

北京邮电大学课程设计报告

课程设计名称	计算机网络课程设计	学 院	计算机	指导教师	吴起凡
班 级	班内序号	学 号	学生姓名	成绩	
2019211319	25	2019212179	彭笑凯		
2019211319	15	2019211865	陈伟杰		
2019211319	19	2019211898	王天一		
课 程 设 计 内 容	<p>① 实验目的：设计一个 DNS 服务器程序，能够读入缓存在本地“域名-IP 地址”对照表，并根据其实现具有不良网站拦截、服务器以及中继功能的 DNS 服务器。</p> <p>② 实验内容：本组基于 C 语言标准库和 Socket 库对 DNS 服务器进行开发，利用红黑树这一数据结构实现了数据的高效查找和存储，完成了不良网站拦截、由本地缓存结果的服务器应答、中继等基本功能，并采用了 LRU 算法的 cache 对中继的结果进行缓存。同时，为了提高服务器处理多客户端的并发性，本程序基于 POSIX 标准库 pthread 进行多线程开发，实现了一个线程池便于对多线程的管理。</p> <p>③ 团队分工：程序整体设计、DNS 报文分析、网络编程及多线程模块主要由彭笑凯进行设计，具体代码编写共同实现，红黑树模块、DNS ID 映射模块、计时器模块主要由陈伟杰进行开发，LRU Cache 模块、命令行处理模块主要由王天一进行开发。程序处理具体细节、后续调试和测试由所有组员共同确认和敲定。</p>				
学生 课程设计 报告 (附页)	设计报告及结果见附页				

目录

一、	系统的功能设计	4
	功能实现	4
	DNS 报文分析	4
	不良网站拦截功能	4
	服务器功能	4
	中继功能	4
	高并发多线程处理功能	4
	DNS ID 转换功能	5
	超时处理功能	5
	命令行用户界面	5
	开发环境	5
二、	模块划分	6
	main	6
	config	6
	dns_server	6
	dns_client	7
	dns_parse	7
	thread_pool	7
	db	7
	dns_id_map	7
	timer	8
	rbtree	8
	cache	8
三、	软件流程图	8
	程序整体处理流程	9
	DNS 报文的接收进行响应的处理流程图	10
	线程池进行多线程调度的处理流程图	11
	计时器模块处理流程图	12
	ID 映射模块处理流程图	12
	红黑树模块的处理流程图	13
	红黑树模块插入操作的处理流程图	14

红黑树模块删除操作的处理流程图.....	15
cache 模块的整体运行流程图.....	17
cache 模块插入处理流程图.....	17
cache 模块查找处理流程图.....	18
四、 测试用例以及运行结果.....	18
命令行界面.....	19
不良网站拦截.....	21
服务器功能.....	22
中继功能	24
DNS ID 转换功能.....	25
超时处理功能	26
五、 调试中遇到并解决的问题	26
六、 心得体会	27

一、 系统的功能设计

功能实现

DNS 报文分析

DNS 即域名系统，是用于进行“域名-IP 地址”转换的一个分布式数据库。其报文结构由报文头部、问题字段、回答字段、权威字段、附加字段这 5 部分组成。其中报文头部固定 12 字节，由 DNS ID、查询/返回标志 QR，返回码 RCODE 等内容组成。其余字段均遵循一定规则进行组织，包含名字、类型、类等字段。

对于域名等名字形式的资源，DNS 报文采取两种表示方法：一，“标签长度-标签内容”，以 0x00 作为结尾标志；二，依据偏移量指向其他字段的压缩表示。

而本程序实现了将 DNS 报文由进行解析的功能，并根据其上述设计规则，组织了对应的数据结构进行存储，以便进行进一步的处理，并实现了字节流报文数据和解析后的 DNS 数据包相互转换的功能。

不良网站拦截功能

程序从本地“域名-IP 地址”映射表中读取表项，获取需要屏蔽的域名内容（即对应 IP 地址为 0.0.0.0 的内容）。在服务器接收到查询报文后，首先在本地数据中进行查询。如果本地查询返回的 IP 地址是全零，则说明该域名是需要被屏蔽的内容，此时程序将返回特殊的数据包，将 DNS 报文头部标志字段里的 RCODE（返回码）置为 3，向询问客户端表示“名字差错”。从而对实现不良网站的拦截。

服务器功能

接收到客户端发来的 DNS 查询报文后，程序首先在本本地“域名-IP 地址”映射表和 Cache 中进行查询，如果请求的域名类型为 A 类请求（即 ipv4 地址查询）且仅查询单个表项，则当域名请求命中后，本程序根据请求报文和对应 IP 地址，直接向客户端返回 DNS 报文内容。

中继功能

当 DNS 查询报文请求的域名类型为其他类型，或者有多个查询字段时，本程序将不再充当服务器，而是将经过处理的请求报文转交给远程服务器，并将从远程服务器返回的回应报文返回给对应的请求客户端。

高并发多线程处理功能

本程序需要考虑允许多个客户端的并发查询，并且允许在第一个查询尚未得到答

案之前就启动处理另外一个客户端查询。

考虑到多进程对性能和资源的占用，以及 `select` 等 I/O 多路复用函数对于 DNS 服务器固定端口的 UDP 连接处理并不方便，本程序采取了多线程的方式实现了并发处理。

从可移植性的角度考虑，本程序主要基于 POSIX 标准库函数 `pthread` 进行多线程编程。同时，为了减小高并发条件下线程频繁创造和销毁的开销，本程序实现了一个线程池，对于并发请求进行预线程化、管理调度和回收资源的处理。

DNS ID 转换功能

由于本程序始终工作在本机的 53 号端口，而且考虑到 DNS 查询请求可能来自多个客户端或主机，因此有可能出现来自不同客户端的 ID 发送碰撞的可能，在程序实现中继功能时，如果不加以处理直接按照原报文请求发送，会造成无法区分远程 DNS 服务器发过来的响应报文具体对应哪个客户端的问题。

因此，在执行中继功能时，程序有必要进行 ID 转换。为了提高查找效率，本程序将目前使用的 ID 作为键，对应转换前的 ID 及其客户端信息作为值存储在红黑树中，用于表示映射关系。每次进行中继发送请求之前，首先尝试将原 ID 插入映射表中，如果原 ID 未被使用，则可以直接以原 ID 作为新的 ID 使用，其红黑树的值设置为本身的 ID 及客户端信息；否则，将随机生成一个新的 ID，在红黑树中反复查询新 ID 是否已经被占用，直至找到一个空闲的 ID 资源，并将其原 ID 和客户端信息保存至红黑树中。而当对方响应到达，或者对应超时，则将在 ID 映射表中删除对应表项，回收 ID 资源。

超时处理功能

为了更好的在多线程中进行超时处理，本程序实现了一个独立线程的定时器，定时器单独设置一个线程，进行定时功能，当每轮定时时间结束后，计时器将执行一个回调函数，而当多轮定时结束或达到最大重设次数后，计时器则会执行一个清理函数并退出。

而在本程序中，超时处理主要是针对中继发往远程服务器重传，当每轮超时结束后，超时重传计时器将会回调重传函数再次发送数据包，而达到最大重传次数后，则会执行清理函数，回收 ID 资源，并且回收内存资源。

命令行用户界面

为了给用户提供更多交互选项和调试信息的提供，本程序还基于 linux 的标准库函数 `getopt` 函数（在 Windows 系统下通过移植库函数实现），实现了一个命令行用户界面，能够根据用户的需求，修改调试等级、打印相关调试信息等功能。

开发环境

语言标准：C11

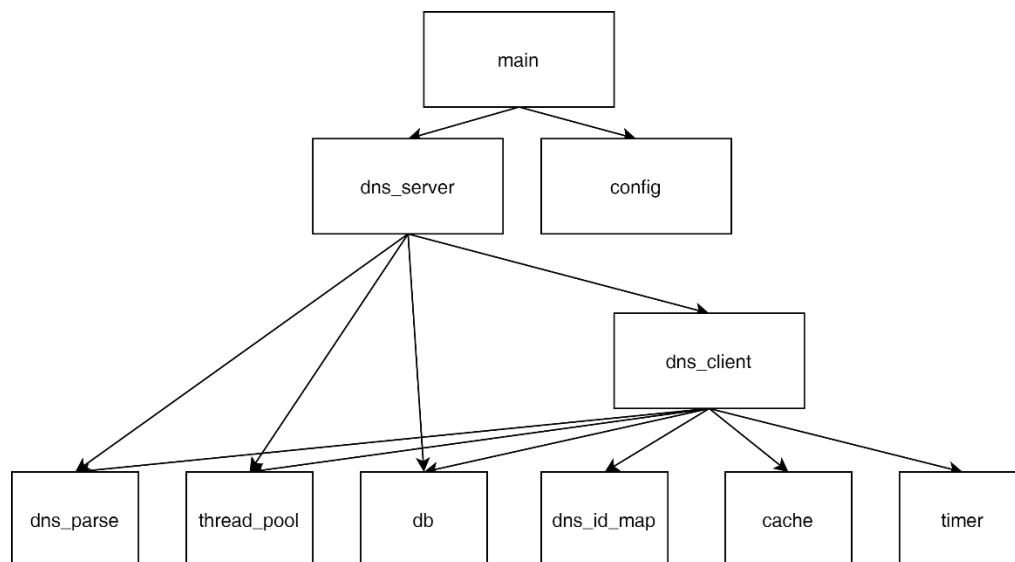
集成开发环境：Visual Studio 2017、Visual Studio 2019

