

1 **Exploration of the T -Interval-Connected Dynamic Graphs:**
2 **the Case of the Ring**

3 **David Ilcinkas · Ahmed M. Wade**

4
5 Received: date / Accepted: date

6 **Abstract** In this paper, we study the T -interval-connected dynamic graphs from the
7 point of view of the time necessary and sufficient for their exploration by a mobile
8 entity (agent). A dynamic graph (more precisely, an evolving graph) is T -interval-
9 connected ($T \geq 1$) if, for every window of T consecutive time steps, there exists
10 a connected spanning subgraph that is stable (always present) during this period.
11 This property of connection stability over time was introduced by Kuhn, Lynch and
12 Oshman [14] (STOC 2010). We focus on the case when the underlying graph is a
13 ring of size n , and we show that the worst-case time complexity for the exploration
14 problem is $2n - T - \Theta(1)$ time units if the agent knows the dynamics of the graph,
15 and $n + \frac{n}{\max\{1, T-1\}}(\delta - 1) \pm \Theta(\delta)$ time units otherwise, where δ is the maximum
16 time between two successive appearances of an edge.

17 **Keywords** Exploration · Dynamic graphs · T-interval-connectivity · Mobile agent

A preliminary version of this paper appeared in the Proceedings of the 20th International Colloquium on Structural Information and Communication Complexity (SIROCCO 2013) [13].

Partially supported by the ANR projects DISPLEXITY (ANR-11-BS02-014) and MACARON (ANR-13-JS02-002).

D. Ilcinkas
LaBRI, CNRS & Univ. Bordeaux, France
Tel.: +33 5 4000 69 12
Fax: +33 5 4000 66 69
E-mail: david.ilcinkas@labri.fr

A.M. Wade
École Polytechnique de Thiès, Senegal
Tel.: +221 77 142 66 70
E-mail: awade@ept.sn

1 Introduction

Partly due to the very important increase of the number of communicating objects that we observe today, the distributed computing systems are becoming more and more dynamic. The computational models for static networks are clearly not sufficient anymore to capture the behavior of these new communication networks. One can nevertheless consider the appearances and disappearances of nodes or edges due to the dynamic nature of the network as (topological) failures. But even the computational models that take into account a certain degree of fault tolerance become insufficient for some very dynamic networks. Indeed, the classical models of fault tolerance either assume that the frequency of fault occurrences is small, which gives enough time to the algorithm to adapt to the changes, or that the system stabilizes after a certain amount of time (as in the self-stabilizing systems for example). Therefore, in the last decade or so, many more or less equivalent models have been developed that take into account the extreme dynamism of some communication networks. An interested reader will find in [4] a very complete overview of the different models and studies of dynamic graphs (see also [15] and [16]).

One of the first developed models, and also one of the most standard, is the model of evolving graphs [7]. To simplify, given a static graph G , called the underlying graph, an evolving graph based on G is a (possibly infinite) sequence of spanning but not necessarily connected subgraphs of G (see Section 2 for precise definitions). Differently speaking, the node set does not change but edges can appear or disappear at each time instant. This model is particularly well adapted for modeling dynamic synchronous networks.

In all its generality, the model of evolving graphs allows to consider an extremely varied set of dynamic networks. Therefore, to obtain interesting results, it is often required to make assumptions that reduce the possibilities of dynamic graphs generated by the model. One example is the assumption of connectivity over time, introduced in [7], which states that there is a journey (path over time) from any vertex to any other vertex. Another example is the assumption of constant connectivity, for which the graph must be connected at all times. This latter assumption, which is very usual, has been recently generalized in a paper by Kuhn, Lynch and Oshman [14] by the notion of T -interval-connectivity (see also [17] for other kinds of generalizations). Roughly speaking, given an integer $T \geq 1$, a dynamic graph is T -interval-connected if, for any window of T consecutive time steps, there exists a connected spanning subgraph which is stable throughout the period. (The notion of constant connectivity is thus equivalent to the notion of 1-interval-connectivity). This new notion, which captures the connection stability over time, allows the finding of interesting results: the T -interval-connectivity allows to reduce by a factor of about $\Theta(T)$ the number of messages that is necessary and sufficient to perform a complete exchange of information between all the vertices [14] (gossip problem).

In this paper, we carry on the study of these T -interval-connected dynamic graphs by considering the problem of exploration. A mobile entity (called agent), moving from node to node along the edges of a dynamic graph, must traverse/visit each of its

61 vertices at least once (the traversal of an edge takes one time unit).¹ This fundamental
62 problem in distributed computing by mobile agents has been widely studied in static
63 graphs since the seminal paper by Claude Shannon [19].

64 As far as highly dynamic graphs are concerned, only the case of periodically-
65 varying graphs had been studied before the preliminary version of this paper [13],
66 both in the absence [10, 12] and in the presence of harmful nodes [8, 9]. Since then,
67 several works also considered the problem of exploring highly dynamic networks.

68 The most related papers are [11], a generalization of our paper to the case when
69 the underlying graph is a tree of cycles (a cactus), and [6], a part of which is a gen-
70 eralization to the cases of the graphs with bounded treewidth, in general and for
71 some specific subclasses, like the rings. Note that our results and those of [6] on
72 the rings were proved independently. In [5], the authors study the impact that syn-
73 chrony, anonymity and topological knowledge have on the computability and com-
74 plexity of the deterministic multi-agent exploration with termination of the 1-interval-
75 connected dynamic graphs based on the ring, in the case when the agents do not know
76 the dynamics of the graph. Finally, [3] presents and proves the correctness of a self-
77 stabilizing algorithm allowing three robots to perpetually explore a dynamic graph
78 based on the ring in which each node can reach infinitely often any other node.

79 Besides, several papers focus on the complexity of computing the optimal explo-
80 ration time of a dynamic graph given as (a centralized) input, in a similar manner as in
81 the Traveling Salesman Problem for static graphs. In the dynamic case, the problem
82 is called Temporal Graph Exploration Problem [6, 18] or Dynamic Map Visitation
83 Problem [1, 2]. In [2], the case of several agents is considered, while [6, 18] and most
84 of [1] consider the case of a single agent. In these papers, several polynomial-time
85 algorithms are given, either exact algorithms for specific graph classes, or approxi-
86 mation algorithms for the general cases. In particular, [1] gives an $O(n^2)$ algorithm to
87 compute the optimal exploration time of a given 1-interval-connected dynamic graph
88 based on the n -node ring. Inapproximability results for the general case are given
89 in [6, 18].

