



(12) 发明专利

(10) 授权公告号 CN 1430751 B

(45) 授权公告日 2010.05.26

(21) 申请号 01809998. X

(56) 对比文件

(22) 申请日 2001.05.18

KR 2000006838 A, 2000.02.07, 全文.

(85) PCT申请进入国家阶段日

审查员 陈汝岩

2002.11.22

(86) PCT申请的申请数据

PCT/US2001/016161 2001.05.18

(87) PCT申请的公布数据

W001/90947 EN 2001.11.29

(73) 专利权人 雅虎公司

地址 美国加利福尼亚州

(72) 发明人 P·G·罗雷克斯 T·A·索拉尼勒  
B·R·豪加尔德

(74) 专利代理机构 北京东方亿思知识产权代理  
有限责任公司 11258

代理人 董方源

(51) Int. Cl.

G06F 17/30 (2006.01)

权利要求书 3 页 说明书 34 页 附图 7 页

(54) 发明名称

用于在数据库搜索系统中识别相关搜索的方法和装置

(57) 摘要

一种生成一搜索结果列表并为一搜索者提供相关的搜索的方法。生成与由该搜索者提交的一搜索请求的一匹配的搜索列表项在包括多个搜索列表项的一竞买搜索列表项数据库中被识别。包含在由该竞买搜索列表项数据库生成的一相关的搜索数据库中的相关的搜索列表项被识别为与该搜索请求有关。包括所识别的搜索列表项和一个或多个所识别的搜索列表项的一搜索结果列表被返回给该搜索者。

1. 一种生成一搜索结果列表的方法,该方法包括:

在一按业绩付费数据库(104)中存储多个搜索列表项,每个搜索列表项包括网页定位器和要由所列出的网页的经营者付的竞买金额以及相关网页的统一资源定位器;

在一相关搜索数据库(106)中存储通过检索每个搜索列表项的统一资源定位器所引用的网页的文本、使用该按业绩付费数据库的搜索列表项生成的相关搜索列表项,每个相关搜索列表项包括与按业绩付费数据库的一个文档相关的关键词和被检索的文本;以及

在搜索引擎网络服务器(108)处接收用户搜索者输入的搜索请求;

识别按业绩付费数据库中具有与该搜索项匹配的关键词的搜索列表项;

利用针对各个识别出的搜索列表项的竞买金额对按业绩付费数据库中所述识别出的搜索列表项排序;

生成包括经排序的搜索列表项中的至少一些搜索列表项的结果列表;

识别具有来自所述用的搜索请求相匹配的文本的相关搜索列表项;

生成相关搜索结果列表(314),所述相关搜索结果列表包括作为推荐的相关搜索的识别出的相关搜索列表项中的至少一些相关搜索列表项的关键词;以及

将一搜索结果列表返回给该搜索者,所述搜索结果列表包括所述结果列表以及所述相关搜索结果列表。

2. 如权利要求1所述的方法,其中识别相关搜索列表项包括:

搜索该按业绩付费数据库的一倒排索引;以及

基于该按业绩付费数据库,搜索元信息的一索引。

3. 如权利要求1所述的方法,进一步包括:

按与该搜索请求的关联排序识别的相关搜索列表项;

选择一预定数量的识别的相关搜索列表项作为最关联的相关搜索列表项;以及

将最关联的相关搜索列表项包括在所述相关搜索结果列表中。

4. 如权利要求3所述的方法,其中排序包括:

根据搜索请求的一查询术语的出现的频率,在相关搜索列表项中选择识别的相关搜索列表项。

5. 如权利要求3所述的方法,其中排序包括:

根据搜索请求的一个或多个查询术语的近似度,在该相关搜索列表项中选择识别的相关搜索列表项。

6. 如权利要求3所述的方法,其中排序包括:

根据预定的加权标准,加权相关搜索列表项;以及

根据相关搜索列表项的加权,选择识别的相关搜索列表项。

7. 如权利要求6所述的方法,其中加权相关搜索列表项包括:

增加包括由一广告商标识的竞买的一个或多个搜索项的一相关搜索列表项的相对加权。

8. 如权利要求6所述的方法,其中加权相关搜索列表项包括:

增加包含在由一广告商标识的一搜索列表项的一说明中的一相关搜索列表项的相对加权。

9. 如权利要求6所述的方法,其中加权相关搜索列表项包括:

增加包含在由一广告商标识的一搜索列表项的一标题中的一相关搜索列表项的相对加权。

10. 如权利要求 6 所述的方法,其中加权相关搜索列表项包括 :

增加包含在由一广告商保持的一网页的元标签关键词中的一相关搜索列表项的相对加权。

11. 如权利要求 6 所述的方法,其中加权相关搜索列表项包括 :

增加包含在由一广告商维护的一网页的文字数据中的一相关搜索列表项的相对加权。

12. 如权利要求 3 所述的方法,其中排序包括 :

根据相关搜索列表项的分布,排列相关搜索列表项;以及

根据相关搜索列表项的等级,选择识别的相关搜索列表项。

13. 如权利要求 12 所述的方法,其中排列包括 :

识别包含在相关搜索列表项中的关键信息;以及

根据在相关搜索列表项中出现的关键信息,增加一相关搜索列表项的等级。

14. 如权利要求 13 所述的方法,其中识别关键信息包括 :

检测在相关搜索列表项中的字段性的广告商数据;以及

检测在相关搜索列表项中的搜索后的数据。

15. 一个系统,包括 :

一按业绩付费数据库 (104),其中存储了多个搜索列表项,每个搜索列表项具有网页定位器和要由所列出的网页的经营者付的竞买金额以及相关网页的统一资源定位器;

一相关搜索数据库 (106),其中存储了通过检索每个搜索列表项的统一资源定位器所引用的网页的文本、使用所述按业绩付费数据库 (104) 的搜索列表项形成的相关搜索列表项,每个相关搜索列表项包括关键词和所述检索的文本;

一搜索引擎网络服务器 (108),与该按业绩付费数据库和该相关搜索数据库耦合,该搜索引擎网络服务器用于接收用户输入的搜索请求,并用来识别所述按业绩付费数据库 (104) 中具有与所述搜索请求相匹配的关键词的搜索列表项,利用针对各个识别出的搜索列表项的竞买金额对所述识别出的搜索列表项排序,并生成包括经排序的搜索列表项中的至少一些搜索列表项的结果列表;

相关搜索网络服务器 (110),用于识别具有与所述用户输入的所述搜索项相匹配的文本的相关搜索列表项,并生成相关搜索结果列表 (314),所述相关搜索结果列表包括用于作为推荐的相关搜索而与所述结果列表一起向用户呈现的、识别出的相关搜索列表项中的至少一些相关搜索列表项的关键词;并且

所述搜索引擎网络服务器 (108) 进一步将所述结果列表和所述相关搜索结果列表提供给所述用户。

16. 如权利要求 15 所述的系统,其中多个搜索列表项的每个搜索列表项进一步包括 :

描述性文字,描述该文档,

一标题,以及

元标签,与该文档相关。

17. 如权利要求 16 所述的系统,其中每个搜索列表项包括 :

该描述性文字与文档相关;

该标题,与该文档相关;以及  
所述元标签,与该文档相关。

18. 一种用于响应对包括多个搜索列表项的一按业绩付费数据库的一搜索请求,形成  
用于识别相关搜索的一相关搜索数据库的方法,该方法包括:

将由该按业绩付费数据库的一搜索列表项引用的每个网页存储为一相关搜索数据库  
输入文字;

为该相关搜索数据库输入创建一倒排索引;以及  
创建用于与该按业绩付费数据库的每个搜索列表项相关的键信息的一索引。

19. 如权利要求 18 所述的方法,其中存储包括:

响应应用于由该按业绩付费数据库的搜索列表项引用的两个或多个网页的统一资源定  
位器的根路径部分以及查询自变量,识别相似的网页。

20. 如权利要求 19 所述的方法,其中识别相似的网页包括:

识别一第一网页的第一关键词;  
识别一第二网页的第二关键词;以及  
比较第一关键词以及第二关键词;

当第一关键词和第二关键词具有一预定的关系时,将第一网页和第二网页识别为相似  
的网页。

## 用于在数据库搜索系统中识别相关搜索的方法和装置

[0001] 附录 / 版权参考

[0002] 本发明文献的公开部分包括属于版权保护的资料。当它出现在 U.S 专利商标局专利文件或记录中，版权人不反对任何人完全相同该专利文献或专利公开，否则留所有版权。

[0003] 附上包含计算机程序源代码的附录。因此该附录特地包含在此作为参考，以及包含属于版权保护的如上所述的资料。

[0004] 发明背景

[0005] 本发明通常涉及用于使用如一基于 Internet 的搜索引擎产生一搜索结果列表的方法和系统。更准确地说，本发明涉及用于从一竞买搜索列表项数据库产生搜索结果以及从一相关的搜索数据库产生相关搜索的一列表的方法和系统。

[0006] 搜索引擎通常被用来搜索计算机网络如万维网上可用的信息以允许用户定位存储在该网络中的感兴趣的信息。为使用一搜索引擎，一用户或搜索者通常输入搜索引擎用来产生一信息列表项如网页的一个或多个搜索项，然后该搜索者能访问和利用。由该搜索产生的信息通常被识别为在该信息和由用户输入的一个或多个搜索项间建立的一关联的结果。不同的搜索引擎使用不同的技术来关联信息和搜索项并识别相关的信息。这些搜索引擎也使用不同的技术来向用户提供被识别的信息。因此，正被查找的信息的似然性作为一搜索的结果改变取决于用来执行该搜索的搜索引擎，

[0007] 这种不确定性是使万维网上信息可用的网页经营商特别关心的。在这种设置，通常几个网页经营商或广告商竞争相同群体的可能的观众或用户。因此，被标识成一搜索结果的一网页的能力对一网页的成功来说通常是很重要的。因此，网页经营商通常试图增加他们的网页将被看成一搜索结果的似然性。

[0008] 使网页经营商具有被看成一搜索的结果的一种更能预测的方法的一种类型的搜索引擎是一种“按业绩付费”协议，其中至少部分基于广告商或网页经营商同意付给该搜索引擎经营者的货币总额显示网页。该网页经营商同意已付适量的货币，通常是指竞买金额 (bid amount)，换得响应一用户输入的一搜索项产生的一组搜索结果组中一特定的位置。一较高的竞买金额将产生在该搜索结果组中一更突出的放置。因此，一网页经营者可试图在一个或多个搜索项上出大价以增加对那个术语来说他们的网页将被看成一搜索结果的似然性。然而，有多个相似的搜索项，对一网页经营者来说在每个潜在地相关的搜索项上竞买是困难的。同样，一竞买将在每个检索术语上获利也是不可能的。因此，一搜索引擎经营者不能从使用没有竞买的某些搜索项执行的搜索接收任何收入。

[0009] 另外，因为现有的网页的数目是不断增加的，对一用户来说找到相关的搜索结果变得更困难。获得相关的搜索结果的困难进一步增加，因为搜索引擎是依靠由该用户输入的搜索项。一用户接收的搜索结果直接地取决于该用户输入的搜索项。一搜索项的输入不能导致关联的搜索结果，但是仅一稍有不同的搜索项的输入可以导致相应的搜索结果。因此搜索项的选择经常是该搜索过程重要的一部分。这可能的对该搜索者和为该搜索者推荐相关的搜索的提供到该搜索引擎的该广告商都有利。然而，当前的搜索引擎不允许搜索引擎经营者向一用户提供相关的搜索项，例如将会产生相应的搜索结果的那些搜索项。需要

克服这些缺陷的一系统。

[0010] 概述

[0011] 仅通过介绍,根据本发明的一实施例,接收来自一搜索者的一搜索请求并用来在一竞买搜索列表项数据库上执行一搜索。在该竞买搜索列表项数据库中存储有包括网页定位器和由所列出的网页的经营者已付的竞买金额的搜索列表项。使用该竞买搜索列表项数据库该搜索产生呈现给该搜索者的搜索结果。该搜索请求也被用来在一相关的搜索数据库上执行一搜索。至少部分地使用该竞买搜索列表项数据库的内容,已经形成该相关的搜索数据库。该相关的搜索数据库的搜索产生呈现给该搜索者的相关的搜索的一列表。

[0012] 根据一第二实施例,使用一竞买搜索列表项数据库创建一相关的搜索的数据库。由该竞买搜索列表项数据库引用的来自网页的所有文字被存储和用来产生一倒排索引。另外的索引被用来提高使用该数据库获得的关联性和相关的搜索结果的分布。

[0013] 上述讨论的本发明的说明性的实施例仅仅用介绍的方式提供。在本部分中不应当视为对定义本发明的范围的下述权利要求的一限制。

[0014] 附图的几个视图的简要的描述

[0015] 图 1 是结合一计算机网络举例说明一数据库搜索系统的一方框图;

[0016] 图 2 是举例说明用于操作图 1 的数据库搜索系统的一方法的流程图;

[0017] 图 3a 和 3b 是举例示出用在本发明一个实施例中的搜索结果列表的显示的图;

[0018] 图 4 是更详细地举例说明如图 2 所示的该方法部分的流程图;

[0019] 图 5 是说明用于按关联排序从一相关的搜索数据库获得的搜索结果的方法的流程图,对应于图 4 中的块 404;

[0020] 图 6 是举例说明用于形成一相关的搜索数据库的一方法的流程图;以及

[0021] 图 7 是举例说明用于从一数据库删除相似的页面信息的一方法的流程图。

[0022] 目前优选实施例的详细说明

[0023] 现在参考图 1,图 1 是所示的结合一计算机网络 102 的一数据库搜索系统 100 的方框图。

[0024] 该数据库搜索系统 100 包括一竞买搜索列表项数据库 104,一相关的搜索数据库 106,一搜索引擎网络服务器 108,一相关的搜索网络服务器 110 以及一搜索引擎网页 114。服务器 104,106,108 可由一广告商网络服务器 120 或一客户计算机 122 在该网络 102 上访问。

[0025] 在所举例说明的实施例中该网络 102 是 Internet 并根据适当的标准例如网际协议提供数据通信。在其他的实施例中,其他的网络系统可能被单独的或结合 Internet 一起使用。在该网络 102 中的通信最好按照网际协议或相似的数据通信标准。其他的数据通信标准也可能被使用来确保可靠的数据通信。

[0026] 数据库搜索系统 100 被设置为一客户机和服务器体系结构的一部分。在一计算机网络例如因特网的环境中,一客户端程序是要求一服务的诸如一程序、任务或应用的一过程,该服务是由诸如一程序、任务或应用的另一过程提供的,该另一过程要求由称为服务器程序的一另一过程提供的一服务。该客户端程序使用所要求的服务而不必了解任何关于该另一个服务器程序或该服务器本身的工作细节。在网络系统中,一客户端程序通常在访问由运行一相应的服务器进程的另一计算机提供的共享网络资源的一计算机之上运行。一服

服务器典型的是在一通信媒介例如一网络上可访问的一远程计算机系统。该服务器充当用于一计算机网络的一信息提供者。因此该系统 100 操作为用于由该客户机例如客户机 122 和该广告商网络服务器 120 访问的服务器。

[0027] 客户计算机 122 可以是常规的个人电脑,工作站或任何大小的计算机系统。每个客户计算机 112 典型的包括一个或多个处理器,存储器,输入输出装置以及一网络接口例如一调制解调器。该广告商网络服务器 120,该搜索引擎网络服务器 108,该相关的搜索网络服务器 110 以及该帐户管理网络服务器 112 可以被相似地配置。然而,该广告商网络服务器 120,该搜索引擎网络服务器 108,该相关的搜索网络服务器 110 以及该帐户管理网络服务器 112 每个可包括由一单独的专用网连接的多个计算机。

[0028] 该客户计算机 112 执行一万维网(“网络”)浏览器程序 124。这种程序例子是可以从 Netscape Communications Corporation 获得的 Navigator 以及可以从微软公司获得的 Internet Explorer。该浏览器程序 124 经由一用户使用来输入将被检索的特殊的网页的地址。这些地址被称为统一资源定位器(URLs)。另外,只要检索到一页面,该浏览器程序 124 可以提供当该用户点击包含在该网页中的其他网页的超链接时对其他页或记录的访问。这种超链接提供一自动方式用于该用户输入另一页的 URL 并检索那页。这些页可以是包括如内容简单的文字信息或更复杂的数字编码的多媒体内容例如软件程序,图形,声频数据,视频数据等等的数据记录。