操作系统：Windows10

二、 模块划分

为了更好地区分模块与模块间的功能，本程序将整个程序划分为了多个模块，分别为：初始化设置模块 `main`、`config`，程序主体逻辑功能的实现模块 `dns_server`、`dns_client`，以及为解决子问题而建立的子模块 `dns_parse`、`dns_id_map`、`db`、`timer`、`thread_pool`、`rbtree`、`cache`。

而模块与模块间的具体调用关系如下图所示：



main

程序的入口，主要进行对 DNS 服务器的启动和命令行设置的调用。

config

负责对命令行界面进行交互的处理。初始化远程服务器地址、本地“域名-IP 地址”映射表、调试信息等级。

dns_server

本程序的服务器端。首先初始化线程池、从程序中读入“域名-IP 地址”映射表、初始化 socket，然后开辟一个新线程，主线程阻塞直至接收到用户的关闭命令。

新线程进行服务器监听的循环，非阻塞式地持续监听本机的 53 号端口。如果接收到数据包，则首先判断 DNS 数据包是响应请求还是查询请求。如果是响应请求，则关闭对应的计时器，同时开辟新线程进行响应中继处理；如果是查询请求，则开辟新线程进行对查询的具体处理，而本线程总是持续监听端口，具体处理交由 `dns_client` 模块，从而实现高并发操作。

dns_client

负责具体处理各种客户端事件的模块。

本模块负责对查询请求进行处理。如果请求能够在“域名-IP 地址”映射表或者 cache 中找到，则生成对应的 DNS 响应，并发给对应的请求客户端；如果请求不能够在本地找到或者请求的类型是本地无法处理的类型，则程序首先将进行 ID 的转换，然后启动对应的超时处理计时器，发往远程服务器发送。如果响应超时，则可能是网络状况不良，本程序将进行重传。如果重传后仍旧无法收到回应，则考虑到可能是远程服务器无法实现功能，此时放弃对请求进行处理，回收 ID 映射表的对应资源。

本模块也负责对响应请求中继进行处理。首先停止对应的重传计时，并通过 ID 映射表得到原本的 ID 和客户端信息，删除 ID 映射表的对应表项以回收 ID 资源。再将中继的结果插入到 LRU Cache 中。最后将响应包返回给原本的请求客户端。

dns_parse

负责进行对 DNS 报文进行解析的模块。

本模块主要提供对 DNS 报文进行格式的解析，根据 DNS 报文的组织结构和规则，完成原始 DNS 报文中字节流向 DNS 报文头部、DNS 名字表示、DNS 询问字段、DNS 资源记录等等字段的双向转换。

thread_pool

负责提供一个管理调度线程资源的线程池。

根据需要的任务数量，进行预线程化，同时根据当前工作繁忙状态，管理调度进行处理的线程数量，并且在线程结束后，回收对应的线程资源。

db

负责管理本地“域名-IP 地址”映射表。

通过红黑树来存储对应的“域名-IP 地址”表项，从而提高资源的查询效率，提供查询、插入等访问数据的接口，同时考虑到多线程时存在共享资源线程不安全的问题，本模块通过互斥锁机制也保证了查询、插入、删除等操作的原子性。

dns_id_map

负责管理 ID 映射表。

通过红黑树来存储 DNS 中继请求的 ID 与其原 ID 及客户端信息。

在发送给远程服务器之前，程序将试图将查询请求的 ID 插入到 ID 映射表中。如果 ID 映射表中不存在对应表项，说明该 ID 资源还没有被占用，则程序将会保持其原 ID，并记录其相关信息；如果 ID 映射表中已经存在对应表项，说明 ID 资源已经被占用，那么 ID 映射表将通过伪随机方法生成新的 ID 并返回，以保证 ID 不冲突。

同样考虑在多线程状态下对共享资源的处理，本模块也通过互斥锁保证了对 ID 映射表的插入、查询等操作的原子性。

timer

负责实现多线程计时器。

每个计时器单独开辟一个线程进行处理，每一轮都会进行定时检查，一旦发现定时超时，则将调用回调函数。若计时重启次数达到外部程序设定的上限，则会退出计时器，并调用对应的清理函数。

rbtree

负责提供红黑树建树、增加节点、删除节点、释放节点、销毁树基本操作。

包含了多种红黑树基本操作辅助函数，例如：改变节点颜色，寻找叔叔节点，寻找祖父节点，左旋右旋操作，以及黑树建树、增加节点、删除节点、释放节点、销毁树基本操作。供程序调用。