90 We focus here on the (worst-case) time complexity of this problem, namely the
91 number of time units used by the agent to solve the problem in the T -interval-con-
92 nected dynamic graphs. The problem of exploration, in addition to its theoretical inter-
93 ests, can be applied for instance to the network maintenance, where a mobile agent
94 has to control the proper functioning of each vertex of the graph.

95 We consider the problem in two scenarios. In the first one, often referred as the
96 offline scenario, the agent knows entirely and exactly the dynamic graph it has to
97 explore. This situation corresponds to predictable dynamic networks such as trans-
98 portation networks for example. In the second scenario, often referred as the online
99 scenario, the agent does not know the dynamics of the graph, that is the times of
100 appearance and disappearance of the edges. This case typically corresponds to net-
101 works whose changes are related to frequent and unpredictable failures. In this sec-
102 ond scenario, Kuhn, Lynch and Oshman [14] noted that the exploration problem is

¹ Note that several specializations of this problem exist, depending on whether the agent has to eventu-
ally detect termination (exploration with stop), return to its starting position (exploration with return),
or even visit each vertex infinitely often (perpetual exploration). The rest of the paper just considers the
general version of the problem.

impossible to solve under the single assumption of 1-interval-connectivity. In fact, it is quite easy to convince oneself that by adding the assumption that each edge of the underlying graph must appear infinitely often, the exploration problem becomes possible, but the time complexity remains unbounded. In this article, and only for the second scenario, we therefore add the assumption of δ -recurrence, for some integer $\delta \geq 1$: each edge of the underlying graph appears at least once every δ time units.

It turns out that the problem of exploration is much more complex in dynamic graphs than in static graphs. Indeed, let us consider for example the first scenario (known dynamic graph). The worst-case exploration time of n -node static graphs is clearly in $\Theta(n)$ (worst case $2n - 3$). On the other hand, the worst-case exploration time of n -node (1-interval-connected) dynamic graphs remains largely unknown. No lower bound better than the static bound is known, while the best known upper bound is quadratic, and directly follows from the fact that the temporal diameter of these graphs is bounded by n . Also, without the 1-interval-connectivity assumption, it is already NP-complete to decide whether exploration is feasible at all, while with this assumption the problem remains hard to approximate (see [6, 18]). In this paper, we focus on the study of T -interval-connected dynamic graphs whose underlying graph is a ring. Note that, in this particular case, the T -interval-connectivity property, for $T \geq 1$, implies that at most one edge can be absent at a given time.

Our results. We determine in this paper the exact time complexity of the exploration problem for the n -node T -interval-connected dynamic graphs based on the ring, when the agent knows the dynamics of the graph. This is essentially $2n - T - 1$ time units (see Section 3 for details). When the agent does not know the dynamics of the graph, we add the assumption of δ -recurrence, and we show that the complexity is $n + \frac{n}{\max\{1, T-1\}}(\delta - 1) \pm \Theta(\delta)$ time units in this case (see Section 4 for details).

2 Model and definitions

This section gives the precise definitions of the concepts and models informally mentioned in the introduction. Some definitions are similar or even identical to the definitions given in [14].

Definition 1 (Evolving graph) An *evolving graph* is a pair $\mathcal{G} = (V, \mathcal{E})$, where V is a static set of n vertices, and \mathcal{E} is a function which maps to every integer $i \geq 0$ a set $\mathcal{E}(i)$ of undirected edges on V .

Definition 2 (Underlying graph) Given an evolving graph $\mathcal{G} = (V, \mathcal{E})$, the static graph $G = (V, \bigcup_{i=0}^{\infty} \mathcal{E}(i))$ is called the *underlying graph* of \mathcal{G} . Conversely, the evolving graph \mathcal{G} is said to be *based* on the static graph G .

In this article, we consider the evolving graphs based on the n -node ring, denoted C_n . Since the cases $n = 1$ and $n = 2$ are trivial and somehow degenerated, we will assume $n \geq 3$.

141 **Definition 3 (T -interval-connectivity)** An evolving graph $\mathcal{G} = (V, \mathcal{E})$ is T -interval-
 142 *connected*, for an integer $T \geq 1$, if for every integer $i \geq 0$, the static graph $G_{[i, i+T[} =$
 143 $(V, \bigcap_{j=i}^{i+T-1} \mathcal{E}(j))$ is connected.

144 **Definition 4 (δ -recurrence)** An evolving graph is δ -*recurrent* if every edge of the
 145 underlying graph is present at least once every δ time steps.

146 **Definition 5 (Temporal diameter)** The *temporal diameter* of an evolving graph is
 147 the maximum time needed to go from any node to any other node starting at any time
 148 when at most one edge can be traversed at each time unit.

149 Note that the temporal diameter of any 1-interval-connected evolving graph is at
 150 most $n - 1$.

151 A mobile entity, called *agent*, operates on these dynamic graphs. We do not as-
 152 sume any limitation in terms of computational capabilities or memory. Nevertheless,
 153 the agent can traverse at most one edge per time unit. It may also stay at the current
 154 node (typically to wait for an incident edge to appear). We say that an agent *explores*
 155 the dynamic graph if and only if it visits all the nodes.

156 3 The agent knows the dynamics of the graph

157 In this section, we assume that the agent perfectly knows the dynamic graph to be
 158 explored.

159 3.1 Upper bound

160 The theorem presented in this subsection, Theorem 1, shows that the worst-case ex-
 161 ploration time is actually small, bounded by $2n$, when the underlying graph is a ring.
 162 Furthermore, it shows that the agent can benefit from the T -interval-connectivity to
 163 spare an additive term T (cf. Figure 1). Note that our upper bound is constructive,
 164 and tight (cf. Theorem 2).

165 The proof of Theorem 1 being quite long and technical, we first present a simpler
 166 and elegant proof, but giving a less precise upper bound, namely $2n - 2$ for any value
 167 of T . Nevertheless, we believe that this simplified result and its proof are of indepen-
 168 dent interest, because of the simplicity of the proof and the fact that the presented
 169 algorithm visits every node in the last $n - 1$ time units. Note that this implies that the
 170 agent is not changing direction and is never blocked during these $n - 1$ steps.

