doi:10.6041/j.issn.1000-1298.2016.06.036

基于 CheerIO 的 MEAN Stack 气象数据网络爬虫研究

胡　戎　冯仲科　蒋君志伟

（北京林业大学精准林业北京市重点实验室，北京 100083）

摘要：为全面、即时搜集分散于互联网上游离的气象数据，满足各行业、各领域、各学科科研部门的数据需求，提出使用 Google MEAN Stack 全栈技术开发基于 CheerIO 的高效定向爬虫，充分利用 Node.js 高性能 I/O 的特性，实现气象信息的快速搜集。同时将技术栈与地理信息系统技术、数据可视化技术以及云计算技术相结合，通过地理信息系统的数据存储、查询、自动制图、统计分析等功能对信息进行分析和处理，在阿里云平台上构建了一个能抓取并存储海量数据、提供实时气象数据的应用系统，提供便捷的检索、查询功能，有较强的实用性。本文结合气象数据爬虫的解决方案，对 MEAN Stack 数据爬虫的开发框架、项目架构以及爬虫核心技术（抓取目标策略、网页分析算法、多线程并发运算等）进行了深入分析研究与实现。

关键词：CheerIO；MEAN Stack；定向爬虫；大气气象数据

中图分类号：TP391.4　　**文献标识码**：A　　**文章编号**：1000-1298(2016)06-0275-08

Web Crawler of Atmosphere and Weather Data Based on MEAN Stack with CheerIO

Hu Rong　Feng Zhongke　Jiang Junzhiwei

(*Precision Forestry Key Laboratory of Beijing*, *Beijing Forestry University*, *Beijing* 100083, *China*)

Abstract: To collect the meteorological data dispersed in various industries, fields and disciplines in a comprehensive and real-time way, and meet the needs of scientific research departments for data, an efficient directional crawler was developed based on Google's full-Stack technology called MEAN (MongoDB + Express + AngularJS + Node.js) Stack and an fast flex Javascript Document Object Model module called CheerIO, the functions such as fast-gathering weather information, information analysis and processing by data storage, query, automatic mapping, statistical analysis, forecasting of GIS were realized. An application system deployed on Alicloud server which can real-timely update and forecast meteorological data was created, and it can also provide practical functions of massive data storage, convenient search and query. An efficient and practical web application system was built, which not only provided effective solutions for scattered online data collection but show people date intuitively by using HTML5 data visualizing technology. In actual project, it offered a great number of data support and example to the weather-related fields, such as forestry and preventive medicine. GIS data visualization is a constantly evolving concept, whose borders are expanding fast. At the age of the internet, especially in the globalization of information, the long-term value of data has been gained more and more recognition and affirmation from small companies to national political decision-making. It should be recognized what really it is and how it can help us.

Key words: CheerIO; MEAN Stack; directional crawler; atmosphere and weather data

收稿日期：2016-02-29　修回日期：2016-03-26

基金项目：国家自然科学基金项目(41371001)和北京林业大学青年教师科学研究中长期项目(2015ZCQ-LX-01)

作者简介：胡戎(1990—)，男，博士生，主要从事 3S 技术集成与开发研究，E-mail：353486474@qq.com

通信作者：冯仲科(1962—)，男，教授，博士生导师，主要从事森林计量学和精准林业研究，E-mail：fengzhongke@126.com

引言

目前,在气象研究及管理领域中,地理系统信息大量运用于气象资料管理、农业气候区划、气候状况跟踪等方面[1-2]。但是数据的跨行业利用不充分,相关领域之间的数据交流不及时,一定程度上浪费了信息资源。爬虫是与所有网页直接面对,也是搜索引擎的全部数据源头,决定着整个系统的内容和信息。使用基于 CheerIO 的气象数据网络爬虫可以大幅度提高爬虫抓取天气数据的工作效率和准确率[3-4]。为将各领域公开发布的气象数据充分利用于科研需求单位,本文提出利用 V8 引擎高吞吐 I/O 的优势,使用 Google MEAN (MongoDB + Express + AngularJS + Node. js) Stack 全栈技术开发高效的定向爬虫即时搜集其他领域及学科的相关数据[5-7]。

