Data Collection in Population Protocols with Non-uniformly Random Scheduler

Joffroy Beauquier, Janna Burman, Shay Kutten, Thomas Nowak, Chuan Xu

Data Collection in Population Protocols with Non-uniformly Random Scheduler

Joffroy Beauquier[1], Janna Burman[1], Shay Kutten[2], Thomas Nowak[1], and Chuan Xu[1]

[1] LRI, Université Paris Sud, CNRS, Université Paris Saclay
{joffroy.beauquier, janna.burman, thomas.nowak, chuan.xu}@lri.fr
[2] Technion - Israel Institute of Technology
kutten@ie.technion.ac.il

Abstract. Contrary to many previous studies on population protocols using the uniformly random scheduler, we consider a more general non-uniform case. Here, pair-wise interactions between *agents* (moving and communicating devices) are assumed to be drawn *non-uniformly* at random. While such a scheduler is known to be relevant for modeling many practical networks, it is also known to make the formal analysis more difficult.
This study concerns *data collection*, a fundamental problem in mobile sensor networks (one of the target networks of population protocols). In this problem, pieces of information given to the agents (e.g., sensed values) should be delivered eventually to a predefined sink node without loss or duplication. Following an idea of the known deterministic protocol TTF solving this problem, we propose an adapted version of it and perform a complete formal analysis of execution times in expectation and with high probability (w.h.p.).
We further investigate the non-uniform model and address the important issue of energy consumption. The goal is to improve TTF in terms of energy complexity, while still keeping good time complexities (in expectation and w.h.p.). Namely, we propose a new parametrized protocol for data collection, called *lazy* TTF, and present a study showing that a good choice of the protocol parameters can improve energy performances (compared to TTF), at a slight expense of time performance.

1 Introduction

Population protocols have been introduced in [7] as a model for passively mobile sensor networks (cf. the journal version [8]). In this model, tiny indistinguishable agents with bounded memory move unpredictably and interact in pairs. That is, when two agents are sufficiently close to each other, they can communicate (i.e., interact). During an interaction, they exchange and update their respective states according to a transition function (the protocol). Such successive interactions contribute to the realization of some global task.

The fact that agent moves are unpredictable is usually modeled by assuming the uniformly random scheduler ([8, 9, 6, 11]). That is, the interactions between any two agents are drawn uniformly at random. However, for some practical sensor networks, this assumption may be unrealistic. Consider, for instance, agents moving at different speeds. In this case, an agent interacts more frequently with a faster agent than with a slower one. In other networks, certain agents may be frequently prevented from communicating with some others, because they move in different limited areas, or disfunction from time to time, etc. In all these examples, the interactions are clearly not uniformly random. There are thus strong arguments for enhancing the basic model.

This paper initiates the study of non-uniform schedulers in the context of population protocols. Considering the scheduler as the generator of sequences of pairwise interactions, non-uniform means that the next interacting pair (i, j) is chosen with a non-uniform probability $P_{i,j}$, depending on i and j.

As a supplementary justification for studying a non-uniform scheduler, notice that many experimental and analytical studies of different (finite boundary) mobile sensor networks show and exploit (respectively) the assumption that the *inter-contact* time of two agents (the time period between two successive interactions of the same two mobile agents) is distributed exponentially (cf. [32, 15, 38, 22]). Similarly, under a non-uniformly random scheduler, it appears that the inter-contact time $T_{i,j}$, of any two agents i and j, follows a geometric distribution ($P[T_{i,j} = t] = (1 - P_{i,j})^{t-1} P_{i,j}$), which is the discrete analogue of the exponential case (observed in practical mobile networks).

The counterpart of considering a non-uniform scheduler is a more complex analysis. Though, it remains feasible in certain cases, as it is shown in this paper. To illustrate this point, consider a fundamental task for mobile sensors, *data collection* (or data gathering). In this task, each agent has initially an input value (for instance, a sensed value). Each value must be gathered exactly once (as a multi-set) by a special agent which we call the base station. In the context of population protocols (assuming non-random schedulers), several data collection protocols have been proposed and their complexity in time has been studied [12]. Notice that the analysis there was only for the worst case. We are not aware of any previous results concerning the average complexity of these protocols. The current paper presents protocols that basically use the simple ideas of the TTF (Transfer To the Faster) protocol of [12]. The new protocols are adapted to a non-uniform scheduler and improve energy consumption, as explained further.

First, consider the original version of TTF. It uses a deterministic parameter called *cover time*, which is an upper bound on the time, counted in the number of global interactions, for an agent to interact with all the others. The data transfer between the agents in TTF depends on the comparison of cover times of two interacting agents. Here we follow this idea. However, as the scheduler is probabilistic, we adapt the corresponding definition of the cover time to be the *expected* (instead of the maximum) number of interactions for an agent to interact with every other agent (see Sect. 2).

The complexity analysis starts with the proofs of two lower bounds on the expected convergence time of any protocol solving data collection (Sect. 3). Then, an analysis of execution times in expectation and with high probability (w.h.p.), for the new version of TTF, is given (Sect. 4). The complexity in expectation indicates how the protocol is good in average, while the complexity w.h.p. tells how it is good almost all the time. We obtain explicit bounds, thus justifying the relevance of the enhanced model in protocol analysis and its operability.

We further investigate the non-uniform model by addressing also energy complexity, which is known to be a crucial issue for sensor networks. The goal is to improve energy consumption of TTF, while keeping good time complexity. For that, we propose a new parametrized protocol, called *lazy* TTF (Sect. 5). As opposed to TTF, it does not execute necessarily the transition of TTF resulting from an interaction. Instead, during an interaction (i, j), TTF is executed with probability p_i (depending on agent i, playing the role of *initiator* in the interaction). Analysis and the corresponding numerical study show that a good choice of the parameters p_i results in lower energy consumption. To find such parameters, we formulate and solve a polynomial-time optimization program. The resulting optimized lazy TTF is compared to TTF in respect with time and energy complexity (Sect. 6). For this analysis, we adopt the energy scheme proposed for population protocols in [37].

Due to the lack of space, most of the proofs and the survey on additional related work have been moved to the appendix.

2 Model and Definitions

Population protocols. The system is represented by an *interaction graph* $G = (\mathbf{A}, \mathbf{E})$, a table $\mathbf{T}$ of *transition rules* and a *scheduler* $S(P)$. All are defined below.

A set $\mathbf{A}$ consists of n anonymous agents and is also called a population. An agent $i \in \mathbf{A}$ represents a finite state sensing and communicating mobile device, which can be seen as a finite state machine. The size of the population n is unknown to the agents. Among the agents, there is a distinguishable one called the *base station* (BST), which can be as powerful as needed, in contrast with the resource-limited agents. The non-BST agents are also called *mobile*. Each agent has a state that is taken from a finite set of states which is the same for all mobile agents, but possibly different for the base station.

A directed edge $(i, j) \in \mathbf{E}$ intuitively represents a possible interaction between two agents. That is, if such an edge exists, then the scheduler (see below) is allowed to schedule an event, called interaction, between i, called then the *initiator*, and j, called the responder for that event. In this work, we consider only complete interaction graphs. What happens in the interaction event is now described.

When two agents i, in state p, and j, in state q, interact (meet), they execute a *transition* $(p, q) \to (p', q')$. As a result, i changes its state from p to p' and j from q to q'. The table $\mathbf{T}$ of all the *transition rules* defines the population protocol. A protocol (respectively, its transition rules) are called *deterministic*, if for every pair of states (p, q), there is exactly one (p', q') such that $(p, q) \to (p', q')$. Otherwise, they are *non-deterministic*. Note that, as interactions are supposed to be asymmetric (with one agent acting as the initiator and the other as the responder), the transition rules for (p, q) and (q, p) may be different.

A *configuration* of the system is defined by the vector of agents' states. If, in a given configuration C, a configuration C' can be obtained by executing one transition of the protocol (between two interacting agents), it is denoted by $C \rightarrow C'$. An *execution* of a protocol is a sequence of configurations $C_0, C_1, C_2, \ldots$ such that C_0 is the *initial configuration* and for each $i \geq 0$, $C_i \rightarrow C_{i+1}$. We consider the number of interactions in an execution as the time reference, i.e., each interaction adds one time unit to the global time. This is similar to the *step complexity*, a common measure in population protocols (cf. [8, 2]) and in distributed computing in general [34].

The sequence of the corresponding interactions in an execution is provided by an external entity called scheduler.

