大规模机器学习算法GBDT及应用

王志伟（冰逸）

个人

- 王志伟（冰逸）
  - 2008-2011： 北航计算机硕士
  - 2011-2012： 网易有道 搜索组
  - 2012-2014： 百度 网页搜索 LTR组
  - 2014-至今： 阿里 推荐&投放算法小组

大纲

- GBDT算法介绍
- GBDT实现原理
- GBDT并行实现
- 应用

GBDT算法介绍

- 一般的有监督机器学习问题
- 训练数据
  - $X=\{x_1, x_2 \ldots, x_n\}$,n个样本
  - $Y=\{y_1, y_2 \ldots, y_n\}$
- 损失函数(loss function)
  - $L(F(X), Y)$
- 目标,寻找一个F
  - $F^* = argmin_F \mathrm{L}(\mathrm{Y}, \mathrm{F}(\mathrm{x}))$

损失函数

- 常见的回归问题
  - Squared error

$$L(Y, F(X)) = \sum_{i=1}^{n} (F(x_i) - y_i)^2$$

  - absolute error

$$L\big(Y, F(X)\big) = \sum_{i=1}^{n} |F(x_i) - y_i|$$

损失函数

- 分类问题
  - SVM hinge loss

$$L\left(Y,F(X)\right)=\sum_{i=1}^{n}\max(0,1-y_i*F(x_i))$$

  - Logistic regression loss

$$L\left(Y,F(X)\right)=\sum_{i=1}^{n}\log(1+\exp(-2*y_i*F(x_i))$$

损失函数

- 排序问题
  - Gbrank

$$L\big(Y, F(X)\big) = \sum_{i,j\varepsilon q \ and \ y_i > y_j} \max(0, -F(x_i) + F(x_j) + \tau)^2$$

  - LambdaMart

$$L\big(Y, F(X)\big) = \sum_{i,j\varepsilon q, and \ y_i > y_j} -log \frac{1}{1 + \exp(-F(x_i) + F(x_j))}$$

最优解F^*

- 对于参数化的模型F

$F(X;P)$

$$F^* = argmin_F L(y, F(X))$$

$$= argmin_P \; L(Y, F(X;P))$$

示例

$$P = \{w^T, b\}, F(X; P) = w^T * X + b$$

$$L\big(Y, F(X)\big) = \sum_{i=1}^{n} (y_i - F(x_i))^2$$

$$F^* = argmin_F L(Y, F(X))$$

$$= argmin_P L(Y, F(X; P))$$

$$= argmin_{w,b} L(Y, W^T * X + b)$$

$$= argmin_{w,b} \sum_{i=1}^{n} (y_i - w^T * x_i - b)^2$$

Gradient descent

参数P的求解

假设有一个初始解P_{m-1},如何寻找一个更优解 P_m

$$p_m = p_{m-1} + \rho * \Delta p$$

Gradient descent

$$L\big(Y, F(X; P)\big) = \varphi(P)$$

$$L(Y, F(X; Pm))$$

$$= \varphi(p_m) = \varphi(p_{m-1} + \rho * \Delta p)$$

$$\approx \varphi(p_{m-1}) + \frac{\partial \varphi(P)}{\partial P}|_{P=P_{m-1}} * \rho * \Delta P$$

$$\Delta P = -\frac{\partial \varphi(P)}{\partial P}|_{P=P_{m-1}}$$

$$\varphi(p_m) < \varphi(p_{m-1})$$

$$P^* = \sum_{i=1}^{M} \rho_i * n g_m$$

Gradient boost

- 最优解F*

$$F^* = \sum_{i=1}^{M} f_i(X)$$

如果有一个初始的$F_{m-1}(X)$,如何找到一个更优解$F_m(X)$

$$F_m(X) = F_{m-1}(X) + \rho * f(x)$$

Gradient boosting

$$L\big(Y, F(X)\big) = \varphi\big(F(X)\big)$$

$$L\big(Y, F_m(X)\big) = \varphi\big(F_{m-1}(X) + \rho f(X)\big)$$

$$\approx \varphi\big(\ F_{m-1}(X)\big) + \frac{\partial\varphi\big(F(X)\big)}{\partial F(X)}|_{F(X)=F_{m-1}(X)}$$

$$* \rho f(x)$$

$$f(x) = -\frac{\partial\varphi\big(F(X)\big)}{\partial F(X)}|_{F(X)=F_{m-1}(X)}$$

Gradient boosting

$$f(x) = -\frac{\partial \varphi\big(F(X)\big)}{\partial F(X)}\Big|_{F(X)=F_{m-1}(X)}$$

$$g_i = -\frac{\partial \varphi\big(F(X)\big)}{\partial F(x_i)}\Big|_{F(X)=F_{m-1}(X)}$$

$$= -\frac{\partial\big(F(x_1), F(x_2), \ldots F(x_n)\big)}{\partial F(x_i)}\Big|_{F(X)=F_{m-1}(X)}$$

$$L\big(Y, F(X)\big) = \sum_{j=1}^{n} \log(1 + \exp(-2 * y_j * F\big(x_j\big))$$

$$\frac{\partial L(Y, F(X))}{\partial F(X_i)} = \frac{\partial \sum_{j=1}^{n} \log(1 + \exp(-2 * y_j * F\big(x_j\big))}{\partial F(x_j)}$$

$$= \frac{\partial \log(1 + \exp(-2 * y_i * F(x_i))}{\partial F(x_i)}$$

$$= \frac{\exp(-2 * y_i * F(x_i)) * (-2 * y_i)}{1 + \exp(-2 * y_i * F(x_i))}$$

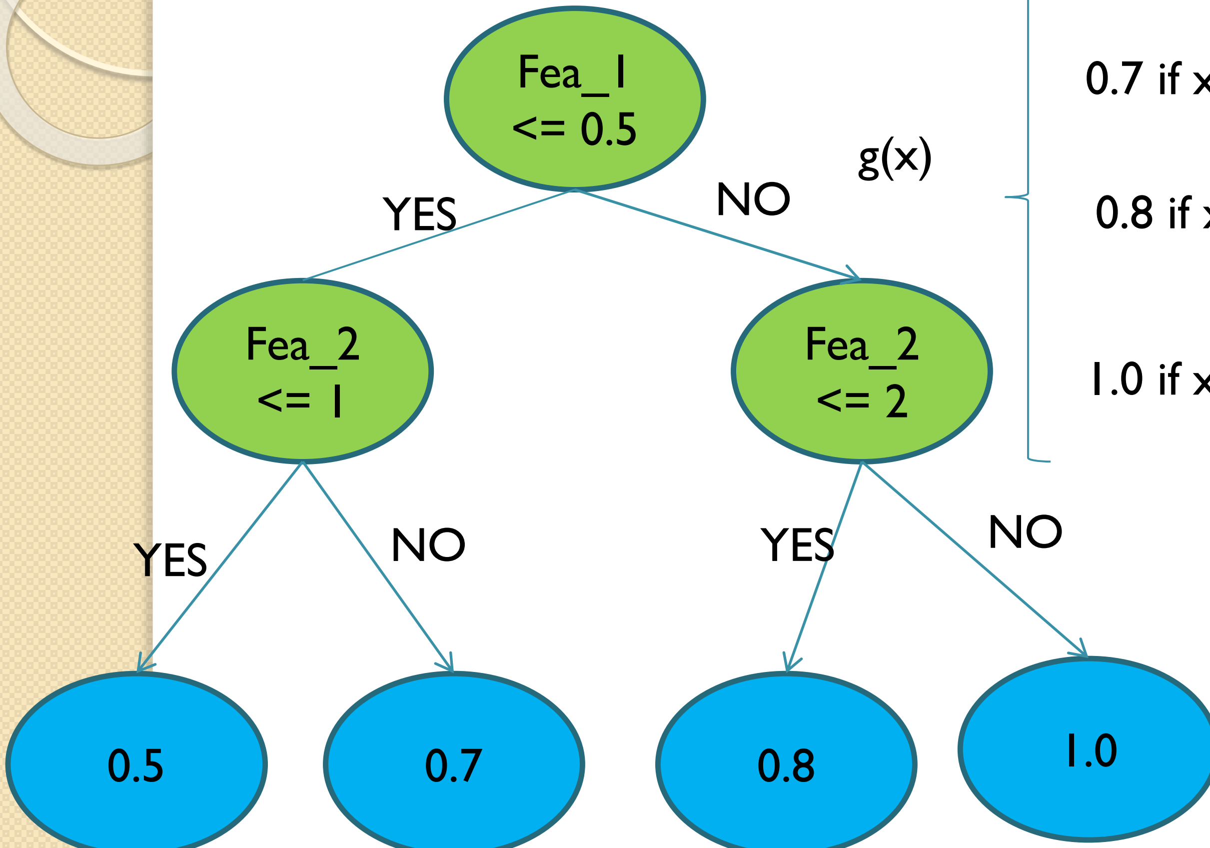
$$f(x_i) = -\frac{\partial L\big(Y, F(X)\big)}{\partial F(x_i)} \Big|_{F(X) = F_{m-1}(X)}$$

$$= -\frac{\exp\big(-2 * y_i * F_{m-1}(x_i)\big) * (-2 * y_i)}{1 + \exp(-2 * y_i * F_{m-1}(x_i))}$$

Gradient boosting

$$f(x) = \begin{cases} g_1 & \text{if } X = x_1 \\ g_2 & \text{if } X = x_2 \\ g_3 & \text{if } X = x_3 \\ \vdots & \\ g_n & \text{if } X = x_n \end{cases}$$

