

基于基址重定位的快速域名压缩算法^①



闫夏莉¹, 王 骞¹, 吕万波¹, 张海阔¹, 岳巧丽¹, 曹 爽²

¹(中国互联网络信息中心, 北京 100190)

²(国家税务总局 电子政务管理中心, 北京 100053)

通讯作者: 吕万波, E-mail: lvwanbo@cnnic.cn

摘 要: 为了提高 DNS 服务器的性能, 提出了基于 M/M/c 排队系统的分析模型. 利用该模型分析了 DNS 服务器的响应时间百分比, 确定了其性能瓶颈在于域名压缩速率. 目前传统压缩算法由于规则的限制只能在查询应答时进行实时的域名压缩, 在高访问量场景存在性能问题. 为了提高压缩速率, 从 DNS 数据特征方面对域名压缩的原理进行了剖析, 并在此基础上结合重定位技术, 提出一种新的域名压缩算法. 新的设计改变了传统的 DNS 数据处理流程, 通过压缩前置, 降低了应答的实时消耗. 实验结果表明, 该算法在压缩比损失很小的条件下提升了系统资源利用率, 达到了优化响应时间百分比的目的.

关键词: 响应时间百分比; 排队模型; 域名压缩; 基址重定位; 相对压缩

引用格式: 闫夏莉, 王骞, 吕万波, 张海阔, 岳巧丽, 曹爽. 基于基址重定位的快速域名压缩算法. 计算机系统应用, 2020, 29(1): 151-157. <http://www.c-s-a.org.cn/1003-3254/7245.html>

Fast Domain Name Compression Algorithm Based on Base Relocation

YAN Xia-Li¹, WANG Qian¹, LYU Wan-Bo¹, ZHANG Hai-Kuo¹, YUE Qiao-Li¹, CAO Shuang²

¹(China Internet Network Information Center, Beijing 100190, China)

²(E-Tax Management Center, State Taxation Administration, Beijing 100053, China)

Abstract: In order to improve the performance of the DNS server, a mathematical model based on M/M/c queue theory was proposed. The probability distribution function of response time was analyzed according to this model, which identifies domain name compression rate as the performance bottleneck. Due to the rule of traditional domain name compression algorithm, DNS servers can only perform real-time domain name compression when the query is answered, which causes a performance problem in the high-traffic scenario. To improve the domain name compression rate, the principle of domain name compression was analyzed from the aspects of DNS data characteristics. Based on this, combing with base relocation technology, a new domain name compression algorithm was proposed. The new design changes the traditional DNS data process, which reduces the real-time consumption during response by pre-compressing. Experimental results show that the algorithm improves the system resource utilization under the condition of small compression loss and achieves the goal of optimizing the percentile response time.

Key words: percentile response time; queue model; name compression; base relocation; relative compression

DNS (Domain Name System) 主要用于承载域名与 IP 地址之间的转换, 是互联网的关键基础设施之一. 权威域名服务器的性能一直是 DNS 的研究重点之一.

在面向 DNS 服务器性能的参数中, 平均响应时间是重要评价指标, 也是用户感受服务器性能的主要体现. 近年来性能测试中又引入了响应时间百分比, 即响应时

① 基金项目: 国家自然科学基金 (61303242)

Foundation item: National Natural Science Foundation of China (61303242)

收稿时间: 2019-06-11; 修改时间: 2019-07-12; 采用时间: 2019-07-22; csa 在线出版时间: 2019-12-27

间小于期望时间的概率百分比来进一步度量和表示其性能, 以期为用户带来更好的体验^[1,2], 这也是本文研究的重要性能参数. 真实的响应时间包括网络传输时间、服务器处理时间等, 用户的网络状况千差万别, 因此本文研究的重点是提升服务器处理时间的百分比.