1 系统设计

1.1 相关技术

(1) CheerIO

CheerIO 是 Node. js 特别为服务端定制的,能够快速灵活地对 JQuery 核心进行实现。它工作于 DOM 模型上,且解析、操作、呈送都很高效。CheerIO 实现了 JQuery 核心的一个子集。CheerIO 删除了从 JQuery 库中与不同浏览器不一致的东西。CheerIO 适用于一些简单的、一致性高的文档对象模型(Document object model, DOM)模型。CheerIO 可以解析几乎所有的超文本标记语言(Hyper text mark-up language, HTML)或可扩展标记语言(Extensible mark-up language, XML)文档。

(2) MEAN Stack

MEAN 是一个 Javascript 平台的现代 Web 开发框架总称,它是 MongoDB + Express + AngularJS + Node. js 4 个框架的第一个字母组合,它与传统基于 PHP 的现代 Web 开发框架(Linux + Apache + Mysql + PHP, LAMP)同样是一种全栈开发技术的简称,MEAN Stack 系统框架如图 1 所示。

(3) ECharts

ECharts 是一个纯 Javascript 的图表库,可以流畅地运行在 PC 和移动设备上,兼容当前绝大部分浏览器,底层依赖轻量级的 Canvas 类库 ZRender,提供直观、生动、可交互、可高度个性化定制的数据可视化图表。它提供了常规的折线图、柱状图、散点图、饼图、K 线图;用于统计的盒形图;用于地理数据可视化的地图、热力图、线图;用于关系数据可视化的关系图、treemap、多维数据可视化的平行坐标;还有用于 BI 的漏斗图、仪表盘;并且支持图与图之间的混搭。

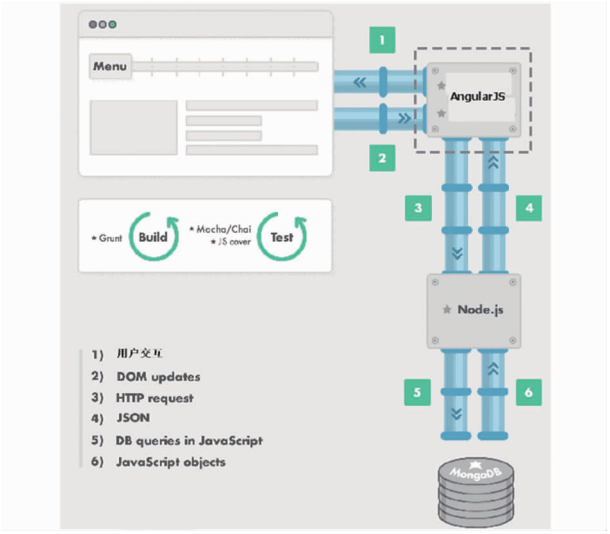

图 1 系统框架

Fig. 1 Architecture of system

1.2 系统结构

本系统使用 MEAN Stack 技术栈旨在开发前后端分离高效友好的气象数据应用网站。采用如图 2 所示的浏览器/服务器模式(Browser/Server, B/S) 3 层结构,分别由前端(浏览器)、后端(服务器)和数据库 3 部分组成。

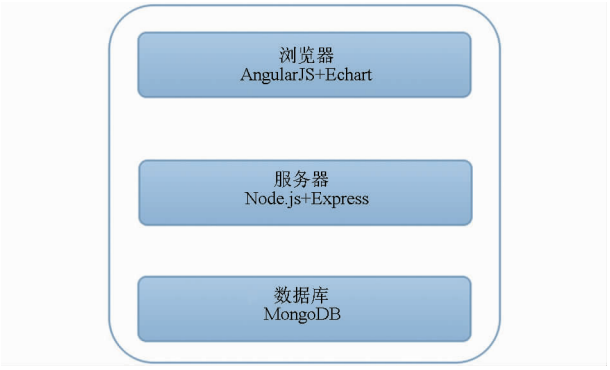

图 2 系统分层结构图

Fig. 2 Layer structure diagram of system

1.2.1 前端部分(浏览器)

前端部分主要使用了以 AngularJS 前端框架为主题的基于 HTML5 的前端技术,使用 Echart3. 0 作为数据可视化的工具实现了包括气象数据的查询,以及生成数据可视化的热力图和 GIS 数据散点图这 3 个主要功能。

前端部分主要分成:①数据集展示模式,通过字段查询,以文字表格的方式将数据呈现给用户,使用 Bootstrap 构建页面。②地图 GIS 展现模式,将数据以热力图的方式直观地展现给用户。

1.2.2 后端部分

后端部分主要使用了基于 ECMAScript6 的

Node.js，并采用 Express 框架实现了无状态请求的软件架构风格（Representational state transfer，RESTful）的气象数据应用程序编程接口（Application programming interface，API）以及作为数据源的基于 CheerIO 的网络爬虫系统，实现了对气象数据时序性的采集、储存、分析和自动制图。

