Space-Optimal Counting in Population Protocols [Extended Version]

Joffroy Beauquier, Janna Burman, Simon Clavière, Devan Sohier

Space-Optimal Counting in Population Protocols

[Extended Version]

Joffroy Beauquier[1⋆], Janna Burman[1⋆⋆], Simon Clavière[2], and Devan Sohier[2]

[1] LRI - CNRS UMR-8623, Université Paris Sud, France
{joffroy.beauquier, janna.burman}@lri.fr
[2] PRiSM - CNRS FRE-3709, Université de Versailles, France
{simon.claviere, devan.sohier}@prism.uvsq.fr

Abstract. In this paper, we study the fundamental problem of *counting*, which consists in computing the size of a system. We consider the distributed communication model of *population protocols* of finite state, anonymous and asynchronous mobile devices (*agents*) communicating in pairs (according to a *fairness* condition). This work significantly improves the previous results known for counting in this model, in terms of (exact) space complexity. We present, prove correct and analyze the time complexities of the first space-optimal protocols solving the problem for two classical types of fairness, *global* and *weak*. Both protocols require no initialization of the counted agents.
The protocol designed for global fairness, surprisingly, uses only one bit of memory (two states) per counted agent. The protocol, functioning under weak fairness, requires the necessary $\log P$ bits (P states, per counted agent) to be able to count up to P agents. Interestingly, this protocol exploits the intriguing Gros sequence of natural numbers, which is also used in the solutions to the Chinese Rings and the Hanoi Towers puzzles. The convergence time of the protocol is exponential, but we prove that this is necessary (within a constant factor) for obtaining a space-optimal solution.

⋆ The work of this author was partially supported by the Israeli-French Maimonide research project.

⋆⋆ Contact Author Details: Laboratoire de Recherche en Informatique, Bât 650, Université Paris Sud, 91405 Orsay Cedex, France, tel: +33 (0)1 69 15 68 60, {janna.burman}@lri.fr. The work of this author was partially supported by the Israeli-French Maimonide and the INS2I PEPS JCJC research projects.

1 Introduction

Counting is a fundamental task in computer science, as illustrated by numerous and important applications of this paradigm in many domains, like network traffic monitoring, database query optimization, or data mining. The context of this work is that of dynamic wireless ad-hoc networks. In this context, many efficient counting protocols have been proposed recently (e.g., [17, 22, 24, 26]).

More precisely, we consider large-scale ad-hoc networks of mobile sensors, in which cheap and tiny devices, with limited communication, memory and computation power, move around and cooperate for achieving some task. Such networks are of an unknown size, fundamentally asynchronous (no common clock), anonymous (no identifiers) and not permanently connected (due to communication limitation). The design of these networks is now focused on complex collections of heterogeneous devices that should be robust, adaptive and self-organizing, serving requests that vary with time. There are many reasons for these devices to fail: extreme external conditions of temperature or pressure, battery exhaustion, failures inherent to their cheap realization, etc. The ability to count them (e.g., for, possibly, replacing some) may be crucial for ensuring that the tasks are performed efficiently. In this work, we propose solutions to this problem, concerning especially the reliability and the size requirements of the memory of the network nodes.

To be able to analyze our solutions, we adopt a formal communication model that suits the considered networks. This is the model of *population protocols* (PP) [3]. In PP, mobile devices, called agents, are anonymous, undistinguishable and asynchronous. Each agent has a finite state, that evolves over the course of interactions. When two agents are sufficiently close one to the other, they interact, and the effect of the interaction is a change of their states. The mobility is modeled in a very general way, by a fairness assumption which is called *global fairness*. In addition to this original fairness of PP, we consider also a classical type of fairness for distributed computing, which we call here *weak*. While global fairness captures the randomization inherent to many real systems, weak fairness only ensures progress of system entities. In general, PP is well adapted to dynamic networks in which the topology changes (like in peer-to-peer networks), or to networks in which nodes move unpredictably (like in mobile sensor networks).

The objective of this paper is to make a step towards a better understanding of the possibilities and limitations of such networks, in studying the feasibility and the complexity of the fundamental task of counting the number of agents. The task of counting anonymous agents in PP has already been studied and several results are known. Basically, we improve these results in terms of exact space complexity. Moreover, the solutions we give are space optimal. Space is a crucial factor, since a low memory is a basic condition in large-scale and unreliable networks.

Current and previous studies on counting in PP consider various parameters of the model that affect attractivity, efficiency and feasibility of the solutions. We list and explain them below together with the related impossibility results:

- The first parameter is the nature of the *fairness*: global or weak. We consider both cases, as already explained above. See formal definitions in Sec. 2.
- The second parameter is the requirement of *initialization* of agent states. On one hand, efficient protocols for dynamic and unreliable networks should not require initialization. There are at least two reasons for that. First, the agents are cheap and prone to failures. So, it should be expected that some memory or communication errors happen. Second, in dynamic and unreliable environments, it should be possible to execute most of the tasks, and counting too, in a repetitive way. In both cases, re-initializing the network could be a real problem. Moreover, it is generally hard to know when such a re-initialization should be done, as termination detection is generally difficult to obtain in such networks.

On the other hand, if no agent state can be initialized, it is impossible to realize counting in PP, under weak or global fairness. This can be proven by using a classical technique of network partitioning (see Prop. 1 in the appendix).
Thus, to be able to solve the problem and still avoid initialization, all previous works, as well as the current one, assume the initialization of *only one* particular (and thus distinguishable) agent called the *base station* (BS).

- For defining the data structures used by finite-state agents in the solutions, all previous studies assume the existence of a known *upper bound P on the number of (non-BS) agents*. The space complexity of the solution is then expressed as a function of the necessary number of states per agent with respect to P. This is justified in the case of weak fairness, since it has been proved in [8] that P (or more) agents cannot be counted with strictly less than P states per agent by deterministic protocols (considered here as well). However, in case of global fairness, we show that this assumption is not needed, by presenting a protocol using only two states per agent.
- Finally, population protocols may be *symmetric* or *asymmetric*. In symmetric protocols, two agents in an interaction (and thus in the corresponding transition) are indistinguishable if their states are identical. Thus, their states are identical also after the transition. In asymmetric protocols, two agents in an interaction can be always distinguished (e.g., there is always an initiator and a responder in the interaction). Our study considers the more difficult and general case of symmetric protocols. Such protocols can be deployed in networks with either symmetric or asymmetric communications.

Most Related Work. Before presenting the contributions, we summarize the previous results about counting in symmetric PP. For the reasons explained above, all these results assume a distinguishable agent BS and do not require any initialization of non-BS agents. Moreover, BS is considered to be a powerful device, so its resources are in general not concerned by the protocol design.
The authors of [17] were the first to be concerned with a space efficient design for counting in PP. Following the results of [8], they improved them from $4P$ to $2P$ states per non-BS agent under weak fairness, and to $\frac{3}{2}P$ under global fairness. This latter result for global fairness was improved to P in [7].
Note that an asymmetric population protocol can be transformed into a symmetric one using the transformer of [9]. However, this transformer requires global fairness and doubles the number of states per agent. This makes it inadequate for obtaining a space efficient symmetric solution from an asymmetric one (in terms of exact space).

Contributions. For the first time, we present and prove correct two space-optimal symmetric population protocols solving the counting problem. One solves the problem under global fairness, and uses only one bit of memory (two states) per non-BS agent (Protocol 1, Sec. 3). It is shown that one agent state is not enough to solve the problem. The other protocol, designed for weak fairness, uses only the necessary P states per non-BS agent (Protocol 2, Sec. 4). Both protocols do not assume any initialization of the counted agents, but the necessary initialization of BS. The protocol assuming weak fairness is *silent* (i.e., no state changes after convergence). However, we show that no silent space-optimal counting protocol exists in our framework under global fairness.

The protocols are given with their time complexity analysis. The model of population protocols being inherently asynchronous and the definitions of the weak (and global) fairness including no bounds, the time complexity must be computed in less classical terms. With weak fairness, it can be done using *non-null transitions* (the ones that change agent states, i.e., make some progress towards the solution), or in terms of *asynchronous rounds* (the shortest fragments of execution where each agent interacts with all the others). We analyze the solution under weak fairness with

these complexity measures, and found that it is exponential. Together with that, we prove that such convergence time is necessary (within a constant factor) for obtaining a space-optimal solution.

For the protocol under global fairness, as is generally done, we give a probabilistic time complexity analysis assuming that the communications between agents follow a probabilistic uniform law. Note that, in this case, the analysis in terms of non-null transitions or in terms of rounds would yield an infinite time complexity, due to the nature of global fairness.

Other Related Work. Apart from the works already mentioned in the context of PP, there are many others related to counting in related models. Many, like [22, 26, 13, 24, 23], consider the synchronous model of dynamic graph. In this model, a computation proceeds by synchronous rounds and, for each round, an adversary chooses the links available for sending messages. Similarly to our case, in the cited works, all nodes execute the same code and have no information about the network (in most cases). In addition, all, except [22], assume anonymous nodes having no unique identifiers. However, in contrast with this work, all nodes have to be initialized, and authors are concerned with *asymptotic* complexity in terms of rounds, bits and messages. All, but [13], study counting. [13] studies a related problem of assigning (short) labels to nodes.
The problem of counting *approximatively* the number of nodes in a network, using probabilities, is known under the term of *size estimation*. A common approach to network size estimation is to use random walks [28, 15] relying on a token being passed around the network to collect information each time it visits an agent. Another strategy is to use randomly generated numbers [21], and then exploit classical results on order statistics to infer the number of participants [6, 30].
In the context of large scale peer-to-peer and dynamic networks in general, probabilistic and gossiping methods have also been proposed for estimating the size of the network [25, 14, 19, 27, 21].
Another problem related to counting is the *resource controller* problem, introduced in [1] and optimized in [20, 12]. The main difference with our model is that topological changes can be delayed until permission has been granted by the controller.
To summarize, the most significant differences of the works mentioned in this section with the current work is that we consider a totally asynchronous model of finite state anonymous and non-initialized deterministic processes. Moreover, in the considered model, termination detection is difficult and in many cases impossible. This makes sequential composition of protocols challenging.

