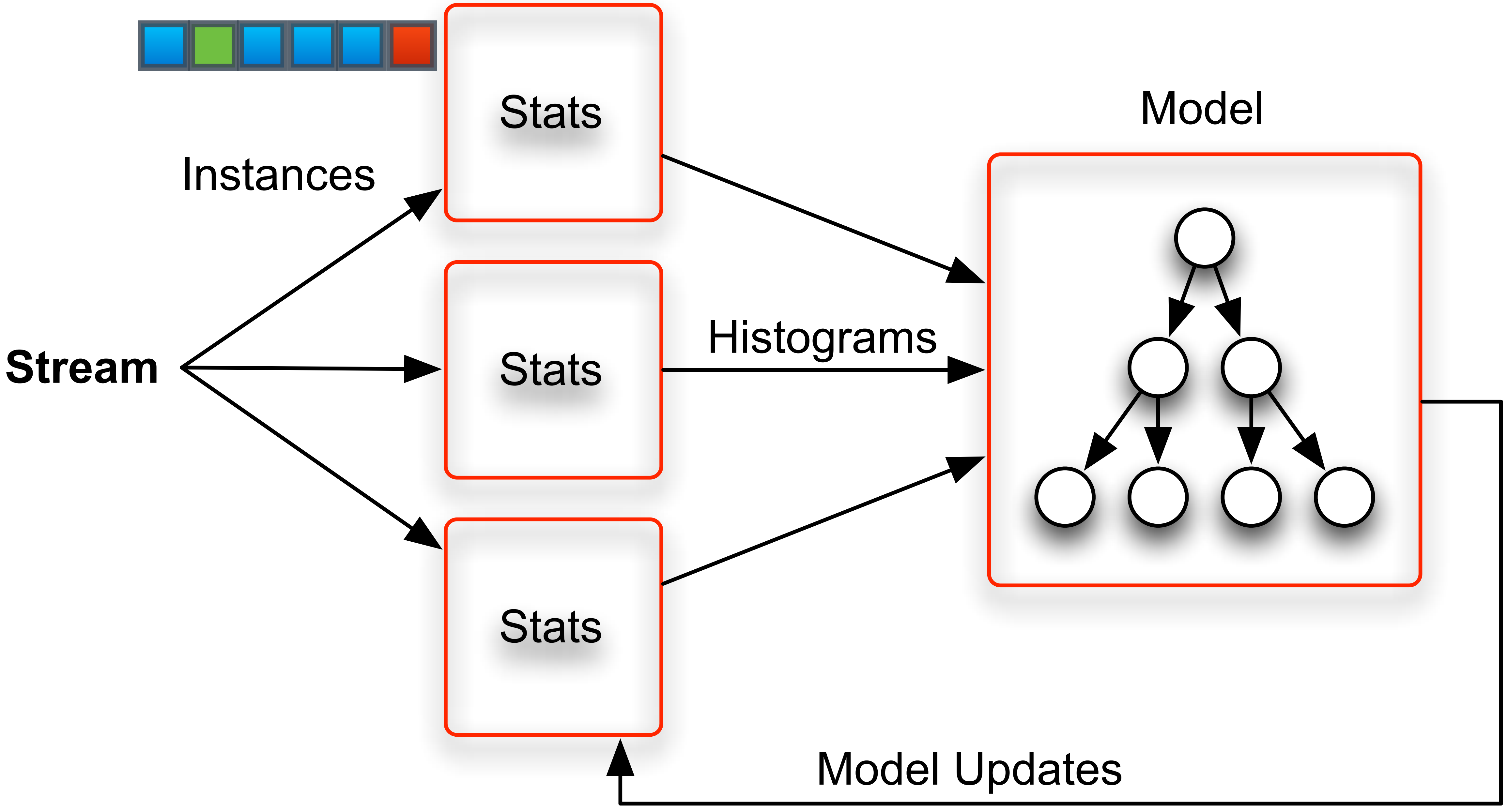
Horizontal Partitioning

Y. Ben-Haim, E. Tom-Tov: “A Streaming Parallel Decision Tree Algorithm”. JMLR (2010)



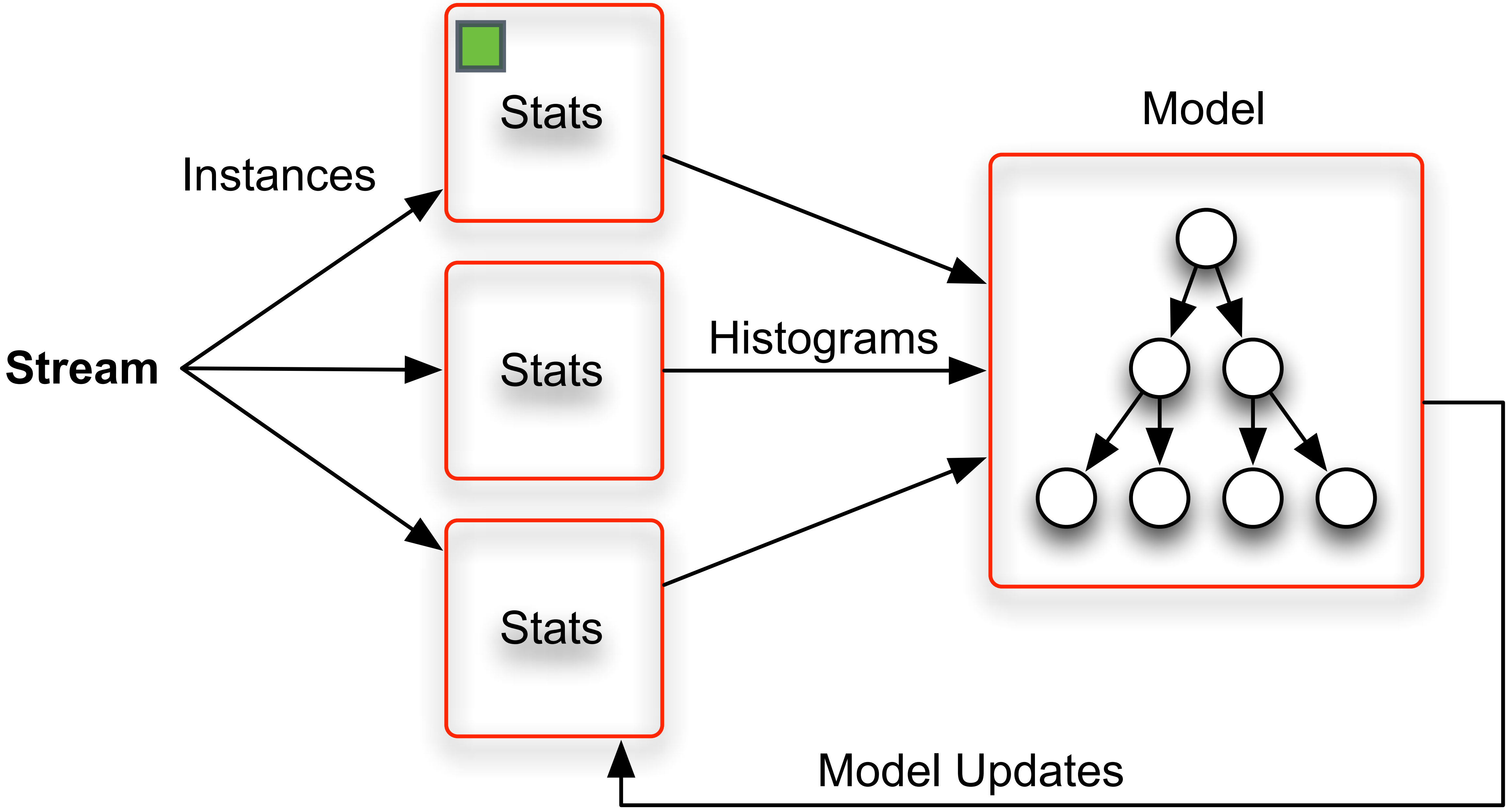
Horizontal Partitioning

Y. Ben-Haim, E. Tom-Tov: “A Streaming Parallel Decision Tree Algorithm”. JMLR (2010)



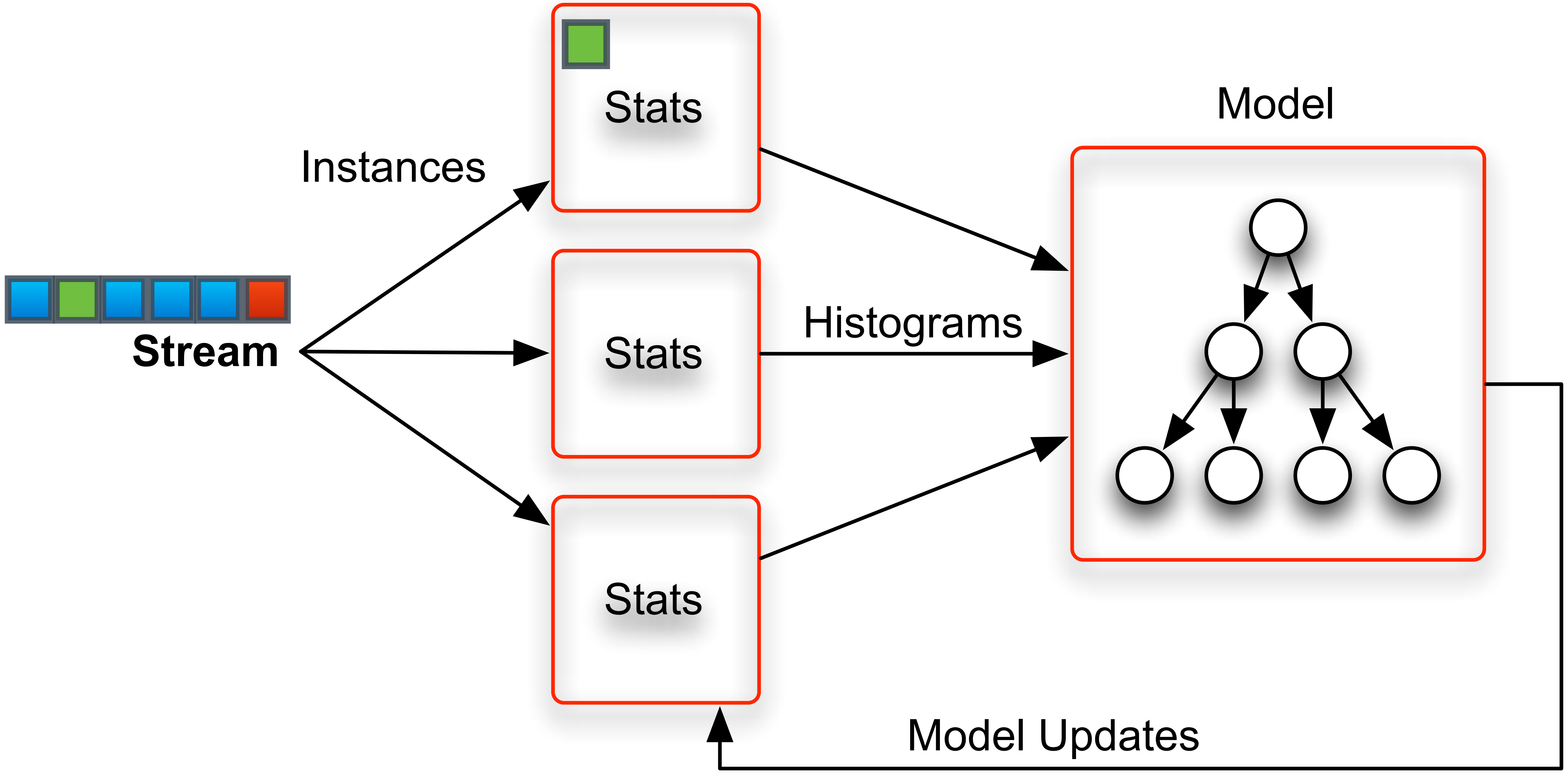
Horizontal Partitioning

Y. Ben-Haim, E. Tom-Tov: “A Streaming Parallel Decision Tree Algorithm”. JMLR (2010)



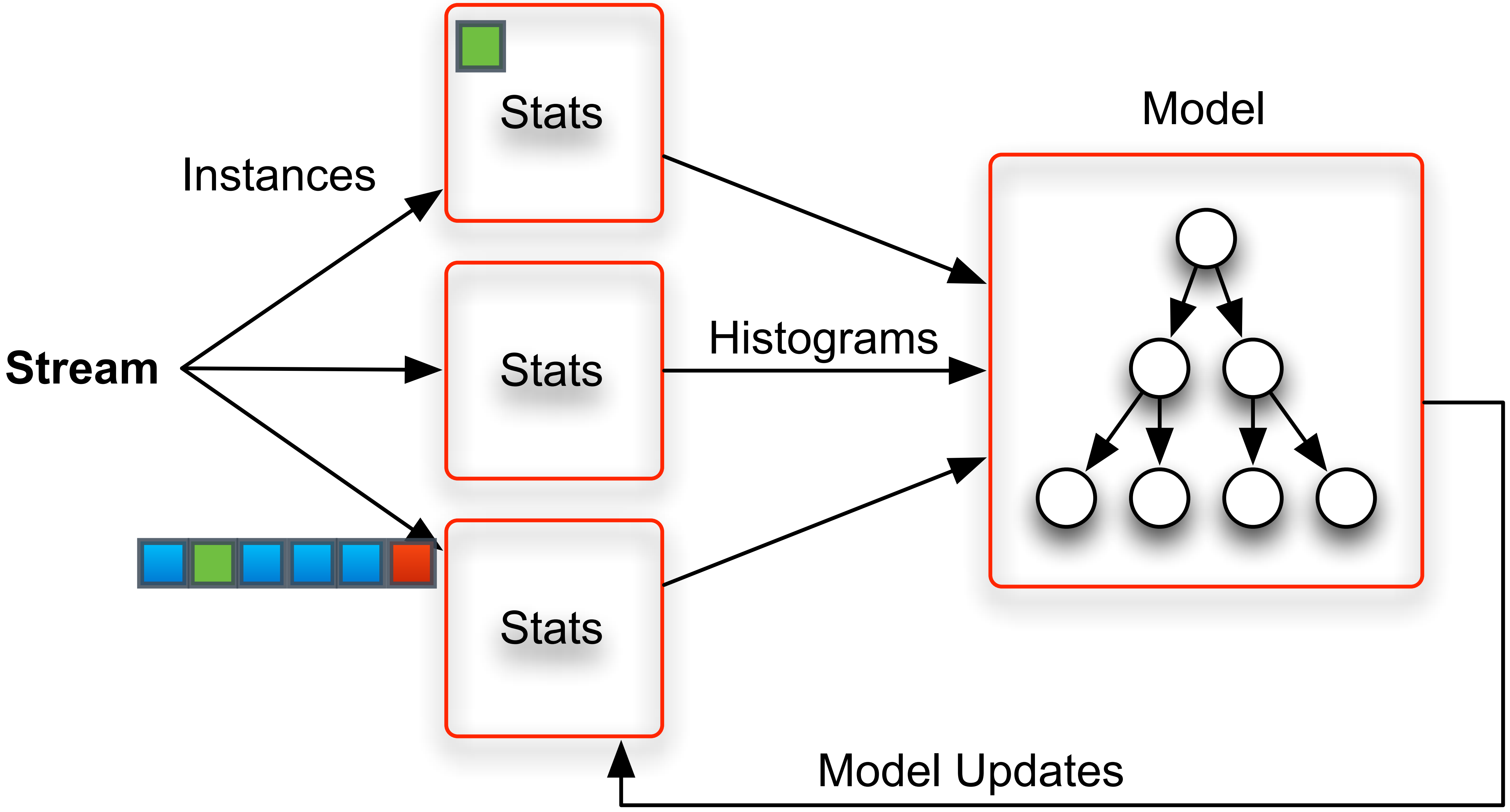
Horizontal Partitioning

Y. Ben-Haim, E. Tom-Tov: “A Streaming Parallel Decision Tree Algorithm”. JMLR (2010)



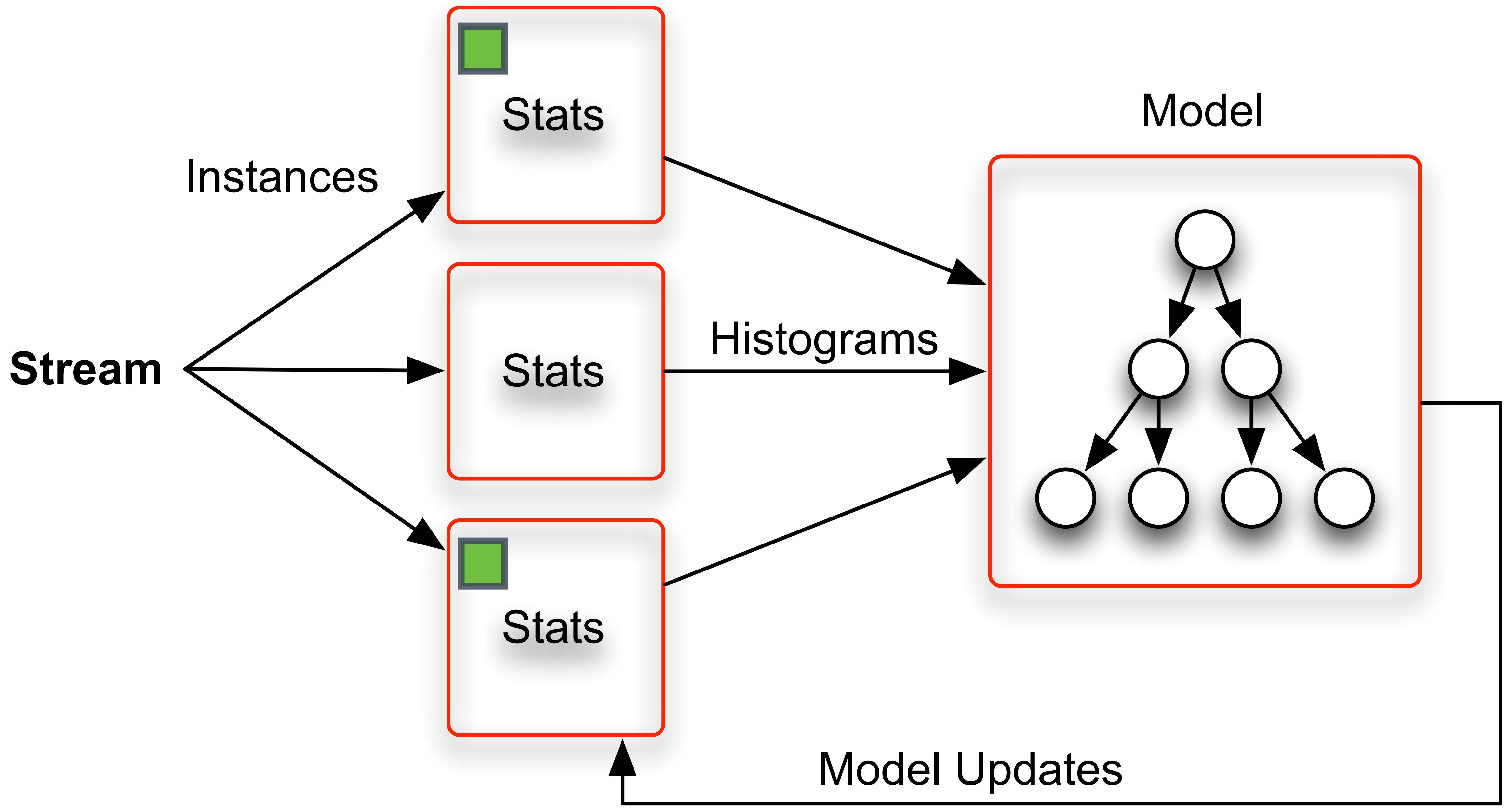
Horizontal Partitioning

