

Internet 网络时延测量中的时钟同步算法

孙海燕, 侯朝桢

(北京理工大学自动控制系, 北京 100081)

摘要: 针对单向网络性能测量过程中存在的时钟同步问题, 提出了基于法向距离最小的优化目标。该文根据优化目标推导了时钟同步优化算法, 从而提高单向网络时延测量的精确性。并针对一个实际的网络时延测量结果进行了分析, 验证了该算法的有效性。

关键词: 网络时延测量; 时钟同步; 时钟偏移; 时滞; 凸集

Clock Synchronization Algorithm for Internet Delay Measurements

SUN Haiyan, HOU Chaozhen

(Department of Automatic Control, Beijing Institute of Technology, Beijing 100081)

【Abstract】 For the problems of clock synchronization in one-way network performance measurement. This paper puts forward the optimizing goal of minimizing normal distance. And the clock synchronization optimization algorithm which improves the accuracy of one-way network delay measurement is deduced based on the optimizing goal. Finally, the algorithm proves effective by analyzing an actual network delay measurement result.

【Key words】 Network delay measurements; Clock synchronization; Clock offset; Clock skew; Convex set

网络性能测量是对 Internet 网络进行精确控制的重要前提条件。在进行单向网络时延和数据包丢失率等网络性能参数测量时, 网络链路两端的时钟是否同步, 是测量结果是否精确的关键。因为时钟不同步导致的时钟误差是单向网络性能测量中的主要误差。因此, 需要利用时钟同步算法, 估算出 2 个时钟之间的误差, 从而得到真实的测量结果。

目前, 国内外许多学者对时钟同步算法进行了深入研究, 先后提出了线性规划化算法、Paxson 算法、分段最小化算法和最小面积算法^[1~5]等。由于网络负载情况是随时变化的, 需要以在线方式实时地对时钟误差进行估计, 因此数据处理量较大, 有可能达到几百或上千个数据。但这些传统的优化方法每次只能处理几十个数据, 计算复杂, 速度跟不上网络的变化情况, 实用性较低。

文献[2]中提出的基于凸集理论的非传统的优化算法, 计算量小, 编程方法简单, 处理数据及时, 非常适用于计算机在线实时估计时钟误差。但基于垂直距离最小的优化目标还不够理想。本文提出了基于法向距离最小的优化目标, 并根据此目标, 利用凸集性质推导得到估算时钟误差的同步算法, 使估算结果更加精确。

1 问题描述

2 个时钟之间存在的“误差”包括时钟偏移和时钟时滞, 相关定义如下:

(1) 时钟偏移(clock offset): 指在某一时刻时钟时间与准确时间之间的误差;

(2) 时钟时滞(clock skew): 指在某一时刻时钟频率与准确时钟频率之间的误差。时钟频率是时钟偏移的一阶导数。

“时钟同步”定义为单向时延测量两端时钟之间的时钟偏移和时钟时滞均为零。在实际的测量过程中, 时钟偏移可以采用网络时间协议(Network Time Protocol, NTP)进行同步设

置。但时钟时滞是时钟本身的固有频率(即速度)之间的偏差, 无法用网络时间协议进行调整。一个实际的单向网络时延测量结果如图 1 所示。

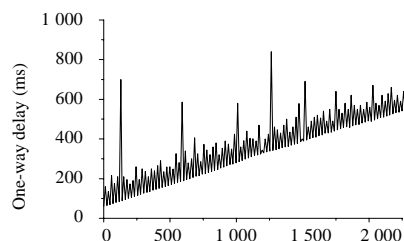


图 1 单向时延测量

在测量数据集合 $\Omega := \{w_i = (t_i, d_i), i = 1, \dots, N\}$ 中, 横坐标轴 x 表示测量时刻 t_i , 纵坐标轴 y 表示网络时延 d_i 。可以看出单向网络时延 d_i 有不断增大的趋势, 这就是时钟时滞对测量结果的不良影响。因此, 本文提出了一种时钟同步改进算法, 估算时钟时滞, 并消除其对单向网络时延测量结果的影响。

