NewaSoft

苏州泥娃软件科技有限公司

SUZHOU NIWA SOFTWARE TECHNOLOGY CORPORATION

分布式爬虫服务系统

操作手册

SUZHOU NIWA SOFTWARE TECHNOLOGY CORPORATION

目录

软件使用说明

1 引言

1 .1 编写目的

本手册主要为系统服务提供指南，主要服务于系统的使用和维护人员。

1 .2 项目背景

项目来源于公司产品部门，主要为全文搜索系统提供爬虫服务的支撑。

1 .3 定义

略

1 .4 参考资料

略

2 软件概述

2 .1 目标

实现分布式爬虫的部署、管理，实现网页 url 的获取和分配，实现网页信息的解析，并输入到全文索引系统的过程。

2.2 功能

主要包括：集群管理、爬虫管理。

3 运行环境

3 .1 硬件

列出软件系统运行时所需的硬件最小配置：

a. 计算机型号、主存容量；PC 和服务器，主存 4G；

b. 存储器；大于 500G；

3 .2 支持软件

操作系统名称及版本号；CENTOS 6.5 以及 7 以上

3 .3 网络环境

系统需要组播的支持。注意设置网络防火墙，打开所需的端口。

4 使用说明

4.1 安装和初始化

系统运行在 linux 环境，主要包括 CentOS 系列。

系统运行程序为：Waiter 0.0.0.0 224.0.0.18 8099 32000 ./page。

系统爬虫可以采用第三方服务，也可以使用系统爬虫服务，NewaSoft 爬虫采用 phantomjs 和 casperjs 组成。下载地址如下：

http://phantomjs.org/download.html

http://casperjs.org/

下载解压，设置环境变量，指向解压目录和解压后的 bin 目录即可。

4.2 运行

./Waiter 0.0.0.0 224.0.0.18 8099 32000 ./page

5 运行说明

5 .1 系统介绍

系统包括：分布式爬虫管理服务系统，url 超级链接管理系统，爬虫服务等三部分组成。

5 .1.1 分布式爬虫管理服务系统

主要管理爬虫服务的发现，组建爬虫管理集群，管理爬虫所在的服务器，提供文件和配置的传输管理等。

5 .1.2 url 超级链接管理系统

记录爬虫获取的 url，为爬虫提供可爬取得链接 url，分析 url 和关联网页的特性，并记载，提供爬虫处理网页的方式和方法。

5 .1.3 爬虫服务

爬虫主要分为以下几个部分：

爬虫任务的加载，失败的处理等；

爬虫服务。处理爬取得 url，获取内容并结构化，通过全文引擎的接口导入到全文索引系统。主要包括：网页的访问管理，内容的获取，对于需要密文搜索的内容进行信息的加密处理，收集的信息存入到全文搜索系统。

