Tight Lower Bounds for Probabilistic Solitude Verification
On Anonymous Rings
by

Karl Abrahamson*
Andrew Adler †
Lisa Higham #
David Kirkpatrick ≠

Technical Report 90-4
January, 1990

* Computer Science Department
Washington State University
Pullman, WA 99164-1210
U.S.A.

† Department of Mathematics
≠ Department of Computer Science
University of British Columbia
Vancouver, B.C. V6T 1W5
Canada

Computer Science Department
University of Calgary
Calgary, Alberta, T2N 1N4
Canada

** This research was supported in part by the Natural Sciences and Engineering Research Council of Canada and the Killam Foundation. (Replaces TR 87-11)

Computer Science Department
University of British Columbia
Vancouver, B.C. V6T 1W5
Canada

Abstract

Tight lower bounds on the expected bit complexity of the Solitude Verification problem on anonymous asynchronous unidirectional rings are established that match the upper bounds demonstrated in a companion paper [5]. In the algorithms of [5], a variety of techniques are applied; In contrast, we find that a single technique, applied carefully, suffices for all of the lower bounds. The bounds demonstrate that, for this problem, the expected bit complexity depends subtly on the processors' knowledge of the size of the ring, and on the type of algorithm (Las Vegas or Monte Carlo / distributive or nondistributive termination).

1 Introduction

Solitude Detection is the problem: given a network of processors and a distinguished nonempty subset of the processors, called contenders, determine whether or not there is just one contender in the network. A weaker problem, *Solitude Verification*, only requires that the processors determine if there is only one contender. (An algorithm for Solitude Verification is allowed, for example, to deadlock or fail to terminate when there is more than one contender).

This paper and its companion paper [5] provide a detailed study of the communication complexity of probabilistic algorithms for these two problems for anonymous, asynchronous, unidirectional rings of processors. The two papers together determine, to within a constant factor, the complexity of these problems measured in bits of communication for all versions of the problems as two additional parameters vary. One parameter is the range of ring sizes for which the algorithm is required to work. The other is the type of termination required of the algorithm. The bit complexity bounds are established for *Monte Carlo* algorithms, that is, algorithms that err with probability at most ϵ. Bounds for *Las Vegas* algorithms, that is, algorithms that terminate with probability one and upon termination are correct, follow as corollaries.

Our motivation is to understand the effect on complexity of seemingly small changes in the problem definition. This necessitates developing a precise model of computation that incorporates enough precision to permit one to ask questions concerning probabilistic mea-

sures of complexity. Because the Solitude Detection problem is one of the simplest non-trivial problems [4] and is a subproblem of many more common problems including the well-studied problem of Leader Election [1, 2, 3, 6, 9, 12, 13, 15, 16, 17, 18, 19], it provides an appropriate domain for the case study.

In this paper the model of computation is formalized and the lower bounds are established. The matching upper bounds all appear in the companion paper [5].

The standard assumption for a distributed computation is that each processor knows when the algorithm has terminated and hence terminates its own computation. This is called distributive termination. A weaker notion of termination, nondistributive termination, is simply that all message traffic has ceased, a situation that may not be detectable. A processor under the control of a nondistributively terminating algorithm may reach some conclusion that it holds only as long as it receives no additional messages. The results of the paper are informally summarized here although the actual theorems to appear make somewhat stronger statements than this summary claims. The probability of error to be tolerated, denoted by ϵ, is a parameter to the algorithm and is assumed to be small. A parameter N is used to describe the size of rings for which the algorithm is required to work.

Throughout this paper, the constant implicit in the Ω notation is truly constant, independent of the algorithm and all of its parameters.

Informal Summary:

1. There is no Monte Carlo distributively terminating algorithm that solves Solitude Verification with error at most $1/4$ on all rings.

2. Any Monte Carlo nondistributively terminating algorithm that solves Solitude Verification on all rings of size n where n is unconstrained has expected bit complexity $\Omega\left(n \log(\frac{1}{\epsilon})\right)$.

3. Any Monte Carlo, distributively terminating algorithm that solves Solitude Verification on all rings of size n where $n \in [1, N]$ has expected bit complexity $\Omega\left(n\sqrt{\log(\frac{N}{n})} + n \log(\frac{1}{\epsilon})\right)$ when ring size is actually less than $N/2$.

4. For every $\rho > 0$, any Monte Carlo (even nondistributively terminating) algorithm that solves Solitude Verification on all rings of size n where $n \in [(\frac{1}{2}+\rho)N, N]$ has expected bit complexity $\Omega\Big(n \min\Big(\log n,\ \log\log(\frac{1}{\epsilon})\Big) + n \min\Big(\log(\frac{1}{\epsilon}),\ \log(\frac{1}{\rho}),\ \log n\Big)\Big)$ on rings of size $\Big\lceil(\frac{1}{2}+\rho)N\Big\rceil$.

5. Let $\nu(n)$ be the smallest non-divisor of n. Any Monte Carlo (even nondistributively terminating) algorithm that solves Solitude Verification on all rings of size n where n is fixed has expected bit complexity $\Omega\Big(n \min\Big(\log \nu(n),\ \log\log(\frac{1}{\epsilon})\Big) + n \min\Big(\log\log n,\ \log\log\log(\frac{1}{\epsilon})\Big)\Big)$.

6. Any Monte Carlo distributively terminating algorithm that solves Solitude Verification on all rings of size n where n is fixed has expected bit complexity $\Omega\left(n \min\left(\sqrt{\log n}\ ,\ \sqrt{\log\log(\frac{1}{\epsilon})}\right)\right)$.

Earlier lower bounds in communication complexity on rings have frequently been achieved by techniques of "cutting and splicing" [2, 4, 7, 10, 11, 14]. The goal is to show that computations of a correct algorithm cannot be "cheap" in the sense that they have small communication cost. These techniques typically start with an assumed cheap computation on a ring, and construct a new ring from the original such that each processor in the new ring has a communication history identical to its corresponding processor in the original ring. Processors in the new ring must therefore reach the same conclusion as those in the original ring. However, the construction is such that this conclusion is incorrect for the new ring.

The lower bounds in this paper are achieved through probabilistic analogues of these techniques. Even in the much simpler error-free setting, subtleties arise in the application of the techniques. For example, care must be taken not to induce deadlock when cutting out pieces of the ring. In the more general probabilistic setting, the techniques analogous to cutting and splicing are much more elaborate. In this setting complete rings of processes must be manipulated rather than just single computations. Also, at each step of the construction, bounds must be maintained on the probabilities of associated computations. This significantly more subtle environment demands a very precise and somewhat formal model

of communication. A central tool that allows the extension of the elementary techniques to a probabilistic setting is our "repeated histories lemma", Lemma 3.6, which establishes the probability, as a function of the expected communication complexity, that two processors separated by a very constrained distance will have identical communication histories. Using tools such as the repeated histories lemma a new ring is constructed by removing, replicating, and splicing together pieces of the original ring. A contradiction is reached because the new ring has more than one contender and too high a probability of asserting solitude.

Although the lower bounds in this paper are all tight, they contrast in many ways with the matching upper bounds [5]. For example, the lower bounds apply to Solitude Verification while the algorithms solve Solitude Detection. The lower bounds impose no restrictions on the type of error that is allowed, while the matching algorithms err (with low probability) only in restricted ways. The lower bounds actually apply to algorithms that have an element of nondeterministic as well as probabilistic behaviour while the upper bounds are achieved by algorithms that are purely probabilistic. The companion paper contains a discussion of the interpretation of this nondeterminism together with a description of the weak requirements of the lower bounds as contrasted with the strong requirement achieved by the matching algorithms. A surprising consequence of this investigation is the discovery that the two parameters, knowledge of ring size and type of termination, have a significant affect on the expected bit complexity of Monte Carlo Solitude Detection on an asynchronous, anonymous unidirectional ring while the complexity is insensitive to other changes in the requirements of the solution.

2 The Model

2.1 Processes, Process Sequences, and Process Rings

A *message* is an element of $\mathcal{M} = \{0,1\}^+$. If $m \in \mathcal{M}$ then the *cost* of m, denoted $\|m\|$, is the length of m. Elements of $\mathcal{B} = \mathcal{M}^*$ (that is, finite sequences of messages)[1] are referred to as

[1] For clarity and conciseness, the * operator (borrowed from operators for strings) is used in this paper to denote finite *sequences*. For example, $\mathcal{M}^*$ is the set of finite *sequences* of messages.

(message) *bundles.* The empty bundle is denoted by $\triangle$. If M is a bundle then we denote by $\|M\|$ the *cost* of M, that is the sum of the costs of all messages in M.

The behaviour of an arbitrary processor can be viewed as a sequence of input and output transmissions, where the actual choice and timing of successive outputs might depend on the preceding sequence (including timing) of outputs and inputs as well as the outcomes of random experiments. In an asynchronous setting, however, links have arbitrary delay. Therefore, it is natural, from an algorithmic point of view, to consider only algorithms whose processor behaviours are independent of timing information. Under the assumption of time-independence, a sequence of output messages produced in response to a single input message can be packaged together to form a single output message. Therefore, processors with time independent behaviour can be assumed to be *message-driven*, with *at most one* output message produced in response to each input message. This message-driven behaviour is modelled by a *probabilistic process*[2] that specifies, for each output state of the process, a probability for each $m \in \mathcal{M}$. The output state of a process records not only the last input message, but also the process's complete communication history. The probability associated with a message m and output state s is the probability that the process responds with message m after the specified sequence of outputs and inputs recorded in state s. For a fixed output state the probabilities of output messages may sum to less than 1, reflecting the possibility that no output message is produced.

There is a natural conservative extension of this view of a process in which inputs are arbitrary bundles and the response to an input bundle M is the bundle formed by concatenating in sequence each response to the individual messages in M. This *bundle-driven* interpretation ensures that *exactly one* output bundle is produced in response to one input bundle. As an additional benefit of this bundle-based interpretation, the initial output produced by a process can be viewed as a response to an empty input bundle. Thereafter, empty input bundles necessarily result in empty output bundles.

Elements of $(\mathcal{B} \times \mathcal{B})^*$ are referred to as *local behaviours*; each pair in the sequence specifies

[2]We use the word *processor* to refer to an informal notion, and *process* to refer to elements of our formal model.

an input bundle and its associated output. Thus a (probabilistic) process π induces, for each local behaviour C and each input bundle M, a probability space on $\mathcal{B}$ (the space of possible output bundles). This interpretation of a process leads to a natural definition of the probability of a local behaviour. First, observe that in general a bundle of output messages may arise from a fixed input bundle and current state in several ways. The probability of local behaviour C is the sum, over all ways that C can arise, of the product of the probabilities associated with each of its output events.

Elements of $\mathcal{H} = \mathcal{B}^*$ (that is, finite length sequences of bundles) are referred to as *(communication) histories*[3]. The *length* of a history h, denoted $|h|$, is the number of bundles in h and the *cost* of h, denoted $\|h\|$ is the sum of the costs of all bundles in h. Every local behaviour C can be decomposed into two equal length histories, the *input history* of C and *output history* of C. The input (respectively, output) history is the sequence of first (respectively, second) elements of the pairs in C. If h and h' are equal length histories, then $h\{\pi\}h'$ denotes the event that process π exhibits a local behaviour with output history h', when it has input history h. Let $h_{(i)}$ denote the length i prefix of history h. The following property is an immediate consequence of the assignment of probabilities to local behaviours:

Property 2.1 *For all histories h and h' and all $1 \leq i \leq |h|$,* $\Pr\left(h\{\pi\}h'\right) \leq \Pr\left(h_{(i)}\{\pi\}h'_{(i)}\right)$.

