



(12)发明专利

(10)授权公告号 CN 104410424 B

(45)授权公告日 2017.06.16

(21)申请号 201410696377.5

(56)对比文件

(22)申请日 2014.11.26

US 5455576 A, 1995.10.03,
 US 7190284 B1, 2007.03.13,
 CN 103138764 A, 2013.06.05,
 CN 103236847 A, 2013.08.07,
 CN 103258030 A, 2013.08.21,
 CN 104125458 A, 2014.10.29,
 CN 103618554 A, 2014.03.05,
 CN 104050269 A, 2014.09.17,

(65)同一申请的已公布的文献号

申请公布号 CN 104410424 A

(43)申请公布日 2015.03.11

审查员 许光华

(73)专利权人 西安电子科技大学

地址 710071 陕西省西安市太白南路2号

(72)发明人 宋彬 李慧玲 秦浩 裴远

(74)专利代理机构 陕西电子工业专利中心

61205

代理人 王品华 黎汉华

(51)Int.Cl.

H03M 7/30(2006.01)

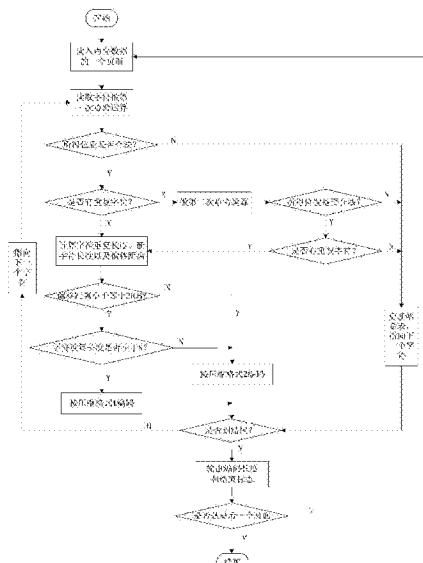
权利要求书2页 说明书4页 附图2页

(54)发明名称

嵌入式设备内存数据的快速无损压缩方法

(57)摘要

本发明公开了一种数据处理技术领域嵌入式设备内存数据的快速无损压缩方法，主要解决已有的压缩方法对内存页面压缩速度低的问题。其主要特点是设计了两种适合内存页面数据的压缩格式：一种是首字节记录字符重复长度、偏移距离和新字符长度，从第二个字节开始依次记录剩余的新字符长度、新字符和剩余的偏移距离；第二种是首字节记录压缩格式标志、偏移距离和新字符长度，从第二个字节开始依次记录剩余的新字符长度、新字符、字符重复长度和剩余的偏移距离。本发明与目前的LZO无损压缩方法相比，提高了内存页面数据的压缩与解压缩速度，并得到更好的压缩率，从而提高了嵌入式设备的内存数据存储容量和利用率，可用于存储受限的嵌入式设备。



1. 一种嵌入式设备内存数据的快速无损压缩方法,包括如下步骤:

(1) 读取嵌入式设备中内存数据的一个内存页面,即按4KB的页面大小逐页读取内存页面;

(2) 判断所读页面的数据是否为新数据,若所读数据未记录在字典中,则判断为新数据,并把新数据位置记入字典中,继续读取内存页面数据,直到未出现新数据为止;

所述字典是根据关键值直接访问的哈希表结构,该关键值是通过哈希函数计算得出;

(3) 对已记录在字典中的所读数据,根据字符重复长度和偏移距离,即字符当前位置与哈希表内记录位置之间的距离,选用不同的压缩格式进行编码:

对于字符重复长度小于8,且偏移距离小于等于2KB的内存页面数据,其首字节记录字符重复长度L、偏移距离D和新字符长度S;从第二个字节开始依次记录剩余的新字符长度M、新字符C和剩余的偏移距离N;

对于不满足字符重复长度小于8,且不满足偏移距离小于等于2KB的内存页面数据,其首字节记录压缩格式标志T、偏移距离D和新字符长度S;从第二个字节开始依次记录剩余的新字符长度M、新字符C、字符重复长度L和剩余的偏移距离N;