Non-uniformly random scheduler. Such a scheduler, denoted by $S(P)$, is defined by a matrix of probabilities $P \in \mathrm{R}^{n \times n}$. During an execution, $S(P)$ chooses the next pair of agents (i, j) to interact (taking i as initiator and j as responder) with the probability $P_{i,j}$. Notice that, in the case of the matrix with entries $P_{i,j} = 1/n(n-1)$ for $i \neq j$, and $P_{i,i} = 0$, the scheduler chooses each pair of agents uniformly at random for each next interaction (i.e., the scheduler is uniformly random).
The matrix P satisfies $\sum_{i=1}^{n} \sum_{j=1}^{n} P_{i,j} = 1$ and $\forall i \in \{1, ..., n\}, P_{i,i} = 0$, since interactions are pairwise. Moreover, for any edge (i, j) in the interaction graph G, $P_{i,j} > 0$. As the considered here G is complete, *every* pair of agents is chosen infinitely often with probability 1.
For a given P, one can compute the *expected* (finite) time for a given agent i to meet all the others. We call it *cover time of agent* i and denote it by cv_i. By resolving the coupon collector's problem with a non-uniform distribution [21], we obtain the cover time of each agent: $\mathrm{cv}_i = \int_0^\infty (1 - \prod_{j \neq i} (1 - e^{-(P_{i,j} + P_{j,i})t}))dt$. Similarly to [12], for two agents i and j, if $\mathrm{cv}_i < \mathrm{cv}_j$, we say that i is *faster* than j, and j is *slower* than i. If $\mathrm{cv}_i = \mathrm{cv}_j$, i and j are said to be in the same *category* of cover times. We denote by m the number of different categories of cover times. We emphasize that agents are not assumed to know their cvs (to conform with the finite state population protocol model). Instead, we do assume that two interacting agents can compare their respective cvs. For instance, this can be implemented by comparing categories instead of *cv*s, in applications where the overall number of categories is likely to be uniformly bounded.

Data Collection. Each agent, except the base station, owns initially a constant input value. Eventually, every input value has to be delivered to the base station, and exactly once (as a multi-set). When this happens, we say that a *terminal configuration* or simply *termination* has been reached. A protocol is said *to solve* data collection if termination is reached in every execution of the protocol.
In the sequel, when describing or analyzing a protocol, the term "transfer an

input value (or token) from agent i to j" means copy it to j's memory and erase it from the memory of i. In particular, this prevents loss or duplication of input values. Moreover, in this preliminary study, we make the assumption that every agent has enough memory to store n values. This assumption is common in the literature [23, 6].

Time Complexity Measures. The *convergence time* of a data collection protocol $\mathcal{P}$ can be evaluated in two ways: first, in terms of expected time until termination, denoted by $T_{\mathrm{E}}(\mathcal{P})$, and second, in terms of time until termination w.h.p.[3], denoted by $T_{whp}(\mathcal{P})$.

Remark 1. The notion of parallel time, which is common when considering the uniformly random scheduler (cf. [9, 10]), is not used in this paper. When using this measure of time, it is assumed that each agent participates in an expected number $\Theta(1)$ of interactions per time unit. With the uniformly random scheduler, this time measure is asymptotically equal to the number of interactions divided by n. However, with non-uniformly random scheduler, this is no more true.

3 Lower Bounds on the Expected Convergence Time

We now give two nontrivial lower bounds on the expected convergence time of data collection protocols. The first one (Th. 1) only depends on the number of agents. The second one (Th. 2) depends on the specific values of the probability matrix P used by the scheduler. The bounds are incomparable in general. To obtain the bounds, we observe that, for performing data collection, each agent has to interact at least once (otherwise, its value simply won't be delivered), and we compute the expected time ensuring that. The proof of Th. 1 uses an analogy with a generalization of the classical coupon collector's problem, which we introduce next.

Let k be a positive integer. Given a probability distribution $(p_1, \dots, p_k)$ on $[k] = \{1, \dots, k\}$, the corresponding k-coupon collector's problem is defined by its *coupon sequence* $(X_1, X_2, \dots)$ of independent and identically distributed (i.i.d.) random variables with $\mathrm{P}(X_t = i) = p_i$ for all $i \in [k]$ and all $t \geq 0$. The k-coupon collector's problem's *expected time* is the expectation of the earliest time T such that $\{X_1, \dots, X_T\} = [k]$, i.e., all coupons were collected at least once.

More generally, given a set $\mathcal{A}$ of subsets of $[k]$ such that $\bigcup_{A \in \mathcal{A}} A = [k]$, and a probability distribution (p_A) on $\mathcal{A}$, the corresponding $\mathcal{A}$-group k-coupon collector's problem is defined by its *coupon group sequence* $(X_1, X_2, \dots)$ of i.i.d. random variables with $\mathrm{P}(X_t = A) = p_A$ for all $A \in \mathcal{A}$ and all $t \geq 0$. Its *expected time* is the expectation of the earliest time T such that $\bigcup_{t=1}^{T} X_t = [k]$, i.e., all coupons were collected in at least one coupon group.

Given an integer $1 \leq g \leq k$, the g-group k-coupon collector's problem is the $\mathcal{A}$-group k-coupon collector's problem where $\mathcal{A} = \{A \subseteq [k] \mid |A| = g\}$. This generalization of the classical coupon collector's problem has been studied, among others, by Stadje [33], Adler and Ross [1], and Ferrante and Saltalamacchia [20].

[3] An event Ξ is said to occur w.h.p., if $\mathrm{P}(\Xi) \geq 1 - \frac{1}{n^c}$, where $c \geq 1$.

The following lemma characterizes the probability distributions that lead to a minimal expected time for the group coupon collector's problem. To the best of our knowledge, this is a new result which generalizes the characterization in the classical coupon collector's problem [21, 27], for which it is known that the uniform distribution leads to the minimal expected time.

Lemma 1. *The expected time of any $\mathcal{A}$-group k-coupon collector's problem is greater than or equal to the $\mathcal{B}$-group k-coupon collectors problem with uniform probabilities where $\mathcal{B} \subseteq \mathcal{A}$ is of minimal cardinality such that $\bigcup \mathcal{B} = [k]$.*

In particular, the expected time of any g-group k-coupon collector's problem is $\Omega(k \log k)$ for every constant $g \geq 1$.

Theorem 1. *The expected convergence time of any protocol solving data collection with non-uniformly random scheduler is $\Omega(n \log n)$.*

Theorem 2. *The expected convergence time of any protocol solving data collection with random scheduler $S(P)$, is $\Omega(\max\limits_i \frac{1}{\sum_{j=1}^{n}(P_{i,j}+P_{j,i})})$.*

The next corollary considers a very simple protocol solving the data collection problem. In this protocol, agents transfer their values only when they interact with the base station. We consider it as a reference, to compare with other proposed protocols. The corollary follows from Th. 2.

Corollary 1. *With random scheduler $S(P)$, the expected convergence time of the protocol solving data collection and where each agent transfers its value only to the base station is $\Omega(\max\limits_i 1/(P_{i,\mathrm{BST}} + P_{\mathrm{BST},i}))$.*

4 Protocol "Transfer To the Faster" (TTF)

Corollary 1 formalizes the straightforward observation that, if the only transfers performed by the agents are towards the base station, the convergence time depends on the slowest agent i. It can be very large, e.g. if $P_{i,\mathrm{BST}} + P_{\mathrm{BST},i} \ll 1/n^2$. Therefore, to obtain better time performances, we propose to study another data collection protocol based on the idea of the TTF protocol of [12]. In the sequel, the studied protocol is called TTF too, since its strategy is the same and there is no risk of ambiguity. The only difference is on the *definition* of the cover time parameter (Sect. 2) used by this strategy (as explained in the introduction).

The strategy of TTF is easy. When agent i meets a faster agent j, i transfers to j all the values it has in its memory (recall that transfer means to copy to the memory of the other and erase from its own). The intuition behind is that the faster agent j is more likely to meet the base station before i. Of course, whenever any agent i meets the base station, it transfers all the values it (still) has in its memory at that time to the base station. As a matter of fact, no transition depends on the actual value held by the agents. It depends only on the comparison between cover times, which are constants. Thus, the input values can be seen as tokens and the states of every agent can be represented by the number of tokens it currently holds. Recall, that in this study, it is assumed that each agent has enough memory for storing the tokens (i.e., an $O(n)$ memory), and each pair of agents interacts infinitely often (i.e., the interaction graph is complete).

