Gathering in Carrier Graphs: Meeting via Public Transportation System

Haozhi Zheng
Graduate School of Science and Technology, Nara Institute of Science and Technology, Japan

Ryota Eguchi
Graduate School of Science and Technology, Nara Institute of Science and Technology, Japan

Fukuhito Ooshita
Faculty of Engineering, Fukui University of Technology, Japan

Michiko Inoue
Graduate School of Science and Technology, Nara Institute of Science and Technology, Japan

Abstract

The *gathering problem* requires multiple mobile agents in a network to meet at a single location. This paper investigates the gathering problem in *carrier graphs*, a subclass of *recurrence of edge* class of *time-varying graphs*. By focusing on three subclasses of single carrier graphs – *circular*, *simple*, and *arbitrary* – we clarify the conditions under which the problem can be solved, considering prior knowledge endowed to agents and obtainable online information, such as the count and identifiers of agents or sites. We propose algorithms for solvable cases and analyze the complexities and we give proofs for the impossibility for unsolvable cases. We also consider general carrier graphs with multiple carriers and propose an algorithm for arbitrary carrier graphs. To the best of our knowledge, this is the first work that investigates the gathering problem in carrier graphs.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases Gathering, Carrier Graph, Time-varying Graph, Mobile agent

Digital Object Identifier 10.4230/LIPIcs.SAND.2024.21

Funding This work was supported by JSPS KAKENHI Grant Numbers 22K11903, 23H03347. The second author was supported by JSPS KAKENHI Grant Number 22H03569.

1 Introduction

Imagine a group of friends attending a music festival in a large park. They have agreed to meet somewhere to enjoy the festival together. However, they become separated in the crowd and have no means of wireless communication. They decide to use the festival shuttle bus, a public transportation carrier that travels around the park on a fixed schedule, to meet at an undetermined location. The challenge of determining when to get on and off the bus and how to ensure that everyone is at the same place is formalized as a *gathering problem*, particularly, gathering problem in a *carrier graph*.

The *gathering problem* requires multiple mobile agents in a network to meet at a single location. The gathering facilitates information sharing among agents working on collaborative tasks. Distributed algorithms for this problem including the *rendezvous* problem (gathering for two agents) have been well studied, especially for static graphs [1, 4, 10, 17].

Recently, distributed algorithms for *highly dynamic graphs* have been intensively studied. These are dynamic graphs whose dynamics are not restricted locally in time or space, that is, graphs are continuously changing over time. Casteigts et al. integrated several concepts of dynamic graphs investigated separately as *time-varying graphs* and sorted them into 15 classes [3]. One meaning class is *connectivity over time* $\mathcal{COT}$ (or *temporally connected* [8]) that is a class of graphs where any pair of two nodes are connected over time in both directions (an agent can reach from one side to another side when it adequately waits and moves in some intermediate nodes). Not surprisingly, due to its weak connectivity assumption, most results for $\mathcal{COT}$ are negative and gathering algorithms achieve only weak properties [2, 16].

As subclasses of $\mathcal{COT}$, more constrained (or more easily handled) classes are defined. Classes *Constant connectivity* $\mathcal{CC}$ and T*-interval connectivity* $\mathcal{INT}_T$ are graphs that guarantee connectivity for any moment. A graph in $\mathcal{INT}_T$ keeps the same spanning tree for any period of T consecutive time steps, and $\mathcal{CC}$ is the same as 1*-interval connectivity* $\mathcal{INT}_1$. These classes have inclusion relation $\mathcal{COT} \supset \mathcal{CC} = \mathcal{INT}_1 \supset \mathcal{INT}_2 \supset \cdots$. For 1-interval connected graphs, especially 1-interval connected rings, gathering, and related problems have been well studied [14, 9, 15, 18].

Another interesting subclasses of $\mathcal{COT}$ are *recurrence of edges* $\mathcal{RE}$, *time-bounded recurrence of edges* $\mathcal{BRE}$ and *periodicity of edges* $\mathcal{PE}$. These are graphs where any edge recurrently appears if it appears at least once. The recurrence of edges is bounded in time in $\mathcal{BRE}$ and it is periodic in $\mathcal{PE}$. The inclusion relation among these classes is $\mathcal{COT} \supset \mathcal{RE} \supset \mathcal{BRE} \supset \mathcal{PE}$. The gathering problem has not been well studied in these classes. There is one work on gathering problem considering $\mathcal{COT}$, $\mathcal{CC}$, $\mathcal{RE}$, $\mathcal{BRE}$ and static graphs [2].

For a class of *carrier graphs*, that is a subclass of $\mathcal{PE}$, the *exploration* problem has been studied [6, 11]. The carrier graph (C-graph) models a system where one or more *carriers* periodically visit sites in the system by following their routes. Agents can move sites with a carrier when some carrier comes to their current site. Practical examples of this model include public transportation systems like buses [20], planes [12] and satellites [19]. Furthermore, this model also finds relevance in the context of ad-hoc data-routing schemes [13, 21]. The exploration requires an agent to visit all the nodes in a network. This is closely related to the gathering since agents need to explore a network to achieve gathering.

This paper considers the gathering problem in carrier graphs. This can be seen as a problem where people try to meet at some station (unknown in advance) in a public transportation system. We consider several assumptions on prior knowledge of agents such as the counts of agents or sites, and, on acquirable information at sites such as identifiers or the number of agents at the site or the site identifier, and for each assumption, clarify the solvability of the problem and propose algorithms for solvable cases.

1.1 Related Works

The gathering problems targeting time-varying graphs are well studied for a family of constantly connected graphs, especially 1-interval connected graphs [2, 14, 15, 18]. Di Luna et al. first investigated the gathering problem in 1-interval connected rings [14]. They first showed gathering at a single node is impossible in 1-interval connected rings and considered the *weak gathering* that allows gathering at the same node, or the two end nodes of the same edge. The feasible initial configurations (initial configurations from which the problem is solvable) were clarified and gathering algorithms were proposed under several assumptions on chirality and cross-detection. Michail et al. expanded targeting graphs beyond rings and examined the solvability of the weak gathering problem for the class of 1-interval connected graphs and initial configurations [15]. Shibata et al. considered the *partial gathering* problem where each agent is required to gather with a group of at least g agents for a given g for 1-interval connected graphs [18]. They clarified the solvability and proposed algorithms for solvable cases under several assumptions on g.

Ooshita and Datta considered weak gathering on rings in $\mathcal{COT}$ [16]. They proved that in $\mathcal{COT}$, weak gathering is impossible when agents cannot leave information at nodes or when all agents should terminate. They also proposed a weak gathering algorithm without termination when agents can leave information at nodes.

Bournat et al. had a unique work on the gathering in time-varying graphs [2]. They considered not only the constant connectivity class $\mathcal{CC}$ but also the family of recurrence of edges classes $\mathcal{RE}$ (recurrence of edges) and $\mathcal{BRE}$ (time-bounded recurrence of edges). They clarified the solvability for four variants of the gathering problem, gathering (all agents gather in bounded time), eventual gathering (all agents gather in finite time), weak gathering (all agents but at most one gather in bounded time, a different definition from [14]) and eventual weak gathering (all agents but at most one gather in finite time), and proposed a single algorithm that solves the strongest feasible variant for $\mathcal{COT}$, $\mathcal{CC}$, $\mathcal{RE}$, $\mathcal{BRE}$ and static graphs. This is the only work that considers the gathering problem for the family of recurrence of edge classes, and there is no work on the gathering problem targeting carrier graphs.