Y. Ben-Haim, E. Tom-Tov: “A Streaming Parallel Decision Tree Algorithm”. JMLR (2010)



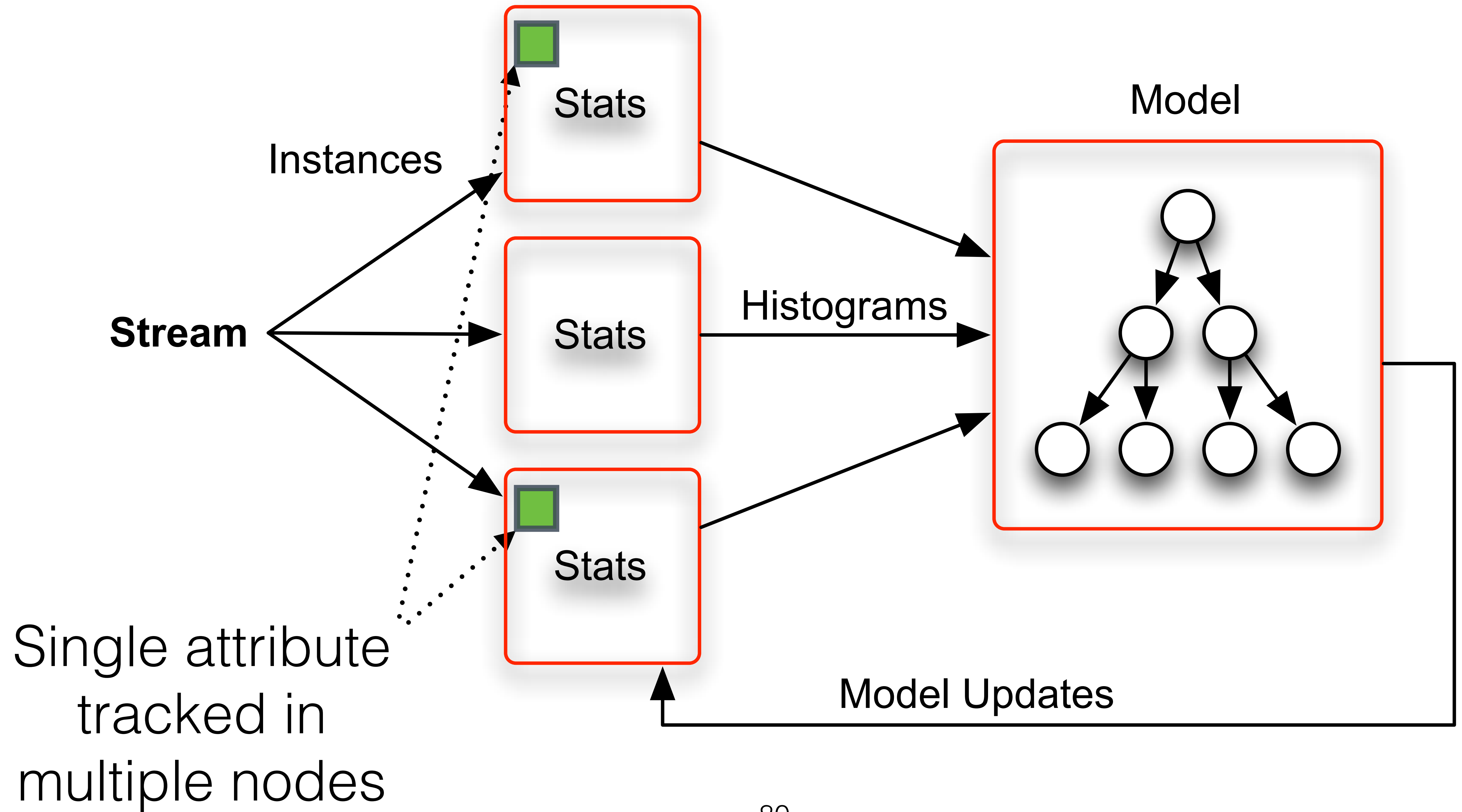
Horizontal Partitioning

Y. Ben-Haim, E. Tom-Tov: “A Streaming Parallel Decision Tree Algorithm”. JMLR (2010)



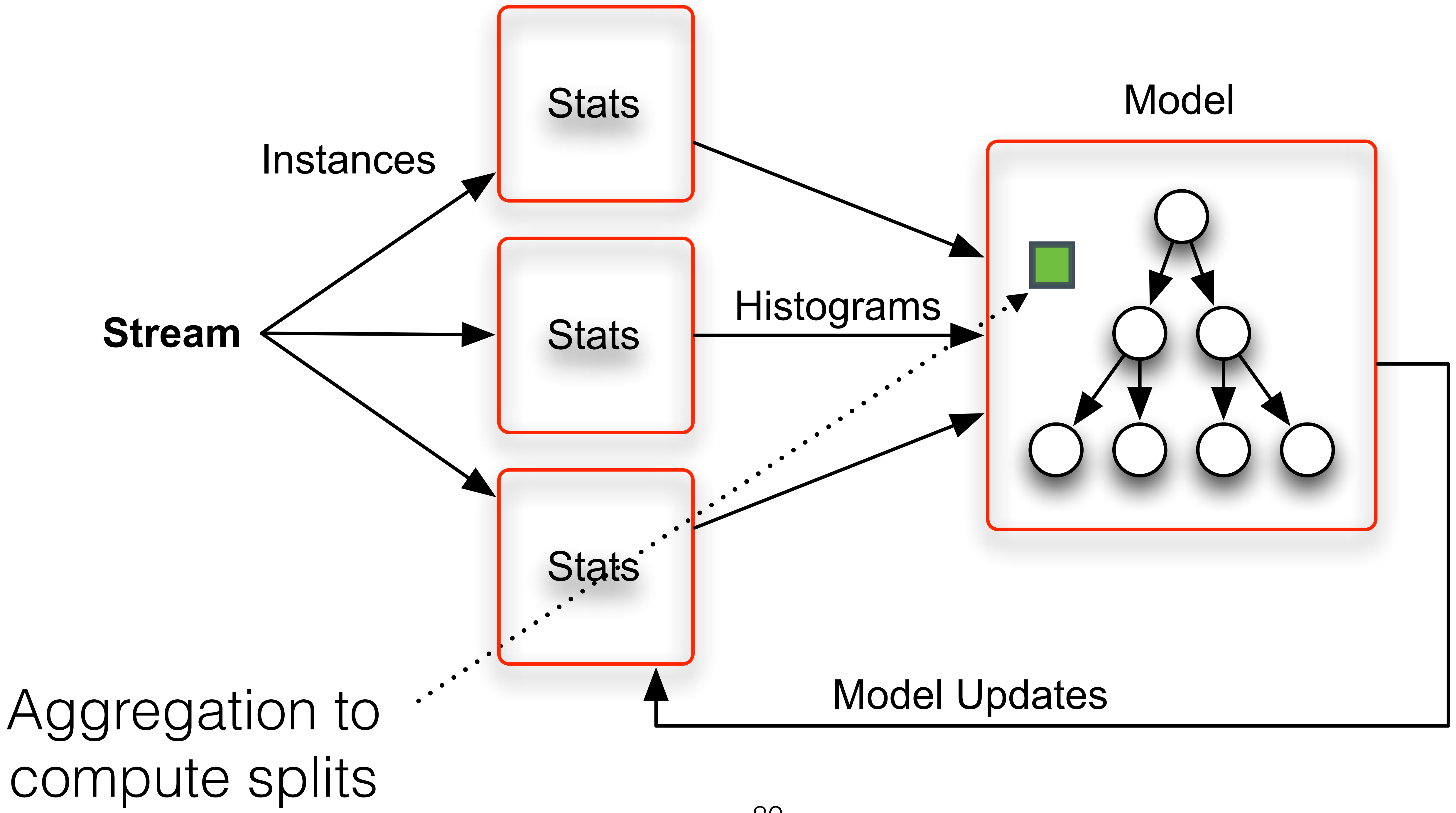
Horizontal Partitioning

Y. Ben-Haim, E. Tom-Tov: “A Streaming Parallel Decision Tree Algorithm”. JMLR (2010)



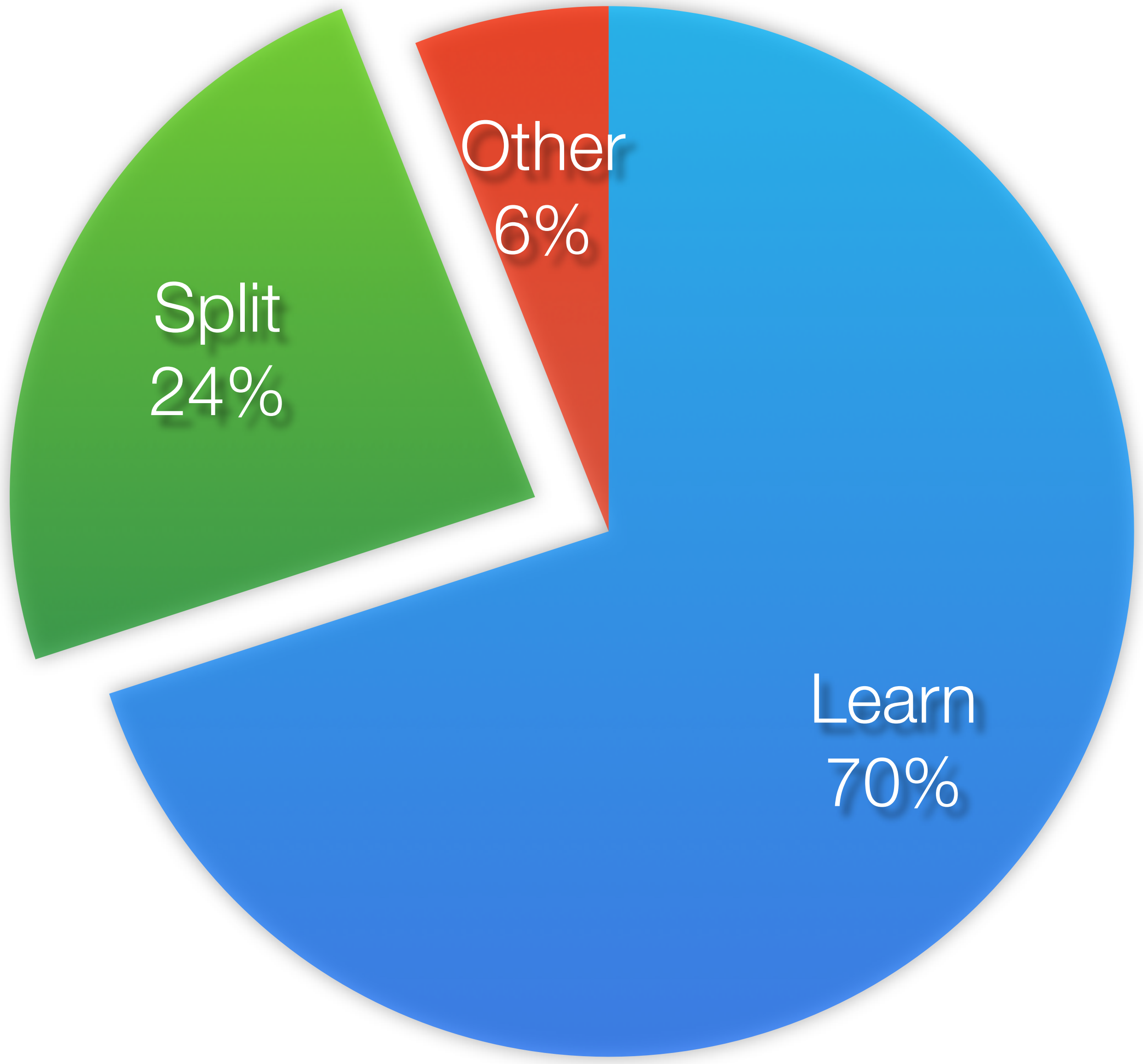
Horizontal Partitioning

Y. Ben-Haim, E. Tom-Tov: “A Streaming Parallel Decision Tree Algorithm”. JMLR (2010)



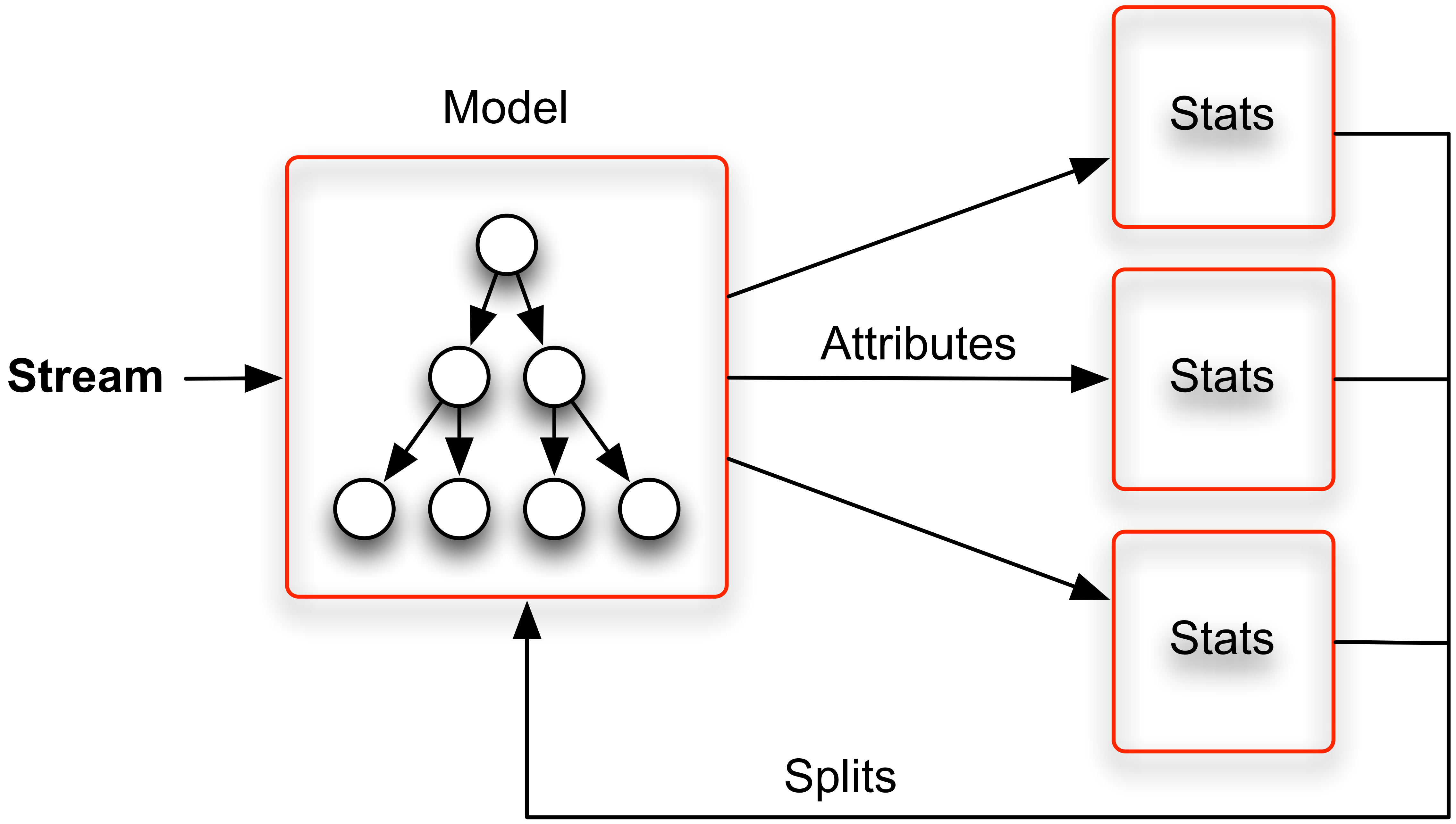
Hoeffding Tree Profiling

Training time for 100 nominal + 100 numeric attributes



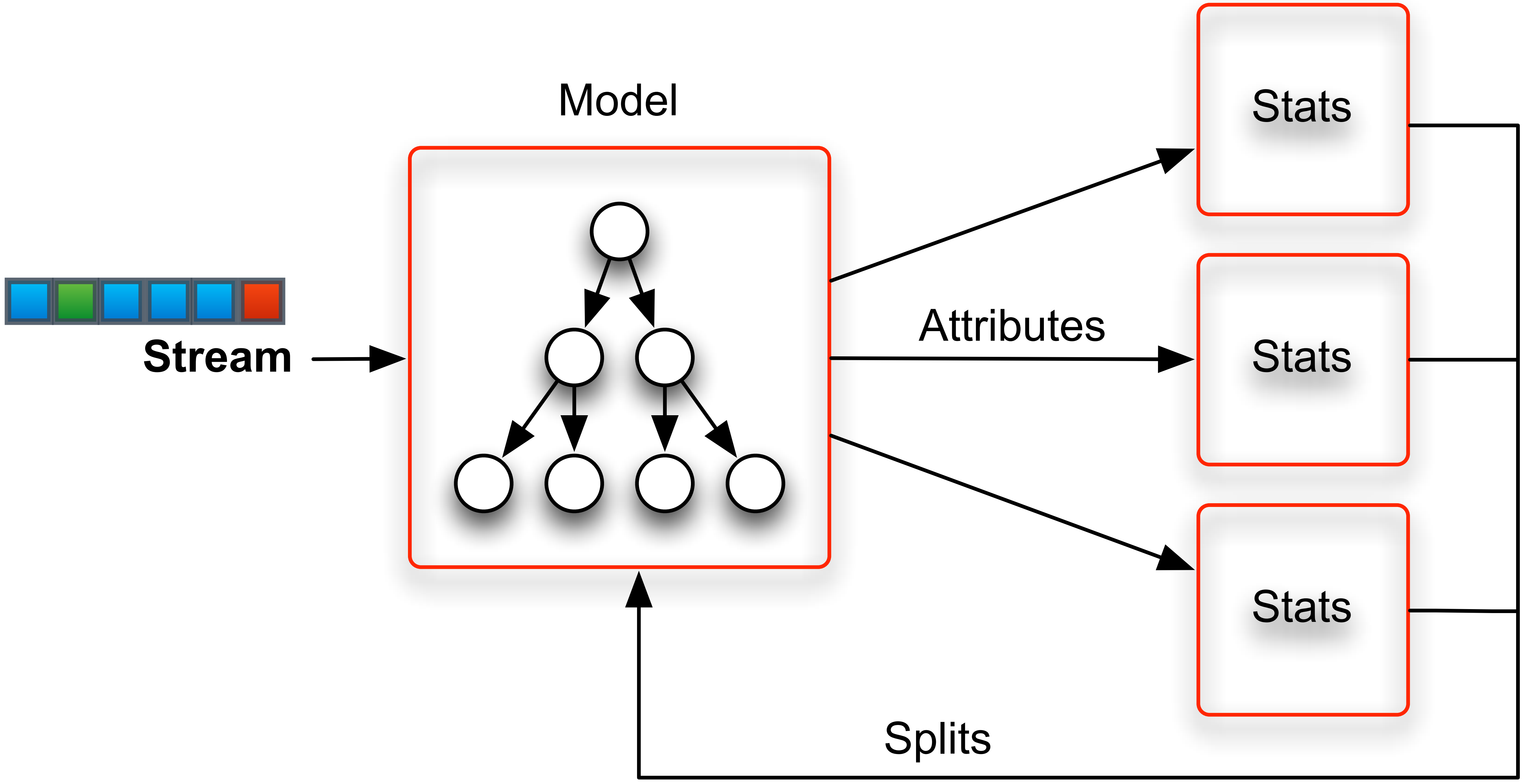
Vertical Partitioning

