计算机科学 1999Vol. 26№. 4

WWW 上 Meta-Search 的研究与实现

Study and Implementation of Meta-Search on the WWW

陈智健

(广东省邮电科学技术研究院开发部 广州 510620)

Abstract Meta-Search is one of the main ways to enhance the capability of Web Document Searching. This paper describes the development of Meta-Searcher and summarizes Meta-Searcher's general architecture. Based on this architecture, a Meta-Searcher-STRET-Search which integrates the capability of Yahoo, Infoseek and AltaVista is implemented.

Keywords Web, Meta-search, Meta-searcher, Index information system, Search engine

1 引言

World Wide Web 是目前全球最大的信息系统，在 WWW 上查询 Web 文档主要依赖于 Internet 上的索引信息系统[1]，如 Yahoo、Infoseek、AltaVista、WebCrawler、Excite、Lycos 等等。由于 WWW 太大又没有良好的结构且 Web 服务器的自治性，所以 Web 文档的查询难以做到全面而精确。衡量 Web 文档查询的质量主要有两个方面：①是否能把所有相关的文档资源找出来，不要有所遗漏。②尽可能保证查询的准确性，不要把无关的东西返回给用户。根据这两个标准，索引信息系统自身不可避免地存在一些局限性。首先，单个索引系统的索引数据库难以涵盖所有的 Web 资源，现状是各个系统的覆盖区域不一样，索引信息的选取、组织都有所不同。其次，文档相关性评估所采用的技术主要依赖于被查询关键字的匹配情况，因此查询不可能是精确的，当前各个索引系统的 Search Engine(搜索引擎)实现的方法不同，对文档相关性的评估存在着较大的差异。另外还有一些使用上的问题，如各个索引系统都有自己的用户界面及使用方式，而且越来越趋于复杂化，在各个系统间的转换会给用户带来许多不便。

Meta-Search(元搜索)研究方法的提出，正是以上述的不足之处为出发点，希望通过对现有索引信息系统的综合利用，来提高 Web 文档查询的质量。

2 Meta-Search 查询

Meta-Search 查询是一种以现有索引信息系统为基础的查询方法，以这种方法构造的 Web 页面查询工具被称作 Meta-Searcher。Meta-Searcher 本身不通过 spider 程序[2]为 WWW 建立索引，因此既不需要维护庞大的索引数据库，也不需要构造复杂的 Search Engine。一般地，它为用户提供了简单、统一的集成查询界面；用户的查询请求被转换成各个索引系统的 Search Engine 所能识别的格式，然后发送给这些 Search Engine，真正的查询过程由它们来完成。因此，索引系统是 Meta-Searcher 的资源库及基础工具。Meta-Searcher 搜集到各个索引系统返回的结果后，经过处理以一定的格式返回给用户。由 Meta-Search 的研究思想可以看出，以这种方法构造的 Web 文档查询工具具备了以下优点。

1)由于 Meta-Search 方式可以一次让多个索引信息系统并发查询，亦即同时利用了多个索引数据库，查询的覆盖面得到了有效的扩大。

2)由于各 Search Engine 对文档相关性的评估存在着较大的差异，用户可以从不同"观点"的结果中进行选择，增大了命中其原意的概率。

3)由于不需要考虑索引数据库的建立与维护工作，Meta-Searcher 的开发者可以把重点放在结果的处理上，以帮助用户尽可能快地找到其所需要的文档。

4)给用户在使用上带来了极大的方便，免除了用户在各个索引系统间不断的转换，不断地适应各种不同的界面、结构、格式的麻烦。尽管 Meta-Searcher 一般不提供各种复杂的 Internet 服务，但它的简单、明了对只需要输入关键字查询的用户来

说是非常受欢迎的。

5)由于 Meta-Searcher 是独立的查询工具，可以方便地配置于客户端，避免诸如服务器过载之类的问题。

Meta-Search 发展至今，事实上经历了两个阶段。

第一阶段 朴素的 Index 阶段。这是以 All-In-One[3]、W3 Search Engines[4]、CUSI (Configurable Unified Search Index)[5]等 Meta-Search Engine 为代表的。这个阶段的 Meta-Searcher 主要为用户提供 Internet 上现有 Search Engine 的查询入口索引。各个 Search Engine 被按照一定的特征归类，用户一次可以选择一个进行查询。查询关键字被发送出去后，用户转为直接与该 Search Engine 相连，Meta-Search Engine 不再处理任何事务。这种类型的 Meta-Searcher 一般在同一页面或简单的几个页面内集成了几十个 Search Engine 的查询入口，普通用户对其中的大部分都是不熟悉的，也很难记住它们的 URL，这种方法方便了用户的使用。