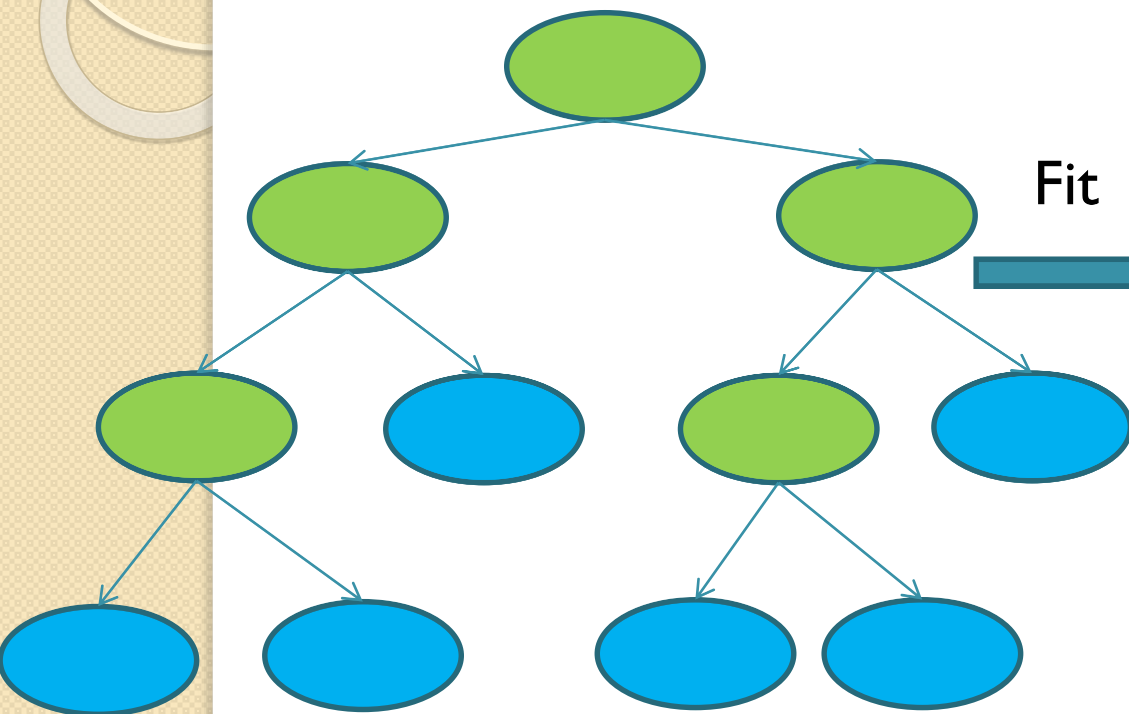
Decision Tree



$$g(x) = \begin{cases} 0.5 & \text{if } x[1] <= 0.5 \ \& \ x[2] <= 1 \\ 0.7 & \text{if } x[1] <= 0.5 \ \& \ x[2] > 1 \\ 0.8 & \text{if } x[1] > 0.5 \ \& \ x[2] <= 2 \\ 1.0 & \text{if } x[1] > 0.5 \ \& \ x[2] > 2 \end{cases}$$

L个叶子的Decision Tree记 $\{R_l\}^L$,第l的叶子节点的值记 r_l

Decision Tree



Fit

$$f(x) \begin{cases} g_1 & \text{if } X = x_1 \\ g_2 & \text{if } X = x_2 \\ g_3 & \text{if } X = x_3 \\ \vdots & \\ g_n & \text{if } X = x_n \end{cases}$$

Gradient boosting Decision Tree

输入{X,Y}, 损失函数L(Y,F(X))

初始模型 $F_0(X) = 0$

For m = 1 to M

Step1:计算在每一个样本点的负梯度gi

$$g_i = -\frac{\partial L(Y, F(X)}{F(X_i)}|_{F(x)=F_{m-1}(X)}$$

Step2:建立一棵L叶子节点的决策树$\{R_l\}_{l=1..L}$,
拟合$\{x_i, g_i\}_{i=1..n}$,记为$g_m(X)$

Step3: $F_m(X) = F_{m-1}(X) + \rho * g_m(X)$

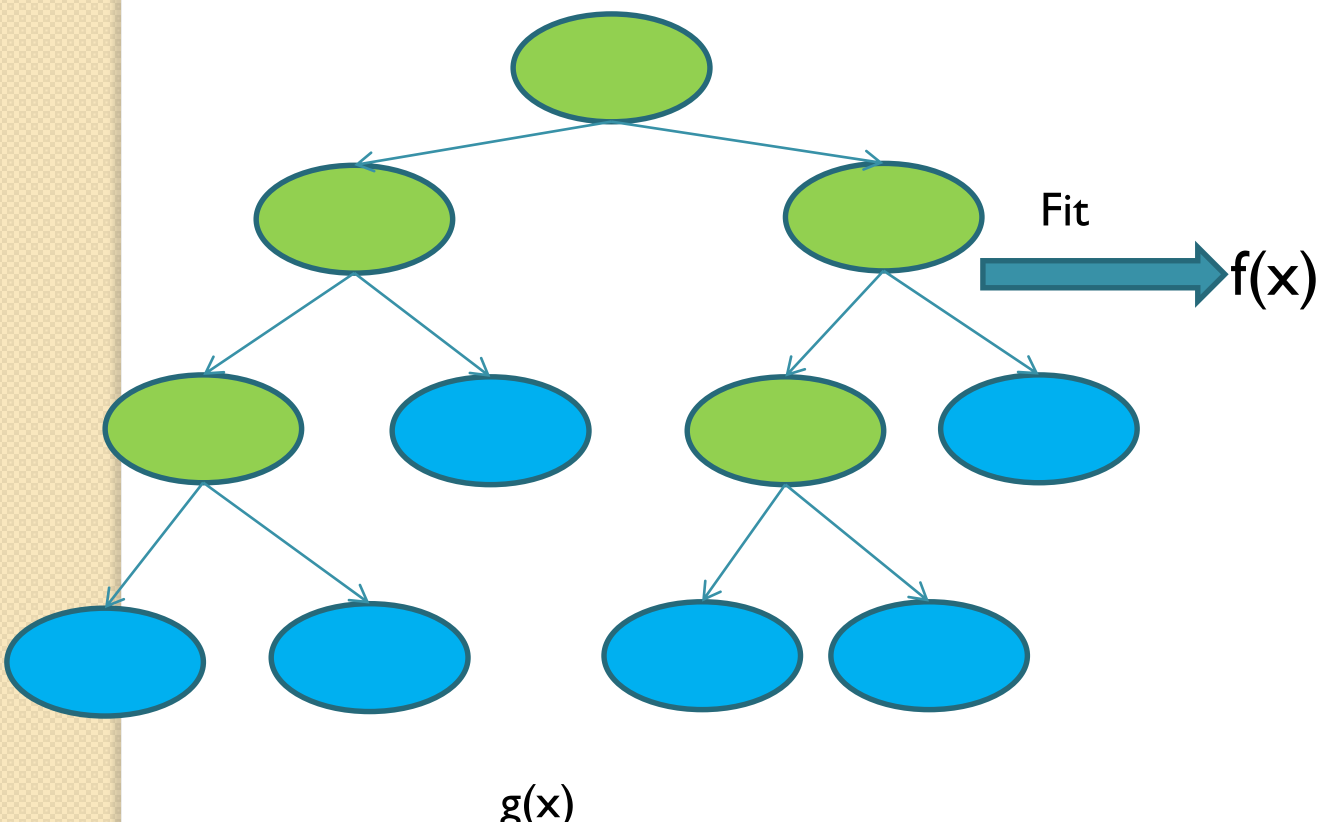
$F(X) = F_M(X)$

大纲

- GBDT算法介绍
- GBDT实现原理
- GBDT并行实现
- 应用

GBDT实现原理

- 建立决策树拟合负梯度函数



$$
\begin{cases}
g_1 & \text{if } X=x_1 \\
g_2 & \text{if } X=x_2 \\
g_3 & \text{if } X=x_3 \\
\vdots & \\
g_n & \text{if } X=x_n
\end{cases}
$$

