УДК 004.032.26:551.583.1
DOI: 10.36979/1694-500X-2024-24-4-18-25

РАЗРАБОТКА КОМПЛЕКСА ПРОГРАММ ПАРСЕРОВ ДЛЯ СБОРА КЛИМАТИЧЕСКИХ ДАННЫХ

Ж.С. Жукова, В.В. Ерофеева, Д.А. Кулагин, К.С. Шварцман, С.Л. Яблочников

Аннотация. Рассматривается разработка программного комплекса парсеров для сбора климатических данных, подчеркивая важность применения информационных технологий в контексте изменения климата. Обсуждается и анализируется актуальность проблемы глобального потепления, необходимость сбора и анализа климатических данных для прогнозирования будущих изменений. Основное внимание уделяется использованию Python для создания специализированных скриптов, способствующих автоматизации сбора данных с метеорологических станций по всему миру. Исследование подчеркивает важность эффективного использования программного обеспечения в экологических исследованиях и предлагает подходы к обработке и анализу больших массивов климатических данных.

Ключевые слова: парсинг; Python; климат; сбор данных; информационные технологии; глобальное потепление.

КЛИМАТТЫК МААЛЫМАТТАРДЫ ЧОГУЛТУУ ҮЧҮН ПАРСЕР ПРОГРАММАЛАРЫНЫН КОМПЛЕКСИН ИШТЕП ЧЫГУУ

Ж.С. Жукова, В.В. Ерофеева, Д.А. Кулагин, К.С. Шварцман, С.Л. Яблочников

Аннотация. Макалада климаттык маалыматтарды чогултуу үчүн талдоочулардын программалык комплексин иштеп чыгуу каралган. Климаттын өзгөрүшүнүн шартында маалыматтык технологияларды колдонуунун маанилүүлүгү баса белгиленген. Глобалдык жылуулуктун актуалдуулугу, келечектеги өзгөрүүлөрдү болжолдоо үчүн климаттык маалыматтарды чогултуу жана талдоо зарылдыгы талкууланып, талданат. Дүйнө жүзүндөгү метеорологиялык станциялардан маалыматтарды чогултууну автоматташтырууга көмөктөшкөн атайын скрипттерди түзүү үчүн Python ду колдонууга басым жасалат. Изилдөө экологиялык изилдөөлөрдө программалык камсыздоону натыйжалуу колдонуунун маанилүүлүгүн баса белгилейт жана климаттык маалыматтардын чоң массивдерин иштетүү жана талдоо ыкмаларын сунуштайт.

Түйүндүү сөздөр: парсинг; Python; климат; маалыматтарды чогултуу; маалыматтык технологиялар; глобалдык жылуулук.

DEVELOPMENT OF A SET OF PARSER PROGRAMS FOR CLIMATE DATA COLLECTION

Zh.S. Zhukova, V.V. Erofeeva, D.A. Kulagin, K.S. Shvartsman, S.L. Yablochnikov

Abstract. The paper discusses the development of a parser software package for climate data collection. Emphasizing the importance of information technology application in the context of climate change. The relevance of the global warming problem and the need to collect and analyze climate data for predicting future changes are discussed and analyzed. The focus is on the use of Python to create specialized scripts to facilitate automation of data collection from weather stations around the world. The study emphasizes the importance of effective use of software in environmental research and suggests approaches for processing and analyzing large climate data sets.

Keywords: parsing; Python; climate; data mining; information technology; global warming.

Развитие информационных технологий позволяет собирать большие массивы статистических, климатических, экономических и т. п. данных, которые постоянно обновляются и доступны в режиме он-лайн. При этом количество данных растёт экспоненциально, а для их сбора и обработки требуется применение новых подходов к анализу и разработка программного обеспечения.

