Weighted and Unweighted Online Connectivity Augmentation via Geometric Hitting Set

Daniel Hathcock* Aditya Subramanian† Seeun William Umboh‡ Michael Zlatin§

arXiv:2611.08570v1 [cs.DS] 5 Nov 2026

Abstract

We study the Online Connectivity Augmentation Problem (OCAP) with cut arrivals. In this problem, we are given a k-edge-connected graph $G = (V, E)$ with $|V| = n$ and a set of links $L \subseteq \binom{V}{2}$. In each iteration, a minimum cut arrives to be augmented, and we seek to maintain a smallest set of links crossing all such cuts. This problem is a special case of online hitting set in which the ground set is L and a sequence of sets to be hit arrive from $\mathcal{C} = \{\delta_L(C) : |\delta_E(C)| = k\}$. As such, OCAP admits a $O(\log^2 n)$-competitive algorithm using the classical algorithm for Online Set Cover of Alon, Awerbuch, Azar, Buchbinder, and Naor. If k is odd, then OCAP reduces to the case where G is a tree. This case was studied by Naor, Umboh, and Williamson, who gave an optimal $O(\log n)$-competitive algorithm, which even holds in weighted setting. Recently, Garg and Subramanian gave an $O(\log n)$-competitive algorithm for unweighted OCAP which holds in the popular node-arrival model, but not for cut-arrivals. No better than $O(\log^2 n)$-competitive algorithm is known for weighted OCAP. In this paper, we provide a nearly optimal $O(\log n \log\log n)$-competitive algorithm for weighted OCAP under cut-arrivals and an optimal $O(\log n)$-competitive algorithm for unweighted OCAP with cut-arrivals. We also give an $O(1)$ algorithm for Online Cycle Augmentation under node-arrivals. Since no $o(\log n)$-competitive algorithm is possible on a cycle with cut-arrivals, this establishes a separation between the best competitive ratio achievable in the two arrival models.

1 Introduction

The defining goal of the field of resilient network design is to construct a network which is robust to failures of its elements. Often, the network designer does not have the luxury of designing their network from scratch, rather there is an existing legacy network in place which we seek to augment by adding additional edges, called links, in the most efficient way. These problems are called Network Augmentation Problems, and have motivated a substantial amount of work in approximation algorithms, combinatorial optimization and polyhedral theory.

A common measure of the resiliency of a network to edge failures is its **edge-connectivity**. Given a graph $G = (V, E)$, a pair of vertices $u, v \in V$ are k-edge-connected if the size of any cut separating u and v is at least k. Similarly, G is (globally) k-edge-connected if every pair of vertices is. This motivates the following problem, called the Weighted Connectivity Augmentation Problem, or WCAP.

*GE Vernova, Advanced Research Center, Niskayuna, NY, USA
†University of Waterloo, ON, Canada
‡University of Melbourne, VIC, Australia
§Department of Computer Science, Pomona College, CA, USA

Problem 1.1 (WCAP)**.** *Given a k-edge-connected graph $G = (V, E)$ and a set of links $L \subseteq \binom{V}{2}$ with nonnegative costs $c : L \to \mathbb{R}_{\geq 0}$, find a minimum-cost subset $F \subseteq L$ such that $(V, E \cup F)$ is $(k+1)$-edge-connected.*

It is useful to observe that this problem can be formulated as a special case of hitting set, where a feasible solution corresponds exactly to a set of links $F \subseteq L$ hitting all k-cuts of G. Here, a link ℓ **hits** a cut $C \subseteq V$ if it crosses that cut, i.e. if $\ell \in \delta_L(C)$. This observation allows us to replace the graph G by another, as long as the structure of its minimum cuts remains the same. In particular, Dinitz, Karzonov, and Lomonosov [DKL76] showed that for any k-edge-connected graph G, there exists an (efficiently computable) cactus graph H for which there is a one-to-one correspondence between their minimum cuts. Furthermore, if k is odd, then H can be chosen to be a tree. Thus, the WCAP problem reduces to the case where G is a cactus (for all k) and where G is a tree (if k is odd) these problems are called WCacAP and WTAP respectively.

There is much work on the offline variants of CAP and TAP. However, there has also been a strong interest in Connectivity Augmentation Problems in other models of computation, including the online and streaming settings [GKR12; NUW22; GS26; Jin+24; Che+25]. We focus on the Online Weighted Connectivity Augmentation Problem (OWCAP) with cut-arrivals, defined as follows.