The carrier graph model was introduced as a subclass of time-varying graphs by Flocchini et al. [6]. This work considers anonymous systems (sites possess no IDs) and non-anonymous systems (sites possess unique IDs), and for both systems established the necessary and sufficient conditions for exploration. To showcase the time complexity differences across various settings, they introduced three carrier graph classes – circular, simple, and arbitrary. The agents in this research are constrained to move exclusively with the carrier and lack the capability to wait on sites, analogous to the context of low-earth orbiting satellite systems.

In contrast, many real-world public transportation systems allow agents to stay at a station, enabling them to await a potentially distinct carrier. Ilcinkas and Wade extended this perspective by allowing agents to leave the carrier and wait on sites [11]. They demonstrated that this added capability enables agents to reduce the number of moves in the worst case. Additionally, this ability allows the agent agents not only to achieve exploration but also to map the whole carrier graph.

Flocchini et al. studied the mapping of carrier graphs with “black holes” which are sites that destroy agents [5, 7]. Their investigations delved into collectively mapping the graph by multiple agents with the ability to leave messages at sites. The goal was to collaboratively construct a map of the carrier graph while minimizing agent loss.

1.2 Our Contributions

This paper considers the gathering problem for carrier graphs. We examine the solvability and propose algorithms (if solvable) for three classes of *circular*, *simple*, and *arbitrary* carrier graphs with a single carrier. The solvability and the time complexity of the gathering problem under several conditions are summarized in Table 1, where p, P, k, and n denote the period, an upper bound of the period, the number of agents, and the number of sites, respectively. Note that some of the results, e.g. for circular or simple graphs with knowledge of P, are automatically derived from the results for the superclass, e.g. for arbitrary graphs with knowledge of P. We also proof the impossibility for unsolvable cases.

Furthermore, we prove the impossibility of gathering in C-graphs with multiple carriers. We also propose a gathering algorithm that terminates in at most $2M + (2p - 2)(m - 1) + 2p$ rounds using existing exploration algorithms, where m and M denote the number of carriers and the termination time of the exploration algorithm, respectively.

2 Preliminaries

2.1 Carrier graphs

We consider a system composed of a set S of n sites and a set C of m *carriers*. The sites have unique identifiers (IDs) or no ID depending on assumptions, and carriers move among the sites. Each carrier c has a unique identifier $id(c)$ and an ordered sequence of sites

Table 1 Time complexity of gathering algorithms on single carrier graphs.

Assumptions		Graph Class		
prior knowledge	observation ability	Circular	Simple	Arbitrary
P	-	$p+P^*$	$p+P^*$	$p+P^*$
k	-	p^*	p^*	p^*
n	-	$2p^*$	$p+n(n-1)^*$	Impossible
n	agent ID	$2p^*$	$p+n(n-1)^*$	Impossible
n	site ID	$2p^*$	$2p$	$2p$
-	agent ID	$3p$	$4p-1$	Impossible
-	site ID	$2p^*$	$2p+1^*$	Impossible
-	agent ID & site ID	$2p^*$	$2p+1^*$	Impossible

$*$) algorithms *with simultaneous termination*

Table 2 Possibilities of gathering algorithms on general carrier graphs.

Assumptions		Possibility
prior knowledge	observation ability	
P	-	Possible
k	site ID	Impossible
n	-	Impossible
n	site ID	Possible

$\pi(c) = \langle s_0, s_1, \ldots, s_{p(c)-1}\rangle, s_i \in S$, called a *route*, where the positive integer $p(c)$ is called a *period* of the route. The carrier c starts at site s_0 at time 0 and then moves to the next site along the route at each time unit in a cyclic manner (moving from $s_{p(c)-1}$ to s_0). We use $s_{p(c)}$ as $s_{p(c)} = s_0$ for convenience. Letting $\pi(c)[j]$ denote a site where c is located at time j, $\pi(c)[j] = s_i$ holds where $i \equiv_{p(c)} j$. A set of all the sites appear in $\pi(c)$ is called a *domain* of c, denoted as $S(c) = \bigcup_{0\le i\le p(c)-1}\{s_i\}$ where $\bigcup_{c\in C} S(c) = S$ holds. We have $|S(c)| \le p(c)$ since the same site can be visited several times along the route.

Each route $\pi(c)$ defines an arc-labelled multi-graph $\vec{G}(c) = (S(c), \vec{E}(c))$, where $\vec{E}(c) = \{(s_i, s_{i+1}, i) : 0 \le i < p(c)\}$. The set of all routes of carriers is denoted by $R = \{\pi(c) : c \in C\}$, and a period of R is defined as $p(R) = \max\{p(c) : c \in C\}$. When no ambiguity arises, we will simply denote $p(R)$ as p. The arc-labelled multi-graph $\vec{G}(C) = (S, \vec{E})$, where $\vec{E} = \bigcup_{c\in C}\vec{E}(c)$, is called *carrier graph*, or shortly, *C-graph*. Especially, a carrier graph with only one carrier is called a *single carrier graph*, or shortly, SC-graph.

For any C-graph $\vec{G}(C)$, we define a static and undirected *meeting graph* $H(C)$ that has C as a set of nodes. In the meeting graph $H(C)$, there is an edge between two nodes c and c' if and only if there exists a site s such that both $s \in S(c)$ and $s \in S(c')$ hold. A C-graph $\vec{G}(C)$ is said to be connected if and only if $H(C)$ is connected. In this paper, we will always consider *connected* C-graphs.

We classify routes by their property into *circular*, *simple*, and *arbitrary* as follows.

▶ **Definition 1.** *A route $\pi(c) = \langle s_0, s_1, \ldots, s_{p(c)-1}\rangle$ is* ***simple*** *if $\vec{G}(c)$ contains no self-loop or multiple arcs, that is $s_i \ne s_{i+1}$, for $0 \le i < p(c)$, and $(x, y, i), (x, y, j) \in \vec{E}(c)$ only if $i = j$.*

▶ **Definition 2.** *A simple route $\pi(c) = \langle s_0, s_1, \ldots, s_{p(c)-1}\rangle$ is* ***circular*** *if it contains no repeated sites, that is $|S(c)| = p(c)$.*

A carrier graph $\vec{G}(C) = (S, \vec{E})$ is said to be *circular* and *simple* if every route $\pi(c) \in R$ is circular and simple, respectively. Let $\mathcal{C}_C$, $\mathcal{C}_S$, and $\mathcal{C}_A$ denote classes of circular, simple, and arbitrary C-graphs, respectively. Obviously, we have $\mathcal{C}_C \subset \mathcal{C}_S \subset \mathcal{C}_A$.

We have the following lemma as $|S(c)| \leq p$.

▶ **Lemma 3.** *Every site is visited by some carrier at least once for any time interval* $[t, t']$ *(*$t' - t \geq p - 1$*).*

2.2 Mobile Agents

There are k mobile computational entities $a_0, a_1, \ldots, a_{k-1} \in A$, called *agents*, in the system, with unique identifiers as $id(a_i) \in \mathbb{N}, 0 \leq i \leq k - 1$. The prior knowledge of the system for agents depends on the assumption considered.