第二阶段 真正意义上的 Meta-Searcher。在第一阶段中，构造 Search Engine 的查询入口索引虽然方便了用户的使用，但并没有实现对多个索引信息系统的综合利用，真正意义上的 Meta-Searcher 是以 MetaCrawler[6]、SavvySearch[7]等 Web 文档查询工具为代表的。这个阶段的 Meta-Searcher 使用了许多基本的 AI 技术实现 Meta-Search 思想，如机器学习、常识推理、软件 agent 技术等等，大大提高了 Web 文档查询的能力。下面以 MetaCrawler、Savvy-Search 为例介绍这一阶段的 Meta-Searcher 的特点。

1)**MetaCrawler**[8]。是美国华盛顿大学的一个 Meta-Search 项目。它有统一的查询界面和查询语言，用户可以选择 WWW、Computer Products、Newsgroups、Files 为查询范围进行查询，查询语言的语法分为"any words"、"all words"、"as a phrase"三种情况。Meta-Crawler 选择了 Lycos、Infoseek、WebCrawler、Excite、AltaVista 和 Yahoo 这六个著名的索引信息系统作为它的查询基础，其控制流如图 1 所示。

用户输入查询关键字，提出查询请求后，Fomulate Query 模块负责将查询请求装配成各个索引系统的 Search Engine 所能识别的命令格式，把查询命令发送出去，真正的查询操作在 Lycos、Excite……等索引系统中并行地进行。Post Process Results 模块在各索引系统返回的数据中分析和提取查询结果(通常是 Web 文档的 URL 及一些简短的说明)。并非每一个索引系统都支持相同的功能，例如 Lycos 目前还不支持短语的查询(在 MetaCrawler 中查询语法的选择为"as a phrase")，但它支持"all words"查询，Post Process Results 模块承担了更深层次的分析任务，直接分析查询到的 Web 文档。在这个例子中，就是在"all words"的查询结果中将符合"as a phrase"查询的 Web 文档筛选出来。Web 文档的下载由 Download References 模块负责。查询结果经 Collate Results 模块及 Remove Duplicates 模块处理后返回给用户。MetaCrawler 并行地使用了六个索引系统，而且在某些情况下还要对查询得到的索引项所指向的 Web 文档进行下载，这是以牺牲查询速度及网络资源为基础的，目前存在的反对意见不少。

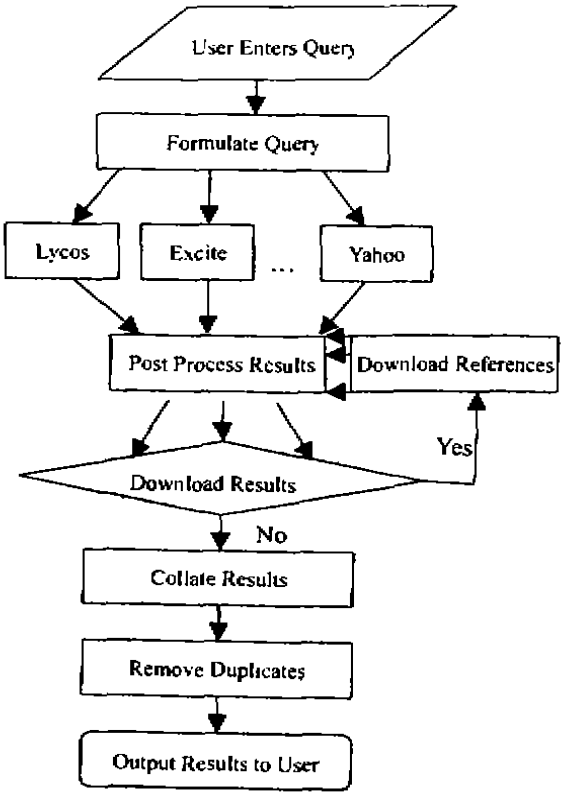

图 1 MetaCrawler 的控制流

2)**SavvySearch**[9]。是一个与 MetaCrawler 相类似的系统，它也有自己统一的查询界面和查询语言，其查询语言的语法跟 MetaCrawler 是一样的，分为"all query terms"、"all query terms,as a phrase"、