A. Murdopo, A. Bifet, G. De Francisci Morales, N. Kourtellis: “VHT: Vertical Hoeffding Tree”. Working paper (2014)



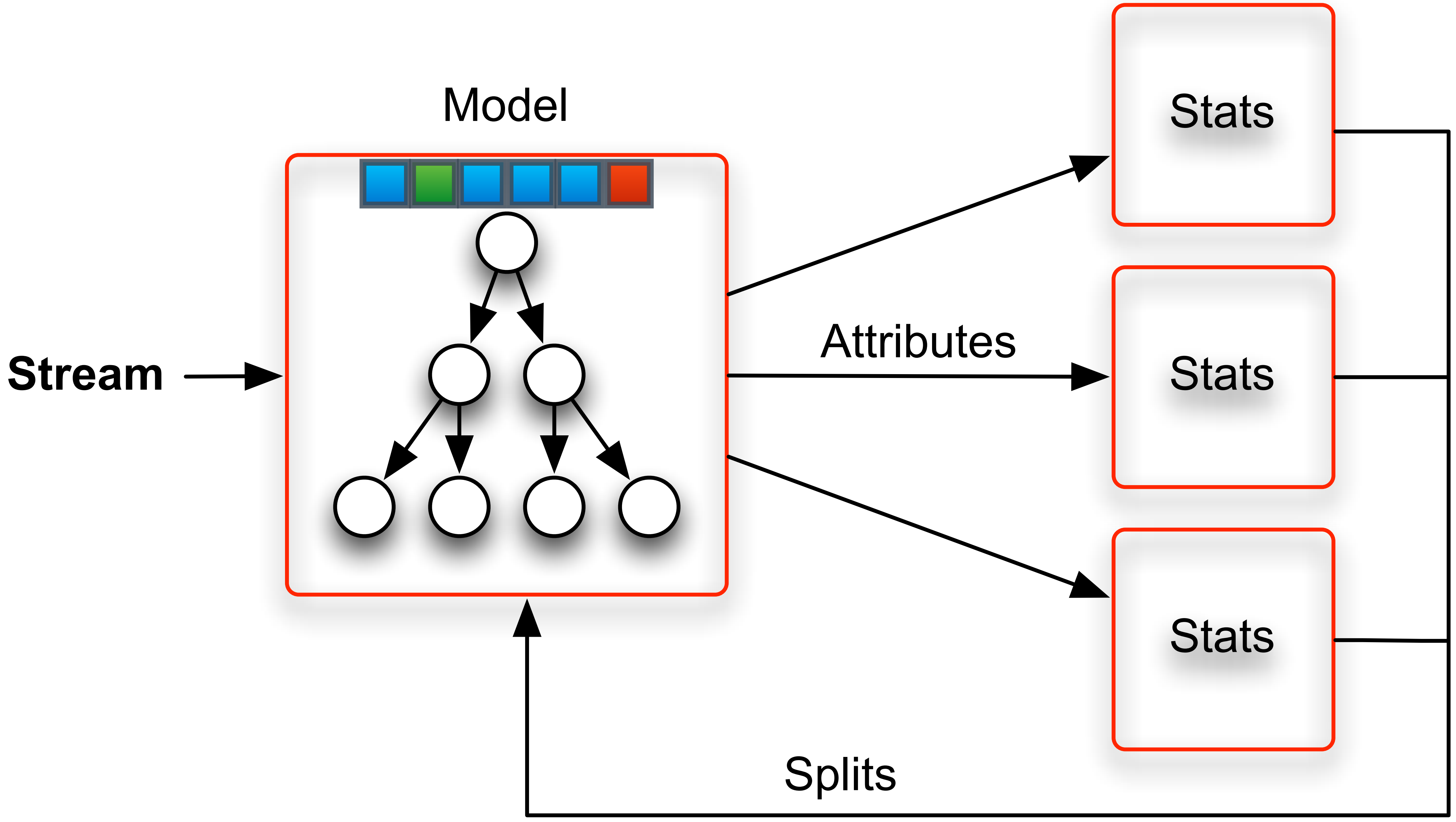
Vertical Partitioning

A. Murdopo, A. Bifet, G. De Francisci Morales, N. Kourtellis: "VHT: Vertical Hoeffding Tree". Working paper (2014)



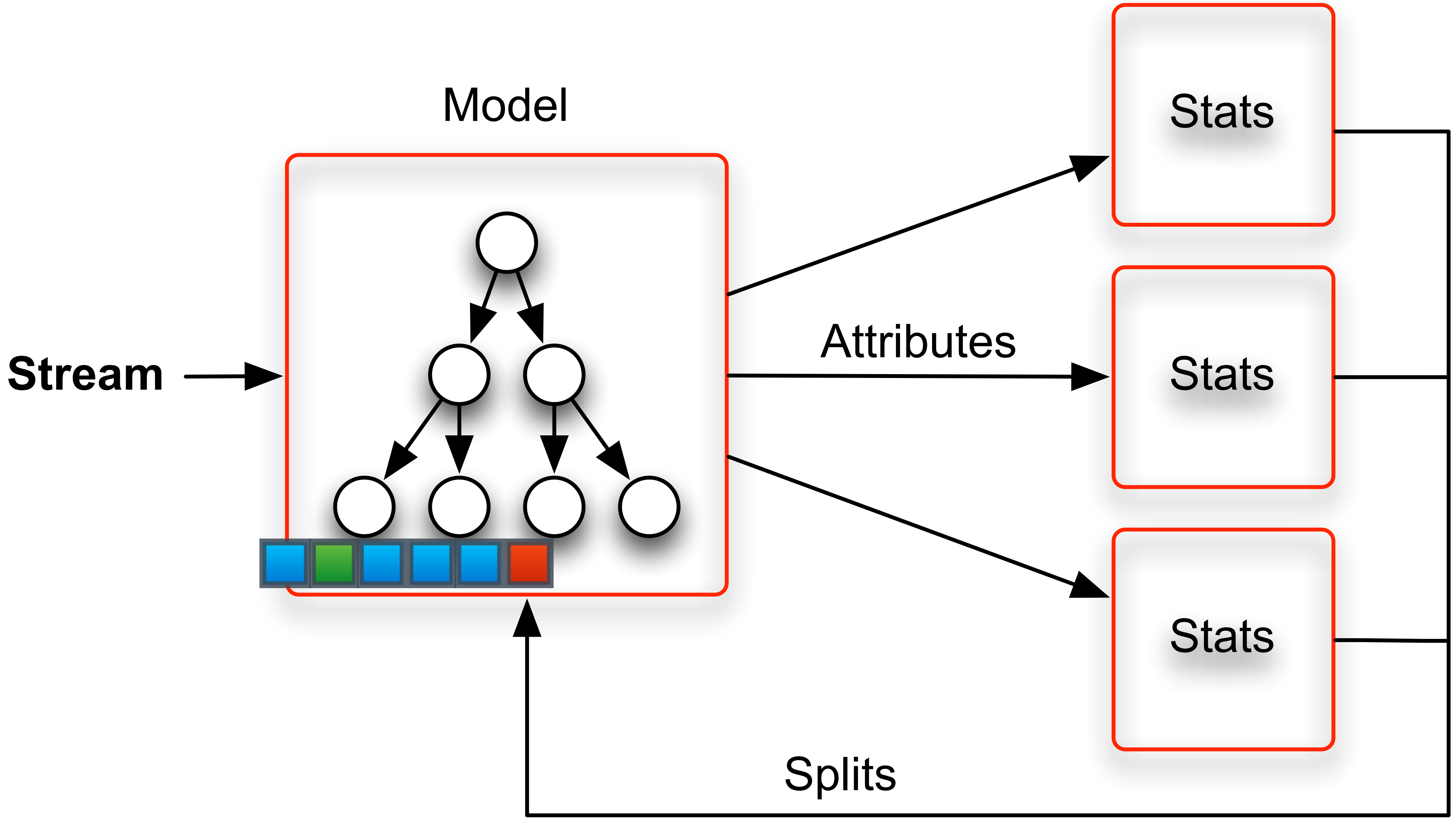
Vertical Partitioning

A. Murdopo, A. Bifet, G. De Francisci Morales, N. Kourtellis: "VHT: Vertical Hoeffding Tree". Working paper (2014)



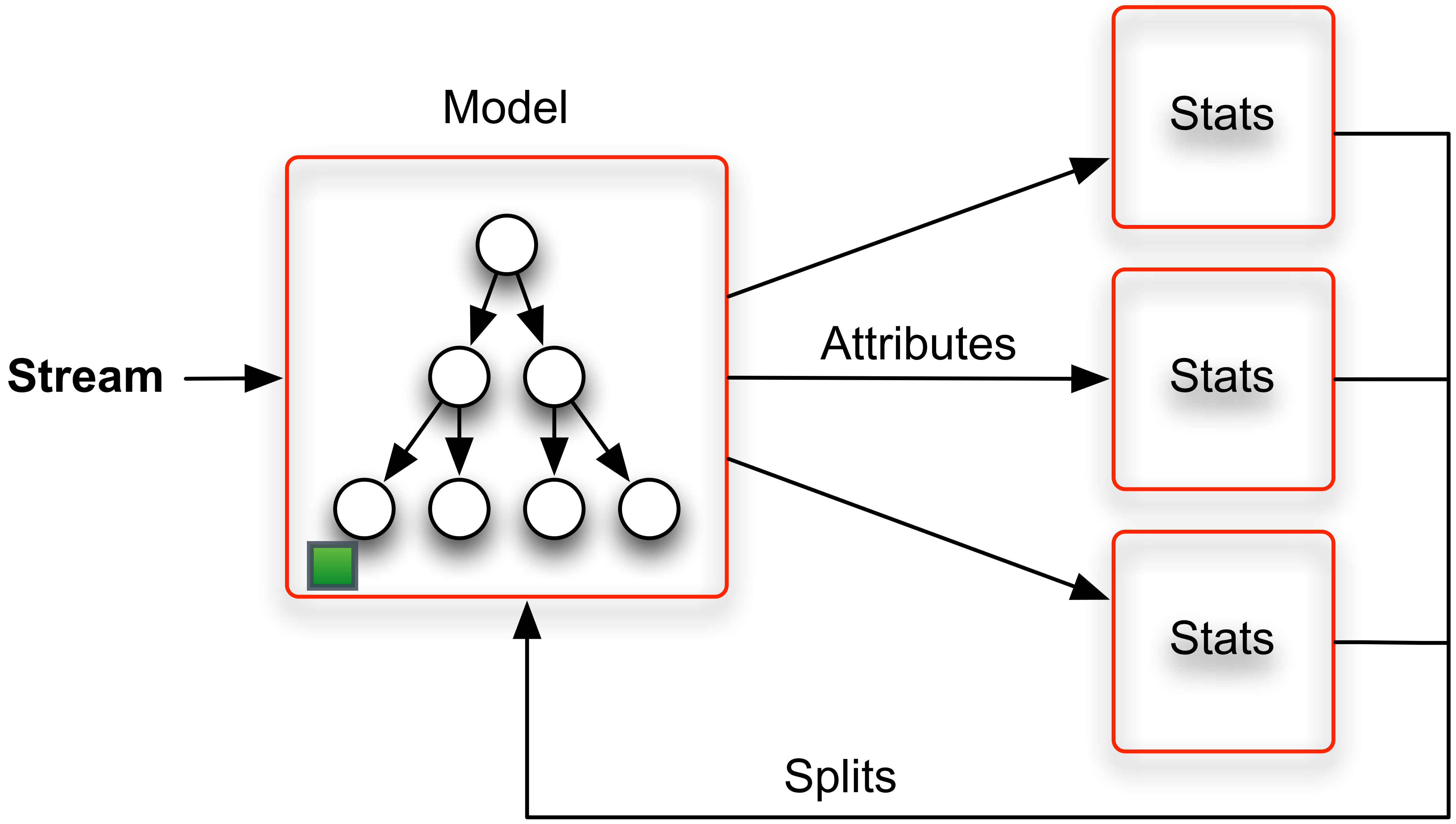
Vertical Partitioning

A. Murdopo, A. Bifet, G. De Francisci Morales, N. Kourtellis: “VHT: Vertical Hoeffding Tree”. Working paper (2014)



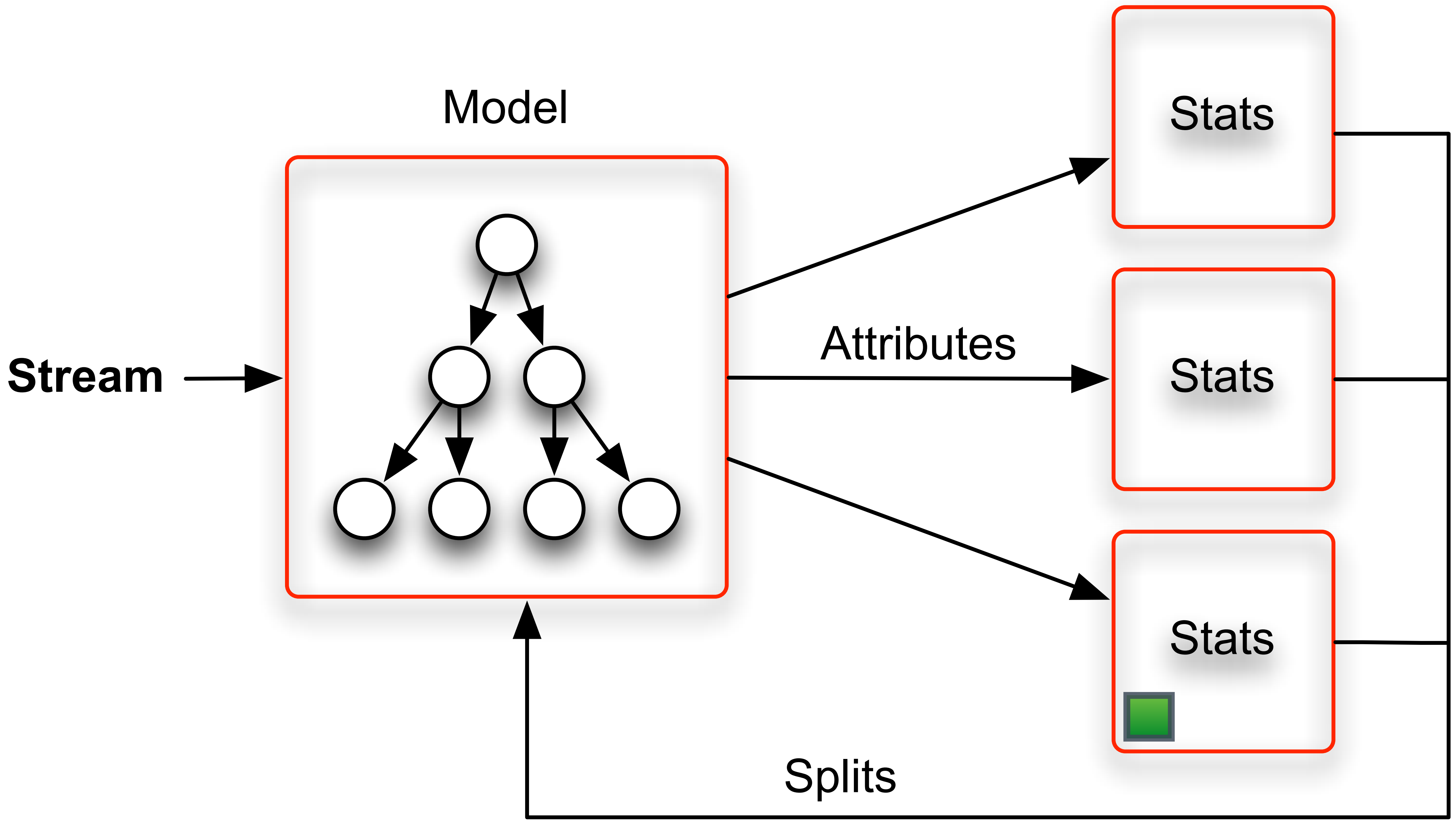
Vertical Partitioning

