采集器对比

Beyebe·小蜜蜂

一个能打的对手都没有

在很长一段时间里，采集器只有能采和不能采的问题，把数据采得 “全” 和 “准” 我认为都是运营的事，和程序本身并没有太大关系，但是对三款市面上的主流采集软件的试用改变了我的认识。

> 都是垃圾，一个能打的对手都没有。
> 都是垃圾，一个能打的对手都没有。
> 都是垃圾，一个能打的对手都没有。

-	小蜜蜂	火车头	八爪鱼	神箭手
多个起始地址	√	√	×	√
探索式翻页	√	×	×	×
增量不为1的翻页	√	×	×	×
获取多种页面类型中的数据	√	√	√	×
对将要采集的url做预处理	√	√	×	×
在起始地址中添加自定义属性	√	×	×	×
事务型采集Step	√	×	×	×
抽取结果检查	√	×	×	×
中心化的操控管理	√	×	×	√
采集配置	√	×	×	√
程序日志	√	×	×	√
采集日志	√	×	×	×
单任务中多线程控制	√	×	×	×
任务按策略自动启动	√	×	×	√
单任务HTTP设置	√	×	×	√
JS注入	√	×	×	√
通过JS判断页面是否已加载到位	√	×	×	×

> 注：由于神箭手的采集任务没有可视化配置界面，使用纯代码编写，以上功能，如果写代码需要超过三行，即认为不具备该功能。

以上是部分功能有没有的对比，除此之外，能做，但是做到的程度不同,这里面的差距也是巨大。

不分轻重缓急，不分先后顺序，挑几个点展开对比一下。

探索式翻页

对于HTML页面的采集，如果涉及到分页，可以使用 “发现下一页的地址” 或者 “点击下一页按钮” 等方法逐页采集。

但是对于json接口的采集，就没有办法从界面上去寻找关于 “下一页” 的元素。

这是百度外卖店铺列表页的数据接口：

> http://waimai.baidu.com/waimai/shoplist/2364115e829cbc29?taste=68&display=json&page=1&count=40

该接口获取的是某地点第一页的店铺列表，分页大小是40。小蜜蜂会根据配置进入下一页（page参数的值做+1），直到获取到的店铺数量小于40。

增量不为1的翻页

有些网站的分页接口并不是页码参数做+1进行递增，而是将数据偏移量以页面大小做递增。

以饿了么为例：

> https://mainsite-restapi.ele.me/shopping/restaurants?extras[]=activities&latitude=22.5483&longitude=113.94444&limit=24&offset=0

该接口获取的是某地点饿了么店铺列表数据，参数 `limit` 表示页面大小， `offset` 表示数据偏移量，使用将 `offset` 每次增加24的方式实现翻页。

对将要采集的url做预处理

没有这个功能就采集不了 `天猫`、`淘宝`、`亚马逊`、`当当` 等网站。

以下两个URL的内容，是等价的。

> http://product.dangdang.com/1265277530.html#ddclick?act=click&pos=1265277530_0_1_m&cat=4010476&key=&qinfo=&pinfo=&minfo=122_1_58&ninfo=&custid=&permid=2017list

等价于：

> http://product.dangdang.com/1265277530.html

假设我们的采集任务是采集当当网所有商品信息，由于相同的商品在不同的时间或者不同的页面进入，都会得到一个不同的url，这会带来一系列问题：

- 会采集到许多内容重复内容的页面。

- 蜘蛛的采集路径可能会进入死循环，使得蜘蛛永远在一个固定深度循环，最终只能采集到少量重复数据。
- 由于抽取到的所有url都与采集过的url不重复，使得采集任务永远无法结束。

所以，蜘蛛必须拥有将收集到的url处理成“标准格式”的能力。

在起始地址中添加自定义属性

没有这个功能就处理不了四个外卖平台的数据。

在百果园项目中，我们首先是对各城市整理出许多坐标点，一个点对应一个url，然后再使用蜘蛛爬取各坐标点周边的店铺信息。

在处理这些采集到的数据时，希望知道采集到的数据来自于哪个地点（经纬度、所在城市），那么在小蜜蜂系统中，可以在起始地址中对起始地址做若干标记，使得蜘蛛在整个过程中保持这一标记，以便于收到数据时获取这些业务上必须的信息。

