**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра Информатики

**Отчет по лабораторной работе №8**

по предмету «Кроссплатформенные технологии программирования»

на тему:

«Модифицированный веб-сканер»

Выполнил: студент группы БВТ 1802

Молчанова А.А.

Руководитель:

Полянцева К.А

Москва 2020

**Цель работы**

В данной лабораторной работе нужно расширить сканер для использования поточной обработки JAVA так, чтобы несколько веб-страниц можно было сканировать параллельно. Это должно привести к значительному повышению производительности, так как время, которое каждый поток сканера тратит на ожидание завершения сетевых операций, может прерываться другими операциями обработки в других потоках.

**Задания**

1. Реализовать класс с именем URLPool, который будет хранить список всех URL-адресов для поиска, а также относительный "уровень" каждого из этих URL-адресов (также известный как "глубина поиска");
2. Необходимо создать класс CrawlerTask, который реализует интерфейс Runnable;
3. Изменить программу так, чтобы она принимала третий параметр через командную строку, который будет определять количество порождаемых потоков веб-сканера;
4. Веб-сканер не должен постоянно опрашивать пул URL-адресов в случае, если он пуст;
5. Обновить пару URL-Depth для использования класса java.net.URL и произведите соответствующие изменения в веб-сканере для того, чтобы он соответствовал и относительным URL-адресам, и абсолютным;
6. Реализовать список URL-адресов, которые были просмотрены, и избегайте возврата к ним. Используйте один из классов коллекций java;
7. Добавить другой дополнительный параметр командной строки.

**Коды программ**

***CrawlerTask.java***

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.Socket;

public class CrawlerTask implements Runnable{

URLPool urlPool;

public static final String URL\_PREFIX = “<a href=\”http”;

public CrawlerTask(URLPool pool){

this.urlPool = pool;}

public static void request(PrintWriter out,URLDepthPair pair){

out.println(“GET “ + pair.getPath() + “ HTTP/1.1”);

out.println(“Host: “ + pair.getHost());

out.println(“Connection: close”);

out.println();

out.flush();}

public static void buildNewUrl(String str,int depth,URLPool pool){

try {

String currentLink = str.substring(str.indexOf(URL\_PREFIX)+9,str.indexOf(“\””, str.indexOf(URL\_PREFIX)+9));

pool.addPair(new URLDepthPair(currentLink, depth + 1));}

catch (StringIndexOutOfBoundsException e) {}}

@Override

public void run(){

while (true){

URLDepthPair currentPair = urlPool.getPair();

try{

Socket my\_socket = new Socket(currentPair.getHost(), 80);

my\_socket.setSoTimeout(1000);

PrintWriter out = new PrintWriter(my\_socket.getOutputStream(), true);

BufferedReader in = new BufferedReader(new InputStreamReader(my\_socket.getInputStream()));

request(out,currentPair);

String line;

while ((line = in.readLine()) != null){

if (line.indexOf(currentPair.URL\_PREFIX)!=-1){

buildNewUrl(line,currentPair.getDepth(),urlPool);}}

my\_socket.close();}

catch (IOException e) {}}}}

***ScannerApp.java***

import java.util.LinkedList;

public class ScannerApp {

public static void showResult(LinkedList<URLDepthPair> resultLink){

for (URLDepthPair c : resultLink)

System.out.println(“Глубина :” + c.getDepth()+”\tСсылка :”+c.getURL());}

public static boolean checkDigit(String line){

boolean isDigit = true;

for (int © = 0; © < line.length() && isDigit; i++)

isDigit = Character.isDigit(line.charAt(i));

return isDigit;}

public static void main(String[] args){

args = new String[]{“http://htmlbook.ru/”, “3”, “1”};

if (args.length == 3&&checkDigit(args[1])&&checkDigit(args[2])){

String lineUrl = args[0];

int numThreads = Integer.parseInt(args[2]);

URLPool pool = new URLPool(Integer.parseInt(args[1]));

pool.addPair(new URLDepthPair(lineUrl, 0));

for (int © = 0; © < numThreads; i++) {

CrawlerTask c = new CrawlerTask(pool);

Thread t = new Thread©;

t.start();}

while (pool.getWait() != numThreads) {

try {

Thread.sleep(500);

} catch (InterruptedException e) {

System.out.println(“Ignoring InterruptedException”);

}}

try {

showResult(pool.getResult());;}

catch (NullPointerException e){

System.out.println(“Not Link”);}

System.exit(0);}

else{

System.out.println(“usage: java ScannerApp <URL> <maximum\_depth> <num\_threads> or second/third not digit”);}}}

***URLDepthPair.java***

import java.util.LinkedList;

import java.net.MalformedURLException;

import java.net.URL;

public class URLDepthPair {

public static final String URL\_PREFIX = “<a href=\”http”;

public String URL;

private int depth;

URL host\_path;

public URLDepthPair (String URL, int depth){

this.URL=URL;

this.depth=depth;

try {

this.host\_path= new URL(URL);}

catch (MalformedURLException e) {

e.printStackTrace();}}

public String getHost(){

return host\_path.getHost();}

public String getPath(){

return host\_path.getPath();}

public int getDepth() {

return depth;}

public String getURL() {

return URL;}

public static boolean check(LinkedList<URLDepthPair> resultLink, URLDepthPair pair) {

boolean isAlready = true;

for (URLDepthPair c : resultLink)

if (c.getURL().equals(pair.getURL()))

isAlready=false;

return isAlready;}}

***URLPool.java***

import java.util.LinkedList;

public class URLPool {

LinkedList<URLDepthPair> findLink;