Изменение климата является актуальной проблемой современного общества. Современная техносфера и цивилизация сформировались в течение голоцена (ок. 11700 лет) в условиях довольно тёплого климата. Изменение температурного режима на планете в сторону значительного потепления или похолодания приведёт к значительным изменениям в балансе биосферы, выработке первичной продукции, доступности ресурсов, что при высоком уровне потребления и численности человеческой популяции, приведёт к нарушениям в техносфере вплоть до её разрушения и угрозы вымирания человечества. При этом тенденция изменения климата – похолодание или потепление – практически не влияет на степень давления на техносферу [1, 2].

Накопленные массивы данных об изменении климата за историю Земли позволяют строить прогностические модели с использованием искусственного интеллекта [3]. Перед авторами стояла цель создать приложение для обработки климатических показателей, однако перед началом работы с данными их требуется скачать для удобной работы локально. Выполнение данной задачи требует написания специальных скриптов. Для работы был выбран язык программирования python (3 версии), который предоставляет множество удобных библиотек, прост в использовании, но при этом является мощным инструментом в решении прикладных задач. За счёт того, что python является интерпретируемым языком (то есть его код исполняется построчно), это делает его универсальным инструментом, код которого можно запускать практически на любой платформе, где установлен интерпретатор языка, преобразующий написанный код в машинные команды и исполняет их [4].

На первом этапе был написан комплекс программ парсеров для сбора числовых значений температур и осадков с сайта, предоставляющего доступ к архивным и текущим данным по температурам и осадкам с метеорологических станций по всему миру.

При разработке бота для скачивания необходимо изначально определить возможность скачивания данных с сайта, т. к. доступ может быть запрещён для скачивания и персонального использования (рисунок 1):

```
User-agent: Yandex
Disallow: /admin/
Disallow: /metar/
Disallow: /photo/
Disallow: /synop/
Disallow: /summary/
Disallow: /usasummary/
Disallow: /extremal.php?
Disallow: /extremal1.php?
Disallow: /dextremal.php?
Disallow: /dextremal1.php?
Host: www.pogodaiklimat.ru
```

Рисунок 1 – Содержание файла robots.txt,
директории в которые не может получить доступ боты

Для работы необходимо получить карту сайта: файл со ссылками на страницы с данными в виде таблицы. Логика работы программы: переход по ссылкам с названиями городов, регионов и занесение всех ссылок в файл для последующего использования.

Первая программа создаёт txt файл со ссылками через функцию map_site, посредством передачи в неё ссылки на сайт со страницей истории. Для удобства пользователей был написан терминальный интерфейс с вариантами выбора файлов и варианта работы скрипта (рисунок 2):

```
redsova@redsova-desktop:~$ python3 conf.py
1) Россия
2) Беларусь
3) Украина
4) Азербайджан
5) Армения
6) Грузия
7) Казахстан
8) Киргизия
9) Молдова
10) Таджикистан
Выполнение: 14.224%
```

Рисунок 2 – Запуск первого скрипта, все страны найдены, начат процесс составления карты ссылок, отображается прогресс выполнения

Видно, о каких странах мы можем получить информацию. Дальше идёт перебор всех стран по списку (текущая страна будет выведена в терминал), скрипт автоматически будет находить регионы и сами станции; также после этого можно увидеть процесс выполнения. После завершения вся имеющаяся информация записывается в файл conf.txt.1 в текущей директории.

Сначала в скрипт импортировали библиотеки и объявили 2 глобальные переменные, которые хранят 2 ссылки на сайт (на историю и на главную страницу соответственно) (рисунок 3):

```
1  #http://www.pogodaiklimat.ru/
2  import requests
3  from bs4 import BeautifulSoup
4  import sys
5  #import os
6
7  #Flags = True
8
9  link_orig="http://www.pogodaiklimat.ru/history.php"
10 link_orig1="http://www.pogodaiklimat.ru/"
11
```

Рисунок 3 – Начало файла conf.py (импорт библиотек и объявление глобальных переменных)