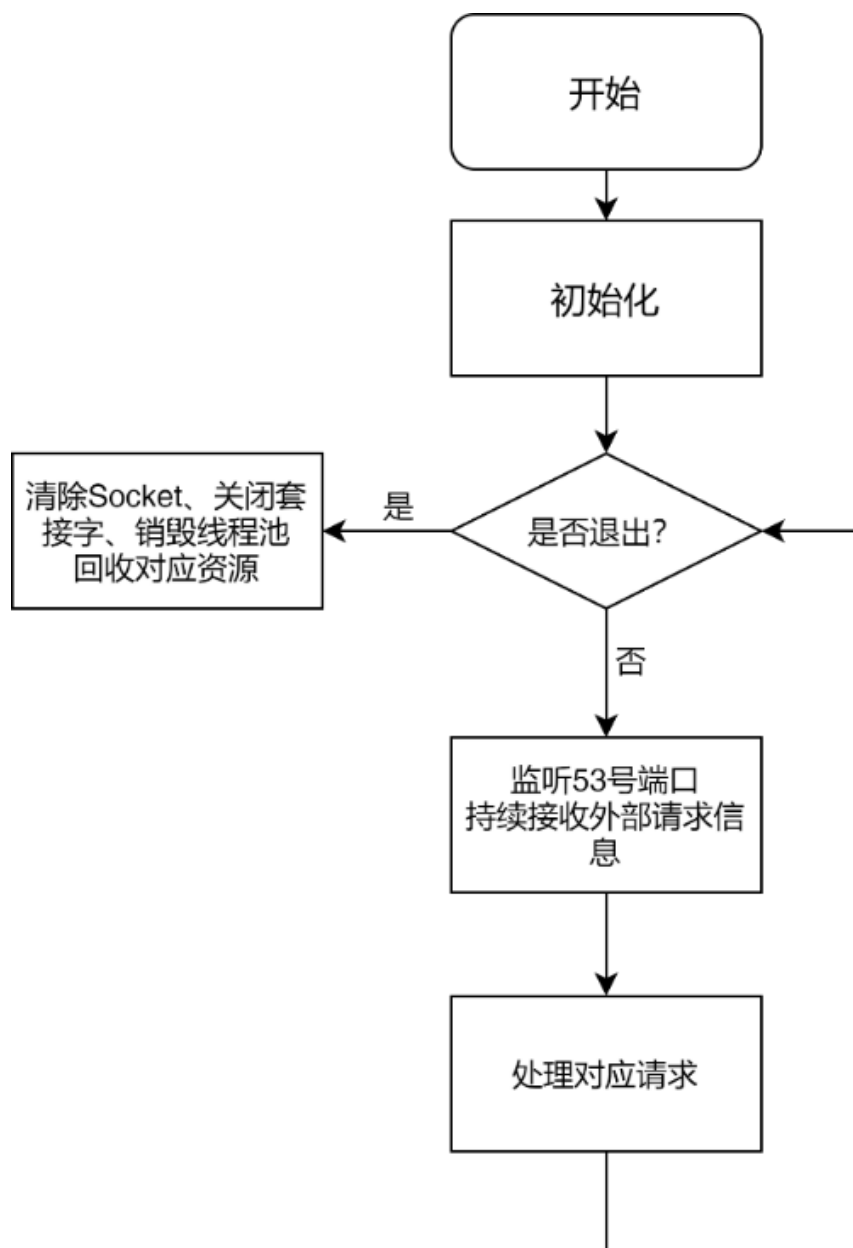
cache

负责“域名-IP 地址”映射的记忆化存储功能。

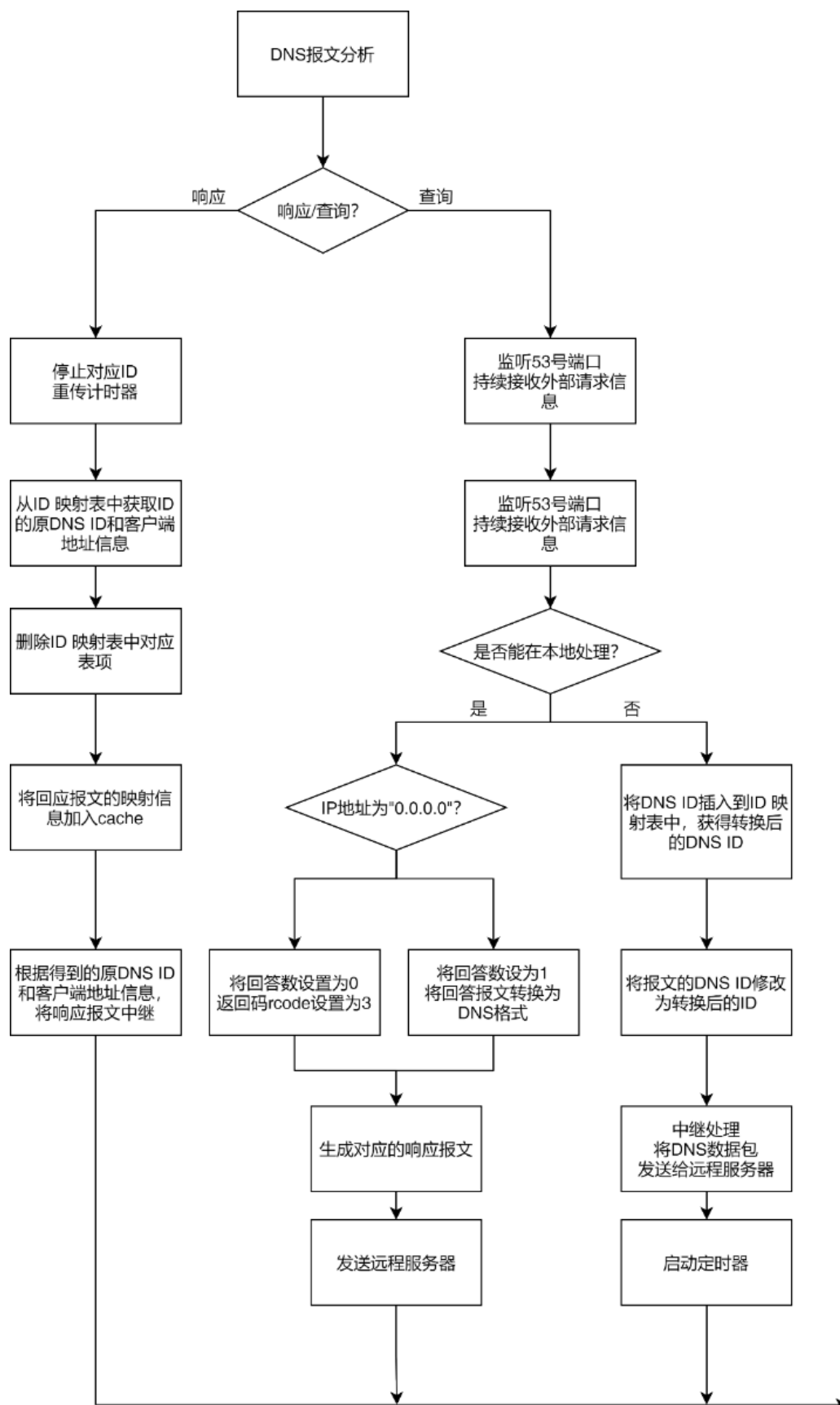
考虑到 DNS 的一个基本特性是使用超高速缓存，当一个名字服务器收到有关映射的信息（主机名字到 IP 地址）时，它会将该信息存放在高速缓存中。这样若以后遇到相同的映射请求，就能直接使用缓存中的结果而无需通过其他服务器查询。它维护一个固定大小的缓存，随当前信息与查询实时更新。