（1）解决的主要问题

在合理的通道中抓取数据，并尽量完整准确地爬取各式维度的数据，对爬虫取得的多维度数据，会结合它的自身特点，运用合理方法来存储数据[8]。一般来说爬虫需要满足以下 4 点要求：①实用性：实现爬取数据的完整准确。②配置性：如多种爬取策略、多种数据获取方式和多种储存格式选取。③灵活性：诸多功能可支持扩展。④通用性：支持多种数据存储格式与数据获取源。

（2）数据的存储策略

在数据中主要包含几种最基本的数据，分别是城市排名数据、城市名称数据以及污染物内容数据。各数据包含字段如表 1、2 所示。在各种平台上大约已有千百亿条天气数据以及海量关系数据，面对如此多的数据，合理存储该数据是重要问题，尤其需要定时备份数据以及建立合适的索引结构以方便对数据的操作[9-10]。

表 1　空气质量数据

Tab.1　Data of air quality

字段名	字段说明	字段类型	字段名	字段说明	字段类型
Rank	排名	String	CO	一氧化碳	String
City	城市名称	String	NO_2	二氧化氮	String
Aqi	Aqi 值	String	O_3	1 h 平均臭氧	String
Ranktype	空气质量指数类别	String	O_3_8 h	8 h 平均臭氧	String
Primary pollution	首要污染物	String	SO_2	二氧化硫	String
PM25	PM2.5 细颗粒物	String	Creat_at	数据创建时间	Date
PM10	PM10 可吸入颗粒物	String			

1.2.3　数据库

数据库部分主要使用了基于 BSON 的 NoSQL—MongoDB。构建了一个支持海量数据存储、分片以及快速查询的数据库系统。

2　系统实现

2.1　前端部分

前端部分使用 AngularJS 框架，采用 Echart 作为数据可视化的工具，具体实现了数据的在线查询与展示功能，AQI 数据热力图和数据散点图，如图 3、4 所示。

表 2　气象数据

Tab.2　Meteorological data

字段名	字段说明	字段类型	字段名	字段说明	字段类型
Id	城市 ID	String	Fl	体感温度（℃）	String
City	城市名称	String	Windspd	风速（km/h）	String
Ion	经度	String	Windsc	风力等级	String
Lat	纬度	String	Winddeg	风向（角度）	String
Tmp	当前温度（℃）	String	Winddir	风向（方向）	String
Cond	天气状况	String	Pres	气压（hPa）	String
Pcpn	降雨量（mm）	String	Vis	能见度（km）	String
Hum	湿度（%）	String	Tim	数据创建时间	Date

图 3　AQI 数据热力图

Fig.3　Thermodynamic chart of AQI data

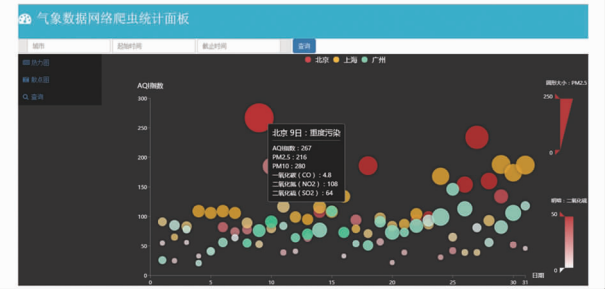

图 4　数据散点图

Fig.4　Scatter plot of data

业务流程：①该模块通过 $http 异步请求后台空气质量数据。②对返回的传入热力图配置选项。③选择需要渲染的 DOM 元素。④渲染生成散点图、热力图。

示例代码如下

```
var dom = document.getElementById("container");
var myChart = echarts.init(dom);
var date = function getpmlist () {
    var deferred = $q.defer(); // 声明延后执行，表示要去监控后面的执行
```

```
    $ http({
      url:"http://xxx:3220/pmlist",
      method:'GET",
      headers: {
        'Content-Type': 'application/json'
      }
    }).
    success(function(data, status, headers, config) {
      deferred.resolve(data);    // 声明执行成功,即
http 请求数据成功,可以返回数据
    }).
    error(function(data, status, headers, config) {
      deferred.reject(data);    // 声明执行失败,即
服务器返回错误
    });
    return deferred.promise;    // 返回承诺,这里并不
是最终数据,而是访问最终数据的 API
} ();
var convertData = function (data) {
    var res = [];
    for (var i = 0; i < data.length; i++) {
        var geoCoord = geoCoordMap[data[i].
name];
        if (geoCoord) {
            res.push({
                name: data[i].name,
                value: geoCoord.concat(data[i].
value)
            });
        }
    }
    return res;
};//数据排序
    option = {…};
myChart.setOption(option, true);
```

