**Answer #1 In-Mapper Pairs approach for co-occurrence matrix:**

class Mapper

method initialize()

H = new AssociativeArray()

method Map(docid a, doc d)

for all term u in record r do

for all term v in Window(u) do

H{(u,v)} = H{(u,v)} + 1 .

method close()

for all term (u,v) in H do

Emit((u,v), H{(u,v)})

class Reducer (**SAME**)

method Reduce(Pair(u, v), Integer [c1, c2, …])

s = 0

for all Integer c in [c1, c2, …] do

s = s + c .

Emit((u, v), s)

**Answer #2 In-Mapper Stripe approach for co-occurrence matrix:**

class Mapper

method initialize()

HFINAL = new AssociativeArray()

method Map(docid a, doc d)

for all term u in record r do

H = new AssociativeArray

for all term v in Window(u) do

H{v} = H{v} + 1 . //Tally words co-occurring with u

HFINAL{u} = HFINAL{u} + H;

method close()

Emit(u, HFINAL)

class Reducer (**SAME**)

method Reduce(term u, AssociativeArray [H1, H2, …])

HFINAL = new AssociativeArray

for all stripe H in [H1, H2, …] do

HFINAL = HFINAL + H //elementwise addition

Emit(u, HFINAL)

Answer #3 Illustrations

1. **Illustrate Pair approach**

Mapper #2 Output

((cat, rat), 1)

((cat, bat), 1)

((cat, rat), 1)

((rat, bat), 1)

((bat, rat), 1)

((bat, mat), 1)

((rat, bat), 1)

((rat, bat), 1)

((rat, bat), 1)

Input Split 2 : [{cat rat bat rat}, {bat mat pat bat}, {pat cat bat mat}]    (Note: 3 records)

1. Illustrate In-Mapper Combining Version of the Pair approach. (The algorithm you wrote in Q1)
2. Illustrate Stripe approach.

Mapper #0 output

(cat, [mat=1, rat=1])

(mat, [rat=1, cat=1])

(rat, [cat=1])

(cat, [bat=1])

(bat, [cat=1, pat=1])

[cat, [pat=1])

(cat, [bat=2, rat=1])

(bat, [rat=1])

(rat, [bat=1])

Mapper #1 Output

(cat, [rat=2, bat=1])

(rat, [bat=1])

(bat, [rat=1])

(bat, [mat=1, pat=1])

(mat, [pat=1, bat=1])

(pat, [bat=1])

(pat, [cat=1, bat=1, mat=1])

(cat, [bat=1, mat=1])

(bat, [mat=1])

Shuffle & Sort

(bat, [[cat=1, pat=1], [rat=1], [rat=1], [mat=1, pat=1], [mat=1]])

(cat, [[mat=1, rat=1], [bat=1], [pat=1], [bat=2, rat=1], [rat=2, bat=1], [bat=1, mat=1]])

(mat, [[rat=1, cat=1], [pat=1, bat=1]])

(pat, [[bat=1], [cat=1, bat=1, mat=1]])

(rat, [[cat=1], [bat=1] , [bat=1]])

Reducer #0 Input

(bat, [[cat=1, pat=1], [rat=1], [rat=1], [mat=1, pat=1], [mat=1]])

(cat, [[mat=1, rat=1], [bat=1], [pat=1], [bat=2, rat=1], [rat=2, bat=1], [bat=1, mat=1]])

Reducer #1 Input

(mat, [[rat=1, cat=1], [pat=1, bat=1]])

(pat, [[bat=1], [cat=1, bat=1, mat=1]])

(rat, [[cat=1], [bat=1] , [bat=1]])

Reducer #0 Output

(bat, [cat=1, pat=2, rat=2, mat=2])

(cat, [mat=2, rat=4, bat=5, pat=1])

Reducer #1 Output

(mat, [rat=1, cat=1, pat=1, bat=1])

(pat, [bat=2, cat=1, mat=1]])

(rat, [cat=1, bat=2])

1. Illustrate In-Mapper Combining Version of the Stripe approach. (The algorithm you wrote in Q2)

Mapper #0 output

(cat, [pat=1, bat=3, mat=1, rat=2])

(mat, [rat=1, cat=1])

(rat, [cat=1, bat=1])

(bat, [rat=1, cat=1, pat=1])

Mapper #1 Output

(bat, [rat=1, mat=2, pat=1])

(cat, [rat=2, bat=2, mat=1])

(mat, [pat=1, bat=1])

(pat, [cat=1, bat=2, mat=1])

Shuffle & Sort

(bat, [[rat=1, cat=1, pat=1], [rat=1, mat=2, pat=1]])

(cat, [[pat=1, bat=3, mat=1, rat=2], [rat=2, bat=2, mat=1]])

(mat, [pat=1, bat=1], [rat=1, cat=1])

(pat, [cat=1, bat=2, mat=1])

(rat, [cat=1, bat=1])

Reducer #0 Input

(bat, [[rat=1, cat=1, pat=1], [rat=1, mat=2, pat=1]])

(cat, [[pat=1, bat=3, mat=1, rat=2], [rat=2, bat=2, mat=1]])

Reducer #1 Input

(mat, [pat=1, bat=1], [rat=1, cat=1])

(pat, [cat=1, bat=2, mat=1])

(rat, [cat=1, bat=1])

Reducer #0 Output

(bat, [rat=2, cat=1, pat=2, mat=2])

(cat, [pat=1, bat=5, mat=2, rat=4])

Reducer #1 Output

(mat, [pat=1, bat=1, rat=1, cat=1])

(pat, [cat=1, bat=2, mat=1])

(rat, [cat=1, bat=2])