[0029] 客户计算机 122 通过该网络 102 与不同的网络信息提供者通信。这些信息提供者包括该广告商网络服务器 120、该帐户管理服务器 112、该搜索引擎服务器 108、以及相关的搜索网络服务器 110。最好,通信功能性是由超级文字传输协议(HTTP) 提供,尽管可使用其他的通信协议如 FTP, SNMP, Telnet 以及在本领域公知的其他的协议。最好,搜索引擎服务器 108,相关的搜索服务器 110 以及帐户管理服务器 112,以及广告商服务器 120 位于万维网上。U.S. 专利申请号 09/322,627,1999 年 5 月 29 日申请,名为“System and Method for Influencing aPosition on a Search Result List Generated by a Computer NetworkSearch Engine”,以及 U.S. 专利申请号 09/494,818,2000 年 1 月 31 日申请,名为“Method and System for Generating a Set of SearchTerms”均被授予本申请的受让人以及在此合并作为参考。这些申请公开了搜索引擎系统的其他的方面。

[0030] 在该举例说明的实施例中的帐户管理网络服务器 112 包括一计算机存储器介质例如一磁碟片系统以及一处理系统。一数据库被存储在该存储介质上并且包含广告商的帐户信息。运行在客户计算机 122 上的常规的浏览器程序 124 可能被用来访问存储在该帐户管理服务器 112 上的广告商账户信息。

[0031] 该搜索引擎网络服务器 108 在定位到该搜索引擎网络服务器 URL 或在能通过一浏览器程序 124 向该搜索引擎网络服务器 148 提交查询的其他网络服务器的站点的基础上允许网络用户键入关键词查询来在网页上的上百万的可用的页中识别感兴趣的页。在本发明的一实施例中,搜索引擎网络服务器 108 产生至少部分地包括从由该帐户管理服务器 112 实施的竞买进程的结果获得的并格式化的相应的输入的一搜索结果列表。该搜索引擎网络服务器 108 对包含与由该用户在一客户计算机 122 输入的搜索项有关的信息的文档产生一超文字链接列表。该搜索引擎网络服务器将该列表用一网页 114 的形式传送给该网络用户,在运行在该客户计算机 122 上的浏览器 124 上显示它。该搜索引擎网络服务器的一实

施例可通过定位到在 URL <http://www.goto.com/> 的网页被找到。

[0032] 搜索引擎网络服务器 108 被连接到该网络 102。在本发明的一个实施例中，搜索引擎网络服务器 108 包括包含多个搜索列表项的一竞买搜索列表项数据库。该数据库 104 包含用来产生响应用户查询的搜索结果的搜索列表项记录的有序的集合。每个搜索列表项记录包含一相关的网页或文档、一标题、叙述性的文字和一竞买金额的 URL。另外，搜索引擎网络服务器 108 也可被连接到该帐户管理服务器 112。该帐户管理服务器 112 也可被连接到该网络 102。

[0033] 另外，在图 1 的举例说明的实施例中，该数据库系统 100 进一步包括一相关的搜索网络服务器 110 和一关联的相关的搜索数据库 106。该相关的搜索网络服务器 110 和数据库 106 用来向一搜索者提供建议的、相关的搜索以及响应他的查询的搜索结果。使用一搜索引擎网络服务器例如服务器 108 实施搜索信息的用户经常执行同该网络站点搜索引擎的索引数据相比被不适当当地聚焦搜索。用户可使用或含糊和概括如一“音乐”或太具体和集中如“在 1950s 前期间来自 New Orleans 的热情的爵士乐”的搜索项。一些用户要求帮助来精炼他们的查询以便从该搜索引擎更好的获得有用信息。该相关的搜索网络服务器 110 向该用户提供非常适合于该竞买搜索列表项数据库 104 的性能的查询建议。

[0034] 在该举例说明的实施例中，该竞买搜索列表项数据库 104 结合经营网络服务器例如广告商网络服务器 120 的广告商被建立。广告商网页 121 被显示在该广告商网络服务器 120 上。一广告商或网络站点创办人通过驻留在该帐户管理服务器 112 上的一账户与其他的广告商参予一竞标进程。一广告商可竞买多个与该广告商网络站点的内容有关的搜索项。

[0035] 通过该网络站点创办人提交的竞买用来控制向使用客户计算机 122 的一搜索者提出的搜索结果。当使用由该广告商竞买的该搜索项的一搜索被执行时较高的竞买在由该搜索引擎网络服务器 108 产生的一搜索结果列表上得到更有利的位置。在一实施例中，由一广告商竞买的金额包括每当经在该搜索结果列表页上的一超链接访问该广告商网络站点时从该广告商的账户扣除的一货币金额。一搜索者用一计算机输入装置如一鼠标点击该超链接来启动一检索请求以便检索与该广告商的超链接有关的信息。最好，每次访问或点击一搜索结果列表超链接被重定向到该搜索引擎网络服务器 108 来将该点击同用于一广告商的该账户标识符关联。对搜索者来说不是显而易见的这个重定向作用在访问该广告商的 URL 前使用由该搜索者点击的该搜索结果列表超链接将访问编码成该搜索结果页的账户信息。在该举例说明的实施例中，在该搜索结果列表页上的该广告商的网络站点说明和超链接附有该广告商的列表项是一已付费列表项的一指示。每个已付表显示与对每次通过该搜索结果列表介绍到该广告商站点的每次点击由该广告商应付的一价格一致的一金额。

[0036] 该搜索者可点击与那个搜索结果页中的每个列表项有关的超级文字连接来访问该相应的网页。该超级文字连接可在该因特网上的任何地方访问网页，以及包括对位于该广告商网络服务器 120 上的广告商网页的已付表。在本发明的一实施例中，该搜索结果列表也包括不是按广告商的竞买结果定价的并且由一常规的搜索引擎如 Inktomi、Lycos、或 Yahoos 搜索引擎产生的不付费表。不付费超级文字连接也可包括由一编辑组 (editorial team) 手动索引到该竞买搜索列表项数据库 104 的链接。最好，该不付费的列表项在该搜索结果页上的该已付费的广告商列表项之后。

[0037] 相关的搜索网络服务器 110 从在客户计算机 122 的搜索者接收使用该搜索引擎网页 114 输入的搜索请求。该相关的搜索数据库 106 中,包括从该竞买搜索列表项数据库 104 产生的相关的搜索列表项,该相关的搜索网络服务器 110 识别与该搜索请求关联的相关的搜索列表项。结合该搜索引擎网络服务器 108,该相关的网络服务器 110 返回包括位于该竞买搜索列表项数据库中的所识别的搜索列表项以及位于该相关的搜索数据库 106 中的一个或多个所识别的相关的搜索列表项的一搜索结果列表给该搜索者。该相关的搜索网络服务器 110 连同相关的搜索数据库的操作将结合图 2-5 在下面描述。该相关的搜索数据库 106 的形成将结合图 6 描述如下。

[0038] 图 2 是举例说明用于操作图 1 的数据库搜索系统 100 的一方法的流程图。该方法从块 200 开始。在此描述的用于实现图 2 的方法的 Java 源代码及其他方法步骤被包括作为一附录。

[0039] 在块 202,一搜索请求被接收。该搜索请求可用任何适当的方法接收。想象一搜索请求将由使用一客户计算机的一搜索者发起来访问实现如图 2 所示的方法的该数据库系统的搜索引擎网页。一搜索请求可被键入作为在一超文字链接点击中的输入文字来启动该搜索请求和搜索过程。

[0040] 在块 202 后,两个平行的过程被启动。在块 204,该数据库搜索系统的搜索引擎网络服务器在该系统的竞买搜索列表项数据库中识别匹配的搜索列表项。另外,该搜索引擎网络服务器可进一步识别不付费的搜索列表项。

[0041] 同样,在块 206,一相关的搜索网络服务器启动一搜索来在该相关的搜索数据库中识别匹配的相关的搜索列表项。通过匹配搜索列表项,表示各个搜索引擎识别包含在各个数据库中的搜索列表项,这产生与该搜索请求的一匹配。如果用于字母原文匹配的一精确的、字母在一竞买的关键词和一搜索项间产生,可产生一匹配。在其他实施例中,如果一竞买的关键词与一搜索项具有预定的关系,可产生一匹配。例如,该预定的关系可包括匹配已经去除后缀的一词根;在一多个词查询中,匹配几个但不是所有词,或用预定数量的词的接近度定位该多个词的查询。

[0042] 如果已经定位搜索结果,在块 208,来自竞买搜索列表项数据库的搜索结果与来自相关搜索数据库的搜索结果结合。在块 210,一搜索结果列表被返回给该搜索者,例如通过在该搜索引擎网页上显示识别后的搜索列表项并在该网络上将该网页数据传送到该客户计算机。该搜索结果和相关的搜索结果可用任何常规的方式显示。

[0043] 用在本发明的一个实施例中的一搜索结果列表显示的例子如图 3 所示,其是由对术语“CD 写入器 (CD burner)”的搜索得出的开头的几个输入项的显示。图 3 的示例性的显示表示包括多个输入 310a、310b、310c、310d、310e、310f、310g、310h、310i 的一搜索结果列表部分、其他搜索类别 (search category) 的一列表项 312 以及一相关的搜索列表项 314。

[0044] 如图 3 所示,一单个输入,如在该搜索结果列表中的输入 310a 包括该网络站点的一说明 320,最好包括一标题和一短的文字的说明以及一超链接 330,当该超链接被一搜索者点击时,将该搜索者浏览器引导到该描述的网络站点所处的 URL。该 URL 340 也可以在该搜索结果列表输入 310a 显示,如图 3 所示。当浏览图 3 的搜索结果项显示 310 的远程搜索者选择或点击该搜索结果项显示 310 的超链接 330 时,该搜索结果项的“点击”发生。

[0045] 搜索结果列表输入 310a-310h 也表示该广告商的搜索列表项的等级值 360a、

360b、360c、360d、360e、360f、360g、360h、360i。该等级值 360a-360i 是一序数值,最好是由该搜索引擎网络服务器的处理系统产生并指定给该搜索列表项的一数字。最好,该等级值 360a-360i 是用软件实现的在该竞买金额、等级和一搜索列表项的搜索项间建立一关联的一过程中被指定。该过程收集匹配一特定的搜索项的所有搜索列表项、按从最高到最低竞买金额对搜索列表项排序,并按顺序对每个搜索列表项指定一等级值。最高竞买金额得到最高的等级值,第二最高的竞买金额得到第二最高的等级值,直到获得最低等级值的最低竞买金额。等级值与竞买金额间的关联如图 3 中所示,每个已付的搜索列表项输入 310a-310h 显示用于那个输入的广告商的竞买金额 350a、350b、350c、350d、350e、350f、350g、350h、350i。如果具有相同搜索项的两个搜索列表项也具有相同的竞买金额,在时间上先接收的竞买将被指定较高的等级值。

[0046] 图 3 的搜索结果列表不包括不付费的列表项。在优选实施例中,不付费列表项不显示一竞买金额以及在最低等级的已付列表项之后显示。由利用对象分布数据库和本领域公知的文字搜索算法的一搜索引擎产生不付费的列表项。这样一种搜索引擎的例子是由 Inktomi Corporation 运作的搜索引擎。由该远程的搜索者输入的最初的搜索查询被用来通过该常规的搜索引擎生成不付费的列表项。

[0047] 该列表项 312 的其他搜索种类表示用于搜索可能与该搜索者的输入搜索项 316 相关的其他可能的种类。其他的搜索种类被选择用于通过识别一组如包含该输入搜索项 316 中的计算机硬件显示。然后在该组中的种类被显示为可由该搜索者点击用于另外的搜索的超链接。这提高了该用户的输入搜索不能找到适当的搜索结果的情况下的用户的方便。

[0048] 该相关的搜索列表项 314 显示使用在此描述的相关的搜索数据库确定的相关的搜索的 6 个输入 318。在其他实施例中,可显示其他的相关的搜索输入的数量。另外,标记为“更多”的一链接 320 允许用户显示另外的相关的搜索输入。在举例说明的实施例中,所显示的输入 318 是在该相关的搜索数据库中最上面的六个最相关和竞买最多的术语。

[0049] 现在参考图 4,在一个实施例中识别在一相关的搜索数据库中匹配的相关的搜索列表项的动作(图 2,动作 206)包括以下动作。在块 400,包含来自包含在该数据库搜索系统的竞买搜索列表项数据库中的所有网页的所有数据的一倒排索引被搜索。该倒排索引被存储在相关的搜索数据库中。在一倒排索引中,一个索引输入被用来参考多个数据库记录。查找每个索引输入的多个匹配当使用倒排索引时通常会更快,因为每个索引输入可引用多个数据库记录。该倒排索引列举可按如字母顺序搜索的词以及附有的每个词是识别包含该词和在每个文档中出现该词的位置的特定的文档的指针。为执行一搜索,代替用词的顺序搜索完这些文档,该计算机定位用于在一搜索查询中识别的特定词的指针并处理它们。该计算机识别具有对该搜索查询术语来说所要求的顺序以及接近关系的文档。

[0050] 在块 402,也为所接收的搜索项搜索元信息。元信息是抽象的,删除有关所接收的数据本身的一次性移出的信息并形成该数据的一说明。元信息是从信息和相关的信息导出的。用于一列表项的元信息描述该列表项与其他列表项的关系,以及用于一列表项的元信息描述主办一列表项的广告商与其他广告商之间的关系。

[0051] 使用命令的脚本来获得元信息以分析该竞买搜索列表项数据库并确定在该数据中出现的信息和关系。在该数据库中收集用于每行的数据的元信息并将其加到那行上。在一个实施例中,在该数据被收集在该数据库中后,该脚本按一批处理运行一次。在其他实施

例中,该脚本被定期地重新运行以更新元信息。

[0052] 包含在竞买搜索列表项数据库中的有关网页的元信息和关键词包括诸如在不同网页域中相似关键词出现的频率以及与一单个的网页相关的不同关键词的数据的信息。该元信息可进一步包括包含在由在该竞买搜索列表项数据库具有竞买的搜索项的网页创办人提供的每个搜索列表项中的信息、广告商标识信息、网络站点主题如赌博或成人内容、以及导出的主题的字段型的广告商数据。最好,在块 400,该元信息用一普通的倒排索引与所搜索的所存储的网页数据结合。

[0053] 块 400 和块 402 的搜索结果是该倒排索引或包含所搜索的信息索引的行的一列表项。每行包含与该竞买搜索列表项数据库的一搜索列表项相关的信息以及与该搜索列表项相关的网页的所有文字。在举例说明的实施例中,搜索列表项包括该广告商的搜索项、该网页的 URL、一标题以及描述性的文字。

[0054] 在块 404,返回的相关的搜索结果按关联排序。可使用任何适当的排序例程。按关联排序搜索结果的最好的处理在图 5 中更详细地说明。

[0055] 在块 406,可提供六个最相关的有关的搜索结果。注意可提供任何适当的搜索结果数量。建议一搜索者选择提供六个有关的搜索是任意的。在块 406 后,控制进行到图 2 的块 208。

[0056] 图 5 是说明用于按关联排序从一相关的搜索数据库获得的搜索结果的方法的流程图,对应于图 4 的块 404。在如图 5 所述的实施例中,用于每个返回的列表项的一关联值被保持。该关联值根据在图 5 中定义的一些特定的关联因素被调整。也可使用其他关联因素。在调整该关联值后,产生一最终的排序并且最高值的列表项被返回。

[0057] 在块 500,根据在每个各自的记录中一查询的搜索项出现的频率,在搜索过程中用于个别记录的关联值增加(块 400、402,图 4)。例如,如果所查询的搜索项在与该搜索列表项相关的文字中频繁地出现,那个列表项的关联则增加。如果查询的搜索项在该列表项中很少或根本不出现,则那个列表的关联值不增加或减小。

[0058] 在块 502,确定在由该搜索者提交的搜索查询中是否有多个搜索项。如果不是,控制进入块 506。如果有多个搜索项,在块 504,个别搜索结果的关联根据在一定位的记录中的近似的搜索项增加。因此,如果两个搜索项是直接最接近的,可实际上增加用于那个记录的关联比数值,建议所识别的搜索列表项对由该搜索者提交的搜索查询来说很相关。另一方面,如果两个搜索项出现如在一相同的句子中但不是很接近,该记录的关联可稍微增加以表示由搜索项的降低的接近暗示的较低的关联。

[0059] 在块 506,确定所定位的记录是否包含一竞买的搜索项。搜索项是由广告商竞买的,该竞买被用来由使用竞买搜索列表项数据库的搜索引擎网络服务器显示搜索结果。如果搜索结果的确包括一竞买的搜索项,该记录的关联被调整,块 508。如果查询不包括一个或多个竞买的搜索项,控制进入块 510。