"any query terms"。

SavvySearch 与 MetaCrawler 最大的区别是在对索引信息系统的使用上。SavvySearch 使用 Meta-Index 方法实现了对 Search Engines 的动态任务分配。Meta-Index 实际上是一个 n×t 矩阵，t 是被查询项(term)的数目，n 是 Search Engines 的数目，矩阵中一个单元的值是对 Search Engine 查询 term 时表现的评价，值为正数时越大，表示 Search Engine 查询 term 的表现越好；如果是负数，表示 Search Engine 对于 term 不会得出有意义的结果。单元中的值的计算依赖于查询过程中的两类事件：No Results 和 Visits。No Results 事件发生在 Search Engine 找不到相关文档索引项的情况下；Visits 则是指用户选择了 Search Engine 查询到的索引项的 URL。当发生在 No Results 事件时单元值下降(假设降幅为 d)；相反地，Visits 事件使单元值上升(假设升幅为 a)。对于关键字组查询的情形，如一次三个单词的查询，如果发生 No Results 事件，那么每个单词(term)与该 Search Engine 所对应的单元值降低 d/3；如果发生的是 Visits 事件，那么升幅则是 a/3。SavvySearch 根据 Meta-Index 中各 Search Engine 的表现动态地选择参与当前查询的 Search Engines，最大数量不超过五个。这样，较好地解决了 Meta-Searcher 中网络资源的使用问题。对于 Search Engines 所返回的结果，SavvySearch 经过整理后(如剔除重复项)返回给用户。

另外，除了上面提到的 Meta-Search 发展所经历的两阶段，还产生了一些过度性的 Meta-Searcher，它们为多个 Search Engine 提供了一个统一的查询入口，并在同一页面上以"FRAME"的形式为各个 Search Engine 提供返回结果的空间，此外不做任何处理。如 All4One[10]。

3 Meta-Searcher 的体系结构

如果说以 MetaCrawler、SavvySearch 为代表的 Web 文档查询工具体现了真正的 Meta-Search 思想，而称之为真正意义下的 Meta-Searcher 的话，那么从它们的工作流中已经可以体现出 Meta-Searcher 的体系结构来。让我们先来看看 MetaCrawler 与 SavvySearch 各自对 Meta-Searcher 体系结构的总结。

1)MetaCrawler 所提出的 Meta-Searcher 体系结构如图 2 所示。

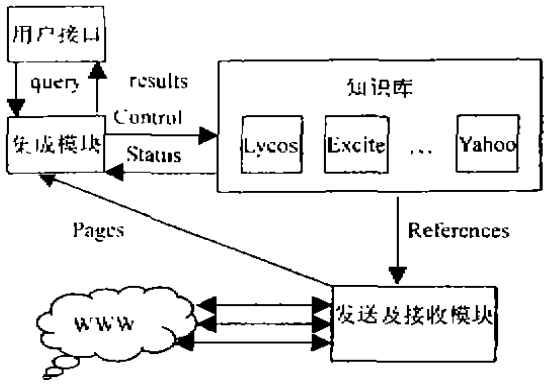

图 2 MetaCrawler 中的体系结构图

· 用户接口：负责将用户的查询请求转换成各个具体的 Search Engine 所能识别的格式。

· 集成模块：负责对 Search Engines 的结果进行后处理，包括提取索引项、剔除重复值，必要时对 Web 文档进行直接的分析，最后将结果整理成一定的格式返回给用户。

· 发送及接收模块：负责向 Search Engines 发送查询请求，及接收返回的页面信息。

· 知识库：记录与 Search Engines 有关的功能状态信息。

2)SavvySearch 提出的 Meta-Searcher 体系结构如图 3 所示。

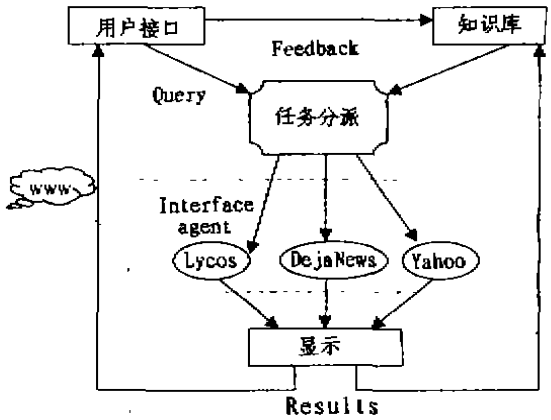

图 3 SavvySearch 提出的 Meta-Searcher 体系结构

SavvySearch 把 Meta-Searcher 的体系结构描述成三大主要部分：

· Dispatch mechanism：这是一个算法，或称为决策方法，决定特定的查询要求应该发给那一个搜索引擎 SE(Search Engine)。

· Interface agents：这是一些小的独立程序，它们负责管理与具体 Search Engine 的交互，主要任务是将用户的查询要求转换为每个具体的搜索引擎所能识别的格式。