本文提到的 DNS 服务器特指权威域名服务器, 其查询请求可分为两类. 一类是针对特定资源记录 RR (Resource Record) 的查询, 如 A 记录、NS 记录查询等, 查询结果返回相应的资源记录, 在本文中用“常规查询”表示. 另一类为 IXFR^[3]/AXFR^[4] (Incremental Zone Transfer/Authoritative Zone Transfer) 查询, 用于主从服务器间的数据同步, 查询结果返回区域数据中变化的资源记录甚至完整的区域数据, 该过程称为增量/全量区域数据传送. 以根服务器为例, 全球共 13 台根服务器及其镜像服务器支撑根区数据解析服务^[5]. 各服务器及其镜像通过全量区域数据传送来保障根区数据的一致性^[6].

为了提高响应时间百分比, 有必要先确定其性能瓶颈, 进行针对性的优化. 在本文中, 为了避免盲目的优化, 首先建立了基于排队论^[7]的数据模型, 将 DNS 服务器抽象为一个 M/M/c 的随机服务系统, 并依据此模型对响应时间百分比进行了分析, 确定性能瓶颈之后, 对 DNS 的数据特征进行了分析, 并结合基址重定位技术提出改进算法. 最后对提出的改进算法进行实验和性能评测.

1 DNS 服务器的数学模型

1.1 模型描述

排队现象由两方面构成, 一方请求服务, 另一方提供服务. 顾客通过排队服务系统要依次经过如下过程: 顾客到达、排队等待、接受服务和离去. DNS 服务器的查询应答过程符合排队服务系统的规律. 服务器收到来自各客户端的查询, 请求按一定的速率到达, 经服务器解析返回应答包. DNS 服务器可以是单个服务台, 也可通过 SO_REUSEPORT 机制, 将多个套接字绑定在同一个端口实现多个服务台.

假设 DNS 服务器为 M/M/c 随机服务系统, 模型如图 1 所示. 该系统具有以下性质: (1) 查询请求为单个到达, 到达的时间间隔符合参数 λ 的泊松分布; (2) 每个请求所需的服务时间独立, 服从 μ 的负指数分布 (忽略常规查询与 IXFR/AXFR 查询的应答差异); (3) 系统

有 $c(c \geq 1)$ 个服务台, 服务的顺序按照先来先服务 FcFs (First come First served) 规则; (4) 系统容量为 $N(N > c)$, 缓冲队列长度为 $N-c$, 请求源无限; (5) 如果请求到来时队列已经被占满, 则出现丢包, 否则进入队列等候.

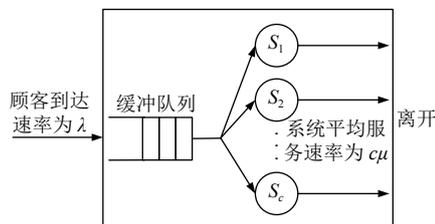


图 1 M/M/c 随机服务系统模型

表 1 显示了建模过程用到的数学符号及相关说明. 其中系统负荷强度为 $\rho = \frac{\lambda}{c\mu}$. 本文只考虑稳定平衡状态 ($\rho < 1$) 的情况.

表 1 DNS 服务器数学建模符号说明

符号	说明
λ	请求到达的速率
μ	服务台服务速率 (解析速率)
c	服务台数量
N	系统容量
ρ	系统负荷强度
P_k	缓冲队列长度为 k 时的概率
W_q	请求平均等待时间
W_s	请求平均逗留时间 (查询响应时间)

1.2 性能瓶颈分析

服务器对查询请求的响应时间 W_s 为请求等待时间 W_q 和解析时间 $\frac{1}{\mu}$ 之和. 响应时间百分比 $\gamma\%$ 可表示为 $P\{W_s < \hat{T} \leq \gamma\% \}$, 其中 \hat{T} 为用户期望时间. 下面将分别针对单服务台模型和多服务台模型分析响应时间百分比的分布概率.

1) $c = 1$ 时, 为 M/M/1 排队模型.