[0060] 在块 510,确定在搜索列表项的说明中是否有搜索项。如图 3 所述,每个那样的列表项包括与该搜索列表项的网络站点的内容的文字性说明。如果搜索项没有包括在说明中,控制进入到块 514。如果搜索项包括在该说明中,在块 512,所定位的记录的关联被相应地调整。

[0061] 在块 514,确定搜索项是否定位在搜索列表项的标题中。如图 3 所示,每个搜索列

表项包括一标题 360。如果搜索项包括在一记录的标题中，在块 516，该记录的关联被相应地调整。如果搜索项没有包括在该标题中，控制进入块 518。

[0062] 在块 518，确定搜索项是否包括在搜索列表项的元标签中。元标签是包括在对用户使用来说不显示的一网络站点中的文字性信息。然而，包含在竞买搜索列表项数据库中的搜索列表项包括用于搜索和其他目的的元标签。在块 518，如果搜索项没有包括在搜索列表项中，控制进入块 522。另一方面，如果搜索项包括在一个或多个搜索列表项的元标签中，在块 520，该记录的关联被相应的调整。

[0063] 在块 522，确定用户的搜索项是否包括在竞买的网页的文字中。如果没有，控制进入块 406，图 4。然而，如果搜索项包括在该网页的文字中，在块 524，该搜索列表项记录的关联被相应地调整。

[0064] 在图 5 中所述的步骤之后，一个或多个以及最好是六个最相关的有关的搜索列表项被返回并连同来自该竞买搜索列表项数据库的搜索结果显示给该搜索者。

[0065] 图 6 说明用于形成用在图 1 的数据库搜索系统中的一相关的搜索数据库的一种方法。该方法从块 600 开始。

[0066] 在块 602，提取用于在竞买搜索列表项数据库中所有网页的所有文字。包括元标签和包含由一 URL 引用的该网页中的其他未显示的文字信息，该 URL 包含在竞买搜索列表项数据库中。在块 604，来自相似的页的文字被省略。这减少必须被处理的数据的总量以形成相关的搜索数据库。用于执行该动作的一个方法的一实施例将结合图 7 在下面描述。另外，这大大地增加了产生相关的搜索数据库的速度。在块 606，该文字被存储在相关的搜索数据库中。

[0067] 在块 608，创建一倒排索引，索引在块 606 中存储的搜索列表项数据以及在块 602 提取的文字。最后得到的倒排索引包括多行数据，每行包括一关键词以及来自与那个关键词有磁的数据库的所有文字。

[0068] 用于相关搜索数据库的内容的一结构的一说明性例子如下。该数据库的每行包括以下的部分：

[0069]	canon_ont	integer	# 有关该搜索结果竞买的不同的搜索列表项数量
[0070]			
[0071]	advertiser_cnt	integer	# 有关该搜索结果竞买的不同的广商的数量
[0072]			
[0073]	related_result	varchar(50)	# 相关结果（竞买的搜索项），规范和单数 (canonicalized and depluralized)
[0074]			
[0075]			
[0076]	raw search text	varchar(50)	# 初始行竞买的搜索项
[0077]	advertiser_ids	varchar(4096)	# 对该搜索结果竞买的所有广告商的须直接付款的列表 (explicit list)
[0078]			
[0079]			
[0080]	words	varchar(65536+)	# 搜索 (crawl) 过的所有网页的
[0081]			全文，包括手工编码的脚本
[0082]	theme	varchar(50)	

[0083] directory\_taxonomy varchar(200)

[0084] 数 canon\_cnt 与数 advertiser\_cnt 不同,因为在不同域中的多个不同的网页可竞买 (bid against) 相同的竞买的搜索项,或多个不同的广告商可仅竞买一条搜索项。特定主题的关键词被包含在具有插入在 advertiser\_cnt 字段中的“标记”的数据库中。如果 “advertiser\_cnt = 999999999”,所出现的查询是一面向成人的查询。在该实施方式中,一可选的提高是禁止在该情况下相关的结果。数 canon\_cnt 和 advertiser\_cnt 是当前导出的数据字段。另外的字段如 theme 和 directory\_taxonomy\_category 能任意地添加以给出更多提高的关联给相关结果匹配,尽管他们在所说明的实施例中并没有使用。

[0085] 在一个实施例中,被查询来获得相关的结果的倒排索引用以下 Java 命令创建:

[0086] SQL > Create metamorph inverted index index02on line ad02(words);

[0087] 这是用于在一文档 (在这里,该文档包含在将通过 TexitThunderstone SQL 命令从 (RelatedSearcherCore. java) 被搜索的一数据库列 (词)) 上创建一任何的文字搜索索引 (mm\_index02) 的厂商认证 (vendor specific) 方法 (使用 Thunderstone-EPI, Inc. 提供的 Texit 关系数据库管理系统):

[0088] " SELECT"

[0089] +" \$rank, " //Num getRow ()arg position 0

[0090] +" canon cnt, " //Int getRow ()arg position 1

[0091] +" raw ~ search ~ text, " //Stri getRow ()arg position 2

[0092] +" cannon search text, " //Stri getRow ()arg position 3

[0093] +" advertiser ~ ids, " //1iStri getRow ()arg position 4

[0094] +" advertiser ~ cnt" //Int getRow ()arg position 5

[0095] +" FROM line ad02" +" WHERE words"

[0096] +" LIKEP\$query ORDER BY 1 desc, advertiser cnt desc;" ;

[0097] 基于在该“words”字段中该搜索短语的出现频率、在该索引词字段中查询的短语部分彼此的近似以及与在该“word”字段中与词的次序相比的词的顺序 (如果 1 > 查询短语词 query phrase word, \$rank 是可编程包含该搜索结果的“关联”的一厂商提供的虚拟数据字段,

[0098] 该“rank”是特定、通过由不同的任何文本搜索引擎 (Free TextSearch Engine) 提供者的各种不同的算法导出的卖主,尽管实际上相当相似于任何卖主的任何文本搜索引擎工作来实现相关搜索功能性。

[0099] " ORDER BY 1 desc[ending], advertiser ~ cnt desc[ending]" 首先通过关联,然后通过竞买该特定的 related\_search\_result 的广告商的数量的导出字段 “advertiser\_cnt”控制排列查询结果 (字段”1”== \$rank)。因此,“关联”是主要的选择标准,以及“用户倾向 (popularity)”是次要的选择标准。

[0100] 在块 610,创建另外的索引并与在块 608 创建的倒排索引一起存储。另外的索引使用与每个搜索列表项相关的关键信息创建。该关键信息包括,如字段性的广告商数据如广告商的标识以及导出的主题如赌博等等。然后该方法在块 612 结束。

[0101] 图 7 是说明用于从一数据库移出相似的页面信息的一种方法的流程图。在所描述的实施方式中该方法继续图 6 的动作 602 的运行。

[0102] 在块 702, 竞买搜索列表项数据库用 URL 数据来检查并且所有的 URLs 是从该数据库中抽取出来的并且形成一列表。在块 704, 该列表被存储并且任何精确的副本被移出。

[0103] 在块 706, 在该列表中的一 URL 被选择并且确定所选择的 URL 是否对在该列表中的在前的 URL 具有相似性。相似性可通过任何适当的方法被确定, 如在该 URL 中多个相同的字符或字段或相同字符的百分比, 或一共同的根或串或字段。

[0104] 在块 708, 如果选择的 URL 与在前的 URL 相似, 所选择的 URL 被添加到一候选的完全相同的 URLs 的列表。在块 710, 搜索 (crawl) 多个预定的每个可能的完全一样的 URL。在所描述的实施例中, 预定数量首先是两个可能的完全相同的 URLs。搜索 (crawl) 最好使用称为一搜索 (crawl) 者的一程序代码来实现。一搜索 (crawl) 者是访问网络站点和读取它们的页面和其他信息的一程序。这种程序是公知的并且也被称为“spider”或 bot”。可选择性地访问和由一搜索 (crawl) 者索引整个站点或特定的页面。在另一实施例中, 由一 URL 引用的每个站点的子集而不是整个网点可被搜索 (crawl) 并比较相似性。

[0105] 在块 712, 由搜索 (crawl) 者返回的数据被检查。该数据可被称为 URL 体并包括来任何该 URL 标识的站点和所有可访问的站点的页面的数据。确定包括包含在 URL 体中的文字和其他信息的数据与包含在在前的 URL 体中的数据是否十分地相似。此外, 可用任何适当的方法如每个页面的文字内容的 Stats 比较确定相似性。如果十分相似, 控制进入块 714 并假定该 URL 与在前的 URL 相同。文字和其他信息体被指定到该相似的 URLs 的剩余部分。

[0106] 如果在块 706 确定所选择的 URL 与在前的 URL 不相似或如果在块 712, 确定该 URL 体与在前的 URL 体不是十分相似, 控制进入块 718。在块 718, 该 URL 被添加到将被搜索 (crawl) 的 URLs 的一列表。在块 720,, 在该列表上的所有 URLs 被搜索 (crawl) 以检索和存储包含在由每个 URL 表示的站点上的信息。

[0107] 在块 716, 来自每个搜索 (crawl) 过的 URL 的信息被加载到相关的搜索数据库 (也称为任意文字数据库 (free text database))。该信息与已经包括在相关的搜索数据库中的搜索列表项数据结合。因此, 在图 7 中描述的方法步骤通过降低被搜索 (crawl) 和存储的 URLs 的数量来降低包含在相关的搜索数据库中的数据总量。完全相同的 URLs (duplicateURLs) 被从该过程中消除并且相近的完全相同品 (near duplicate) URLs 用内容的相似性被核查。结果是降低对该最终结果的数据库的存储要求以及在该数据库上更快、更有效的查询。通过提高性能提高用户的方便。

[0108] 从以上所述, 可以看出本发明提供用于产生呈现给一搜索者在一竞买搜索列表项数据库中搜索的相关的搜索的一种改进的方法和装置。相关的搜索在使用该竞买搜索列表项数据库形成的一相关的搜索数据库中被执行。来自该相关的搜索者的数据库的搜索结果按关联排序用于呈现给该用户。因此, 如果一用户的最初的搜索太窄或太宽, 该用户具有可被用来产生更可用的结果的有效相关的搜索。另外, 该相关的搜索已经使用按竞买的搜索项引用的搜索列表项产生。这有利于对在该数据库搜索系统中的广告付费的广告商。这增加了由一搜索者使用该数据库系统访问一广告商的网络站点的可能性。

[0109] 同时本发明的一特定的实施例已经示出并描述, 也可做修改。因此, 覆盖所有这种变化和修改的附加的权利要求落在本发明的精神和范围内。

[0110] 源代码附录

[0111] 这实际上是执行来自任何文字搜索引擎 (任何文本搜索引擎 (FreeText Search

Engine)) (来自的 Thunderstone-EPI, Inc. 的 TexitRDBMS) 核心的 Java 部分并且后处理该结果, 过滤有关广告商出现的频率和主题 (“adult”-ness)

```
[0112] package com.go2.search.related; import java.util.Vector;
[0113] import java.util.Hashtable;
[0114] import java.util.StringTokenizer;
[0115] import java.rmi.RemoteException;
[0116] import com.go2.texit.*;
[0117] /***&commat ;author Phil Rorex*&commat ;version*/
[0118] class Callback implements TErrorMsgIF{private static int err115 = 0;
public int getErr115() {return(err115);} public void ErrorMsgDelivery(String msg, int level, int msgNumber) {switch(msgNumber) {case 2:
[0119] System.out.println (" FATAL:msg:" +msg+" level:" +level+" msgNumber:
" +msgNumber);
[0120] System.exit(2); } case 100:break; case 115:err115++;break; default:
[0121] System.out.println(" UNUSUAL:msg:" +msg+" level:" +level+" msgNumber:
" +msgNumber);}}
[0122] /**/**run as a stand-alone JVM, since the Free TextSearcher*being used
is best connected with as a JNI-based Clanguage按一单机的 JVM运行, 因为所使用的
任何文字搜索器最好与作为一基于 C 语言的 JNI 库接口 API 连接。
[0123] **/public class RelatedSearcherCore implements Runnable{
[0124] //cache an instance of Texit server and Query 高速缓存 Texit 服务器和查
询的一个实例
[0125] private static Server texit = null;
[0126] private static Query texitQuery = null;
[0127] private static Query texitPlurQuery = null;
[0128] private static Query texitAdultQuery = null ;
[0129] private static long timeClock;
[0130] //used to coordinate time-outs on extra long queries用来调整有关特别长
的查询的超时
[0131] private static final Integer PRE ~ QUERY = new Integer(1);
[0132] private static final Integer MID ~ QUERY = new Integer(2);
[0133] private static final Integer POST ~ QUERY = new Integer(3);
[0134] private static Integer semaphore = PRE ~ QUERY;
[0135] //time out process 超时处理
[0136] Thread watchDog;
[0137] // 线程开始等待如果 Core() 没有超过为它所设置的超时可能永不会被使用的很
长时间
[0138] long globalTimeOut = 0;
[0139] //The magic adult flag 不可思议的成人标记
```

[0140] //If a related search free-text search returns a row //which has this field set, it's automatically " themed " //as an adult-oriented related search//This particular " Magic datarow " is pre-loaded with//all the " adult-oriented" terms which are typical//in this theme. Same should be done for CASINO FLAG//CURRENT ~ NEWS ~ FLAG, and any other theme desired. 如果一相关的搜索任何的文字搜索返回具有该字段组的一行,它被自动地“主题化”为一面向成人的相关的搜索。该特定的“不可思议的数据行”是用在该主题内很典型的所有“面向成人的”术语预载。对 CASINO FLAG、CURRENT ~ NEWS ~ FLAG 和所要求的其他主题也应当执行相同的处理

```

[0141] private static final int ADULT ~ FLAG = 999999999 ;
[0142] //How many pluralized tries of the query//used to search for singular and plural//version of up to[square root of]MAX_PLURAL_QRY//terms 多少次查询尝试用来查询单一的和达到 MAX_PLURAL_QRY 的 [ 平方根 ] 的多个版本
[0143] private static final int MAX_PLURAL_QRY = 4 ;
[0144] //Limit Texis to this many rows 将 Texis 限定到多个行
[0145] //This is the initial#of pre-filtered free-text//searched rows coming back from the search engine 这是初始的从该搜索引擎返回的预先过滤的任何文字搜索的行
[0146] private static final int MARROWS = 60 ;//
[0147] //controls the ' looseness ' of the post-search filter// that filters out related searches based on the//derived-data element of (#-of-different-advertisers//bidding on this related search term). Set to 0 is//this element means " how many times we can ignore seeing//the identical advertiser before we start ignoring//related searches bid on by him " 0 is strongest reject, //larger numbers reject less stringently (usu. not > than//1, if ratio of webpages:related search terms is > than//about 10 控制后搜索过滤器的“压缩”, 该过滤器基于 ( 在相关的搜索项上竞买的不同的广告商的 #) 导出的数据部分过滤出相关的搜索。设置为 0, 表示在我们开始忽略由广告商竞买的相关的搜索前我们可忽略多少次查看相同的广告商, 0 是最强烈地拒绝, 更大数量的拒绝不是很严格 ( 如果网页 : 相关的搜索项的比率大于 10, 则 usu. not > 1 )
[0148] private static final int ADVERTISER_THRESHOLD = 0 ;
[0149] //the SQL query used to talk to Texis(the FTS engine) 用于与 Texis(FTS 引擎) 对话的 SQL 查询
[0150] private static final String TEXIS_SQL = " SELECT " + !$rank, " // Num getRow ()arg position 0+ " canon_cnt, " //Int getRow ()arg position 1+ " rawsearchtext, " //Stri getRowf ()arg position 2+ " cannonsearchtext, " //Stri getRow ()arg position 3+ " advertiser_ids, " //Stri getRow ()arg position 4+ " advertiser_cnt" //Int getRow ()arg position 5+ " FROM linead02" +" WHERE words" +" LIKEP ? ORDER BY 1 desc, advertiser_cnt desc ; " ;