· Display mechanism：集合整理各个搜索引擎返回的结果（按某种方式，如：去掉重复的，依照相关度的大小从高到低排列），将其显示给用户。

从以上两个模型中可以看出，除去二者本身独有的特点，它们所描述的 Meta-Searcher 在基本构架上是一致的，都遵循了 Meta-Search 的根本思想。由此，我们可以总结出 Meta-Searcher 的一般框架结构图，如图 4 所示。

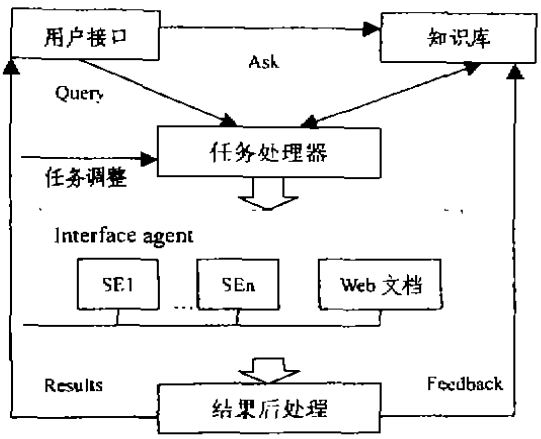

图 4　Meta-Searcher 的一般结构图

· 用户接口：用户提出查询请求、接收查询结果的界面。

· 知识库：记录 Search Engines 的功能特征、用户的反馈信息、查询的历史状态信息。

· 任务处理器：依据当前的查询请求和知识库中的知识，确定完成当前查询任务的方法。

· 结果后处理：负责对 Search Engines 返回的结果进行处理，以一定的格式返回给用户。

4　STRET-Search

STRET-Search 是针对国内 Internet/Intranet 用户而构造的一个 Meta-Searcher，它采用 Java Servlet 方式，以 Solaris 2.5 + JavaWebServer1.1Beta 为软件系统平台，在 SUN SPARCStation 20 上实现。

考虑当前国内网络资源的使用情况及用户的特点，STRET-Search 选择了 Yahoo、Infoseek 和 Alta Vista 这三个国内普通用户使用最多、查询能力强、速度快的索引系统作为其基础查询工具。这三个索引系统的技术分析如下。

1）**Yahoo**。是最早跟国内用户见面的索引信息系统之一，它采用主题索引的方式进行定向搜索。由于 Yahoo 的目录系统有着良好的组织结构，它往往能检索出相关性较好的网页。

Yahoo 查询语言的语法支持"any words"、"all words"、"as a phrase"三种方式；支持通配符，允许用户用"*"匹配任意多个任意的字符；对关键字的大小写不敏感，如：输入"SoftWare Agent"与输入"software agent"，可以得到相同的查询结果；Yahoo 的索引项是由人工来组织的，不排除无实际意义的虚词，但在进行虚词查询时表现出来的能力很有限。一般来说，Yahoo 不支持自然语言查询，但 Yahoo 可以向 AltaVista 求助。Yahoo 已经工作了很长时间，并在运行中不断地加入新的网页，其功能强大和使用方便得到了大多数网络用户的认同和喜爱。

2）**Infoseek**。曾经被著名的浏览器 Netscape 内定为自己缺省的网络搜索引擎，以使用方便、搜索速度快而著称。它号称为 Web 最大的目录系统，有超过 5000 万个索引项被纳入多层目录结构。

Infoseek 查询语言的语法支持"any words"、"all words"、"as a phrase"三种方式；支持通配符，允许用户用"*"匹配任意多个任意的字符，但对关键字的大小写敏感；Infoseek 支持词形的变化，如：用户输入查找关键字"mouse"，Infoseek 会检测所有包含"mouse"、"mice"的文档；Infoseek 的索引采用全文索引的方式，索引范围覆盖了文档除 comment 域以外的几乎所有内容，索引项包括了虚词；Infoseek 支持简单的自然语言查询（plane English query）。

Infoseek 在不断更新完善的过程中，新提供了一项称为 Ulta-match 的技术，它能利用所谓的类神经网络与 cookie 技术，进行记录、追踪、分析来访用户的兴趣所在，从而达到瞄准行为目标的目的。

3）**AltaVista**。被网络用户评价为目前表现最好的索引信息系统之一，有的索引系统将 AltaVista 引用为自己的"伙伴"系统，上面 Yahoo 进行自然语言查询就是其中一例。AltaVista 的索引数据库大概有超过 3100 万个的 Web 页面。

AltaVista 查询语言的语法支持"any words"、"all words"、"as a phrase"三种方式；支持通配符，允许用户用"*"匹配任意多个任意的字符；对关键字的大小写也有敏感性反应：当用户输入的关键字都为小写字母时，不分大小写匹配该关键字的所有形

