# 整卷

1.Assume that a 10000 server cluster is built from reliable machines with a mean-time between failures (MTBF) of 2000 days then what is the failure rate?

**Failure rate = No of server / MTBF = 10000 / 2000 days = 5servers / day**

**Failure rate = No of server / MTBF = 5000/ 2000 days = 2.5servers / day**

2.Compute the time required to read 1 TB data using on a machine having 2 I/O channels (or 2 hard drives) such that each can read 400 MB/sec

**(1024 \* 1024) / (2 \* 400 \* 60 ) = 21.84 mins**

**(1024\*1024) / (200 \* 2 \* 60 ) = 43.7 mins ( 1TB,2 hard,200Mb/s)**

3.List four distinct features of the problem from which you can conclude it is not suitable for MapReduce framework.

**1.Latenc - Due to the fixed cost incurred by each MapReduce job submitted, application that requires low latency time or random access to a large set of data is infeasible.**

**2.batch processing- Since MapReduce is suitable only for batch processing jobs, implementing interactive jobs and models becomes impossible.**

**3.iterative - mplementing iterative map reduce jobs is expensive due to the huge space consumption by each job.**

**4.Caching - In Hadoop, MapReduce cannot cache the intermediate data in-memory for a further requirement which diminishes the performance of hadoop**

**4.**What is the minimum block size in Hadoop?

**The minimum block size in hadoop is 64Mb**

**5.**What is the specific function of the secondary namenode in Hadoop (Be **clear and specific**.)?

Specific functions of secondary name node in hoadoop is to read the metadata from the RAM of primary name node and store that data in the hard disk. This helps retrieve the metadata in case when the primary name node crashes

6.Assume that there are five data blocks (A, B, C, D, E, F), four racks (R1, R2, R3, R4) such that R1 is closer to R2, R2 is closer to R3, R3 is closer to R4 and R4 is closer to R3. Also assume that each rack has 5 datanodes. Illustrate a possible placement of block in HDFS under default settings

|  |  |  |  |
| --- | --- | --- | --- |
| R1 | R2 | R3 | R4 |
| A | BE | C | D |
| B | F | D | C |
| B | F | D | C |
| F | A | E |  |
|  | A | E |  |

|  |  |  |  |
| --- | --- | --- | --- |
| R1 | R2 | R3 | R4 |
| A | A | B | C |
| D | AB | B | C |
|  | D | CE | F |
|  | DE | E | F |
|  |  | F |  |

7.Explain how map and fold (of functional programming) can be used to count difference between number of odd numbers and number of even numbers. In particular, write pseudo-code for functions f and g.

Examples: [9, 7, 2, 1] = 2 (Since 3 – 1 = 2) [2, 9, 4] = -1 (Since 1 – 2 = -1)

**Function f(x){**

**If(x % 2 == 0) return -1;**

**Else return 1;**

**}**

**F(x,y) = x + y inital value = 0**

**8.**Explain how map and fold (of functional programming) can be used to determine whether or not **a sequence of four or more zeroes** is present. In particular, write pseudo-code for functions f and g. If the answer is “yes” it must return 1; else any value other than 1.

Examples:

[6, 7, 5, **0, 0, 0, 0**, 3, 4, 0, 0, 1] = 1 (since there is a sequence of 4 zeroes)

[6, 7, 5, 0, 0, 0, 3, 4, **0, 0, 0, 0, 0,** 1] = 1 (since there is a sequence of 5 zeroes)

[6, **0, 0, 0, 0,** 9, **0, 0, 0, 0,** 8, 1] = 1 (since there is a sequence of 4 zeroes)

[5, 0, 0, 0, 3, 4, 0, 0, 8, 0, 0, 1] = some value other than 1, may be 2 (since there is no sequence of four or more zeroes)

[5, 0, 0, 3, 4, 0, 0, 8, 0, 7, 1] = some value other than 1, may be 5 (since there is no sequence of four or more zeroes)

**(a)**Write a **MapReduce** algorithm to compute the average length of all words that begins with letter A or a, the average length of all words that begins with letter B or b, the average length of all words that begins with letter C or c and so on in a collection of English documents. **PLEASE DO NOT USE COMBINER OR IN-MAPPER COMBINING.**

For example, if the document is [apple bat Day Zebra] the result is as follows:

(A, 5) // Words that begin with A: {apple}. Average length (5)/1 = 5.

