信息检索系统 Web 搜索引擎 (3)

计算机科学1998Vol. 25 №. 5

Web信息检索服务系统与搜索引擎*)

Web Information Retrieval Systems and Search engines

常桂然 张晓辉

(东北大学软件中心 沈阳110006)

摘 要 This article discusses the basic functions and architecture of robot-based Web indexing and search systems and analyzes the document discovery algorithms for indexing mode and real-time search modes. Algorithms for data collection control and some other problems with the development of Chinese-English Web indexing and search servers are also discussed.

关键词 WWW, Indexing, Search, Web, Search engine, Document discovery, Data collection control

1 引言

由于WWW的信息量在不断地增加,而且不断地动态更新,人们研制了许多Web索引与检索服务系统以帮助用户查找信息。按照信息收集和索引的方式,大体上可将Web索引与检索服务系统分为三类。第一种是将手工收集到的信息编成HTML文件,按某种次序排列组织,使用户可以通过索引进行查阅。第二种是,在此基础上,有些站点也提供检索查询功能,例如WAIS等检索工具,这种索引服务系统的生成方法是用手工键入新的URL地址,并由系统的管理员将数据输入到数据库中去,需要大量的人力来进行搜集、排序、编制HTML文件并进行维护。这两种方法速度很慢,更新周期长,无法适应Web数据迅速扩展变化的形势。

因此,第三种是,人们希望Web信息索引与检索服务系统能够在较短的时间内、在指定的范围内自动发现新的信息,对其所覆盖的资料进行自动更新,并根据检索规则和从其它Web服务器得到的数据的类型对数据进行加工处理、自动建立索引。根据信息收集与索引方式的不同,这种检索工具又可分为两种。一种是通过相对集中的方法管理信息收集与索引过程,用户可通过电子邮件等手段注册自己的URL(统一资源定位器),例如ALIWEB和Harvest,另一种是通过搜索引擎即软件robot来自动进行数据收集和索引工作,如AltaVista、WebCrawler、InfoSeek、Lycos等,本文主要讨论这种检索工具的基本功能、系统结构和软件robot的工作原理,并介绍某些系统在索引建立模式和实时检索模式下的Web文档发现算法以及数据收集控制算法。

2 基于robot的Web索引与检索服务系统

Web搜索引擎robot是一种软件程序,通过检索一个Web文件并递归地检索该文件所引用的其它文件,robot能够自动遍历整个Web的超链结构。robot是这种软件程序的通称,又常被称为spider,有的也称为"web wanderer"、"web worm"、"web crawler"等。这些称呼的意义仅略有不同,其中使用最多的是spider。第一个robot称为web wanderer,用来发现Internet上的Web服务器,并计算其数目。早在1994年已出现了一批robot,其中最知名的有WebCrawler、Lycos和WWW Worm等。

2.1 系统功能

基于robot的Web索引与检索服务系统的功能主要有两个方面:数据收集和用户查询服务。搜索引擎自动在Internet上搜寻WWW、gopher和ftp等站点资源,返回相应数据资源,然后对它们进行排序,建立索引,并由此产生一个索引数据库。查询则是面向最终用户的,是人类查询者和建有索引的资

*) 本文得到辽宁省科学技术基金的资助。常桂然 教授、博士,主要研究领域为计算机网络、多媒体技术和软件工程。张晓辉 研究生,研究领域为计算机网络和多媒体技术。

源数据库之间的界面，用户可以通过浏览器访问由robot生成的索引数据库。在有些系统中，用户也可通过调用robot的客户端程序，根据需要自行搜索Web主页。

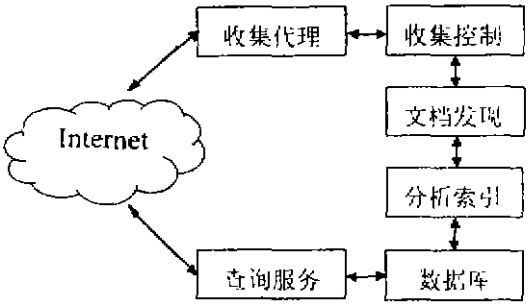

图1 基于robot的信息检索服务系统结构

2.2 总体结构

为了实现系统服务功能，基于robot的Web索引与检索服务系统需要进行下列各项工作。首先是数据收集或文档发现，具体来说就是查找文档的"身份标识"，即URL。在发现一个URL后，应能确定该文档是否值得进行搜集和索引。在收集到一个文档之后，需要执行以下三项操作：做出该文档已被访问的标记，分析该文档到外部的链接，对文档内容加以索引组织。所有这些步骤都涉及到将信息存放到数据库中，从而生成索引数据库，以便用户查询。