Функция main – главная функция программы, которая вызывает все другие, и в которой осуществляется запись полученных значений в файл (мною было принято решение записать массив через разделитель по средствам функции print). Также надо отметить, что запись в данном случае не происходит поточно. Сначала формируется массив с данными, а потом он целиком помещается в файл (каждая ссылка с новой строчки); в данном случае такое решение может быть применимо, поскольку скрипт выполняется относительно недолго (примерно 1–2 часа). Также следует отметить, что на работу скрипта может влиять достаточно много параметров: скорость интернет-соединения, скорость записи (в данной программе она почти мгновенная):

```python
def main():
    dict_dat = state_instaion(staion_country(map_site(link_orig)))
    mas_url=[]
    for i in range(1,len(dict_dat)):
        mas_url.append(str(link_orig1 + str(dict_dat[i][3]).strip('/')))
    mas_url1=remove_duplicates(mas_url)
    sys.stdout=open("conf.txt.1" ,"w")
    print(*mas_url1, sep='\n')

```

Рисунок 4 – Функция main

Функция get_html, которая принимает на вход ссылку на страницу выполняет GET-запрос к странице и возвращает html-код, полученный от сервера. Узнать метод, используемый на сайте, можно следующим образом: открыть код страницы, перейти на вкладку network(сеть) и посмотреть заголовки запроса, в Яндексе это можно сделать, нажав F12) (рисунки 5, 6):

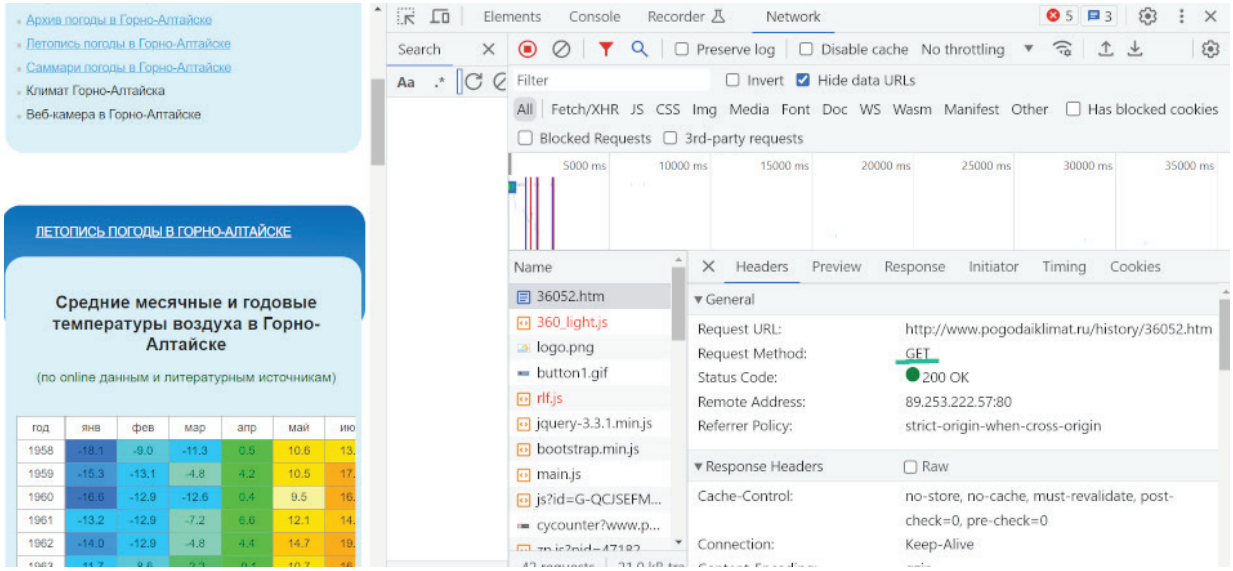

Рисунок 5 – Узнаём метод для запроса

```python
def get_html(link):
    response = requests.get(link)
    response.encoding = response.apparent_encoding
    return response.text

