第35卷 第24期 计 算 机 工 程 2009年12月
Vol.35 No.24 Computer Engineering December 2009
·软件技术与数据库· 文章编号：1000—3428(2009)24—0072—03 文献标识码：A 中图分类号：TP311

SIP信令压缩动态字典方案

马 庆，吕玉琴
(北京邮电大学电子工程学院，北京 100876)

摘 要：在对SIP信令进行动态压缩的过程中，需要使用大量的状态内存空间来存储状态信息，造成状态内存的极大浪费。提出用于SIP信令动态压缩的结构化动态字典设计方案，并结合LZSS算法设计一种基于动态字典的SIP压缩机制，改进SIP信令压缩。实验结果表明，该压缩机制在一定程度上降低了压缩算法的时间复杂度，并实现了状态内存的高效利用。

关键词：信令压缩；动态字典；动态压缩

Dynamic Dictionary Scheme in SIP Signaling Compression

MA Qing, LV Yu-qin
(School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876)

【Abstract】A large amount of state memory is consumed when using dynamic compression in SIP signaling compression, which is a great waste of state memory. This paper proposes an effective solution based on structured dynamic dictionary for SIP signaling dynamic compression, and a SIP compression mechanism based on dynamic-dictionary is designed for improving SIP signaling compression combined with LZSS algorithm. Experimental result shows that the new mechanism reduces the time complexity of compression algorithm to some extent and makes effective use of state memory.

【Key words】signaling compression; dynamic dictionary; dynamic compression

1 概述

SIP(Session Initial Protocol)[1]是IETF提出的会话控制协议，3GPP R5提出将其作为IMS中的多媒体会话控制协议，用于会话过程的建立和保持。SDP(Session Description Protocol)用于多媒体信息流的描述。然而，SIP和SDP都是基于文本的协议，消息冗长，若直接应用于无线网络，不但浪费宝贵的带宽资源，而且会造成极大的会话建立时延。

为了对无线网络中传输的文本信令进行压缩，IETF提出了信令压缩(Signaling Compression, SigComp)技术[2]。扩展操作[3]与SigComp配合，包括动态压缩和共享压缩，使用SigComp提出的UDVM及反馈机制，能够显著提高信令压缩效果。

在动态压缩过程中，需要将已发送的消息保存为状态信息。对于状态信息的简单存储会消耗大量状态内存，不但存储效率低下，而且不利于压缩过程中对状态信息的使用。为此提出一种动态字典结构，并结合LZSS算法对SIP信令进行动态压缩，提高了状态内存的利用率，并在一定程度上降低了算法的时间复杂度。

2 信令压缩

信令压缩是对文本信令进行压缩的技术体系，位于应用层与传输层之间，将应用层信令压缩后送入传输层，支持多种协议，如TCP, UDP和SCTP。

2.1 SIP/SDP静态字典

在SIP/SDP压缩过程中，只有经过开始少量的几条消息交互后，压缩率才能达到最高。为了尽可能减少会话建立时延，IETF提出了静态字典[4]，它可以作为状态信息，对开始的几条SIP消息进行压缩，在对第一条SIP消息进行压缩时效果最明显。

2.2 扩展操作

扩展操作的提出，是为了实现SigComp压缩体制。与基于每条消息的压缩方式相比，它可以显著提高信令的压缩效率。扩展操作包括动态压缩和共享压缩，本文只考虑前者。

动态压缩的实现需要使用之前发送过的消息，因此，压缩器需要知道这些消息是否被对端成功接收。对于可靠的传输协议，这一点可以保证。对于不可靠的传输协议，SigComp提供显式确认机制来实现反馈功能。

3 文本压缩算法

文本压缩算法，可以分为基于字典、基于变换和基于模型3类。通过对SIP消息的分析，可以发现基于字典的压缩算法最为适用。当用于SIP信令压缩过程时，在众多的文本压缩算法中，综合考虑压缩速度和压缩率，Deflate算法的压缩效果最好[5]。