如果不具备这样的功能，就不知道这些数据分别来自于哪个城市。

页面中JS注入

没有这个功能就采集不了美团外卖。

小蜜蜂支持在采集到的页面中执行JS，使用js执行加载数据、修改页面内容等行为。

通过JS判断页面是否已加载到位

没有这个功能就采集不了京东。

许多网站的页面内容是由js异步加载出来的，也就是说，打开页面之后，要等一会，内容才会被js加载并渲染呈现。

其它采集器要么不支持js渲染，要么粗暴地设置一个超时时间。

直接设置一个固定的时间有两个弊端，如果设置得太长，页面可能早就加载好了，剩下的时间做无谓的等待，非常浪费。如果设置得太短，超过时间之后不管有没有加载到位都直接结束页面的加载，会造成采集不到数据。

小蜜蜂支持通过js来判断页面是否加载到位，同时还支持超时，默认30秒。也就是说，在30秒内会不断检查页面是否加载到位，如果判断为true，会立即结束等待，返回页面内容。如果在30秒内一直等不到true，会示意该页面下载失败。

事务型采集

没有这个功能就采集不了百度外卖。

百度外卖需要先使用一个请求告知服务器客户端的经纬度，然后再访问该经纬度对应的地点，才能拿到该地点的店铺数据。

步骤1：

> http://waimai.baidu.com/waimai?qt=shoplist&lat=4828483.6823517&lng=12933686.676485&address=address

步骤2：

> http://waimai.baidu.com/waimai/shoplist/d246740c572f1145?taste=10&display=json&page=1&count=40

那么在多线程环境下，就需要两个功能来保证这件事按预期执行。
1. Cookie隔离。线程间不能共享Cookie，但是单线程内的多次url下载要共享Cookie。
2. 两个url在一个线程内按顺序执行，并且使用同一浏览器对象（由于小蜜蜂支持多种浏览器类型，所以这里仅仅在线程内共享Cookie是不够的）。

对采集路径的精确控制

如果没有这个，百果园、天虹等项目一个都做不了。

先设立一个场景，我们需要进入居家类目下的子类，但是不进入活动频道页：

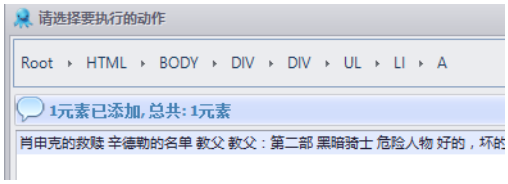
```
<h1>手机</h1>
<ul>
    <li><a class="first" href="https://search.jd.com/Search?keyword=老人机">老人机</a></li>
    <li><a href="https://search.jd.com/search?keyword=拍照神器">拍照神器</a></li>
    <li><a href="https://search.jd.com/search?keyword=双卡双待">双卡双待</a></li>
</ul>

<h1>居家</h1>
<ul>
    <li><a class="first" href="https://search.jd.com/Search?keyword=扫地机器人">扫地机器人</a></li>
    <li><a href="https://jieri.jd.com/" style="color:red">京东狂欢节</a></li>
    <li><a href="https://search.jd.com/search?keyword=置物盒">置物盒</a></li>
</ul>
```

神箭手：

```
var configs = {
    helperUrlRegexes: [
        "https://search.jd.com/search\?keyword="
    ],
};
```