(B, 2.5) // Words that begin with B: {bat, be}. Average length (3 + 2)/2 = 2.5

(D, 3) // Words that begin with D: {done, do, Day}. Average length (4+2+3)/3 = 3.

(Z, 5) // Words that begin with Z: {Zebra}. Average length (5)/1 = 5.

**(b)**Assume that there are two mappers and two reducers. Note that Mapper 1 and Reducer 1 run on the same machine. Similarly, Mapper 2 and Reducer 2 run on the same machine. **Illustrate the behavior of your algorithm if there are four Data blocks. That means, show mapper output, reducer input and reducer output.**

Data block 1 : [care rope marks card]

Data block 2 : [may crime boat made]

Data block 3 : [boy paid me be ]

Data block 4 : [pay cat bat parks]

Also assume that File1 and File 2 are assigned to Mapper 1, File 3 and File 4 are assigned to Mapper 2. Further, let the partitioner assign all words less than letter ‘k’ to Reducer 1 and everything else to Reducer 2.

**(c) Modify your program in (a) by adding a combiner**

* Write the pseudo-code for the mapper.
* List the key value pairs emitted by Mapper 1. List the key value pairs emitted by Mapper 2.
* List the input for Reducer 1. List the input for Reducer 2,

**(d) Modify your program in (a) so that it uses in-mapper combining design pattern.**

* Write the pseudo-code for the mapper.
* List the key value pairs emitted by Mapper 1. List the key value pairs emitted by Mapper 2.
* Count the number of key value pairs send across the network.

**Decode the following assuming varInt coding was in effect.**

**0110 1001 1011 0100 1100 0001 0010 1010 0111 0000 1000 1101**

**Step1. 0110 1001 ---> 64 + 32 + 8 + 1 = 105**

**Step2. 1011 0100 ---> 32 + 16 + 4 + 52**

**Step3. 1100 0001 ---->64 + 1 = 65**

**Step4. 0010 1010 ---->32+ 8 + 2 = 42**

**Step5. 0111 0000 ---->64 + 32 + 16 = 112**

**Step6. 1000 1101 ---->8 + 4 + 1= 13**

|  |  |  |
| --- | --- | --- |
| **105 \* 27 +52** | **65** | **42 \* (27)2 + 112 \* 27 + 13** |

1. **( a) write an efficient mapreduce algorithm to compute the "expected frequency “of every term in a collection of documents , assume that your mapper do not use in-mapper combiner or a combiner.**

**We define " expected frequency " as the number of times the term appear in all documents divided by total number of terms .**

**Example :**

**File 1 : [ cat mat]**

**File 2 : [ cat pat cat ]**

**File 3 : [ cat rat bat ]**

**File 4 : [ mat bat]**

**File 5 : [pat bat ]**

**Number of “cat”=4 , number of "bat" is 3 , number of " pat"= 2 , number of “rat”= 1 , number of " mat"= 2 . total number of all words is 12 . so " expected frequency of "cat" =4 / 12 , expected frequency of "bat " =3 / 12 , expected frequency of " pat"= 2 / 12, " expected frequency "rat " =1/12 and "expected frequency "mat "= 2 / 12 .**

**( b ) assume that there are two mappers and two reducers . note that mapper i and Reducer i run on the same machine . similarly , mapper 2 and reducer 2 run on the same machine illustrate the behavior of your algorithm if there are five files :**

**File 1 : [ cat mat]**

**File 2 : [ cat pat cat ]**

**File 3 : [ cat rat bat ]**

**File 4 : [ mat bat]**

**File 5 : [pat bat ]**

**Also assume that file 1 and file 2 are assigned to mapper 1 , file 3 , file 4 and file 5 is assigned to mapper 2 . further , let the partitioner assign all words less than letter “1” to reducer 1 and everything else to reducer 2**

