Brief Announcement: Time-Travel Planning with Tenet Turnstiles

Thibaut Blanc
ENS Lyon, France

Quentin Bramas
Strasbourg University, CNRS, ICUBE, France

Jean-Romain Luttringer
Strasbourg University, CNRS, ICUBE, France

Sébastien Tixeuil
Sorbonne University, CNRS, LIP6, IUF, France

Abstract

We study routing in dynamic graphs when an agent may use backward time travel (BTT) devices. Minimizing *delay* (arrival time minus departure time) is the primary objective; the number of time inversions is the secondary cost. Building on the framework of Bramas et al., we introduce two space-time online settings – *ST-online-easy* and *ST-online-hard* – and analyze the *Tenet* model, where BTT is performed by entering a turnstile that reverses the direction of time flow. We obtain a polynomial-time offline algorithm, tight competitive ratios for the T-online and S-online settings, a tight quadratic competitive ratio for ST-online-easy, and we prove that no finite competitive ratio exists for ST-online-hard.

2012 ACM Subject Classification Theory of computation → Dynamic graph algorithms

Keywords and phrases dynamic graphs, time travel, online algorithms

Digital Object Identifier 10.4230/LIPIcs.SAND.2026.22

1 Introduction

Dynamic (temporal) graphs model systems whose connectivity changes over time. Most algorithmic work assumes that time only moves forward; an agent may wait but cannot revisit the past. We study traversal in dynamic graphs when an agent has access to a *backward time travel* (BTT) device. The goal is to travel from a source to a destination minimizing *delay* (final time minus start time, which can be 0 thanks to BTT), and then the BTT cost.

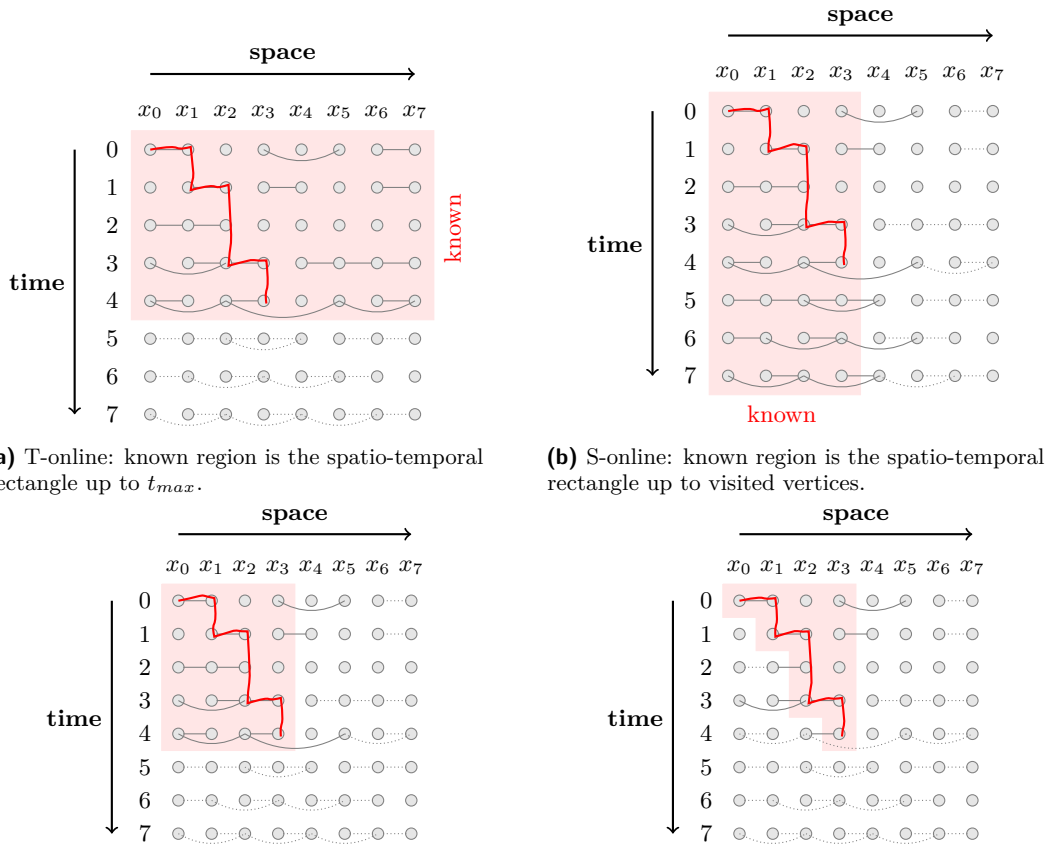
Bramas et al. [1, 2, 3] initiated the formal study of space-time travel in dynamic networks. They defined offline (full knowledge), T-online (the agent learns all edges up to the current time at each step), and S-online (the agent learns all edges incident to already-visited vertices across all times) information settings. They provided polynomial-time offline algorithms, a T-online algorithm achieving optimal competitive ratio with cost at most twice optimal, and strong lower bounds for S-online algorithms, complemented by matching upper bounds for several cost functions.