Problem 1.2 (OWCAP; cut-arrivals)**.** *Given a k-edge-connected graph $G = (V, E)$, and a set of links $L \subseteq \binom{V}{2}$ on V with non-negative costs $c : L \to \mathbb{R}_{\geq 0}$, a sequence of k-cuts of G arrive online. Upon each arrival, we must irrevocably purchase links so that every k-cut that has arrived so far is crossed by a link we have purchased. The objective is to minimize the total cost of purchased links.*

There is also the problem of Online Connectivity Augmentation with node-arrivals in which pairs of vertices $(u_t, v_t) \in \binom{V}{2}$, referred to as *terminals*, arrive online. Upon each arrival, we must irrevocably purchase links so that every requested pair is $(k+1)$-edge-connected in the augmented graph. Notice that each terminal-pair request corresponds to covering a collection of cut requests—namely those k-cuts which separate u from v in G. Hence, any algorithm for OCAP in the cut-arrival model can be used to solve OCAP with node-arrivals by processing all k-cuts that separate pairs of requested vertices. However, the converse is not true: an algorithm which handles node-arrivals may not apply to the cut-arrival setting.

As a special case of Online Hitting Set, the OWCAP with cut-arrivals problem readily admits an $O(\log^2 n)$-competitive algorithm by applying the classical algorithm of [Alo+09]. In particular, [Alo+09] provides an algorithm for weighted online hitting set which is $O(\log m \log N)$-competitive, where m is the number of sets to be hit and N is the size of the ground set of elements. In OCAP with cut arrivals, our ground set is L and the sets to be hit arrive from $\mathcal{C} = \{\delta_L(C) : |\delta_E(C)| = k\}$, each set corresponding to a min-cut of G. It follows from [DKL76] that the number of minimum cuts in any graph with n vertices is at most n^2, hence, we have $N = |L| \leq n^2$ and $m \leq n^2$ yielding an $O(\log^2 n)$ competitive ratio overall.

Following this, subsequent work has obtained improved algorithms in various important special cases. For example, Naor, Umboh and Williamson [NUW22] considered the special case of Online WTAP, in which G is a tree. They essentially resolve this case, giving a deterministic $O(\log n)$-competitive algorithm for Online WTAP, matching a lower bound showing that no $o(\log n)$-competitive ratio can be achieved [GKR12]. More recently, Garg and Subramanian [GS26] considered the general case of Online CAP, giving an optimal $O(\log n)$-competitive algorithm for

unweighted CAP in the node-arrival model. However, their techniques do not extend to cut arrivals. Interestingly, as observed in [NUW22], the node-arrival model and the cut-arrival model are equivalent for Online WTAP. No better than $O(\log^2 n)$ is known for weighted OCAP in either arrival model. Thus, our work is motivated by two central questions:

> **Question 1:** Is there a separation between the best achievable competitive ratios in the node-arrival and cut-arrival models of Online Connectivity Augmentation?
>
> **Question 2:** Can we obtain an optimal $O(\log n)$-competitive algorithm for weighted Online Connectivity Augmentation?

More generally, for which special cases of Online Hitting Set can we exploit the underlying structure to obtain improved competitive ratios? In the next section, we describe how we address each of these questions.

1.1 Our Results

We essentially resolve Question 1 and resolve Question 2 up to a factor of $\log \log n$. Our first result is a nearly optimal competitive ratio for OWCAP with cut arrivals.

Theorem 1.3. *There is a randomized $O(\log n \log \log n)$-competitive algorithm for OWCAP with cut arrivals.*

Since the cut arrivals model subsumes the node arrival model, this also implies a $O(\log n \log \log n)$-competitive result for node arrivals, improving the earlier $O(\log^2 n)$ bound.

We also obtain an optimal $O(\log n)$-competitive algorithm for unweighted OCAP with cut arrivals. This generalizes the recent result of Garg and Subramanian [GS26] who achieved a matching result only for node arrivals. Furthermore, our approach is distinct, and hence also provides an alternate algorithm for the node arrival case, with matching guarantees.

Theorem 1.4. *There is an $O(\log n)$-competitive algorithm for unweighted OCAP under cut arrivals.*

Thus, Theorem 1.4 gives a partial answer to Question 1, namely that there is no difference in complexity between node-arrival and cut-arrival model in the unweighted case of OCAP for general graphs. However, we show that for unweighted OCAP where G is a cycle, which we call OCycAP, there is a separation between the two models.

Theorem 1.5. *There is a $O(1)$-competitive algorithm for unweighted OCycAP under node-arrivals. However, there is no $o(\log n)$-competitive algorithm for unweighted OCycAP under cut-arrivals, even fractionally.*

Along the way to proving Theorem 1.3 and Theorem 1.4 we establish a surprising connection between connectivity augmentation problems and a particular case of 2D geometric hitting set, which we call Online TL-Rectangle hitting set. We elaborate on this further in the later sections, but we state the result here, as it is interesting in its own right.