A process π is an *initiator* if $\Pr\left(\triangle\{\pi\}\triangle\right) = 0$ and a *non-initiator* if $\Pr\left(\triangle\{\pi\}\triangle\right) = 1$. There is no essential loss of generality in assuming that each process is either an initiator or a non-initiator; at an additional cost of n bits of communication, we can insist that all processes with $\Pr\left(\triangle\{\pi\}\triangle\right) < 1$ start by transmitting a 1-bit "wakeup" message to the next such process (via the intervening non-initiators) and thereafter continue as before. We distinguish a subset $\mathcal{M}_a \subseteq \mathcal{M}$ called *accepting messages*, and a subset $\mathcal{M}_r \subseteq \mathcal{M} - \mathcal{M}_a$ called *rejecting messages*. A history is an *accepting history* (respectively, *rejecting history*) if and only if its last message (that is, the last message in its last non-empty bundle) is an accepting

[3] In this paper, the concatenation of two histories is to be interpreted as the concatenation of two sequences. Thus, if $h_1 = M_{1,1}\cdots M_{1,i}$ and $h_2 = M_{2,1}\cdots M_{2,j}$ then h_1h_2 denotes the new history sequence comprising the bundles $M_{1,1}\cdots M_{1,i}M_{2,1}\cdots M_{2,j}$.

(respectively, rejecting) message. The set of all accepting (respectively, rejecting) histories is denoted by $\mathcal{H}_a$ (respectively, $\mathcal{H}_r$). A process is in an *accepting state* (respectively, *rejecting state*) if its last message sent was an accepting (respectively, rejecting) message.

A sequence of t processors can be linked together into a *line* by letting the output of the i^{th} processor be the input of the $(i+1)^{st}$ processor, for $i = 1, \ldots, t-1$. In many respects a line of processors behaves like a single processor whose input is the input of the first processor and whose output is the output of the last. This is formalized as follows. The sequence of histories $h_0, h_1, \ldots, h_t$ describes a *computation* of a sequence of t processes in which the i^{th} process has input history h_{i-1} and output history h_i, for $i = 1, \ldots, t$. The *cost* of such a computation is $\sum_{i=1}^{t} \|h_i\|$. Note that history h_0 does not contribute to this cost. We denote by $h_0 \{\pi_1\} h_1 \cdots \{\pi_t\} h_t$ the event that the process sequence $\pi_1, \pi_2, \ldots, \pi_t$ produces the computation described by $h_0, h_1, \ldots, h_t$, when π_1 has input history h_0. Since the probabilistic choices of the individual processes are assumed to be independent, we have:

Property 2.2 $\Pr\Big(h_0 \{\pi_1\} h_1 \cdots \{\pi_t\} h_t\Big) = \prod_{i=1}^{t} \Pr\Big(h_{i-1} \{\pi_i\} h_i\Big)$.

If $\Pi = \pi_1, \ldots, \pi_t$ is a sequence of processes then $h_0 \{\Pi\} h_t$ denotes the disjunction of the events of the form $h_0 \{\pi_1\} h_1 \cdots \{\pi_t\} h_t$ taken over all arbitrary histories $h_1, \ldots, h_{t-1}$. Thus,

$$\Pr\Big(h_0 \{\Pi\} h_t\Big) = \sum_{h_i \in \mathcal{H},\ 1 \le i < t} \Pr\Big(h_0 \{\pi_1\} h_1 \cdots \{\pi_t\} h_t\Big).$$

As a consequence of the definition of an initiator:

Property 2.3 *If* $\Pr\Big(\triangle \{\Pi\} M\Big) > 0$ *where* $M \neq \triangle$, *then* Π *contains at least one initiator.*

The preceding definitions allow us to study the communication of a *line* of processors as a function of the communication of the individual processors. Our objective, however, is to study the communication of *rings* of processors. Because the communication of processors on a ring is assumed to be independent of timing information, the computation can be scheduled in any convenient way. Imagine a scheduler that proceeds as follows. Some initiating processor is chosen to send its initial message. Starting with this initiator, the scheduler proceeds once

around the ring and each processor in turn sends a sequence of messages consisting of the processor's initial output message followed by its response to each message in sequence in its input queue. Thereafter, the scheduler continues around the ring and each processor responds in sequence to each message in its input queue. The meaningful part of the computation ends when some processor reads all the messages in its input queue without generating any output messages since then there necessarily are no outstanding messages. This is just a convenient way to schedule computation. A processor is not allowed to look ahead at messages in the remainder of its input queue before responding to the current message. Let a *round* be one pass of the scheduler around the ring. The scheduler just described serves to partition the sequence of input and output messages of each processor into bundles that correspond to each round. Round one is initiated when the scheduler delivers an empty input bundle to an arbitrary processor, p_1. Processor p_1 generates a bundle (either containing a single message or empty) in response to this input, which in turn become the input bundle of the following processor, p_2. Similarly, in each round, on the ring of processors $p_1, \ldots, p_t$, for $1 \leq i \leq t-1$, the output bundle of p_i is the input bundle of p_{i+1}. The output bundle of p_t in round j is the input bundle of p_1 in round $j+1$. By convention, the computation terminates when p_t produces an empty output bundle; this ensures that the number of input and output bundles is the same for all processors. Observe that for each processor all input and output bundles except possibly the first and last are non-empty. This reflects the fact that once a null output bundle has been produced with this scheduling of the ring, all subsequent outputs are null. Because processor behaviours are independent, the probability of the computation on the ring is the product of the probabilities of the behaviours of each processor. Note that a computation of a ring of processors may have different descriptions depending on the choice of the first processor (in effect, the point at which the ring is broken into a line). Nevertheless, the probability space of sequences of messages transmitted by each processor remains the same and hence the expected cost and outcome of the computation is independent of this choice.

The preceding informal view of a *scheduled ring* motivates the definition of a computation

of a ring of processors in terms of the computation of a line of processors. A scheduled ring is modelled by a sequence of processes that satisfies some additional properties. The input and output histories associated with any process must be of the form $\mathcal{B}(\mathcal{B} - \triangle)^*\mathcal{B}$. Such histories are referred to as *ring histories*. Furthermore, the output history h_t associated with the last processor and the input history h_0 associated with the first processor must satisfy $h_0\triangle = \triangle h_t$. Note that the placement of empty bundles is essential; it is quite possible that $h\,\{\Pi\}\,h$ holds with probability 1, and yet when Π is executed as a ring it produces no messages for lack of an initiator. Accordingly, a history sequence $h_0, h_1, \ldots, h_t$ is defined to be a *ring computation* if each h_i is a ring history and $h_0\triangle = \triangle h_t$. The event that the process sequence Π produces a ring computation with output history $h\triangle$ corresponds to the event $\triangle h\,\{\Pi\}\,h\triangle$ as long as h contains no empty bundles.

2.2 Decisions and Termination

Some lower bounds in this paper require a restriction to processes that do not change an accepting decision once made, while other lower bounds apply even to processes that may reverse their decision throughout the computation. We capture this restriction on processes by defining a process π to be *irrevocably accepting* if π never outputs another message after having output an accepting message. A process that does not accept irrevocably may be left in an accepting state after having sent its last output message without being able to detect that the computation has terminated. Notice that the restriction to irrevocably accepting processors permits rejecting decisions to be changed. This asymmetry permits slightly more generality than if both accepting and rejecting decisions were required to be firm. It also allows us to argue (below) that, without any essential loss of generality, the processes in any ring computations can be assumed to reach unanimous decisions. The unanimity condition tends to reduce the probability of erroneous computations. Also, perhaps more significantly, it allows us to simplify our notation in discussing computations, since the outcome of every computation is reflected in the history of *each* of its processors.

We now observe that it is straightforward to modify processes, at a total cost of $O(cn)$

bits of communication when there are $O(c)$ initiators, so that if any process terminates in an accepting state, then they all do. It suffices to replace empty output bundles (other than those that initiate computation) by one of two distinguished *poll* messages. Poll messages are initiated by processes that would otherwise produce an empty output bundle in response to a non-empty input bundle. A poll message has type *accept* if at least one of the processes that it has encountered (including its initiator) has reached an accepting decision. Non-accepting poll messages are forwarded to the next process that initiated a poll message while accepting poll messages continue to be forwarded until they reach a process that has sent an accepting poll message. Since any computation with c initiators results in at most c initiated poll messages, each process sends $O(c)$ bits in addition to its normal communication. For the remainder of this paper it will be assumed that all processes have been modified in this way. Hence, it can be assumed that if any process has a decision at the end of a computation, then the decision is unanimous. This is justifiable only because our proofs are insensitive to the complexity of the algorithm when there are two or more initiators. A computation $h_0, \ldots, h_t$ *accepts* if $h_i \in \mathcal{H}_a$, for $i = 0, \ldots, t$, and *rejects* if $h_i \in \mathcal{H}_r$, for $i = 0, \ldots, t$. Let $\hat{\mathcal{H}}_a$ be the subset of $\mathcal{H}_a$ containing those histories with no empty bundles.

As a consequence of the assumed unanimity of assertions:

Property 2.4 *The probability that the computation of a process sequence* Π *accepts on a ring is* $\sum_{h \in \hat{\mathcal{H}}_a} \Pr\big(\triangle h \{\Pi\} h \triangle\big)$.

The following two properties, which are also immediate consequences of the definitions in subsection 2.1, are used to draw conclusions about rings of arbitrary processes and rings of irrevocably accepting processes respectively.

Property 2.5 *Let* Π *be any process sequence and* h *any accepting history. Then, with probability at least* $\Pr\big(\triangle h \{\Pi\} h \triangle\big)$, *computations of* Π *on a ring accept.*

Let $\triangle \{\Pi\} *$ denote the disjunction of events of the form $\triangle \{\Pi\} h$ over all $h \in \mathcal{H}_a$.

Property 2.6 *Let* Π *and* Φ *be sequences of irrevocably accepting processes. Then, with probability at least* $\Pr\big(\triangle \{\Pi\} *\big)$, *computations of* $\Pi\Phi$ *on a ring accept.*

2.3 Solitude Detection Algorithms

The decision of a Solitude Detection algorithm is expressed by associating the current decision state of a process with the type of its most recent output message. Recall that decisions of processes can be assumed to be unanimous. Accordingly, a computation *asserts solitude* if it accepts and *asserts nonsolitude* if it rejects.

Let $\mathcal{A}$ denote the set of all probabilistic processes. A distributed probabilistic algorithm for Solitude Detection would normally specify a fixed initiating process from $\mathcal{A}$ for all contenders and a fixed non-initiating process for all non-contenders. (Certainly all of the algorithms of the companion paper [5] satisfy this property). It is convenient to generalize this notion of a distributed algorithm to permit arbitrary assignments from a set of initiating processes to contenders, and from a set of non-initiating processes to non-contenders. Note that there is no loss of generality in identifying initiators with contenders. Such a scheme can be imposed by having all contenders begin by sending a 1-bit "wakeup" message to the next contender (via the intervening non-contenders). Upon receipt of a "wakeup" message all processes proceed as before. We define an *algorithm* to be just the set $\alpha \subseteq \mathcal{A}$ of both initiating and non-initiating processes that are available for assignment.