三、 软件流程图

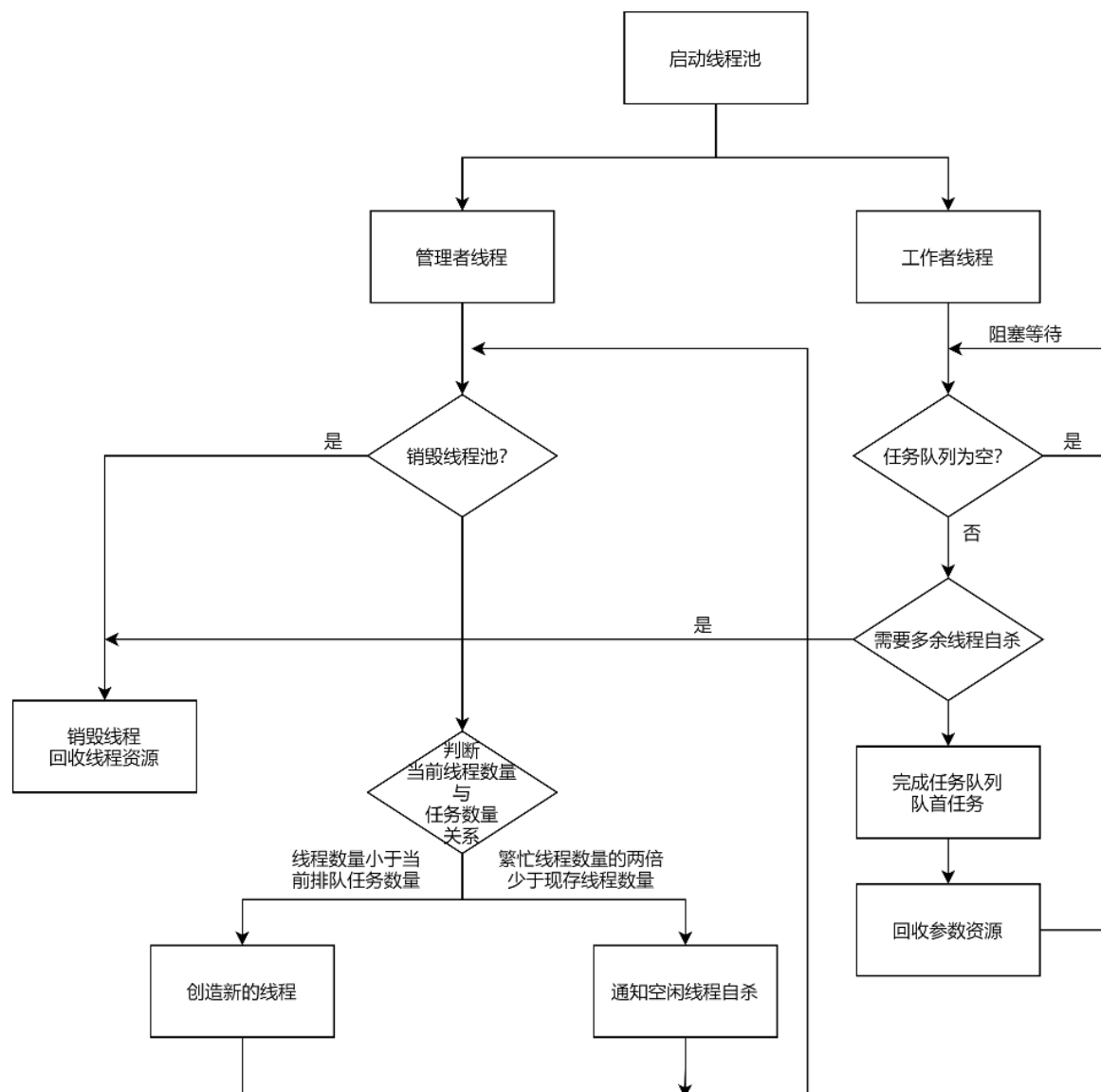
程序整体处理流程



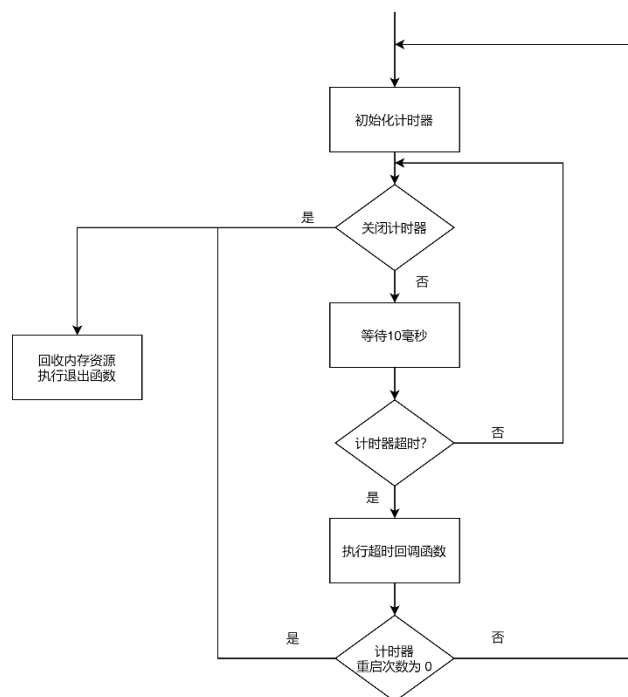
DNS 报文的接收进行响应的处理流程图



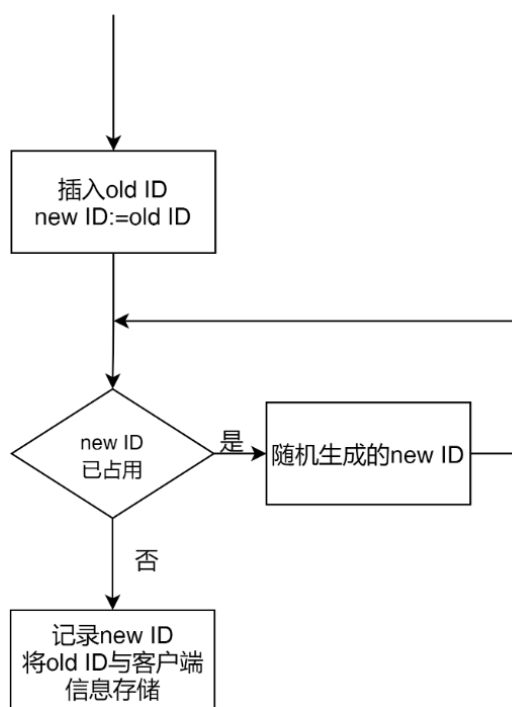
线程池进行多线程调度的处理流程图



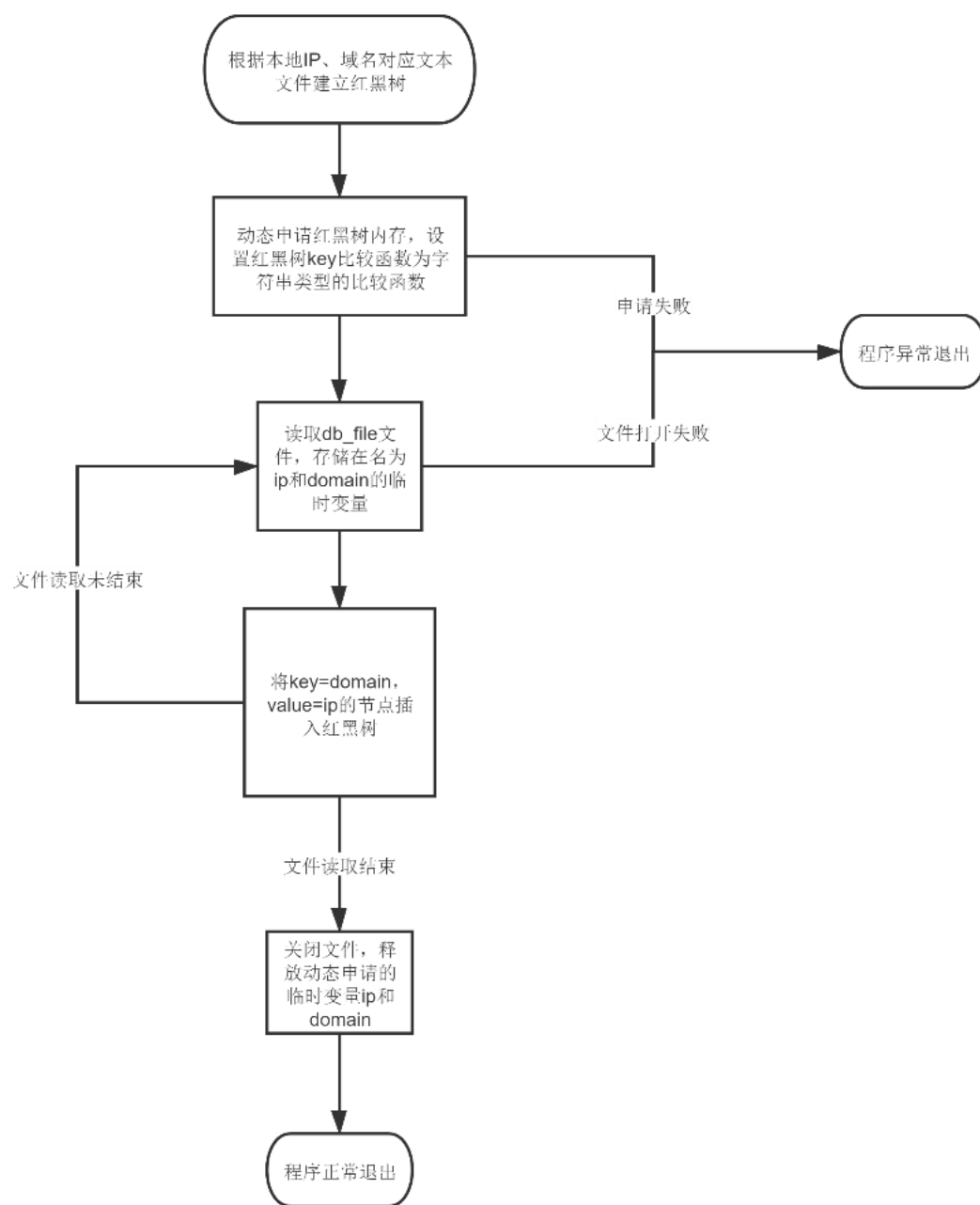
计时器模块处理流程图



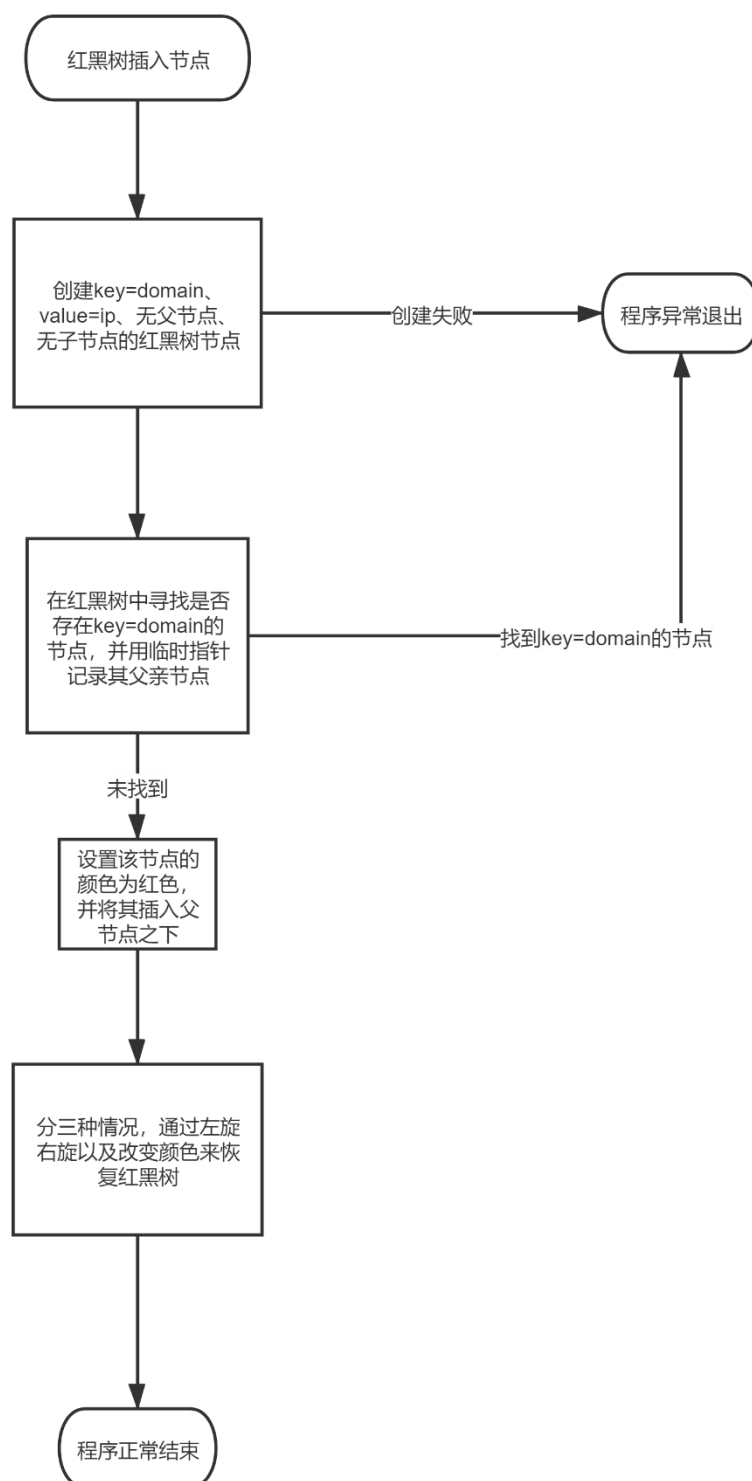
ID 映射模块处理流程图



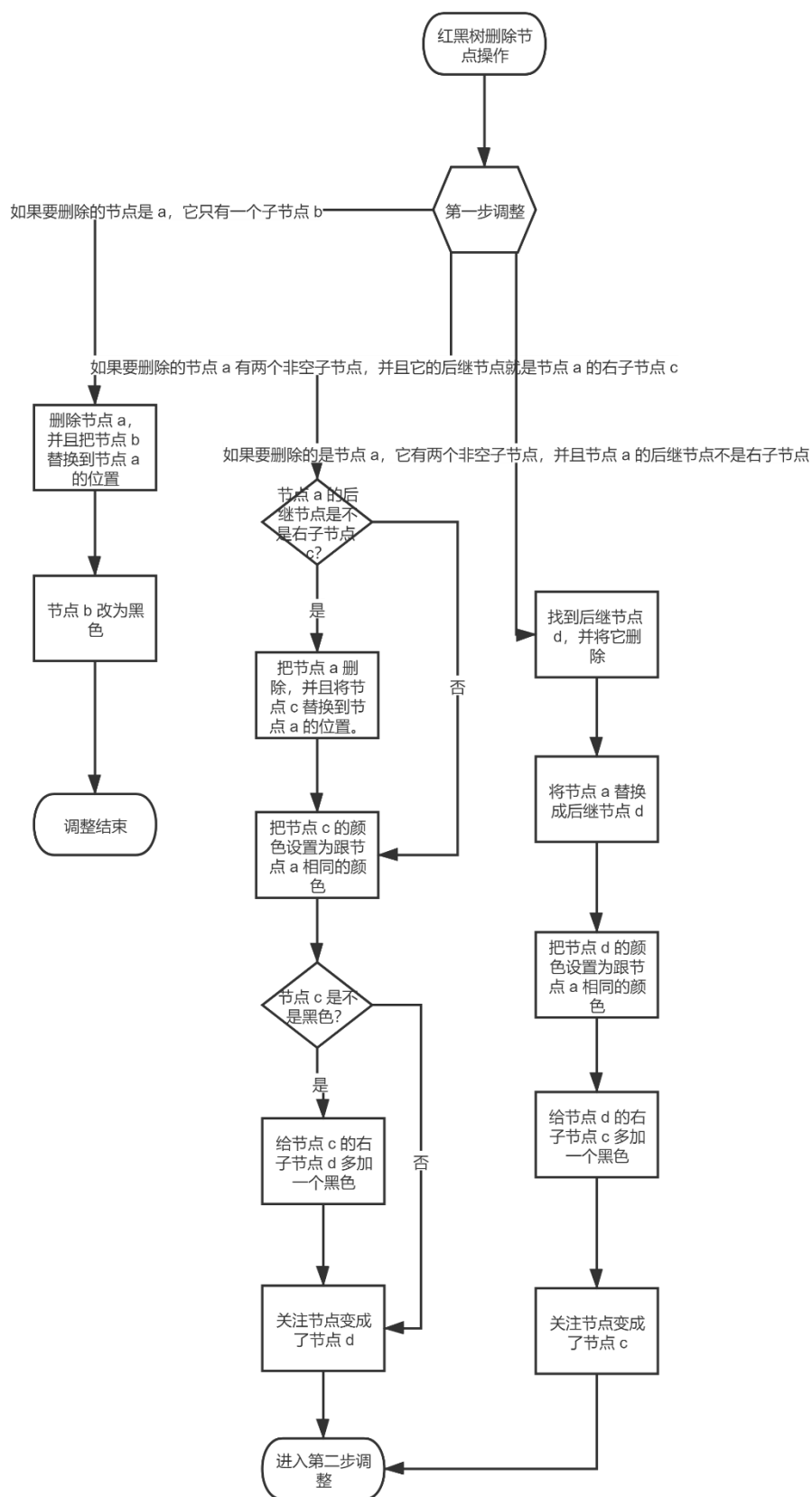
红黑树模块的处理流程图

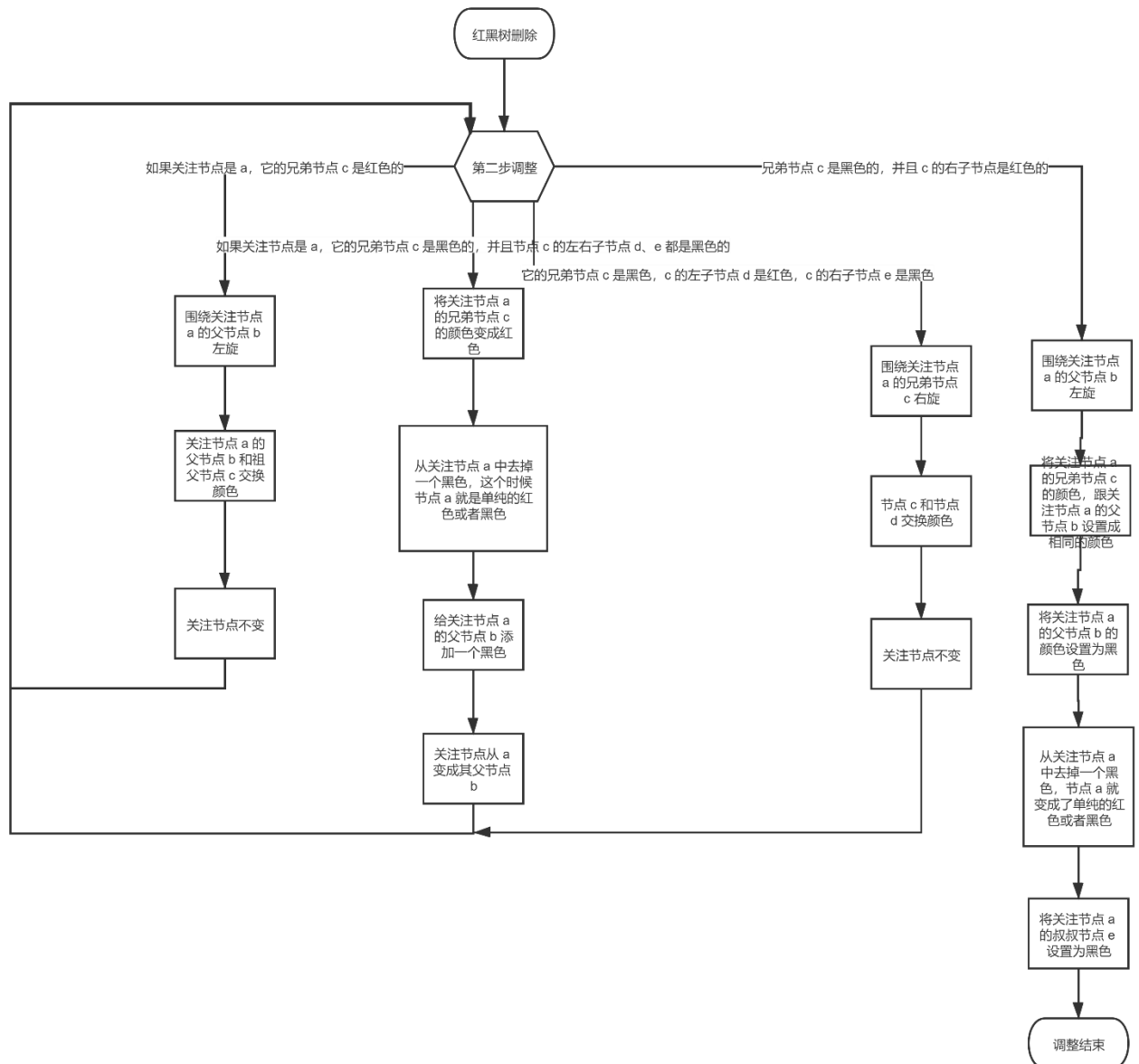


红黑树模块插入操作的处理流程图

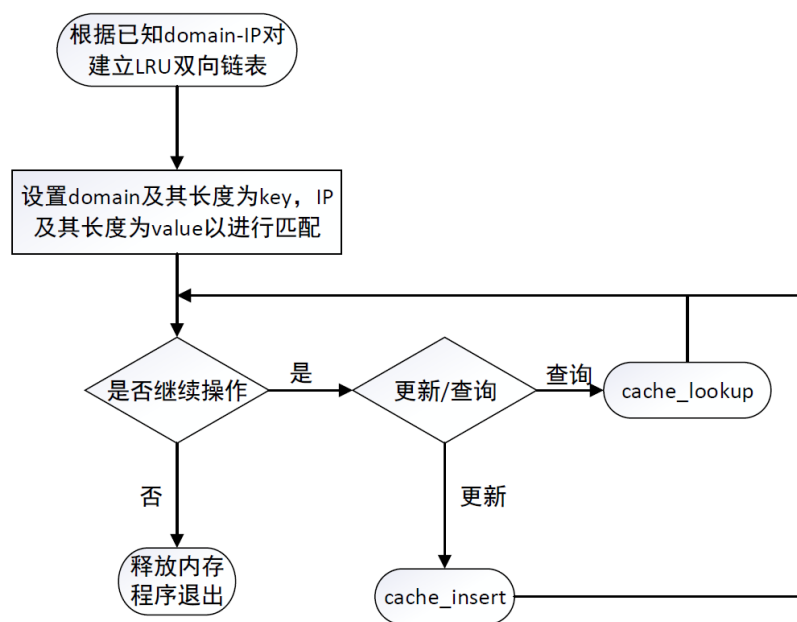


红黑树模块删除操作的处理流程图

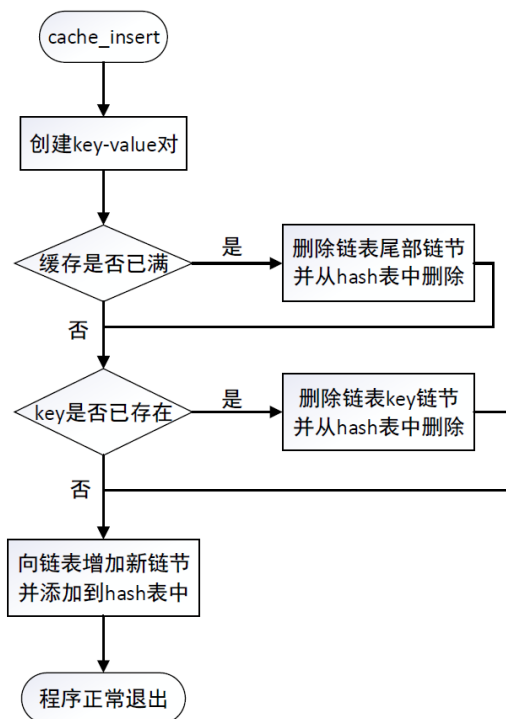




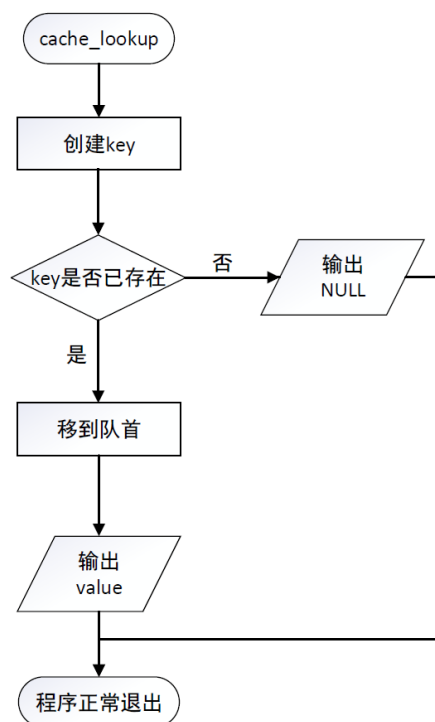
cache 模块的整体运行流程图



cache 模块插入处理流程图



cache 模块查找处理流程图



四、 测试用例以及运行结果

经过测试，本程序成功实现了不良网站拦截、服务器功能、中继功能，并且能够进行超时重传等处理机制，由于使用了多线程机制，本程序在多客户端访问、多主机访问等并发访问的情况下都有较好的表现。

首先，我们可以测试单主机情况下的运行情况。将本机的 DNS 服务器地址改为回环地址 “127.0.0.1”



命令行界面

本程序实现了一个命令行交互界面，用于让用户设置远程服务器 IP 地址、“域名-IP 地址”映射表以及 debug 等级。

在命令行选项中输入“-?”或者“--help”，则会跳出本程序命令行交互的使用方法。

```
C:\Windows\System32\cmd.exe
Microsoft Windows [版本 10.0.19042.985]
(c) Microsoft Corporation. 保留所有权利。

D:\computer_network\DNSRELAY\Debug>.dnsrelay -?
DNSRELAY, Build: Mon May 31 00:12:48 2021

Usage:
  .dnsrelay <options>

Options :
  -?  --help : print this
  -d  --debug=<0-2>: debug level set, default 0
  -l  --log=<filename> : using assigned file as db file
  -a  --address=<remote dns ip address> : set remote dns server ip address