171 **Proposition 1** *For every integers $n \geq 3$ and $T \geq 1$, and for every T -interval-con-*
 172 *ected dynamic graph based on C_n , there exists an agent (algorithm) exploring this dy-*
 173 *namic graph in time at most $2n - 2$ such that this algorithm visits every node in the*
 174 *last $n - 1$ time units.*

175 *Proof* We first prove that, for any time t , there exists a node $v(t)$ such that an agent
 176 starting from $v(t)$ at time t and moving in the clockwise direction is not blocked in
 177 the $n - 1$ following time units, implying that such an agent has visited all nodes by

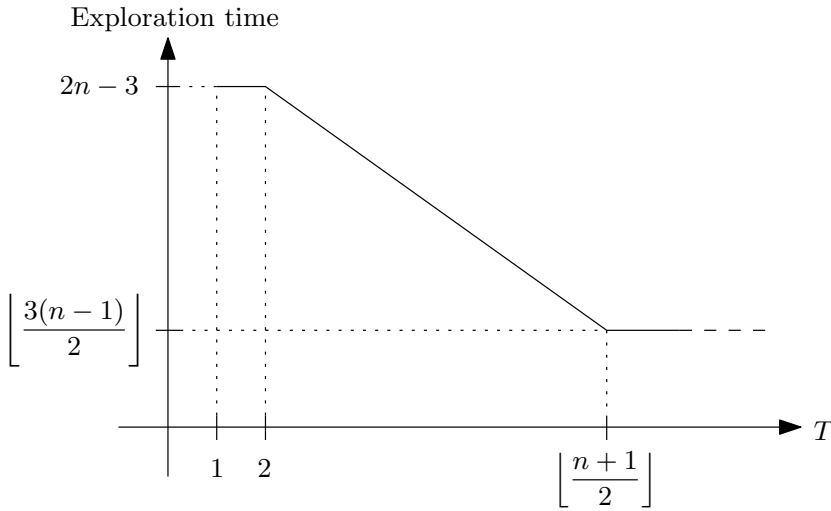


Fig. 1 Worst-case exploration time of the T -interval-connected dynamic graphs based on C_n as a function of T .

178 time $t + n - 1$. Indeed, consider that n virtual agents are placed on the n nodes of the
 179 graph at time t (one virtual agent on each node). Further consider that all these virtual
 180 agents go clockwise from time t on. At the first round (from time t), all virtual agents
 181 are trying to traverse different edges, so at most one virtual agent may be blocked.
 182 More generally, at each round, only one additional virtual agent can be blocked. Since
 183 there are n virtual agents, there exists at least one of them which is not blocked from
 184 time t to time $t + n - 1$. Its starting node $v(t)$ satisfy the desired properties.

185 We can now describe the algorithm satisfying the statement of the proposition.
 186 Let u be the starting node of the agent and let v be the node $v(n - 1)$ described in
 187 the first part of this proof. The algorithm simply consists in going from u to v in time
 188 at most $n - 1$ (recall that the temporal diameter of any 1-interval-connected dynamic
 189 graph is at most $n - 1$, so this is always possible), then in waiting at v until time $n - 1$
 190 (if v is reached sooner), and finally in going clockwise from v for $n - 1$ time units.
 191 The proposition follows from the properties of the node v . \square

192 Before proceeding with the formal theorem and its proof, let us informally de-
 193 scribe the key ingredients of the proof of the most general case.

194 We consider two algorithms, being the algorithms always going in the clockwise,
 195 resp. counter-clockwise, direction, traversing edges as soon as the dynamic graph
 196 allows it. At the beginning of the process, the two agents executing these algorithms,
 197 starting from the same initial position, try to traverse distinct edges and thus, at each
 198 time step, at least one of them progresses. During this phase, the average speed of the
 199 two agents is thus $1/2$ (edge traversals per time unit). However, when the agents are
 200 about to meet each other on an edge or on a node (thus after time at most n), their
 201 progression may be stopped by the absence of a unique edge e .

202 If this edge e is absent for at least $n - 1$ time steps, then any agent has enough time
 203 to change its direction and to explore all the nodes of the graph in the other direction,
 204 hence completing exploration within $2n$ steps, see Fig. 2.

205 If the edge e does not stay absent long enough and reappears at time t , we mod-
 206 ify the two algorithms as follows. The agent previously progressing in the clockwise,
 207 resp. counter-clockwise, direction, starts now by exploring the ring in the opposite di-
 208 rection, before going back in the usual direction the latest possible so that it reaches
 209 the edge e at time at most t . At time t , the two modified algorithms cross each other,
 210 and then continue their progression in their usual direction until one of them termi-
 211 nates the exploration. Note that, after time t , we have again the property that, at each
 212 time step, at least one agent progresses. See Fig. 3.

213 Globally, except during the period when e is absent, the average speed of the two
 214 agents is $1/2$. Besides, the modification of the algorithms generally allows each of the
 215 agents to explore an additional part of the ring. Unfortunately, these parts of the ring
 216 are traversed twice instead of once. Nevertheless, in the general case, the speed of
 217 both the modified agents is 1 during the period when e is absent. This compensates
 218 the loss induced by traversing twice some parts of the ring. Overall, the average speed
 219 is thus globally of at least $1/2$, which implies that at least one of the two modified
 220 agents performs exploration within time $2n$.

221 In order to obtain a better upper bound thanks to the T -interval connectivity, we
 222 use the following observation.

223 *Observation.* When a dynamic graph based on a ring is T -interval connected, all
 224 edges are present during $T - 1$ steps between the removal of two different edges.

225 We use this observation to gain an additive term of $T - 1$ on the exploration time,
 226 yielding to a time of roughly $2n - T$. A much more precise analysis of the modified
 227 algorithms allows us to obtain the exact claimed bounds.

228 **Theorem 1** For every integers $n \geq 3$ and $T \geq 1$, and for every T -interval-connected
 229 dynamic graph based on C_n , there exists an agent (algorithm) exploring this dynamic
 230 graph in time at most

$$\begin{cases} 2n - 3 & \text{if } T = 1 \\ 2n - T - 1 & \text{if } 2 \leq T \leq (n + 1)/2 \\ \lfloor \frac{3(n-1)}{2} \rfloor & \text{if } T > (n + 1)/2 \end{cases}$$

232 *Proof* Fix $n \geq 3$ and an arbitrary dynamic graph based on the ring C_n . Let v_0, \dots, v_{n-1}
 233 be the vertices of C_n in clockwise order. Assume that the agent starts exploration
 234 from v_0 at time 0. In order to prove this theorem, we will describe various algorithms,
 235 and we will show that at least one of them will allow the agent to perform exploration
 236 within the claimed time bound. Fix T to be any positive integer, and let \mathcal{T} be this
 237 bound.

238 First assume that at most one edge e is absent during the time interval $[0, \mathcal{T}]$.
 239 Then, an agent going to the closest extremity of e (in time at most $\lfloor (n - 1)/2 \rfloor$) and
 240 then changing direction (for $n - 1$ steps) will explore all nodes of the ring in time at

241 most $\lfloor 3(n-1)/2 \rfloor \leq \mathcal{T}$ (see Fig. 1). So let us assume from now on that at least two
 242 different edges are absent at least once each during the time interval $[0, \mathcal{T})$.

