Trade-off Results for Connection Management

Marios Mavronicolas
University of Cyprus, Nicosia, Cyprus

Nikos Papadakis
University of Crete, Heraklion, Greece

(December 15, 2000)

A preliminary version of this work appears in the *Proceedings of the 11th International Symposium on Fundamentals of Computation Theory,* B. S. Chlebus and L. Czaja, eds., pp. 340–351, Lecture Notes in Computer Science, Vol. 1279, Springer-Verlag, Krakow, Poland, September 1997.

This work has been supported in part by funds for the promotion of research at University of Cyprus, and by AT&T/NCR research funds.

Authors' addresses: M. Mavronicolas, Department of Computer Science, University of Cyprus, Nicosia, CY-1678, Cyprus; N. Papadakis, Department of Computer Science, University of Crete, Heraklion 711 10, Greece.

Abstract. A *connection management* protocol establishes and handles a connection between two hosts across a wide-area network to allow reliable message delivery. We continue previous work of Kleinberg *et al.* (*Proceedings of the 3rd Israel Symposium on the Theory of Computing and Systems,* pp. 258–267, January 1995) to study the precise impact of the level of *synchrony* provided by the processors' clocks on the performance of connection management protocols, under common assumptions on the pattern of failures of the network and the host nodes.

Two basic timing models are assumed: clocks that exhibit certain kind of a *drift* from the rate of real time, and clocks that display a pattern of *synchronization* to real time. We consider networks that can duplicate and reorder messages, and nodes that can crash. We are interested in simultaneously optimizing the following performance parameters: the *message delivery time,* which is the time required to deliver a message, and the *quiescence time,* which is the time that elapses between successive periods of *quiescence,* at which the receiving host deletes all earlier connection records and returns to an initial state.

We establish natural trade-offs between message delivery time and quiescence time, in the form of tight lower and upper bounds, for each combination of the timing models and failure types. Several of our trade-off results significantly improve upon or extend previous ones shown by Kleinberg *et al.*

Key words: distributed computation, communication networks, connection management, protocols, lower bounds, trade-offs, message delivery time, quiescence time, synchrony.

Contents

1 Introduction

1.1 Motivation-Overview

Transport layer protocols [19, Chapter 6], such as the TCP/IP Internet Suite (see, e.g. [21] or [17, Chapter 3]), provide a reliable connection between two remote hosts, a *sender* and a *receiver,* across a wide-area network. The sender wishes to establish a connection to the receiver, transmit information, and later release the connection. A *connection management protocol* coordinates the establishment and release of the connection. In turn, protocols built over the transport layer provide the ground for ftp, telnet, remote procedure calls, and a number of other useful communication primitives that rely on reliable connections.

In a large network, each sender typically maintains a number of parallel sessions. Moreover, there can be a sufficient number of different *incarnations* of any session with a single receiver; in each incarnation, the connection is opened, closed and opened again. In the presence of network failures, even as benign as message reordering and duplication, it is necessary to maintain records at each receiver keeping track of which packets have been received, acted on, and so forth. Based on its own local records, the receiver must *deliver* each individual message from the sender once and never twice, even if it receives multiple packets that are duplicates of the message; the *message delivery time* is the time required to deliver a message. As the number of parallel sessions increases, however, memory limitations do not allow processing nodes to keep history records for very long. So, the receiver must periodically *quiesce* by deleting past connection records and returning to an initial state; the *quiescence time* is the time that elapses between periods of quiescence.

Message delivery time determines the latency of data transmission; thus, for applications with short incarnations, such as remote procedure calls, it is particularly desirable to keep message delivery time as small as possible. On the other hand, the amount of information that needs to be stored at each node is proportional to quiescence time; so, for applications involving steady stream-like traffic with stringent requirements on transmission rate, it is even necessary to keep quiescence time as small as possible, so that available buffer space at each processing node does not run over. Naturally, a large number of protocols have been proposed in the practical literature to minimize either message delivery time or quiescence time [5, 10, 18, 20, 24, 25]. In short, all of these protocols rely on using some combination of timers, synchronized clocks, packet delay bounds, and unique incarantion identifiers; these protocols have attracted much attention in the literature on the verification of communication protocols.[1] On the one extreme, timer-based protocols (see, e.g., [10]) achieve small message delivery time; on the other extreme, the *three-packet handshake* protocol (see, e.g., [2, 4, 5, 20]) guarantees small quiescence time.

Timer-based protocols require knowledge of the *maximum packet lifetime* μ; roughly speaking, μ is the largest amount of time a duplicate of any message may survive in the network before reaching the receiver.[2] The receiver can deliver immediately if it is prepared to maintain a record for an amount of time equal to the maximum packet lifetime; in this way, the receiver is certain that a duplicate will not arrive after the record is deleted. The catch, however, is that μ can, in general, be quite large, while duplicates may, in fact, survive for significantly shorter than μ in "normal" executions. On the opposite extreme, the three-packet handshake protocol imposes no overhead in terms of clocks or connection records. Instead, each processing host uses a source of *unique identifiers*: upon request from the host, the source yields an identifier that has not been generated before. Each message is handled in a "three-way handshake" fashion, which, roughly speaking, has as follows. First, the sender sends a unique identifier x to the receiver; in response, the receiver generates a unique identifier y and replies with $\langle x, y \rangle$. Finally, the sender sends the message together with y, and the receiver delivers the message, being sure it is not delivering a duplicate.[3] Unfortunately, however, the three-packet handshake protocol incurs a rather large message delivery time, since it requires three round-trips of communication between the sender and the receiver. Indeed, it has been a natural belief among practitioners that there are some sort of inherent trade-offs between message delivery time and quiescence time in connection management protocols, rendering these protocols inefficient in either one or the other of the two performance measures.

Kleinberg *et al.* [9] have been the first to establish mathematically precise trade-offs between message delivery time and quiescence time in a number of natural settings. More specifically, Kleinberg *et al.* have studied the connection management problem from the perspective of the amount of synchrony provided by the clocks of the sender and the receiver; their results indicate that the trade-offs between message delivery time and quiescence time depend in a critical and subtle way on this amount of synchrony. The trade-off results of Kleinberg *et al.* [9] have been expressed as non-trivial, simultaneous lower bounds on message delivery time and quiescence time under particular synchrony assumptions; these lower bounds have been accompanied by

[1] Such works attempt to verify known protocols for correctness; however, work on investigating the necessity of the model assumptions on which such protocols rely has been much less voluminous.

[2] We assume that all such duplicates eventually reach the receiver, so that μ is a finite quantity.

[3] For a concise and more accurate description of the three-packet handshake protocol, we refer the reader to [9, Section 3].

corresponding protocols whose performance guarantees nearly match the lower bounds.

In this paper, we continue the work of Kleinberg *et al.* [9] by further studying the effect of the behavior of the sender and receiver's clocks with respect to real time on the performance of connection management protocols; we still adopt all assumptions from [9] on the timing properties of the clocks, and on the pattern of failures of the network and the host nodes. We establish new, natural trade-offs between message delivery time and quiescence time, in the form of tight lower and upper bounds, for each combination of timing assumptions and failure types. Several of our trade-off results significantly improve upon or extend the ones shown by Kleinberg *et al.* [9].

Our lower bounds use the technique of "shifting" executions, originally introduced by Lundelius and Lynch for showing lower bounds on the precision achievable by clock synchronization algorithms [11]. Roughly speaking, this technique amounts to simultaneously "retime" events occurring at processes and "shift" their clocks by corresponding amounts, so that individual processes behave mistakenly in the resulting execution due to their inability to tell the two executions apart. We note that the "shifting" technique has been the one used for showing lower bounds by Kleinberg *et al.* [9]. Furthermore, one of our upper bounds is based on a substantial improvement of a specific "time-stamping" technique introduced by Kleinberg *et al.* [9, Section 5].

1.2 Failure Types, Timing Models and Timing Parameters

Throughout, we focus on *network failures,* which allow duplication and reordering of messages. We also consider *node failures,* where the receiver, but not the sender, may fail by crashing.[4] Both network and node failures have been considered by Kleinberg *et al.* [9].

We consider two basic timing models. In the *drifting clocks* model, each of the sender and receiver's clocks runs at a rate that may vary with time but always remains within a factor of $1/\rho$ and ρ to the rate of real time, for some fixed (and known) constant $\rho \geq 1$, called *drift.* In the *approximately synchronized clocks* model, each of the clocks is always within ε of real time, for some fixed (and known) constant $\varepsilon \geq 0$, called *precision.* Both the drifting clocks and the approximately synchronized clocks models have been studied in the preceding work of Kleinberg *et al.* [9].

[4] We assume, however, that the receiver may not maintain in stable storage the time of its last crash, since, otherwise, if it is not required to deliver any message whose initial packet was sent before this time, there is a general reduction to the case of message duplications (cf. [9, Section 6.2]).

We follow [9] to express our bounds on message delivery time and quiescence time in terms of two main timing parameters describing packet delays. The first of these parameters refers to a specific execution e of the system and is called the *maximum packet delay in execution e,* denoted d_e; that is, d_e is the supremum of the times that elapse between the sending of a message and the receipt of (a duplicate of) it in execution e. The second parameter of interest is the *maximum packet lifetime μ,* already introduced in Section 1.1; notice that μ is the maximum, over all executions e, among all d_e. While we may sometimes assume that μ is known, in contrast, neither the sender nor the receiver may know d_e *a priori* in execution e. Kleinberg *et al.* [9, Section 1] provide excellent motivation for the use of d_e in expressing bounds on message delivery time and quiescence time:[5]

> "We wish to be able to prove time bounds that hold *for every* execution of a protocol, not just in a worst-case sense. Thus, for instance, while it is correct to say that the time required before delivery by the three-packet handshake is at most 3μ, one can make the stronger statement that the time required is at most $3d_e$ in execution e. In this way, one can consider whether a given protocol has the following desirable property: in "good executions" (those with $d_e \ll \mu$), the time required is small relative to d_e."

Moreover, we introduce two additional timing parameters describing the behavior of the clocks in any specific execution of the timing models we consider. much in the same way d_e describes packet delays. For the drifting clocks model, we define the *worst drift in execution e,* denoted ρ_e, to be the maximum rate observed on any of the sender and receiver's clocks in execution e; for the approximately synchronized clocks model, the *worst precision in execution e,* denoted ε_e, is defined to be the maximum absolute deviation from real time observed on any of the sender and receiver's clocks in execution e. Clearly, $1/\rho \leq \rho_e \leq \rho$ and $0 \leq \varepsilon_e \leq \varepsilon$. It turns out that the parameters ρ_e and ε_e, together with the parameter d_e, determine the dependency of time bounds on message delivery time and quiescence time achievable in execution e on timing properties that are inherent to execution e in a more accurate way than ρ, ε and μ, respectively, do.

[5] It appears that similar motivations have recently led several researchers to study a notion of optimality per each particular execution for *clock synchronization* algorithms; this notion is stronger than the more common notion of worst-case optimality [3, 16].

1.3 Detailed Description and Relation to Previous Work

Sections 1.3.1 and 1.3.2 describe our results for the cases of network failures, and combined network and node failures, respectively.

1.3.1 Network Failures

We start with the case where there are network failures but not node failures. Our point of departure is an ingenious connection management protocol designed by Kleinberg *et al* [9, Section 5] for the approximately synchronized clocks model in the presence of network failures. Roughly speaking, this protocol relies on a conservative estimation, made by the receiver, of the maximum delay in any specific execution; the estimates are obtained through a "time-slicing" technique requiring both the sender and the receiver to use their (approximately synchronized) clocks in order to send to each other one time-stamped packet per each "time-slice". In turn, these estimates enable the receiver to determine when to deliver or quiesce.

We observe that the safety condition satisfied by this protocol, namely that it does not deliver a message twice, holds *independently* of the particular timing assumptions made for the approximately synchronized clocks model.[6] This observation makes this protocol a natural candidate of a *generic* connection management protocol which guarantees at-most-once message delivery in the presence of network failures for *any* model in which clocks are available to the sender and the receiver. Such a generic protocol would enjoy nice portability properties across models for which the available clocks satisfy different timing assumptions, while it would still run correctly for models in which the timing properties of the clocks are non-amenable to a precise formalization, or even completely *unknown*.

There is, however, an additional, natural performance requirement on a generic connection management protocol. Indeed, different applications may present different needs regarding which one between message delivery time and quiescence time to minimize while still retaining the other *bounded*; so, a connection management protocol is truly competitive in performance only if it allows such appropriate trade-offs between its message delivery time and quiescence time. Unfortunately, as we explain below, the connection management protocol of Kleinberg *et al.* [9, Section 5] fails to do so.

[6] An inspection of the proof of [9, Theorem 5] reveals that the timing assumptions in the approximately synchronized clocks model are explicitly used in the analysis of the performance of this protocol, namely in deriving upper bounds on the message delivery time and quiescence time it achieves; however, these timing assumptions are not used in its correctness proof.

For the approximately synchronized clocks model, the connection management protocol of Kleinberg *at al.* [9, Section 5] achieves upper bounds of $(1 + 2/\delta)d_e + (4 + 4/\delta)\varepsilon + c$ and $(\delta + 2)d_e + (2\delta + 6)\varepsilon + c$ on message delivery time and quiescence time, respectively, for any constant $c > 0$, where $\delta \geq 1$ is a "trade-off" parameter (cf. [9, Theorem 5]). Increasing δ lowers the upper bound on message delivery time but raises the upper bound on quiescence time; on the other hand, decreasing δ raises the upper bound on message delivery time but lowers the upper bound on quiescence time. Moreover, the upper bound on message delivery time increases as δ decreases down to 1, while still remaining bounded above by a *finite* quantity, namely $3d_e + 8\varepsilon + c$; unfortunately, the same does not hold on the way the upper bound on quiescence time increases with δ: the limit of the upper bound on quiescence time, as δ becomes large, is infinite. Thus, the connection management protocol of Kleinberg *et al.* [9, Section 5] may become non-competitive in performance for the approximately synchronized clocks model, due to unbounded increase in the amount of connection records per node, for applications requiring the latency of packet transmission to become arbitrarily small.

Call a connection management protocol *bounded* if the upper bounds it achieves on message delivery time and quiescence time are both bounded functions of any involved trade-off parameters. The work of Kleinberg *et al.* [9] leaves open the question of whether there exists or not a connection management protocol that is both generic and bounded. We resolve this question by a judicious adjustment of the timing conditions which the receiver uses to determine when to deliver or quiesce in the generic protocol of Kleinberg *et al* [9]; the result is another generic connection management protocol which is also bounded for the approximately synchronized clocks model.

We also present another generic connection management protocol that is both simple and natural. This protocol employs a timer and relies on knowledge of the maximum packet lifetime μ. The receiver delivers immediately each time it receives a new packet; it then counts off some time on its local clock before quiescing in order to make sure that the elapsed real time is no less than μ.

Drifting Clocks

We first consider the case of drifting clocks, for which we establish a trade-off lower bound result between message delivery time and quiescence time.

The three-packet handshake protocol [5, 20] still works for the drifting clocks model to achieve upper bounds of $3d_e$ on both message delivery time and quiescence time.[7] Kleinberg *et al.* [9, Section 4.2] describe a natural timer-based protocol achieving upper bounds of d_e and $\rho^2\mu + d_e$ on message delivery time and quiescence time, respectively. This protocol requires knowledge by the receiver of the maximum packet lifetime μ; moreover, the upper bound on quiescence time achieved by the protocol of Kleinberg *et al.* [9] is particularly large for systems whose maximum packet lifetime is large. However, Kleinberg *et al.* establish almost optimality of this protocol by presenting a nearly matching trade-off between message delivery time and quiescence rime that must hold for *some* execution of any connection management protocol. More specifically, Kleinberg *et al.* [9, Theorem 4] show that for any connection management protocol there exists an execution e with $d_e < \mu/3$ for which either a lower bound of $3d_e$ on message delivery time holds or a lower bound of $\rho^2(\mu - 3d_e)$ on quiescence time holds.

We establish a more precise trade-off between message delivery time and quiescence time that must still hold for *some* execution of any connection management protocol. More specifically, we show that for any fixed constant δ, $0 \leq \delta \leq 2$, either a lower bound of $(3 - \delta\rho)d_e$ on message delivery time holds, or a lower bound of $\rho^2(\mu - (3 - \delta)d_e)$ on quiescence time holds for some execution e of any arbitrary connection management protocol. Our result extends and improves upon [9, Theorem 4] in a significant way: it is a substantial refinement of [9, Theorem 4] that achieves to incorporate the trade-off parameter δ; note that [9, Theorem 4] is but the special case of our result with $\delta = 0$.

Approximately Synchronized Clocks

We next turn to the case of approximately synchronized clocks, for which we present both lower and upper bounds.

We start with lower bounds. Kleinberg *et al.* [9, Section 5] consider the special case of *perfect clocks* (i.e., approximately synchronized clocks with $\varepsilon = 0$); in particular, Kleinberg *et al.* show that a certain trade-off between message delivery time and quiescence time must hold for *some* execution of any connection management protocol in the perfect clocks model. In more detail, Kleinberg *et al.* [9, Theorem 6] show, assuming $\varepsilon = 0$, that for any connection management protocol, for any constant δ' where $0 < \delta' < 2$, there exists some execution e for which either a lower bound of $(1 + \delta')d_e$ on message delivery time holds, or a lower bound of

[7] Processors may read off *unique* time stamps from their clocks; these time-stamps may be used to implement unique identifiers, required by the three-packet handshake protocol, in cases where unique identifiers are not separately available.

$\min\{\mu, 2d_e/\delta'\}$ on quiescence time holds; notice, however, that the latter lower bound never exceeds μ. Kleinberg *et al.* remark [9, Section 5]:

> "For general $\varepsilon > 0$, we do not know how to obtain a correspondingly tight lower bound, and leave this as an open question."

We resolve this open question of Kleinberg *et al.* by presenting a corresponding trade-off result for the case of general $\varepsilon > 0$. More specifically, we show that for any fixed constant $\delta > 1$, either a lower bound of $(3-2/\delta)d_e + \varepsilon$ on message delivery time holds, or a lower bound of $(\delta/(\delta-1))d_e + \varepsilon$ on quiescence time holds for some execution e of any arbitrary connection management protocol.

For purpose of direct comparison to the trade-off result of Kleinberg *et al.* [9, Theorem 6], which holds just for the special case where $\varepsilon = 0$, set $\delta' = 2(1-1/\delta)$ where $\delta > 1$. Under this substitution, the lower bounds on message delivery time and quiescence time in their result can be expressed as $(3-2/\delta)d_e$ and $\min\{\mu, (\delta/(\delta-1))d_e\}$, respectively; these expressions are almost identical to those obtained by setting $\varepsilon = 0$ in the corresponding lower bounds we have shown. Our trade-off result implies that the timing uncertainty ε in the approximately synchronized clocks model incurs an additive overhead proportional to ε on each of the message delivery time and the quiescence time.

Our trade-off result improves upon the corresponding result of Kleinberg *et al.* [9, Theorem 6] in two significant ways. First, it extends [9, Theorem 6] to the case of general $\varepsilon \geq 0$. Second, when specialized for the case where $\varepsilon = 0$, the lower bound of $(\delta/(\delta-1))d_e$ on quiescence time improves in some cases upon the corresponding lower bound of $\min\{\mu, (\delta/(\delta-1))d_e\}$, shown in [9, Theorem 6]; this is so because $\min\{\mu, (\delta/(\delta-1))d_e\} \leq \mu$, while it can be verified that $(\delta/(\delta-1))d_e$ exceeds μ in the case where $d_e < \mu$ if δ is chosen so that $\delta < \mu/(\mu - d_e)$.

We continue with upper bounds. We use the timing assumptions made for the approximately synchronized clocks model to carry out a careful timing analysis of our generic connection management protocol. This analysis reveals upper bounds on message delivery time and quiescence time which not only still incorporate the trade-off parameter δ, but also improve substantially upon the corresponding upper bounds achieved by the corresponding protocol in [9, Theorem 5]. More specifically, we show upper bounds of $(3-1/\delta)d_e + (4-1/\delta)2\varepsilon + c$ and $(3+1/\delta)d_e + (4+1/\delta)2\varepsilon + c$ on message delivery time and quiescence time, respectively, for any constant $c > 0$; $\delta \geq 1$ is a "trade-off" parameter. We remark that each of these upper bounds converges to the finite quantity $3d_e + 8\varepsilon + c$ as δ approaches infinity; this implies that our generic connection management protocol is bounded for the case of the approximately synchronized clocks model. In contrast, the generic connection management protocol of Kleinberg *et al.* [9, Section 5] achieves upper bounds of $(1+2/\delta)d_e+(4+4/\delta)\varepsilon+c$ and $(\delta+2)d_e+(2\delta+6)\varepsilon+c$ on message delivery time and quiescence time, respectively; these bounds imply that the (generic) protocol of Kleinberg *et al.* is not bounded for the approximately synchronized clocks model.

