$Mx = x \cdot \lambda$

M is $n \times n$ matrix M

vector

Given M and a vector v

compute $y = M \cdot v$

$$y_i = \sum_{j=1}^{n} m_{ij} \cdot v_j$$

Assumption.
v could fit in memory but M cannot fit in memory & M is sparse.

If M is sparse, you can represent the elements of M as (i, j, m_{ij})

Map: input (i, j, m_{ij})
output: $(i, \underline{m_{ij} \cdot v_j})$

reduce (i, l)
sum = 0
for each element $x \in l$
sum += x
return (i, sum)

Eigenvectors and eigenvalue:

$R(A_1, \ldots, A_n)$

Relational Algebra defines some ops. on relational tables. (i.e. R)

$\sigma_C(R) \rightarrow$ select tuples that satisfy C.

$\Pi_A(R) \rightarrow$ Projection, $R \cup S$, $R \cap S$, $R - S$

$R_1 \times R_2 \rightarrow$ cartesian product

$R_1 \bowtie R_2 \rightarrow$ Natural join of two tables

$R_1 \bowtie R_2 = \sigma_C (R_1 \times R_2)$

↳ check all common attributes are equal.

$\phi_X(R)$ where X is a grouping attribute

or $f(A)$ where

$f \in \{$ sum, count, min, max, avg, ~~std~~

$\phi_{Nationality,\ count(A)}$ (student table)

$\sigma_C(R)$ on map-reduce

map: input is a tuple t of R

output: if t satisfies C (condition)
than output
(t, t)

reduce : (t, t)
does nothing

$\Pi_A(R)$ on map-reduce

map: input tuple t
project attributes of t based on A.
Let's call projected values t'

output: (t', t')
key value

Reduce (t' as key
and list of $[t', \ldots t']$

reduce will convert

$(t', [t', \ldots, t'])$ to

(t', t')

output (t', t')

---

union : $R \cup S$

Map: output (t, t) for both R & S.

reduce: $(t, [t, \ldots t])$

output one $(t, t) \longrightarrow$ union

For intersection: change the reduce such that if the value list has two tuples than output (t, t)

Assume we want to compute $R - S$

Map: output (t, R) for R

output (t, S) for S.

Reduce input (t, l)

If $l = [R, S] \mid l = [S, R]$
output nothing

else $l = [S]$
output nothing

else (// $L = [R]$)

output (t, t)

$R \underset{B}{\bowtie} S$: $R(A, B)$ $S(B, C)$

Map: output $(b, (a, R))$ for R

// $(b, (c, S))$ for S

reduce : input (b, l)
for each (a_i, R) in l
for each (c_i, S) in l
~~out~~ create (a_i, b, c_i)

Output:

$$\left(b,\ (a_1, b, c_1),\ (a_2, b, c_2)\ .\ (a_n, b, c_n)\right)$$

Grouping & Aggregation by Map-reduce

Given $R(A,B,C)$ $\qquad \gamma_{A,\,\phi(B)}(R)$

$\phi \to \{\text{count}, \text{avg}, \ldots\}$

Map: for each tuple of R

output (a, b)

Reduce: input $\to (a, [b_1 \ldots b_\ell])$

output $\to (a, \phi(b_1, \ldots, b_\ell))$

Given M is $n \times t$ matrix

N is $t \times r$ matrix

Our goal is to compute $P = M.N$

$$p_{ik} = \sum_{j=1}^{t} m_{ij} \cdot n_{jk}$$

$M(I, J, V)$ $\quad$ $N(J, K, W)$

for each (i, j, m_{ij})

output $(j, (M, i, m_{ij})$

for each (j, k, n_{ij})

output $(j, (N, k, n_{jk}))$

Reduce: Input (j, l)

output $(j, (i_1, k_1, m_{i_1 j} \cdot n_{jk_1})$ . . . .

$(i_l, k_l, m_{i_l j} \cdot n_{jk_l})$)

second map-reduce step

map: (key, value)

$\hookrightarrow [(i_1, k_1, v_1), \ldots (i_l, k_l, v_l)]$

output $((i_i, k_i), v_i)$ for each (i_l, k_l, v_l) in the value list.

reduce

input: $((i,k), \quad [v_1, \ldots, v_t])$

output: $\left((i,k), \sum_{i=1}^{t} v_i \right)$

Exercise: Do this only with one map-reduce!