网络机器人的分布性和可控性设计

国防科技大学计算机学院　阳爱民 杨岳湘 瞿国平

本文提出一种新的Robot的设计和实现方案，描述了系统的设计思想、组成、工作流程，并提供一些技术参数；重点说明了多个Robot分布并行工作的设计及其行为可控性的设计。

引　言

随着Internet的快速普及和发展，信息资源与站点越来越多，如何从浩瀚的信息海洋中找到有用的信息是日益重要的问题，而网络机器人(Robot)[1]是一种自动在WWW网上获取信息并进行漫游的程序，其作用是为网络资源有效的信息建立目录说明。

然而，由于网络信息的多样化：语言的多样化，单词格式的多样化(电子邮件、链接、压缩编码……)；文件格式的多样化(文本文件、图像文件、声音文件……)，再加上空间，时间的限制，设计不当的网络机器人(Robot)，会导致相当严重的后果，如数据重复，数据量过大，造成服务器失常等。本文提出一种新的Robot的设计和实现方案，将分布处理技术用于其上，实现多个网络机器人(Robots)同时并行工作，系统管理员通过友好的界面对Robots的行为进行全方位的控制。

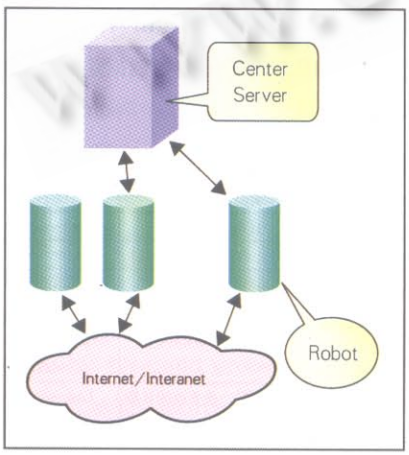

图 1　结构图

基本设计思想

我们将软件设计成独立的两部分，一部分安装在一个Center Server上，称之Control Server组件；另一部分，分别安装在远方的其他计算机上，称为Robot组件(如图1)。

Robot组件的主要功能是接收Control Server组件发送的配置文件，以这些配置文件来规范自己的行为，并在网上搜索符合要求的信息资源，经处理后，定时地向Center Server传送。

Control Server组件主要的功能是向各个Robot发送配置文件，接收Robot送来的信息资源，经协调后，写入数据库中，同时，为用户提供查询服务。配置指示文件由系统管理员，通过Control Server组件的界面设置完成；每个Robot都有一个ID号，用来鉴别身份。

Robot的结构设计

1. 内部状态图(如图2)

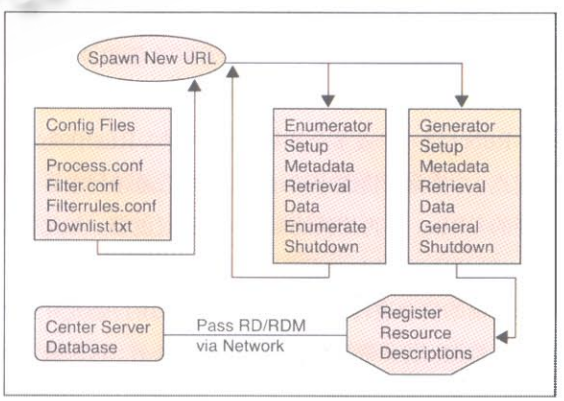

图 2　内部状态图

2. 各组成部分说明

如图2所示每个Robot主要由列举器(Enumerator)、

生成器(Generator)和一些配置文件组成,每个列举器和生成器分别对应一个过滤器。

(1)列举器和生成器: 列举器的主要作用是利用网络协议和机制,寻找、定位或发现新的网络资源,列举器测试每个资源看它是否应该被列举,以便发现新的资源。例如: 列举器可以从一个html文件中抽取一个超文本的链,并用这条链去发现新的资源; 生成器主要是测试每个资源,看它是否可以被创建一个资源说明(RD),若能,则创立一个RD,经登记后,通过网络传送到Center Server中。列举器(Enumerator)和生成器(Generator)分别对应一个过滤器(filter),每个过滤器在过滤时分六个阶段进行,分别是Setup、Metadata、Retrieval、Data Shutdown以及Enumerate和Generate。