2 Model and Notations

As a basic model we use the model of *population protocols* of Angluin et al. [5] with some adaption as detailed below. In this model, a system consists of a collection $\mathcal{A}$ of pairwise interacting agents, also called a population. Each agent represents a finite state sensing and communicating mobile device. Among the agents, there may be a distinguishable agent called the *base station* (BS), which can be as powerful as needed, in contrast with the resource-limited non-BS agents. The non-BS agents are also called *mobile*, interchangeably. The size of the population is the number of mobile agents, denoted by N, and is unknown (a priori) to the agents.

A *protocol* in this model (a population protocol) can be modeled as a *finite* transition system whose states are called *configurations*. A *configuration* is a function that associates each agent with its state. Each agent has a state taken from a finite set of states, the same for all mobile agents (denoted Q), but generally different for BS.

In this transition system, every transition between two configurations is modeled by a *transition* between two agents happening during an interaction. That is, when two agents x, in state p, and y, in state q, interact (meet), they execute a transition $(p, q) \to (p', q')$. As a result, x changes its state from p to p' and y from q to q'. If $p = p'$ and $q = q'$, the corresponding transition is called *null* (such

transitions are specified by default), and non-null otherwise.[3] The transitions are *deterministic*, if for every pair of states (p, q), there is exactly one (p', q') such that $(p, q) \to (p', q')$. We consider only deterministic transitions and thus, only *deterministic protocols*. Transitions and protocols can be *symmetric* or *asymmetric*. Symmetric means that, if $(p, q) \to (p', q')$ is a possible transition, then $(q, p) \to (q', p')$ is also a possible transition. In particular, if $(p, p) \to (p', q')$ is symmetric, $p' = q'$. Asymmetric is the contrary of symmetric.

Let C and C' be configurations. Then, $C \to C'$ is a transition (between two configurations), if C' can be obtained from C by a single transition of two agents in an interaction. This means that C contains two states p and q and C' is obtained from C by replacing p and q by p' and q' respectively, where $(p, q) \to (p', q')$ is a transition. If there is a sequence of configurations $C = C_0, C_1, \ldots, C_k = C'$, such that $C_i \to C_{i+1}$ for all $i, 0 \leq i < k$, we say that C' *is reachable* from C, denoted $C \xrightarrow{*} C'$.

An *execution* of a protocol is an infinite sequence of configurations $C_0, C_1, C_2, \ldots$ such that C_0 is the starting configuration and for each $i \geq 0$, $C_i \to C_{i+1}$. In a real distributed execution, interactions could take place simultaneously, but when writing down an execution we can order those simultaneous interactions arbitrarily. An *asynchronous round* (or simply, a round) is defined as a shortest segment in an execution where each agent interacts with every other.

An execution is said *weakly fair*, if every pair of agents in $\mathcal{A}$ interacts infinitely often. An execution is said *globally fair*, if for every two configurations C and C' such that $C \to C'$, if C occurs infinitely often in the execution, then C' also occurs infinitely often in the execution. This definition together with the finite state space assumption, implies that, if in an execution there is an infinitely often reachable configuration, then it is infinitely often reached [4]. Global fairness can be viewed as simulating randomized systems (without introducing randomization explicitly) [18].

A *problem* is defined by a predicate $\mathcal{D}$ on executions. A population protocol $\mathcal{PP}$ is said to *solve a problem* $\mathcal{D}$, if and only if every execution of $\mathcal{PP}$ satisfies the conditions defining $\mathcal{D}$. The problem of *counting* is defined by the following conditions: eventually, in any execution, there is at least one agent (BS, in our case) obtaining a value of N in some variable and this value does not change. Note that the counting predicate is required to be satisfied only eventually (and forever after). When it happens, we say that the protocol has *converged*. The *convergence time* of a protocol in terms of (non-null) transitions, or in terms of asynchronous rounds, is defined by the maximum possible number of (non-null) transitions, or (respectively) rounds, in an execution, till convergence. We consider only *semi-uniform* protocols in the sense that the size of the population N is not used by a protocol and all agents, except BS, are (a priori) indistinguishable and interact according to the same possible transitions [11, 29]. A protocol is called *silent*, if in any execution, eventually, no state of an agent changes [10].

For simplicity, we do not present the rules of our protocols under the form of possible transitions, but under the equivalent form of a pseudo-code.

3 Space-Optimal Counting under Global Fairness

In this section, we present a space-optimal protocol (Protocol 1 below) solving the counting problem under global fairness. The protocol uses only one bit of memory, i.e., only two states per mobile (non-BS) agent. Its convergence time is $\Theta(2^N)$ transitions in average (see below).

[3] In general, during an interaction, both agent states are required for a transition and a possible application have to provide this. However, in case of interactions with BS, the computations can be done on the side of BS. The non-BS agent only receives from BS the resulting state change of the transition. Moreover, in the protocols of this work, the maximum that the two mobile interacting agents have to be able to do is to compare their states, for detecting the identical ones.
In any case, here and as is done in the previous related studies, the size of Q is considered as a metric of the space complexity analysis.

It is easy to see that with only one state per mobile agent, counting is impossible. Indeed, in this case, BS cannot distinguish between populations of one or more mobile agents (see the proof of Prop. 2 in the appendix). In addition, a partitioning argument can be used to show why no silent (uniform) counting protocol exists with only two states per agent (see the proof of Prop. 3 in the appendix).

Protocol 1 Description. Each mobile agent x has one bit $mark_x$, which is flipped at each interaction of x with BS. Between any two mobile agents, there are only null transitions. BS maintains a variable $size\_total_{BS}$ that eventually and forever holds the size of the population N. In addition, it maintains an array $size_{BS}[2]$ of two elements, where $size_{BS}[0]$ holds an estimation for the number of mobile agents currently marked 0 (i.e., with $mark = 0$), and similarly, $size_{BS}[1]$ estimates the number of agents currently marked 1. Eventually, these estimations become correct forever and $size\_total_{BS}$ too, because the latter is computed at each transition as the sum of $size_{BS}[0]$ and $size_{BS}[1]$ (line 6). The protocol itself can be described in a simple way. Whenever an agent marked 0 interacts with BS, BS flips its mark (to 1), decrements the estimation of 0 marked agents, i.e., $size_{BS}[0]$ (if it is not 0), and increases the estimation of 1 marked agents, i.e., $size_{BS}[1]$ (similarly for an agent marked 1).

The idea behind this solution is to try to reach a configuration, using the force of global fairness, where all agents are marked similarly, let us say, by 0 (the proof of Theorem 1 shows that it is possible). From such a configuration, there is always a *possible* segment of execution where each agent x interacts with BS, exactly once. In each such interaction, the mark of x is flipped, to "remember" that it has been "counted". By the end of such an execution segment, all agents are marked 1 (i.e., as "counted"). Moreover, both estimations of the number of agents marked 1 and 0 in $size_{BS}[1]$ and in $size_{BS}[0]$, respectively, are correct and stay correct from this moment on. Thus, the estimation of the size of the population (in $size\_total_{BS}$) becomes also correct. By global fairness, the scenario above appears in any execution of Protocol 1 (Theorem 1).

Protocol 1 Space-Optimal Counting under Global Fairness (one bit per agent)

Variables at BS:
$size_{BS}[2]$: array of two non-negative integers, initialized to 0
$size\_total_{BS}$: non-negative integer initialized to 0; eventually holds N
Variable at a mobile agent x:
$mark_x$: in $\{0, 1\}$, initialized *arbitrarily*

1: **when** a mobile agent x interacts with BS **do**
2: **if** $size[mark_x] > 0$ **then**
3: $size[mark_x] \leftarrow size[mark_x] - 1$
4: $mark_x \leftarrow 1 - mark_x$
5: $size[mark_x] \leftarrow size[mark_x] + 1$
6: $size\_total_{BS} \leftarrow size_{BS}[0] + size_{BS}[1]$

Correctness of Protocol 1. Let us denote by $\#0(C)$, respectively $\#1(C)$, the number of agents marked 0 (i.e., with $mark = 0$), respectively 1, in a configuration C. The proof of Lemma 1 is done by induction on the index of a configuration in an execution

Lemma 1. *For every configuration C, $size_{BS}[0] \leq \#0(C)$ (resp. $size_{BS}[1] \leq \#1(C)$).*

Proof. First, let us prove the lemma for $size_{BS}[0]$. We prove by induction on the index $k \geq 0$ of a configuration in an execution $(C_0, C_1, C_2, \ldots, C_k, \ldots)$. At the starting configuration C_0, $k = 0$, the lemma holds because of the initialization of $size_{BS}[0]$ to 0. Let us assume that the lemma holds

for $k = k'$ and prove it for $k = k' + 1$. Then, $size_{BS}[0] \leq \#0(C_{k'})$. From any configuration, and from $C_{k'}$ in particular, the only possible interaction (BS, x) is of two types, either x is marked 0 ($mark_x = 0$), or 1:
- If x is marked 0, during the following transition, its mark is flipped to 1 (line 4) and thus $\#0(C_{k'+1}) = \#0(C_{k'}) - 1$. At line 3, $size_{BS}[0]$ is decremented too (if it is not 0), and this is the only line that changes $size_{BS}[0]$ in this transition (line 5 changes $size_{BS}[1]$). Thus, after this transition, in $C_{k'+1}$, $size_{BS}[0] \leq \#0(C_{k'+1})$.
- If, during an interaction (BS, x) at $C_{k'}$, x is marked 1, during the following transition, its mark is flipped to 0 (line 4) and thus $\#0(C_{k'+1}) = \#0(C_{k'}) + 1$. At line 5, $size_{BS}[0]$ is incremented too, and this is the only line that changes $size_{BS}[0]$ in this transition (line 3 changes $size_{BS}[1]$).Thus, after this transition, in $C_{k'+1}$, $size_{BS}[0] \leq \#0(C_{k'+1})$.
Thus, the lemma holds for $size_{BS}[0]$. As $size_{BS}[1]$ is managed exactly in the same (but symmetric) way as $size_{BS}[0]$, the lemma also holds for $size_{BS}[1]$. □

As $size\_total_{BS}$ is always set to the sum of $size_{BS}[0]$ and $size_{BS}[1]$ (line 6), we have the following corollary. Lemma 2 below is easily obtained by observing the pseudo-code.