对于上述各项工作，系统中都需要有一个相应的组成部分。数据收集代理程序(collector，agent)负责从Web上收集文档，在具体实现时通常需要调用WWW的函数库(libWWW)。数据库负责存放文档元数据、文档间的链接以及索引结构，并保持数据的一致性。为了对用户查询做出响应，需要有一个用户查询服务器，这部分应提供适当的查询方法和良好的用户界面。robot的核心部分亦可称为收集控制引擎，确定探索哪一个新的文档和初始启动搜索过程，根据一定的算法调用数据收集代理，对取回的文档进行分析并建立索引；此外，还需要有一个主控模块，负责控制和协调整个系统各部分的工作。一个典型的系统结构如图1所示，其中收集控制和文档发现两部分构成所谓的收集控制引擎，略去了主控模块。

2.3 软件robot的主要功能

虽然有人把整个Web索引与检索服务系统叫做robot，为便于讨论，本文中的搜索引擎软件robot只包括前文所述的收集控制引擎以及收集代理程序。由收集控制引擎负责，robot从已知文档开始，分析这些文档中指向外部的链接，沿着指向新文档的链接进行探索。为了使收集的文档更具有代表性，在索引中应包括尽可能多的Web服务器。因而用于建立索引的robot通常采用基于广度优先的图遍历算法。此外，收集控制引擎还要对文档类型进行分析，对于无法索引的文档，如图象、声音或二进制文件等，将不予收集。由于Web索引与检索服务系统需要对文档进行分析，建立索引，所以采用的具体索引方式将对系统的实现和效率产生很大影响。

从Web上实际收集文档的工作是由数据收集代理程序来完成的。收集控制引擎将URL作为参数通过程序调用接口交给数据收集代理，而数据收集代理则通过HTTP协议使用WWW函数库访问Web上相应文档的内容。如果访问成功，则将取回文档的内容交给收集控制引擎，否则将说明访问失败的原因。数据收集代理的工作原理比较简单，因此我们主要讨论收集控制引擎的原理与实现问题。收集控制引擎将已经预处理过的数据传递给分析索引模块，由其提取出相应的关键词和数据。robot的数据获取，又可分为文档URL发现和收集控制两部分，我们将分别进行讨论。

2.4 索引数据库组织

每个基于robot的Web索引与检索服务系统都与一定的索引和检索技术相联系，与数据的存储组织方式相关。将HTML格式文件取到本地以后，由一个小程序将其中的辅助部分去掉，并按一定的策略将其中可用于查询的部分(如关键字和一些特定词等)存储到数据库中，形成本地查询数据库，以后如果进行查询，就不必到远地去重新获取HTML格式文件了。一般来说，对索引数据库的要求主要有两点：数据查询的效率和数据组织的效率，也就是速度和存储空间。数据库通常由两部分组成：一个全文索引和一个Web的有向图表示。数据库存储在硬磁盘上，每当增加文档时进行更新。

数据库中存储关于文档、链接和服务器的数据部分，所存储的不是完整的URL，而是把它们划分成分别描述服务器和文档的客体。一个文档中的链接，只是指向另一个文档的指针，每个客体存储在磁盘上单独的btree中；文档在一个树中，服务器在另一个树中，而链接在最后一个树中。用这种方式将数据分开，可以快速地扫描服务器链表，以选择未搜索过的服务器，或上次访问时间与当前时间间隔最长的服务器。

