无线传感器网络流量的冗余抑制机制研究

褚轶昕,宗 平
(南京邮电大学 物联网学院,江苏 南京 210003)

摘 要:冗余抑制是一种通过在接收节点缓存频繁传输的内容以避免多次发送重复数据的网络流量抑制技术。现有的解决方式需要有大量的内存,用于分析当前冗余流量并维持缓存。由于硬件的限制,无线传感器节点在同一时刻不能够提供相应的资源;协议和通信模式的多样性还使得通信中冗余发生的频率及其所占比例难以预测。因此,给出了一种改进的冗余抑制协议,通过概率流量分析以确定和缓存那些造成最多流量储存的冗余传输子集,实现性能表现接近于建立在全面分析和无约束缓存基础上的冗余抑制解决方案。

关键词:无线传感器网络;冗余抑制;数据频繁项
中图分类号:TP393 **文献标识码**:A **文章编号**:1673-629X(2011)07-0067-04

Mechanism of Suppressing Redundancy Protocol in Wireless Sensor Network Traffic

CHU Yi-xin,ZONG Ping
(College of Internet of Things,Nanjing University of Post and Telecommunications,Nanjing 210003,China)

Abstract:Redundancy suppression is a network traffic compression technique by caching recurring transmission contents at receiving nodes, avoids repeatedly sending duplicate data. Existing implementations require abundant memory both to analyze recent traffic for redundancy and to maintain the cache. Wireless sensor nodes at the same time cannot provide such resources due to hardware constraints. The diversity of protocols and traffic patterns in sensor networks furthermore makes the frequencies and proportions of redundancy in traffic unpredictable. Such difficulties made us devise a novel protocol that conducts a probabilistic traffic analysis to identify and cache only the subset of redundant transfers that yields most traffic savings. Verified this approach to perform close enough to a solution built on exhaustive analysis and unconstrained caching to be practicable.

Key words:wireless sensor network;redundancy suppression;frequent items of data stream

0 引 言

人们一直在致力于研究如何避免不必要的数据传输而造成的网络资源浪费。常见的解决方案包括依据数据请求的频率缓存结果[1],或者是在发送前对所传输数据进行整体压缩以达到降低传输数据量的目的[2]。冗余抑制是一种防止相同数据在网络链路上重复传输的方法。其基本思想是在接收节点将某些传入的数据保存在内存中。如果之后需要传输相同的数据,则可以将其从先前的缓存数据中提取出来并在本地重构,而不必再次发送。

近期的研究集中在如何使用多种指纹[3]及分块[4]算法,从而可以在任意数据集中确定重复数据子目录。考虑到无线传感器网络设计的资源约束因素,文中提出了一种改进的冗余抑制协议。该解决方案的优点体现在分析流量时抛开了对冗余要素的假设,比如其发生的频率及粒度等,它不会像现有的解决方法一样限制数据分析的搜索空间,从而使其可以适用于不可预知的、任意的通信内容。同时,该协议着眼于发现那些在传输过程中抑制冗余时造成最多存储消耗的重叠数据。由于节点内存容量的限制,只有高端部分的冗余数据将被缓存。在技术层面上,通过将分块算法用于冗余分析,结合带有概率置信度的频率计算数据结构,以保证获取的冗余信息的准确性,并作为是否缓存数据的依据;通过排除最小相关数据以及在运行时调整精度参数的方式来处理连续数据流,以进一步优化解决方案的实施性能。

收稿日期:2010-12-06;**修回日期**:2011-03-10
基金项目:江苏省科技支撑项目(BE2009157)
作者简介:褚轶昕(1984-),女,硕士研究生,研究方向为计算机系统结构;宗 平,博士,教授,研究方向为计算机网络、信息安全等。

1 冗余抑制协议

无线传感器网络有以下特点:无线传感器网络包

括了大面积的空间分布，能源受限制，网络自动配置、自动识别节点，网络的自动管理和高度协作性[5]。而在无线传感器网络中，数据流模型的主要特征表现在四个方面：数据总量被假定是无限的；数据到达速率非常快；数据到达次序不受应用所约束；除非可以保存，每个元素均只能“看”一次。由于无线传感器节点的内存是有限的，使得用于确定及缓存重复数据段的内存空间相当少。为了适应这一特性，文中旨在找到一种方式来识别所需数据段，同时将选择的数据段放入缓存，分析时仅针对拥有最大重传频率比的数据段，而不是当前的所有数据分组。此外，数据分析的过程必须适应冗余的动态性，不应当在初始阶段就缩小查询参数而使得结果有失偏颇。文中给出一种改进的冗余抑制机制，利用 Winnowing 算法对数据流进行分块加以分析，提出了改进的 hCount 算法，在参数为动态变化的情况下确定冗余并控制数据结构的争用。

