Brief Announcement: A Tight Lower Bound for Clock Synchronization in Odd-Ary M-Toroids

Reginald Frank[1]
Texas A&M University, College Station, TX, USA
reginaldfrank77@tamu.edu
https://orcid.org/0000-0002-0423-1071

Jennifer L. Welch[2]
Texas A&M University, College Station, TX, USA
welch@cse.tamu.edu

Abstract

In this paper we show a tight closed-form expression for the optimal clock synchronization in k-ary m-cubes with wraparound, where k is odd. This is done by proving a lower bound of $\frac{1}{4}um\left(k-\frac{1}{k}\right)$, where k is the (odd) number of processes in each of the m dimensions, and u is the uncertainty in delay on every link. Our lower bound matches the previously known upper bound.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases Clock synchronization, Lower bound, k-ary m-toroid

Digital Object Identifier 10.4230/LIPIcs.DISC.2018.47

Related Version Find the full paper at `https://arxiv.org/abs/1807.05139`.

1 Introduction

Synchronizing clocks in a distributed system in which processes communicate through messages with uncertain delays is subject to inherent errors. A body of work has sought bounds on how closely the clocks can be synchronized when there is no drift in the hardware clocks and there are no failures. Lundelius and Lynch [5] showed that, in an n-process clique with the same uncertainty u on every link, the best synchronization possible is $u\left(1-\frac{1}{n}\right)$. Subsequently, Halpern et al. [4] considered arbitrary topologies in which each link may have a different uncertainty and showed that the optimal clock synchronization is the solution of an optimization problem. This work was generalized by [1, 6] in which algorithms were given for finding the optimal clock synchronization in any given execution. In contrast to the more general lower bounds of [4, 1, 6], Biaz and Welch [3] gave a collection of *closed-form* upper and lower bounds on the optimal clock synchronization in the worst case for k-ary m-cubes (m-dimensional hypercubes with k processes in every dimension), both with and without wraparound, in which every link has the same uncertainty, u. When there is no wraparound, the tight bound is $\frac{1}{2}um\left(k-1\right)$. When there is wraparound and k is even, the tight bound is $\frac{1}{4}umk$. However, when there is wraparound and k is odd, there is a gap between the upper bound of $\frac{1}{4}um\left(k-\frac{1}{k}\right)$ and the lower bound of $\frac{1}{4}um\left(k-1\right)$.

[1] Supported in part by CRA-W and CDC's DREU program and NSF grant CNS-0540631.
[2] Supported in part by NSF grant 1526725.

In this paper, we consider k-ary m-cubes with wraparound ("m-toroids") and odd k. We show a lower bound of $\frac{1}{4}um\left(k-\frac{1}{k}\right)$, which matches the previously known upper bound. We use the same shifting technique from previous lower bounds for clock synchronization (e.g., [5, 4, 3]). The key insight in our improved lower bound is to exploit the fact that the graph is a collection of rings in each dimension and to use multiple shifted executions instead of one.

2 Preliminaries

We first present our model and problem statement (following [5, 2, 3]). We consider a graph of k^m processes, where $k \geq 3$ is odd and $m \geq 1$, in which each process id is a tuple $\langle p_0, p_1, ..., p_{m-1}\rangle$ where each $p_i \in \{0, 1, ..., k-1\}$. There are links in both directions between any two processes $\vec{p}$ and $\vec{q}$ if and only if their ids differ in exactly one component, say the i-th, such that $p_i = q_i + 1$ (addition on process id components is modulo k throughout). Each process $\vec{p}$ has a *hardware clock* modeled as a function $H_{\vec{p}}$ from reals (real time) to reals (clock time). We assume there is no drift, so $H_{\vec{p}}(t) = t + c_{\vec{p}}$ for some constant $c_{\vec{p}}$. Each *process* is modeled as a state machine whose transition function takes as input the current state, current value of the hardware clock, and current *event* (receipt of a message or some internal occurrence), and produces a new state and a message to send over each incident link.

