Thi Hong Yen Truong
ID: 614745

W1D5

Q1: Write an in-mapper combiner algorithm modifying Co-occurrence Matrix (Pairs approach) algorithm.

```
class Mapper
        method initialized()
                H = new AssociativeArray()

        method Map(docid a, doc b)
                for all term u in record r do
                        for all term v in Window(u) do
                                if (H{(u, v)} is null)
                                        H{(u, v)} = 1
                                else
                                        H{(u, v)} = H{(u, v)} + 1


        method close()
                for all Pair(u, v) in H do
                        Emit((u, v), H{(u, v)})


class Reducer
        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)
```

Q2: Write an in-mapper combiner algorithm modifying Co-occurrence Matrix (Stripe approach)

```
class Mapper
        method initialize()
                H = new AssociativeArray()

        method Map(docid a, doc d)
                for all term u in record r do
                        H_u = new AssociativeArray()
                        for all term v in Window(u) do
```

```
                    if (H_u{v} is null)
                        H_u{v} = 1
                    else
                        H_u{v} = H_u{v} + 1
                if (H{u} is null)
                    H{u} = H_u
                else
                    H{u} = H{u} + H_u

    method close()
        for all stripe u in H do
            Emit(u, H{u})


class Reducer
    method Reduce(term u, AssociativeArray [H1, H2, ...])
        H_FINAL = new AssociativeArray()
        for all stripe H in [H1, H2, ...] do
            H_FINAL = H_FINAL + H
        Emit(u, H_FINAL)
```

Q3: Assume that there are two input splits and two reducers. Note that Mapper 1 and Reducer 1 run on the same machine. Mapper 2 and Reducer 2 run on the same machine.

Further, let the partitioner assign all words less than letter 'K' to Reducer 1 and everything else to Reducer 2.

Input Split 1: [{cat mat rat, cat}, {cat bat cat pat}, {cat bat rat bat}] (Note: 3 records)

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

Let the window of X, W(X) be set of all term after X and before the next X.

Example: Let Data block be [a b c a d e]

W(a) = {b, c}, W(b) = {c, a, d, e}, W(c) = {a, d, e}, W(a) = {d, e}, W(d) = {e}, W{e} = {}

Machine 1	Machine 2
Input Split 1	Input Split 2
cat mat rat cat	cat rat bat rat

cat bat cat pat cat bat rat bat	bat mat pat bat pat cat bat mat

1. Illustrate Pair approach

Machine 1	Machine 2
Input Split 1	Input Split 2
cat mat rat cat cat bat cat pat cat bat rat bat	cat rat bat rat bat mat pat bat pat cat bat mat
W(X) 1	W(X) 2
W(cat) = {mat, rat) W(mat) = {rat, cat} W(rat) = {cat}	W(cat) = {rat, bat, rat} W(rat) = { bat} W(bat) = { rat}
W(cat) = {bat} W(bat) = {cat, pat} W(cat) = {pat}	W(bat) = { mat, pat} W(mat) = { pat, bat} W(pat) = { bat}
W(cat) = {bat, rat, bat} W(bat) = { rat} W(rat) = {bat}	W(pat) = { cat, bat, mat} W(cat) = {bat, mat} W(bat) = { mat}
Mapper 1	Mapper 2
((cat, mat), 1) ((cat, rat), 1) ((mat, rat), 1) ((mat, cat), 1) ((rat, cat), 1)	((cat, rat), 2) ((cat, bat), 1) ((rat, bat), 1) ((bat, rat), 1)
((cat, bat), 1) ((bat, cat), 1) ((bat, pat), 1)	((bat, mat), 1) ((bat, pat), 1) ((mat, pat), 1)