The sequel concerns analytical results on the time performance of TTF. Firstly, we associate to each configuration a vector of non-negative integers representing the number of tokens held by each agent. Then, it is shown that the evolution of such vectors during executions can be expressed by a *stochastic* linear system. Next, $T_{\text{whp}}(\text{TTF})$ is expressed in terms of distances between the configuration vectors (Th. 3) and, by applying stochastic matrix theory ([35, 25, 30]) an upper bound on $T_{\text{whp}}(\text{TTF})$ is obtained (Th. 4). Finally, using this result, we obtain also an upper bound on the convergence time in expectation, $T_{\text{E}}(\text{TTF})$ (Th. 5).

Formally, we represent a configuration by a non-negative integer vector $x \in \mathrm{N}^n$ that satisfies $\sum_{i=1}^{n} x_i = n-1$. By abusing the terminology, we sometimes call such a vector a configuration. We denote the configuration vectors' space by $\mathbb{V}$. By convention, the first element of x is the number of tokens held by the base station. Since, at the beginning of an execution, every mobile agent owns exactly one token and no token is held by the base station, the initial configuration is $x_{\text{init}} = \mathbf{1} - \mathbf{e}_1$, where $\mathbf{e}_i = \big(0, \ldots, 0, 1, 0, \ldots, 0\big)^T$ is the $n \times 1$ unit vector with the i^{th} component equal to 1. The terminal configuration is $x_{\text{end}} = (n-1)\mathbf{e}_1$.

Let $x(t) \in \mathbb{V}$ be the discrete random integer vector that represents the configuration just after t^{th} interaction in executions of TTF. We can see that $\mathrm{P}(x(0) = x_{\text{init}}) = 1$, and since the base station never transfers tokens to others, $\mathrm{P}(x(t+1) = x_{\text{end}}) \geq \mathrm{P}(x(t) = x_{\text{end}})$. Moreover, since at any moment there is a positive probability for delivering any of the tokens to the base station, $\lim_{t\to\infty} \mathrm{P}(x(t) = x_{\text{end}}) = 1$. Furthermore, the time complexities of TTF can be formalized using $x(t)$ by $T_{\text{E}}(\text{TTF}) = \sum_{t=1}^{\infty} t \cdot (\mathrm{P}(x(t) = x_{\text{end}} \wedge x(t-1) \neq x_{\text{end}}))$ and $T_{\text{whp}}(\text{TTF}) = \inf\left\{t \mid \mathrm{P}(x(t) = x_{\text{end}}) \geq 1 - \frac{1}{n}\right\}$.
To evaluate these time complexities, we study the evolution of $x(t)$ during executions of TTF. Given time t, consider a transition rule applicable from a configuration represented by a vector v^t and resulting in a configuration with vector v^{t+1}. Suppose that at time t, the interaction (i, j) is chosen by the scheduler. If neither i nor j are the base station and i is faster than j ($\text{cv}_i < \text{cv}_j$), agent j transfers all its tokens to i. Thus, $v_i^{t+1} = v_i^t + v_j^t$ and $v_j^{t+1} = 0$. The relation between v^t and v^{t+1}, in this case, can be expressed by the linear equation $v^{t+1} = W(t+1)v^t$, where $W(t+1) = I + \mathbf{e}_i\mathbf{e}_j^T - \mathbf{e}_j\mathbf{e}_j^T \in \{0,1\}^{n\times n}$. If $\text{cv}_i = \text{cv}_j$, no token is transferred and $v^{t+1} = v^t$. We still have $v^{t+1} = W(t+1)v^t$, but with $W(t+1) = I$. On the other hand, if j is the base station, $W(t+1) = I + \mathbf{e}_i\mathbf{e}_j^T - \mathbf{e}_j\mathbf{e}_j^T$, as agent i transfers all of its tokens to the base station.
As the pair of agents is chosen independently with respect to P, $W(t+1)$ can be seen as a *random* matrix such that with probability $P_{i,j} + P_{j,i}$:

$$W(t+1) = \begin{cases} I + \mathbf{e}_i\mathbf{e}_j^T - \mathbf{e}_j\mathbf{e}_j^T & \text{if } \text{cv}_i < \text{cv}_j \text{ or } i = 1 \text{ or } j = 1 \\ I & \text{if } \text{cv}_i = \text{cv}_j \end{cases} \tag{1}$$

By comparing the resulting probability distributions, we readily verify that the relation between $x(t)$ and $x(t+1)$, i.e., $x(t+1) = W(t+1)x(t)$, is a stochastic linear system with the matrices specified in (1).

Distance. Consider a function $d_\gamma(x) : \mathbb{V} \to \mathrm{R}$. It associates any x in $\mathbb{V}$ to a real number representing a "weighted" Euclidian norm distance between the configuration vector x and the vector representing a terminal configuration. That is, $d_\gamma(x) = ||(x - x_{\text{end}}) \circ \gamma||_2$, where $\gamma \in \mathrm{R}^n$ is a real vector, $\circ$ the entry-wise product, and $||\cdot||_2$ the Euclidean norm. The vector γ can be viewed as a weight vector. We choose γ in such a way that, if there is a transfer of tokens in interaction $t+1$, configuration v^t, then $d_\gamma(v^{t+1})$ is smaller than $d_\gamma(v^t)$. Intuitively this means that, when a transfer is performed, the resulting configuration is closer to termination.

Lemma 2. *Let i and j be two agents with $\mathrm{cv}_i < \mathrm{cv}_j$. Consider an interaction between i and j in a configuration represented by v^t and resulting in v^{t+1}. If $\gamma_j/\gamma_i \geq \sqrt{2n-3}$, then $d_\gamma(v^{t+1}) \leq d_\gamma(v^t)$.*

Theorem 3. *The convergence time with high probability of* TTF, $T_{\text{whp}}(\text{TTF})$*, is equal to* $\inf\left\{t \mid \mathrm{P}\left(\frac{d_\gamma(x(t))}{d_\gamma(x_{\text{init}})} < (2n)^{\frac{-(m-1)}{2}}\right) \geq 1 - 1/n\right\}$ *if* $\gamma_{\text{BST}} = 0$ *and* $\gamma_j/\gamma_i \geq \sqrt{2n}$ *whenever* $\mathrm{cv}_i < \mathrm{cv}_j$*. Recall that $m \leq n$ denotes the number of cover time categories (Sect. 2).*

We are now ready to state and prove our main upper bound on the convergence time of TTF, $T_{\text{whp}}(\text{TTF})$ (Th. 4). To prove it, we apply stochastic matrix theory to the stochastic linear system defined above for $x(t)$.

Without loss of generality, we assume that $\mathrm{cv}_2 \leq \mathrm{cv}_3 \leq \cdots \leq \mathrm{cv}_n$. We choose $\gamma \in \mathrm{R}^n$ by setting $\gamma_1 = 0$, $\gamma_2 = 1$, and $\gamma_{i+1} = \gamma_i$, if $\mathrm{cv}_{i+1} = \mathrm{cv}_i$, and $\gamma_{i+1} = \gamma_i\sqrt{2n}$, if $\mathrm{cv}_{i+1} > \mathrm{cv}_i$. In particular, $\gamma_n = (2n)^{(m-1)/2}$.

Theorem 4. *With a non-uniformly random scheduler $S(P)$, the convergence time of* TTF *is at most* $\frac{m \log 2n}{\log \lambda_2(\tilde{W})^{-1}}$ *with high probability, where γ is defined above.* $\mathit{\Gamma}_{i,j} = \gamma_i/\gamma_j$, $\tilde{W} = \sum_{i<j \wedge \mathrm{cv}_i < \mathrm{cv}_j} (P_{i,j} + P_{j,i}) W_{ij}^{\mathit{\Gamma}^2} + \sum_{i<j \wedge \mathrm{cv}_i = \mathrm{cv}_j} (P_{i,j} + P_{j,i}) I$, $W_{ij}^{\mathit{\Gamma}^2} = I + \mathit{\Gamma}_{i,j}(e_i e_j^T + e_j e_i^T) + (\mathit{\Gamma}_{i,j}^2 - 1) e_j e_j^T$*, and $\lambda_2(A)$ denotes the modulus of the second largest eigenvalue of matrix A.*

Now, we study the performance of TTF with respect to the convergence time in expectation, i.e. $T_{\text{E}}(\text{TTF})$.