Corollary 1. *In any configuration, $size\_total_{BS} \leq N$.*

Lemma 2. *The value of $size\_total_{BS}$ never decreases.*

Proof. The value of $size\_total_{BS}$ can decrease only by executing line 3, $size[mark_x] \leftarrow size[mark_x] - 1$. Whenever this line is executed in a transition, line 5 is executed in the same transition too. Due to line 4, in line 5, $size[1 - mark_x] \leftarrow size[1 - mark_x] + 1$. Thus, if line 3 is executed in some transition, $size\_total_{BS}$ does not change in this transition. In all other cases, it can only increase. □

Theorem 1. *Under global fairness, (symmetric) Protocol 1 solves the counting problem. Eventually, $size\_total_{BS} = N$ and does not change anymore.*

Proof. To prove the theorem, we show below that, from any possible configuration, there is a reachable configuration C^* s.t., in C^*, $size\_total_{BS} = N$. Then, by global fairness, such configuration is eventually reached. Finally, by corollary 1 and lemma 2, we have $size\_total_{BS} = N$ in all subsequent configurations.

Now we show why C^* is always reachable. Consider a configuration C. In C, let $size_{BS}[0] = x_0, size_{BS}[1] = x_1$, where x_0, x_1 are non-negative integers $\leq N$. By lemma 1, there are $0 \leq x'_0, x'_1 \leq N$ s.t. $\#0(C) = x_0 + x'_0$ and $\#1(C) = x_1 + x'_1$. Then, from C, there is the following possible execution (that reaches C^*). First, $x_1 + x'_1$ agents marked 1 interact with BS, each one exactly once. It is easy to verify, by the code of Protocol 1, that at the end of this segment of execution, $size_{BS}[0] = x_0 + x_1 + x'_1, size_{BS}[1] = 0$ and $\#0(C) = x_0 + x'_0 + x_1 + x'_1 = N, \#1(C) = 0$ (all agents are marked 0). Now, $x_0 + x_1 + x'_1$ agents interact with BS (each one exactly once), what results in $size_{BS}[0] = 0, size_{BS}[1] = x_0 + x_1 + x'_1$ and $\#0(C) = x'_0, \#1(C) = x_0 + x_1 + x'_1$. Finally, x'_0 agents marked 0 interact with BS, each one exactly once. Now, $size_{BS}[0] = 0, size_{BS}[1] = x_0 + x'_0 + x_1 + x'_1 = N$ and $\#0(C) = 0, \#1(C) = x_0 + x'_0 + x_1 + x'_1 = N$ (all agents are marked 1). In this configuration, $size\_total_{BS} = N$, and thus C^* is reachable from (any configuration) C. □

Time Complexity Analysis. Global fairness only applies to infinite schedules and expresses a condition on which there are no bounds. In consequence, a complexity study in terms of real time, the number of interactions, or even in terms of rounds or non-null transitions would yield an infinite complexity. On the other hand, there is a strong relation between global fairness and randomization (of interactions) [18]. This is why, generally in population protocols the time complexity analysis is

based on a probabilistic law on the interactions between agents. Since it is anyhow an approximation, and for the sake of simplicity, the uniform law is used. We follow a similar approach and evaluate the convergence time of Protocol 1 in term of the average number of transitions. Below, we sketch the analysis and we give the details in the appendix.

To calculate the convergence time, we use the observation (stated by Lemma 11 appeared and proven in the appendix) that Protocol 1 must first reach a configuration with all mobile agents in the same state, and then a configuration with all the agents in the other state (recall that there are only two mobile agent states).

Thus, consider a population of N agents, and let t_k be the average number of transitions that happen before all agents are in state 0, starting from a configuration with k agents in state 1. Then, $t_0 = 0$ trivially. For $1 \leq k \leq N-1$, $t_k = 1 + \frac{k}{n} t_{k-1} + \frac{N-k}{N} t_{k+1}$. This is because, at the current step, there are k chances out of N that an agent in state 1 meets BS, leading to a configuration with $k-1$ agents in state 1; and $N-k$ chances out of N that an agent in state 0 interacts with BS resulting in a configuration with $k+1$ agents in state 1. Finally, $t_N = 1 + t_{N-1}$.

From that we have $t_N = 2^{N-1} \sum_{k=0}^{N-1} \frac{1}{\binom{N-1}{k}} = 2^N + o(2^N)$ (see the appendix for details), and the average convergence time of Protocol 1 is $\Theta(2^N)$ transitions.

4 Space-Optimal Counting under Weak Fairness

In this section, we present a silent symmetric space-optimal protocol (Protocol 2 below) solving the counting problem under weak fairness (see Theorem 2 and Corollary 2). Its convergence time is $\Theta(2^N)$ (Corollary 3), and we prove that such a convergence time is necessary, within a constant factor, for obtaining a space-optimal solution. The protocol is correct starting from arbitrary states in mobile agents, but BS. It uses at most P states per agent, which is *necessary* in the current conditions for solving counting in populations with at most P mobile agents ($N \leq P$) [8].

Protocol 2 General Description. In this protocol, BS eventually counts the mobile agents and stores the value in variable n. To realize this, BS successively attempts to guess the number of mobile agents in the population, starting from 1 and ending with N (this guess is stored in n). For each guess $n < P$, BS tries to name (differently) mobile agents in state 0 (zero-state) interacting with BS (lines 3 and 9). That is, BS tries to assign to these agents distinct states from $\{1, \ldots, n\}$ (also called here names). State 0 has a special technical role. Whenever two agents with identical names (*homonyms*) interact, they change their state to 0 (line 12). Thus, this state indicates to BS that, either it has created homonyms before, or that homonyms (or, simply, agents in state 0) existed already in the population in the starting configuration.

Thus, zero-state mobile agents are named by BS. The names are given one by one following some finite sequence U^* of names (line 9). For simplicity, in the presented protocol, this sequence is computed in advance and depends on P. However, for an optimized version, the required prefix of U^*, U_N, can be computed on the fly, during an execution (see Remark 1). Sequence U^* guarantees that, if there are $N < P$ agents, whatever their starting states are, the naming succeeds. If no naming succeeds, BS concludes that there are more than $P-1$ agents, that is $N = P$. Thus, the protocol actually realizes a (consecutive minimal) naming for any $N < P$ in order to realize finally a counting for any $N \leq P$.

Another important property of U^* is that, for every guess n, if all the terms of U^*, from the first to some l_n^{th} term, have been used by BS to name interacting agents, then BS can conclude that the guess of n is wrong. It is safe then to switch to the next guess $n+1$ (line 8). In the sequel, we denote the prefix of U^* of length l_n by U_n ($l_n = |U_n|$). Any term of U_n is in $\{1, \ldots, n\}$. Thus, if BS meets an agent in a state $> n$, it can conclude that it has never seen this agent before. Hence, it can safely deduce that $N > n$, and switch to the next guess $n+1$ (lines 5 - 8).

As long as there are agents in state 0 or in a state $> n$, and $n < P$ (line 2), the base station continues renaming and counting, because all these agents will eventually interact with BS (by weak fairness). If there are homonyms, eventually they meet too and switch to state 0 (again, by fairness).

Protocol 2 Space-Optimal Counting under Weak Fairness (P states per agent)

Variables at BS:
n: non-negative integer initialized to 0 // guess of the population size; eventually holds N
k: non-negative integer initialized to 0 // pointer to the k^{th} element of U^*
Shortcuts at BS:
U^*: constant sequence of elements in $[1, \ldots, P-1]$ computed in advance by the recursion $U_1 \equiv 1, U^* \equiv U_{P-1} \equiv U_{P-2}, P-1, U_{P-2}$
$U^*(k)$: returns the k^{th} element of U^*
$l_n = 2^n - 1$ ($\equiv |U_n|$)

Variable at a mobile agent x:
$name_x$: non-negative integer in $[0, \ldots, P-1]$, initialized *arbitrarily*

1: **when** a mobile agent x interacts with BS **do**
2: **if** $n < P \wedge (name_x = 0 \vee name_x > n)$ **then**
3: **if** $name_x = 0$ **then**
4: $k \leftarrow k + 1$ // advance k to point to the next element of U^*
5: **else if** $name_x > n$ **then**
6: $k \leftarrow l_n + 1$ // because agent x with a name $> n$ could not be seen before by BS, the population must be larger than n, so k is advanced accordingly
7: **if** $k > l_n$ **then**
8: $n \leftarrow n + 1$ // pointer k indicates that the population is larger
9: $name_x \leftarrow U^*(k)$ // set the name of x to the the k^{th} element of U^*
10: **when** two mobile agents x and y interact **do**
11: **if** $name_x = name_y$ **then**
12: $name_x \leftarrow name_y \leftarrow 0$ // set homonym states to 0

4.1 Naming Sequence U^* - the Gros Sequence

As a matter of fact, sequence U^* is not unique. We choose and define one of the possible such sequences. We also prove the properties claimed about it above.
To define the sequence U^*, we consider the infinite sequence U_∞, whose left prefix U_n is defined recursively by $U_n \equiv U_{n-1}, n, U_{n-1}$, where $U_1 \equiv 1$.
Sequence U^* is obtained for $n = P - 1$, i.e., $U^* \equiv U_{P-1} \equiv U_{P-2}, P-1, U_{P-2}$.
For example, the prefix U_4 of U_∞ is: $1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, 1, 2, 1$.

