MapReduce

Samara Ribeiro Silva
Instituto Tecnológico de Aeronáutica, Laboratório de Processamento Distribuído (CES-27). Professor Celso Hirata e Professora Juliana de Melo Bezerra, São José dos Campos, São Paulo, 25 de outubro de 2021.

1. Introdução

O MapReduce é um modelo de programação utilizado para processar *data sets* muito extensos. As tarefas são divididas em um conjunto de tarefas independentes. As duas etapas principais são a de mapeamento (Map) e a de redução (Reduce). A fase de mapeamento ocorre após a divisão da entrada em porções e em cada porção da entrada as palavras são separadas e retornam para a próxima etapa um par <*key, value*> onde *key* corresponde a palavra e *value* é inicialmente definido como 1. Já na etapa de redução cada *worker* verifica o conjunto de entrada recebido e caso haja palavras repetidas ele elimina essa repetição e soma uma unidade para cada repetição no *value*, ou seja, a saída da fase de redução contém pares com valores de *key* únicos e com *value* indicando quantas vezes a palavra indicada na *key* aparece no seu conjunto de entrada.

O MapReduce pode ser implementado de maneira sequencial utilizando apenas um processo ou distribuída. Na execução distribuída é necessário a presença de um processo *master*. Esse processo seleciona os *workers* disponíveis e atribui uma tarefa para cada um deles.

Para a realização desse laboratório foi fornecido um conjunto de arquivos contendo o código a ser completado e modificado, conforme mostrado na seção de implementação e resultados.

2. Implementação e Resultados

2.1. Parte 1

2.1.1. Tarefa 1.1 Complete a função mapFunc (no arquivo wordcount.go) para identificar as palavras de cada fatia do arquivo inicial. Para facilitar, toda palavra deve ser convertida para minúsculo. Vamos trabalhar da seguinte forma: registrar cada palavra com valor um. Não precisa ter ordenação alfabética entre as palavras.

```
func mapFunc(input []byte) (result []mapreduce.KeyValue) {
    (...)

    for _, word := range words {
        //COMPLETAR ESSE CÓDIGO
        //Basta colocar em result os itens <word,"1">
        //Lembrando: word em minúsculo!
        word = strings.ToLower(word)
```

```
        result = append(result, mapreduce.KeyValue{Key: word,
            Value: "1"})
    }

    //fmt.Printf("%v\n", result) //Para ajudar nos testes. Precisa da biblioteca fmt (acima comentada)

    return result
}
```

Nessa tarefa a função mapFunc foi completada. Essa função recebe a porção do arquivo de entrada e separa as palavras utilizando a biblioteca strings na variável *words.* O código acima mostra que cada palavra de *word* foi transformada para minúscula e em seguida foi adicionada em *result* (variável de saída) como par <*word, "1"*>.

2.1.2. Tarefa 1.2 Implemente a função reduceFunc (no arquivo wordcount.go) para consolidar a contagem das palavras. Novamente não precisa ter ordenação alfabética entre as palavras.

```
func reduceFunc(input []mapreduce.KeyValue) (result []mapreduce.KeyValue) {
    //COMPLETAR ESSE CÓDIGO!!!

    var mapAux map[string]int = make(map[string]int)
    for _, item := range input {
        _, ok := mapAux[item.Key]
        if ok {
            mapAux[item.Key]++
        } else {
            mapAux[item.Key] = 1
        }

    }
    //fmt.Printf("%v\n", mapAux)

    for key, value := range mapAux {
        result = append(result, mapreduce.KeyValue{Key: key,
            Value: strconv.Itoa(value)})
    }

    //fmt.Printf("%v\n", result) //Para ajudar nos testes. Precisa da biblioteca fmt (acima comentada)

    return result
}
```

O código acima mostra a função reduceFunc onde verifica-se se há repetição no conjunto de entrada e incrementa a variável *value*.

2.1.3. Tarefa 1.3 Rode o programa com o arquivo teste.txt. Use o seguinte comando para rodar:

wordcount.exe -mode sequential -file files/teste.txt -chunksize 100 -reducejobs 2

```
PS C:\Program Files\Go\src\labMapReduce\wordcount> .\wordcount.exe -mode sequential
-file files\teste.txt -chunksize 100 -reducejobs 2
2021/10/25 14:34:30 Running in sequential mode.
2021/10/25 14:34:30 Running RunSequential...
2021/10/25 14:34:30 Fanning in file map\map-0
2021/10/25 14:34:30 Fanning in file map\map-1
2021/10/25 14:34:31 Fanning out file result\result
2021/10/25 14:34:31 Fanning out file result\result-1
```

teste.txt
Teste para ver o correto funcionamento da contagem de palavras. Por exemplo, a palavra teste deve ocorrer apenas tres vezes, sendo que a ultima ocorrencia e esta: teste.

O arquivo teste.txt foi dividido em porções respeitando o tamanho máximo definido em *chunksize*.

map-0	**map-1**
Teste para ver o correto funcionamento da contagem de palavras. Por exemplo, a palavra teste	deve ocorrer apenas tres vezes, sendo que a ultima ocorrencia e esta: teste.