(4) 判断编码位置是否为当前读入内存页面结尾,若是,则输出压缩后的数据和数据的长度,并记录结束标志,执行步骤(5),否则,返回步骤(2),继续读入新数据;

(5) 判断当前页面是否为内存数据包的最后一个内存页面,若是,则编码结束,否则,返回步骤(1)读入下一个内存页面。

2. 根据权利要求1所述的嵌入式设备内存数据的快速无损压缩方法,其特征在于:步骤

(3) 所述的首字节记录字符重复长度L、偏移距离D和新字符长度S,按如下规则记录:

首字节的前3位记录字符重复长度L,L<8;

首字节的第4、第5、第6位记录偏移距离D的后3比特,即每一位记录1比特;

判断新字符长度S是否大于3,若不是,则首字节的最后2位记录新字符长度S,即每一位记录1比特,若是,则首字节的最后2位记录为0作为标志,并用新字符长度S减3,得到剩余的新字符长度M。

3. 根据权利要求1所述的嵌入式设备内存数据的快速无损压缩方法,其特征在于:步骤

(3) 所述的从第二个字节开始依次记录剩余的新字符长度M、新字符C和剩余的偏移距离N,按如下规则记录:

判断剩余的新字符长度M是否大于255,若不是,则记录剩余的新字符长度M,若是,则记录一个字节0,并用剩余的新字符长度M减255,直到剩余的新字符长度小于255,记录该剩余的新字符长度,再记录新字符C;

记录新字符完成之后,记录剩余的偏移距离N,该剩余的偏移距离N是偏移距离D的前8比特。

4. 根据权利要求1所述的嵌入式设备内存数据的快速无损压缩方法,其特征在于:步骤

(3) 所述的首字节记录压缩格式标志T、偏移距离D和新字符长度S,按如下规则记录:

首字节的前2位记录为01作为压缩格式标志T,即第1位记录为0,第2位记录为1;

首字节的第3、第4、第5、第6位记录偏移距离D的后4比特,即每一位记录1比特;

判断新字符长度S是否大于3,若不是,则首字节的最后2位记录新字符长度S,即每一位记录1比特,若是,则首字节的最后2位记录为0作为标志,并用新字符长度S减3,得到剩余的

新字符长度M。

5.根据权利要求1所述的嵌入式设备内存数据的快速无损压缩方法，其特征在于：步骤(3)所述的从第二个字节开始依次记录剩余的新字符长度M、新字符C、字符重复长度L和剩余的偏移距离N，按如下规则记录：

判断剩余的新字符长度M是否大于255，若不是，则记录剩余的新字符长度M，若是，则记录一个字节0，并用剩余的新字符长度M减255，直到剩余的新字符长度小于255，记录该剩余的新字符长度，再记录新字符C；

记录新字符完成之后，判断字符重复长度L是否大于255，若不是，则记录字符重复长度L，若是，则记录一个字节0，并用字符重复长度L减255，直到字符重复长度小于255，记录该字符重复长度；

记录字符重复长度完成之后，记录剩余的偏移距离N，该剩余的偏移距离N为偏移距离D的前8比特。

嵌入式设备内存数据的快速无损压缩方法

技术领域

[0001] 本发明属于数据处理技术领域,涉及嵌入式设备内存数据的数据压缩方法,本发明在数据压缩时根据内存数据的特征采用新的数据压缩格式提高了压缩的速度,可用在存储受限的嵌入式设备中。

背景技术

[0002] 内存是计算机的重要部件之一,它是与CPU进行沟通的桥梁。计算机中所有程序的运行都是在内存中进行的。内存的性能对计算机的影响很大,而在体积、存储容量受限的嵌入式便携设备中,内存对设备性能和用户体验的影响尤为突出。近些年来,随着移动互联网的发展,嵌入式便携设备如手机、平板电脑等已经成为人们必备的一种通信工具。因此对内存数据进行压缩,挺高内存存储能力和利用率将大大提高设备的整体性能。随着社会发展,信息量不断增长,人们对嵌入式设备的系统性能也提出了更高的要求,如更高的速度、更低的耗能、更小的体积、能存取更多的信息等等。为了达到上面的各种性能要求,人们提出了各种改进的方法。相较于高额的硬件技术的突破,更加快速有效的方法之一就是无损数据压缩技术。如若在嵌入式设备中运用无损数据压缩技术,则可以在相同的硬件存储空间中存取更多的数据,提高内存利用率,降低成本,挺高设备性能和用户体验。鉴于上述技术的各种优点,运用这种简单而廉价的改进嵌入式系统性能的技术,研究无损数据压缩技术是很有必要的。