Theorem 1.6. *There is a randomized $O(\log |P| \log \log |P|)$ competitive algorithm for online hitting set with P points where each arriving shape is the union of a leftless rectangle and a topless rectangle.*

As far as the authors can tell, Theorem 1.10 is the only result for any non-trivial special case of general weighted geometric online hitting set which beats the doubly logarithmic factor obtainable via [Alo+09] for general weighted hitting set.

1.2 Our Techniques

Our approach to Online Connectivity Augmentation with cut-arrivals consists of three phases: reduction to a cycle, then reduction to a particular case of geometric hitting set we call TL-Rectangle Hitting Set, then solving TL-Rectangle Hitting Set with the appropriate bounds.

1.2.1 From arbitrary graphs to cycles

First we reduce the general OCAP problem (both weighted and unweighted) to the same problem on a cycle. This reduction is a standard tool when dealing with Connectivity Augmentation Problems in the weighted case. Essentially, one can introduce links of weight 0 to "unfold" any cactus into a cycle, where the minimum cuts in the cycle which are not crossed by 0 cost links are in one-to-one correspondence with the minimum cuts of the original cactus (c.f. Theorem 4 in [Gál+21], Lemma 2.2 in [TZ23]).

Since this introduces links of cost 0, the reduction does not go through for unweighted CAP. However, we show that in the online setting with cut arrivals, we may always assume that the graph to be augmented is a cycle.

Theorem 1.7. *If there is an α-competitive algorithm for OCycAP with cut-arrivals, then there is an α-competitive algorithm for OCAP with cut-arrivals. Similarly, if there is an α-competitive algorithm for OWCycAP with cut-arrivals, then there is an α-competitive algorithm for OWCAP with cut arrivals.*

To prove Theorem 1.7, we utilize the Theorem on Circumference Disposition of Dinitz, Karzonov, and Lomonosov [DKL76]. This result states that the vertices of any graph can be arranged in a cyclic order so that any minimum cut in G splits the vertices into two arcs of this cycle. In other words, for any graph $G = (V, E)$ there exists a cycle $H = (V, E')$ such that if $S \subseteq V$ is a min-cut in G, then S is a min-cut H. While there may be additional min-cuts in H which do not correspond to min-cuts in G, in the cut-arrival model, these cuts simply never request to be augmented.

We emphasize that Theorem 1.7 *only* applies to the cut-arrival model. It is not true for node-arrivals, as Theorem 1.5 demonstrates. In particular, there is a constant competitive algorithm for unweighted OCycAP in the node-arrival model, but no $o(\log n)$-competitive algorithm exists for unweighted OCAP in the node-arrival model.

1.2.2 From cycle-cuts to TL-rectangles

Our second step is to cast OCycAP as a 2D geometric hitting set problem we call TL-rectangle hitting set. This problem is an Online 2D geometric hitting set problem in which each arriving shape is the union of two particularly oriented 3-sided rectangles - one is topless and the other is leftless. These shapes are defined as follows:

Definition 1.8. *A topless rectangle is a region $R^T = \{(x, y) \in \mathbb{R}^2 : a \leq x \leq b, c \leq y\}$ for some $a, b, c \in \mathbb{R}$. Equivalently, $R^T = [a, b] \times [c, \infty)$. A leftless rectangle is a region $R^L = \{(x, y) \in \mathbb{R}^2 : x \leq a, b \leq y \leq c\} = (-\infty, a] \times [b, c]$ for some $a, b, c \in \mathbb{R}$.*

Then, the Online TL-rectangle hitting set problem is the standard geometric hitting set problem in 2D where each arriving shape is the union of a topless and leftless rectangle.

Problem 1.9 (Online TL-Rectangle Hitting Set)**.** *We are given a finite point set $P \subseteq \mathbb{R}^2$. In each time step, a shape $S \subseteq \mathbb{R}^2$ arrives to be hit where $S = R^L \cup R^T$ is the union of a leftless rectangle R^L and a topless rectangle R^T. Upon each arrival, we may irrevocably purchase points to maintain a set $F \subseteq P$ which hits all shapes which have arrived so far. The goal is to minimize $|F|$. In the weighted case, each point has an associated non-negative cost c_p and the goal is to minimize the total cost of F.*

Perhaps surprisingly, we show that OCycAP with cut-arrivals reduces to Online TL-rectangle hitting set in both the weighted and unweighted case.