Conforme selecionado em *reducejobs* obteve-se dois reduces. Observe nas tabelas abaixo que a coluna identificada como **result** contém os pares da coluna **reduce-X** considerando as repetições. Note que o resultado condiz com o esperado.

result-0	**reduce-0**	**reduce-0-0**	**reduce-1-0**
{"Key":"a","Value":"2"} {"Key":"e","Value":"1"} {"Key":"teste","Value":"3"} {"Key":"de","Value":"1"} {"Key":"da","Value":"1"} {"Key":"por","Value":"1"} {"Key":"palavra","Value":"1"} {"Key":"sendo","Value":"1"} {"Key":"ocorrencia","Value":"1"} {"Key":"esta","Value":"1"} {"Key":"ver","Value":"1"} {"Key":"o","Value":"1"} {"Key":"vezes","Value":"1"} {"Key":"que","Value":"1"}	{"Key":"teste","Value":"1"} {"Key":"ver","Value":"1"} {"Key":"o","Value":"1"} {"Key":"da","Value":"1"} {"Key":"de","Value":"1"} {"Key":"por","Value":"1"} {"Key":"a","Value":"1"} {"Key":"palavra","Value":"1"} {"Key":"teste","Value":"1"} {"Key":"vezes","Value":"1"} {"Key":"sendo","Value":"1"} {"Key":"que","Value":"1"} {"Key":"a","Value":"1"} {"Key":"ocorrencia","Value":"1"}	{"Key":"teste","Value":"1"} {"Key":"ver","Value":"1"} {"Key":"o","Value":"1"} {"Key":"da","Value":"1"} {"Key":"de","Value":"1"} {"Key":"por","Value":"1"} {"Key":"a","Value":"1"} {"Key":"palavra","Value":"1"} {"Key":"teste","Value":"1"}	{"Key":"vezes","Value":"1"} {"Key":"sendo","Value":"1"} {"Key":"que","Value":"1"} {"Key":"a","Value":"1"} {"Key":"ocorrencia","Value":"1"} {"Key":"e","Value":"1"} {"Key":"esta","Value":"1"} {"Key":"teste","Value":"1"}

	{"Key":"e","Value":"1"} {"Key":"esta","Value":"1"} {"Key":"teste","Value":"1"}		

result-1	reduce-1	reduce-0-1	reduce-1-1
{"Key":"contagem","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"exemplo","Value":"1"} {"Key":"tres","Value":"1"} {"Key":"ultima","Value":"1"} {"Key":"para","Value":"1"} {"Key":"funcionamento","Value":"1"} {"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"} {"Key":"correto","Value":"1"}	{"Key":"para","Value":"1"} {"Key":"correto","Value":"1"} {"Key":"funcionamento","Value":"1"} {"Key":"contagem","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"exemplo","Value":"1"} {"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"} {"Key":"tres","Value":"1"} {"Key":"ultima","Value":"1"}	{"Key":"para","Value":"1"} {"Key":"correto","Value":"1"} {"Key":"funcionamento","Value":"1"} {"Key":"contagem","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"exemplo","Value":"1"}	{"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"} {"Key":"tres","Value":"1"} {"Key":"ultima","Value":"1"}

2.1.4. Tarefa 1.4 Rode o programa com outro arquivo de entrada. Use diferentes valores para chunksize e reducejobs.

```
PS C:\Program Files\Go\src\labMapReduce\wordcount> .\wordcount.exe -mode sequential
-file .\files\pedra.txt -chunksize 50 -reducejobs 3
2021/10/25 16:34:59 Running in sequential mode.
2021/10/25 16:34:59 Running RunSequential...
2021/10/25 16:34:59 Fanning in file map\map-0
2021/10/25 16:34:59 Fanning in file map\map-1
2021/10/25 16:34:59 Fanning in file map\map-2
2021/10/25 16:35:00 Fanning out file result\result-0
2021/10/25 16:35:00 Fanning out file result\result-1
2021/10/25 16:35:00 Fanning out file result\result-2
```

pedra.txt
No meio do caminho tinha uma pedra Tinha uma pedra no meio do caminho Tinha uma pedra No meio do caminho tinha uma pedra

map-0	map-1
No meio do caminho tinha uma pedra Tinha uma	pedra no meio do caminho Tinha uma pedra No

map-2
meio do caminho tinha uma pedra

reduce-0	reduce-0-0	reduce-1-0	reduce-2-0
{"Key":"no","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"} {"Key":"no","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"} {"Key":"no","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"}	{"Key":"no","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"} {"Key":"uma","Value":"1"}	{"Key":"pedra","Value":"1"} {"Key":"no","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"} {"Key":"no","Value":"1"}	{"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"}

reduce-1	reduce-0-1	reduce-1-1	reduce-2-1
{"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"} {"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"} {"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"}	{"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"}	{"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"}	{"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"}

reduce-2	reduce-0-2	reduce-1-2	reduce-2-2
{"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"} {"Key":"tinha","Value":"1"} {"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"} {"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"}	{"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"} {"Key":"tinha","Value":"1"}	{"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"}	{"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"}

result-0	result-1	result-3
{"Key":"uma","Value":"4"} {"Key":"pedra","Value":"4"} {"Key":"no","Value":"3"}	{"Key":"meio","Value":"3"} {"Key":"caminho","Value":"3"}	{"Key":"do","Value":"3"} {"Key":"tinha","Value":"4"}

Nessa etapa utilizou-se o arquivo pedra.txt como entrada *chunksize* $= 50$ e *reducejobs* $= 3$. Note que os resultados estão corretos e conforme o esperado.