Our contributions

We build on this framework along two axes. First, we introduce two *space-time online* variants, *ST-online-easy* and *ST-online-hard*, which capture two levels of information disclosure when space and time are discovered simultaneously. Second, we introduce and fully analyze the *Tenet* model [4], where BTT cost equals the number of time-flow inversions performed. We obtain: (i) an $O(m \log n)$ offline algorithm; (ii) a tight $\Theta(n)$ competitive ratio for T-online; (iii) linear upper and lower bounds for S-online; (iv) a tight $\Theta(n^2)$ competitive ratio for ST-online-easy; and (v) impossibility of any finite competitive ratio for ST-online-hard.

5th Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2026).
Editors: George B. Mertzios and Andréa W. Richa; Article No. 22; pp. 22:1–22:6

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany



(a) T-online: known region is the spatio-temporal rectangle up to t_{max}.

(b) S-online: known region is the spatio-temporal rectangle up to visited vertices.

(c) ST-online-easy: known region is the spatio-temporal rectangle up to t_{max} and visited vertices.

(d) ST-online-hard: known region follows the exact space-time footprint of the travel prefix.

Figure 1 Information states for the four new online settings (red line = travel prefix; dotted edges are not yet revealed).

2 Model and Problem

▶ **Definition 1** (Dynamic Graph). *A* dynamic graph *is a pair* $G = (V, (E_t)_{t\in\mathbb{N}})$ *where* V *is a finite vertex set and* $E_t \subseteq V \times V$ *is the edge set at time* t. *The* footprint *of* G *is* $\mathcal{F}(G) = (V, \bigcup_{t\in\mathbb{N}} E_t)$.

▶ **Definition 2** (Space-Time Travel). *A* space-time travel *of length* k *is a sequence* $T = ((u_0, t_0), \ldots, (u_k, t_k))$ *such that for each* i: $u_i \in V$, $t_i \in \mathbb{N}$; *and if* $u_i \neq u_{i+1}$, *then* $t_i = t_{i+1}$ *and* $(u_i, u_{i+1}) \in E_{t_i}$. *The* delay *of* T *is* $t_k - t_0$.

▶ **Definition 3** (ODOC). *The* optimal-delay optimal-cost *(ODOC) problem asks for a travel from* $(src, 0)$ *to* dst *that (i) minimizes the delay, and (ii) among all minimum-delay travels, minimizes the model-dependent cost. Because BTT is always available, any travel to a reachable destination can achieve delay* 0, *so the ODOC problem reduces to finding a minimum-cost travel from* $(src, 0)$ *to* $(dst, 0)$. *An online algorithm is* ρ-competitive *if on every instance where an ODOC travel exists, its cost is at most* ρ *times the optimal cost.*

Online information settings

We define four settings specifying what the agent knows during execution.
T-online: At global time t, the agent knows $E_{t'}$ for all $t' \leq t$.
S-online: The agent knows edges in E_t incident to already-visited vertices, for all t.
ST-Easy: Let t_{max} be the latest time reached so far. The agent knows edges in E_t incident to visited vertices, for all $t \leq t_{max}$.
ST-online-hard: The agent knows $(u, v) \in E_t$ if it has visited (u, t') or (v, t') for some $t' \geq t$.

The four settings are illustrated in Figure 1.

3 The Tenet Model

An agent in the Tenet model can: (i) traverse an edge; (ii) wait one step; or (iii) enter a *turnstile* to invert the direction of time flow (initially forward). The *direction of time* at step i of a travel $T = ((u_0, t_0), \ldots, (u_k, t_k))$, written $\delta(T, i)$, is the sign of $t_{j+1} - t_j$ for the largest $j < i$ such that $t_j \neq t_{j+1}$ (defaulting to $+1$).