This generalization gives algorithms both probabilistic and nondeterministic attributes. Like conventional probabilistic algorithms, an algorithm is said to solve a problem with probability p if, for *all* possible process assignments, the resulting computation reaches the desired conclusion with probability at least p. Like conventional nondeterministic algorithms, it is said to solve a problem efficiently if for *some* choice of process assignments the resulting computation has low expected cost.[4]

More formally, let I denote an interval of positive integers and let $\mathcal{R}_I$ denote the class of all rings of size t where $t \in I$. If $\alpha \subseteq \mathcal{A}$ is an algorithm, α^t denotes the set of sequences π_1, ..., π_t where $\pi_i \in \alpha$ for $1 \leq i \leq t$, and α^I denotes $\bigcup_{t \in I} \alpha^t$. Therefore, α^I corresponds to the

[4] This use of "nondeterminism" should not be confused with the use of the same term by some authors to refer to the undetermined behaviour of the scheduler. Since the scheduler cannot affect the communication on a unidirectional ring, the results in this paper hold for all schedulers. The term here is used in the stronger automata-theoretic sense and refers to the analysis under the assumption of lucky choices.

set of all assignments of processes in α to processors on rings in the set $\mathcal{R}_I$.

An algorithm is usually understood to be distributively terminating if the processors can detect when the computation is finished. In this case, there is no need for processors to reach tentative conclusions which may change throughout the computation as additional message arrive; instead, each processor can make one (final) decision and send that decision in its last message before terminating. Therefore an algorithm α *terminates distributively* if every $\pi \in \alpha$ makes irrevocable decisions. For lower bounds on Solitude Verification, we weaken this constraint slightly. A Solitude Verification algorithm α *accepts distributively* if every $\pi \in \alpha$ is irrevocably accepting.

This paper and its companion [5] are concerned with three closely related problems, called *Solitude Detection, Solitude Verification,* and *Weak Solitude Verification*, defined as follows. Let I denote an interval of positive integers.

Solitude Detection. α solves *Solitude Detection* with confidence $1 - \epsilon$ on rings in $\mathcal{R}_I$ if (i) for any element of α^I containing exactly one initiator, solitude is asserted with probability at least $1 - \epsilon$, and (ii) for any element of α^I containing more than one initiator, nonsolitude is asserted with probability at least $1 - \epsilon$.

Solitude Verification. α solves *Solitude Verification* with confidence $1 - \epsilon$ on rings in $\mathcal{R}_I$ if (i) for any element of α^I containing exactly one initiator, solitude is asserted with probability at least $1 - \epsilon$, and (ii) for any element of α^I containing more than one initiator, solitude is not asserted with probability at least $1 - \epsilon$.

Weak Solitude Verification. α solves *Weak Solitude Verification* with confidence $1 - \epsilon$ on rings in $\mathcal{R}_I$ if, for any element of α^I containing more than one initiator, solitude is not asserted with probability at least $1 - \epsilon$.

These definitions make it clear that Weak Solitude Verification is a subproblem of Solitude Detection. Lower bounds for Weak Solitude Verification imply lower bounds for Solitude Detection. Nonsolitude can be ascertained with a low expected cost by a simple exchange of coin tosses [5]. But the problem we focus on is the cost of verifying, with high probability, that there is only one initiator. Therefore the complexity of a Weak Solitude Verification

algorithm is defined to be the expected complexity when solitude is correctly asserted. (In the case of algorithms that never correctly assert solitude, the complexity is undefined.)

More formally, let α be an algorithm that solves Weak Solitude Verification with confidence $1-\epsilon$. Let $\mathcal{S} = \{\Pi \in \alpha^t \mid \Pr(\Pi \text{ asserts solitude}) \geq 1-\epsilon\} \neq \emptyset$. The complexity of α on rings of size t is[5] $\inf_{\Pi \in \mathcal{S}} \mathrm{E}\Big((\|h_0\| + \cdots + \|h_t\|) \mid h_0, \ldots, h_t \text{ is a computation of } \Pi \text{ that asserts solitude} \Big)$.

3 Tools for Deriving Lower Bounds

The symbol χ is reserved to denote a number (which will be given a specific value whenever necessary) called the *cheapness threshold.* A computation of a process sequence is said to be *cheap* if it has total cost at most χ. We denote by $h_0 \langle \pi_1 \rangle h_1 \cdots \langle \pi_t \rangle h_t$ (respectively, $h_0 \langle \Pi \rangle h_t$) the conjunction of the event $h_0 \{\pi_1\} h_1 \cdots \{\pi_t\} h_t$ (respectively, $h_0 \{\Pi\} h_t$) and the event that the computation is cheap.

Let Π_1, Π_2 and Π_3 be process sequences. The sequence $\Pi_1\Pi_2$ is said to be formed by *concatenation* of Π_1 and Π_2 and the sequence $\Pi_1\Pi_2\Pi_3$ is formed by *splicing* Π_2 into the sequence $\Pi_1\Pi_3$. The following properties and lemmas, which are consequences of the definitions in Section 2, are used to relate the probability of computations of the sequence $\Pi_1\Pi_2\Pi_3$ to the probabilities of related computations of the sequences $\Pi_1\Pi_3$ and Π_2.

Property 3.1 $\Pr\Big(h_1 \{\Pi_1\} h_2 \{\Pi_2\} h_2 \{\Pi_3\} h_3\Big) = \Pr\Big(h_1 \{\Pi_1\} h_2 \{\Pi_3\} h_3\Big) \cdot \Pr\Big(h_2 \{\Pi_2\} h_2\Big)$ *and* $\Pr\Big(h_1 \langle \Pi_1 \rangle h_2 \langle \Pi_2 \rangle h_2 \langle \Pi_3 \rangle h_3\Big) \leq \Pr\Big(h_1 \langle \Pi_1 \rangle h_2 \langle \Pi_3 \rangle h_3\Big) \cdot \Pr\Big(h_2 \langle \Pi_2 \rangle h_2\Big)$.

Let $h = M_1 \cdots M_k$ and $h' = M'_1 \cdots M'_l$. Then h and h' are *message-equivalent* if the sequence of messages formed from the concatenation of M_1 through M_k is the same as the sequence of messages formed from the concatenation of M'_1 through M'_l. That is, the histories are the same up to packaging of messages into bundles. Let Π be a process sequence. The concatenation of k copies of Π, denoted Π^k, is said to be formed by *replication* of Π. The following lemma relates the probabilities of certain computations of Π^k to those of Π.

[5] $\mathrm{E}(x)$ denotes the expected value of x.

Lemma 3.2 *For any history h and integer $k \geq 1$ there is a message-equivalent history h' such that* $\Pr\Big(\triangle h' \{\Pi^k\} h' \triangle\Big) \geq \Big(\Pr\Big(\triangle h \{\Pi\} h \triangle\Big)\Big)^k$.

Proof: Suppose that $\Pr\Big(\triangle h \{\Pi\} h \triangle\Big) = p$ and $h = M_1 \cdots M_r$, where $M_i \in \mathcal{B}$. Define $M_0 = \triangle$ and $M_s = \triangle$, for $s > r$. Define M_i^j to be the bundle formed by concatenation of the sequences of messages in the subsequence of bundles $M_i, \ldots, M_{i+j-1}$. Let $\lambda = \lceil r/k \rceil$. It will suffice to show that, for all $t \geq 1$, $\Pr\Big(\triangle M_1^k M_{k+1}^k \cdots M_{(\lambda-1)k+1}^k \{\Pi^t\} M_1^t M_{t+1}^k \cdots M_{(\lambda-1)k+t+1}^k\Big) \geq p^t$. The proof is by induction on t. The basis, $t = 1$, is a straightforward consequence of the bundle-driven nature of processes. For $t > 1$,

$$\begin{aligned}
&\Pr\Big(\triangle M_1^k M_{k+1}^k \cdots M_{(\lambda-1)k+1}^k \{\Pi^t\} M_1^t M_{t+1}^k M_{k+t+1}^k \cdots M_{(\lambda-1)k+t+1}^k\Big) \\
&\quad \geq \Pr\Big(\triangle M_1^k M_{k+1}^k \cdots M_{(\lambda-1)k+1}^k \{\Pi^{t-1}\} M_1^{t-1} M_t^k M_{k+t}^k \cdots M_{(\lambda-1)k+t}^k\Big) \\
&\qquad \cdot \Pr\Big(M_1^{t-1} M_t^k M_{k+t}^k \cdots M_{(\lambda-1)k+t}^k \{\Pi\} M_1^t M_{t+1}^k M_{k+t+1}^k \cdots M_{(\lambda-1)k+t+1}^k\Big) \\
&\quad = p \cdot \Pr\Big(\triangle M_1^k M_{k+1}^k \cdots M_{(\lambda-1)k+1}^k \{\Pi^{t-1}\} M_1^{t-1} M_t^k M_{k+t}^k \cdots M_{(\lambda-1)k+t}^k\Big)
\end{aligned}$$

by the bundle-driven nature of processes. Setting $t = k$, it follows that $\Pr\Big(\triangle h' \{\Pi^k\} h' \triangle\Big) \geq p^t$, where $h' = M_1^k M_{k+1}^k \cdots M_{(\lambda-1)k+1}^k$. ∎

Lemma 3.3 *If Π is any sequence of irrevocably accepting processes and $\Pr\Big(\triangle \{\Pi\} *\Big) = p$, then $\Pr\Big(\triangle \{\Pi^k\} *\Big) \geq 1 - (1-p)^k$, for $k \geq 1$*

Proof: It suffices to observe that for $k > 1$,

$$\Pr\Big(\triangle \{\Pi^k\} *\Big) \geq \Pr\Big(\triangle \{\Pi\} *\Big) + \Big(1 - \Pr\Big(\triangle \{\Pi\} *\Big)\Big) \Pr\Big(\triangle \{\Pi^{k-1}\} *\Big).$$

∎

The preceding properties and lemmas allow us to perform probabilistic analogues of collapsing, replicating and splicing once certain events and their probabilities have been identified. The lemmas to follow provide the necessary probabilities. Lemma 3.4 counts the number of distinct ring histories of cost at most k, and is required by the others. Lemma 3.5 allows us to cut a ring of processes at some link, and to treat it as a line. Lemma 3.6 locates repeated histories, and Lemmas 3.7 and 3.8 use the repeated histories to collapse the line to a desired size.

Lemma 3.4 *There are fewer than* $2 \cdot 6^k$ *distinct ring histories of cost at most* k.

Proof: Recall that ring histories contain no internal empty bundles and that messages are never empty. Imagine that bundles are separated by end-of-bundle markers and messages within bundles are separated by end-of-message markers. Then a sequence of bits can be parsed into a history by placing between between any pair of bits either an end-of-message marker or an end-of-bundle marker or no marker. Since, in addition, the history might begin or end with an empty bundle, there are $2^{k+2}3^{k-1}$ histories of cost exactly k. Therefore, the number of ring histories of cost at most k is $1 + \sum_{i=1}^{k} 2^{i+2}3^{i-1} < 2 \cdot 6^k$. ∎

Given a process sequence Π with low expected complexity, the following lemma provides a fixed short ring history and a cyclic permutation Φ of Π such that, with reasonably high probability, Φ produces a ring computation with this history as its output.

Lemma 3.5 *Let* $\epsilon < \frac{1}{2}$ *and let* $\Pi \in \mathcal{A}^t$ *be any process sequence that asserts solitude on a ring with probability at least* $1 - \epsilon$. *Suppose that the expected bit complexity of computations of* Π *that assert solitude is at most* $\chi/2$. *Then there exists a cyclic permutation* Φ *of* Π *and an accepting history* h *with* $\|h\| \leq \chi/t$ *such that* $\Pr\left(\triangle h \langle \Phi \rangle h \triangle\right) > 6^{-\chi/t-2}$.

Proof: Let $\Pi = \pi_1, \ldots, \pi_t$. Since the expected cost of computations that assert solitude is at most $\chi/2$, the probability that an arbitrary computation of Π asserts solitude and communicates fewer than χ bits is at least $(1 - \epsilon)/2 > 1/4$. The remaining probability calculations are implicitly conditional on the computation asserting solitude and being cheap.