**(i ) count the number of key value pairs emitted by mappers 1 , 2 ~~and 3~~**

**(ii ) count the number of key value pairs send across the network**

**(C ) write a combiner .**

**(D ) assume everything in ( b ) plus the presence of the combiner you wrote in ( C) . Count the number of key value pairs send across the network .**

1. **Part (A ) Given user’s first name, last name and time used , your job is to write a Mapreduce algorithm ( with No combiner; No in-mapper combining to compute the maximum and minimum time used by each user.**

**See the sample input and the corresponding output produced . your algorithm must accomplish the same. Illustrate your algorithm provide pseudo code for partitioner and comparator consistent with reducer output**

**Mapper 1:**

**Input split 1**

**Mark dove 38**

**Chris dove 27**

**Mark dove 16**

**Input split 2**

**Joy cox 30**

**Chris dove 41**

**Mark cox 51**

**Mapper 2 :**

**Input split 3**

**Adam day 42**

**Chris dove 36**

**Mark cox 29**

**Input s plit 4**

**Joy cox 88**

**Chris dove 100**

**Joy day 67**

**Reducer 1 output**

**Cox Joy 88 30**

**Cox Mark 51 29**

**Reducer 2 output**

**Dove Chris 100 27**

**Dove Mark 38 16**

**Day Adam 42 42**

**Part (b) Write a mapreduce algorithm with combiner ( no in-mapper combining to solve the same problem Illustrate your algorithm**