1.1 冗余分析

无线传感器网络由资源和能量都受到约束的节点构成，节点采用电池作为能量源。在无线传感器网络中，通信能耗是无线传感器网络能耗中比重最大的部分，除此之外，节点的运算能耗也不可忽视。针对其传输流量易重复冗余的特性，可以考虑在发送方和接收方建立统一的冗余确定机制，使用带概率置信度的频率计算方法标识冗余数据，并采用压缩算法对数据段进行处理，以期减少重复数据在网络中的传输，降低传输能耗。

通常确定冗余的方法要求调整初始参数或保存实际数据用以比较，计算开销比较大，从而增加了传输开销[6]。文中设计的冗余分析策略是：首先建立若干数据分块，且每个分块都设置了不同的块大小，再应用 Winnowing 算法[7]，定义不同分块的连续截点间的距离为 w。Winnowing 算法是采用步进式递减的方式，使得距离参数 w 从总数据大小 n 减少到 1，通过多次运行此算法，得到内容对齐的分块，其范围从单个的全分块彻底分裂成单字节。此过程可以理解为在所选截点处反复切分数据，使其成为越来越细的块。在每一次切分后，记录下切分形成了哪些块，并采用指纹标识产生的数据分块。然后对分块的指纹进行比较，以识别匹配的数据段，同时记录它们出现的次数。与现有协议处理策略不同的是，文中没有通过抽样来过滤指纹，也放弃使用定位器位置匹配实际数据作详细比较的方法。这里忽略了连续的数据流本身，而是依赖于某种假定 —— 取自不同粒度的分块所形成的大量指纹可以匹配任意长度的重复数据。为了确定最值得缓存的数据段，文中通过块大小来规范化每个数据块重复的计数。

1.2 冗余抑制协议的设计

在传感器网络中，各个传感器节点都采集数据，发送到中心主机进行处理[8]。传感器节点发送数据的速率和外部条件紧密相关，当特定事件发生时，有些节点会发送更多的数据。对于数据流算法而言，当外部条件变化时，能够根据数据流变化及时调整参数，进而提高性能就非常重要。

尽管 Winnowing 算法将任意大小的数据映射成常数大小的指纹，但是如果大量存储最近传输数据包中提取的数据段，仍会消耗相当大的内存[9]。因此有必要设计一种压缩方式以保存持续的流量指纹计数。文中的主要设计思想是仅缓存反复性最强的那部分数据，那么需要考虑的是如何确定一个有合理准确度的频繁出现的块集合，而不是精确地保存所有指纹及其计数。因此这里采用了概率的频率计算方法。

如何挖掘数据流上的频繁元素是数据流研究的一个基本问题，解决此问题的各种方法区别于衡量参数，计算误差的置信度以及内存的利用。近期研究中认为 hCount[10] 的数据结构在严格的内存限制下具有优越的性能[11]。hCount 是一种基于梗概的数据结构，由 h 个集构成，每个集有 m 个计数器。算法核心数据结构是大小为 $m * h$ 的哈希表 $S[m][h]$[12]，还需要 h 个独立的哈希函数辅助计算。每个哈希函数 H_i 都能够将数字从 $[0,\cdots,M-1]$ 均匀映射到 $[0,\cdots,m-1]$，选用的哈希算法：

$$H_i(e) = ((a_i \cdot e + b_i) \bmod P) \bmod m, 1 \leq i \leq h$$

其中 a_i 和 b_i 是两个随机数，P 是一个大素数。任何一个元素 e 均有 h 个对应的关联计数器，表示为：$\{S[H_0(e)][0], S[H_1(e)][1], \cdots, S[H_{h-1}(e)][h-1]\}$。当检测到指纹重复时则其所有关联计数器的值均增加 1。然后检查哈希表，输出频繁元素及其该元素的估计值，其估计值就是该元素所对应的关联计数器中最小的一个。M 为数据流中相异元素个数，给定三个用户自定义参数：(1) 门槛参数 $\alpha, \alpha \in (0,1)$；(2) 误差参数 $\varepsilon \in (0,1)$；(3) 概率参数 $\rho \approx 1$。在任何时间点上，仅给定一个小的内存空间，该算法都能输出一列元素，且满足：输出所有频率超过门槛参数 α 的元素；不输出频率小于 α 的元素；对于任一个元素，至少以 ρ 的概率保证真实值与估计值之间的差距不超过 ε。