根据参考文献^[8], 有:

$$\int_0^{\hat{T}} \mu(1-\rho)e^{-\mu(1-\rho)t} dt = 1 - e^{-\mu(1-\rho)\hat{T}} \geq \gamma\% \quad (1)$$

或者

$$\mu \geq \frac{-\ln(1-\gamma\%)}{\hat{T}} + \lambda \quad (2)$$

对于 DNS 服务器, 如果不通过流量控制等策略进行人工干预, 请求到达速率 λ 为不可控因素. 为了分析的方便, 在此假设 λ 不变, 用户期望时间 \hat{T} 也不变. 根据

式(2)可知,响应时间百分比 $\gamma\%$ 随着解析速率 μ 的增加而增加。

2) $c > 1$ 时,为 M/M/c 排队模型。

查询请求到达时,如果缓冲队列已满,新的请求无法响应,服务器出现丢包,此时的概率称为损失概率。由参考文献[8]可知,损失概率与缓冲队列长度有关,增加缓冲队列的长度可降低损失概率。为了分析的方便,本文重点讨论没有请求损失的场景。因此,可假设缓冲队列的长度无限大,推导出响应时间的分布函数^[7]为:

$$W_s(t) = \begin{cases} 1 - e^{-\mu t} \left[1 + \mu t \cdot \frac{P_c}{1-\rho} \right], \rho = 1 - \frac{1}{c} \\ 1 - e^{-\mu t} - \frac{P_c}{c(1-\rho)^2 - (1-\rho)} \left[e^{-\mu t} - e^{-\mu t c(1-\rho)} \right], \rho \neq 1 - \frac{1}{c} \end{cases} \quad (3)$$

其中,

$$P_c = \frac{c^c \rho^c}{c!} \cdot \frac{1}{\sum_{k=0}^{c-1} \frac{(c\rho)^k}{k!} + \frac{(c\rho)^c}{c!} \cdot \frac{1}{\rho}} \quad (4)$$

由上述公式可知,在请求到达速率 λ 不变的情况下,响应时间与服务台数量 c 和解析速率 μ 有关。DNS服务器实现多服务台处理查询请求的原理是提高了多核CPU的利用率,多服务台的数量与CPU数量相关。考虑到运行成本,增加多服务台数量的方式不作考虑。在服务台数量固定的情况下,需要通过提高解析速率 μ ,来优化响应时间百分比。

查询请求的解析过程依次为:接收请求、解压缩请求包、查找匹配资源记录、组装应答包、域名压缩和发送应答包。解压缩请求包和组装应答包规则简单,耗时少,在此不做讨论。而数据包接收依赖于网络通信框架,查找匹配资源记录依赖于数据结构和查找算法,改进这两个过程成本较高。因此本文将对域名压缩算法进行针对性优化。

2 DNS 域名压缩现状

2.1 传统压缩算法

为了分析域名压缩,需要对DNS服务器的数据处理流水线进行分析,见图2。为了分析的完整性,流水线中同时考虑了服务器的数据来源和数据出口。DNS服务器通过动态更新或区域数据传送对本区域数据进行更新,收到更新数据后进行解压缩再存储到本地。服务器收到查询请求后在存储数据中查找匹配资源记录,

然后对查找结果进行组装、压缩,最后返回应答包给请求端。对DNS服务器来说,数据更新频率远低于查询请求的频率,即大部分的查询使用同一版本的数据进行应答。

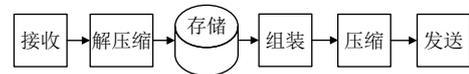


图2 传统 DNS 数据处理流程图

域名压缩通过减少DNS数据中域名的冗余来降低带宽占用。不论是常规查询应答或IXFR/AXFR查询应答,其出口带宽大于入口带宽,尤其是AXFR查询应答,这种差异更加明显。以根区的全量区域数据传送为例,数据包共包含2万多条资源记录。而CN、COM等顶级域的资源记录总数则达到了千万数量级,域名压缩的重要性可见一斑。域名压缩算法直接影响着解析性能。

传统域名压缩算法衍生于LZ77^[9]。该算法基于数据本身包含有重复的字符序列这个特性,使用指针来代替已经出现过的字符序列来达到压缩的目的。域名压缩利用指向数据包中已经出现过的域名的指针来代替整个域名或者部分域名^[10]。该压缩算法的压缩比高,但压缩过程耗时,主要消耗在域名的匹配过程^[11]。在高查询量场景下,在解析过程进行实时的域名压缩非常消耗系统资源。这是制约解析速率的主要原因。