LinkedList<URLDepthPair> viewedLink;

int maxDepth;

int cWait;

public URLPool(int maxDepth){

this.maxDepth = maxDepth;

findLink = new LinkedList<URLDepthPair>();

viewedLink = new LinkedList<URLDepthPair>();

cWait = 0;}

public synchronized URLDepthPair getPair(){

while (findLink.size() == 0) {

cWait++;

try{

wait();}

catch (InterruptedException e){

System.out.println(“Ignoring InterruptedException”);}

cWait--;}

URLDepthPair nextPair = findLink.removeFirst();

return nextPair;}

public synchronized void addPair(URLDepthPair pair){

if(URLDepthPair.check(viewedLink,pair)){

viewedLink.add(pair);

if (pair.getDepth() < maxDepth){

findLink.add(pair);

notify();}}}

public synchronized int getWait(){

return cWait;}

public LinkedList<URLDepthPair> getResult(){

return viewedLink;}}

**Результат работы кода**

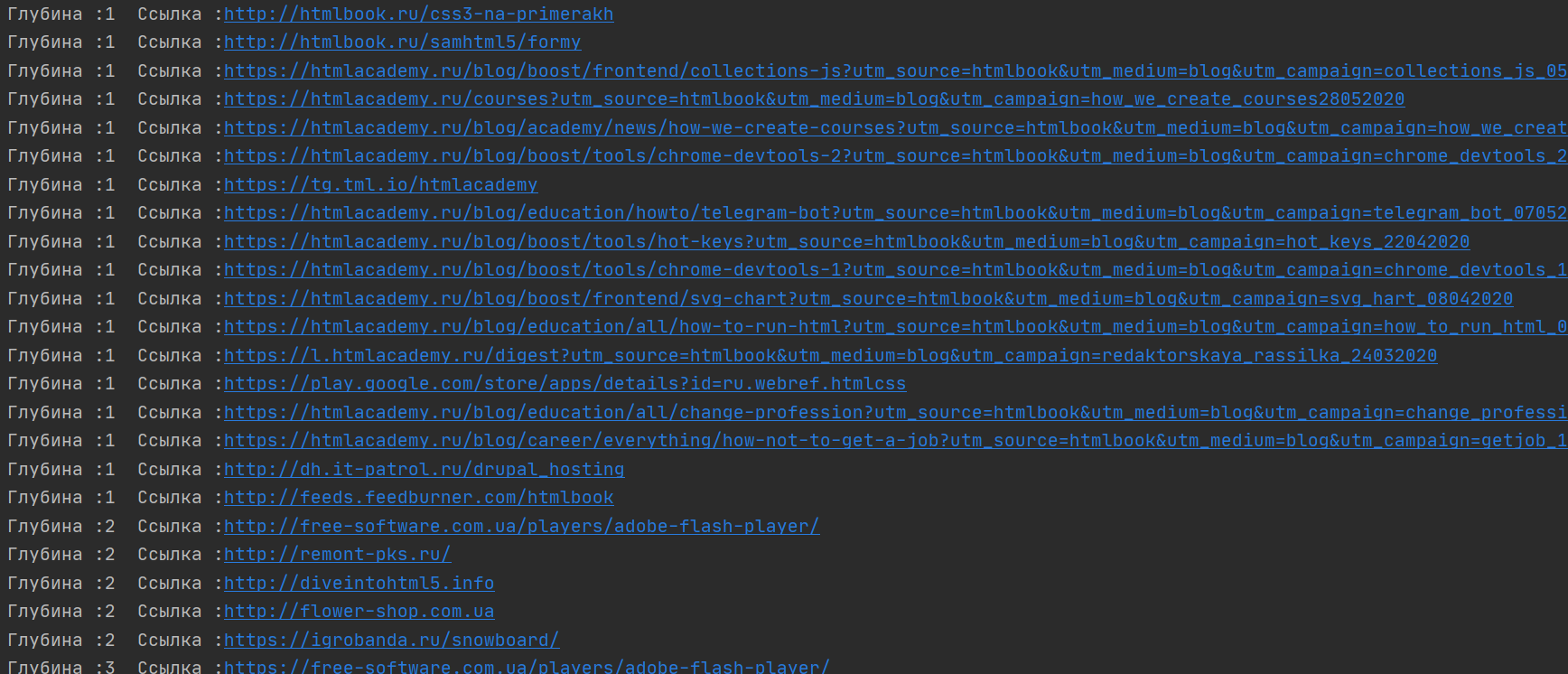


Рисунок 1 – Результат работы программы веб-сканера

**Вывод**

В ходе лабораторной работы был реализован класс с именем URLPool, который должен хранить список всех URL-адресов для поиска, создан класс CrawlerTask, который реализует интерфейс Runnable. Программа была изменена так, что она принимает третий параметр через командную строку, который будет определять количество порождаемых потоков веб-сканера. Веб-сканер не опрашиваеь пул URL-адресов в случае, если он пуст. Обновлена пара URL-Depth для использования класса java.net.URL, реализован список URL-адресов, которые были просмотрены, добавлен другой дополнительный параметр командной строки.