Let $l_n \equiv |U_n|$. By construction of sequence U_∞, $l_1 = 1$, and $l_{n+1} = 2l_n + 1$, which gives $l_n = 2^n - 1$. Then, using the recursive definition of U_∞ we obtain that $\forall n, U_\infty(2^n) = n + 1$ and $\forall n, \forall 1 \leq k < 2^n, U_\infty(2^n + k) = U_\infty(k)$.

Remark 1. Based on this alternative description, U_∞, and U^* in particular, can be defined iteratively. The k^{th} term of U_∞ is one plus the index of the least significant non-zero bit in the binary decomposition of k. Thus, BS does not need to store the whole sequence of names in advance. It can compute the next state to assign to a mobile agent based on a single integer variable. Such computation of the sequence does not depend on P, but on the number of the sequence terms which will be actually used. For this sequence, the number of terms used to name n agents is at most

$l_n = 2^n - 1$. In consequence, the number of interactions (before convergence) between BS and an agent in state 0 or $> n$ is at most l_N.

Remark 2. It appears that U_∞ is known in the literature under the name of Gros sequence. This sequence can be found all over mathematics. It has remarkable properties with respect to the binary numeration, generating a Gray code. It encodes an Hamiltonian cycle on the edges of a n-dimensional cube. It is also the "greediest" square-free sequence (if one builds the sequence in choosing at each step the smallest integer that does not produce a square). Finally, the Gros sequence solves the Chinise Rings puzzle and, surprisingly, solved the Tower of Hanoi puzzle long before the latter was at all invented. For details refer to [16, 2].

One of the intuitions behind the use of the Gros sequence for counting is related to the Hamiltonian cycle property on a cube. Consider a multi-dimensional cube whose vertices are labeled by the multi-sets of n names and edges connect vertices that differ by exactly one name. Whatever the initial names are (the agents can be arbitrarily initialized), the Gros sequence leads, by traveling along the Hamiltonian cycle it encodes, to the vertex where all names are distinct. In the corresponding configuration the counting can be performed.

Now we give a more precise, but also more technical, explanation why, by using this particular sequence U^*, BS correctly counts $N (\leq P)$ agents. Consider the prefix $U_n = U_{n-1}, n, U_{n-1}$ of U^*. By assigning successively the numbers given by U_n, and in particular by the prefix U_{n-1}, BS can assign *distinct* names from $\{1, \ldots, n-1\}$ to *all* agents, only if $N \leq n-1$. If it is not the case ($N > n-1$), BS eventually detects it whenever it meets an agent x, either in state $> n-1$, or in state 0 *after* the last name in U_{n-1} has been assigned (i.e., homonyms still exist). Then, BS guesses that $N = n$, and continues naming with the sub-sequence (n, U_{n-1}). That is, it assigns state n to agent x which becomes unique, if effectively $N = n$. If this is the case, BS should successfully rename the remaining $n-1$ agents with $n-1$ states from $\{1, \ldots, n-1\}$, following, once again, the naming sequence defined by U_{n-1}. From now on, the procedure repeats for the sequence $U_{n+1} = U_n, n+1, U_n$. If the guess of $N = n$ was wrong, BS eventually detects it (at least, by the end of the prefix U_n), and switches to guess $n+1$. That is, it will continue naming according to $(n+1, U_n)$. This continues until the guess of BS is correct, or till all attempts have failed, meaning that $N = P$.

Remark 3. We would like to better put into perspective the strategy of Protocol 2 to resolve the name conflicts, in explaining the similarities and the differences between the strategy here and of the protocol in [17] using $2P$ states (in the same conditions). First, intuitively, as BS cannot detect homonyms, this have to be the responsibility of the mobile agents. In both considered protocols, mobile agents inform BS about detected homonyms. In Protocol 2, they do it by a special state 0, and in [17], by a special state D_i ($i \in \{1 \ldots P\}$), which also indicates the specific state S_i of the detected homonyms. This helps BS to effectively assign new names and advance towards the solution. On the contrary, in Protocol 2, there is no available state space in mobile agents to provide any additional useful information to BS about the specific states of the detected homonyms. Thus, BS cannot name the agents efficiently and have to use a sort of exhaustive renaming, using the recursive Gros sequence. These intuitions have a formal basis proven in Sec. 4.3.

4.2 Correctness of Protocol 2

In the proofs below, we consider a set E of non-zero-states associated to a configuration C s.t., for every $s \in E$, the number of mobile agents in state s in C is odd. This allows to focus only on the transitions involving BS, and not on transitions between homonyms, which will happen eventually and do not change the parity of the number of agents in any state. Moreover, for any

$E, E' \subseteq \{1, \ldots, n\}$, we denote by $E \triangle E' \equiv E \cup E' - E \cap E'$ their symmetric difference. In particular, $E \triangle \{e\}$ ($e \in \{1, \ldots, n\}$) is $E \cup \{e\}$ if $e \notin E$, and $E - \{e\}$ if $e \in E$.

In Lemma 3 below, we prove that when the sequence U^* is used by the protocol, it guarantees that, if $N < P$, E evolves until $E = \{1, \ldots, N\}$, where all mobile agents have distinct names. Then, using Lemma 3 we obtain the main Theorem 2.

Lemma 3. *Let $E_0 \subset \{1, \ldots, n\}$ and $E_{k+1} = E_k \triangle \{U_\infty(k+1)\}$. There exists some $1 \leq j \leq 2^n - 1$ such that $E_j = \{1, \ldots, n\}$.*

Proof. Let $\mathcal{H}_n$ ($n \in \mathbb{N}$) be the induction hypothesis *"for any subset $E_0 \subset \{1, \ldots, n\}$ and such that $E_{k+1} = E_k \triangle \{U_\infty(k+1)\}$, there is $E_j = \{1, \ldots, n\}$ for some $1 \leq j \leq 2^n - 1$"*.

Let us prove the basis for $n = 1$, i.e., for $\mathcal{H}_1$. As $U_\infty(1) = 1$, if $E_0 = \emptyset$, then $E_1 = \{1\}$ and $j = 1$. If $E_0 = \{1\}, j = 0$. Thus $\mathcal{H}_1$ is true.

Assume that, for $n \in \mathbb{N}$, $\mathcal{H}_n$ is true, and consider $E_0 \subset \{1, \ldots, n+1\}$.

First, consider the case where $n+1 \in E_0$. Set $E'_0 = E_0 - \{n+1\}$ and $E'_{k+1} = E'_k \triangle \{U_\infty(k+1)\}$. For all $k \leq 2^n - 1$, $E_k = E'_k \cup \{n+1\}$. According to $\mathcal{H}_n$, there exists j such that $E'_j = \{1, \ldots, n\}$. Then, $E_j = E'_j \cup \{n+1\} = \{1, \ldots, n+1\}$.

Now consider $n+1 \notin E_0$. For all $k \leq 2^n - 1$, $U_\infty(k) \leq n$, and $n+1 \notin E_k$. Then, as $U_\infty(2^n) = n+1$, $E_{2^n} = E_{2^n - 1} \cup \{n+1\}$. Set $E'_0 = E_{2^n - 1}$ and $E'_{k+1} = E'_k \triangle \{U_\infty(k+1)\}$. For all $k \leq 2^n - 1$, $E_{2^n + k} = E'_k \cup \{n+1\}$. According to $\mathcal{H}_n$, there exists j such that $E'_j = \{1, \ldots, n\}$. Then, $E_{2^n + j} = \{1, \ldots, n+1\}$. By induction, the lemma is true. □

Theorem 2. *Protocol 2 solves the counting problem, under weak fairness, for up to P mobile agents, each with P states. Moreover, the protocol names up to $P - 1$ mobile agents with distinct names (for any $N < P$, the names are in $\{1, \ldots N\}$).*

Proof. Consider an execution $(C_0, C_1, C_2, \ldots)$ of the protocol. For every $i \geq 0$, let E_i denote the set of states s.t., for every $s \in E_i$, the number of mobile agents in state s in C_i is odd. If $E_i = \{1, \ldots, N\}$, then all the agents have distinct states. Let n_i and k_i denote (respectively) the values of the variables n and k of BS in a configuration C_i.

Lemma 3 implies that, for any $N < P$, if $E_0 \subset \{1, \ldots, N\}$ and $E_{k+1} = E_k \triangle \{U^*(k+1)\}$, there exists some $1 \leq j \leq 2^N - 1$ ($l_N = 2^N - 1$) such that $E_j = \{1, \ldots, N\}$ ($|U^*| = 2^P - 1$).

If $n_i < N$, agents cannot all have distinct non-zero-states in $\{1, \ldots, n_i\}$. Consider a configuration where $n_i < N$. There are two cases (i) and (ii) concerning possible transitions with BS. In case (i), there are agents in state 0, or/and there are different agents in the same state (homonyms), that will eventually interact and change their states to 0 (line 12). In both sub-cases, a mobile agent in state 0 eventually meets BS; and in the corresponding transition, in line 4, k increases. Once $k_j > l_{n_j}$ ($j > i$), n_j is incremented (lines 7 - 8). In case (ii), there exists a mobile agent x with $name_x > n_i$, what causes n to increase too. Thus, eventually, $n_j = N$. We show now that the protocol converges to $n = N$ and not a larger value.

- First, assume that the case (ii) does not occur. Consider the first configuration (C_i) with $n_i = N$, and suppose $N < P$. Starting from this configuration, BS assigns states to agents following U_N. $E_i \in 2^{\{1, \ldots, N\}}$, $k_i > l_{N-1}$ (lines 7 - 8), and only the following transitions between C_i and C_{i+1} are possible:
  1. a transition between homonyms (lines 11 - 12), which results in $E_{i+1} = E_i$;
  2. a transition between BS and an agent in state 0 (lines 3, 4 and 9), which results in $E_{i+1} = E_i \triangle \{U^*(k_i)\}$.

