一种基于 IE 缓存的 Web 图片获取方法①

An Approach for Grabbing Image from Web Based on IE Cache

欧永红　姚耀文　(华南理工大学 计算机科学与工程学院 广东 广州 510006)

摘　要： 当前基于网页的抓取技术只是简单地获取整个页面的 HTML 文件并抽取所需的文本信息，而对于其它 MIME 对象，如图片，特别是通过 ASP、JSP 或 Servlet 等服务器端组件查询后返回的图片对象，则很难获取。为了解决这个问题，提出了一种基于 Internet Explorer 缓存的 Web 图片获取方法：在 Visual C++中采用多线程技术构造一个浏览器对象，将选定的样本图片页面在浏览器对象中打开，然后在 IE 缓存中搜索并获取所需的 Web 图片，最后将它存入 Oracle 数据库中分析使用。这种方法的优点是能够方便、快捷的获取任何来源形式的 Web 图片对象。

关键词： IE 缓存　图片获取　多线程　线程同步　事件

1　引言

随着 Internet 的迅猛发展，互联网已经成为了信息发布和获取的主要场所之一，并且其发布的数据量仍在不断地呈几何级数的快速增长，显然互联网对全球用户获取具有价值的信息资源的重要性不言而喻。因此，快速、有效的利用这些信息具有极其重要的意义。当前基于网页的抓取技术只是简单地返回整个页面的 HTML 文件，再通过对 HTML 文件进行抽取得到所需的文本信息[1]，这种方法虽然易于实现，但无法获取其它基于结构化信息的粒度更细 MIME (Multipurpose Internet Mail Extensions)对象，如图片对象等。本文提出了一种基于 Internet Explorer 缓存的 Web 图片获取方法，将选定的样本图片页面在构造的浏览器对象中打开，然后在 IE 缓存中查询并获取 Web 图片，最后将它存入 Oracle 数据库中供用户分析使用。

2　Web图片获取技术分析

Web 文本信息的获取比较容易实现，只需要与指定的 URL 建立一个 Session，并读取接收的数据即可获得，再通过对 HTML 文件的进行抽取得到所需的文本信息。而相对于文本信息而言，图片对象的获取要困难得多。对于图片对象的地址是物理链接地址的，在 Visual C++中可以直接通过 URLDownload ToFile 函数实现下载。而一些采用相对路径存储的图片对象，特别是通过 ASP、JSP 或 Servlet 等服务器端组件查询后返回的图片对象，很难直接从链接地址进行访问和获取。一般的方法是采用 WinPcap 等工具抓取网卡协议数据包[2]，通过分析，将包含图片信息的数据包进行提取、拼接并还原成图片文件，再保存入数据库中。这种方法的缺点在于图片信息被分割并封装到多个协议包后，对协议包的抓取、分析、拼接存在一定困难，且实现效率较低。本文提出了一种基于 IE 缓存的 Web 图片获取方法，大大降低了实现难度，提高了图片获取的效率，实现了对各种来源的图片对象的获取。

3　Web图片获取概述

需要抓取 Web 站点的信息，显示特点一般是:同一 Web 站点的同类数据信息，通常都是采用相同模板或者基于某种动态网页技术生成的，通过 HTML 或者 XHTML 格式表现出来，具有一定的相似性[3]，而其中图片对象主要是基于 ASP、JSP 或 Servlet 等服务器端组件执行后反馈给 IE 浏览器的。因此，图片对象的获取是针对相似的动态生成的 Web 网页。图片对象获取

① 收稿时间:2009-03-25

的过程如下：选定样本页面、定义图片缓存名、构造浏览器对象并打开样本页面、搜索 IE 缓存、获取图片对象并清空 IE 缓存、数据入库。

运行 Web 图片获取器，程序主界面如图 1 所示。

图 1　程序主界面

4　样本和模式

样本是指包含图片对象的特征性页面，分析样本可以获取所有同类网页的相关对象。如某网上书城中的图书，其网址为 http://网上书城/product.aspx?product_id=20503923。其图书信息页面如表 1 所示。