243 Before proceeding with the rest of the proof, we introduce the following nota-
 244 tions. Given a time interval I and two algorithms A and B , let d_A^I be the number of
 245 edge traversals performed by agent A during the time interval I , let α_A^I , resp. $\alpha_{A,B}^I$, be
 246 the number of time steps in I for which agent A , resp. both agents A and B , do(es)
 247 not move. (For all the algorithms we will consider, the reason why an agent will not
 248 move will always be the same: the edge it wants to traverse is absent.) Finally, let β^I
 249 be the number of time steps in I for which no edges are absent.

250 Let us now consider two simple algorithms. L , respectively R , is the algorithm
 251 always going in the counter-clockwise, resp. clockwise, direction, traversing edges
 252 as soon as the dynamic graph allows it. Now consider the sum of the number of
 253 edges traversed by each of the two algorithms until some time t . Since only one edge
 254 can be absent at a given time, this sum increases by at least one (and obviously by at
 255 most two) at each time step, until this sum is larger than or equal to $n-1$. So let e be
 256 the unique unexplored edge when this sum reaches $n-1$. If the sum jumps directly
 257 from $n-2$ to n , then fix e to be any of the last two unexplored edges. In both cases,
 258 let t_1 be the first time one of the two agents reaches one extremity of e . We consider
 259 two cases.

260 **Case 1.** The edge e is absent during the whole interval $[t_1, t_1 + n - 1)$.

261 In this case, the first agent to reach an extremity of e , at time t_1 , goes back in
 262 the opposite direction and explores the ring in $n-1$ further steps. This gives an
 263 exploration time of at most $t_1 + n - 1$. Let $I_1 = [0, t_1)$. We have

$$264 \quad t_1 = \begin{cases} d_L^{I_1} + \alpha_L^{I_1} & (1) \\ d_R^{I_1} + \alpha_R^{I_1} & (2) \end{cases}$$

265 and, since L and R are always trying to traverse distinct edges during I_1 and at
 266 most one edge may be removed at any time, we also have

$$267 \quad \alpha_L^{I_1} + \alpha_R^{I_1} + \beta^{I_1} \leq t_1. \quad (3)$$

268 Besides, by definition of t_1 , we have $d_L^{I_1} + d_R^{I_1} \leq n - 1$. (4)

269 Recall that we are considering the case when there are at least two removed differ-
 270 ent edges during the whole interval $[0, t_1 + n - 1)$. As mentioned before, when the
 271 dynamic graph is T -interval connected, all edges must be present during $T-1$
 272 steps between the removal of two different edges. This implies that during the
 273 whole interval $[0, t_1 + n - 1)$, there are at least $T-1$ steps when no edges are
 274 absent. By definition of Case 1, these steps must occur before time t_1 . Thus, we
 275 have

$$276 \quad \beta^{I_1} \geq T - 1 \quad (5)$$

277 Summing the first five (in)equalities, we obtain

$$278 \quad (1)+(2)+(3)+(4)+(5) \rightarrow t_1 + T \leq n,$$

279 or equivalently $t_1 + n - 1 \leq 2n - T - 1$, which gives the exact claimed bound for
 280 the general case $T \geq 2$. Fig. 2 illustrates the trajectories analyzed in Case 1.

281 For $T = 1$, this bound is one unit larger than the claimed bound. So let us further
 282 study the case $T = 1$. If the inequality (4) is in fact strict, then the correct bound is
 283 obtained. Otherwise, it means that at time $t_1 - 1$, both agents were free to move.

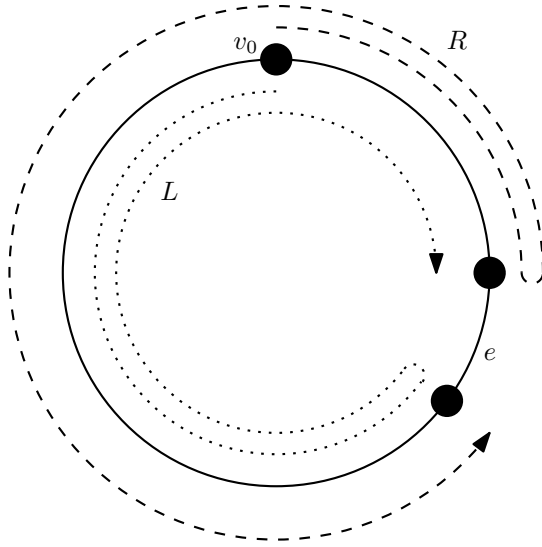


Fig. 2 In the case when edge e is absent for a long time (Case 1 in the proof), one of the dotted and dashed trajectories explores all nodes in the desired time.

284 This implies that either $\beta^{I_1} \geq 1$ (the inequality (5) is strict) or that the inequality
 285 (3) is strict. In both cases, this also gives the correct bound.

286 **Case 2.** The edge e is not absent during the whole interval $[t_1, t_1 + n - 1)$.

287 Then let t_2 be the time such that $t_2 + 1$ is the first time at which every edge has
 288 been explored by L or R (or both). Note that this definition implies that $t_2 \geq t_1$.
 289 We now define two new algorithms, one of which will explore the dynamic graph
 290 within time \mathcal{T} .

291 Let L' be the algorithm that is equal to L until some time t , at which L' goes back
 292 in the other direction forever. More precisely, L' is the algorithm for which t is
 293 the largest possible value such that L' and R share the same position at time t_2
 294 (intuitively, L' just has time to catch back R at time at most t_2). Similarly, let R'
 295 be the algorithm that is equal to R until some time t , at which R' goes back in
 296 the other direction forever. More precisely, R' is the algorithm for which t is the
 297 largest possible value such that R' and L share the same position at time t_2 .

298 In order to analyze the algorithms L' and R' , we introduce two other algorithms.
 299 Let L'' , respectively R'' be the algorithm defined as L' , resp. R' , but turning back
 300 exactly one time unit later than L' , resp. R' .

301 Finally, let \mathcal{T}_{exp} be the exploration time of the first between L' and R' exploring
 302 the dynamic graph, and let $I_1 = [0, t_1)$, $I_2 = [t_1, t_2)$, $I_{1,2} = [0, t_2)$, $I_3 = [t_2, \mathcal{T}_{exp})$,
 303 and $I = [0, \mathcal{T}_{exp})$.