g(x)拟合f(x) 拟合指标

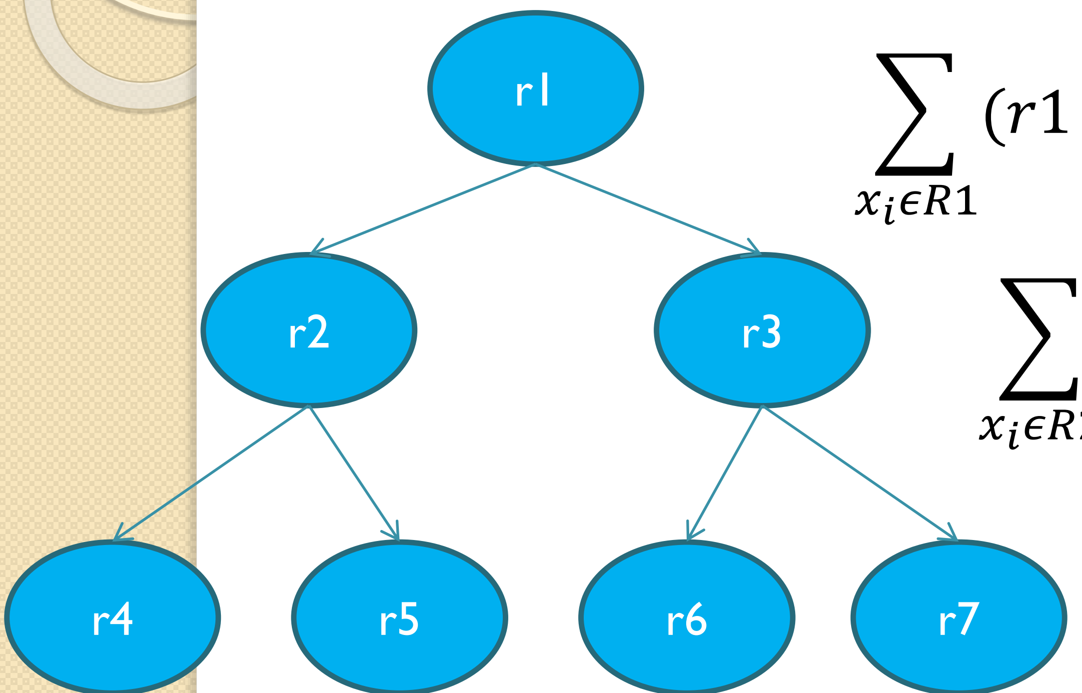
- g(x)与f(x)的diff
- absolute error

$$\sum_{i=1}^{n} |g(x_i) - f(x_i)|$$

- square error

$$\sum_{i=1}^{n} (g(x_i) - f(x_i))^2$$

Greedy algorithm



$$\sum_{x_i \epsilon R1} (r1 - g_i)^2$$

$$\sum_{x_i \epsilon R2} (r2 - g_i)^2 + \sum_{x_i \epsilon R3} (r3 - g_i)^2$$

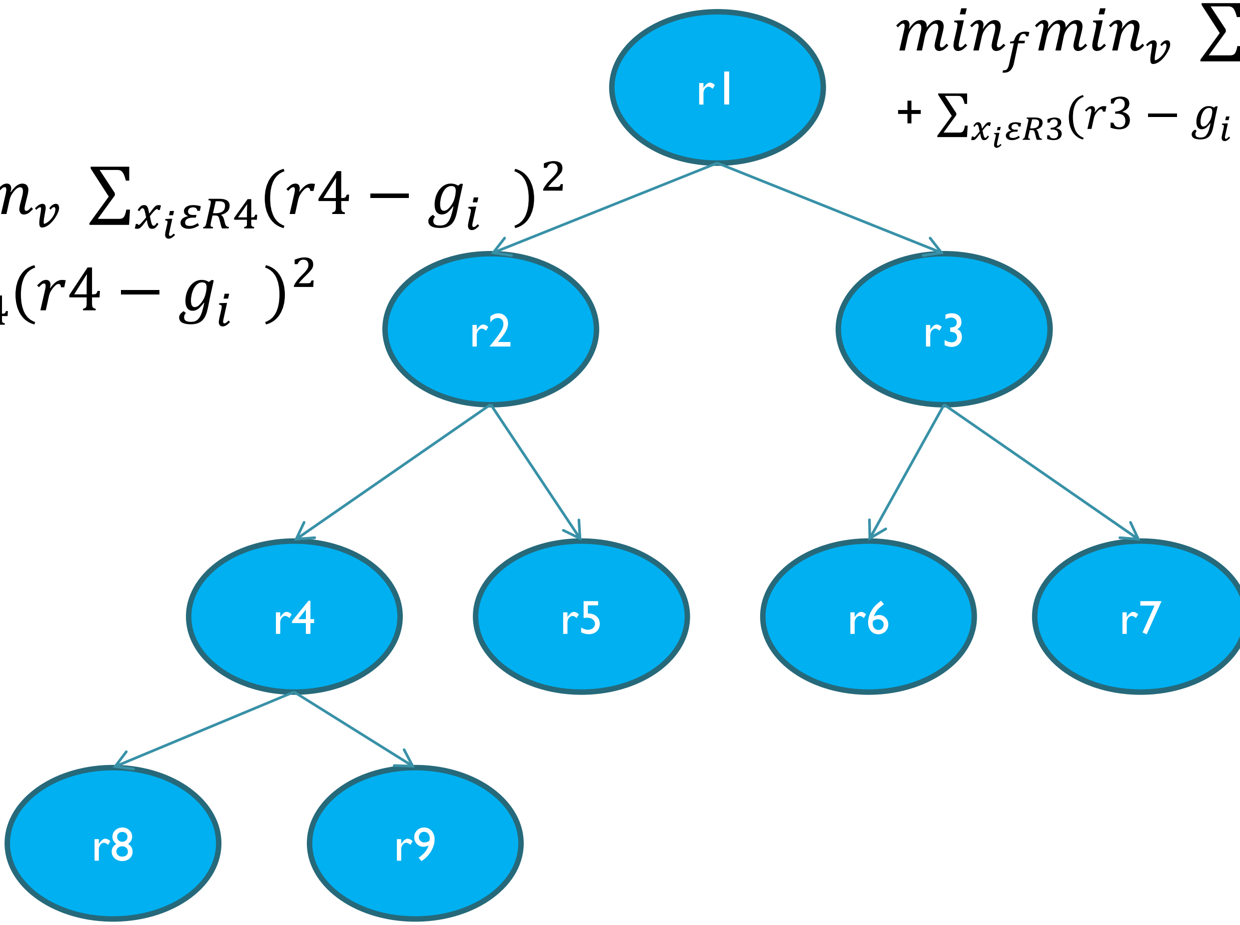
$$\sum_{x_i \epsilon R4} (r4 - g_i)^2 + \sum_{x_i \epsilon R5} (r5 - g_i)^2$$

$$+ \sum_{x_i \epsilon R6} (r6 - g_i)^2 + \sum_{x_i \epsilon R7} (r7 - g_i)^2$$

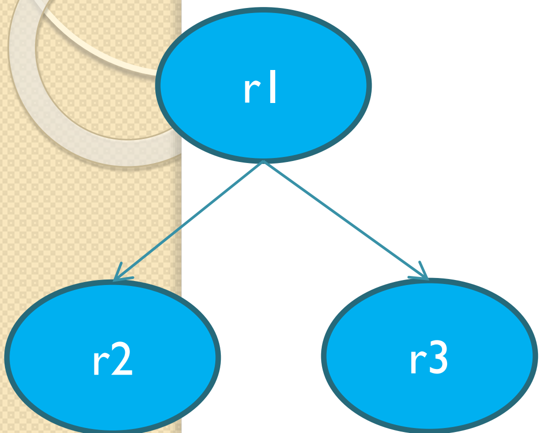
Greedy algorithm

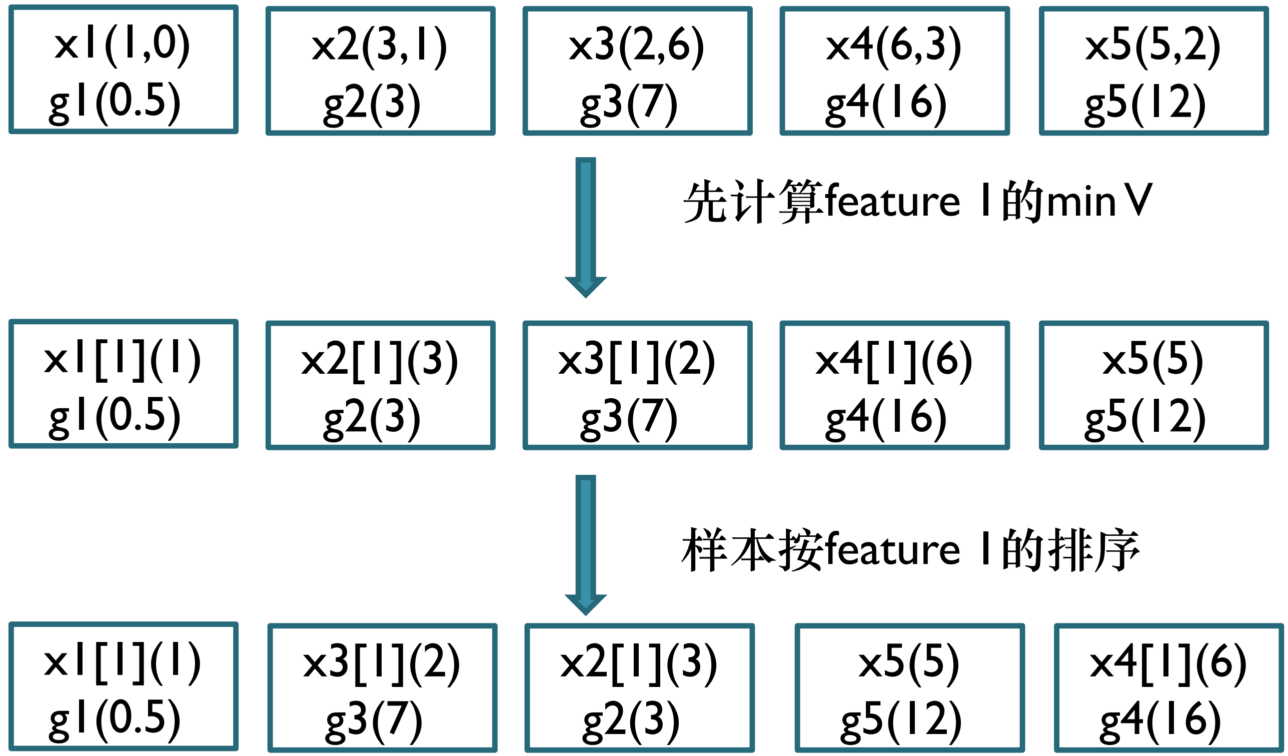
$$min_f min_v \sum_{x_i \varepsilon R2}(r2 - g_i\ )^2 + \sum_{x_i \varepsilon R3}(r3 - g_i\ )^2$$

$$min_f min_v \sum_{x_i \varepsilon R4}(r4 - g_i\ )^2 + \sum_{x_i \varepsilon R4}(r4 - g_i\ )^2$$

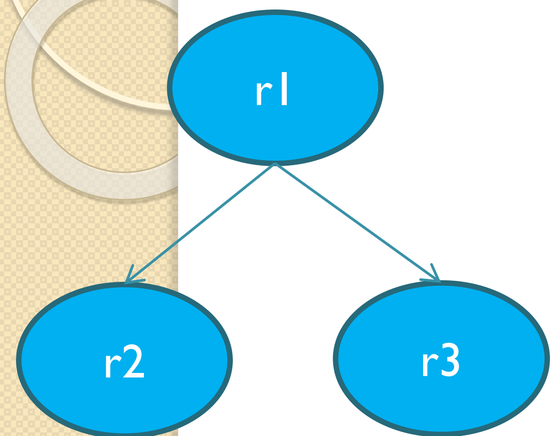


Min F min V


r1
r2
r3


x1(1,0) g1(0.5)
x2(3,1) g2(3)
x3(2,6) g3(7)
x4(6,3) g4(16)
x5(5,2) g5(12)
先计算feature 1的min V
x1[1](1) g1(0.5)
x2[1](3) g2(3)
x3[1](2) g3(7)
x4[1](6) g4(16)
x5(5) g5(12)
样本按feature 1的排序
x1[1](1) g1(0.5)
x3[1](2) g3(7)
x2[1](3) g2(3)
x5(5) g5(12)
x4[1](6) g4(16)