文中选用 hCount 算法，同时需进一步解决频率计数算法的相关问题。因为，这些算法依赖于固定计数容量和属性的初始参数配置。然而在连续的数据流中，某些属性如相异元素个数 M 是不可预知的，也是动态变化的；另一方面，这些参数也界定了数据结构的能力，计算的准确性也难以保证。因此，对于无穷尽的

数据分组,需要注意以及控制数据结构的争用,如在何时清除旧的项等。

hCount 假定相异元素个数 M 是已知的,但是在真实应用中,这个假设却并不总是成立的。因此扩展 hCount 算法,允许其在运行时可以适应参数的变化,使用分阶段建立哈希表的方法来解决相异元素 M 不能够确定的情况。首先设置初始值 M_0 及 M_1,其中 $M_0 = 0$,为了节省内存,$M_1 << M$。然后,创建一个哈希表 S_1,供所有在范围 $[0, M_1 - 1]$ 内的元素使用。当"看到"数据流上一个新的元组,且它超出范围 $[0, M_1 - 1]$ 时,创建一个新的哈希表 S_2。S_2 被用来处理范围 $[M_1, \cdots, M_2 - 1]$ 之间的元素,其中 $M_2 = M_0 + M_1$。依次类推,可以逐步地创建一系列的哈希表来达到目的。当新创建的哈希表仍然不能够满足需求时,上述步骤将被重复执行。即新哈希表 S_{i+1} 的处理数据范围是 $[M_i, M_{i+1} - 1]$,其中 $M_{i+1} = M_{i-1} + M_i$。这种操作在某种程度上会给某阶段的哈希表过度分配内存,超出 M 的需求造成内存的浪费,因此需要在此过程中同时执行以下操作来压缩数据结构。若 S_1、S_2 中,参数 $h_1 \leq h_2, m_1 = 2^x, m_2 = 2^{x+k} (x \geq 0, k \geq 0)$,则丢弃 S_2 保存的 $(h_2 - h_1)$ 个多余计数器集。此外,对于 S_2 中剩余的每一个计数器集,在 S_1 中都有计数器集与之一一对应,将这些集中每一个计数器的二进制索引位置值转化为 k 比特,并将其值加在要转换的 S_2 的索引位置的计数器上。这样使得 S_1 保存了所有的计数器状态,S_2 则被丢弃,且任何 S_2 中附加的计数器的准确性都在此压缩过程中丢失。这里对于是否输出数据的门槛参数 α 也不做具体的初始定义,每次只是输出处于计数器顶端的那些元素,使其可以动态适应数据流的变化。

接下来考虑如何处理无限数据流不停到达时保存计数的问题。当数据结构的争用等级达到内存界限时,就必须清除一些内容。尽管从 hCount 中移除这些数据很容易,但却不能够这么做,因为如果抛弃了所有的原始通信数据,那么所有在此数据段上的信息均不可用。事实上,问题的关键在于并不在乎确切的元素计数,而只关心确定那些重发次数最多的数据内容。基于这一考虑,需要控制梗概的争用等级,将所有正值的内部计数器等量递减,直到争用等级低于允许的内存限制。通过这一方法逐步摒弃了最低频繁项(关联度最低),释放了空间以保存新进项。

上述功能的扩展,一方面使得我们有能力进行动态过滤,并且可以在持续的数据流中依据内存的限制保留数据结构中那些最大频繁项的计数。另一方面允许调整 hCount 的参数以达到与当前保存项属性的最佳匹配。因此无论如何动态分配计数器,内存空间都能得到充分的利用。协议按如下步骤实现。分析要传输的报文,使用 Winnowing 算法进行分块,提取出来的分块在其 hCount 哈希表的关联计数器的值相应增加。同时通过估算当前保存的计数器的计算误差来衡量数据结构的争用等级。如果误差超出阀值或者 hCount 超出现有的 hCount 内存限制,则通过下述方法来进行调整。添加新的 hCount 阶段来扩展容量,所有计数器逐步减少,直到回到误差阀值和内存限制的范围内,同时抛弃在此过程中值减为 0 的阶段。若 hCount 的争用等级允许,多个阶段可以通过压缩进行合并。