i.e.
  .dnsrelay -d2 -a 8.8.8.8 -l ./dnsrelay.txt
  .dnsrelay --debug=2 --address=8.8.8.8 --log=./dnsrelay.txt

D:\computer_network\DNSRELAY\Debug>.
```

在命令行选项中输入“-l”或者“--log=”，随后输入本地“域名-IP 地址”映射表的位置，则能够选择对应的文件作为本地的映射表。

```
C:\Windows\System32\cmd.exe - .\dnrelay -l .\dnrelay.txt
Microsoft Windows [版本 10.0.19042.985]
(c) Microsoft Corporation. 保留所有权利。

D:\computer_network\DNSRELAY\Debug>.dnrelay -l .\dnrelay.txt
DNSRELAY, Build: Mon May 31 00:18:58 2021

Remote dns: 10.3.9.44
Data file: .\dnrelay.txt
Debug level: 0
If you want to terminate the program, just type ESC.

[db]db load successfully.
```

在命令行选项中输入“-a”或者“--address=”，随后输入远程服务器的 IP 地址，则能够选择对应的远程服务器作为中转请求的远程服务器。

```
C:\Windows\System32\cmd.exe - .\dnrelay -a 10.3.9.44
Microsoft Windows [版本 10.0.19042.985]
(c) Microsoft Corporation. 保留所有权利。

D:\computer_network\DNSRELAY\Debug>.dnrelay -a 10.3.9.44
DNSRELAY, Build: Mon May 31 00:20:57 2021

Remote dns: 10.3.9.44
Data file: .\dnrelay.txt
Debug level: 0
If you want to terminate the program, just type ESC.

[db]load db error
```

在命令行选项中输入“-d”或者“--debug=”，随后紧跟数字 0-2，则能够选择对应的 debug 等级。在不同的 debug 等级下，程序将会在命令行界面上打印不同的调试信息，对于 debug 等级为 0，则几乎不会打印程序运行过程中的各种详细信息。而对于 debug 等级为 1，程序将会输出每次处理的信息；debug 等级为 2，程序将会输出每次处理的具体 DNS 报文解析后的结果。

```
C:\Windows\System32\cmd.exe - .\dnrelay -d0
