Finding triangle using mapreduce

you need to examine the friend lists of at least two members of a potential friend triangle. If A is friends with B and C, and B is friends with A and C, then A,B,C is a friend triangle.

Mapper will emit all possible friend triangles given a single friend list, and reducer will emit friend triangles emitted by at least two mappers.

Mapper will read in a line of input. It will then emit every combination of the 'host' (owner of the list) and two friends in the list as a key and then 1 as a value. For example, if given:

"A B C"

as a line of input where A is the 'host' and B and C are the friends, then my Map function will emit:

"<A,B,C> -> 1", "<A,C,B> -> 1", "<B,A,C> -> 1", "<B,C,A> -> 1", "<C,A,B> -> 1", "<C,B,A> -> 1"

Note that the mapper will emit the same three elements in all different orderings.

The mapper takes as input a string representing a friend list. The mapper will then output a set of keys representing every possible friend triangle containing the owner of the friend list. A single key will be a string representing a single possible friend triangle. To validate the friend triangle, another mapper processing another friend list must emit the same key. For each key, the mapper emits the integer '1' as a value. This helps the reducer keep count of the number of mappers emitting the same key by aggregating all values with the same key together. If the aggregate value is greater than one, then the friend triangle is validated.

The mapper emits every

possible friend triangle present in this graph. The Reducer's job is to validate which potential friend triangles are real and which are incomplete. If a Reducer receives two or more messages with the same key, then there exists a friend triangle described by that key. If a reducer ever only receives one copy of a given key, then that key doesn't represent a valid friend triangle.

Friends of A are B, C, D, E , F.

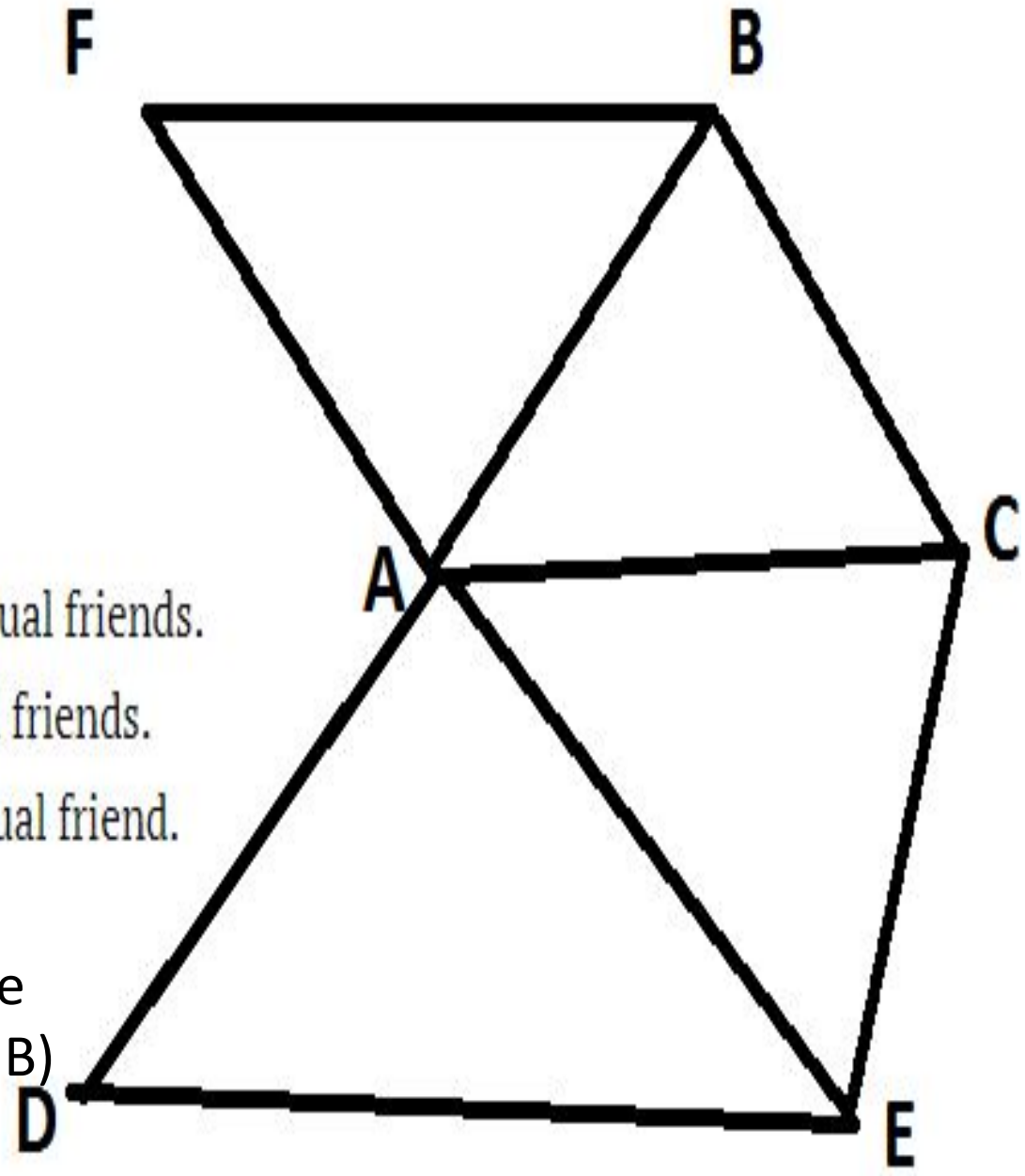
Friends of B are A, C, F.

