日志格式：

175.44.30.93 - - [29/Sep/2013:00:10:16 +0800] "GET /structure/heap/ HTTP/1.1" 200 22539 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1;)"

“175.44.30.93”为访问机器的IP， “[29/Sep/2013:00:10:16 +0800]”是访问时间，"GET /structure/heap/ HTTP/1.1"中“/structure/heap/”是访问的网站子目录，即访问的页面为“http://dongxicheng.org/structure/heap/”，其他字段含义可暂时不关注，每一行字段数目是相同且规整的

1）统计（所有日志）独立ip数目，即不同ip的总数

程序：

package com.wordcount.test;

import java.io.IOException;  
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;  
import org.apache.hadoop.fs.Path;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.Job;  
import org.apache.hadoop.mapreduce.Mapper;  
import org.apache.hadoop.mapreduce.Reducer;  
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;  
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;  
import org.apache.hadoop.util.GenericOptionsParser;

public class WordCount {  
    public static enum MY\_COUNTER{DIFFERENT\_IP\_COUNTER}  
    public static class TokenizerMapper extends Mapper<Object,Text,Text,IntWritable>{  
        private final static IntWritable one = new IntWritable(1);  
        private Text ip = new Text();  
        @Override  
        protected void map(Object key, Text value, Mapper<Object, Text, Text, IntWritable>.Context context)  
                throws IOException, InterruptedException {  
            String[] strs = value.toString().split(" ");  
            ip.set(strs[0]);  
            context.write(ip, one);  
        }  
    }  
      
      
    public static class IntSumReducer extends Reducer<Text,IntWritable,Text,IntWritable>{  
        private IntWritable result = new IntWritable();  
        @Override  
        protected void reduce(Text key, Iterable<IntWritable> values,  
                Reducer<Text, IntWritable, Text, IntWritable>.Context context) throws IOException, InterruptedException {  
                
              int sum = 0;  
              for (IntWritable val : values) {  
                sum += val.get();  
              }  
              result.set(sum);  
              context.write(key, result);  
            }  
          
    }  
      
      
      
    public static void main(String[] args) throws Exception {  
        Configuration conf = new Configuration();  
        String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();  
        if(otherArgs.length < 2){  
            System.err.println("Usage:wordcount <in>[<in>...]<out>");  
            System.exit(2);  
        }  
          
        Job job = new Job(conf,"word count");  
        job.setJarByClass(WordCount.class);  
        job.setMapperClass(TokenizerMapper.class);  
        job.setCombinerClass(IntSumReducer.class);  
        job.setReducerClass(IntSumReducer.class);  
        job.setOutputKeyClass(Text.class);  
        job.setOutputValueClass(IntWritable.class);  
          
        for(int i = 0; i < otherArgs.length -1;++i){  
            FileInputFormat.addInputPath(job, new Path(otherArgs[i]));  
        }  
          
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[otherArgs.length-1]));  
        System.exit(job.waitForCompletion(true) ? 0 : 1);  
    }  
      
}

生成的文件为ip 和 出现的次数

例如：

101.226.167.203    4  
101.226.167.204    3  
101.226.167.205    2

对生成的文件再次进行Mapreduce处理，统计ip个数

修改IntSumReducer逻辑

private static IntWritable sum = new IntWritable(0);  
    private static int c = -1;  
    private static Text pv = new Text("pv");  
      
    public static class IntSumReducer extends Reducer<Text,IntWritable,Text,IntWritable>{  
        private IntWritable result = new IntWritable();  
        @Override  
        protected void reduce(Text key, Iterable<IntWritable> values,  
                Reducer<Text, IntWritable, Text, IntWritable>.Context context) throws IOException, InterruptedException {  
            c++;  
            sum.set(c);;  
            context.write(pv, sum);  
              
        }  
    }  
      
输入文件指定为刚生成的part-r-00000

运行得uv数

[root@xfyan output2]# cat part-r-00000  
uv    13349

2）  统计（所有日志）每个子目录访问次数

程序：

package com.wordcount.test;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;  
import org.apache.hadoop.conf.Configured;  
import org.apache.hadoop.fs.Path;  
import org.apache.hadoop.io.IntWritable;  
import org.apache.hadoop.io.Text;  
import org.apache.hadoop.mapreduce.Job;  
import org.apache.hadoop.mapreduce.Mapper;  
import org.apache.hadoop.mapreduce.Reducer;  
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;  
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;  
import org.apache.hadoop.util.GenericOptionsParser;

public class WordCount\_n2 {  
    public static class TokenizerMapper extends Mapper<Object,Text,Text,IntWritable>{  
        private final static IntWritable one = new IntWritable(1);  
        private Text dir = new Text();  
        @Override  
        protected void map(Object key, Text value, Mapper<Object, Text, Text, IntWritable>.Context context)  
                throws IOException, InterruptedException {  
            String[] strs = value.toString().split(" ");  
            dir.set(strs[7]);  
            context.write(dir, one);  
        }  
    }  
      
      
    public static class IntSumReducer extends Reducer<Text,IntWritable,Text,IntWritable>{  
        private IntWritable result = new IntWritable();  
        @Override  
        protected void reduce(Text key, Iterable<IntWritable> values,  
                Reducer<Text, IntWritable, Text, IntWritable>.Context context) throws IOException, InterruptedException {  
            int sum = 0;  
            for(IntWritable val : values){  
                sum+=val.get();  
            }  
            result.set(sum);  
            context.write(key, result);  
        }  
    }  
      
      
      
    public static void main(String[] args) throws Exception {  
        Configuration conf = new Configuration();  
        String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();  
        if(otherArgs.length < 2){  
            System.err.println("Usage:wordcount <in>[<in>...]<out>");  
            System.exit(2);  
        }  
          
        Job job = new Job(conf,"word count");  
        job.setJarByClass(WordCount.class);  
        job.setMapperClass(TokenizerMapper.class);  
        job.setCombinerClass(IntSumReducer.class);  
        job.setReducerClass(IntSumReducer.class);  
        job.setOutputKeyClass(Text.class);  
        job.setOutputValueClass(IntWritable.class);  
          
        for(int i = 0; i < otherArgs.length -1;++i){  
            FileInputFormat.addInputPath(job, new Path(otherArgs[i]));  
        }  
          
        FileOutputFormat.setOutputPath(job, new Path(otherArgs[otherArgs.length-1]));  
        System.exit(job.waitForCompletion(true) ? 0 : 1);  
    }  
      
}