```

Рисунок 6 – Get_html

Функция remove_duplicates служит для удаления дубликатов в массиве и возвращения массива (рисунок 7):

```python
def remove_duplicates(array):
    # Создаем пустой словарь для хранения уникальных элементов
    unique_elements = {}

    # Проходим по каждому элементу в исходном массиве
    for element in array:
        # Добавляем текущий элемент в словарь с ключом равным самому элементу
        unique_elements[element] = None

    # Преобразуем словарь обратно в список
    array = list(unique_elements.keys())

    return array
```

Рисунок 7 – Функция remove_duplicates

Функция map_site принимает ссылку, обрабатывает (отступает от имеющихся на сайте ненужных нам гиперссылок) и заносит в словарь ссылки. Словарь имеет следующий вид: номер страны (объекта, указанного на сайте), в котором содержится массив из 2-х значений – название страны и ссылка на страницы с ней (рисунки 8 и 9):

Функция state_instaion отвечает за формирование словаря, ключом в котором является номер (порядковый номер записи). Формирование словаря происходит за счёт нескольких циклов, которые направлены на то, чтобы найти на странице нужные нам данные (такие как ссылки, название городов и станций). Именно они и являются значениями словаря и представляются в виде массива, который состоит из следующих элементов: ссылка на страницу с другими ссылками, но уже на станцию, страна, город/станция, ссылка на сам город и станцию. Пример такой конструкции представлен на рисунке 9:

После того как функция отработала, она возвращает результат в функцию main, в которой и происходит запись в другой словарь, который имеет область определения в рамках данной функции. После того как переменная получила значение с помощью цикла словаря и уже сохранённой ранее ссылки на главную страницу, происходит формирование уже готовых ссылок на страницы сайта, все они сохраняются в массиве mas_url1, который с помощью перенаправления стандартного потока вывода (sys.stdout) записывается в файл.

Результат работы функции state_instaion – словарь размером 5242968 байт, что составляет 40 Мбайт. Поскольку в файл записывается только часть этого словаря, то итоговый его вес 879 Кбайт, а состоит он из 19264 строчек, каждая из которых является ссылкой на страницу с климатическими данными. На рисунке 10 приведен результат работы первого скрипта:

Итог работы программы: все ссылки сохранились в файле conf.txt.1. Теперь его можно использовать для последующего анализа содержимого страниц, которые находятся по данным ссылкам. Также если немного видоизменить скрипт (например, на рисунке 4 в функции main элемент 3 заменить на 2, получим список стран, которые есть на сайте). Главная цель – собрать информацию, а точнее показать, как её можно собирать с помощью python – достигнута.

```
def map_site(link):
    dict_country = {}
    tmp_mas=[]

    text_orig = get_html(link)
    soup = BeautifulSoup(text_orig,'lxml')
    stations_rus = soup.find_all("li", class_="big-blue-billet__list_link")
    str_tmp = ""
    for i in range(len(stations_rus) - 16):#16
        href = stations_rus[i].find("a")['href']
        if not("profile" in href):
            str_tmp = stations_rus[i].text
            tmp_mas.append(str_tmp)
            tmp_mas.append(href)
            dict_country[i+1]=tmp_mas
            tmp_mas = []
            #print(href)
        else:
            continue

    #print(dict_country)
    for j in range(len(dict_country)):
        print(j + 1, end=") ")
        print(dict_country[j + 1][0])

    #country_number = int(input("Введте номер страны (0-все страны): "))
    country_number=0#del in prod
    dict_station={}
    mas_new_tmp=[]
```

```
    if country_number == 0:
        for i in range(len(dict_country)):
            #print(i+1, end=") ")
            #print(dict_country[i+1][0],end=" ")
            lin_new=link_orig1+dict_country[i+1][1]
            mas_new_tmp.append(dict_country[i+1][0])
            mas_new_tmp.append(lin_new)
            dict_station[i+1]=mas_new_tmp
            mas_new_tmp=[]
            #print(lin_new)
    #print(dict_station)
    '''
    for i in range(len(dict_station)):
        print(dict_station[i+1])
    '''
    return dict station
```

