doi:10.3969/j.issn.1001-893x.2014.06.022
引用格式:喻歆. 一种基于传播时延修正的网络时间互同步算法[J]. 电讯技术,2014,54(6):813-817. [YU Xin. A Mutual Network Synchronization Algorithm with Considering Propagation Delay[J]. Telecommunication Engineering,2014,54(6):813-817.]

一种基于传播时延修正的网络时间互同步算法*

喻　歆**

(中国西南电子技术研究所,成都 610036)

摘　要:介绍了一种时间互同步算法的基本原理,并通过仿真试验分析了该算法在不同环境下的性能,指出其应用在地理覆盖范围较大的网络中的局限性。为改善该算法在实际应用场景中的性能,提出了一种基于传播时延修正的新策略。仿真结果表明,改进后的时间同步算法在传播时延不能忽略的场景中的性能远远优于原来的算法,其同步精度由原算法的 1 ms 左右提高到几微秒到几十微秒的量级,在实际工程中具有较高的应用价值。

关键词:无线传感器网络;时间同步;互同步;传播时延;时钟偏移;时钟频率偏移
中图分类号:TN915.04;TP802　　**文献标志码**:A　　**文章编号**:1001-893X(2014)06-0813-05

A Mutual Network Synchronization Algorithm with Considering Propagation Delay

YU Xin

(Southwest China Institute of Electronic Technology, Chengdu 610036, China)

Abstract: The elements of a mutual network algorithm are given. The performance of the algorithm is analyzed empirically in different situations considering network node density, network geographic coverage, network dynamics and uncertainties. The degradation of the algorithm in performance is observed in situations where the propagation delay cannot be ignored. In order to improve the performance, a new strategy based on correcting the propagation delay is proposed. Simulation results show that the improved algorithm outperforms the original algorithm largely with respect to the synchronization error and the synchronization error is reduced from about 1 millisecond to a few to tens of microseconds, which shows good prospect of the improved algorithm in practice.

Key words: wireless sensor network; time synchronization; mutual synchronization; propagation delay; clock offset; clock skew

1　引　言

随着微机电系统和无线通信技术的不断发展,无线传感器网络(Wireless Sensor Network,WSN)在人体、动植物和环境监控,生化威胁探测,国土安全,以及诸多军事应用领域已经开始发挥出重要的功用[1-2]。网络时间同步指的是将网络中的所有时钟校正到同一个时间标准的过程,其对 WSNs 诸多操作和功能的正常运转起到了至关重要的作用[2]。

全球导航卫星系统(Global Navigation Satellite Systems,GNSS)能够很容易地解决时间同步问题,并且获得较高的精度[3]。但该方法对每个网络节点的硬件配置提出了额外要求,增加了成本,并且满足硬件配置要求的节点也无法与缺少相关硬件配置的网络节点进行时间同步。此外,即使所有网络节点

* 收稿日期:2014-01-26;修回日期:2014-04-08　Received date:2014-01-26;Revised date:2014-04-08
** 通讯作者:hughdeudeu@gmail.com　**Corresponding author**:hughdeudeu@gmail.com

都具备了通过 GNSS 进行时间同步的硬件配置,受通信系统所处环境(如室内、地下、森林等)影响,GNSS 也无法总是提供可靠的网络时钟参考。因此,必须设计额外的网络时间同步机制。

过去十多年间,学者们提出了许多有效的时间同步算法[4-10],这些算法大致可以分为中心式和分布式两类。中心式方法中,网络中存在一个参考时钟或称为主时钟,网络中的其他时钟都以主时钟为基准进行校正,而分布式方法则不需要参考时钟。较为流行的中心式方法包括 TPSN(Time-synchronization Protocol for Sensor Networks)[6]、FTSP(Flooding Time Synchronization Protocol)[7] 和 RBS(Reference Broadcast Synchronization)[8]。TPSN 和 FTSP 中,当节点失效或者在移动环境中,选择新的根节点需要额外的通信开销。此外,叶子节点到根节点的距离会直接影响其同步精度。RBS 中,主节点的选择和网络分簇的过程中都会引入较大的通信开销。分布式时间同步方法没有主时钟节点,因此不受单点失效的影响,具有较强的健壮性。同时,没有对主时钟节点的选择、管理和维护也使得这类时间同步方法具备较小的开销。