**Part (c ) Write a mapreduce algorithm with in-mapper combining to solve the same problem Illustrate your algorithm**

**Part(a)**

**class Mapper {**

**method map(DocID id, Doc d) {**

**for each record do {**

**fname <- getFname();**

**lname <- getLname();**

**time <- getTime();**

**Emit( new Pair <lname, fname>, int time);**

**}**

**}**

**}**

**class Reducer {**

**method reduce(Pair p, int [t1, t2, …]) {**

**t\_min = t1;**

**t\_max = t1;**

**for each t in [t2, …] do {**

**if (t\_min > t) t\_min = t;**

**if (t\_max < t) t\_max = t;**

**}**

**Emit( Pair p, new ValuePair <t\_max, t\_min> );**

**}**

**}**

**class Partitioner {**

**method getPartition(Key (k1, k2)) {**

**ch = UPPERCASE(k1).charAt(0);**

**if ( ch < ‘D’) return 1;**

**else return 2;**

**}**

**}**

**class Pair {**

**Lname : String;**

**Fname : String;**

**method compareTo(Pair p1, Pair p2){**

**// returns -1 if p1 < p2; returns 1 if p1 > p2; returns 0 if p1 == p2**

**If (p1.Lname < p2.Lname) return 1;**

**else if (p1.Lname > p2.Lname) return -1;**

**else if (p1.Fname < p2.Fname) return -1;**

**else if (p1.Fname > p2.Fname) return 1;**

**else return 0;**

**}**

**}**

**Part(b)**

**class Mapper {**

**method map(DocID id, Doc d) {**

**for each record do {**

**fname <- getFname();**

**lname <- getLname();**

**time <- getTime();**

**Pair p = new Pair < Lname, Fname>;**

**ValuePair t = new ValuePair <time, time>;**

**Emit(p, t);**

**}**

**}**

**class Combiner {**

**method combine(Pair p, ValuePair [ t1, t2, t3, …]) {**

**ValuePair temp = t1;**

**for each t in [t2, t3, …] do {**

**if (temp.value1 < t.value1) temp.value1 = t.value1;**

**if (temp.value2 > t.value2) temp.value2 = t.value2;**

**}**

**Emit( Pair p, ValuePair temp );**

**}**

**}**

**class Reducer {**

**method reduce(Pair p, ValuePair [ t1, t2, t3, …]) {**

**ValuePair temp = t1;**

**for each t in [t2, t3, …] do {**

**if (temp.value1 < t.value1) temp.value1 = t.value1;**

**if (temp.value2 > t.value2) temp.value2 = t.value2;**

**}**

**Emit( Pair p, ValuePair temp );**

**}**

**}**

**Part(c)**

**class Mapper{**

**AssociatedArray H;**

**method initialize () {**

**H = new AssociatedArray();**

**}**

**method map(DocID id, Doc d) {**

**for each record do {**

**fname <- getFname();**

**lname <- getLname();**

**time <- getTime();**

**Pair p = new Pair < Lname, Fname>;**

**if(H{p} == null)**

**H.add( p, new ValuePair <time, time> );**

**if (H{p}.value1 < time) H{p}.value1 = time;**

**else (H{p}.value2 > time) H{p}.value2 = time;**

**}**

**method close(){**

**for each Key p in H do**

**Emit( p, H{p});**

**}**

**}**

**class Reducer {**

**method reduce(Pair p, ValuePair [ t1, t2, t3, …]) {**

**ValuePair temp = t1;**

**for each t in [t2, t3, …] do {**

**if (temp.value1 < t.value1) temp.value1 = t.value1;**

**if (temp.value2 > t.value2) temp.value2 = t.value2;**

**}**

**Emit( Pair p, ValuePair temp);**

**}**

**}**

**(a) – No combiner, no in-mapper combiner**

|  |  |
| --- | --- |
| **Machine 1** | **Machine 2** |
| **Mapper 1 – Input Split 1** | **Mapper 2 – Input Split 3** |
| **(<Dove, Mark>, 38)**  **(<Dove, Chris>, 27)**  **(<Dove, Mark>, 16)** | **(<Day, Adam>, 42)**  **(<Dove, Chris>, 36)**  **(<Cox, Mark>, 29)** |
| **Mapper 1 – Input Split 2** | **Mapper 2 – Input Split 4** |
| **(<Cox, Joy>, 30)**  **(<Dove, Chris>, 41)**  **(<Cox, Mark>, 51)** | **(<Cox, Joy>, 88)**  **(<Dove, Chris>, 100)**  **(<Day, Joy>, 67)** |
| **Reducer 1** | **Reducer 2** |
| **(<Cox, Joy>, [30, 88])**  **(<Cox, Mark>, [29, 51])** | **(<Dove, Chris>, [27, 41, 36, 100])**  **(<Dove, Mark>, [38, 16])**  **(<Day, Adam>, [42])**  **(<Day, Joy>, [67])** |

**(b) - With combiner, no in-mapper combiner (assuming that the combiner will work all the time)**

|  |  |
| --- | --- |
| **Machine 1** | **Machine 2** |
| **Mapper 1 – Input Split 1** | **Mapper 2 – Input Split 3** |
| **(<Dove, Mark>, <38, 38>)**  **(<Dove, Chris>, <27, 27>)**  **(<Dove, Mark>, <16, 16>)** | **(<Day, Adam>, <42, 42>)**  **(<Dove, Chris>, <36, 36>)**  **(<Cox, Mark>, <29, 29>)** |
| **Combiner 1- Input Split 1** | **Combiner 2 – Input Split 3** |
| **(<Dove, Mark>, <38, 16>)**  **(<Dove, Chris>, <27, 27>)** | **(<Day, Adam>, <42, 42>)**  **(<Dove, Chris>, <36, 36>)**  **(<Cox, Mark>, <29, 29>)** |
| **Mapper 1 – Input Split 2** | **Mapper 2 – Input Split 4** |
| **(<Cox, Joy>, <30, 30>)**  **(<Dove, Chris>, <41, 41>)**  **(<Cox, Mark>, <51, 51>)** | **(<Cox, Joy>, <88, 88>)**  **(<Dove, Chris>, <100, 100>)**  **(<Day, Joy>, <67, 67>)** |
| **Combiner 1 – Input Split 2** | **Combiner 2 – Input Split 4** |
| **(<Cox, Joy>, <30, 30>)**  **(<Dove, Chris>, <41, 41>)**  **(<Cox, Mark>, <51, 51>)** | **(<Cox, Joy>, <88, 88>)**  **(<Dove, Chris>, <100, 100>)**  **(<Day, Joy>, <67, 67>)** |
| **Reducer 1** | **Reducer 2** |
| **(<Cox, Joy>, [<30, 30>, <88, 88>])**  **(<Cox, Mark>, [<29, 29>, <51, 51>])** | **(<Dove, Chris>, [<27, 27>, <41, 41>,**  **<36, 36>, <100,100>])**  **(<Dove, Mark>, [<38, 16>])**  **(<Day, Adam>, [<42, 42>])**  **(<Day, Joy>, [<67, 67>])** |