Theorem 1.10. *If there is an α-competitive algorithm for (weighted) Online TL-Hitting Set, then there is an α-competitive algorithm for (weighted) OCycAP.*

To see why this is true, consider any instance of OCycAP with cycle $G = (V, E)$ and links L. We will represent each link as a point in $\mathbb{R}^2$, and each 2-cut of G will correspond to the union of a topless and leftless rectangle. We will do this in such a way so that each link ℓ crosses the min-cut S if and only if its associated point is contained in the shape associated to S.

Indeed, we name the vertices of G in cyclic order as $V = (v_0, v_1, \ldots, v_{n-1}, v_0)$ with an arbitrary vertex selected as the "left-most" vertex v_0. Then a min-cut $C \subseteq V \setminus v_0$ can be represented as an *interval* in the cyclic order $C = \{v_a, v_{a+1}, \ldots, v_b\}$, where $1 \leq a \leq b \leq n-1$. A link $\ell = (v_i, v_j)$ crosses the cut C if it enters the interval through its left boundary or leaves it through its right boundary. We can represent this with a pair of inequalities: for ℓ to enter C from the left, we must have $i < a \leq j \leq b$. Similarly, a link $\ell = (i, j)$ enters C from the right exactly when $a \leq i \leq b < j$. Thus, the link $\ell = (v_i, v_j)$ crosses C exactly when the point (i, j) is contained in $S = R^L \cup R^T$, where

$$R^L(C) = (-\infty, a-1] \times [a, b] \text{ and } R^T(C) = [a, b] \times [b+1, \infty)$$

are leftless and topless rectangles, respectively. Hence, for a cycle with n vertices, we recover an instance of Online TL hitting set with at most n^2 points which lie on the $n \times n$ integer grid. See Figure 1 for an illustration of the reduction.

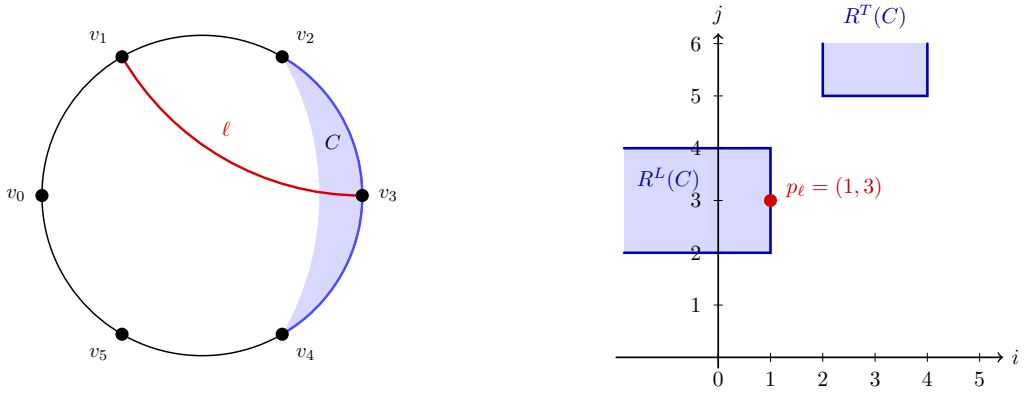


Figure 1: The minimum cut C is crossed by a link $\ell = (v_1, v_3)$. This link is represented by the point $(1, 3) \in \mathbb{Z}^2$, which is contained in the union of the leftless rectangle $R^L(C)$ and the topless rectangle $R^T(C)$.

1.2.3 Hitting TL-rectangles online

It now suffices to provide competitive bounds for the Online TL-rectangle hitting set problem. The weighted case is our main technical contribution, and is much more involved than the unweighted case. Hence, we explain each separately, and start with the simpler unweighted setting.

Unweighted Online TL-rectangle Hitting We mainly utilize a previously known logarithmic-competitive algorithm for hitting bottomless rectangles, given by De, Singh, and Tóth [DST25]. By rotation, their result easily extends to hitting leftless rectangles, or to hitting topless rectangles. Can we leverage this algorithm to obtain a logarithmic-competitive algorithm which hits unions of such shapes? One natural idea is to try to run both algorithms in parallel, essentially hitting **both** the topless and leftless rectangle on each arrival, and hope that this blows up the competitive ratio by a factor of 2 at most. Unfortunately, this does not work in general, but we are able to use a stronger property of their analysis which allows us to handle unions.