[0003] Lempel和Ziv于1977年提出了一种高效率的无失真压缩技术,即LZ77无损数据压缩算法,该压缩算法的主要原理是利用较短的标记代表前面出现过的重复字串,标记格式为重复长度,偏移距离,如abcdeka_{bcd}ha,则可以编码成abcde(5,6)ha表示,这样从整体上而言,较短的信息代替较长的信息,从而达到了压缩的效果。1982年,James Storer和Thomas Szymanski在LZ77基础上将算法进行改进提出了LZSS算法,提高了压缩效率。后来Lempel-Ziv-Oberhumer又在LZSS的基础上将算法进行改进提出了LZO算法,极大地提高了压缩编码速度。LZO算法是一种基于字典的无损的数据压缩算法,具有压缩速度快、即时性的特点。该算法根据不同的重复长度和偏移距离设计了五种数据压缩格式,编码端按照匹配对,即重复长度,偏移距离的大小选择某一种压缩格式编码,解码端通过压缩格式的首字节大小区分这五种不同的格式,最大的偏移距离可以达到48K。该方法存在的不足之处是,自内存“分页机制”提出之始,内存页面的默认大小便被设置为4096字节,即4KB。虽然在原则上计算机的内存页面大小是可配置的,但绝大多数的操作系统在实现中仍然采用默认的4KB页面。为便于内存数据管理,应对内存数据采用逐页压缩的方式,而LZO初始设计目的是压缩长度不定的数据,它在压缩内存数据时,只能取得很低的压缩率,不能有效的提高内存利用率,并且压缩与解压缩速度都很慢。因此对于内存数据,用目前的LZO的压缩方式和压缩格式都不能适用。

发明内容

[0004] 本发明的目的在于克服上述已有技术的不足,提出了一种嵌入式设备内存数据的快速无损压缩方法,以能够更快速的压缩与解压缩内存数据,从而有效提高内存存储能力和利用率。

[0005] 实现本发明的技术方案是:根据内存页面的数据特征,设计一种适合内存页面的压缩格式,针对内存页面数据进行压缩编码,具体步骤包括如下:

[0006] (1) 读取嵌入式设备中内存数据的一个内存页面,即按4KB的页面大小逐页读取内存页面;

[0007] (2) 判断所读页面的数据是否为新数据,若所读数据未记录在字典中,则判断为新数据,并把新数据位置记入字典中,继续读取内存页面数据,直到未出现新数据为止;

[0008] 所述字典是根据关键值直接访问的哈希表结构,该关键值是通过哈希函数计算得出;

[0009] (3) 对已记录在字典中的所读数据,根据字符重复长度和偏移距离,即字符当前位置与哈希表内记录位置之间的距离,选用不同的压缩格式进行编码:

[0010] 对于字符重复长度小于8,且偏移距离小于等于2KB的内存页面数据,其首字节记录字符重复长度L、偏移距离D和新字符长度S;从第二个字节开始依次记录剩余的新字符长度M、新字符C和剩余的偏移距离N;

[0011] 对于不满足字符重复长度小于8,且偏移距离小于等于2KB的内存页面数据,其首字节记录压缩格式标志T、偏移距离D和新字符长度S;从第二个字节开始依次记录剩余的新字符长度M、新字符C、字符重复长度L和剩余的偏移距离N;

[0012] (4) 判断编码位置是否为当前读入内存页面结尾,若是,则输出压缩后的数据和数据的长度,并记录结束标志,执行步骤(5),否则,返回步骤(2),继续读入新数据;