式，如果输入的关键字含大写字母，则只匹配当前的形式；跟 Infoseek 一样，AltaVista 也采用全文索引的方式，索引项包括了虚词，支持简单的自然语言查询。AltaVista 采用 Digital 公司开发的 64 位运算技术，其搜索速度比一般的搜索引擎要快。

STRET-Search 依图 4 所示的框架结构捆绑了上述三个索引系统的能力，在知识库的设计中，参考了 SavvySearch 的 Meta-Index 的方法，建立查询关键字与 Search Engines 的关联，但并不是用这种相关性来实行对 Search Engines 的动态任务分配。STRET-Search 更注重对查询结果的调节。我们也许会注意到，各个索引系统对用户输入的查询关键字常常能得到数千、数万个匹配结果，然后根据相关度的大小分批返回，用户一般没有耐心通过"Next 20"的超连接把所有结果都看完，一般来说，前面 50 条索引匹配项对用户是最有用的。有的查询页面允许用户定制一次返回的数量，但用户并不知道 Search Engines 对当前查询的关键字是否有好的表现，一次返回过多的结果很不利于用户的选择，而且造成时间上的延时。STRET-Search 的任务处理器根据 Search Engines 与查询关键字的相关性，动态控制各个 Search Engines 一次返回结果的数量，使用户更加方便、迅速地选出其所需要的 Web 文档。

在结果返回的处理上，STRET-Search 采用表决的策略，三者同时匹配到的文档被认为相关性最好，最先显示；两两相同者次之；最好分别显示各自不同的查询结果。

结束语 Web 上的文档查询是一种很不精确的查询，Meta-Searcher 大大增强了这种查询的能力。但并非索引信息系统使用得越多就越好，太多的不同"观点"的查询结果往往会干扰用户的选择，而且依当前网络资源的现状，使用过多的索引系统已被认为是一种不道德的行为。本文认为，选择 3～5 个适当的索引系统是合适且有益的。

参 考 文 献

1 阳小华，周龙骧. World Wide Web 的索引与查询技术. 计算机科学，1997，24(3)：29～34

2 沈达阳，林作铨. Internet 上的软件 Agent. 计算机科学，1997，24(2)：14～19

3 Available at：http：//www. albany. net/allinone/

4 Available at：http：//cuiwww. unige. ch/meta-index-html

5 Available at：http. //www. nexor. com/public/cusi/cusi. html

6 Available at：http. //www. metacrawler. com/

7 Available at：http：//www. cs. colostate. edu/～dreiling/smartform. html

8 Erik Selberg Oren Etzioni. The MetaCrawler Architecture for Resource Aggregation on the Web. Available at：http：//www. cs. washington. edu/homes/selberg

9 Dreilinger D. Description and Evaluation OF Meta-Search Agent，In partial fulfillment of the requirements for the degree of Master of Science. Colorado State University，FortCollins，Colorado，Summer 1996

10 Available at：http：//www. all4one. com/

（上接第 9 页）

制地增大 p，使得 $S\hat{Y}^{(p)}=S_N$，此即所谓的遍历搜索。很显然，用无穷次繁衍来填满一个原本有限的集合 S_N 的做法是低效率的。低效率的原因在于，为实现 $S^{*}\in S\hat{Y}^{(p)}$，使 $S\hat{Y}^{(p)}=S_N$ 并无必要。

SA 存在的问题类似，限于篇幅不再赘述。

参 考 文 献

1 Rumelhart D E，et al. Learning Representation by Backpropagation Errors. Nature，1986，323(6188)：533～536

2 Honik K. Approximation Capabilities of Multilayer Feedforward Networks. Neural Networks，1991，4：551～557

3 Brooks R. Intelligence without Reason. IJCAI-91，1991

4 董聪. 大脑、知觉模型和计算机模拟. 科技导报，1997(7)：7～10

5 董聪. 多层前向网络的逼近机理与拓扑结构学习方法. 通讯学报，1998，19(3)：29～34

6 董聪，刘西拉. 广义 BP 算法及网络容错性和泛化能力的研究. 控制与决策，1998，13(2)：120～124

7 Kirkpatrick S. Optimization by Simulated Annealing：Quantitative Studies. J. Statis. Phys.，1984，34：975～986

8 Holland J H. Adaptation in Natural and Artificial Systems. MIT Press，1992

9 董聪，郭晓华. 广义遗传算法的逻辑结构及全局收敛性的证明. 计算机科学，1998，25(6)

10 董聪. 前向网络全局最优化问题研究. 中国科学基金，1997(1)：23～29