Theorem 1.11 (De, Singh, and Tóth [DST25])**.** *Let $P \subseteq [0, N)^2 \cap \mathbb{Z}^2$ be a point set. Then there is a deterministic algorithm for online bottomless rectangle hitting set satisfying:*

1. *If a bottomless rectangle is satisfied on arrival, the algorithm adds no new points. Otherwise, it adds at most two points.*
2. *For every point $p \in P$, there are $O(\log N)$ rectangles containing p that are unsatisfied on arrival.*

The above two properties easily imply an $O(\log N)$-competitive algorithm for hitting bottomless rectangles if the points lie on the $N \times N$ integer grid. Moreover, we are able to use the refined guarantee to design an $O(\log N)$-competitive algorithm for Online TL-rectangle hitting via a simple primal-dual fitting argument.

Weighted Online TL-rectangle Hitting Our algorithm for the weighted case is significantly more involved. We ultimately obtain an $O(\log n \log\log n)$-competitive algorithm for weighted online TL-rectangle hitting set with n points. We provide a high level overview of the techniques here.

Our main approach involves three-steps. First, as with any online hitting set problem with n elements, we can use the classical algorithm of [Alo+09] to obtain an $O(\log n)$-competitive *fractional* solution x^t. Our next goal is to design an online (randomized) rounding algorithm which takes as input an arbitrary increasing fractional solution x^t at each time, and yields a feasible integral online solution. We show how to design such an online rounding algorithm for weighted bottomless rectangles with a loss of at most $O(\log\log n)$. Finally, our third step is a general observation that, given online rounding algorithms for two classes of shapes $\mathcal{S}_1$ and $\mathcal{S}_2$ with losses $\alpha_1(n)$ and $\alpha_2(n)$, we can construct an online rounding algorithm for the class consisting of unions of shapes from $\mathcal{S}_1$ and $\mathcal{S}_2$ with loss $2(\alpha_1(n) + \alpha_2(n))$. Applying this where $\mathcal{S}_1$ consists of leftless rectangles and $\mathcal{S}_2$ consists of topless rectangles yields an online rounding algorithm for TL-rectangles with loss $O(\log\log n)$. Thus, combining this with the $O(\log n)$-competitive fractional solution yields the desired bound.

Steps 1 and 3 are relatively straightforward, so we elaborate here on our online rounding algorithm for bottomless rectangles. In particular, we prove the following:

Lemma 1.12. *Given an instance of Online Bottomless-Rectangle hitting set with $|P| = n$, arriving request sequence R^t, and an online fractionally feasible solution x^t, there is an online randomized*

rounding algorithm which outputs an increasing sequence of feasible integral hitting sets $H^t \subseteq P$ so that $\Pr[p \in H^t] \leq O(\log \log n) x_p^t$.

To prove Lemma 1.12, we exploit a structural property that has been used to obtain improved algorithms for hitting geometric set systems. The *shallow cell complexity (SCC)* $\varphi(\ell, k)$ bounds the number of sets of size k in any restriction of the set system to ℓ elements. It is known that Bottomless rectangles, in particular, have low SCC. The concept of SCC was first introduced by Chan, Grant, Könemann, and Sharpe [Cha+12] to obtain improved approximations for geometric set systems in the *offline* setting. First, they solve the natural LP relaxation to obtain a feasible fractional solution x. In classical randomized rounding, one would then sample elements with probability proportional to their value in the fractional solution. Here, they instead create a number of duplicates of each element e, where the number of copies of e is proportional to x_e (after zeroing elements of small x value). They then employ a method called Quasi-uniform sampling which exploits the low-SCC of the instance to obtain random integral solution with improved bounds.

There are two issues when trying to employ Quasi-uniform sampling in the online setting. First, it is unclear how to correlate the irrevocable samples in each time step to preserve the correct marginals. Second, since the values of x^t are increasing, we need to use an increasing number of copies of each element over time. A recent paper of Bhore, Gupta, and Kumar [BGK26] attempts to handle these exact issues. In particular, they *successfully* design an online sampler which incurs a low loss if the SCC of the duplicated set system remains low. But, their claim that the duplicated set system preserves low SCC is false, and the paper was subsequently retracted.

However, we can get around this hurdle. The key is to define a fixed set system upfront with plenty of copies of each element. Then, as the value of x_e increases over the course of the algorithm, we "activate" an increasing number of copies of e, so that each arriving shape is hit only by activated copies. We then directly show that the SCC remains low for the resulting set family, and employ (a specialization of) the sampler of [BGK26] to obtain our result.

1.2.4 Unweighted OCycAP with node-arrivals

It is striking that our methods can obtain optimal, or nearly optimal competitive ratios even in the more general setting of cut-arrivals. One might wonder whether there is any difference in complexity between the two arrival models. We show that, for the case of unweighted Online Cycle Augmentation, there is a separation.