A. Murdopo, A. Bifet, G. De Francisci Morales, N. Kourtellis: "VHT: Vertical Hoeffding Tree". Working paper (2014)



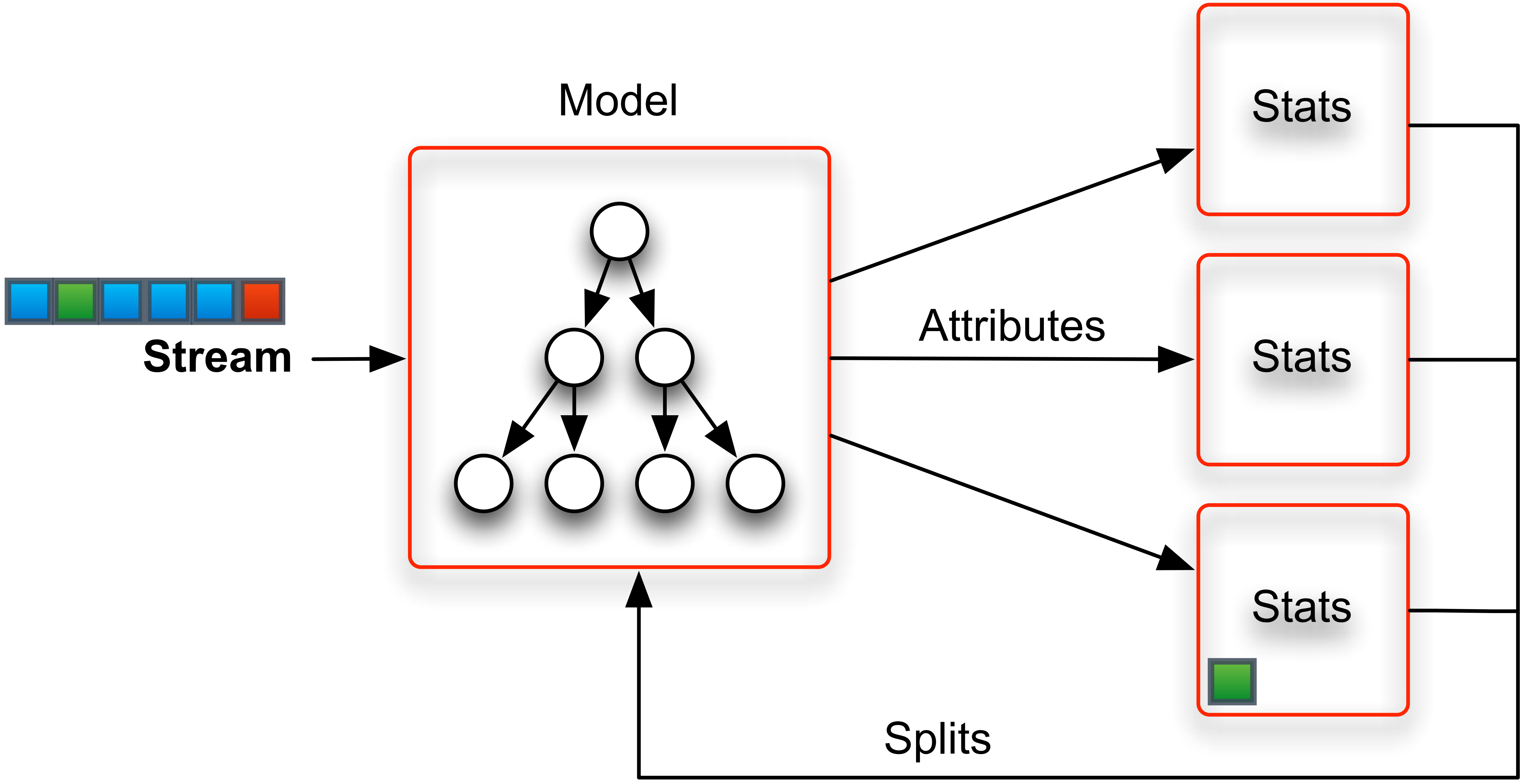
Vertical Partitioning

A. Murdopo, A. Bifet, G. De Francisci Morales, N. Kourtellis: “VHT: Vertical Hoeffding Tree”. Working paper (2014)



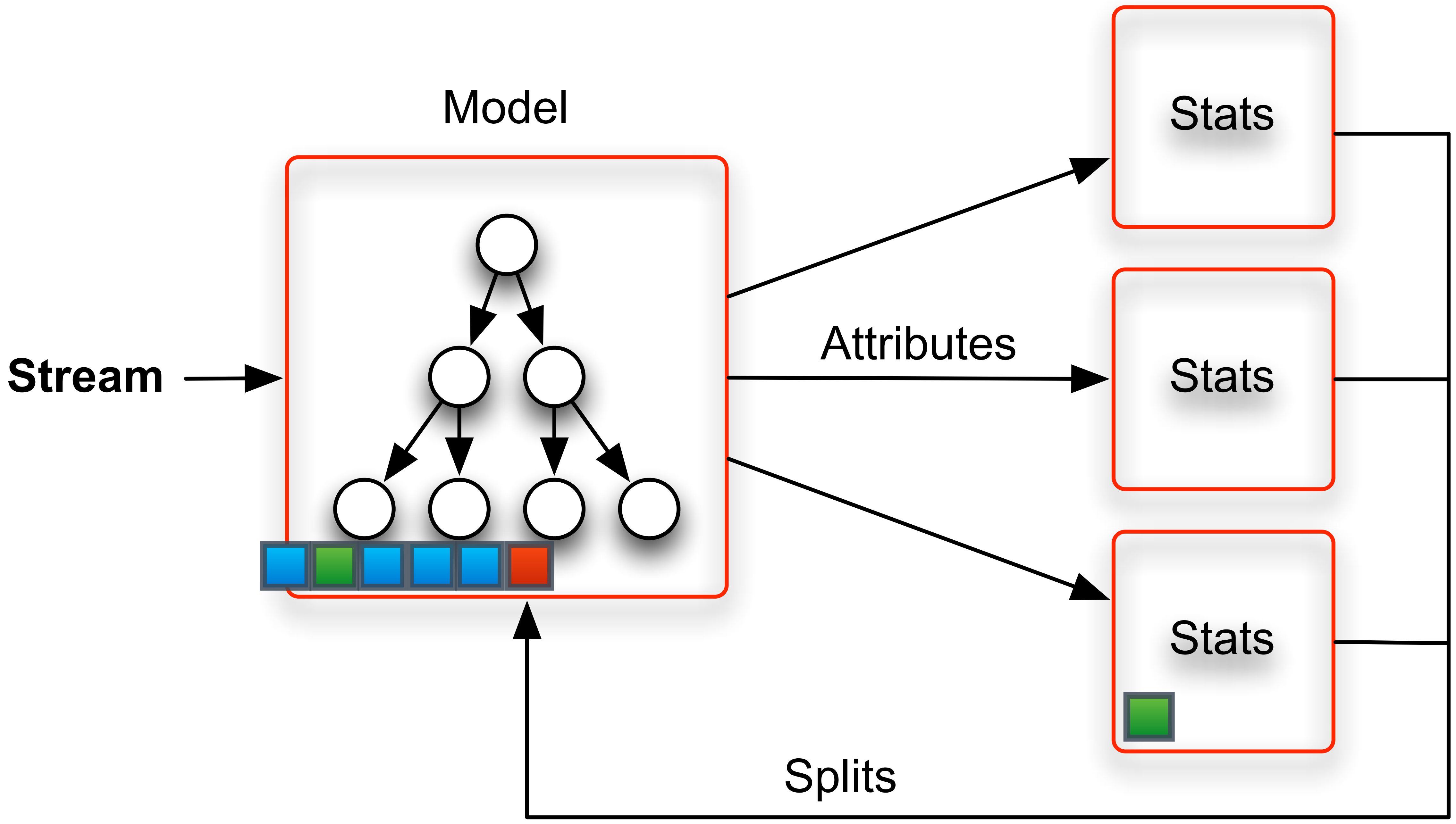
Vertical Partitioning

A. Murdopo, A. Bifet, G. De Francisci Morales, N. Kourtellis: “VHT: Vertical Hoeffding Tree”. Working paper (2014)



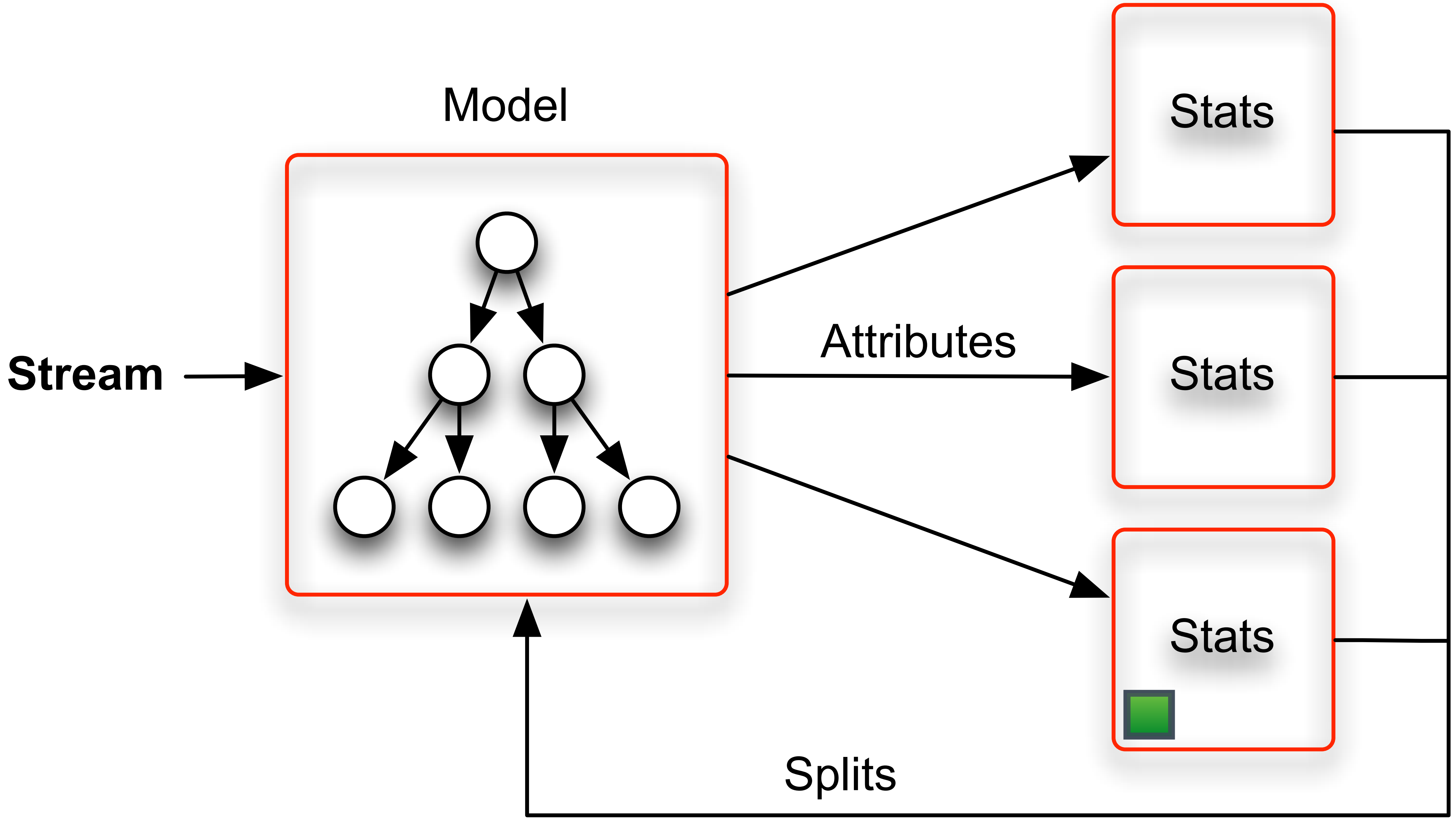
Vertical Partitioning

A. Murdopo, A. Bifet, G. De Francisci Morales, N. Kourtellis: “VHT: Vertical Hoeffding Tree”. Working paper (2014)



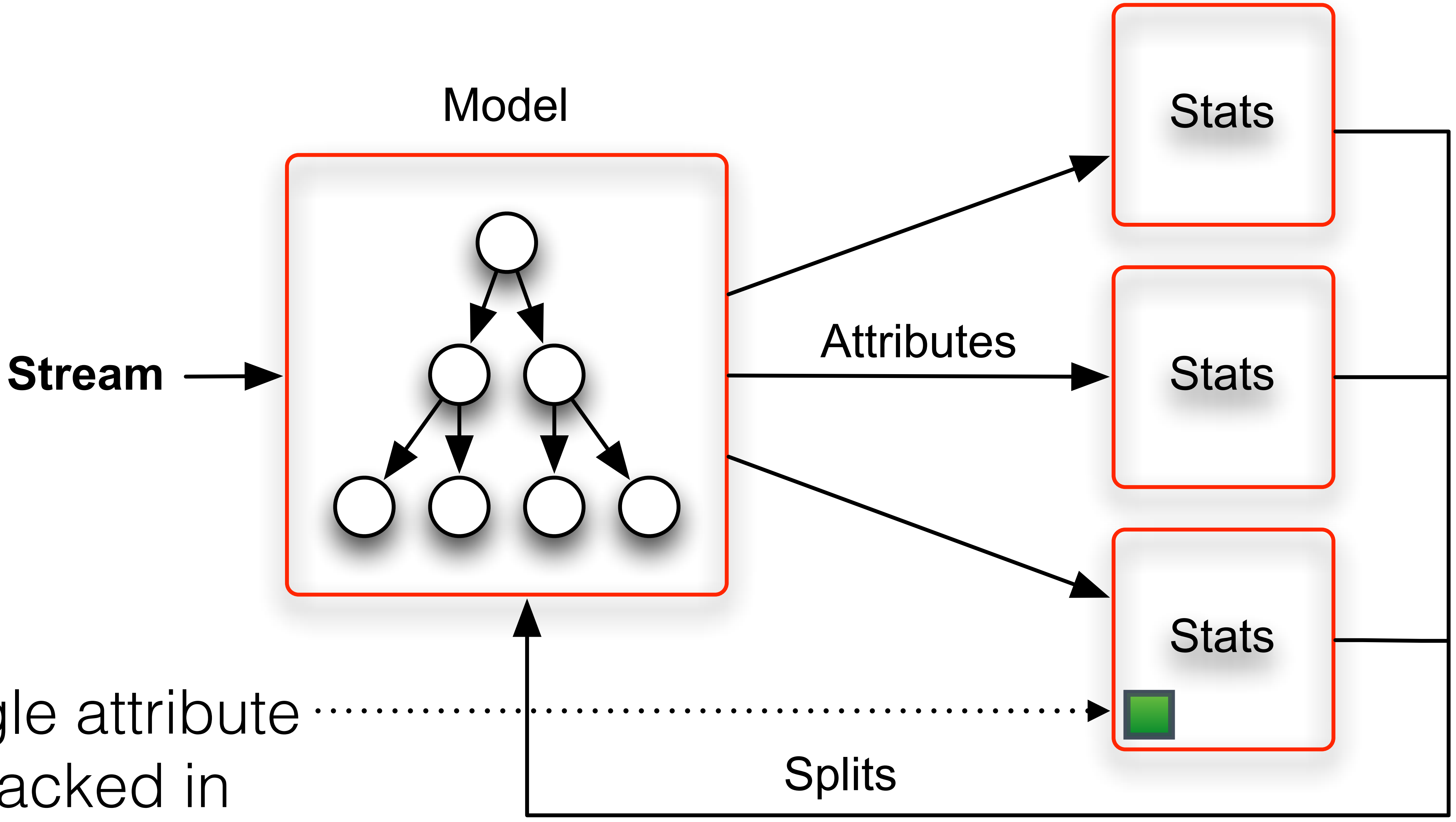
Vertical Partitioning

