Information and Computation 303 (2025) 105265

Contents lists available at ScienceDirect

Information and Computation

journal homepage: www.elsevier.com/locate/yinco

Efficient assignment of identities in anonymous populations ☆

Leszek Gąsieniec [a], Jesper Jansson [b], Christos Levcopoulos [c,*], Andrzej Lingas [c,*]

[a] Department of Computer Science, University of Liverpool, Street, L69 38X, UK
[b] Graduate School of Informatics, Kyoto University, Kyoto, Japan
[c] Department of Computer Science, Lund University, Lund, Sweden

ARTICLE INFO

Article history:
Received 28 April 2024
Received in revised form 21 December 2024
Accepted 1 January 2025
Available online 8 January 2025

ABSTRACT

We consider the fundamental problem of assigning distinct labels to agents in the probabilistic model of population protocols. Our protocols operate under the assumption that the size n of the population is embedded in the transition function. W.h.p. (with high probability), they are silent, i.e., eventually each agent reaches its final state and remains in it forever, and they are safe, i.e., never change a label that has already been assigned to an agent. We provide efficient protocols for this problem complemented with tight lower bounds. Our fast labeling protocol uses only $O((n\log n)/\varepsilon)$ interactions w.h.p., $(2+\varepsilon)n + O(n^a)$ states, and the label range $[1,(1+\varepsilon)n]$, where $1 \geq \varepsilon > 0$ and $0 < a < 1$, while our nearly state-optimal protocol uses only $n + 5\sqrt{n} + O(\log\log n)$ states, the label range $[1,n]$, and w.h.p., $O(n^3)$ interactions.

1. Introduction

The problem of assigning unique identifiers to entities in distributed systems is one of the core problems related to network integrity, and a solution to this problem is often an important preprocessing step for more complex distributed algorithms. The tighter the range that the identifiers are drawn from, the harder the assignment problem becomes [1,2].

In this article, we assume the probabilistic population protocol model and study the problem of assigning distinct identifiers, which we refer to as *labels*, to all agents. In the literature [1,16,9], the problem is also known as *naming* or *renaming*, and when the label range is equal to the number of agents, as *ranking* or *tight naming/renaming*. The population protocol model was originally intended to model large systems of agents with limited resources (state space) [4]. In this model, the agents are prompted to interact with one another towards a solution of a shared task. The execution of a protocol in this model is a sequence of pairwise interactions between randomly chosen agents. During an interaction, each of the two agents, the *initiator* and the *responder* (the asymmetry assumed in [4]), updates its state in response to the observed state of the other agent according to the predefined (global) transition function. For the problem of assigning unique labels to the agents (the unique labeling problem), we assume that all agents are initially in the same state. The number of agent states used by a population protocol is its *state complexity* while the number of interactions of the protocol (in expectation or with high probability) required to achieve a desired output configuration is its *time complexity*. For more details about the

☆ This article is an extended version of the paper that appeared in Proceedings of the 25th International Conference on Principles of Distributed Systems (OPODIS 2021), *LIPIcs*, Article No. 12, pp. 12:1–12:21, 2021.

* Corresponding author.
E-mail addresses: L.A.Gasieniec@liverpool.ac.uk (L. Gąsieniec), jj@i.kyoto-u.ac.jp (J. Jansson), Christos.Levcopoulos@cs.lth.se (C. Levcopoulos), Andrzej.Lingas@cs.lth.se (A. Lingas).

https://doi.org/10.1016/j.ic.2025.105265

computational model of population protocols, see Section 2. Also, for a discussion on applications of labeled populations, see the final remarks in Section 6.

Our protocols for the unique labeling problem assume that the number n of agents is known in advance. They will also work if only an upper bound on the number of agents is known to the agents; in fact, in this case, the problem becomes easier as the range from which the labels are drawn is larger. It is natural to impose a limit on n since without it, there is no limit on the number of states to be used. Indeed, many existing population protocols such as [14,15] rely on knowledge of n.

Our labeling protocols include a preprocessing step to elect a *leader*, i.e., an agent singled out from the population, which improves coordination of more complex tasks and processes. For some examples of leader-based computation see [5].

In the unique labeling problem studied here, the number of utilized states needs to reflect the number of agents n. Also, $\Omega(n \log n)$ is a natural lower bound on the expected number of interactions required to solve not only the labeling problem but any nontrivial problem by a population protocol. The reason is that $\Omega(n \log n)$ interactions are needed to achieve a positive constant probability that each agent is involved in at least one interaction [12].

Perhaps the simplest protocol for unique labeling in population networks (named Protocol 1) is as follows [15] (cf. [13] and Table 3). Initially, all agents hold the label 1, which is equivalent to all agents being in state 1. In due course, whenever two agents with the same label i interact, the responder updates its own label to $i + 1$. The advantage of this simple protocol is that it does not need any knowledge of the population size n and it utilizes only n states and assigns labels from the smallest possible range $[1, n]$, where for $0 \leq p \leq q$, $[p, q]$ denotes the set $\{p, p + 1, \ldots, q\}$ of integers. The severe disadvantage is that it needs at least a cubic in n number of interactions to achieve the configuration in which the agents have distinct labels. This is because getting rid of the last multiple label i, for all $i = 1, \ldots n-1$, requires a quadratic number of interactions in expectation.

In the following two examples of protocols for unique labeling (named Protocol 2 and 3), we assume that the population size n is embedded in the transition function. Such protocols are commonly used and known as *non-uniform* protocols [3].

The first solution (Protocol 2) assumes that one agent has already been designated as the leader [5]. The protocol then lets the leader assign the labels $n, n-1, ..., 2$ in this order to the other unlabeled agents as it encounters them and finally lets it assign the label 1 to itself. The protocol uses only $2n - 1$ states (n states utilized by the leader and $n - 1$ states by the other agents) and assigns unique labels in the smallest possible range $[1, n]$ to the n agents.

Unfortunately, this simple protocol requires $\Omega(n^2 \log n)$ interactions because as more agents get their labels, interactions between the leader and agents without labels become less likely. The probability of such an encounter drops from $\Omega(\frac{1}{n})$ at the beginning to $O(\frac{1}{n^2})$ at the end of the process.

By using randomization, we obtain a much faster solution (Protocol 3) as follows. Each agent picks a number in $[1, n^3]$ uniformly at random as its label. The probability that a given pair of agents gets the same label is only $\frac{1}{n^3}$. Therefore, this protocol assigns unique labels to the agents with probability at least $1 - \frac{1}{n}$. It requires only $O(n \log n)$ interactions w.h.p. (with high probability)[1] [20]. The drawback is that it uses $O(n^3)$ states and the large range $[1, n^3]$. This method also needs a large number of independent random bits for each agent.

Besides the efficiency and population size aspects, there are other deep differences between the three examples of labeling protocols. An agent in Protocol 1 does not always know whether there currently exists another agent with the same label as itself. In Protocol 2, this deficiency cannot happen since labeled agents always have different labels. In Protocol 3, two agents may share the same label, but it takes place with a small probability.

The labeling protocols presented in this paper are *silent* and *safe*. We say that a (not necessarily labeling) protocol is silent if eventually each agent reaches its final state and remains in it forever. We say that a labeling protocol is safe if it never updates the label assigned to any single agent during an execution of the protocol. Importantly, there are dedicated label states and none of the agents is in such a state at the beginning of the labeling process. While the concept of a silent population protocol is well established in the literature [14,17], the concept of a safe labeling protocol is new. The latter property is useful in situations where the protocol producing a valid labeling has to be terminated before completion due to some unexpected emergency or running out of time.

Observe that among the three examples of labeling protocols, only the Protocols 2 and 3 are both silent and safe. Protocol 1 is silent [14] but not safe.

1.1. Our contributions

The primary objective of this paper is to provide efficient labeling protocols complemented with tight lower bounds in terms of the number of states utilized by agents, the size of the range from which the labels are drawn, and the number of interactions required in expectation or w.h.p. by our solutions.

In particular, we provide positive answers to the two following natural questions under the assumption that the number n of agents is known at the beginning.

[1] That is, with probability at least $1 - \frac{1}{n^\alpha}$, where α is any constant not less than 1 and n is the number of agents.

Table 1
Upper bounds on the number of states, the number of interactions, and the range used by the safe labeling protocols presented in this paper. In Theorems 1 and 4, ε is $\Omega(n^{-1})$.

Theorem	# states	# interactions	Range
1, $\varepsilon = 1$	$O(n)$	$O(n \log n)$ w.h.p.	$[1, 2n]$
1	$(2+\varepsilon)n + O(n^a)$, any $a < 1$	$O((n \log n)/\varepsilon)$ w.h.p.	$[1, (1+\varepsilon)n]$
2	$n + 5 \cdot \sqrt{n} + O(\log \log n)$	$O(n^3)$ w.h.p.	$[1, n]$
4	$n(1 + 9\sqrt{\varepsilon}) + O(\log \log n)$	$O(n^2/\varepsilon)$ w.h.p.	$[1, n]$

Table 2
Lower bounds on the number of states or/and the number of interactions required by labeling protocols. [1] Any labeling protocol that is capable of producing a valid labeling. [2] The silent protocol in Theorem 5 is assumed to produce a valid labeling and to be safe with probability greater than $1 - \frac{1}{n}$. [3] The silent protocol in Theorem 6 is assumed to produce a valid labeling and to be safe with probability 1.

Protocol type	# states	# interactions	Theorem
any[1]	n	$\Omega(n \log n)$ w.h.p.	Remark 2
silent, safe[2]	$n + \sqrt{\frac{n-1}{2}} - 1$	-	5
silent, safe[3], $n + t < 2n$ states	-	expected $\frac{n(n-1)}{2t+1}$	6
pool, range $[1, n+r]$	-	expected $\frac{n(n-1)}{2(r+1)}$	7

1. Can one design a protocol for the labeling problem using the asymptotically optimal number $O(n \log n)$ of interactions w.h.p., utilizing the asymptotically optimal number $O(n)$ of states and an asymptotically minimal label range of size $O(n)$?
2. Can one design a silent and safe protocol for the labeling problem utilizing substantially fewer states than $2n$ and possibly the minimal label range $[1, n]$?

We first present a silent w.h.p. and safe labeling protocol that draws labels from the range $[1, (1+\varepsilon)n]$, where $1 \geq \varepsilon > 0$. It requires $O((n \log n)/\varepsilon)$ interactions in order to complete the assignment of distinct labels from the aforementioned range to the n agents, w.h.p. The protocol uses $(2+\varepsilon)n + O(n^a)$ states, for any positive constant $a < 1$. In particular, for $\varepsilon = 1$, w.h.p. the protocol requires the asymptotically optimal number $O(n \log n)$ of interactions to assign distinct labels from the range $[1, 2n]$. For $\varepsilon = 1$, the protocol uses also the asymptotically optimal number $O(n)$ of states.

Furthermore, we consider a natural class of population protocols for the unique labeling problem that we call *pool protocols* which includes our fast labeling protocols. We show that for any protocol in this class that picks the labels from the range $[1, n+r]$, the expected number of interactions to label the agents correctly is at least $\frac{n(n-1)}{2(r+1)}$.

Next, we provide a labeling protocol which uses only $n + 5\sqrt{n} + O(\log \log n)$ states and the label range $[1, n]$. The number of interactions required by the protocol is $O(n^3)$ both in expectation and w.h.p. Once a unique leader is elected, it produces a valid labeling and is silent and safe. On the other hand, we show that (even if a unique leader is given in advance) any silent protocol that produces a valid labeling and is safe with probability larger than $1 - \frac{1}{n}$ must use at least $n + \sqrt{\frac{n-1}{2}} - 1$ states. It follows that our protocol is nearly state-optimal. In addition, we present a variant of this protocol which uses $n(1 + 9\sqrt{\varepsilon}) + O(\log \log n)$ states. The number of interactions required by this variant is $O(n^2/\varepsilon)$, where $\varepsilon \geq n^{-1}$, both in expectation and w.h.p. On the other hand, we show that for any silent and safe labeling protocol utilizing $n + t < 2n$ states, the expected number of interactions required to achieve a valid labeling is at least $\frac{n(n-1)}{2t+1}$.