Theorem 5. *The expected convergence time of the TTF protocol is* $O\left(\frac{m \log n}{\log \lambda_2(\tilde{W})^{-1}}\right)$ *where $\tilde{W}$ is the matrix defined in Theorem 4.*

5 Lazy TTF

The strategy of TTF may result in a long execution when an input value is transferred many times before being finally delivered to the base station. These transfers are certainly energy consuming. Then a natural issue is to transform TTF in order to save energy, while keeping the time complexity as low as possible. The idea is to prevent certain data transfers, for example, when it is more likely to meet soon a faster agent and thus possibly make fewer transfers in overall. We propose a simple protocol based on TTF, called *lazy* TTF. In contrast with TTF, lazy TTF does not necessarily execute the transition resulting from an interaction. It chooses randomly to execute it or not. Formally, during an interaction (i, j), with agent i acting as initiator, TTF is executed with probability p_i, where $p \in \mathrm{R}^n$ is a vector of probabilities.

Notice that the choice of executing TTF depends uniquely on the initiator i. In practical terms, an initiator represents an agent that, by sensing the environment, has detected another agent j. At this moment i takes the random decision (with probability p_i) whether a TTF transition should be executed and the interaction itself should take place, or not. In the latter case, not only the energy for the eventual data transfer is saved, but also the energy for establishing the interaction.

Observe that when p is the vector of all ones, lazy TTF behaves as TTF and its energy consumption is the same as for TTF. However, when p is the vector of all zeros, lazy TTF does not solve the problem of data collection as no value is ever transferred to the base station, but no energy is consumed for transferring of data or establishing interactions. Depending on p, time complexities of lazy TTF can be worse than of TTF, given the same scheduler. At the same time, longer executions of lazy TTF may be more energy efficient. Thus, there is a trade-off between time and energy performance depending on the values of p. We now investigate the choice of p for obtaining good time/energy trade-off. Firstly, we give upper bounds on the time complexities of lazy TTF. Then, we introduce an optimization problem that takes p as a variable. Finally, numerical results in Sect. 6 demonstrate energy efficiency of lazy TTF, given the optimal p.

5.1 Convergence Time of Lazy TTF

To obtain an upper bound on the convergence time of lazy TTF, we show a particular equivalence of lazy TTF under scheduler $S(P)$ with TTF under scheduler $S(P \circ (p \cdot \mathbf{1}^T))$, where $\mathbf{1}$ is the vector of all ones and $\circ$ presents the entry-wise product. This equivalence is on the level of distribution of configurations of the two protocols. Precisely, as we show below, the random vector $x(t)$ for these two protocols is exactly the same, allowing to use Th. 4 to obtain a time complexity upper bound for lazy TTF.

Let us express $x(t)$ in case of lazy TTF in a similar way as we did before for TTF in Sect. 4. First, $\mathrm{P}(x(0) = x_{\text{init}}) = 1$ is the same as for TTF. Then, $x(t+1) = W(t+1)x(t)$ and $W(t+1)$ can be seen as a *random* matrix such that, with probability $P_{i,j} \times p_i + P_{j,i} \times p_j$, $W(t+1)$ is as in Eq. 1. Notice that $x(t)$ in case of TTF under $S(P \circ (p \cdot \mathbf{1}^T))$ is expressed exactly in the same way (Sect. 4). Thus, by applying Th. 4 for TTF under $S(P \circ (p \cdot \mathbf{1}^T))$, we obtain the upper bound on $T_{whp}(\text{lazy TTF}(p))$.

Theorem 6. *With a non-uniformly random scheduler $S(P)$, the convergence time with high probability of lazy* TTF *is at most $\frac{m \log 2n}{\log \lambda_2(\tilde{W})^{-1}}$,*

$$\text{where } \tilde{W} = \sum_{cv_i < cv_j} (P_{i,j}p_i + P_{j,i}p_j) W_{ij}^{\Gamma^2} + \sum_{cv_i < cv_j} (P_{i,j}(1-p_i) + P_{j,i}(1-p_j) I + \sum_{cv_i = cv_j} (P_{i,j} + P_{j_i}) I, \text{ and } W_{ij}^{\Gamma^2} = I + \Gamma_{i,j}(\mathbf{e}_i \mathbf{e}_j^T + \mathbf{e}_j \mathbf{e}_i^T) + (\Gamma_{i,j}^2 - 1)\mathbf{e}_j \mathbf{e}_j^T. \quad (2)$$

Then, the upper bound on $T_{\mathrm{E}}(\text{lazy TTF}(p))$ can be obtained in the same way as in Th. 5.

To summarize, note that, as executions of lazy TTF are equivalent to those of TTF under $S(P \circ (p \cdot \mathbf{1}^T))$ in the sense explained above, one can imagine that lazy TTF transforms the matrix of interaction probabilities "on the fly" (during executions). It can be also seen as if it transforms the interaction graph itself. Indeed, certain vectors p may make some pairs of agents to interact with extremely small probability (or not interact at all), thus effectively remove these pairs from the graph. This is illustrated in the appendix. Next, we are looking for vectors p, optimizing an upper bound on the time performance of lazy TTF(p) to ensure a good time energy trade-off. Equivalently, we are looking for schedulers (matrices P) for which the original TTF is efficient in this sense.
Thus, the goal is to find a vector p minimizing the upper bound on $T_{whp}(\text{lazy TTF}(p))$ (Th. 6). To that end, an optimization program OP_1, taking p as a variable, is proposed as follows:
$OP_1: \min_{p \in \mathrm{R}^n} \lambda_2(\tilde{W})$ s.t Eq. 2, $0 \le p_i \le 1$.
By Th. 6, minimizing the upper bound of $T_{whp}(\text{lazy TTF}(p))$ is equivalent to minimizing the second largest eigenvalue of $\tilde{W}$. Then, we reformulate OP_1 as a semi-definite program [36, 24] OP_2 which is convex and can be solved in polynomial time.
$OP_2: \min_{p \in \mathrm{R}^n, s} s$ s.t $sI - \tilde{W} \succeq 0$, Eq. 2, $0 \le p_i \le 1$.

Let $\hat{p}$ be the optimal solution of OP_2. We can see that if $\hat{p}$ is all ones vector, lazy TTF$(\hat{p})$ performs as TTF. Otherwise, lazy TTF$(\hat{p})$ outperforms TTF in terms of the upper bounds on time. This optimized upper bound ensures that lazy TTF$(\hat{p})$ converges in a reasonable time. In the next section, by the numerical results obtained for different small examples, we demonstrate the efficiency of lazy TTF$(\hat{p})$, in terms of energy consumption.

6 Numerical Results

6.1 The Relation between T_{whp}(TTF) and its Upper Bound

The goal of this section is to justify the relevance of the method used here to obtain the optimal probability vector p for lazy TTF. To justify this, we show

by simulations that the time upper bound value for TTF is well correlated with the exact value of its time complexity (calculated by Markov chains, for small systems). This implies the same correlation for lazy TTF, because the bounds in Th. 4 and Th. 6 are obviously well correlated too (one is obtained from the other; see Sect. 5). That is why the optimal probability vector p for the upper bound of lazy TTF is close to the optimal vector for the real (tight) convergence time.
From Th. 4, we have an upper bound on time w.h.p. for TTF, denoted here by $T_{\text{upp}}(\text{TTF})$. In this section, we show the relation between $T_{\text{upp}}(\text{TTF})$ and $T_{\text{whp}}(\text{TTF})$. In our experiment, two systems of size 4 and 5 are considered and 100 schedulers are generated randomly for each system. Since the system is of small size, for each scheduler s, the exact value of $T^s_{\text{whp}}(\text{TTF})$ can be obtained by constructing the corresponding Markov Chain. The upper bound, $T^s_{\text{upp}}(\text{TTF})$, can be calculated by Th. 4. Then, for every generated s, we plot $T^s_{\text{whp}}(\text{TTF})$ and $T^s_{\text{upp}}(\text{TTF})$ on the figure with x-axis for $T_{\text{whp}}(\text{TTF})$ and y-axis for $T_{\text{upp}}(\text{TTF})$.

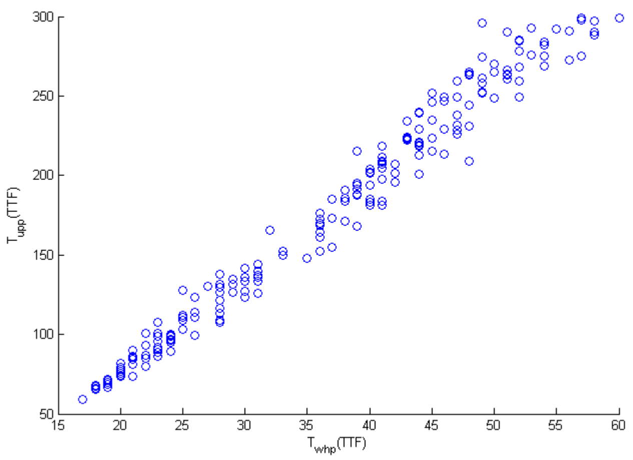

Fig. 1. Relation between $T_{\text{whp}}(\text{TTF})$ and $T_{\text{upp}}(\text{TTF})$.