D:\computer_network\DNSRELAY\Debug>.dnrelay -d0
DNSRELAY, Build: Mon May 31 00:24:45 2021

Remote dns: 10.3.9.44
Data file: .\dnrelay.txt
Debug level: 0
If you want to terminate the program, just type ESC.

[db]db load successfully.
```

debug 等级为 0，运行过程中不显示任何具体调试信息

```
C:\Windows\System32\cmd.exe - \dnsmasq -d1
D:\computer_network\DNSRELAY\Debug>. \dnsmasq -d1
DNSRELAY, Build: Mon May 31 00:28:03 2021

Remote dns: 10.3.9.44
Data file: .\dnsmasq.txt
Debug level: 1
If you want to terminate the program, just type ESC.

[db]load db...
[db]db load successfully.
[server]server created.

[query] received a packet id = 0x2b12
[query] id map[id 2b12 -> 2b12]
[query] id 0x2b12 query have to send to remote dns.
[response]get response packet id 2b12(client id 2b12)
[query] id map[id 2b12 delete id 2b12 successfully.
[client]id 2b12 timer stop.
[query] received a packet id = 0x1263
[query] id map[id 1263 -> 1263]
[query] id 0x1263 query have to send to remote dns.
[response]get response packet id 1263(client id 1263)
[query] id map[id 1263 delete id 1263 successfully.
[client]id 1263 timer stop.
[query] received a packet id = 0x7c7e
[query] id map[id 7c7e -> 7c7e]
[query] id 0x7c7e query have to send to remote dns.
[response]get response packet id 7c7e(client id 7c7e)
[query] id map[id 7c7e delete id 7c7e successfully.
```