```

```
[0151] //+" LIKEP ? ; " ;
[0152] //+" LIKEP ? ORDER BY advertiser cnt desc ;按 advertiser_cnt 的降序排序" ;
[0153] private static final String TEXIS_PLUR_SQL = " SELECTplural" +" FROM
plurals " + " WHERE singular " +tt = . ? ;111 private static final String
TEXISADULTSQL = " SELECTcannon_search_text " + " FROM adult " + " WHERE
words" +" LIKE ? ; " ;
[0154] private static Callback cb = new Callback() ;
[0155] public void init(String texisHome, long timeOut){globalTimeOut =
timeOut ;init(texisHome) ;}
[0156] public void init(String texisHome){/**
[0157] *Instantiate Taxis connection object and perform*Taxisquery
initialization*示例 Taxis 连接对象以及执行 Taxis 查询初始化
[0158] *Called one time to setup the Related Search query. 调用一次来安装相关的
搜索查询
[0159] *Must be called before findRelate is ever called. 总是在调用 findRelate
前必须被调用
[0160] *//Perform Taxis initialization and precache an instance//of Taxis
Server and Taxis Query 执行 Taxis 初始化以及预先高速缓冲 Taxis 服务和 Taxis 查询
的一实例
[0161] try{
[0162] Taxis texisRDBMS = new Taxis() ;texis = (Server)texisRDBMS.
createServer(texisHome) ;
[0163] Vector n = new Vector(200) ;//Vector n = texis.getNoise() ;
n.addElement( " a " ) ;n.addElement( " about " ) ;n.addElement( " after " ) ;
n.addElement( " again " ) ;n.addElement( " ago " ) ;n.addElement( " all " ) ;
n.addElement(" almost" ) ;n.addElement(" also" ) ;n.addElement(" always" ) ;
n.addElement( " am " ) ;n.addElement( " an " ) ;n.addElement( " and " ) ;
n.addElement(" another" ) ;n.addElement(" any" ) ;n.addElement(" anybody" ) ;
n.addElement( " anyhow " ) ;n.addElement( " anyone " ) ;n.addElement(
" anything " ) ;n.addElement( " anyway " ) ;n.addElement( " are " ) ;
n.addElement( " as " ) ;n.addElement( " at " ) ;n.addElement( " away " ) ;
n.addElement(" be" ) ;n.addElement(" became" ) ;n.addElement(" because" ) ;
n.addElement(" been" ) ;n.addElement(" before" ) ;n.addElement(" being" ) ;
n.addElement( " but " ) ;n.addElement( " by " ) ;n.addElement( " came " ) ;
n.addElement( " can " ) ;n.addElement( " cannot " ) ;n.addElement( " com " ) ;
n.addElement(" come" ) ;n.addElement( " could" ) ;n.addElement( " de" ) ;
n.addElement( " del " ) ;n.addElement( " der " ) ;n.addElement( " did " ) ;
n.addElement( " do " ) ;n.addElement( " does " ) ;n.addElement( " doing " ) ;
```

```
n.addElement( " done " );n.addElement( " down " );n.addElement( " each " );
n.addElement( " else " );n.addElement( " even " );n.addElement( " ever " );
n.addElement( " everything " );n.addElement( " for " );
n.addElement( " from " );n.addElement( " front " );n.addElement( " get " );
n.addElement( " getting " );n.addElement( " go " );n.addElement( " goes " );
n.addElement( " going " );n.addElement( " gone " );n.addElement( " got " );
n.addElement( " gotten " );n.addElement( " had " );n.addElement( " has " );
n.addElement( " have " );n.addElement( " having " );n.addElement( " he " );
n.addElement( " her " );n.addElement( " here " );n.addElement( " him " );
n.addElement( " his " );n.addElement( " how " );n.addElement( " i " );
n.addElement( " if " );n.addElement( " in " );n.addElement( " into " );
n.addElement( " is " );n.addElement( " isn't " );n.addElement( " it " );
n.addElement( " jpg " );n.addElement( " just " );n.addElement( " last " );
n.addElement( " least " );n.addElement( " left " );n.addElement( " less " );
n.addElement( " let " );n.addElement( " like " );n.addElement( " make " );
n.addElement( " many " );n.addElement( " may " );n.addElement( " maybe " );
n.addElement( " me " );n.addElement( " mine " );n.addElement( " more " );
n.addElement( " most " );n.addElement( " much " );n.addElement( " my " );
n.addElement( " myself " );n.addElement( " net " );n.addElement( " never " );
n.addElement( " no " );n.addElement( " none " );n.addElement( " not " );
n.addElement( " now " );n.addElement( " of " );n.addElement( " off " );
n.addElement( " on " );n.addElement( " one " );n.addElement( " onto " );
n.addElement( " org " );n.addElement( " our " );n.addElement( " ourselves " );
n.addElement( " out " );n.addElement( " over " );n.addElement( " per " );
n.addElement( " put " );n.addElement( " putting " );n.addElement( " same " );
n.addElement(" saw11");n.addElement( " see " );n.addElement( " seen " );
n.addElement( " shall " );n.addElement( " she " );n.addElement( " should " );
n.addElement( " so " );n.addElement( " some " );n.addElement( " somebody " );
n.addElement( " someone " );n.addElement( " something " );
n.addElement( " stand " );n.addElement( " such " );n.addElement( " sure " );
n.addElement( " take " );n.addElement( " than " );n.addElement( " that " );
n.addElement( " the " );n.addElement( " their " );n.addElement( " them " );
n.addElement( " then " );n.addElement( " there " );n.addElement( " these " );
n.addElement( " they " );n.addElement( " this " );n.addElement( " those " );
n.addElement( " through " );n.addElement( " till " );n.addElement( " to " );
n.addElement( " too " );n.addElement( " two " );n.addElement( " unless " );
n.addElement( " until " );n.addElement( " up " );n.addElement( " upon " );
n.addElement( " us " );n.addElement( " very " );n.addElement( " was " );
```

```

n.addElement( " we " ) ;n.addElement( " went " ) ;n.addElement( " were " ) ;
n.addElement(" what" ) ;n.addElement(" what's" ) ;n.addElement(" whatever" ) ;
n.addElement(" when" ) ;n.addElement(" where" ) ;n.addElement(" whether" ) ;
n.addElement( " which " ) ;n.addElement( " while " ) ;n.addElement( " who " ) ;
n.addElement( " whoever" ) ;n.addElement( " whom" ) ;n.addElement( " whose" ) ;
n.addElement( " why " ) ;n.addElement( " will " ) ;n.addElement( " with " ) ;
n.addElement( " within" ) ;n.addElement( " without" ) ;n.addElement("won' t' l) ;
n.addElement( " would" ) ;n.addElement( " wouldn't" ) ;n.addElement(" www" ) ;
n.addElement( " yet" ) ;n.addElement( " you" ) ;n.addElement( " your" ) ;texis.
setNoise(n) ;texisQuery = (Query)texis.createQuery() ;texisPlurQuery = (Query)
texis.createQuery() ;texisAdultQuery = (Query)texis.createQuery() ;

```

[0164] /\*Query.api()'s affect ALL queries,not just ones seton Query.api() 受所有查询的影响,而不仅是确定的查询

```

[0165] /*texisQuery.setlikeprows(MARROWS) ;texisQuery.allinear(0) ;
texi sQuery.alpostproc(0) ;texi sQuery.prepSQL(TEXISSQL) ;texisPlurQuery.
prepSQL(TEXIS_PLUR_SQL) ;texisAdultQuery.prepSQL(TEXISADULTSQL) ;

```

```

[0166] TErrorMsg.RegisterMsgDelivery(cb) ;watchDog = new Thread(this) ;
watchDog.setPriority(Thread.NORM_PRIORITY+1) ;watchDog.start() ;
catch(TException te){te.printStackTrace() ;throw new RuntimeException(" Could
not initialize Texis:Failed with: " +te.getMsg()+code: " +te.getErrorCode()) ;
catch(RemoteExceptionre){throw new RuntimeException( " Unexpected
RemoteException:+re) ;}}/**/

```

[0167] \*Perform a Texis related search query and package results(if any)\*into an array of RelatedResults objects 执行一Texis相关的搜索查询以及将结果(若有的话) 打包成RelatedResult对象的一数组

```

[0168] /*public RelatedResult[] findRelated(String rawQuery, String
canonQuery, int maxResults, int maxResultLength) throwsException{try{return
findRelated(rawQuery, canonQuery, maxResults, maxResultLength, 2000) ;
catch(Exception e){e.printStackTrace() ;throw new Exception( " overloadedf
indRelated " +e.getMessage()) ;}}public RelatedResult[] findRelated(String
rawQuery, String canonQuery, int maxResults, int maxResultLength, long timeOut)
throwsRelatedSearchException{//local vars

```

[0169] Vector resultVector = new Vector() ;

[0170] Vector thisRow = new Vector() ;

[0171] RelatedResult[] results = null ;int resultCount = 0 ;int rank = 0 ;int
canonCnt = 0 ;

[0172] Integer advertiser\_id = null ;int advertiserCnt = 0 ;

[0173] String advertiserIds = null ;

[0174] String rawSearchText = null ;

```
[0175] String canonSearchText = null ;
[0176] Runtime rt = Runtime.getRuntime() ;long mem = rt.totalMemory() ;
long free = rt.freeMemory() ;//System.out.println( " totalMemory(): " +mem) ;
//System.out.println( " freeMemory(): " +free) ;try{if(canonQuery == null)
return(null) ;
[0177] Vector queryArgs = new Vector() ;
[0178] String newQuery ;if(timeOut != 0) {
[0179] //get the loop out of wait() mode 使该循环离开 wait() 模式
[0180] synchronized(watchDog){globalTimeOut = timeOut ;//System.out.println
//System.currentTimeMillis()//+//+timeClock//+//+(System.currentTimeMillis()
()-timeClock)//+" Core :setting MID_QUERY" ) ;semaphore=MID_QUERY ;timeClock
= System.currentTimeMillis() ;
[0181] //thread better be waiting on eternity 线程最好永远等待 watchDog.
notify() ;}
[0182] //if no Raw Query,probably have canonical ized version only 如果没有行
查询,最好仅具有规范的版本
[0183] if(rawQuery != null) {
[0184] //usual serving site,don't re-pluralize,just use //rawquery 普通的服务
站点,不是多数,正好使用行查询
[0185] ;newQuery = stripNoiseChars(rawQuery) ;}else{
[0186] //only have a canonQuery to work with,so make //a roughapproximation of
a raw term to include in search //try andgenerate queries to cover up to MAX_
PLURAL_QRY possible//re-plural ized forms of the query 仅具有一 canonQuery 来工
作,因此做出一行术语的粗略的近似以包括在查询尝试中并生成查询来覆盖该查询的重新
使成复数形成的 MAX_PLURAL_QRY
[0187] newQuery = pluralize(stripNoiseChars(canonQuery)) ;}//if(newQuery ==
null) return null ;if(isAdult(newQuery)) returnnull ;
[0188] //Set up the(stack allocated)query parameters 建立(分配堆栈的)查询参
数
[0189] queryArgs.removeAllElements() ;queryArgs.addElement(newQuery) ;
[0190] //performJNI calls here 在这里执行 JNI 调用
[0191] texisQuery.setParam(queryArgs) ;texisQuery.execSQL() ;
[0192] //Iterate over the rows 在行上重复
[0193] String lastCanon = rawQuery ;
[0194] Hashtable advertisers = new Hashtable(MAX_ROWS*200) ;
[0195] Hashtable used = new Hashtable(MXX_ROWS)
[0196] Vector resultSet = texisQuery.getRows() ;//VectorresultSet =
getRowsLocal() ;
[0197] //make 2passes. 做 2 遍
```

```

[0198] //first time de-dup on advertisers 第一次不复制有关广告商
[0199] //second time don't dedup 第二次打开
[0200] //System.out.println(" got rows: " +resultSet.size());for(int pass
= 0;pass < 2;pass++) {if(resultCount > = maxResults), break;for(int i =
0;i < resultSet.size();i++) {thisRow = (Vector)resultSet.elementAt(i);
if(thisRow.size() == 6){rank = ((Number)(thisRow.elementAt(0))).intValue();
//System.out.println(thisRow.elementAt(0).getClass().toString());canonCnt =
((Integer)(thisRow.elementAt(1))).intValue();rawSearchText = (String)thisRow.
elementAt(2);canonSearchText = (String)thisRow.elementAt(3);advertiserIds =
(String)thisRow.elementAt(4);advertiserCnt = ((Integer)thisRow.elementAt(5)).
intValue();
[0201] //Drop out early if we detect magic ADULT_FLAG 如果检测到不可思议的
ADULT_FLAG,先离开
[0202] if(advertiserCnt == ADULT_FLAG) return null;if(canonCnt == ADULT
FLAG) return null;if(false) {
[0203] System.out.println(" rank: " +rank + " cnt: " +canonCnt + " rst
: " +rawSearchText + " cst : " +canonSearchText + " aids : " +advertiserIds
+ " adcnt: " +advertiserCnt);}else (throw newRelatedSearchException(" Texit
query failed, protocolviolation" );)
[0204] //De-dup the results, and also don't return a related//search term which
canonically matches the original query 不完全相同该结果,并且也不返回规范地匹配
最初查询的一相关的搜索项
[0205] if(( ! canonSearchText.equalsIgnoreCase(rawQuery))&&( !
canonSearchText.equals(canonQuery))&&( ! rawSearchText.
equalsIgnoreCase(rawQuery))&&( ! rawSearchText.equalsIgnoreCase(canonQuery))
&&(canonSearchText.length() < = maxResultLength)){//System.out.println(" got
cst:" +canonSearchText) ;
[0206] //look for this advertiser in the hash table 在散列表中查询该广告商
[0207] //if there, increment occurrences count //and if above threshold, we've
seen enough //terms suggested by this advertiser, so go to //next term //if not
seen this advertiser yet, put it in the //hashtable and process 如果有,递增出现
数并且如果超出阀值,我们可以看到由该广告商建议的很多的术语,因此到另一术语,如果
未看到该广告商,将其放入该散列表和过程中
[0208] StringTokenizer st = new StringTokenizer(advertiserIds, " ")i //
if(st.countTokens() != advertiserCnt){//System.out.println(" toks:" //+st.co
untTokens()//+" Cnt:" //+advertiserCnt); //throw newRelatedSearchException("//"
Texit query suspect, wrongadvertiser count" );//}if(pass == 0){int dupAdvCnt
= 0;boolean Next = false;
[0209] //Parse all the advertiserID's out of the returned row 分析不在返回行