Deflate算法是LZ77算法的扩展，它先对文本使用LZ77算法压缩，然后对输出的标记(包括匹配距离、长度和原始字符)进行动态Huffman编码。

LZ77算法将一个可以跟随压缩进程滑动的窗口作为术语字典，若被压缩的字符串在窗口中出现，则输出一个三元符号组(距离，长度，下一字符)。

LZSS算法使用自适应字典模型，将已编码信息作为字典。若需要编码的字符串已出现过，则输出匹配距离和长度，否则输出新字符串。LZSS算法压缩率高，编译码算法较简单，因此，本文采用该算法作为动态压缩方案的算法设计思想。

基金项目：国家自然科学基金资助项目(60772111)
作者简介：马 庆(1986－)，男，硕士研究生，主研方向：多媒体通信系统；吕玉琴，教授
收稿日期：2009-07-15 **E-mail：**maqing2007@gmail.com

4 SIP 动态字典的设计

4.1 SIP 消息语法

SIP 协议是一个基于文本的协议，使用 UTF-8 字符集。一个 SIP 消息既可以是一个从客户端到服务器端的请求，也可以是一个从服务器端到客户端的应答。即使在字符集上和语法细节上有所不同，请求和应答消息都基于 RFC2822 格式。

下面是一条典型 SIP 消息的内容摘要：

```
1   INVITE sip:bob@biloxi.com SIP/2.0
2   Via: SIP/2.0/UDP pc33.atlanta.com;
    branch=z9hG4bKnashds8
3   To: Bob <bob@biloxi.com>
…
10  Content-Length: 147
11
12  v=0
13  o=UserA 2890844526 2890844526 IN IP4 here.com
…
18  a=rtpmap:0 PCMU/8000
```

消息格式：

起始行　　第 1 行
*消息头　　第 2 行~第 10 行
CRLF　　第 11 行
[消息体]　　第 12~18 行

说明：起始行为请求行或状态行，其中，“ * ”表示该消息头部可包含一个或多个，“ [ ] ”表示该参数为可选项。起始行、每一个消息头部以及空行都必须使用回车换行字符(CRLF)来表示行终结。

4.2 动态字典方案

4.2.1 动态字典的结构设计

LZSS 算法是基于滑动窗口的压缩，如何在滑动窗口中查找最长匹配字符串，直接影响算法效率。算法中空间和时间的消耗集中于匹配串的查找。

在使用动态压缩方案时，需要将前面发送过的消息存储为状态信息。直接保存状态信息会消耗大量状态内存，存储效率低下。

为了解决上述问题，结合 SIP 消息的特点，将状态消息保存在结构化的“ 动态字典 ”中。为了与 SIP 消息的 3 个部分结构相适应，动态字典也设计为 3 层结构，包括起始行字典(SLDict)、消息头字典(HeaderDict)、消息体字典(BodyDict)。

下面是本文中用到的 2 个术语说明：

(1)item

由关键字(keyword)和值(value)2 个字符串类型的字段组成的数据结构。其中，value 为 SIP 消息中某行的内容，keyword 由 value 字段得到。关于动态字典 item 的说明如表 1 所示。

表 1　动态字典 item 的说明

动态字典	值	关键字
SLDict	INVITE sip:bob@biloxi.com SIP/2.0	INVITE
HeaderDict	To: Bob <bob@biloxi.com>	To
BodyDict	m=audio 49172 RTP/AVP 0	m

(2)SIP 消息解析

按行读取 SIP 消息，每行作为一个 item，并将 item 归类到相应的字典类型，SLDict, HeaderDict 或 BodyDict。

动态字典是由“ 词条 ”(item)数组构成的数据结构。对于 item，获得 keyword 的方法如下：

SLDict 选取 value 字符串中第一个空格前的子字符串作为 keyword，表示 SIP 方法，如“ INVITE ”；HeaderDict 选取 value 字符串中第一个冒号之前的子字符串作为 keyword，表示消息头域名称，如“ Via ”, “ To ”, “ From ”等；BodyDict 选取 value 字符串中第一个等号之前的子字符串作为 keyword，表示会话描述字段名称，如“ v ”,“ o ”等，若 value 中不含等号，则 keyword 选为前 5 个字母。

“ 动态字典 ”方案与 SIP 消息的语法特点紧密结合，其核心思想体现在 keyword 上。首先，keyword 为压缩算法提供了字符串匹配操作的索引，可以大幅度提高匹配字符串的查找效率。其次，保存状态信息时，不需要存储整条信息，只需按 keyword 对动态字典进行更新即可，提高了存储效率。