debug 等级为 1，会显示具体的处理信息

```
C:\Windows\System32\cmd.exe - \dnsmasq -d2
Remote dns: 10.3.9.44
Data file: .\dnsmasq.txt
Debug level: 2
If you want to terminate the program, just type ESC.

[db]load db...
[db]db load successfully.
[server]server created.

[query] received a packet id = 0x6ffa
[query] detailed query packet info id = 0x6ffa
[query] received a packet id = 0x0220
[query] detailed query packet info id = 0x0220

[DNS message]
[Raw data]:
6f fa 01 00 00 01 00 00
00 00 00 00 03 68 6b 6a
07 73 70 65 65 64 65 72
02 70 77 00 00 01 00 01
[DNS Header]
Transaction ID: 6ffa
Flags: query
ra=0, rd=1, tc=0, rcode=0
qd_cnt=1, an_cnt=0, ns_cnt=0, ar_cnt=0
[DNS Queries]
query:hkj.speeder.pw.
[DNS answer]
```

debug 等级为 2，会显示出处理的每条 DNS 数据包报文内容

不良网站拦截

如果需要拦截某些网站，可以在“域名-IP 地址”映射表中添加表项“0.0.0.0 域名”这一字段，从而实现对某些不良网站的拦截。在本次测试中，我们首先在映射表中加入 www.baidu.com 作为待屏蔽的不良网站，首先输入“ipconfig/flushdns”，清除当前本地的 DNS 缓存，随后再打开本程序，使用 nslookup 进行域名查看，则此时显示“UnKnown 找不到 www.bupt.edu.cn: Non-existent domain”，而对于不被屏蔽的网站，则返回正常结果。

```
C:\Windows\System32\cmd.exe - nslookup

D:\computer_network\DNSRELAY\Debug>nslookup
默认服务器: Unknown
Address: 127.0.0.1

> www.baidu.com
服务器: Unknown
Address: 127.0.0.1

*** Unknown 找不到 www.baidu.com: Non-existent domain
> www.bupt.edu.cn
服务器: Unknown
Address: 127.0.0.1

非权威应答:
名称: vn46.bupt.edu.cn
Addresses: 2001:da8:215:4038::161
           10.3.9.161
Aliases: www.bupt.edu.cn
>
```

此时登录浏览器，同样发现域名为 www.baidu.com 的网站由于 DNS 解析的问题，无法正常访问。



而对于正常的网站，本程序则可以正常访问。



服务器功能

结合命令行工具 nslookup 与命令行界面给出的调试信息对检测结果进行确认。

在“域名-IP 地址”映射表中，写入“10.3.9.161 www.bupt.edu.cn”这一行，打开程序，将调试等级调整至 2 级。

```
CA\Windows\System32\cmd.exe - \DNSRELAY -d2
D:\computer_network\DNSRELAY\Debug>. \DNSRELAY -d2
DNSRELAY, Build: Mon May 31 19:28:02 2021

Remote dns: 101.199.128.54
Data file: .\dnsrelay.txt
Debug level: 2
If you want to terminate the program, just type ESC.

[db]load db...
[db]db load successfully.
[server]server created.
```

此时在 nslookup 中输入“www.bupt.edu.cn”进行查询，最终成功在本地中找到对应表项并成功发出响应报文。不过，由于本程序并不支持对于 DNS 问题类型 CNAME 或 AAAA 的解析处理，因而返回的响应报文结果中仅有其 ipv4 的 IP 地址，对于别名或 ipv6 结果，则由本程序中继远程服务器所获知。

```
选择C:\Windows\System32\cmd.exe - \DNSRELAY -d2

[query] received a packet id = 0x0002
[query] detailed query packet info id = 0x0002

[DNS message]
<Raw data>:
00 02 01 00 00 01 00 00
00 00 00 00 03 77 77 77
04 62 75 70 74 03 65 64
75 02 63 6e 00 00 01 00
01
<DNS Header>
Transaction ID: 0002
Flags: query
ra=0, rd=1, tc=0, rcode=0
qd_cnt=1, an_cnt=0, ns_cnt=0, ar_cnt=0
<DNS Queries>
query:www.bupt.edu.cn.
<DNS answer>

[client]id 0x0002 query found in db.
[response]ready to send response packet id = 0x0002

[DNS message]
<Raw data>:
00 02 81 80 00 01 00 01
00 00 00 00 03 77 77 77
04 62 75 70 74 03 65 64
75 02 63 6e 00 00 01 00
01 00 00 01 00 01 00 00
00 78 00 04 0a 03 09 a1
<DNS Header>
Transaction ID: 0002
Flags: response
ra=1, rd=1, tc=0, rcode=0
qd_cnt=1, an_cnt=1, ns_cnt=0, ar_cnt=0
<DNS Queries>
query:www.bupt.edu.cn.
<DNS answer>
query:
addr: 10.3.9.161
```

查询“www.bupt.edu.cn”时，程序在 debug 等级为 2 时的调试结果。

```
命令提示符 - nslookup
C:\Users\PENG\XK>nslookup
默认服务器: UnKnown
Address: 127.0.0.1

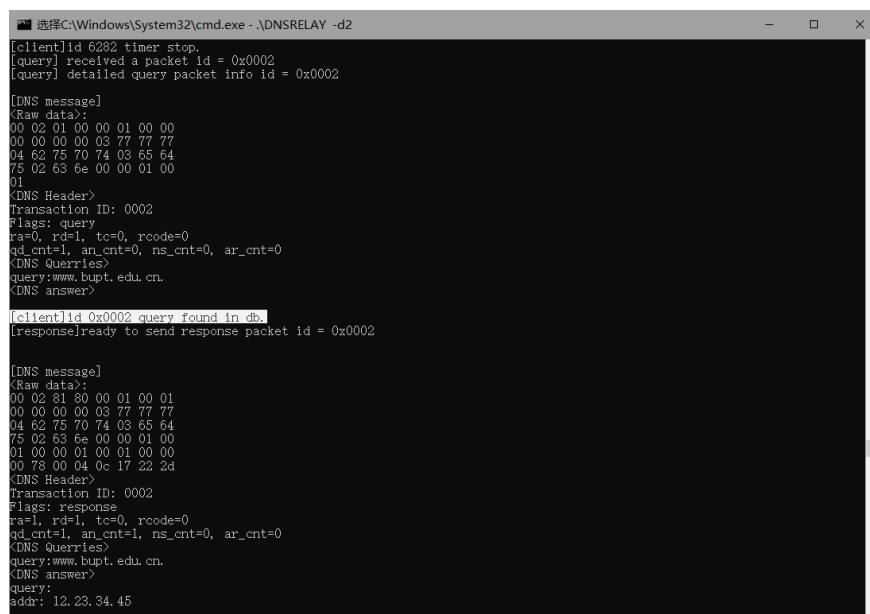
> www.bupt.edu.cn
服务器: UnKnown
Address: 127.0.0.1

非权威应答:
名称: www.bupt.edu.cn
Addresses: 2001:da8:215:4038::161
10.3.9.161
Aliases: www.bupt.edu.cn

>
```

查询“www.bupt.edu.cn”时，nslookup 程序返回的结果

由于本地返回的结果依照“域名-IP 地址”映射表中的表项，故将 IP 地址改为自己设定的任意结果，nslookup 的返回值也应该由本地“域名-IP 地址”映射表所决定。



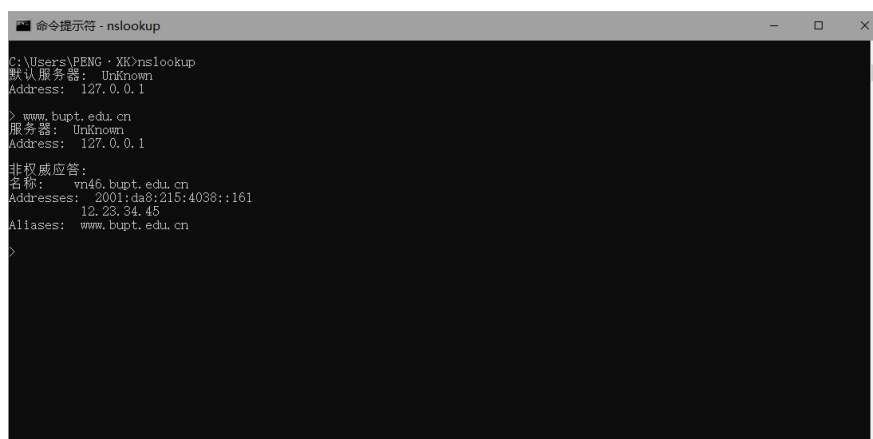
```
选择C:\Windows\System32\cmd.exe - \DNSRELAY -d2
[client]id 6282 timer stop.
[query] received a packet id = 0x0002
[query] detailed query packet info id = 0x0002

[DNS message]
<Raw data>:
00 02 01 00 00 01 00 00
00 00 00 00 03 77 77 77
04 62 75 70 74 03 65 64
75 02 63 6e 00 00 01 00
01
<DNS Header>
Transaction ID: 0002
Flags: query
ra=0, rd=1, tc=0, rcode=0
qd_cnt=1, an_cnt=0, ns_cnt=0, ar_cnt=0
<DNS Queries>
query:www.bupt.edu.cn.
<DNS answer>

[client]id 0x0002 query found in db.
[response]ready to send response packet id = 0x0002

[DNS message]
<Raw data>:
00 02 81 80 00 01 00 01
00 00 00 00 03 77 77 77
04 62 75 70 74 03 65 64
75 02 63 6e 00 00 01 00
01 00 00 01 00 01 00 00
00 78 00 04 0c 17 22 2d
<DNS Header>
Transaction ID: 0002
Flags: response
ra=1, rd=1, tc=0, rcode=0
qd_cnt=1, an_cnt=1, ns_cnt=0, ar_cnt=0
<DNS Queries>
query:www.bupt.edu.cn.
<DNS answer>
query:
addr: 12.23.34.45
```