```

的所有广告商的 ID

```
[0210] while(st.hasMoreTokens()) {  
[0211] Integer advertiserId = Integer.valueOf(st.nextToken()); //if( !  
advertisers.containsKey(advertiserId))  
[0212] //if this advertiser is new to us (over whole query)//putin the hash如果该广告商对我们来说是新的（在整个查询上），放入散列中  
[0213] Integer cnt = (Integer)advertisers.get(advertiserId); if(cnt == null)  
(advertisers.put(advertiserId, new Integer(0)); else {  
[0214] //Seen this advertiser before, so increment his//tally以前见过该广告商，  
因此递增其计数  
[0215] advertisers.put(advertiserId, new  
[0216] Integer(cnt.intValue() + 1)); //System.out.println(advertiserId+ " dups  
:" +(cnt.intValue() + 1));  
[0217] //If he's (now) past the threshhold, don't use//biddedterm(yet) 如果他  
(现在)超过该阀值，不使用竞买的术语  
[0218] if(cnt.intValue() >= ADVERTISER_THRESHOLD) {  
[0219] Next = true; break;} dupAdvCnt++; } } if(Next == true) {continue;}  
else{if( ! used.containsKey(canonSearchText)) {used.put(canonSearchText,  
new Boolean(true));}} else{if( ! used.containsKey(canonSearchText)) {used.  
put(canonSearchText, new Boolean(true));} else{continue;}} //if (dupAdvCnt >=  
ADVERTISER_THRESHOLD) {  
[0220] //continue ;继续  
[0221] //System.out.println(" dups :" +dupAdvCnt); //}/**if(pass == 0){  
[0222] //first time thru see if we've used this advertiser第一次，通过查看我  
们是否已经使用该广告商  
[0223] Integer cnt = (Integer)advertisers.get(advertiser_id); if(cnt ==  
null) (advertisers.put(advertiser_id, new Integer(0)); else{advertisers.  
put(advertiser_id, new  
[0224] Integer(cnt.intValue() + 1)); if(cnt.intValue() >= ADVERTISER_  
THRESHOLD) {cont inue; } if( ! used.containsKey(canonSearchText)) {used.  
put(canonSearchText, newBoolean(true));} else{//this is a second(or more)time  
thru.  
[0225] //see if we've already used this term这是第一次（或更多）次通过查看我  
们是否已经使用过该术语  
[0226] if( ! used.containsKey(canonSearchText)) {used.put(canonSearchText, new  
Boolean(true));} else{continue;}} **/if(resultCount < maxResults) {resultVector.  
addElement(new RelatedResult(rawSearchText,  
[0227] RelatedResult.NON_CACHED)); resultCount++; } else{break;} //if-else//  
if//for}//for) catch(TException te) {throw new RelatedSearchException
```



```

startsWith( " www" )term = term.substring(3 ;} } } } } switch(term.length()) { case
0 : case 1 : return( null ) ; default : { switch(term.charAt(term.length() - 1)) { case ('m')
: case ('M') : case ('t') : case ('T') : case ('g') : case ('G') : case ('f') : case ('F') : {
[0238]   String lowerTerm = term.toLowerCase() ; if(lowerTerm.endsWith( " dot
com" ))term = term.length() > 8 ? term.substring(0, term.length() - 8) : null ; else
if(lowerTerm.endsWith(" dotcom" ))term = term.length() >
[0239]   7 ? term.substring(0, term.length() - 7) : null ; else if(lowerTerm.
endsWith( " com" ))term = term.length() > 4 ? term.substring(0, term.
length() - 4) : null ; else if(lowerTerm.endsWith( " net" ))term = term.
length() > 4 ? term.substring(0, term.length() - 4) : null ; elseif(lowerTerm.
endsWith( " org" ))term = term.length() > 4 ? term.substring(0, term.
length() - 4) : null ; else if(lowerTerm.endsWith( " gif" ))term = term.
length() > 4 ? term.substring(0, term.length() - 4) : null ; else if(lowerTerm.
endsWith( " jpg" ))term = term.length() > 4 ? term.substring(0, term.
length() - 4) : null ; } } } } } //Debug: System.out.println( " term:[ " +term.trim()
zur return(term == null ? null : term.trim()) ; } private boolean isAdult(String
query) throws RelatedSearchException { if(query == null) return false ;
[0240]   Vector queryArgs = new Vector() ;
[0241]   Vector thisRow = new Vector() ; queryArgs.addElement(query) ; try {
[0242]     //perform JNI calls 执行 JNI 调用
[0243]     texisAdultQuery.setParam(queryArgs) ; texisAdultQuery.execSQL() ;
if((thisRow = texisAdultQuery.getRow()).size() != 0) { return(true) ; }
else { return(false) ; } catch(TException te) { throw new RelatedSearchException(
" Texisinterface failed with: " +te.getMsg(), te) ; } catch(RemoteException
re) { throw new RelatedSearchException( " Got a RemoteException that should
never occur: " +re) ; } } private String pluralize(String token) throws
RelatedSearchException { if(token == null) return null ;
[0244]   Vector queryArgs = new Vector() ;
[0245]   String pluralToken ;
[0246]   Vector thisRow = new Vector() ;
[0247]   StringTokenizer st0 = new StringTokenizer(token, " " ) ;
[0248]   String[] terms = new String[st0.countTokens()] ;
[0249]   String[] fullQuery = new String[MAX_PLURAL_QRY] ; int fullQueryCnt = 0 ;
[0250]   //Iterate over each token to see if there's a plural version 使每个重
复来查看是否多个版本
[0251]   for(int ele0 = 0 ; st0.hasMoreTokens() ; ele0++) { terms[ele0] = st0.
nextToken() ; for(int element = 0 ; element < terms.length && fullQueryCnt <
[0252]   MAX_PLURAL_QRY ; element++) {
[0253]     //Do plurals lookup on this term from texis db 从该Texis数据库执行多次

```

有关该术语的查找

```
[0254]     queryArgs.removeAllElements() ;queryArgs.addElement(terms[element]) ;
try {
[0255]     //perform JNI calls 执行 JNI 调用
[0256]     texisPlurQuery.setParam(queryArgs) ;texisPlurQuery.execSQL() ;
[0257]     //retrieve the row 检索该行
[0258]     if((thisRow = texisPlurQuery.getRow(##.size(# ! = 0#{
[0259]         String term = null ;
[0260]         //loop thru the terms 循环完术语
[0261]         for(int ele1 = 0 ;ele1 < terms.length ;ele1++) {if(ele1 == element)
(if(ele1 == 0) {term = (String)(thisRow.elementAt(0)) ;} else {term+
= " " +(String)(thisRow.elementAt(0)) ;}} else {if(ele1 == 0) term =
terms[ele1] ;else term+ = " " +terms[ele1] ;}} fullQuery[fullQueryCnt] = term ;
fullQueryCnt++ ;}) catch(TException te) {throw new RelatedSearchException(" Texi
sinterface failed with:+te.getMsg() , te) ;} catch(RemoteException re) {throw new
RelatedSearchException(" Got a
[0262]     RemoteException that should never occur:" +re) ;})
[0263]     //Build the new expanded query 连编该新扩展的查询
[0264]     if(fullQueryCnt > 0) {pluralToken = " (" +token ;for(int i = 0 ;
i < fullQueryCnt ;i++) {pluralToken = pluralToken + " , " +fullQuery[i] ;}
pluralToken = pluralToken+ " ) " } else {pluralToken = token ;}
return(pluralToken) ;} public Vector getRowsLocal() throws TException,
RemoteException {
[0265]     Vector set = new Vector() ;int e ;synchronized(APIToken.Lock)
{while(true) {
[0266]         Vector row = new Vector() ;row = texisQuery.getRow() ;if(row.size() ==
0) break ;set.addElement(row) ;} return set ;} public synchronized void run()
{while(true) {try {
[0267]             //start our timeout 开始我们的超时
[0268]             synchronized(watchDog) { //System.err.println("//System.
currentTimeMillis()///+timeClock ///+(System.currentTimeMillis()-timeClock
)//+" run:starting wait of" //+globalTimeOut) ;watchDog.wait(globalTimeOut) ;
[0269]             //just got woke up, //see why 正好被唤醒,查看为什么
[0270]             if(semaphore.equals(PRE-QUERY)) { //System.err.println("//System.
currentTimeMillis()//+" - " //+timeClock//+" = " //+(System.currentTimeMillis-
timeClock)//+" run:got PRE_QUERY " ) ;continue ;} elseif(semaphore.
equals(POST_QUERY)) { //System.err.println("//System.currentTimeMillis()//+
/+timeClock//+" = " //+(System.currentTimeMillis()-timeClock)//+" run:go
t POST_QUERY " ) ;continue ;} else if(semaphore.equals(MID-QUERY)) {if (System.
```

```
currentTimeMillis() - timeClock >= globalTimeOut) {  
[0271] // we timed out, but semaphore wasn't set, so hoseourselves 我们时间到了, 但没有设置信号标志, 因此我们自己停止  
[0272] System.out.println(  
[0273] System.currentTimeMillis() + timeClock + (System.currentTimeMillis() - timeClock) + " Fatal: timeout " + globalTimeOut + " usec exceeded" );  
[0274] System.exit(1); } else { // System.out.println("// System.currentTimeMillis() //++/+timeClock//+++(System.currentTimeMillis() - timeClock)//+" run:gotMID-QUERY, but OK ! " ); } } else  
[0275] System.out.println(  
[0276] System.currentTimeMillis() + timeClock + (System.currentTimeMillis() - timeClock) + " run:ARGHH got no_QUERY, Hmmmm ! " ); } } catch (Exception e) {  
[0277] System.out.println(" got wait() exception" ); }  
[0278] The following code is used to implement the cached resultslookup, first to see if we've seen this related search before, to save time and not do the algorithmic lookup during the relatedsearch execution. 以下的代码被用来实现高速缓冲结果的查找, 首先查看我们以前是否已经看出过该相关的搜索结果以节约时间而且在相关搜索执行期间不执行算法查找  
[0279] package com.go2.search.related; // import atg.nucleus.  
GenericRMIService; import atg.nucleus.GenericService; import atg.nucleus.  
ServiceException; import atg.service.resourcepool.JDBCConnectionPool; import  
atg.service.resourcepool.ResourceObject; import atg.service.resourcepool.  
ResourcePoolException; import java.rmi.RemoteException; import java.net.*;  
import java.io.*; import java.sql.*; import java.util.Vector;  
[0280] /** This is the top level interface to the related search*system it is  
meant to be used as a dynamo service available*to other dynamo services**/ 这是  
对该相关搜索系统来说最高级的接口, 它表示可被用作对其他动态服务来说可用的一动态  
服务 // public class RelatedSearcherImpl extends GenericRMIService public class  
RelatedSearcherImpl extends GenericService implements RelatedSearcher { // my pool  
of Texis/UDP 我的 Texis/UDP 库  
[0281] private TexisUDPConnectionPool texisUDPConnectionPool; // my pool of  
connections to Oracle cache 我的连接到 Oracle 高速缓冲存储器上的库  
[0282] private JDBCConnectionPool relatedCacheConnectionPool; // Statistics  
properties Stats 学特性  
[0283] private int requestCount = 0; private int oracleCacheHits = 0;  
private int texisRequests = 0; private int texisTimeoutMillis = 0; private int  
slowTexisRequestCount = 0; // private constants 专用常数  
[0284] private static String CACHE_SQL = " SELECT*FROM RESEARCH  
[0285] WHERE CANON_QUERY =? " ; private static int BUFFER_SIZE = 512; //
```

parameters 参数

[0286] private boolean texisEnabled = false ;private boolean oracleEnabled = false ;private boolean systemEnabled = false ;private long cummulativeOracleTime = 0 ;private long cumulativeTexisTime = 0 ;

[0287] /\*\*Create and export and instance of RelatedSearcher overRMI 在 RMI 上 RelatedSearcher 的产生以及输出以及实例

[0288] \*/public RelatedSearcherImpl () throws RemoteException{super () ;//java. rmi. registry. LocateRegistry. createRegistry(1111). rebind(" RelatedSearcher " , this) ;}/\*\*

[0289] \*This method was created in VisualAge. 该方法用 VisualAge 创建

[0290] \*&commat ;return RelatedResult []\*&commat ;param canonQuery java. lang. String\*&commat ;param maxResults int\*&commat ;param maxLength int \*/private RelatedResult[] findFromCache(String canonQuery, int maxResults, int maxLength) throws RelatedSearchException(

[0291] Vector resultVector = new Vector() ;

[0292] RelatedResult[] results = null ;

[0293] PreparedStatement ps = null ;

[0294] ResultSet rs = null ;try{//Get a Connection 得到一连接

[0295] ResourceObject resource = null ;try{.resource = getRelatedCacheConnectionPool().checkout(getAbsoluteName()) ;

[0296] Connection conn = (Connection)resource.getResource() boolean success = false ;try{//Here's where we get the goods from Oracle 这里我们可从 Oracle 获得商品

[0297] ps = conn.prepareStatement(CACHE\_SQL) ;ps.setString(1, canonQuery) ;rs = ps.executeQuery() ;//prime the cursor topoint to the one and only row we // expect from Oracle if no matching rows were found //then we'll simply drop thru to the end 最初将光标指向我们期望的来自 Oracle 的一个或仅一行, 如果现有匹配行已经找到, 那么我们将简单地落到结尾

[0298] if(rs.next()){//Extract the data we need if there was something如果有, 抽取我们需要的数据

[0299] int numTerms = rs.getInt(2) ;if(numTerms == 0)//The cache tells us that there won't be //any results so we'll bail early throw new 高速缓存器告诉我们没有任何结果, 因此我们将先委托扔出新的

[0300] RelatedSearchException( " No related Results " ) ;int cacheFlag = rs.getInt(3) ;//iterate over results retrieving up to maxResults //of those of them that are maxLength or smaller 重复有关是 maxLength 或更小的那些 maxResults 检索的结果

[0301] int resultCount = 0 , rowCount = 0 ;while(resultCount < maxResults&&rowCount < numTerms) {

```
[0302] String term=rs.getString(4+rowCount); //push this term into the result vector if its good 将该术语推入该结果矢量中,如果它的商品
[0303] if(term.length() <= maxLength) {resultVector.addElement(new
[0304] RelatedResult(term, cacheFlag)); resultCount++; rowCount++;} } conn.
commit(); success = true;} finally{//Cleanup result set 整理结果集合
[0305] if(rs != null) rs.close(); //Cleanup prepared statement 整理准备的语句
[0306] if(ps != null) ps.close(); //Cleanup connection 整理连接
[0307] if(! success && conn != null) conn.rollback(); } //try-finally 最后一次尝试
[0308] } finally{//Check the Connection back in 再次登记该连接
[0309] if(resource != null) getRelatedCacheConnectionPool().checkIn(resource); } //try-finally 最后一次尝试
[0310] ) catch(ResourcePoolException exc) {if(isLoggingError())
{logError(" Unable to get Oracle cache connection ", exc); } throw new
RelatedSearchException(" Unable to get Oracle cache connection ", exc); }
catch(SQLException se) {if(isLoggingError()) {logError(" Interface with Oracle
cache failed ", se); } throw newRelatedSearchException(" Interface with Oracle
cache failed ", se); } //try-catch 尝试捕获
[0311] if(resultVector.size() == 0) return null; else {resultVector.
copyInto(results = new
[0312] RelatedResult[resultVector.size()]); return results; }
[0313] *Communicate to Texis thru TexisConnectionPool 通过TexisConnectionPool
连接到 Texis
[0314] *&commat; return RelatedResult[]*&commat; param canonQuery java.
lang.String*&commat; param maxResults int*&commat; param maxLength int*/
private RelatedResult[] findFromUDPTexis(String rawQuery, String canonQuery,
int maxResults, int maxLength) throws RelatedSearchException{
[0315] RelatedResult[] results = null; //Get a UDPTexisConnection 获 得 一
UDPTexisConnection
[0316] ResourceObject resource = null;
[0317] TexisUDPConnection tc = null; try {resource =
getTexisUDPConnectionPool().checkOut(getAbsoluteName()); tc =
(TexisUDPConnection)resource.getResource();
[0318] DatagramSocket socket = tc.getSocket(); //do this at runtime to be able
to switch Dynamo at run time 在运行时执行该操作以便能在运行时转换 Dynamo
[0319] socket.setSoTimeout(getTexisTimeoutMillis()); //package data to send 打
包数据以便发送
[0320] TexisRequest request = new TexisRequest(); request.
setRawQuery(rawQuery); request.setCanonQuery(canonQuery); request.
```

```
setMaxResults(maxResults) ;request.setMaxChars(maxLength) ;request ;
setSequenceNumber(++tc.sequenceNumber) ;request.setTimeout(getTexitisTimeoutMillis()) ;

[0321]  ByteArrayOutputStream baos = new ByteArrayOutputStream() ;
ObjectOutputStreamous = new ObjectOutputStream(baos) ;ous.
writeObject(request) ;ous.flush() ;baos.close() ;byte[] sendData = baos.
toByteArray() ;//send it off to theserver 从服务器发送它

[0322]  if(isLoggingDebug()){logDebug( " About to send to Texitis at
endpoint:" +tc.getHost0+" :" +tc.getPort()) ;}//send it 发送它

[0323]  DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.
length, tc.getHost(), tc.getPort()) ;socket.send(sendPacket) ;//wait for a reply
upto timeOut 等待有关超过毫秒的一响应

[0324]  long startWait = System.currentTimeMillis() ;while(true){//pull off
inboud packets and check them the the rightsequenceNumber完成 inboud信息包并用
正确的顺序号核对它们 //throws a java.io.InterruptedIOException on timeout

[0325]  DatagramPacket receivePacket = new

[0326]  DatagramPacket(new byte[BUFFER_SIZE], BUFFER_SIZE) ;socket.
receive(receivePacket) ;

[0327]  ObjectInputStream ois = new ObjectInputStream(new
[0328]  ByteArrayInputStream(receivePacket.getData())) ;

[0329]  TexitisResponse response = (TexitisResponse)ois.readObject() ;ois.close() ;
if(response.getSequenceNumber() != tc.sequenceNumber){//we got a stale
response 我们获得一失效的响应

[0330]  long midPoint = System.currentTimeMillis() ;int remainder = (int)(get
TexitisTimeoutMillis().-(midPoint-startWait)) ;if(remainder > 0){//if we can
stillwait some more before a timeout 如果我们在一超时前还能等待更久 //reset
socket timeOut to the remaining time 对剩余的时间重新设置插槽的 timeOut

[0331]  socket.setSoTimeout(remainder) ;}else{//give up atthis point break ;在
该点中止

[0332]  } } //if-wrong-sequence-number else{results = response.getResults() ;
break ;} } //while} catch(ResourcePoolException rpe) {if(isLoggingError())
{logError( " Unable to get or checkin a Texitis connection " ,
rpe) ;} } catch(ClassNotFoundException cnfe) {if(isLoggingError())
{logError( " Class not found Exception " , cnfe) ;} } catch(SocketException
se) {if(isLoggingError()) logError( " Socket Except ion talking to
Texitis " , se) ;} } catch(StreamCorruptedException sce) {if(isLoggingError())
{logError(" Corrupted return from Texitis" , sce) ;} } catch(InterruptedIOException
ioie) {if(isLoggingDebug()) {logDebug(" Timed out talking to Texitis" , ioie) ;} }
catch(IOException ioe) {if(isLoggingDebug) f logDebug( " Timedout talking
```