4.2.2 动态字典的初始化与更新

在解压器成功解压第一条 SigComp 消息后，得到最初的状态信息，用来对动态字典进行初始化。在一次 SIP 会话中，第一条消息通常为 REGISTER 消息，因而一般由 REGISTER 消息来完成动态字典的初始化。

通过字典更新的操作，来实现状态信息的保存。在成功解压出一条 SIP 消息后进行更新操作。对解压得到的 SIP 消息进行解析，每得到一条 item 都对动态字典进行更新，按照其 keyword 在动态字典中存在与否，更新操作分为“ 增量 ”更新和“ 替换 ”更新：

(1)“ 增量 ”更新

keyword 不存在，对动态字典进行扩充。若状态内存充足，则将此 item 添加到动态字典中，实现“ 增量 ”更新，否则，返回失败。

(2)“ 替换 ”更新

keyword 已存在，根据 item 所属的字典类型进行相应的“ 替换 ”更新操作。对于不同的字典类型，有相同 keyword 的 item，其 value 在多条 SIP 消息之间有不同的变化特点，故将 item 分为 2 类，并采用不同的更新策略：

1)发生变化的部分可能再次出现在后续消息中，对 item 进行 value 替换。

2)发生变化的部分不会再次出现在后续消息中，如 nonce 随机字符串的值，即使对 item 进行 value 替换，对于后续消息的压缩也无作用，因此不替换。

具体而言，SLDict 中的 item 属于类型 1，HeaderDict 中的 item 属于类型 2，BodyDict 中的 item 属于类型 1。

5 动态字典压缩机制的设计

动态字典压缩机制是指利用前面设计的动态字典，对 SIP 文本消息进行压缩的过程，它主要分为 2 个步骤：

(1)标记字符串压缩；

(2)标记字符串编码。

标记字符串压缩是指利用动态字典将 SIP 消息表示为标记字符串。如图 1 所示，首先对 SIP 消息按行进行解析得到 item，然后对每个 item 在动态字典中进行 keyword 检索，若查找成功，则按 LZSS 算法进行标记，得到由<距离，长度>二元组以及原始字符混合组成的标记字符串，并将其合并到输出 1 中；若查找失败，则将此 item 的 value 直接合并到输出 1 中。此过程循环进行，直到到达消息尾。

标记字符串编码是对标记字符串进行编码的过程。标记字符串中含有大量相同的标记字符，需要进行编码，以提高压缩效果。动态 Huffman 编码实现简单，效率较高，所以选作编码算法。图 1 中的输出 1 作为编码输入。特别地，对第一条 SIP 消息进行 Huffman 编码前要采用静态字典进行压缩。

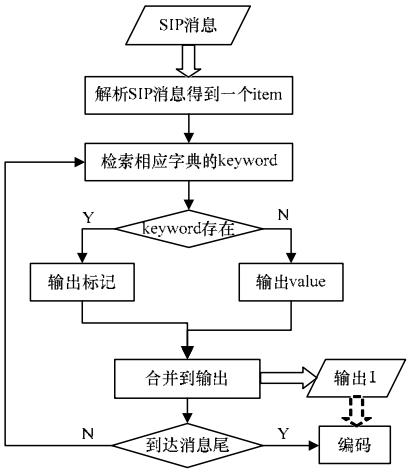

图 1　压缩机制流程

动态 Huffman 编码是一种可变字长编码，它使用动态 Huffman 树，每处理完一个字符都要修改 Huffman 树。编码和解码使用相同的方法修改 Huffman 树，不需要为解码保存 Huffman 树的信息。

6　压缩结果及分析

为了对压缩算法进行性能分析，使用一次 SIP 会话的消息作为输入。另外，为了仅考虑动态压缩，使用同一终端的消息作为状态信息，此处使用从 X-Lite 客户端发出的 10 条 SIP 消息。

从表 2 的结果来看，第 1 条消息的压缩效果最差，这是因为发送第 1 条消息时，动态字典为空，无法为压缩器提供可用于压缩的信息，所以动态压缩的效果从第 2 条消息开始体现，表 2 最后一行是消息序号为 2~10 的统计结果。第 2 条~第 10 条消息的动态压缩效果很明显，平均压缩率低于 30%。