Let e_i denote the expected cost of the output history of process π_i, over all cheap ring computations of Π that assert solitude. For some i, $e_i \leq \chi/2t$, so with probability at least $1/2$, π_i has an output history with cost no more than χ/t. But by Lemma 3.4 there are fewer than $2 \cdot 6^{\chi/t}$ distinct ring histories with at most χ/t bits and hence, with probability greater than $(1/4)6^{-\chi/t}$, π_i outputs some fixed accepting history h, where $\|h\| \leq \chi/t$. Removing the conditioning on cheap computations that assert solitude, it follows that $\Pr\left(\triangle h \langle \Phi \rangle h \triangle\right) \geq (1/16)6^{-\chi/t} > 6^{-\chi/t-2}$ where $\Phi = \pi_{i+1}, \ldots, \pi_t, \pi_1, \ldots, \pi_i$. ∎

At the heart of our lower bound proofs is the observation that a sequence of histories of sufficiently small total cost must contain the same history twice. Lemma 3.6 refines this observation to a probabilistic setting, and provides information about the separation between the repetitions. It shows that, given a process sequence Π and an integer l, with reasonably high probability, the computation of Π contains two identical histories whose separation is a small integer multiple kl of l. The probability depends on the bound τ on k and on the cheapness threshold χ.

Lemma 3.6 *Let τ and l be positive integers with τ sufficiently large. Let Π be any sequence of processes with $|\Pi| \geq \tau l$. Suppose that $\chi < (|\Pi| \log_6 \tau)/36$. Let h^0 and h^1 be arbitrary histories in $\mathcal{H}$. Then there exist non-empty process sequences Φ_1, Φ_2 and Φ_3 where $\Pi = \Phi_1\Phi_2\Phi_3$ and a history h^* such that*

i) $l \leq |\Phi_2| < \tau l$ and l divides $|\Phi_2|$, and

ii) $\Pr\left(h^0 \langle \Phi_1 \rangle h^* \langle \Phi_2 \rangle h^* \langle \Phi_3 \rangle h^1\right) \geq \tau^{-1} \Pr\left(h^0 \langle \Pi \rangle h^1\right)$.

Proof: Let $\xi = \Pr\left(h^0 \langle \Pi \rangle h^1\right)$, and suppose that $\xi > 0$, since otherwise the lemma is trivial. Let $\delta = \log_6 \tau$ and suppose that $\Pi = \pi_1, \ldots, \pi_t$. For $1 \leq i < t$, let e_i be the expected cost of the output history of π_i, conditional on $h^0 \langle \Pi \rangle h^1$. That is, $e_i = (1/\xi) \sum_h \|h\| \cdot \Pr\left(h^0 \langle \pi_{1,i} \rangle h \langle \pi_{i+1,t} \rangle h^1\right)$. If $e_i < \delta/12$, say that link i is *quiet*. If $\|h\| \leq \delta/4 - 1$, say that history h is *cheap*.

Suppose that link i is quiet and denote by h_i^* the cheap history that maximizes $\Pr\left(h^0 \langle \pi_{1,i} \rangle h_i^* \langle \pi_{i+1,t} \rangle h^1\right)$. Refer to h_i^* as the *preferred* history on link i. By Lemma 3.4, there are fewer than $2 \cdot 6^{\delta/4-1} = 6^{\delta/4}/3 = \tau^{1/4}/3$ cheap histories. So $\Pr\left(h^0 \langle \pi_{1,i} \rangle h_i^* \langle \pi_{i+1,t} \rangle h^1\right) > \frac{\xi}{\tau^{1/4}}$, since otherwise $\sum_{\|h\| \leq \delta/4-1} \Pr\left(h^0 \langle \pi_{1,i} \rangle h \langle \pi_{i+1,t} \rangle h^1 | h^0 \langle \Pi \rangle h^1\right) < (\tau^{1/4}/3)/\tau^{1/4} = 1/3$ and hence $e_i > (2/3)(\delta/4 - 1)$. This is at least $\delta/12$ for $\tau \geq 6^8$, thus contradicting the assumption that link i is quiet.

Let $B_{u,v} = \{u \cdot \tau l + v + kl \ : \ 0 \leq k \leq \tau - 1\}$, where $0 \leq u \leq \left\lfloor \frac{t-1}{\tau l} \right\rfloor - 1$ and $1 \leq v \leq l$. Note that any two members of any set $B_{u,v}$ are separated by kl where k is an integer between 1 and $(\tau - 1)$. Of the $t - 1$ internal links of Π all but $(t - 1) \bmod \tau l < t/2$ belong to $\bigcup_{u,v} B_{u,v}$.

Choose u and v such that at least $1/3$ of the τ members of $B_{u,v}$ are quiet links, given $h^0 \langle \Pi \rangle h^1$. Such a pair u, v must exist, since otherwise at least $2/3$ of at least $t/2$ links are not quiet, contradicting the assumption that $\sum_{i=1}^{t} e_i \leq \chi < t\delta/36$.

Again, because there are fewer than $\tau^{1/4}/3$ cheap histories, at least $w = \left\lfloor \tau^{3/4} \right\rfloor$ of the cheap members of $B_{u,v}$ have identical preferred histories. Let $i_1, \ldots, i_w$ be w such members, and let h^* denote their common preferred history. Let D_s denote the event $h^0 \langle \pi_{1,i_s} \rangle h^* \langle \pi_{i_s+1,t} \rangle h^1$, for $1 \leq s \leq w$. By the inclusion-exclusion principle, $\sum_{r<s} \Pr(D_r \,\&\, D_s) \geq (\sum_s \Pr(D_s)) - \xi$. Since $\Pr(D_s) > \frac{\xi}{\tau^{1/4}}$, there must exist distinct r and s such that

$$\begin{aligned} \Pr(D_r \,\&\, D_s) &\geq \frac{\left(\frac{w}{\tau^{1/4}} - 1\right)\xi}{\binom{w}{2}} \\ &> \frac{\xi}{\tau^{1/4} w} \qquad \text{for } \tau \geq 16 \\ &\geq \frac{\xi}{\tau} \end{aligned}$$

Thus, assuming $\tau \geq 6^8$, it follows that $\Pr(D_r \,\&\, D_s) \geq \tau^{-1} \Pr\left(h^0 \langle \Pi \rangle h^1\right)$. So it suffices to choose $\Phi_1 = \pi_{1,i_r}$, $\Phi_2 = \pi_{i_r+1,i_s}$ and $\Phi_3 = \pi_{i_s+1,t}$. ∎

The following two lemmas are used to collapse a ring from its initial size to below some target size t^*, overshooting as little as possible. Both lemmas apply Lemma 3.6 and Property 3.1 repeatedly to determine the probability of an event on a short sequence of processes from the probability of a related event on the original sequence. They differ in that Lemma 3.8 employs a more delicate and sophisticated strategy for collapsing than does Lemma 3.7 and thus achieves a stronger result. Although Lemma 3.8 subsumes Lemma 3.7, both are included because the naive approach of Lemma 3.7 sometimes suffices and the proof is simpler.

Lemma 3.7 *Let τ, l and t^* be positive integers with $t^* \geq \tau l$ and τ sufficiently large. Let Π be any sequence of processes with $|\Pi| \geq t^*$. Suppose that $\chi < (t^* \log_6 \tau)/36$. Let h^0 and h^1 be any histories. Then there exists a (non-contiguous) subsequence Φ of Π such that*

i) $t^* - \tau l < |\Phi| < t^*$,

ii) $|\Phi| \equiv |\Pi| \pmod{l}$, *and*

iii) $\Pr\left(h^0 \langle \Phi \rangle h^1\right) \geq \tau^{-1-\frac{|\Pi|-t^*}{l}} \Pr\left(h^0 \langle \Pi \rangle h^1\right)$.

Proof: When applied to τ, l and a process sequence Γ meeting the required conditions, Lemma 3.6 identifies process sequences Φ_1, Φ_2 and Φ_3 satisfying

1. $l \leq |\Phi_2| < \tau l$ and l divides $|\Phi_2|$, and
2. $\Pr\left(h^0 \langle \Phi_1 \rangle h^* \langle \Phi_2 \rangle h^* \langle \Phi_3 \rangle h^1\right) \geq \tau^{-1} \Pr\left(h^0 \langle \Pi \rangle h^1\right)$.

Property 3.1 ensures that $\Pr\left(h^0 \langle \Phi_1 \Phi_3 \rangle h^1\right) \geq \tau^{-1} \Pr\left(h^0 \langle \Gamma \rangle h^1\right)$. The application of Lemma 3.6 and Property 3.1 to obtain the new sequence $\Gamma' = \Phi_1 \Phi_3$ from Γ is called a *collapsing* step.

Starting with sequence Π, collapsing steps are repeatedly applied as long as the conditions of Lemma 3.6 are met, that is, as long as the resulting sequence has length at least t^*. Each collapsing step removes a subsequence that has length kl where $1 \leq k \leq \tau - 1$. Let the final sequence Φ be the first sequence obtained by successive collapsing that has length less than t^*. It follows that:

1. At most $\frac{|\Pi|-t^*}{l} + 1$ collapsing steps are required to obtain Φ.
2. $|\Phi| > t^* - \tau l$.
3. $|\Phi| \equiv |\Pi| \pmod{l}$.

Each collapsing step multiplies the probability bound by τ^{-1}, so

$$\Pr\left(h^0 \langle \Phi \rangle h^1\right) \geq \tau^{-1-\frac{|\Pi|-t^*}{l}} \Pr\left(h^0 \langle \Pi \rangle h^1\right).$$

■

Lemma 3.8 *Let τ, l and t^* be positive integers with $t^* \geq \tau l$ and τ sufficiently large. Let Π be any sequence of processes with $|\Pi| \geq t^*$. Suppose that $\chi < (t^* \log_6 \tau)/36$. Let h^0 and h^1 be any histories. Then there exists a (non-contiguous) subsequence Φ of Π such that*

i) $t^* - \tau l < |\Phi| < t^*$,

ii) $|\Phi| \equiv |\Pi| \pmod{l}$, *and*

iii) $\Pr\left(h^0 \langle \Phi \rangle h^1\right) \geq \tau^{-2-\tau \ln \frac{|\Pi|-t^*}{l}} \Pr\left(h^0 \langle \Pi \rangle h^1\right)$.

Proof: As in Lemma 3.7, we apply Lemma 3.6 and Property 3.1 repeatedly, each time eliminating some processes between repeated histories. At a given collapsing step, the remaining sequence Γ has some length t', where $t^* \leq t' \leq |\Pi|$, and $t' \equiv |\Pi| \pmod{l}$, and $\Pr\left(h^0 \langle \Gamma \rangle h^1\right) = p$.