近年来,一种基于时钟采样的网络时间互同步(Clock-Sampling Mutual Network Synchronization,CS-MNS)[9]算法因其简单有效的特性备受关注,并且已经被用于实际系统当中[10]。CS-MNS 算法中,每个节点与其 1 跳邻居周期交互同步消息,并基于该信息修正调整时钟偏移和时钟频率偏移的校正因子。CS-MNS 算法具有诸多优点,例如无需直接访问物理时钟,无需主时钟,实现简单,同步精度较高等[9]。然而,CS-MNS 并没有考虑各种时延对其性能的影响,这可能导致其在某些场景尤其是在具有较大地理覆盖范围的网络中性能不能满足要求。因此,本文首先简单介绍 CS-MNS 算法并分析其性能,然后对其进行改进,以期提高该算法在覆盖更大地理范围和处于更复杂环境中的 Ad Hoc 网络中的性能。

2 CS-MNS 算法简介及分析

2.1 时钟定义及问题描述

一个包含 N 个节点的网络,每个节点的时钟都有不同的时钟频率偏移和初始时间,即第 i 个节点时钟的时间过程 $T_i(t)$ 表示为

$$T_i(t)=\beta_i t+T_i(0),i=\{1,2,\cdots,N\} \quad (1)$$

式中,t 为真实时间,β_i 表示第 i 个时钟相对于真实时间的时钟频率偏移,$T_i(0)$ 为第 i 个时钟的初始时间。

时间同步算法的目标是使网络中的所有时钟的时间过程满足:对于任意的 $i\neq j$,都有

$$\| T_i(t>t_c)-T_j(t>t_c) \| \leqslant \Delta T$$

其中,t_c 表示算法的收敛时间,ΔT 表示容忍的同步误差。

2.2 CS-MNS 算法简介

CS-MNS 对虚拟时钟进行同步,虚拟时戳由真实时戳乘上校正因子获得;CS-MNS 根据收到的虚拟时戳消息对校正因子进行修正,从而达到同步虚拟时钟的目的[9]。虚拟时戳和校正因子的修正表达式分别为

$$\overline{T_i}(n\tau)=s_i(n\tau)\beta_i n\tau+s_i(n\tau)T_i(0),\ i=\{1,2,\cdots,N\},n\in Z^+ \quad (2)$$

$$s_i((n+1)\tau)=s_i(n\tau)+k_p\frac{\overline{T_{rt}}(n\tau)-\overline{T_i}(n\tau)}{\overline{T_i}(n\tau)-\overline{T_i}(0)} \quad (3)$$

式中,τ 为采样周期,常数 k_p 称为控制增益,$\overline{T_{rt}}$ 为第 i 个节点收到的发送时钟发送的时戳消息。

从公式(3)可以看出,每个节点发送时戳消息时只需包含发送该消息时本地的虚拟时戳;一个节点一旦接收到一条有效的时戳消息,就能立即独立地对时钟偏移和时钟频率偏移进行校正。

需要注意的是,CS-MNS 不能为网络提供统一的绝对参考时间,但是统一的虚拟时钟能够保证如 TDMA 类协议的正常运转。

2.3 CS-MNS 算法的仿真分析

2.3.1 仿真环境设置

文献[9]在只有两个节点的场景中理论证明了 CS-MNS 的稳定性和收敛性,并通过仿真分析了 CS-MNS 在较大的网络规模(节点数 60 ~ 140)、不同的控制增益 k_p(取值 0.1 ~ 1.95)和不同的晶振稳定性(时钟偏移频率在某个时间段内发生突变)时的性能。为了更深入地分析和理解 CS-MNS 在不同场景中的性能,并考虑到本研究的应用场景,本文针对相对较小规模和较稀疏的网络展开一系列研究和分析。

每一次仿真时,算法每完成一次同步迭代操作,我们就记录当前网络中所有节点虚拟时间之间的最大差值的绝对值;针对每一组参数,我们都执行了 100 次仿真,最终仿真结果由 100 次仿真的平均值

得到。算法使用 MATLAB 编码实现,开发环境为 MATLAB Version 7.1.0.246 (R14) Service Pack 3,平台环境为 Windows XP Professional SP 3 操作系统,Intel Core 2 Quad CPU Q8400 @ 2.66 GHz 2.66 GHz的 CPU,以及1.96 GB的内存。

2.3.2　仿真结果及分析

本文一共进行了 5 组仿真,每组仿真采用的公共参数设置如表 1 所示。

5 组仿真中,仿真 1 研究网络节点密度,即节点的平均单跳邻居数目的影响;仿真 2 研究消息交互周期的影响;仿真 3 研究消息接收成功率的影响;仿真 4 研究不同拓扑结构(体现在网络节点密度和网络覆盖范围)的影响;仿真 5 综合研究各种因素包括网络覆盖范围、网络节点密度、网络动态性以及消息传播时延的影响。每组仿真各自的仿真参数如表 2 所示。