表 1　图书信息页面示例

图书名称	论语
作者	XXX
出版社	XX 出版社
图书图片	
价格	10.00 元

通过分析样本的网址，发现图书图片是服务器端执行 ASP.NET 查询组件后反馈给 IE 浏览器得到的，无法直接从服务器上下载。通过分析还发现所有同类网页的区别在于 product_id 不同，当设置不同的 product_id 时，可以自动获取某区间段之内的所有图书图片对象的页面地址。

模式是指网页中要获取的图片对象在 IE 缓存中对应的文件名。可以通过打开 IE 缓存目录，查找所需的图片对象，将其拷贝到缓存目录之外的任意文件目录中(IE 缓存中的文件名并不是其真正的文件名)，即可发现其命名规则。如表 1 中的图片复制到 IE 缓存目录外后文件名为 20503923_[b].jpg，则可以定义图片存储在 IE 缓存的模式为：product_id+_[b].jpg。对于不同的样本，需要定义不同的模式。

```
//下面是以样本页面设定的变量
CString sURL;//样本页面的网址
CString sProduct_ID;//商标编号
CString sPicture_Name_In_Cache= sProduct_ID + "_[b].jpg" ; //图片在 IE 缓存的名称
CString sFilePathName;//图片对象的存储目录
```

5　Web图片获取器设计

在 Visual C++中，Web 图片获取器主要通过 Wininet.lib 库实现的。具体设计如下：

5.1 建立 Session 连接

```
//导入 Wininet 库
#include " Wininet.h"
#pragma comment(lib,"Wininet.lib")
//与样本页面建立 Session 连接
m_pSession=newCInternetSession(NULL,0);
CHttpFile*pFile=(CHttpFile*) m_pSession->OpenURL (sURL);
```

以上代码实现了与样本页面建立 Session 连接，建立 Session 连接的目的是：很多服务器端代码在 IE 浏览器访问服务器时都会检查 Session 连接，只有建立了 Session 连接，才允许访问相应页面。

5.2 构建浏览器对象并显示样本页面

```
//构建显示图片的浏览器视图 CMapHtmlView：从 CHtmlView 继承
class CMapHtmlView : public CHtmlView{}
//在浏览器对象中显示样本网页
Navigate2(sURL,NULL,NULL);
```

5.3 在 IE 缓存中搜索要获取的图片对象

搜索 IE 缓存中的文件主要通过 Wininet 库中的 FindFirstUrlCacheEntry() 和 FindNextUrlCacheEntry()实现[4]。实现代码如下：

```
CString sIECacheFile;//IE 缓存中的文件名
LPINTERNET_CACHE_ENTRY_INFO
lpCacheEntry = NULL;// 包含缓存信息的结构体指针
DWORD dwTrySize;//保存信息的结构体的缓存大小
BOOL bSuccess;//查找 IE 缓存后的返回值
bSuccess=FindFirstUrlCacheEntry(NULL,lpCacheEntry, &dwTrySize);//开始查找 IE 缓存文件
While(bSuccess)
{
//将缓存文件名(含路径)存入 sIECacheFile 中
sIECacheFile=lpCacheEntry->lpszLocalFileName;
//将文件名(去除路径后)与图片的模式文件名比较
if(sIECacheFile.Right(sPicture_Name_In_Cache.GetLength()) == sPicture_Name_In_Cache)
CopyFile(lpCacheEntry->lpszLocalFileName, sFilePathName, FALSE); //将图片存入指定目录中
DeleteUrlCacheEntry(lpCacheEntry->lpszSour
ceUrlName);//删除已经查询的 IE 缓存文件，提高下次查询的效率
bSuccess=(FindNextUrlCacheEntry(hCacheDir, lpCacheEntry, &dwTrySize);//查询下一个缓存文件
}
```

上述代码实现了在 IE 缓存中搜索并获取需要的图片对象，将它拷贝到指定路径下后，立刻清空缓存中的文件，提高了下次搜索 IE 缓存的效率。

5.4 图片对象入库

为了提高系统的效率，将图片对象存储在指定目录中，而数据库中仅存储图片对象的存储路径。

系统采用 ADO 方式与 Oracle 数据库进行连接，系统中设计了两个类：CADOConnection 和 CADORecordset 分别实现连接数据库和操作数据库的功能。入库操作通过执行相应的 SQL 语句实现，SQL 语句如下：

```
insert into system.BookDB(PictureName) values(sFilePathName)
```