Let l' be the largest multiple of l that does not exceed $l + \frac{t'-t^*}{\tau}$. Then $l' > (t' - t^*)/\tau$. By Lemma 3.6, with l' playing the role of l and t' playing the role of t, Γ transmits repeated histories, separated by ml where $l' \leq ml < \tau l' \leq \tau l + t - t^*$, with probability at least p/τ. By Property 3.1, there exists a (non-contiguous) subsequence Γ' of Γ with length less than $t' - l'$, such that $\Pr\left(h^0 \langle \Gamma' \rangle h^1\right) \geq p/\tau$. By starting with Π and collapsing some number g times in this fashion, we eventually construct a sequence Φ where

1. $t^* - \tau l < |\Phi| < t^*$,
2. $|\Phi| \equiv |\Pi| \pmod{l}$, and
3. $\Pr\left(h^0 \langle \Phi \rangle h^1\right) \geq \tau^{-g} \Pr\left(h^0 \langle \Pi \rangle h^1\right)$

For each collapsing step except the last, the value of $t' - t^*$ decreases by a factor of at least $1 - 1/\tau$. Since the last collapse is by at least l processes, it follows that $g \leq 1 + \hat{g}$ where $\hat{g}$ is the smallest integer such that $(|\Pi| - t^*)\left(1 - \frac{1}{\tau}\right)^{\hat{g}} < l$. Taking logarithms to the base e, and using the fact that $\ln\left(1 - \frac{1}{\tau}\right) < -\frac{1}{\tau}$, we get that $\hat{g} \leq 1 + \tau \ln \frac{|\Pi|-t^*}{l}$. ∎

To summarize the notation, π represents a probabilistic process, and Π represents a sequence of probabilistic processes. If h is a communication history, then $h_{(i)}$ is the length i prefix of h. The symbol $\triangle$ represents the empty bundle, which only occurs at the beginning or end of a ring history. Notation $h \{\Pi\} h'$ denotes the event that probabilistic process Π has output history h' when its input history is h, and $h \{\Pi\} *$ is the event that Π asserts solitude when its input history is h. Notation $h \langle \Pi \rangle h'$ denotes the event consisting of the conjunction of $h \{\Pi\} h'$ and the event that the total number of bits transmitted by Π with input history h

is at most some preassigned value χ. The event notation is extended to multiple histories by letting, for example, $h_0\,\{\pi_1\}\,h_1\,\{\pi_2\}\,h_2$ represent the conjunction of $h_0\,\{\pi_1\}\,h_1$ and $h_1\,\{\pi_2\}\,h_2$.

4 Overview of Lower Bound Proofs

The lower bound proofs all proceed similarly. In this section the common structure of the proofs is highlighted. The proofs convert a size $n \in I_1$ single contender ring of processes whose computations are correct with high probability, and have low expected bit complexity, to another ring of processes with size $n' \in I_2$ and with two or more contenders whose computations err with unacceptably high probability. Because of the unanimity of decisions, it suffices to show that, with unacceptably high probability, some process in the final ring terminates in an (erroneous) accepting state. For this reason the proofs are referred to as fooling arguments. A standard fooling argument applies to arbitrary message-driven processes and thus holds for even nondistributively terminating algorithms. From a fooling argument it can be concluded that with high probability there is an erroneous accepting history in the final ring when all message traffic has ceased. A weak fooling argument applies only to irrevocably accepting processes and therefore holds for distributively terminating algorithms. From a weak argument it can be concluded that with high probability there is some accepting history at some point in the computation. Since the processes' accepting decisions are irrevocable it is unnecessary to assure that message traffic has ceased in order to conclude that such a computation is in error.

More precisely, a *fooling argument* for an algorithm α consists of the following steps.

Schema 4.1

1. Assume that for any $n \in I_1$, there is a process sequence $\Pi \in \alpha^n$ that has exactly one contender and asserts solitude with confidence at least $1 - \epsilon$ and has expected bit complexity less than $\chi/2$, where χ is chosen as an appropriate function of ϵ and n.

2. Apply Lemma 3.5 to conclude that there is a cyclic permutation Γ of Π and a history $h \in \mathcal{H}_a$ where $\|h\| \leq \chi/n$ such that $\Pr\left(\triangle h\,\langle\Gamma\rangle\,h\triangle\right) \geq 6^{-\chi/n-2}$.

3. Using the collapsing, replicating and splicing lemmas, produce a *fooling sequence* $\Phi \in \alpha^{I_2}$ with two or more contenders such that $\Pr\Big(\triangle h \{\Phi\} h \triangle\Big) > \epsilon$.

4. Apply Property 2.5 to conclude that, with probability greater than ϵ, computations of Φ erroneously assert solitude.

Steps 1, 2 and 4 are essentially the same for all the lower bound proofs. Therefore the proofs begin by stating the values of χ, I_1 and I_2 and proceeding with step 3. Step 3 differs in each of the lower bound proofs. We refer to this step as the *core* of the fooling argument. The *conclusion* of a fooling argument for α is that if α solves Weak Solitude Verification with confidence $1 - \epsilon$ on rings in $\mathcal{R}_{I_2}$, then the expected bit complexity of α on ring of size $n \in I_1$ is $\Omega(\chi)$.

A *weak fooling argument* for a distributively terminating algorithm α differs from a standard fooling argument in that the core need only establish that $\Pr\Big(\triangle \{\Phi\} *\Big) > \epsilon$ for some process sequence $\Phi \in \alpha^{[1,b]}$ where $I_2 = [a, b]$. Thereafter, step 4 applies Property 2.6 to reach the desired conclusion.

Our proofs must keep track of probabilities of events, and each collapsing, replicating or splicing operation decreases the known probability bound. Therefore, it is necessary to keep the number of steps small. In some cases, the need for efficiency results in relatively difficult proofs.

It will be useful in general to have a linear lower bound.

Theorem 4.2 *If α solves Weak Solitude Verification on rings of size n with confidence greater than 3/4, then the expected bit complexity of α is at least $n/6$ on rings of size n.*

Proof: Suppose that process sequence $\Pi \in \alpha^n$ with a single contender asserts solitude with confidence greater than 3/4 and has expected bit complexity less than $n/6$. Then, with probability at least 2/3, correct computations of Π have complexity less than $n/2$. But, by the message-driven nature of computations, it follows that, with probability greater than 2/3, message traffic in correct computations of Π travel less than half way around the ring beyond the contender. In this case the contender in Π concludes that it is alone without receiving

any communication. Thus with probability at least $(3/4)(2/3) = 1/2$ arbitrary computations of Π conclude solitude with message traffic on less than the first half of the ring.

Now consider the process sequence Φ constructed by splicing together two pieces of Π both starting with the contender: one piece consisting of $\lfloor n/2 \rfloor$ processes and the other piece $\lceil n/2 \rceil$ processes of Π. Then with probability $(1/2)^2 = 1/4$, both contenders in Φ erroneously conclude they are alone after message traffic has ceased. ■

In the interest of ease of presentation, little attempt is made to establish good asymptotic constants in the lower bounds.

5 Bounds for Ring Size Loosely Known

We say that an algorithm α *knows* only that ring size n is in $[a, N]$ if α must work for all rings in $\mathcal{R}_{[a,N]}$. We prove two lower bounds on the expected bit complexity of Monte Carlo Solitude Detection algorithms that know only that $a \leq n \leq N$, where $a \leq N/2$, namely (1) an $\Omega\left(n \log(\frac{1}{\epsilon})\right)$ lower bound for nondistributively terminating algorithms (which of course applies also to distributively terminating algorithms), and (2) an $\Omega\left(n\sqrt{\log(\frac{N}{n})}\right)$ lower bound for distributively terminating algorithms. The $\Omega\left(n \log(\frac{1}{\epsilon})\right)$ bound only holds when n happens to be at most $N/2$, although the algorithm only *knows* that $a \leq n \leq N$. When $n > N/2$, the complexity can be lower than $\Omega\left(n \log(\frac{1}{\epsilon})\right)$ [5].

The $\Omega\left(n \log(\frac{1}{\epsilon})\right)$ bound is the simpler of the two. The core involves no collapsing and only a single replication.

Theorem 5.1 *Let $0 < \epsilon < 1/4$ and let α be any (even nondistributively terminating) algorithm that solves Weak Solitude Verification with confidence $1 - \epsilon$ on any ring in $\mathcal{R}_{[a,N]}$ where $a \leq N/2$. Then the expected bit complexity of α on rings of size $n \in [a, N/2]$ is $\Omega\left(n \log(\frac{1}{\epsilon})\right)$.*

Proof: By the linear lower bound of Theorem 4.2, and we can assume that ϵ is small. A fooling argument is given by following Schema 4.1 with $\chi = (2n/5)\log_6(\frac{1}{\epsilon})$, $I_1 = [a, N/2]$ and $I_2 = [2a, N]$. After applying steps 1 and 2 of the schema, we have $\Gamma \in \alpha^{[a,N/2]}$ with exactly one contender and $h \in \mathcal{H}_a$ such that $\Pr\left(\triangle h \langle \Gamma \rangle h \triangle\right) \geq 6^{-\chi/n-2}$.

Consider the sequence $\Phi = \Gamma^2 \in \alpha^{I_2}$ formed by splicing together two copies of Γ. By Lemma 3.2, there exists a history $h' \in \mathcal{H}_a$ such that

$$\begin{aligned} \Pr\left(\triangle h' \{\Phi\} h' \triangle\right) &\geq \left(6^{-\chi/n-2}\right)^2 \\ &= 6^{-4}\epsilon^{4/5} \qquad \text{since } \chi = (2n/5)\log_6(\tfrac{1}{\epsilon}) \\ &> \epsilon \end{aligned}$$

for $\epsilon < 6^{-20}$. Since Φ has two contenders, the argument is completed by appealing to Property 2.5. ∎

The $\Omega\left(n\sqrt{\log(\frac{N}{n})}\right)$ lower bound is based on a weak fooling argument and hence only applies to distributively terminating algorithms. Like the previous lower bound, there is no collapsing in the core of the argument. However, more replication is required.

Theorem 5.2 *Let α be any distributively accepting algorithm that solves Weak Solitude Verification with confidence greater than $3/4$ on rings in $\mathcal{R}_{[a,N]}$. Then the expected bit complexity of α on rings of size $n \in [a, N]$ is $\Omega\left(n\sqrt{\log(\frac{N}{n})}\right)$.*

Proof: By Theorem 4.2 we can assume that N/n is large. A weak fooling argument is given by following Schema 4.1 with $\chi = n\sqrt{(1/2)\log_6\left(\frac{N}{n}\right)}$ and $I_1 = I_2 = [a, N]$. After applying steps 1 and 2 of the schema, we have $\Gamma \in \alpha^{[a,N]}$ with exactly one contender and $h \in \mathcal{H}_a$ satisfying $\|h\| \leq \chi/n$ and $\Pr\left(\triangle h \langle\Gamma\rangle h \triangle\right) \geq 6^{-\chi/n-2}$.

Consider the sequence Γ^k where $k = |h| \leq \chi/n$. By Lemma 3.2, there exists an accepting bundle M (formed by concatenating all the bundles in h), such that $\Pr\left(\triangle M \{\Gamma^k\} M \triangle\right) \geq 6^{-k(\chi/n+2)} \geq 6^{-(\chi/n)(\chi/n+2)}$. Hence, by Property 2.1, $\Pr\left(\triangle \{\Gamma^k\} *\right) \geq 6^{-(\chi/n)(\chi/n+2)}$.

Let $t = \lfloor N/(nk) \rfloor$. But $2(\chi/n)^2 = \log_6(N/n)$ and $k \leq \chi/n$. So $t \geq 6^{(2(\chi/n)^2)}/k - 1 > 6^{(\chi/n)^2+2(\chi/n)} > 1$. Let $\Phi = \Gamma^{tk}$. By Lemma 3.3, $\Pr\left(\triangle \{\Phi\} *\right) \geq 1 - \left(1 - 6^{-(\chi/n)(\chi/n+2)}\right)^t > 1 - 1/e > 1/2$. Since $\Phi \in \alpha^{[a,N]}$ has more than one contender, the argument is completed by appealing to Property 2.6. ∎

6 Bounds for Ring Size Approximately Known

By the matching upper bound results [5], the lower bounds of the preceding section are tight to within a constant factor as long as the algorithm knows at best that $N/2 \leq n \leq N$. But suppose that all processors know that $(\frac{1}{2}+\rho)N \leq n \leq N$ for some given positive $\rho < 1/2$. We prove two lower bounds on the expected bit complexity of Solitude Detection: (1) an $\Omega\left(n \min(\log\log(\frac{1}{\epsilon}), \log N)\right)$ bound, showing that, for sufficiently large N, the bit complexity is doubly logarithmic in $1/\epsilon$, and (2) an $\Omega\left(n \min(\log\rho, \log(\frac{1}{\epsilon}), \log N)\right)$ bound, showing that the cost is logarithmic in $1/\rho$ when N is large and ϵ is small. Both bounds apply to nondistributively terminating algorithms. For simplicity, we prove the first bound for $\rho = 1/4$, although a modified proof applies to any positive $\rho < 1/2$.