to Texis " , ioe );})finally{//Check theConnection back in if we got it in the first place try{if(resource ! = null)getTexisUDPConnectionPool().checkIn(resource) ;}catch(ResourcePoolException rpe) /\*ignore this one 忽略这一个 \*/}return results ;}\*&commat ;returnRelatedResult[]-an array of RelatedResult objectsRelatedResult对象的一个数组\*which is ordered by relevance fromhigh to low or null if the system is disable or no\*relatedresults were found 按关联从高到低或如果该系统被禁止或没有找到相关的结果的空排序 \*&commat ; param rawQuery java.lang.String-raw query for which related searches are needed 用于相关的搜索所需的原始查询 \*&commat ;param canonQuery java.lang.String-canonicalized for of the raw query 原始查询的规范化 \*&commat ;param maxResults int-maximum number of resultsrequested 所请求的最大结果数据 \*&commat ;parammaxResultLength int-maximum lenght of a result in characters 在字符串中一结果的最大长度 \*/public RelatedResult[]findRelated(String rawQuery, String canonQuery, int maxResults, intmaxResultLength)//throws [0333] RelatedSearchException throws [0334] RelatedSearchException, RemoteException{requestCount++ ;//Return fast if system is disabled 如果系统被禁止,则返回快 [0335] if( ! getSystemEnabled())return null ; [0336] RelatedResult[]results = null ;//first try getting datafrom the Oracle pool(if enabled)(如果允许) 第一次尝试从 Oracle 取数 [0337] if(getOracleEnabled()) {try{long startOracle = System.currentTimeMillis() ;//keep timing stats 继续计时 Stats [0338] results = .findFromCache(canonQuery, maxResults, maxResultLength) ; oracleCacheHits++ ;//fixed statistics bugcumulativeOracleTime+ = (System.currentTimeMillis()-startOracle) ;}catch(RelatedSearchException rse){//If Oracle told us that this search has no related//i.e.editorially-excluded porn, then drop out early 如果 Oracle 告诉我们该搜索没有相关的即编辑上排除了色情部分,则先退出 [0339] if(rse.getRootCause() == null) {return null ;}else{//log it otherwise for post mortem 否则对后 mortem 登录它 [0340] if(isLoggingError())logError(" Failed to interface toOracle cache,will try Texis " , rse) ;}catch(Exception e){if(isLoggingError())logError(" Failed to interface toOracle " , e) ;}}//if Oracle enabled 如果 Oracle 允许 // ifunsuccessfull then try Texis pool if enabled 如果未成功,则如果允许的话尝试 Texis [0341] if(getTexisEnabled()&&results == null) {try{longstartTexisQuery = System.currentTimeMillis() ;//keep texistiming stats 继续 Texis 计时 Stats [0342] texisRequests++ ;results = findFromUDPTexis(rawQuery, canonQuery, maxResults, maxResultLength) ;long texisQueryMillis = System.currentTimeMillis()

() -startTexitQuery ;' //logabnormally long request time 记录异常的长的请求时间  
[0343] if (texisQueryMillis > getTexitTimeoutMillis ()) slowTexitRequestCount ++ ;  
cummulativeTexitTime + = texisQueryMillis ; } catch (Exception e)  
{ if (isLoggingError ()) logError ( " Texit interface failed with: " + e.  
getMessage (), e ); } //if texisEnabled return results ; 如 果 texisEnabled 返回  
结果 )/\*\*Stats accessor Stats 访问 程序 \*&commat ;return String \*/ public  
String getCummulativeOracleTime () { return (cummulativeOracleTime / 1000.0)  
+ " seconds " } /\*\*Stats accessor Stats 访问 程序 \*&commat ;return long \*/  
public String getCummulativeTexitTime () { return (cummulativeTexitTime / 1000  
.0) + " seconds " ; } /\*\*Stats accessor Stats 访问 程序 \*&commat ;return int \*/  
public int getOracleCacheHits () { return oracleCacheHits ; } /\*\*Stats accessor  
Stats 访问 程序 \*&commat ;return boolean \*/ public boolean getOracleEnabled  
f return oracleEnabled ; } /\*\*Accessor for relatedCacheConnectionPool  
用 于 relatedCacheConnectionPool 的 访 问 程 序 \*&commat ;return atg.  
service. resourcepool. JDBCConnectionPool \*/ public JDBCConnectionPool  
getRelatedCacheConnectionPool () { return relatedCacheConnectionPool ; } /\*\*Stats  
accessor Stats 访问程序

[0344] &commat ;return int \*/ public int getRequestCount () { return  
requestCount ; } /\*\*Stats accessor Stats 访 问 程 序 \*&commat ;return int \*/  
public int getSlowTexitRequestCount () { return slowTexitRequestCount ; } /\*\*  
\*Stats accessor Stats 访 问 程 序 \*&commat ;return boolean \*/ public boolean  
getSystemEnabled () { return systemEnabled ; } /\*\*Stats accessor Stats 访 问  
程 序 \*&commat ;return boolean \*/ public boolean getTexitEnabled () { return  
texisEnabled ; } /\*\*stats accessor Stats 访问程序 \*&commat ;return int \*/ public  
int getTexitRequests () { return texisRequests ; } /\*\*configu param accessor 配置  
参数访问程序 \*&commat ;return int \*/ public int getTexitTimeoutMillis () { return  
texisTimeoutMillis ; } /\*\*This method was created in VisualAge. 该 方 法 用  
VisualAge 创建

[0345] \*&commat ;return com. go2. search. related. TexitUDPConnectionPool \*/  
public TexitUDPConnectionPool getTexitUDPConnectionPool () { return texisUD  
PConnectionPool ; } /\*\*mutator\*&commat ;param newValue boolean \*/ public void  
setOracleEnabled (boolean newValue) { this. oracleEnabled = newValue ; } /\*\*Mutator  
for relatedCacheConnectionPool 用 于 relatedCacheConnectionPool 的  
Mutator\*&commat ;param newValue atg. service. resourcepool. JDBCConnectionPool \*/  
public void setRelatedCacheConnectionPool (JDBCConnectionPool newValue)  
{ this. relatedCacheConnectionPool = newValue ; } /\*\*mutator\*&commat ;param  
newValue boolean \*/ public void setSystemEnabled (boolean newValue) { this.  
systemEnabled = newValue ; } /\*\*mutator\*&commat ;param newValue boolean \*/  
public void setTexitEnabled (boolean newValue) { this. texisEnabled =

newValue ; }/\*\*parameter mutator 参 数 mutator\*&commat ;param newValue int \*/  
 public voidsetTexitisTimeoutMillis(int newValue) {this.texitisTimeoutMillis =  
 newValue ; }/\*\*This method was created in VisualAge. 该方法是用 VisualAge 创建  
 [0346] \* & commat ; param newValue com.go2.search.related.  
 TexitisUDPConnectionPool\*/public voidsetTexitisUDPConnectionPool(TexitisUDPConnecti  
 onPool newValue) {this.texitisUDPConnectionPool = newValue ; }

[0347] The following code is control code that controls the dumping of search  
 listing database, loading the crawled text, and inverted-indexing all the related  
 search indexing, including building the 'derived-data' elements 以下代码是控制搜  
 索列表项数据库的信息转储、载入搜索 (crawl) 过的文字以及倒排索引所有的相关搜索索  
 引的控制码,包括连编“所导出的数据”部分 :

[0348] :#! /bin/ksh-x export PATH = /usr/local/morph3/bin:::\$PATH#../.zshrc  
 export TMP = /export/home/goto/tmp export TEMP = \$TMP export TEMPDIR = \$TMP  
 export TMPDIR = \$TMPTMPTABLE = linead0

[0349] TMPTABLE2 = linead

[0350] TERMSTABLE = terms

[0351] INC = 02NEWTABBE = line\_ad\${INC}

[0352] CRAWLDATA = /home/goto/rs/DONE/ALL.UNIQ CRAWLTABLE = line ad4

[0353] SPOOL = /home/goto/list

[0354] DB = /home/goto/crawladb#####

[0355] Log() {echo' \n####' \$(date" +% m/% d% H:% M:% S" ):" \${\*}" , ####' }  
 log 0. timport crawled data

[0356] Log 0.1 Create line\_ad4 and unique index in preparation for 'crawl' import  
 tsql-d\$DB#drop table line\_ad4;create table linead4(id counter,

[0357] ad\_url varchar(300), crawltitle varchar(750), crawlmeta varchar(500),  
 crawlbody varchar(8000)) i drop index idx4ad\_url;create unique index idx4ad\_url  
 on line\_ad4(adurl); ! timport-database \$DB-table \$CRAWLTABLE-s/home/goto/rs/  
 DONE/crawl.sch-file \$CRAWLDATA

[0358] Log 1. extract line\_ads from live\_ADMIN into column delimited spool file  
 upadadm \$SPOOL

[0359] Log 2. timport-database \$DB-table \$TMPTABLE-s newrs.sch-file \${SPOOL}  
 timport-database \$DB-table \$TMPTABLE-snewrs.sch-file \${SPOOL}

[0360] Log 3. build canon index on \${TMPTABLE##tsql-d\$DB} #drop index idx0cst;  
 create index idx0cst on \${TMPTABLE} (cannon search text); !

[0361] Log 4. add counts of canons to \$TMPTABLE texis DB = \$DBTMPTABLE =  
 \$TMPTABLE updatecnt

[0362] Log 5. build url index on \$TMPTABLE tsql-d\$DB <<! drop index idx0url;  
 create index idx0url on \${TMPTABLE} (aa\_url);

[0363] Log 6. merge crawled text w/original tsql-d\$DB < <! drop index

```

idx4url ;create index idx4url on ${CRAWLTABLE} (ad_url) ;drop table $TMPTABLE2 ;
[0364]   CREATE TABLE $TMPTABLE2
[0365]     AS
[0366]   SELECT a.price price, a.rating rating, a.ad_id ad_id, a.bit-date
        bid-date, a.raw_search_text raw_search_text, a.cannon_search_text cannon_
        search_text, a.adspectitleadspectitle, a.ad_spec_desc ad_spec_desc, a.ad_url
        ad_url, a.resource resource_id, b.crawltitle crawltitle, b.crawlmetacrawlmeta,
        b.crawlbody crawlbody, a.canon cntfrom$TMPTABLE a, ${CRAWLTABLE} b where a.adurl
        = b.adurl order by price desc ;#texisDB = $DB CRAWLTABLE = ${CRAWLTABLE} TMPTABLE
        = $TMPTABLE updateit
[0367]   Log 7. collapse 0 onto 01
[0368]   Log 7.1 first-make the table tsql-d${DB}#! drop table ${NEWTABLE} ;
        tsql-d${DB} < <! create table ${NEWTABLE} (canon cnt integer, cannon_search_
        text varchar(50), raw_search_text varchar(50), advertiser_ids varchar(4096),
        advertiser_cnt integer, words varchar(65536)) ; !
[0369]   Log 7.2second, build the uniq, sorted list of search terms
[0370]   Log 7.2select cannon-search-text from ${TMPTABLE} ;tsql-d ${DB}
        < <! #sort-u I timport-s termstable.sch-fileselect cannonsearchtext
        from ${TMPTABLE} ;
[0371]   Log 7.3third, build uniq index on terms table
[0372]   Log 7.3create unique index idxterm on terms term tsql-d${DB} < < !
        create unique index idxterm on terms(term) ;
[0373]   Log 7.3.9prepare for collapse
[0374]   Log 7.3.9create index idxcstcol on ${TMPTABLE2}cannon_search_text
[0375]   Log 7.3.9create index idxadidcol on ${TMPTABLE2} adid tsql-d${DB}
        <<! create index idxcstcol on ${TMPTABLE2}(cannon_search_text) ;create index
        idxadidcol on ${TMPTABLE2}(ad_id) ; !
[0376]   Log 7.4fourth collapse around csts
[0377]   Log 7.4texis SRCTABLE = ${TMPTABLE2} TGTTABLE = ${NEWTABLE}
[0378]   TERMSTABLE = ${TERMSTABLE} collapse texis db = ${DBSRCTABLE} = ${TMPTABLE2}
        TGTTABLE = ${NEWTABLE}
[0379]   TERMSTABLE = ${TERMSTABLE} collapse
[0380]   Log 8 do the porn line buildporn#timport-database ${DB}-table ${NEWTABLE}-s
        rsporn.schfile
[0381]   Log 9 do the porn table newporn#timport-database ${DB}-srsnewporn.
        sch-file
[0382]   Log 10 metamorph index words column tsql-d${DB} <<! create metamorph
        inverted index mmx${INC}w on ${NEWTABLE} (words) ;
[0383]   Log 10 all done

```

```

[0384]    Dumps bid-for-placement search listings data# ! /bin/ksh
[0385]    TXSORAUSER = XXXXXX TXSORAPWD = XXXXXX
[0386]    LVSRVPWD = XXXXX#SPOOL = pipe
[0387]    SPOOL = ${1}
[0388]    SERVER = XXXXX
[0389]    Log() {echo' \n####' $(date" +% m/% d% H:% M:% S" ):;" ${1}" '####')
[0390]    Log " start dump " sqlplus-$TXSORAUSER/$TXSORAPWD&commat;
1#${SERVER} > /dev/null ! //set heading off setlinesize 750set pagesize
0set arraysize 1set maxdata 50000setbuffer 50000set crt off;set termout
off spool$(SPOOL)selectrpad(to_char(advertiser_id),8)11rpad(raw_search,
30)#rpad(canon-search,30)#rpad(title,100)#rpad(description,280)#rpad(url,
200)#rpad(resource_id,20)#rpad(to_char(price*100),5)11rpad(rating,
2)11rpad(to_char(search_id),8)#rpad(resource_id,18)#rpad(to_char(line_ad_id),
8)11to_char(bid_date,'YYYYMMDDHHMMSS')from ads where status = 5and rating
='G' and canon_search<>'grab bag' and rownum < 10000;spool off;quit;
[0391]    Log" end dump" exit 0
[0392]    Counts#of occurrences of pagebids for each particularpotential related
search result<script language = vortex><timeout = -1></timeout><a name =
main><DB = "/home/goto/crawldb" ><SQL ROW" select distinctcannonsearchtext
cst from" $TMPTABLE><SQL ROW" select count(*)cnt from" $TMPTABLE " where
cannon_search_text = $cst" ><SQL NOVARS" update" $TMPTABLE" set canon_cnt =
$cnt wherecannon_search_text = $cst" ></SQL></SQL></SQL></a></script>
[0393]    Aggregates web page body-text and listings based on therelated-search
result, while collecting and creatingderived-data of 1, how many different
advertisers have web-pagesassociated with the related-search result. 基于相关的
搜索结果,总的网页文字字和列表项,同时收集和创建导出的数据 1,多少不同的广告商具
有与该相关的搜索结果相关的网页
[0394]    <script language = vortex><timeout = -1></timeout><DB = /home/
goto/crawldb><a name = main><! --get allcanon-terms from tmp table--><SQL
ROW" select term cstfrom" $TERMSTABLE><$words = ><$rst = ><$csts = ><$asts
= ><$asds = ><$cts = ><$cms = ><$cbs = ><$advs = ><$last_adv = ><$adv_cnt
= 0><! --get all rows w/this canon term from tmp table--><SQLROW" select
cannon-cent cc, ad_id aid, raw-search-text rst, cannon_search_text cts, ad_
spec_title ast, .ad_spec_desc asd, crawltitle ct, crawlbody cb, crawlmeta
cm from" $SRCTABLE " wherecannon_search_text = $cst order by adi
d " ><! --aggregatethe text to prepare for collapsed insert--><$rst =
t$rst+$rst><$rst = ($csts+$cst)><$asts = ($asts+$ast)><$asds
= ($adsds+$asd)><$cts = ($cts+$ct)><$cms = ($cms+$cm)><$cbs =
($cbs+$cb)><if$aid! = $last_adv><! --add advertiser to list if not seen him

