# 基于 C++ 的网络爬虫的设计与实现

宋子衿　白俊鸽
成都锦城学院计算机与软件学院 四川 成都 611731

【摘要】随着设计行业以及人工智能产业逐渐发展，人们对于各种图片的需求量也与日俱增。但手动一个一个的下载费时费力，因此就需要使用爬虫来收集图片。本文讲述了一款基于 C++ 的爬虫的设计与具体实现，该爬虫主要针对 http 协议的网站，通过获取用户输入的网址进行进一步分析，进而自动爬取图片。

【关键词】爬虫；C++；http；图片

## 1 概述

在现代，设计行业需要大量的图片作为素材或者进行参照。人工智能领域实现各种训练也可能会需要用到大量的图片，而在庞大的网络中对图片资源进行搜集和下载成为了困扰许多人的问题之一。网络爬虫是依靠预先设定好的规则而进行自动的抓取网络信息的过程。[1] 该爬虫项目主要对 http 协议网站中的图片进行爬取，通过获取网站的 html 分析查找其中的 src，通过找到的 src 进行图片的批量下载。

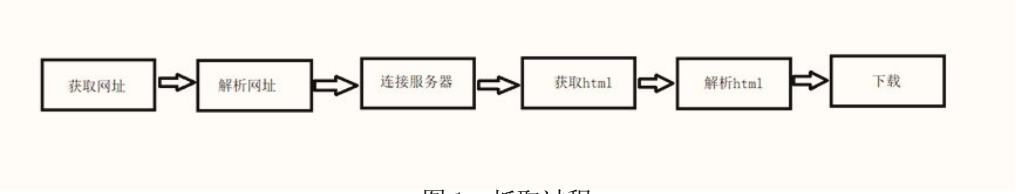


图 1：抓取过程

该爬虫可以粗略的分为获取网址、解析网址、连接服务器、获取 html、分析 html、下载六大部分。该爬虫可以还对一些网站中出现的 src 不完整的现象进行识别和判断，使得该爬虫对不同网站的适应性得到了提升。

## 2 获取和分析网址

程序开始运行时使用 CreateDirectory() 函数创建一个文件夹，以供储存爬取的图片。接下来程序会提醒用户输入要爬取的网址，让后将用户输入的网址存入一个 string 类型的变量中以供后续使用。

接下来将网址装入到一个 char 类型的数组中并通过 strstr() 函数检查输入的网址是否为 http 协议的网址。如果确认输入的网址为 http 协议的网址，那么接下来就通过 sscanf() 函数找到“http://”后面的第一个“/”，sscanf() 函数可以找到一个字符串中的指定的子串，并分别将找到的子串的前后部分装入到不同的变量中去。通过寻找“http://”后面的第一个“/”，我们可以获取其前后两部分的内容，前面的就是主机名，后面的则是资源路径。

## 3 连接服务器

接下来我们就要进行连接服务器并发送 get 请求的操作。

我们首先初始化套接字库，并创建一个套接字。然后使用 gethostbyname() 解析主机 IP，并使用指针 p 接收其返回值。

然后声明一个 sockaddr_in 变量，sockaddr_in 是一种结构体，它是由系统封装的，其中的成员变量 sin_family 表示使用哪种地址族、sin_port 则被用来保存端口号、sin_addr 则保存了 IP 地址的相关信息。我们将其 sin_family 设置为 AF_INET，表示要使用 TCP/IP 协议；将其 sin_port 设置为 htons(80)，表示使用 http 默认的 80 端口号。再将指针 p 中的 h_addr 传给 sokaddr_in 变量的 sin_addr。最后通过 connect() 函数对服务器进行连接，connect() 函数通过套接字和 sockaddr_in 变量作为参数请求连接，如果成功则返回 0，连接失败则会返回错误码。

如果连接成功，就进行下一步——发送 get 请求。这里要先声明一个 char 类型的数组存储 get 请求，并使用 sprintf() 函数将请求的内容装入数组中；需要装入的内容如下：”GET %s HTTP/1.1\r\nHost: %s\r\nConnection: Close\r\n\r\n”。这里的第一个“%s”是之前分析出来的资源路径，第二个则是主机名。

