Finding mutual friends for every two friends

MapReduce Java

Sample Input

A->(B,D,E)

B->(A,C)

C->(B,E)

D->(A)

E->(A,C)

Description: A->(B,D,E) indicates that person ‘A’ has friends (B,D,E). Hence, it is required for B, D and E to have ‘A’ in their friend list.

Expected output

Output Sample:

(A,C)=>(B,E)

(B,D)=>(A)

(B,E)=>(A,C)

(D,E)=>(A)

Description: Line by line output, (A,C)=>(B,E) indicates that B and E are mutual friends of A and C. Meaning B and E are common friends to A and C.

Know that for manual solution simplicity's sake names within input and output are chosen alphabetically sorted. Otherwise input as well output can be any random ordered names. Know that in output we do not expect pairs if there are no common friends.

(A,B)=>() (A,D)=>() (A,E)=>() (B,C)=>() (C,D)=>() (C,E)=>() are not displayed as empty result set. No mutual friends.

map(offsetkey, linevalue) -> (intermediate key, intermediate value)

Linevalue#1

A->(B,D,E)

Result of map()

(ikey, ivalue)

<(B,D),A>

<(B,E),A>

<(D,E),A>

Linevalue#2

B->(A,C)

Result of map()

(ikey, ivalue)

<(A,C),B>

Linevalue#3

C->(B,E)

Result of map()

(ikey, ivalue)

<(B,E),C>

Linevalue#4

D->(A)

Result of map()

(ikey, ivalue)

empty

Linevalue#5

E->(A,C)

Result of map()

(ikey, ivalue)

<(A,C),E>

Sort and shuffle

Input All Pairs resulted from mapper

<(B,D),A> <(B,E),A> <(D,E),A> <(A,C),B> <(B,E),C> <(A,C),E>

Output <key,[list of values]>

<(B,D),A> <(B,E),[A,C]>

<(D,E),A> <(A,C),[B,E]>

***** Time to guess, what will be the logic written in reducer function provided above is input?

Reducer

The function with no logic.

Empty body. Just a forward input to output.

Framework dumps all context added <key,value> pairs, which were generated as result of reducer to HDFS output location.