A. Murdopo, A. Bifet, G. De Francisci Morales, N. Kourtellis: “VHT: Vertical Hoeffding Tree”. Working paper (2014)



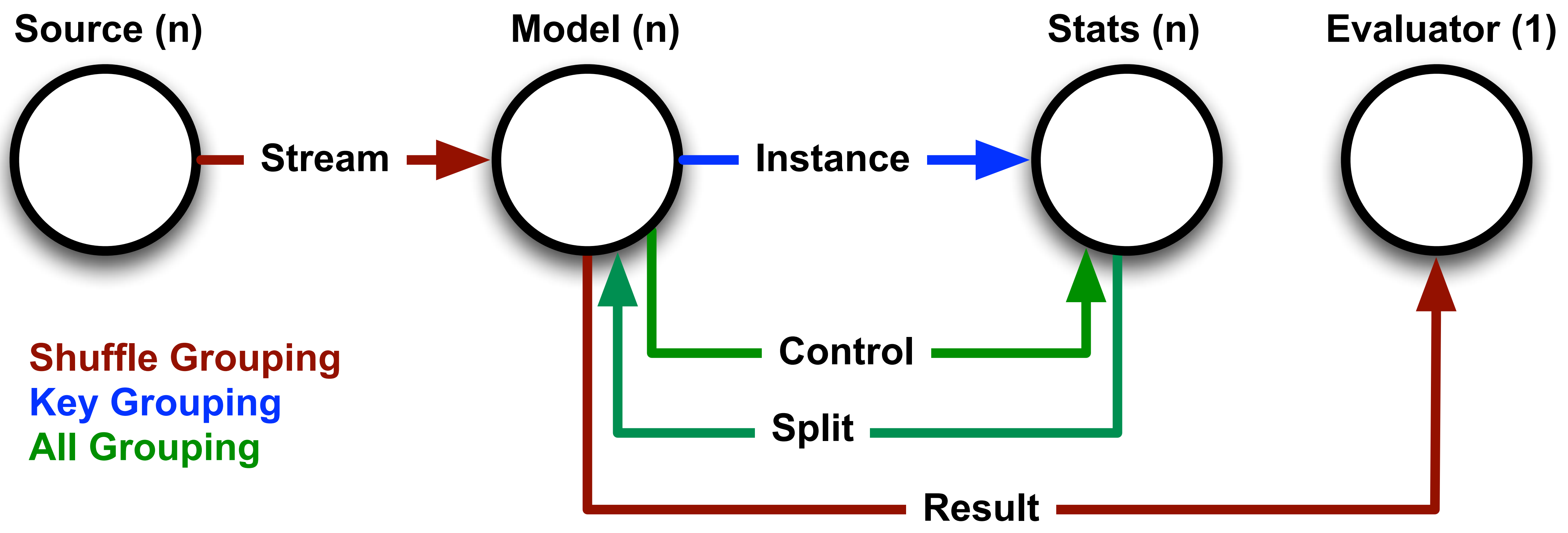
Vertical Partitioning

A. Murdopo, A. Bifet, G. De Francisci Morales, N. Kourtellis: "VHT: Vertical Hoeffding Tree". Working paper (2014)



Single attribute tracked in single node

Vertical Hoeffding Tree



Shuffle Grouping
Key Grouping
All Grouping

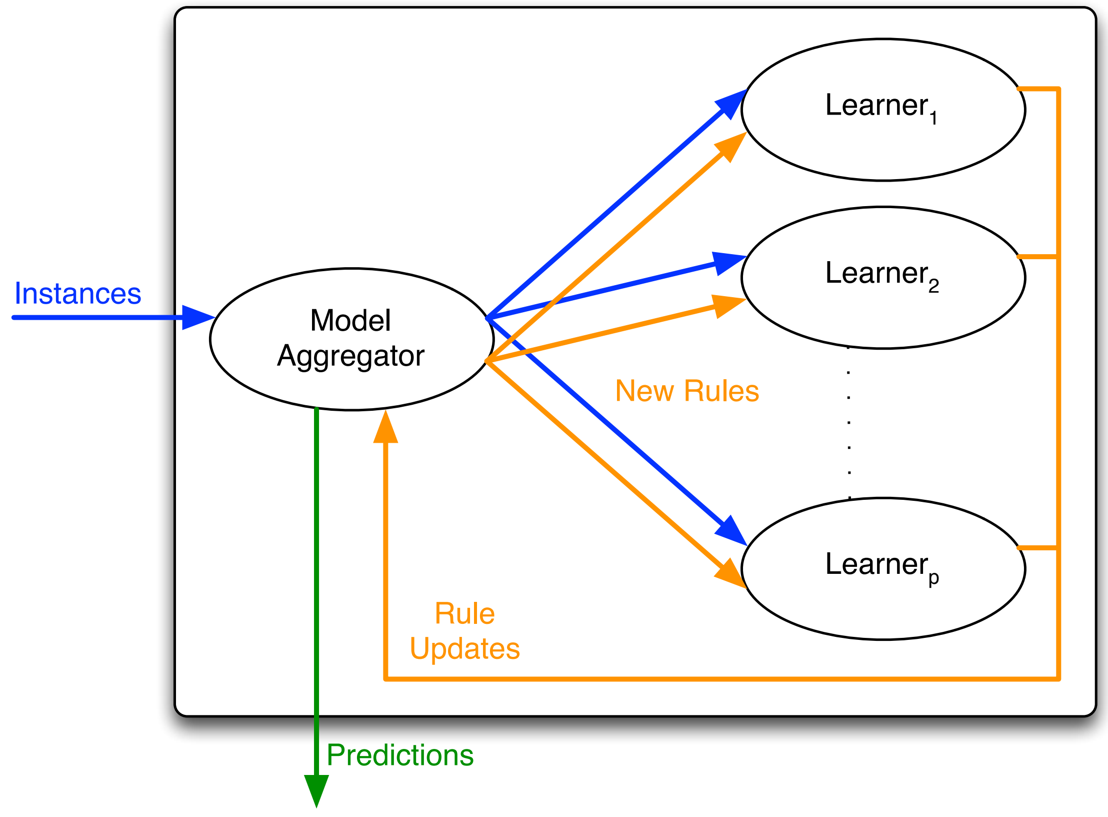
Advantages of Vertical Parallelism

- High number of attributes => high level of parallelism (e.g., documents)
- vs. task parallelism
    - Parallelism observed immediately
- vs. horizontal parallelism
    - Reduced memory usage (no model replication)
    - Parallelized split computation