装入后则要通过系统调用函数 send() 来进行发送。该函数需要程序正处于链接状态时才能使用，因此放在 connect() 的后面，然后通过加入套接字、装有 get 请求的 char 类型数组以及 strlen() 函数计算出的数组的大小作为为参数发送 get 请求到服务器。

## 4 获取并解析 html

爬取目标网站之前，首先要确定采集目标，明确采集字段的位置信息和逻辑结构。[2] 因此我们在发送完 get 请求后就需要获取 html，首先要声明一个数组和一个字符串来存储 html。接下来则通过 recv() 函数接收从服务器返回的数据，并通过 recv() 函数的返回值为判断条件的 while 循环判断是否仍有数据发送过来，并在每次循环中将数组中的数据装入字符串中。如果数据传输结束，循环就会终止，最终我们将得到该网页完整的

html。

获取到网页的 html 后，我们需要找到其中表示图片的 src，并根据 src 所给出的 url 进行图片下载。因此，我们需要一个正则表达式来进行判断，寻找符合条件的 src，正则表达式的内容如下：”src=\”(.*?\\.jpg)\””。另外我们还要通过一个 smatch 变量 mat 获取符合表达式的部分。

我们需要 regex_search() 函数进行搜索匹配，将开始和结尾设置为 html 的开始和结尾，并通过 while 循环不断查找，放入 mat 中去。在循环内，我们将找到的 src 通过 mat[1].first 和 mat[1].second 转换为 string 变量，并作为参数传入到负责下载的函数中去。

首次负责下载的函数调用后我们将 mat[0].second 设置为开始，意为将找到的符合表达式的第二个部分的开头设置为开始。接下来的不断循环会将所有符合表倒是条件的 src 传入到下载函数中去进行下载，下载结束后的将 mat[0].second 设置为开始的这一步骤会使每次循环的查找范围缩小，直到查找范围中没有符合条件的 src，至此所有图片将会被下载完成。我们就爬取到了该网页的所有图片信息。

## 5 下载

获得到 src 后将它传给负责下载的函数，但是有些 src 并不能用于直接下载，我们需要将 src 补全为 url 后再进行进一步的操作。url 需要有协议名以及主机名，但有些 src 缺少这两部分或其中的一个部分。

```
char* pos2 = strstr(szurl, "http://");
char* pos3 = strstr(szurl, ".com");
char* pos4 = strstr(szurl, "data-original=");
if (pos4 != NULL)
{
    char t_url[256],y_url[256];
    int a = sscanf(szurl, "%[^=]%s", y_url,t_url);
    char* rt_url = t_url + 2;
    int b = sizeof(rt_url);
    string t_str(rt_url);
    url = t_str;
}
if (pos2 == NULL)
{
    if (pos3 == NULL)
        url = "http://" + str + url;
    else
    {
        if (szurl[0] == '/')
        {
            url = "http:" + url;
        }
        else
            url = "http://" + url;
    }
}
```

图 2：需要补全的几种情况

此外，我们在获取 src 时还会获取到后面有 data-original 的 src，例如：“src=“//www.yesky.com/TLimages2009/yesky/images/delaypic.gif” data-original=”d-pic-image.yesky.com/105x140/uploadImage/2019/007/05/RO9WXOPGB9N_H.jpg””这种情况需要取 data-original= 后面的部分作为 src 才能进行正常下载，否则就会下载失败。我们通过 strstr() 函数如果找到获取的 src 中存在 data-original 后就使用 sscanf() 函数获取到 data-original= 后面的部分进行进一步处理。

接下来我们遇到的有些 src 有以下几种情况，首先是没有“http://”也没有主机名的，其次是没有“http://”但是有主机名的，再者就是有主机名没有“http:”但是有后面的“//”的。我们将使用 strstr() 函数判断出这几种情况，并根据判断的结果对 src 进行相应的补全，以使其成为能进行正常下载图片的 url。