2.2 改进压缩算法

根据DNS数据的特点,域名压缩只能采用无信息损失的无损压缩算法^[12-14]。本文对无损压缩算法进行了充分的调研^[15-21],结果如表2所示,只有部分LZ77系列算法适用于DNS域名压缩。统计编码和字典编码中的LZ78算法不符合DNS标准协议。

近年来LZ77算法的改进方向可大致分为两类:一类是尝试与其他算法结合以获取更好的压缩效果,如与霍夫曼编码结合产生的DEFLATE算法等,这类改进由于结合了统计编码同样不符合DNS协议。另一类则通过优化数据结构和算法等方式提高压缩效率,如LZSS, LZO等,此类算法适用于DNS,但是数据处理流程与传统压缩算法相比并未发生实质性变化,因而同样无法减少资源消耗,提升解析速率。

根据上述分析,为了提高压缩速率,可以考虑将压缩模块前置,将实时压缩转为写时压缩,即在数据存储

时进行压缩处理. 结合 DNS 服务器的特征, 压缩前置可以充分利用读写操作的不对称性, 提高系统资源的利用率. 但是, 现有域名压缩依赖于域名位于数据包中的绝对位置, 对于查询应答, 服务器无法提前预知需要回复的资源记录, 因而现有的压缩算法无法实现压缩模块前置.

表 2 常见无损压缩算法列表

算法分类	算法及改进算法	原理	DNS 适用
统计编码	香农-范诺编码		否
	游程编码	基于字符(串)的出现频率来	否
	算术编码	对字符(串)本身重新编码	否
	霍夫曼编码		否
字典编码	LZSS		是
	LZ77		是
	DEFLA	将字典中出现过的字符(串)利用符号代替	否
	TE		否
	LZ78	HLZ	否
		DTLZW	否

3 基于基址重定位的快速域名压缩算法

3.1 DNS 数据特征分析

为了改进压缩算法, 需要对 DNS 域名压缩的原理进行分析. 压缩是基于数据的冗余度进行的减小数据存储空间的过程^[9]. DNS 根据域名空间倒置的树形结构进行区域的划分. 区域数据以资源记录为最小单位进行存储, 区域内资源记录的所有者(owner)都是其区域顶点(zone apex)的子域^[10], 因此区域内的域名具有冗余性, 这是域名压缩实施的基础. 此外, DNS 的资源记录之间还有其他相关性, 可以进行进一步的域名压缩.

对权威域名服务器来说, 无论是常规查询还是 IXFR/AXFR 查询, 其应答包大部分情况下包含多条相关的资源记录. 这些资源记录可能是查询的特定类型的资源记录, 也可能是帮助请求端进一步获取最终信息的相关记录. AXFR 查询则为特殊情况, 其应答返回了区域内的所有资源记录. 分析查询应答结果后, 发现 DNS 数据具有以下相关性. 图 3 是域名相关性示意图.

(1) I 类: 相同域名, 相同类型的资源记录

DNS 查询应答包中, 域名和类型都相同的资源记录从不单独出现, 称为 RRset^[10] (资源记录集合). AXFR 查询应答时也同样如此. 因而, 域名和类型都相同的资源记录具有最基本的相关性, 称为 I 类相关记录. 该类记录可以进行域名压缩.

(2) II 类: 相同域名, 不同类型的资源记录

IXFR/AXFR 查询应答常常包含域名相同但类型不同的资源记录. 常规查询应答时, A 记录/AAAA 记录也常作为 glue 记录^[10]一同返回. 部署 DNSSEC^[22]后, RRSIG 记录与其相关记录也会同时返回给请求端. 这些相关记录包含相同的域名, 同样可以进行域名压缩, 称为 II 类相关记录.

(3) III 类: NS 记录及其 glue 记录

对大部分查询请求来说, 获取 IP 地址才是其最终目的. 因此, 返回 NS 记录时, A 记录/AAAA 记录常作为附加信息同时返回给请求端. 此时, NS 记录中的授权服务器的域名(NSNAME^[23])与 A 记录/AAAA 记录的所有者是相同的, 同样可以进行域名压缩. 这种相关记录称为 III 类相关记录.