2.2 后端部分

后端部分主要实现了网络爬虫定时对天气气象数据以及空气质量数据的抓取、保存以及入库,主要功能流程如图 5 所示。

本系统应业务需要共构建了 2 个业务流分别采集天气气象数据,以及空气质量数据。每天北京时间 12:00 采集 1 次天气气象数据并入库,每小时采集 1 次空气质量数据并入库,且每天生成 1 份 Excel 保存数据,以生成可靠的时序性数据用于科研分析。

(1) Schedule 定时器

本系统使用 Later.js 构建定时器,在每天每个整点的 30 分采集 1 次空气质量数据,每天采集 24 次;每天北京时间 12:00 采集 1 次天气气象数据;每天北京时间 12:00 将当天采集的数据从数据库中取出并生成 Excel。

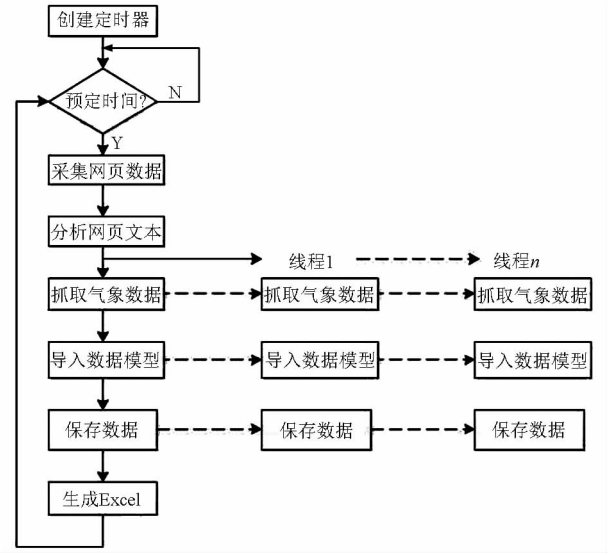

图 5 后端主要功能流程图

Fig. 5 Flowchart of backend main function

业务流程:①同步应用程序与系统时间的格式(如 CST、GMT 等)。②使用 Recur 语法脚本设定定时器时间。设定 3 种时间表:时间表 1:每个整点的 30 分;时间表 2:每天北京时间 12:00(04:00:00GMT);时间表 3:每天北京时间 23:55(15:55:00GMT)。③设定定时器定时执行任务。时间表 1 执行天气质量数据抓取业务流;时间表 2 执行天气气象数据抓取业务流;时间表 3 执行 Excel 生成业务流。

示例代码如下

```
later.date.localTime();//同步系统时间
var sched3 = later.parse.recur().on(23).hour
().on(55).minute();
var task3 = later.setInterval(function() {
asyncWriteExcel.asyncWriteExcel();
}, sched3);时间表 3
```

(2) 流程控制及多线程模块

模块介绍:本系统使用 Async 模块,管理业务流程以及进行多线程性能优化[11]。由于 nodejs 的单线程特性,流程控制需要使用回调(callback)的方式进行,Async 模块封装了回调的流程,简化了文档的结构,本系统使用 Asyncwaterfall 进行流程控制来管理系统的业务流,同时使用 Async map 进行并发操作,对采集的数据分批入库,提高运行的效率[12]。

业务逻辑:①使用瀑布流执行采集网页功能。②从收集到的网页中抓取并分析数据。③进行多线程操作,并发执行整理和保存功能。

示例代码如下

```
async.waterfall([
    function(callback){
      pmservice.getHtml(callback);//采集网页
    },function(data,callback) {
       pmservice.catchdate(data,callback);//分析数据
    },function(data,cb){
    //并发数据整理与保存
    async.map(data,function(item,callback){
      var PMdata = new PMEntity(item);
       DailyPMDao.saveEntity(PMdata, function(data){
                    console.log(data);
                    callback(null,data);
                });
        }, function(err,results) {
            if (err)
                console.log(err);
            cb(null,1);
        });
    },function(callback){
        datapump.writeXlsxPM();
    }
],function(err,results){
});
```