The number of non-zero homonyms in a given configuration is finite, and transitions of type 1 decrease this number, so that an infinite sequence of transitions of this type is impossible. Thus, while $E_i \neq \{1, \ldots, N\}$ (meaning that there are homonyms or agents in state 0), transitions of type 2 happen. These transitions also increment k_i. Let i_1, i_2, ... denote the indexes of transitions of type 2: $E_{i_{j+1}} = E_{i_j} \triangle \{U^*(k_j)\}$. Lemma 3 implies that there is some j such that $E_{i_j} = \{1, \ldots, N\}$. At this point, all agents are in distinct states, and the protocol has converged with $n = N$, because n increases only if the naming with n states has failed, i.e., when $k > l_n$, (lines 7 - 8) and this impossible in the considered case.

– If case (ii) happens, i.e., BS interacts with an agent x with $name_x > n_i$. Agent x has not been assigned before, since otherwise, it would have been given a state $\leq n_i$. The naming with $n_i - 1$ agents would have failed already, while this agent had no interaction. Thus, the execution up to step i is undistinguishable from an execution with at least n_i agents, but with the agent currently meeting BS, there are at least $n_i + 1$ agents. Thus, $N \geq n_i + 1$. In any case, $n_i \leq N$.

□

Corollary 2. *Protocol 2 is silent.*

Proof. By Theorem 2, for any $N < P$, the protocol finally names all mobile agents with distinct names in $\{1, \ldots N\}$, and thus the condition at line 2 stops being satisfied. Hence, in this case, eventually, no agent changes its state. In the remaining case of $N = P$, the condition at line 2 stops being satisfied when n reaches and stays equal to N (what happens, by Theorem 2). After that, no agent can change its state. □

Corollary 3. *The convergence time of Protocol 2 is $\Theta(2^N)$ effective transitions and $O(2^N)$ rounds.*

Proof. Following the study of sequence U_∞ and remark 1, the number of terms in U^* used by Protocol 2 to name n agents is $l_n = 2^n - 1$. In consequence, the number of (effective) interactions before convergence, between BS and an agent in state 0 or $> n$, is at most 2^N. Other possible effective transitions are between homonyms. Each agent in such a transition changes its state to 0, and its next effective transition necessarily involves BS. Thus, there cannot be more than 2^N effective transitions between homonyms. Hence, the first part of the corollary follows.
Now, notice that, in any (asynchronous) round, at least one effective transition happens (otherwise the protocol has converged). Thus, there are at most 2^N rounds to accommodate $\Theta(2^N)$ effective transitions. This implies the second part of the corollary. □

We prove below that this complexity is necessary for the optimal memory space. Intuitively, starting from an arbitrary configuration with P mobile agents, and with only P available states, no protocol at BS can detect the lacking names (states) in the population, during a worst case execution. That is why, in this case, BS cannot advance in naming (required for counting) faster than by following a sequence of at least $O(2^P)$ names. This length is necessary, because there exist $O(2^P)$ different starting configurations and from *any* such configuration, a sequence of at least $O(2^P)$ names is required to BS (in the worst case) in order to obtain a configuration with distinctly named mobile agents, and count them.

4.3 Protocol 2 is Time-Optimal among all Space-Optimal ones

Definition 1. *Given an execution $E = (C_1, C_2, \ldots)$ we introduce the following notation that also indicates the specific interactions at each execution step: $\mathcal{E} = (C_1, (a_1, b_1), C_2, (a_2, b_2), \ldots)$.*

The trace *of an execution segment $\mathcal{E}$ for an agent a is the sequence of states of this agent, and of the state of the agents it meet in the execution, denoted by*

$Tr_a(\mathcal{E}) = (C_1(a), C_k(b_k), C_k(a), C_l(b_l), \ldots)$ *(with* $k, l, \ldots$ *the steps at which the interactions involve* a*). When no precision is given, we consider the trace for the base station.*

The trace describes what an agent *"sees"*, so that two execution segments having the same trace for a given agent lead to configurations in which this agent is in the same state.

The following notation will be useful to describe executions with an extra agent, undistinguishable from the original one, which we will use to prove impossibility results.

Notation 2 *For a configuration* C *of a system with* n *agents, we denote by* $C[a'(s)]$ *the configuration with an extra* $n+1$*th agent* a' *in state* s*,* $C[a \to a', a(s)]$ *the configuration with an extra* $n+1$*th agent in the state of* a *in* C*, and* a *in state* s*. For a (finite or infinite) execution segment* $\mathcal{E} = (C_1, (a_1, b_1), C_2, \ldots)$*, we denote by* $\mathcal{E}[a'(s)] = (C_1[a'(s)], (a_1, b_1), C_2[a'(s)], \ldots)$ *and* $\mathcal{E}[a \to a'] = (C_1[a \to a', a(s)], (a'_1, b'_1), C_2[a \to a', a(s)], \ldots)$*, with* $s = C_1(a)$ *the state of agent* a *in configuration* C_1*, and* $a'_i = a_i$ *if* $a_i \neq a$*, and* $a'_i = a'$ *otherwise,* $b'_i = b_i$ *if* $b_i \neq a$*, and* $b'_i = a'$ *otherwise. Note that* $\mathcal{E}[a \to a']$ *is an execution segment: interactions that do not involve* a *remain unaffected, and interactions concerning* a *are applied to* a' *instead of* a*, so that the transition from one configuration to the following remains legal.*

Considering two execution segments $\mathcal{E}_1 = (C_1, (a_1, b_1), \ldots, C_k)$ *and* $(C_k, (a_k, b_k), \ldots, C_l)$*, such that the last configuration in* $\mathcal{E}_1$ *is the first configuration of* $\mathcal{E}_2$*, we denote by* $\mathcal{E}_1.\mathcal{E}_2 = (C_1, \ldots, C_l)$*.*

$\mathcal{E}' = \mathcal{E}[a'(s)]$ describes the execution segment built from $\mathcal{E}$ by adding a new agent a' that is held back from any interaction in the whole segment, while other agents behave just like in $\mathcal{E}$.

$\mathcal{E}' = \mathcal{E}[a \to a']$ describes the execution segment built from $\mathcal{E}$ by adding a new agent a', in the same state as a at the beginning, and taking the place of a while a is held back from any interaction. Each time a interacts in $\mathcal{E}$, a' does the same interaction in $\mathcal{E}'$, while a is held back from any interaction and remains in its starting state.

Consider a symmetric protocol counting up to P agents under weak fairness.

Lemma 4. *If there are* $2 < n < P$ *agents present in the system, no two mobile agents can be infinitely often simultaneously in the same state.*

In particular, the protocol has at least $P - 1$ states.

Proof. Consider a weakly fair execution $(C_1, (a_1, b_1), C_2, \ldots)$ with a set of n agents $\mathcal{A}$, with two mobile agents a_1 and a_2 infinitely often simultaneously in the same state.

For any step t, by weak fairness, there is a step $t' > t$ such that all pair of agents interact in the sequence $(C_t, \ldots, C_{t'})$, and, by the above remark, a step $t'' > t'$ such that $C_{t''}(a_1) = C_{t''}(a_2) = s$.

Let $t_0 < t_1 < \ldots$ be an infinite sequence of steps such that $C_{t_i}(a_1) = C_{t_i}(a_2) = s$ and any pair of agents interact in $\mathcal{E}_i = (C_{t_i}, (a_{t_i}, b_{t_i}), C_{t_i+1}, \ldots, C_{t_{i+1}})$.

Now, we can build an execution of the protocol with $n + 1 \leq P$ agents by introducing a new agent a_3 and setting for any i: $\mathcal{E}'_{3i} = \mathcal{E}_{3i}[a_3(s)]$, $\mathcal{E}'_{3i+1} = \mathcal{E}_{3i+1}[a_2 \to a_3]$ and $\mathcal{E}'_{3i} = \mathcal{E}_{3i}[a_3(s)]$, $\mathcal{E}'_{3i+1} = \mathcal{E}_{3i+1}[a_1 \to a_3]$. Now, by construction, the final configuration of $\mathcal{E}'_j$ and the initial configuration of $\mathcal{E}'_{j+1}$ are the same since a_1, a_2 and a_3 are then all in state s. Thus, $\mathcal{E}' = \mathcal{E}'_0.\mathcal{E}'_1 \ldots$ is an execution of the protocol. This execution is weakly fair: any pair of agent interacts in any sequence $(C'_{t_i}, \ldots, C'_{t_{i+3}})$, thus infinitely often. Now, from the point of view of the base station, the sequence of interactions is the same in $\mathcal{E}$ and $\mathcal{E}'$: $Tr_{BS}(\mathcal{E}) = Tr_{BS}(\mathcal{E}')$. Thus, the result of the counting process is the same, and the base station counts only n agents in C', which is a contradiction. □

We denote by $\to^*$ the transitive closure of the $\to$ relation between pairs of states.

Lemma 5. *A state m such that $(m,m) \to^* (m,m)$ cannot be occupied infinitely often if there are less than P agents.*

Proof. Consider a weakly fair execution $(C_1, (a_1, b_1), C_2, \ldots)$ with $n < P$ agents, the state m being occupied infinitely often. As there is only a finite number of agents, an agent a is in state m infinitely often.

For any step t, by weak fairness, there is a step $t' > t$ such that all pair of agents interact in the sequence $(C_t, (a_t, b_t), \ldots, C_{t'})$, and, by the above remark, a step $t'' > t'$ such that $C_{t''}(a) = m$.

Let $t_1 < t_2 < \ldots$ be an infinite sequence of steps such that $C_{t_i}(a) = m$ and any pair of agents interact in $\mathcal{E}_i = (C_{t_i}, (a_{t_i}, b_{t_i}), C_{t_i+1}, \ldots, C_{t_{i+1}})$. For any i, let $\mathcal{E}'_{2i} = \mathcal{E}_{2i}[a'(m)]$ and $\mathcal{E}'_{2i+1} = \mathcal{E}_{2i+1}[a \to a']$. Let $\mathcal{F}_i$ be the execution segment obtained by making a and a' interact in configuration C_{t_i} (in which they are both in state m) until they are both again in state m (the final configuration of $\mathcal{F}_i$ is thus still C_i).