From Fig. 1, we can see that $T_{\text{upp}}(\text{TTF})$ has a nearly linear relation with $T_{\text{whp}}(\text{TTF})$. It means that $T_{\text{upp}}(\text{TTF})$ in Th. 4 captures well the relation of the scheduler's behavior to the time performance of TTF in most of the cases. Moreover, it demonstrates that, for lazy TTF, minimizing $T_{whp}(\text{lazy TTF}(p))$ in Sect. 5, is reasonable for improving the energy performance.

6.2 Gaps on Time and Energy between TTF and Lazy TTF($\hat{p}$)

For energy consumption analysis, we consider the energy model of [37] proposed for population protocols. In this model, an agent senses its vicinity by proximity sensor, consuming a negligible amount of energy [29]. Once the interaction is established, each participant consumes a fixed amount of energy $\mathcal{E}_{wkp}$ (mainly for switching on its radio, which is known to be very energy consuming; cf.[28]). Now, recall that, with lazy TTF, the choice of executing TTF depends on the probability p_i of the initiator i. If TTF should not be executed, the initiator does not proceed to establish the interaction neither (i.e., $\mathcal{E}_{wkp}$ is not spent), as explained in Sect. 5.

We study the expectation of the total energy consumption of a protocol $\mathcal{P}$, denoted $\mathcal{E}(\mathcal{P})$. According to the energy scheme explained above, $\mathcal{E}(\mathcal{P})$ is evaluated by the expected total energy spent for establishing all the interactions till convergence. It is proportional to the time expectation $T_{\mathrm{E}}(\mathcal{P})$. In particular, $\mathcal{E}(\mathrm{TTF}) = 2T_{\mathrm{E}}(\mathrm{TTF}) \cdot \mathcal{E}_{wkp}$ and $\mathcal{E}(\mathrm{lazyTTF}(p)) = 2T_{\mathrm{E}}(\mathrm{lazy\,TTF}(p)) \times \sum_i \sum_j (P_{i,j}p_i + P_{j,i}p_j) \times \mathcal{E}_{wkp}$.

For the systems of small size with a scheduler s, the exact values of $T^s_{\mathrm{E}}(\mathrm{TTF})$ and $T^s_{\mathrm{E}}(\mathrm{lazy\,TTF}(\hat{p}^s))$ can be calculated by constructing the corresponding Markov Chain. In the experiments, systems of size 4,5,6,7 and 8 are considered and for each size n, 10000 different schedulers are generated randomly. Denote by $\mathcal{S}(n)$ the set of these schedulers. For each scheduler $s \in \mathcal{S}(n)$, $T^s_{\mathrm{E}}(\mathrm{TTF})$, $\hat{p}^s$, $T^s_{\mathrm{E}}(\mathrm{lazy\,TTF}(\hat{p}^s))$, $\mathcal{E}^s(\mathrm{TTF})$ and $\mathcal{E}^s(\mathrm{lazy\,TTF}(\hat{p}^s))$ are evaluated. Then, the gaps on time and on energy between lazy $\mathrm{TTF}(\hat{p}^s)$ and TTF^s are denoted by $Gap(T_{\mathrm{E}}, n)$ and $Gap(\mathcal{E}, n)$, respectively, and are computed as follows.
$Gap(T_{\mathrm{E}}, n) = \left(\sum_{s\in\mathcal{S}(n)} \frac{T^s_{\mathrm{E}}(\mathrm{lazy\,TTF}(\hat{p}^s)) - T^s_{\mathrm{E}}(\mathrm{TTF})}{T^s_{\mathrm{E}}(\mathrm{TTF})}\right)/10000$ and
$Gap(\mathcal{E}, n) = \left(\sum_{s\in\mathcal{S}(n)} \frac{\mathcal{E}^s(\mathrm{lazy\,TTF}(\hat{p}^s)) - \mathcal{E}^s(\mathrm{TTF})}{\mathcal{E}^s(\mathrm{TTF})}\right)/10000$.
Results appear in Table. 1.

Size n	$Gap(T_{\mathrm{E}}, n)$	$Gap(\mathcal{E}, n)$
4	11.60%	-15.32%
5	17.10%	-23.60%
6	22.04%	-30.79%
7	26.31%	-36.99%
8	27.41%	-39.07%

Table 1. Gaps on time and energy.

In column 3, it can be seen that lazy TTF consumes less energy than TTF for all systems. Lazy TTF saves at least 15% of energy. The counterpart is (a slight) increase in the execution time, as shown in column 2.

References

1. Ilan Adler and Sheldon M. Ross. The coupon subset collection problem. *Journal of Applied Probability*, 38:737–746, 2001.
2. Dan Alistarh, James Aspnes, David Eisenstat, Rati Gelashvili, and Ronald L. Rivest. Time-space trade-offs in population protocols. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2560–2579, 2017.
3. Dan Alistarh, James Aspnes, David Eisenstat, Rati Gelashvili, and Ronald L Rivest. Time-space trade-offs in population protocols. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2560–2579. SIAM, 2017.

4. Dan Alistarh, James Aspnes, and Rati Gelashvili. Space-optimal majority in population protocols, Technical Report 2017.
5. Dan Alistarh and Rati Gelashvili. Polylogarithmic-time leader election in population protocols. In *International Colloquium on Automata, Languages, and Programming*, pages 479–491. Springer, 2015.
6. Dan Alistarh, Rati Gelashvili, and Milan Vojnović. Fast and exact majority in population protocols. In *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing*, pages 47–56. ACM, 2015.
7. Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. In *Proceedings of the Twenty-third Annual ACM Symposium on Principles of Distributed Computing*, PODC '04, pages 290–299, New York, NY, USA, 2004. ACM.
8. Dana Angluin, James Aspnes, Zoë Diamadi, Michael J Fischer, and René Peralta. Computation in networks of passively mobile finite-state sensors. *Distributed computing*, 18(4):235–253, 2006.
9. Dana Angluin, James Aspnes, and David Eisenstat. Fast computation by population protocols with a leader. In *International Symposium on Distributed Computing*, pages 61–75. Springer, 2006.
10. Dana Angluin, James Aspnes, and David Eisenstat. A simple population protocol for fast robust approximate majority. *Distributed Computing*, 21(2):87–102, 2008.
11. James Aspnes, Joffroy Beauquier, Janna Burman, and Devan Sohier. Time and space optimal counting in population protocols. In Panagiota Fatourou, Ernesto Jiménez, and Fernando Pedone, editors, *OPODIS 2016*, volume 70 of *LIPIcs*, pages 13:1–13:17. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.
12. Joffroy Beauquier, Janna Burman, Julien Clement, and Shay Kutten. On utilizing speed in networks of mobile agents. In *Proceedings of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pages 305–314. ACM, 2010.
13. Luca Becchetti, Andrea E. F. Clementi, Emanuele Natale, Francesco Pasquale, Prasad Raghavendra, and Luca Trevisan. Friend or foe? population protocols can perform community detection. *CoRR*, abs/1703.05045, 2017.
14. Stephen Boyd, Arpita Ghosh, Balaji Prabhakar, and Devavrat Shah. Randomized gossip algorithms. *IEEE/ACM Transactions on Networking (TON)*, 14(SI):2508–2530, 2006.
15. Han Cai and Do Young Eun. Crossing over the bounded domain: from exponential to power-law inter-meeting time in manet. In *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 159–170. ACM, 2007.
16. Ioannis Chatzigiannakis, Shlomi Dolev, Sándor Fekete, Othon Michail, and Paul Spirakis. On the fairness of probabilistic schedulers for population protocols. In *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2010.
17. Rachel Cummings, David Doty, and David Soloveichik. Probability 1 computation with chemical reaction networks. *Natural Computing*, 15(2):245–261, 2016.
18. Alexandros G Dimakis, Soummya Kar, José MF Moura, Michael G Rabbat, and Anna Scaglione. Gossip algorithms for distributed signal processing. *Proceedings of the IEEE*, 98(11):1847–1864, 2010.
19. David Doty and David Soloveichik. Stable leader election in population protocols requires linear time. In *International Symposium on Distributed Computing*, pages 602–616. Springer, 2015.