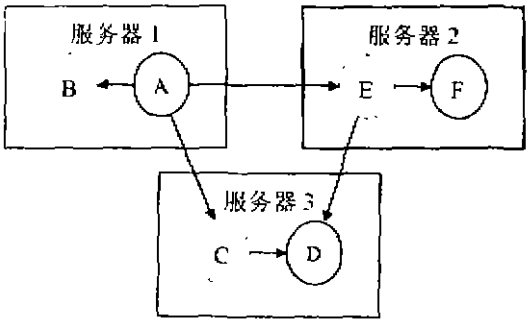

图2 Web的有向图表示

3 索引建立模式文档发现算法

robot的基本文档发现方式如下：从一个或一组URL开始，访问该URL并进行本地索引，同时记录该URL所指HTML文件中所有新的URL锚链(anchor)；然后再以这些新的URL为起始点，继续进行本地索引，直到再没有满足条件的新URL为止。在记录新URL时，通过分析和判断去掉不需要或不想要的URL。

图2为Web一部分的有向图表示。假设已经访问过服务器1上的文档A以及服务器2上的文档E，现在需要确定下一个应该访问的新文档。文档A中具有链接指向文档B、C和E，而文档E具有到文档D和F的链接。这时，需要根据所执行的搜索类型和文档发现策略从B、C、D或F选择一个来访问。robot的工作目的可以是为了建立索引，也可能是帮助用户实时查询。根据robot执行的是索引建立模式还是实时检索模式，采用的文档发现策略将有所不同。本节我们讨论几个robot程序在索引建立模式下所采用的文档发现策略。

3.1 WWW Worm的文档发现策略

WWW Worm的数据收集部分称为wwww，它从WWW上收集URL，并记录在文档库中。wwww的调用命令为：

wwww archive levels HTML1 HTML2 … > new archive

这里archive是上一次运行wwww产生的文档库，这个文档库可以为空；levels表示递归收集的深度；HTML1，HTML2等是给定的起始URL。wwww读取由URL给定的每个文件，并对这些文件中所发现的任何HTML引用，递归地调用自身。这种递归调用进行到levels规定的深度为止。

当wwww打开一个HTML文件时，它读取并记录文件的标题字段，将标题字段和该HTML文件并联起来。每当wwww在一个打开的HTML文件中遇到一个URL时，它读取并记录相伴随的超文本锚链。这个超文本和包含它的HTML文件便与该URL相关联。在处理完一个HTML文档后，wwww将它标记为"已读取"，以避免由于其它HTML文件的引用而再次访问该文件，从而大大提高效率，减轻对网络的影响。

wwww使用UNIX软插座(socket)来打开HTML文档，如果第一次尝试失败，将再进行几次尝试。它还能察觉和记录文档中明显的语法错误。

wwww的文档发现算法有一定的局限性，只有起始HTML文件和文档库文件中在某一递归深度内引用到的URL，才能被wwww发现。因此，在外部没有得到任何引用以及离起点较远(层次较深)的文档或文档树，将无法被wwww所发现。

3.2 RBSE Spider的文档发现策略

为了提高基于robot的索引与检索服务系统的性能，对RBSE Spider系统提出了如下要求：将Web文档图结构发现和索引建立过程分开，从而可在不"重新发现"已知Web图结构的前提下更新索引结构；可以对一个已知子图重新起动spider，而不需"重新发现"已知结构；对Web的影响很小，只限于检索HTML文档；取得初步结果所需的工作量较小。

RBSE Spider实际上由两个软件程序组成。Spider本身负责生成Web图、存入数据库结构中，进行操作处理，并遍历带有"*.html"或"http:*/"的链接。通过对一个已知子图重新起动spider的机制，实现文档发现策略，调用重新起动机制的方式有以下几种：

(1)从一个作为参数传递的给定URL开始进行广度优先搜索；

(2)从一个作为参数传递的给定URL开始执行有限的深度优先搜索；

(3)从数据库中已知的未访问过的URL开始进行广度优先搜索；

(4)从数据库中已知的未访问过的URL开始执行有限的深度优先搜索。

在访问了每个URL以后，spider将执行一些记录操作，以便在发生运行中断的情况下，需要重新访问的文档达到最小。

另一个程序具体负责收集文档，它由spider调用并创建单独的进程。Spider读取发现的URL，按

一定格式存放在数据库中。RBSE Spider 事实上采用的是有限深度-广度优先遍历算法，检索 Web 中包含 HTML 的那一部分，为了避免访问不感兴趣的文档，有可能查不到不明显的 HTML 文档。