表 2　压缩结果

序号	原始消息/Byte	压缩消息/Byte	压缩率/(%)
1	539	351	65.1
2	694	141	20.3
3	538	138	25.7
4	389	145	37.2
5	699	131	18.7
6	851	378	44.4
7	323	113	35.0
8	1 019	257	25.2
9	591	101	17.1
10	630	192	30.5
总计	5 734	1 596	27.8

如图 2 所示，随着压缩过程的进行，未采用动态字典时，状态内存呈线性增长，采用动态字典后，状态内存增长缓慢。因此，动态字典方案在很大程度上减少了状态内存的消耗。

在算法效率上，动态字典压缩算法用 keyword 的查找代替了 LZSS 算法中最佳匹配位置的查找，降低了算法的实现复杂度和时间复杂度。为了更好地说明动态压缩的效果，图 3 仅为对第 2 条~第 10 条消息的消息头部分进行标记串压缩的结果，动态字典算法与原始 LZSS 算法程序的标记串压缩率均为 35%左右。因此，动态字典算法在保证压缩率的同时，减少了程序的运行时间。

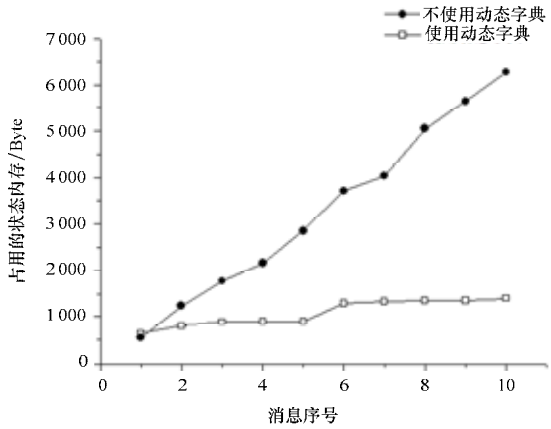

图 2　动态字典对状态内存的影响

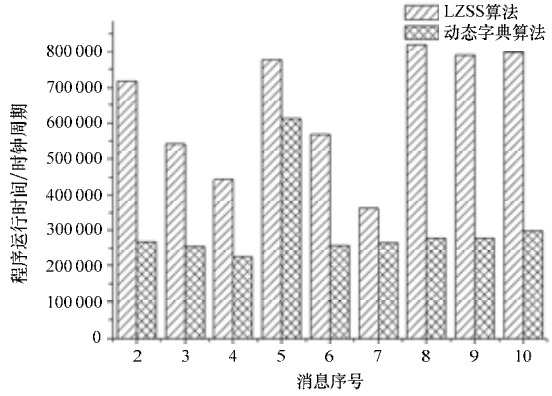

图 3　动态字典对运行时间的影响

7　结束语

本文提出的动态字典压缩方案，与 SIP 消息的特点紧密结合，较好地实现了扩展操作中的动态压缩，降低了压缩算法复杂度并提高了存储效率。

本文中的动态压缩机制在应用中，首先需要与 SigComp 中的反馈应答机制相结合，对状态字典的更新结果进行返回；其次，引入共享压缩机制提高压缩效果；最后，需要实现 UDVM 虚拟机对文中压缩算法的支持，以达到通用解压的目的。

参考文献

[1] Rosenberg J, Schulzrine H, Camarillo G. SIP: Session Initiation Protocol[S]. IETF, RFC 3261, 2002.

[2] Price R, Bormann C, Christoffersson J. Signaling Compression (SigComp)[S]. IETF, RFC 3320, 2003.

[3] Hannu H, Christoffersson J, Vorsgren S. Signaling Compression (SigComp)——Extended Operations[S]. IETF, RFC 3321, 2003.

[4] Garcia-Martin M. The Session Initiation Protocol(SIP) and Session Description Protocol(SDP) Static Dictionary for Signaling Compression(SigComp)[S]. IETF, RFC 3485, 2003.

[5] Jin Haipeng, Mahendran A C. Using SigComp to Compress SIP/SDP Messages[C]//Proc. of IEEE International Conference on Communications. Seoul, Korea: [s. n.], 2005: 3107-3111.

编辑　顾逸斐