(3) 网页采集模块

模块介绍:使用 http 模块以及 buffer 模块对网页进行采集和保存 http 模块的 get 功能,获取网页的 html 信息,并保存在 buffer 中,以提供文本分析的效率、减少内存的占用[13]。

业务逻辑:①调用 Node.js HTTP 这部分功能获取目标地址 URL 的 HTML 超文本。②通过 BufferHelper 将其生成 Buffe。③将 Buffer 作为参数传入文本分析模块去执行。

示例代码如下

```
Pm25service.prototype.getHtml = function(callback){
    var req = http.get(this.url,function(res){
      var bufferHelper = new BufferHelper();
      res.on('data',function(chunk){
          bufferHelper.concat(chunk);
      });
      res.on('end',function(){
                this.html = iconv.decode(bufferHelper.toBuffer(),'utf8');
                callback(null,this.html);
        });
    });
    req.end();
};
```

(4) 文本分析模块

模块介绍:使用 CheerIO 解析 Buffer 中的 HTML 文本数据,筛选出天气气象和空气质量数据,生成相应实体,具体实体字段如表 3、4 所示。

表 3　空气质量索引

Tab.3　Index of air quality data

字段名	参数数组索引	字段名	参数数组索引
Rank	pmarr[0]	CO	pmarr[7]
City	pmarr[1]	NO_2	pmarr[8]
Aqi	pmarr[2]	O_3	pmarr[9]
Ranktype	pmarr[3]	O_3_8h	pmarr[10]
Primarypollution	pmarr[4]	SO_2	pmarr[11]
PM25	pmarr[5]	Creat_at	pmarr[12]
PM10	pmarr[6]		

表 4　气象数据索引

Tab.4　Index of meteorological data

字段名	参数数组索引	字段名	参数数组索引
Id	pmarr[0]	Fl	pmarr[8]
City	pmarr[1]	Windspd	pmarr[9]
Ion	pmarr[2]	Windsc	pmarr[10]
Lat	pmarr[3]	Winddeg	pmarr[11]
Tmp	pmarr[4]	Winddir	pmarr[12]
Cond	pmarr[5]	Pres	pmarr[13]
Pcpn	pmarr[6]	Vis	pmarr[14]
Hum	pmarr[7]	Tim	pmarr[15]

业务逻辑:①将上一步的采集网络模块生成的 Buffer 数据进行解析。②一一对照索引表,将数据存入数组的对应索引。③把保存结果的数组传给下一步进行多线程操作。

具体实现代码如下

```
Pm25service.prototype.catchdate = function(data,callback){
    var $ = CheerIO.load(data,{decodeEntities:false});
    var results = [];
     $('tbody').children().each(function(i,elem){
        var arr = [];
        $(this).children().each(function(j,el){
            arr.push($(el).text());
        });
```

```
        results.push(arr);
    });
    callback(null,results);
};
```

(5)实体对象和 DAO 数据访问对象

模块介绍:通过编写 Model Entities 数据实体模块对数据进行归一化的整理,生成对应的天气气象数据对象以及空气质量数据对象,以方便相对应数据访问对象的操作。通过编写 Mongoose DAO(data access object)数据访问对象,实现后端应用程序与 MongDB 的数据对接;通过构建通用的数据访问对象,作为具体数据访问对象的基类;提供数据访问接口,让具体数据访问对象实现或继承,主要实现的功能如图 6 所示。

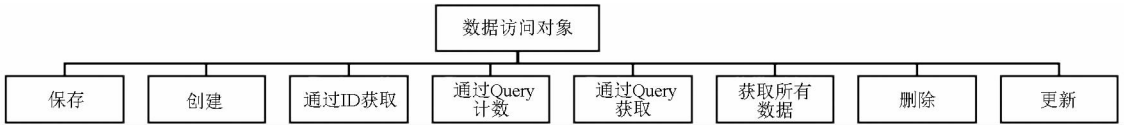

图 6 DAO 主要功能图

Fig. 6 Main function diagram of DAO

业务逻辑:①数据实体模块接受文本分析模块处理的结果,生成每个城市的数据实体对象。②数据访问模块对生成的数据实体对象的字段进行检查,防止错误产生。③通过数据访问模块操纵数据库保存数据。

示例代码如下

```
function DaoBase (Model){
    this.model = Model;
}
//save
DaoBase.prototype.getMongoEntity = function(entity)
{
    var MDBEntities = new this.model(entity);
    return MDBEntities;
};
DaoBase.prototype.saveEntity = function (entity,
callback){
    console.log(entity);
    this.save(this.getMongoEntity(entity),
callback);
};
DaoBase.prototype.save =function(entity,callback){
    entity.save(function(error,doc) {
        if (error) {
            console.log("error :" + error);
        } else {
            //console.log(doc);
            callback(doc);
        }
    });
};
```