In any execution segment $\mathcal{F}_{2i}.\mathcal{E}_{2_i}.\mathcal{F}_{2i+1}.\mathcal{E}_{2i+1}$, any two agents interact: if they are distinct from a and a', in $\mathcal{E}_{2i}$ and $\mathcal{E}_{2i+1}$, if one of them is a, in $\mathcal{E}_{2i}$, if one of them is a', in $\mathcal{E}_{2i+1}$, and if they are a and a', in $\mathcal{F}_{2i}$ and $\mathcal{F}_{2i+1}$. Obviously $Tr_{BS}(\mathcal{F}_{2i}.\mathcal{E}_{2_i}.\mathcal{F}_{2i+1}.\mathcal{E}_{2i+1}) = Tr_{BS}(\mathcal{E}_{2i}.\mathcal{E}_{2i+1})$.

Thus, the execution $\mathcal{F}_0.\mathcal{E}'_0.\mathcal{E}'_1.\mathcal{F}_1 \ldots$ is a weakly fair execution of the protocol, undistinguishable for the base station from the original execution $\mathcal{E}$. The base station will count n agents, which is a contradiction. □

From the two Lemmas above, we can deduce that a symmetric population protocol counting up to P agents has at least P states, as already proven in [**?**]. Indeed, consider any state s. Consider $(s,s) = (s_1,s_1) \to (s_2,s_2) \to (s_3,s_3) \to \ldots$. Since the number of state is finite, there exist $i < j$ such that $s_i = s_j$. Let $m = s_i$: we have $(m,m) \to^* (m,m)$. Consider an execution with $P-1$ agents. By Lemma 4, all agents have distinct state, and by Lemma 5, they cannot be in state m. Thus, the protocol has at least $(P-1)+1 = P$ states.

Now, suppose that the protocol is symmetric and has P states.

Lemma 6. *There exists a state m such that for any state s, $(s,s) \to^* (m,m)$. Moreover, $(m,m) \to (m,m)$.*

Proof. Consider any state s and $(s,s) = (s_1,s_1) \to (s_2,s_2) \to (s_3,s_3) \to \ldots$. Since the number of state is finite, there exist $i < j$ such that $s_i = s_j$. Let $m_s = s_i$.

State m is unique: for any s and s', $m_s = m_{s'} = m$. Indeed, all such states cannot be infinitely often occupied by Lemma 5, and the protocol can count $P-1$ agents that will end in distinct states (Lemma 4).

Let m' be a state such that $(m,m) \to (m',m')$: since $(m,m) \to^* (m,m)$, we have $(m',m') \to^* (m,m) \to (m',m')$, so that $m' = m$ by unicity of m. □

Remark 4. A non-deterministic protocol cannot be faster than a deterministic one in a weakly fair schedule: indeed, weak fairness tells nothing about the rule to be used when several agents meet, and one can design a weakly fair execution by choosing always the same rule in case of two agents meeting in given states.

Thus, in the sequel, we will consider only deterministic protocols.

Denote by BS_0 the initial state of the base station, and by BS_k and x_k the states such that $(BS_{k-1}, m) \to (BS_k, x_k)$: BS_k is the state of the base station after it has met k agents in state m (and no agent in other states), and x_k the name it gives to this kth agent.

Lemma 7. *If BS_k can be reached in an execution with $n < P$ agents (in other words, if BS_k is not a configuration in which the base station has counted P agents), then there exists $l \geq k$ such that $x_l \neq m$.*

Proof. Suppose that there exists a k such that an execution segment $\mathcal{E}$ with $n < P$ agents ends with $(BS_{k-1}, m) \to (BS_k, x_k)$, and $x_l = m$ for all $l \geq k$.

Let C be the terminal configuration of $\mathcal{E}$, and x an agent in state m (there is at least one, the one that met the base station in $\mathcal{E}$ during the last interaction $(BS_k, m) \to (BS_{k+1}, m)$): $\mathcal{E}.(BS, x).C[BS(BS_k)].(BS, x).C[BS(BS_{k+1})] \dots (BS, x).C[BS(BS_l)]$ is always a valid execution segment, and BS_l is reachable in a configuration with $n < P$ agents for all l.

Now, let C_0 be a configuration with P agents such that there is one agent in state x if and only if x appears an odd number of times in $x_1, x_2, \dots, x_k$; other agents are in state m. Consider the execution segment consisting in repeating k times the following segment: make an agent in state m meet the base station, and if it creates homonyms, make immediately these homonyms meet until they reach state m. This execution is possible: since there are P agents and P states, either one is in state m, or there are homonyms that meet and lead to agents in state m. After this execution segment, the base station is in state BS_k, and all other agents are in state m. Thus, since $(m, m) \to (m, m)$, and $(BS_l, m) \to (BS_{l+1}, m)$, the remaining of the execution will consist in meeting between agents in state m, and between the base station and agents in state m.

Any prefix of this execution with P agents has thus the same trace as the one of an execution with $n < P$ agents: the base station remains in states BS_l reachable in a configuration with $n < P$ agents. Thus, the protocol cannot count P agents, which is a contradiction. □

Theorem 3. *In the worst case, under weak fairness, a counting protocol with P states needs at least $2^{P-1} - 1$ non-null interactions to count P agents.*

Proof. Consider an execution $\mathcal{E}$ with $P - 1$ agents. In particular, according to Lemma 7, in any execution, the base station will eventually rename an agent in state m if, due to the interaction schedule or to the absence of any agent in a state $\neq m$, it does not meet any agent in a state $\neq m$ for an arbitrary long time.

As in Lemma 7, we will restrict our attention to execution prefixes in which the base station only meets agents in state m, as long as there exist agents in state m: as shown by Lemma 7, the base station cannot *"know"* if there are agents in states other than m, and thus cannot wait to meet an agent in a state $\neq m$ to take actions. Thus, when meeting continuously agents in state m, BS will continue the execution of the protocol until no agent in state m remains, or P agents have been counted.

The execution segments described below are prefixes of weakly fair executions, and, as such, their convergence time is a lower bound on the (worst-case) convergence time of the protocol under weak fairness (even if some executions may not follow the same pattern).

Considering a configuration C, and let $E(C)$ be the set of mobile agent states $\neq m$ that are occupied by an odd number of agents in configuration C.

We consider a family of executions, indexed by their starting configurations (we restrict ourselves to starting configurations such that no two agents have the same name $\neq m$), built in the following way:

1. an agent in state m meets the base station;
2. if the state allocated to this agent is different from m and already occupied by an agent, they meet until they are both in state m.

This process is repeated until a naming is obtained (after Lemmas 4 and 5, convergence cannot be reached before a naming is obtained; after Lemma 7, sequence x is long enough for the counting to be achieved, and thus the naming realized). Transitions of the form 2 do not change the parity of the number of agents in any state. Thus, if $C \to C'$ with a transition of form 2, $E(C') = E(C)$. With a transition of form 1, if $x_k = m$, then $E(C') = E(C)$, else $E(C') = E(C) \triangle \{x_k\}$. Now,

convergence can be reached only in a configuration C in which all agents have distinct states $\neq m$, which implies $E(C) = \mathcal{S} \setminus \{m\}$.

In such an execution, considering a starting configuration such that $E(C_0) = I$ (one agent is in each state of I, remaining agents are in state m), the consecutive values of $E(C)$, ignoring consecutive configurations with the same value of E, will be defined by $E_0(I) = I$, and $E_{l+1}(I) = E_l(I) \triangle \{x_{k_l}\}$, with k_l the indices at which $x_{k_l} \neq m$.
Thus, $E_l = I \triangle \{x / x$ appears an odd number of times in $x_1, x_2, \ldots, x_l\}$, and the protocol has converged to a naming when $\{x / x$ appears an odd number of times in $x_1, x_2, \ldots, x_l\} = I$. Since there are 2^{P-1} possible values for I (all subsets of $\mathcal{S} \setminus \{m\}$, which is of cardinal $P-1$), the sequence $x_1, x_2, \ldots$ needs to be at least of length $2^{P-1}-1$ (excluding the starting configuration $I = \mathcal{S} \setminus \{m\}$). Thus, in the worst starting configuration, $2^{P-1} - 1$ interactions between the base station and a mobile agent will be required for the protocol to give distinct names to $P-1$ agents.

In particular, to count P agents, the protocol will need that the naming with $P-1$ states has failed, and will take at least $2^{P-1}-1$ interactions between the base station and a mobile agent. □

Corollary 4. *Protocol 2 is time-optimal among all space-optimal counting protocols within a constant factor.*

***"Unicity"* of Protocol 2**

The following lemmas show that, with the difference of at most one possible extra transition between mobile agents, and of the base station counting strategy (which cannot improve the complexity of the protocol; see above), the proposed protocol is the only one that can count agents with P states.

Lemma 8. *There is no transition of the form $(s_1, s_2) \to (s_3, s_4)$ with $s_1 \neq s_2$, $s_1, s_2 \neq m$ and $\{s_1, s_2\} \neq \{s_3, s_4\}$.*

Proof. Suppose that such a transition exists. Now, consider an execution with $P-1$ agents. By Lemma 4, consider a configuration starting from which all agents are infinitely often in distinct states, and states that are different from m; thus all states $\neq m$ are occupied. By weak fairness, an agent in state s_1 and an agent in state s_2 will eventually interact, and if, w.l.o.g., $s_3 \neq s_1, s_2$, state s_3 is already occupied, and there are two agents in the same state s_3, which is a contradiction. □

Lemma 9. *There is no transition of the form $(m, s) \to (s', s)$ with $s' \neq m$.*

Proof. Suppose such a transition exists, and consider an execution $\mathcal{E}$ with $P-1$ agents. Let $\mathcal{E}_1$ be an execution segment starting from the initial configuration and leading to a configuration starting from which all agents have distinct states (Lemma 4). For $i > 1$, let $\mathcal{E}_i$ be an execution segment in which all agents interact. The proof of the previous Lemma shows that agents, starting from the last configuration of $\mathcal{E}_1$, remain in the same state: let a be the agent in state s, and a' the agent in state s'. Denote by C this configuration.