304 As in the first case, we have

$$305 t_1 = \begin{cases} d_L^{I_1} + \alpha_L^{I_1} & (1) \\ d_R^{I_1} + \alpha_R^{I_1} & (2) \end{cases}$$

306 On I_1 , we have

$$\alpha_{L''}^{I_1} + \alpha_{R''}^{I_1} - \alpha_{L'',R''}^{I_1} + \beta^{I_1} \leq t_1. \quad (3)$$

Besides, L and R are always trying to traverse distinct edges during I_1 . Moreover, by definition, the algorithm L'' , resp. R'' , does not catch R , resp. L , before time t_2 (and thus t_1). This gives

$$\alpha_L^{I_1} + \alpha_R^{I_1} + \alpha_{L'',R''}^{I_1} + \beta^{I_1} \leq t_1 \quad (4)$$

$$(1)+(2)+(3)+(4) \rightarrow \alpha_{L''}^{I_1} + \alpha_{R''}^{I_1} + 2\beta^{I_1} \leq d_L^{I_1} + d_R^{I_1} \quad (5)$$

On $I_{1,2}$, we have

$$t_2 = \begin{cases} d_{L''}^{I_{1,2}} + \alpha_{L''}^{I_{1,2}} \\ d_{R''}^{I_{1,2}} + \alpha_{R''}^{I_{1,2}} \end{cases} \quad (6)$$

$$d_{L''}^{I_{1,2}} \leq d_{L'}^{I_{1,2}} + 1 \quad (8)$$

$$d_{R''}^{I_{1,2}} \leq d_{R'}^{I_{1,2}} + 1 \quad (9)$$

$$(6)+(7)+(8)+(9) \rightarrow 2t_2 \leq d_{L'}^{I_{1,2}} + d_{R'}^{I_{1,2}} + \alpha_{L''}^{I_{1,2}} + \alpha_{R''}^{I_{1,2}} + 2 \quad (10)$$

Note that, by definition of t_1 and t_2 , the edge e is absent during the whole interval I_2 . Besides, neither L' nor R' reaches an extremity of the edge e before turning back, because otherwise it would reach the other extremity too late, namely at time at least $t_1 + n - 1$, which is larger than t_2 by definition of Case 2. (By the way, this proves that $\mathcal{T}_{exp} > t_2$.) This implies that L'' and/or R'' may reach an extremity of e (at time t_1) but in this case turn back immediately before trying to traverse it. Moreover, L'' and R'' cannot reach an extremity of edge e while going clockwise, resp. counter-clockwise, before time t_2 . This means that they are never blocked during the time interval I_2 . This translates into

$$\alpha_{L''}^{I_{1,2}} = \alpha_{L'}^{I_1} \quad (11)$$

$$\alpha_{R''}^{I_{1,2}} = \alpha_{R'}^{I_1} \quad (12)$$

$$(5)+(10)+(11)+(12) \rightarrow 2t_2 + 2\beta^{I_1} \leq d_L^{I_1} + d_R^{I_1} + d_{L'}^{I_{1,2}} + d_{R'}^{I_{1,2}} + 2 \quad (13)$$

Starting from time $t_2 + 1$, the algorithms L' and R' are always trying to traverse distinct edges. Since L' and R' are not blocked at time t_2 , this means that, on I_3 , we have

$$\alpha_{L'}^{I_3} + \alpha_{R'}^{I_3} + \beta^{I_3} \leq \mathcal{T}_{exp} - t_2 \quad (14)$$

and

$$\mathcal{T}_{exp} - t_2 = \begin{cases} d_{L'}^{I_3} + \alpha_{L'}^{I_3} \\ d_{R'}^{I_3} + \alpha_{R'}^{I_3} \end{cases} \quad (15)$$

$$(14)+(15)+(16) \rightarrow \mathcal{T}_{exp} - t_2 + \beta^{I_3} \leq d_{L'}^{I_3} + d_{R'}^{I_3} \quad (17)$$

$$(17)+\frac{1}{2}(13) \rightarrow$$

$$\mathcal{T}_{exp} + \beta^{I_1} + \beta^{I_3} \leq \frac{1}{2}(d_L^{I_1} + d_R^{I_1} + d_{L'}^{I_{1,2}} + d_{R'}^{I_{1,2}}) + d_{L'}^{I_3} + d_{R'}^{I_3} + 1 \quad (18)$$

Let x , resp. y , be the number of edges traversed by L' , resp. R' , before turning back. Then

$$d_{L'}^{I_{1,2}} = 2x + d_R^{I_{1,2}} \quad (19)$$

$$d_{R'}^{I_{1,2}} = 2y + d_L^{I_{1,2}} \quad (20)$$

Counting the number of edges that still need to be traversed by L' and R' until exploration is performed, we obtain

$$d_{L'}^{I_3} \leq n - 1 - x - d_R^{I_{1,2}} \quad (21)$$

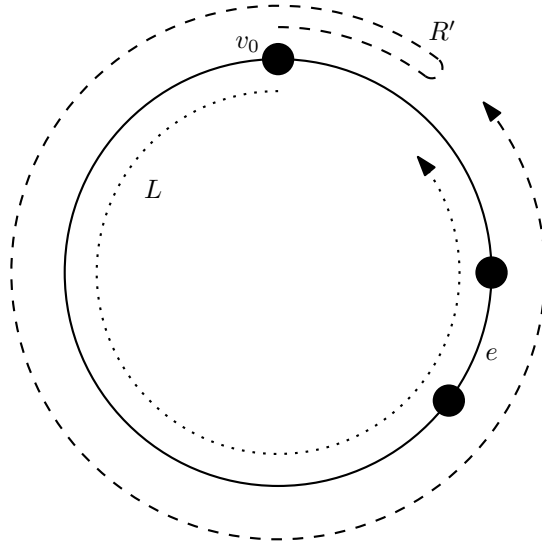


Fig. 3 In the case when edge e is absent for a short time (Case 2 in the proof), the dashed trajectory is used (or its equivalent in the other direction). The turning time of the dashed trajectory is defined at the latest possible time such that both the dashed and the dotted trajectories traverse edge e at the same time.

$$d_{R'}^{I_3} \leq n - 1 - y - d_L^{I_1,2} \quad (22)$$

Similarly as in Case 1, because of the T -interval connectivity and the hypotheses that at least two different edges are removed, we have

$$\beta^I \geq T - 1. \quad (23)$$

Besides, note that $\beta^{I_1} + \beta^{I_3} = \beta^I$.