图 3 域名相关性示意图

上述 3 类相关记录中的域名是域名压缩的主要对象. 以根区数据为例, 其资源记录类型多为 NS 记录、DS 记录、A 记录/AAAA 记录. 全量区域数据传送时有 82% 的域名进行了压缩, 其中根据上述 3 类相关性进行压缩的域名占总压缩域名中的 95%(I 类、II 类、III 类分别占 36.4%、31.6%、27%). 除了上述 3 类相关性外, 根区数据还利用了 NSEC 记录中的 next owner^[22]进行域名压缩, 但此类压缩不具备通用性. NSEC 记录在其他区域的数据中并不常见, 通常 DNSSEC 部署更倾向于采用 NSEC3 机制^[22]. 常规查询应答时, 利用 NSEC 记录的 next owner 进行压缩的可能性也很小. 因此该相关性在此不做考虑. 类似 MX、SRV 等类型的资源记录可参考 NS 记录及其 glue 记录的方式实现域名压缩, 在此不再赘述.

3.2 本文改进算法

为了实现压缩模块前置, 需要取消域名压缩时对数据包的依赖, 因此, 本文在新的算法中引入了基址重

定位技术. 基址重定位^[24]是把程序的逻辑地址空间转换成内存中的实际物理地址空间的过程. 重定位表 (Relocation Table) 用于记录重定位时需要修改的地址的位置 (重定位入口的偏移), 以便进行内存地址的修正. 类似地, 在域名压缩时, 可以先进行域名的相对压缩, 再利用重定位表, 修正偏移量, 最终完成传统域名压缩.

结合 DNS 数据特征的分析结果, 本文改进算法只需聚焦于 I 类、II 类、III 类相关记录的域名压缩, 即可保障原始压缩比. 首先, 在数据更新时完成 I 类相关记录中域名的相对压缩, 称为分组压缩. 之后, 在应答时无需查找, 直接利用重定位表实现 I 类相关记录中压缩域名的压缩偏移量修正以及 II 类相关记录中域名

的压缩. 最后实现 III 类相关记录的快速压缩, 称为关联压缩. 运用本文改进算法后, 数据处理流程图见图 4.



图4 本文改进算法的数据处理流程图

本文改进算法的具体步骤如下,

(1) 分组压缩

将区域数据根据 RRset、域名进行分组. 对 RRset 中的域名进行相对压缩, 偏移量以 RRset 首字节为基准, 并为同一域名下的所有 RRset 建立重定位表. 重定位表在系统启动后即可建立, 并在数据更新时进行同步更新. 相对压缩结果如图 5 所示. 重定位表如表 3 所示.