We finally argue that the timer-based protocol described before achieves upper bounds of d_e and $\mu+4\epsilon$ on message delivery time and quiescence time, respectively, when specialized to the approximately synchronized clocks model.

1.3.2 Network and Node Failures

We next turn to the case where there are both network and node failures.

Drifting Clocks

We first consider the case of drifting clocks, for which we show a lower bound on message delivery time.

We establish a lower bound on message delivery time that must hold for *some* execution of any connection management protocol. More specifically, we show that for any arbitrary connection management protocol, there exists an execution e of it with $d_e < \mu/(3\rho+1)$ for which a lower bound of $3\rho d_e$ holds on message delivery time. No corresponding lower bound had been shown in the preceding work of Kleinberg *et al.* [9].

Approximately Synchronized Clocks

We next turn to the case of approximately synchronized clocks, for which we show two lower bounds on message delivery time that trade-off strength and generality.

First, we show that for any connection management protocol, there exists an execution e of it such that $\varepsilon \leq d_e < \mu/3$ for which a lower bound of $d_e+2\varepsilon$ holds on message delivery time. Second, we show that a stronger assumption on the execution e suffices to allow a larger lower bound on message delivery time. More specifically, we show that, under the assumption $\varepsilon \leq d_e < (\mu-6\varepsilon)/5$, a lower bound of $3d_e+2\varepsilon$ on message delivery time holds.

Our second result improves upon the corresponding result of Kleinberg *et al.* [9, Theorem 8] in two singificant ways. First, it extends [9, Theorem 8] to the case of general $\varepsilon \geq 0$. Second, when specialized for the case where $\varepsilon = 0$, the lower bound of $3d_e$ on message delivery time holds for more executions. More specifically, we show that a lower bound $3d_e$ on message delivery time holds for some execution e with $d_e < \mu/5$, while Kleinberg *et al.* show that a lower bound $3d_e$ on message delivery time holds for some execution e with $d_e < \mu/9$.

Figures 1 (a) and (b) provide a summary of the lower and upper bounds on message delivery time and quiescence time known so far for each of the timing models we have considered, for the cases of network failures, and combined network and node failures, respectively.

1.4 Organization

The rest of the paper is organized as follows. Section 2 contains formal definitions and some preliminary facts. Part **??** deals with network failures; it consists of Sections 3, 4, and 5. In Section 3, two generic protocols are presented that solve connection management. for a general model with clocks. The drifting clocks model and the approximately synchronized clocks model are treated in Sections 4 and 5, respectively. Part **??** considers combined network and node failures; it consists of Sections 6 and 7, which treat the drifting clocks model and the approximately synchronized clocks model, respectively. We conclude, in Section 8, with a discussion of our results and some open problems.

2 Definitions and Preliminaries

Our definitions closely match corresponding ones in [9, Section 2].

The system we model consists of two nodes S (*sender*) and R (*receiver*), corresponding users U_S and U_R at the nodes, and a network connecting the two nodes. Thus, U_S and U_R are the two users at the opposite ends of a connection, while S and R are the network interfaces for U_S and U_R, respectively. The sender S wishes to transmit a single *message* to the receiver R; the receiver is required to eventually deliver the message, but never to deliver it for a second time. S and R communicate through *packets* sent along the network. Throughout, denote $\Re$ the domain of *real time*.

This section is organized as follows. Section 2.1 introduces clock types and corresponding timing models. Definitions for the formal system model appear in Section 2.2, while Section 2.3 defines the connection management problem.

Bounds	Drifting clocks	Approximately synchronized clocks
Lower	$\mathbf{D}(e) \geq (3 - \delta\rho)d_e$ or $\mathbf{Q}(e) \geq \rho^2(\mu - (3 - \delta)d_e)$, for any δ, $0 \leq \delta \leq 2$	$\mathbf{D}(e) \geq (3 - 2/\delta)d_e + \varepsilon$ or $\mathbf{Q}(e) \geq (\delta/(\delta - 1))d_e + \varepsilon$, for any $\delta > 1$, if $\varepsilon \leq d_e < (1 - 1/\delta)(\mu - \varepsilon$
	$\mathbf{D}(e) \geq 3d_e$ or $\mathbf{Q}(e) \geq \rho^2(\mu - 3d_e)$ if $d_e < \mu/3$ [9, Theorem 4]	$\mathbf{D}(e) \geq (3 - 2/\delta)d_e$ $\mathbf{Q}(e) \geq \min\{\mu, (\delta/(\delta - 1))d_e\}$, for any $\delta \geq 1$ and $\varepsilon = 0$ [9, Theorem 6]
Upper	$\mathbf{D}(e) \leq d_e$ and $\mathbf{Q}(e) \leq \rho^2\mu + d_e$ [9, Section 4.2]	$\mathbf{D}(e) < (3 - 1/\delta)d_e + (4 - 1/\delta)2\varepsilon + c$ and $\mathbf{Q}(e) < (3 + 1/\delta)d_e + (4 + 1/\delta)2\varepsilon + c$, for any $\delta \geq 1$ and $c > 0$
		$\mathbf{D}(e) \leq d_e$ and $\mathbf{Q}(e) \leq \mu + 4\varepsilon$
		$\mathbf{D}(e) < (1 + 2/\delta)d_e + (4 + 4/\delta)\varepsilon + c$ and $\mathbf{Q}(e) < (2 + \delta)d_e + (6 + 2\delta)\varepsilon + c$, for any $\delta \geq 1$ and $c > 0$ [9, Theorem 5]

(a) Network failures

Bounds	Drifting clocks	Approximately synchronized clocks
Lower	$\mathbf{D}(e) \geq 3\rho d_e$, if $d_e < \mu/(3\rho + 1)$	$\mathbf{D}(e) \geq d_e + 2\varepsilon$, if $\varepsilon \leq d_e < \mu/3$
		$\mathbf{D}(e) \geq 3d_e + 2\varepsilon$, if $\varepsilon \leq d_e < (\mu - 6\varepsilon)/5$
		$\mathbf{D}(e) \geq 3d_e$, if $d_e < \mu/9$ and $\varepsilon = 0$ [9, Theorem 8]
Upper	—	—

(b) Network and node failures

Figure 1: Summary of bounds on message delivery time and quiescence time

2.1 Clock Types and Timing Models

A *clock* is a strictly increasing (and unbounded), piece-wise continuous function of real time $\gamma : \Re \rightarrow \Re$; denote γ^{-1} the *inverse* of γ. In the *generic clocks model,* clocks γ_S and γ_R are associated with S and R, respectively.

We consider two main clock types: clocks that may "drift" away from the rate of real time, and clocks that are approximately synchronized with respect to real time.

Drifting Clocks

Fix any constant $\rho \geq 1$, called *drift.* A *ρ-drifting clock,* or *drifting clock* for short, is a clock $\gamma : \Re \rightarrow \Re$ such that for all real times $t_1, t_2 \in \Re$ with $t_1 < t_2$,

$$\frac{1}{\rho} \;\leq\; \frac{\gamma(t_2) - \gamma(t_1)}{t_2 - t_1} \;\leq\; \rho\,.$$

Roughly speaking, a ρ-drifting clock runs at a rate between $1/\rho$ and ρ times the rate of real time; note that the rate of a ρ-drifting clock may itself vary with real time.

A *non-drifting clock* is a ρ-drifting clock $\gamma : \Re \rightarrow \Re$ with $\rho = 1$. Thus, for all real times $t_1, t_2 \in \Re$, $\gamma(t_2) - \gamma(t_1) = t_2 - t_1$; in other words, a *non-drifting clock* runs at the rate of real time.

In the *drifting clocks model* [9], each of γ_S and γ_R is a drifting clock.

Approximately Synchronized Clocks

Fix any constant $\varepsilon \geq 0$, called *precision.* An *ε-synchronized clock,* or *approximately synchronized clock,* is a clock $\gamma : \Re \rightarrow \Re$ such that for each real time $t \in \Re$, $|\gamma(t) - t| \leq \varepsilon$. Roughly speaking, an ε-synchronized clock remains always within ε of real time. An immediate implication of the definition of a ε-synchronized clock is that for any real times $t_1, t_2 \in \Re$, $|(\gamma(t_2) - \gamma(t_1)) - (t_2 - t_1)| \leq 2\varepsilon$.

A *perfect clock* is an ε-synchronized clock $\gamma : \Re \rightarrow \Re$ with $\varepsilon = 0$. Thus, for each real time $t \in \Re$, $\gamma(t) = t$. Note that a perfect clock is also a non-drifting clock, but not vice versa.

The *approximately synchronized clocks model* [9] is defined by assuming that each of γ_S and γ_R is an approximately synchronized clock; in the *perfect clocks model* [9], each of γ_S and γ_R is a perfect clock.

An immediate implication of the definition of the approximately synchronized clocks model is that $|\gamma_S(t) - \gamma_R(t)| \leq 2\varepsilon$. The *weakly synchronized clocks model* is defined as a weaker variant of the approximately synchronized clocks model in which we assume that this implication holds, and also that for any real times $t_1, t_2 \in \Re$, both $|(\gamma_S(t_2) - \gamma_S(t_1)) - (t_2 - t_1)| \leq 2\varepsilon$, and $|(\gamma_R(t_2) - \gamma_R(t_1)) - (t_2 - t_1)| \leq 2\varepsilon$, while we relax the assumption that each of the individual clocks of S and R be ε-synchronized. The following is an immediate implication of the three timing conditions defining the weakly synchronized clocks model, which will be useful in our later proofs.

Lemma 2.1 *In the weakly synchronized clocks model, for any real times* $t_1, t_2 \in \Re$,

$$|\gamma_S(t_2) - \gamma_R(t_1) - (t_2 - t_1)| \quad \leq \quad 2\varepsilon\,.$$

Intuitively, Lemma 2.1 establishes how much the clocks of S and R at *different* real times may at most differ from each other in the weakly synchronized clocks model (in particular, in the approximately synchronized clocks model), as a function of the difference between these times.

2.2 System Model

Each of U_S, U_R, S and R is modeled as an automaton with a (possibly infinite) set of *states,* and a *transition function.* In general, we shall not be concerned with the structure of U_S and U_R; U_S simply provides a *message* $\mathtt{m}$ to S, which must be delivered to U_R by R; thus, it suffices to take each of U_S and U_R to be an I/O automaton [12]. In contrast, more state structure is needed for S and R; each state of S and R consists of an *internal* component, and a *clock* component; thus, we take each of S and R to be a *timed automaton* [8, 13, 14]. A *protocol* is a pair of timed automata, one for each of S and R.

Initially, the internal components of the states of S and R are equal to initial values $q_{0,S}$ and $q_{0,R}$, respectively; no local action is enabled in an initial state. The clock components of S and R, also called their *local times,* are their clocks γ_S and γ_R, repsectively; neither S nor R can modify its clock. No access to real time is provided to S and R; instead, each of S and R obtains its only information about time from its clock and from messages it exchanges. The local times of S and R will be sometimes called S*-time* and R*-time,* respectively. An S*-interval* (resp., R*-interval*) is an interval of S-times (resp., R-times).

We list the *events* that can occur at each of S and R, together with an informal explanation.

-- *Packet-send events*– $\mathsf{send}(\pi, S)$ and $\mathsf{send}(\pi, R)$, for all packets π: S (resp., R) sends packet π to R (resp., S).

-- *Packet-receive events*– $\mathsf{receive}(\pi, S)$ and $\mathsf{receive}(\pi, R)$, for all packets π: S (resp., R) receives packet π from R (resp., S).

-- *Timer-set events*– $\mathsf{timerset}(\tau, S)$ and $\mathsf{timerset}(\tau, R)$, for all clock times τ: S (resp., R) sets a timer to go off when its clock reads τ.

-- *Timer-expire events*– $\mathsf{timerexpire}(\tau, S)$ and $\mathsf{timerexpire}(\tau, R)$, for all clock times τ: a timer that was set for time τ on S's clock (resp., R's clock) goes off.

-- *Message-input event*– $\mathsf{input}(\mathtt{m}, R)$: U_S provides $\mathtt{m}$ to S as input;

-- *Message-deliver event*– $\mathsf{deliver}(\mathtt{m}, R)$: R delivers $\mathtt{m}$ to U_R;

-- *Quiesce event*– $\mathsf{quiesce}(R)$: R quiesces;

-- *Crash event*– $\mathsf{crash}(R)$: R crashes.

The packet-receive, timer-expire, message-input and crash events are *interrupt* events; the packet-send, timer-set, message-deliver, and quiesce events are *react* events.

Each interrupt event at S or R causes an application of its transition function, which runs from states and interrupt events to states, and sets of react events. Roughly speaking, the transition function of S (resp., R) takes as input its current state, clock time, and interrupt event, and produces a new state, a (possibly empty) set of messages to be sent to R (resp., to S), a (possibly empty) set of timers to be set for the future, and nothing else (resp., possibly a message-deliver event, or a quiesce event, or both). Formally, a *step* of S or R is a tuple $\langle q, i, q', \mathtt{R}\rangle$, where q and q' are states, i is an interrupt event, and $\mathtt{R}$ is a set of react events. Thus, a step is taken on occurrence of an interrupt event. For any step $\langle q, \mathsf{quiesce}(R), q', \mathtt{R}\rangle$ or $\langle q, \mathsf{crash}(R), q', \mathtt{R}\rangle$ of R, we assume that $q' = q_{0,R}$; thus, a quiesce or crash event causes a transition to a state whose internal component gets its initial value, while the clock component is not affected.

A *history* h of S or R is a mapping associating to each real time $t \in \Re$, a (possibly empty) finite sequence of steps so that the following hold:

1. There is only a finite number of times $t' < t$ such that the corresponding sequence of steps is nonempty (thus, the concatenation of all such sequences in real time order is also a sequence).

2. The interrupt event in the first step of a history of S is the message-input event; furthermore, there are no other message-input events in a history.

3. The old state of each subsequent step is the new state of the previous step.

4. There is at most one timer-set event in each sequence, and it is ordered after all other events in the same sequence.

5. A timer expires at S (resp., R) at clock time τ if and only if S (resp., R) has previously set a timer for clock time τ.

An *execution* is a pair of histories $\langle h_S, h_R \rangle$ for S and R, such that there exists a function ϕ, which maps each packet-receive event $\mathsf{receive}(m, S)$ to a packet-send event $\mathsf{send}(m, R)$, and each packet-receive event $\mathsf{receive}(m, R)$ to a packet-send event $\mathsf{send}(m, R)$. We model packet duplication by assuming that ϕ need not be one-to-one; that is, there may be *different* packet-receive events $\mathsf{receive}(m, S)$ (resp., $\mathsf{receive}(m, R)$) that are mapped by ϕ to the *same* message-send event $\mathsf{send}(m, R)$ (resp., $\mathsf{send}(m, S)$). However, we require that each single packet may be duplicated only a finite number of times; this is modeled by assuming that for each packet-send event $\mathsf{send}(\pi, S)$ (resp., $\mathsf{send}(\pi, R)$), there may exist only a finite number of packet-receive events $\mathsf{receive}(\pi, R)$ (resp., $\mathsf{receive}(\pi, S)$) that are mapped by ϕ to $\mathsf{send}(\pi, S)$ (resp., $\mathsf{send}(\pi, R)$).

We use the function ϕ to define the *delay* incurred by packet π in execution e as the difference of the real times of occurrences of the events $\mathsf{receive}(\pi, S)$ (resp., $\mathsf{receive}(\pi, R)$) and $\phi(\mathsf{receive}(\pi, S))$ (resp., $\phi(\mathsf{receive}(\pi, S))$) in the corresponding histories. Define d_e, the *maximum packet delay in execution e,* to be the maximum delay over all packets. The *maximum packet lifetime* μ is the maximum d_e over all executions e. For an execution e, denote $\gamma_S^{(e)}$ and $\gamma_R^{(e)}$ the clocks of S and R, respectively, in execution e; the superscript will be omitted when the execution is clear from context.

A cornerstone of our lower bound proofs is the notion of equivalent executions with respect to either S or R (or both). Roughly speaking, two executions are equivalent with respect to S (resp., R) if they are indistinguishable to S (resp., R); however, an outside observer who has access to the real time can tell them apart. To formalize this notion, define the *view* of S (resp., R) in history h_S (resp., h_R) to be the concatenation of the sequence of steps in h_S (resp., h_R) in real-time order. (Note that the view includes the clock times.) The real times of occurrence of events are not represented in the view. The *view of S in execution e* (resp., *view of R in execution e*), denoted $e \mid S$ (resp., $e \mid R$) is the view of S (resp., R) in h_S (resp., h_R). Two executions e and e' are *equivalent with respect to S* (resp., *equivalent with respect to*

R) if $e \mid S = e' \mid S$ (resp., $e \mid R = e' \mid R$). Two executions e and e' are *equivalent* if they are both equivalent with respect to S and equivalent with respect to R.

Define the *view* of S (resp., R) in history h_S (resp., h_R) for some S-interval (resp., R-interval) to be the concatenation of the sequence of steps in h_S (resp., h_R) in real-time order for which the S-time (resp., R-time) is in the S-interval (resp., R-interval); note that the view of S (resp., R) in history h_S (resp., h_R) for some S-interval (resp., R-interval) is a (possibly empty) subsequence of the view of S (resp., R) in history h_S (resp., h_R). The *view of S in execution e for some S-interval I_S* (resp., *view of R in execution e for some R-interval I_R*), denoted $e(I_S) \mid S$ (resp., $e(I_R) \mid R$) is the view of S (resp., R) in h_S (resp., h_R) for the S-interval I_S (resp., R-interval I_R). Two executions e and e' are *equivalent with respect to S for the S-interval I_S* (resp., *equivalent with respect to R for the R-interval I_R*), denoted $e \overset{I_S}{\equiv} e'$ (resp., $e \overset{I_S}{\equiv} e'$) if $e(I_S) \mid S = e'(I_S) \mid S$ (resp., $e(I_R) \mid R = e'(I_R) \mid R$).

2.3 Connection Management Protocols

A protocol $\mathcal{P}$ *solves connection management* if it satisfies the following condition. For every execution e of $\mathcal{P}$, there is exactly one $\mathsf{deliver}(\mathtt{m}, R)$ event followed by exactly one $\mathsf{quiesce}(R)$ event. Assume that these events occur at real times $\mathbf{D}(e)$ and $\mathbf{Q}(e)$, respectively. A *connection management protocol* is a protocol that solves connection management.

A *trade-off* connection management protocol $\mathcal{P}$ is a connection management protocol for which there exists a parameter $\delta \geq 0$ such that for any timed execution e of $\mathcal{P}$ both $\mathbf{D}(e)$ and $\mathbf{Q}(e)$ are bounded above by (non-constant) functions of δ, one of which is an ascending function of δ and the other is a descending function of δ. A *bounded* connection management protocol is a trade-off connection management protocol for which the one of the functions bounding $\mathbf{D}(e)$ and $\mathbf{Q}(e)$ that is an ascending function of δ converges to a finite upper bound as δ approaches infinity.

In all of our lower bound proofs, we will construct sequences of executions in the last of which a message is delivered twice. We will illustrate these executions using appropriate execution diagrams; in these diagrams, events will be depicted using conventions summarized in Figure 2.

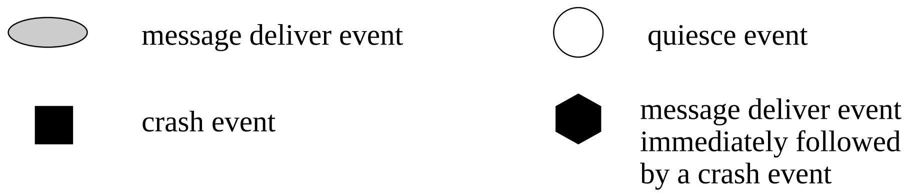

Figure 2: Conventions for events

3 Generic Protocols

In this section, we present two (generic) protocols that solve connection management in the generic clocks model.

3.1 A Protocol Based on Time Stamps

We present a generic protocol $\mathcal{P}_1$ that employs time stamps. Section 3.1.1 describes $\mathcal{P}_1$ and shows certain preliminary properties of it; the correctness of $\mathcal{P}_1$ is established in Section 3.1.2.

3.1.1 Description and Preliminaries

Throughout, fix any constant $c > 0$, and let δ be any real parameter such that $\delta \geq 1$. Define c' to be a function of c and δ,

$$c' = \frac{\delta c}{7\delta + 2} .$$

Notice that c' converges to the finite quantity $c/7$ as the parameter δ becomes arbitrarily large.

For any real time $t \in \Re$, say that $\gamma_S(t)$ (resp., $\gamma_R(t)$) is a *discrete S-time* (resp., *discrete R-time*), if it is a positive integral multiple of c'. For each integer $l \geq 1$, the *lth discrete S-time* is the discrete S-time lc'; the *lth discrete R-time* is defined in a corresponding way.