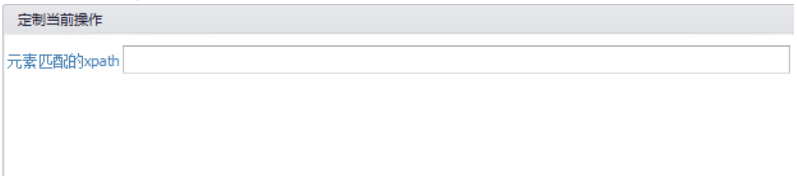
这里正则的匹配范围是整页，所以仅仅匹配url的形状，会抓到很多不需要的url。

八爪鱼：
八爪鱼有两种方式来选择需要进入的url。
方式一，在界面中点选：



很惭愧，点了很久也没能配置成想要的样子。这个方式Beyebe在五年前就证明是行不通的。

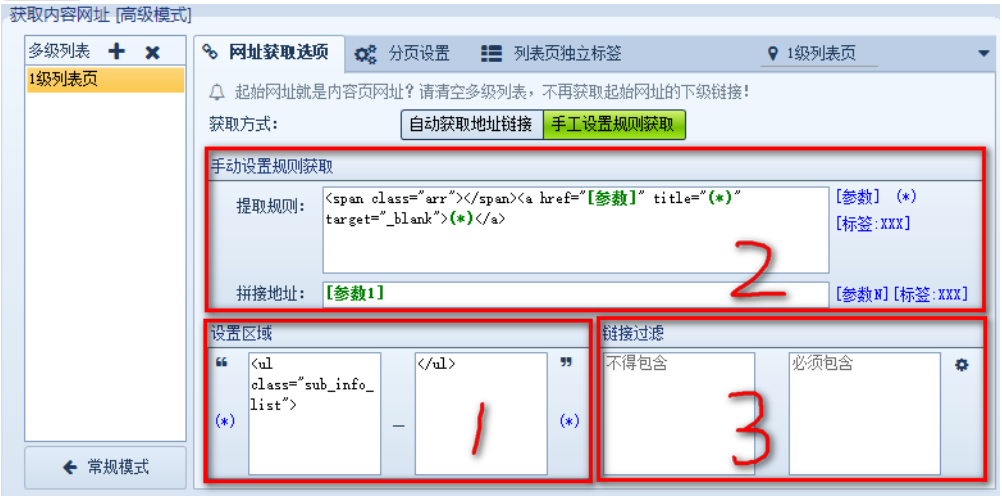
方式二，使用xpath：



xpath这种写表达式的方式非常精准，功能也强大，但是精准的范围只是对html节点的选取非常精准，在我们这个场景中，还是会抽取到不需要的url。

另外，八爪鱼这里只能填写一个表达式，如果要从页面中的多个区域寻找url就无能为力了。另，八爪鱼还只支持寻找a标签内href里的url，如果url不在这个地方，就无法爬取。

火车头：



火车头的配置思路就靠谱许多，先选择一个区域，然后再在这个区域内找满足格式的url，最后还能 “正选” 加 “反选” 双重过滤。

但是它的问题在于，寻找区域和提取规则这两件事都是基于字符串精确匹配来处理的，这会产生很多问题。

1. 如果有多个区域的格式是一样的，会一把抓，无法区分，所以依然不能满足上面预设的场景。
2. 鲁棒性非常差，目标网站在html源码内多一个空格或者换行，都会造成配置失效。
3. 对于不合规范的html会无法处理，如果希望选择的ul并没有结束标签。
4. 对于相似结构的嵌套，也处理不了。例如ul的li内如果还有ul，那么这种基于字符串精确匹配的方式是无法控制区域到底是外层的ul还是内层的ul。

关于对数据的抽取

按照正常的逻辑，蜘蛛应当不局限于采集一种页面类型，也就是说，假设我们去采集京东的购物网站，应当允许保存任意页面上的所需数据，商品列表页、商品详情页、商品评论页、店铺详情页、店铺活动页等所有页面。

火车头和神箭手都只支持在整个采集任务中获取一种数据。

火车头会对所有页面进行数据抽取，这也不科学。

关于对数据的检查

在一个页面中抽取数据，如果某属性抽取不到，找不到那个包含数据的html节点；或者找到的节点个数不符合预期；虽

然找到了指定的节点，但是节点里并不包含想要的数据……

由于网站改版，或者配置的时候没有做足够数量的抽样检查，都会导致实际做数据抽取时，所遇到的情况与用户的预期大相径庭。

在以上做对比的几款采集程序中，有的考虑到了这件事，但是没有提供反查和解决的途径，有的直接没有考虑这种情况。

直接没有考虑到这种情况的，在抽取不到数据的时候，会直接跳过，忽视掉这件事；还有的是抽到什么是什么，不检查。

考虑了这件事的，会输出日志，但日志内容仅仅只是一句“抽取失败，请检查”之类的话而已，用户根本不知道失败的时候发生了什么，到底是一个什么样的页面导致了抽取失败。

所以，一个合格的爬虫，应该把当时的情况告诉用户，当时采集到了一个什么样的页面，我们当时的配置是怎样的，结果这个配置在这个页面里没有达到什么样的预期，这样用户便可追溯可反查可修复。

演示，编写任务，用于采集Beyebe导购网站“手机通讯”类目下的所有商品信息。