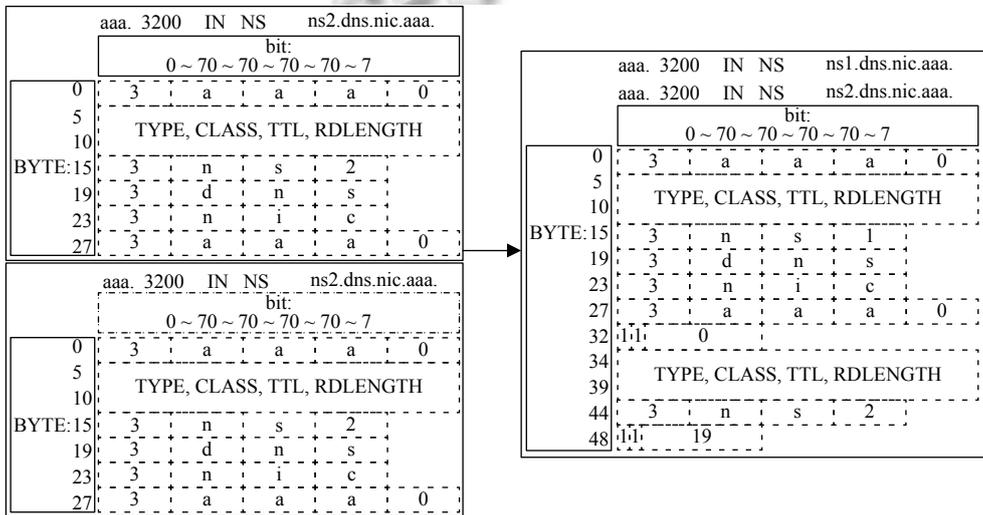


图5 RRset 相对压缩示意图

表3 重定位表

RRset 类型	原始域名	偏移量	当前位置	压缩结果
DS	aaa.	/	0	/
	aaa.	0	39	C000
	aaa.	/	0	/
NS	ns1. dns. nic. aaa.	/	15	/
	aaa.	0	32	C000
	ns2. dns. nic. aaa.	19	44	C013

(2) 利用重定位表修正压缩偏移量

在组装应答包的过程中, 利用重定位表修正相对压缩的偏移量. 同时, 完成 II 类资源记录的域名压缩. 上述过程支持零查找.

(3) 关联压缩

对于 III 类资源记录, 在存储时如果不做特殊处理,

无法提前完成关联压缩. 在不改变解析软件数据结构的前提下, 可以在应答时通过动态字典实现关联压缩. 动态字典只在应答过程建立, 保存已添加的 NS 记录的相关信息. 在添加 NS 记录或 A/AAAA 记录时, 通过搜索动态字典, 即可完成 III 类资源记录的域名压缩. 传统压缩算法会对应答的资源记录中的所有域名, 甚至其父域名建立字典, 进行匹配域名的查找, 字典量大, 搜索速率低. 而用于关联压缩的编码字典只有少量数据, 搜索速率很高. 表 4 和表 5 分别是相同域名资源记录的压缩结果和最终压缩结果.

4 实验结果与分析

为了验证本文改进算法的性能, 本文使用一台

Linux 服务器作为测试平台, CPU 为 Intel Xeon, 2*16 核, 单核主频 2.00 GHz. 测试软件采用 BIND, 上文已经阐述适合于 DNS 的改进算法是 LZ0 与 LZSS, LZ0 与 LZSS 相比算法复杂度更低, 本文选择 LZ0 算法进行对比实验. 分别采用传统算法, 改进算法 (LZ0) 及本文算法实现压缩过程, 并以根区数据为样本进行测试.

表 4 相同域名的资源记录压缩结果

RRset 类型	原始域名	当前位置	压缩结果
DS	aaa.	0	/
	aaa.	39	C000
	aaa.	78	C000
NS	ns1. dns. nic. aaa.	93	/
	aaa.	110	C000
	ns2. dns. nic. aaa.	122	C061

表 5 最终压缩结果

RRset 类型	原始域名	压缩结果
NS	ns1. dns. nic. aaa.	/
	ns2. dns. nic. aaa.	C061
A	ns1. dns. nic. aaa.	C05D
AAAA	ns1. dns. nic. aaa.	C05D

4.1 压缩比

压缩比是衡量域名压缩的重要参数. 本文对常规查询和 AXFR 查询分别做了测试, 分析压缩比变化.

对于常规查询场景, 测试结果如表 6 所示. 表中 (+ED) 代表 DNSSEC 查询. 结果表明, 对于常规查询 3 种算法的压缩比相同.

对于 AXFR 查询, 利用根区数据进行了对比测试. 采用传统算法和改进算法, 根区的全量数据共 79 个数据包, 1341 181 字节; 采用本文算法时, 根区的全量数

据共 79 个数据包, 1350 026 字节. 本文算法在全量区域数据传送场景下, 与另外两种算法相比仅增加了 0.6% 的数据量. 测试结果说明本文算法完全满足 DNS 服务器的压缩比实际需求.