3.3 索引建立模式下 WebCrawler 的文档发现策略

在索引建立模式下，目标是为尽可能多的 Web 页面建立索引。在有足够存储空间的情况下，WebCrawler 采用的一种解决方案是使得建立了索引的页面来自尽可能多的 Web 服务器。因此，采用了一种经过修改的广度优先算法，保证每一台 Web 服务器至少有一个文档在数据库中建立了索引，这样做的优点是数据索引的覆盖面较宽。

WebCrawler 建立索引的详细过程是：每当在一个新的 Web 服务器上发现了一个新的文档时，该服务器将被置于一个马上要访问的服务器表中；在访问任何其它文档之前，将从每一个新的服务器上索取一个文档，并建立索引；当所有已知的服务器都被访问过之后，将按顺序对表中所有服务器的文档建立索引，直到发现一个新的服务器为止，这时将重复上述过程。这种建立索引的过程将在执行某一定量的时间后结束，或者执行到探索了一定量的文档为止。

小结 广度优先算法可保证每一台 Web 服务器至少有一个文档在数据库中建立了索引，使数据索引的覆盖面较宽，对用户查询比较有利。同时还可将建立索引的工作负担适当地分散到不同的 Web 服务器，减轻对 Web 正常工作的影响。

如果有一组 URL 地址没有被组外 URL 所链接到，那么 robot 和 spider 就找不到它们。同时由于 robot 和 spider 不能更新太快、难免有不能及时加入的新 Web 地址，所以很多拥有 robot 和 spider 的 Web 索引和检索服务站点同时提供一项由用户加入新 Web 地址的功能。

4 实时检索模式文档发现算法

4.1 实时检索模式下 WebCrawler 的文档发现策略

在实时检索模式下，目标是发现与用户查询内容最接近的文档。WebCrawler 使用的策略为：沿着与用户所要求的内容相近的文档中的链接查询，与沿任意链接查询相比，找到相关文档的可能性要大一些。这与人们浏览 Web 的方式基本一致。

为了找出最初的一组相近文档，WebCrawler 首先在自己的索引库中查找用户要求的内容。对这一组文档中最相关的文档做出标记，并对这些文档中未搜索过的链接进行搜索。在索取一个新文档之后，将对其建立索引，并放到数据库中，然后重新执行用户的查询。查询的结果将按相关性进行排序，接近排序表顶部的新文档将作为进一步搜索的对象。这一过程将重复执行，直到找到足够多的相近文档，可使用户满意，或者达到一个时间期限为止。

关于这种文档发现策略的一个问题是：WebCrawler 可能会漫无目的地沿文档中的链接进行搜索，从而导致毫无关系的路径。为了能象人那样选择最有意义的链接，即以一个锚链文本与查询的相似性为基础来评价每个锚链，WebCrawler 根据文档中的锚链文本生成一个很小的全文索引，用于用户查询，从而选择最相关的链接。

4.2 Fish-搜索算法

Fish-搜索算法仿真鱼群繁殖和寻找食物的过程。鱼群由一个存放 URL 的链表来仿真，每次搜索产生的下一代 URL 的平均数目称为宽度，在未找到相关文档的情况下可以执行的搜索步骤称为深度。在实时操作过程中搜索信息，需要进行以下三项活动：

(1)找到好的搜索起始点，一种方式是使用一个连接性良好的文档，另一种是使用基于 robot 的索引库；

(2)收集文档，在接收(客户)方扫描文档中的相关信息；

(3)扫描收集到的文档，以发现指向其它 Web 文档的链接。

这种搜索算法的有效性在很大程度上取决于发现相关文档的能力，以及避免下载不相关文档的能力。关于搜索算法中使用的导航策略，经过研究得到两个重要结论：在发现超文档结构方面，无论是何种结构，深度优先算法都是最优的，即能发现最大数量的交叉引用链接。而且，这种最优性是稳定的，即在略微偏离严格的深度优先算法时，仍能得到接近最优的结果，"链接加亮"设施可以最大限度地改进浏览查询效率，链接加亮的意思是对可引导至已访问节点的链接做出标记。

Fish-搜索算法的具体实现方法为：每当找到一个相关文档后，其中的链接将被加到链表的前面；如果一个文档是不相关的，其中的链接将被加到链表中标记为在相关文档中出现的所有链接后面。这种偏离严格深度优先算法的做法，可以保证集中精力