5.5 多线程进行批量处理

以上程序实现了对单个页面图片的获取，如果要实现对海量信息的图片对象获取，需要分析其 URL 之间的关系。如样本页面中的区别在于 product_id 不同，可以根据将 product_id 设置为循环变量进行批量图片获取。若页面之间没有关系，则需将页面地址存放在文本文件中，下载时从文件中循环读出地址。

批量图片获取时，系统将运行较长的时间，且在运行期间用户无法进行操作。因此，必须采用多线程方式来实现批量处理。系统中设置了两个线程：主线程和图片获取线程。主线程负责系统的初始化和响应用户操作。图片获取线程复杂批量获取图片对象。

Visual C++中的 MFC 支持两类线程：工作者线程和用户界面线程。其区别在于工作者线程没有消息循环，而用户界面线程有自己的消息队列和消息循环[5]。由于系统中需要进行消息传递，因此采用用户界面线程来实现。下面代码实现了图片对象批量下载。

```
// 构造线程 CDownLoadThread 类：从 CWinThread 类中继承
class CDownLoadThread : public CWinThread{…}//线程类中有实现批量下载的代码，原理同上，略。
//定义指向 CDownLoadThread 的指针
CDownLoadThread* m_pDownLoadThread;
//在主线程中启动线程
m_pDownLoadThread=(CDownLoadThread*)AfxBeginThread(RUNTIME_CLASS(CDownLoadThread));
```

5.6 线程同步

多线程实现了图片对象的批量下载，但还有不少问题需要解决。比如，用户在主线程中要结束图片下载线程，但图片下载线程正在下载某个图片对象，如果立刻中止它，它的很多资源就会来不及释放，造成系统的浪费，多次操作后甚至会造成系统的崩溃。这就需要线程同步来解决这个问题。本系统采用事件(CEvent)来实现线程同步。

事件是一个允许一个线程在某种情况发生时，唤醒另外一个线程的同步对象。在 MFC 中，CEvent 类对象有两种类型：人工事件和自动事件。一个自动 CEvent 对象在被至少一个线程释放后会自动返回到无信号状态；而人工事件对象获得信号后，释放可利用线程，但直到调用成员函数 ReSetEvent()才将其设置为无信号状态。本系统中采用自动事件可实现线程同步。

```
//在主线程中定义全局变量：图片对象下载完成事件
CEvent eventDownLoadComplete;
//在主线程中止下载响应函数中设置等待事件信号，等待时间为无限
WaitForSingleObject(eventDownLoadComplete.m_hObject,INFINITE);
//在下载线程中的 OnDownloadComplete 响应函数中设置事件信号
eventDownLoadComplete.SetEvent();
```

上述代码实现了线程同步，当用户想中止下载时，启动等待下载完成事件信号函数 WaitForSingleObject()，当正在下载的图片对象下载完成时，触发事件信号，释放占有的资源后，结束下载。

6 结语

本文介绍了一种基于 IE 缓存的 Web 图片获取方法，解决了以往的网页数据抓取仅是针对文本信息而无法获取图片对象的问题，在实际中对用户非常有用。同时，该方法实现了对各种来源的图片对象的获取，而不必考虑图片对象是何种服务器端组件(ASP、JSP 或 Servlet 等)查询后返回的，在设计上有一定的创新性。但如何利用尽量少的样本，取得比较好的效率，如何从 HTML 网页中或者其它的信息实体中得到粒度更细的对象，仍需要进一步的研究。

参考文献

1 任仲晟,薛永生. 基于页面标签的Web结构化数据抽取.计算机科学,2007,10:133－136.

2 胡晓元,史浩山. WinPcap 包截获系统的分析及其应用.计算机工程,2005,2:96－98.

3 刘书华,陈国奎.基于 PowerBuilder 的网页数据抓取.计算机系统应用, 2009,18(2):171－175.

4 王福建,郭月强,焦祝军,等. Visual C++6.0 编程高手.北京:希望电子出版社, 2000.144－146.

5 Kruglinski DJ, Wingo S, Shepherd G. Visual C++6.0 技术内幕.北京:希望电子出版社, 1999.260－275.