(2)控制Robot行为的配置文件。Robot的行为由四个配置文件来控制,它们是: Process.conf,Filter.conf,Filterrules.conf和Downlist.txt。Process.conf为Robot定义了大部分操作参数,包括告诉Robot应该使用哪些过滤器(这些过滤器是在filter.conf中已经定义了的); Filter.conf包括了列举过滤器和生成过滤器等过滤器的定义,该文件可以包括多个过滤器的定义; Filterrules.conf包括了Robot搜索的起始点或种子URLs(Seed URL)以及过滤器的过滤的规则; Downlist.txt是所有已经下载网页的索引,其作用是避免Robots的重复下载。

上述四个配置文件是纯文本文件,你可以用任意一种标准的文本编辑器编辑它。

(3)过滤过程: Robots用过滤器来控制哪些网络资源被处理以及怎样处理,一旦当Robot发现资源的一些参考消息(或资源本身)时,它运用过滤器来列举它(为了发现更多的资源),并且决定它是否能被产生一个资源说明(RD),若能产生RD,则产生,并将它暂时存放在Cache中。

Robot以一个或多个开始点(或种子URLs)作为工作的起点,并为每个资源申请过滤器,然后通过列举器来产生新的URLs,并再为新的URLs申请过滤器,如此往复下去。过滤器开始时,执行一些必需的初始化操作,然后,它对当前的资源进行比较测试,测试的目的,是允许或拒绝这一资源; 若一资源被允许,这就意味着,这一资源允许继续被过滤; 若一资源被拒绝,Robot将最终列举它,并尝试发现更多的资源,生成器也可能为它创建一个资源说明(RD); 注意这些操作没有必然的联系,有的资源被列举,有的资源被生成了RD,许多资源两者都有; 例如,一个HTML文档,它包含一些连接到其他文档的链,生成器可以为它本身产生一个RD,而列举器也能列举它所链接到的文档。

(4)过滤过程的几个阶段: 在过滤器的过滤过程中,列举过滤器和生成过滤器,它们都经过五个阶段,其中有四个阶段相同: Setup,Metadata,Data和Shutdown;注意,Retrieval不是过滤过程的一个阶段,因为它的作用是从网上检索和找回某一网页。每一阶段的工作情况如下:

Setup阶段:

在这一阶段,过滤器忙于初始化,执行一些初始化操作,不能获得有关资源的URL或content的消息; 在一个Robot的生命周期中只发生一次。

Metadata阶段:

在这一阶段,Robot已经遇到资源的URL[1,3]但没有下载资源的内容(content),因此,有关资源内容的信息仍然无效,而有关URL自身的一些信息以及来自Filter.conf文件的数据是有效,有效的资源如表1所示; 在这一阶段,Robot根据这些有效的Metadata信息,来过滤资源。

表 1

CompleteURL	资源的位置	http://www.nudt.edu.cn/
Protocol	URL的访问协议	http,ftp,file
Host	URL的地位部分	www.nudt.edu.cn:80
IP address	URL地址的数字表示	210.43.211.30
URI	URL的路径部分	/index.htm
Depth	搜索的深度	10
Enumeration-filter	列举过滤器的名称	enuneration-filer=my-enumerator
Generation-filter	生成过滤器的名称	eneration-filer=my-generator

Data阶段:

在这一阶段,Robot已经下载了URL资源的内容(在Retrieval阶段),Robot可以访问有关content的数据,如description,the author等等,若资源是一个HTML文件,则Robot将分析HTML头部〈META〉标记内的数据; 因此,包含在〈META〉标记内的数据是有效的,有效的数据如表2所示:

表 2

Data	含义	举例
Content-charset	资源所使用的字符集	
Content-encoding	压缩方法	content-encoding:X-gzip
Content-length	资源的大小(以byte为单位)	Content-length:3495
Content-Type	媒体的类型	Content-Type: text/html(imgc/jpeg)
Expires	资源过期的日期和时间	Expires:Thu,01 Oct 1999 08:00:00 GMT
Last-modified	资源的最后一次更改的日期	Last-modified:Fri,15 Jan 1999 12: 45: 26 GMT
Data in <META> tags	附加信息	Author,Description Kywords