2.2. Parte 2

2.2.1. Tarefa 2.1 Complete o código da função handleFailingWorkers (do arquivo master.go).

```go
func (master *Master) handleFailingWorkers() {
    //////////////////////////
    // YOUR CODE GOES HERE //
    //////////////////////////
    for elem := range master.failedWorkerChan {
        master.workersMutex.Lock()
        delete(master.workers, elem.id)
        master.workersMutex.Unlock()
        log.Printf("Removing worker '%v' from master list\n", elem.id)
    }
}
```

Na função handleFailingWorkers verifica-se o canal failedWorkerChan que armazena os *works* que falharam na execução de alguma tarefa e o retira da variável *workers* do processo master possibilitando assim a finalização correta dos *workers*.

Na execução exemplificada abaixo o *worker 0* foi forçado a falhar na sua terceira tarefa e na sequência foi removido da lista.

```
PS C:\Program Files\Go\src\labMapReduce\wordcount> .\wordcount.exe -mode distributed -type
master -file .\files\pg1342.txt -chunksize 102400 -reducejobs 5
2021/10/25 15:34:15 Running in distributed mode.
2021/10/25 15:34:15 NodeType: master
2021/10/25 15:34:15 Reduce Jobs: 5
2021/10/25 15:34:15 Address: localhost
2021/10/25 15:34:15 Port: 5000
2021/10/25 15:34:15 File: .\files\pg1342.txt
2021/10/25 15:34:15 Chunk Size: 102400
2021/10/25 15:34:15 Running Master on localhost:5000
2021/10/25 15:34:15 Scheduling Worker.RunMap operations
2021/10/25 15:34:15 Accepting connections on 127.0.0.1:5000
2021/10/25 15:34:17 Registering worker '0' with hostname 'localhost:50001'
2021/10/25 15:34:17 Running Worker.RunMap (ID: '0' File: 'map\map-0' Worker: '0')
2021/10/25 15:34:18 Running Worker.RunMap (ID: '1' File: 'map\map-1' Worker: '0')
2021/10/25 15:34:18 Running Worker.RunMap (ID: '2' File: 'map\map-2' Worker: '0')
2021/10/25 15:34:19 Operation Worker.RunMap '2' Failed. Error: read tcp
127.0.0.1:50089->127.0.0.1:50001: wsarecv: Foi forçado o cancelamento de uma conexão
existente pelo host remoto.
2021/10/25 15:34:19 Removing worker '0' from master list
2021/10/25 15:34:22 Registering worker '1' with hostname 'localhost:50002'
2021/10/25 15:34:22 Running Worker.RunMap (ID: '3' File: 'map\map-3' Worker: '1')
2021/10/25 15:34:23 Running Worker.RunMap (ID: '4' File: 'map\map-4' Worker: '1')
2021/10/25 15:34:23 Running Worker.RunMap (ID: '5' File: 'map\map-5' Worker: '1')
2021/10/25 15:34:24 Running Worker.RunMap (ID: '6' File: 'map\map-6' Worker: '1')
2021/10/25 15:34:25 Running Worker.RunMap (ID: '2' File: 'map\map-2' Worker: '1')
```

```
2021/10/25 15:34:26 7x Worker.RunMap operations completed
2021/10/25 15:34:28 Scheduling Worker.RunReduce operations
2021/10/25 15:34:28 Running Worker.RunReduce (ID: '0' File: 'reduce\reduce-0' Worker: '1')
2021/10/25 15:34:28 Running Worker.RunReduce (ID: '1' File: 'reduce\reduce-1' Worker: '1')
2021/10/25 15:34:29 Running Worker.RunReduce (ID: '2' File: 'reduce\reduce-2' Worker: '1')
2021/10/25 15:34:29 Running Worker.RunReduce (ID: '3' File: 'reduce\reduce-3' Worker: '1')
2021/10/25 15:34:29 Running Worker.RunReduce (ID: '4' File: 'reduce\reduce-4' Worker: '1')
2021/10/25 15:34:30 5x Worker.RunReduce operations completed
2021/10/25 15:34:30 Closing Remote Workers.
2021/10/25 15:34:30 Done.
```

2.2.2. Tarefa 2.2 Altere o código fornecido de forma que as operações que falhem sejam executadas. Uma solução simples utiliza canais de forma eficiente para comunicar operações que falharam entre goroutines.

```go
//master.go
type Master struct {
    (...)
    // Fault Tolerance
    failedOperationChan chan *Operation
    wg                  sync.WaitGroup
}
```