Рисунок 8 – Составление карты сайта (словаря)

```python
def state_instaion(dict_staions_in_country, saved_i=0):
    mas_data=[]
    dict_data={}
    count=0
    for i in range(saved_i, len(dict_staions_in_country)):
        for j in range(len(dict_staions_in_country[i])):
            #if not("?" in dict_staions_in_country[i][j][1]):
            link_orig_tmp = link_orig1.strip("/") + dict_staions_in_country[i][j][1]
            #print(dict_staions_in_country[i][j][2], end=" ")
            #print(dict_staions_in_country[i][j][0], end=" ")
            #print(link_orig_tmp)
            if (("?" in link_orig_tmp) and not("#" in link_orig_tmp) and not("profil" in link_orig_tmp)):
                html=get_html(link_orig_tmp)
                soup = BeautifulSoup(html,'lxml')
                regions = soup.find_all("a", class_="")

                for reg in regions[21:-29]:
                    if not("profile" in reg["href"]):
                        count+=1
                        mas_data.append(link_orig_tmp)
                        #mas_data.append(dict_staions_in_country[i][j][2])
                        mas_data.append(dict_staions_in_country[i][j][0])
                        mas_data.append(reg.text)
                        mas_data.append(reg["href"])
                        #print(dict_staions_in_country[i][j][2], end=" ")
                        #print(dict_staions_in_country[i][j][0], end=" ")
                        #print(reg.text,end=" ")
                        #print(reg["href"])
                        dict_data[count]=mas_data
                        mas_data=[]
        sys.stdout.write('\r' + "Выполнение: " + str(round(float(i/len(dict_staions_in_country) * 100),nd
    return dict_data
```

Рисунок 9 – Функция state_instation

```
http://www.pogodaiklimat.ru/history/37477.htm
http://www.pogodaiklimat.ru/history/37251.htm
http://www.pogodaiklimat.ru/history/37479.htm
http://www.pogodaiklimat.ru/history/37472.htm
http://www.pogodaiklimat.ru/history/37259.htm
http://www.pogodaiklimat.ru/history/37474.htm
http://www.pogodaiklimat.ru/history/37089.htm
http://www.pogodaiklimat.ru/history/37475.htm
http://www.pogodaiklimat.ru/history/37461.htm
http://www.pogodaiklimat.ru/history/37079.htm    358,1          1%
```

Рисунок 10 – Сохранение ссылок в скрипт

Заключение. В статье проанализирована значимость разработанного программного комплекса парсеров для мониторинга и анализа климатических данных в контексте глобального изменения климата. Использование автоматизированных инструментов для сбора и анализа данных позволяет более эффективно прогнозировать климатические изменения, что критически важно для разработки стратегий адаптации и смягчения последствий. Исследование открывает новые перспективы для дальнейших разработок в области экологического мониторинга и предлагает основу для глубокого изучения взаимосвязей между климатом и экосистемами.

Поступила: 12.02.24; рецензирована: 27.02.24; принята: 29.02.24.

Литература

1. *Weninger B. et al*. Climate forcing due to the 8200 cal yr BP event observed at Early Neolithic sites in the eastern Mediterranean / B. Weninger et al. // Quaternary Research. № 66. Pp. 401–420 (2006).
2. *Котляков В.М.* Что нового мы узнали о снеге и льде в Антарктиде в период Международного геофизического года и в последующие 10–20 лет / В.М. Котляков // Вопросы географии. 2020. № 150. С. 75–99. EDN HCWXAM.
3. *Жукова Ж.С.* Исследование вариативности температурных показателей Антарктиды / Ж.С. Жукова, В.В. Ерофеева // Вопросы науки. 2023. № 3. С. 53–57. EDN WRUQZV.
4. *Коржов А.Н.* Парсинг как способ автоматизированного поиска информации / А.Н. Коржов // Матер. межд. научн.-технич. конф. молодых ученых БГТУ им. В.Г. Шухова, Белгород, 01–20 мая 2019 г. Белгород: Белгор. гос. технол. ун-т им. В.Г. Шухова, 2019. С. 2932-2935. EDN WNXOZZ.