```

```
before--><$advs = ($advs+$aid)><$adv_cnt = ($adv_cnt+1)><$lastadv = $aid></if></SQL><$canon_cnt = $loop><$words = ($rst+$csts+$asts+$asds+$cms+$cbs+$cts)><! --pickoff zeroeth element only from$rst array--><loop$rst><$Rst = $rst><break></loop><strlen$words><$wlen = $ret><strlen$advs><! --display which rowwe're working on-->$wlen$ret$cst<! --insert to collapsed row--><SQL NOVARS " insert into " $TGTTABLE " (canon_cnt, cannon_search_text, raw_search_text, advertiser_ids, advertiser_cnt, words) VALUES ($cc, $cst, $Rst, $advs, $adv_cnt, $words) " ></SQL><! --$words***>$ret = (text2mm($words, 50))>$ret--></SQL></a></script>
```

[0395] Database schema layout used to upload bidden search listings 用来载入竞买的搜索列表项的数据库大纲布局

[0396] database/home/goto/crawldb #droptableline\_ad1 droptableline\_ad0 table line\_ad0 createtable col #keepfirst trimspace#multiple datefmt yyyymmdd HHMMSS#Name Type Tag default valfield advertiserid varchar(8) 1-8 field raw-search-textvarchar(40) 9-48 field cannon\_search\_text varchar(40) 49-88 field ad\_spec\_title varchar(100) 89-188 field adspecdescvarchar(2000) 189-2188 field adurl varchar(200) 2189-2388 field resource varchar(20) 2389-2408 field price integer 2409-24130 field rating char(2) 2414-2415 field ad\_id integer 2416-24230 field bid\_date date 2424-24380 field canon\_cnt integer 0 field crawlwords varchar(40)

[0397] Manual join of search listing data with crawled web page data into a single merged table 手工将搜索列表项数据与搜索 (crawl) 过的网页数据连接到一个单个的合并表中

[0398] <script language = vortex><timeout = -1></timeout><a name = main><DB = "/home/goto/crawldb" ><SQLROW" select ad\_url myurl,crawltitle ct,crawlmeta cm,crawlbodycb from" \$CRAWLTABLE><SQL NOVARS" update" \$TMPTABLE" setcrawltitle = \$ct,crawlmeta = \$cm,crawlbody = \$cb where ad\_url = \$myurl" ></SQL></SQL></a></script>

[0399] Code to duplicate URL Crawl elimination 完全相同 URL 搜索 (crawl) 消除的代码

[0400] /\*\*\*Insert the type's description here. 这里插入类型的说明

[0401] \*Creation date: (02/18/2000 11:12:12AM) 创建时间 \*&commat ;author:\*/

[0402] import java.io.\*; //import of java classes needed for input/output 用于输入 / 输出所需的 Java 类的输入

```
[0403] import java.util.*; //import corejava.*;import java.lang.String;
public class Url{*****
*****Compare URLs Address 比较 URLs 地址 *****
*****public static void main(String args[])
throws Exception{//Decalarations of the input and output File 输入和输出文件的
```

## Decalaration

```
[0404] BufferedReader inputFile ;
[0405] PrintWriter nonDupFile ;
[0406] PrintWriter dupFile ;//Initialization 初始化
[0407] String firstUrl = " " ;
[0408] String secondUrl = " " ;
[0409] String urlBufferA, urlBufferB = " " , urlBufferC = " " ;
[0410] String compareDomainA = " " ;
[0411] String compareDomainB = " " ;
[0412] String compareDomainC = " " ;
[0413] String newFlag = " false" ;inputFile = new BufferedReader(new
[0414] FileReader(" /home/lauw/urls.lau" )) ;nonDupFile = new PrintWriter(new
FileWriter(" /home/lauw/nonDupFile.real" )) ;dupFile = new PrintWriter(new
[0415] FileWriter(" /home/lauw/dupFile.real" )) ;nonDupFile.close() ;dupFile.
close() ;firstUrl = inputFile.readLine() ;secondUrl = inputFile.readLine() ;
urlBufferC = inputFile.readLine() ;urlBufferA = firstUrl ;urlBufferB =
secondUrl ;do
[0416] Slash CcompareDomainA = new Slash() ;
[0417] Slash ccompareDomainB = new Slash() ;
[0418] Slash ccompareDomainC = new Slash() ;compareDomainA = ccompareDomainA.
Slash(urlBufferA) ;compareDomainB = ccompareDomainB.Slash(urlBufferB) ;
compareDomainC = ccompareDomainC.Slash(urlBufferC) ;
[0419] Compare compareSub = new Compare() ;newFlag = compareSub.
Compare(compareDomainA, compareDomainB, compareDomainC, urlBufferB, newFlag) ;
urlBufferA = urlBufferB ;urlBufferB = urlBufferC ;urlBufferC = inputFile.
readLine() ;}while (urlBufferC != null) ;/////////////////////////////Loop for
first Null value 用于第一 Null 值的循环
[0420] urlBufferC = firstUrl ;
[0421] Slash ccompareFirstNullDomainA = new Slash() ;
[0422] Slash ccompareFirstNullDomainB = new Slash() ;
[0423] Slash ccompareFirstNullDomainC = new Slash () ;compareDomainA =
ccompareFirstNullDomainA.Slash(urlBufferA) ;compareDomainB =
ccompareFirstNullDomainB.Slash(urlBufferB) ;compareDomainC =
ccompareFirstNullDomainC.Slash(urlBufferC) ;
[0424] Compare compareFirstNullSub = new Compare() ;newFlag =
compareFirstNullSub.Compare(compareDomainA, compareDomainB, compareDomainC,
urlBufferB, newFlag) ;/////////////////////////////Loop for last Null value 用于最后
的 Null 值的循环
[0425] urlBufferA = urlBufferB ;urlBufferB = firstUrl ;urlBufferC = secondUrl ;
```

```
[0426]     Slash ccompareLastNullDomainA = new Slash() ;
[0427]     Slash ccompareLastNullDomainB = new Slash() ;
[0428]     Slash ccompareLastNullDomainC = new Slash() ;compareDomainA
= ccompareLastNullDomainA.Slash(urlBufferA) ;compareDomainB =
ccompareLastNullDomainB.Slash(urlBufferB) ;compareDomainC =
ccompareLastNullDomainC.Slash(urlBufferC) ;
[0429]     Compare compareLastNullSub = new Compare() ;newFlag =
compareLastNullSub.Compare(compareDomainA, compareDomainB, compareDomainC,
urlBufferB, newFlag) ;inputFile.close() ;} } class Slash(
[0430]     String Slash(String buffer) {int domainSlashEnd = 0 ;intdomainSlashStart
= 0 ;boolean domainIndex = false ;boolean startFound = false ;boolean newFlag =
false ;
[0431]     String comparedomain ;comparedomain = " " ;for(intdomainSlashLoop =
8 ;domainSlashLoop < = (buffer.length()-1) ;domainSlashLoop++) {if((buffer.
substring(domainSlashLoop, (domainSlashLoop+1)).equals( " / " ))(buffer.
substring(domainSlashLoop, (domainSlashLoop+1)).equals(" ? " )))(if(startFound
== false) {///Check the Urlswith Domain Name only 仅用域名核对该 URLs
[0432]         if((domainSlashLoop+1) == buffer.length()) {comparedomain = buffer.
substring(0, (buffer.length())) ;domainIndex = true ;domainSlashLoop = buffer.
length()+500 ;}//end for domain name only 仅结束域名
[0433]         domainSlashStart = domainSlashLoop+1 ;startFound = true ;} else
{domainSlashEnd = domainSlashLoop ;domainSlashLoop = buffer.length ()+500 ;///
add 5to get out of the loop 加 5 来退出循环
[0434]     } } //end for Loop 结束循环
[0435]     if(domainSlashEnd == 0) domainSlashEnd = buffer.length() ;}
if(domainIndex == false) {comparedomain = buffer.substring(domainSlashStart,
domainSlashEnd) ;} } returncomparedomain ;}) import java.io.* ;//import of java
classesneeded for input/output class Compare 用于输入类比较所需的 Java 类的输入
[0436]     {
[0437]     String Compare(String aCompareDomainA,, String aCompareDomainB,
[0438]     String aCompareDomainC, String aUrlBufferB, String newFlag) throws
[0439]     Exception (
[0440]     PrintWriter nonDupFile ;
[0441]     PrintWriter dupFile ;nonDupFile = new PrintWriter(new
[0442]     FileWriter( " /home/lauw/nonDupFile.real " , true) , true) ;dupFile = new
PrintWriter(new
[0443]     FileWriter( " /home/lauw/dupFile.real " , true) , true) ;
if(aCompareDomainC.equals(aCompareDomainB)) if(newFlag.equals( " true " ))
(dupFile.println(" New" ) ;newFlag = " false" );}
```

```
[0444] System.out.println(" Duplicate " );dupFile.println(aUrlBufferB);  
else {if(aCompareDomainB.equals(aCompareDomainA))if(newFlag.equals(" true"))  
{dupFile.println(" New" );newFlag =" false";}  
[0445] System.out.println(" print a Duplicat in second time" );  
[0446] System.out.println(" Sec Duplicate" );dupFile.println(aUrlBufferB);  
else  
[0447] System.out.println(" non Dup " );nonDupFile.println(aUrlBufferB);  
newFlag =" true";}  
[0448] System.out.println(" *****S, );}  
[0449] return newFlag;  
[0450] }  
[0451] }。
```