Foram adicionadas duas variáveis na struct Master: o canal failedOperationCha (armazenará as operações que falharam para possibilitar sua execução posteriormente) e wg.

```go
//master_scheduler.go
func (master *Master) runOperation(remoteWorker *RemoteWorker, operation *Operation, wg *sync.WaitGroup)
{
    (...)
    if err != nil {
        //wg.Done()
        (...)
        master.failedOperationChan <- operation
    } else {
        (...)
    }
}
```

A função runOperation foi alterada conforme mostrado acima adicionando a operação falha no canal failedOperationChan e comentando a linha wg.Done() das operações falhas.

```go
//master_scheduler.go
func (master *Master) manageFailOp(wg *sync.WaitGroup) {
    var worker *RemoteWorker
    for op := range master.failedOperationChan {
```

```
        worker = <-master.idleWorkerChan
        go master.runOperation(worker, op, &master.wg)
    }
}
```

```
//master_scheduler.go
func (master *Master) schedule(task *Task, proc string, filePathChan chan string) int {
    (...)
    counter = 0
    for filePath = range filePathChan {
        (...)
    }
    go master.manageFailOp(&master.wg)
    (...)
}
```

Para executar as operações falhas foi criada uma função manageFailOp (optou-se pela criação dessa função para evitar problemas com o fechamento do canal). Essa função é chamada como uma go routine.

2.2.3. Tarefa 2.3 Rode o programa com o arquivo teste.txt. Utilize um worker falho e outro normal. Emule falha numa operação map.

Observe que mesmo o *worker 0* falhando na sua terceira tarefa `Worker.RunMap '2'` os resultados obtidos nas tabelas abaixo foram corretos e de acordo com o esperado.

```
PS C:\Program Files\Go\src\labMapReduce\wordcount> .\wordcount.exe -mode distributed -type
master -file files/teste.txt -chunksize 80 -reducejobs 3
2021/10/25 17:51:27 Running in distributed mode.
2021/10/25 17:51:27 NodeType: master
2021/10/25 17:51:27 Reduce Jobs: 3
2021/10/25 17:51:27 Address: localhost
2021/10/25 17:51:27 Port: 5000
2021/10/25 17:51:27 File: files/teste.txt
2021/10/25 17:51:27 Chunk Size: 80
2021/10/25 17:51:27 Running Master on localhost:5000
2021/10/25 17:51:27 Scheduling Worker.RunMap operations
2021/10/25 17:51:27 Accepting connections on 127.0.0.1:5000
2021/10/25 17:51:41 Registering worker '0' with hostname 'localhost:50002'
2021/10/25 17:51:41 Running Worker.RunMap (ID: '0' File: 'map\map-0' Worker: '0')
2021/10/25 17:51:41 Running Worker.RunMap (ID: '1' File: 'map\map-1' Worker: '0')
2021/10/25 17:51:42 Running Worker.RunMap (ID: '2' File: 'map\map-2' Worker: '0')
2021/10/25 17:51:43 Operation Worker.RunMap '2' Failed. Error: read tcp
127.0.0.1:64129->127.0.0.1:50002: wsarecv: Foi forçado o cancelamento de uma conexão
existente pelo host remoto.
2021/10/25 17:51:43 Removing worker '0' from master list
```

```
2021/10/25 17:51:48 Registering worker '1' with hostname 'localhost:50003'
2021/10/25 17:51:48 Running Worker.RunMap (ID: '2' File: 'map\map-2' Worker: '1')
2021/10/25 17:51:48 3x Worker.RunMap operations completed
2021/10/25 17:51:48 Scheduling Worker.RunReduce operations
2021/10/25 17:51:48 Running Worker.RunReduce (ID: '0' File: 'reduce\reduce-0' Worker: '1')
2021/10/25 17:51:49 Running Worker.RunReduce (ID: '1' File: 'reduce\reduce-1' Worker: '1')
2021/10/25 17:51:49 Running Worker.RunReduce (ID: '2' File: 'reduce\reduce-2' Worker: '1')
2021/10/25 17:51:49 3x Worker.RunReduce operations completed
2021/10/25 17:51:49 Closing Remote Workers.
2021/10/25 17:51:50 Done.
```

map-0	**map-1**
Teste para ver o correto funcionamento da contagem de palavras. Por exemplo,	a palavra teste deve ocorrer apenas tres vezes, sendo que a ultima ocorrencia e

map-2
esta: teste.

reduce-0	**reduce-0-0**	**reduce-1-0**	**reduce-2-0**
{"Key":"correto","Value":"1"} {"Key":"da","Value":"1"} {"Key":"palavra","Value":"1"} {"Key":"sendo","Value":"1"} {"Key":"ultima","Value":"1"} {"Key":"ocorrencia","Value":"1"} {"Key":"esta","Value":"1"}	{"Key":"correto","Value":"1"} {"Key":"da","Value":"1"}	{"Key":"palavra","Value":"1"} {"Key":"sendo","Value":"1"} {"Key":"ultima","Value":"1"} {"Key":"ocorrencia","Value":"1"}	{"Key":"esta","Value":"1"}