表 6 压缩比对比结果

查询域名	查询类型	查询结果 (字节数)	
		传统/改进算法	本文算法
.	SOA	103	103
.	NS	251	251
.	NS(+ED)	537	537
aaa.	NS	415	415
aaa.	NS(+ED)	870	870
notexist.	A	112	112
notexist.	A(+ED)	1027	1027
cn.	AAAA	360	360
cn.	AAAA(+ED)	695	695

4.2 ASL

域名查找算法对域名压缩性能有重要影响. ASL (Average Search Length)^[25] 是衡量查找速率的主要标准. ASL 指平均查找长度, 其中查找成功的 ASL 指找到已有元素的平均探测次数, 查找失败的 ASL 指找到元素插入位置的平均探测次数. 本文对 4.1 节常规查询的相同场景做了 ASL 对比分析, 结果如表 7 所示 (“/” 代表无压缩, 0 代表不需要查找).

采用传统算法和改进算法时, 其字典保存了应答数据包中的所有域名及其子域名. 采用本文算法时, 其字典中只包含需要关联压缩的域名, 保存的域名数要远小于另两种算法. 由结果可知, 本文改进算法减少了压缩时查找算法的 ASL, 有效提高了压缩速率.

表 7 ASL 对比

查询域名	查询类型	传统算法		改进算法		本文算法	
		查找成功	查找失败	查找成功	查找失败	查找成功	查找失败
.	SOA	/	1	/	1	/	0
.	NS	1	1	1	1	0	/
.	NS(+ED)	1	1	1	1	0	/
aaa.	NS	2.3	2.4	1	1	0.9	/
aaa.	NS(+ED)	2.1	2.8	1	1	0.8	/
notexist.	A	/	1	/	1	/	0
notexist.	A(+ED)	2	1.1	1	1	0	0
cn.	AAAA	1.2	1.1	1	1	0.4	/
cn.	AAAA(+ED)	1.2	1.1	1	1	0.4	/

4.3 响应时间百分比

图 6 显示了各不同查询响应时间的百分比的测试结果对比.

与传统算法和改进算法相比, 本文的算法可以有

效缩短响应时间, 将 90% 以上的耗时控制在 0.5 ms 以下, 而传统与改进算法只能将 90% 以上的耗时控制在 0.65 ms 以下, 只有 20% 的耗时分布在 0.5 ms 以下. 采用本文算法后 DNS 服务器的查询响应性能有明显提

升,达到了预期目标。

5 结论

本文基于排队模型的分析结果,提出了一种基于基址重定位的快速域名压缩算法。新算法充分利用了DNS服务器的数据特征,在不改变原有数据结构的前提下,通过重定位表和动态编码字典实现快速压缩,提高了压缩速率。相比于传统算法和改进算法,本文算法提高了响应时间百分比,可以为用户带来更好的体验。

