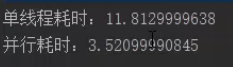
Python的并行化爬虫

# 使用map可以实现多线程爬虫

**代码**：（Python 2.x）

*#-\*-coding:utf8-\*-***from** multiprocessing.dummy **import** Pool **as** ThreadPool  
**import** requests  
**import** time  
  
**def** getsource(url):  
 html = requests.get(url)  
  
urls = []  
  
**for** i **in** range(1,21):  
 newpage = **'http://tieba.baidu.com/p/3522395718?pn='** + str(i)  
 urls.append(newpage)  
  
time1 = time.time()  
**for** i **in** urls:  
 print i  
 getsource(i)  
time2 = time.time()  
print **u'单线程耗时：'** + str(time2-time1)  
  
**pool = ThreadPool(4)  
time3 = time.time()  
results = pool.map(getsource, urls)  
pool.close()**  
pool.join()  
time4 = time.time()  
print **u'并行耗时：'** + str(time4-time3)

**结果**：



# 百度贴吧爬虫

（这是利用Python 2.x写的）

*#-\*-coding:utf8-\*-***from** lxml **import** etree  
**from** multiprocessing.dummy **import** Pool **as** ThreadPool  
**import** requests  
**import** json  
**import** sys  
  
reload(sys)  
  
sys.setdefaultencoding(**'utf-8'**)  
  
**'''重新运行之前请删除content.txt，因为文件操作使用追加方式，会导致内容太多。'''  
  
def** towrite(contentdict):  
 f.writelines(**u'回帖时间:'** + str(contentdict[**'topic\_reply\_time'**]) + **'\n'**)  
 f.writelines(**u'回帖内容:'** + unicode(contentdict[**'topic\_reply\_content'**]) + **'\n'**)  
 f.writelines(**u'回帖人:'** + contentdict[**'user\_name'**] + **'\n\n'**)  
  
**def** spider(url):  
 html = requests.get(url)  
 selector = etree.HTML(html.text)  
 content\_field = selector.xpath(**'//div[@class="l\_post l\_post\_bright "]'**)  
 item = {}  
 **for** each **in** content\_field:  
 reply\_info = json.loads(each.xpath(**'@data-field'**)[0].replace(**'&quot'**,**''**))  
 author = reply\_info[**'author'**][**'user\_name'**]  
 content = each.xpath(**'div[@class="d\_post\_content\_main"]/div/cc/div[@class="d\_post\_content j\_d\_post\_content "]/text()'**)[0]  
 reply\_time = reply\_info[**'content'**][**'date'**]  
 print content  
 print reply\_time  
 print author  
 item[**'user\_name'**] = author  
 item[**'topic\_reply\_content'**] = content  
 item[**'topic\_reply\_time'**] = reply\_time  
 towrite(item)  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 pool = ThreadPool(4)  
 f = open(**'content.txt'**,**'a'**)  
 page = []  
 **for** i **in** range(1,21):  
 newpage = **'http://tieba.baidu.com/p/3522395718?pn='** + str(i)  
 page.append(newpage)  
  
 results = pool.map(spider, page)  
 pool.close()  
 pool.join()  
 f.close()

# 使用scrapy 框架爬虫

**Scrapy内部使用的就是多线程爬虫**。



## items.py

import scrapy  
class BaidutiebaItem(scrapy.Item):  
 NAME = scrapy.Field()  
 TIME = scrapy.Field()  
 CONTENT = scrapy.Field()

## settings.py

ROBOTSTXT\_OBEY = False  
LOG\_FILE = './log.txt'  
ITEM\_PIPELINES = {  
 'BaiduTieba.pipelines.BaidutiebaPipeline': 300,  
}

## pipiline.py

class BaidutiebaPipeline(object):  
 num = 0  
 def process\_item(self, item, spider):  
 self.num += 1  
 f = open('./result.txt','a',encoding='utf-8');  
 f.write('\*\*\*\*\*\*\*\*\*第'+str(self.num)+"个回复\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")  
 f.write("NAME : "+item['NAME']+'\n')  
 f.write("TIME : "+item['TIME']+'\n')  
 f.write("CONTENT : "+item['CONTENT']+'\n'+'$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$'+'\n')  
 f.close()  
 return item

## tieba.py (spider)

# -\*- coding: utf-8 -\*-  
import scrapy  
from BaiduTieba.items import BaidutiebaItem  
import logging  
from bs4 import BeautifulSoup  
import json  
import string  
logger = logging.getLogger('ZhaoHong')  
  
class TiebaSpider(scrapy.Spider):  
 name = 'tieba'  
 # allowed\_domains = ['baidu.com']  
 num = 0  
  
 def start\_requests(self):  
 url = 'http://tieba.baidu.com/p/3522395718?pn=1'  
 yield scrapy.Request(url = url,callback=self.parse,meta={'flag':'True'})  
  
 def parse(self, response):  
 self.num += 1  
 logger.info('-----开始写源代码到文件中-------'+str(self.num))  
 soup = BeautifulSoup(response.text,'html.parser')  
 # self.write\_html(soup.prettify())  
 logger.info('-----写入完毕--------------')  
 item = BaidutiebaItem()  
 # 解析源代码  
 try:  
 divList = response.xpath('//div[starts-with(@class,"l\_post j\_l\_post l\_post\_bright")]')  
 dataList = divList.xpath('@data-field').extract()  
 logger.info(len(divList))  
 logger.info(len(dataList))  
  
 nameList = divList.xpath('div[@class="d\_author"]/ul[@class="p\_author"]/li[@class="d\_name"]'  
 '/a[@alog-group="p\_author"]/text()').extract()  
  
 logger.info(len(nameList))  
 contentList = divList.xpath('div[starts-with(@class,"d\_post\_content\_main")]/div[starts-with(@class,"p\_content")]/cc/div[starts-with(@class,"d\_post\_content")]/text()').extract()  
 logger.info(str(len(contentList))+"NNNNN")  
  
 except Exception as e:  
 logger.error(e)  
 # 解析结束  
 for i in range(len(nameList)):  
 item['NAME'] = nameList[i].strip()  
 dataDic = json.loads(dataList[i])  
 item['TIME'] = dataDic['content']['date'].strip()  
 item['CONTENT'] = contentList[i].strip()  
 yield item  
 if response.meta['flag']=='True':  
 for i in range(20):  
 if i < 2:  
 continue  
 yield scrapy.Request(url = response.url+'?pn='+str(i),callback=self.parse,meta={'flag':'False'})  
  
 def write\_html(self,html):  
 try:  
 logger.info('##############################')  
 with open('./html.html','w',encoding='utf-8') as f:  
 f.write(html)  
 except Exception as e:  
 logger.error(e)

结果：