All our labeling protocols include a preprocessing step for electing a unique leader and assume the knowledge of the population size n. However, our nearly state-optimal protocol (Single-Cycle protocol) can be made independent of n (see Section 4).

Our results are summarized in Tables 1 and 2.

1.2. Main ideas of our protocols

Our first fast labeling protocol essentially operates as follows for $\varepsilon = 1$. The leader initially has the label 1 and the interval of labels $[2, n]$. During the first phase, encountered unlabeled agents also receive a label and an interval of labels that they can distribute among other agents. When a labeled agent with a non-empty interval interacts with an unlabeled agent, the latter agent receives a label from the interval. If the remaining part of the interval has length ≥ 2 then part of it is given to the latter agent. After $O(n \log n)$ interactions, a sufficiently large fraction of agents is labeled and has no additional labels to distribute w.h.p. The leader counts its own interactions up to $O(\log n)$ in order to trigger the second phase by broadcasting. In the second phase, an agent with a label x and without a non-empty interval can distribute one

Table 3
Upper bounds on the number of interactions, the number of states, and the range used by the previously published labeling protocols. In the first column "un" and "kn" stand for "unknown" and "known", respectively. In case of the self-stabilizing labeling protocols in [14], the "safe" property can eventually hold only for their initialized versions.

n	# states	# interactions	Range	Properties	Ref.
un	n	$O(n^3)$ w.h.p.	$[1, n]$	silent	[15]
un	$n^{O(1)}$	$O(n \log n \log\log n)$ w.h.p.	$[1, n^{O(1)}]$	silent	[19]
kn	$O(n)$	$O(n^2)$ expected	$[1, n]$	silent, safe	[14]
kn	$2^{O(n^{\log n} \log n)}$	$O(n \log n)$ w.h.p.	$[1, n]$	safe	[14]

additional label $x + n$. During the first phase, the labels from the range $[1, n]$ are rapidly distributed among the agents. In the second phase, the unlabeled agents still have a high chance of communicating with an agent capable of distributing a label. Our general fast labeling protocol, where $1 \geq \varepsilon > 0$, restricts the set of agents that may distribute the labels $x + n$ in the second phase to those having labels in the range $[1, n\varepsilon]$.

The main idea of the nearly state-optimal labeling protocol (Single-Cycle protocol) is to use the leader and an auxiliary leader nominated by the leader to jointly dispense the n labels among the remaining free agents without assigned labels. The leader dispenses the first part of each individual label, while the auxiliary leader provides the second part. When a free agent receives both partial labels, it combines them into its individual label and then informs the leaders about this. The two leaders operate in two embedded loops. For each of roughly $\sqrt{n}$ partial labels of the leader, the auxiliary leader makes a full round of dispensing its roughly $\sqrt{n}$ partial labels. In the generalized version of the protocol (k-Cycle protocol), the process is partially parallelized by letting the leader form k pairs of dispensers, where each pair labels agents in a distinct range of size n/k.

1.3. Related work

There are several papers concerning labeling of processing units (also known as renaming or naming) in different communication models [16]. E.g., Berenbrink et al. [9] present efficient algorithms for so-called loose and tight renaming in shared memory systems, improving on or providing alternative algorithms to the earlier algorithms by Alistarh et al. [1,2]. Loose renaming, where the label space is larger than the number of units, was shown to admit substantially faster algorithms than tight renaming [1,9].

The problem of assigning unique labels to agents has been studied in the model of population protocols by Beauquier et al. [7,13]. In [13], the emphasis is on estimating the minimum number of states which are required by non-safe protocols. In [7], the authors provide among other things a generalization of a leader election protocol to include a distribution of m labels among n agents, where $m \leq n$. In the special case of $m = n$, all agents will receive unique labels. No analysis on the number of interactions required by the protocol is provided in [7]. Their focus is on the feasibility of the solution, i.e., that the process eventually stabilizes to the final configuration. Their protocol seems inefficient in the state space aspect as it needs many states/bits to keep track of all the labels.

Doty et al. considered the labeling problem in [19] and presented a subroutine named "UniqueID" for it based on the technique of traversing a labeled binary tree and associating agents with nodes in the tree. The subroutine requires $O(n \log n \log\log n)$ interactions.

The labeling problem has also been studied in the context of self-stabilizing protocols where the agents start in arbitrary (not predefined) states; see [14,15]. In [15], Cai et al. propose a solution that we reviewed as one of the examples in Section 1, where it was referred to as Protocol 1. In [14], Burman et al. study both slow and fast labeling protocols, with the latter utilizing an exponential number of states. The protocols in both papers require exact knowledge of n. The work by [14] focuses on self-stabilizing protocols which, by definition, cannot be safe. To provide a meaningful comparison, we consider our protocols in relation to the initialized versions of the protocols in [14]. For instance, the leader-driven initialized (silent) ranking protocol in [14] (see Lemma 4.1) requires $O(n^2)$ interactions, uses $O(n)$ states, and is safe. An analogous variant of the fast ranking protocol from [14] requiring $O(n \log n)$ interactions and an exponential number of states is also safe but not silent. Table 3 summarizes the known labeling protocols.

The most closely related problem studied extensively in the literature is that of counting the population size, i.e., determining the number of agents. This topic has recently been investigated by Aspnes et al. in [6] and Berenbrink et al. in [12]. In this paper, we assume that the population size is initially known. Alternatively, it can be computed by using the protocol counting the exact population size described in [12]. The aforementioned protocol computes the population size in $O(n \log n)$ interactions w.h.p., using $\tilde{O}(n)$ states. Another option is to employ the protocol computing the approximate population size, presented in [12]. The latter protocol requires $O(n \log^2 n)$ interactions to compute the approximate size w.h.p., using only a poly-logarithmic number of states. For references to earlier papers on protocols for counting or estimating the population size, especially the papers that introduced the counting problem and that include the original algorithms on which the improved algorithms of Berenbrink et al. are based, see [12].

Our protocols include a preprocessing step for electing a unique leader and its synchronization with the proper labeling protocol (see Subsection 2.4). There is a vast literature on population protocols for leader election [11,18,20,21]. For our

purposes, the most relevant is the protocol that elects a unique leader from a population of n agents using $O(n \log n)$ interactions and $O(n^a)$ many states, for any positive constant $a < 1$, w.h.p. This result is implicit in [12,18] (see Subsection 2.4). The newest results elaborate on state-optimal leader election protocols utilizing $O(\log \log n)$ states. These include the fastest possible protocol [11] requiring $O(n \log n)$ interactions in expectation, and a slightly slower protocol [21] requiring $O(n \log^2 n)$ interactions with high probability.

Our population protocols for unique labeling also use the known population protocol for (one-way) epidemics, or broadcasting. It completes spreading a message in $\Theta(n \log n)$ interactions w.h.p. and uses only two states [20] (see also Fact 5 below).

1.4. Organization of the paper

In the next section, we provide some basic facts about probabilistic inequalities and population protocols for broadcasting, counting, and leader election. In Section 3, we present our fast, silent w.h.p., and safe protocol for unique labeling in the range $[1, n(1+\varepsilon)]$. Section 4 is devoted to the nearly state-optimal, silent, and safe protocol with the label range $[1, n]$ and its variation. Section 5 derives lower bounds on the number of states or the number of interactions required by silent, safe, and pool protocols for unique labeling. We conclude with final remarks.

2. Preliminaries

2.1. The computational model of population protocols

Assume that a population of n agents is given. The agents can pairwise interact to change their states and in this way perform a computation. A population protocol can be formally specified by providing a set Q of possible states, a set O of possible outputs, a transition function $\delta : Q \times Q \to Q \times Q$, and an output function $o : Q \to O$. The current state $q \in Q$ of an agent is updated during its interactions. Consequently, the current output $o(q)$ of the agent also becomes updated during interactions. The overall current state of the set of n agents is represented by a vector in Q^n describing the current states of the agents. A computation of a population protocol is specified by a sequence of pairwise interactions between agents. At each time step, an ordered pair of agents is selected for interaction by a probabilistic scheduler independently and uniformly at random. The first agent in the selected pair is referred to as the initiator while the second one is the responder. During the interaction, the states of the two agents are updated according the transition function δ. The sequence of pairwise interactions among selected agents constitutes a *run* of the protocol, and the computation of the protocol specified by the sequence is the *execution* of the run. The *state complexity* of a population protocol is determined by the number $|Q|$ of states used in the protocol.

We can specify a problem to be solved by a population protocol by providing the set of input configurations, the set O of possible outputs, and the desired output configurations for given input configurations. For the unique labeling problem, all agents are initially in the same state q_0. The set O is just the set of positive integers. A desired configuration is when all agents output their distinct labels.

The *stabilization time* of an execution of a protocol is the number of interactions (in expectation or w.h.p.) until the states of agents form a desired configuration from which no sequence of pairwise interactions can lead to a configuration outside the set of desired configurations. The *time complexity* of a population protocol for a given problem is the number of interactions of the protocol (in expectation or w.h.p.) required to achieve a desired output configuration when starting from the input configuration.

2.2. Probabilistic bounds

Fact 1. *(The union bound) For a sequence $A_1, A_2, ..., A_r$ of events, $Prob(A_1 \cup A_2 \cup ... \cup A_r) \leq \sum_{i=1}^{r} Prob(A_i)$.*

Fact 2. *(multiplicative Chernoff bounds) Suppose $X_1, ..., X_n$ are independent random variables taking values in $\{0, 1\}$. Let X denote their sum and let $\mu = E[X]$ denote the sum's expected value.*

Then, for any $\delta \in [0, 1]$, $Prob(X \leq (1-\delta)\mu) \leq e^{-\frac{\delta^2 \mu}{2}}$ holds.

Similarly, for any $\delta \geq 0$, $Prob(X \geq (1+\delta)\mu) \leq e^{-\frac{\delta^2 \mu}{2+\delta}}$ holds.

Fact 3. *(Chernoff-Janson bound, Theorem 2.1 in [23]) Let $X = \sum_{i=1}^{n} X_i$, where $n \geq 1$ and X_i are independent geometric variables such that the probability that $X_i = k$ is equal to $p_i(1-p_i)^{k-1}$ for $p_i \in (0, 1]$ and $k = 1, 2, ...$ Let p^* be the minimum of $p_1, ..., p_n$, and let μ be the expected value of X. Then, for any $\lambda \geq 1$, the probability that $X \geq \lambda\mu$ does not exceed $e^{-p^*\mu(\lambda - 1 - \ln \lambda)}$.*

Fact 4. *[20] For all $C > 0$ and $0 < \delta < 1$, during $Cn \log n$ interactions, with probability at least $1 - n^{-O(\delta^2 C)}$, each agent participates in at least $2C(1-\delta)\log n$ and at most $2C(1+\delta)\log n$ interactions.*

2.3. Broadcasting and counting

We shall refer to the following broadcast process which can be completed during $\Theta(n \log n)$ interactions w.h.p. Each agent is either in a state of M-type (got the message) or in a state of ¬M-type (has not got any message). Whenever an agent in a state of M-type interacts with an agent in a state of ¬M-type, the latter changes its state to a state of M-type (gets the message). The process starts when the first agent gets the message and completes when all agents have the message.

Fact 5. *[20] There exists a constant c_0, such that for $c \geq c_0$, the broadcast process completes in $cn \log n$ interactions with probability at least $1 - n^{-\Theta(c)}$.*

Berenbrink et al. [12] obtained among other things the following result on counting the population size, i.e., the number of agents.

Fact 6. *There exists a protocol for a population of an unknown number n of agents such that w.h.p., after $O(n \log n)$ interactions the protocol stabilizes and each agent holds the exact population size. The protocol uses $\tilde{O}(n)$ states.*

2.4. Leader election preprocessing

In our fast protocols, we employ a preprocessing step to compute the unique leader utilizing $O(n \log n)$ interactions with high probability. The preprocessing uses $O(n^a)$ states, for any positive constant $a > 0$. It is a state-efficient implementation of the following straightforward leader election protocol, analogous to the third example of a simple labeling protocol (Protocol 3) presented in the introduction.