Agents operate in a LOOK-COMPUTE-MOVE manner in each synchronous round j, which is the interval between time $j - 1$ and time j where $j \in \mathbb{N}^+$. At the beginning of each round, an agent gets information on the current site (LOOK operation). The number of agents at the same site can be observed by default, while the ability to observe agents' IDs and the site's ID is endowed by certain assumptions. The agent then determines whether it will move or stay at the current site (COMPUTE operation). Then the agent performs the move depending on the decision (MOVE operation). Agents' memory is persistent across rounds (non-oblivious). We assume that agents cannot observe other agents' memory.

An agent can stay at the current site or move with one of the carriers. An agent a_i can move with or switch to a carrier c only when it is placed at the same site as c at the same time. Agent a_i at site s at time t will be at site s' at time $t + 1$ if it moves with or switches to carrier c, where $(s, s', i) \in \vec{E}$ and $t \equiv_{p(c)} i$ hold. Otherwise, agent a_i stays at site s.

2.3 The Gathering Problem

The goal of the *gathering* problem is to gather all the agents within finite time, that is to let every agent $a \in A$ move to the same site $s \in S$ and terminate, within finite time, regardless of the starting position. Moreover, the gathering problem *with simultaneous termination* requires all the agents to terminate an algorithm simultaneously at the same site.

We assume that an agent starts its execution either spontaneously upon encountering a carrier or upon encountering another moving agent. To evaluate the time complexity of an algorithm, we measure the number of rounds from time 0 (representing the initial configuration) to the point at which all agents have terminated their operations. We designate the initiation of execution by the first agent as round 1, and subsequently, time 1 shows the configuration after the execution of round 1.

3 Gathering with One Carrier

This section explores the gathering problem on SC-graphs. The process of congregating all agents in an SC-graph is straightforward: agents are instructed to move with the carrier whenever they encounter it. Consequently, all agents are assured to be located at the same site at any round $t \geq p$. However, the challenge is how each agent decides when to terminate its execution. We will showcase the impossibility of gathering under certain assumptions. Next, we will show several feasible assumptions under which we propose algorithms to solve the gathering problem. We will prove their correctness and assess their time complexity.

3.1 Impossibilities

At first, we show the impossibility of gathering for arbitrary SC-graphs.

▶ **Theorem 4.** *There does not exist any algorithm that solves the gathering problem in arbitrary SC-graphs when the agents have no prior knowledge even when they can observe IDs of agents and the current site ID.*

Proof. For contradiction, suppose that there exists an algorithm $\mathcal{A}$ that solves the gathering problem in the settings of the statement of the theorem.

Consider a SC-graph G with n sites $s_0, \dots, s_{n-1}$. Assume that a single carrier c has a route $\pi(c) = \langle s_0, \dots, s_{n-1} \rangle$ and n agents $a_0, \dots, a_{n-1}$ exist at sites $s_0, \dots, s_{n-1}$ initially. By algorithm $\mathcal{A}$, there exists T such that all agents gather and terminate at some site $s \in S(c)$ in T rounds.

We consider another SC-graph G' with $n'+1$ sites $s'_0, \dots, s'_{n'}$. Assume that a single carrier c' has a route $\pi(c') = \langle s'_0, \dots, s'_0, s'_1, \dots, s'_{n'} \rangle$ where c' visits s'_0 repeatedly T times and then visits $s'_1, \dots, s'_{n'}$. Also assume that n' agents $a'_0, \dots, a'_{n'-1}$ exist at sites $s'_1, \dots, s'_{n'}$ initially. Note that the agents do not observe a carrier during the first T rounds. By algorithm $\mathcal{A}$, there exists T' such that all agents gather and terminate at some site $s' \in \pi(c')$ in T' rounds.

Lastly, we construct another SC-graph G'' from the above two graphs G and G'. The sites of G'' include $s_0, \dots, s_{n-1}, s'_0, \dots, s'_{n'}$. Assume that n agents $a_0, \dots, a_{n-1}$ are initially placed at the same positions as G and n' agents $a'_0, \dots, a'_{n'-1}$ are initially placed at the same positions as G'. In this graph, we consider a single carrier c'' such that c'' moves similarly to G during the first T rounds and moves similarly to G' after that. That is, $\pi(c'')[j] = \pi(c)[j]$ for $0 \le t < T$ and $\pi(c'')[j] = \pi(c')[j]$ for $T \le t < T'$. Let us consider the behavior of agents $a_0, \dots, a_{n-1}$ in graph G''. During the first T rounds, carrier c'' moves similarly to G and consequently agents $a_0, \dots, a_{n-1}$ behave similarly to G. Hence, agents $a_0, \dots, a_{n-1}$ terminate at site s in T rounds. Next consider the behavior of agents $a'_0, \dots, a'_{n'-1}$. During the first T rounds, none of them observes a carrier, which is the same as G'. During time T to T', carrier c'' moves similarly to G'. Hence, agents $a'_0, \dots, a'_{n'-1}$ behave similarly to G' and terminate at site s' in T' rounds. This implies that agents terminate at different two nodes s and s'. This is a contradiction. ◀

▶ **Theorem 5.** *There does not exist any algorithm that solves the gathering problem in arbitrary SC-graphs when the agents only have knowledge of n, even when they can observe IDs of agents.*

Proof. For contradiction, suppose that there exists an algorithm $\mathcal{A}$ that solves the gathering problem in the settings of the statement of the theorem. In the proof, we assume $n = 2k + 2$.

Consider a SC-graph G_0 with n sites $s_0, \dots, s_{n-1}$. Assume that a single carrier c_0 has a route $\pi(c_0) = \langle s_0, \dots, s_{n-1} \rangle$ and k agents $a_0, \dots, a_{k-1}$ exist at sites $s_0, \dots, s_{k-1}$ initially. By algorithm $\mathcal{A}$, there exists T such that all agents gather and terminate in T rounds. Let $S_{half} = \{s_0, \dots, s_k\}$. The following claim shows that we can change the movement of the carrier so that agents gather and terminate in T rounds without going out from S_{half}.

▷ Claim 6. There exists a SC-graph G_1 with n sites $s_0, \dots, s_{n-1}$ such that (1) the single carrier c_1 visits only sites in S_{half} during the first T rounds, and (2) when k agents $a_0, \dots, a_{k-1}$ are initially placed at the same positions as G_0, they gather and terminate in T rounds.

Proof. First, we introduce some terms. The state of an agent is a tuple of its ID and the values of all variables. A state of a site is a tuple of states of agents if agents exist, or empty otherwise. We say a site is occupied if some agents exist on the site. For G_i ($i \in \{0, 1\}$), we define $\phi_i^t(s)$ as the state of site s at time t in G_i. Let $S = \{s_0, \dots, s_{n-1}\}$.

In the following, we inductively prove that, by defining $\pi(c_1)$ carefully, for $0 \le t \le T$, (a) $\pi(c_1)[t] \in S_{half}$ holds, (b) every agent exists on a node in S_{half} at time t, and (c) there exists a bijection $f^t : S \to S$ that maps site s of G_0 to site $f^t(s)$ of G_1 so that $\phi_1^t(f^t(s)) = \phi_0^t(s)$ holds for any $s \in S$ and $\pi(c_1)[t] = f^t(\pi(c_0)[t])$ holds. Clearly, this implies the claim.