修改本地表项，查询“www.bupt.edu.cn”时，程序在 debug 等级为 2 时的调试结果。



```
命令提示符 - nslookup
C:\Users\PENG >nslookup
默认服务器: Unknown
Address: 127.0.0.1

> www.bupt.edu.cn
服务器: Unknown
Address: 127.0.0.1

非权威应答:
名称: www.bupt.edu.cn
Address(es): 2001:da8:215:4038::161
12.23.34.45
Aliases: www.bupt.edu.cn

>
```

修改本地表项，查询“www.bupt.edu.cn”时，程序在 debug 等级为 2 时的调试结果。

中继功能

对于不符合“域名-IP 地址”映射表处理要求的 DNS 请求，本程序将起到与远程 DNS 服务器之间的中继作用，我们同样可以结合 nslookup 的返回结果与程序的调试信息对结果进行验证。


```

命令提示符 - nslookup
> www.sina.com
服务器: Unknown
Address: 127.0.0.1

非权威应答:
名称:      spool.grid.sinaedge.com
Addresses: 2409:8c3c:4:f::68:83
          120.192.83.98
Aliases:   www.sina.com
>

```

对于不在“域名-IP 地址”映射表中的域名请求，本程序仍旧能够中继返回正确的结果。

```

选择C:\Windows\System32\cmd.exe - .\dnsrelay -d2
[query] received a packet id = 0x0005
[query] detailed query packet info id = 0x0005

[DNS message]
<Raw data>:
00 05 01 00 00 01 00 00
00 00 00 00 03 77 77 77
04 73 69 6e 61 03 63 6f
6d 00 00 1c 00 01
<DNS Header>
Transaction ID: 0005
Flags: query
ra=0, rd=1, tc=0, rcode=0
qd_cnt=1, an_cnt=0, ns_cnt=0, ar_cnt=0
<DNS Queries>
query:www.sina.com.
<DNS answer>
[dns id map]id 0005 -> 0005
[query]id 0x0005 query have to send to remote dns.
[client]id 0004 timer stop.
[response]get response packet id 0005(client id 0005)
[dns id map]delete id 0005 successfully.

[DNS message]
<Raw data>:
00 05 81 80 00 01 00 02
00 00 00 00 03 77 77 77
04 73 69 6e 61 03 63 6f
6d 00 00 1c 00 01 c0 0c
00 05 00 01 00 00 00 cd
00 16 05 73 70 6f 6f 6c
04 67 72 69 64 08 73 69
6e 61 65 64 67 65 c0 15
c0 2a 00 1c 00 01 00 00
00 7a 00 10 24 09 8c 3c
00 04 00 0f 00 00 00 00
00 68 00 83
<DNS Header>
Transaction ID: 0005
Flags: response
ra=1, rd=1, tc=0, rcode=0
qd_cnt=1, an_cnt=2, ns_cnt=0, ar_cnt=0
<DNS Queries>
query:www.sina.com.
<DNS answer>
query:www.sina.com.
cname
query:spool.grid.sinaedge.com.

```

对于不在“域名-IP 地址”映射表中的域名请求，程序在 debug 等级为 2 时的调试结果，具体显示出了请求 DNS 报文与中继的响应 DNS 报文。

DNS ID 转换功能

由于考虑到多个客户端同时访问本 DNS 服务器，本程序也能处理将同时访问的多个客户端程序相同的 DNS ID 进行映射，使其能够区分。而当远程 DNS 服务器返回，或者是定时器结束，则本程序将删除对应表项，回收 ID 资源。

```

[server]dns collision occur!
[dns id map]id 3bb8 -> c04b
[query]id 0xc04b query have to send to remote dns.

```

DNS ID 冲突时，本程序采取随机算法生成一个新的 DNS ID，保证发往远程服务器的 DNS 请求的 DNS

ID 不同，使其可以被区分。

```
[response]get response packet id c04b(client id 3bb8)
[dns id map]delete id c04b successfully.
[client]id c04b timer stop.
```

对应上述 DNS 冲突，对应的远程服务器返回时，程序能够正确找到其原 DNS ID，并删除对应 DNS ID 的表项，以回收 ID 资源。

超时处理功能

为了更好地进行超时处理，本程序实现了一个独立线程的定时器，定时器单独设置一个线程，进行定时功能，当每轮定时时间结束后，计时器将进行一次超时重传，超时重传仍旧没有回应后，计时器将会执行清理程序，回收对应的 ID 资源并放弃处理。

[illegible]

通过抓包工具 Wireshark, 我们能够找到定时超时重发的报文

五、调试中遇到并解决的问题

在程序的开发过程中，我们遭遇了不少的问题，其中有的问题是出现在程序设计逻辑上的，有的问题是出现在对网络编程的处理上的，还有的问题则是处于编程习惯不良导致的疏忽问题，这些问题具体表现为以下几个问题：

● 底层数据结构及工具实现

由于本实验要求使用 C 语言进行编程实现，而且除了标准库和 Socket 库之外不允许调用其他函数库，所以很多原本可以通过 C++ STL 或者其他工具库实现的函数和数据结构，我们都必须自行使用 C 语言进行实现。这给我们的编程带来了一定的困难，尤其是对于红黑树、LRU Cache 等较复杂的数据结构，我们需要花费一定的时间去理解其原理，再选择合适的物理结构进行实现。

例如，在红黑树的实现过程中，我们原本对于红黑树的销毁操作采用的是递归实现，然而在调试的过程中，我们发现递归实现对于较多表项而言会导致调用栈次数过多等问题，最后我们又将其通过非递归的方式进行实现，最终使得程序能够正常运行。

● 多线程开发问题

为了提高并发性，本程序采取了多线程的方案来处理较多，而且考虑到移植性和

跨平台的特性，相较 windows 自带的多线程模块，我们选择了 POSIX 标准多线程库 pthread 来进行多线程开发。

在进行多线程开发的过程中，我们建立了一个管理和调度线程的线程池以提高多线程的性能。然而，在进行多线程编程的过程中，我们也遭遇了一系列问题，比如对于共享资源的访问问题，如果多线程同时操作共享资源往往会导致线程不安全，我们就不得不通过互斥锁或者信号量对共享资源进行保护，而不谨慎地使用互斥锁很可能导致性能下降或者死锁。

如何保证多线程的线程安全，并保持线程间的线程同步，给我们的编程带来了很大的挑战。所幸，最后经过不断调试和尝试，最终能够较为稳定的运行。

- 其他问题

由于编程习惯、动态分配内存管理、疏忽等等原因，在编程过程中不免会出现一大堆让人困惑又难以定位的问题，例如文件读写没有安全检测，导致内存泄露，或者是动态分配内存时分配内存片的大小有误等等。

六、 心得体会

在本次计算机网络课程设计的实践过程中，我们小组成员相互合作、互相帮助，采用多线程和网络编程的方法，在仅使用 C 语言的标准库函数和 Socket 函数的前提下，完成了具有中继、不良网站拦截、服务器功能的 DNS 中继服务器。

在本次课程设计的前期工作中，我们主要是通过查阅资料、阅读文档的方式对 DNS 进行了解，由于 RFC1035 文档及一些辅助资料结构清晰、介绍翔实，随着对计算机网络课程的进一步了解和掌握，DNS 的机制和原理也逐渐变得更加清晰。

课程设计真正的难点则在于程序的编程实现过程。由于我们小组对于多线程编程和网络编程相对而言比较陌生，故花费了不少时间成本和精力去进行相关入门的学习和探索，尽管如此，完成程序的过程中仍旧出现了不少困难和未曾预料的 BUG。然而，当我们从对多线程的机制一窍不通，到能够自己实现一个可以运行的线程池。从网络编程时屡屡受挫到最后完成功能。这中间的辛苦和困难固然不计其数，但随之而来的成就感以及积累的经验 and 知识，终究是让人感到收获满满的。

总而言之，在付出了大量的时间和精力之后，我能够切身地感受到本次计算机网络课程设计为我们日后进一步理解计算机网络知识和网络编程、并发编程的技术奠定了坚实的基础，而其中收获的经验教训，也必将在日后的学习生涯中使我们裨益良多。