The protocol $\mathcal{P}_1$ is the "parallel composition" of a "sub-protocol" $\mathcal{P}_1^{ts}$ that generates and handles timestamps, and a "sub-protocol" $\mathcal{P}_1^{dq}$ that uses timestamps in order to infer when to deliver and quiesce. The "sub-protocol" $\mathcal{P}_1^{ts}$ is identical to the corresponding "sub-protocol" of the generic protocol proposed by Kleinberg *et al.* [9, Section 5]; however, for the sake of completeness, we repeat in this paper its description and proof of correctness at a somewhat higher level of formalism. The "sub-protocol" $\mathcal{P}_1^{dq}$ builds upon the corresponding "sub-protocol" of the generic protocol proposed by Kleinberg *et al.* [9, Section 5].

The Protocol $\mathcal{P}_1^{ts}$

For each integer $l \geq 0$, S sends a packet to R at the lth discrete S-time. Assume that r_0 is the smallest integer such that R has received a packet from S by the r_0th discrete R-time; for each integer $l \geq r_0$, R sends a packet to S at the lth discrete R-time.

Define *threshold functions* $\mathtt{Th}_S : \mathcal{N} \rightarrow \mathcal{N} \cup \{\perp\}$ and $\mathtt{Th}_R : \mathcal{N} \rightarrow \mathcal{N} \cup \{\perp\}$ as follows. For each integer $l \geq 0$, $\mathtt{Th}_S(l) \neq \perp$ if and only if there exists some integer $s \geq 0$ such that:

- for each integer $s' \leq s$, S has received by discrete S-time lc' a packet sent by R at the s'th discrete R-time;
- no packet sent by R at the $(s+1)$th discrete R-time has been received by S by discrete S-time lc'.

In this case, $\mathtt{Th}_S(l) = s$.

We proceed to define the function $\mathtt{Th}_R$. For $l = r_0 - 1$, $\mathtt{Th}_R(r_0 - 1) = 0$. For each integer $l \geq 0$ such that $l \neq r_0 - 1$, $\mathtt{Th}_R(l) \neq \perp$ if and only if there exists some integer $r \geq 0$ such that:

- for each integer $r' \leq r$, R has received by discrete R-time lc' a packet sent by S at the r'th discrete S-time;
- no packet sent by S at the $(r+1)$th discrete S-time has been received by R by discrete R-time lc'.

In this case, $\mathtt{Th}_R(l) = r$.

The content of each packet sent by S to R at the lth discrete S-time is a function of l. For $l = 0$, S sends $\langle 0, \mathtt{m} \rangle$ where the first component indicates that the packet is sent at the 0th discrete S-time. For $l > 0$, S sends $\langle l, \mathtt{Th}_S(l) \rangle$. Similarly, R sends $\langle l \rangle$ to S at the lth discrete R-time, where $l \geq r_0$.

R maintains three finite sets $\mathcal{S}_1$, $\mathcal{S}_2$ and $\mathcal{S}_3$, which are updated at each discrete R-time; denote $\mathcal{S}_1^{(l)}$, $\mathcal{S}_2^{(l)}$ and $\mathcal{S}_3^{(l)}$ the values attained by $\mathcal{S}_1$, $\mathcal{S}_2$, and $\mathcal{S}_3$ at the lth discrete R-time. Formally,

$$\begin{aligned} & \mathcal{S}_1^{(l)} \\ = \; & \{l' - \mathtt{Th}_R(l') \mid l' \leq l \text{ and } \mathtt{Th}_R(l') \neq \perp\}, \end{aligned}$$

$$\begin{aligned} & \mathcal{S}_2^{(l)} \\ = \; & \{ l - \texttt{Th}_S(l) \;\mid\; R \text{ has received } \langle l', \texttt{Th}_S(l' \rangle \text{ by the } l\text{-th discrete } R\text{-time and } \texttt{Th}_S(l') \neq \bot \} \, , \end{aligned}$$

and

$$\begin{aligned} & \mathcal{S}_2^{(l)} \\ = \; & \{ l' - r_0 \;\mid R \text{ has received } \langle l', \texttt{Th}_S(l') \rangle \text{ by the } l\text{-th discrete } R\text{-time and } \texttt{Th}_S(l') = \bot \} \, . \end{aligned}$$

R uses the sets $\mathcal{S}_1^{(l)}$, $\mathcal{S}_2^{(l)}$, and $\mathcal{S}_3^{(l)}$ to define the *maximum function* $\texttt{Mx}_R : \mathcal{N} \rightarrow \mathcal{N}$ as follows. For each integer $l \geq 0$,

$$\texttt{Mx}_R(l) \;\; = \;\; \max \mathcal{S}_1^{(l)} \cup \mathcal{S}_2^{(l)} \cup \mathcal{S}_3^{(l)} + c' \, .$$

For any execution e, denote

$$\texttt{Mx}^*_R(e) \;\; = \;\; \max_{lc' \leq \mathbf{Q}(e)} \texttt{Mx}_R(l) \, .$$

We have:

Lemma 3.1 (Kleinberg, Attiya and Lynch [9]) $\texttt{Mx}_R(r_0) > |r_0|$

Proof: By the first rule,

$$\begin{aligned} M^{(r_0 - c')} & \geq r_0 - c' - s \\ & = r_0 - c' \\ & \quad \text{(since by Claim ?? } s = 0) \, . \end{aligned}$$

It follows that:

$$\begin{aligned} l^{(r_0)} & = M^{(r_0)} + c' \\ & > M^{(r_0)} \\ & \geq M^{(r_0 - c')} \\ & \quad \text{(since } M^{(t)} \text{ is an ascending function)} \\ & \geq r_0 \, , \end{aligned}$$

so that

Claim 3.2 $l^{(r_0)} > r_0$

The first packet from S sent at S-time 0, so each S-packet has time-stamp ≥ 0. By the third rule,

$$\begin{aligned} M^{(r_0)} &\geq s' - r_0 \\ &\geq -r_0 ; \end{aligned}$$

hence

$$\begin{aligned} l^{(r_0)} &= M^{(r_0)} + c' \\ &> M^{(r_0)} \\ &\geq -r_0 , \end{aligned}$$

which implies that $l^{(r_0)} > -r_0$. By Claim 3.2, this implies that $l^{(r_0)} >| r_0 |$, as needed. ■

The Protocol $\mathcal{P}_1^{dq}$

We are now ready to present the algorithm. R delivers at the first discrete R-time t' when $t' > (3 - 1/\delta) l^{(t')}$ and quiesces at the first discrete R-time t'' when $t^{('')} > (3 + 1/\delta) l^{(t'')}$ It then sends a done message to S; S quiesces immediately upon receiving this done message. If at any time S report a non trivial threshold that is less than r_0 (i.e. one can conclude that R is hearing replays), R aborts the connection without delivering and sends an error message to S.

For any time t, define $r(t)$ to be the discrete R-time at which the maximum value for $l^{(t)}$ was attained; that is, $r(t)$ is the largest $r \leq t$ for which $l^{(r)} = l^{(t)}$. We show:

Lemma 3.3

$$d_e > \gamma_R^{-1}(r) - \gamma_S^{-1}(r - l^{(t)} + 2c')$$

Proof: Assume, without loss of generality, that $l^{(t)}$ was updated using the first rule (the other cases are similar). Consider the discrete R-time r at which the maximum value for $l^{(t)}$ was attained -i.e. the first $r \leq t$ for which $l^{(t)} = l^{(r)}$. Let s to be the threshold of R at time r. Since the lag was updated at time r using the first rule, we have that $M^{(r)} = r - s$. Also since $l^{(r)} = M^{(r)} + c'$, $l^{(t)} = M^{(t)} + c'$ and $l^{(t)} = l^{(r)}$, we have that $M^{(r)} = M^{(t)}$. Thus, $M^{(t)} = r - s$. It implies that

$$\begin{aligned} s &= r - M^{(t)} \\ &= r - (l^{(t)} - c') \\ &= r - l^{(t)} + c' . \end{aligned}$$

It immediately follows that the threshold of R at discrete time r is equal with $r - l^{(t)} + c'$. By definition of threshold, R has received all S-packet with time stamp $\leq r - l^{(t)} + c'$; thus, any S-packet sent at discrete S-time $r - l^{(t)} + 2c'$ has not yet arrived. It follows:

$$d_e \ > \ \gamma_R^{-1}(r) - \gamma_S^{-1}(r - l^{(t)} + 2c')\,,$$

as needed. ■

3.1.2 Correctness Proof

We continue to show that $\mathcal{P}_1$ is a connection management protocol. We need to prove that R does not deliver any message for a second time. First we argue that R will not quiesce until it has received an S-packet with non-trivial threshold. Let ψ denote the S-packet with minimal time-stamp that reports a non-trivial threshold, and consider discrete R-time r at which R has not yet received ψ. Let $r - u_1$ be the time-stamp of the most recent S-packets, and set $v = r - u_1 - r_0$. It follows that $r - u_1 \geq r_0$. By the first rule,

$$\begin{array}{rcl} l^{(r)} & = & M^{(r)} + c' \\ & > & r - (r - u_1) \\ & = & u_1\,, \end{array}$$

which implies $l^{(r)} > u_1$. Also

$$\begin{array}{rcl} l^{(r)} & = & M^{(r)} + c' \\ & \geq & M^{(r_0)} + c' \\ & & \text{(since } M^{(t)} \text{ is an asceding function } \ M^{(r)} \geq M^{(r_0)}) \\ & = & l^{(r_0)} \\ & > & r_0 \\ & & \text{(by Lemma 3.1)}\,, \end{array}$$

which implies $l^{(r)} > r_0$. Also

$$\begin{array}{rcl} l^{(r)} & = & M^{(r)} + c' \\ & \geq & M^{(r-u_1)} + c' \\ & & (M^{(t)} \text{ is an ascending function}) \\ & > & M^{(r-u_1)} \\ & \geq & r - u_1 - r_0\,, \end{array}$$

by the third rule, $M^{(r-u_1)} \geq r - u_1 - r_0$ (since at time $(r - u_1)$, R has not yet receive a non trivial threshold). It implies that $l^{(r)} > u$. Thus, $r = r_0 + u_1 + u < 3l^{(r)} < (3 + 1/\delta)l^{(r)}$, so R will not yet quiesce.

Now let l^* (resp. M^*) denote the maximum value of $l^{(t)}$ (resp. $M^{(t)}$) over all discrete R-times t up to quiescence, and s_1 denote the time-stamp of S-packet ψ. Indeed, the time-stamped $s_1 - c'$ reports a trivial threshold, so by the third rule for estimating the lag, $M^* \geq s_1 - c' - r_0$. It follows $l^* \geq s_1 - r_0$. Since $l^{(t)}$ is an ascending function, $l^* \geq l^{(r_0)}$. By Lemma 3.1, this implies that $l^* > r_0$; adding, we obtain:

Claim 3.4 *$s_1 < 2l^*$.*

Finally, suppose $\mathbf{T} > t''$ and a replay of the original message arives at time $\mathbf{T}$. We will show that if $\mathbf{T}' \geq \mathbf{T}$ is some time at which R has not received a replay of S-packet ψ, it is not required to deliver. Since ψ has not been received at $\mathbf{T}'$, by the first rule for estimating the lag we have

$$\begin{aligned} l^{(\mathbf{T}')} &\geq \mathbf{T}' - s_1 \\ &> \mathbf{T}' - 2l^* \\ &\quad \text{(by Claim 3.4)}. \end{aligned}$$

Since R quiesces at R-time T and l^* denote the maximum value $l^{(t)}$ over all discrete R-times t up to quiescence, by protocol we have

$$(3 + \frac{1}{\delta})\, l^* \leq \mathbf{T}\,,$$

so that

$$l^* \leq \frac{\delta}{3\delta + 1}\mathbf{T}\,.$$

Thus,

$$\begin{aligned} l^{(\mathbf{T}')} &> \mathbf{T}' - 2l^* \\ &\geq \mathbf{T}' - \frac{2\delta}{3\delta + 1}\mathbf{T} \\ &\geq \mathbf{T}' - \frac{2\delta}{3\delta + 1}\mathbf{T}' \\ &= (1 - \frac{2\delta}{3\delta + 1})\,\mathbf{T}' \\ &= \frac{\delta + 1}{3\delta + 1}\mathbf{T}'\,, \end{aligned}$$

which implies that

$$\begin{aligned}\mathbf{T}' &< \frac{3\delta+1}{\delta+1}\, l^{(\mathbf{T}')} \\ &= \frac{3(\delta+1)-2}{\delta+1}\, l^{(\mathbf{T}')} \\ &= (3-\frac{2}{\delta+1})\, l^{(\mathbf{T}')} \\ &\leq (3-\frac{1}{\delta})\, l^{(\mathbf{T}')} .\end{aligned}$$

Thus, R does not deliver at time $\mathbf{T}'$. Recall that T is the R-time which R receives a replay of original message and $\mathbf{T}' \geq \mathbf{T}$ is a R-times at which R has not receives a replay of ψ. It implies that R before delivery it receives a replay of ψ. But ψ reports a threshold $(= r_0)$ smaller than $\mathbf{T}$,which is the discrete R-time at which R first started sending packets to S following quiescence. By the protocol, R will abort the connection in this case. Thus, R never delivers the message a second time. It immediately follows:

Proposition 3.5 $\mathcal{P}_1$ *is a connection management protocol.*

3.2 A Timer-Based Protocol

In this section, we present a generic protocol $\mathcal{P}_2$ that employs a timer and relies on knowledge of the maximum packet lifetime μ.

R delivers immediately each time it receives a new packet. It then counts off on its clock so that local time a elapses, in a way that real time at least μ elapses; it then quiesces.

We show that $\mathcal{P}_2$ is a connection management protocol. Consider any packet π send by S to R at real time t. Thus, π arrives at R at real time $\mathbf{D} > t$. Then, R delivers immediately. After R counts off its clock to pass local time a so that the real time which elapses is at least μ; then, R quiesces at time $\mathbf{Q} \geq \mu + \mathbf{D} > \mu + t$. Assume that a replay of π arrives at R at time $\mathbf{T}$. Since the maximum packet lifetime is equal to μ, $\mathbf{T} \leq \mu + t$. It follows that $\mathbf{Q} > \mathbf{T}$. Thus, R never delivers twice. It immediatelly follows:

Proposition 3.6 $\mathcal{P}_2$ *is a connection management protocol.*

4 Drifting Clocks

In this section, we present our lower bounds for the drifting clocks model in the presence of network failures. We show:

Theorem 4.1 *Consider the drifting clocks model in the presence of network failures. Then, for any connection management protocol $\mathcal{P}$, for any constant δ such that $0 \leq \delta \leq 2$, there exists an execution e of $\mathcal{P}$ such that either*

$$\mathbf{D}(e) \quad \geq \quad (3 - \delta\rho)d_e\,,$$

or

$$\mathbf{Q}(e) \quad \geq \quad \rho^2(\mu - (3 - \delta)d_e)\,.$$

Proof: Assume, by way of contradiction, that there exists a connection management protocol $\mathcal{P}$ for the drifting clocks model in the presence of network failures, and a constant δ, $0 \leq \delta \leq 2$, such that for every execution e of $\mathcal{P}$, both $\mathbf{D}(e) < (3 - \delta\rho)d_e$ and $\mathbf{Q}(e) < \rho^2(\mu - (3 - \delta)d_e)$. We construct an execution of $\mathcal{P}$ containing two message-deliver events.

We start with an informal outline of our proof. We construct a sequence of executions e, e', f and f', so that R delivers the message twice in f'. e is a slow execution. f' is the "concatenation" of e' and f. In e and f, the clocks of R and S are "slow", while in e', the clocks of R and S are "fast". We start with e, which terminates immediately after R quiesces. By modifying R's clock, we "perturb" e to obtain f, which S cannot distinguish from e; in f, only delivery. We continue to construct e', which S cannot distinguish from e to S, while R still delivers in e' and quiesces. Finally, we construct f' as the "concatenation" of e' and f; in f', R first delivers and quiescences, before it receives replays of all packets in a way that R "sees" them arriving as in f. This leads R to deliver again, which contradicts the correctness of $\mathcal{P}$. We now present the details of the formal proof.

Consider an execution e of $\mathcal{P}$ for which $\gamma_S^{(e)}(t) = \gamma_R^{(e)}(t) = t/\rho$; thus, both clocks run "slow" in e and initially hold the value 0. Furthermore, assume that each packet incurs a delay of d_e in the execution e. Finally, assume that the last step in e is taken on occurrence of a quiesce event at R.

By our assumption on $\mathcal{P}$, the message-deliver and quiesce events occur in e at real times $\mathbf{D}(e) < (3 - \delta\rho)d_e$, and $\mathbf{Q}(e) < \rho^2(\mu - (3 - \delta)d_e)$, respectively; thus, these events occur at R's local times

$$\begin{aligned}
\gamma_R^{(e)}(\mathbf{D}(e)) &< \gamma_R^{(e)}((3-\delta\rho)d_e) \\
&\quad \text{(since } \mathbf{D}(e) < (3-\delta\rho)d_e \text{ and } \gamma_R^{(e)} \text{ is strictly increasing)} \\
&= \frac{(3-\delta\rho)\,d_e}{\rho} \\
&\quad \text{(by definition of } \gamma_R^{(e)}\text{)} \\
&= (\frac{3}{\rho} - \delta)\,d_e\,,
\end{aligned}$$

and

$$\begin{aligned}
\gamma_R^{(e)}(\mathbf{Q}(e)) &< \gamma_R^{(e)}(\rho^2(\mu-(3-\delta)d_e)) \\
&\quad \text{(since } \mathbf{Q}(e) < \rho^2(\mu-(3-\delta)\rho)d_e \text{ and } \gamma_R^{(e)} \text{ is strictly increasing)} \\
&= \frac{\rho^2(\mu-(3-\delta)d_e)}{\rho} \\
&\quad \text{(by definition of } \gamma_R^{(e)}\text{)} \\
&= \rho\,(\mu-(3-\delta)d_e)\,,
\end{aligned}$$

respectively.

Since all packet delays are equal to d_e in the execution e, R receives a packet from S no earlier than time d_e. Since no local actions are enabled in the initial state of R, it follows that R sends a packet to S no earlier than time d_e. Since all packet delays are equal to d_e in the execution e, it follows that S receives a packet from R no earlier than time $2d_e$. By definition of $\gamma_S^{(e)}$, this immediately implies:

Lemma 4.2 *In the execution e, S receives a packet from R no earlier than S-time $2d_e/\rho$.*

We continue to construct an execution e' of $\mathcal{P}$ as follows.

- Each step occurring at real time t in e is scheduled to occur at real time t/ρ^2 in the sequence e'; in addition, e' preserves the ordering of steps in e;
- define $\phi_{e'} = \phi_e$; thus, e' preserves the correspondence between packet-receive and packet-send events in e;
- finally, set $\gamma_S^{(e')}(t) = \gamma_R^{(e')}(t) = \rho t$; thus, both clocks run "fast" in e' and initially hold the value 0.

Note that, by definition of e, our construction implies that the last step in e' is taken on occurrence of a quiesce event at R. Moreover, we show:

Lemma 4.3 *e' is an execution of $\mathcal{P}$.*

Proof: Since e is an execution of $\mathcal{P}$, both $e \mid S$ and $e \mid R$ are histories of S and R, respectively. Consider any step occurring at real times t and t/ρ^2 in e and e', respectively. The corresponding local times at either S or R are t/ρ and $\rho\, t/\rho^2 = t/\rho$, respectively. Since these local times are equal and e is an execution of $\mathcal{P}$, it follows that both $e' \mid S$ and $e' \mid R$ are histories of S and R, respectively.

It remains to show that $d_{e'} \leq \mu$. Take any packet-send and packet-receive events π_1 and π_2 occurring at real times t_1 and t_2, respectively, in e. By definition of e, the delay of the packet in e is $t_2 - t_1 = d_e$. By construction of e', these events occur at real times t_2/ρ^2 and t_1/ρ^2, respectively, and their correspondence is preserved. Thus, the delay of the packet in e' is

$$\begin{aligned}
\frac{t_2}{\rho^2} - \frac{t_1}{\rho^2} &= \frac{t_2 - t_1}{\rho^2} \\
&\leq t_2 - t_1 && \text{(since } \rho \geq 1\text{)} \\
&= d_e && \text{(by definition of } e\text{)} \\
&\leq \mu && \text{(since } e \text{ is an execution of } \mathcal{P}\text{)},
\end{aligned}$$

as needed. ■

By construction of e' and Lemma 4.3, it immediately follows:

Lemma 4.4 *e' is an execution of $\mathcal{P}$ that is equivalent to e.*

Lemma 4.4 implies that the message-deliver and quiesce events in e' occur at R's local times less than $(3/\rho - \delta)d_e$ and $\rho(\mu - (3 - \delta)d_e)$, respectively. By definition of $\gamma_R^{(e')}$, it follows that the message-deliver and quiesce events in e' occur at real times less than $(3/\rho - \delta)d_e/\rho$ and $\mu - (3 - \delta)d_e$, respectively.