2　协议的实现及验证

2.1　协议结构及工作流

发送方对所有发送的数据进行彻底的冗余分析,与接收方同步缓存结果,即只保存每个传入链接的缓存表。当某项被选中进行缓存,发送方使用缓存表索引及报文中的位置偏移量来取代数据段,以便接收方可以从其缓存表中重构数据。这样缓存内容完全由发送方控制,在接收方被利用。这样使得这种同步的工作流更适用于发送方和接收方都独立拥有流量分析的情况。图 1 给出了该协议工作流在发送方的详细处理流程。

2.2　协议分析及仿真

基于分块的冗余分析,这里首先测量仅通过对比块来确定数据段的性能损失。将分析所有潜在数据子集后计算的输出作为基准。图 2 显示了一个从地球物理数据集产生的示例结果。此例刚开始时包含的冗余数据非常少,这一条件使得基于分块的方法得到传输最坏的结果。

由于不能提供所有的在空间受限的情况下的详细结果,这里对若干实验的数据做出总结。实验得到了用于精确计算的 hCount 内存空间是 10kB 和 100kB 情况下缓存性能接近的曲线。图 3 描绘了一个数据集示例,10kB 中有接近一半的性能用于精确计算,而 100 kB 的只有约 10% 的性能损失。即使缓存比 hCount 的内存少几个数量级,只有几 kB 到几十 kB,其性能仍表现良好,在实践中将在收到的相邻节点间被进一步拆分。实验也表明仅对确定的重复冗余块的高端进行缓存,使得缓存的同步开销只增加几个百分点,几乎可以忽略不计。

3　结束语

文中给出了一种实现无线传感器网络冗余抑制的可行解决方案,主要解决了传感器节点严格的内存约束以及寻求一种在传输数据时对其分布和频率不做任何假设的分析冗余的方法。该方法通过基于分块的算法合并指纹,扩展了基于梗概的频率计数算法,并且通