Regression

VAMR

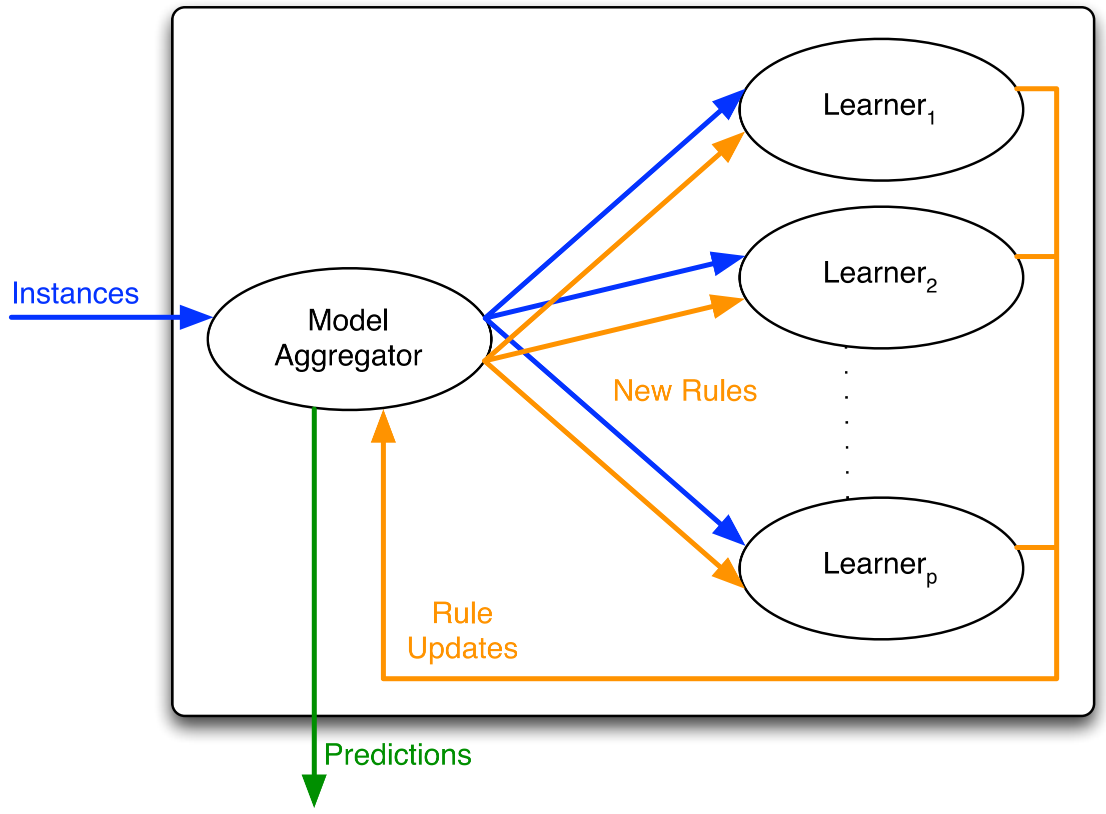
A. T. Vu, G. De Francisci Morales, J. Gama, A. Bifet: “Distributed Adaptive Model Rules for Mining Big Data Streams”. BigData ‘14



VAMR

A. T. Vu, G. De Francisci Morales, J. Gama, A. Bifet: “Distributed Adaptive Model Rules for Mining Big Data Streams”. BigData ‘14

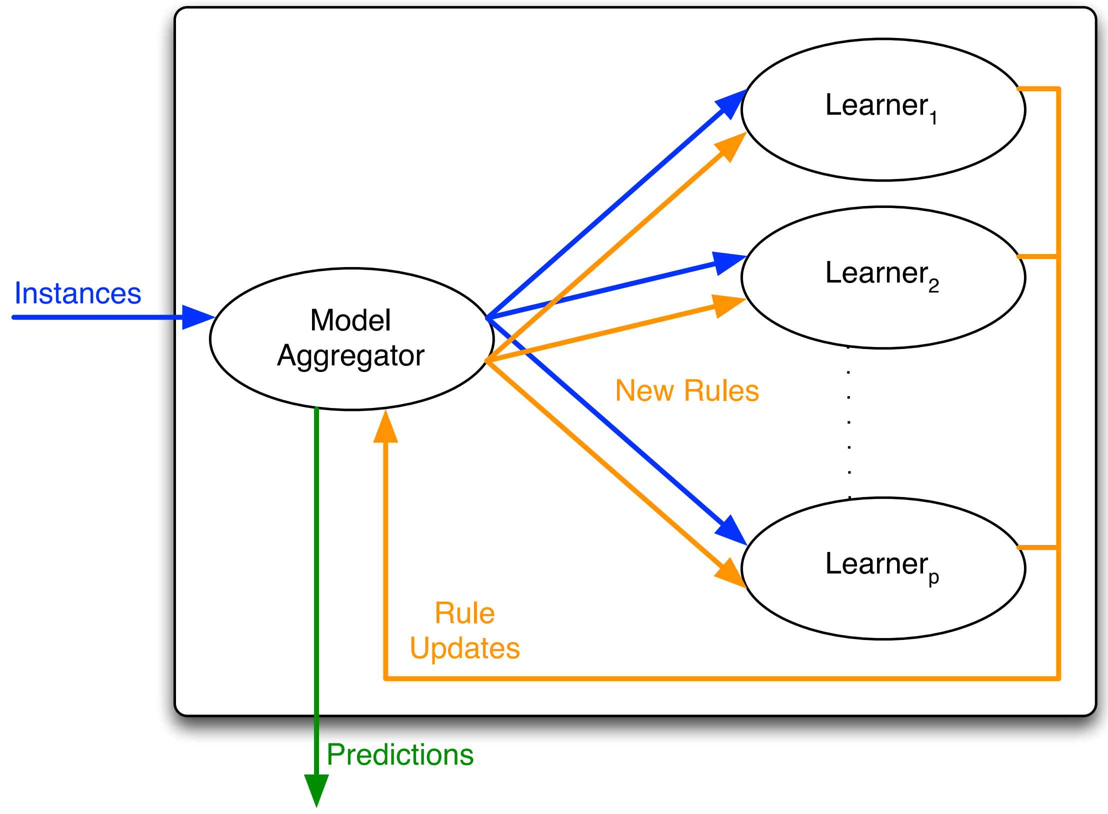
- Vertical AMRules



VAMR

A. T. Vu, G. De Francisci Morales, J. Gama, A. Bifet: “Distributed Adaptive Model Rules for Mining Big Data Streams”. BigData ‘14

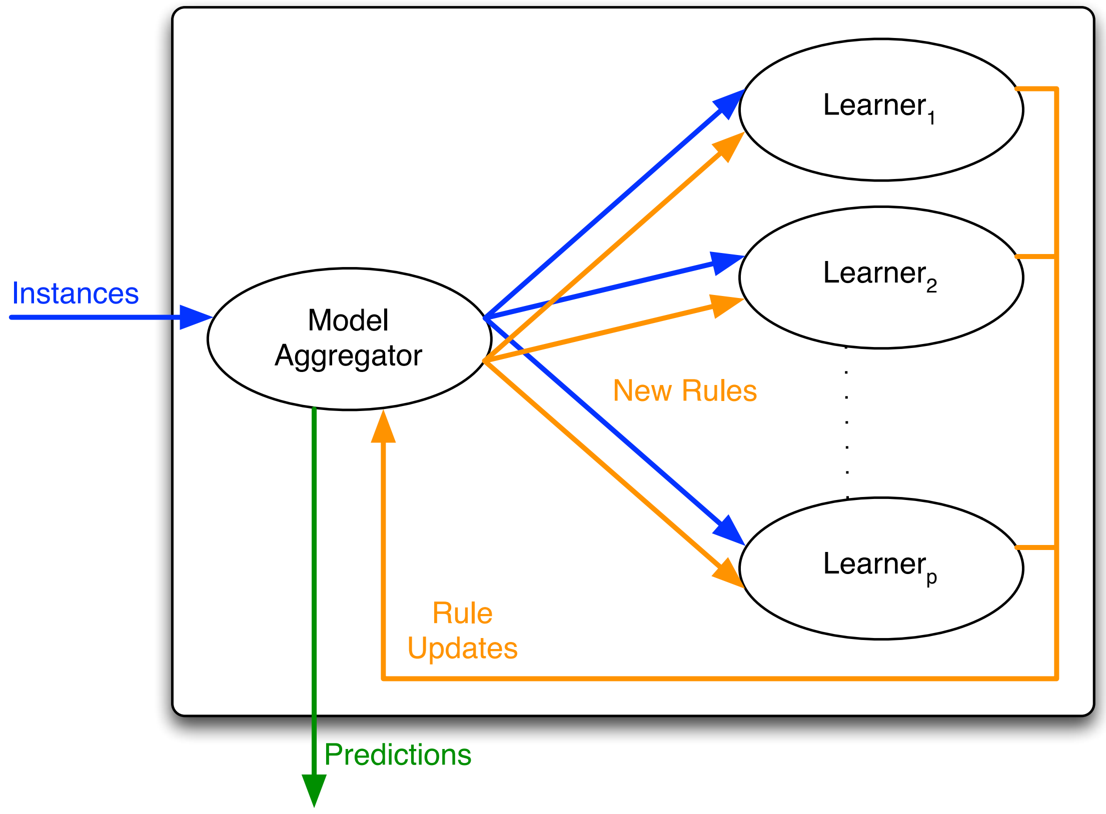
- Vertical AMRules
- Model: rule body + head
    - Target mean updated continuously with covered instances for predictions
    - Default rule (creates new rules)



VAMR

A. T. Vu, G. De Francisci Morales, J. Gama, A. Bifet: “Distributed Adaptive Model Rules for Mining Big Data Streams”. BigData ‘14

- Vertical AMRules
- Model: rule body + head
    - Target mean updated continuously with covered instances for predictions
    - Default rule (creates new rules)
- Learner: statistics
    - Vertical: Learner tracks statistics of independent subset of rules
    - One rule tracked by only one Learner
    - Model -> Learner: key grouping on rule ID



HAMR

A. T. Vu, G. De Francisci Morales, J. Gama, A. Bifet: “Distributed Adaptive Model Rules for Mining Big Data Streams”. BigData ‘14

HAMR

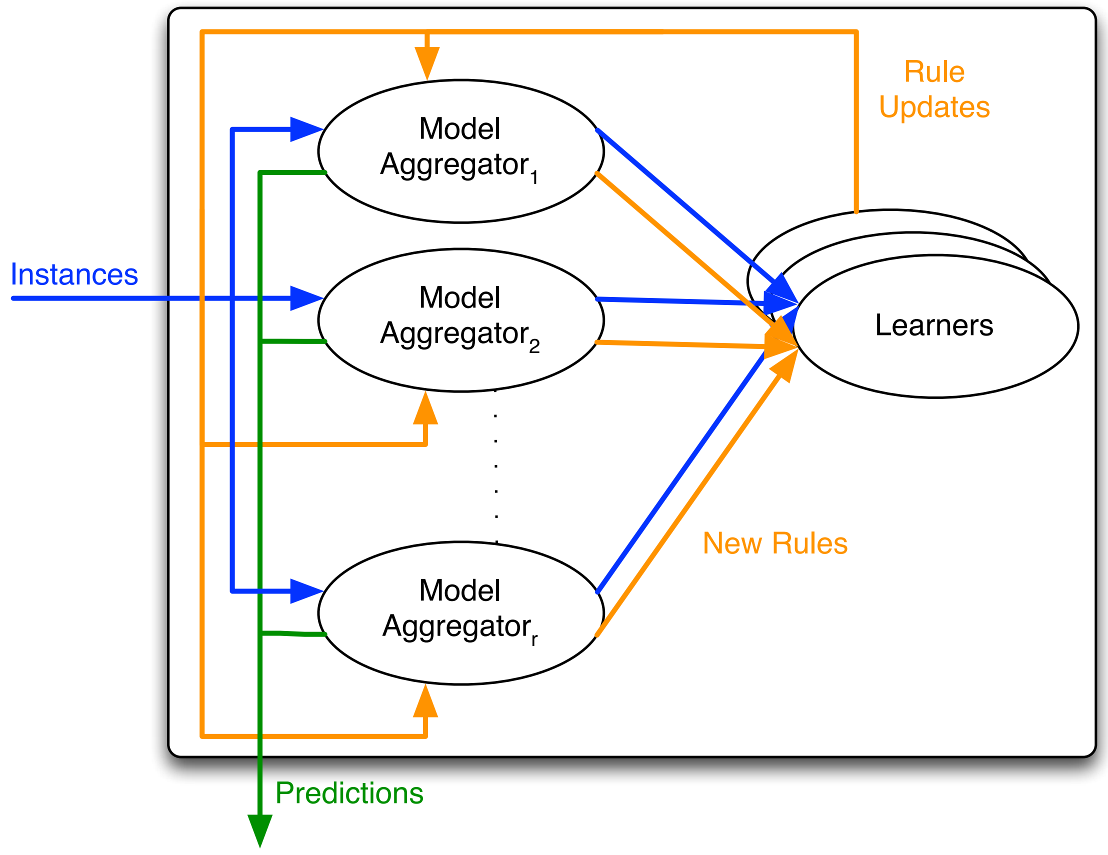
A. T. Vu, G. De Francisci Morales, J. Gama, A. Bifet: “Distributed Adaptive Model Rules for Mining Big Data Streams”. BigData ‘14

- VAMR single model is bottleneck

HAMR

A. T. Vu, G. De Francisci Morales, J. Gama, A. Bifet: “Distributed Adaptive Model Rules for Mining Big Data Streams”. BigData ‘14