First, let us explain why there is no $o(\log n)$-competitive algorithm for unweighted OCycAP for cut arrivals. In fact, this result already follows from our earlier result which establishes an equivalence between OCycAP and OCAP under the cut-arrival model. Since there is no $o(\log n)$-competitive algorithm for Online TAP [NUW22], even in the unweighted setting, there cannot be such an algorithm for OCAP. Hence, by Theorem 1.7, there is no $o(\log n)$-competitive algorithm for OCycAP either.

That being said, there is also a clean, direct lower bound which takes place directly on a cycle instance, which we prove in Section 5.2. The lower bound also applies to fractional, and hence randomized algorithms as well.

OCycAP Algorithm for Node Arrivals In light of this, we turn to the node-arrival model. Here, we take advantage of the structure of the *link intersection graph*. If G is a cycle, then every link $\ell \in L$ can be viewed as a chord of the cycle. Informally, the link intersection graph is the graph

whose node set corresponds to L, with an edge between ℓ and ℓ' if they intersect as chords. It is well-known [Bas+14] in the node arrival setting that satisfying a terminal-pair (u, v) is equivalent to purchasing a set of links containing a path in the link intersection graph from a link incident to u to a link incident to v.

The algorithm maintains the set of *active* connected components of the link intersection graph that contain any terminals seen so far. When a terminal pair (u, v) arrives, we iteratively expand the component containing u by either choosing two links that merge it with another active component, or if no such choice exists, choosing two links that expand it farthest in both directions along the cycle. Using a careful charging argument, we can independently bound the total cost attributed to all merges, as well as the total cost of all steps in which a merge could not be performed, each by a constant times the cost of the optimal solution.

1.3 Related Work

Offline Connectivity Augmentation There is a great deal of work on Connectivity Augmentation in the offline setting. Focusing on the weighted case, the problem was first introduced by Eswaran and Tarjan [ET76], who showed WCAP is NP-complete. Later, a 2-approximation was given by Frederickson and JáJá for WTAP [FJ81]. After a long line of work on various special cases [CN13; Fio+18; TZ22], Traub and Zenklusen gave a $1.5 + \varepsilon$ approximation for offline WTAP, and then matched this for offline WCAP [TZ25; TZ23]. More recently, Cohen-Addad, Drygala, Klein, and Svensson, gave an α-approximation for WTAP where $\alpha < 1.5$ [Coh+26]. After various improvements over the years [Adj19; CG18a; CG18b; Eve+09; GKZ18], the current best approximation factor for unweighted TAP and CAP is 1.393 due to Cecchetto, Traub and Zenklusen [CTZ21].

Online Geometric Hitting Set An instance of online Geometric Hitting Set is defined by a set P of points in Euclidean space and a collection of shapes which arrive online to be hit. As mentioned, the algorithm of [Alo+09] can be applied to any such problem to obtain an $O(\log p \log m)$-competitive ratio if there are p points and m possible arriving sets. This holds even in the weighted case. However, improved competitive ratios can be achieved in certain geometric settings. Even and Smorodinsky [ES14] obtained optimal $O(\log p)$ competitive ratios for the unweighted case when the arriving shapes are are intervals on a line, or half-planes, or unit disks in the plane. Khan, Lonkar, Rahul, Subramanian and Wiese [Kha+23] gave an $O(\log N)$-competitive algorithm for axis-aligned squares when the points lie on an $N \times N$ integer grid, again in the unweighted setting. More recently, De, Singh, and Tóth [DST25] obtained an $O(\log N)$-competitive algorithm for hitting bottomless rectangles on an $N \times N$ integer grid, as well as an $O(\log M \log p)$-competitive algorithm for disks with radii in $[1, M]$. As far as we can tell, prior to this work, there are no improvements over the algorithm of [Alo+09] for any weighted geometric special cases of Online Hitting Set, excluding the retracted paper of [BGK26].

1.4 Organization of the paper

In Section 2, we formalize the problem models, online hitting set formulations, and preliminaries. In Section 3, we present the optimal $O(\log n)$-competitive algorithm for unweighted OCAP under cut-arrivals via reductions to cycles and TL-rectangles. In Section 4, we present the $O(\log n \log \log n)$-competitive randomized algorithm for weighted OCAP under cut-arrivals using our online rounding

framework. Finally, in Section 5, we establish the separation on cycles by giving a $O(1)$-competitive algorithm for node-arrivals and an $\Omega(\log n)$ lower bound for cut-arrivals.

2 Preliminaries

In all online connectivity augmentation models, we are given a k-edge-connected graph $G = (V, E)$ with $|V| = n$ and a set of candidate links $L \subseteq \binom{V}{2}$ upfront. The links have non-negative costs given by $c : L \to \mathbb{R}_{\geq 0}$. In the unweighted setting, we assume $c(\ell) = 1$ for all $\ell \in L$.