搜索相关文档的邻近区域。而且，指向位于不同站点上的文档的链接将得到优先考虑，以便将注意力适当地分散到 Web 上。在向链表添加 URL 时、首先检查该 URL 是否已存在于链表中、从而起到了"链接加亮"的作用。在链表中已存在的 URL，将不会被再次添加到链表中；但如果包含该 URL 的节点中含有相关信息，则可把它移到链表的起始处。搜索的结果是一个相关文档组成的链表，每一个文档带有一个"相关性分数"。

5 数据收集控制算法

数据收集控制模块负责根据文档发现部分得到的新 URL 调用数据收集代理，因为等待服务器响应和网络传输是数据收集的瓶颈，robot 的研制者采用了各种方法来减少自身操作对 Web、Internet 和 Web 服务器的影响。

5.1 WebCrawler 的数据收集控制算法

WebCrawler 使用独立运行的数据收集代理进程、可并行地运行多达15个。当文档发现部分选择了一个新的 URL 后，数据收集控制模块尝试找到一个空闲的数据收集代理程序，请求它索取该 URL。在这个代理程序做出响应后，将把索取工作交给它来做。从实现的角度来说，使用独立的代理进程、可帮助将主进程与存储器问题以及代理程序和 libWWW 中的错误隔离开来。

5.2 AltaVista Scooter 的动态适应数据收集控制算法

如果某些 robot 以很快的速率集中索取大量 HTML 文档，将会堵塞 Web 服务器、使之无法对正常的用户访问及时做出响应。AltaVista 的搜索引擎 Scooter 采用了动态适应的方式来解决这个问题，可以根据一个 Web 站点的处理和通信能力来确定索取 Web 页面的速率。具体算法是：每当从某一 Web 站点索取一个页面后，计算出所需的时间，这主要是网络传输所需要的时间、而不是主机 CPU 时间；然后将该时间间隔乘以100，即所谓"好人系数"，得出在索取下一页面之前应该等待的时间。

采用这种算法之后，保证 Scooter 数据收集占用 Web 服务器的资源不超过其总资源的1%。从而使得 Scooter 既可以按照每秒1页的速率从一个具有 T3传输线路的 Web 站点索取文档、也可以按照每5分钟1页的速率从一个以28.8Kbps 速率与 Internet 相连的 Web 站点索取文档。保证了不影响被访问 Web 站点的正常工作。

结束语 对 Web 信息发现和搜索引擎的许多技术、研究人员正在进行更为深入的研究，我们前面介绍的 robot 实例都是单独工作的，为了提高数据获取效率与利用率，希望各索引服务器之间进行相互协调，数据获取部分与索引建立部分之间也要相互协调、Harvest 项目在这方面做了很多工作。

因为现有的 Web 信息发现和搜索引擎不能令人满意地解决中文索引与查询问题，我们有必要在汉语分词、信息提取、数据的存储、检索算法等方面进行研究，编制出自己的 robot 和分析服务系统。

参 考 文 献

[1] C. M. Bowman, et al., The harvest information discovery and access system, Proc. 2nd Intl. WWW Conf., Chicago, U. S. A, Oct. 1994

[2] D. Eichmann, The RBSE Spider-Balancing Effective Search Against Web Load, Proc. 1st Intl. www Conf., Geneva, Switzerland, May 1994

[3] P. M. E. De Bra and R. D. J. Post, Information Retrieval in the World-Wide Web: Making Client-Based Searching Feasible, Same to[2]

[4] O. Etzioni, The World-Wide Web: Quagmire or Gold Mine? CACM. 39(11)1996

[5] M. Koster, Robots in the Web: threat or treat?, ConneXions, 9(4)1995

[6] O. A. McBryan, GENVL and WWWW: Tools for Taming the Web, Same to[2]

[7] D. Morton and S. Silcot, Systems for Providing Searching Access to Collections of HTML Documents, AusWeb95, http: // www. scu. edu. au/ ausweb95/papers/indexing/morton/

[8] B. Pinkerton, Finding What People Want: Experiences with the WebCrawler, Same to[1]

[9] J. Udell, Search Again, Byte, Jan. 1997