Consider now an execution f of $\mathcal{P}$ for which $\gamma_S^{(f)}(t) = t/\rho$, and $\gamma_R^{(f)}(t) = t/\rho + \rho(\mu - (3 - \delta)d_e)$; thus, both clocks are "slow", but the clock of S is initially 0, while the clock of R is initially $\rho(\mu - (3 - \delta)d_e)$. Furthermore, assume that each packet incurs a delay of d_f in the execution f. Assume that $d_f = d_e$. Finally, assume that the last step in f is taken on occurrence of a message-deliver event at R.

Since all packet delays are equal to d_f in the execution f, R receives a packet from S no earlier than time $d_f = d_e$. Since no local actions are enabled in the initial state of R, it follows that R sends a packet to S no earlier than time d_e. Since all packet delays are equal to d_e in the execution f, it follows that S receives a packet from R no earlier than time $2d_e$. By definition of $\gamma_S^{(f)}$, this immediately implies:

Lemma 4.5 *In the execution f, S receives a packet from R no earlier than S-time $2d_e/\rho$.*

We continue to show that e and f are equivalent with respect to S in an initial interval of its local time.

Lemma 4.6 $f \mid S \overset{[0,\gamma_R^{(f)}(\mathbf{D}(f))-\rho(\mu-(3-\delta)d_e)-d_e/\rho]}{\equiv} e \mid S$

Proof: By Lemmas 4.2 and 4.5, it suffices to show that

$$\gamma_R^{(f)}(\mathbf{D}(f)) - \rho(\mu-(3-\delta)d_e) - \frac{d_e}{\rho} \quad < \quad \frac{2d_e}{\rho}\,.$$

Clearly,

$$\begin{aligned}
& \gamma_R^{(f)}(\mathbf{D}(f)) - \rho(\mu-(3-\delta)d_e) - \frac{d_e}{\rho} \\
< \;& \gamma_R^{(f)}((3-\delta\rho)d_e) - \rho(\mu-(3-\delta)d_e) - \frac{d_e}{\rho} \\
& \text{(since } \mathbf{D}(f) < (3-\delta\rho)d_e \text{ and } \gamma_R^{(f)} \text{ is strictly increasing)} \\
= \;& \frac{(3-\delta\rho)\,d_e}{\rho} + \rho(\mu-(3-\delta)d_e) - \rho(\mu-(3-\delta)d_e) - \frac{d_e}{\rho} \\
& \text{(by definition of } \gamma_R^{(f)}\text{)} \\
= \;& \frac{(3-\delta\rho)\,d_e}{\rho} - \frac{d_e}{\rho} \\
= \;& (\frac{3}{\rho} - \delta)\,d_e - \frac{d_e}{\rho} \\
\leq \;& \frac{3}{\rho}\,d_e - \frac{d_e}{\rho} \\
& \text{(since } \delta \geq 0\text{)} \\
= \;& \frac{2d_e}{\rho},
\end{aligned}$$

as needed. ■

By Lemma 4.4, Lemma 4.6 immediately implies:

Corollary 4.7 $f \mid S \overset{[0,\gamma_R^{(f)}(\mathbf{D}(f))-\rho(\mu-(3-\delta)d_e)-d_e/\rho]}{\equiv} e' \mid S$

We continue to show a timing property of packet-send and packet-receive events in f.

Lemma 4.8 *Consider any packet π sent from S to R at S-time*

$$\tau \quad \in \quad [0, \gamma_R^{(f)}(\mathbf{D}(f)) - \rho(\mu - (3-\delta)d_e) - \frac{d_e}{\rho}] \, .$$

Then, π arrives at R at R-time $d_e/\rho + \tau + \rho(\mu - (3-\delta)d_e)$.

Proof: By definition of $\gamma_S^{(f)}$, π is sent at real time $\rho\tau$. By construction of f, π arrives at R at real time $\rho\tau + d_f = \rho\tau + d_e$. By definition of $\gamma_R^{(f)}$, it follows that π arrives at R at R-time $(d_e + \rho\tau)/\rho + \rho(\mu - (3-\delta)d_e) = d_e/\rho + \tau + \rho(\mu - (3-\delta)d_e)$, as needed. ∎

Finally, we construct the execution f'. Set $\gamma_S^{(f')}(t) = \gamma_R^{(f')}(t) = \rho t$; thus, both clocks run "fast" in e' and initially hold the value 0. Take $f' = e' f_1$, where the sequence of steps f_1 is defined as follows.

- Each step at R occurring at real time t in f is scheduled to occur at real time $t/\rho^2 + \mu - (3-\delta)d_e$ in f_1; in addition, the ordering of steps in f is preserved.
- Consider any packet-send event at S occurring in e' at real time $t > Q(e') - d_e/\rho^2$; a step on a corresponding packet-receive event is scheduled to occur at real time $t + \mu$ in f_1.

In Figure 3, we present the sequence f'. We show:

Lemma 4.9 *f' is an execution of $\mathcal{P}$.*

Proof: We start by defining the function $\phi_{f'}$.

- The restriction of $\phi_{f'}$ on packet-receive events in e' is equal to $\phi_{e'}$.
- Consider any packet-receive event π in f, mapped by ϕ_f to some packet-send event in f. Use the equivalence of e' and f established in Lemma 4.7 to determine the corresponding packet-send event in e' to which $\phi_{f'}$ maps π.

- Any packet-send event at S occurring in e' at real time $t > Q(e') - d_e/\rho^2$ is the image under $\phi_{f'}$ of the corresponding packet-receive event (scheduled to occur at real time $t+\mu$ in f_1).

We show:

Claim 4.10 $d_{f'} \leq \mu$

Proof: We proceed by case analysis.

1. Since the restriction of $\phi_{f'}$ on packet-receive events in e' is equal to $\phi_{e'}$, the delay of each packet in e' is at most $d_{e'} \leq \mu$, by Lemma 4.3.

2. Consider any packet-receive event π at R occurring at real time $t/\rho^2 + \mu - (3-\delta)d_e$ in f_1. By construction of f_1, there is a corresponding packet-receive event at R occurring at real time t in f. By construction of f, the corresponding packet-send event at S occurs at real time $t - d_f = t - d_e$ in f. By definition of $\gamma_S^{(f)}$, this packet-send event occurs at S-time $(t-d_e)/\rho$ in f. By Lemma 4.7, an identical packet-send event at S occurs at S-time $(t-d_e)/\rho$ in e'; by definition of $\phi_{f'}$, this is the packet-send event to which π is mapped. By definition of $\gamma_S^{e'}$, this packet-send event at S occurs at real time $(t-d_e)/\rho^2$ in e'. By construction of f', this packet-send event event at S occurs at real time $(t-d_e)/\rho^2$ in f'. Hence, the delay of π in f' is

$$\begin{aligned}
\frac{t}{\rho^2} + \mu - (3-\delta)d_e - \frac{t-d_e}{\rho^2} &= \mu - (3-\delta)d_e + \frac{d_e}{\rho^2} \\
&\leq \mu - d_e + \frac{d_e}{\rho^2} \qquad \text{(since } \delta \leq 2\text{)} \\
&\leq \mu - d_e + d_e \qquad \text{(since } \rho \geq 1\text{)} \\
&= \mu\,,
\end{aligned}$$

   as needed.

3. By construction and definition of $\phi_{f'}$, the delay of any packet-receive event at R in f_1 in correspondence to a packet-send event at S occurring in e' at real time $t > Q(e') - d_e/\rho^2$ is exactly μ.

This completes our proof. ■

Since the last step in f is taken on occurrence of a message-deliver event at R, this step is taken at real time $\mathbf{D}(f)$ in f. By construction of f_1, this step is scheduled to occur at real time $\mathbf{D}(f)/\rho^2 + \mu - (3-\delta)d_e$ in f_1. Also, by construction of f_1, any step on a packet-receive event correspondent to a packet-sent event at S in e' is scheduled to occur at real time greater than $\mathbf{Q}(e') - d_e/\rho^2 + \mu$. Clearly,

$$\begin{aligned}
& \mathbf{Q}(e') - \frac{d_e}{\rho^2} + \mu - (\frac{\mathbf{D}(f)}{\rho^2} + \mu - (3-\delta)d_e) \\
= \;& \mathbf{Q}(e') - \frac{d_e}{\rho^2} - \frac{\mathbf{D}(f)}{\rho^2} + (3-\delta)d_e \\
\geq \;& -\frac{\mathbf{D}(f)}{\rho^2} + (3-\delta)d_e && \text{(since } \mathbf{Q}(e') \geq d_{e'} = d_e/\rho^2\text{)} \\
> \;& -\frac{(3-\delta\rho)d_e}{\rho^2} + (3-\delta)d_e && \text{(since } \mathbf{D}(f) < (3-\delta\rho)d_e\text{)} \\
= \;& 3(1-\frac{1}{\rho^2})d_e - \delta(1-\frac{1}{\rho})d_e \\
\geq \;& 3(1-\frac{1}{\rho^2})d_e - 2(1-\frac{1}{\rho^2})d_e && \text{(since } \delta \leq 2 \text{ and } \rho \geq 1\text{)} \\
= \;& (1-\frac{1}{\rho^2})d_e \\
\geq \;& 0 && \text{(since } \rho \geq 1\text{)}.
\end{aligned}$$

It follows that the last step in f scheduled to occur in f_1 precedes any step occurring on a packet-receive event correspondent to a packet-sent event at S in e' that is also scheduled to occur in f_1. Consider now any of the latter steps, occurring at real time t in f. By definition of $\gamma_R^{(f)}$, this step occurs at R-time $t/\rho + \rho(\mu - (3-\delta)d_e)$ in f. By construction of f_1, this step is scheduled to occur at real time $t/\rho^2 + \mu - (3-\delta)d_e$ in f_1. By definition of $\gamma_R^{(f')}$, this step occurs at R-time $t/\rho + \rho(\mu - (3-\delta)d_e)$ in f_1. Since the local times at which this step occurs in f and f' are equal, and f is an execution of $\mathcal{P}$, it follows that f_1 is equivalent to f in the R-interval $[\rho(\mu - (3-\delta)d_e), \gamma_R^{(f)}(\mathbf{D}(f))]$. It follows that f' is an execution of $\mathcal{P}$, as needed. ■

By Lemma 4.9, f' is an execution of $\mathcal{P}$ containing two message-deliver events. A contradiction. ■

The lower bounds on message delivery time and quiescence time shown in Theorem 4.1 are simultaneously non-negative, and, hence, non-trivial, if (and only if) both $3 - \delta\rho \geq 0$ and $\mu - (3-\delta)d_e \geq 0$. Eliminating δ and assuming $\rho > 1$ yields

$$d_e \leq \frac{\rho}{\rho - 1}\frac{\mu}{3}$$

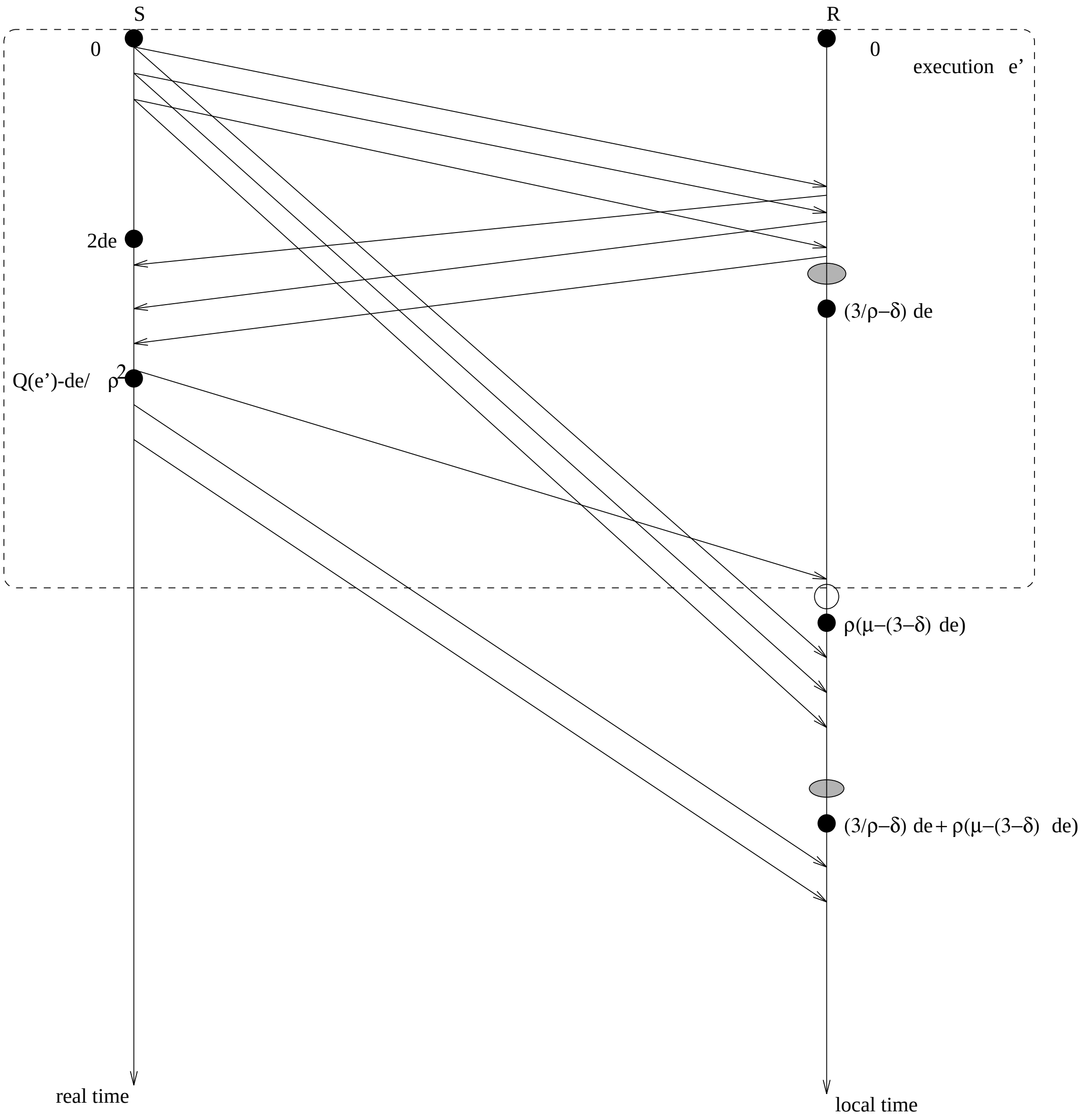

Figure 3: The execution f'

as a necessary condition for any timed execution e for which the trade-off lower bounds shown in Theorem 4.1 are non-trivial; Kleinberg *et al.* [9, Theorem 4] argue that $d_e \leq \mu/3$ is a corresponding necessary condition. Since

$$\frac{\rho}{\rho - 1}\frac{\mu}{3} \geq \frac{\mu}{3}$$

for $\rho > 1$, this implies that the trade-off lower bound shown in Theorem 4.1 is non-trivial for a wider range of executions than the trade-off lower bound shown in [9, Theorem 4].

5 Approximately Synchronized Clocks

In this section, we present our lower and upper bounds for the approximately synchronized clocks model, in the presence of network failures.

5.1 Lower Bound

We show:

Theorem 5.1 *Consider the approximately synchronized clocks model, in the presence of network failures. Then, for any connection management protocol $\mathcal{P}$, for any constant $\delta > 1$, there exists an execution e of $\mathcal{P}$ with*

$$\varepsilon \leq d_e < (1 - \frac{1}{\delta})(\mu - \varepsilon)\,,$$

such that either

$$\mathbf{D}(e) \geq (3 - \frac{2}{\delta})d_e + \varepsilon\,,$$

or

$$\mathbf{Q}(e) \geq \frac{\delta}{\delta - 1}\,d_e + \varepsilon\,.$$

Proof: Assume, by way of contradiction, that there exists a connection management protocol $\mathcal{P}$ for the approximately synchronized clocks model in the presence of network failures, and a constant $\delta > 1$, such that for every execution e of $\mathcal{P}$ with $\varepsilon \leq d_e < (1 - 1/\delta)(\mu - \varepsilon)$, both

$$\mathbf{D}(e) < (3 - \frac{2}{\delta})\,d_e + \varepsilon\,,$$

and

$$\mathbf{Q}(e) \;<\; \frac{\delta}{\delta-1}\, d_e + \varepsilon\,.$$

We construct an execution of $\mathcal{P}$ containing two message-deliver events.

We start with an informal outline of our proof. We construct a sequence of executions e, e', f, f' such that R delivers a message twice in f'. We start with execution e which terminates with R quiesces following its delivery. We "pertub" e to obtain e', which S and R cannot distinguish from e, while some message incur delay larger than corresponding in e. We continue to construct f which is indistinquishable from e to S, while R only delivers. The message incur larger corresponding one in e. Finally we construct f' as the "concatenation" of e' and f;In f', R first delivers and follows quiesces and next receives replay of all packets in such a way that R "sees" all packets arriving as in f. By construction of f, R delivers again, which constradicts the correctness of $\mathcal{P}$. We now present the details of the formal proof.

We start with a simple property of *any* execution e of $\mathcal{P}$ with

$$\varepsilon \;\leq\; d_e \;<\; (1-\frac{1}{\delta})(\mu-\varepsilon)\,.$$

Lemma 5.2 *Fix any execution e of $\mathcal{P}$ with*

$$\varepsilon \;\leq\; d_e \;<\; (1-\frac{1}{\delta})(\mu-\varepsilon)\,.$$

Then, $\mathbf{D}(e) < 3d_e$.

Proof: We proceed by case analysis on δ. Assume first that $\delta \geq 2$, so that $\delta/(\delta-1) \leq 2$. Then,

$$\begin{aligned}
\mathbf{D}(e) &\leq \mathbf{Q}(e) \\
&< \frac{\delta}{\delta-1}\, d_e + \varepsilon && \text{(by assumption on } \mathcal{P} \text{ and } \delta) \\
&\leq 2d_e + \varepsilon \\
&\leq 3d_e && (\text{since } \varepsilon \leq d_e)\,,
\end{aligned}$$

as needed. Assume now that $1 < \delta < 2$, so that $3 - 2/\delta < 2$. Then,

$$\begin{aligned}
\mathbf{D}(e) &< (3-\frac{2}{\delta})\, d_e + \varepsilon && \text{(by assumption on } \mathcal{P} \text{ and } \delta) \\
&< 2d_e + \varepsilon \\
&\leq 3d_e && (\text{since } \varepsilon < d_e)\,,
\end{aligned}$$

as needed. ■

Consider an execution e of $\mathcal{P}$ for which $\gamma_S^{(e)}(t) = t - \varepsilon$ and $\gamma_R^{(e)}(t) = t$; thus, the clock of S initially holds the value $-\varepsilon$, while the clock of R initially holds the value 0. Furthermore, assume that each packet incurs a delay of d_e, where $\varepsilon \leq d_e < (1 - 1/\delta)(\mu - \varepsilon)$, in the execution e. Finally, assume that the last step in e is taken on occurrence of a quiesce event at R. By our assumption on $\mathcal{P}$ and δ,

$$\mathbf{D}(e) \quad < \quad (3 - \frac{2}{\delta})d_e + \varepsilon \, ,$$

and

$$\mathbf{Q}(e) \quad < \quad \frac{\delta}{\delta - 1} d_e + \varepsilon \, .$$

Since all packet delays are equal to d_e in the execution e, R receives a packet from S no earlier than time d_e. Since no local action are enabled in the initial state of R, it follows that R sends a packet to S no earlier than time d_e. Since all packet delays are equal to d_e in the execution e, it follows that S receives a packet from R no earlier than time $2d_e$. By definition of $\gamma_S^{(e)}$, this immediately implies:

Lemma 5.3 *In the execution e, S receives a packet from R no earlier than S-time $2d_e - \varepsilon$.*

Consider now an execution f of $\mathcal{P}$ for which $\gamma_S^{(f)}(t) = t - \varepsilon$ and $\gamma_R^{(f)}(t) = t + \varepsilon$; thus, the clock of S is initially $-\varepsilon$, while the clock of R is initially ε. Furthermore, assume that each packet incurs a delay of d_f in the execution f. Assume that $d_f = (\delta/(\delta - 1))d_e$. Finally, assume that the last step in f is taken on occurrence of a message-deliver event at R.