20. Marco Ferrante and Monica Saltalamacchia. The coupon collector's problem. *Materials matemàtics*, 2014(2):35 pp., 2014.
21. Philippe Flajolet, Daniele Gardy, and Loÿs Thimonier. Birthday paradox, coupon collectors, caching algorithms and self-organizing search. *Discrete Applied Mathematics*, 39(3):207–229, 1992.
22. Wei Gao and Guohong Cao. User-centric data dissemination in disruption tolerant networks. In *INFOCOM, 2011 Proceedings IEEE*, pages 3119–3127. IEEE, 2011.
23. Rachid Guerraoui and Eric Ruppert. Names trump malice: Tiny mobile agents can tolerate byzantine failures. In *ICALP (2)*, pages 484–495, 2009.
24. Christoph Helmberg, Franz Rendl, Robert J Vanderbei, and Henry Wolkowicz. An interior-point method for semidefinite programming. *SIAM Journal on Optimization*, 6(2):342–361, 1996.
25. Ali Jadbabaie, Jie Lin, and A Stephen Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on automatic control*, 48(6):988–1001, 2003.
26. Yves Mocquard, Emmanuelle Anceaume, and Bruno Sericola. Optimal proportion computation with population protocols. In *Network Computing and Applications (NCA), 2016 IEEE 15th International Symposium on*, pages 216–223. IEEE, 2016.
27. Toshio Nakata. Coupon collector's problem with unlike probabilities. Preprint, 2008.
28. V. Rajendran, K. Obraczka, and J. J. Garcia-Luna-Aceves. Energy-efficient, collision-free medium access control for wireless sensor networks. *Wireless Networks*, 12(1):63–78, 2006.
29. Mohammad Abdur Razzaque and Simon Dobson. Energy-efficient sensing in wireless sensor networks using compressed sensing. *Sensors*, 14(2):2822–2859, 2014.
30. Wei Ren, Randal W Beard, et al. Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Transactions on automatic control*, 50(5):655–661, 2005.
31. Devavrat Shah et al. Gossip algorithms. *Foundations and Trends® in Networking*, 3(1):1–125, 2009.
32. Gaurav Sharma and Ravi R Mazumdar. Scaling laws for capacity and delay in wireless ad hoc networks with random mobility. In *Communications, 2004 IEEE International Conference on*, volume 7, pages 3869–3873. IEEE, 2004.
33. Wolfgang Stadje. The collector's problem with group drawings. *Advances in Applied Probability*, 22(4):866–882, 1990.
34. Gerard Tel. *Introduction to Distributed Algorithms (2nd ed.)*. Cambridge University Press, 2000.
35. John N Tsitsiklis, Dimitri P Bertsekas, and Michael Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. In *1984 American Control Conference*, pages 484–489, 1984.
36. Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM review*, 38(1):49–95, 1996.
37. Chuan Xu, Janna Burman, and Joffroy Beauquier. Power-aware population protocols, 2017. To appear in ICDCS 2017.
38. Hongzi Zhu, Luoyi Fu, Guangtao Xue, Yanmin Zhu, Minglu Li, and Lionel M Ni. Recognizing exponential inter-contact time in vanets. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–5. IEEE, 2010.

A Additional Related Work

The uniformly random scheduler has been introduced and studied in the context of population protocols in the seminal paper [7]. Later, leader election protocols (cf. [5, 19]) and exact majority protocols (cf. [6]) have been proposed in this uniform model. Their performances are evaluated by the parallel expected convergence time and by the number of states available at each agent (space complexity). In several papers (cf. [3]), trade-offs between time and space complexities of protocols solving these problems are studied. Any leader election or majority protocol converges in $\Omega(n/\mathit{polylog}\ n)$ expected time using $O(\log\log n)$ states. A recent work [4] shows that, by employing "phase clocks", both problems can be solved in $O(\log^2 n)$ expected time, using $O(\log n)$ states. Other complex problems, such as counting [11], community detection [13] and proportion computation [26], have been also studied under similar uniform scheduler model.
Besides the uniformly random scheduling independent of the agents states, there are works assuming scheduling dependent on the states of agents, like the *transition function* scheduler in [16] or the scheduling of reactions in CRN (Chemical Reaction Network model) according to the model of stochastic chemical kinetics (cf. [17]).
We should mention also the randomized gossip algorithm in [14] designed for the problem of averaging in an arbitrarily connected network. Each node runs an independent Poisson clock (asynchronous time model), and at each clock tick, the node randomly selects a neighbor, with the probability given by the algorithm. Then, it averages its value with the chosen neighbor. Observe that this algorithm can be seen as a population protocol under non-uniformly random scheduler, in which two meeting agents average their values. For more details on gossip algorithms, refer to the surveys [31] and [18].
For the discussion and related work on the energy consumption scheme for population protocols that we adopt here refer to [37]. Additional related references on data collection population protocols can be found in [37] too.

B Proof of Lemma 1

We say that a set $\mathcal{B} \subseteq \mathcal{A}$ of coupon groups is *covering* if $\bigcup \mathcal{B} = [k]$.

For every $\mathcal{B} \subseteq \mathcal{A}$, denote by $T_{\mathcal{B}}$ the time until all coupon groups in $\mathcal{B}$ appear at least once in the coupon group sequence. Denoting by T the time until all coupons appear at least once in the coupon group sequence, we have

$$T = \min\left\{T_{\mathcal{B}} \mid \mathcal{B} \text{ is covering}\right\} \ . \tag{3}$$

For every covering $\mathcal{B}$, let $F_{\mathcal{B}}$ be the event that $\mathcal{B}$ is the first covering coupon group set to completely appear in the coupon group sequence. The law of total expectation gives

$$\mathrm{E}(T) = \sum_{\mathcal{B} \text{ covering}} \mathrm{E}\left(T \mid F_{\mathcal{B}}\right) \cdot \mathrm{P}(F_{\mathcal{B}}) \geq \min_{\mathcal{B} \text{ covering}} \mathrm{E}\left(T \mid F_{\mathcal{B}}\right) \ . \tag{4}$$

By definition of $F_{\mathcal{B}}$, we have $\mathrm{E}\left(T \mid F_{\mathcal{B}}\right) = \mathrm{E}\left(T_{\mathcal{B}} \mid F_{\mathcal{B}}\right)$. The latter is greater than or equal to the expected time of a $|\mathcal{B}|$-collector's problem, which can be seen by shifting probabilities of non-$\mathcal{B}$ coupon groups into $\mathcal{B}$. By [27, Theorem 1], this time is then at most that of the $\mathcal{B}$-coupon collector's problem with uniform probabilities, i.e., $|\mathcal{B}|H(|\mathcal{B}|)$ where $H(m) = \sum_{\ell=1}^{m} 1/\ell$ denotes the m^{th} harmonic number. This proves the first part of the lemma.

To show the second part, we note that $\lceil k/g \rceil$ coupon groups of size g are needed to cover the set $[k]$, i.e., $|\mathcal{B}| \geq \lceil k/g \rceil$, which means

$$\mathrm{E}(T) \geq \left\lceil \frac{k}{g} \right\rceil \cdot H\left(\left\lceil \frac{k}{g} \right\rceil\right) \sim \frac{k}{g} \cdot \log \frac{k}{g} = \Omega(k \log k) \tag{5}$$

as $k \to \infty$ if g is a constant.

C Proof of Theorem 1

For data collection, each agent has to transfer its value at least once, and the base station has to receive values at least once. Therefore, in any execution, each agent has to interact at least once. The expected time of every agent interacting at least once is that of a 2-group n-coupon collector's problem, i.e., is $\Omega(n \log n)$ by Lemma 1.

D Proof of Theorem 2

For any agent i, it is required at least $\sum_{j=1}^{n} 1/(P_{i,j} + P_{j,i})$ time in expectation to establish one interaction. Thus, to complete one data collection, for which it is required that each agent interacts at least once, it takes at least $\max_i \frac{1}{\sum_{j=1}^{n}(P_{i,j}+P_{j,i})}$ expected time.