[0013] (5) 判断当前页面是否为内存数据包的最后一个内存页面,若是,则编码结束,否则,返回步骤(1)读入下一个内存页面。

[0014] 本发明由于所采用的压缩格式简单,从而提高了内存页面数据的压缩与解压缩速度,并得到更好的压缩率,能够较大幅提高嵌入式设备的内存数据存储容量和利用率。

[0015] 测试结果表明:本发明与目前的LZO无损压缩方法相比,其压缩时间提高了14.52%,解压缩时间提高了98.84%,压缩率提高了1.1%。

附图说明

[0016] 图1是本发明的压缩流程图;

[0017] 图2是本发明中的压缩格式图。

具体实施方式

[0018] 下面结合图对本发明作进一步详细描述:

[0019] 参照图1,本发明的实现步骤如下:

[0020] 步骤1:从嵌入式设备的内存数据包中读入一个内存页面,即按4KB的页面大小逐页读取内存页面。

[0021] 步骤2:从所读内存页面读入四个字符,做第一次哈希运算,即通过第一个哈希函数计算得出关键值,该哈希函数为目前的LZO无损压缩方法中的第一个哈希函数。

[0022] 步骤3:根据步骤2中关键值判断字符的位置是否合法,若合法,则进入步骤4,若不合法,则更新哈希表,该哈希表是根据关键值直接访问的数据结构,再返回步骤2。

[0023] 所述的合法,是指哈希表中所存的每一个位置都只能根据一个关键值访问。

[0024] 步骤4:判断当前哈希表所存位置中的字符是否与读入字符相同,若相同,则进入步骤7,若不相同,则进入步骤5。

[0025] 所述的当前哈希表所存位置,是指根据步骤2中关键值直接访问的哈希表所存的一个位置。

[0026] 步骤5:用步骤2中得到的关键值做第二次哈希运算,即通过第二个哈希函数计算得出第二个关键值,该哈希函数为目前的LZO无损压缩方法中的第二个哈希函数;再根据第二个关键值判断字符位置是否合法,若合法,则进入步骤6,若不合法,则更新哈希表,返回步骤2。

[0027] 步骤6:判断该哈希表所存地址中的字符是否与读入字符相同,若相同,则进入步骤7,若不相同,则判定读入字符为新字符C,并更新哈希表,返回步骤2。

[0028] 所述的哈希表所存地址,是指根据步骤5中第二个关键值直接访问的哈希表所存的一个地址。

[0029] 步骤7:计算新字符长度S、字符重复长度L和偏移距离D,即字符当前位置与哈希表内记录位置之间的距离。

[0030] 该计算方法与目前的LZO无损压缩方法中的计算方法相同。

[0031] 步骤8:判断字符重复长度是否小于8且偏移距离是否小于等于2KB,若是,则执行步骤9;若不是,则执行步骤10。

[0032] 步骤9:将字符按压缩格式1的记录规则进行编码。

[0033] 参照图2(a),本步骤按照压缩格式1的记录规则进行编码的步骤如下:

[0034] 9.1)首字节的前3位记录字符重复长度L,首字节的第4、第5、第6位记录偏移距离D的后3比特,即每一位记录1比特;

[0035] 9.2)判断新字符长度S是否大于3,若不是,则首字节的最后2位记录新字符长度S,并从第二个字节开始记录新字符C;若是,则首字节的最后2位记录为0作为标志,并用新字符长度S减3,得到剩余的新字符长度M,再判断剩余的新字符长度M是否大于255,若不是,则记录剩余的新字符长度M,若是,则记录一个字节0,并用剩余的新字符长度M减255,直到剩余的新字符长度小于255,记录该剩余的新字符长度,再记录新字符C;

[0036] 9.3)记录新字符完成之后,记录剩余的偏移距离N,该剩余的偏移距离N是偏移距离D的前8比特;然后进入步骤11。

[0037] 步骤10:将字符按压缩格式2的记录规则进行编码。

[0038] 参照图2(b),本步骤按照压缩格式2的记录规则进行编码的步骤如下:

[0039] 10.1)首字节的前2位记录为01作为压缩格式标志T,即第1位记录为0,第2位记录为1,首字节的第3、第4、第5、第6位记录偏移距离D的后4比特,即每一位记录1比特。

[0040] 10.2)判断新字符长度S是否大于3,若不是,则首字节的最后2位记录新字符长度S,即每一位记录1比特,并从第二个字节开始记录新字符;若是,则首字节的最后2位记录为0作为标志,并用新字符长度S减3,得到剩余的新字符长度M,再判断剩余的新字符长度M是否大于255,若不是,则记录剩余的新字符长度M,若是,则记录一个字节0,并用剩余的新字

符长度M减255,直到剩余的新字符长度小于255,记录该剩余的新字符长度,再记录新字符C;

[0041] 10.3) 记录新字符完成之后,判断字符重复长度L是否大于255,若不是,则记录字符重复长度L,若是,则记录一个字节0,并用字符重复长度L减255,直到字符重复长度小于255,记录该字符重复长度;

[0042] 10.4) 记录字符重复长度完成之后,记录剩余的偏移距离N,该剩余的偏移距离N为偏移距离D的前8比特。

[0043] 步骤11:判断编码位置是否与当前读入内存页面结尾位置相同,若相同,则输出编码后的数据和数据的长度,并记录结束标志,然后进入步骤12,若不相同,则返回步骤2。

[0044] 所述的结束标志,是指记录三个字节的数据,即第1个字节记录为17,第2和第3个字节都记录为0。

[0045] 步骤12:判断当前内存页面是否为内存数据包的最后一个内存页面,即嵌入式设备的内存数据包中的所有数据是否都已经被读取,若是,则编码结束,若不是,则返回步骤1。

[0046] 下面结合实验对本发明的效果做进一步说明:

[0047] 本实验采用C语言来编写发明所提出的压缩方法,通过比较本发明与目前的LZO无损压缩方法对内存页面数据的压缩效果,来说明本发明方法压缩与解压缩速度的优点。LZO是目前最好的无损压缩方法。本实验所采用的内存数据为典型移动设备的4KB大小的内存页面数据,实验使用数据为内存页面数据包,数据包大小为453MB。在VS2010程序开发环境中,分别用本发明和目前的LZO无损压缩方法压缩内存页面数据,实验结果如表1所示:

[0048] 表1

VS2010 实验结果			
	压缩时间 (s)	解压缩时间 (s)	压缩率
[0049]	LZO 方法	4.223	2.13
	本发明	3.61	0.025
	性能提高	14.52%	98.84%
			1.1%

[0050] 表1中的时间是整个压缩包所有内存页面的压缩时间与解压缩时间,表格中数据是运行了1000次取平均的结果。由表1可以看出,本发明的压缩率提高了1.1%,同时压缩时间与解压缩时间分别提高了14.52%和98.84%,从而提高了嵌入式内存数据存储容量和利用率。