For a subset of vertices $C \subseteq V$, we denote by $\delta_E(C)$ the set of edges in E with exactly one endpoint in C, and by $\delta_L(C)$ the set of candidate links in L with exactly one endpoint in C. Since G is k-edge-connected, every cut C has $|\delta_E(C)| \geq k$. A cut C is a *minimum cut* (or k-cut) if $|\delta_E(C)| = k$. A link $\ell \in L$ is said to *cover* or *cross* a cut C if $\ell \in \delta_L(C)$.

Requests arrive online over time in discrete time steps. We use $t \geq 1$ to denote the current time step.

2.1 Online Connectivity Augmentation Models

Depending on the setting, requests either take the form of minimum cuts that need to be covered, or terminal vertex pairs that need to be connected.

Problem 2.1 (OWCAP and OCAP with cut arrivals)**.** *A sequence of k-cuts $C_1, C_2, \ldots$ of G arrive online one at a time. When a cut C_t arrives at time step t, we must irrevocably purchase a subset of links to add to our solution. At the end of time step t, our cumulative purchased link set $F_t \subseteq L$ must satisfy $|\delta_{F_t}(C_i)| \geq 1$ for all $i \leq t$. The objective in the Weighted Online Connectivity Augmentation Problem (OWCAP) is to minimize the total cost $\sum_{\ell \in F_t} c(\ell)$. In the unweighted version (OCAP), the goal is to minimize $|F_t|$.*

Problem 2.2 (OWCycAP and OCycAP with cut arrivals)**.** *This is the special case of OWCAP (or OCAP) with cut arrivals in which the given base graph G is a cycle and $k = 2$.*

Problem 2.3 (OWCAP and OCAP with node arrivals)**.** *In each time step t, a pair of terminal vertices $(s_t, \tau_t) \in \binom{V}{2}$ arrives. We must irrevocably purchase links so that the vertices s_t and τ_t are $(k+1)$-edge-connected in the augmented graph. That is, at the end of time step t, the cumulative purchased link set $F_t \subseteq L$ must be such that every cut C separating s_t and τ_t has $|\delta_{E \cup F_t}(C)| \geq k+1$. As before, OWCycAP and OCycAP denote the special cases where G is a cycle.*

2.2 Online Hitting Set and Competitive Analysis

The Connectivity Augmentation problems defined above can be formulated as instances of the Online Hitting Set problem.

Problem 2.4 (Online Hitting Set)**.** *Initially, we are given a set system $(U, \mathcal{F})$ where U is a universe of N elements with non-negative costs $c : U \to \mathbb{R}_{\geq 0}$ and $\mathcal{F}$ is a family of non-empty subsets of U. A sequence of requests $R_t \in \mathcal{F}$ arrive online. At each time step t, we must maintain an integral hitting set $H_t \subseteq U$ such that $H_t \cap R_{t'} \neq \emptyset$ for every $t' \leq t$, and $H_t \supseteq H_{t-1}$. The goal is to minimize $\sum_{e \in H_t} c(e)$.*

Fractional Hitting Set and LP Formulation. In the fractional relaxation of the problem, we must maintain a monotone fractional solution $x^t \in \mathbb{R}^U_{\geq 0}$ over time. The problem can be expressed as an online covering linear program:

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in U} c(e) x_e^t \\ \text{subject to} \quad & \sum_{e \in R_{t'}} x_e^t \geq 1 \quad \forall t' \leq t, \\ & x_e^t \geq x_e^{t-1} \quad \forall e \in U, \\ & x_e^t \geq 0 \quad \forall e \in U. \end{aligned}$$

Competitive Analysis. The performance of our online algorithms is measured using competitive analysis.

- **Deterministic Competitive Ratio:** A deterministic online algorithm is α-competitive if, for any valid request sequence, the cost of the algorithm's solution is at most $\alpha \cdot \text{OPT}$, where OPT is the cost of the optimal offline solution that knows the entire request sequence in advance.
- **Randomized Competitive Ratio:** For randomized algorithms, we evaluate the performance against an *oblivious adversary*, which has knowledge of the algorithm but cannot adapt to its random coin flips. An algorithm is α-competitive in expectation if, for any request sequence generated by an oblivious adversary, $\mathbb{E}[\text{cost}(\text{ALG})] \leq \alpha \cdot \text{OPT}$.
- **Fractional Competitive Ratio:** An online fractional algorithm is α-competitive if it maintains a valid fractional solution x^t whose total cost is at most α times the cost of the optimal offline fractional solution.

3 An $O(\log n)$-competitive algorithm for OCAP under cut arrivals

In this section, we give a simple $O(\log n)$ competitive algorithm for OCAP under cut arrivals.