Each agent selects $\lceil 3 \log n \rceil$ random bits (representing random numbers in the range $[0, 2^{\lceil 3 \log n \rceil}]$). W.h.p., there exists a unique agent holding the largest of the drawn numbers. This agent is selected as the leader by using broadcasting. Note that each agent can draw random bits during consecutive interactions: 1 when acting as the initiator, and 0 when acting as the responder.

To limit the number of states to $O(n^a)$, the protocol is implemented in $\lceil 3/a \rceil$ rounds. In each round, every agent draws only $\lceil a \log n \rceil$ bits due to the state restriction and the agents reach consensus (through the broadcast process from Fact 5) on the largest number drawn in that round w.h.p. After each round, the set of the remaining candidates for the leader is narrowed down to those that drew largest random numbers in all previous rounds. This way, the leader which would be picked if we drew $\lceil 3 \log n \rceil$ bits in one attempt survives all the rounds.

We still need to show that it is possible to provide a proper separation into the $\lceil 3/a \rceil$ rounds w.h.p. From the various clock mechanisms in the literature, we select one proposed by Berenbrink et al. [10] to clock the aforementioned rounds. Within this mechanism, we initialize the average AVE of values stored in all agent interaction counters to 0. Following each sequence of n consecutive interactions, AVE is increased by 1.

Let MIN, MAX be the smallest and largest current values stored in the counters, both initially set to 0. Importantly, for each agent its (interaction) counter is set to 1 after its first interaction. The clocking mechanism has the following properties.

Any time beyond the first $\Theta(n \log n)$ interactions, the following inequalities and equalities hold w.h.p.

1. $MAX - AVE < \log \log n$
2. $AVE - MIN = O(\log n)$

Thus, all counters are concentrated w.h.p., and at any given time $MAX - MIN < O(\log n)$.

Recall that during each round, every remaining leader candidate must (A) collect $\lceil a \log n \rceil$ random bits and then (B) distribute those bits (in the form of a single $O(n^a)$ number) to find the maximum. Both tasks (A) and (B) require $\Theta(n \log n)$ interactions each.

As a consequence, we need to create rounds long enough to accommodate tasks (A) and (B) separately, and to have silent breaks between actions in (A) and (B) (including those coming from different rounds). We need silent periods as the counters may differ by $O(\log n)$ and we cannot execute actions from (B) while some agents still execute actions from (A), and vice versa. For a similar reason, the subrounds have to be long enough so when some agent joins this subround late, sufficiently many interactions are available to complete the relevant task.

Finally, note that if an agent a_i (according to its counter) is still in the silent period but it encounters another agent a_j that already executes the next task, a_i can begin executing the next task earlier.

Considering the design of the phases, the fact that their number is constant, and the aforementioned properties of the clocking mechanism that hold w.h.p., we conclude that the agent that drew the largest (composite) number is selected as the leader w.h.p. like in the straightforward leader election protocol.

Therefore, the following fact, implicit in [12] and Section 3.3 in [18], holds.

Fact 7. *[12,18] There exists a preprocessing protocol that elects a unique leader from a population of n agents using $O(n \log n)$ interactions w.h.p. and $O(n^a)$ many states, for any positive constant $a < 1$.*

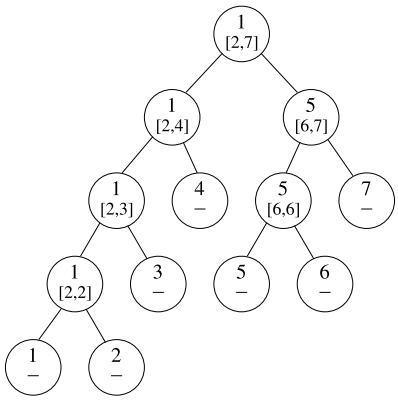

Fig. 1. An example of the partition tree of the start interval [1,7].

For our nearly state-optimal, slower protocols, we use the following fact established in [21].

Fact 8. *[21] There exists a preprocessing protocol that elects a unique leader from a population of n agents using $O(n \log^2 n)$ interactions w.h.p. and $O(\log \log n)$ many states.*

In both cases, the leader election preprocessing step is synchronized with our proper labeling protocols using $O(n \log n)$ interactions w.h.p. as follows. When the counter of the leader reaches an appropriate constant multiple of $\log n$, so the number of interactions required by the preprocessing has taken place w.h.p. by Fact 4, the leader initiates the labeling process. All the preprocessing states are identified with an initial state for the labeling protocol. The initial state has an L-component which is set to 1 in case of the leader and to 0 otherwise.

3. Labeling with asymptotically optimal number of interactions, nearly optimal number of states and range

In this section, we present a silent w.h.p. and a safe labeling protocol that assigns unique labels from the range $[1, (1+\varepsilon)n]$, where $0 < \varepsilon \leq 1$, to n agents. The protocol requires $O((n \log n)/\varepsilon)$ interactions w.h.p. We assume that $0 < \varepsilon \leq 1$, and ε does not have to be a constant; it can even be as small as $O(n^{-1})$. The protocol utilizes only $(2+\varepsilon)n + O(n^a)$ states, where a is any positive constant less than 1.

3.1. Informal description

The overall concept of the protocol is explained in the first paragraph of Subsection 1.2. The protocol consists of two main phases preceded by the leader election preprocessing described in Subsection 2.4. The preprocessing involves $O(n \log n)$ interactions w.h.p. and $O(n^a)$ states, where a is any positive constant less than 1.

The idea of the first phase is reminiscent of load balancing [12]. The difference is that the tokens (in our case labels and interval sub-ranges) are distinct. At the start of this phase, the leader assigns the label 1 and also temporarily the interval $[2, n]$ to itself. Subsequently, when two agents interact, one with a label and a temporarily assigned interval $[q, r]$ where $r > q$ and the other without a label, the former agent shrinks its interval to $[q, \lfloor \frac{q+r}{2} \rfloor]$ and gives away (and assigns) the label $\lfloor \frac{q+r}{2} \rfloor + 1$ and if $\lfloor \frac{q+r}{2} \rfloor + 2 \leq r$ also the sub-interval $[\lfloor \frac{q+r}{2} \rfloor + 2, r]$ to the latter agent. Additionally, when an agent possessing a label and a temporarily assigned singleton interval $[q, q]$ interacts with an agent lacking a label, the former agent nullifies the interval and assigns the label q to the latter agent. In all other cases, interactions do not affect the states of the agents. By the (complete) *partition tree* (of the start interval $[1, n]$), we mean the final binary tree with leaves assigned distinct labels within the range $[1, n]$, resulting from the aforementioned process; see Fig. 1. Note that the partition tree is unique and has a logarithmic depth. During the first phase, a (binary) subtree of the partition tree (with the same root) is formed. Observe that when an agent at an intermediate node of the tree interacts with an unlabeled agent, the former agent migrates to the left child of the node, while the latter agent lands at the right child of the node.

In the second phase, suppose that an agent A interacts with an agent B and the following conditions are satisfied:
(i) A has a label i such that $i \leq n\varepsilon$;
(ii) A is located at a leaf of the partition tree;
(iii) B does not have a label;
(iv) at the leaf, A has not previously interacted with any unlabeled agents.
Then, B is assigned the label $i + n$. Otherwise, the agents interact as in the first phase.

To put the two phases together, the leader agent counts its interactions. When the number of interactions of the leader in the first phase surpasses an appropriate multiple of $(\log n)/\varepsilon$ then the following happens by Fact 4. The total number of interactions in the first phase meets a lower bound of the form $(cn \log n)/\varepsilon$ required to guarantee that the number

of unlabeled agents falls below $n\varepsilon/4$ w.h.p. (see Lemma 1). This means that at this point in time, the leader can start broadcasting the message about the transition to the second phase to the other agents. To reduce the required number of states, the broadcasting is limited to agents with labels within $[1, n\varepsilon]$. In this way, the second phase starts. In fact, only $n\varepsilon$ agents will ever update their component state to phase 2 by the definition of the protocol.

For the analysis of the synchronization of the leader election preprocessing with the two phases as well as that of putting the two phases together, see Subsection 2.4 and the proof of Theorem 1.

3.2. Formal description

We can describe the states and the transition function of our fast labeling protocol more compactly and formally as follows.

State components

- $L \in \{0,\ 1\}$ **# 1 for the leader otherwise 0**
- $\texttt{counter} \in [0, \lceil (c \log n)/\varepsilon \rceil - 1]$
- $\texttt{phase} \in \{$ 1,2,done $\}$ **# initially all agents are assumed to be in phase 1**
- $\texttt{label} \in \{\texttt{undefined}\} \cup [1, 2n]$
- $\texttt{interval} \in \{$ the set of intervals at the nodes of the partition tree (see Fig. 1)$\}$

For an estimate of the number of required states, refer to Theorem 1 below. The protocol can utilize significantly fewer states than the product of the sizes of state components' ranges, as only a small subset of possible component state-values combinations needs to be considered.

Transition function

Leader election preprocessing

Initially, we run the leader election protocol described in Subsection 2.4 in a preprocessing step. We assume that as the result of the preprocessing, only the agent selected for the leader has L set to 1. Furthermore, we assume that for each agent, $\texttt{phase} = 1$, $\texttt{label} = \texttt{undefined}$, and $\texttt{interval} = \emptyset$ hold initially.

The transition function is described by a finite set of schematic state transition rules for two interacting agents. The rules take the form
$(\texttt{Agent1}[\texttt{list1}], \texttt{Agent2}[\texttt{list2}]) \rightarrow (\texttt{Agent1}[\texttt{list3}], \texttt{Agent2}[\texttt{list4}])$.
Here, list1 and list2 are lists of constraints on relevant component state values that have to be satisfied in order to apply the transition rule. Meanwhile, `list3` and `list4` are lists of updates of affected component state values. Notably, the component state values are expressed in terms of variables. Also, the values of component states not appearing on the lists `list3` and `list4` remain unchanged. If one of the agents participating in the interaction remains unaffected, it is omitted on the right side of the rule. Finally, we apply the convention that whenever the left sides of one or more rules apply to a pair of interacting agents, all these rules are executed.

Phase 1

- $(\texttt{Agent1}[L = 1, \texttt{label} = \texttt{undefined}], \texttt{Agent2})$
  $\rightarrow (\texttt{Agent1}[\texttt{label} = 1, \texttt{interval} = [2, n], \texttt{counter} = 0])$
- $(\texttt{Agent1}[\texttt{interval} = [q, r]\ \texttt{where}\ q < r\ \texttt{and}\ \lfloor \frac{q+r}{2} \rfloor + 2 \leq r], \texttt{Agent2}[\texttt{label} = \texttt{undefined}])$
  $\rightarrow (\texttt{Agent1}[\texttt{interval} = [q, \lfloor \frac{q+r}{2} \rfloor]], \texttt{Agent2}[\texttt{interval} = [\lfloor \frac{q+r}{2} \rfloor + 2, r], \texttt{label} = \lfloor \frac{q+r}{2} \rfloor + 1])$
- $(\texttt{Agent1}[\texttt{interval} = [q, r]\ \texttt{where}\ q < r\ \texttt{and}\ \lfloor \frac{q+r}{2} \rfloor + 2 > r], \texttt{Agent2}[\texttt{label} = \texttt{undefined}])$
  $\rightarrow (\texttt{Agent1}[\texttt{interval} = [q, \lfloor \frac{q+r}{2} \rfloor]], \texttt{Agent2}[\texttt{label} = \lfloor \frac{q+r}{2} \rfloor + 1])$
- $(\texttt{Agent1}[\texttt{interval} = [q, q]], \texttt{Agent2}[\texttt{label} = \texttt{undefined}])$
  $\rightarrow (\texttt{Agent1}[\texttt{interval} = \emptyset], \texttt{Agent2}[\texttt{label} = q])$

Transition from Phase 1 to Phase 2

- $(\texttt{Agent1}[L = 1, \texttt{counter} = k < \lceil (c \log n)/\varepsilon \rceil - 1], \texttt{Agent2})$
  $\rightarrow (\texttt{Agent1}[\texttt{counter} = k + 1])$
- $(\texttt{Agent1}[L = 1, \texttt{counter} = \lceil (c \log n)/\varepsilon \rceil - 1, \texttt{Agent2}[\texttt{phase} = 1, \texttt{label} \leq n\varepsilon])$
  $\rightarrow (\texttt{Agent1}[\texttt{phase} = 2], \texttt{Agent2}[\texttt{phase} = 2])$
- $(\texttt{Agent1}[\texttt{phase} = 2], \texttt{Agent2}[\texttt{phase} = 1, \texttt{label} \leq n\varepsilon])$
  $\rightarrow (\texttt{Agent2}[\texttt{phase} = 2])$

Phase 2

- $(\texttt{Agent1}[\texttt{phase} = 2, \texttt{label} = q \leq n\varepsilon, \texttt{interval} = \emptyset], \texttt{Agent2}[\texttt{label} = \texttt{undefined}])$
  $\rightarrow (\texttt{Agent1}[\texttt{phase} = \texttt{done}], \texttt{Agent2}[\texttt{label} = q + n])$

3.3. Analysis

The following lemmata play a central role in demonstrating that $O(n \log n)$ interactions are sufficient w.h.p. to implement our protocol.