爬虫任务的调度模块，包括：url 的获取和记录，分配，获取的内容的设置，全文搜索的接口等。

任务特别项的处理：

是否需要登录；

是否需要输入验证码；

是否可以爬取等。

5 .2 分布式爬虫管理系统

系统主界面主要包括：

集群主机的发现，网络拓扑的编制和管理等。

5 .2.1 系统主界面

主要菜单：

收集。主要通过组播发现爬虫主机，点击该菜单，会自动收集主机的信息，并添加到拓扑图中；

保存。保存编制好的拓扑结构图；

下载。提供拓扑结构文件的下载；

编辑。自定义拓扑的形式，定义爬虫集群。

5 .2.2 爬虫主机管理页面

点击主界面的节点，选取集群或者单独的主机进行管理。

主要的菜单包括：

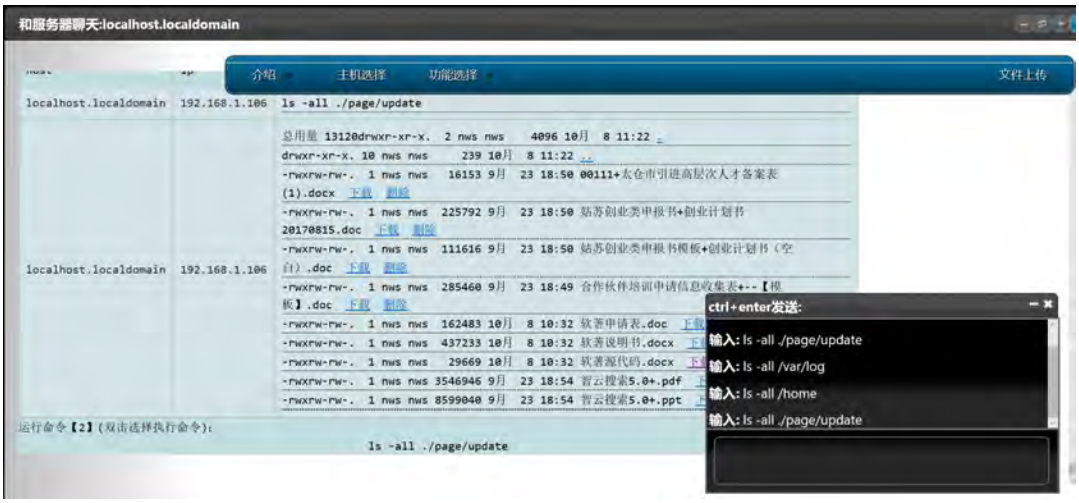
主机选择。可以再次选择管理的主机或者集群。

功能选择。选择管理具体功能。

主要分为：

目录浏览

提供用户目录、系统日志和上传文件目录的浏览。示例：查看上传文件目录。



服务查询

提供内存、磁盘、进程和时间的查询。

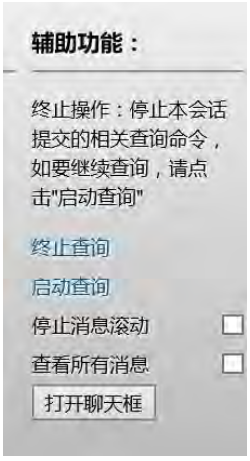
爬虫服务

主要管理爬虫，提供爬虫的启动和停止。

点击“启动爬虫”，点击“开始爬取”即可实现爬虫服务。例如：

点击“开始爬取”：

辅助功能



主要包括：

终止查询。暂时停止该主机的相关查询服务；

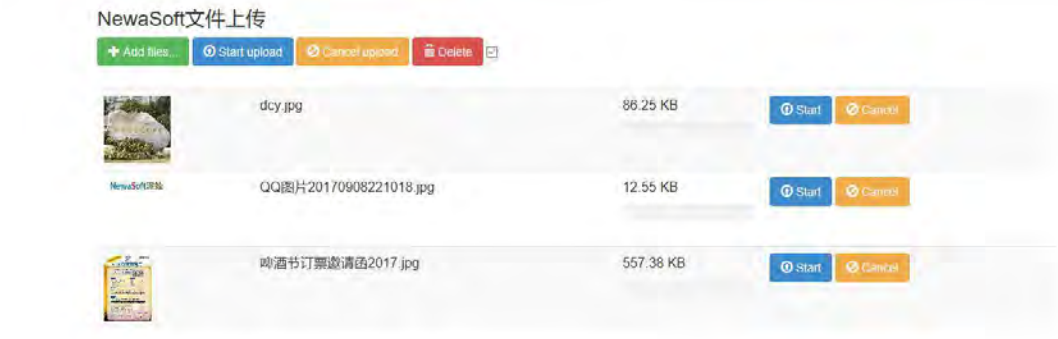
启动查询。启动被停止的查询服务；

停止消息滚动。选中，网页的消息不滚动，未选中，消息继续加载；

查看所有消息。查看集群内所有查询的消息；

打开聊天框。可以输入需要在主机或者集群运行的程序或者脚本。在运行前确认该脚本是否被允许。

文件上传。上传文件到 web 服务提供的主机。



5.3 爬虫服务

爬虫服务主要依托 phantomjs 和 casperjs。下载地址如下：

http://phantomjs.org/download.html

http://casperjs.org/

下载解压，设置环境变量，指向解压目录和解压后的 bin 目录即可。

支持第三方爬虫。

爬虫启动脚本（startSpider.sh)示例：

```
#!/bin/bash
export PATH=$PATH:/home/nws/pacong/bin
cd ./spider
casperjs --start-url=$1 --required-values=$4 --mabiao=$6 --skipped-values=$5 --startnum=1 --web-security=no --load-images=no --limit=100 --mysearch=$2 $3
```