For the base case, consider time 0. We define $\pi(c_1)[0] = \pi(c_0)[0] = s_0 \in S_{half}$. This implies condition (a) for $t = 0$. Since all agents in G_1 are initially placed at the same positions as G_0, conditions (b) and (c) hold for $t = 0$ by defining $f^0(s) = s$ for any $s \in S$.

To prove inductive cases, for $t = \ell < T$, assume that conditions (a) to (c) hold. That is, $\phi_1^\ell(f^\ell(s)) = \phi_0^\ell(s)$ holds for any $s \in S$ and $\pi(c_1)[\ell] = f^\ell(\pi(c_0)[\ell])$ holds. Let $v_0 = \pi(c_0)[\ell]$, $w_0 = \pi(c_0)[\ell+1]$, and $v_1 = \pi(c_1)[\ell] = f^\ell(v_0)$. We define $\pi(c_1)[\ell+1]$ as follows: $\pi(c_1)[\ell+1] = f^\ell(w_0)$ if $f^\ell(w_0) \in S_{half}$ holds, and otherwise $\pi(c_1)[\ell+1] = w \in S_{half}$ such that w is not occupied in G_1 at time ℓ. In the latter case, from condition (b), w_0 is not occupied in G_0 at time ℓ, and hence we use some non-occupied site $w \in S_{half}$ instead of $f^\ell(w_0) \notin S_{half}$. From $k < |S_{half}|$, such w definitely exists. This definition implies conditions (a) and (b) for $r = \ell + 1$.

In the following, we prove condition (c) for $t = \ell + 1$. Let $w_1 = \pi(c_1)[\ell+1]$ and define u_0 as a site satisfying $w_1 = f^\ell(u_0)$. We define bijection $f^{\ell+1}$ as follows:

- If $w_1 = f^\ell(w_0)$ holds, we define $f^{\ell+1}(s) = f^\ell(s)$ for any $s \in S$.
- If $w_1 \neq f^\ell(w_0)$ holds, we define $f^{\ell+1}$ by exchanging sites mapped from w_0 and u_0, that is, $f^{\ell+1}(w_0) = f^\ell(u_0) = w_1$, $f^{\ell+1}(u_0) = f^\ell(w_0)$, and $f^{\ell+1}(s) = f^\ell(s)$ for any $s \in S \setminus \{w_0, u_0\}$.

In both cases, $w_1 = f^{\ell+1}(w_0)$ and hence $\pi(c_1)[\ell+1] = f^{\ell+1}(\pi(c_0)[\ell+1])$ hold.

In the rest, we prove $\phi_1^{\ell+1}(f^{\ell+1}(s)) = \phi_0^{\ell+1}(s)$ for any $s \in S$. We first consider an arbitrary site $x_0 \in S \setminus \{w_0, u_0\}$. Let $x_1 = f^{\ell+1}(x_0) = f^\ell(x_0)$. In this case, x_0 (resp., x_1) is not the destination of a carrier in G_0 (resp., G_1) during the round between time ℓ to $\ell+1$. If x_0 is not occupied in G_0 at time ℓ, x_1 is also not occupied in G_1 from $\phi_1^\ell(x_1) = \phi_0^\ell(x_0)$. If x_0 is occupied, from $\phi_1^\ell(x_1) = \phi_0^\ell(x_0)$ and $\pi(c_1)[\ell] = f^\ell(\pi(c_0)[\ell])$, agents on x_1 in G_1 and agents on x_0 in G_0 observe the same site state and existence of a carrier, agents on x_1 in G_1 behave as the same as those on x_0 in G_0. This implies $\phi_1^{\ell+1}(f^{\ell+1}(x_0)) = \phi_1^{\ell+1}(x_1) = \phi_0^{\ell+1}(x_0)$.

Next consider the case of $x_0 = w_0$ (including the case of $u_0 = w_0$). In this case, during the round between time ℓ to $\ell+1$, carrier c_0 moves from v_0 to w_0 in G_0, and carrier c_1 moves from $v_1 = f^\ell(v_0)$ to $w_1 = f^{\ell+1}(w_0)$ in G_1. We can observe $\phi_1^\ell(w_1) = \phi_0^\ell(w_0)$. Indeed this is trivial if $w_1 = f^\ell(w_0)$, and otherwise w_0 and w_1 are not occupied at time ℓ and hence $\phi_1^\ell(w_1) = \phi_0^\ell(w_0)$ holds. Similarly to the previous discussion, agents on v_1 (resp., w_1) in G_1 and those on v_0 (resp., w_0) in G_0 behave similarly. Consequently, carrier c_1 carries agents (if any) to w_1 such that the agents have the same states as G_0. Hence, $\phi_1^{\ell+1}(f^{\ell+1}(w_0)) = \phi_1^{\ell+1}(w_1) = \phi_0^{\ell+1}(w_0)$ holds.

Lastly, consider the case of $x_0 = u_0 \neq w_0$. In this case, u_0 in G_0 and $f^{\ell+1}(u_0)$ in G_1 are not occupied at time $\ell+1$. This implies $\phi_1^{\ell+1}(f^{\ell+1}(u_0)) = \phi_0^{\ell+1}(u_0)$. ◁

Next we consider another SC-graph G'_0 with n sites $s'_0, \ldots, s'_{n-1}$. Assume that a single carrier c'_0 has a route $\pi(c'_0) = \langle s'_0, \ldots, s'_0, s'_1, s'_2, \ldots, s'_{n-1} \rangle$ where c'_0 visits s'_0 repeatedly T times and then visits $s'_1, \ldots, s'_{n-1}$. Also assume that k agents $a'_0, \ldots, a'_{k-1}$ exist at sites $s'_1, \ldots, s'_k$ initially. Note that the agents do not observe a carrier during the first T rounds. By algorithm $\mathcal{A}$, there exists T' such that all agents gather and terminate in T' rounds. Let $S'_{half} = \{s'_0, \ldots, s'_k\}$. Similarly to Claim 6, we can prove the following claim.

▷ Claim 7. There exists a SC-graph G'_1 with n sites $s'_0, \ldots, s'_{n-1}$ such that (1) the single carrier c'_1 visits only sites in S'_{half} during the first T' rounds, and (2) when k agents $a'_0, \ldots, a'_{k-1}$ are initially placed at the same positions as G'_0, they do not observe a carrier during the first T rounds, and gather and terminate in T' rounds.