Lemma 1. *Let $0 < \varepsilon \leq 1$. There is a constant c such that after $(cn \log n)/\varepsilon$ interactions in the first phase the number of unlabeled agents falls below $n\varepsilon/4$ w.h.p.*

Proof. Let c be a constant whose value will be specified later. For the purpose of obtaining a contradiction, suppose that there exists a set F of at least $n\varepsilon/4$ agents that after $(cn \log n)/\varepsilon$ interactions still have not been assigned any labels. Note that for any agent A, the probability that a given interaction will be between A and a member in F is at least $\frac{\varepsilon}{4n}$.

We model the first phase by growing a subtree of the partition tree. The subtree initially consists of a single root node with the leader agent starting with the interval $[2, n]$. Specifically, to obtain a contradiction, we will show that after $(cn \log n)/\varepsilon$ interactions, the subtree becomes fully developed (i.e., it becomes the partition tree) w.h.p., so such a set F cannot exist w.h.p.

Consider an arbitrary path P from the root to a leaf in the partition tree. Note that several agents during distinct interactions can appear on the path. Define as a success an interaction between an agent currently at a leaf of the growing subtree, where the leaf belongs to P, with an unlabeled agent. Note that if the leaf of the subtree is not a leaf of the partition tree, the interaction results in attaching two children to it, thereby creating two new leaves in the extended subtree. To be precise, the subpath of P in the growing subtree extends to one of the two new leaves, and the two interacting agents relocate to the two new leaves, respectively.

During $(cn \log n)/\varepsilon$ interactions, the expected number of interactions of a leaf of the growing tree belonging to P with an agent in F, i.e., the expected number of successes μ is at least $\frac{c}{4} \log n$ by our assumption on F. Using the first multiplicative Chernoff bound from Fact 2 with δ equal to $\frac{1}{2}$, we can choose a sufficiently large constant c such that the probability of achieving at least $(1 - \frac{1}{2})\frac{c}{4} \log n \geq \log_2 n + 1$ successes will be at least $1 - e^{-\frac{1}{8}\frac{c}{4}\log n} \geq 1 - \frac{1}{n^{1+\alpha}}$, for a given $\alpha > 1$. However, due to the construction of the partition tree, P cannot have more than $\log_2 n + 1$ edges. Consequently, the path P will be completed in the growing subtree after the $(cn \log n)/\varepsilon$ interactions with probability at least $1 - \frac{1}{n^{1+\alpha}}$.

Applying the union bound (Fact 1), we find that the probability of at least one of the n paths from the root to the leaves in the partition tree remaining incomplete after the $(cn \log n)/\varepsilon$ interactions is at most $\frac{1}{n^{\alpha}}$. In summary, after the $(cn \log n)/\varepsilon$ interactions, all the n paths are completed, ensuring that all agents receive labels with probability at least $1 - \frac{1}{n^{\alpha}}$. W.h.p., this contradicts the assumption that a set F of at least $n\varepsilon/4$ agents remains without assigned labels after $(cn \log n)/\varepsilon$ interactions. □

Lemma 2. *Let $0 < \varepsilon \leq 1$. If the second phase starts after $cn \log n$ interactions, where c is the constant from Lemma 1, then only $O((n \log n)/\varepsilon)$ interactions are needed to assign labels in $[1, (1 + \varepsilon)n]$ to the remaining unlabeled agents, w.h.p.*

Proof. According to Lemma 1, the number of unlabeled agents at the start of the second phase is at most $n\varepsilon/4$ w.h.p. Therefore, at the beginning of this phase, the number of agents with labels less or equal to εn is at least $\frac{3\varepsilon n}{4}$ w.h.p. Note that not all of the latter agents need to be located at the leaves of the partition tree; some of them may still reside at intermediate nodes of the tree and can assign labels within the range $[1, n]$. An agent with a label $i \leq n\varepsilon$ at a leaf of the partition tree can assign the label $i + n$ to an unlabeled agent only once. Since this can happen at most $\frac{n\epsilon}{4}$ times, the number of agents with labels in $[1, \varepsilon n]$ that can assign a label to an unlabeled agent is always at least $\frac{n\varepsilon}{2}$ w.h.p. We conclude that for an unlabeled agent the probability of an interaction with an agent that can assign a label is at least $\frac{\varepsilon}{2n}$. Hence, after every $O(n/\epsilon)$ interactions, the expected number of unlabeled agents reduces by half. It follows that the expected number of such interaction rounds is $O(\log n)$.

We obtain the $O((n \log n)/\varepsilon)$ bound on the number of interactions w.h.p. by using Fact 4 with the constant C to be specified shortly and $\delta = \frac{1}{2}$. Subsequently, each agent will interact at least $(C \log n)/\varepsilon$ times with other agents w.h.p. over the course of $(Cn \log n)/\varepsilon$ interactions. The probability that an encountered agent can assign a label is at least $\frac{n\varepsilon/2}{n} = \frac{\varepsilon}{2}$. Consequently, the probability that a given agent does not interact with any agent capable of providing a label over the course of the aforementioned interactions is at least $(1 - \frac{\varepsilon}{2})^{(C \log n)/\varepsilon}$. By picking a large enough C, we conclude that each agent (in particular, those unlabeled) will interact with at least one agent capable of providing a label during the $(Cn \log n)/\varepsilon$ interactions w.h.p. □

Remark 1. During both phases, no pair of agents have the same label.

Proof. The uniqueness of the label assignments in the first phase follows from the separation of the labels and intervals assigned to agents before and after each interaction. This argument also applies to labels that do not exceed n and are assigned later in the second phase. Furthermore, the uniqueness of labels in the form $i + n$ stems from the uniqueness of labels assigned to agents that pass on these labels. □

We shall also use the following auxiliary lemma on broadcasting limited to a subset of agents in order to reduce the number of states necessary to implement the phase transition.

Lemma 3. *The leader can inform $\Theta(n\varepsilon)$ agents with labels not exceeding $O(n\varepsilon)$ about the phase transition utilizing only these agents in $O((n\log n)/\varepsilon)$ interactions w.h.p.*

Proof. During the initial part of the broadcasting process, after every $O(n/\varepsilon)$ interactions, the expected number of agents participating in the broadcasting process doubles. Thus, after $O((n\log n)/\varepsilon)$ interactions, the expected number of informed agents becomes $\Omega(n\varepsilon)$. The expected number of uninformed agents reduces by half after every $O(n/\varepsilon)$ interactions. Consider rounds, each consisting of $O(n/\varepsilon)$ interactions. To estimate the expected number of such rounds needed to complete the broadcasting let us call such a round successful if it reduces at least by half the number of uniformed agents if this number is positive. The expected number of successful iterations in a sequence of $d\log n$ rounds is at least $\frac{d}{2}\log n$ so the expected number of rounds necessary to complete the broadcasting is $O(\log n)$. It remains to convert the latter bound into a w.h.p. one.

To facilitate the probabilistic analysis of the doubling part, we define a binary broadcast tree. After interacting with an uninformed agent, an informed agent at an intermediate node migrates to one of its two child nodes. The other agent becomes informed and migrates to the other child of the node (cf. the partition tree in the proof of Lemma 1). Then, we apply the method described in the proof of Lemma 1 to show that w.h.p. only $O((n\log n)/\varepsilon)$ interactions are necessary to achieve a configuration where only a constant fraction of the agents participating in the broadcasting remain uninformed.

To establish an analogous asymptotic upper bound on the number of interactions needed for the halving part w.h.p., we employ Fact 4 analogously as in the proof of Lemma 2. □

Theorem 1. *Let $1 \geq \varepsilon > 0$. There is a safe protocol for a population of n agents that assigns unique labels within the range $[1, (1+\varepsilon)n]$ to n agents equipped with $(2+\varepsilon)n + O(n^a)$ states, for any positive constant $a < 1$, during $O((n\log n)/\varepsilon)$ interactions w.h.p. The protocol is also silent w.h.p.*

Proof. Assuming that the leader election preprocessing yields a unique leader, the correctness of label assignment in both phases w.h.p. and the fulfilling of the definition of a silent w.h.p. and safe protocol can be inferred from Lemmata 1, 2, 3, Remark 1, and the protocol's specification. To be precise, the correctness of label assignment follows from Remark 1, whereas the fact that the protocol is silent w.h.p. follows from Lemma 2, Remark 1, and the protocol's specification. Finally, the protocol is safe since it never updates labels assigned to the agents.

The leader election preprocessing, as described in Subsection 2.4, and its synchronization with the proper labeling protocol in two phases add $O(n\log n)$ interactions w.h.p. and $o(n^a)$ states for any positive $a < 1$. By Fact 7, the leader election preprocessing provides a unique leader w.h.p. It follows that w.h.p. the whole protocol provides a correct labeling, and is silent and safe. (To achieve the probability of at least $1 - \frac{1}{n^\alpha}$, we need to increase α slightly in both Fact 7 and Lemmata 1, 2.)

Both phases require $O((n\log n)/\varepsilon)$ interactions w.h.p. by Lemmata 1 and 2.

Recall that to combine the two phases, analyzed in Lemmata 1 and 2, we let the leader agent count its interactions. When the number of interactions of the leader in the first phase surpasses an appropriate multiple of $(\log n)/\epsilon$, the total number of interactions in the first phase reaches the required lower bound from Lemma 1 w.h.p. by Fact 4. Subsequently, according to Lemma 1, the number of unlabeled agents during the first phase falls below $n\varepsilon/4$ w.h.p. Therefore, the leader starts broadcasting the message on the transition to the second phase to the other agents.

By Lemma 3, the broadcasting requires $O((n\log n)/\varepsilon)$ interactions w.h.p. since only the $\Theta(n\varepsilon)$ agents in states corresponding to labels in $[1, n\varepsilon]$ are involved in it.

Starting the estimation of the number of needed states, let us generally remark that a straightforward approach to upper bound the number of states for a protocol with multiple phases is to compute the size of the Cartesian product of the sets of component states in respective phases. In the following refined state analysis of our protocol, we occasionally exploit the fact that some combinations of the component states specified by elements of the Cartesian product are never used, eliminating the need to take them into account. For instance, once an agent is elected as the leader, it can take only the leftmost path in the partition tree and utilize only $O(\log n)$ component states in the labeling process.

The leader election preprocessing uses $O(n^a)$ states, for any positive constant $a < 1$. When the leader initiates the labeling process, all preprocessing states of encountered agents (some of them may not yet have been informed about the termination of the preprocessing) are seen as the initial state for the labeling part of the protocol.