结合管理界面，参数$1 为爬取网址，参数$2 为全文索引地址，参数$3 为爬虫程序，参数$4 对应处理网址，参数$5 为忽略网址，参数$6 为密文密钥（用于密文检索用，主要用于信息加密传递到全文搜索服务）。

爬虫 spider.js 示例：

```
/*
模块名称：spider.js
模块功能：爬取信息，并传输到搜索引擎
模块编制：
    苏州泥娃软件科技有限公司
    2017.9.22
 */
(function (window, document, undefined) {
    'use strict';
    var postnum = 0;
    var utils = require('utils'),
```

```
            helpers = require('./helpers'),
            flcodec = require('./flcodec'),
            fs = require('fs'),
            config = require('./config');

    var flago=false;
      // ###################  WORKING CODE  ##################

      // Create Casper
      var casper = require('casper').create({
        verbose: config.verbose,
        logLevel: config.logLevel,
        pageSettings: {
          loadImages: config.loadImages,
          loadPlugins: config.loadPlugins
        }
      });

      // Echo options hash to screen
      if (config.logLevel != 'error') {
          utils.dump(casper.cli.options);
      }

      // ###################  Initializing Vars  ##################

      // URL arrays
      var visitedUrls = [], pendingUrls = [], skippedUrls = [];
      var times = [];
      var visitedResourceUrls = [];

      // required and skipped values
      var  requiredValues  =  casper.cli.get('required-values')  ||
config.requiredValues,
          skippedValues   =   casper.cli.get('skipped-values')   ||
config.skippedValues,
          linkLimit     = casper.cli.get('limit') || config.limit;
```

```
// setting hard value for linkLimit so it doesn't go on forever
//if (linkLimit === 0) {
  //linkLimit = 100;
//}


// look for a command line cookie and then for a cookie in the config
var cookie = false;

if (typeof casper.cli.get('cookie') === 'string') {
  try {
    cookie = JSON.parse(cookie);
  } catch (e) {
    casper.die('User defined cookie is not valid JSON.');
  }
} else if (casper.cli.get('cookie') === true) {
  cookie = config.cookie_data;
}

// Initializing Data Object
var dataObj = {
  start: casper.cli.get('start-url') || config.startUrl,
 mysearch:casper.cli.get('mysearch') || config.mysearch,
 startnum:casper.cli.get('startnum') || config.startnum,
 mabiao:casper.cli.get('mabiao') || config.mabiao,
  date: new Date(),
  dateFileName:       casper.cli.get('date-file-name')       ||
config.dateFileName,
  requiredValues: helpers.prepareArr(requiredValues),
  skippedValues: helpers.prepareArr(skippedValues),
  cookie: cookie,
  links: [],
  errors: [],
  times: [],
```

```
  messages: [],
  skippedLinksCount: 0,
  logFile: '',
  linkCount: 1,
  userAgent: casper.cli.get('user-agent') || config.userAgent
};


// ###################  Spider Function  #################
//que = dataObj.startnum;
var spider = function(url) {
  // Add the URL to visited stack
 visitedUrls.push(url);

  // Add cookie
  if (dataObj.cookie) {
    casper.page.addCookie(dataObj.cookie);
  }

  // add userAgent if supplied
  if (typeof dataObj.userAgent !== 'undefined') {
    casper.userAgent(dataObj.userAgent);
  }

  // Open the URL and modify
  casper.open(url).then(function() {

    // ###################  Setup Link Data  #################

    // Get current response status of URL
    var status = this.status().currentHTTPStatus;

    // Log url
   if(status==undefined || status==403)
   {
     return this.exit(2);
```

```
        }
         if(status >= 400) {
                    this.echo("* " + this.colorizer.format(status,
helpers.statusColor(status)) + ' ' + url);

         } else {
             this.echo("      " + this.colorizer.format(status,
helpers.statusColor(status)) + ' ' + url);
         }

         // Instantiate link object for log
         var link = {
           url: url,
           status: status
         };

         // Push links to dataObj
        dataObj.links.push(link);
       var title = this.getTitle();

       var postsearch = {};
       postsearch = casper.evaluate(function (url, title, posturl) {

           var postsearch = {};
           try {
               postsearch._id = url;
               postsearch.title_s = title;
               postsearch.title = title;

               postsearch.url = url;
           } catch (e) {
                __utils__.echo("Error in fetching json object " +