Algorithm 1 Gather-With-Number-Of-Agents.

```
1: k ← total number of agents
2: for each round do
3:     agents ← the number of agents at the same site, carrier ← carrier at the same site
4:     if agents = k then
5:         Terminate
6:     end if
7:     if carrier ≠ ∅ then
8:         Move with carrier
9:     end if
10: end for
```

Lastly, we construct a SC-graph G_2 from the above two graphs G_1 and G_1'. The set of sites of G_2 is $S_{half} \cup S'_{half}$. Assume that initially k agents $a_0, \ldots, a_{k-1}$ are located at sites $s_0, \ldots, s_{k-1}$ like G_1 and other k agents $a'_0, \ldots, a'_{k-1}$ are located at sites $s'_1, \ldots, s'_k$ like G'_1. We consider a single carrier c_2 such that c_2 moves similarly to G_1 during the first T rounds and moves similarly to G'_1 after that up to round T' (and it can move arbitrarily). That is, $\pi(c_2)[j] = \pi(c_1)[j]$ for $0 \le j < T$ and $\pi(c_2)[j] = \pi(c'_1)[j]$ for $T \le j < T'$. Let us consider the behavior of agents $a_0, \ldots, a_{k-1}$ in graph G_2. During the first T rounds, carrier c_2 moves similarly to G_1 and consequently agents $a_0, \ldots, a_{k-1}$ behave similarly to G_1. Hence, agents $a_0, \ldots, a_{k-1}$ terminate at a node in S_{half} in T rounds. Next consider the behavior of agents $a'_0, \ldots, a'_{k'-1}$. During the first T rounds, none of them observes a carrier, which is the same as G'_1. During time T to T', carrier c_2 moves similarly to G'_1. Hence, agents $a'_0, \ldots, a'_{k'-1}$ behave similarly to G'_1 and terminate at a node in S'_{half} in T' rounds. This implies that agents terminate at different two sites s and s'. This is a contradiction. ◀

3.2 Possibilities

In this subsection, we present algorithms for SC-graphs for several scenarios on the initial knowledge of agents and/or the information acquired in the LOOK operations.

3.2.1 With Prior Knowledge of agents amount k

We start with a simple case where each agent knows the number k of agents. This allows the agents to determine when to terminate as they can observe the count of agents at the same site. Each agent moves with a carrier whenever it encounters the carrier, and terminates once it can see k agents.

▶ **Theorem 8.** *Algorithm 1 solves the gathering problem with simultaneous termination within p rounds for arbitrary SC-graphs if agents know the total number k of agents.*

Proof. In this algorithm, any agent moves with the carrier whenever the carrier arrives at its current site. Through this process, all k agents inevitably gather in p rounds and terminate simultaneously, as a consequence of lines 5, 6, 8, 9, and Lemma 3. ◀

3.2.2 With Prior Knowledge of Period p

We present the algorithm when the agents initially know an upper bound of the period of a given SC-graph in Algorithm 2. Let P denote an upper bound on the period p of the single carrier. The idea of the algorithm is simple: The agents maintain a variable *steps* initially set to 0, and increment the variable in each round. When the carrier appears in the

Algorithm 2 Gather-With-Period.

```
1: P ← an upper bound on the carrier's period, steps ← 0, pre-agents ← 1
2: for each round do
3:     agents ← the number of agents at the same site, carrier ← carrier at the same site
4:     if steps = P then
5:         Terminate
6:     else if agents = pre-agents then
7:         steps ← steps + 1
8:     else
9:         steps ← 0
10:    end if
11:    pre-agents ← agents
12:    if carrier ≠ ∅ then
13:        Move with carrier
14:    end if
15: end for
```

current site, the agent moves sites along the carrier. The value of *steps* is reset to 0 when the number of agents in the carrier increases, which is detected by comparing the number of agents at the current round and the number of agents in the previous round. The number of agents in the previous round is maintained by another variable *pre-agents*. Finally, the agents terminate when the value of *steps* is equal to the value of the upper bound of the period. The algorithm finishes gathering within $p + P$ rounds.

▶ **Lemma 9.** *All the agents operating Algoritm 2 are located at the same site and have identical steps values at the end of any round* $t \geq p$*.*

Proof. As illustrated in lines 15 and 16, an agent moves with the carrier upon encounter and remains on the carrier until termination. From this observation, we can see that all agents are present at the same site after round p, by Lemma 3. Let $t_0(\leq p)$ be the round when the carrier encounters the last agent. Each agent operates *steps* $\leftarrow 0$ at round t_0, and increments it by one in each round after round t_0. That is all the agents have identical *step* values at the end of each round $t \geq p > t_0$. ◀

▶ **Theorem 10.** *Algorithm 2 solves the gathering problem for arbitrary SC-graphs within* $p + P$ *rounds if agents know an upper bound* P *of the period* p *of the carrier.*

Proof. In the case of $k = 1$, the variable *steps* of the only agent increases every round until *steps* $= P$. Then the agent terminates at the P-th round. In the case of $k > 1$, all k agents congregate at the same location on the carrier with identical *steps* values by the end of any round $t \leq p$ from Lemma 9. Hence, at a certain round $r \leq p + P$, all k agents terminate their execution at the same site simultaneously. ◀

Algoritm 2 can solve the problem when the initial knowledge of the agents is n (the number of sites) in circular SC-graphs. This is because we have $n = p$ by the definition of the circular route (in Definition 2). Therefore, by substituting n for p, the agents can gather within $2p$ rounds.

▶ **Corollary 11.** *The gathering problem can be solved for circular SC-graphs within* $2p$ *rounds if agents know the number* n *of sites.*

In simple SC-graphs, the agent can obtain an upper bound $n(n-1)$ of the period, since $p \leq n(n-1)$ holds from the definition of the simple route (in Definition 1). Therefore, the agents can gather within $p + n(n-1)$ rounds.

Algorithm 3 Gather-With-Site-ID-and-n.

```
n ← total number of sites, sites ← ∅, explore-mode ← true
for each round do
    current ← current site ID, sites ← {current} ∪ sites, carrier ← carrier at the same site
    if |sites| = n then
        explore-mode ← false
    end if
    if carrier ≠ ∅ then
        if explore-mode = false ∧ current = min (sites) then
            Terminate
        else
            Move with carrier
        end if
    end if
end for
```

▶ **Corollary 12.** *The gathering problem can be solved for simple SC-graphs within $p+n(n-1)$ rounds if agents know the number n of sites.*

The same strategy also can be applied to the situation where agents can observe the ID of the current site in each LOOK operation in circular and simple SC-graphs. In this case, agents find the period p while executing Algorithm 2. Each agent checks site IDs and the number of moves when it moves with the carrier. When it encounters the same site (arc in the case of simple SC-graphs) twice, it detects the end of the first cycle and gets the period p. Let $t_0(\leq p)$ be the round when the carrier encounters the last agent. The value of *steps* reaches to p at round $t_0+p \leq 2p$, while the last agent gets the value p at round t_0+p in circular SC-graphs and at t_0+p+1 in simple SC-graphs. The strategy can work well if agents decide termination when *steps* value becomes $p+1$ in circular and simple SC-graphs. The modified algorithm solves the gathering problem within $2p$ and $2p+1$ rounds in circular and simple SC-graphs.

▶ **Corollary 13.** *The gathering problem can be solved for circular and simple SC-graphs within $2p$ and $2p+1$ rounds if agents can observe the current site's ID.*

Algorithms derived from Algorithm 2 also solve the problem *with simultaneous termination* since Lemma 9 guarantees that the *steps* value is identical for all the agents before terminating.

3.2.3 With Prior Knowledge of n and Ability to Observe Site ID

Another straightforward approach directs the agents to gather at the site with the smallest ID. Agents determine the minimum site ID while exploring the entire graph with knowledge of the total number n of sites and the ability to observe the current site's ID.