To reduce the state count in modeling the first phase, instead of having states corresponding to all possible sub-intervals of $[1, n]$, we focus on states corresponding to the nodes of the partition tree (see Fig. 1) whose subtree is formed in the

first phase. Specifically, with each intermediate node of the partition tree that does not correspond to a label in $[1, n\varepsilon]$ (i.e., that is not an ancestor of a leaf corresponding to a label in $[1, n\varepsilon]$), we associate a single state. (Recall that when an agent at an intermediate node of the partition tree encounters an unlabeled agent, the former agent shifts to the left child of the node.) With each intermediate node corresponding to a label in $[1, n\varepsilon]$, we associate two states. They indicate whether the agent at the node has already received the message about the phase transition. Furthermore, with each leaf of the partition tree with a label i in $[1, n\varepsilon]$, we associate four states. They indicate whether the agent at the leaf has already received the message about the phase transition, and whether the agent has already passed the label $i+n$ to some agent without a label, respectively. With each of the remaining leaves, we associate only a single state.

We emphasize that while modeling the two phases, there is no need to duplicate the aforementioned number of states to indicate if an agent is the leader. Simply, after the first interaction of the leader in the labeling process, the leader follows the leftmost path in the partition tree. In other words, the states associated with the nodes on the leftmost path also indicate the leadership property, while those associated with remaining nodes in the tree indicate the absence of this property.

By Lemma 1 and Fact 4, we also need $O((\log n)/\varepsilon)$ additional component states for the leader to keep track of the number of its own interactions in order to initiate broadcasting the message about the transition to phase two at the appropriate time step. We can eliminate the $O(\frac{1}{\varepsilon})$ factor here by letting the leader approximately count each $\Theta(1/\varepsilon)$-th interaction. To achieve this, the leader can count only interactions with agents which have got labels not exceeding $O(\varepsilon n)$. The number of the latter interactions is a fraction $\Theta(\varepsilon)$ of the leader's overall interactions w.h.p. Thus, for each interaction which the leader counts, there will be additional $\Theta(1/\varepsilon)$ interactions w.h.p. which the leader does not have to count. Hence, in order to determine that sufficiently many, i.e., $\Theta((\log n)/\varepsilon)$, interactions with the leader have taken place w.h.p., the leader has only to count the $\Theta(\log n)$ interactions with agents which have got labels not exceeding $O(\varepsilon n)$. It follows that the leader's state component corresponding to the leader's counter requires only $O(\log n)$ component states w.h.p. (here again we have to amplify the probability in order to ensure high probability for the whole labeling process).

Since the leader can only occupy the $O(\log n)$ nodes along the leftmost path in the partition tree, the total number of states utilized by the leader in the labeling process is $O(\log n) \times O(\log n) = O(\log^2 n)$.

Finally, we have $n\varepsilon$ states corresponding to the labels in $[n+1, (1+\varepsilon)n]$. Hence, the total number of states required is only $(2 + O(\varepsilon))n + O(n^a)$, where a is any positive constant less than 1. To eliminate the constant factor, say c, at ε, we can run the protocol for an $\varepsilon' = \frac{\varepsilon}{c}$. This adjustment does not alter the asymptotic upper bound on the number of required interactions w.h.p. and, in fact, narrows down the label range. □

By combining the protocol from Theorem 1 with that of Berenbrink et al. for exact counting the population size [12] (Fact 6), we obtain the following corollary on unique labeling when the population size is initially unknown to the agents.

Corollary 1. *Let $1 \geq \varepsilon > 0$. There exists a protocol for a population of n agents that assigns unique labels within the range $[1, (1+\varepsilon)n]$ to the agents equipped with $\tilde{O}(n)$ states and initially not knowing the number n during $O((n \log n)/\varepsilon)$ interactions w.h.p.*

Proof. We run first the protocol for exact counting (Fact 6) and then our protocol for unique labeling (Theorem 1) using the leader elected by the counting protocol. We can synchronize the three protocols similarly to how we synchronized the two phases of our protocol, using an additional $O((n \log n)/\varepsilon)$ interactions w.h.p. and $O(\log n)$ states. □

4. State- and range-optimal labeling

In this section we propose and analyze nearly state-optimal protocols, which are silent and safe once a unique leader is elected. These protocols utilize labels from the smallest possible range $[1, n]$. Recall that the simple n-state Protocol 1 [15] described in the introduction is unfortunately not safe. As in the previous sections, we assume the number of agents n to be known. We introduce a silent and safe labeling protocol named *Single-Cycle* which utilizes $n + 5\sqrt{n} + O(\log \log n)$ states. The number of interactions required by the protocol is $O(n^3)$ both in expectation and w.h.p. In Section 5, we show that any silent and safe labeling protocol requires $n + \sqrt{\frac{n-1}{2}} - 1$ states; see Theorem 5. Thus, our protocol is nearly state-optimal. Finally, we introduce a partially parallelized version of the *Single-Cycle* protocol. When $1/n \leq \varepsilon < 1/2$, it uses $O(n^2/\varepsilon)$ interactions w.h.p. and $n(1 + 9\sqrt{\varepsilon}) + O(\log \log n)$ states.

4.1. Labeling protocol

The state-efficient labeling protocol begins with a preprocessing step which elects a unique leader using the $O(\log \log n)$-state protocol from Fact 8 in [21]. The leader is reported, and the preprocessing step concludes after $O(n \log^2 n)$ interactions with high probability. Note here that the trivial constant-state $O(n^2)$-time leader election protocol [20] cannot be directly used instead as it remains in the leader election mode.

The core concept of the proper labeling protocol involves two agents: A, the leader computed during the preprocessing, and B, an agent nominated by A in its first action. These two agents collectively serve as partial *label dispensers*. Together, these agents distribute unique labels to the remaining *free* (not yet labeled) agents in the population. Agent A provides the

first part, while agent B handles the second part of each individual label. To simplify the presentation, we first assume that n is a square of some integer. During the protocol's execution, agent A uses partial labels $\texttt{label}(a) \in \{0, \ldots, \sqrt{n}-1\}$ and B uses partial labels $\texttt{label}(b) \in \{1, \ldots, \sqrt{n}\}$. Both dispensers assign a unique pair of partial labels $(\texttt{label}(a), \texttt{label}(b))$ to each agent. The combination (i, j) is interpreted as the integer label $i \cdot \sqrt{n} + j$. The protocol first labels all free agents according to the sequence $(\sqrt{n}-1, \sqrt{n})$, $(\sqrt{n}-1, \sqrt{n}-1)$, $(\sqrt{n}-1, \sqrt{n}-2)$, $\ldots$, $(\sqrt{n}-1, 1)$, $(\sqrt{n}-2, \sqrt{n})$, $\ldots$, $(0, 4)$, $(0, 3)$ and finally assigns $(0, 2)$ to agent B and $(0, 1)$ to agent A.

In a nutshell, once the leader A during its first interaction nominates the second dispenser B, the actual labeling process starts. This process is based on repeated utilization of a fixed cycle of interactions between dispensers A, B, and the free agents. In each cycle a new label is dispensed, i.e., exactly one free agent becomes labeled. At the beginning of each cycle, agent A first awaits an interaction with any free agent F. Once such an interaction occurs, A dispenses its current partial label $\texttt{label}(a)$ to F. Next, F awaits an interaction with B in order to receive the second part of its label. When this happens, F enters its final state with the combined label, and B subsequently awaits an interaction with A to signal the transition to the next cycle. On the conclusion of this interaction, if $\texttt{label}(b) > 1$, agent B adopts new partial label $\texttt{label}(b) - 1$. Otherwise, B adopts $\texttt{label}(b) = \sqrt{n}$, and agent A adopts new label $\texttt{label}(a) - 1$. The sole exception occurs when $\texttt{label}(a) = 0$ and $\texttt{label}(b) = 3$. In this case, agent B adopts label $(0, 2)$ and agent A adopts label $(0, 1)$ and both agents conclude the labeling process.

The state utilization and transition function in the labeling protocol are specified as follows. (We shall specify the transition function using the same form of transition rules as in the preceding section. For abbreviation, we shall denote $X[X.\textit{suffix}]$, where X is an agent, $X \in \{A, B, F\}$, $X.\texttt{suffix}$ is a state of X, $\texttt{suffix} \in \{\texttt{init}, \texttt{final}\}$, by $X.\texttt{suffix}$. Similarly, $X.\texttt{suffix} = y$ will stand for $X[X.\texttt{suffix} = y]$.)

State utilization in *Single-Cycle* protocol

[Agent A] Since $\texttt{label}(a) \in \{0, \ldots, \sqrt{n}-1\}$ dispenser A utilizes $2 \cdot \sqrt{n} + 2$ states including:

- $A.\texttt{init} = (1)$ **# the initial (leadership) state of dispenser A,**
- $A[\texttt{label}(a), \texttt{await}(F)]$ **# dispenser A carrying partial label $\texttt{label}(a)$ awaits interaction with a free agent F,**
- $A[\texttt{label}(a), \texttt{await}(B)]$ **# dispenser A carrying partial label $\texttt{label}(a)$ awaits interaction with dispenser B,**
- $A.\texttt{final} = (0, 1)$ **# the final state of A.**

[Agent B] Since $\texttt{label}(b) \in \{1, \ldots, \sqrt{n}\}$ dispenser B utilizes $2 \cdot \sqrt{n} + 1$ states including:

- $B[\texttt{label}(b), \texttt{await}(F)]$ **# dispenser B carrying partial label $\texttt{label}(b)$ awaits interaction with a free agent F,**
- $B[\texttt{label}(b), \texttt{await}(A)]$ **# dispenser B carrying partial label $\texttt{label}(b)$ awaits interaction with dispenser A**
- $B.\texttt{final} = (0, 2)$ **# the final state of B.**

[Agent F] Since free agents carry partial labels $\texttt{label}(a) \in \{0, \ldots, \sqrt{n}-1\}$ and eventually adopt one of the $n-2$ destination labels (excluding dispensers) they utilize $n + \sqrt{n} - 1$ states including:

- $F.\texttt{init} = (0)$ **# the initial (non-leader) state of F**
- $F[\texttt{label}(a), \texttt{await}(B)]$ **# free agent F carrying partial label $\texttt{label}(a)$ awaits interaction with dispenser B,**
- $F.\texttt{final} = (\texttt{label}(a), \texttt{label}(b))$ **# the final state of F.**

In total, the *Single-Cycle* protocol requires $n + 5 \cdot \sqrt{n} + 2$ states.

Transition function in *Single-Cycle* protocol

[Step 0] Initialization During the first interaction of A with a free agent the second dispenser B is nominated. Both dispensers adopt their largest labels. Following this interaction, A will await a free agent in the initial state, while B will await a free agent carrying a partial label obtained from A.

- $(A.\texttt{init}, F.\texttt{init})$
  $\rightarrow (A[\texttt{label}(a) = \sqrt{n} - 1, \texttt{await}(F)], B[\texttt{label}(b) = \sqrt{n}, \texttt{await}(F)])$,

The three steps C_1, C_2, and C_3 of the labeling cycle are given below.
[Step C_1] Agent A dispenses partial label When agent A interacts with a free agent F, the current partial label $\texttt{label}(a)$ is dispensed to F. Both A and the partially labeled F will then await interactions with B, who is currently waiting to interact with F but is not yet ready to interact with A.

- $(A[\texttt{label}(a), \texttt{await}(F)], F.\texttt{init})$
  $\rightarrow (A[\texttt{label}(a), \texttt{await}(B)], F[\texttt{label}(a), \texttt{await}(B)])$ **# Go to Step C_2**

[Step C_2] Agent B dispenses partial label When agent B interacts with a free agent F carrying the partial label $\texttt{label}(a)$, the complementary current partial label $\texttt{label}(b)$ is provided to F. Agent F then reaches the final state with the combined label $(\texttt{label}(a), \texttt{label}(b))$. Agent B is now ready for interaction with A.

- $(B[\texttt{label}(b), \texttt{await}(F)], F[\texttt{label}(a), \texttt{await}(B)])$
  $\rightarrow (B[\texttt{label}(b), \texttt{await}(A)], F.\texttt{final} = (\texttt{label}(a), \texttt{label}(b)))$ **# Go to Step** C_3

[Step C_3] Agents A and B negotiate a new label or conclude In the case when $\texttt{label}(a) = 0$ and $\texttt{label}(b) = 3$ the dispensers A and B conclude in states $(0, 1)$ and $(0, 2)$ respectively; see the first transition. Otherwise, A and B update their partial labels to form the next pair and the protocol goes back to Step C_1.