Since all packet delays are equal to d_f in the execution f, R may receive a packet from S no earlier than time d_f. Since no local actions are enabled in the initial state of R, it follows that R may send a packet to S no earlier than time d_f. Hence, since all packet delays are equal to d_f in f, S may receive a packet from R no earlier than time $2d_f$. Also $d_f = (\delta/(\delta - 1))d_e > d_e$, since $\delta > \delta - 1$. It implies that in f, S may receive a packet from R no earlier than time $\mathbf{D}(e) - d_e < 2d_e$. By definition of $\gamma_S^{(f)}$, this immediately implies that:

Lemma 5.4 *In the execution f, S receives a packet from S no earlier than S-time $2\mathbf{D}(e) - d_e - \varepsilon$.*

By our assumption on $\mathcal{P}$ and δ, the message-deliver event occurs in f at real time

$$\mathbf{D}(f) \quad < \quad (3 - \frac{2}{\delta})\, d_f + \varepsilon$$

$$
\begin{aligned}
&= (3 - \frac{2}{\delta}) \frac{\delta}{\delta -} d_e + \varepsilon \\
&= \frac{3\delta - 2}{\delta - 1} d_e + \varepsilon \, ;
\end{aligned}
$$

thus, this event occurs at R's local time

$$
\begin{aligned}
&\gamma_R^{(f)}(\mathbf{D}(f)) \\
&\quad < \gamma_R^{(f)}(\frac{3\delta - 2}{\delta - 1} d_e + \varepsilon) \\
&\quad\phantom{<}\ (\text{since } \mathbf{D}(f) < \tfrac{3\delta-2}{\delta-1} d_e + \varepsilon \text{ and } \gamma_R^{(f)} \text{ is strictly increasing}) \\
&\quad = \frac{3\delta - 2}{\delta - 1} d_e + 2\varepsilon \, .
\end{aligned}
$$

We continue to show a timing property of packet-send and packet-receive events in f.

Lemma 5.5 *Consider any packet π sent from S to R at S-time τ in f. Then, π arrives at R at R-time $\tau + 2\varepsilon + (\delta/(\delta - 1))d_e$.*

Proof: By definition of $\gamma_S^{(f)}$, π is sent at real time $\tau + \varepsilon$ in f. By construction of f, π arrives at R at real time $\tau + \varepsilon + d_f$. By definition of $\gamma_R^{(f)}$, it follows that π arrives at R at R-time $\tau + \varepsilon + d_f + \varepsilon = \tau + 2\varepsilon + (\delta/(\delta - 1))d_e$, as needed. ■

By Lemma 5.3, Lemma 5.4 immediately implies:

Corollary 5.6 *$e \mid S = f \mid S$ in the S-interval $[-\varepsilon, \mathbf{D}(e) - d_e - \varepsilon)$.*

Finally, we construct an execution f' in which R delivers the message twice. A prefix of f' will be equal to e'. The e' finish with quiensce of R at time $\mathbf{Q}(e)$. The remainder of f' is an execution fragment f_1' which begins at time $\mathbf{Q}(e)$. In f_1', $\gamma_S^{(f_1')}(t) = t - \varepsilon$ and $\gamma_R^{(f_1')}(t) = t$. In the Figure 4 we present the execution f' in which R delivers twice. We replay all packets sent by S in the S-interval $[-\varepsilon, \mathbf{D}(e) - d_e - \varepsilon)$ so that they incurs a delay of

$$\frac{\delta}{\delta - 1} d_e + \varepsilon > \mathbf{Q}(e) \, .$$

We show:

Lemma 5.7

$$\frac{\delta}{\delta - 1} d_e + \varepsilon < \mu \, .$$

Proof: We have $d_e < (1 - 1/\delta)(\mu - \varepsilon)$. It implies that

$$\frac{\delta}{\delta - 1} d_e + \varepsilon < \mu \, .$$

$\mu > (\delta/(\delta - 1))d_e + \varepsilon$, as needed. ■

We continue to show certain timing properties of send-packet and receive-packet events in f_1'.

Lemma 5.8 *Consider any replay packet π from S to R at S-time $t \in [-\varepsilon, \mathbf{D}(e) - d_e - \varepsilon)$, it arrive at R at R-time*

$$\frac{\delta}{\delta - 1} d_e + 2\varepsilon + t \, .$$

Proof: By definition of $\gamma^{(e')}_{(R)}$, π sent by S at real time $t + \varepsilon$. It follows that π arrives at R at real time

$$\frac{\delta}{\delta - 1} d_e + 2\varepsilon + t \, .$$

Since $\gamma^{(f_1')}_{R}(t) = t$, π arrives at R at R-time

$$\frac{\delta}{\delta - 1} d_e + 2\varepsilon + t \, .$$

■

By constraction of e', each packet sent by S in the interval $[0, \mathbf{Q}(e) - d_e)$ it arrive at R before R-time $\mathbf{Q}(e)$. We show:

Lemma 5.9 *Consider any packet π sent from S to R at time $t > \mathbf{Q}(e) - d_e$. Then, π arrive at R after R-time*

$$\frac{3\delta - 2}{\delta - 1} d_e + 2\varepsilon \, .$$

Proof: By constraction of e', π incur a delay μ to arrive at R. It follows that π arrive at R at R-time

$$\begin{aligned} \mu + t &> \frac{\delta}{\delta - 1} d_e + \varepsilon + t \\ & \quad (d_e < \tfrac{\delta}{\delta - 1}(\mu - \varepsilon)) \\ &\geq \frac{\delta}{\delta - 1} d_e + \varepsilon + 2d_e + \varepsilon \\ &= \frac{3\delta - 2}{\delta - 1} d_e + 2\varepsilon \, . \end{aligned}$$

■

By Lemma 5.6, we have $f'_1 \mid R = f \mid R$ in the interval

$$[\frac{\delta}{\delta-1}d_e + \varepsilon, \frac{3\delta-2}{\delta-1}d_e + 2\varepsilon)\,.$$

Also

$$\gamma_R^{(f)}(\mathbf{D}(f)) \; < \; (3-\frac{2}{\delta})\frac{\delta}{\delta-1}d_e + 2\epsilon\,.$$

Thus, R delivery in f'_1 at local time $\gamma_R^{(f)}(\mathbf{D}(f))$. Hence, R delivers the message twice in execution $f' = e'f'_1$. A contradiction. ■

Since the weakly synchronized clocks model is no stronger than the approximately synchronized clocks model, Theorem 5.1 immediately implies.

Corollary 5.10 *Consider the weakly synchronized clocks model, in the presence of network failures. Fix any parameter $\delta > 1$. Then, for any connection management protocol $\mathcal{P}$, there exists an execution e of $\mathcal{P}$ with $\varepsilon < d_e < (1-1/\delta)(\mu-\varepsilon)$ such that either*

$$\mathbf{D}(e) \;\; \geq \;\; (3-\frac{2}{\delta})\, d_e + \varepsilon\,,$$

or

$$\mathbf{Q}(e) \;\; \geq \;\; \frac{\delta}{\delta-1}\, d_e + \varepsilon\,.$$

5.2 Upper Bounds

We show:

Theorem 5.11 *Consider the approximately synchronized clocks model in the presence of network failures. Then, for any constants $\delta \geq 1$ and $c > 0$, there exists a connection management protocol $\mathcal{P}$ such that for every execution e of $\mathcal{P}$,*

$$\mathbf{D}(e) \;\; < \;\; (3-\frac{1}{\delta})\, d_e + (4-\frac{1}{\delta})\, 2\varepsilon + c\,,$$

and

$$\mathbf{Q}(e) \;\; < \;\; (3+\frac{1}{\delta})\, d_e + (4+\frac{1}{\delta})\, 2\varepsilon + c\,.$$

Proof: Let $\mathcal{P}_1$ be the generic connection management protocol introduced in Section 3. Fix any execution e of $\mathcal{P}_1$. We start by showing a lower bound on d_e.

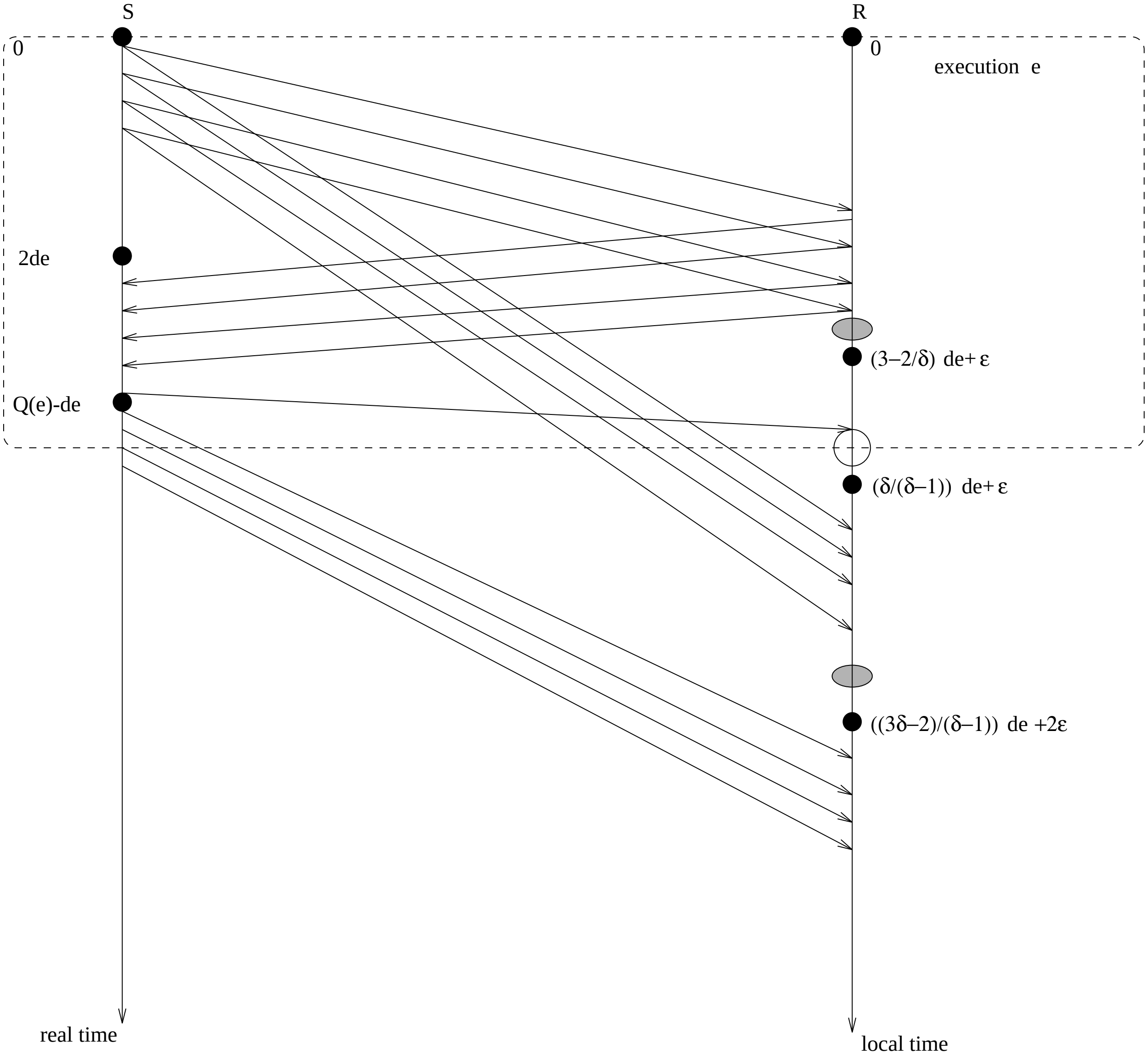

Figure 4: The execution f'

Lemma 5.12 *For any real time $t \leq \mathbf{Q}(e)$, $l^{(t)} - 2\varepsilon - 2c' \ < \ d_e$.*

Proof: Since the clocks of R and S are approximately synchronized, it follows that $\mid r - \gamma_R^{-1}(r) \mid \leq \varepsilon$ and $\mid (r - l^{(t)} + 2c') - \gamma_S^{-1}(r - l^{(t)} + 2c') \mid \leq \varepsilon$; this implies that $\gamma_R^{-1}(r) \geq r - \varepsilon$ and $\gamma_S^{-1}(r - l^{(t)} + 2c') \leq r - l^{(t)} + 2c' + \varepsilon$, respectively. By Lemma 3.3, this implies that

$$\begin{aligned} d_e &> (r - \varepsilon) - (r - l^{(t)} + 2c' + \varepsilon) \\ &= l^{(t)} - 2c' - 2\varepsilon \,, \end{aligned}$$

as needed. ■

We continue to show an upper bound on $\mathbf{D}(e)$. By Lemma 5.12, $l^{(t)} < d_e + 2\varepsilon + 2c'$. At the maximum discrete R-time not delivery,

$$\begin{aligned} t' - c' &\leq (3 - \frac{1}{\delta})\, l^{(t')} \\ &< (3 - \frac{1}{\delta})\, (d_e + 2\varepsilon + 2c') \,, \end{aligned}$$

which implies that:

$$\begin{aligned} t' &< (3 - \frac{1}{\delta})\, (d_e + 2\varepsilon) + (7 - \frac{2}{\delta})\ c' \\ &= (3 - \frac{1}{\delta})\, (d_e + 2\varepsilon) + (7 - \frac{2}{\delta})\, (7 + \frac{2}{\delta})^{-1}\ c \\ &< (3 - \frac{1}{\delta})\, (d_e + 2\varepsilon) + (7 + \frac{2}{\delta})\, (7 + \frac{2}{\delta})^{-1}\ c \\ &= (3 - \frac{1}{\delta})\, (d_e + 2\varepsilon) + c \,. \end{aligned}$$

Since the clocks are approximately synchronized, it follows that $|\gamma_S^{-1}(0) - 0| \leq \varepsilon$ and $|\gamma_R^{-1}(t') - t'| \leq \varepsilon$. This implies that $\gamma_S^{-1}(0) \geq -\varepsilon$ and $\gamma_R^{-1}(t') \leq t' + \varepsilon$, respectively. The initial send event was at real time $\gamma_S^{-1}(0)$ and the time required for R to delivery is $\gamma_R^{-1}(t')$. Thus,

$$\begin{aligned} \mathbf{D}(e) &= \gamma_R^{-1}(t') - \gamma_S^{-1}(0) \\ &\leq t' + \varepsilon - (-\varepsilon) \\ &= t' + 2\varepsilon \\ &< (3 - \frac{1}{\delta})(d_e + 2\varepsilon) + c + 2\varepsilon \\ &= (3 - \frac{1}{\delta})d_e + (8 - \frac{2}{\delta})\varepsilon + c \\ &= (3 - \frac{1}{\delta})d_e + (4 - \frac{1}{\delta})2\varepsilon + c \,, \end{aligned}$$

as needed.

We continue to show an upper bound on $\mathbf{Q}(e)$. By Lemma 5.12, $l^{(t)} < d_e + 2\varepsilon + 2c'$. So at the maximum discrete R-time not quiescence,

$$\begin{aligned} t'' - c' &\leq (3 + \frac{1}{\delta})\, l^{(t'')} \\ &< (3 + \frac{1}{\delta})\,(d_e + 2\varepsilon + 2c') \\ &= (3 + \frac{1}{\delta})\,(d_e + 2\varepsilon) + (6 + \frac{2}{\delta})\, c'\,, \end{aligned}$$

which implies that

$$\begin{aligned} t'' &< (3 + \frac{1}{\delta})\,(d_e + 2\varepsilon) + (7 + \frac{2}{\delta})\, c' \\ &= (3 + \frac{1}{\delta})\,(d_e + 2\varepsilon) + c\,. \end{aligned}$$

Since the clocks are approximately synchronized, it follows that $|0 - \gamma_S^{-1}(0)| \leq \varepsilon$ and $|t'' - \gamma_R^{-1}(t'')| \leq \varepsilon$; these imply that $\gamma_S^{-1}(0) \geq -\varepsilon$ and $\gamma_R^{-1}(t'') \leq t'' + \varepsilon$, respectively. The initial send event was at real time $\gamma_S^{-1}(0)$ and the time required for R to quiensce is $\gamma_R^{-1}(t'')$. Thus,

$$\begin{aligned} \mathbf{Q}(e) &= \gamma_R^{-1}(t'') - \gamma_S^{-1}(0) \\ &\leq t'' + \varepsilon - (-\varepsilon) \\ &= t'' + 2\varepsilon \\ &< (3 + \frac{1}{\delta})\,(d_e + 2\varepsilon) + c + 2\varepsilon \\ &= (3 + \frac{1}{\delta})\, d_e + (8 + \frac{2}{\delta})\,\varepsilon + c \\ &= (3 + \frac{1}{\delta})\, d_e + (4 + \frac{1}{\delta})\, 2\varepsilon + c\,, \end{aligned}$$

as needed. ■

We next consider the weakly synchronized clocks model.

Theorem 5.13 *Consider the weakly synchronized clocks model, in the presence of network failures. Then, for any constants $\delta \geq 1$ and $c > 0$, there exists a connection management protocol $\mathcal{P}$ such that for every execution e of $\mathcal{P}$,*

$$\mathbf{D}(e) < (3 - \frac{1}{\delta})\, d_e + (4 - \frac{1}{\delta})\, 2\varepsilon + c\,,$$

and

$$\mathbf{Q}(e) < (3 + \frac{1}{\delta})\, d_e + (4 + \frac{1}{\delta})\, 2\varepsilon + c\,.$$

Proof: Let $\mathcal{P}_1$ be the generic connection management protocol introduced in Section 3. Fix any execution e of $\mathcal{P}_1$. We start by showing a lower bound on d_e.

Lemma 5.14 *For any real time* $t \leq \mathbf{Q}(e)$, $d_e \; > \; l^{(t)} - 2\varepsilon - 2c'$.

Proof: Since the clocks of R and S are weakly appoximately synchronized, it follows that $| \, \gamma_R(t_2) - \gamma_S(t_1) - (t_2 - t_1) \, | \leq \; 2\varepsilon$. It implies that $| \, (r - (r - l^{(t)} + 2c')) - (\gamma_R^{-1}(r) - \gamma_S^{-1}(r - l^{(t)} + 2c')) \, | \leq \; 2\varepsilon$, which implies that $\gamma_R^{-1}(r) - \gamma_S^{-1}(r - l^{(t)} + 2c' \; \geq \; l^{(t)} - 2c' - 2\varepsilon$. By Lemma 3.3, this implies that $d_e \; > \; l^{(t)} - 2c' - 2\varepsilon$, as needed. ■

We continue to show an upper bound on $\mathbf{D}(e)$. By Lemma 5.14, $l^{(t)} < d_e + 2\varepsilon + 2c'$. At the maximum discrete R-time not delivery,

$$\begin{aligned} t' - c' \; &\leq \; (3 - \frac{1}{\delta}) \, l^{(t')} \\ &< \; (3 - \frac{1}{\delta}) \, (d_e + 2\varepsilon + 2c') \, , \end{aligned}$$

which implies that

$$\begin{aligned} t' \; &< \; (3 - \frac{1}{\delta}) \, (d_e + 2\varepsilon) + (7 - \frac{2}{\delta}) \; c' \\ &= \; (3 - \frac{1}{\delta}) \, (d_e + 2\varepsilon) + (7 - \frac{2}{\delta}) \, (7 + \frac{2}{\delta})^{-1} \, c \\ &< \; (3 - \frac{1}{\delta}) \, (d_e + 2\varepsilon) + (7 + \frac{2}{\delta}) \, (7 + \frac{2}{\delta})^{-1} \, c \\ &= \; (3 - \frac{1}{\delta}) \, (d_e + 2\varepsilon) + c \, . \end{aligned}$$

Since the clocks are weakly approximately synchronized, it follows that $| \, (t' - 0) - (\gamma_R^{-1}(t') - \gamma_S^{-1}(0)) \, | \leq \; 2\varepsilon$. It implies that $\gamma_R^{-1}(t') - \gamma_S^{-1}(0) \; \leq \; t' + 2\varepsilon$. The initial send event was at real time $\gamma_S^{-1}(0)$ and the time required for R to delivery is $\gamma_R^{-1}(t')$.