reduce-1	**reduce-0-1**	**reduce-1-1**	**reduce-2-1**
{"Key":"teste","Value":"1"} {"Key":"para","Value":"1"} {"Key":"ver","Value":"1"} {"Key":"o","Value":"1"} {"Key":"contagem","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"por","Value":"1"} {"Key":"exemplo","Value":"1"} {"Key":"a","Value":"1"} {"Key":"teste","Value":"1"} {"Key":"tres","Value":"1"} {"Key":"vezes","Value":"1"} {"Key":"a","Value":"1"} {"Key":"teste","Value":"1"}	{"Key":"teste","Value":"1"} {"Key":"para","Value":"1"} {"Key":"ver","Value":"1"} {"Key":"o","Value":"1"} {"Key":"contagem","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"por","Value":"1"} {"Key":"exemplo","Value":"1"}	{"Key":"a","Value":"1"} {"Key":"teste","Value":"1"} {"Key":"tres","Value":"1"} {"Key":"vezes","Value":"1"} {"Key":"a","Value":"1"}	{"Key":"teste","Value":"1"}

reduce-2	**reduce-0-2**	**reduce-1-2**	**reduce-2-2**

{"Key":"funcionamento","Valu e":"1"} {"Key":"de","Value":"1"} {"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"} {"Key":"que","Value":"1"} {"Key":"e","Value":"1"}	{"Key":"funcionamento","Valu e":"1"} {"Key":"de","Value":"1"}	{"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"} {"Key":"que","Value":"1"} {"Key":"e","Value":"1"}	

result-final	**result-0**	**result-1**	**result-2**
{"Key":"correto","Value":"1"} {"Key":"da","Value":"1"} {"Key":"palavra","Value":"1"} {"Key":"sendo","Value":"1"} {"Key":"ultima","Value":"1"} {"Key":"ocorrencia","Value":"1 "} {"Key":"esta","Value":"1"} {"Key":"teste","Value":"3"} {"Key":"para","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"por","Value":"1"} {"Key":"ver","Value":"1"} {"Key":"o","Value":"1"} {"Key":"contagem","Value":"1" } {"Key":"exemplo","Value":"1"} {"Key":"a","Value":"2"} {"Key":"tres","Value":"1"} {"Key":"vezes","Value":"1"} {"Key":"que","Value":"1"} {"Key":"e","Value":"1"} {"Key":"funcionamento","Valu e":"1"} {"Key":"de","Value":"1"} {"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"}	{"Key":"correto","Value":"1"} {"Key":"da","Value":"1"} {"Key":"palavra","Value":"1"} {"Key":"sendo","Value":"1"} {"Key":"ultima","Value":"1"} {"Key":"ocorrencia","Value":"1 "} {"Key":"esta","Value":"1"}	{"Key":"teste","Value":"3"} {"Key":"para","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"por","Value":"1"} {"Key":"ver","Value":"1"} {"Key":"o","Value":"1"} {"Key":"contagem","Value":"1" } {"Key":"exemplo","Value":"1"} {"Key":"a","Value":"2"} {"Key":"tres","Value":"1"} {"Key":"vezes","Value":"1"}	{"Key":"que","Value":"1"} {"Key":"e","Value":"1"} {"Key":"funcionamento","Valu e":"1"} {"Key":"de","Value":"1"} {"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"}

2.2.4. Tarefa 2.4 Rode o programa com o arquivo teste.txt. Utilize um worker falho e outro normal. Emule falha numa operação reduce. Obs: Aqui tem que testar diferentes casos para achar um que falha no reduce.

Observe que mesmo o *worker 0* falhando na sua terceira tarefa `Worker.RunReduce '0'` os resultados obtidos nas tabelas abaixo foram corretos e de acordo com o esperado.

```
PS C:\Program Files\Go\src\labMapReduce\wordcount> .\wordcount.exe -mode distributed -type
master -file files/teste.txt -chunksize 100 -reducejobs 3
2021/10/25 18:00:27 Running in distributed mode.
2021/10/25 18:00:27 NodeType: master
2021/10/25 18:00:27 Reduce Jobs: 3
2021/10/25 18:00:27 Address: localhost
2021/10/25 18:00:27 Port: 5000
2021/10/25 18:00:27 File: files/teste.txt
```

```
2021/10/25 18:00:27 Chunk Size: 100
2021/10/25 18:00:27 Running Master on localhost:5000
2021/10/25 18:00:27 Scheduling Worker.RunMap operations
2021/10/25 18:00:27 Accepting connections on 127.0.0.1:5000
2021/10/25 18:00:31 Registering worker '0' with hostname 'localhost:50002'
2021/10/25 18:00:31 Running Worker.RunMap (ID: '0' File: 'map\map-0' Worker: '0')
2021/10/25 18:00:31 Running Worker.RunMap (ID: '1' File: 'map\map-1' Worker: '0')
2021/10/25 18:00:32 2x Worker.RunMap operations completed
2021/10/25 18:00:32 Scheduling Worker.RunReduce operations
2021/10/25 18:00:32 Running Worker.RunReduce (ID: '0' File: 'reduce\reduce-0' Worker: '0')
2021/10/25 18:00:32 Operation Worker.RunReduce '0' Failed. Error: read tcp
127.0.0.1:49657->127.0.0.1:50002: wsarecv: Foi forçado o cancelamento de uma conexão
existente pelo host remoto.
2021/10/25 18:00:32 Removing worker '0' from master list
2021/10/25 18:00:34 Registering worker '1' with hostname 'localhost:50003'
2021/10/25 18:00:34 Running Worker.RunReduce (ID: '1' File: 'reduce\reduce-1' Worker: '1')
2021/10/25 18:00:34 Running Worker.RunReduce (ID: '0' File: 'reduce\reduce-0' Worker: '1')
2021/10/25 18:00:35 Running Worker.RunReduce (ID: '2' File: 'reduce\reduce-2' Worker: '1')
2021/10/25 18:00:35 3x Worker.RunReduce operations completed
2021/10/25 18:00:35 Closing Remote Workers.
2021/10/25 18:00:35 Done.
```