- $(A[\texttt{label}(a) = 0, \texttt{await}(B)], B[\texttt{label}(b) = 3, \texttt{await}(A)])$
  $\rightarrow (A.\texttt{final} = (0, 1), B.\texttt{final} = (0, 2))$ **# Conclude the labeling process**
- $(A[\texttt{label}(a) = 0, \texttt{await}(B)], B[\texttt{label}(b) > 3, \texttt{await}(A)])$ **or**
  $(A[\texttt{label}(a) > 0, \texttt{await}(B)], B[\texttt{label}(b) > 1, \texttt{await}(A)])$
  $\rightarrow (A[\texttt{label}(a), \texttt{await}(F)], B[\texttt{label}(b) - 1, \texttt{await}(F)])$ **# Go to Step** C_1
- $(A[\texttt{label}(a) > 0, \texttt{await}(B)], B[\texttt{label}(b) = 1, \texttt{await}(A)])$
  $\rightarrow (A[\texttt{label}(a) - 1, \texttt{await}(F)], B[\texttt{label}(b) = \sqrt{n}, \texttt{await}(F)])$ **# Go to Step** C_1

Theorem 2. Single-Cycle *utilizes* $n + 5 \cdot \sqrt{n} + O(\log\log n)$ *states and the minimal label range* $[1, n]$*. The number of interactions required by the protocol is* $O(n^3)$ *both in expectation and w.h.p. Once a unique leader is elected, it produces a valid labeling of n agents and it is silent and safe.*

Proof. Assume that the leader election preprocessing provides a unique leader. Then, the protocol is silent and safe by its definition. All labels are distributed sequentially and the labeling process concludes when both dispensers finalize their respective labels. Specifically, once the two dispensers A and B are set up they operate in a short cycle consisting of steps C_1, C_2, and C_3, sequentially labeling all free agents in the population. One can observe that the sequence of cycles resembles the structure of two nested loops: the external loop iterates over the partial labels of A, while the internal one iterates over partial labels of B. Overall, the labeling cycle is employed $n - 2$ times. We show below that the number of interactions required by the labeling process beyond leader election preprocessing is $O(n^3)$, both in expectation and w.h.p.

Recall that each utilization of the labeling cycle requires executing the three steps C_1, C_2, and C_3. Step C_1 involves an interaction of leader A with a free agent. At the start of the labeling process, when all agents but A and B are free, the probability of a successful interaction is $\frac{2(n-2)}{n(n-1)}$. Towards the end of the process, when only one free agent remains, this probability decreases to $\frac{2}{n(n-1)}$. In contrast, steps C_2 and C_3 are based on interactions between specific pairs of agents; therefore, the probability of a successful interaction is always $\frac{2}{n(n-1)}$. Finally, we also need to consider an interaction during which leader A nominates dispenser B, which occurs with probability $\frac{2}{n}$.

Let X_i be a random variable standing for the number of interactions needed to draw the i-th successful interaction, given that the preceding successful interactions have already occurred. Specifically, X_0 corresponds to the number of interactions needed to nominate B, while the remaining $3(n-2)$ random variables $X_1, \ldots, X_{3(n-2)}$ stand for the number of interactions needed to conclude each step of the labeling cycle, which is utilized $n - 2$ times. Consequently, the expected number of interactions required to conclude the labeling process is given by $ET = \sum_{i=0}^{3(n-2)} E(X_i) = \frac{n}{2} + \sum_{i=1}^{n-2} \frac{n(n-1)}{2(n-i-11)} + 2\sum_{i=1}^{n-2} \frac{n(n-1)}{2} = n + \frac{n(n-1)}{2}(\sum_{i=1}^{n-2} \frac{1}{n-i-1} + 2n - 4) = n^3 - o(n^3)$.

Since the random variables X_i are independent, we can apply the Chernoff-Janson bound as described in Fact 3 to X. Given that $p^* = \frac{2}{n(n-1)} \geq \frac{1}{n^2}$ and $\mu = n^3 - o(n^3)$ in our scenario, choosing $\lambda = 1.1$ yields an exponentially small probability that ET exceeds λn^3. For this reason, the protocol requires $O(n^3)$ interactions w.h.p.

Finally, considering that the preprocessing (leader election) requires $O(n \log^2 n)$ interactions w.h.p. (see Subsection 2.4), we conclude that the entire labeling process requires $O(n^3)$ interactions w.h.p. By the definition of the protocol the range of assigned labels is $[1, n]$. And, as the preprocessing utilizes $O(\log\log n)$ states and the labeling is based on $n + 5 \cdot \sqrt{n} + 2$ states, we conclude that the total state utilization is limited to $n + 5 \cdot \sqrt{n} + O(\log\log n)$. □

As described above, protocol *Single-Cycle* assumes that n is a perfect square, but it can be generalized to work for any positive integer n as follows. Define the label set for agent A as $\{0, \ldots, \lfloor \frac{n-1}{\lceil\sqrt{n}\rceil} \rfloor\}$ and the label set for agent B as $\{1, \ldots, \lceil\sqrt{n}\rceil\}$. Additionally, modify the formula for computing labels from partial labels to $i \cdot \lceil\sqrt{n}\rceil + j$. (Here, a simple upper bound on the largest element in the label set for agent A is $\lfloor \frac{n-1}{\lceil\sqrt{n}\rceil} \rfloor = \lceil \frac{n}{\lceil\sqrt{n}\rceil} \rceil - 1 \leq \lceil\sqrt{n}\rceil - 1$.) To ensure that no labels larger than n will be assigned to the agents using this formula, modify [Step 0] so that the process starts with $\texttt{label}(a) = \lfloor \frac{n-1}{\lceil\sqrt{n}\rceil} \rfloor$ and $\texttt{label}(b) = ((n-1) \bmod \lceil\sqrt{n}\rceil) + 1$.

We also note that when the exact value of n is embedded in the transition function of *Single-Cycle* on the conclusion, all agents become dormant, i.e., they stop participating in the labeling process. The protocol could be redesigned so that

the labels are dispensed by A and B in increasing order using a diagonal method, e.g., $(0,0)$, $(0,1)$, $(1,0)$, $(0,2)$, $(1,1)$, $(2,0)$, $(0,3)$, $(1,2)$, $(2,1)$, $(3,0)$ etc., where agent A gets label $(0,0)$, agent B gets label $(0,1)$, the first labeled free agent gets $(1,0)$, the second $(0,2)$, then $(1,1)$ and $(2,0)$, when A and B start using the next diagonal, and etc. Each pair (i,j) is interpreted as $(i+j)(i+j+1)/2+i$. For instance, $(0,1)=1$, $(0,2)=3$, $(0,3)=6$ and in general $(0,j)=j(j+1)/2$, $(1,j-1)=j(j+1)/2+1$, $(1,j-2)=j(j+1)/2+2$,....,$(j,0)=j(j+1)/2+j=(j+1)(j+2)/2-1=(0,j+1)-1$. In this variant, the size of the population does not need to be known in advance. However, the two dispensers will never stop searching for free agents yet to be labeled (nevertheless the protocol remains silent).

4.2. Faster labeling

We observe that the *Single-Cycle* protocol can be partially parallelized by instructing the leader A to form k pairs of dispensers, where each pair labels agents in a distinct range of size n/k. In this scenario, the new *k-Cycle* protocol requires an additional $2k$ states to enable the leader A to initialize the labeling process (by creating two dispensers) for all k independent cycles. Thus, the total number of states is bounded by $n+2k+k\cdot(5\sqrt{n/k}+2)=n+4k+5k\cdot\sqrt{n/k}<n+9\sqrt{nk}$, as $k<\sqrt{nk}$, plus the number of states required by the leader election preprocessing. For the leader election preprocessing, we employ the protocol described in Fact 8 ([21]), along with the method to synchronize it with the proper labeling protocol outlined in Subsection 2.4. Similarly, it adds $O(n\log^2 n)$ interactions w.h.p. and $O(\log\log n)$ states.

Applying methods from the proof of Theorem 2, we can show that the expected number of interactions required by the *k-Cycle* protocol is bounded by $O(n^2\cdot k+n^3/k)$, which is minimized for $k=\sqrt{n}$. And indeed, note that to initialize k cycles leader A must communicate with $2k-1$ free agents. The search for dispensers occurs concurrently with the actual label dispensing processes in some cycles, so we can only assume that the probability of drawing an interaction which yields a new dispenser is at least $\frac{2}{n(n-1)}$. Since we only need $2k-1$ successful interactions, the dispenser creation concludes after $O(n^2\cdot k)$ interaction w.h.p. Once this process is complete, all k cycles are ready to dispense their labels. In fact, some of these cycles have already begun dispensing labels. Examine each of these cycles individually. It dispenses its n/k labels utilizing $O(n^3/k)$ (global) interactions w.h.p., and the likelihood of significantly more interactions is exponentially small. Hence, we can apply the union bound to show that all k cycles finish on time, i.e., utilizing $O(n^3/k)$ interactions w.h.p.

In summary, we obtain the following theorem.

Theorem 3. *For $k=\sqrt{n}$, and the minimal label range $[1,n]$, the k-*Cycle *protocol provides a valid labeling that utilizes $O(n^{5/2})$ interactions w.h.p. and $n+O(n^{3/4})$ states.*

Proof. The correctness of the labeling follows from the protocol definition. The time complexity follows from the argument provided before the theorem. Regarding the space utilization, given that the number of states is bounded by $n+9\sqrt{nk}+O(\log\log n)$ and $k=\sqrt{n}$, we achieve the desired bound of $n+O(n^{3/4})$. □

Note that one can enhance the time complexity of *k-Cycle* protocol by deferring work in each cycle until all dispensers have been created. Once this is done, leader A notifies all dispensers to begin their tasks via the broadcast process from Fact 5, executed by a subpopulation formed of leader A and $k=\varepsilon\cdot n$ dispensers, where $0<\varepsilon<1/2$. As ε is a positive constant, the broadcast is run on a constant fraction of the population and is completed in $O(n\log n)$ interactions w.h.p. Moreover, as the number of all dispensers k equals $\varepsilon\cdot n$, leader A needs only $O(n^2)$ interactions to create all dispensers. This is because before the last dispenser is created the probability of choosing an interacting pair containing leader A and a free agent is at least $\frac{(1-\varepsilon)n}{n^2}=\frac{1-\varepsilon}{n}$. Meaning that the expected number of interactions needed to create $\varepsilon\cdot n$ dispensers is at most $\frac{\varepsilon\cdot n^2}{1-\varepsilon}=O(n^2)$, as $\frac{\varepsilon\cdot n^2}{1-\varepsilon}\cdot\frac{1-\varepsilon}{n}=\varepsilon\cdot n$. One can multiply the number of interactions by a constant factor to create $\varepsilon\cdot n$ dispensers in $O(n^2)$ time w.h.p. As discussed earlier, the duration in terms of interactions of the second stage is $O(n^3/k)=O(n^2/\varepsilon)$, for $k=\varepsilon\cdot n$, which dominates the time complexity of the enhanced *k-Cycle* protocol. Moreover, to achieve the speed-up one needs to employ a larger number of states to accommodate the delay of the second stage and the broadcasting process. This is accomplished by adding an additional state to each dispenser, enabling them to wait for the broadcast message. Upon receiving the broadcast message, each dispenser can then proceed to broadcast and dispense their respective labels. Note that leader A is allowed to broadcast and dispense labels just after the last dispenser is created. Thus the total number of states is bounded by $n+\varepsilon\cdot n+9\sqrt{nk}=n(1+\varepsilon+9\sqrt{\varepsilon})$, where $k=\varepsilon\cdot n$. The following trade-off holds.