表 1　每组仿真采用的公共参数设置

Table 1 Common parameters setting of all simulations

考虑因素及参数	取值
时钟频率偏移	每个节点取值: $[-40, 40]\times10^{-6}$ 中的随机值
初始时钟	每个节点取值: [0, 1 000] μs 中的随机值
校正因子 $s_i(0)$	1
控制增益 k_p	0.3

表 2　每组仿真采用的参数设置(区间符号表示取区间内的随机值)

Table 2 Other parameters setting of each simulation (interval notation means random values within the range)

考虑因素		仿真 1	仿真 2	仿真 3	仿真 4	仿真 5
节点数		2 ~ 10,步进 2	9	9	9	9
覆盖范围		单跳	单跳	单跳	由图 1 拓扑决定	所有节点分布在 1 000 km×1 000 km 范围内
节点密度		节点数-1	8	8	由图 1 拓扑决定	由覆盖范围、节点位置和通信距离决定; 每个节点最大通信距离 300 km
网络动态性		静止	静止	静止	静止	每个节点每经过[a,b]s 移动到 1 000 km×1 000 km 内的随机位置; 其中[a, b]s:[1, 5]s、[6, 10]s、[11, 15]s
不确定性	消息交互周期/s	1	1,[1,3],[1,5], [1,7],[1,9]	[1,5]	[1,5]	[1,5]
	消息传播时延	忽略	忽略	忽略	忽略	由通信距离决定
	消息接收成功率	100%	100%	0.2 至 1, 步进 0.2	100%	100%

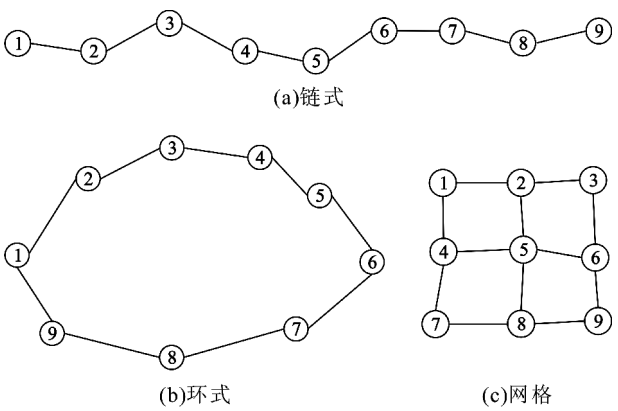


图 1　仿真 4 中测试的 3 种拓扑结构:链式、环式和网格

Fig. 1 Three network topologies tested in the fourth simulation: tandem, ring and grid

根据仿真结果可以观察到的现象以及得出的结论分析如下:

首先,在消息接收成功率 100% 的前提下,网络节点密度越大,算法的收敛速度越快。网络节点密度较小时(2 个节点和 4 个节点),算法甚至会先经历一个发散的过程,最终才会收敛到一个稳定状态。网络节点密度越大,每个节点修正自己时间时,就能参考网络中更多节点的折衷时间信息,从而加速了收敛的过程;

其次,消息交互周期越长,算法收敛速度越慢;消息接收成功率越低,算法收敛速度越慢。原因很直观,因为消息交互周期的变长和消息接收成功率的降低都会导致节点之间时戳消息交互次数的减少,从而减少了算法的成功迭代次数。对于一些基于竞争的 MAC(Media Access Control)层接入协议,网络节点密度的增大将增加节点之间碰撞的概率,导致消息接收成功率降低[11],所以使用此类 MAC

协议的网络中,节点密度的增大并不总是会加快 CS-MNS 的收敛过程。此外,虽然缩短消息交互周期可以提高算法收敛速度,但是增加了网络开销。还需注意的是,即使在消息接收成功率只有 20% 时,算法仍然能够收敛;

第三,仿真网格拓扑结构收敛最快,链式拓扑结构收敛最慢。仔细分析各拓扑结构,链式结构中,节点之间最小间隔跳数的最大值为 7,节点平均邻居数约为 1.78;环式结构中,节点之间最小间隔跳数的最大值为 4,节点平均邻居数为 2;网格结构中,节点之间最小间隔跳数的最大值为 3,节点平均邻居数为 2.67。由前面分析结果,在 100% 的消息接收成功率前提下,节点平均邻居数越多,收敛越快;此外,节点之间间隔跳数越多,单跳之间时间相互修正的结果传播到全网就需要越长的时间,因此算法在网格结构中收敛最快,链式结构中收敛最慢;