e.message);
           }

             postsearch.content = __utils__.findOne("body").innerText;
```

```
            return postsearch;
        }, url, title, dataObj.mysearch);

        var str = JSON.stringify(postsearch);
        var tt = "[" + str + "]";
        this.echo("Content:  " + tt);
        if (postsearch.title != "" && postsearch.title != undefined) {
            this.echo("@title_s: " + postsearch.title);
            if(dataObj.mabiao==""||dataObj.mabiao==undefined)
             {
            this.echo("No flcodec!");
            }else
            {
            this.echo("Start flcodec!");
            postsearch.title_s = flcodec.trans_s(postsearch.title);
                postsearch.title = flcodec.trans(postsearch.title);
                if (postsearch.content != "" && postsearch.content !=
undefined) {
                    postsearch.content_s                                =
flcodec.trans_s(postsearch.content);
                    postsearch.content                                  =
flcodec.trans(postsearch.content);
                }

                postsearch.url = flcodec.trans(postsearch.url);
            }
        }

        var str = JSON.stringify(postsearch);
        var tt = "[" + str + "]";

        if (status == 200 && tt.length > 0) {
             try {
                    postnum = casper.evaluate(function (posturl, tt,
postnum) {
```

```
                try {
            JSON.parse(__utils__.sendAJAX(posturl, "POST", tt, false,
{ contentType: "application/x-www-form-urlencoded" }));
                    postnum++;
                    return postnum;
                } catch (e) {
                    __utils__.echo("Error in fetching json object "
+ e.message);
                    postnum++;
                    return postnum;
                }
            }, dataObj.mysearch, tt, postnum)

            this.echo("@postnum: " + postnum);
        } catch (e) {
            console.log("Error in: " + e.message);
        }
    }

    //    ###################       Process    Links    on    Page
##################

    var baseUrl = this.getGlobal('location').origin;

    // Find links on the current page
    var localLinks = this.evaluate(function() {
      var links = [];
      __utils__.findAll('a[href]').forEach(function(e) {
        links.push(e.getAttribute('href'));
      });
      return links;
    });

    // iterate through each localLink
    this.each(localLinks, function(self, link) {
```

```
            // if url contains text
            var containsText = function (element, index, array) {
              return (newUrl.indexOf(array[index]) >= 0);
            };

            // Get new url
            var newUrl = helpers.absoluteUri(baseUrl, link);

            // If url is not visited, pending or skipped:
            if (pendingUrls.indexOf(newUrl) === -1 &&
                visitedUrls.indexOf(newUrl) === -1 &&
                skippedUrls.indexOf(newUrl) === -1) {

              // if newUrl is not does not contain skipped, and does have
required
              if (!dataObj.skippedValues.some(containsText) &&
                  dataObj.requiredValues.every(containsText)) {
                    pendingUrls.push(newUrl);

              } else {

                // add it to skipped array
                skippedUrls.push(newUrl);

                casper.log('Skipping ' + newUrl, 'debug');

                // add to counted skipped links
                dataObj.skippedLinksCount++;

                return;
              }
            } // eof visited, pending, skipped
          }); // eof each links

          // If there are any more URLs, run again.
          //if (dataObj.linkCount < linkLimit)
```

```
            {
             var nextUrl= pendingUrls.shift();
             if(nextUrl!=""&&nextUrl!=undefined)
             spider(nextUrl);
            }
          //else {
            //  casper.log('There are no more URLs to be processed!',
'Warning');
            ///}
          }); // eof page function
        }; // eof spider function


        // Start Spidering!
        casper.start(dataObj.start, function() {
            this.echo('Starting to spider ' + dataObj.start, 'info');
