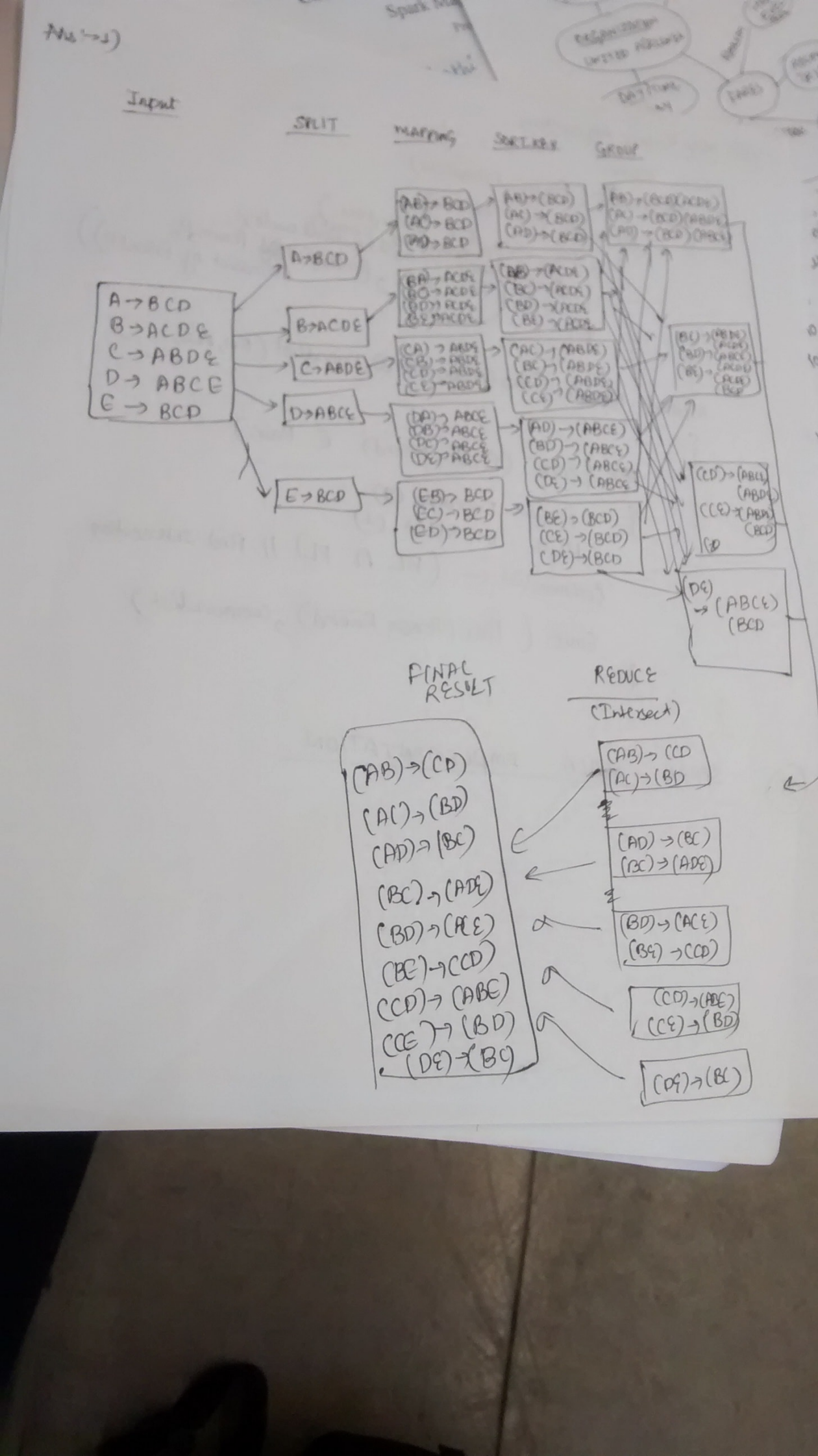
Input
SPLIT
MAPPING
SHUFFLE
GROUP
A→BCD
B→ACDE
C→ABDE
D→ABCE
E→BCD
FINAL RESULT
REDUCE
(Intersect)

(2) Map Reduce Algorithm

```
class Mapper
{
  Method Map (Person, Friendlist)
  {
    (for each P ∈ Personlist)
      { (for each F ∈ Friendlist)
        FL ← (Friendlist of friend)
        { Sort P and F in alphabetical order
          emit ( Pair (P, F) , (Friendlist of Person P,
                                Friendlist of friend F))
  } } }
  Method Reduce ( Pair(Person, Friend), Friendlist (FL1, FL2, ....)
  {
      For (each (Person, Friend) ∈ Pair)
      {
          FL ← Friendlist (2)
          PL ← Friend list (1)
        Common_list ← (FL ∩ PL) // find Intersection

          Emit ( Pair (Person, Friend) , Common_list)
      }
  }
```

(3) SPARK SCALA IMPLEMENTATION

PairFlatMapFunction <T, K, V>

T = Iterable <Tuple2 <K, V>>

Reduced():

// key = Tuple (user1, user2)

// values = List ((list (user))

Reduce (key, values)

{

commonfriends = intersection (values);

emit (key, friends);

}

JavaPairRdd <Tuple2 <>, Iterable <Iterable >>

grouped = Pairs.groupByKey ();