(6) DataPump 数据泵

模块介绍:使用 DataPump 数据泵来实现 MongDB 数据的导出,匹配 Excel 工作簿的列名与 MongDB 数据库的字段名,将 Excel 的文件名设置为当天日期。

业务流程:①获取当天日期,并获取当天的年月日以方便文件命名。②选择 Excel 数据的数据源为服务器后台的 Mongodb 的 Collections。③设定数据赛选规则。④设定导出的 Excel 文件名以及工作簿名。⑤确定 Excel 列名,并将数据库的对应字段与列索引匹配。⑥生成 Excel。

示例代码如下

```
module.exports.writeXlsxPM = function(){
var date = new Date();
var now = new Date(date.getFullYear(),date.
getMonth(),date.getDate());
pump
    .mixin(MongodbMixin('mongodb://
username:password@serverip/bjfuweather'))//
    .useCollection('dailypms')
    .from(pump.find({create_at:{$gte:
now}}))//绑定数据源
    .mixin(ExcelWriterMixin())//注入 Excel
写入模块
    .createWorkbook('./resources/tmp/
temppm'+now.getFullYear()+(now.getMonth()+
1)+now.getDate()+'.xlsx')//设定创建文件脚本
    .createWorksheet('AQI')//设定工作簿
.writeHeaders(['rank', 'city','aqi','ranktype','
primarypollution','pm25','pm10','co','no2','o3','
o3_8h','so2','time']) //绑定 Excel 表头
    .process(function(AQI) {
        return pump.writeRow([AQI.rank,
AQI.city, AQI.aqi, AQI.ranktype, AQI.
primarypollution, AQI.pm25, AQI.pm10, AQI.co,
AQI.no2,AQI.o3,AQI.o3_8h,AQI.so2,AQI.create_
at.toString()]);
```

```
    }) //设定 Excel 写入规则
    .logErrorsToConsole()
    .run()
    .then(function() {//结束反馈
        console.log("Done writing contacts to file");
    });
};
```

3　行业应用

3.1　林火防护应用

为了早日实现森林防火工作的科学化、信息化、规范化，系统应充分考虑功能扩容性和技术升级、兼容、扩展性，以适应当代信息技术迅猛发展的要求，求得最佳的性能价格比[14-15]。即时的气象爬虫数据，如风速、温度、湿度、降水量等，能通过层次分析法得出的公式来直观、快速地在森林火灾发生前及时预测火情、发现火情等，从而起到预防的目的[16]。将爬虫收集的天气气象数据计算出的气象因子与植被因子、地形因子与人为因子结合，得出北京市森林火险等级，如图 7 所示。为指挥中心指挥调度提出了宝贵的参考意见，最大限度地降低了火灾发生的可能，从而对火灾进行预先处理，对初发火情，做到及时发现、及时救护，使火灾隐患消亡在萌芽状态。森林火险等级指数为

$$F_{FI} = \sum_{j=1}^{\infty} X_j W_i \qquad (1)$$

式中　F_{FI}——森林火险等级指数

X_j——火险指标

W_i——火险因子权重，见表 5

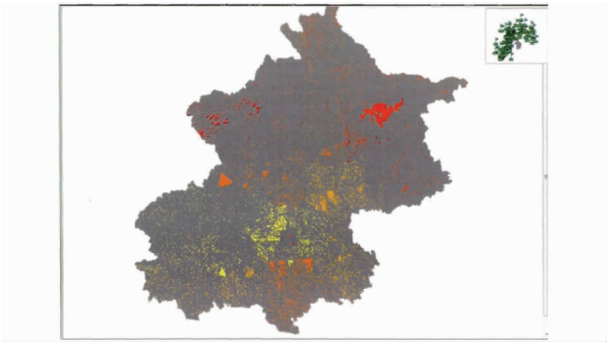

图 7　北京市森林火险等级图

Fig. 7　Forest fire danger rating chart of Beijing

3.2　其他行业应用

3.2.1　城市森林应用

城市森林对 PM2.5 有一定的削减作用。根据 PM2.5 浓度受到林分结构和气象因子的影响，在特定的条件下，林内的 PM2.5 浓度会高于林缘浓度。这种浓度升高的现象和林内相对湿度较高、风速低有关。因此 API 指数随着距离的增大其自身的相关性依然是呈现降低的趋势，但局部区域随着风速等的影响，其相关性又会发生变化[17]。林分内的 PM2.5 浓度与环境因子（浓度、温度、湿度、风速）显著相关，这对于城市雾霾与植被规划有着决定性的影响。