▶ **Definition 4** (Cost). $\zeta(T) = \#\{i \in [1; k-1] \mid \delta(T, i) \neq \delta(T, i+1)\}$, *i.e., the number of time inversions.*

Figure 2 shows three travels: the red travel uses 0 inversions, the blue uses 1 (which is optimal for delay 0), and the green uses 3.

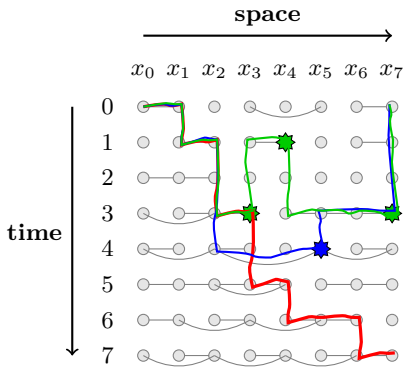


Figure 2 Three feasible Tenet travels. Stars mark time inversions: red = 0, blue = 1 (optimal), green = 3.

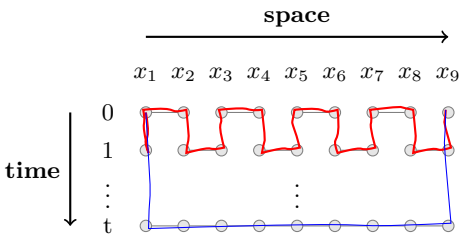


Figure 3 Adversarial instance G_8^t used in the T-online lower bound.

4 Main Results

Offline Setting

▶ **Theorem 5.** *The ODOC problem in the Tenet model can be solved in time $O(m \log n)$, where $m = |\mathcal{F}(G)|$.*

Algorithm and key ideas

We define a total order on pairs $(d, t) \in \mathbb{N} \times \mathbb{N}$: $(d, t) < (d', t')$ iff $d < d'$, or $d = d'$ and (d even and $t < t'$), or (d odd and $t > t'$). Intuitively, after an even number of inversions the agent moves forward (minimizing arrival time), while after an odd number it moves backward (maximizing arrival time, so it can reach the destination earlier by then going forward).

The offline algorithm is a Dijkstra-like scan over vertices. Each vertex u carries a key $(d[u], tmp[u])$: the minimum number of inversions to reach u and the associated optimal time under the order above. A priority queue extracts candidates in this order; for each extracted (u, d, t), the algorithm scans neighbors v and inserts the best reachable (v, d', t') into the queue. Correctness follows from showing that the order is consistent with optimal substructure of Tenet travels: any prefix of an optimal travel is itself optimal. Since each vertex is finalized at most once and the queue size is bounded by n, the total running time is $O(m \log n)$.

T-Online Setting

▶ **Theorem 6.** *No T-online algorithm has competitive ratio $< n$. The strategy that at each step follows an ODOC travel if one is fully determined by current knowledge, and waits otherwise, achieves competitive ratio exactly n.*

Key ideas

Lower bound. We use an indistinguishability argument. For odd n and large t, consider the graph G_n^t with $V = \{1, \ldots, n\}$, alternating edges at times 0 and 1, and a full path at time t as illustrated in Figure 3. Instances G_n^∞ and $G_n^{t_{max}+1}$ (where t_{max} is the furthest time reached by the algorithm) are indistinguishable, yet their optimal costs differ by a factor of n: the algorithm must pay at least n inversions to traverse G_n^∞, while OPT pays 1 on $G_n^{t_{max}+1}$ using the late full path.

Upper bound. Any simple travel uses at most $n-1$ edges and thus at most $n-1$ inversions. Since our strategy may initially wait until the ODOC travel is identifiable, it requires at most one extra inversion, giving a total cost bounded by n.

S-Online Setting

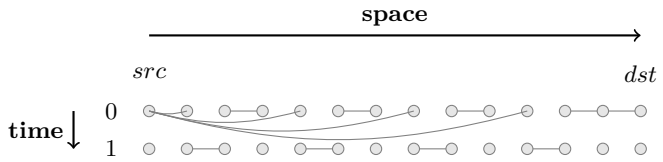


Figure 4 Adversarial graph G_4 used in the S-online lower bound.