A *history* of process $\vec{p}$ is a sequence of alternating states and pairs of the form (event, hardware clock value), beginning with $\vec{p}$'s initial state. Each state must follow correctly from the previous one according to $\vec{p}$'s transition function and the hardware clock values must increase. A *timed history* of $\vec{p}$ is a history together with an assignment of a real time t to each pair (e, T) in the history such that $H_{\vec{p}}(t) = T$. An *execution* is a set of k^m timed histories, one per process, with a bijection for each link between the set of messages sent over the link and the set of messages received over the link. The *delay* of a message is the difference between the real time when it is received and the real time when it is sent. An execution is *admissible* if every message has delay in $[0, u]$ where u is a fixed value called the *uniform uncertainty*.

We assume each process $\vec{p}$ has a local variable $adj_{\vec{p}}$ as part of its state and we define its *adjusted clock* $A_{\vec{p}}(t)$ to be equal to $H_{\vec{p}}(t) + adj_{\vec{p}}(t)$. An execution has *terminated* once all processes have stopped changing their *adj* variables. We say the algorithm *achieves* ϵ-*synchronized clocks* if every admissible execution eventually terminates with $|A_{\vec{p}}(t) - A_{\vec{q}}(t)| \leq \epsilon$ for all processes $\vec{p}$ and $\vec{q}$ and all times t after termination.

"Shifting" an execution changes the real times at which events occur [5]. Let $\mathbf{x}$ be an m-dimensional matrix of real numbers with k elements in each dimension, which we call a *shift matrix*; elements of $\mathbf{x}$ are indexed by process ids. Define *shift*(α,$\mathbf{x}$) be the result of adding $x_{\vec{p}}$ to the real time associated with each event in $\vec{p}$'s timed history in α. Shifting changes the hardware clocks and message delays as follows [5, 2]:

▶ **Lemma 1.** *Let α be an execution with hardware clocks $H_{\vec{p}}$ and let $\mathbf{x}$ be a shift matrix. Then* shift(α,$\mathbf{x}$) *is a (not necessarily admissible) execution in which*

(a) the hardware clock of each $\vec{p}$, denoted $H'_{\vec{p}}(t)$, equals $H_{\vec{p}}(t) - x_{\vec{p}}$ and

(b) every message from $\vec{p}$ to $\vec{q}$ has delay $\delta - x_{\vec{p}} + x_{\vec{q}}$, where δ is the message's delay in α.

3 Lower Bound

▶ **Theorem 2.** *For any algorithm that achieves ϵ-synchronized clocks in a k-ary m-toroid with uniform uncertainty u, where k is odd, it must be that $\epsilon \geq \frac{1}{4}um\left(k-\frac{1}{k}\right)$.*

The complete proof and an example for the $k = 5$ case are in the full paper.

Proof sketch. Let $\mathcal{A}$ be any algorithm that achieves ϵ-synchronized clocks in a k-ary m-toroid with uniform uncertainty u, where $k = 2r + 1$ for some integer $r \geq 1$. Let α be the admissible execution of $\mathcal{A}$ in which $H_{\vec{p}}(t) = t$ for each process $\vec{p}$, every message from $\vec{p}$ to $\vec{q}$, where $\vec{q}$ is $\vec{p}$'s neighbor in the h-th dimension such that $q_h = p_h + 1$, has the same fixed delay $\delta_{\vec{p},\vec{q}}$, which is 0 if $0 \leq p_h < r$ and is u if $r \leq p_h < k$, and every message from $\vec{q}$ to $\vec{p}$ has the same fixed delay $\delta_{\vec{q},\vec{p}} = u - \delta_{\vec{p},\vec{q}}$.