表 5　森林火险因子权重

Tab. 5　Forest fire danger index weights

目标层	准则层	权重	指标层	权重
森林火险等级	气象因子	0.390 9	降雨量	0.120 8
			温度	0.035 6
			湿度	0.056 4
			风力	0.155 2
			风向	0.028 8
	植被因子	0.147 0	植被类型	0.067 3
			树种	0.029 5
			树龄	0.032 7
			郁闭度	0.017 5
	地形因子	0.118 1	海拔	0.023 1
			坡度	0.058 3
			坡向	0.036 7
	人为因子	0.344 0	道路	0.037 2
			居民点	0.105 3
			节日	0.148 9
			历史多发地点	0.052 6

3.2.2　疫情预报预测

平均气压、平均蒸发量和平均降水量与消化道和呼吸道传染病发病率关系密切；温度与虫媒传染病的发病率的关系密切[18]。而这些气象数据都可以通过爬虫技术获取，利用实时更新的数据，专家可以依此判断疫情蔓延的趋势和缓急程度。实现了疫情信息管理，更能利用数学模型进行疫情预测。对未来一段时间内疫病流行趋势预测，为疫情防制部门制定防制决策提供支持。

4　结束语

提出了一种基于 MEAN Stack 的气象爬虫地理信息系统，采用 MEAN Stack 全栈技术，构建利用高效，实用的 Web 应用程序，为搜集网上零散的有效数据提供了更加有效地解决方案。且在实际项目应用中，对气象相关的林业与相关学科提供了大量的数据支持和例证[19-20]。GIS 数据可视化是一个处于不断演变之中的概念，其边界也在不断地扩大[21]。通过网络爬虫不断搜集游离在互联网上的海量数据，将地理信息技术与数据挖掘技术相结合进行大数据分析，从而得到更多知识，推动技术发展。

参 考 文 献

1 吴焕萍. GIS 技术在气象领域中的应用[J]. 气象,2010, 36(3):90 - 100.
WU Huanping. Application of GIS in mteorology[J]. Meteorological Monthly,2010, 36(3):90 - 100. (in Chinese)

2 黄青,周清波,王利民,等. 基于遥感的冬小麦长势等级与气象因子相关性分析[J]. 农业机械学报,2014,45(12):301 - 307.
HUANG Qing, ZHOU Qingbo, WANG Limin, et al. Relationship between winter wheat growth grades obtained from emote-sensing and meteorological factor[J]. Transactions of the Chinese Society for Agricultural Machinery, 2014, 45(12): 301 - 307. (in Chinese)

3 孙懿慧. 基于 GIS 多源数据融合的湖北省中稻增产潜力及影响因子的研究[D]. 武汉:华中农业大学,2015.
SUN Yihui. Yield-increasing potential of middle-season rice in Hubei province based on GIS and multi-source data[D]. Wuhan: Huazhong Agricultural University,2015. (in Chinese)

4 王文生,郭雷风. 关于我国农业大数据中心建设的设想[J]. 大数据,2016(1):28 - 34.
WANG Wensheng, GUO Leifeng. Envisagement of the construction of national agricultural big data center[J]. Big Data Research, 2016(1):28 - 34. (in Chinese)

5 钱继来. 基于 REST 与 RIA 的 Web 应用研究与实现[D]. 武汉:武汉理工大学,2011
QIAN Jilai. Research and realization of Web application based on REST and RIA[D]. Wuhan: Wuhan University of Technology, 2011. (in Chinese)

6 DEAN J, GHEMAWAT S. MapReduce: simplified data processing on large clusters[C]// Proceedings of the 6th Conference on Symposium on Operating Systems Design & Implementation,2004:10.

7 周德懋,李舟军. 高性能网络爬虫:研究综述[J]. 计算机科学,2009, 36(8):26 - 29,53.
ZHOU Demao, LI Zhoujun. Survey of high-performance web crawler[J]. Computer Science, 2009, 36(8): 26 - 29, 53. (in Chinese)

8 罗一纾. 微博爬虫的相关技术研究[D]. 哈尔滨:哈尔滨工业大学,2013.
LUO Yishu. Research on the microblogging crawler related technologies[D]. Harbin: Harbin Institute of Technology, 2013. (in Chinese)

9 SRINIVASAN P, MENCZER F, PANT G. A general evaluation framework for topic crawler[J]. Information Retrieval, 2005, 8(3): 417 - 447.