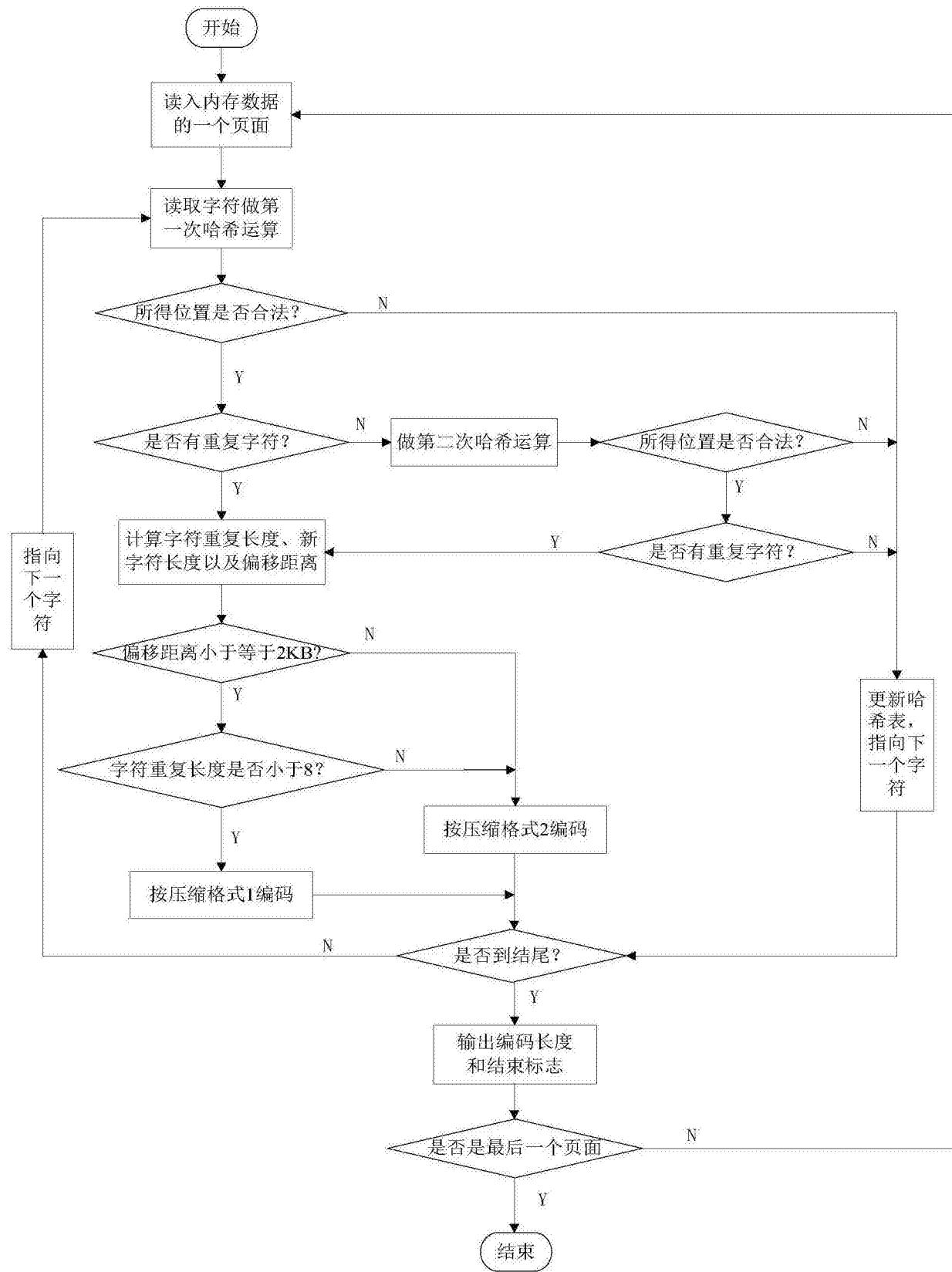
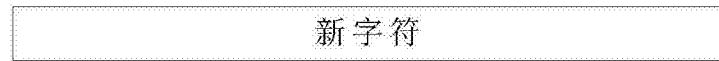
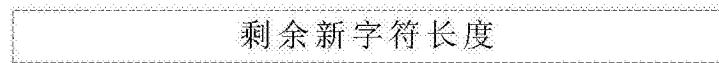


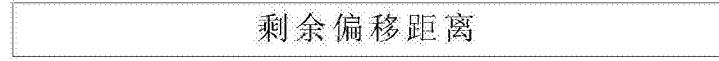
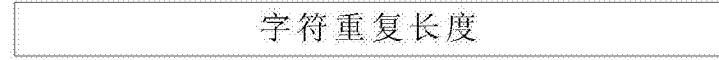
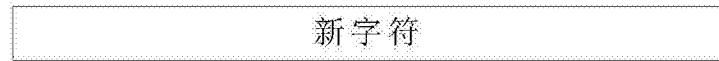
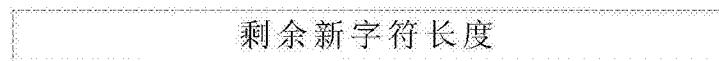
图1

3 比特	3 比特	2 比特
字符重复长度	偏移距离	新字符长度



(a)

2 比特	4 比特	2 比特
0 1	偏移距离	新字符长度



(b)

图2