Theorem 4. *For any constant $0<\varepsilon<1/2$, where $k=\varepsilon\cdot n$ is a natural number, and the minimal label range $[1,n]$, the enhanced k-*Cycle *protocol provides a valid labeling utilizing $O(n^2/\varepsilon)$ interactions w.h.p. and $n(1+\varepsilon+9\sqrt{\varepsilon})+O(\log\log n)$ states.*

5. Lower bounds

In this section, we derive several lower bounds on the number of states or interactions required by silent, safe, or the so-called pool protocols for unique labeling. Importantly, these lower bounds also hold in our model assuming that the population size is known to the agents initially and also when a unique leader is available initially.

The following general lower bound, valid for any range of labels, follows immediately from the definitions of a population protocol and the problem of unique labeling, respectively.

Remark 2. The problem of assigning unique labels to n agents requires $\Omega(n \log n)$ interactions w.h.p. and the agents have to be equipped with at least n states.

Proof. The requirement for $\Omega(n \log n)$ interactions w.h.p. arises since each agent has to interact at least once (e.g., see the introduction in [12]). The lower bound on the number of states follows from the symmetry of agents. Any agent has to be prepared to be assigned an arbitrary label represented by a logarithmic number of bits. □

5.1. A sharper lower bound on the number of states

We establish a lower bound of $n + \sqrt{\frac{n-1}{2}} - 1$ on the number of states required by a silent protocol which ensures a valid labeling of the n agents and is safe w.h.p. The lower bound holds even if the protocol is provided with a unique leader and the knowledge of the number of agents. It almost matches the upper bound established in the previous section.

The basic concept behind the lower bound proof is as follows. We examine a finite run of a silent protocol that generates a unique labeling and is safe with probability greater than $1 - \frac{1}{n}$. Thus, in the run all agents reach final states associated with distinct labels and we may assume that no agent updates its label. Next, we categorize the agents into two groups: these who attained their final state (in the run) by an interaction with an agent that already reached its final state, and those that achieved their final state through an interaction with an agent that has not yet achieved its final state. Consider the states of the agents in the first group directly preceding their final states. Note that the former states cannot belong to the set of final states in the run by the definition of the final states. Subsequently, we claim that there exists a successful run in which any pair of agents from the first group attains distinct states directly preceding their final states in the run. The proof of this claim is essential since it implies that the number of states outside the set of final states is at least as large as the number of agents in the first group. This combined with the facts that the total size of these two agent groups sums up to n, and that the number of states outside the final state set must be approximately at least the square root of the second group's size, we obtain the lower bound through straightforward calculations.

The proof of the lower bound is organized as follows. We start with an introductory section specifically designed to introduce essential concepts and notation based on the lower bound assumptions. Next, we prove the crucial claim (Lemma 4). Finally, we show Theorem 5 stating the lower bound, by using Lemma 4.

Consider the set I consisting of ordered pairs of the n agents. We can interpret I as the set of possible pairwise interactions between the agents.

Let Z be a finite run of a labeling protocol, i.e., a finite sequence of pairs in I. Suppose that the execution of Z is successful, meaning that each agent attains a final state with a distinct label, and no agent gets assigned two or more distinct labels during the run.

Let F_Z be the set of final states reached by the agents after the execution of the run Z. The equality $|F_Z| = n$ holds because we assume the execution of Z to be successful. Additionally, let R_Z stand for the set of remaining states utilized in this run. Note that if an agent is in a state in F_Z, it has a label.

Consider an agent x. Define $f_Z(x) \in F_Z$ as the last state attained by this agent in the run Z, and let $pred_Z(x)$ be the next to the last state achieved by the same agent in the run. Because at most one agent can have the common initial state as its final state, $pred_Z(\ )$ is defined for at least $n-1$ agents.

Remark 3. If $pred_Z(x)$ is defined then $pred_Z(x) \in R_Z$.

Proof. If $pred_Z(x) \in F_Z$ and $pred_Z(x)$ assigns a distinct label from that assigned by $f_Z(x)$ to x then we have a contradiction with our assumptions on Z. Specifically, this contradicts the assumption that no agent gets two or more distinct labels during the run. Conversely, if $pred_Z(x) \in F_Z$ and $pred_Z(x)$ assigns the same label as that assigned by $f_Z(x)$ to x, we face a contradiction with the validity of the final labeling resulting from Z. □

Next, let A_Z be the set of agents x that achieved their final state in the run Z through an interaction of x in the state $pred_Z(x)$ with an agent in a state in F_Z. Suppose next that there exists a pair of agents $x,\ y \in A_Z$, where $pred_Z(x) = pred_Z(y)$ and x achieves its final state $f_Z(x)$ not later than y. Note that $x,\ y$ cannot change to their final states in the same interaction by the definition of A_Z. Therefore, there exists a prefix $Z_1 i_1 Z_2 i_2$ of Z in which both agents achieve their final states in F_Z. Here Z_i is a subsequence of Z consisting of consecutive pairs in Z for $i = 1, 2$. i_1 is a pair specifying the interaction between the agent x and an agent x' in a state in F_Z turning x to the state $f_Z(x)$. (Additionally, x' cannot be in a final state different from its own, i.e., in $F_Z \setminus \{f_Z(x')\}$, as this would require updating its label contradicting the assumption on Z.) i_2 is a pair specifying an interaction between the agent y and another agent in a state in F_Z turning y to the state $f_Z(y)$. We shall refer to such a shortest prefix $Z_1 i_1 Z_2 i_2$ of Z as the (x, y)-*prefix of* Z. Furthermore, we can transform this prefix by replacing the pair i_2 in $Z_1 i_1 Z_2 i_2$ with the pair i_3 specifying the interaction between y and the agent x' in the state $f_Z(x')$ analogous to i_1. We shall refer to the resulting prefix as the *transformed* (x, y)-*prefix of* Z.

Remark 4. Let Z be a finite run of a protocol in which the agents achieve final states with distinct labels and no single agent is assigned two or more distinct labels in the run. Suppose that there exists a pair of agents $x,\ y \in A_Z$, such that $pred_Z(x) = pred_Z(y)$ and x reaches its final state $f_Z(x)$ not later than y. Then, neither the transformed (x, y)-prefix of Z nor any of its extensions can provide a valid labeling of the agents without updating labels for some of them.

Proof. Let $Z_1 i_1 Z_2 i_3$ be the transformed (x, y)-prefix of Z. It is sufficient to observe that in $Z_1 i_1 Z_2 i_3$ the agent y reaches the same state $f_Z(x)$ as the agent x, given that $pred_Z(x) = pred_Z(y)$. Thus, neither the run $Z_1 i_1 Z_2 i_3$ nor any of its extensions can yield a valid labeling of the agents without updating labels for some of them. □

Lemma 4. *Consider a silent protocol which produces a valid labeling of the n agents and is safe with probability larger than* $1 - \frac{1}{n}$. *There exists a finite run Z of the protocol, such that after the execution of Z, each agent is in a final state with a distinct label, no single agent is assigned two or more distinct labels during Z, and for any pair of distinct agents $x,\ y \in A_Z$, $pred_Z(x) \neq pred_Z(y)$ holds.*

Proof. The proof of the lemma is by a contradiction with the assumptions on the labeling protocol.

To obtain the contradiction, we assume that for each finite run Z in which the agents achieve final states with distinct labels without assigning two or more distinct labels to any single agent during the run, there is a pair of agents x and y in A_Z, such that $pred_Z(x) = pred_Z(y)$. Consider such a pair of agents x and y in A_Z that minimizes the length of the (x, y)-prefix $Z_1 i_1 Z_2 i_2$ of Z (as specified directly before Remark 4) in which both agents achieve their final states in F_Z. Based on Remark 4, we infer that neither the transformed (x, y)-prefix $Z_1 i_1 Z_2 i_3$ (also specified before Remark 4) nor any of its extensions can result in a valid labeling of the agents without updating labels for some of them. Importantly, the runs $Z_1 i_1 Z_2 i_2$ and $Z_1 i_1 Z_2 i_3$ are equally probable (*).

We initialize two sets S_{valid} and $S_{invalid}$ of strings (sequences) over the alphabet I. Then, for each run Z in which the agents achieve final states with distinct labels without updating the label of any single agent, we identify the subsequences $Z_1,\ Z_2$ and interactions i_1, i_2, i_3. We insert the prefix $Z_1 i_1 Z_2 i_2$ into S_{valid} and the corresponding sequence $Z_1 i_1 Z_2 i_3$ into $S_{invalid}$. Note that by the minimum length property of $Z_1 i_1 Z_2 i_2$, no string in S_{valid} is a prefix of another string in S_{valid}. The analogous property follows for $S_{invalid}$. By the construction of the sets, each run Z in which the agents achieve final states with distinct labels without updating the label of any single agent has to overlap with or be an extension of a string in S_{valid}. Furthermore, no run of the protocol that overlaps with a string in $S_{invalid}$ or extends a string in $S_{invalid}$ results in a valid labeling without updating the label of any single agent. Define the function $g : S_{valid} \to S_{invalid}$ by $g(Z_1 i_1 Z_2 i_2) = Z_1 i_1 Z_2 i_3$. According to the property (*), the probability that a string over I equals $Z_1 i_1 Z_2 i_2$ or extends $Z_1 i_1 Z_2 i_2$ is not greater than the probability that a string over I equals $g(Z_1 i_1 Z_2 i_2)$ or extends $g(Z_1 i_1 Z_2 i_2)$. The function g does not have to be a bijection. Suppose that $g(Z_1 i_1 Z_2 i_2) = g(Z'_1 i'_1 Z'_2 i'_2)$. Then we obtain $Z_1 i_1 Z_2 i_3 = Z'_1 i'_1 Z'_2 i'_3$. Moreover, the strings $Z_1 i_1 Z_2 i_2$ and $Z'_1 i'_1 Z'_2 i'_2$ may differ only in the last interaction, specifically, i_2 may differ from i'_2. However, both i_2 and i'_2 must include the same agent (y in the earlier construction) present in $i_3 = i'_3$. In conclusion, the two aforementioned strings in S_{valid} can differ by at most one agent in the last interaction. It follows that the function g maps at most $n - 1$ strings in S_{valid} to the same string in $S_{invalid}$. Consequently, the probability P_{valid} of the event that the agents eventually achieve their final states yielding a valid labeling without updating the label of any single agent is at most $n - 1$ times larger than the probability $P_{invalid}$ of the complementary event. It follows that $P_{valid} + P_{invalid} = 1$ and $P_{invalid} \geq \frac{1}{1+(n-1)}$. We obtain $P_{valid} \leq 1 - \frac{1}{n}$ which yields a contradiction with the lemma assumption $P_{valid} > 1 - \frac{1}{n}$. This completes the proof of the lemma. □

Theorem 5. *A silent protocol that produces a valid labeling of the n agents and is safe with probability greater than* $1 - \frac{1}{n}$ *requires at least* $n + \sqrt{\frac{n-1}{2}} - 1$ *states.*

Proof. Consider a run Z of the protocol that satisfies Lemma 4. As a result, $|R_Z| \geq |A_Z|$ holds.

Let B_Z be the set of remaining agents that got their final state in F_Z in an interaction, where both agents were in states outside F_Z (i.e., in R_Z). Since the agents in B_Z achieved distinct final states with distinct labels during the aforementioned interactions, we deduce that $2|R_Z|^2 \geq |B_Z|$, implying that $|R_Z| \geq \sqrt{|B_Z|/2}$. Simply, there are $|R_Z|^2$ ordered pairs of states in R_Z, and when agents in the states forming such a pair interact, they can achieve at most two distinct states in F_Z. In other words, if $2|R_Z|^2 < |B_Z|$, there would be a pair of agents in B_Z that would achieve the same final state in the run and hence would have the same label at the end of the considered run.

By straightforward calculations, we find that
$|R_Z| \geq \max\{|A_Z|, \sqrt{\frac{n-1-|A_Z|}{2}}\} \geq \sqrt{\frac{n-1}{2}} - 1$. This yields the theorem. □

By reusing the arguments from the proof of Theorem 5, we can also relatively easily derive the following trade-off under stronger assumptions.