Theorem 6.1 *Let $0 < \epsilon < 1/4$, and let α be any (even nondistributively terminating) algorithm that solves Weak Solitude Verification with confidence $1-\epsilon$ on rings in $\mathcal{R}_{[3N/4,N]}$. Then the expected bit complexity of α on rings of size $n \in [3N/4, N]$ is $\Omega\left(n \min\left(\log\log(\frac{1}{\epsilon}), \log N\right)\right)$.*

Proof: By Theorem 4.2 we can assume that N is large and ϵ is small. A fooling argument is given by following Schema 4.1 with $\chi = \frac{n}{146}\left\lfloor \min\left(\log_6\log(\frac{1}{\epsilon}), \log_6 N\right)\right\rfloor$ and $I_1 = I_2 = [3N/4, N]$. After applying steps 1 and 2 of the schema, we have $\Gamma \in \alpha^{I_1}$ with exactly one contender and $h \in \mathcal{H}_a$ such that $\Pr\left(\triangle h \langle\Gamma\rangle h \triangle\right) \geq 6^{-\chi/n-2}$.

Let $\tau = 6^{73\chi/n}$, $l = \left\lfloor \frac{N}{8\tau} \right\rfloor$ and $t^* = N/2$. Since $\chi \leq (n/146)\log_6 N$, it follows that $N \geq 6^{146\chi/n} = \tau^2 > 8\tau$, implying $1 \leq l$. Moreover, $t^*(\log_6\tau)/36 = \frac{N}{2}\cdot\frac{73\chi}{n}\cdot\frac{1}{36} > \chi$. Hence, by Lemma 3.7, there exists a non-contiguous subsequence Υ of Γ such that $3N/8 \leq t^* - \tau l < |\Upsilon| < t^* = N/2$ and

$$\begin{aligned}
\Pr\left(\triangle h \langle\Upsilon\rangle h \triangle\right) &\geq \tau^{-1-\frac{n-t^*}{l}} \Pr\left(\triangle h \langle\Gamma\rangle h \triangle\right) \\
&\geq \tau^{-5\tau} 6^{-\chi/n-2} \\
&> \tau^{-6\tau}
\end{aligned}$$

By Property 2.3, Υ contains a contender. Now consider the sequence $\Phi = (\Upsilon)^2$. By Lemma 3.2, $\Pr\left(\triangle h' \{\Phi\} h' \triangle\right) \geq \tau^{-12\tau}$ where h' is equivalent to h. But $\chi \leq (n/146)\log_6\log(\frac{1}{\epsilon})$ and

$\tau = 6^{73\chi/n}$, implying $\tau \leq (\log(\frac{1}{\epsilon}))^{1/2}$. So $\tau^{-12\tau} > \epsilon$. Since $\Phi \in \alpha^{I_2}$ has two contenders, the argument is completed by appealing to Property 2.5. ■

The next theorem shows the dependence of the bit complexity of Solitude Detection on ρ. Notice that it only applies at the lower end of the interval $[(\frac{1}{2}+\rho)N, N]$.

Theorem 6.2 *Let $0 < \epsilon < 1/4$ and let α be any (even nondistributively terminating) algorithm that solves Weak Solitude Verification with confidence $1-\epsilon$ on rings in $\mathcal{R}_{[(\frac{1}{2}+\rho)N,N]}$ where $0 < \rho < 1/2$. Then the expected bit complexity of α on rings of size $\left\lceil(\frac{1}{2}+\rho)N\right\rceil$ is $\Omega\left(n \min\left(\log(\frac{1}{\rho}), \log(\frac{1}{\epsilon}), \log N\right)\right)$.*

Proof: By Theorem 4.2 we can assume that N is large, and both ρ and ϵ are small. Abbreviate $(\frac{1}{2}+\rho)N$ by n and assume n is an integer. Let z be the largest multiple of 3 not exceeding $\min\left(\log_6(\frac{1}{\rho}), \log_6(\frac{1}{\epsilon}), \log_6 N\right)$. A fooling argument is given by following Schema 4.1 with $\chi = \lfloor nz/109 \rfloor$, $I_1 = [n, n]$ and $I_2 = [n, N]$. After applying steps 1 and 2 of the schema, we have Γ of length n with exactly one contender and $h \in \mathcal{H}_a$ such that $\Pr\left(\triangle h \langle \Gamma \rangle h \triangle\right) \geq 6^{-\chi/n-2}$.

The core of the argument begins with a single collapsing step. Apply Lemma 3.6 to Γ with $l = \lfloor N6^{-z} \rfloor$ and $\tau = 6^{z/3}$. The condition $\chi < (|\Gamma| \log_6 \tau)/36$ is easily verified. Also $l \geq \max(1, \rho N)$ since $z \leq \min(\log_6 N, \log_6(\frac{1}{\rho}))$, and $l < \tau l < N/4 < n$ since z is large. So a single collapsing step removes some number k of processors from Γ, where $\rho N \leq k < N/4$. The result is a sequence $\Upsilon \in \alpha^{[(\frac{1}{4}+\rho)N, N/2]}$ such that $\Pr\left(\triangle h \langle \Upsilon \rangle h \triangle\right) \geq \tau^{-1} 6^{-\chi/n-2}$. By Property 2.3, Υ contains a contender.

Let $\Phi = (\Upsilon)^2$. By Lemma 3.2, $\Pr\left(\triangle h' \{\Phi\} h' \triangle\right) > \tau^{-2} 6^{-2\chi/n-4}$ where h' is equivalent to h. But $\log_6(\tau^2 6^{2\chi/n+4}) = 2z/3 + 2\chi/n + 4 < 0.7z + 4 < \log_6(\frac{1}{\epsilon})$, since $z \leq \log_6(\frac{1}{\epsilon})$ is large. So $\Pr\left(\triangle h' \{\Phi\} h' \triangle\right) > \epsilon$. Since $\Phi \in \alpha^{[(\frac{1}{2}+\rho)N, N]}$ has two contenders, the argument is completed by appealing to Property 2.5. ■

7 Bounds for Ring Size Exactly Known

We say that the algorithm α knows n exactly if α is only required to work on rings of size n. In that case, the lower bound proofs become more difficult, since the core must produce a new sequence of processes whose length is exactly the same as the length of the original one. For distributive termination it is easy to pad out the new sequence to the desired length, but for nondistributive termination there must be an exact size match.

We prove three lower bounds on the expected bit complexity of Monte Carlo Solitude Detection: (1) an $\Omega\Big(n \min\Big(\sqrt{\log n}, \sqrt{\log\log(\frac{1}{\epsilon})}\Big)\Big)$ bound for distributively terminating algorithms, (2) an $\Omega\Big(n \min\Big(\log \nu(n), \log\log(\frac{1}{\epsilon})\Big)\Big)$ bound for nondistributively terminating algorithms, where $\nu(n)$ is the smallest positive nondivisor of n, and (3) an $\Omega\Big(n \min\Big(\log\log n,$ $\log\log\log(\frac{1}{\epsilon})\Big)\Big)$ bound for nondistributively terminating algorithms.

Theorem 7.1 *Let $0 < \epsilon < 1/4$ and let α be a distributively accepting algorithm that solves Weak Solitude Verification with confidence $1 - \epsilon$ on rings of size n. Then the expected bit complexity of α on rings of size n is $\Omega\Big(n \min\Big(\sqrt{\log n}, \sqrt{\log\log(\frac{1}{\epsilon})}\Big)\Big)$.*

Proof: By the linear lower bound of Theorem 4.2, and we can assume that n is large and ϵ is small. Let $z = \min\Big(\sqrt{\log_6 n}, \sqrt{\log_6\log(\frac{1}{\epsilon})}\Big)$. A weak fooling argument is given by following Schema 4.1 with $\chi = nz/9$ and $I_1 = I_2 = [n, n]$. After applying steps 1 and 2 of the schema, we have Γ of length n with exactly one contender and $h \in \mathcal{H}_a$ satisfying $\|h\| \leq \chi/n = z/9$ and $\Pr\Big(\triangle h \langle\Gamma\rangle h \triangle\Big) \geq 6^{-z/9-2}$.

The core of the argument applies Lemma 3.8 to collapse Γ, followed by Lemma 3.2 to replicate the result. Let target size $t^* = \lceil 9n/z \rceil$, and parameters $\tau = \lfloor 6^{z^2/2} \rfloor \leq \sqrt{n}$ and $l = \lfloor \frac{9n}{z\tau} \rfloor$. Then $(t^* \log_6 \tau)/36 > nz/9 = \chi$ and $\tau l \leq t^*$ and $l > 1$. So the conditions of Lemma 3.8 are met. The result is a sequence Υ such that $|\Upsilon| < t^*$ and $\Pr\Big(\triangle h \langle\Upsilon\rangle h \triangle\Big) \geq \tau^{-2-\tau \ln\left(\frac{n-t^*}{l}\right)} 6^{-z/9-2}$. But $(n - t^*)/l < n/l < (z\tau)/18$, and τ is large, so $\Pr\Big(\triangle h \langle\Upsilon\rangle h \triangle\Big) > \tau^{-2}\tau^{-\tau \ln \tau}\tau^{-\tau \ln(z/18)} 6^{-z/9-2} \geq \tau^{-\tau \log \tau}$. By Property 2.3, Υ has one contender.

Let $\Phi = (\Upsilon)^k$ where $k = \max(|h|, 2) \leq z/9$. By Lemma 3.2, $\Pr\Big(\triangle M \{\Phi\} M \triangle\Big) > \tau^{-(z\tau \log \tau)/9}$, where M is the accepting bundle formed by concatenating all the bundles in h.

Hence, applying Property 2.1, $\Pr\left(\triangle\{\Phi\}*\right) \geq \Pr\left(\triangle\{\Phi\}M\right) \geq \tau^{-(z\tau\log\tau)/9}$. But for large enough n, $\log_6 \log \tau^{(z\tau\log\tau)/9} < 2\log_6 \tau \leq 2(z^2/2) \leq \log_6 \log(\frac{1}{\epsilon})$. Hence, $\Pr\left(\triangle\{\Phi\}*\right) > \epsilon$. Since Φ has length $k|\Upsilon| < z/9\left(\left\lceil\frac{9n}{z}\right\rceil - 1\right) \leq n$ and has more than one contender, the argument is completed by appealing to Property 2.6. ■

Corollary 7.2 *The complexity of any distributively accepting Las Vegas algorithm that solves Weak Solitude Verification with confidence at least* $1 - 1/2^n$ *on rings of size* n *is* $\Omega\left(n\sqrt{\log n}\right)$ *bits.*

The previous proof does not require that the fooling sequence Φ be constructed to a precise length since, when we are only interested in distributively terminating algorithms, Property 2.6 allows us to pad Φ out to length n with an additional sequence of arbitrary processes. For nondistributive termination, much more care is needed to ensure a fooling sequence of length exactly n. It is therefore not surprising to find divisibility properties of n not only in the proofs, but in the complexity bounds as well.