map-0	**map-1**
Teste para ver o correto funcionamento da contagem de palavras. Por exemplo, a palavra teste	deve ocorrer apenas tres vezes, sendo que a ultima ocorrencia e esta: teste.

result-0	**reduce-0**	**reduce-0-0**	**reduce-1-0**
{"Key":"correto","Value":"1"} {"Key":"da","Value":"1"} {"Key":"palavra","Value":"1"} {"Key":"sendo","Value":"1"} {"Key":"ultima","Value":"1"} {"Key":"ocorrencia","Value":"1"} {"Key":"esta","Value":"1"}	{"Key":"correto","Value":"1"} {"Key":"da","Value":"1"} {"Key":"palavra","Value":"1"} {"Key":"sendo","Value":"1"} {"Key":"ultima","Value":"1"} {"Key":"ocorrencia","Value":"1"} {"Key":"esta","Value":"1"}	{"Key":"correto","Value":"1"} {"Key":"da","Value":"1"} {"Key":"palavra","Value":"1"}	{"Key":"sendo","Value":"1"} {"Key":"ultima","Value":"1"} {"Key":"ocorrencia","Value":"1"} {"Key":"esta","Value":"1"}

result-1	**reduce-1**	**reduce-0-1**	**reduce-1-1**
{"Key":"o","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"por","Value":"1"} {"Key":"exemplo","Value":"1"} {"Key":"vezes","Value":"1"} {"Key":"teste","Value":"3"} {"Key":"para","Value":"1"} {"Key":"a","Value":"2"} {"Key":"tres","Value":"1"}	{"Key":"teste","Value":"1"} {"Key":"para","Value":"1"} {"Key":"ver","Value":"1"} {"Key":"o","Value":"1"} {"Key":"contagem","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"por","Value":"1"} {"Key":"exemplo","Value":"1"}	{"Key":"teste","Value":"1"} {"Key":"para","Value":"1"} {"Key":"ver","Value":"1"} {"Key":"o","Value":"1"} {"Key":"contagem","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"por","Value":"1"} {"Key":"exemplo","Value":"1"}	{"Key":"tres","Value":"1"} {"Key":"vezes","Value":"1"} {"Key":"a","Value":"1"} {"Key":"teste","Value":"1"}

{"Key":"ver","Value":"1"} {"Key":"contagem","Value":"1" }	{"Key":"a","Value":"1"} {"Key":"teste","Value":"1"} {"Key":"tres","Value":"1"} {"Key":"vezes","Value":"1"} {"Key":"a","Value":"1"} {"Key":"teste","Value":"1"}	{"Key":"a","Value":"1"} {"Key":"teste","Value":"1"}	

result-2	reduce-2	reduce-0-2	reduce-1-2
{"Key":"que","Value":"1"} {"Key":"e","Value":"1"} {"Key":"funcionamento","Valu e":"1"} {"Key":"de","Value":"1"} {"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"}	{"Key":"funcionamento","Valu e":"1"} {"Key":"de","Value":"1"} {"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"} {"Key":"que","Value":"1"} {"Key":"e","Value":"1"}	{"Key":"funcionamento","Valu e":"1"} {"Key":"de","Value":"1"}	{"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"} {"Key":"que","Value":"1"} {"Key":"e","Value":"1"}

result-final	result-0	result-1	result-2
{"Key":"correto","Value":"1"} {"Key":"da","Value":"1"} {"Key":"palavra","Value":"1"} {"Key":"sendo","Value":"1"} {"Key":"ultima","Value":"1"} {"Key":"ocorrencia","Value":"1 "} {"Key":"esta","Value":"1"} {"Key":"o","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"por","Value":"1"} {"Key":"exemplo","Value":"1"} {"Key":"vezes","Value":"1"} {"Key":"teste","Value":"3"} {"Key":"para","Value":"1"} {"Key":"a","Value":"2"} {"Key":"tres","Value":"1"} {"Key":"ver","Value":"1"} {"Key":"contagem","Value":"1" } {"Key":"que","Value":"1"} {"Key":"e","Value":"1"} {"Key":"funcionamento","Valu e":"1"} {"Key":"de","Value":"1"} {"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"}	{"Key":"correto","Value":"1"} {"Key":"da","Value":"1"} {"Key":"palavra","Value":"1"} {"Key":"sendo","Value":"1"} {"Key":"ultima","Value":"1"} {"Key":"ocorrencia","Value":"1 "} {"Key":"esta","Value":"1"}	{"Key":"o","Value":"1"} {"Key":"palavras","Value":"1"} {"Key":"por","Value":"1"} {"Key":"exemplo","Value":"1"} {"Key":"vezes","Value":"1"} {"Key":"teste","Value":"3"} {"Key":"para","Value":"1"} {"Key":"a","Value":"2"} {"Key":"tres","Value":"1"} {"Key":"ver","Value":"1"} {"Key":"contagem","Value":"1" }	{"Key":"que","Value":"1"} {"Key":"e","Value":"1"} {"Key":"funcionamento","Valu e":"1"} {"Key":"de","Value":"1"} {"Key":"deve","Value":"1"} {"Key":"ocorrer","Value":"1"} {"Key":"apenas","Value":"1"}