▶ **Theorem 14.** *Algorithm 3 solves the gathering problem for arbitrary SC-graphs within $2p$ rounds if agents can obtain the current site's ID and know the total number n of sites.*

Proof. In this algorithm, the operation is organized into two distinct phases denoted by the parameter *explore-mode*. Initially, each agent operates in *explore-mode*, where the agent explores the entire graph with the carrier. The agent exits *explore-mode* by round p when all sites in the graph have been visited, as indicated by the condition $|sites| = n$. Subsequently, the agent moves with the carrier up to p rounds toward the site with the smallest ID, where it finally terminates. Since all the agents terminate the same site with the smallest ID, the gathering is achieved within $2p$ rounds. ◀

Algorithm 4 Gather-With-Agent-ID-Circular.

```
leader ← ⊥, landmark ← ⊥, leader-acknowledged ← false
for each round do
    agents ← {IDs at the same site}, carrier ← carrier at the same site
    if carrier ≠ ∅ then
        if leader = ⊥ ∨ min(agents) < leader then
            leader ← min(agents), leader-acknowledged ← false
        end if
        if leader = id then
            if landmark ≠ ⊥ ∧ landmark ∈ agents then          ▷ found the landmark
                Terminate
            else if landmark = ⊥ ∧ |agent| ≥ 2 then
                landmark ← min (agents \ {id})
            end if
            Move with carrier
        else if leader-acknowledged = false ∧ leader ∈ agents then
            leader-acknowledged ← true
            Stay at the site                ▷ start to wait for a leaderless carrier
        else if leader ∉ agents then
            Move with carrier                                 ▷ head to the leader
        else
            Terminate                                         ▷ found the leader
        end if
    end if
end for
```

Algoritm 3 does not attain *gathering with simultaneous termination* due to the lack of information about other agents' exploration progress.

3.2.4 With Ability to Observe Other Agents' ID

An alternative approach involves designating a special agent as a leader, who terminates first. Then, the remaining agents move to and terminate at the leader's position. To determine the leader agent, agents observe each other's unique ID. The leader's termination serves as a signal for the other agents to gather at the designated position.

The algorithm for gathering in circular SC-graphs relies on the election of two special agents, namely the *leader* and the *landmark*. All agents will learn the leader's ID when the election is over since the leader will meet every other agent during the leader election process. The leader then moves to the location of the landmark and terminates with the landmark. Subsequently, other agents observing a carrier without a leader will recognize that it is time to head toward the leader. They will terminate upon encountering the leader again. The following theorem holds for Algorithm 4. The proof is in the Appendix.

▶ **Theorem 15.** *Algorithm 4 solves the gathering problem within $3p$ rounds for circular SC-graphs when $k > 1$, if agents can obtain the IDs of other agents at the same site.*

Proof. We show that Algorithm 4 makes all agents gather within $3p$ rounds. The proof proceeds with showing some claims.

▷ Claim 16. There are no terminated agents at round $t < p$.

Proof. If an agent terminates as a leader (line 10), it first sets some ID to *landmark* variable and meets again its landmark to terminate. If an agent terminates as non-leader (line 21), it first meets with the leader and meets the leader again to terminate. Both cases require agents to encounter the carrier at least twice. To encounter the carrier twice, any agent needs $p + 1$ rounds. ◁

▷ Claim 17. By round p, the agents riding the carrier include the agent with the smallest agent ID, and it moves with the carrier until termination.

Proof. In the first cycle of the carrier's move, every agent first encounters the carrier and moves with the carrier as a leader if it has the smallest ID among agents at the site (line 14). This implies the agent a_{min} with the smallest ID becomes a leader (sets its own *id* to *leader* variable) when it first encounters the carrier at round t_0 ($\leq p$) and solely moves with the carrier until termination at round $p + 1$ or later by Claim 16. ◁

We then show that a_{min} again meets its landmark by round $2p$.

▷ Claim 18. The agent a_{min} arrives at the landmark's site by round $t = 2p$.

Proof. When a_{min} first meets other agents, it sets one agent ID a_{land} to its *landmark*. We first show that a_{min} first meets a_{land} by round p. When a_{min} first encounters the carrier, if there are other agents at the same site, a_{min} selects a_{land} among these agents, that is, a_{min} first meets a_{land} by p. Otherwise, since one agent moves with the carrier after the carrier encounters some agents, this means that a_{min} is the first agent that the carrier encounters and will meet a_{land} at its initial location by round p. Let t ($\leq p$) be the round when a_{min} meets a_{land} first time. Since a_{land} does not become a leader, it remains to stay at the site. Since a_{min} moves with the carrier until termination by Claim 17, it again meets a_{land} at time $t + p$ ($\leq 2p$). ◁

After reaching the landmark's location, the leader terminates its execution. Subsequently, any agents situated at distinct sites observe the leaderless carrier and move along it. Following an additional p rounds from the leader's termination, the carrier returns to the leader's site, bringing all agents to this site.

In summary, the algorithm guarantees the gathering of all agents to a solitary site. This gathering is accomplished within $3p$ rounds for circular SC-graphs. ◀

The leader election process on circular SC-graphs does not work for simple SC-graphs as it is, since a site may appear multiple times along a simple route. Fortunately, a simple route includes no repeated arcs. We can use an arc as a landmark. Algorithm 5 for simple SC-graphs has a similar approach to that for circular SC-graphs. It elects the smallest agent as the leader, and the leader sets *landmark* by placing two agents *pre-landmark* and *post-landmark* at both endpoints of the landmark when it first meets two other agents. The leader's termination is triggered upon observing pre-landmark and post-landmark in this order in consecutive two rounds.

In Algorithm 5, two variables *pre-landmark* and *post-landmark* are used to indicate a landmark. An agent sets a landmark when first meets two other agents. To do so, agents move with the carrier while there are two or fewer agents together before setting a landmark (while *leader-acknowledged* is false). The leader terminates if it again meets *pre-landmark* and *post-landmark* in this order in consecutive rounds. Other agents behave similarly to Algorithm 4. They stay at the site to wait for a leaderless carrier, but if it is selected as a post-landmark (*leader-acknowledged* is false and it has the second smallest ID among agents

Algorithm 5 Gather-With-Agent-ID-Simple.

```
leader ←⊥, pre-landmark ← ⊥, post-landmark ← ⊥
leader-acknowledged ← false, visit-pre-landmark ← false, post-landmark-move ← false
for each round do
    agents ← {IDs at the same site}, carrier ← carrier at the same site
    if carrier ≠ ∅ then
        if leader = ⊥ ∨ min (agents) < leader then
            leader ← min (agents), leader-acknowledged ← false
        end if
        if leader = id then
            if pre-landmark ≠ ⊥ ∧ pre-landmark ∈ agents then
                visit-pre-landmark ← true
            else if visit-pre-landmark then
                if post-landmark ∈ agents then                ▷ found the landmark
                    Terminate
                else
                    visit-pre-landmark ← false
                end if
            else if pre-landmark = ⊥ ∧ |agents| ≥ 3 then
                pre-landmark ← 3rd-min(agents), post-landmark ← 2nd-min(agents)
            end if
            Move with carrier
        else if leader-acknowledged = false ∧ |agents| ≥ 3 ∧ leader ∈ agents then
            leader-acknowledged ← true
            if 2nd-min(agents) = id then
                post-landmark-move ← true
                Move with carrier
            else
                Stay at the site                ▷ start to wait for a leaderless carrier
            end if
        else if post-landmark-move = true then
            post-landmark-move ← false
            Stay at the site                ▷ start to wait for a leaderless carrier
        else if leader-acknowledged = false ∨ leader ∉ carrier then
            Move with carrier                ▷ elect or head to the leader
        else
            Terminate                ▷ found the leader
        end if
    end if
end for
```

in the current site) it has one more move. After the leader terminates, the other agents see a leaderless carrier and head to the leader with the carrier. The following theorem holds for Algorithm 5.