Finally, since $d_L^{I_1} - d_L^{I_1,2}$ and $d_R^{I_1} - d_R^{I_1,2}$ are less than or equal to 0, we get $(18) + \frac{1}{2}(19) + \frac{1}{2}(20) + (21) + (22) + (23) \rightarrow \mathcal{T}_{exp} \leq 2n - T$.

In fact, we claim that this last inequality is strict. We will prove this claim by contradiction, assuming that inequalities (14), (21), (22), and (23) are in fact equalities. The equalities for (21) and (22) imply that the algorithms L' and R' are not blocked during the last step of I_3 , which allow them to simultaneously terminate exploration at this last step. Equation (14) being an equality, this implies that this last step is counted in β^{I_3} . However, this step where no edges are absent being the last one of I , it cannot belong to the $T - 1$ consecutive steps with no absent edges that occur between the removal of two different edges in the same interval I . This contradicts the fact that (23) is an equality, concluding the proof for $T \geq 2$. Fig. 3 illustrates the trajectories analyzed in Case 2.

For $T = 1$, the bound obtained so far is one unit larger than the claimed bound. For the purpose of contradiction, assume that $\mathcal{T}_{exp} = 2n - 2$. This implies that all inequalities are in fact equalities except exactly one of the inequalities (14), (21), (22), and (23). In the latter case, we proved more precisely that $\beta^I = 1$ because $\beta^{I_3} = 1$. This implies that $\beta^{I_1} = 0$ in all four cases. We will now come to

369 a contradiction by proving that one of the inequalities (3), (4), (8), or (9) must be
370 strict.

371 The only way for Equation (8), resp. (9), to be an equality is that L'' , resp. R'' ,
372 traverses an edge just before turning back, that is at the step, say $t_{L'}$, resp. $t_{R'}$,
373 when L' , resp. R' , turns back. Differently speaking, Equation (8), resp. (9), being
374 an equality implies that L'' and thus L , resp. R'' and thus R , are not blocked at
375 step $t_{L'}$, resp. $t_{R'}$. Since (3) is assumed to be an equality, and because $\beta^{I_1} = 0$,
376 at least one of L'' and R'' is blocked at each time step of the interval I_1 . This
377 is in particular true for the times $t_{L'}$ and $t_{R'}$. Therefore, the two times $t_{L'}$ and
378 $t_{R'}$ must be different. Without loss of generality, assume that $t_{L'} < t_{R'}$. Let us
379 now consider the step at time $t_{R'}$. At this step, R'' and thus R as well are not
380 blocked. This implies that L must be blocked at this step because of (4) being an
381 equality. However, L'' has already turned back and does not travel with L anymore,
382 and thus cannot be blocked at this step. This would lead to (3) being strict, the
383 contradiction concluding this proof. \square

384 3.2 Lower bound

385 We now prove that the precise bound given in Section 3.1 is actually the exact worst-
386 case time complexity of the exploration problem.

387 **Theorem 2** *For every integers $n \geq 3$ and $T \geq 1$, there exists a T -interval-connected*
388 *dynamic graph based on C_n such that any agent (algorithm) needs at least*

$$389 \begin{cases} 2n - 3 & \text{if } T = 1 \\ 2n - T - 1 & \text{if } 2 \leq T \leq (n + 1)/2 \\ \lfloor \frac{3(n-1)}{2} \rfloor & \text{if } T > (n + 1)/2 \end{cases}$$

390 *time units to explore it.*

391 *Proof* For any integers $n \geq 3$, and $2 \leq T \leq \lceil (n + 1)/2 \rceil$, we define a T -interval-
392 connected dynamic graph $\mathcal{G}_{n,T}$ based on C_n . Let v_0, v_1, \dots, v_{n-1} be the vertices of C_n
393 in clockwise order. Assume that the exploration starts from v_0 at time 0. In $\mathcal{G}_{n,T}$, the
394 edge $\{v_0, v_1\}$, respectively $\{v_{T-1}, v_T\}$, is absent in the time interval $[0, n - 2T + 1)$,
395 respectively $[n - T, 2n)$. See Figure 4. Note that this dynamic graph is indeed T -
396 interval-connected.

397 Consider any agent (algorithm). We will now prove that the time it uses to ex-
398 plore $\mathcal{G}_{n,T}$ is at least $2n - T - 1$. Since the agent must explore all vertices, it must in
399 particular explore both v_{T-1} and v_T . We consider two cases.

400 **Case 1.** v_{T-1} is explored before v_T .

401 To visit v_{T-1} without going through v_T , the agent must traverse the edge $\{v_0, v_1\}$.
402 By construction, this edge is absent until time $n - 2T + 1$. Moreover, the length of
403 the path between v_0 and v_{T-1} without going through v_T is $T - 1$. Thus the agent
404 needs at least $n - T$ time units to reach v_{T-1} for the first time. Since the edge
405 $\{v_{T-1}, v_T\}$ is absent in the time interval $[n - T, 2n)$, the fastest way of reaching
406 v_T is to traverse the whole ring through v_0 , inducing $n - 1$ additional time units.
407 So in this first case, the agent needs at least $2n - T - 1$ time units to explore $\mathcal{G}_{n,T}$.

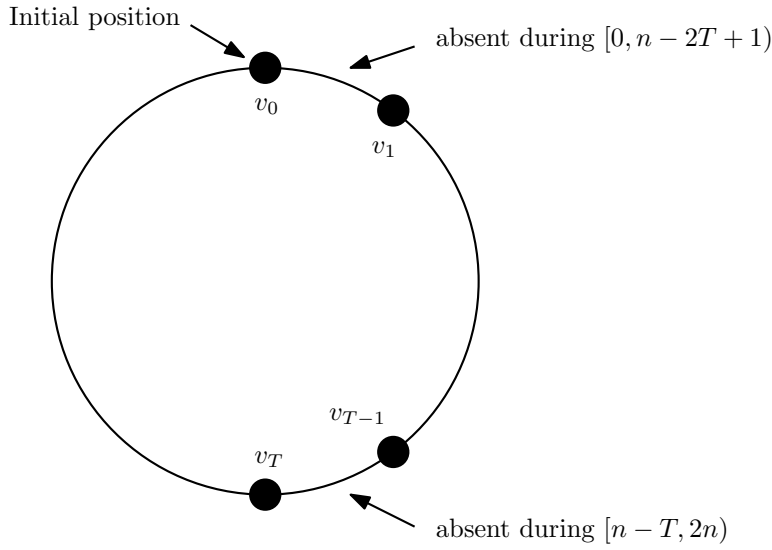


Fig. 4 T -interval-connected dynamic graph based on C_n achieving the worst-case exploration time, for $2 \leq T \leq \lceil (n+1)/2 \rceil$.