For $0 \leq i < k$, define $\alpha^i = \mathit{shift}(\alpha,\mathbf{x}^i)$, where the $\vec{p}$-th element of the shift matrix $\mathbf{x}^i$, denoted $x^i_{\vec{p}}$, is defined as $\sum_{j=0}^{m-1} \mathbf{W}^i_{p_j}$, where $\mathbf{W}$ is defined as follows:

range of $i \in \{0,\ldots,m-1\}$			
$0 \leq i < r$		$r \leq i < k$	
range of p_j	$\mathbf{W}^i_{p_j}$	range of p_j	$\mathbf{W}^i_{p_j}$
$0 \leq p_j \leq i$	0	$0 \leq p_j \leq i - r$	$p_j u$
$i < p_j \leq r$	$(p_j - i)u$	$i - r < p_j \leq r$	$(i - r)u$
$r < p_j \leq r + i + 1$	$(r - i)u$	$r < p_j \leq i$	$(i - p_j)u$
$r + i + 1 < p_j \leq 2r$	$(2r - p_j + 1)u$	$i < p_j \leq 2r$	0

The idea behind the shift amounts in $\mathbf{W}$ is to cause two processes that are farthest apart in the graph to be shifted as far apart in real time as possible – thus achieving a large skew between their adjusted clocks – while maintaining valid message delays between all neighbors. By considering multiple shifted executions, we can cancel out terms involving adjusted clocks, leaving behind only terms that involve the system parameters ϵ and u, and the graph parameters k and m.

In the full paper we show that all shifted executions are admissible, i.e., that all message delays are in $[0, u]$:

▶ **Lemma 3.** *For all i, $0 \leq i < k$, α^i is admissible.*

Fix any i with $0 \leq i < r$. We focus on two processes that are maximally far away from each other. Since α^i is admissible by Lemma 3, $\mathcal{A}$ must ensure that $A^i_{\langle i,\ldots,i\rangle} - A^i_{\langle i+r+1,\ldots,i+r+1\rangle} \leq \epsilon$, where $A^i_{\vec{p}}$ denotes the adjusted clock of process $\vec{p}$ after termination in α^i. By definition of α^i and Lemma 1(a), $A^i_{\langle i,\ldots,i\rangle} = A_{\langle i,\ldots,i\rangle}$ and $A^i_{\langle i+r+1,\ldots,i+r+1\rangle} = A_{\langle i+r+1,\ldots,i+r+1\rangle} - m(r - i)u$. Thus by substituting we get $A_{\langle i,\ldots,i\rangle} - A_{\langle i+r+1,\ldots,i+r+1\rangle} \leq -m(r - i)u + \epsilon$, for $0 \leq i < r$. Similarly, we can show $A_{\langle i,\ldots,i\rangle} - A_{\langle i-r,\ldots,i-r\rangle} \leq -m(i - r)u + \epsilon$, for $r \leq i < k$.

Adding together these k inequalities and simplifying gives $\epsilon \geq \frac{1}{4}um\left(k - \frac{1}{k}\right)$. ◀

References

1 Hagit Attiya, Amir Herzberg, and Sergio Rajsbaum. Optimal clock synchronization under different delay assumptions. *SIAM J. Comput.*, 25(2):369–389, 1996.

2 Hagit Attiya and Jennifer L. Welch. *Distributed Computing: Fundamentals, Simulations, and Advanced Topics, Second Edition.* John Wiley & Sons, Hoboken, NJ, 2004.

3 Saad Biaz and Jennifer L. Welch. Closed form bounds for clock synchronization under simple uncertainty assumptions. *Inf. Process. Lett.*, 80(3):151–157, 2001.

4 Joseph Y. Halpern, Nimrod Megiddo, and Ashfaq A. Munshi. Optimal precision in the presence of uncertainty. *J. Complexity*, 1(2):170–196, 1985.

5 Jennifer Lundelius and Nancy Lynch. An upper and lower bound for clock synchronization. *Inform. Control*, 62(2/3):190–204, 1984.

6 Boaz Patt-Shamir and Sergio Rajsbaum. A theory of clock synchronization (extended abstract). In *Proc. 26th Annual ACM Symp. Theory of Comput.*, pages 810–819, 1994.