随后我们将用到 URLDownloadToFile() 函数对图片进行下载，但该函数需要图片的存储路径和存储名称，且该函数下载的图片同名时会相互覆盖，因此我们要采用尽量不重复的图片存储名称。我在这里采用的是 substr() 函数截取 src 中的图片名称来解决的，首先通过 find_last_of() 函数确定最后一个“/”所在的位置，之后将位置传给 substr() 函数，该函数会自动截取位置后面的内容，并通过一个 string 类型的变量接收截取的部分作为图片的保存名称。最后将保存名称和之前创建的文件夹的路径加在一起，就成为了图片最终的保存路径。

URLDownloadToFile() 函数中所需的 url 以及保存路径都需要宽字符的字符串，我们现在的 url 和保存路径并非这种类型。因此，我们要使用函数 MultiByteToWideChar() 对这两个字符串进行转换。该函数需要被转换字符的缓冲区的大小作为参数，但我们事先并不知道这个值，因此我们要先将该函数的倒数第二个参数设置为空、倒数第一个参数设置为 0，这样该函数的功能就会变成返回转换字符的缓冲区所需要的大小。之后我们将再次调用该函数，这次该函数的倒数第二个变量将会设置为一个宽字符变量，倒数第一个变量则是之前获得的缓冲区的大小。只有我们将用此种方式将 url 和存储路径都转换为宽字符类型的字符串。

上述准备完成后我们就开始用 URLDownloadToFile() 函数来进行下载，该函数有五个参数，第一个参数是 ActiveX 组件的接口地址，我们这里用不到，将它设置为空；第二个参数就是 url，第三个是存储路径；第四个为保留值，设置为 0；第五个则是 ibindstatuscallback 接口地址，这里设置为空即可。之后该函数会根据所给的 url 和保存路径将图片下载到指定的路径中去，通过函数的返回值判断是否下载成功并返回提示。通过解析 html 中的 while 循环不断地将获得的 src 传入下载函数进行处理和下载，我们就能得到该网页的所有图片。

## 6 深度爬取

以上的过程为爬取一个网页内所有图片的过程，但仅能爬取用户输入网址的那个网页，而当我们有更大量的图片需求时，我们则希望该爬虫能够对其他的相关网页内的图片也进行爬取。因此，根据该需求我们可以声明一个队列，在获取 html 后、解析 src 之前插入一个步骤，来实现对其他相关网页的深度爬取。

在获取到 html 后我们采用和寻找 src 一样的方式，用正则表达式找到 html 中的 href，表达式内容有如下两种情况：”href=\”(.*?\\.html)\””、”href=\”(.*?\\.shtml)\””。这些都

是与输入网址相关的网页，但他们有的也和 src 一样缺少协议名或者是主机名，这里我们依旧采用处理 src 的方式将缺少的部分进行补全，随后我们将补全后的网址存入到一个队列中去。在一个页面的内容爬取结束之后，程序将会自动从队列中取出网址，然后进行上述循环，不断进行图片的爬取。

在对图片的需求较大时，这种方法能使你在只输入一个网址的情况下爬下所有相关网页的图片，在短时间内获得大量图片。

## 7 结束语

爬虫可以高效提取有价值的信息，降低技术成本，提高业务效率。[3] 本文详细阐述了基于 C++ 实现的针对 http 协议的网络爬虫的设计思路和实现过程，该程序的结构清楚，思路清晰，适于刚接触爬虫和 C++ 网络编程的新手学习。

同时该项目代码量不大，开发成本低；也具有较大的改造空间，当遇到一些本项目中未考虑到的情况时，可以通过添加原程序中未出现的情况来使该项目的适用范围增加。

## 【参考文献】

[1] 杨月．Python 网络爬虫技术的研究 [J]. 电子世界 ,2021,(10):57-58.

[2] 陈皓，周传生．基于 Python 和 Scrapy 框架的网页爬虫设计与实现 [J]. 电脑知识与技术 ,2021,17(13):3-5.

[3] 黄子豪，张舒．网络爬虫对互联网安全的影响及“反爬”策略的研究 [J]. 科学技术创新 ,2021,(10):120-121.