如果在网络性能测量中不考虑时钟调整, 可以用一条固定的、非零斜率的直线表示时钟时滞 $L := \{(x, y) | y = \alpha x + \beta\}$ 。因此, 估算时钟时滞的问题可以转化为找一条表示时钟时滞的直线 L , 它低于在集合 Ω 中所有的时延测量点, 并在某种意义上“最接近于”集合 Ω ^[2]。根据测量得到离散的时延数据 w_i , 利用时钟同步算法计算并优化后, 得到 α 和 β 的估计

基金项目: 国防预研基金资助项目

作者简介: 孙海燕(1975—), 女, 博士生, 主研方向: 网络控制系统; 侯朝桢, 教授、博导

收稿日期: 2005-09-28 **E-mail:** sun_hai_yan@bit.edu.cn

值 $\hat{\alpha}$ 和 $\hat{\beta}$ ，从而确定了表示时钟时滞 \bar{d}_i 的直线 $L: \bar{d}_i = \hat{\alpha}t_i + \hat{\beta}$ 。在每个测量时刻， t_i 的真实时延 τ_i ，由测量时延 d_i 减去时钟时滞 \bar{d}_i 得到，即 $\tau_i = d_i - \bar{d}_i$ 。

2 时钟同步算法

在保证一条直线 L 最接近于一些离散点集合 Ω 的目标下，得到这条直线 L 的参数 $\hat{\alpha}$ 和 $\hat{\beta}$ ，是一个优化问题^[1,2]。因此，时钟同步算法是一个优化算法。其限制条件就是使得集合 Ω 中所有的点都在表示时钟时滞的直线 L 之上，保证真实时延 τ_i 的非负性^[2]。即

$$d_i \geq \bar{d}_i = \hat{\alpha}t_i + \hat{\beta} \quad (1)$$

在满足这一条件的直线中，找到最接近于离散测量点集合 Ω 的直线。

2.1 优化目标

有许多度量一条直线和一些离散点在某种意义下的“接近”程度的方法，可以依据不同度量方法，提出表示直线 L 最接近于集合 Ω 的优化目标。文献^[1,2]中提出的优化目标是使所有离散点到直线的垂直距离之和为最小，并估算出时滞直线参数 $\hat{\alpha}$ 和 $\hat{\beta}$ 。目标函数^[1,2]为

$$obj := \sum_{i=1}^N (d_i - \bar{d}_i) = \sum_{i=1}^N (d_i - \hat{\alpha}t_i - \hat{\beta}) = \sum_{i=1}^N d_i - \hat{\alpha} \sum_{i=1}^N t_i - N\hat{\beta} \quad (2)$$

其中， N 表示测量的次数。

用几何图形表示一个点到一条直线的“距离”如图 2 所示。

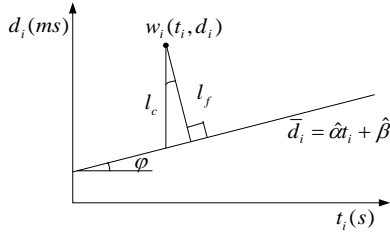


图 2 点到直线的距离

其中，离散点 d_i 到直线 L 的垂直距离为 l_c ，法线上的法向距离为 l_f ， φ 为直线 L 与横坐标轴的夹角。则有

$$\hat{\alpha} = tg(\varphi) \quad (3)$$

$$l_c = d_i - \bar{d}_i = d_i - t_i tg(\varphi) - \hat{\beta} \quad (4)$$

$$l_f = l_c \cos(\varphi) \quad (0 < \varphi < 90^\circ) \quad (5)$$

因此， $l_f < l_c$ ，法向距离小于垂直距离。所以，在法向距离表示下的离散点与直线的接近程度，优于用垂直距离表示的接近程度。

根据以上分析，本文提出了一个基于法线距离最小的优化目标，使得已知离散点集合 Ω 到所求直线 L 到法向距离之和为最小，目标函数为

$$obj_1 := \sum_{i=1}^N l_{f_i} = \sum_{i=1}^N (d_i - t_i tg(\varphi) - \hat{\beta}) \cos(\varphi) \quad (6)$$

$$= \cos(\varphi) \sum_{i=1}^N d_i - \sin(\varphi) \sum_{i=1}^N t_i - N\hat{\beta} \cos(\varphi)$$