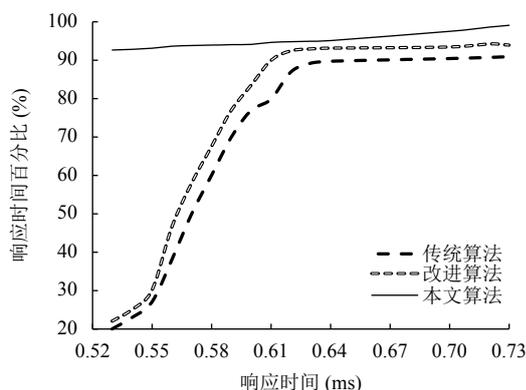


图6 响应时间百分比对比图

参考文献

- Zheng WB, Zhou MC, Wu L, *et al.* Percentile performance estimation of unreliable IaaS clouds and their cost-optimal capacity decision. *IEEE Access*, 2017, 5: 2808–2818. [doi: 10.1109/ACCESS.2017.2666793]
- 何怀文. 保证响应时间百分比下的云中心资源优化配置. *电信科学*, 2015, 31(8): 91–98. [doi: 10.11959/j.issn.1000-0801.2015197]
- Hoenes A. DNS incremental zone transfer protocol (IXFR). IETF. 2012.
- Wang C, Jin J, Han F, *et al.* A mechanism for synchronization across name servers on zone creation. IETF. 2010.
- 延志伟, 耿光刚, 李洪涛, 等. DNS根服务体系的发展研究. *网络与信息安全学报*, 2017, 3(3): 00150.
- 童小海. DNS根区数据的传输与更新机制研究[硕士学位论文]. 北京: 中国科学院大学, 2016.
- 唐应辉, 唐小我. 排队论: 基础与分析技术. 北京: 科学出版社, 2006. 29–39
- 何怀文, 傅瑜, 杨毅红, 等. 基于 $M/M/n/n+r$ 排队模型的云计算中心服务性能分析. *计算机应用*, 2014, 34(7): 1843–1847. [doi: 10.11772/j.issn.1001-9081.2014.07.1843]
- Sayood K. 数据压缩导论. 贾洪峰, 译. 北京: 人民邮电出版社, 2014. 105–109.
- Mockapetris PV. Domain names-implementation and specification. RFC. 1987.
- 刘爱东, 李知宇, 王丰, 等. 一种支持嵌入式标校系统的数据压缩算法. *计算机与数字工程*, 2018, 46(12): 2607–2610. [doi: 10.3969/j.issn.1672-9722.2018.12.044]
- 宋秉玺. 高效无损压缩算法的研究与实现[硕士学位论文]. 西安: 西安电子科技大学, 2014.
- 吕炳赞. 常用无损数据压缩算法的分析及研究. *电子世界*, 2019, (4): 5–6.
- 王波涛, 李昂, 陈月梅, 等. H-PCPIR-V: 基于 Huffman 编码的 PCPIR-V 优化算法. *计算机工程与科学*, 2018, 40(3): 418–430. [doi: 10.3969/j.issn.1007-130X.2018.03.006]
- 李英, 崔艳鹏, 高新波. 一种基于算术编码的文本数据压缩算法. *电子科技大学学报*, 2016, 46(6): 929–933. [doi: 10.3969/j.issn.1001-0548.2016.06.009]
- Gulliver TA, Speidel U, Rebenich N. A note on Lempel-Ziv parser tails and substring lengths. *Proceedings of 2018 International Symposium on Information Theory and Its Applications*. Singapore. 2018. 550–554. [doi: 10.23919/ISITA.2018.8664395]
- Köpl D, Sadakane K. Lempel-Ziv Computation in Compressed Space (LZ-CICS). *Proceedings of 2016 Data Compression Conference*. Snowbird, UT, USA. 2016. 3–12. [doi: 10.1109/DCC.2016.38]
- Tucker G, Oursler R, Stern J. ISA-L Igzip: Improvements to a fast deflate. *Proceedings of 2017 Data Compression Conference*. Snowbird, UT, USA. 2017. 465. [doi: 10.1109/DCC.2017.88]
- Sharma K, Gupta K. Lossless data compression techniques and their performance. *Proceedings of 2017 International Conference on Computing, Communication and Automation*. Greater Noida, India. 2017. 256–261. [doi: 10.1109/CCAA.2017.8229810]
- Kutwin M, Plandowski W, Zaroda A. Generalized word equations: A new approach to data compression. *Proceedings of the 2019 Data Compression Conference*. Snowbird, UT, USA. 2019. 585. [doi: 10.1109/DCC.2019.00097]
- Gupta A, Bansal A, Khanduja V. Modern lossless compression techniques: Review, comparison and analysis. *Proceedings of the 2017 2nd International Conference on Electrical, Computer and Communication Technologies*. Coimbatore. 2017. 1–8. [doi: 10.1109/ICECCT.2017.8117850]
- Vasant S. DNSSEC-authenticated denial of existence: Understanding zone enumeration[Master's Thesis]. Boston: Boston University, 2014.
- Mockapetris P. Domain names-concepts and facilities. RFC 1034. ISI. 1987. DOI: 10.17487/RFC1034
- 威利. Windows PE 权威指南. 北京: 机械工业出版社, 2011. 189–195.
- Xiong B, Li F, Jiang LL, *et al.* An efficient oriented-Hash table for connection record management in high-speed networks. *Journal of Huazhong University of Science and Technology (Natural Science Edition)*, 2011, 39(2): 19–22, 31.