Hence,

$$\begin{aligned} \mathbf{D}(e) \; &= \; \gamma_R^{-1}(t') - \gamma_S^{-1}(0) \\ &\leq \; t' + 2\varepsilon \\ &< \; (3 - \frac{1}{\delta})(d_e + 2\varepsilon) + c + 2\varepsilon \\ &= \; (3 - \frac{1}{\delta})d_e + (4 - \frac{1}{\delta})2\varepsilon + c \, , \end{aligned}$$

as needed. We continue to show an upper bound on $\mathbf{Q}(e)$. By Lemma 5.14, $l^{(t)} < d_e + 2\varepsilon + 2c'$. So at the maximum discrete R-time not quiescence,

$$\begin{aligned} t'' - c' &\leq (3 + \frac{1}{\delta})\, l^{(t'')} \\ &< (3 + \frac{1}{\delta})\,(d_e + 2\varepsilon + 2c') \\ &= (3 + \frac{1}{\delta})\,(d_e + 2\varepsilon) + (6 + \frac{2}{\delta})\,c'\,, \end{aligned}$$

which implies that

$$\begin{aligned} t'' &< (3 + \frac{1}{\delta})\,(d_e + 2\varepsilon) + (7 + \frac{2}{\delta})\,c' \\ &= (3 + \frac{1}{\delta})\,(d_e + 2\varepsilon) + c\,. \end{aligned}$$

Since the clocks are weakly approximately synchronized, it follows that $\mid (t'' - 0) - (\gamma_R^{-1}(t'') - \gamma_S^{-1}(0) \mid \leq 2\varepsilon$. It imples that $\gamma_R^{-1}(t'') - \gamma_S^{-1}(0) \leq t'' + 2\varepsilon$. The initial send event was at real time $\gamma_S^{-1}(0)$ and the time required for R to quiesce is $\gamma_R^{-1}(t')$. Thus,

$$\begin{aligned} \mathbf{Q}(e) &= \gamma_R^{-1}(t'') - \gamma_S^{-1}(0) \\ &\leq t'' + 2\varepsilon \\ &< (3 + \frac{1}{\delta})\,(d_e + 2\varepsilon) + c + 2\varepsilon \\ &= (3 + \frac{1}{\delta})\,d_e + (4 + \frac{1}{\delta})\,2\varepsilon + c \end{aligned}$$

as needed. ■

We continue to show a second upper bound for the approximately and weakly approximately syncronized clocks.

We slightly modify the algorithm which we present in Section 3.2. in order to take the advatage of property $\mid \gamma_R(t_2) - \gamma_S(t_1) - (t_2 - t_1) \mid \leq 2\varepsilon$, which approximately synchronized and weakly approximately synchronized clock satisfy. Each packet which S sent to R contains both the message and the current local time. When R receive a packet estimate the $u = r - s$, where r is the local R-time at which arrive the packet at R, while s is the local time in the packet. Then R delivers immediately. After counts off $\mu - u + 2\varepsilon$ in its clock and then quiensce.

We continue to show that $\mathcal{P}_2$ for approximately synchronized and weakly approximately synchronized clocks model is a connection management protocol. We needed to prove that R will not delivery any message for a second time. Assume that a packet π send at real time t.

It follows that π arrive at R at real time $d+t$, where d is the delay incur the packet to arrive. Then R estimate the $u = \gamma_R(d+t) - \gamma_S(t)$. Then R delivery immediatelly. Assume that a replay of π arrives at R at time $\mathbf{T} > d+t$. Since the maximum packet lifetime is equal to μ, it follows that $\mathbf{T} \leq \mu + t$. We prove that R does not quiensce before time $\mu + t$. Let $\mathbf{Q}$ to be the time at which R quiensce. By the protocol, we have that $\gamma_R(\mathbf{Q}) - \gamma_R(d+t) \; = \; \mu - u + 2\varepsilon$. Since the clocks are approximately synchronized or weakly approximately syncronized, it follows that $\mid \gamma_R(\mathbf{Q}) - \gamma_S(t) \; - (\mathbf{Q} - t) \mid \leq \; 2\varepsilon$. It implies that

$$\begin{aligned}
\mathbf{Q} &\geq \gamma_R(\mathbf{Q}) - \gamma_S(t) \; + t - 2\varepsilon \\
&= \gamma_R(\mathbf{Q}) - \gamma_R(d+t) \; + \gamma_R(d+t) - \gamma_S(t) \; + t - 2\varepsilon \\
&\geq \mu - u + 2\varepsilon - 2\varepsilon + u + t \\
&\quad (\text{since } u = \gamma_R(d+t) - \gamma_S(t) \text{ and} \\
&\quad \gamma_R(\mathbf{Q}) - \gamma_R(d+\chi) \; = \mu - u + 2\varepsilon) \\
&= \mu + t \, .
\end{aligned}$$

Since $T \leq \mu + t$, this implies that $\mathbf{Q} \geq \mathbf{T}$. It follows that R never delivers a message a second time. Thus,

Theorem 5.15 *For the approximately and weakly approximately synchronized clocks models, $\mathcal{P}_2$ is connection management protocol.*

We show:

Theorem 5.16 *Consider the weakly approximately synchronized clocks model in the presence of network failures. Then , there exists a connection management protocol $\mathcal{P}$ such that for every execution e of $\mathcal{P}$,*

$$\mathbf{D}(e) \leq d_e \, ,$$

and

$$\mathbf{Q}(e) \leq \mu + 4\varepsilon \, .$$

Proof: Let $\mathcal{P}_2$ be the connection management protocol introduced in the beging of this Section. Fix any execution e of $\mathcal{P}_2$.

Assume that S send the initial packet at local time 0. The packet incur a delay of $d \leq d_e$ to arrive at R. Thus, the packet arrive at R at time $d + \gamma_S^{-1}(0)$. By the protocol $\mathcal{P}_2$, when R receive the initial packet estimate the $u = \gamma_R(d + \gamma_S^{-1}(0)) - 0$. After R delivery immmediately at time $d + \gamma_S^{-1}(0)$. It follows that R delivery at local time $\gamma_R(d + \gamma_S^{-1}(0))$. Since the packet sent at time $\gamma_S^{-1}(0)$ and R delivery at time $d+\gamma_S^{-1}(0)$, it immediatelly follows that $\mathbf{D}(e) = d + \gamma_S^{-1}(0) - \gamma_S^{-1}(0) \leq d_e$. By the protocol $\mathcal{P}_2$, after R wait to elapses local time $\mu - u + 2\varepsilon$ and then quiensce at local time $\mathbf{T} = \gamma_R(d + \gamma_S^{-1}(0)) + \mu - u + 2\varepsilon \; = \; \mu + 4\varepsilon$, since $u = \gamma_R(d + \gamma_S^{-1}(0))$. Since the clocks are weakly approximate syncronized by the Lemma 2.1, we have that $|\mathbf{T} - 0 - (\gamma_R^{-1}(\mathbf{T}) - \gamma_S^{-1}(0))| \; \leq \; 2\varepsilon$. It implies that $\gamma_R^{-1}(\mathbf{T}) - \gamma_S^{-1}(0) \; \leq \; \mu + 4\varepsilon$. Since R quiensce at local R-time $\mathbf{T}$, and the initial packet send at S-time 0, we have that:

$$\begin{aligned} \mathbf{Q}(e) &= \gamma_R^{-1}(\mathbf{T}) - \gamma_S^{-1}(0) \\ &\leq \mu + 4\varepsilon \,, \end{aligned}$$

as needed. ■

Since the weakly approximately synchronized clocks model is no stronger than approximately synchronized clocks model, Theorem 5.16 implies,

Corollary 5.17 *Consider the approximately synchronized clocks model in the presence of network failures. Then , there exists a connection management protocol $\mathcal{P}$ such that for every execution e of $\mathcal{P}_1$,*

$$\mathbf{D}(e) \leq d_e \,,$$

and

$$\mathbf{Q}(e) \leq \mu + 4\varepsilon \,.$$

6 Drifting Clocks

In this section, we present our lower bound for the drifting clocks model, under network and node failures.

Theorem 6.1 *Consider the drifting clocks model in the presence of network and node failures. Then, for any connection management protocol $\mathcal{P}$, there exists an execution e of $\mathcal{P}$ with $d_e < \mu/(3\rho + 1)$ such that*

$$\mathbf{D}(e) \;\; \geq \;\; 3\rho d_e \,.$$

Proof: Assume, by way of contradiction, that there exists a connection management protocol $\mathcal{P}$ for the drifting clocks model in the presence of network and node failures such that for every execution e of $\mathcal{P}$ with $d_e < \mu/(3\rho+1)$, $\mathbf{D}(e) < 3\rho d_e$. We construct an execution of $\mathcal{P}$ containing two message-deliver events.

We start with an informal outline of our proof. We construct a sequence of executions e, e', f and f', so that R delivers a message twice in f'. In e and f, the clocks of R and S are "slow", while in e', the clocks of R and S are "fast". We start with e, which terminates immediately after R delivery(when R delivery crash immediately). We continue to construct e', which S cannot distinguish from e to S, while R still delivers in e' and after crash. By modifying R's clock, we "perturb" e to obtain f, which S cannot distinguish from e; still, f terminates immediately after R crash. Finally, we construct f' as the "concatenation" of e' and f; in f', R first delivers and then crashes, before it receives replays of all packets in a way that R "sees" them arriving as in f. This leads R to deliver again, which contradicts the correctness of $\mathcal{P}$. We now present the details of the formal proof.

We construct an execution e of $\mathcal{P}$ in which

$$\gamma_S^{(e)}(t) \quad = \quad \frac{t}{\rho} - 3(1-\frac{1}{\rho})d_e$$

and

$$\gamma_R^{(e)}(t) \quad = \quad \frac{t}{\rho}.$$

Thus, the clock of S is initially $-3(1-1/\rho)d_e$, while the clock of R is initially 0. Each packet incurs a delay of d_e. We construct e so that S sends its intial packet at local time $-3(1-1/\rho)d_e$ and the second packet send at local time 0. It implies that S send the initial packet at real time 0 and the second packet at time $3(\rho-1)d_e$. We construct e so that R crash immediately when receive the initial packet from S, so that R cannot respond to S. Assume that up to $(3\rho-2)d_e$, when R receive a replay of initial packet crash immediately so it cannot respond to S. Also assume that one replay of the initial packet arrive at R at the moment $(3\rho-2)d_e$. Note that the second packet sent from S to R arrive at R at time $(3\rho-2)d_e$. It implies that the replay of the initial packet and the second packet arrive at R at local time $(3-2/\rho)d_e$. Thus, since no local actions are enabled in the intial state of R, R may sent a packet to S no earlier than time $(3\rho-2)d_e$. Hence, since all packet delays are equal to d_e in e, S may receive a packet from R no earlier than time $(3\rho-1)d_e$. It follows that:

Claim 6.2 *In the execution e, the inputs that S receives in the interval $[0,(3\rho-1)d_e)$ is the initial input from U_S.*

By our assumption on $\mathcal{P}$, in e, R delivers at real time $\mathbf{D}(e) < 3\rho d_e$; thus, R delivers at local time

$$\begin{aligned} \gamma_R^{(e)}(\mathbf{D}(e)) &< \gamma_R(3\rho d_e) \\ &\quad \text{(since } \mathbf{D}(e) < 3\rho d_e \text{ and } \gamma_R^{(e)} \text{ is strictly increasing)} \\ &= 3d_e\,. \end{aligned}$$

We have then R crash immediately.

We continue to construct an execution e' for which

$$\gamma_S^{(e')}(t) = \rho t - 3(1 - \frac{1}{\rho})d_e$$

and

$$\gamma_R^{(e')}(t) = \rho t\,.$$

Thus, both clocks are "fast". Also the clock of S is initially $-3(1 - 1/\rho)d_e$, while the clock of R is initially 0. Futhermore, assume that each packet incurs a delay of $d_{e'} = d_e/\rho^2$. We construct e' so that S sends its intial packet at local time $-3(1 - 1/\rho)d_e$ and the second packet send at local time 0. We construct e' so that R crashes immediately on receipt of the initial packet from S, so that R cannot respond to S. Assume that up to R-time $(3 - 2/\rho)d_e$, when R receive a replay of initial packet crash immediately so it cannot respond to S. Also assume that one replay of the initial packet arrive at R at local time $(3 - 2/\rho)d_e$. Note that the second packet sent from S to R arrive at R at local time $(3 - 2/\rho)d_e$. Also notice that, by construction the ammount of local time that elapses between the send and the receipt of every packet is the same in e and e'. By construction, $e' \mid S = e \mid S$ and $e' \mid R = e \mid R$. So in e', R delivers before local time $3d_e$. It follows that R delivers before real time $3d_e/\rho$. We have then R crash immediately.

We continue to construct an execution f for which

$$\gamma_S^{(f)}(t) = \frac{t}{\rho} - 3(1 - \frac{1}{\rho})d_e$$

and

$$\gamma_R^{(f)}(t) = \frac{t}{\rho} + 3d_e\,.$$

Thus, both clock are "slow" and the clock of S is initially $-3(1 - 1/\rho)d_e$, while the clock of R is initially $3d_e$. Futhermore, each packet incurs a delay of $d_f = d_e$. We construct f so that S sends its intial packet at time 0 and the second packet at time $3(\rho - 1)d_e$. Assume that up to $(3\rho - 2)d_e$, when R receive a replay of initial packet crash immediately so it cannot respond to S. Assume that R crashes immediately when receive the initial packet so from S, that R cannot respond to S. Also assume that one replay of the initial packet arrive in R at time $(3\rho - 2)d_e$. Note that the second packet sent from S to R arrive at R at time $(3\rho - 2)d_e$. It implies that the replay of initial packet and the second replay arrive at R at local time $(3 - 2/\rho)d_e + 3d_e$. Since no local actions are enabled in the intial state of R, R may sent a packet to S no earlier than time $(3\rho - 2)d_e$. Hence, since all packet delays are equal to d_e in the timed execution f, S may receive a packet from R no earlier than time $(3\rho - 1)d_e$. It follows that:

Claim 6.3 *In f, the inputs that S receives in the interval $[0, (3\rho - 1)d_e)$ is the initial input from U_S.*

By our assumption on protocol $\mathcal{P}$, in f, R delivers at time

$$\mathbf{D}(f) < 3\rho d_f = 3\rho d_e\,;$$

thus, R delivers at local time

$$\begin{aligned}\gamma_R^{(f)}(\mathbf{D}(f)) &< \gamma_R^{(f)}(3\rho d_e)\\ &\quad \text{(since } \mathbf{D}(f) < 3\rho d_e \text{ and } \gamma_R^{(f)} \text{ is strictly increasing)}\\ &= 6d_e\,.\end{aligned}$$

Since all packet delays are equal to d_e in the execution f, the only input that R receives before delivery are the packets sent by S in the interval $[0, \mathbf{D}(f) - d_e)$. It follows that:

Claim 6.4 *The inputs that R receives before delivery are the packets sent by S in the S-interval*

$$[-3(1 - \frac{1}{\rho})d_e, \gamma_R^{(f)}(\mathbf{D}(f)) - \frac{d_e}{\rho} - 3d_e)\,.$$

We continue to show certain timing properties of send-packet and receive-packet events in f.

Lemma 6.5 *Consider any packet π sent from S to R at S-time*

$$t \in [-3(1 - \frac{1}{\rho})d_e, \gamma_R^{(f)}(\mathbf{D}(f)) - 3d_e - \frac{d_e}{\rho})\,.$$

Then π arrives at R at R-time

$$3d_e + \frac{d_e}{\rho} + t\,.$$

Proof: By definition of $\gamma_S^{(f)}$, π is sent at real time ρt. By construction of f, π arrive at R at real time $\rho t + d_e$. It follows that π arrives at R at R-time

$$\begin{aligned} \gamma_R^{(f)}(d_e + \rho t) &= \frac{d_e + \rho t}{\rho} + 3d_e \\ &= 3d_e + \frac{d_e}{\rho} + t\,, \end{aligned}$$

as needed. ■

We continue to show that e' and f are indistinguishable to S in an initial interval of its local time.

Lemma 6.6 *$f \mid S = e' \mid S$ in the S-interval*

$$[-3(1 - \frac{1}{\rho})d_e, \gamma_R^{(f)}(\mathbf{D}(f)) - \frac{d_e}{\rho} - 3d_e)\,.$$

Proof: By our assumption on protocol $\mathcal{P}$, we have that $\mathbf{D}(f) < 3\rho d_e$, it follows that $\mathbf{D}(f) - d_e < (3\rho - 1)d_e$. By Claim 6.3, this impies that: in f, the only input that S receives in the interval $[0, \mathbf{D}(f) - d_e)$ is the initial input from U_S. Also by Claim 6.2, in e, the inputs that S receives in the interval $[0, \mathbf{D}(f) - d_e)$ is the initial input from U_S. Thus, $e \mid S = f \mid S$ in the interval $[0, \mathbf{D}(f) - d_e)$. Also by construction $e \mid S = e' \mid S$. Thus, $f \mid S = e' \mid S$ in the interval $[0, \mathbf{D}(f) - d_e)$. By definition of $\gamma_R^{(f)}$ and $\gamma_S^{(f)}$, $f \mid S = e' \mid S$ in the S-interval

$$[-3(1 - \frac{1}{\rho})d_e, \gamma_R^{(f)}(\mathbf{D}(f)) - \frac{d_e}{\rho} - 3d_e)\,.$$

■

Finally, we construct an execution f' in which R delivers the message twice. A prefix of f' is equal to e'. The remainder of f' is an execution fragment f_1', which begins at time $\mathbf{D}(e')$ with R in its initial state. In f_1', $\gamma_S^{(f_1')}(t) = \gamma_R^{(f_1')}(t) = \rho t$. The execution f' is shown in Figure 5. We replay the initial packet so that incurs a delay of $(3/\rho - 2/\rho^2)d_e + 3d_e/\rho$ to arrive. It follows that the replay of initial packet arrive at R at local time $(3 - 2/\rho)d_e + 3d_e$. We replay all packets sent by S in the interval $[0, \gamma_R^{(f)}(\mathbf{D}(f)) - d_e/\rho - 3d_e)$, so that each incurs a delay of $3d_e/\rho + d_e/\rho^2$ to arrive.

We continue to show certain timing properties of send-packet and receive-packet events in f_1'.

Lemma 6.7 *Consider any replay of packet π sent by S to R at S-time*

$$t \in [-3(1-\frac{1}{\rho})d_e, \gamma_R^{(f)}(\mathbf{D}(f)) - \frac{d_e}{\rho} - 3d_e)\,.$$

Then π arrive at R at R-time

$$3d_e + \frac{d_e}{\rho} + t\,.$$

Proof: By definition of $\gamma_R^{(f_1')}$, π is sent at real time t/ρ. By construction of f_1', π arrives at R at real time $t/\rho + 3d_e/\rho + d_e/\rho^2$. It follows that π arrive at R at R-time

$$\gamma_R^{(f_1')}(\frac{t}{\rho} + \frac{3d_e}{\rho} + \frac{d_e}{\rho^2}) \;=\; 3d_e + \frac{d_e}{\rho} + t\,,$$

as needed. ∎

By Lemmas 6.5 and 6.7, the replays arriving at R as they did in execution f. Any other packets sent by S incur a delay of μ to arrive at R;hence any other packets sent by S take $\rho\mu$ units of local time to arrive at R. Thus, any each packet sent by S after S-time $\gamma_R^{(f)}(\mathbf{D}(f)) - d_e/\rho - 3d_e$, it will receive from R after R-time

$$\begin{aligned}
\gamma_R^{(f)}(\mathbf{D}(f)) - \frac{d_e}{\rho} - 3d_e + \rho\mu \;&>\; \gamma_R^{(f)}(\mathbf{D}(f)) - \frac{d_e}{\rho} - 3d_e\; + 4\rho d_e \\
&\quad\; (d_e < \mu/(3\rho+1)) \\
&=\; \gamma_R^{(f)}(\mathbf{D}(f)) + (4-3-\frac{1}{\rho})d_e \\
&\geq\; \gamma_R^{(f)}(\mathbf{D}(f))\,,
\end{aligned}$$

since $\rho \geq 1$. So we can ensure that any packet sent by S after S-time $\gamma_R^{(f)}(\mathbf{D}(f)) - d_e/\rho - 3d_e$ will not interfre with this part of the construction. By our construction, the only inputs that R receives following delivery and up to time $\gamma_R^{(f)}(\mathbf{D}(f))$ are replays of packets sent by S in the S-interval $[-3(1-\frac{1}{\rho})d_e, \gamma_R^{(f)}(\mathbf{D}(f)) - 3d_e - \frac{d_e}{\rho})$. Also, by Claim 6.4, in f, the inputs that R receives up to time $\gamma_R^{(f)}(\mathbf{D}(f))$ are replay of packets sent by S in the S-interval $[-3(1-\frac{1}{\rho})d_e, \gamma_R^{(f)}(\mathbf{D}(f)) - 3d_e - \frac{d_e}{\rho})$. By Lemma 6.6, $f \mid S = e' \mid S$ in the S-interval $[-3(1-1/\rho)d_e, \gamma_R^{(f)}(\mathbf{D}(f)) - 3d_e - d_e/\rho)$. Thus, we can construct the fragment f_1' so that $f \mid R = f_1' \mid R$ in the R-interval $[3d_e, \gamma_R^{(f)}(\mathbf{D}(f))]$. Thus, R delivers the message in f_1'. Hence, R delivers the message twice in the execution $f' = e'f_1'$. A contradiction. ∎