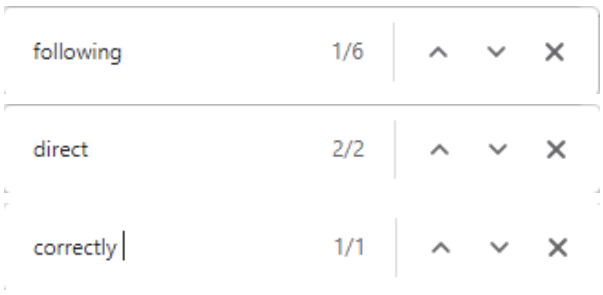
2.2.5. Tarefa 2.5 Rode o programa com outro arquivo de entrada. Use diferentes valores para chunksize e reducejobs. Use um master e dois (ou mais) workers falhos.

```
PS C:\Program Files\Go\src\labMapReduce\wordcount> .\wordcount.exe -mode distributed -type
master -file .\files\distributed_computing.txt -chunksize 204800 -reducejobs 6
2021/10/25 16:28:44 Running in distributed mode.
```

```
2021/10/25 16:28:44 NodeType: master
2021/10/25 16:28:44 Reduce Jobs: 6
2021/10/25 16:28:44 Address: localhost
2021/10/25 16:28:44 Port: 5000
2021/10/25 16:28:44 File: .\files\distributed_computing.txt
2021/10/25 16:28:44 Chunk Size: 204800
2021/10/25 16:28:44 Running Master on localhost:5000
2021/10/25 16:28:44 Scheduling Worker.RunMap operations
2021/10/25 16:28:44 Accepting connections on 127.0.0.1:5000
2021/10/25 16:28:54 Registering worker '0' with hostname 'localhost:50002'
2021/10/25 16:28:54 Running Worker.RunMap (ID: '0' File: 'map\map-0' Worker: '0')
2021/10/25 16:28:55 1x Worker.RunMap operations completed
2021/10/25 16:28:55 Registering worker '1' with hostname 'localhost:50001'
2021/10/25 16:28:56 Scheduling Worker.RunReduce operations
2021/10/25 16:28:56 Running Worker.RunReduce (ID: '0' File: 'reduce\reduce-0' Worker: '0')
2021/10/25 16:28:56 Running Worker.RunReduce (ID: '1' File: 'reduce\reduce-1' Worker: '1')
2021/10/25 16:28:56 Registering worker '2' with hostname 'localhost:50003'
2021/10/25 16:28:56 Running Worker.RunReduce (ID: '2' File: 'reduce\reduce-2' Worker: '2')
2021/10/25 16:28:56 Running Worker.RunReduce (ID: '3' File: 'reduce\reduce-3' Worker: '1')
2021/10/25 16:28:56 Operation Worker.RunReduce '0' Failed. Error: read tcp
127.0.0.1:58379->127.0.0.1:50002: wsarecv: Foi forçado o cancelamento de uma conexão
existente pelo host remoto.
2021/10/25 16:28:56 Removing worker '0' from master list
2021/10/25 16:28:56 Running Worker.RunReduce (ID: '4' File: 'reduce\reduce-4' Worker: '2')
2021/10/25 16:28:56 Running Worker.RunReduce (ID: '0' File: 'reduce\reduce-0' Worker: '1')
2021/10/25 16:28:56 Running Worker.RunReduce (ID: '5' File: 'reduce\reduce-5' Worker: '2')
2021/10/25 16:28:57 Operation Worker.RunReduce '0' Failed. Error: read tcp
127.0.0.1:58388->127.0.0.1:50001: wsarecv: Foi forçado o cancelamento de uma conexão
existente pelo host remoto.
2021/10/25 16:28:57 Removing worker '1' from master list
2021/10/25 16:28:57 Running Worker.RunReduce (ID: '0' File: 'reduce\reduce-0' Worker: '2')
2021/10/25 16:28:57 6x Worker.RunReduce operations completed
2021/10/25 16:28:57 Closing Remote Workers.
2021/10/25 16:28:57 Done.
```

O arquivo distributed_computing.txt é muito grande e portanto optou-se por não colocar os arquivos e resultados no relatório. Para exemplificar o funcionamento selecionou-se algumas palavras.