因此，本文提出的优化目标与文献^[1, 2]中提出的优化目标相比，所求直线与测量的离散点更“接近”，优化结果更理想。

此外，还有一种优化目标是最小化离散点组成的曲线与时滞直线之间的面积^[2]。优化目标函数为^[2]

$$obj_2 := \sum_{i=1}^{N-1} (d_i - \hat{\alpha}t_i - \hat{\beta} + d_{i+1} - \hat{\alpha}t_{i+1} - \hat{\beta}) \frac{(t_{i+1} - t_i)}{2} \quad (7)$$

$$= \sum_{i=1}^{N-1} \frac{(d_i + d_{i+1})(t_{i+1} - t_i)}{2} - \frac{t_N^2 - t_1^2}{2} \hat{\alpha} - (t_N - t_1) \hat{\beta}$$

以上提出的不同优化目标，是从不同的特征角度来表示点到直线的接近程度。每个目标都可能在某些情况下非常适用，但不太适合其他情况。因此，在进行时钟同步算法优化计算时，要综合考虑优化目标，从而得到更好的计算效果。

2.2 凸集理论^[2]

利用凸集理论，针对离散测量点集合 Ω 定义一个凸集为

$$co(\Omega) := \left\{ x \mid x = \sum_i \mu_i w_i, \mu_i \geq 0, \sum_i \mu_i = 1, w_i \in \Omega \right\} \quad (8)$$

使得所有的离散点都包含在凸集中。凸集的最低边界由分段线性方程组成。如果凸集 $co(\Omega)$ 的最低边界位于直线 L 之上，可以保证凸集中所有的点都位于直线 L 的上方。可以找到一个算法，最多进行 $2N$ 次计算，就可以找到凸集的最低边界。

求解凸集最低边界的算法步骤如下：

(1) 初始化： w_1 压栈； w_2 压栈；

(2) For $i=3$ to N

If (w_i 在堆栈顶点和堆栈次点连线的上方)

{ w_i 压栈； }

Else

While (w_i 在堆栈顶点和堆栈次点连线的下方 and 堆栈尺寸

大于 1)

{ 堆栈顶点出栈； w_i 压栈； }

(3) End

直线 L 最接近于离散测量点集合 Ω 的程度是在凸集 $co(\Omega)$ 之下，但至少有一个点与凸集“接触”，这样的接触点既在凸集边界上，又在直线 L 上。如图 3 所示。

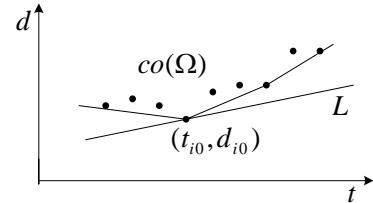


图 3 凸集和时滞直线的关系

因此，可以先求得凸集最低边界的最小点坐标 $w_{i_0}(t_{i_0}, d_{i_0})$ ，从而得到所求直线 L 上的一个已知点的坐标 (t_{i_0}, d_{i_0}) 。

2.3 凸集优化算法

本文利用凸集方法对基于法向距离最小的优化目标进行优化算法设计，具体描述如下：

首先，依照 2.2 节的求解凸集的算法步骤，利用凸集理论得到包含所有离散测量点的凸集 $co(\Omega)$ 。

然后，如果固定 $\hat{\alpha}$ ，则最小面积优化目标式(7)变为

$$f(\hat{\alpha}) = \min_{\hat{\beta}} (q_1 - q_2 \hat{\alpha} - q_3 \hat{\beta}) \quad (9)$$

$$s.t. t_i \hat{\alpha} + \hat{\beta} \leq d_i, i = 1, \dots, N.$$

其中， $q_1 = \sum_{i=1}^{N-1} \frac{(d_i + d_{i+1})(t_{i+1} - t_i)}{2}$ ， $q_2 = \frac{t_N^2 - t_1^2}{2}$ ， $q_3 = t_N - t_1$ 。