$\mathcal{E}_1[a''(m)].\mathcal{E}_2[a''(m)].(a, a'').C[a''(s')].(a', a'').C[a'(m), a''(m)].(a, a'').C[a'(m), a''(s)].$
$\mathcal{E}_3[a' \to a'', a'(m)].(a, a').C[a'(s'), a''(m)].(a', a'').C[a'(m), a''(m)].(a, a').C[a'(s), a''(m)]. \ldots$ is a weakly fair execution of the protocol with P agents, undistinguishable from $\mathcal{E}$ from the point of view of the base station. □

Lemma 10. *There is at most one state s such that there are transitions of the form $(m, s) \to (s', s'')$ with $\{m, s\} \neq \{s', s''\}$.*

Proof. Suppose the protocol has such a rule. First, consider an execution with $P-1$ agents. Consider a configuration with exactly one agent in state m. If there are two agents (or more) in state s, they can interact, leading to a configuration with three agents in state m. If there is an agent in state s in this configuration, let these two agents interact. Then, the resulting configuration contains no agent in states s and m; since there are $P-1$ agents and P states, at least one state is occupied by two agents: if the next interaction is between these two agents, we obtain a configuration with two agents in state m. Now, by Lemmas 4 and 5, at some point of the execution, there must be a last configuration with an agent in state m. Thus, to ensure convergence, the protocol must guarantee that, eventually, there is one agent in state m, and no agent in state s. In particular, at most one state s can be present in such a rule. □

5 Conclusion and Perspectives

In this paper, we presented two population protocols for counting, under two classical fairness assumptions. Under global fairness, we gave a protocol with only two states per agent and, under weak fairness, a protocol with P states (P being an upper bound on the size of the system). In terms of exact space complexity, both protocols are optimal in space and considerably improve the best solutions known up to now, presenting a totally different angle of attack.

Using a memory of only one bit has certainly practical advantages in applications for large-scale networks connecting very simple artifacts. Moreover, the assumption of global fairness, necessary for the correctness of the corresponding protocol, can be realized approximatively in practice. As described in [5], this is because in practice, a variety of parameters and events (like power-supply, local clock frequency or movement of nodes) affect the scheduling of a system in a random way, making the assumption of global fairness realistic. The average time complexity of the 1-bit protocol, assuming a fairness with probabilistic interactions (otherwise, the nature of global fairness yields an infinite convergence time), is exponential, and we are currently investigating if this is tight, for such a minimum memory requirement.

The second protocol, under weak fairness, solves the challenge of counting up to P with exactly P states per agent. Nevertheless, due to the nature of the Gros sequence, its time complexity, in terms of non-null transitions or in terms of (asynchronous) rounds, is exponential. This is because, in the worst case, the number of non-null transitions (or rounds) till convergence depends on the number of times BS renames a mobile agent. This is $2^{P-1}-1$ times, due to the length of the used Gros sequence. We have shown that this complexity is necessary. It can be seen as the price for using the minimum possible number of states.

A complete study of the trade off between space and time complexities for counting algorithms in population protocols could be a valuable sequel to the present work. Considering existing counting protocols designed for weak fairness, we can identify the following tendency. With $\log P$ bits of memory per mobile agent, the space-optimal protocol that we present in this paper has an exponential complexity. An additional bit of memory allows to design protocols like in [17] with a logarithmic round complexity, while another additional bit allows to solve this problem in a constant number of rounds [8]. It will be interesting to study whether such drastic trade-offs are necessary.

For global fairness, much less studies about counting protocols and especially about their complexity analysis exist. This is certainly an additional interesting research direction.

Finally, another possible perspective concerns the space complexity of BS. One may imagine a system, where all agents including the distinguishable BS are resource-limited, motivating the study of the necessary space requirements for BS.

Remark 5. One may notice that the proposed protocols look more like centralized protocols than distributed ones. We note that it should be so due to the strong memory constraints and the

nature of the problem. First, as without BS the problem is impossible, any solution has to use some sort of centralization; otherwise BS would not be necessary. Second, reducing the memory to the minimum, strongly limits the useful information that mobile agents can exchange to progress towards the solution. One can identify this in both protocols.

Acknowledgments

The authors would like to thank Jean-Paul Allouche and Jean Berstel for identifying the Gros sequence, and the anonymous reviewers for their thoughtful and helpful remarks.

References

1. Y. Afek, B. Awerbuch, S. A. Plotkin, and M. E. Saks. Local management of a global resource in a communication network. In *Symposium on Foundations of Computer Science*, pages 347–357, 1987.
2. J.-P. Allouche and J. O. Shallit. *Automatic Sequences - Theory, Applications, Generalizations*. Cambridge Univ. Press, 2003. ISBN 978-0-521-82332-6.
3. D. Angluin, J. Aspnes, Z. Diamadi, M. J. Fischer, and R. Peralta. Computation in networks of passively mobile finite-state sensors. *Dist. Comp.*, 18(4):235–253, 2006.
4. D. Angluin, J. Aspnes, D. Eisenstat, and E. Ruppert. The computational power of population protocols. *Dist. Comp.*, 20(4):279–304, 2007.
5. D. Angluin, J. Aspnes, M. J. Fischer, and H. Jiang. Self-stabilizing population protocols. *ACM Trans. Auton. Adapt. Syst.*, 3(4), 2008.
6. C. Baquero, P. S. Almeida, R. Menezes, and P. Jesus. Extrema propagation: Fast distributed estimation of sums and network sizes. *IEEE Trans. Parallel Distrib. Syst.*, 23(4):668–675, 2012.
7. J. Beauquier, J. Burman, and S. Clavière. Comptage et nommage simples et efficaces dans les protocoles de populations symétriques. In *ALGOTEL 2014*, pages 1–4, June 2014.
8. J. Beauquier, J. Clement, S. Messika, L. Rosaz, and B. Rozoy. Self-stabilizing counting in mobile sensor networks with a base station. In *DISC*, pages 63–76, 2007.
9. O. Bournez, J. Chalopin, J. Cohen, and X. Koegler. Playing with population protocols. In *CSP*, pages 3–15, 2008.
10. S. Dolev, M. G. Gouda, and M. Schneider. Memory requirements for silent stabilization. *Acta Inf.*, 36(6):447–462, 1999.
11. S. Dolev, A. Israeli, and S. Moran. Self-stabilization of dynamic systems assuming only read/write atomicity. *DC*, 7(1):3–16, 1993.
12. Y. Emek and A. Korman. New bounds for the controller problem. In *DISC*, pages 22–34, 2009.
13. P. Fraigniaud, A. Pelc, D. Peleg, and S. Perennes. Assigning labels in an unknown anonymous network with a leader. *Dist. Comp.*, 14(3):163–183, 2001.
14. A. J. Ganesh, A.-M. Kermarrec, E. Le Merrer, and L. Massoulié. Peer counting and sampling in overlay networks based on random walks. *Dist. Comp.*, 20(4):267–278, 2007.
15. C. Gkantsidis, M. Mihail, and A. Saberi. Random walks in peer-to-peer networks: Algorithms and evaluation. *Perform. Eval.*, 63(3):241–263, 2006.
16. A. M. Hinz, S. Klavzar, U. Milutinovic, and C. Petr. *The Tower of Hanoi - Myths and Maths*. Birkhäuser Basel, 2013. ISBN 3034802366, 9783034802369.
17. T. Izumi, K. Kinpara, T. Izumi, and K. Wada. Space-efficient self-stabilizing counting population protocols on mobile sensor networks. *Theor. Comput. Sci.*, 552:99–108, 2014.
18. H. Jiang. *Distributed Systems of Simple Interacting Agents*. PhD thesis, Yale University, 2007.
19. D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *FOCS*, pages 482–491, 2003.
20. A. Korman and S. Kutten. Controller and estimator for dynamic networks. *Inf. Comput.*, 223:43–66, 2013.
21. D. Kostoulas, D. Psaltoulis, I. Gupta, K. P. Birman, and A. J. Demers. Active and passive techniques for group size estimation in large-scale and dynamic distributed systems. *Journal of Systems and Software*, 80(10): 1639–1658, 2007.
22. F. Kuhn, N. A. Lynch, and R. Oshman. Distributed computation in dynamic networks. In *STOC*, pages 513–522, 2010.
23. G. Di Luna, R. Baldoni, S. Bonomi, and I. Chatzigiannakis. Conscious and unconscious counting on anonymous dynamic networks. In *ICDCN*, pages 257–271, 2014.

24. G. Di Luna, R. Baldoni, S. Bonomi, and I. Chatzigiannakis. Counting in anonymous dynamic networks under worst-case adversary. In *ICDCS*, pages 338–347, 2014.
25. E. Le Merrer, A.-M. Kermarrec, and L. Massoulié. Peer to peer size estimation in large and dynamic networks: A comparative study. In *HPDC*, pages 7–17, 2006.
26. O. Michail, I. Chatzigiannakis, and P. G. Spirakis. Naming and counting in anonymous unknown dynamic networks. In *SSS*, pages 281–295, 2013.
27. D. Mosk-Aoyama and D. Shah. Computing separable functions via gossip. In *PODC*, pages 113–122, 2006.
28. B. F. Ribeiro and D. F. Towsley. Estimating and sampling graphs with multidimensional random walks. In *ACM SIGCOMM*, pages 390–403, 2010.
29. G. Tel. *Introduction to Distributed Algorithms (2nd ed.)*. Cambridge University Press, 2000.
30. D. Varagnolo, G. Pillonetto, and L. Schenato. Distributed statistical estimation of the number of nodes in sensor networks. In *IEEE Conference on Decision and Control, CDC*, pages 1498–1503, 2010.

Appendix

Preliminary (Impossibility) Results

In this section we present some simple impossibility results related to the counting problem in population protocols. The proofs use the classical partitioning argument.