((cat, pat), 1)	((mat, bat), 1) ((pat, bat), 1)
((cat, bat), 2) ((cat, rat), 1) ((bat, rat), 1) ((rat, bat), 1)	((mat, bat), [1]) ((mat, cat), [1]) ((mat, rat), [1]) ((mat, pat), [1]) ((pat, bat), [1, 1]) ((pat, cat), [1]) ((pat, mat), [1]) ((rat, bat), [1,1]) ((rat, cat), [1])
Reducer 1 Input	Reducer 2 Input
((bat, cat), [1]) ((bat, mat), [1, 1]) ((bat, pat), [1,1]) ((bat, rat), [1,1]) ((cat, bat), [1,1,1,1]) ((cat, mat), [1,1]) ((cat, pat), [1]) ((cat, rat), [1,1, 2])	((mat, bat), [1]) ((mat, cat), [1]) ((mat, rat), [1]) ((mat, pat), [1]) ((pat, bat), [1, 1]) ((pat, cat), [1]) ((pat, mat), [1]) ((rat, bat), [1,1]) ((rat, cat), [1])
Reducer 2 Output	Reducer 2 Output
((bat, cat), 1) ((bat, mat), 2) ((bat, pat), 2) ((bat, rat), 2) ((cat, bat), 4) ((cat, mat), 2) ((cat, pat), 1)	((mat, bat), 1) ((mat, cat), 1) ((mat, rat), 1) ((mat, pat), 1) ((pat, bat), 2) ((pat, cat), 1) ((pat, mat), 1)

((cat, rat), 4)	((rat, bat), 2) ((rat, cat), 1)

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

Machine 1	Machine 2
W(X) 1	W(X) 2
W(cat) = {mat, rat) W(mat) = {rat, cat} W(rat) = {cat}	W(cat) = {rat, bat, rat} W(rat) = { bat} W(bat) = { rat}
W(cat) = {bat} W(bat) = {cat, pat} W(cat) = {pat}	W(bat) = { mat, pat} W(mat) = { pat, bat} W(pat) = { bat}
W(cat) = {bat, rat, bat} W(bat) = { rat} W(rat) = {bat}	W(pat) = { cat, bat, mat} W(cat) = {bat, mat} W(bat) = { mat}
Mapper 1	Mapper 2
((cat, mat), 1) ((cat, rat), 2) ((mat, rat), 1) ((mat, cat), 1) ((rat, cat), 1) ((cat, bat), 3) ((bat, cat), 1) ((bat, pat), 1) ((cat, pat), 1) ((bat, rat), 1) ((rat, bat), 1)	((cat, rat), 2) ((cat, bat), 2) ((rat, bat), 1) ((bat, rat), 1) ((bat, mat), 2) ((bat, pat), 1) ((mat, pat), 1) ((mat, bat), 1) ((pat, bat), 2) ((pat, cat), 1) ((pat, mat), 1) ((cat, mat), 1)
Reducer 1 Input	Reducer 2 Input

((bat, cat), [1]) ((bat, mat), [1, 1]) ((bat, pat), [1,1]) ((bat, rat), [1,1]) ((cat, bat), [3,2]) ((cat, mat), [1,1]) ((cat, pat), [1]) ((cat, rat), [2, 2])	((mat, bat), [1]) ((mat, cat), [1]) ((mat, rat), [1]) ((mat, pat), [1]) ((pat, bat), [2]) ((pat, cat), [1]) ((pat, mat), [1]) ((rat, bat), [1,1]) ((rat, cat), [1])
Reducer 1 Output	Reducer 2 Output
((bat, cat), 1) ((bat, mat), 2) ((bat, pat), 2) ((bat, rat), 2) ((cat, bat), 5) ((cat, mat), 2) ((cat, pat), 1) ((cat, rat), 4)	((mat, bat), 1) ((mat, cat), 1) ((mat, rat), 1) ((mat, pat), 1) ((pat, bat), 2) ((pat, cat), 1) ((pat, mat), 1) ((rat, bat), 2) ((rat, cat), 1)

3. Illustrate Stripe approach.

Machine 1	Machine 2
W(X) 1	W(X) 2
W(cat) = {mat, rat) W(mat) = {rat, cat} W(rat) = {cat}	W(cat) = {rat, bat, rat} W(rat) = { bat} W(bat) = { rat}
W(cat) = {bat} W(bat) = {cat, pat} W(cat) = {pat}	W(bat) = { mat, pat} W(mat) = { pat, bat} W(pat) = { bat}