根据凸集理论可知，在凸集中，目标函数 $f(\hat{\alpha})$ 的任一局部极小点都是 $f(\hat{\alpha})$ 在非空可行集上的全局极小点。因此有

$$\begin{aligned} f(\hat{\alpha}) &= q_1 - q_2 \hat{\alpha} - q_3 \min \{d_i - t_i \hat{\alpha}\} \\ &= q_1 - q_2 \hat{\alpha} + q_3 \max \{t_i \hat{\alpha} - d_i\} \end{aligned} \quad (10)$$

设 $w_{i_0}(t_{i_0}, d_{i_0})$ 为凸集 $co(\Omega)$ 的最低点, 直线 L 必然经过该点。因此, 该点既在凸集集合 $co(\Omega)$ 中, 又在直线 L 上。则有

$$f(\hat{\alpha}) = q_1 - q_2 \hat{\alpha} + q_3 (t_{i_0} \hat{\alpha} - d_{i_0}) \quad (11)$$

对该式求导, 得到

$$f'(\hat{\alpha}) = -q_2 + q_3 t_{i_0} \quad (12)$$

在凸集 $co(\Omega)$ 的最低点 $w_{i_0}(t_{i_0}, d_{i_0})$ 处, 必然存在 $f'(\hat{\alpha}) = 0$ 。则有

$$t_{i_0} = \frac{q_2}{q_3} \quad (13)$$

此时, 最优化目标 obj_2 取得了最小值。

当求得最低点的横坐标 t_{i_0} 后, 就可以通过测量点 (t_{i_0}, d_{i_0}) 求对应的纵坐标 d_{i_0} 。下面通过法向距离最小的目标得到所求直线的斜率 $\hat{\alpha}$ 或 $tg(\varphi)$ 。

如果 φ 的大小固定, 则法向距离最小的优化目标式(6)变为

$$\begin{aligned} f(\varphi) &= \min_{\hat{\beta}} (p_1 \cos(\varphi) - p_2 \sin(\varphi) - p_3 \cos(\varphi) \hat{\beta}) \\ \text{s.t. } t_i tg(\varphi) + \hat{\beta} &\leq d_i, i = 1, \dots, N \end{aligned} \quad (14)$$

其中, $p_1 = \sum_{i=1}^N d_i$, $p_2 = \sum_{i=1}^N t_i$, $p_3 = N$ 。根据凸集理论可知,

在凸集中, 目标函数 $f(\varphi)$ 的任一局部极小点, 都是 $f(\varphi)$ 在非空可行集上的全局极小点。因此, 有

$$\begin{aligned} f(\varphi) &= p_1 \cos(\varphi) - p_2 \sin(\varphi) - p_3 \cos(\varphi) \min \{d_i - t_i tg(\varphi)\} \\ &= p_1 \cos(\varphi) - p_2 \sin(\varphi) + p_3 \cos(\varphi) \max \{t_i tg(\varphi) - d_i\} \end{aligned} \quad (15)$$

将凸集 $co(\Omega)$ 的最低点 $w_{i_0}(t_{i_0}, d_{i_0})$ 的坐标代入该方程, 得到

$$f(\varphi) = p_1 \cos(\varphi) - p_2 \sin(\varphi) + p_3 \cos(\varphi) (t_{i_0} tg(\varphi) - d_{i_0}) \quad (16)$$

对其求一阶倒数, 有

$$\begin{aligned} f'(\varphi) &= -p_1 \sin(\varphi) - p_2 \cos(\varphi) + p_3 t_{i_0} \cos(\varphi) + p_3 d_{i_0} \sin(\varphi) \\ &= (p_3 d_{i_0} - p_1) \sin(\varphi) + (p_3 t_{i_0} - p_2) \cos(\varphi) \end{aligned} \quad (17)$$

在凸集 $co(\Omega)$ 的最低点 $w_{i_0}(t_{i_0}, d_{i_0})$ 处, 必然存在 $f'(\varphi) = 0$ 。则有

$$tg(\varphi) = \frac{p_2 - p_3 t_{i_0}}{p_3 d_{i_0} - p_1} \quad (18)$$

此时, 最优化目标 obj_1 取得最小值。

因此, 利用凸集理论对优化目标 obj_1 和 obj_2 进行优化计算, 得到了通过所求直线 L 的点 $w_{i_0}(t_{i_0}, d_{i_0})$ 和直线 L 的斜率 $tg(\varphi)$, 可以唯一确定表示时钟时滞的直线 L 。