MIN v



V = 1.5

x1[1](1) g1(0.5)	x3[1](2) g3(7)	x2[1](3) g2(3)	x5(5) g5(12)	x4[1](6) g4(16)
X1(0.5)	X3(7)	X2(3)	X5(12)	X4(16)

$$\sum_{x_i \epsilon R2} (r2 - g_i\ )^2 + \sum_{x_i \epsilon R3} (r3 - g_i\ )^2$$

$$= (r2 - 0.5)^2 + (r3 - 7)^2 + (r3 - 3)^2 + (r3 - 12)^2 + (r3 - 16)^2$$

$$r2 = 0.5$$

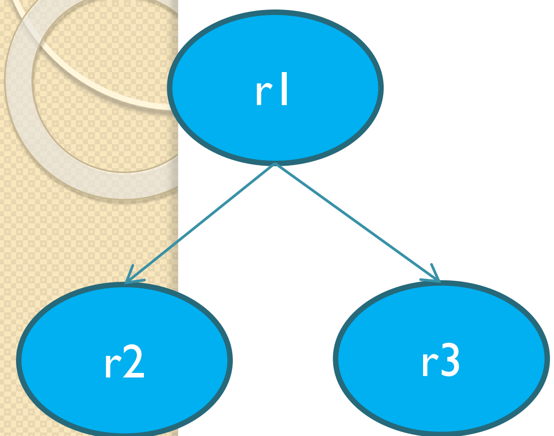
$$r3 = \frac{7 + 3 + 12 + 16}{4} = 9.5$$

$$error: 97$$

MIN v

V = 2.5

x1[1](1) g1(0.5) | x3[1](2) g3(7) | x2[1](3) g2(3) | x5(5) g5(12) | x4[1](6) g4(16)



x1(0.5)
X3(7)

x2(3)
X5(12)
X4(16)

$$\sum_{x_i \epsilon R2} (r2 - g_i)^2 + \sum_{x_i \epsilon R3} (r3 - g_i)^2$$

$$= (r2 - 0.5)^2 + (r2 - 7)^2 + (r3 - 3)^2 + (r3 - 12)^2 + (r3 - 16)^2$$

$$r2 = \frac{0.5 + 7}{2} = 3.75$$

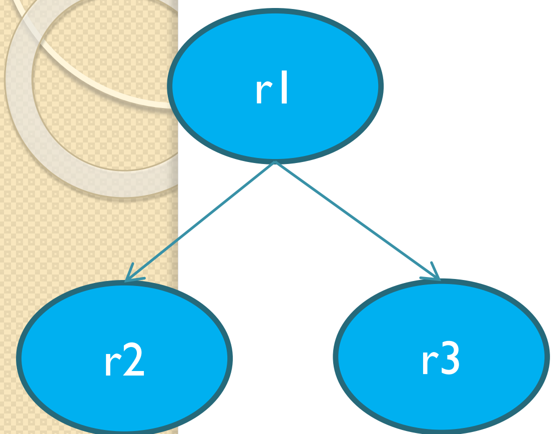
$$r3 = \frac{3 + 12 + 16}{3} = 10.33$$

$$error: 109.7917$$

MIN v

V = 4

x1[1](1) g1(0.5)	x3[1](2) g3(7)	x2[1](3) g2(3)	x5(5) g5(12)	x4[1](6) g4(16)



$$\sum_{x_i \epsilon R2} (r2 - g_i\ )^2 + \sum_{x_i \epsilon R3} (r3 - g_i\ )^2$$

x1(0.5)
x3(7)
x2(3)

X5(12)
X4(16)

$$= (r2 - 0.5)^2 + (r2 - 7)^2 + (r2 - 3)^2 + (r3 - 12)^2 + (r3 - 16)^2$$

$$r2 = \frac{0.5 + 7 + 3}{3} = 3.5$$

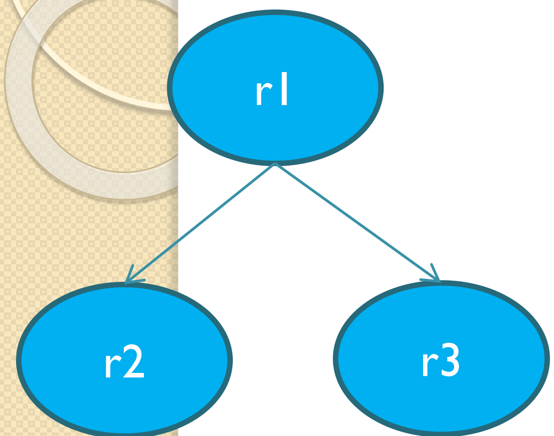
$$r3 = \frac{12 + 16}{2} = 14$$

$error$: 29.5

MIN v

V = 5.5

x1[1](1) g1(0.5)	x3[1](2) g3(7)	x2[1](3) g2(3)	x5(5) g5(12)	x4[1](6) g4(16)



x1(0.5)
X3(7)
X2(3)
X5(12)

X4(16)

$$\sum_{x_i \epsilon R2} (r2 - g_i)^2 + \sum_{x_i \epsilon R3} (r3 - g_i)^2$$

$$= (r2 - 0.5)^2 + (r2 - 7)^2 + (r2 - 3)^2 + (r2 - 12)^2 + (r3 - 16)^2$$

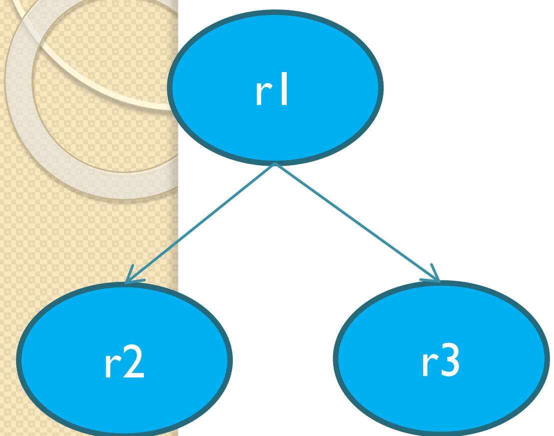
$$r2 = \frac{0.5 + 7 + 3 + 12}{4} = 5.62$$

$$r3 = 16$$

$$error: 75.68$$

MIN v

V = 5.5



x1[1](1) g1(0.5)	x3[1](2) g3(7)	x2[1](3) g2(3)	x5(5) g5(12)	x4[1](6) g4(16)

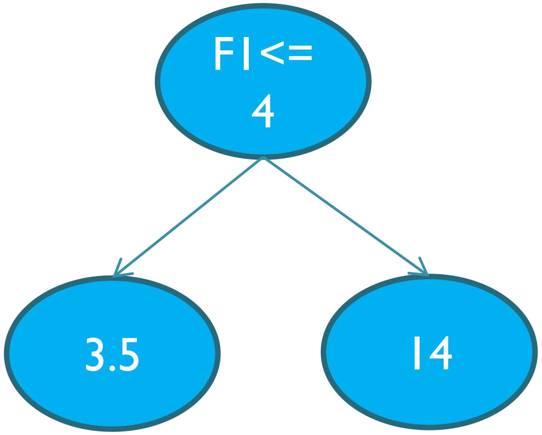
V=1.5 error :97

V=3.5 error :109.7917

V=4 error :29.5

V=5.5 error :75.68

Min V



Min V

- 计算复杂度
  - N个样本，一共有n个可选的分裂点
  - 对于一个分列V,求error n次运算
  - 复杂度 O(N^2)

Min v

$$min \sum_{x_i \epsilon R2} (r2 - g_i)^2 + \sum_{x_i \epsilon R3} (r3 - g_i)^2$$

$$= min \sum_{x_i \epsilon R2} (ave_{R2} - g_i)^2 + \sum_{x_i \epsilon R3} (ave_{R3} - g_i)^2$$

$$= min \sum_{x_i \epsilon R2} {g_i}^2 - count_{R2} * {ave_{R2}}^2 + \sum_{x_i \epsilon R3} {g_i}^2 - count_{R3} * {ave_{R3}}^2$$

$$= min \sum_{x_i \epsilon R1} {g_i}^2 - \frac{{sum_{R2}}^2}{count_{R2}} - \frac{{sum_{R3}}^2}{count_{R3}}$$

constant

$$= min - \frac{{sum_{R2}}^2}{count_{R2}} - \frac{{sum_{R3}}^2}{count_{R3}} \qquad = max \frac{{sum_{R2}}^2}{count_{R2}} + \frac{{sum_{R3}}^2}{count_{R3}}$$

MIN v

ALL_SUM=0.5+7+3+12+16=38.5
ALL_COUNT=5

x1[1](1) g1(0.5)	x3[1](2) g3(7)	x2[1](3) g2(3)	x5(5) g5(12)	x4[1](6) g4(16)
R2_SUM=0.5 R2_COUNT=1	R2_SUM=7.5 R2_COUNT=2	R2_SUM=10.5 R2_COUNT=3	R2_SUM=22.5 R2_COUNT=4	
R3_SUM=38 R3_COUNT=4	R3_SUM=31.5 R3_COUNT=3	R3_SUM=28.5 R3_COUNT=2	R3_SUM=16 R3_COUNT=1	
0.5^2/1+38^2/4=361.25	7.5^2/2+31.5^2/3=358.5	10.5^2/3+28.5^2/2=442.875	22.5^2/4+16^2/1=382.5625	

O(N)