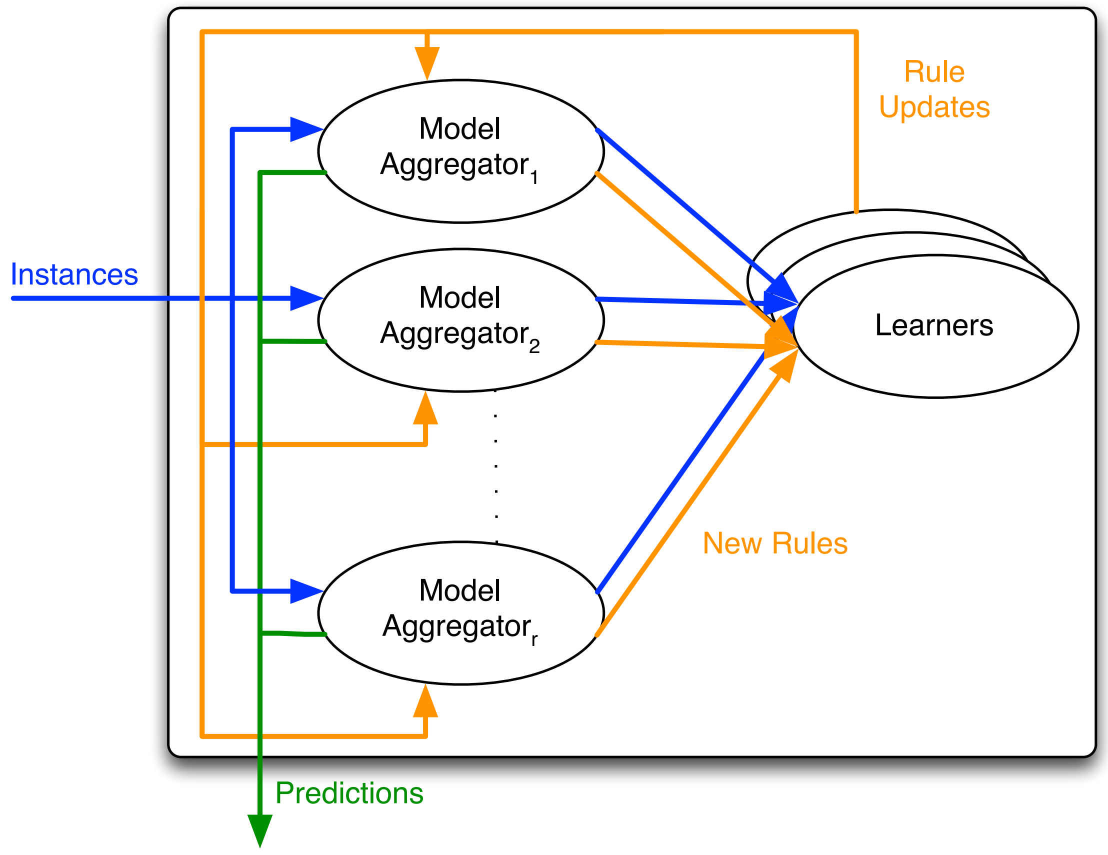
- VAMR single model is bottleneck
- Hybrid AMRules
  (Vertical + Horizontal)



HAMR

A. T. Vu, G. De Francisci Morales, J. Gama, A. Bifet: “Distributed Adaptive Model Rules for Mining Big Data Streams”. BigData ‘14

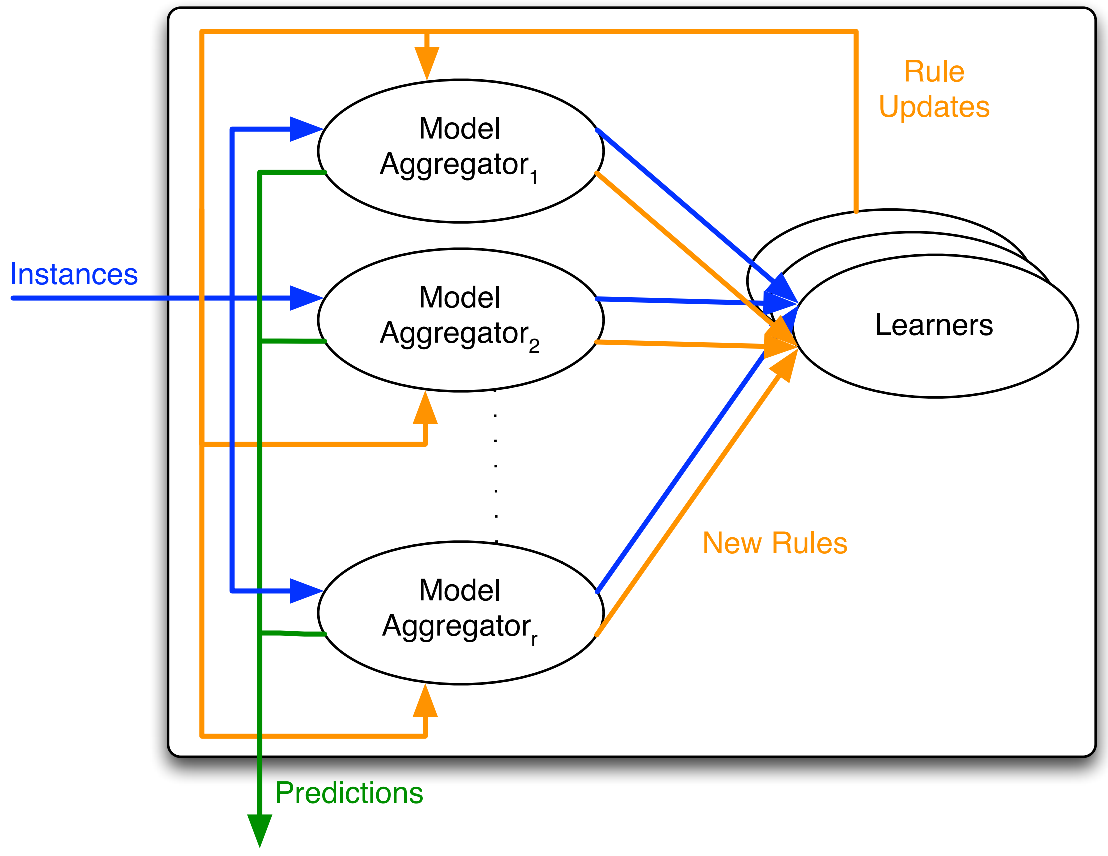
- VAMR single model is bottleneck
- Hybrid AMRules (Vertical + Horizontal)
    - Shuffle among multiple Models for parallelism



HAMR

A. T. Vu, G. De Francisci Morales, J. Gama, A. Bifet: “Distributed Adaptive Model Rules for Mining Big Data Streams”. BigData ‘14

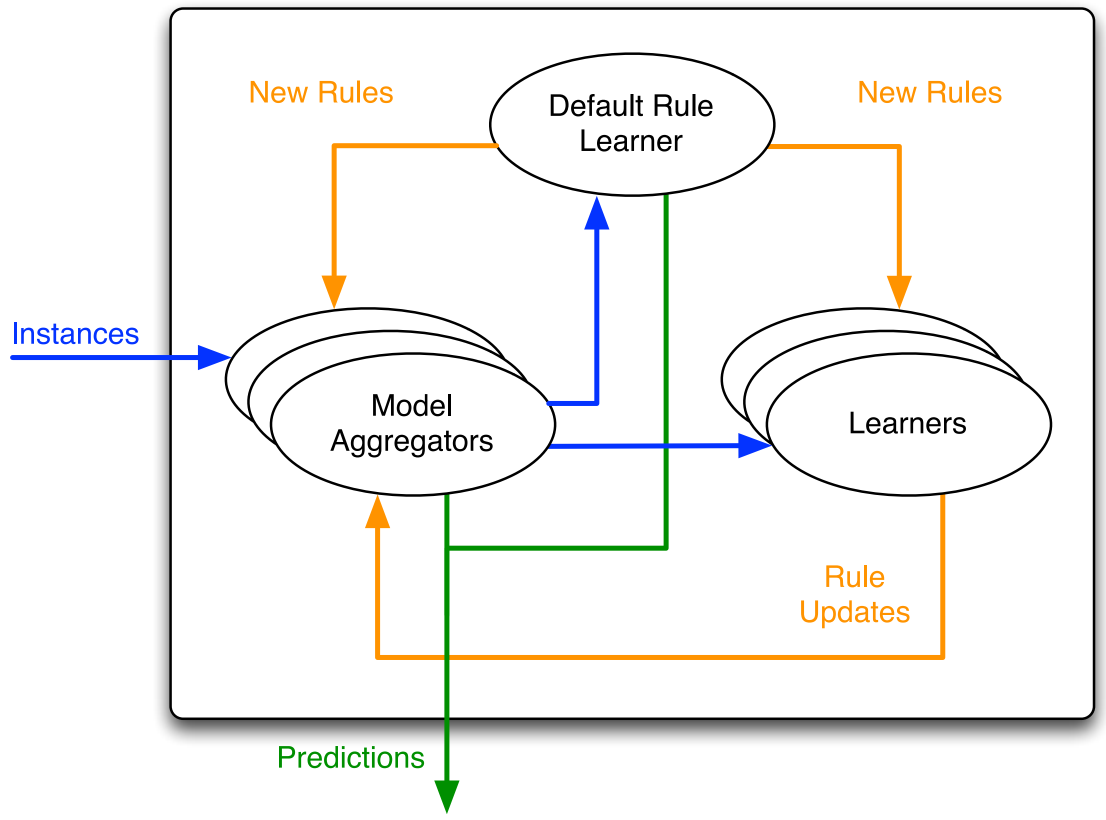
- VAMR single model is bottleneck
- Hybrid AMRules (Vertical + Horizontal)
  - Shuffle among multiple Models for parallelism
- Problem: distributed default rule decreases performance



HAMR

A. T. Vu, G. De Francisci Morales, J. Gama, A. Bifet: “Distributed Adaptive Model Rules for Mining Big Data Streams”. BigData ‘14

- VAMR single model is bottleneck
- Hybrid AMRules (Vertical + Horizontal)
    - Shuffle among multiple Models for parallelism
- Problem: distributed default rule decreases performance
    - Separate dedicate Learner for default rule



Conclusions

Summary

- Streaming useful for finding approximate solutions with reasonable amount of time & limited resources

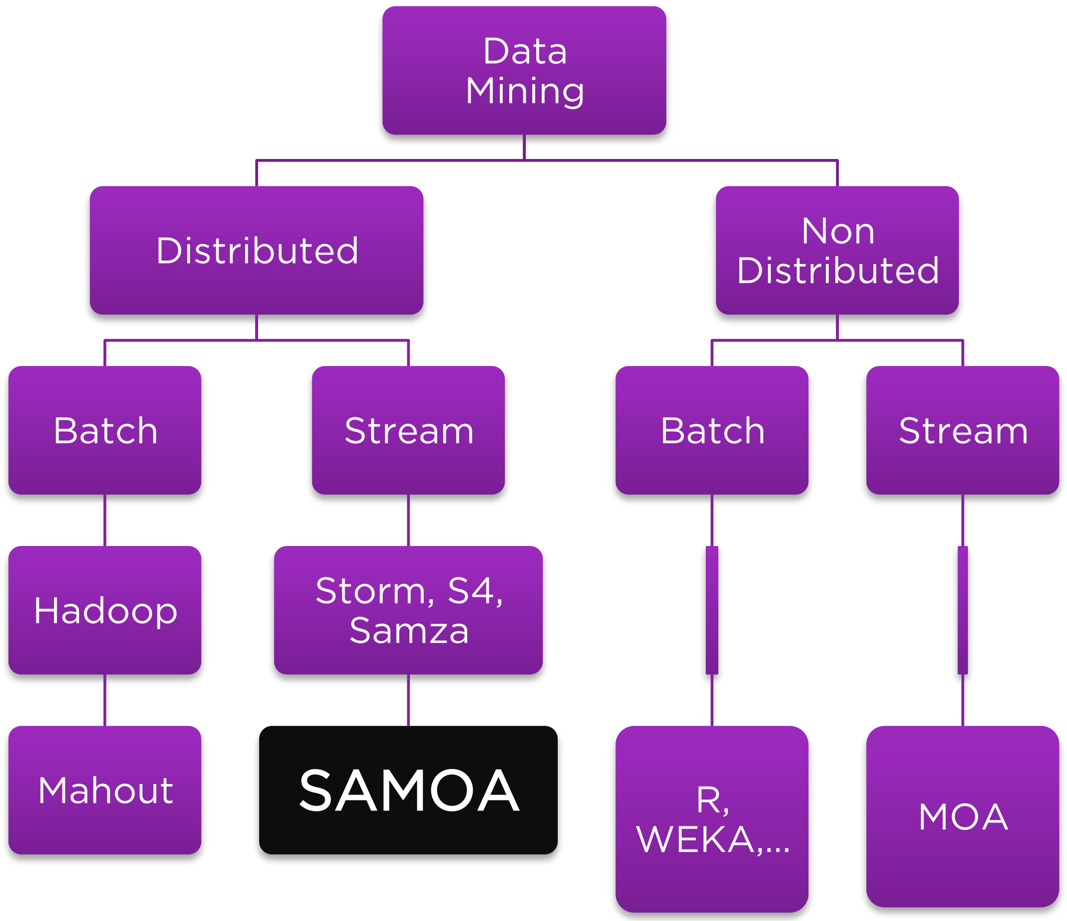
- Algorithms for classification, regression, clustering, frequent itemset mining