▶ **Theorem 19.** *Algorithm 5 solves the gathering problem within* $4p - 1$ *rounds for simple SC-graphs when* $k > 2$ *if agents can obtain the IDs of other agents at the same site.*

Proof. We will see a difference from Algorithm 4. Let a_{min} denote the agent with the smallest ID.

▷ Claim 20. The agent a_{min} arrives at the landmark's arc by round $3p - 1$.

Proof. When a_{min} first meets two other agents, it sets two agent IDs a_{pre_land} and a_{post_land} to its *pre-landmark* and *post-landmark*, respectively. We first show that a_{min} first meets a_{pre_land} and a_{post_land} by round $2p-2$. The agent a_{min} meets at least one agent by round p

as in Algorithm 4. However, if a_{min} meets only one agent a' at that time, it has to move more with the carrier to meet more agents (a' accompanies a_{min} since its *leader-acknowledged* is $false$). The worst case is that a_{min} is initially located at $\pi(c)[p-1]$, and one agent is located at $\pi(c)[0]$ and the other agents are located at $\pi(c)[p-3]$. In this case, The second-smallest agent a' once set its landmark at $(\pi(c)[p-3], \pi(c)[p-2])$ at rounds $p-2$ and $p-3$ and all the agents except a_{min} and a' stay one of the endpoints. Thus, a_{min} takes another $p-2$ moves to meet the pre-landmark of a' at $\pi(c)[p-3]$ and sets a_{min}'s landmark at $(\pi(c)[p-3], \pi(c)[p-2])$ at round $2p-2$ and $2p-3$. Then a_{min} takes another p moves to again meet its *pre-landmark* and *post-landmark* in this order. That is, a_{min} again arrives at the landmark's arc at time $3p-1$. ◁

After reaching the landmark's location, the leader terminates its execution. Subsequently, any agents situated at distinct sites observe the leaderless carrier and move along it. Following an additional p rounds from the leader's termination, the carrier returns to the leader's site, bringing all agents to this site.

In summary, the algorithm guarantees the gathering of all agents to a solitary site. This gathering is accomplished within $4p-1$ rounds for circular SC-graphs. ◀

Given the inherently asynchronous nature of the termination process, achieving the goal of *gather with simultaneous termination* remains unattainable.

4 Gathering on Multi-Carrier C-Graphs

4.1 Impossibility

According to Flocchini et al. [6], exploration on anonymous C-graphs (sites' IDs not observable) is impossible without knowledge of P, even if n and k are given, and is impossible without knowledge of either P or n when the site ID is visible, even if k is given. Thus, gathering on anonymous C-graphs is impossible without prior knowledge of P as shown by Theorem 21.

▶ **Theorem 21.** *There is no algorithm that solves the problem of gathering on C-graphs that agents terminate before every site is visited.*

Proof. Assume algorithm $\mathcal{A}$ solves the gathering problem without making all agents visit every site. Let G_1 be a C-graph that a site $s_1 \in S(G_1)$ will not be visited by an agent A_1 operating $\mathcal{A}$ starting from a certain initial configuration. Similarly, let G_2 be a copy of G_1 so that an agent A_2 will not visit a site $s_2 \in S(G_2)$ in an execution starting from the same initial configuration as before. Then, let G_3 be a C-graph that is a combination of G_1 and G_2. That is, $S(G_3)$ has all the carriers and sites of G_1 and G_2, and all the carriers and agents are located at the same sites as before at the round 0. Let a carrier c move periodically between s_1 and s_2, so that the C-graph G_3 becomes connected. Because A_1 will terminate without visiting s_1 and so for A_2 and s_2, the two agents will never meet.This is a contradiction. ◀

In non-anonymous C-graphs, solving the gathering problem is trivial because agents must explore, as indicated by Theorem 21. This exploration leads agents to move to the site with the smallest ID.

Algorithm 6 Gather-With-Multiple-Carriers.

```
1: For each agent i, explores and maps the graph G⃗(C_i) by A_E
2: p ← p(C_i), m ← |C_i|, n ← |S(C_i)|
3: destination ← min(C_i)
4: Compute the foremost path to destination then move to it
5: Wait until round M + (2p − 2)(m − 1)
6: Wait until an agent comes for at most M rounds
7: Operate Algorithm 2 with period p
```

4.2 Gathering on Anonymous C-graphs

When gathering on an anonymous C-graph with multiple carriers, we adopt a two-tiered approach. Initially, agents gather in the meeting graph $H(C)$, which means arriving at the same carrier's route. Then, agents use algorithms designed for SC-graphs to gather on that carrier. If agents have learned about the identities (IDs), routes, and timetables of all carriers, as well as the topology of the C-graph, agents can directly gather at the carrier with the smallest ID with at most $(2p-2)(m-1)$ rounds as Lemma 22 shows. Another $2p$ round is then required for operating Algorithm 2 to gather all the agents on the same site since agents know the period of the carrier of the smallest ID.

▶ **Lemma 22.** *The foremost path (the path that an agent arrives at its destination at the earliest time) from any site to any carrier costs at most* $(2p-2)(m-1)$ *rounds.*

Proof. The basic scenario involves two connected carriers c_1 and c_2. Since there are at most p sites in each carrier, the longest path from c_1 to c_2 is $p-1$. In the worst case, the agent at a shared site needs to wait for the carrier's coming for $p-1$ rounds. Therefore, the accumulative time for the foremost path is at most $2p-2$.

Since the longest foremost path involves at most m carriers with $m-1$ times carrier switching, the foremost path costs at most $(2p-2)(m-1)$ rounds. ◀

For an unknown C-graph, agents can learn the required knowledge by assigning names to the visited sites so that each agent will make a private map of the C-graph after visiting every site. Suppose we have an algorithm $\mathcal{A}_E$ that maps the C-graph in M rounds, the gathering problem can be solved in at most $M+(2p-2)(m-1)+2p$ rounds.

▶ **Theorem 23.** *The Algorithm 6 solves the gathering problem for C-graphs in* $2M+(2p-2)(m-1)+2p$ *rounds with a mapping algorithm* $\mathcal{A}_E$*, which maps the C-graph in* M *rounds.*

Proof. The topology and the dynamics are known to the agent by round M as the exploration ends. Therefore, every agent will be at a site belonging to the route of the carrier with the smallest ID by the $M+(2p-2)(m-1)$-th round. That allows Algorithm 2 to solve the gathering problem in another $2p$ rounds.

In a scenario where each agent starts up at the beginning of round 0, agents agree on the global time so that every agent begins Algorithm 2 at round $2M+(2p-2)(m-1)$ that ensures the gathering at round $2M+(2p-2)(m-1)+2p$.

Otherwise, when each agent starts up separately, there is a delay d between the first and last started agents. As we suppose an agent will start its gathering process after encountering another moving agent, the delay d can be at most M rounds. The Line 6 eliminates the effects of the delay and leads to a gathering time of $2M+(2p-2)(m-1)+2p$ rounds. ◀

Ilcinkas and Wade [11] propose an exploration algorithm EXPLORE-WITH-WAIT that maps a C-graph. Given the a priori knowledge of an upper bound $B=\mathcal{O}(p)$ on the maximum period p, the worst-case time complexity is $\Theta(np)$. Subsequently, the time complexity of a EXPLORE-WITH-WAIT version of Algorithm 6 is $\Theta(np)$ as $m=\mathcal{O}(n)$.