Theorem 6. *If a silent protocol, that produces a valid labeling of the n agents and is safe with probability* 1, *uses $n + t$ states, where $t < n$, then the expected number of interactions required by the protocol to yield a valid labeling is at least* $\frac{n(n-1)}{2t+1}$.

Proof. Recall the notation and concepts given in the introductory part prior to the statement of Lemma 4, specifically for the run Z, the states $f_Z(x)$, $pred_Z(x)$, the sets A_Z, R_Z, and the set B_Z defined in the second paragraph of the proof of Theorem 5.

To prove the theorem, we need $|R_Z| \geq |A_Z|$ to hold for any run Z resulting in a valid labeling of the agents without updating the label of any single agent. The existence of such a run Z implied by Lemma 4 is insufficient to derive a lower bound on the expected number of required interactions. The stronger assumptions on the silent protocol in the statement of the theorem, requiring the protocol to always provide a valid labeling without altering the label of any single agent, solve the problem. Namely, if $pred_Z(x) = pred_Z(y)$ for $x, y \in A_Z$, where x reaches its final state no later than y does, then according to Remark 4, neither the transformed (x, y)-prefix $Z_1 i_1 Z_2 i_3$ (as specified before Remark 4) nor any of its extensions can yield a valid labeling without modifying the label of any single agent. We obtain a contradiction with the aforementioned assumptions. This shows that the inequality $|R_Z| \geq |A_Z|$ holds for any arbitrary run Z that concludes with a valid labeling without updating the label of any single agent.

We can assume w.l.o.g. that $|A_Z| < n$, as otherwise $t \geq |R_Z| \geq |A_Z| \geq n$. Accordingly, the set B_Z of agents is non-empty. Consider a last agent x in B_Z such that, while in the state $pred(x)$, it reaches its final state $f_Z(x)$ through an interaction with another agent y in a state s. If y is a member of B_Z then both x and y are the two last agents in B_Z that simultaneously achieve their final states in F_Z through the same interaction. The probability of the interaction between them is only $\frac{2}{n(n-1)}$. Suppose in turn that y belongs to A_Z. We know that $t \geq |R_Z| \geq |A_Z|$ from the preceding part. Therefore, there are at most t agents in A_Z in the state s that the agent x, in the state $pred_Z(x)$, could interact with. The probability of such an interaction is at most $t\frac{2}{n(n-1)}$. We conclude that the probability of an interaction between the agent x and the agent y, resulting in x reaching its final state $f_Z(x)$, is less than $\frac{2t+1}{n(n-1)}$. This proves the theorem. □

Corollary 2. *Fix $\varepsilon > 0$. If a silent protocol that produces a valid labeling of the n agents and is safe with probability 1 uses only $n + O(n^{1-\varepsilon})$ states then the expected number of interactions required by the protocol to achieve a valid labeling is $\Omega(n^{1+\varepsilon})$.*

5.2. A lower bound for pool protocols with the range $[1, n + r]$

Our fast protocols presented in Section 3 exemplify a class of natural protocols for the unique labeling problem, which we refer to as *pool protocols*.

At each step of a pool protocol, a subset of agents possesses explicit or implicit pools of labels. These are pairwise disjoint and their union falls within the assumed range of labels. When two agents interact, they can redistribute the combined contents of their pools between them. Initially, only one agent (the leader) possesses a pool of labels. This initial pool aligns with the assumed range of labels. Any agent can only be labeled with a label from its own pool. Once assigned, the label is removed from the pool and cannot be changed. Finally, an unlabeled agent cannot transfer its entire pool to another agent during an interaction without receiving a portion of the other agent's pool.

Theorem 7. *The expected number of interactions required by a pool protocol to assign unique labels in the range $[1, n + r]$, where $r \geq 0$, to the population of n agents is at least $\frac{n(n-1)}{2(r+1)}$.*

Proof. We shall say that an agent has the P property if the agent owns a non-empty pool or a label has been assigned to the agent. Note that if an agent attains the P property during the execution of a pool protocol, it keeps it indefinitely. Also note that all agents must eventually achieve the P property to successfully finish the label assignment task. In each interaction of a pool protocol at most one additional agent can acquire the P property. Since at the beginning only one agent possesses the P property, there must exist an interaction after which only one agent lacks this property. Due to the disjointedness of the pools and labels, the assumed label range, and the definition of a pool protocol, there exist at most $r+1$ agents that could contribute a sub-pool or label from their own pool to the agent lacking the P property. The expected number of interactions leading to an interaction between the agent lacking the P property and one of the at most $r + 1$ agents is not less than $\frac{n(n-1)}{2(r+1)}$. □

6. Final remarks

Our upper bound of $n + 5 \cdot \sqrt{n} + O(\log \log n)$ on the number of states achieved by a protocol for unique labeling that is silent and safe once a unique leader is elected almost matches our lower bound of $n + \sqrt{\frac{n-1}{2}} - 1$. We can integrate our protocols for unique labeling with the recent protocols for counting or approximating the population size due to Berenbrink et al. [12] in order to eliminate the assumption that the population size is known to the agents initially. Since the aforementioned protocols from [12] either require $\tilde{O}(n)$ states or $O(n \log^2 n)$ interactions, the resulting combinations lose some of the near-optimality or optimality properties of our protocols (cf. Corollary 1). The related question if one can design a protocol for counting or closely approximating the population size simultaneously requiring $O(n \log n)$ interactions w.h.p. and at most cn states, where c is a small constant, is of interest in its own right.

In the original model of population protocols [4], agents were equipped only with a constant number of states so an assignment of unique labels to them was not possible. The state restriction has been gradually relaxed and agents with $O(n)$ states could be assigned unique identifiers. The problem of assigning unique labels to agents is related to that of leader election [14]. The agent with the smallest label can be naturally identified as the leader. Thus, when the label range is minimal, a unique labeling preprocessing is strictly stronger than a leader election preprocessing [14]. But even in the case of small but not necessarily minimal label range, the labeling preprocessing provides a more overwhelming breaking of the symmetry of the agents than just dividing them into a leader and non-leaders. Interestingly, once $O(n)$ state space is available, several population protocols include some labeling mechanism. This concerns among other things the self-stabilizing task of leader election [15] and counting [8]. It is much easier to design and implement protocols that partition agents into groups which are assigned different tasks when the agents are uniquely labeled. For example, several fault-tolerant population protocols assume that agents are given unique names *a priori* [22]. More generally, a unique labeling preprocessing supports diversification of the states of the agents.

CRediT authorship contribution statement

Leszek Gąsieniec: Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Conceptualization. **Jesper Jansson:** Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Conceptualization. **Christos Levcopoulos:** Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Conceptualization. **Andrzej Lingas:** Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors are grateful to the anonymous reviewers for their valuable comments and suggestions. The research was in part supported by Swedish Research Council grant 621-2017-03750 and 2018-04001.

During the preparation of this manuscript the authors used Chat GPT-4 in a very limited way, in order to verify or improve some phrases in English. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Data availability

No data was used for the research described in the article.

References

[1] D. Alistarh, H. Attiya, S. Gilbert, A. Giurgiu, R. Guerraoui, Fast randomized test-and-set and renaming, in: Proc. of the International Symposium on Distributed Computing, DISC, in: Lecture Notes in Computer Science, vol. 6343, Springer, 2010, pp. 94–108.
[2] D. Alistarh, O. Denysyuk, L. Rodrigues, N. Shavit, Balls-into-leaves: sub-logarithmic renaming in synchronous message-passing systems, in: Proc. of the 2014 ACM Symposium on Principles of Distributed Computing, PODC, ACM, 2014, pp. 232–241.
[3] D. Alistarh, R. Gelashvili, Recent algorithmic advances in population protocols, ACM SIGACT News 49 (3) (2018) 63–73.
[4] D. Angluin, J. Aspnes, Z. Diamadi, M.J. Fischer, R. Peralta, Computation in networks of passively mobile finite-state sensors, Distrib. Comput. 18 (4) (2006) 235–253.
[5] D. Angluin, J. Aspnes, D. Eisenstat, Fast computation by population protocols with a leader, in: Proc. of the International Symposium on Distributed Computing, DISC, in: Lecture Notes in Computer Science, vol. 4167, Springer, 2006, pp. 61–75.
[6] J. Aspnes, J. Beauquier, J. Burman, D. Sohier, Time and space optimal counting in population protocols, in: Proc. of the 20th International Conference on Principles of Distributed Systems, OPODIS, in: LIPIcs, vol. 70, Dagstuhl - LZI, 2016, pp. 13:1–13:17.
[7] J. Beauquier, J. Burman, L. Rosaz, B. Rozoy, Non-deterministic population protocols, in: Proc. of the 20th International Conference on Principles of Distributed Systems, OPODIS, in: Lecture Notes in Computer Science, Springer, 2012, pp. 61–75.
[8] J. Beauquier, J. Clement, S. Messik, L. Rosaz, B. Rozoy, Self-stabilizing counting in mobile sensor networks with a base station, in: Proc. of International Symposium on Distributed Computing (DISC), in: Lecture Notes in Computer Science, Springer, 2007, pp. 63–76.
[9] P. Berenbrink, A. Brinkmann, R. Elsässer, T. Friedetzky, L. Nagel, Randomized renaming in shared memory systems, in: Proc. of the IEEE Int. Parallel and Distributed Processing Symp., IPDPS, IEEE Computer Society, 2015, pp. 542–549.
[10] P. Berenbrink, A. Czumaj, A. Steger, B. Vocking, Balanced allocations: the heavily loaded case, SIAM J. Comput. 35 (6) (2006) 1350–1385.
[11] P. Berenbrink, G. Giakkoupis, P. Kling, Optimal time and space leader election in population protocols, in: Proc. of the 52nd ACM-SIGACT Symposium on Theory of Computing, STOC, ACM, 2020, pp. 119–129.
[12] P. Berenbrink, D. Kaaser, T. Radzik, On counting the population size, in: Proc. of the ACM Symposium on Principles of Distributed Computing, PODC, ACM, 2019, pp. 43–52.
[13] J. Burman, J. Beauquier, D. Sohier, Space-optimal naming in population protocols, in: Proc. of the 33rd International Symposium on Distributed Computing, DISC, in: LIPIcs, vol. 146, Dagstuhl - LZI, 2019, pp. 9:1–9:16.
[14] J. Burman, H. Chen, H. Chen, D. Doty, T. Nowak, E. Severson, C. Xu, Time-optimal self-stabilizing leader election in population protocols, in: Proc. of the ACM Symposium on Principles of Distributed Computing, PODC, ACM, 2021, pp. 33–44.

[15] S. Cai, T. Izumi, K. Wada, How to prove impossibility under global fairness: on space complexity of self-stabilizing leader election on a population protocol model, Theory Comput. Syst. 50 (2012) 433–445.
[16] A. Castañeda, S. Rajsbaum, M. Raynal, The renaming problem in shared memory systems: an introduction, Comput. Sci. Rev. 5 (3) (2011) 229–251.
[17] S. Dolev, M.G. Gouda, M. Schneider, Memory requirements for silent stabilization, Acta Inform. 36 (6) (1999) 447–462.
[18] D. Doty, M. Eftekhari, A survey of size counting in population protocols, Theor. Comput. Sci. 894 (2021) 91–102.
[19] D. Doty, M. Eftekhari, O. Michail, P.G. Spirakis, M. Theofilatos, Exact size counting in uniform population protocols in nearly logarithmic time, arXiv preprint, arXiv:1805.04832, 2018.
[20] R. Elsässer, T. Radzik, Recent results in population protocols for exact majority and leader election, Bull. Eur. Assoc. Theor. Comput. Sci. 126 (2018).
[21] L. Gąsieniec, G. Stachowiak, Enhanced phase clocks, population protocols, and fast space optimal leader election, J. ACM 68 (1) (2021) 2.
[22] R. Guerraoui, E. Ruppert, Names trump malice: tiny mobile agents can tolerate Byzantine failures, in: Proc. of ICALP (2) 2009, in: LNCS, Springer, 2009, pp. 484–495.
[23] S. Janson, Tail bounds for sums of geometric and exponential variables, Stat. Probab. Lett. 135 (April 2018) 1–6.