Theorem 7.3 *Let* $0 < \epsilon < 1/4$ *and let* α *be any (even nondistributively terminating) algorithm that solves Weak Solitude Verification with confidence* $1 - \epsilon$ *on rings of size* n. *Then the expected bit complexity of* α *on rings of size* n *is* $\Omega\left(n\min\left(\log\nu(n), \log\log(\frac{1}{\epsilon})\right)\right)$, *where* $\nu(n)$ *is the smallest nondivisor of* n.

Proof: By the linear lower bound of Theorem 4.2, we can assume that n and $\nu(n)$ are large and ϵ is small. In particular, n is even. Let z be the largest even integer not exceeding $\min\left(\nu(n) - 1, \log^{1/4}(1/\epsilon)\right)$. A fooling argument is given by following Schema 4.1 with $\chi = (n/75)(\log_6 z)$, and $I_1 = I_2 = [n, n]$. After applying steps 1 and 2 of the schema, we have Γ of length n with exactly one contender and $h \in \mathcal{H}_a$ such that $\Pr\left(\triangle h\langle\Gamma\rangle h\triangle\right) \geq 6^{-\chi/n-2}$.

The core of the argument has four steps. (1) Identify a short contiguous subsequence of Γ, which will be used in step 3 to pad a sequence to length exactly $n/2$. (2) Collapse the two other pieces found in step 1 to short sequences. (3) Splice replications of the piece found in step 1 into the shortened pieces found in step 2 to form a sequence of length exactly $n/2$. (4) Replicate the sequence from step 3 once.

Let $\lambda(x)$ denote the least common multiple of the positive integers not exceeding x. Apply Lemma 3.6 to Γ with $\tau = \tau_1 = z/2$, and $l = l_1 = n/\lambda(z)$. Condition $\chi < (|\Gamma| \log_6 \tau)/36$ follows immediately from the choice of χ. Also $\tau_1 l_1 < n$, and l_1 is a positive integer, since $z < \nu(n)$. According to Lemma 3.6, $\Gamma = \Upsilon\Psi\Theta$ where $|\Psi| = ml_1$ for some positive integer $m < \tau_1$, and, for some history h^*, $\Pr\left(\triangle h \langle \Upsilon \rangle h^* \langle \Psi \rangle h^* \langle \Theta \rangle h \triangle\right) \geq \tau_1^{-1} 6^{-\chi/n-2} > z^{-2}$. By Property 3.1, $\Pr\left(\triangle h \langle \Upsilon \rangle h^*\right)$, $\Pr\left(h^* \{\Psi\} h^*\right)$ and $\Pr\left(h^* \langle \Theta \rangle h \triangle\right)$ are all greater than z^{-2}.

Notice that $n/(2|\Psi|) = \lambda(z)/(2m)$ is an integer, since $m < \tau_1 = z/2$. So $|\Psi|$ divides $n/2$, a fact that will be crucial for step 3.

Step 2 consists of collapsing Υ and Θ each to size less than $n/4$. Apply Lemma 3.8 to each, using target size $t^* = n/4$, and parameters $l = l_2 = |\Psi|$, and $\tau = \tau_2 = z^2$. Then $\tau_2 l_2 < nz^3/(2\lambda(z))$. Since z is large, the conditions $1 \leq l_2 < \tau_2 l_2 < t^*$ and $\chi < (t^* \log_6 \tau_2)/36$ of Lemma 3.8 are easily verified. Let Υ' and Θ' be the results of applying Lemma 3.8 to Υ and Θ respectively. Then $n/2 - 2\tau_2 l_2 = n/2 - 2z^2|\Psi| < |\Upsilon'\Theta'| < n/2$. According to the lemma, $\Pr\left(\triangle h \langle \Upsilon' \rangle h^*\right)$ and $\Pr\left(h^* \langle \Theta' \rangle h \triangle\right)$ are both at least $z^{-2}\tau_2^{-2-\tau_2 \ln(\frac{n-t^*}{l_2})}$. Since $(n - t^*)/l_2 < n/|\Psi| \leq \lambda(z)$ and $\ln(\lambda(x)) \sim x$ [8] both of those probabilities are at least z^{-3z^3}.

For step 3 observe (using Lemma 3.8) that $|\Upsilon'\Theta'| \equiv (|\Upsilon| + |\Theta|) \pmod{|\Psi|}$. But $|\Upsilon\Psi\Theta| = n$, and $|\Psi|$ divides $n/2$, so $|\Psi|$ divides $|\Upsilon'\Theta'|$. Therefore, there must be some positive integer $k \leq 2z^2$ such that $|\Upsilon'\Theta'| + k|\Psi| = n/2$.

For step 4 let $\Phi = (\Upsilon'\Psi^k\Theta')^2 \in \alpha^n$. Because $\Pr\left(\triangle h \langle \Upsilon'\Theta' \rangle h \triangle\right) > 0$, Property 2.3 ensures that either Υ' or Θ' has a contender. Hence, Φ contains two contenders. Moreover, for some history h' equivalent to h, $\Pr\left(\triangle h' \{\Phi\} h' \triangle\right) \geq \left(z^{-3z^3} \cdot z^{-2k} \cdot z^{-3z^3}\right)^2 \geq z^{-12z^3-4k} > 2^{-z^4} \geq \epsilon$, since $z \leq \log^{1/4}(\frac{1}{\epsilon})$. The argument is completed by appealing to Property 2.5. ∎

When nondistributive termination is acceptable, the bound of Theorem 7.1 can be beaten [5]. In that case, the following theorem applies.

Theorem 7.4 *Let $0 < \epsilon < 1/4$ and let α be any (even nondistributively terminating) algorithm which solves Weak Solitude Verification with confidence $1 - \epsilon$ on rings of size n. Then the expected bit complexity of α on rings of size n is $\Omega\left(n \min\left(\log\log n, \log\log\log(\frac{1}{\epsilon})\right)\right)$.*

Proof: By the linear lower bound of Theorem 4.2, assume that n is large and ϵ is small. Let $z = \min\left(\ln n, \log_6 \log_6(\frac{1}{\epsilon})\right)$. A fooling argument is given by following Schema 4.1 with $\chi = (n/75)\log_6 z$ and $I_1 = I_2 = [n, n]$. After applying steps 1 and 2 of the schema, we have Γ of length n with exactly one contender and $h \in \mathcal{H}_a$ such that $\Pr\left(\triangle h \langle \Gamma \rangle h \triangle\right) \geq 6^{-\chi/n-2} = z^{-(1/75)}6^{-2}$.

The core of this proof is more involved than preceding proofs and proceeds in 6 steps. In Step 1, a relatively short subsequence Θ_1 is found in Γ, which will be used to make fine adjustments in the length of the constructed sequence Φ. Step 2 isolates a longer sequence Θ_2 in Γ, which will be used to make large adjustments in the length of Φ. Step 3 performs the main collapsing of Γ. Step 4 pads the collapsed sequence with replications of Θ_2. Step 5 pads further with replications of Θ_1. Step 6 does the final replication to arrive at a contradiction. The length of Θ_2 is crucial to the argument. The need for the other sequence Θ_1 is to correct for slight imprecision in our ability to control the length of Θ_2.

Step 1: Isolate Θ_1. Apply Lemma 3.6 to Γ with $l = l_1 = 1$ and $\tau = \tau_1 = \lceil z^{1/2} \rceil$. Then $\tau l < n$ and $(n \log_6 \tau)/36 \geq n \log_6 z^{1/2}/36 > \chi$. So the conditions of Lemma 3.6 are met, implying $\Gamma = \Psi\Theta_1\Upsilon$ where $1 \leq |\Theta_1| < z^{1/2}$ and, for some history h_1, $\Pr\left(\triangle h \langle \Psi \rangle h_1 \langle \Theta_1 \rangle h_1 \langle \Upsilon \rangle h \triangle\right) \geq z^{-1/2} z^{-1/75} 6^{-2} > z^{-0.9}$. By Property 3.1 each of $\Pr\left(\triangle h \langle \Psi\Theta_1 \rangle h_1\right)$, $\Pr\left(h_1 \langle \Theta_1 \rangle h_1\right)$ and $\Pr\left(h_1 \langle \Theta_1\Upsilon \rangle h \triangle\right)$ are greater than $z^{-0.9}$. Let $d_1 = |\Theta_1|$.

Step 2: Isolate Θ_2. In Step 3, Γ will be collapsed by multiples of $|\Theta_2|$ down to a size close to $n' = \lfloor n/(d_1 + 1) \rfloor$. It is crucial that $|\Theta_2|$ be chosen carefully, so that a length quite close to n' can be achieved in a small number of individual collapsing and splicing steps, which keeps the probabilities sufficiently large.

Let $\lambda(x)$ denote the least common multiple of the integers not exceeding x. Apply Lemma 3.6 to the larger of $\Psi\Theta_1$ and $\Theta_1\Upsilon$, which has length at least $n/2$ and which without loss of generality can be assumed to be $\Theta_1\Upsilon$. Let $\tau = \tau_2 = \lceil z \rceil$ and $l = l_2 = d_1 \left\lceil \frac{n-n'}{d_1 \lambda(z)} \right\rceil$. Then $\tau_2 l_2 < n/2$ because z is much smaller than $\lambda(z)$. Also $(|\Theta_1\Upsilon| \log_6 \tau)/36 \geq (n \log_6 z)/72 > \chi$. So the conditions of Lemma 3.6 are met, implying $\Theta_1\Upsilon = \Xi\Theta_2\Lambda$ where $|\Theta_2| = ml_2$ for some positive integer $m < z$, and for some history h_2, $\Pr\left(h_1 \langle \Xi \rangle h_2 \langle \Theta_2 \rangle h_2 \langle \Lambda \rangle h \triangle\right) > \tau_2^{-1} z^{-0.9} > z^{-2}$. By

Property 3.1 each of $\Pr\left(h_1 \langle \Xi \rangle h_2\right)$, $\Pr\left(h_2 \langle \Theta_2 \rangle h_2\right)$ and $\Pr\left(h_2 \langle \Theta_2 \Lambda \rangle h\triangle\right)$ are greater than z^{-2}. Let $d_2 = |\Theta_2|$. Since $m < z$, m divides $\lambda(z)$, so d_2 divides $\lambda(z)l_2$.

To summarize, there are two subsequences Θ_1 and Θ_2 of Γ, of lengths d_1 and d_2, respectively, where $1 \leq d_1 \leq l_2 \leq d_2 < l_2 z$ and $d_1 < z^{1/2}$. Known divisibility properties are that $d_1 \mid l_2$ and $l_2 \mid d_2$ and $d_2 \mid \lambda(z)l_2$. Sequence $\Gamma = \Psi\Xi\Theta_2\Lambda$.

Step 3: Collapse pieces of Γ. Collapse each of Ψ, Ξ and $\Theta_2\Lambda$ separately to size at most $t^* = \lfloor n'/3 \rfloor$. Leave any segment that is already no longer than t^* unchanged. For each of those that are longer than t^*, apply Lemma 3.7 using the parameters t^*, $l = l_3 = d_2$ and $\tau = \tau_3 = \left\lceil z^{3\sqrt{z}} \right\rceil$. To verify the conditions of Lemma 3.7, notice that $l_3\tau_3 < l_2 z \tau_3 = d_1 z \left\lceil \frac{n}{d_1\lambda(z)} \right\rceil \left\lceil z^{3\sqrt{z}} \right\rceil < z^{2+3\sqrt{z}} \left\lceil \frac{n}{\lambda(z)} \right\rceil$. But $\ln \lambda(z) \sim z$ and $z < \ln n$, so $l_3\tau_3 < \frac{n}{z} + z^{2+3\sqrt{z}} < t^*$. Also notice that $(t^* \log_6 \tau_3)/36 \geq (\lfloor n'/3 \rfloor \log_6(z^{3\sqrt{z}})/36 > \frac{n \log_6 z}{72} \frac{\sqrt{z}}{d_1+1} > \chi$ since $d_1 < \sqrt{z}$.