408 **Case 2.** v_T is explored before v_{T-1} .

409 To visit v_T without going through v_{T-1} , the agent must use the path v_0, v_{n-1} , up
 410 to v_T , which is of length $n - T$. When at node v_T , and since the edge $\{v_{T-1}, v_T\}$
 411 is absent in the time interval $[n - T, 2n)$, the fastest way of reaching v_{T-1} is to
 412 traverse the whole ring through v_0 , inducing $n - 1$ additional time units. Thus
 413 also in the second case, the agent needs at least $2n - T - 1$ time units to explore
 414 $\mathcal{G}_{n,T}$.

415 This proves the theorem for values of T in $[2, \lceil (n+1)/2 \rceil]$. In fact, this also proves
 416 the theorem for $T = 1$ because $\mathcal{G}_{n,2}$ is obviously also 1-interval-connected, and thus
 417 the bound $2n-3$ proved for $T = 2$ is also valid for $T = 1$. Besides, note that only
 418 one edge is ever removed in $\mathcal{G}_{n, \lceil (n+1)/2 \rceil}$. This dynamic graph is therefore T -interval-
 419 connected for any T , and thus the theorem is also proved for values of T larger than
 420 $(n+1)/2$. \square

421 **4 The agent does not know the dynamics of the graph**

422 In this section, we assume that the agent does not know the dynamics of the graph,
 423 i.e., it does not know the times of appearance and disappearance of the edges. As
 424 explained in the introduction, we assume here the δ -recurrence property, for a given
 425 $\delta \geq 1$, in order for the problem to be solvable in bounded time.

426 4.1 Upper bound

427 We first prove that there exists a very simple algorithm that is able to explore all the
428 δ -recurrent T -interval-connected dynamic graphs based on the ring. This algorithm
429 consists in moving as much and as soon as possible in a fixed arbitrary direction, see
430 Algorithm 1.

Algorithm 1 STUBBORN-TRAVERSAL(dir)

Input: a direction dir
for each time step **do**
 if the edge in the dir direction is present **then**
 traverse it
 else
 wait
 end if
end for

431 **Theorem 3** For every integers $n \geq 3$, $T \geq 1$ and $\delta \geq 1$, and for any direction dir , Al-
432 gorithm STUBBORN-TRAVERSAL(dir) explores any δ -recurrent T -interval-con-
433 nected dynamic graph based on C_n in time at most

$$434 \quad n - 1 + \left\lceil \frac{n - 1}{\max\{1, T - 1\}} \right\rceil (\delta - 1).$$

435 *Proof* Fix an arbitrary direction dir and let us analyze the algorithm STUBBORN-
436 TRAVERSAL(dir). Note first that it will complete exploration after traversing exactly
437 $n - 1$ edges. To bound its exploration time, it thus remains to bound the number of
438 time steps when the agent cannot move.

439 Since the dynamic graph is δ -recurrent, an edge cannot be absent for more than
440 $\delta - 1$ consecutive time steps. Furthermore, since the dynamic graph is T -interval-
441 connected, two time steps in which two different edges are absent must be separated
442 by at least $T - 1$ time steps in which all edges are present. Therefore, the agent can
443 traverse at least $\max\{1, T - 1\}$ edges between two consecutive blocks at different
444 nodes. To summarize, the agent can be blocked at most $\left\lceil \frac{n-1}{\max\{1, T-1\}} \right\rceil$ times during at
445 most $\delta - 1$ time steps.

446 Putting everything together, the agent will perform edge traversals for $n - 1$ time
447 steps and will wait for at most $\left\lceil \frac{n-1}{\max\{1, T-1\}} \right\rceil (\delta - 1)$ time steps, which gives the
448 claimed bound. \square

449 4.2 Lower bound

450 It turns out that the simple and natural Algorithm 1, described and analyzed in Sec-
451 tion 4.1, is almost optimal, up to an additive term proportional to δ .

452 **Theorem 4** For every integers $n \geq 3$, $T \geq 1$, and $\delta \geq 1$, and for every agent (algo-
 453 rithm), there exists a δ -recurrent T -interval-connected dynamic graph based on C_n
 454 such that this agent needs at least

$$455 \quad n - 1 + \left\lfloor \frac{n - 3}{\max\{1, T - 1\}} \right\rfloor (\delta - 1)$$

456 time units to explore it.

457 This result holds even if the agent knows n , T and δ .

458 *Proof* Let $n \geq 3$, $T \geq 1$, and $\delta \geq 1$. Fix an arbitrary agent (algorithm) A . We construct
 459 as follows the δ -recurrent T -interval-connected dynamic graph $\mathcal{G}_{n,T,\delta}(A)$ based on C_n
 460 that this agent will fail to explore in less than the claimed bound.

461 Let v_0, v_1, \dots, v_{n-1} be the vertices of C_n in clockwise order. Assume that the agent
 462 starts exploration from v_0 at time 0. For any integer $1 \leq i \leq n - 1$, if the node v_i is
 463 explored by going from v_0 in the counter-clockwise direction, then node v_i is den-
 464 noted v_{i-n} . Finally, let $\tilde{T} = \max\{1, T - 1\}$.

465 In the dynamic graph $\mathcal{G}_{n,T,\delta}(A)$, only the edges $\{v_{\tilde{T}+1}, v_{\tilde{T}+2}\}$, $\{v_{2\tilde{T}+1}, v_{2\tilde{T}+2}\}$,
 466 and so on, and $\{v_0, v_{-1}\}$, $\{v_{-\tilde{T}}, v_{-\tilde{T}-1}\}$, $\{v_{-2\tilde{T}}, v_{-2\tilde{T}-1}\}$, and so on, may be absent.
 467 The actual times of appearance and disappearance of these edges depend on the algo-
 468 rithm A . For any integer $i \geq 0$, each time the agent arrives at node $v_{-i\tilde{T}}$ in the counter-
 469 clockwise direction, the edge $\{v_{-i\tilde{T}}, v_{-i\tilde{T}-1}\}$ is removed until either the δ -recurrence
 470 forces the edge to reappear or the agent leaves the node $v_{-i\tilde{T}}$ to go on $v_{-i\tilde{T}+1}$. Simi-
 471 larly, for any integer $i \geq 1$, each time the agent arrives at node $v_{i\tilde{T}+1}$ in the clockwise
 472 direction, the edge $\{v_{i\tilde{T}+1}, v_{i\tilde{T}+2}\}$ is removed until either the δ -recurrence forces the
 473 edge to reappear or the agent leaves the node $v_{i\tilde{T}+1}$ to go on $v_{i\tilde{T}}$. Note that between
 474 two time steps with two different absent edges, there are at least $T - 1$ time steps for
 475 which no edges are absent. The dynamic graph is therefore T -interval-connected. It
 476 is also δ -recurrent by construction.