V=4

Create Tree

输入$\{x_i, g_i\}_{i=1..n}$

Step1.将所有样本放入root结点（R1），标记为可分结点

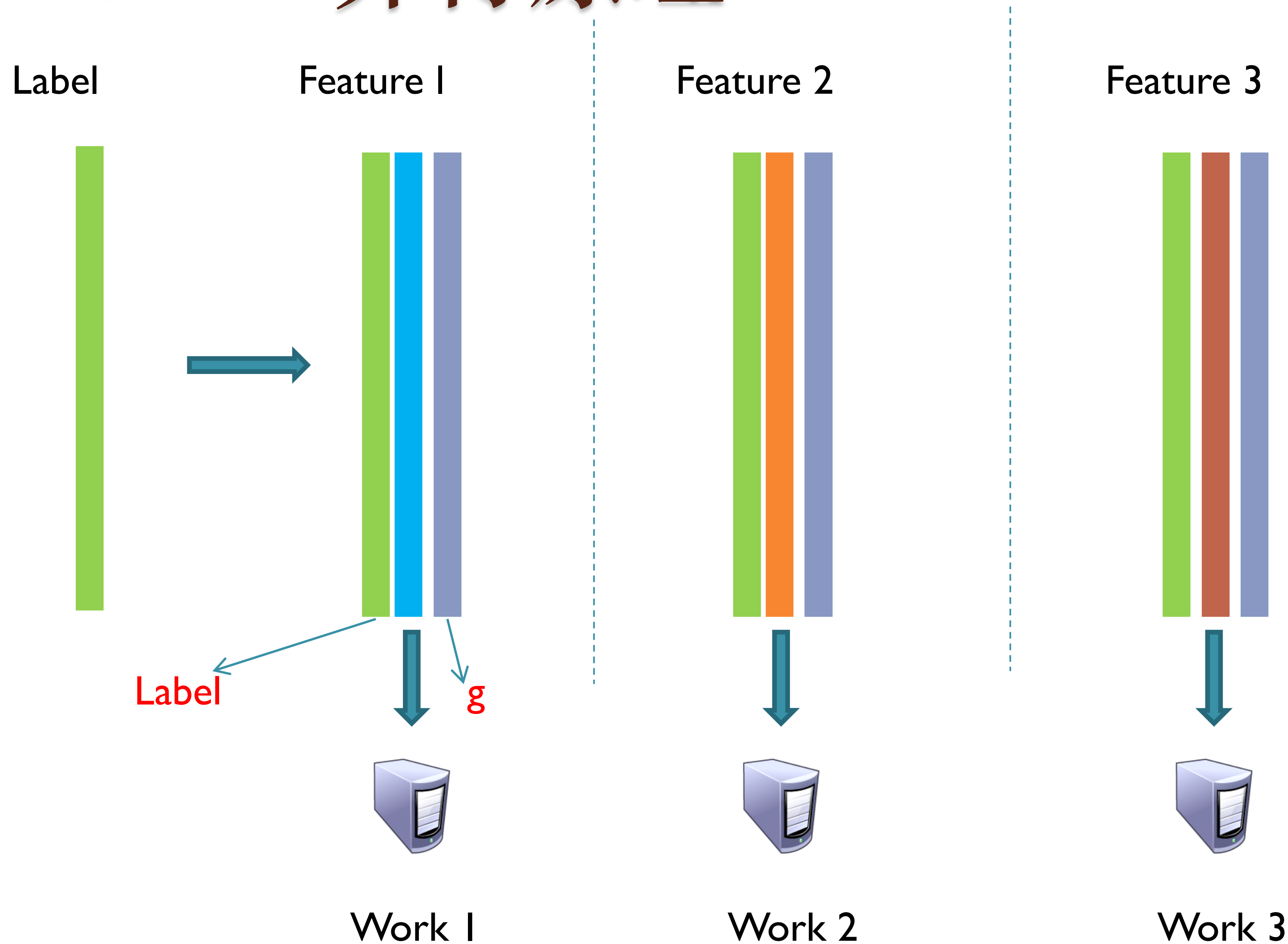
Step2.对于一个可分的结点，寻找一个最优的分裂点(fid,value),分裂结点，并将该结点标记为不可分，同时将新生成的子结点标记为可分结点。