# 选择

**1.In a 550 server cluster, built from reliable machines with MTBF of 350 days the failure error rate is :**

**a.157% b.1.57% c 15.7% d.636% e.0.636% f.63.6%**

**2.select all that applies to the Stripes technique .**

**a Stripes produces less intermediate keys than Pairs.**

**b.It must be used in combination with combiners.**

**c the value in the key value pair is more complex.**

**d.Ad in -mapper cannot be applied.**

**e Complex memory management, since the associative array may grow too large.**

**f. Experimental results prove Stripes is faster tan Pairs.**

**1.In a 1600server cluster, built from reliable machines with MTBF of 400 days the failure error rate is :**

**a.400% b.40% c 4% d.0.25% e.2.5% f.25%**

**2.Select all the apply to secondary sorting**

**a.It exists in Hadoop.**

**b.It exists in Google.**

**c.Only possible if there is a secondary nameNode.**

**d.Sort the emits based on key then value if the keys are equal.**

**e.Sorting is done inside the mapper method.**

**f.Sorting is done by the framework**

# True Or False

1.one of the three V’s in in Big Data is Volume,and it means how big the data files are.

True,because Big Data is huge in its volume ,It consists of different types of data distributed over the different computer network.The biggest challenge in big data is to find meaningful information from large data set therefore, volume of big data is enormous.

2.The map function is executed independently once for each input split.

True, each input split are executed independently in map function and generates intermediate key value pairs and framework will take care of it.

3.Even though using an in-mapper combiner reduces network traffic, it does have drawbacks.

True, in-mapper combiner use local aggregation that generates less number of key-value pairs and the framework will have less task of doing sort and shuffle which reduces network traffic but it has

Drawbacks of memory overflow

1.one of the three V’s in in Big Data is Volume,and it means how many computers process that data.

2.A Hadoop system is designed to deal with failure not prevent it.

3.Local aggregation means to combine a cluster of nodes on the same rack and only have them work on solving a problem in big data in isolation of other nodes in the system

# **表格**

Answer three out of four

The table below shows SCI principles and big data topics or techniques . Fill in the blank under each section , where you either provide the missing SCI point or big data point . If you provide a point that is not from the lecture you must provide reasonable connection to get the full credit . If you use a point from class or main points you do not have to provide the connection.

By stating the analogy instead of the SCI principle you get partial credit .

|  |  |  |
| --- | --- | --- |
| SCI point | Big Data Point | Connection |
|  | Mapper |  |
| All information in nature is ultimately in the Unified Field. |  |  |
|  | All associative arrays with the same key will be brought together in the reduce phase |  |
| Unification of the Self with itself |  |  |

|  |  |  |
| --- | --- | --- |
| SCI point | Big Data Point | Connection |
|  | HDFS |  |
| The subjective means of gaining knowledge starts with the wholeness of the non-changing Absolute, which is the basis of the changing relative |  |  |
|  | Mapper |  |
| Tm promotes the ability to see the wholeness as will as the point value,the larger picture as well as the details |  |  |

1.solve the problem below using mapreduce ( 30 points )

One of the ways to rank professors is to find out how many publications their name appears on. A professor gains points when their name appears in the publication, when the publication is cited , and the type of conference at which the paper is published . the table below shows the points for each option,

Write a MapReduce program that can go over magnitudes of publications and order professors based on the score of each one.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1stAuthor | 2ndAuthor | 3rdAuthor | 4thAuthor | 5thAuthor | #of citations | A level conference | B level conference | C level conference |
| 5pts | 4pts | 3pts | 2pts | 1point | Point for each citation | 10points | 7points | 5points |

Each data file consists of many records. A sample of record is as follows:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Publication ID | Date | 1stAuthor | 2ndAuthor | 3rdAuthor | 4thAuthor | 5thAuthor | Conference Type | Citations |
| 54867 | 5/2003 | Jack S. | Jim K. | Jill L. | None | None | A | 73 |
| 97634 | 7/2009 | Jane I. | Jill l. | Joe Y. | Jim s. | Jane F. | C | 15 |

Write all the methods needed to solve this problem using MapReduce.