Friends of C are A, B, E

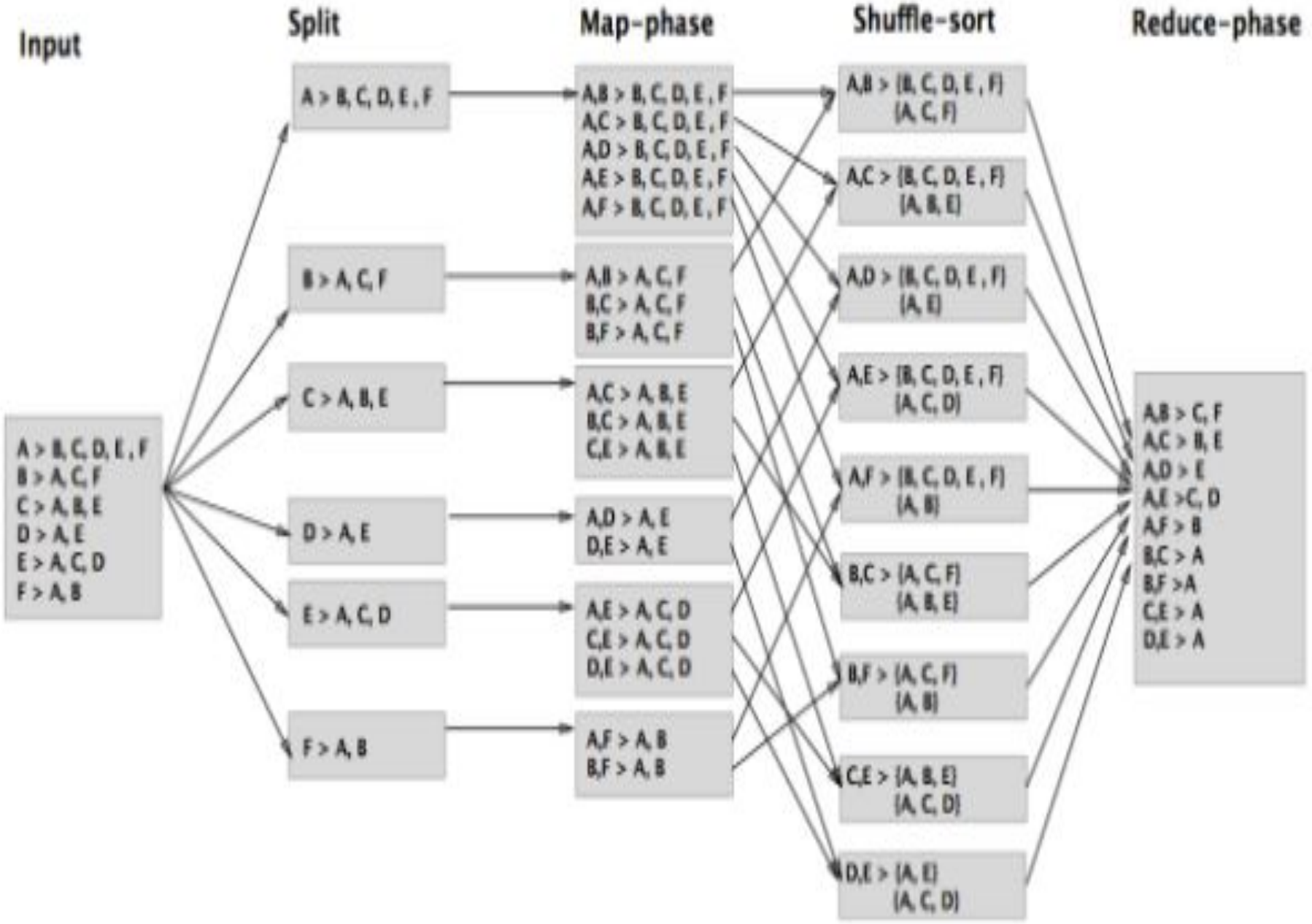
So A and B have C, F as their mutual friends.

A and C have B, E as their mutual friends.

B and C have only A as their mutual friend.



So when A visits B's profile
We can quickly lookup (A B)
and see that they have
two friends in common, (C F).


Input
Split
Map-phase
Shuffle-sort
Reduce-phase
A > B, C, D, E, F
B > A, C, F
C > A, B, E
D > A, E
E > A, C, D
F > A, B
A,B > B, C, D, E, F
A,C > B, C, D, E, F
A,D > B, C, D, E, F
A,E > B, C, D, E, F
A,F > B, C, D, E, F
A,B > A, C, F
B,C > A, C, F
B,F > A, C, F
A,C > A, B, E
B,C > A, B, E
C,E > A, B, E
A,D > A, E
D,E > A, E
A,E > A, C, D
C,E > A, C, D
D,E > A, C, D
A,F > A, B
B,F > A, B
A,B > [B, C, D, E, F]
[A, C, F]
A,C > [B, C, D, E, F]
[A, B, E]
A,D > [B, C, D, E, F]
[A, E]
A,E > [B, C, D, E, F]
[A, C, D]
A,F > [B, C, D, E, F]
[A, B]
B,C > [A, C, F]
[A, B, E]
B,F > [A, C, F]
[A, B]
C,E > [A, B, E]
[A, C, D]
D,E > [A, E]
[A, C, D]
A,B > C, F
A,C > B, E
A,D > E
A,E > C, D
A,F > B
B,C > A
B,F > A
C,E > A
D,E > A