E Proof of Lemma 2

During an interaction (i, j) where i is faster than j, j transfers its tokens to i. Suppose that agent i holds a tokens in v^t. To ensure $d_\gamma(v^{t+1}) \leq d_\gamma(v^t)$, it suffices to have $((a+1)^2 - a^2)\gamma_i^2 \leq \gamma_j^2$, which is equivalent to $\gamma_j/\gamma_i \geq \sqrt{2a+1}$. If agent j has tokens, agent i cannot have more than $n-2$ tokens in v^t, thus $a \leq n-2$. Therefore, if $\gamma_j/\gamma_i \geq \sqrt{2n-3}$, the lemma is satisfied.

F Proof of Theorem 3

Given a configuration vector v^t at time t, if $v^t \neq x_{\text{end}}$, we have $d_\gamma(v^t) \geq \gamma_{\min}$ where $\gamma_{\min} = \min\{\gamma_i \mid i \neq \mathrm{BST}\}$. Further, using the relation $\gamma_j/\gamma_i \geq \sqrt{2n}$, we have $d_\gamma(x_{\text{init}}) = ||\gamma||_2 \leq \sqrt{\sum_{i=0}^{m-2}(2n)^i + (n-m+1)(2n)^{m-1}} \leq \sqrt{(2n)^{(m-1)/2}}$. Therefore, if $v^t \neq x_{\text{end}}$, we have $\frac{d_\gamma(v^t)}{d_\gamma(x_{\text{init}})} \geq \frac{1}{||\gamma||_2} \geq (2n)^{-(m-1)/2}$.

Thus, if v^t satisfies $\frac{d_\gamma(v^t)}{d_\gamma(x_{init})} < (2n)^{-(m-1)/2}$, it is necessarily the terminal configuration. Since $T_{\text{whp}}(\text{TTF}) = \inf\{t \mid \text{P}(x(t) = x_{\text{end}}) \geq 1 - 1/n\}$, we obtain the result of Th. 3.

G Proof of Theorem 4

Firstly, we study the evolution of the vectors $y(t) = \big(x(t) - x_{\text{end}}\big) \circ \gamma$, which appear in the formulation of $T_{\text{whp}}(\text{TTF})$ (Th. 3). As $x_{\text{end}} = (n-1)e_1$ and $\gamma_1 = 0$, $y(t)$ reduces to $x(t) \circ \gamma$. Since $x(t+1) = W(t)x(t)$, we obtain

$$y(t+1) = x(t+1) \circ \gamma = \big(W(t)x(t)\big) \circ \gamma = W^{\Gamma}(t)y(t) \tag{6}$$

where $W^{\Gamma}(t) = W(t) \circ \Gamma$ is the entry-wise product and matrix $\Gamma \in \text{R}^{n \times n}$ has entries $\Gamma_{i,j} = \gamma_i/\gamma_j$ when $j \neq 1$ and $\Gamma_{i,1} = 0$ for all $i \in \{1, \ldots, n\}$. Hence we get:

$$\text{E}\left(y(t+1)^T y(t+1) \mid y(t)\right) = y(t)^T\, \text{E}\left(W^{\Gamma}(t)^T \cdot W^{\Gamma}(t)\right) y(t) \tag{7}$$

From (1), we know that with probability $P_{i,j} + P_{j,i}$, matrix $W^{\Gamma}(t)^T \cdot W^{\Gamma}(t)$ is equal to

$$\begin{cases} I + \Gamma_{i,j}(e_i e_j^T + e_j e_i^T) + (\Gamma_{i,j}^2 - 1)e_j e_j^T & \text{if } \text{cv}_i < \text{cv}_j \text{ or } i = 1 \text{ or } j = 1 \\ I & \text{if } \text{cv}_i = \text{cv}_j \end{cases} \tag{8}$$

Then, setting $W_{ij}^{\Gamma^2} = I + \Gamma_{i,j}(e_i e_j^T + e_j e_i^T) + (\Gamma_{i,j}^2 - 1)e_j e_j^T$, we have:

$$\begin{aligned} \tilde{W} &= \text{E}\left(W^{\Gamma}(t)^T \cdot W^{\Gamma}(t)\right) \\ &= \sum_{i<j \wedge \text{cv}_i < \text{cv}_j} (P_{i,j} + P_{j,i}) W_{ij}^{\Gamma^2} + \sum_{i<j \wedge \text{cv}_i = \text{cv}_j} (P_{i,j} + P_{j,i}) I \end{aligned} \tag{9}$$

In particular:

$$\tilde{W}_{1,1} = 1 \qquad \text{and} \qquad \tilde{W}_{i,1} = \tilde{W}_{1,i} = 0 \quad \text{for all } i \in \{1, \ldots, n\} \tag{10}$$

Since $W^{\Gamma}(t)^T \cdot W^{\Gamma}(t)$ is symmetric and positive semi-definite, so is its expectation $\tilde{W}$. Now, we turn to study the properties of the eigenvalues in $\tilde{W}$. By (10), matrix $\tilde{W}$ is of the form

$$\tilde{W} = \begin{pmatrix} 1 & 0 \cdots & 0 \\ 0 & & \\ \vdots & W' & \\ 0 & & \end{pmatrix} \tag{11}$$

for some $W' \in \text{R}^{(n-1)\times(n-1)}$. Denoting the kth largest eigenvalue of matrix A by $\lambda_k(A)$, we have $\lambda_1(W') \leq \|W'\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^{n} W'_{i,j}$ where $\|\cdot\|_\infty$ denotes the operator norm with respect to the supremum norm on R^{n-1}, i.e., the largest 1-norm of rows of the matrix. According to (9), the i^{th} row sum of $\tilde{W}$ is

$$\sum_{j=1}^{n} \tilde{W}_{i,j} = \sum_{w \neq i} \sum_{v \neq w} P_{w,v} + \sum_{j:\mathrm{cv}_j = \mathrm{cv}_i} (P_{i,j} + P_{j,i}) \tag{12}$$

$$+ \sum_{j:\mathrm{cv}_j < \mathrm{cv}_i} \mathit{\Gamma}_{j,i}^2 (P_{i,j} + P_{i,j}) + \sum_{j:\mathrm{cv}_j > \mathrm{cv}_i} \mathit{\Gamma}_{i,j} (P_{i,j} + P_{i,j}) \tag{13}$$

for all $i \in \{2, \ldots, n\}$. As $\mathit{\Gamma}_{i,j} = \frac{\gamma_i}{\gamma_j} < 1$ whenever $\mathrm{cv}_i < \mathrm{cv}_j$, we obtain $\sum_j W'_{i,j} < 1$ for all i. By the block decomposition (11), we thus have $\lambda_1(\tilde{W}) = 1$ and

$$\lambda_2(\tilde{W}) = \lambda_1(W') \leq \|W'\|_\infty = \max_i \sum_j W'_{i,j} < 1 \ . \tag{14}$$

Now, using the Rayleigh quotient, we have:

$$\mathrm{E}\left(y(t+1)^T y(t+1) \mid y(t)\right) \leq \lambda_2(\tilde{W}) \cdot \|y(t)\|_2^2 \tag{15}$$

Repeatedly using (15), we obtain the bound

$$\mathrm{E}\|y(t)\|_2^2 = \mathrm{E}\left(y(t)^T y(t)\right) \leq \lambda_2(\tilde{W})^t \|y(0)\|_2^2 \ . \tag{16}$$

Applying Markov's inequality, we obtain

$$\mathrm{P}\left(\frac{d_\gamma(x(t))}{d_\gamma(x_{\mathrm{init}})} \geq (2n)^{\frac{-(m-1)}{2}}\right) = \mathrm{P}\left(\frac{\|y(t)\|_2^2}{\|y(0)\|_2^2} \geq (2n)^{-(m-1)}\right)$$

$$\leq (2n)^{m-1} \frac{\|y(t)\|_2^2}{\|y(0)\|_2^2} \leq (2n)^{m-1} \lambda_2(\tilde{W})^t \tag{17}$$

Thus, if $(2n)^{m-1}\lambda_2(\tilde{W})^t \leq 1/n$, i.e., $t \geq \frac{m \log 2n}{\log \lambda_2(\tilde{W})^{-1}}$, then $\mathrm{P}\left(\frac{d_\gamma(x(t))}{d_\gamma(x_{\mathrm{init}})} \geq (2n)^{\frac{-(m-1)}{2}}\right) \leq 1/n$. So, we obtain $T_{\mathrm{whp}}(\mathrm{TTF}) \leq \frac{m \log 2n}{\log \lambda_2(\tilde{W})^{-1}}$.