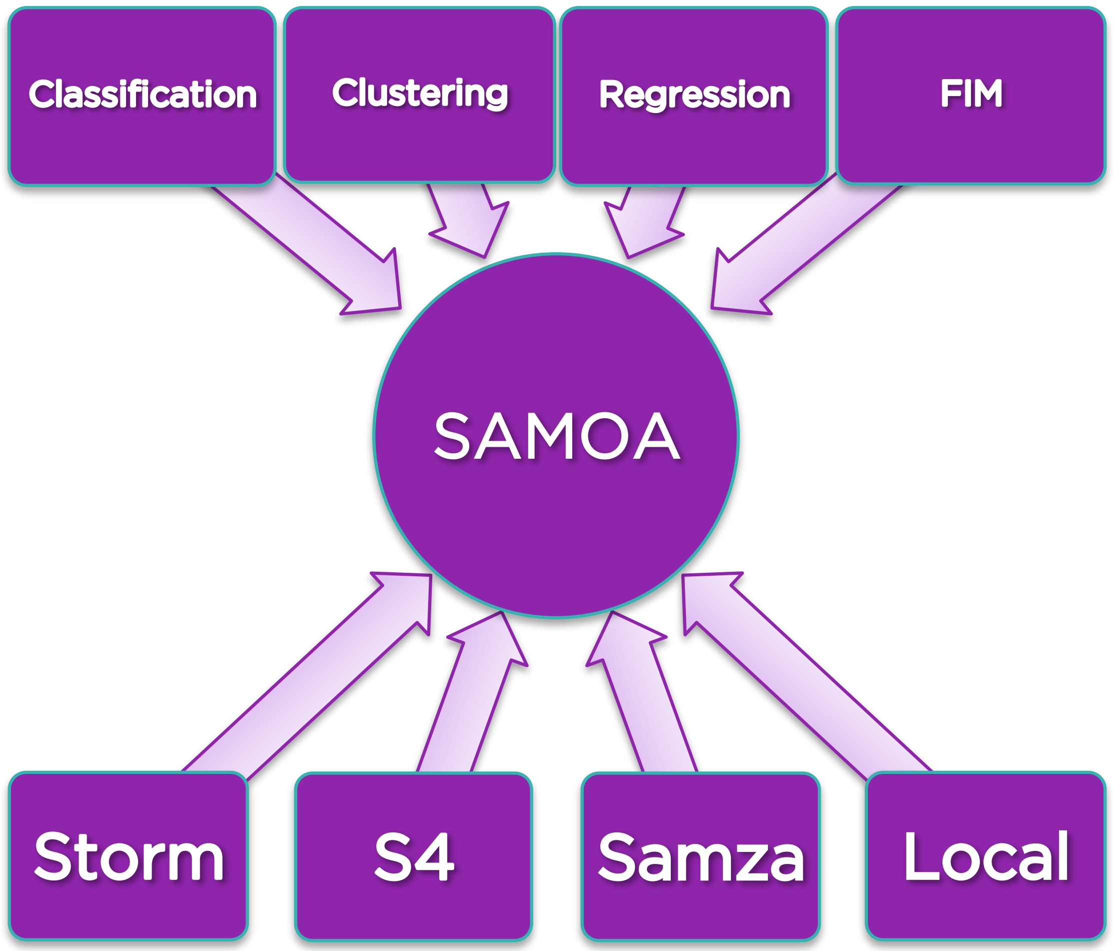
- Single machine for small streams

- Distributed systems for very large streams http://samoa-project.net

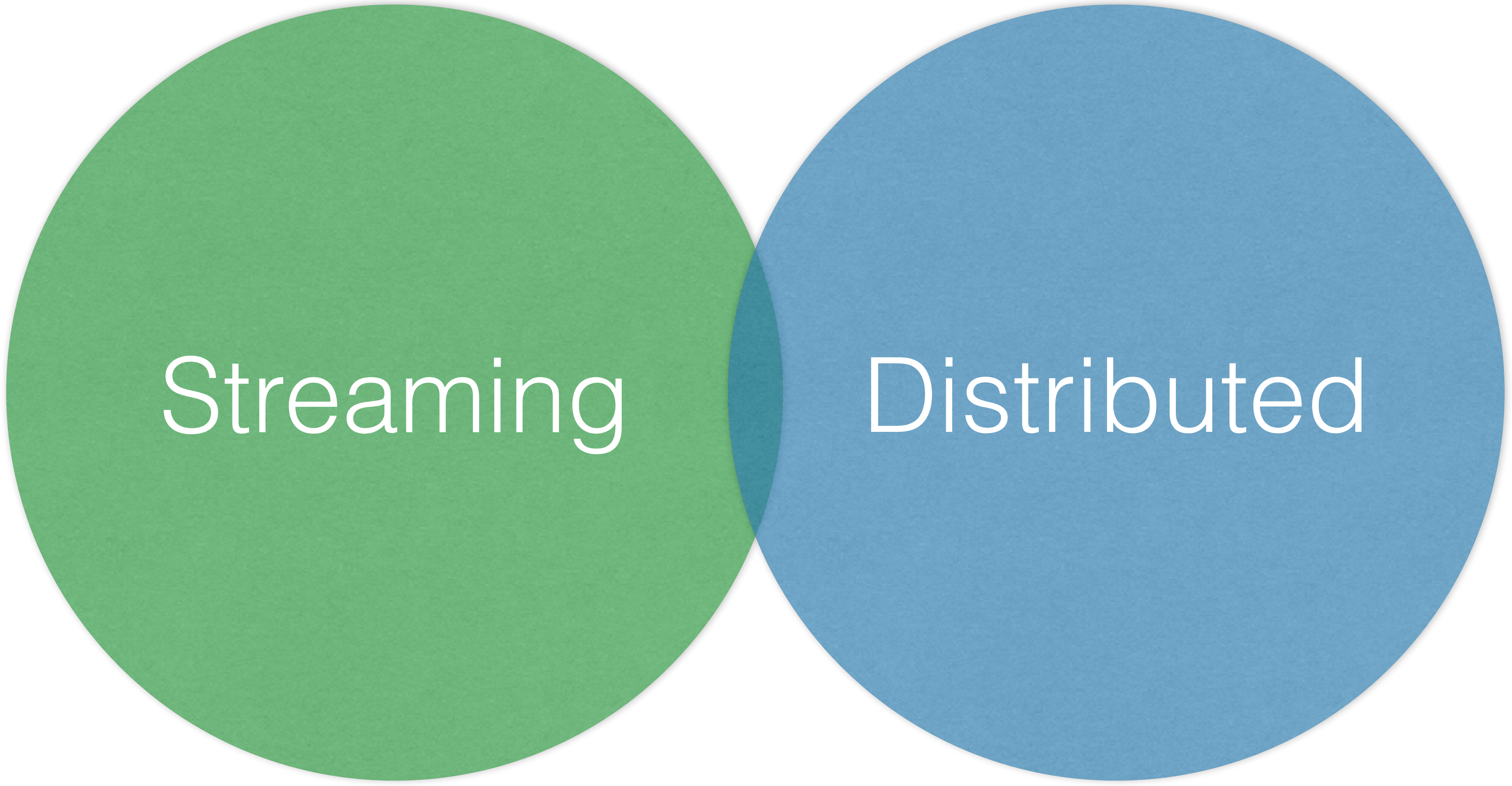
SAMOA

G. De Francisci Morales, A. Bifet: “SAMOA: Scalable Advanced Massive Online Analysis”. JMLR (2014)

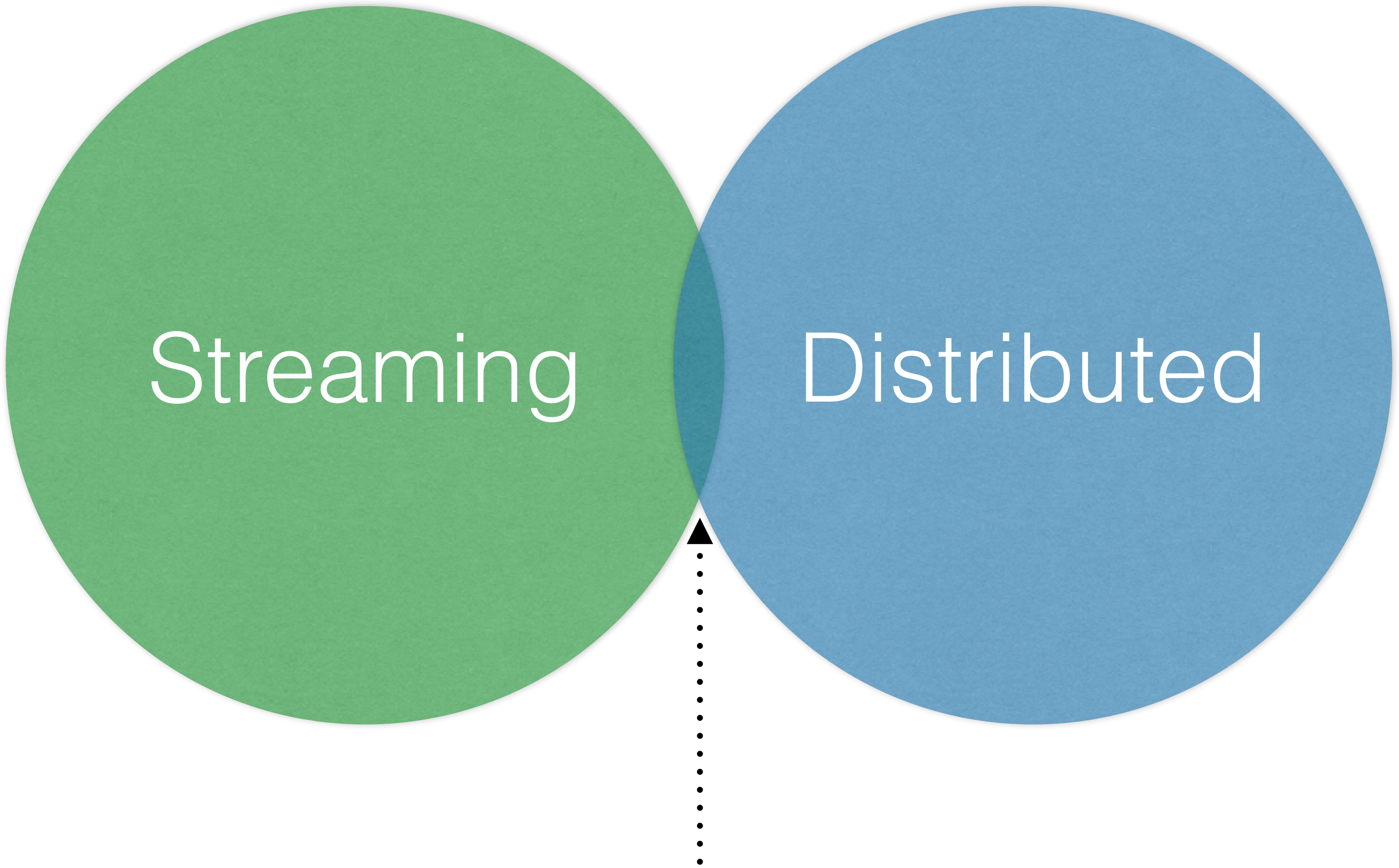




Vision

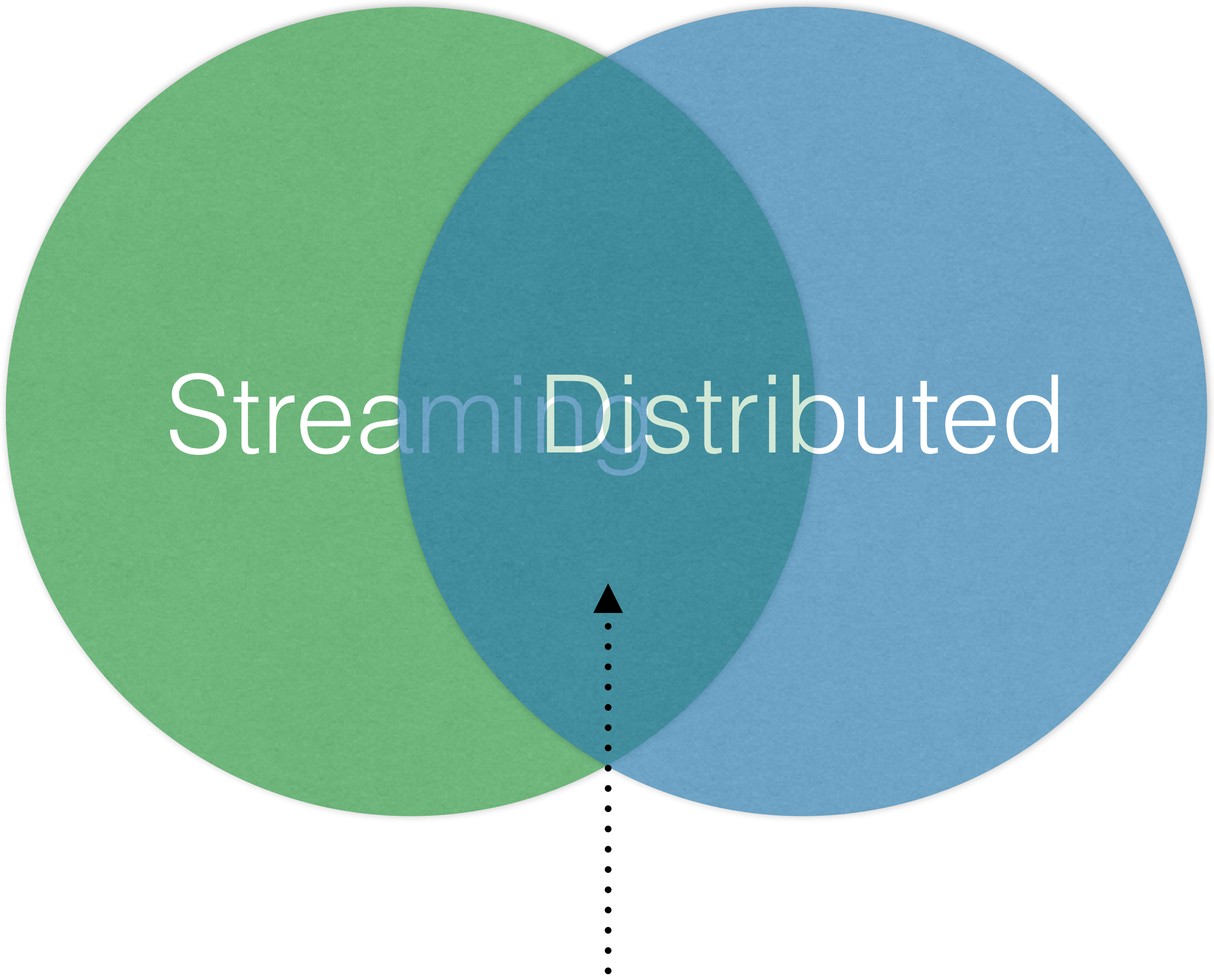


Vision



Big Data Stream Mining

Vision



Big Data Stream Mining

Open Challenges

- Structured output
- Multi-target learning
- Millions of classes
- Representation learning
- Ease of use

References

- IDC’s Digital Universe Study. EMC (2011)
- P. Domingos, G. Hulten: “Mining high-speed data streams”. KDD ’00
- J Gama, P. Medas, G. Castillo, P. Rodrigues: “Learning with drift detection”. SBIA’04
- G. Hulten, L. Spencer, P. Domingos: “Mining Time-Changing Data Streams”. KDD ‘01
- J. Gama, R. Fernandes, R. Rocha: “Decision trees for mining data streams”. IDA (2006)
- A. Bifet, R. Gavaldà: “Adaptive Parameter-free Learning from Evolving Data Streams”. IDA (2009)
- A. Bifet, R. Gavaldà: “Learning from Time-Changing Data with Adaptive Windowing”. SDM ’07
- E. Almeida, C. Ferreira, J. Gama. "Adaptive Model Rules from Data Streams”. ECML-PKDD ‘13
- H. Kremer, P. Kranen, T. Jansen, T. Seidl, A. Bifet, G. Holmes, B. Pfahringer: “An effective evaluation measure for clustering on evolving data streams”. KDD ’11
- T. Zhang, R. Ramakrishnan, M. Livny: “BIRCH: An Efficient Data Clustering Method for Very Large Databases”. SIGMOD ’96
- C. C. Aggarwal, J. Han, J. Wang, P. S. Yu: “A Framework for Clustering Evolving Data Streams”. VLDB ‘03
- M. Ester, H. Kriegel, J. Sander, X. Xu: “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”. KDD ‘96

- F. Cao, M. Ester, W. Qian, A. Zhou: “Density-Based Clustering over an Evolving Data Stream with Noise”. SDM ‘06
- G. S. Manku, R. Motwani: “Approximate frequency counts over data streams”. VLDB '02
- Y. Chi , H. Wang, P. Yu , R. Muntz: “Moment: Maintaining Closed Frequent Itemsets over a Stream Sliding Window”. ICDM ’04
- C. Giannella, J. Han, J. Pei, X. Yan, P. S. Yu: “Mining frequent patterns in data streams at multiple time granularities”. NGDM (2003)
- M. Stonebraker U. Çetintemel: “‘One Size Fits All’: An Idea Whose Time Has Come and Gone”. ICDE ’05
- A. Agarwal, O. Chapelle, M. Dudík, J. Langford: “A Reliable Effective Terascale Linear Learning System”. JMLR (2014)
- Y. Ben-Haim, E. Tom-Tov: “A Streaming Parallel Decision Tree Algorithm”. JMLR (2010)
- A. T. Vu, G. De Francisci Morales, J. Gama, A. Bifet: “Distributed Adaptive Model Rules for Mining Big Data Streams”. BigData ’14
- G. De Francisci Morales, A. Bifet: “SAMOA: Scalable Advanced Massive Online Analysis”. JMLR (2014)
- J. Gama: “Knowledge Discovery from Data Streams”. Chapman and Hall (2010)
- J. Gama: “Data Stream Mining: the Bounded Rationality”. Informatica 37(1): 21-25 (2013)

Contacts

- https://sites.google.com/site/bigdatastreamminingtutorial
- **Gianmarco De Francisci Morales**
  gdfm@yahoo-inc.com @gdfm7
- **João Gama**
  jgama@fep.up.pt @JoaoMPGama
- **Albert Bifet**
  abifet@waikato.ac.nz @abifet
- **Wei Fan**
  david.fanwei@huawei.com @fanwei