Step3.重复step2，直至无可分结点或者叶子结点数达到域值终止。

大纲

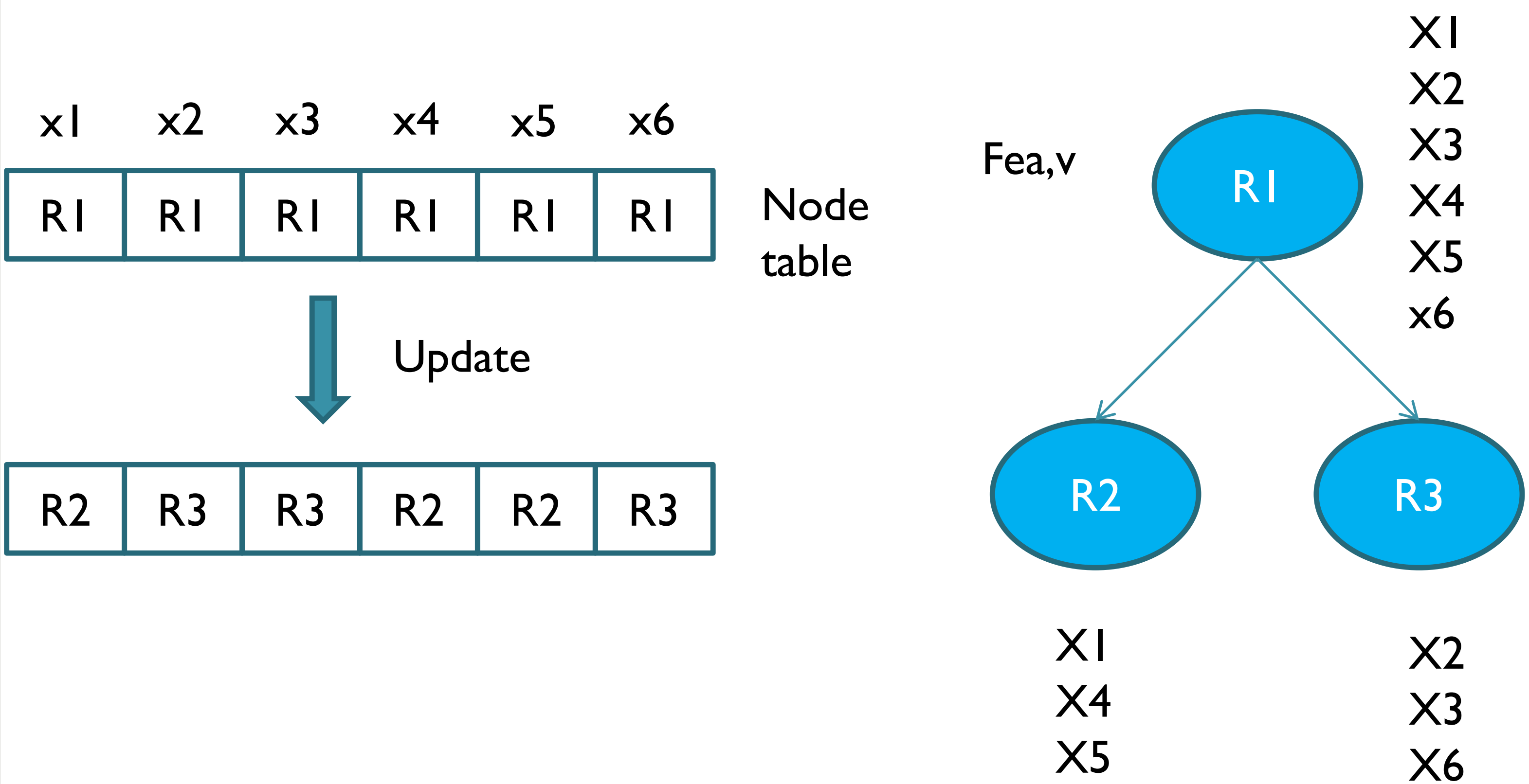
- GBDT算法介绍
- GBDT实现原理
- GBDT并行原理
- 应用

GBDT并行原理



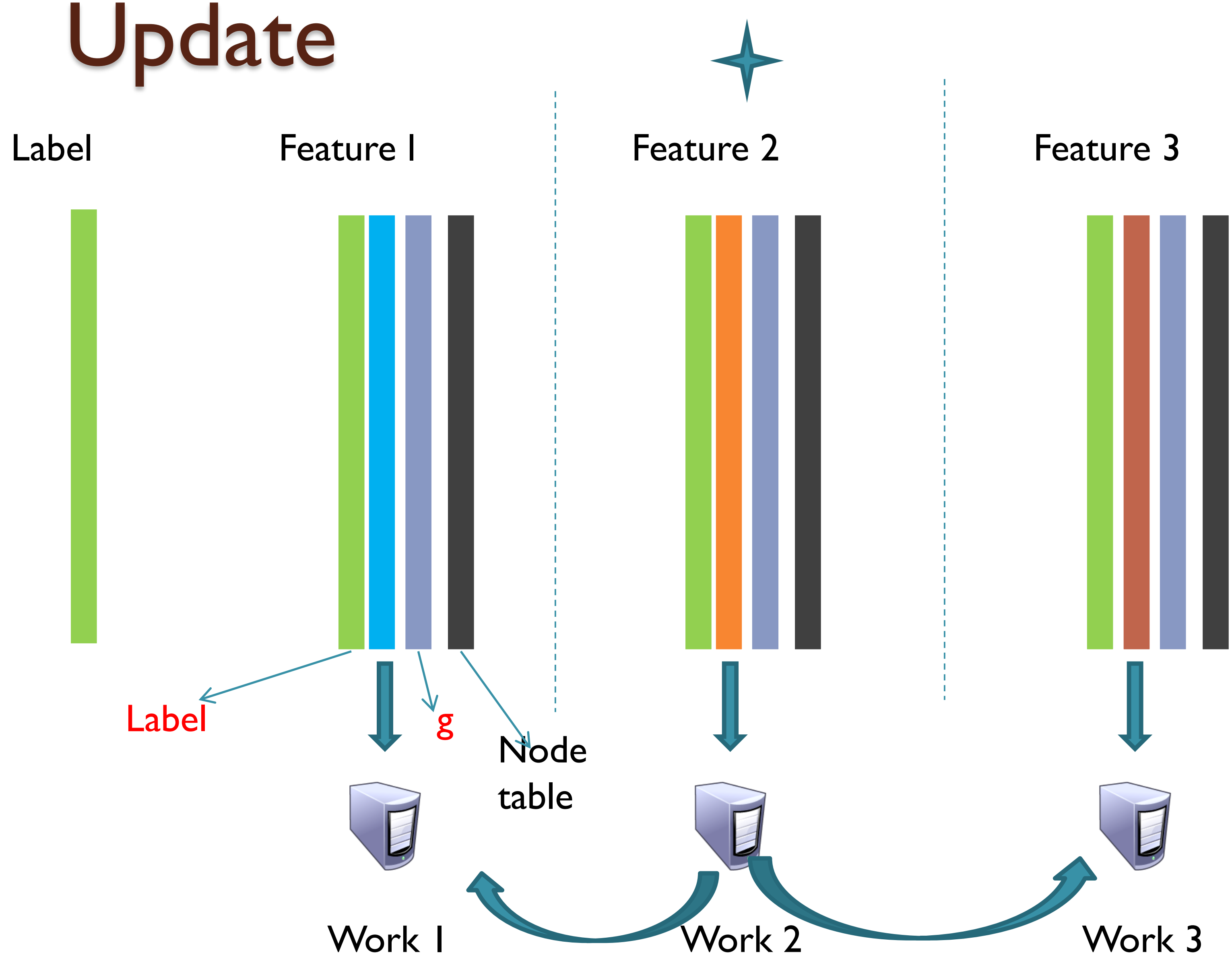
Label
Feature 1
Feature 2
Feature 3
Label
g
Work 1
Work 2
Work 3

Split

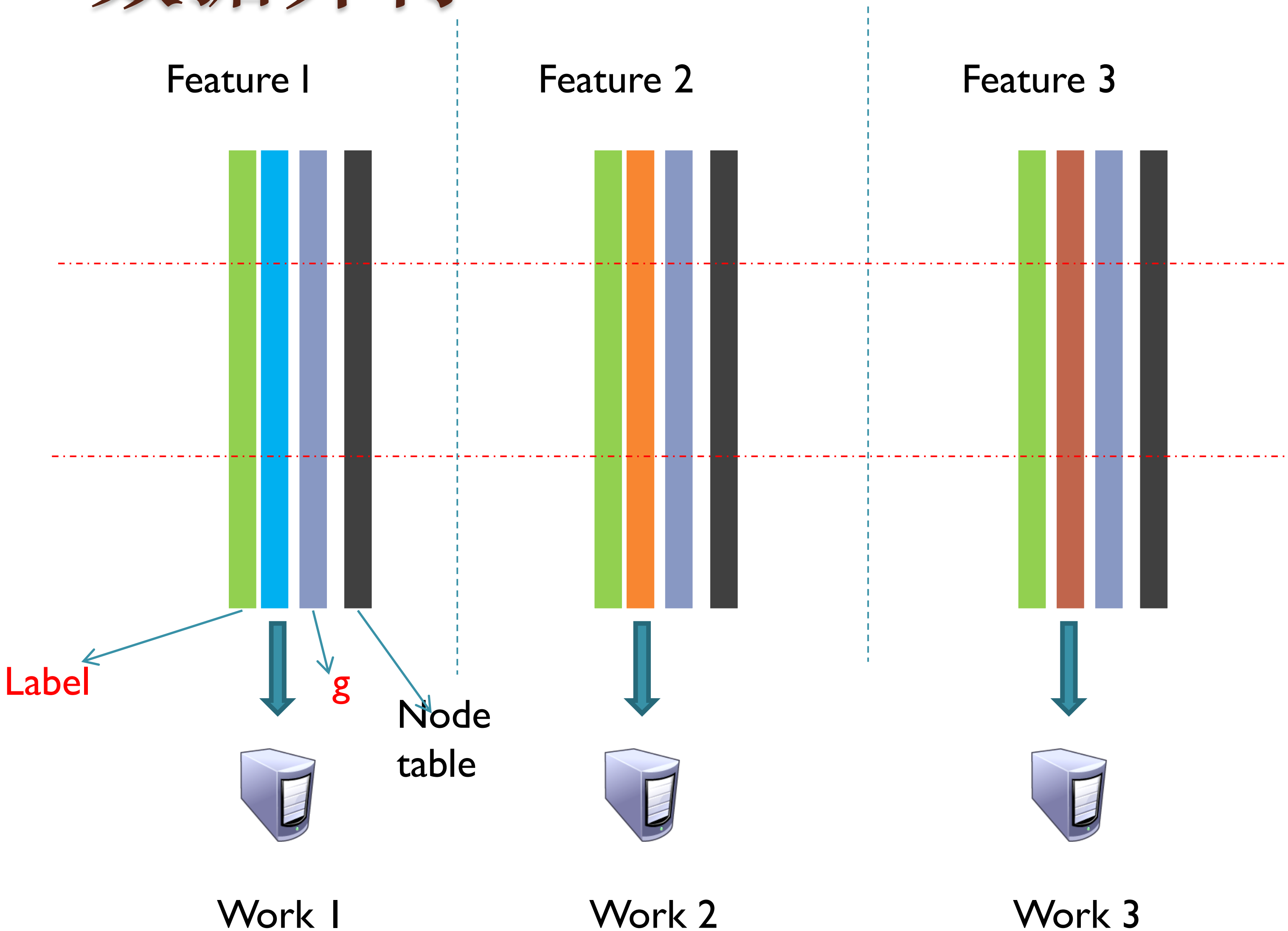


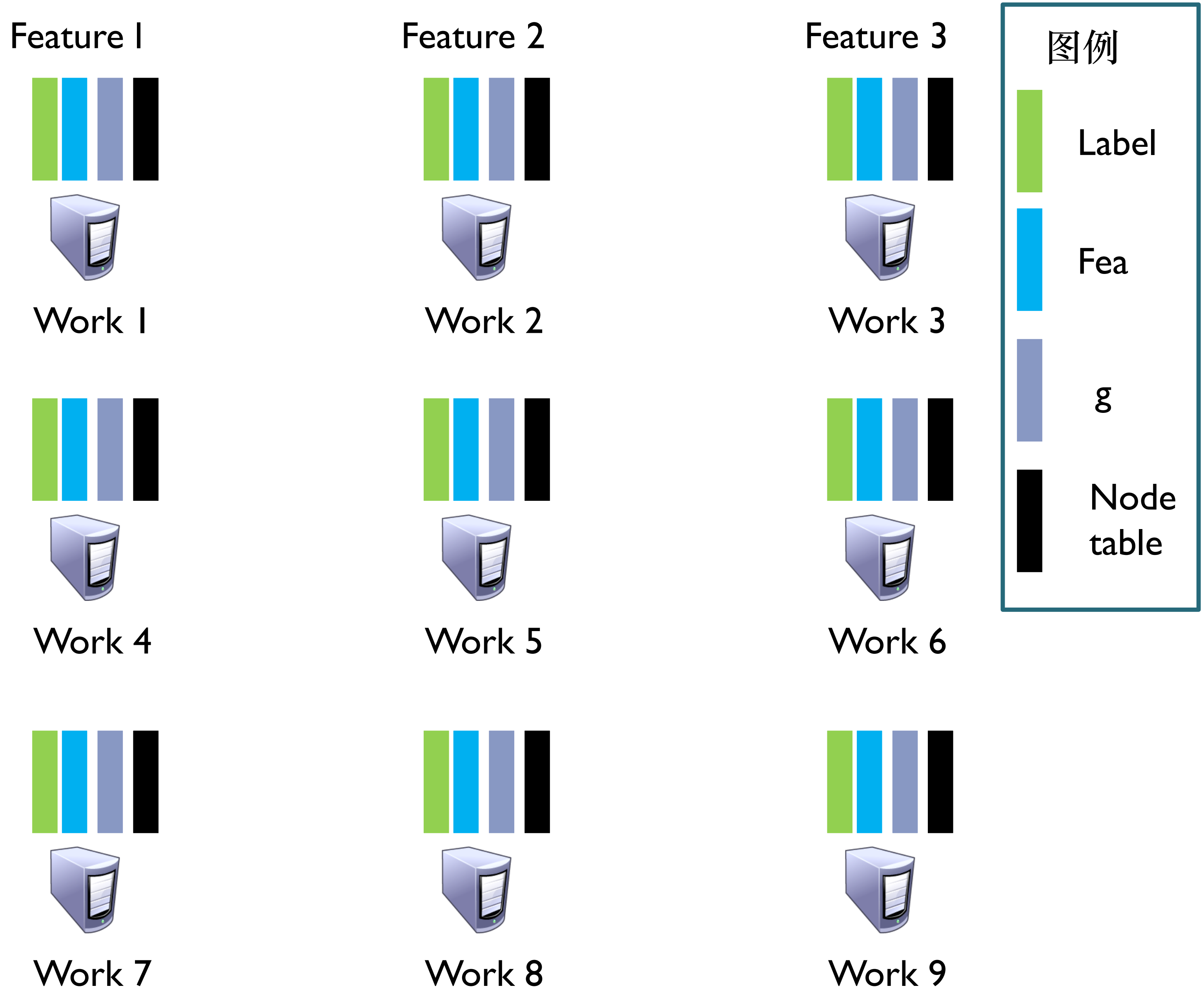
x1 x2 x3 x4 x5 x6
R1 R1 R1 R1 R1 R1
Node table
Update
R2 R3 R3 R2 R2 R3
Fea,v
R1
X1 X2 X3 X4 X5 x6
R2
R3
X1 X4 X5
X2 X3 X6