W(cat) = {bat, rat, bat} W(bat) = { rat} W(rat) = {bat}	W(pat) = { cat, bat, mat} W(cat) = {bat, mat} W(bat) = { mat}
Mapper 1	Mapper 2
(cat, [mat: 1, rat: 1]) (mat, [rat: 1, cat: 1]) (rat, [cat: 1])	(cat, [rat: 2, bat: 1]) (rat, [bat: 1, rat: 1])
(cat, [mat: 1, rat: 1]) (bat, [cat: 1, pat: 1]) (cat, [pat: 1])	(bat, [mat: 1, pat: 1]) (mat, [pat: 1, bat: 1]) (pat, [bat: 1])
(cat, [bat: 2, rat: 1]) (bat, [rat: 1]) (rat, [bat: 1])	(pat, [cat: 1, bat: 1, mat: 1]) (cat, [bat: 1, mat: 1]) (bat, [mat: 1])
Reducer 1 Input	Reducer 2 Input
(bat, [[cat: 1, pat: 1], [rat: 1], [mat: 1, pat: 1], , [mat: 1]]) (cat, [[mat: 1, rat: 1], [mat: 1, rat: 1], [pat: 1], [bat: 2, rat: 1], [rat: 2, bat: 1], [bat: 1, mat: 1]])	(mat, [[pat: 1, bat: 1], [bat: 1], [cat: 1, bat: 1, mat: 1]]) (pat, [[cat: 1, bat: 1, mat: 1], [bat: 1]) (rat, [[cat: 1], [bat: 1], [bat: 1, rat: 1]])
Reducer 1 Output	Reducer 2 Output
(bat, [cat: 1, mat: 2, pat: 2, rat: 1]) (cat, [bat: 4, mat: 3, pat: 1, rat: 5])	(mat, [bat: 3, cat: 1, mat: 1, pat: 1]) (pat, [cat: 1, bat: 2, mat: 1]) (rat, [bat: 2, cat: 1, rat: 1])

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

Machine 1	Machine 2
Input Split 1	Input Split 2

cat mat rat cat cat bat cat pat cat bat rat bat	cat rat bat rat bat mat pat bat pat cat bat mat

Machine 1	Machine 2
W(X) 1	W(X) 2
W(cat) = {mat, rat) W(mat) = {rat, cat} W(rat) = {cat}	W(cat) = {rat, bat, rat} W(rat) = { bat} W(bat) = { rat}
W(cat) = {bat} W(bat) = {cat, pat} W(cat) = {pat}	W(bat) = { mat, pat} W(mat) = { pat, bat} W(pat) = { bat}
W(cat) = {bat, rat, bat} W(bat) = { rat} W(rat) = {bat}	W(pat) = { cat, bat, mat} W(cat) = {bat, mat} W(bat) = { mat}
Mapper 1	Mapper 2
(bat, [cat: 1, pat: 1, rat: 1]) (cat, [bat: 2, mat: 2, pat: 1, rat: 3]) (mat, [rat: 1, cat: 1]) (rat, [bat: 1, cat: 1])	(bat, [mat: 2, pat: 1]) (cat, [bat: 2, mat: 1, rat: 2]) (mat, [pat: 1, bat: 1]) (pat, [cat: 1, bat: 2, mat: 1]) (rat, [bat: 1, rat: 1])
Reducer 1 Input	Reducer 2 Input
(bat, [[cat: 1, pat: 1,rat: 1], [mat: 2, pat: 1]]) (cat, [[bat: 2, mat: 2, pat: 1, rat: 3], [bat: 2, mat: 1, rat: 2]])	(mat,[[rat: 1, cat: 1], [pat: 1, bat: 1]]) (pat, [[cat: 1, bat: 2, mat: 1]]) (rat, [[bat: 1, rat: 1], [bat: 1, cat: 1]])
Reducer 1 Output	Reducer 2 Output
(bat, [cat: 1, mat: 2, pat: 2, rat: 1]) (cat, [bat: 4, mat: 3, pat: 1, rat: 5])	(mat, [bat: 3, cat: 1, mat: 1, pat: 1]) (pat, [cat: 1, bat: 2, mat: 1]) (rat, [bat: 2, cat: 1, rat: 1])