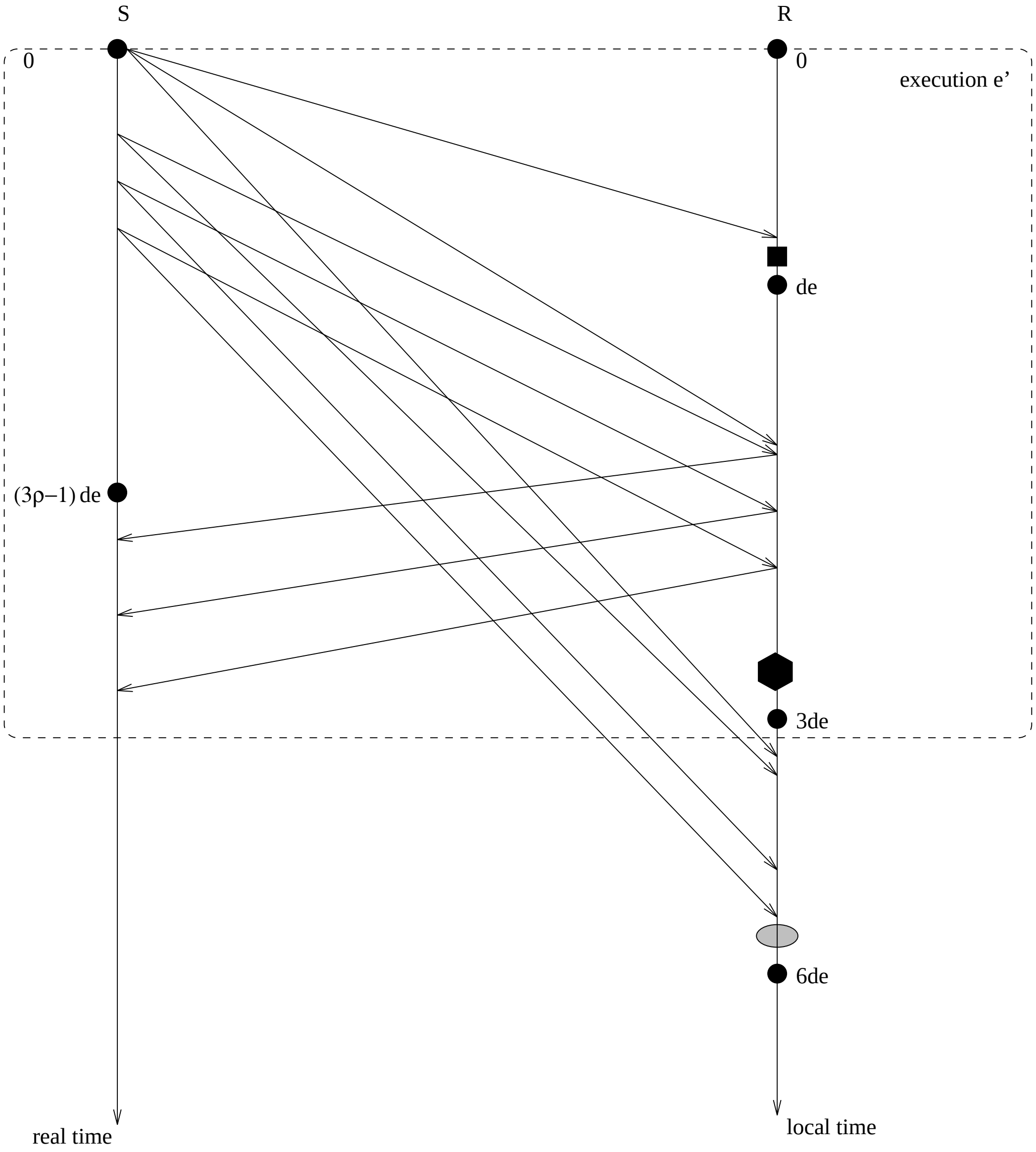

Figure 5: The execution f'

7 Approximately Synchronized Clocks

In this section, we present two lower bounds for the approximately synchronized clocks model, under both network and node failures. The first is more general but less strong.

Theorem 7.1 *Consider the approximately synchronized clocks model, under both network and node failures. Then, for any connection management protocol $\mathcal{P}$, there exists an execution e of $\mathcal{P}$ with $\varepsilon \leq d_e < \mu/3$, such that*

$$\mathbf{D}(e) \quad \geq \quad d_e + 2\varepsilon \, .$$

Proof: Assume, by way of contradiction, that there exists a connection management protocol $\mathcal{P}$ such that for every execution e of $\mathcal{P}$ with $\varepsilon \leq d_e < \mu/3$, $\mathbf{D}(e) < d_e + 2\varepsilon$. We construct an execution of $\mathcal{P}$ containing two message-deliver events.

We start with an informal outline of our proof. We construct a sequence of executions e, e', and f, so that R delivers a message twice in f. In all of these executions, the clock of R "lags" by ε that of S. We start with any execution e which terminate with R delivering a message and immediately crashing. We "perturb" e to abtain e' which is indistinguishable from e to either S , while all messages incur a delay larger than the corresponding one in e. Finnally we continue to construct f as "concatenation" of e and e';in f, R first delivers and crashes and next receives replays of all packets in such way that R "sees" all packets arriving as in f. By construction of f, R delivers again , which cotradicts the correctness of $\mathcal{P}$. We now present the details of the formal proof.

Consider an execution e of $\mathcal{P}$ for which $\gamma_S^{(e)}(t) = t$, and $\gamma_R^{(e)}(t) = t + \varepsilon$; thus, the clocks of S and R hold the initial values 0 and ε, respectively. Futhermore, assume that each packet incurs a delay of d_e in the execution e, where $\varepsilon \leq d_e < \mu/9$. Finally, assume that the last step in e is taken on occurrence of a crash event at R, which immediately follows a message-deliver event at R.

By assumption on $\mathcal{P}$, in e, R delivers at real time $\mathbf{D}(e) < d_e + 2\varepsilon$. Since all packet delays are equal to d_e in the execution e, R may receive a packet no earlier than time d_e. Since no local actions are enabled in the initial state of R , it immediately follows that R may send a packet to S no earlier than time d_e. It follows in e, S does not receive a packet from R before real time $2d_e$. Also $2\varepsilon \leq 2d_e$, since $d_e \geq \varepsilon$. It follows that:

Claim 7.2 *In e, the only input S receives in the interval $[0, 2\epsilon)$ is the initial input from U_S.*

We construct an execution e' of $\mathcal{P}$ in $\gamma_S^{(e')}(t) = t$, and $\gamma_R^{(e')}(t) = t + \varepsilon$. Thus, the clock of S is initially 0 while the clock of R is initially ε . Futhermore, each packet incurs a delay of $d_{e'} = d_e + 2\varepsilon$ in the execution e'. By assumption on $\mathcal{P}$, in e', R delivers at real time $\mathbf{D}(e') < d_{e'} + 2\varepsilon = d_e + 4\varepsilon$. Thus, R delivers at R-time

$$\begin{aligned}\gamma_R^{(e')}(\mathbf{D}(e')) &< \gamma_R^{(e')}(d_e + 4\varepsilon)\\ &\quad (\text{since } \mathbf{D}(e') < d_e + 4\varepsilon \text{ and } \gamma_R^{(e')} \text{ is strictly increasing})\\ &= d_e + 5\varepsilon\\ &\quad (\text{by definition of } \gamma_R^{(e')}).\end{aligned}$$

Since all packet delays are equal to $d_e + 2\varepsilon$ in the execution e', R may receive a packet no earlier than real time $d_e + 2\varepsilon$. Since, no local actions are enabled in the initial state of R, it immediately follows that R may send a packet to S no earlier than real time $d_e + 2\varepsilon$. It follows that:

Claim 7.3 *In e', S does not receive a packet from R before real time $2d_e + 4\varepsilon$.*

We continue to show certain timing properties of send-packet and receive-packet in e'.

Lemma 7.4 *Consider any replay packet π sent from S to R at S-time $t \in [0, 2\varepsilon)$. Then π arrives at R at R-time $t + d_e + 3\varepsilon$.*

Proof: By definition of $\gamma_S^{(e')}$, π sent at real time t. By construction of e', π arrives at R at real time $t + d_e + 2\varepsilon$. By definition of $\gamma_R^{(e')}$, it follows that π arrives at R at R-time $t + d_e + 3\varepsilon$, as needed. ■

We continue to show:

Lemma 7.5 *$e' \mid S = e \mid S$ in the S-interval* $[0, 2\varepsilon)$.

Proof: Since $d_e \geq \varepsilon$, $2\epsilon \leq 2d_e \leq 2d_e + 4\epsilon$. By Claim 7.2 and Claim 7.3, this implies that $e' \mid S = e \mid S$ in the interval $[0, 2\varepsilon)$. By definitions of $\gamma_S^{(e')}$ and $\gamma_S^{(e)}$, it follows that $e' \mid S = e \mid S$ in the S-interval $[0, 2\varepsilon)$, as needed. ■

Finally, we construct an execution f in which R delivers the message twice. A prefix of f is equal to e. The remainder of f is an execution fragment f_1 which begins at time $\mathbf{D}(e)$ with R in its initial state. In f_1, $\gamma_S^{(f_1)}(t) = t$ and $\gamma_R^{(f_1)}(t) = t + \varepsilon$. In the Figure 6 we present the execution f in which R delivers twice. We replay all packets sent by S so that each incurs a delay of $d_e + 2\epsilon$ to arrive. We continue to show certain timing properties of send-packet and receive-packet in e'.

Lemma 7.6 *Consider any replay packet π sent from S to R at S-time $t \in [0, 2\varepsilon)$. Then π arrives at R at R-time $t + d_e + 3\varepsilon$.*

Proof: By definition of $\gamma_S^{(e)}$, π sent at real time t. By construction of f, π arrives at R at real time $t + d_e + 2\varepsilon$. By definition of $\gamma_R^{(f_1)}$, it follows that π arrives at R at R-time $t + d_e + 3\varepsilon$, as needed. ■

We show:

Lemma 7.7 *Consider any replay packet π sent from S to R at S-time $t \geq 2\varepsilon$. Then π arrives at R after R-time $d_e + 5\varepsilon$.*

Proof: By construction of f, π incur a delay $d_e + 2\varepsilon$ to arrives at R. It follows that π arrives at R at real time $t + d_e + 2\varepsilon > d_e + 4\varepsilon$. By definition of $\gamma_R^{(f_1)}$, it follows that π arrives at R at R-time

$$\begin{aligned}
\gamma_R^{(f_1)}(t + d_e + 2\varepsilon) &\geq \gamma_R^{(f_1)}(d_e + 4\varepsilon) \\
&\quad \text{(since } t + d_e + 2\varepsilon > d_e + 4\varepsilon \text{ and } \gamma_R^{(f_1)} \text{ is strictly increasing)} \\
&= d_e + 5\varepsilon \\
&\quad \text{(by definitions of } \gamma_R^{(f_1)}\text{)},
\end{aligned}$$

as needed. ■

By Lemma 7.4, 7.5, 7.7 and 7.6, we have that $f_1 \mid R = e' \mid R$, in the R interval $[d_e + 2\varepsilon, d_e + 5\varepsilon)$. Since $\gamma_R^{(e')}(\mathbf{D}(e')) < d_e + 5\varepsilon$, it follows that: in f_1, R delivers at R-time $\gamma_R^{(e')}$ Thus, R delivers twice in the execution $f = ef_1$. A contradiction. ■

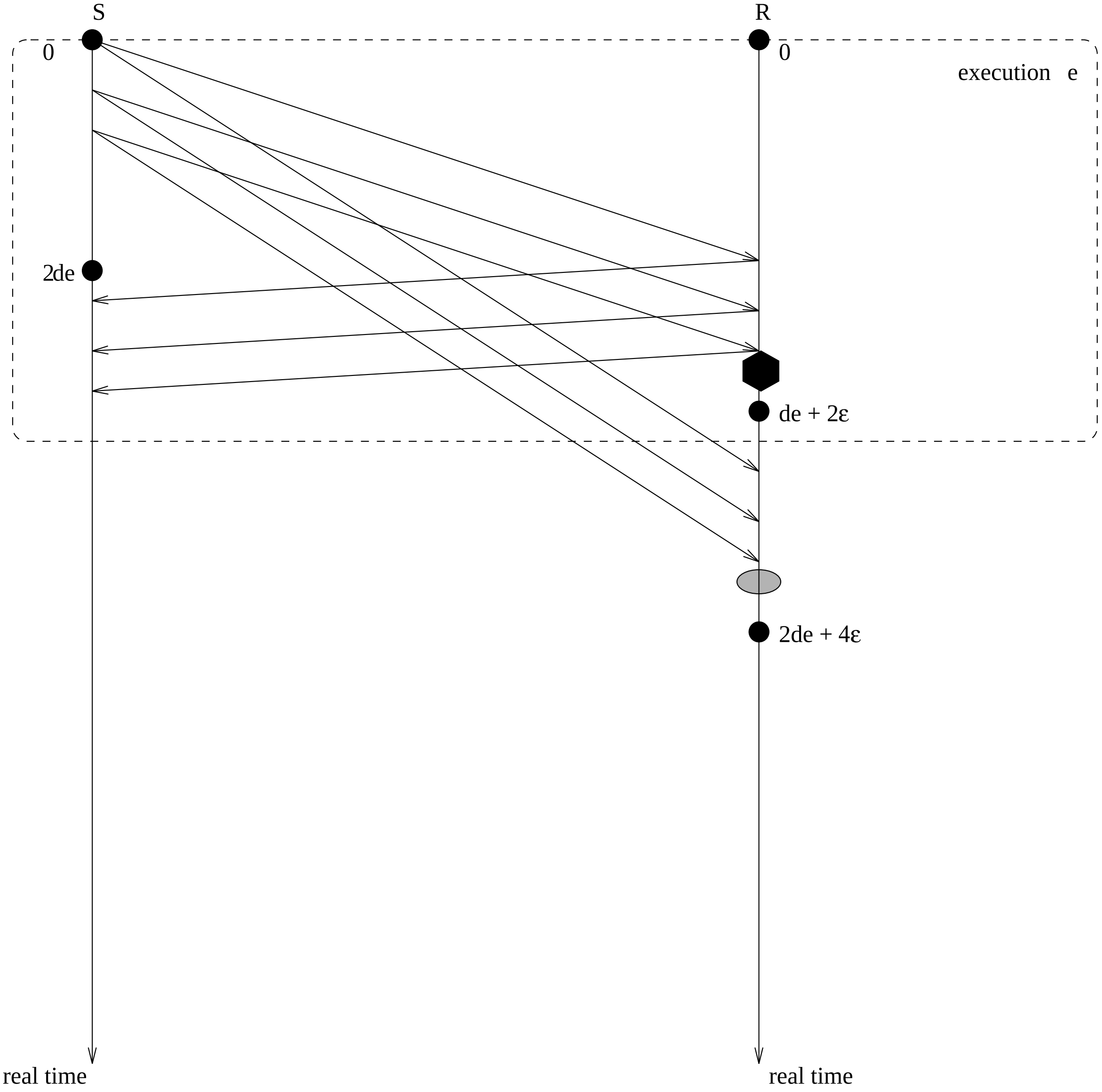

Figure 6: The execution f

Since the weakly synchronized clocks model is no stronger than the approximately synchronized clocks model, Theorem 7.1 immediately implies:

Corollary 7.8 *Consider the weakly synchronized clocks model, under both network and node failures. Then, for any connection management protocol $\mathcal{P}$, there exists an execution e of $\mathcal{P}$ with $\varepsilon \leq d_e < \mu/3$. for which*

$$\mathbf{D}(e) \ \geq \ d_e + 2\varepsilon \, .$$

We continue to show a stronger but less general lower bound.

Theorem 7.9 *Consider the approximately synchronized clocks model, under both network and node failures. Then, for any connection management protocol $\mathcal{P}$, there exists an execution e of $\mathcal{P}$ with $\varepsilon \leq d_e \leq (\mu - 6\varepsilon)/5$, such that*

$$\mathbf{D}(e) \ \ \geq \ \ 3d_e + 2\varepsilon \, .$$

Proof: Assume, by way of contradiction, that there exists a connection management protocol $\mathcal{P}$ for the approximately synchronized clocks model in the presence of both network and node failures such that for every execution e of $\mathcal{P}$ with $\varepsilon \leq d_e < (\mu - 6\varepsilon)/5$, $\mathbf{D}(e) < 3d_e + 2\varepsilon$. We construct an execution of $\mathcal{P}$ containing two message-deliver events.

We start with an informal outline of our proof. Our proof constructs a sequence of executions e, e', f, f' so that R delivers a message twice in f'. We start with execution e which terminates with R delivering a message and immediately crashing. We "perturb" e to obtain e' which is indistinguishable from e to either S or R, while all messages from R to S take time μ in e'; so, R still delivers and immediately crashes by the end of e'. We continue to construct f which is indistinguishable from e' to S, while R only delivers in f but does not crash; the construction uses the fact that communication from R to S is slow in e'. Finally, we construct f' as the "concatenation" of e' and f; in f', R first delivers and crashes and next receives replays of all packets in such a way that R "sees" all packets arriving as in f. By the construction of f, R delivers again, which contradicts the correctness of $\mathcal{P}$. We now present the details of the formal proof.

Consider an execution e of $\mathcal{P}$ for which $\gamma_S^{(e)}(t) = t - \varepsilon$, and $\gamma_R^{(e)}(t) = t$. Thus, the clocks of S and R initially hold the values $-\varepsilon$ and 0, respectively. Furthermore, assume that each packet incurs a delay of d_e in the execution e. We construct e so that S sends its initial packet at time 0. It implies that S sends its initial packet at S-time $-\epsilon$. By the Theorem 7.1, we have that R delivers at time $\mathbf{D}(e) \geq d_e + 2\varepsilon$. We construct e so that R crash immediately when receive a packet so that R cannot respond to S. The last crash happen just before the moment $d_e + 2\varepsilon$. Notice that the moment just before $d_e + 2\varepsilon$ R is in initial state. Also assume that one replay of each packet has received by R before the moment $d_e + 2\varepsilon$ arrive at R at time $d_e + 2\varepsilon$. Thus, since no local actions are enabled in the initial state of R, R may sent a packet to S no earlier than time $d_e + 2\varepsilon$. Hence, since all packet delays are equal to d_e in e, S may receive a packet from R no earlier than time $2d_e + 2\varepsilon$. It implies that:

Claim 7.10 *In e , the only input that S receives in the interval $[0, 2d_e + 2\varepsilon)$ is the initial input from U_S.*

By our assumption on $\mathcal{P}$, in e, R delivers at real time $\mathbf{D}(e) < 3d_e + 2\varepsilon$. By definition of $\gamma_R^{(e)}$ R delivery at R-time $\gamma_R^{(e)}(\mathbf{D}(e)) = \mathbf{D}(e)$. We have then R crash immediately.

We continue to show certain timing properties of send-packet and receive-packet events in e.

Lemma 7.11 *Consider any packet π sent from S to R at S-time $t \in [-\varepsilon, \varepsilon)$. Then π arrives at R at R-time $d_e + 2\varepsilon$. Consider any packet π_1 sent from S to R at S-time $t \in [\varepsilon, \mathbf{D}(e) - d_e - \varepsilon)$. Then π_1 arrives at R at R-time $t + d_e + \varepsilon$.*

Proof: By construction of e, one replay of π arrives at R at R-time $d_e + 2\varepsilon$. By definition of $\gamma_S^{(e)}$, π_1 is sent at real time $t + \varepsilon$. By definition of $\gamma_S^{(e)}$, π_1 is sent at real time $t + \varepsilon$. By construction of e, π_1 arrives at R at real time $t + d_e + \varepsilon$. By definition of $\gamma_R^{(e)}$, π_1 arrives at R at R-time $t + d_e + \varepsilon$. ■

We construct an execution e' in which $\gamma_S^{(e')}(t) = t - \varepsilon$ and $\gamma_R^{(e')}(t) = t$. Thus, the clock of S is initially $-\varepsilon$ while the clock of R is initially 0. Assume that each packet sent from S to R in the interval $[0, 2\varepsilon)$ arrives at R at real time $d_e + 2\varepsilon$. Also assume that any other packet sent from S to R incurs a delay of d_e. Each packet from R to S incurs a delay of μ. Since all packets which sent from S to R in the interval $[0, 2\varepsilon)$ arrives at R at real time $d_e + 2\varepsilon$ and all other packet delays from S to R are equal to d_e in the execution e', R may receive a packet from S no earlier than time $d_e + 2\varepsilon$. Since no local actions are enabled initial state of R, it follows that R may send a packet to S no earlier than time $d_e + 2\varepsilon$. Hence, since all packet delays from R to S are equal to μ in e', S may receive a packet from R no earlier than time

$$\mu + d_e \quad > \quad 5d_e + d_e + 8\varepsilon$$

$$\begin{aligned} & \quad (\text{since } d_e < (\mu - 6\varepsilon)/5) \\ = & \quad 6d_e + 8\varepsilon\,. \end{aligned}$$

It follows that:

Claim 7.12 *In e', the input that S receives in the interval $[0, 6d_e + 8\varepsilon)$ is the initial input from U_S.*

We continue to show certain timing properties of send-packet and receive-packet events in e'.