5 Conclusion

In this study, we started an exploration of a variant of the gathering problem: gathering in carrier graphs, a particular class of time-varying graphs.

Throughout our investigation, we analyzed several factors that affect the feasibility and the time complexity of gathering in single carrier graphs. Then, we extended our algorithms to solve the gathering problem in general carrier graphs.

Open questions remaining include exploring the impact of communication abilities on the gathering problem in C-graphs, identifying additional factors that may influence the feasibility of gathering, and extending our findings to encompass a wider array of dynamic graph classes, such as bounded-recurrent-edge graphs.

References

1 Steve Alpern and Shmuel Gal. *The theory of search games and rendezvous*, volume 55 of *International series in operations research and management science*. Kluwer, 2003. `doi:10.1007/b10080`.

2 Marjorie Bournat, Swan Dubois, and Franck Petit. Gracefully degrading gathering in dynamic rings. In *Stabilization, Safety, and Security of Distributed Systems - 20th International Symposium*, volume 11201 of *Lecture Notes in Computer Science*, pages 349–364. Springer, 2018. `doi:10.1007/978-3-030-03232-6_23`.

3 Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. Time-varying graphs and dynamic networks. *Int. J. Parallel Emergent Distributed Syst.*, 27(5):387–408, 2012. `doi:10.1080/17445760.2012.668546`.

4 Yoann Dieudonné, Andrzej Pelc, and David Peleg. Gathering despite mischief. *ACM Trans. Algorithms*, 11(1):1:1–1:28, 2014. `doi:10.1145/2629656`.

5 Paola Flocchini, Matthew Kellett, Peter C. Mason, and Nicola Santoro. Finding good coffee in paris. In Evangelos Kranakis, Danny Krizanc, and Flaminia L. Luccio, editors, *Fun with Algorithms - 6th International Conference*, volume 7288 of *Lecture Notes in Computer Science*, pages 154–165. Springer, 2012. `doi:10.1007/978-3-642-30347-0_17`.

6 Paola Flocchini, Bernard Mans, and Nicola Santoro. On the exploration of time-varying networks. *Theoretical Computer Science*, 469:53–68, 2013. `doi:10.1016/j.tcs.2012.10.029`.

7 Paola Flocchini, Nicola Santoro, Peter C. Mason, and Matthew Kellett. Black hole search in the network and subway models. *Theoretical Computer Science*, 50:158–184, 2012. `doi:10.1007/s00224-011-9341-8`.

8 Tsuyoshi Gotoh, Paola Flocchini, Toshimitsu Masuzawa, and Nicola Santoro. Tight bounds on distributed exploration of temporal graphs. In Pascal Felber, Roy Friedman, Seth Gilbert, and Avery Miller, editors, *23rd International Conference on Principles of Distributed Systems*, volume 153 of *LIPIcs*, pages 22:1–22:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. `doi:10.4230/LIPICS.OPODIS.2019.22`.

9 Tsuyoshi Gotoh, Yuichi Sudo, Fukuhito Ooshita, and Toshimitsu Masuzawa. Dynamic ring exploration with (h, S) view. *Algorithms*, 13(6):141, 2020. `doi:10.3390/A13060141`.

10 Jion Hirose, Junya Nakamura, Fukuhito Ooshita, and Michiko Inoue. Gathering with a strong team in weakly byzantine environments. In *Proceedings of the 22nd International Conference on Distributed Computing and Networking*, ICDCN '21, pages 26–35. Association for Computing Machinery, 2021. `doi:10.1145/3427796.3427799`.

11 David Ilcinkas and Ahmed M. Wade. Exploration of carrier-based time-varying networks: The power of waiting. *Theoretical Computer Science*, 841:50–61, 2020. `doi:10.1016/j.tcs.2020.07.003`.

12 Ari Keränen and Jörg Ott. DTN over aerial carriers. In Martin May, Gunnar Karlsson, and Ellen W. Zegura, editors, *Proceedings of the 4th ACM workshop on Challenged networks*, pages 67–76. ACM, 2009. `doi:10.1145/1614222.1614234`.

13 Cong Liu and Jie Wu. Scalable routing in cyclic mobile networks. *IEEE Trans. Parallel Distributed Syst.*, 20(9):1325–1338, 2009. doi:10.1109/TPDS.2008.218.

14 Giuseppe Antonio Di Luna, Paola Flocchini, Linda Pagli, Giuseppe Prencipe, Nicola Santoro, and Giovanni Viglietta. Gathering in dynamic rings. *Theor. Comput. Sci.*, 811:79–98, 2020. doi:10.1016/J.TCS.2018.10.018.

15 Othon Michail, Paul G. Spirakis, and Michail Theofilatos. Beyond rings: Gathering in 1-interval connected graphs. *Parallel Process. Lett.*, 31(4):2150020:1–2150020:31, 2021. doi:10.1142/S0129626421500201.

16 Fukuhito Ooshita and Ajoy K. Datta. Brief announcement: Feasibility of weak gathering in connected-over-time dynamic rings. In *Stabilization, Safety, and Security of Distributed Systems - 20th International Symposium*, volume 11201 of *Lecture Notes in Computer Science*, pages 393–397. Springer, 2018. doi:10.1007/978-3-030-03232-6_27.

17 Andrzej Pelc. Deterministic rendezvous algorithms. In Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro, editors, *Distributed Computing by Mobile Entities, Current Research in Moving and Computing*, volume 11340 of *Lecture Notes in Computer Science*, pages 423–454. Springer, 2019. doi:10.1007/978-3-030-11072-7_17.

18 Masahiro Shibata, Yuichi Sudo, Junya Nakamura, and Yonghwan Kim. Partial gathering of mobile agents in dynamic rings. In Colette Johnen, Elad Michael Schiller, and Stefan Schmid, editors, *Stabilization, Safety, and Security of Distributed Systems - 23rd International Symposium*, volume 13046 of *Lecture Notes in Computer Science*, pages 440–455. Springer, 2021. doi:10.1007/978-3-030-91081-5_29.

19 Shin-Ywan Wang, J. Leigh Torgerson, Joshua Schoolcraft, and Yan Brenman. The deep impact network experiment operations center monitor and control system. In *2009 Third IEEE International Conference on Space Mission Challenges for Information Technology*, pages 34–40, 2009. doi:10.1109/SMC-IT.2009.13.

20 Xiaolan Zhang, Jim Kurose, Brian Neil Levine, Donald F. Towsley, and Honggang Zhang. Study of a bus-based disruption-tolerant network: mobility modeling and impact on routing. In Evangelos Kranakis, Jennifer C. Hou, and Ram Ramanathan, editors, *Proceedings of the 13th Annual International Conference on Mobile Computing and Networking*, pages 195–206. ACM, 2007. doi:10.1145/1287853.1287876.

21 Wenrui Zhao and M.H. Ammar. Message ferrying: proactive routing in highly-partitioned wireless ad hoc networks. In *The Ninth IEEE Workshop on Future Trends of Distributed Computing Systems*, pages 308–314, 2003. doi:10.1109/FTDCS.2003.1204352.