最后,仿真网络动态性越高(节点移动速度越快),算法收敛越快。这是因为,动态性越高的网络,节点的平局邻居数在时间轴上作平均后大于动态性较低的网络中对应的值。此外,可以发现由于考虑了传播时延,虽然算法能够收敛,但是同步精度受到了较大的影响。

综上,CS-MNS 算法具有很强的健壮性,其在不同场景和环境中都能够保证收敛。但是该算法受消息传播时延影响较大,同步精度性能较差(达到了 1 ms左右)。

3 CS-MNS 算法的改进

3.1 基于传播时延修正的 CS-MNS 算法

从对 CS-MNS 的仿真分析中可以发现传播时延对其同步精度的性能影响较大,而在实际的应用场景中,尤其是在节点之间间距较远时传播时延是不能忽略的。因此,本文设计了一种基于传播时延修正的 CS-MNS 算法,记为 CS-MNS-d,以改进算法在实际应用场景中的同步精度性能。

具体地,CS-MNS-d 根据传播时延对公式(3)所示 CS-MNS 的校正因子递归控制规则进行了修正。新的校正因子递归控制规则如下所示:

$$s_i((n+1)\tau)=s_i(n\tau)+k_p\frac{\overline{T_{xj}}(n\tau)-\overline{T_{ri}}(n\tau)-TP_{ij}}{\overline{T_i}(n\tau)-\overline{T_i}(0)} \quad (4)$$

其中,$\overline{T_{xj}}(n\tau)$ 为节点 j 发送其时间戳消息时节点 j 的虚拟时戳,$\overline{T_{ri}}(n\tau)$ 是节点 i 收到节点 j 的该时戳消息时节点 i 的虚拟时戳,$\overline{T_i}(n\tau)$ 为节点 i 更新校正因子时节点 i 的虚拟时戳,TP_{ij} 为根据节点 i 和 j 的真实时戳(真实时戳为稳定时戳)计算的该时戳消息的传播时延。

根据真实时戳计算节点间传播时延原理如图 2 所示。计算时,忽略了节点 i 和 j 之间的时钟频率偏移差别。

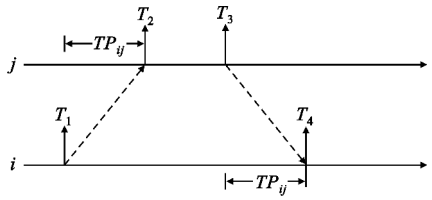


图 2 传播时延计算原理图

Fig. 2 The principle of calculating propagation delay

令 ΔT_{ij} 表示之间的时钟偏差,则有

$$T_1+\Delta T_{ij}+TP_{ij}=T_2 \quad (5)$$

$$T_4+\Delta T_{ij}-TP_{ij}=T_3 \quad (6)$$

由公式(5)和公式(6)可得

$$TP_{ij}=\frac{(T_4-T_1)-(T_3-T_2)}{2} \quad (7)$$

注意,T_1 和 T_4 为节点 i 的真实时戳,T_2 和 T_3 为节点 j 的真实时戳。因此,每个节点发送的同步消息中需要同时包含虚拟和真实时戳。

3.2 仿真性能对比

采用与上一节仿真 5 完全一样的场景和算法参数设置,经过 100 次仿真后,CS-MNS-d 的仿真结果如图 3 所示。

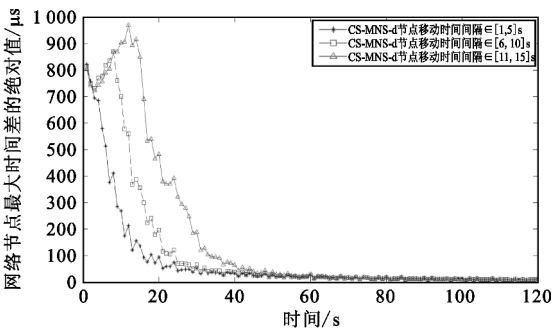


图 3 CS-MNS-d 在仿真 5 场景中的仿真结果

Fig. 3 Results of the CS-MNS-d algorithm in the scenario of the fifth simulation

由图 3 所示结果可以看出,CS-MNS-d 在 3 种动态场景中都能快速收敛,即使在最慢的情况下(节点移动时间间隔∈[11, 15]s),算法在不到40 s

的时间内同步误差能够达到100 μs以下,在不到60 s的时间内同步误差能够达到50 μs以下,而 CS-MNS 在最好情况下在30 s左右就接近收敛,并且最终收敛时的同步误差也在1 000 μs以上。因此,相对于 CS-MNS,CS-MNS-d 更适用于传播时延不能忽略的覆盖了较大地理范围的实际无线通信网络中。