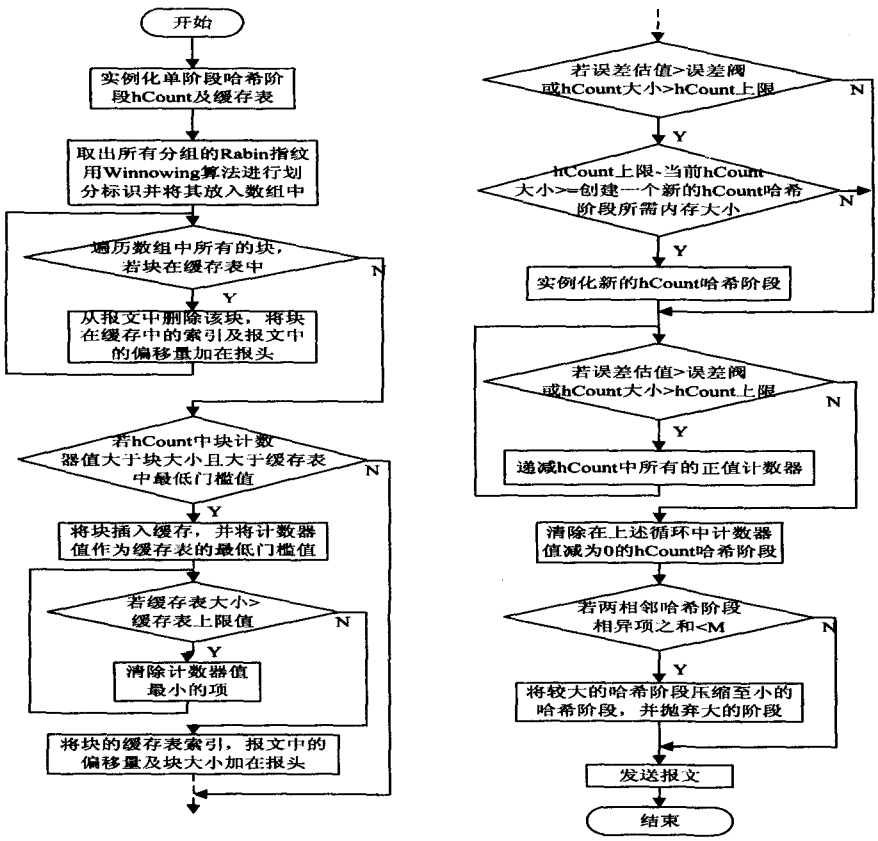

图1 发送方的冗余抑制协议处理流程

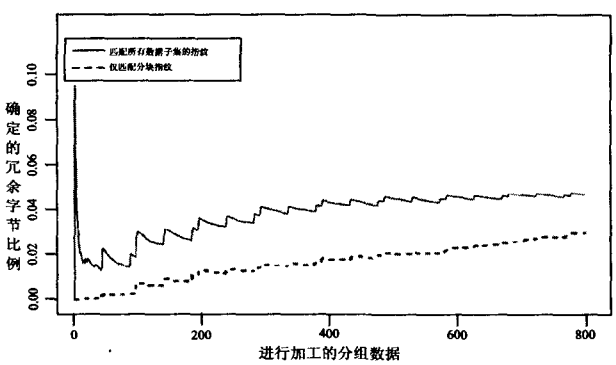

图2 传输数据中确定的冗余比例

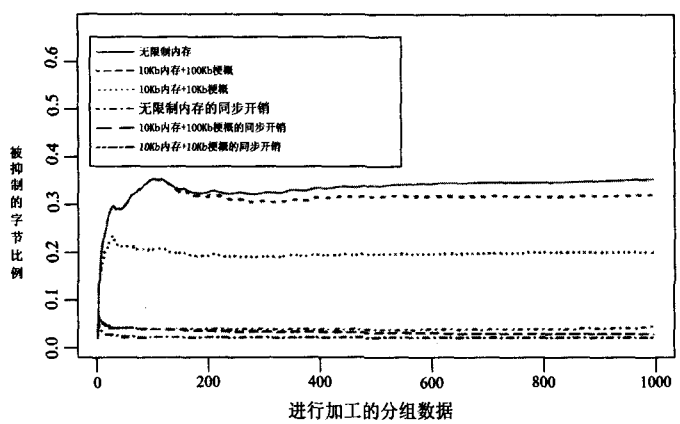

图3 实际的内存限制下冗余抑制的性能表现

过利用它们来选择缓存冗余数据集顶端的数据；同时通过调整其参数以及动态清除内容的方法来优化hCount梗概的内存使用，以达到足够的性能。进一步的研究工作可以考虑协调多个传入节点的链接以及在多跳范围扩展缓存规模，扩展包括通过多路径路由协议实现跨层协调，以便通过这种方式划分流量使得相似数据沿着相同的路线进行路由来达到冗余抑制。

参考文献：

[1] Prabh K P, Abdelzaher T F. Energy - conserving data cache placement in sensor networks [J]. ACM Transactions on Sensor Networks, 2005, 1(2): 178-203.

[2] Kimura N, Latifi S. A survey on data compression in wireless sensor networks [J]. Information Technology: Coding and Computing, 2005(2): 8-13.

[3] Rabin M. Fingerprinting by random polynomials [R]. USA: Harvard University Department of Computer Science, 1981.

[4] Bjorner N, Blass A, Gurevich Y. Content-dependent chunking for differential compression, the local maximum approach [J]. Journal of Computer and System Sciences, 2009, 76(3-4): 154-203.

[5] 储昭勋，胡艳军. 无线传感器网络技术[J]. 计算机技术与发展，2006，16(4)：64-65.

[6] 王 娟，王汝传，孙力娟. 数据融合在传感器网络协议中的节能性分析[J]. 计算机技术与发展，2006，16(11)：4-6.

[7] Schleimer S, Wilkerson D S, Aiken A. Winnowing: local algorithms for document fingerprinting [C]//Proceedings of the-ACM Sigmod. San Diego, CA, USA: ACM Press, 2003: 76-85.

[8] 王 殊，阎睿杰. 无线传感器网络的理论及应用[M]. 北京：北京航空航天大学出版社，2007.

[9] 边肇祺，张学工. 模式识别[M]. 第2版. 北京：清华大学出版社，1999.

[10] Jin C, Qian W, Sha C, et al. Dynamically maintaining *frequent items over a data stream* [C]//Proceedings of the 12th ACM CIKM. San Diego, CA, USA: ACM Press, 2003: 287-294.

[11] Manerikar N, Palpanas T. Frequent items in streaming data: An experimental evaluation of the state-of-theart [J]. Data and Knowledge Engineering, 2009, 68(4): 415-430.

[12] 金澈清. 数据流上若干查询处理算法的研究[D]. 上海：复旦大学，2005：23-27.