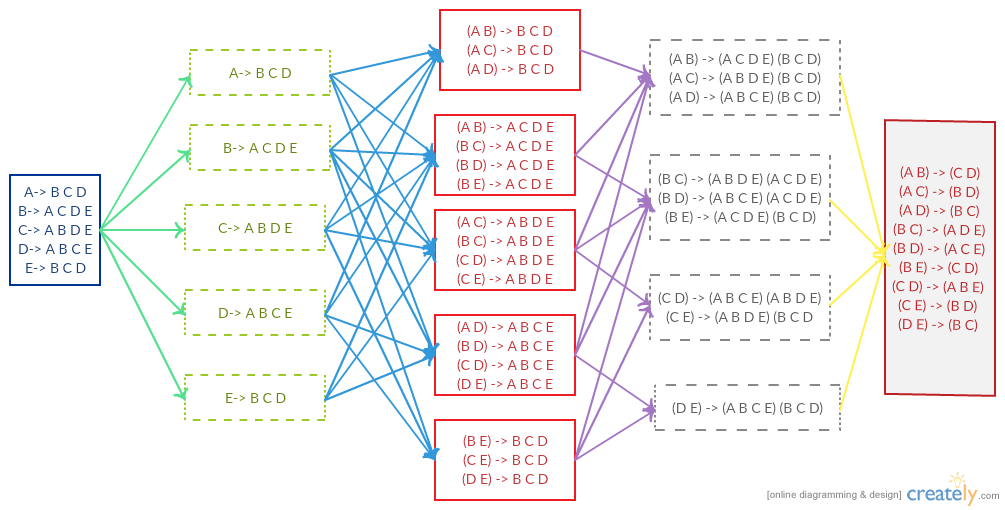
**Knowledge Discovery and Management  
Problem Set (PS-2B)**

Name: Kalyan Kilaru

Class ID: 10

1) Draw a MapReduce diagram similar to the word count diagram below.



2) Sketch a MapReduce algorithm for the common Facebook friends.

**MAP():**

//Key is the person

//value is a list of friends for this key=person

//value=(friend1, friend2…..friendN)

Class Mapper

Method Map(key, value)

{

reducerValue=(friend1, friend2,….friendN);

for all friend in (friend1, friend2,….friendN)

{

reducerKey= buildSortedKey(person,friend);

emit(reducerKey, reducerValue);

}

}

**BuildStoreKey():**

Tuple2 buildStoreKey(person1, person2){

If(person1 < person2){

Return new Tuple2(person1, person2);

}

Else{

Return new Tuple2(person2, person1);

}

}

**Reduce():**

//key=Tuple(person1,person2)

//value= list{list\_1,List\_2,……List\_L}

//Each list is a set of unique user ID’s

Reduce(key, value){

Output=key;

OutputValue=intersection(List\_1,List\_2……..)

Emit(outputKey, outputValue);

}

3) Sketch Spark Scala implementation (referring to the word count code below).

**Map():**

Map(P, {Friend\_1, Friend\_2,….Friend\_n}) {

Friends = {Friend\_1, Friend\_2,….Friend\_n};

For(f: friends) {

Key=buildSortedTuple(P,f);

Emit(key,friends);

}

}

\*

PairFlatMapFunction<T, K, V>

T= Iterable<Tuple2<k, v>>

**Reduce():**

//key=Tuple(user1,user2)

//values=List(list(user))

Reduce(key, values) {

commonFriends = intersection(values);

emit(key, friends);

}

\*

JavaPairRdd<Tuple2<>, Iterable<Iterable>>

grouped= Pairs.groupByKey();