The following similar result was already presented in [8] for weak fairness. We show that it holds for global fairness as well.

Proposition 1. *If no agent state can be initialized, it is impossible to realize counting in population protocol model, under weak or global fairness.*

Proof. Assume, by contradiction, that such a protocol B exists and take a population of $2n$ agents. Both fairness conditions allow the population to be disconnected (partitioned) for any unbounded, but finite period. Thus, partition $2n$ agents into two sub-populations, each of n agents, till B converges in every sub-population and counts to n. Then, make the two sub-populations interact, till B converges and counts to $2n$. Repeat the whole scenario infinitely many times (possible under both fairness conditions) to obtain the contradiction. □

Thus, to be able to solve the counting problem and still avoid the initialization, all previous works, as well as the current one, assume the initialization of only one particular (and thus distinguishable) agent called the base station (BS).

Proposition 2. *There is no population protocol realizing counting with only* one *state per non-BS agent.*

Proof. Assume, by contradiction, that such a protocol exists. Consider a population of $n+1$ mobile agents. Take an execution in which one agent x does not interact with the other n agents and BS, for a long enough period, such that BS eventually counts correctly the n mobile agents it interacts with. Then, when x starts interacting with the population, trivially, no interaction can change its state and BS cannot distinguish this agent from any other. Thus, it won't be possible to count all the $n+1$ agents. □

Proposition 3. *No* silent *(uniform) counting population protocol exists with only two states available in every non-BS agent.*

Proof. Assume, by contradiction, that such a silent protocol exists. Consider a population of $n+1$ mobile agents. Take an execution in which an agent x does not interact with others for a long period such that, by the end of this period, the counting for the other n agents is accomplished

and no agent state changes (whenever x still does not interact) due to the silence of the assumed solution. Assume also that, in this latter configuration, the state of x is similar to the state of two other mobile agents x_1 and x_2 ($N > 2$). When x reconnects, no interaction involving x can be distinguished from an interaction with x_1 or x_2. Hence, it won't be possible to count all the $n+1$ agents. □

This proof can be easily generalized to show that there is no silent counting protocol with less than $N-1$ agent states. Note that, under weak fairness, a similar claim is correct and tight for less than N agent states (i.e., no silent counting protocol exists with less than N agent states). This is simply because no counting solution exists with less than N states under weak fairness [8], and a silent solution exists with at least N states (Protocol 2).

Time Complexity Analysis of Protocol 1 - Details

Lemma 11. *In the first configuration C^1 after the convergence of Protocol 1, i.e., the first time when size_total $= N$ (and does not change after), all agents have the same mark $m \in \{0,1\}$. Moreover, there is a configuration C^0 s.t. $C^0 \xrightarrow{*} C^1$, and all agents in C^0 are in state $1-m$.*

Proof. Thus, the counting is achieved in C^1. This happens following an interaction of BS with an agent, let us say w.l.o.g., in state 0. By definition, *size_total* $=$ *size*$[0]$ $+$ *size*$[1] = N$, and since *size_total* increases in this interaction, we had and we still have *size*$[0] = 0$ (otherwise, *size_total* cannot increase). Then, after this transition, *size*$[1]$ becomes N.

We show now that at the last transition with *size*$[1] = 0$, before C^1 has been reached (at least the first step is such), all agents were in state 0 (this will prove the existence of C^0). Denote by r the number of 1-0 transitions (transitions changing a state of a mobile agent from 1 to 0). Then, the number of 0-1 transitions is $N+r$, since 1-0 transitions increment *size_total*, and 0-1 ones decrement it (*size*$[1]$ never goes to 0, by assumption). Thus, BS meets $N+r$ agents in state 0 that turn to 1, and r in state 1 that turn to 0. This creates N new agents in state 1 thus, at the step when *size*$[1] = 0$, all agents were in state 0. □

Thus, consider a population of N agents, and let u_k be the average number of transitions that happen before all agents are in state 0, starting from a configuration with k agents in state 1.

We have the following relations:

- $u_0 = 0$ by definition;
- for $1 \leq k \leq N-1$, $u_k = 1 + \frac{k}{n}u_{k-1} + \frac{N-k}{N}u_{k+1}$: at the current step, there is k chances in N that an agent with mark 1 meets the base station, leading to a configuration with $k-1$ agents with mark 1, and $N-k$ chances in N that an agent marked 0 interacts with BS resulting in a configuration with $k+1$ agents marked 1;
- $u_N = 1 + u_{N-1}$.

For $0 \leq k \leq N-1$, set $v_k = u_{k+1} - u_k$. We have:

- $v_{N-1} = 1$
- $\forall 1 \leq k \leq N-1, v_{k-1} = u_k - u_{k-1} = u_k - \frac{N}{k}\left(u_k - 1 - \frac{N-k}{N}u_{k+1}\right) = \frac{k-N}{k}u_k + \frac{N}{k} + \frac{N-k}{k}u_{k+1} = \frac{N-k}{k}v_k + \frac{N}{k}$

Thus, for $0 \leq k \leq N-1$

$$v_{N-k} = \prod_{i=N-k+1}^{N-1} \frac{N-i}{i} + \sum_{i=1}^{k-1}\prod_{j=i}^{k} \frac{j}{N-j}\frac{N}{i} = \frac{(N-k)!(k-1)!}{(N-1)!} + N\sum_{i=1}^{k-1}\frac{(k-1)!(N-k)!}{i!(N-i)!} =$$

$$= N\sum_{i=0}^{k-1} \frac{(k-1)!(N-k)!}{i!(N-i)!}$$

$$v_{N-k} = \frac{1}{\binom{N-1}{k-1}} + N\sum_{i=1}^{k-1} \frac{(k-1)!(N-k)!}{N!}\frac{N!}{i!(N-i)!} = \frac{1}{\binom{N-1}{k-1}} + \sum_{i=1}^{k-1} \frac{\binom{N}{i}}{\binom{N-1}{k-1}} = \frac{\sum_{i=0}^{k-1}\binom{N}{i}}{\binom{N-1}{k-1}}$$

From that, we get:

$$u_N = \sum_{k=0}^{N-1} v_k = \sum_{k=0}^{N-1} \frac{\sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}}$$

First, consider the case when N is even:

$$u_N = \sum_{k=0}^{N/2-1} \frac{\sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}} + \sum_{k=N/2}^{N-1} \frac{\sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}}$$

$$u_N = \sum_{k=0}^{N/2-1} \frac{\sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}} + \sum_{k=N/2}^{N-1} \frac{2^N - \sum_{i=k+1}^{N}\binom{N}{i}}{\binom{N-1}{k}}$$

since $\sum_{i=0}^{N}\binom{N}{i} = 2^N$

$$u_N = \sum_{k=0}^{N/2-1} \frac{\sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}} + \sum_{k=N/2}^{N-1} \frac{2^N - \sum_{i=0}^{N-k-1}\binom{N}{N-i}}{\binom{N-1}{N-k-1}}$$

by setting $i' = N - i$

$$u_N = \sum_{k=0}^{N/2-1} \frac{\sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}} + \sum_{k=N/2}^{N-1} \frac{2^N - \sum_{i=0}^{N-k-1}\binom{N}{i}}{\binom{N-1}{N-k-1}}$$

$$u_N = \sum_{k=0}^{N/2-1} \frac{\sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}} + \sum_{k=0}^{N/2-1} \frac{2^N - \sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}}$$

by setting $k' = N - k - 1$

$$u_N = 2^N \times \sum_{k=0}^{N/2-1} \frac{1}{\binom{N-1}{k}}$$

$$u_N = 2^{N-1} \times \sum_{k=0}^{N-1} \frac{1}{\binom{N-1}{k}}$$

The case when N is odd is similar:

$$u_N = \sum_{k=0}^{(N-3)/2} \frac{\sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}} + \frac{\sum_{i=0}^{(N-1)/2}\binom{N}{i}}{\binom{N-1}{(N-1)/2}} + \sum_{k=(N+1)/2}^{N-1} \frac{\sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}}$$

And, similarly

$$u_N = \sum_{k=0}^{(N-3)/2} \frac{\sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}} + \sum_{k=0}^{(N-3)/2} \frac{2^N - \sum_{i=0}^{k}\binom{N}{i}}{\binom{N-1}{k}} + \frac{\sum_{i=0}^{(N-1)/2}\binom{N}{i}}{\binom{N-1}{(N-1)/2}}$$

by setting $k' = N - k - 1$

$$u_N = 2^N \times \sum_{k=0}^{(N-3)/2} \frac{1}{\binom{N-1}{k}} + 2^{N-1} \times \frac{1}{\binom{N-1}{(N-1)/2}}$$

$$u_N = 2^{N-1} \times \sum_{k=0}^{N-1} \frac{1}{\binom{N-1}{k}}$$

Now, for $2 \leq k \leq N-3$, $\frac{1}{\binom{N-1}{k}} \leq \frac{1}{\binom{N-1}{2}} = \frac{2}{(N-1)(N-2)}$, so that

$$2 = \frac{1}{\binom{N-1}{0}} + \frac{1}{\binom{N-1}{N-1}} \leq \sum_{k=0}^{N-1} \frac{1}{\binom{N-1}{k}} \leq 2 + \frac{2}{N-1} + \frac{2(N-4)}{(N-1)(N-2)} = 2 + O\left(\frac{1}{N}\right)$$

Thus, $u_N \geq 2^N$, and $u_N \sim_{N\to\infty} 2^N$.

The average complexity of the protocol is $\Theta(2^N)$. The best starting configurations, for the complexity, is when all agents have the same mark. The average complexity is then $2^N + o(2^N)$. Starting from any other configuration, the protocol first has to reach a configuration where all agents have identical marks. This takes less than $2^N + o(2^N)$ transitions, since starting with all agents having the same mark, and switching it, makes the protocol traverse all configurations. Hence, in this case, the overall complexity is less than $2 \times (2^N + o(2^N))$.