总结时钟同步算法的计算步骤如下:

- (1) 利用凸集理论得到包含所有离散测量点的凸集 $co(\Omega)$ 。
- (2) 根据最小面积优化目标进行推导, 直接利用式(13)求得凸集

最低点的横坐标 t_{i_0} , 并通过测量点 $w_{i_0}(t_{i_0}, d_{i_0})$, 直接得到凸集最低点的纵坐标 d_{i_0} 。因此, 得到存在于直线 L 上的一个点的坐标 (t_{i_0}, d_{i_0}) 。

(3) 根据法向距离之和为最小的优化目标推导, 直接利用式(18)求得直线 L 的斜率 $tg(\varphi)$ 。则根据直线 L 上的点 (t_{i_0}, d_{i_0}) 和斜率 $tg(\varphi)$ 唯一确定时钟时滞直线 L 。

(4) 在每个测量点, 用测量时延 d_i 减去时钟时滞直线 L 上对应的时滞 \bar{d}_i , 就得到真实时延 τ_i , $\tau_i = d_i - \bar{d}_i$ 。达到时钟同步的目的。

3 时钟同步算法分析

以下利用算法评价标准^[1,3], 分析本文提出的算法的可行性和优越性:

(1) 计算复杂度: 评价一个算法的计算复杂度, 通常利用算法的时间复杂度来表示, 本文提出的算法的复杂度是线性阶 $O(N)$, 计算复杂度低。一个算法花费的时间与算法中语句的执行次数成正比。本文提出的时钟同步算法步骤(1)需要执行 n 次, 其它步骤各需要执行一次, 因此总的语句执行次数为 $T(n) = n + 3$ 次。而传统的优化算法语句执行次数较多, 如线性规划算法需要执行次数为 $T(n) = n^2/2 + n/2 + 2$ 次。因此, 本文提出的非传统的优化方法计算复杂度低, 执行所消耗的时间较少。并且, 随着网络数据的增多, 这种计算速度快的优势更加明显。

(2) 消除时钟时滞后的时延非负性: 因为网络时延是非负的, 因此要保证在消除时钟时滞后, 真实时延的非负性。本文在利用凸集理论推导时钟同步算法的同时, 将其作为约束条件。因此, 该算法充分考虑和保证了时延的非负性。

(3) 算法鲁棒性: 即通过算法估计出来的时钟时滞的误差与时钟时滞的值大小无关。本文提出的时钟同步算法估计时滞只与网络测量时延的变化有关, 与时钟时滞的大小无关。因此, 完全能保证算法的鲁棒性。

4 算例

依据本文的法向距离最小的优化目标和相应的时钟同步算法, 估算出来的时钟时滞直线 L 如图 4 所示。时滞直线低于测量时延, 说明本文的优化目标和相应的算法保证了真实时延的非负性。时滞直线非常接近测量时延, 说明本算法计算精确高。

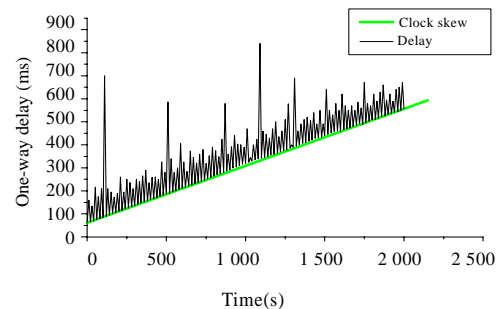


图 4 时钟同步算法计算结果

5 结束语

本文综合考虑了法向距离最小的优化目标及面积最小化目标, 利用凸集理论, 得到了时钟同步优化算法。本算法的主要优点在于计算和编程简单, 程序运行较快, 可以及时处理大量的数据。因此, 比较适合于计算机的在线运行。本文的优化算法是在网络性能测量过程中, 不考虑时钟调整情况下推导得到的。关于时钟调整, 本算法还需要作进一步的研究

究和改进。

(下转第 48 页)