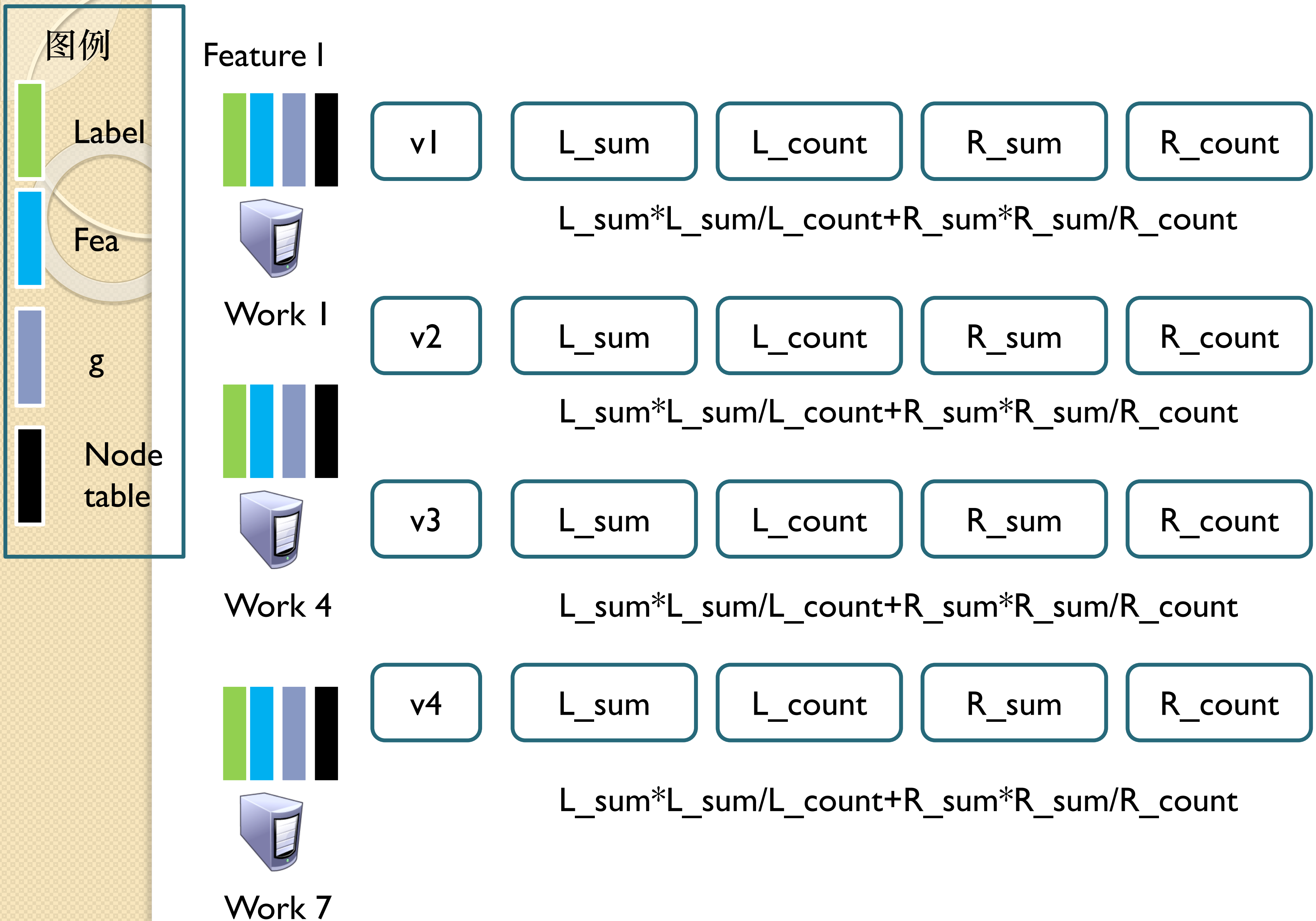
Update

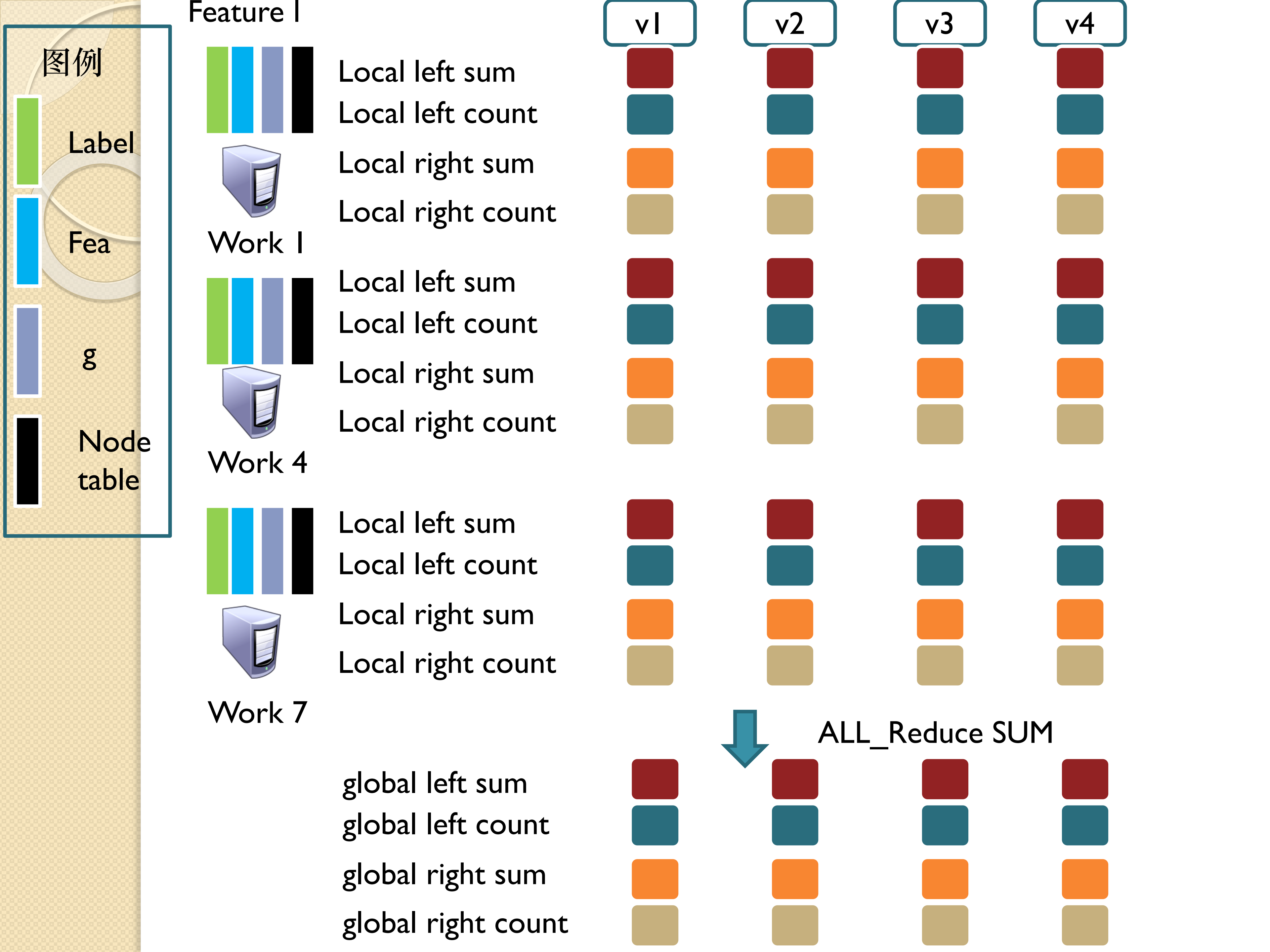

Label
Feature 1
Feature 2
Feature 3
Label
g
Node table
Work 1
Work 2
Work 3

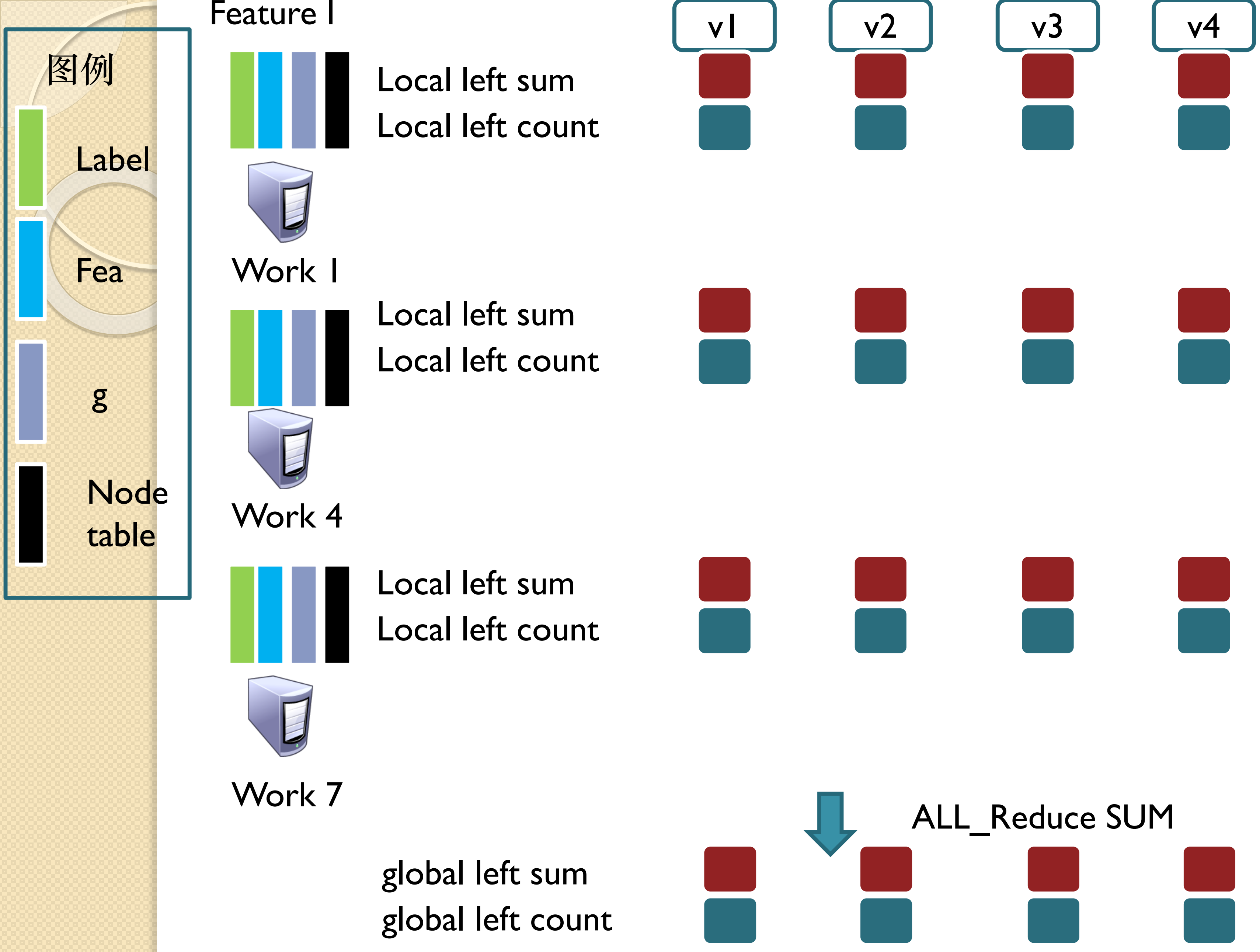
数据并行


Feature 1
Feature 2
Feature 3
Label
g
Node
table
Work 1
Work 2
Work 3


Feature 1
Feature 2
Feature 3
Work 1
Work 2
Work 3
Work 4
Work 5
Work 6
Work 7
Work 8
Work 9
图例
Label
Fea
g
Node table


图例
Label
Fea
g
Node table
Feature I
Work I
Work 4
Work 7
v1
L_sum
L_count
R_sum
R_count
L_sum*L_sum/L_count+R_sum*R_sum/R_count
v2
L_sum
L_count
R_sum
R_count
L_sum*L_sum/L_count+R_sum*R_sum/R_count
v3
L_sum
L_count
R_sum
R_count
L_sum*L_sum/L_count+R_sum*R_sum/R_count
v4
L_sum
L_count
R_sum
R_count
L_sum*L_sum/L_count+R_sum*R_sum/R_count


Feature 1
图例
Label
Fea
g
Node table
v1
v2
v3
v4
Local left sum
Local left count
Local right sum
Local right count
Work 1
Local left sum
Local left count
Local right sum
Local right count
Work 4
Local left sum
Local left count
Local right sum
Local right count
Work 7
ALL_Reduce SUM
global left sum
global left count
global right sum
global right count