1. Solve the problem below using map and fold ( 30 points )

Due to the vast number of credit card customers and the high volume of credit card transactions identifying credit card fraud is a very hard task . Write a program that can determine if a customer is a victim of some credit card theft . Given a customer id identify if the customer may have had his her credit card compromised , based on the following rules.

Conditions for an account to be compromised

1-if a customer has more than 50 transactions a day

2-if a customer has transactions in two states during the same day .

3-if a customer has 3 cash back requests and a transaction more than s100

4-if a customer has 5 cash back requests

Exceptions ( not compromised account is safe )

1-if a customer has transactions in two states during the same day , but one transaction is a transportation transaction

1. - if a customer has 5 cash back requests and then a balance check .

Sample entries in the system

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| CustomerID | Date | Amount | State | Transaction type | Type | Card type |
| 24587 | 5 / 5 / 2005 | 12.45 | IA | Credit | Restaurant | VISA |
| 54867 | 8 / 8 / 2008 | 4.76 | WA | Credit | Transportation | Mastercard |
| 35786 | 4 / 7 / 2015 | 15.75 | IL | Credit | Grocery | VISA |
| 35786 | 4 / 7 / 2015 | 20 | IL | Cashback | --- | Discover |
| 24578 | 8 / 7 / 2014 | --- | NY | Statement | --- | AmericanExpress |

# 伪代码

Solve the problem below using map and fold.

A company wishes to hire some students. It manages to collect the transcripts of .. graduating from university for the last 35 years. Last year1.8 million students go ... degree from the US alone. The company is interested in hiring a low experienced ...the following credentials:

a- Graduated no more than three years ago.

b- Took a Big Data course(CS522) and got A or A-.

c- Or Took Algorithms(CS435) and got A or A- and Parallel Programming....B or higher.

d- Sort the candidates according to GPA.

Write the f and g functions, and initial value. Your program should return the

Int f(int x){

If(years <=3){

If(course==”big data” && (grade == “A” || grade == “A-”)){

Return 1;

}

If( (course == “Algorithms” && grade == “A” || grade == “A-” ) || (course == “Parallel Programming” && grade >= “B” ) ){

Return 2

}

}

1.Explain the reason behind the assumption “Failures are common”, What steps are taken in HDFS to overcome this problem?

2.Explain what is meant by the statement “Instead of moving data program goes where data is”,In particular, explain how a MapReduce program runs in Hadoop.

3.A MapReduce program consists of five parts. They are\_\_\_\_\_\_\_\_\_.

4.A datanode in HDFS is similar to \_\_\_\_\_\_\_\_ in GFS? A namenode in HDFS is similar to \_\_\_\_\_\_\_in GFS?

5.What is the mechanism used in Hadoop to determine the “health” of a datanode?What happens if a datanade “dies”.

6.Define an array to be fine if it contains exactly 5 zeros to least once.Write a map and fold(of functional programming ) algorithm.illustrate your algorithm on following test data set.

Examples:

[7,6,0,0,0,0,0,3,1] //fine [7,6,0,0,0,0,0 ] //fine

[0,0,0,0,0] //fine [7,3,2,-5,1] //NOT fine

[7,6,0,0,0,0,0,3,1,0,0] //fine [0,0,0,0,0,3,0,0,0,0,0] //fine

[3,0,0,0,0,5,0] //NOT fine [0,6,0,0,-4,0] //NOT fine

[0,0,0,0,0,0] //NOT fine

7,Define an array to be perfect if all elements of the array the multiples of the first number.Writhe a map and fold(of functional programming) algorithm to determine whether of not the array if perfect.

Examples:

[3,12,18,9] //perfect

[3,12,0,9] //perfect

[3,12,-18,9] //perfect

[3,-12,18,-9,0] //perfect

[2,20,50,5,44] //NOT perfect, 5 is not a multiple of 2

[4,8,36,2,16] //NOT perfect, 2 is not a multiple of 4