4 结　语

本文通过仿真分析,说明了 CS-MNS 算法具有较强的健壮性,以及其在不同场景和环境中具有较好的收敛性,但是该 CS-MNS 算法受消息传播时延影响较大。基于此,本文提出了一种基于传播时延修正的改进算法,仿真结果表明,改进后的算法在不能忽略传播时延的环境中同步精度大大提高,具有一定的实际应用价值。CS-MNS 还存在一些需要进一步改善的地方,例如算法迭代初期,网络中的时钟会有一个发散的过程,因此增加了算法收敛的时间;再如,如果网络初始时间相差非常大时 CS-MNS 会不稳定等,这些都是进一步研究的重点。

参考文献:

[1] 任丰原,黄海宁,林闯. 无线传感器网络[J]. 软件学报,2003,14(7):1282-1291.
REN Feng-yuan, HUANG Hai-ning, LIN Chuang. Wireless Sensor Networks [J]. Journal of Software, 2003, 14(7): 1282-1291. (in Chinese)

[2] Wu Y-C, Chaudhari Q, Serpedin E. Clock Synchronization of Wireless Sensor Networks[J]. IEEE Signal Processing Magazine, 2011, 28(1): 124-138.

[3] 黄琳,马文杰,李国军,等. 基于 GNSS 的通信星座系统的时间同步研究[J]. 航天器工程,2010,19(6):67-74.
HUANG Lin, MA Wen-jie, LI Guo-jun, et al. Study on GNSS-based Time Synchronization for a Communications Satellite Constellation System[J]. Spacecraft Engineering, 2010, 19(6): 67-74. (in Chinese)

[4] Sivrikaya F, Yener B. Time Synchronization in Sensor Networks: A Survey[J]. IEEE Network, 2004, 18(4): 45-50.

[5] 易胜蓝. 嵌入式 Linux 下 IEEE 1588 时间同步的实现[J]. 电讯技术,2012,52(5):800-803.
YI Sheng-lan. Implementation of IEEE 1588 Time Synchronization under Embedded Linux[J]. Telecommunication Engineering, 2012, 52(5): 800-803. (in Chinese)

[6] Ganeriwal S, Kumar R, Srivastava M B. Timing-sync Protocol for Sensor Networks [M]. New York: ACM Press, 2003:138-149.

[7] Maroti M, Kusy B, Simon G, et al. The Flooding Time Synchronization Protocol [M]. New York: ACM Press, 2004: 39-49.

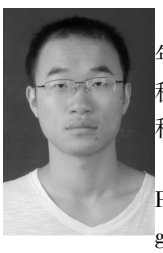
[8] Elson J, Girod L, Estrin D. Fine-grained Network Time Synchronization Using Reference Broadcasts [C]//Proceedings of the Fifth Symposium on Operating System Design and Implementation. Boston, MA: IEEE, 2002: 163-167.

[9] Rentel C H, Kunz T. A Mutual Network Synchronization Method for Wireless Ad Hoc and Sensor Networks [J]. IEEE Transactions on Mobile Computing, 2008, 7(5): 633-646.

[10] Kunz T, MacNeil M. Implementing Clock Synchronization in WSN: CS-MNS vs. FTSP[C]// Proceedings of the 2011 IEEE 7th International Conference on Wireless and Mobile Computing, Networking and Communications. Wuhan: IEEE, 2011: 157-164.

[11] 蹇强,龚正虎,朱培栋,等. 无线传感器网络 MAC 协议研究进展[J]. 软件学报,2008,19(2):389-403.
JIAN Qiang, GONG Zheng-hu, ZHU Pei-dong, et al. Overview of MAC Protocols in Wireless Sensor Networks [J]. Journal of Software, 2008, 19(2): 389-403. (in Chinese)

作者简介:



喻　歆(1984—),男,四川成都人,2011年获计算机应用技术专业博士学位,现为工程师,主要研究方向为计算智能、无线通信、移动自组织网络等。

YU Xin was born in Chengdu, Sichuan Province, in 1984. He received the Ph. D. degree in Computer Application Technology in 2011. He is now an engineer. His research interests include computational intelligence, wireless communication, and mobile ad hoc networks.

Email: hughdeudeu@gmail.com