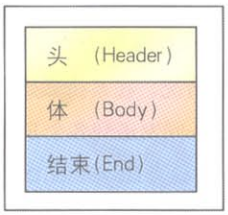

图 3 过滤器图

在上述的数据中，除〈META〉中的数据外，其他数据均来自远方服务器(如WWW服务器)的HTTP应答(response)头部(在检索和找回这些资源时)；在这一阶段，Robot根据这些有效的Data信息，来过滤资源。

Eumerate阶段：

在这一阶段，Robot列举当前的资源，看是否有指向其他能被检查的资源，并产生新的URLs。Generate阶段：

这一阶段，Robot将为资源产生一个资源说明(RD)，登记RD，并存在RD到Cache中。

Shutdown阶段：

执行一些必要的终止操作，在Robot的一个生命周期中只发生一次。

(5)过滤器语法：文件filter.con包含了列举过滤器和生成器的定义，这个文件可以包含多个过滤器。每个过滤器都有一个很好的定义结构，如图3所示”)；

体(Body)由一系列过滤器的指示(directives)组成，这些指示定义了过滤器在“Setup,Metadata,Enumerate(或Generate)和Shutdown阶段的行为；Robot用六种指示来配置过滤各阶段的操作每个指示有它自己专用的Robot应用函数和对此函数有效的参数与之对应；例如，列举函数有对应的Enunerate directive，生成函数有对应的Generate directive等等。

结束(End)由〈1Filter〉标记，下面是一个例子：

〈Filter name="enumeration1"〉 # 说明过滤器的名称，标志过滤器定义的开始。

说明filterrules.conf文件的位置；

Setup fn=filterrules-setup config=/config/filterrules.conf

MetaData fn=filterrules-process #Process the rules

#Filter by type and process rules again

Data fn=assign-source dst=type src=content-type

Data fn=filterrules-process

#Perform the enumeration on HTML only

Enumerate enable=true fn=enumerate-urls max=1024 type=text/html

Shutdown fn=filterrules=shutdown #Cleanup

</Filter>

注：符号#后面的文字是注释。

(6)文件Process.conf参数设置：文件Process.conf为Robot定义了许多参数，其中有两个最重要的进程参数就是：enumeration-filter和generation-filter；这两个参数决定Robot在列举(enumeration)和生成(generation)阶段使用哪些过滤器；我们定义这两个参数的缺省值为enumeration-default和generation-default。注意，所有的过滤器必须是在文件filter.conf中定义了；下面这些参数，系统管理员可以设定：

auto-proxy:代理设置；例如auto-proxy=”http://dnsl.xtnu.edu.cn: 80/”

connect-timeout:允许网络应答一个连接请求的最长时间；例如，connect-timeout=120

convert-timeout：允许文档转换的最长时间；例如，convert-timeout=600

depth:搜索深度；例如，depth=10;

enable-ip：是否为每个被创立的RD的URL生成一个IP地址；例如，enable-ip= [true|false]

enable-robots.text:在访问每个节点是否检查Robots.txt文件；可设置为yes或no;

engine-concurrent:预前为Robot创建的线程数，engine-concurrent= [1..100];

enumeration-filter:指定enunerator的过滤器；

generation-filter:指定generator的过滤器；

loglevel:log文件的级别，logleve= [0..5]；这些值的含义如下：

level 0:log文件只记录一些严重的错误；

level 1：记录生成和列举行为；

level 2：记录检索的行为；

level 4：记录产生新URL的行为；

level 5：记录检索进程进行的情况；

max-concurrent:一个Robot能同时并发检索的资源的最大数目，max-concurrent= [1..100];

max-filesize-kb:Robot检索文件的最大长度(以kb为单位)；例如：max-filesize-kb=1024

max-memory-per-urL/max-memory:指定每个

URL拥有的最大内存数(以bytes为单位);

onCompletion:决定在Robot完成一个运行周期后，Robot该做什么事情；有以下几种方式可以选择：idle mode、loopback mode、start again mode or quit mode;

password:用于http的身份鉴定和FTP的连接，password=string;

remote-access:是否允许Robot能从远方的主机接受命令，remote-access= [yes|no];

server-delay:这是两次访问同一个Web节点的时间间隔，将阻止Robot频繁地访问同一个节点，server-delay-delay_in _seconds;

tmpdir: 指定Robot创建临时文件的位置;

username: 这是运行Robot的用户的名称，它将被用于http的身份鉴定和ftp的连接。

Bindir: 为Robot指定外部程序所在的位置，以便于Robot运行外部程序，Bindir=path;