477 By definition of the dynamics of the graph, the agent needs to wait $\delta - 1$ time units
 478 to go from $v_{-i\tilde{T}}$ to $v_{-i\tilde{T}-1}$, for $i \geq 0$, or to go from $v_{i\tilde{T}+1}$ to $v_{i\tilde{T}+2}$, for $i \geq 1$. Also,
 479 except near the origin in the clockwise direction, the agent cannot traverse more than
 480 \tilde{T} new edges before having to traverse such a blocking edge. Hence, to explore all the
 481 vertices, the agent needs to perform at least $\left\lfloor \frac{n-3}{\tilde{T}} \right\rfloor$ such traversals. This lower bound
 482 is obtained in the case when the agent explores the ring by always going clockwise
 483 (starting in the counter-clockwise direction and/or changing direction during the ex-
 484 ploration do not help). The waiting time of the agent is thus at least $\left\lfloor \frac{n-3}{\tilde{T}} \right\rfloor (\delta - 1)$.
 485 Since the agent needs also at least $n - 1$ time units to traverse enough edges so that
 486 all vertices are explored, we obtain the claimed bound. \square

487 5 Conclusion

488 We studied in this paper the problem of exploration of the T -interval-connected dy-
 489 namic graphs based on the ring in two scenarios, when the agent is specific to the
 490 dynamic graph, and when the agent does not know the dynamics of the graph. The

491 next objective is obviously to extend these results to larger families of underlying
 492 graphs. Unfortunately, this problem is much more difficult than it seems: proving that
 493 any dynamic graph based on a tree of cycles (a cactus) can be explored in time $O(n)$
 494 is already a challenging open problem.

495 References

- 496 1. E. Aaron, D. Krizanc, and E. Meyerson. DMVP: Foremost Waypoint Coverage of Time-
 497 Varying Graphs. In *40th International Workshop on Graph-Theoretic Concepts in Computer
 498 Science (WG)*, LNCS 8147, pages 29–41, 2014.
- 499 2. E. Aaron, D. Krizanc, and E. Meyerson. Multi-Robot Foremost Coverage of Time-Varying
 500 Graphs. In *10th International Symposium on Algorithms and Experiments for Sensor Systems,
 501 Wireless Networks and Distributed Robotics (ALGOSENSORS)*, LNCS 8847, pages 22–38,
 502 2014.
- 503 3. M. Bournat, A. K. Datta, and S. Dubois. Self-Stabilizing Robots in Highly Dynamic Environ-
 504 ments. In *18th International Symposium on Stabilization, Safety, and Security of Distributed
 505 Systems (SSS 2016)*, LNCS 10083, pages 54–69, 2016.
- 506 4. A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro. Time-varying graphs and dy-
 507 namic networks. *International Journal of Parallel, Emergent and Distributed Systems*, volume
 508 27(5), 2012.
- 509 5. G. A. Di Luna, S. Dobrev, P. Flocchini, and N. Santoro. Live Exploration of Dynamic Rings.
 510 In *IEEE 36th International Conference on Distributed Computing Systems (ICDCS)*, pages
 511 570–579, 2016.
- 512 6. T. Erlebach, M. Hoffmann, and F. Kammer. On Temporal Graph Exploration. In *42nd Interna-
 513 tional Colloquium on Automata, Languages, and Programming (ICALP)*, LNCS 9134, pages
 514 444–455, 2015.
- 515 7. A. Ferreira, Building a reference combinatorial model for MANETs. *Network*, IEEE, volume
 516 18(5), pages 24–29, 2004.
- 517 8. P. Flocchini, M. Kellelt, P. C. Mason, and N. Santoro. Searching for black holes in subways.
 518 *Theory of Computing Systems*, 50(1), pages 158–184, 2012.
- 519 9. P. Flocchini, M. Kellelt, P. C. Mason, and N. Santoro. Finding Good Coffee in Paris. In *6th
 520 International Conference on Fun with Algorithms (FUN)*, LNCS 7288, pages 154–165, 2012.
- 521 10. P. Flocchini, B. Mans, and N. Santoro. On the exploration of time-varying networks. *Theoret-
 522 ical Computer science*, volume 469, pages 53–68, 2013.
- 523 11. D. Ilcinkas, R. Klasing, and A. M. Wade. Exploration of Constantly Connected Dynamic
 524 Graphs Based on Cactuses. In *21st International Colloquium on Structural Information and
 525 Communication Complexity (SIROCCO)*, LNCS 8576, pages 250–262, 2014.
- 526 12. D. Ilcinkas and A. M. Wade. On the Power of Waiting when Exploring Public Transportation
 527 Systems. In *15th International Conference On Principles Of Distributed Systems (OPODIS)*,
 528 LNCS 7109, pages 451–464, 2011.
- 529 13. D. Ilcinkas and A. M. Wade. Exploration of the T-Interval-Connected Dynamic Graphs: the
 530 Case of the Ring. In *20th International Colloquium on Structural Information and Communi-
 531 cation Complexity (SIROCCO)*, LNCS 8179, pages 13–23, 2013.
- 532 14. F. Kuhn, N.A. Lynch, and R. Oshman, Distributed computation in dynamic networks. In *42nd
 533 ACM symposium on Theory of computing (STOC)*, pages 513–522, 2010.
- 534 15. F. Kuhn and R. Oshman, Dynamic networks: models and algorithms. *ACM SIGACT News*,
 535 volume 42(1), pages 82–96, 2011.
- 536 16. O. Michail. An Introduction to Temporal Graphs: An Algorithmic Perspective. *Internet Math-
 537 ematics*, volume 12(4), pages 239–280, 2016.
- 538 17. O. Michail, I. Chatzigiannakis, and P. G. Spirakis. Causality, influence, and computation in
 539 possibly disconnected synchronous dynamic networks. *Journal of Parallel and Distributed
 540 Computing*, volume 74(1), pages 2016–2026, 2014.
- 541 18. O. Michail and P. G. Spirakis. Traveling salesman problems in temporal graphs. *Theoretical
 542 Computer science*, volume 634, pages 1–23, 2016.
- 543 19. C. E. Shannon, Presentation of a maze-solving machine. 8th Conf. of the Josiah Macy Jr.
 544 Found. (Cybernetics), pages 173–180, 1951.