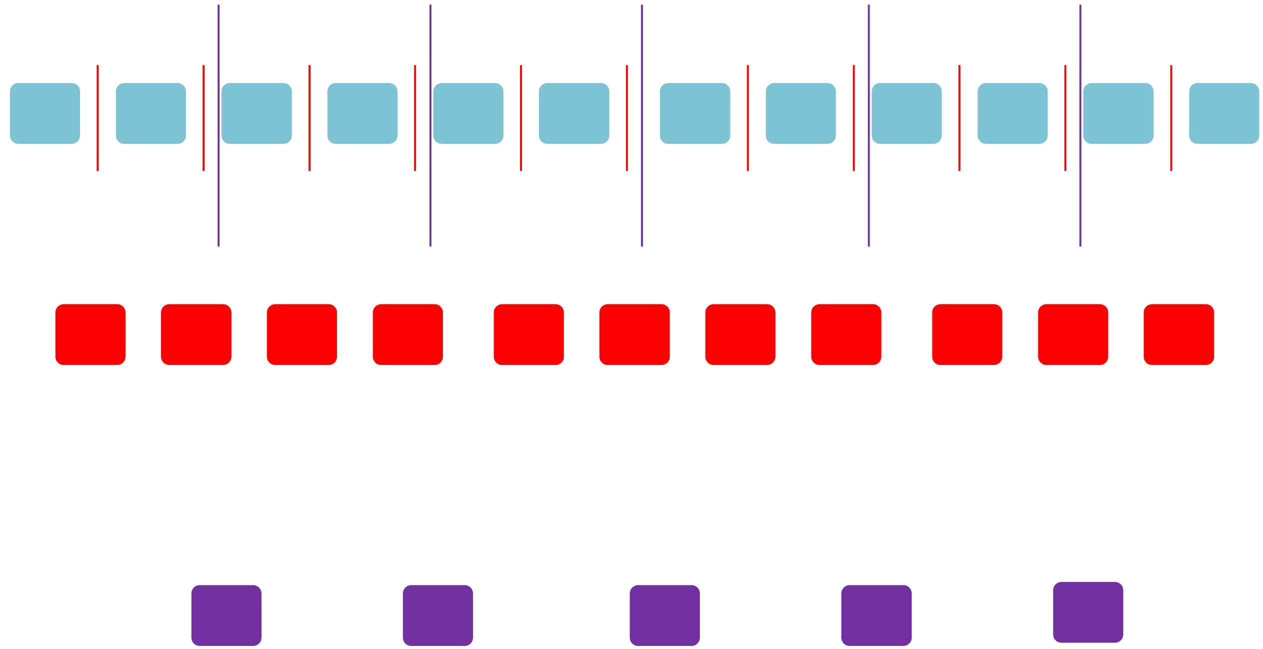

图例
Label
Fea
g
Node table
Feature 1
v1
v2
v3
v4
Local left sum
Local left count
Work 1
Local left sum
Local left count
Work 4
Local left sum
Local left count
Work 7
ALL_Reduce SUM
global left sum
global left count

split value list

样本

Split value list

Split value list



通信时间

特征个数 fea_count, 数据并行度 K组， 总work数 fea_count * K

Min V : Reduce local sum and weight

$$Num\_splits * 2 * 4 * log_2 K \approx 8kb * log_2 K$$

Min Feature

$$2 * 4 * log_2(fea_{count} * K) \quad byte$$

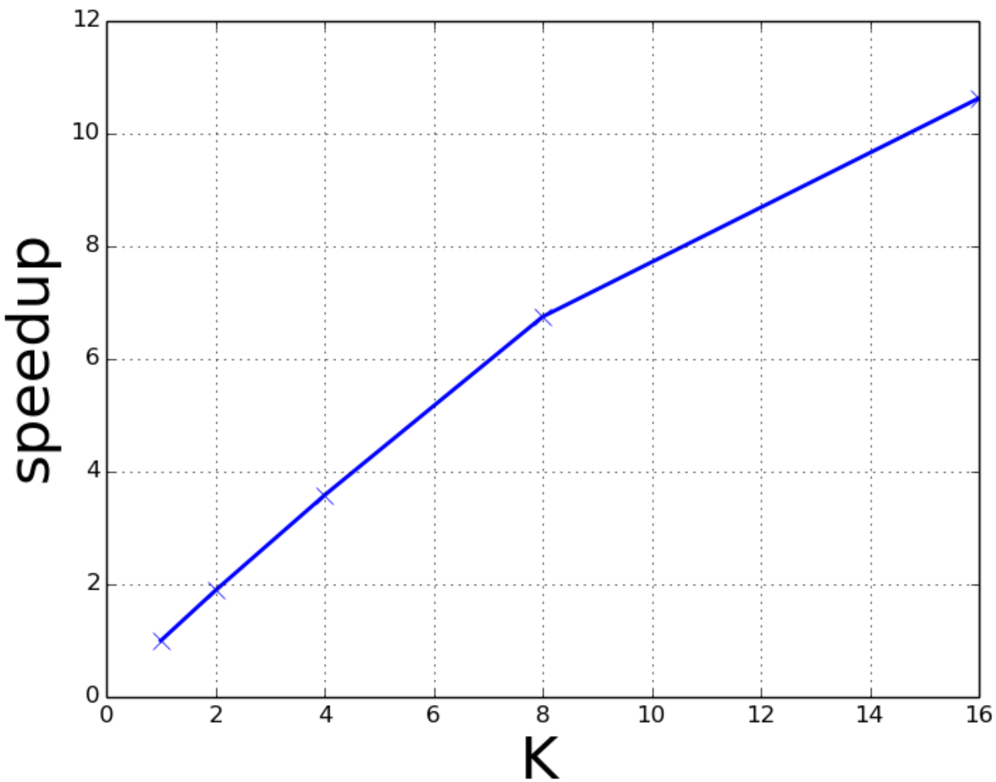
Node table

$$\frac{N}{8 * K} \text{ byte}$$

性能测试

数据量1亿2千万数据，特征150
单棵树叶子结点32 500棵树

数据度K
1
2
4
8
16



大纲

- GBDT算法介绍
- GBDT实现原理
- GBDT并行实现
- 应用

推荐场景

- 协同过滤

- LR点击率预测

Ranking- pairwise optimization

- 构建pair
  - Click >> non click
  - Cart & buy >> click

Multi-obojective

- Click & price
  - Price high >> price low

结果

- 收藏夹-猜你喜欢
  - 点击率、转化率、客单价、引导成交的全面提升
- 手淘试用品推荐
  - 点击率显著提升
- 已买到宝贝-猜你喜欢
  - 点击率显著提升

集团内业务应用

- 共享平台、搜索、B2B、蚂蚁金服、阿里妈妈、CDO、天猫、OS、iDST、淘宝技术部、菜鸟、国际事业部等十几个BU的重要业务应用

算法开放&集团外部应用

御膳房算法平台

http://pai.yushanfang.com

Q & A

Thank you