Cmd-hook:指定一个外部命令，这是Robot在完成一个运行周期后，要执行的外部命令。

Command-port:指定Robot监听的端口号，Robot从这里接受其他程序的命令;

Command-port=port_number;

以上这些参数，系统管理员通过浏览器在任何与Internet相连的主机上，可以方便的设置。

(7)Robot的工作过程简述：Robot从filterrules.conf读取种子URLs(Seed URLs)，检查URLs和与他们相关联的网络资源，每资源都要经过列举过滤器和生成过滤器检验；若资源通过了列举过滤器的检验，则Robot就列举它，并产生新的URLs，新的URLs又可以被列举过滤器检验，再列举，再产生新的URLs，如此下去，不断发现新的资源；若资源通过了生成过滤器的检验，则Robot对它产生一个资源说明，并经处理，存到Cache中，再在某一设定的时间，将它们传送到Center Server database。

各Robots相互协调工作的设计

在Robots的工作过程中，每个Robot从各自的种子URLs(Seed URLs)出发，不断地到网上搜索资源，虽然它们的种子URLs不一样，但不能排除不搜索到相同的资源；而且每个Robot都要知道Center Server 的database中已存在的资源；否则，这两种情况都会造成资源重复下载。为了解决这个问题，我们对各Robot协调工作作了如下的设计(图4所示):

Center Server中保存一个Downlist.txt文件，该文件记录下载资源的情况，在系统开始工作时，该文件和Config Files 文件一同发送到各个Robot中，Robots在获取某一资源时，先检查Downlist.txt文件，看是否记录了这个资源；若有，就不下载了。同时，在Center Server中，还设计了一个可以定时产生令牌[2]的模块；该令牌包含了每个Robot的ID号和传递次序，在产生令牌的同时，还产生了一个list.txt文件。Center Server在Robot工作一段时间T后，产生令牌和list.txt，此时，list.txt文件为空；令牌和list.txt文件被发送到1号 Robot，1号Robot处理令牌，将自己的ID号去掉，同时，将自己在T时间内下载的资源情况，按一定格式(与Downlist.txt文件格式一样)写入list.txt中；然后，将令牌和list.txt传送给2号Robot，2号Robot从令牌中去掉自己的ID，并打开list.txt文件，检查自己在T时间内下载的资源在list.txt是否存在，将已存在的资源去掉，并将list.txt没有记载的资源的消息，写入到list.txt中；然后，传给3号Robot，…；令牌最终传回到Center Server,Center server将list.txt文件合并到Downlist.txt中去；同时，去掉令牌和list.txt文件；再过T时间，又重复上述过程。我们采用这种方法解决资源的重复下载这个问题。■

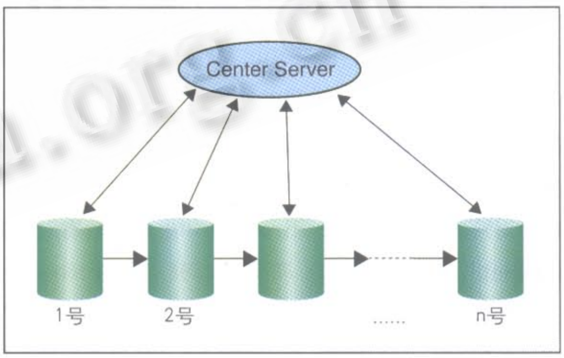

图4 Robots相互协调过程图

参考文献

1 王玉琳，凌涛，沈美娥，WEB程序设计教程，北京：电子工业出版社，1997.206-248

2 朱海滨，分布处理技术，长沙：国防科技大学出版社，1997.10-30

3 T.Berner-Lee,L.Masinter,M.McCahill,Uniform Resource Locators,RFC 1738,1-96

4 D.Hardy,M.Schwartz,U.Colorade,D.Wessels,Harvest User'S Manua(Version 1.4),University of Collorado at Boulder,Technical Report CU-CS-743-94,1996,3-51,76-77