So the conditions of Lemma 3.7 are met. Let Φ_1, Φ_2 and Φ_3 be the sequences resulting from Ψ, Ξ and $\Theta_2\Lambda$ respectively and combine the results supplied by Lemma 3.7 for each of Φ_1, Φ_2 and Φ_3. Note that each collapsing step removes at least d_2 processors. Thus the total number of steps, summed over Φ_1, Φ_2 and Φ_3, cannot exceed n/d_2. But $n/d_2 \leq n/l_2 \leq n\lambda(z)/(n - n') < 2\lambda(z) < 3^z$, for large z. Let $n'' = |\Phi_1\Phi_2\Phi_3|$. According to Lemma 3.7,

(a) $n' - 3\tau_3 d_2 < n'' < n'$

(b) $n'' \equiv n \pmod{d_2}$ and

(c) $\Pr\left(\triangle h \{\Phi_1\} h_1\right) \cdot \Pr\left(h_1 \{\Phi_2\} h_2\right) \cdot \Pr\left(h_2 \{\Phi_3\} h\triangle\right) \geq \tau_3^{-3-n/d_2} z^{-3}$
$\geq \left\lceil z^{3\sqrt{z}} \right\rceil^{-3-n/d_2} z^{-3} > 6^{-4^z}$.

Step 4: Pad with Θ_2. The object is to pad Φ_2 to form sequence Φ_2' such that $n' - \lambda(z)d_1 < |\Phi_1\Phi_2'\Phi_3| \leq n'$. If $n'' = |\Phi_1\Phi_2\Phi_3|$ is already greater than $n' - \lambda(z)d_1$, simply let $\Phi_2' = \Phi_2$. Otherwise, $n'' \leq n' - \lambda(z)d_1$. Since $(n - n')/\lambda(z) \leq l_2 < (n - n')/\lambda(z) + d_1$, conclude that $n' - \lambda(z)d_1 < n - \lambda(z)l_2 \leq n'$. Hence, it suffices to pad to length exactly $n - \lambda(z)l_2$. Since $n'' \equiv n \pmod{d_2}$, and $\lambda(z)l_2 \equiv 0 \pmod{d_2}$, then $n'' \equiv n - \lambda(z)l_2 \pmod{d_2}$. So there must be a positive integer $k < 3\tau_3$ such that $n'' + kd_2 = n - \lambda(z)l_2$. Let $\Phi_2' = \Phi_2(\Theta_2)^k$.

Since $\Pr\left(h_2\{\Theta_2\}h_2\right) > z^{-2}$ and $\Pr\left(h_1\{\Phi_2\}h_2\right) > 6^{-4^z}$, then $\Pr\left(h_1\{\Phi_2'\}h_2\right) > z^{-6\tau_3}6^{-4^z}$. Let $\Phi_4 = \Phi_1\Phi_2'\Phi_3$. Then $\Pr\left(\triangle h\{\Phi_4\}h\triangle\right) > 6^{-5^z}$. By Property 2.3, Φ_4 must contain a contender. Also, $|\Phi_4| \equiv n \pmod{d_2}$.

Step 5: Pad with Θ_1. Since $|\Phi_4| \equiv n \pmod{d_2}$ and d_1 divides d_2, $|\Phi_4| \equiv n \pmod{d_1}$. Since $n' - \lambda(z)d_1 < |\Phi_4| \leq n'$, conclude that $n - \lambda(z)d_1(d_1+1) - d_1 \leq (d_1+1)|\Phi_4| \leq n$. So there must be a nonnegative integer $K < (\lambda(z)+1)(d_1+1)$ such that $(d_1+1)|\Phi_4| + Kd_1 = n$. Let $\Phi_5 = \Phi_1(\Theta_1)^K\Phi_2'\Phi_3$. Then $\Pr\left(\triangle h\{\Phi_5\}h\triangle\right) > 6^{-5^z}z^{-(\lambda(z)+1)(d_1+1)}$.

Step 6: Replication. Let $\Phi = \Phi_5(\Phi_4)^{d_1}$. Then $|\Phi| = n$, and Φ has more than one contender. Also, $\Pr\left(\triangle h\{\Phi\}h\triangle\right) > 6^{-5^z(d_1+1)}z^{-(\lambda(z)+1)(d_1+1)} > 6^{-6^z}$, since $d_1 < \sqrt{z}$, $\lambda(z) < 3^z$ and z is large. Recall that $z \leq \log_6\log_6(\frac{1}{\epsilon})$. Therefore $\Pr\left(\triangle h\{\Phi\}h\triangle\right) > \epsilon$. The argument is completed by appealing to Property 2.5. ■

Corollary 7.5 *The complexity of any nondistributively terminating Las Vegas algorithm that solves Weak Solitude Verification with confidence at least $1 - 1/2^n$ on rings of size n is $\Omega(n \log\log n)$ bits.*

8 Concluding Remarks

The lower bounds of this paper apply to Monte Carlo algorithms, that is, to algorithms that err with small probability (at most ϵ). The matching upper bounds [5] are also achieved with Monte Carlo algorithms. A stricter requirement is to insist on Las Vegas algorithms, that is, algorithms that must be correct upon termination and must terminate with probability one.

The elaborate model used in this paper can be substantially simplified when the results are required only for Las Vegas algorithms. The Monte Carlo proofs all proceed by deriving a ring of processes that errs with too high a probability from the assumption of a correct ring of processes that has a low expected complexity. Since Las Vegas algorithms must have zero probability of error, the proofs are relieved of the burden of maintaining bounds on the probability of error of the new ring at every stage of the construction. Furthermore, there is no need to start with a probability space of computations for a ring of processes; rather,

the manipulations are applied to a single computation. It is only necessary to derive some possible erroneous computation from the assumption that there is some computation with low complexity.

This simplified perspective leads to stronger statements of theorems as well as less complicated proofs. The results apply to the *best case* complexity of a correct algorithm, rather than to the expected case. Furthermore, the results hold even if, with high probability, the algorithm deadlocks or fails to terminate. All that is required of the algorithm is that it is correct if and when it terminates.

This weak requirement for correctness together with the interpretation of complexity as the best case complexity reflect the automaton-theoretic notion of nondeterminism. Thus we call such algorithms *nondeterministic (distributed)* algorithms. The details of the general nondeterministic model for communication complexity of distributed algorithms can be found elsewhere [4]. In this model, the results of Section 7 become:

Theorem 8.1 *The complexity of any distributively accepting nondeterministic algorithm that solves Weak Solitude Verification for rings in* $\mathcal{R}_{[n,n]}$ *is* $\Omega\left(n\sqrt{\log n}\right)$ *bits.*

Theorem 8.2 *The complexity of any nondistributively terminating nondeterministic algorithm that solves Weak Solitude Verification for rings in* $\mathcal{R}_{[n,n]}$ *is* $\Omega\left(n \log\log n\right)$ *bits.*

The lower bound of Theorem 8.1 can be extended to a lower bound of $\Omega\left(n\sqrt{\log n}\right)$ bits for general non-constant function evaluation, and applies to any nondeterministic distributed algorithm for a ring of size n [4]. This bound is tight; there is a cyclic non-constant Boolean function that can be computed in $O\left(n\sqrt{\log n}\right)$ expected bits on an asynchronous anonymous unidirectional ring. The Monte Carlo result of Theorem 7.1 can similarly be extended to Monte Carlo non-constant function evaluation. However, the bound of Theorem 7.3 is not known to hold for non-constant function evaluation with distributed termination. There is a gap in the known upper and lower bounds for Monte Carlo non-constant function evaluation.

The generalization from Las Vegas to nondeterministic algorithms can be mimicked to derive a more general notion than the strict Monte Carlo model of an algorithm. As described

in subsection 2.3, this more general model, which has both probabilistic and nondeterministic properties, is the one for which the lower bounds of this paper actually hold.

References

[1] K. Abrahamson, A. Adler, R. Gelbart, L. Higham, and D. Kirkpatrick. The bit complexity of probabilistic leader election on a unidirectional ring. In *Distributed Algorithms on Graphs*, pages 1–12. Carleton University Press, 1986. Proc. 1st International Workshop on Distributed Algorithms.

[2] K. Abrahamson, A. Adler, R. Gelbart, L. Higham, and D. Kirkpatrick. The bit complexity of randomized leader election on a ring. *SIAM Journal on Computing*, 18(1):12–29, 1989.

[3] K. Abrahamson, A. Adler, L. Higham, and D. Kirkpatrick. Probabilistic evaluation of common functions on rings of known size. Technical Report 88-15, University of British Columbia, 1988.

[4] K. Abrahamson, A. Adler, L. Higham, and D. Kirkpatrick. Randomized function evaluation on a ring. *Distributed Computing*, 3(3):107–117, 1989.

[5] K. Abrahamson, A. Adler, L. Higham, and D. Kirkpatrick. Optimal algorithms for probabilistic solitude detection on anonymous rings. Technical Report TR 90-3, University of British Columbia, 1990.

[6] Y. Afek. *Distributed Algorithms for Election in Unidirectional and Complete Networks.* PhD thesis, University of California at Los Angeles, 1985.

[7] D. Angluin. Local and global properties in networks of processors. In *Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing*, pages 82–93, 1980.

[8] T. M. Apostol. *Introduction to Analytic Number Theory.* Springer-Verlag, New York, 1976.

[9] H. Attiya, N. Santoro, and S. Zaks. From rings to complete graphs — $\theta(n \log n)$ to $\theta(n)$ distributed leader election. Technical Report SCS-TR-109, Carleton University, 1987.

[10] H. Attiya and M. Snir. Better computing on the anonymous ring. In *Proc. Aegean Workshop on Computing*, pages 329–338, 1988.

[11] H. Attiya, M. Snir, and M. Warmuth. Computing on an anonymous ring. In *Proc. 4th Annual ACM Symp. on Principles of Distributed Computing*, pages 196–203, 1985.

[12] H. L. Bodlaender. New lower bound techniques for distributed leader finding and other problems on rings of processors. Technical Report RUU-CS-88-18, Rijksuniversiteit Utrecht, 1988.

[13] D. Dolev, M. Klawe, and M. Rodeh. An $O(n \log n)$ unidirectional distributed algorithm for extrema finding in a circle. *J. Algorithms*, 3(3):245–260, 1982.

[14] P. Duris and Z. Galil. Two lower bounds in asynchronous distributed computation (preliminary version). In *Proc. 28nd Annual Symp. on Foundations of Comput. Sci.*, pages 326–330, 1987.

[15] G. Fredrickson and N. Lynch. The impact of synchronous communication on the problem of electing a leader in a ring. In *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, pages 493–503, 1984.

[16] A. Itai and M. Rodeh. Symmetry breaking in distributed networks. In *Proc. 22nd Annual Symp. on Foundations of Comput. Sci.*, pages 150–158, 1981.

[17] J. Pachl. A lower bound for probabilistic distributed algorithms. Technical Report CS-85-25, University of Waterloo, Waterloo, Ontario, 1985.

[18] J. Pachl, E. Korach, and D. Rotem. Lower bounds for distributed maximum finding. *J. Assoc. Comput. Mach.*, 31(4):905–918, 1984.

[19] G. Peterson. An $O(n \log n)$ algorithm for the circular extrema problem. *ACM Trans. on Prog. Lang. and Systems*, 4(4):758–752, 1982.