▶ **Theorem 7.** *No S-online algorithm has competitive ratio $< \frac{4}{3}n - \frac{17}{3}$. The DFS-based exploration algorithm achieves competitive ratio $2n - 4$.*

Key ideas

Lower bound. For $n = 3k+2$, consider a graph G_k with k gadgets hanging off the source: each gadget has a spine of length 3 with its middle edge available only at time 1, as illustrated in Figure 4. Under S-online, the agent cannot distinguish which gadget leads to the destination before exploring all of them, forcing linear cost ($\frac{4}{3}n - \frac{17}{3}$ inversions) while OPT uses one inversion.

Upper bound. The DFS strategy maintains a spanning tree of explored vertices via parent pointers, prioritizing the destination when available. Each tree edge is traversed at most twice (once down, once back), except the last two (no backtrack from destination, and the penultimate is always explored first). Each traversal may require one inversion; including a possible final inversion to return to time 0 gives at most $2(n-1)-2 = 2n-4$ inversions.

ST-Online-Easy Setting

▶ **Theorem 8.** *There exists an ST-online-easy-Online algorithm with competitive ratio* $\frac{(2n-1)^2}{12} + 1$*, and no ST-online-easy algorithm achieves a better ratio.*

Key ideas

Call a vertex a *cliff* if its predecessor and one of its successors in the footprint tree have an earlier time of occurrence. Cliffs force the agent to perform extra inversions: visiting a cliff-hanger (a descendant reachable from the cliff backward in time) and returning costs 2 inversions.

The *ST-online-easy algorithm* greedily follows a cost-optimal travel to an unvisited vertex at each step, exploring cliff-hangers as they appear. If k is the maximum number of cliffs on any root-to-leaf path, the total cost is bounded by $f(k) = -3k^2 + (2n-1)k + 1$. This concave quadratic is maximized at $k = (2n-1)/6$, yielding $\frac{(2n-1)^2}{12} + 1$.

Tightness. An adaptive adversary reveals edges incrementally: each time the algorithm commits to exploration, the adversary places the next useful edge further in the future, forcing the algorithm to accumulate cliffs. This construction matches the bound $f(k)$ for every integer $k \in [2, (n-4)/2]$, confirming optimality.

ST-Online-Hard Setting

▶ **Theorem 9.** *No ST-online-hard algorithm admits a finite competitive ratio for* $n \geq 3$*.*

Key idea

Fix any algorithm A with claimed ratio ρ. Start with edge $(src, x) \in E_0$. The agent must explore x (otherwise we add a direct edge from x to dst invisible to the agent). After the agent has oscillated between src and x for ρ rounds, costing 2ρ inversions, we place the edge $(src, dst) \in E_{t+1}$ where t is the current furthest time reached. Then OPT pays 1 and the algorithm pays more than ρ, a contradiction.

5 Conclusion

We analyzed the Tenet BTT model across five information settings. The offline setting admits $O(m \log n)$ computation and the landscape of the online settings is sharply stratified: T-online achieves $\Theta(n)$; S-online achieves $\Theta(n)$ (with a remaining constant-factor gap); ST-online-easy exhibits tight $\Theta(n^2)$ behavior driven by the cliff parameter; and ST-online-hard admits no competitive ratio.

Open directions include characterizing the minimal additional information that restores competitiveness in ST-online-hard, closing the constant-factor gap in the S-online bounds, extending these models to multiple cooperating agents, and studying robustness under learning-augmented temporal predictions.

References

1 Quentin Bramas, Jean-Romain Luttringer, and Sébastien Tixeuil. Offline constrained backward time travel planning. In Shlomi Dolev and Baruch Schieber, editors, *Stabilization, Safety, and Security of Distributed Systems - 25th International Symposium, SSS 2023, Jersey City, NJ, USA, October 2-4, 2023, Proceedings*, volume 14310 of *Lecture Notes in Computer Science*, pages 466–480. Springer, 2023. doi:10.1007/978-3-031-44274-2_35.

2 Quentin Bramas, Jean-Romain Luttringer, and Sébastien Tixeuil. Online space-time travel planning in dynamic graphs. In Arnaud Casteigts and Fabian Kuhn, editors, *3rd Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2024)*, volume 292 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 7:1–7:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.SAND.2024.7.

3 Quentin Bramas, Jean-Romain Luttringer, and Sébastien Tixeuil. On time-travel planning in dynamic graphs. *Theor. Comput. Sci.*, 1054:115501, 2025. doi:10.1016/J.TCS.2025.115501.

4 Christopher Nolan. Tenet, 2020.