10 程锦佳. 基于 Hadoop 的分布式爬虫及其实现[D]. 北京:北京邮电大学,2010.
CHENG Jinjia. Research and implementation of distributed web crawler based on Hadoop architecture[D]. Beijing: Beijing University of Posts and Telecommunications,2010. (in Chinese)

11 胡廉民,张泽斌,徐威迪,等. 基于分层结构保留的增量网络爬虫算法[J]. 计算机应用研究,2013,30(8):2381 - 2385.
HU Lianmin, ZHANG Zebin, XU Weidi, et al. Improved crawler algorithm based on hierarchical structure preservation [J]. Application Research of Computers,2013,30(8):2381 - 2385. (in Chinese)

12 徐伟恒,苏志芳,张晴晖,等. 基于物联网架构和 WebGIS 的森林火灾监测系统研究[J]. 安徽农业科学,2012, 40(1):589 - 593.
XU Weiheng, SU Zhifang, ZHANG Qinghui, et al. Research on forest fire monitoring system based on internet of things and WebGIS[J]. Journal of Anhui Agricultural Sciences,2012, 40(1):589 - 593. (in Chinese)

13 李欢. 基于 API 天气数据抓取的特定网络爬虫的研究与实现[D]. 秦皇岛:燕山大学,2014.
LI Huan. Based on the specific web crawler API weather data fetching in the research and implementation[D]. Qinhuangdao: Yanshan University,2014. (in Chinese)

14 何诚,巩垠熙,张思玉,等. 基于 MODIS 数据的森林火险时空分异规律研究[J]. 光谱学与光谱分析,2013, 33(9):2472 - 2477.
HE Cheng, GONG Yinxi, ZHANG Siyu, et al. Forest fire division by using MODIS data based on the temporal-spatial variation law[J]. Spectroscopy and Spectral Analysis,2013,33(9):2472 - 2477. (in Chinese)

15 张冬有,冯仲科,臧淑英. 基于 3S 技术的三维扑火队伍跟踪监控指挥系统研究[J]. 北京林业大学学报,2007,29(增刊 2):103 - 106.
ZHANG Dongyou, FENG Zhongke, ZANG Shuying. Three-dimensional monitoring and directing system responding to forest fire prevention teams based on 3S technology[J]. Journal of Beijing Forestry University,2007,29(Supp. 2):103 - 106. (in Chinese)

16 岳金柱,冯仲科,姜伟. 我国森林火灾应急响应分级与处置相关问题的研究探讨[J]. 森林防火,2008(3):36 - 39.
YUE Jinzhu, FENG Zhongke, JIANG Wei. China's forest fires emergency response classification and disposal of related issues to the research discussion[J]. Forest Fire Prevention,2008(3):36 - 39. (in Chinese)

17 冯仲科,毛海颖,李虹. 环首都圈植被分布与可吸入颗粒物的空间相关性[J]. 农业工程学报,2015, 31(1):220 - 227.
FENG Zhongke, MAO Haiying, Li Hong. Spatial correlation between vegetation distribution and respirable particulate matter around capital region[J]. Transactions of the CSAE,2015, 31(1):220 - 227. (in Chinese)

18 施海龙,曲波,郭海强,等. 干旱地区呼吸道传染病气象因素及发病预测[J]. 中国公共卫生,2006, 22(4):417 - 418.
SHI Hailong, QU Bo, GUO Haiqiang, et al. Effect of meteorological factors on epidemic situation of aspiratory infectious diseases in drought area[J]. Chinese Journal of Public Health,2006, 22(4):417 - 418. (in Chinese)

19 孙忠富,杜克明,郑飞翔,等. 大数据在智慧农业中研究与应用展望[J]. 中国农业科技导报,2013(6):63 - 71.
SUN Zhongfu, DU Keming, ZHENG Feixiang, et al. Perspectives of research and application of big data on smart agriculture[J]. Journal of Agricultural Science and Technology,2013(6):63 - 71. (in Chinese)

20 孙晓勇,刘子玮,孙涛,等. 大数据在农业研究领域中的应用与发展[J]. 中国蔬菜,2015(10):1 - 5.

21 王文生,郭雷风. 农业大数据及其应用展望[J]. 江苏农业科学,2015(9):1 - 5.