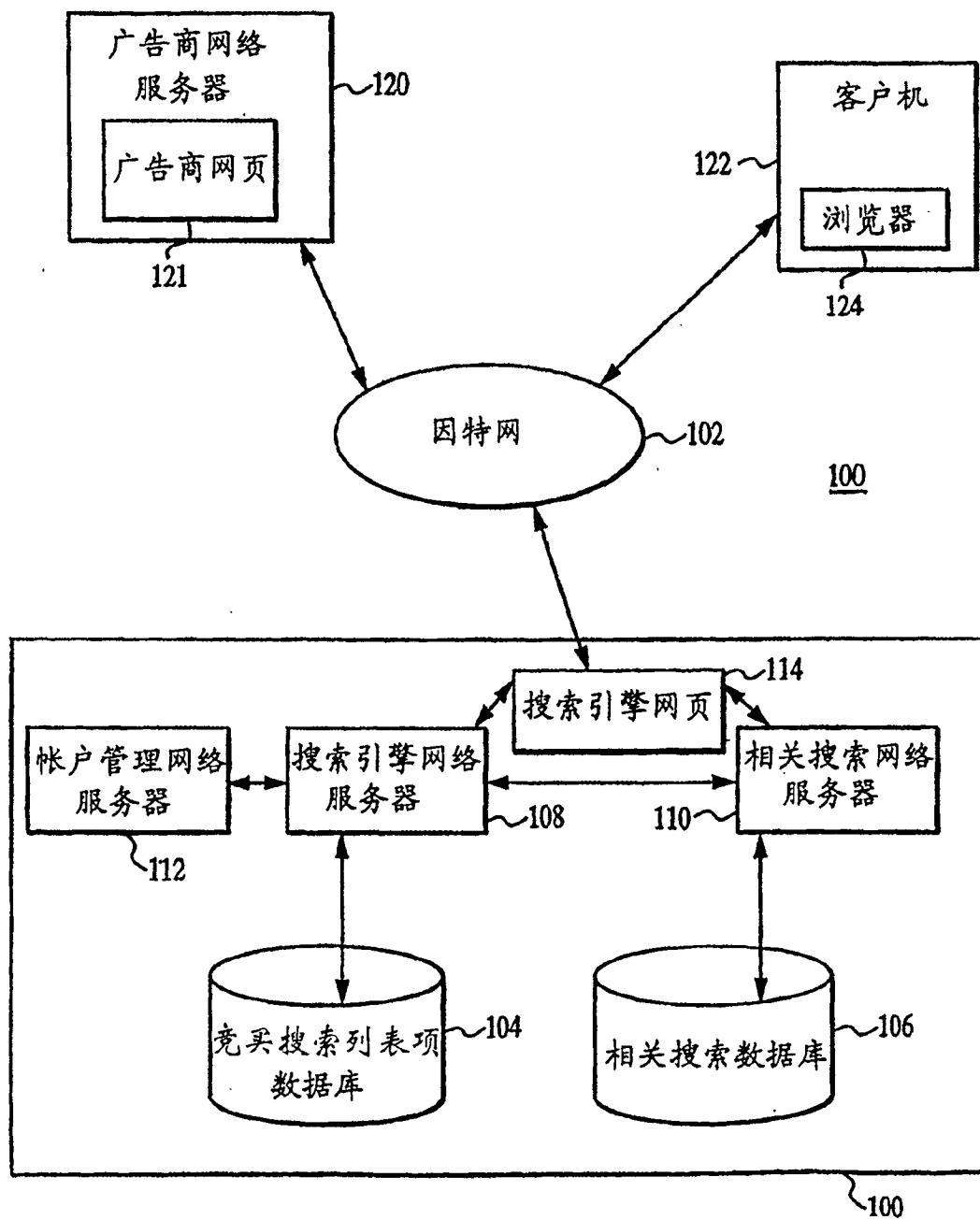


图 1

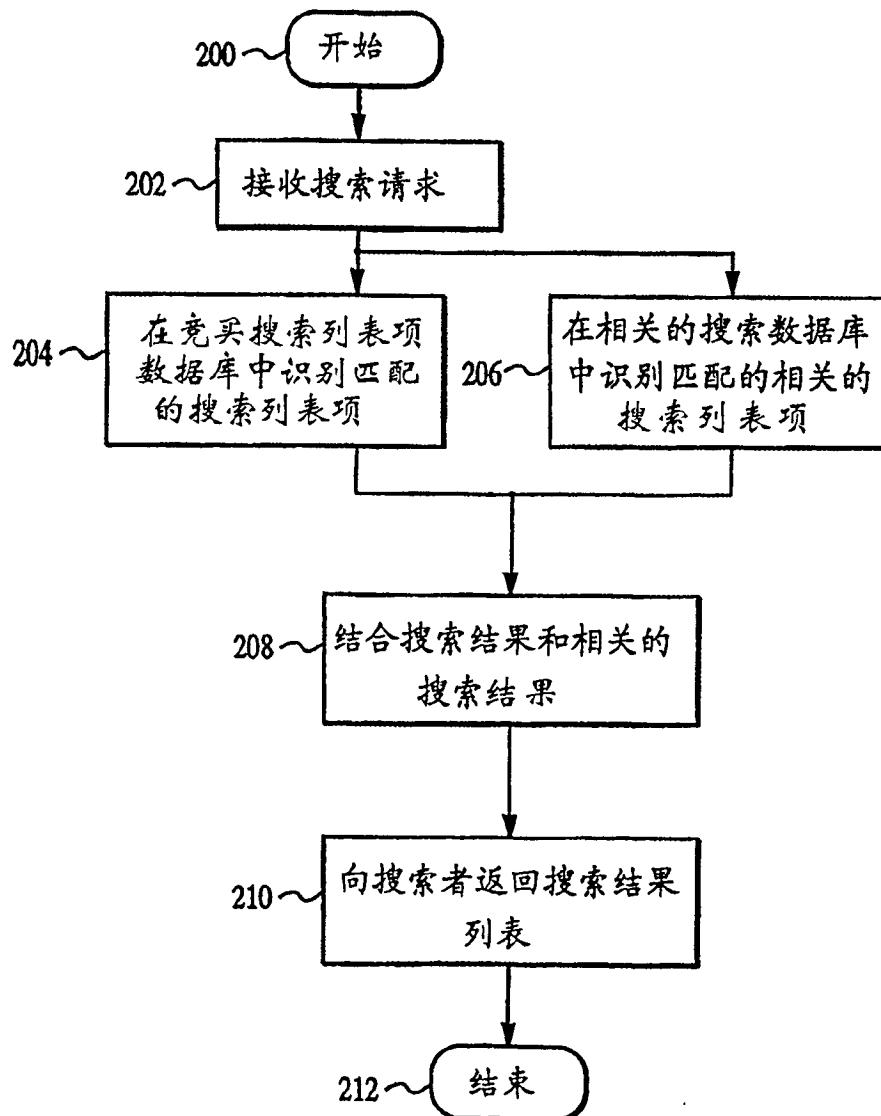


图 2

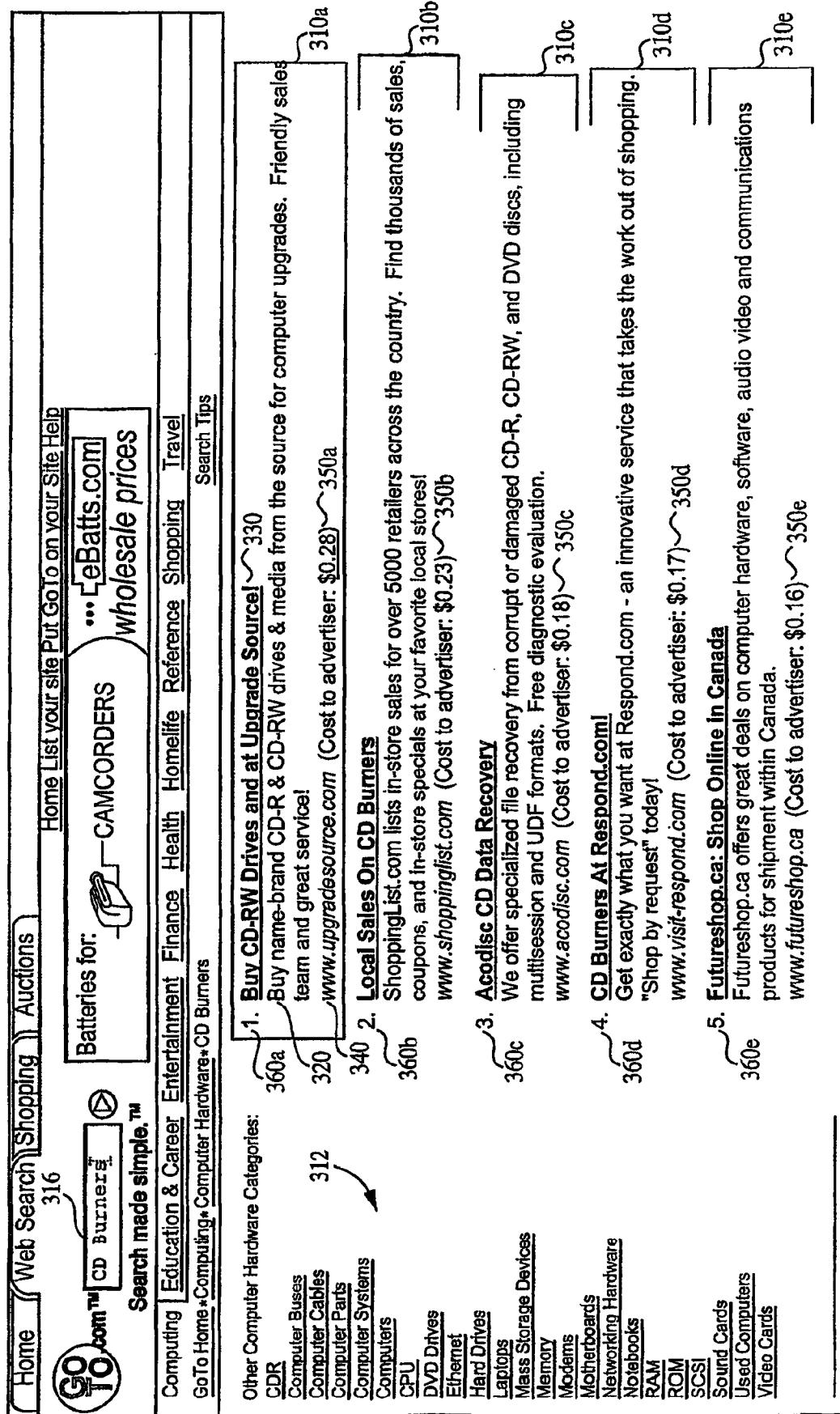


图 3a

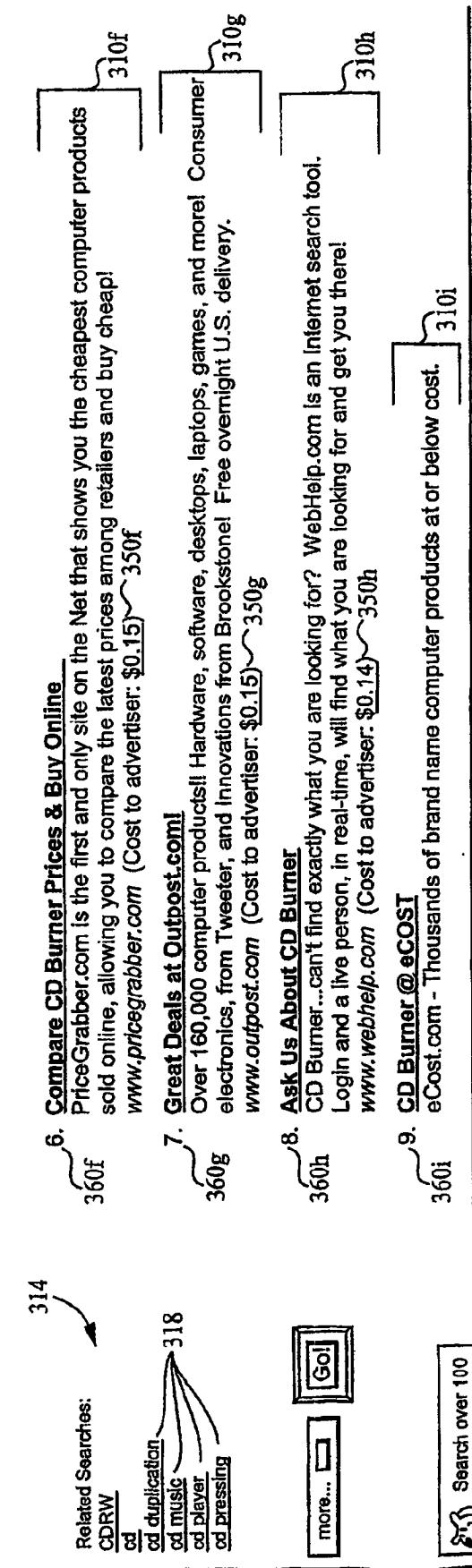


图 3b

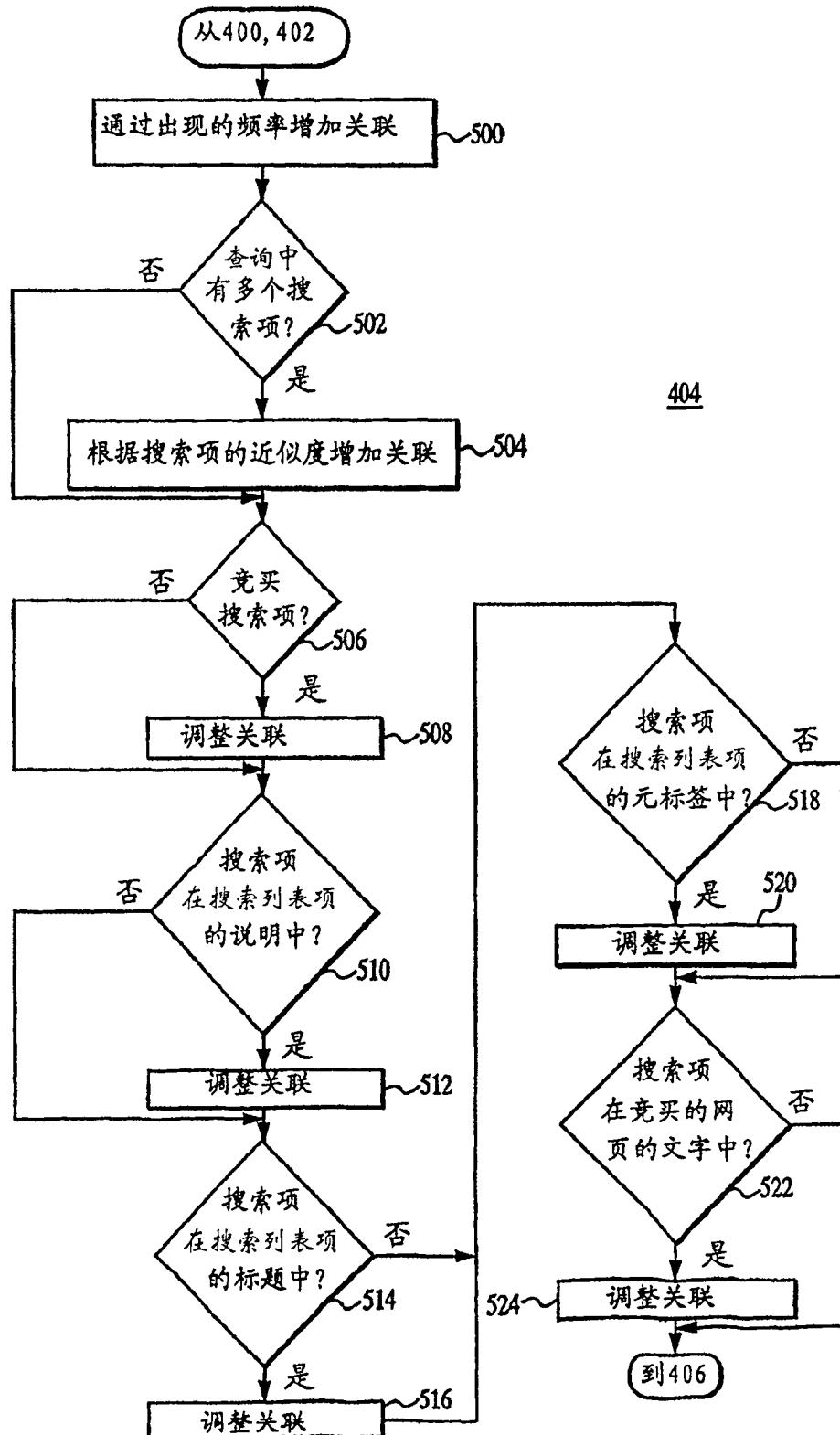


图 5

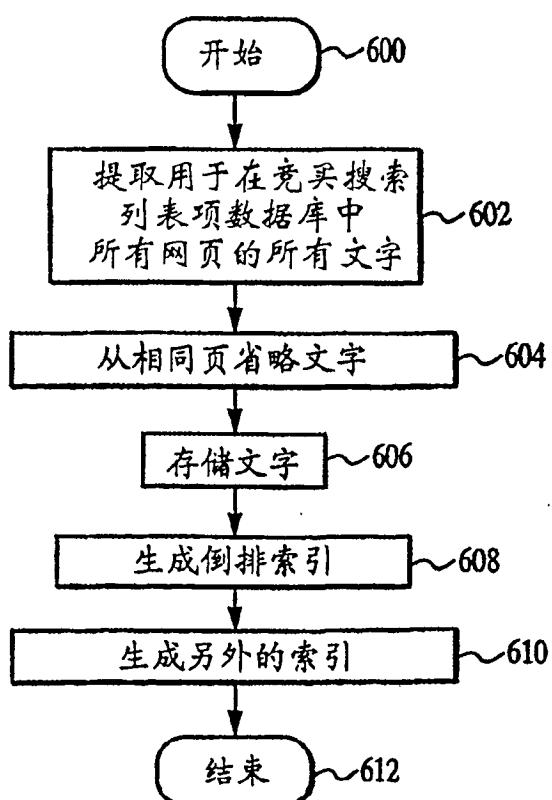
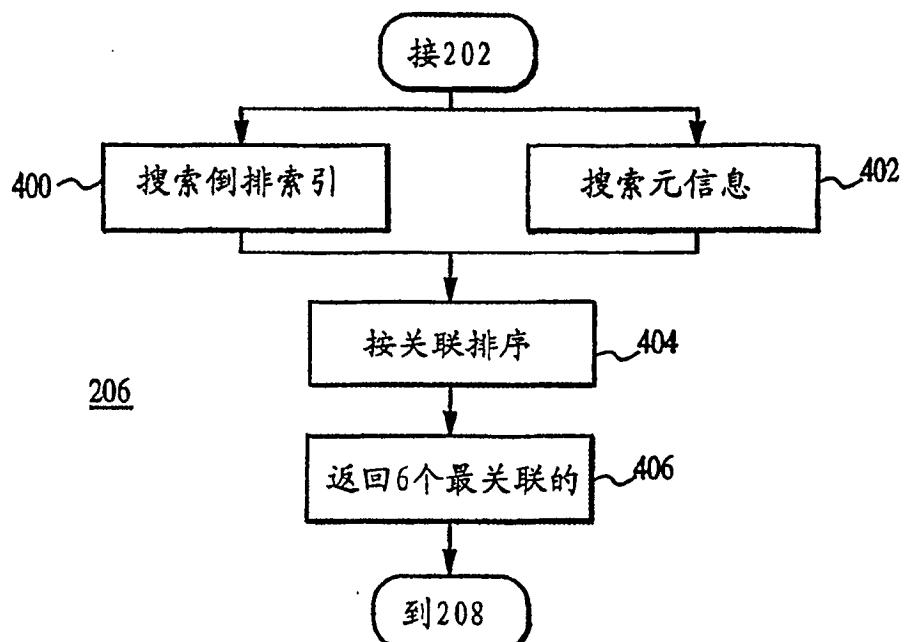


图 6

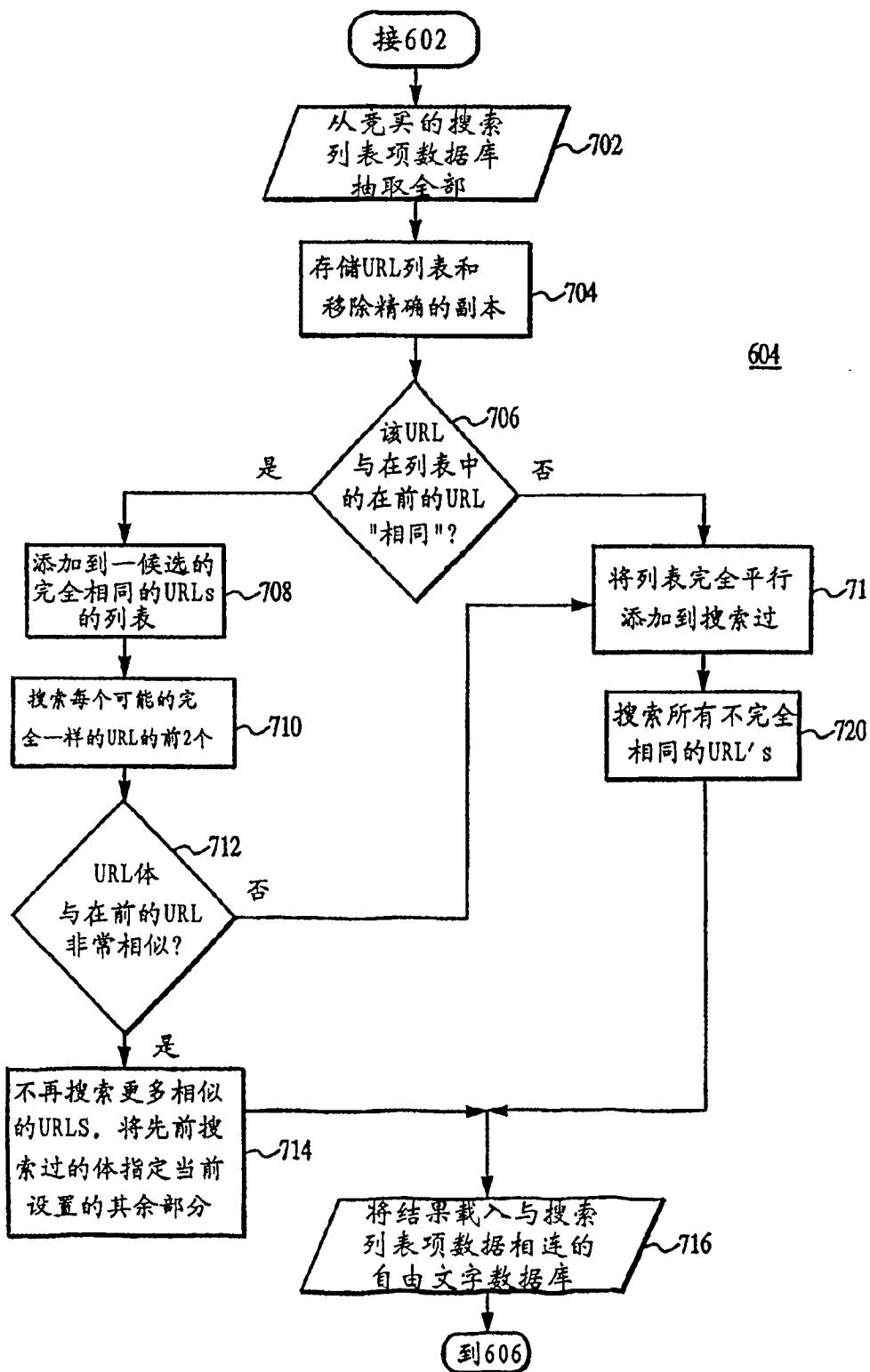


图 7