            spider(dataObj.start);
        });

        casper.run();

        // if console error exists
        casper.on('page.error', function(msg, trace) {
          var error = {
            msg: msg,
            file: trace[0].file,
            line: trace[0].line,
            func: trace[0]['function']
          };

          this.echo('* ERROR: ' + error.msg, 'error');
          this.echo('    file: ' + error.file, 'warning');
          this.echo('    line: ' + error.line, 'warning');
          this.echo('    function: ' + error.func, 'warning');

          dataObj.errors.push(error);
```

```
});

// if console message exists
casper.on('remote.message', function(msg) {
  this.log('MESSAGE: ' + msg, 'WARNING');
  var message = {
    url: casper.getGlobal('location').href,
    msg: msg
  };
  dataObj.messages.push(message);
});

// stop crawl if there's an internal error
casper.on('error', function(msg, backtrace) {
  this.log('INTERNAL ERROR: ' + msg, 'ERROR' );
  this.log('BACKTRACE:' + backtrace, 'WARNING');
  this.die('Crawl stopped because of errors.');
});

// Find the longuest request
casper.on('resource.requested', function(resource) {
    times[resource.id] = {
        start: new Date().getTime(),
        url: resource.url
    };
});

casper.on('resource.received', function(resource) {
    if (resource.stage == 'end') {
        times[resource.id].time = new Date().getTime() -
times[resource.id].start;
        times[resource.id].status = resource.status;
        dataObj.times.push(times[resource.id]);
        if (visitedUrls.indexOf(resource.url) == -1 &&
visitedResourceUrls.indexOf(resource.url) == -1) {
            visitedResourceUrls.push(resource.url);
```

```
                if(resource.status >= 400) {
                    casper.echo("*                 "                +
this.colorizer.format(resource.status,
helpers.statusColor(resource.status)) + ' ' + resource.url);
                } else {
                    casper.echo("#                       "          +
this.colorizer.format(resource.status,
helpers.statusColor(resource.status)) + ' ' + resource.url);
                }
            }

if(!flago&&(resource.url.search(/abuseip/i)>0||resource.url.search(/u
nhuman/i)>0 || resource.url.search(/imhuman/i)>0))
            {
                flago = true;
                this.echo("ABUSEIP:                       "          +
this.colorizer.format(resource.status,
helpers.statusColor(resource.status)) + ' ' + resource.url);
                this.exit(2);
            }
        }
    });

    // after crawl is complete, write json file with results
    casper.on('run.complete', function() {
      var   fileLocation   =   casper.cli.get('file-location')   ||
config.fileLocation;
      var filename;

      // set filename for logging
      if (dataObj.dateFileName) {
        filename = helpers.getFilename(fileLocation) + '-data.json';
      } else {
        filename = fileLocation + 'data.json';
      }
```

```
        dataObj.logFile = filename;

        var data = JSON.stringify(dataObj, undefined, 2);

        // write json file
        fs.write(filename, data, 'w');

        if (typeof config.cb === 'function') {
          config.cb(data);
        }

        // Find the longest request.
        var longest = times.sort(function(reqa, reqb) {
            return reqb.time - reqa.time;
        })[0];
        this.echo('', 'INFO');
        this.echo(utils.format('Longest request: %s (%s) with %dms',
longest.url, longest.status, longest.time), 'INFO');
        this.echo('', 'INFO');

        this.echo('Crawl has completed!', 'INFO');
        this.echo('Data file can be found at ' + filename + '.', 'INFO');
      });
    })(this, this.document);
```

非常规过程

系统启动命令：./waiter 0.0.0.0 224.0.0.18 8099 32000 ./page，参数说明如下：启动程序名称 web 服务侦听地址 组播侦听地址 web 服务端口 组播端口 网页文件所在目录。

系统停止该进程，killall waiter；或者 ps -ef|grep Waiter，获取进程号，然后 kill -9 进程号。

防火墙设置。打开 tcp 的 web 服务端口和 udp 的组播端口。

文件和配置文件传递。先上传文件，然后通过对话框给集群下命令实现。命令格式如下：

wget -O 本地文件 http://服务主机/上传文件。

操作命令一览表

ls 、tar 、cat、free、top 等

程序文件（或命令文件）和数据文件一览表

Waiter startSpider.sh start.sh ./spider ./page

苏州泥娃软件科技有限公司
SUZHOU NIWA SOFTWARE TECHNOLOGY CORPORATION
Add：江苏苏州太仓宁波东路66号德国留学生创业园519室
Tel：0512-33021366