H Proof of Theorem 5

Let T be the convergence time and $T_\theta = \inf\left\{t \mid \mathrm{P}(x(t) = x_{\mathrm{end}}) > 1 - \theta\right\} = \inf\left\{t \mid \mathrm{P}(T \geq t) \leq \theta\right\}$. Analogously to (17) in Th. 4, we know that

$$T_\theta \leq \frac{\log 2n^{(m-1)}\theta^{-1}}{\log \lambda(\tilde{W})^{-1}} = \frac{(m-1)\log 2n}{\log \lambda(\tilde{W})^{-1}} + \frac{\log(1/\theta)}{\log \lambda(\tilde{W})^{-1}} = A + B\log(1/\theta) \ . \tag{18}$$

Since T is a non-negative random number, we have

$$\begin{aligned}
T_{\mathrm{E}}(\mathrm{TTF}) &= \sum_{t=1}^{\infty} \mathrm{P}(T \geq t) \\
&= \mathrm{P}(T \geq 1) + \mathrm{P}(T \geq 2) + ... + \mathrm{P}(T \geq T_\theta) + \sum_{t=1+T_\theta}^{\infty} \mathrm{P}(T \geq t) \\
&\leq T_\theta + \sum_{t=1+T_\theta}^{T_{\theta/2}} \mathrm{P}(T \geq t) + \sum_{t=1+T_{\theta/2}}^{T_{\theta/4}} \mathrm{P}(T \geq t) + ... \\
&\leq T_\theta + T_{\theta/2} \cdot \theta + T_{\theta/4} \cdot \theta/2 + ... = T_\theta + \sum_{i=1}^{\infty} T_{\theta/2^i} \cdot \frac{\theta}{2^{i-1}} \\
&\leq T_\theta + \sum_{i=1}^{\infty} (A + B \log \frac{2^i}{\theta}) \cdot \frac{\theta}{2^{i-1}} \\
&= T_\theta + A\theta \cdot \sum_{i=1}^{\infty} \frac{1}{2^{i-1}} + B\theta \cdot (\log 2 \cdot \sum_{i=1}^{\infty} \frac{i}{2^{i-1}} + \log \theta^{-1} \sum_{i=1}^{\infty} \frac{1}{2^{i-1}}) \\
&= T_\theta + (A\theta + B\theta \log \theta^{-1}) \cdot \sum_{i=1}^{\infty} \frac{1}{2^{i-1}} + 2B\theta \log 2 \cdot \sum_{i=1}^{\infty} \frac{i}{2^i} \\
&\leq (1 + 2\theta) \cdot A + B \log(1/\theta) + 4B\theta \log 2
\end{aligned}$$

Choosing $\theta = 1/n$ leads to

$$T_{\mathrm{E}}(\mathrm{TTF}) \leq \frac{(m-1)[(1+2/n)\log 2n] + [(1+2/n)\log n + \log 16/n]}{\log \lambda_2(\tilde{W})^{-1}}.$$

I TTF vs. Lazy TTF($\hat{p}$) in Terms of Time Complexity Bounds

The comparison is made considering the upper bounds given in Th. 4 and Th. 6. The justification (by numerical experiments) that such a comparison makes sense appears in Sect. 6.1. The main goal of this section is to illustrate the effect of the application of the optimal vector $\hat{p}$ on TTF and its scheduler, in the sense explained in Sect. 5.

The numerical experiments, in this section, are performed as follows. First, we generate randomly the probability matrix P to simulate a non-uniformly random scheduler $S(P)$. Here, due to the lack of space, we present results for only 3 representative schedulers. Second, by solving OP_2 (Sect. 5), we get the best $\hat{p}$ for lazy TTF(p) and we compare the upper bounds on convergence time w.h.p. of TTF and of lazy TTF($\hat{p}$) (Th. 4 and Th. 6, see Tab. 2). At last, we interpret the values of $\hat{p}$ for a better understanding of lazy TTF (Fig. 3).

The systems under consideration are composed of four agents, the base station (BST), the fastest agent f, the slowest agent nf_2, and an intermediate

non-fastest agent (nf_1). Thus, $\mathrm{cv}_f < \mathrm{cv}_{nf_1} < \mathrm{cv}_{nf_2}$.
The matrix P, for each considered $S(P)$ of the three, is encoded by edge labels in a complete interaction graph corresponding to the considered population. The three schedulers are depicted in Fig. 2. In the graph (a), take for instance the label 0.021 between f and BST. It means that the probability that the next interaction concerns the fastest agent (as initiator) and BST, is 0.021. Table 2

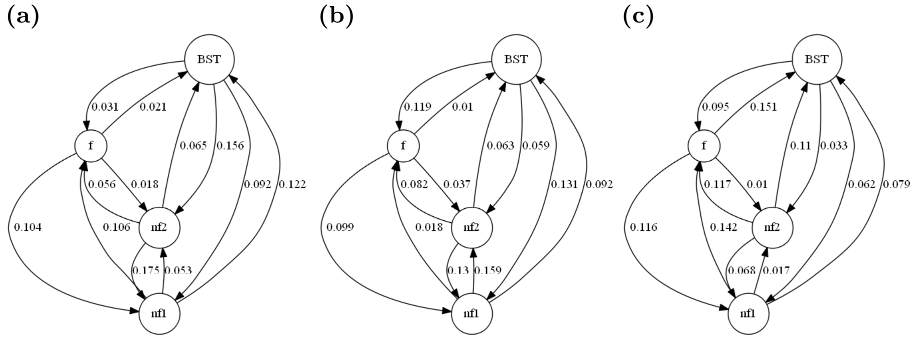

Fig. 2. Graph illustrations for three random schedulers.

below presents the numerical results for each scheduler given in Fig. 2. Second column gives the vector $\hat{p}$ in the order $[\mathrm{BST}, f, nf_1, nf_2]$. The third and forth columns present the upper bounds on the time w.h.p. for TTF and lazy TTF($\hat{p}$), computed using Th. 4 and Th. 6 respectively and rounded down to integers.

Lines **a** and **b** in Table 2 show that the tentative to save energy by inhibiting some interactions, e.g., $\hat{p}_{nf_2}(\mathbf{a}) = 0$ and $\hat{p}_{nf_2}(\mathbf{b}) = 0$, is not really significant for the execution time bounds (80 vs. 67, 23 vs. 22). In line **c** (in case of scheduler (c), Fig. 2) lazy TTF behaves as TTF.

S	$\hat{p}$	TTF	*lazy*
a	$[1, 1, 0, 0]$	80	67
b	$[1, 0.35, 1, 0]$	23	22
c	$[1, 1, 1, 1]$	13	13

Table 2. Time complexity of TTF vs. lazy TTF($\hat{p}$).

As explained in Sect. 5, lazy TTF($\hat{p}$) under $S(P)$ is equivalent to TTF under $S(P')$, where $P'_{i,j} = P_{i,j} \times \hat{p}_i, \forall (i,j) \in \mathbf{E}$. Fig. 3 presents the interaction graphs encoding P' for schedulers **(a)** and **(b)**.

Observe that according to graph (**a**) in Fig. 2, for both nf_1 and nf_2, the probability to meet BST is about the same ($0.092 + 0.122 = 0.214$ vs. $0.065 + 0.156 = 0.221$). Moreover, the probability for a token to be transferred to BST from nf_1 or nf_2 through f is very small. Thus, when meeting nf_1, a better heuristic for nf_2 is to wait for meeting BST, rather than to transfer tokens to

nf_1. The value obtained for $\hat{p}$ confirms this heuristic, since $\hat{p}_{nf_1} = \hat{p}_{nf_2} = 0$ (line **a**, Table 2).
According to graph (**b**), Fig. 2, nf_1 and nf_2 have a better probability to meet BST than f. Thus, when nf_1 or nf_2 meets f, transferring tokens to f does not seem to be a good choice. This intuition is confirmed by the computed value for $\hat{p}$, since f executes TTF with a small probability 0.35 (line **b**, Table 2).

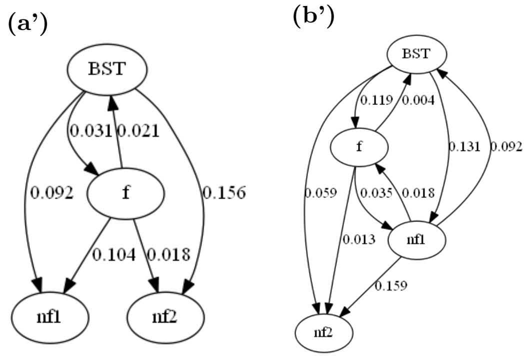

Fig. 3. Resulting schedulers with $\hat{p}$.