Lemma 7.13 *Consider any packet π sent from S to R at S-time $t \in [-\varepsilon, \varepsilon)$. Then π arrives at R at R-time $d_e + 2\varepsilon$. Consider any packet π_1 sent from S to R at S-time $t \in [\varepsilon, \mathbf{D}(e) - d_e - \varepsilon)$. Then π_1 arrives at R at R-time $t + d_e + \varepsilon$.*

Proof: By construction of e', one replay of π arrives at R at R-time $d_e + 2\varepsilon$. By definition of $\gamma_S^{(e')}$, π_1 is sent at real time $t + \varepsilon$. By definition of $\gamma_S^{(e')}$, π_1 is sent at real time $t + \varepsilon$. By construction of e', π_1 arrives at R at real time $t + d_e + \varepsilon$. By definition of $\gamma_R^{(e')}$, π_1 arrives at R at R-time $t + d_e + \varepsilon$. ■

We show:

Lemma 7.14 *$e \mid S = e' \mid S$ in the interval $[0, \mathbf{D}(e) - d_e)$ and $e \mid R = e' \mid R$ in the interval $[0, \mathbf{D}(e))$.*

Proof: We have that

$$\begin{aligned} 6d_e + 8\varepsilon & > 2d_e + 2\varepsilon \\ & > \mathbf{D}(e) - d_e \\ & \quad (\text{since } \mathbf{D}(e) < 3d_e + 2\varepsilon)\,. \end{aligned}$$

By Claim 7.10 and Claim 7.12, this implies that $e \mid S = e' \mid S$ in the interval $[0, \mathbf{D}(e) - d_e)$. By definitions of $\gamma_S^{(e')}$ and $\gamma_S^{(e)}$, it implies that $e \mid S = e' \mid S$ in the S-interval $[-\varepsilon, \mathbf{D}(e) - d_e - \varepsilon)$. By Lemma 7.11, 7.13 it implies that $e \mid R = e' \mid R$ in the R-interval $[0, \gamma_R^{(e)}(\mathbf{D}(e)))$. By definitions of $\gamma_R^{(e')}$, it implies that $e \mid R = e' \mid R$ in interval $[0, \mathbf{D}(e))$. ■

By Lemma 7.14 follows that in e' R delivery at time $\mathbf{D}(e)$. We have then R crash immediately.

We now construct an execution f for which $\gamma_S^{(f)}(t) = t - \varepsilon$ and $\gamma_R^{(f)}(t) = t$. Thus, the clock of S is initially $-\varepsilon$, while the clock of R is initially 0. Each packet incurs a delay of $d_f = \mathbf{D}(e)$. We construct f so that S sends its initial packet at time 0. By the Theorem 7.1, we have that R delivers at time $\mathbf{D}(f) \geq \mathbf{D}(e) + 2\varepsilon$. We construct f so that R crash immediately when receive a packet so that R cannot respond to S. The last crash happen just before the moment $\mathbf{D}(e) + 2\varepsilon$. Also assume that one replay of each packets has received by R before the moment $\mathbf{D}(e) + 2\varepsilon$ arrive at R at time $\mathbf{D}(e) + 2\varepsilon$. Thus, since no local actions are enabled in the initial state of R, R may sent a packet to S no earlier than time $\mathbf{D}(e) + 2\varepsilon$. Hence, since all packet delays are equal to $\mathbf{D}(e)$ in f, S may receive a packet from R no earlier than time $2\mathbf{D}(e) + 2\varepsilon$. It follows that:

Claim 7.15 *In f, the input that S receives in the interval $[0, 2\mathbf{D}(e) + 2\varepsilon)$ is the initial input from U_S.*

By our assumption, in f, R delivers at time

$$\begin{aligned}\mathbf{D}(f) &< 3d_f + 2\varepsilon \\ &= 3\mathbf{D}(e) + 2\varepsilon\,.\end{aligned}$$

We then have R crash immediately. Also

$$\begin{aligned}\mathbf{D}(f) - \mathbf{D}(e) &< 2\mathbf{D}(e) + 2\varepsilon \\ &\quad (\text{since } \mathbf{D}(f) < 3\mathbf{D}(e) + 2\varepsilon) \\ &< 6d_e + 6\varepsilon \\ &\quad (\text{since } \mathbf{D}(e) < 3d_e + 2\varepsilon)\,.\end{aligned}$$

By Claim 7.12 and Claim 7.15, this implies that $e' \mid S = f \mid S$ in the interval $[0, \mathbf{D}(f) - \mathbf{D}(e))$. By definition of $\gamma_S^{(f)}$, it implies that:

Lemma 7.16 *$e' \mid S = f \mid S$ in the S-interval $[-\varepsilon, \mathbf{D}(f) - \mathbf{D}(e) - \varepsilon)$.*

Since all packet delays are equal to $\mathbf{D}(e)$ in the execution f, we have that in f the input that R receives before delivery are the packets sent by S in the interval $[0, \mathbf{D}(f) - \mathbf{D}(e))$. By definition of $\gamma_S^{(f)}$, it implies that:

Lemma 7.17 *In f the input that R receives before delivery are the packets sent by S in the S-interval $[-\varepsilon, \mathbf{D}(f) - \mathbf{D}(e) - \varepsilon)$.*

We continue to show certain timing properties of send-packet and receive-packet events in f.

Lemma 7.18 *Consider any packet π sent from S to R at S-time $t \in [-\varepsilon, \varepsilon)$. Then π arrives at R at R-time $\mathbf{D}(e) + 2\varepsilon$. Consider any packet π_1 sent from S to R at S-time $t \in [\varepsilon, \mathbf{D}(f) - \mathbf{D}(e) - \varepsilon)$. Then π_1 arrives at R at R-time $t + \mathbf{D}(e) + \varepsilon$.*

Proof: By construction of f, one replay of π arrives at R at R-time $\mathbf{D}(e) + 2\varepsilon$. By definition of $\gamma_S^{(f)}$, π_1 is sent at real time $t+\varepsilon$. By construction of f, π_1 arrives at R at real time $t+\mathbf{D}(e)+\varepsilon$. By definition of $\gamma_R^{(f)}$, π_1 arrives at R at R-time $t + \mathbf{D}(e) + \varepsilon$. ■

Finally, we construct an execution f' in which R delivers the message twice. A prefix of f' is equal to e'. By construction, in e' R do delivery at $\mathbf{D}(e)$ and immediately crash. The remainder of f' is an execution fragment f_1' which begins at $\mathbf{D}(e)$ with R in its initial state (R do crash at time $\mathbf{D}(e)$). In f_1', $\gamma_S^{(f_1')}(t) = t - \varepsilon$ and $\gamma_R^{(f_1')}(t) = t$. In the Figure 7 we present the execution f' in which R delivers twice. We replay all packets sent by S in the S-interval $[-\varepsilon, \varepsilon)$ so that each replay arrive at R at time $\mathbf{D}(e) + 2\varepsilon$. Also we replay all packets sent by S in the S-interval $[\varepsilon, \mathbf{D}(f) - \mathbf{D}(e) - \varepsilon)$ so that each incurs a delay of $\mathbf{D}(e)$ to arrive. We have all other packets incurs a delay of μ. We show:

Lemma 7.19 *Consider any replay packet π sent by S at S-time $t \in [-\varepsilon, \mathbf{D}(f) - \mathbf{D}(e) - \varepsilon)$, it arrive at R before R-time $\gamma_R^{(f)}(\mathbf{D}(f))$.*

Proof: By definition of $\gamma_S^{(e')}$, π sent at real time $t + \varepsilon$. By construction of f', π arrives at R at rela time

$$\begin{aligned} t + \mathbf{D}(e) &< \mathbf{D}(f) - \mathbf{D}(e) + \mathbf{D}(e) \\ &= \mathbf{D}(f). \end{aligned}$$

By definitions of $\gamma_R^{(f_1')}$, $\gamma_R^{(f)}$ and $\gamma_S^{(e')}$, it follows that in f_1', each replay packet sent by S at S-time $t \in [-\varepsilon, \mathbf{D}(f) - \mathbf{D}(e) - \varepsilon)$, arrives at R before R-time $\gamma_R^{(f)}(\mathbf{D}(f))$. ■

We show:

Lemma 7.20 *Consider any replay packet π sent by S after S-time $\mathbf{D}(f) - \mathbf{D}(e) - \varepsilon$. Then π at R sfter R-time $\gamma_R^{(f)}(\mathbf{D}(f))$.*

Proof: By definition of $\gamma_S^{(e')}$, π is sent after real time $\mathbf{D}(f) - \mathbf{D}(e)$. By construction of f', π arrives at R after real time

$$\begin{aligned}
\mathbf{D}(f) - \mathbf{D}(e) + \mu \quad &> \quad \mathbf{D}(f) - \mathbf{D}(e) + 5d_e + 2\varepsilon \\
&\qquad \text{(since } d_e < (\mu - 6\varepsilon)/5) \\
&\geq \quad \mathbf{D}(f) - 3d_e - \varepsilon + 5d_e + 2\varepsilon \\
&\qquad \text{(since } \mathbf{D}(e) < 3d_e + \varepsilon) \\
&= \quad \mathbf{D}(f) + 2d_e + \varepsilon \\
&> \quad \mathbf{D}(f)\,.
\end{aligned}$$

By definition of $\gamma_R^{(f_1')}$, π arrives at R after R-time $\gamma_R^{(f)}(\mathbf{D}(f))$. ■

We continue to show certain timing properties of send-packet and receive-packet events in f_1'.

Lemma 7.21 *Consider any replay packet π sent from S to R at S-time $t \in [-\varepsilon, \varepsilon)$. Then π arrives at R at R-time $\mathbf{D}(e) + 2\varepsilon$. Consider any replay packet π_1 sent from S to R at S-time $t \in [\varepsilon, \mathbf{D}(f) - \mathbf{D}(e) - \varepsilon)$. Then π_1 arrives at R at R-time $t + \mathbf{D}(e) + \varepsilon$.*

Proof: By construction of f_1', one replay of π arrives at R at R-time $\mathbf{D}(e) + 2\varepsilon$. By definition of $\gamma_S^{(e')}$, π_1 is sent at real time $t + \varepsilon$. By construction of f_1', π_1 arrives at R at real time $t + \mathbf{D}(e) + \varepsilon$. By definition of $\gamma_R^{(f_1')}$, π_1 arrives at R at R-time $t + \mathbf{D}(e) + \varepsilon$. ■

By Lemma 7.16, 7.17, 7.18, 7.19, 7.20 and 7.21, we have that $f_1' \mid R = f \mid R$ in the interval $[\mathbf{D}(e) + 2\epsilon, \mathbf{D}(f))$. It implies that R delivery in f_1'. Thus, R delivers twice in the execution $f' = e'f_1'$. A contradiction. ■

Since the weakly synchronized clocks model is no stronger than the approximately synchronized clocks model, Theorem 7.9 immediately implies:

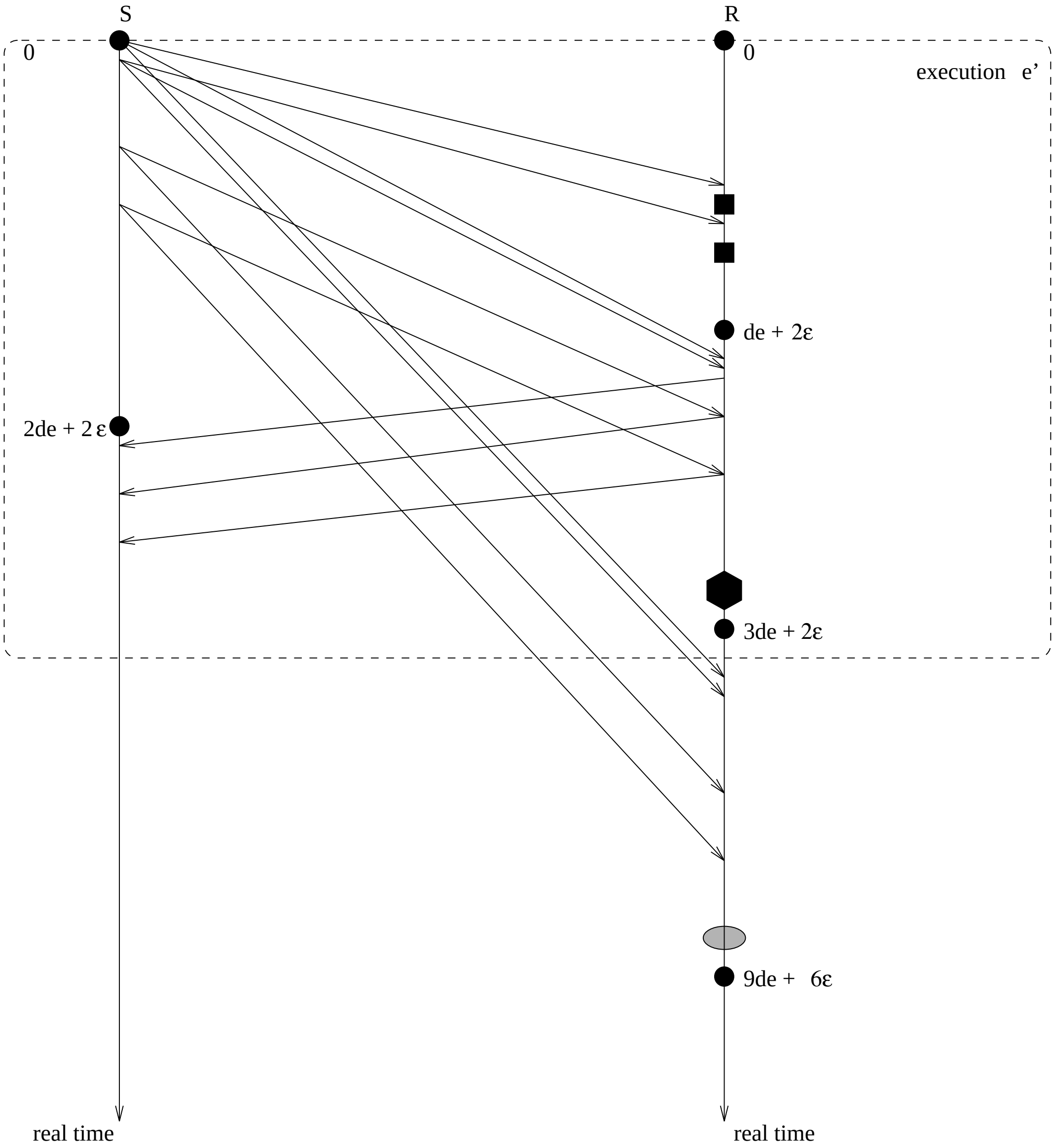

Figure 7: The execution f'

Corollary 7.22 *Consider the weakly synchronized clocks model, where messages can be duplicated and reordered, and R can crash but does not remember the time of its last crash. Then, for any connection management protocol $\mathcal{P}$, there exists an execution e of $\mathcal{P}$ with $\varepsilon \leq d_e \leq (\mu - 6\varepsilon)/5$ for which*

$$\mathbf{D}(e) \quad \geq \quad 3d_e + 2\varepsilon \, .$$

8 Discussion and Directions for Further Research

We have presented a collection of trade-offs between message delivery time and quiescence time, in the form of tight lower and upper bounds, for connection management protocols, over a number of natural settings

Acknowledgments:

We wish to thank Hagit Attiya, Jon Kleinberg and Nancy Lynch whose work [9] has inspired our work.

References

[1] Y. Afek, H. Attiya, A. Fekete, M. Fischer, N. Lynch, Y. Mansour, D.-W. Wang, and L. Zuck, "Reliable Communication over Unreliable Channels," *Journal of the ACM,* Vol. 41, No. 6, pp. 1267–1297, November 1994.

[2] H. Attiya, S. Dolev and J. L. Welch, "Connection Management Without Retaining Information," *Information and Computation,* Vol. 123, No. 2, pp. 175–191, December 1995.

[3] H. Attiya, A. Herzberg, and S. Rajsbaum, "Optimal Clock Synchronization under Different Delay Assumptions," *SIAM Journal on Computing,* Vol. 25, No. 2, pp. 369–389, April 1996.

[4] H. Attiya and R. Rappoport, "The Level of Handshake Required for Establishing a Connection," *Proceedings of the 8th International Workshop on Distributed Algorithms,* Lecture Notes in Computer Science, Vol. 857 (G. Tel and P. Vitanyi, eds.), Springer-Verlag, pp. 179–193, September/October 1994.

[5] D. Belsnes, "Single-Message Communication," *IEEE Transactions on Communications,* Vol. 24, No. 2, February 1976.

[6] D. Comer, *Internetworking with TCP/IP, Volume 1: Principles, Protocols and Architecture,* second edition, Prentice-Hall, 1991.

[7] A. Fekete, N. Lynch, Y. Mansour, and J. Spinelli, "The Impossibility of Implementing Reliable Communication in the Presence of Crashes," *Journal of the ACM,* Vol. 40, No. 5, pp. 1087–1107, November 1993.

[8] R. Gawlick, R. Segala, J. Sogaard-Andersen and N. Lynch, "Liveness in Timed and Untimed Systems", *Proceedings of the 21st International Colloqium on Automata, Languages and Programming,* Lecture Notes in Computer Science, Vol. 820 (S. Abiteboul and E. Shamir, eds.), Springer-Verlag, pp. 166–177, July 1994.

[9] J. Kleinberg, H. Attiya and N. Lynch, "Trade-offs Between Message Delivery and Quiesce Times in Connection Management Protocols," *Proceedings of the 3rd Israel Symposium on the Theory of Computing and Systems,* pp. 258–267, January 1995.

[10] B. Liskov, L. Shrira and J. Wroclawski, "Efficient At-Most-Once Messages Based on Synchronized Clocks," *ACM Transactions on Computer Systems,* Vol. 9, No. 2, pp. 125–142, 1991.

[11] J. Lundelius and N. Lynch, "An Upper and Lower Bound for Clock Synchronization," *Information and Control,* Vol. 62, No. 2/3, pp. 190–204, August/September 1984.

[12] N. Lynch and M. Tuttle, "An Introduction to Input/Output Automata," *CWI Quarterly,* Vol. 2, No. 3, pp. 219–246, September 1989.

[13] N. Lynch and F. Vaandrager, "Forward and Backward Simulations for Timing-Based Systems," in J. W. de Bakker, C. Huizing, W. P. de Roever and G. Rozenberg (editors), *Real-Time: Theory in Practice,* Lecture Notes in Computer Science, Vol. 600, pp. 397–446, Springer-Verlag, June 1991.

[14] N. Lynch and F. Vaandrager, "Forward and Backward Simulations – Part II: Timing-Based Systems," Technical Memo MIT/LCS/TM-487.b, Laboratory for Computer Science, Massachusetts Institute of Technology, September 1993.

[15] L. Murphy and A. Shankar, "Connection Management for the Transport Layer: Service Specification and Protocol Verification," *IEEE Transactions on Communications,* Vol. 39, pp. 1762–1775, 1991.

[16] B. Patt-Shamir and S. Rajsbaum, "A Theory of Clock Synchronization," *Proceedings of the 26th Annual ACM Symposium on Theory of Computing,* pp. 810–819, June 1994.

[17] R. W. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols,* Addison-Wesley Professional Computing Series, 1994.

[18] C. Sunshine and Y. Dalal, "Connection Management in Transport Protocols," *Computer Networks,* Vol. 2, pp. 454–473, 1978.

[19] A. Tanenbaum, *Computer Networks,* Prentice Hall, 1988.

[20] R. Tomlinson, "Selecting Sequence Numbers," *ACM Operating Systems Review,* Vol. 3, 1975.

[21] *Transmission Control Protocol,* DARPA Network Working Group Report RFC-793, University of Southern California, September 1981.

[22] D.-W. Wang and L. Zuck, "Tight Bounds for the Sequence Transmission Problem," *Proceedings of the 8th Annual ACM Symposium on Principles of Distributed Computing,* pp. 73–83, August 1989.

[23] D.-W. Wang and L. Zuck, "Real-Time Sequence Transmission Problem," *Proceedings of the 10th Annual ACM Symposium on Principles of Distributed Computing,* pp. 111–123, August 1991.

[24] R. W. Watson, "Timer-Based Mechanisms in Reliable Transport Protocol Connection Management," *Computer Networks,* Vol. 5, pp. 47–56, 1981.

[25] R. W. Watson, "The Delta-t Transport Protocol: Features and Experience," *Proceedings of the IEEE International Conference on Local Computer Networks,* pp. 399–407, 1989.