Palavras selecionadas	**Resultado da busca (Ctrl + F)**
{"Key":"articles","Value":"2"} {"Key":"examples","Value":"6"} {"Key":"following","Value":"6"} {"Key":"direct","Value":"2"} {"Key":"correctly","Value":"1"}	

- Utilizando o arquivo pedra.txt.

```
PS C:\Program Files\Go\src\labMapReduce\wordcount> .\wordcount.exe -mode distributed -type
master -file .\files\pedra.txt -chunksize 80 -reducejobs 3
2021/10/25 17:41:57 Running in distributed mode.
2021/10/25 17:41:57 NodeType: master
2021/10/25 17:41:57 Reduce Jobs: 3
2021/10/25 17:41:57 Address: localhost
2021/10/25 17:41:57 Port: 5000
2021/10/25 17:41:57 File: .\files\pedra.txt
2021/10/25 17:41:57 Chunk Size: 80
2021/10/25 17:41:57 Running Master on localhost:5000
2021/10/25 17:41:57 Scheduling Worker.RunMap operations
2021/10/25 17:41:57 Accepting connections on 127.0.0.1:5000
2021/10/25 17:42:08 Registering worker '0' with hostname 'localhost:50002'
2021/10/25 17:42:08 Running Worker.RunMap (ID: '0' File: 'map\map-0' Worker: '0')
2021/10/25 17:42:09 Running Worker.RunMap (ID: '1' File: 'map\map-1' Worker: '0')
2021/10/25 17:42:09 2x Worker.RunMap operations completed
2021/10/25 17:42:10 Scheduling Worker.RunReduce operations
2021/10/25 17:42:10 Running Worker.RunReduce (ID: '0' File: 'reduce\reduce-0' Worker: '0')
2021/10/25 17:42:10 Operation Worker.RunReduce '0' Failed. Error: read tcp
127.0.0.1:62126->127.0.0.1:50002: wsarecv: Foi forçado o cancelamento de uma conexão
existente pelo host remoto.
2021/10/25 17:42:10 Removing worker '0' from master list
2021/10/25 17:42:14 Registering worker '1' with hostname 'localhost:50001'
2021/10/25 17:42:14 Running Worker.RunReduce (ID: '1' File: 'reduce\reduce-1' Worker: '1')
2021/10/25 17:42:14 Running Worker.RunReduce (ID: '0' File: 'reduce\reduce-0' Worker: '1')
2021/10/25 17:42:15 Operation Worker.RunReduce '0' Failed. Error: read tcp
127.0.0.1:62133->127.0.0.1:50001: wsarecv: Foi forçado o cancelamento de uma conexão
existente pelo host remoto.
2021/10/25 17:42:15 Removing worker '1' from master list
2021/10/25 17:42:17 Registering worker '2' with hostname 'localhost:50003'
2021/10/25 17:42:17 Running Worker.RunReduce (ID: '2' File: 'reduce\reduce-2' Worker: '2')
2021/10/25 17:42:17 Running Worker.RunReduce (ID: '0' File: 'reduce\reduce-0' Worker: '2')
2021/10/25 17:42:18 3x Worker.RunReduce operations completed
2021/10/25 17:42:18 Closing Remote Workers.
2021/10/25 17:42:18 Done.
```

map-0	**map-1**

No meio do caminho tinha uma pedra Tinha uma pedra no meio do caminho Tinha	uma pedra No meio do caminho tinha uma pedra

result-0	**reduce-0**	**reduce-0-0**	**reduce-1-0**
{"Key":"no","Value":"3"} {"Key":"uma","Value":"4"} {"Key":"pedra","Value":"4"}	{"Key":"no","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"} {"Key":"no","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"} {"Key":"no","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"}	{"Key":"no","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"} {"Key":"no","Value":"1"}	{"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"} {"Key":"no","Value":"1"} {"Key":"uma","Value":"1"} {"Key":"pedra","Value":"1"}

result-1	**reduce-1**	**reduce-0-1**	**reduce-1-1**
{"Key":"meio","Value":"3"} {"Key":"caminho","Value":"3"}	{"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"} {"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"} {"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"}	{"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"} {"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"}	{"Key":"meio","Value":"1"} {"Key":"caminho","Value":"1"}

result-2	**reduce-2**	**reduce-0-2**	**reduce-1-2**
{"Key":"do","Value":"3"} {"Key":"tinha","Value":"4"}	{"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"} {"Key":"tinha","Value":"1"} {"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"} {"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"}	{"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"} {"Key":"tinha","Value":"1"} {"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"}	{"Key":"do","Value":"1"} {"Key":"tinha","Value":"1"}

result-final
{"Key":"do","Value":"3"} {"Key":"no","Value":"3"} {"Key":"caminho","Value":"3"} {"Key":"uma","Value":"4"} {"Key":"meio","Value":"3"} {"Key":"tinha","Value":"4"} {"Key":"pedra","Value":"4"}

3. Conclusão

Este trabalho serviu para exercitar e fixar os conhecimentos do modelo de programação MapReduce de forma sequencial e distribuída.

Os resultados obtidos descritos na seção de implementação e resultados foram condizentes com o esperado teoricamente e, portanto, corroboram para a classificação da implementação como adequada do algoritmo proposto.