3.1 The cycle case

We begin by giving an $O(\log n)$-competitive algorithm for OCycAP on a cycle with cut arrivals. Later, we show how to use this to handle the case of a general graph.

Problem 3.1 (OCycAP with cut arrivals)**.** *Upfront, there is a cycle $G = (V, E)$ and a collection of links $L \subseteq \binom{V}{2}$. At time step t, a minimum cut C_t arrives to be augmented. We seek to maintain a set of links $F_t \subseteq L$ of minimum cardinality so that F_t crosses all min-cuts which have arrived by time t.*

We now reduce the problem to an online geometric hitting set problem where the sets to be hit correspond to cuts, and points correspond to links. In particular, we convert each minimum cut C into the union of a leftless rectangle and a topless rectangle as follows. Order the vertices of G as $V = (v_0, v_1, \ldots, v_{n-1}, v_0)$ along the cycle, with an arbitrary vertex selected as v_0. A minimum cut C separates the cycle into two paths; we choose the path that does not contain v_0 to represent

the cut: $C = \{v_a, v_{a+1}, \ldots, v_b\}$, where $1 \le a \le b \le n-1$. We refer to this cut as $C = [a, b]$. For a link $\ell = (v_i, v_j)$ with $i < j$, the link crosses the cut $C = [a, b]$ precisely when it enters the interval through its left boundary or leaves it through its right boundary. For the link ℓ to enter from the left, we must have $i < a \le j \le b$. Similarly, the link $\ell = (i, j)$ enters $C = [a, b]$ from the right exactly when $a \le i \le b < j$.

We can now encode this as a geometric hitting set problem as follows. For each link $\ell = (v_i, v_j)$, we create a point $p_\ell = (i, j) \in \mathbb{Z}^2$. For each arriving cut $C = [a, b]$, we create two rectangles which are subsets of $\mathbb{R}^2$:

$$R^L(a, b) = (-\infty, a-1] \times [a, b]$$

and

$$R^T(a, b) = [a, b] \times [b+1, \infty).$$

Notice that R^L is a leftless rectangle and R^T is a topless rectangle, and all corners lie on the integer lattice.

Proposition 3.2. *A link $\ell = (v_i, v_j)$ with $i < j$ crosses the minimum cut $C = [a, b]$ if and only if $p_\ell = (i, j) \in R^L(a, b) \cup R^T(a, b)$.*

Hence, we have reduced the problem to Online TL-rectangle hitting set, Problem 1.9. We now show how to leverage a recent result on hitting set for bottomless rectangles to yield an $O(\log n)$ competitive algorithm for the problem. We use a stronger property of the analysis which allows us to handle unions.

Theorem 1.11 (De, Singh, and Tóth [DST25])**.** *Let $P \subseteq [0, N)^2 \cap \mathbb{Z}^2$ be a point set. Then there is a deterministic algorithm for online bottomless rectangle hitting set satisfying:*

1. *If a bottomless rectangle is satisfied on arrival, the algorithm adds no new points. Otherwise, it adds at most two points.*
2. *For every point $p \in P$, there are $O(\log N)$ rectangles containing p that are unsatisfied on arrival.*

The above two properties imply an $O(\log N)$-competitive algorithm for bottomless rectangle hitting set where $P \subseteq [0, N)^2 \cap \mathbb{Z}^2$. By rotation, this algorithm can easily be adapted to handle topless and leftless rectangles with the same guarantee. We now give the $O(\log n)$-competitive algorithm for OCycAP. See Algorithm 1.

Algorithm 1: OCycAP under cut arrivals

Input: A cycle $G = (V, E)$ where $|V| = n$, a set of links $L \subseteq \binom{V}{2}$, and an online sequence of minimum cuts $C_1, \ldots, C_m$.

Let

$$P = \{(i, j) : (v_i, v_j) \in L,\ i < j\} \subseteq [0, n)^2 \cap \mathbb{Z}^2.$$

Initialize $F \leftarrow \emptyset$.

Initialize a leftless-rectangle hitting-set algorithm $\mathcal{A}^L$ on P

Initialize a topless-rectangle hitting-set algorithm $\mathcal{A}^T$ on P.

For each arriving cut $C = [a, b]$:

  If C is already covered by F, do nothing.

  If C is not covered by F:

    Let

$$R^L = (-\infty, a-1] \times [a, b] \qquad \text{and} \qquad R^T = [a, b] \times [b+1, \infty).$$

    Feed the leftless rectangle R^L to the algorithm $\mathcal{A}^L$ and let Q^L be the set of points purchased.

    Feed the topless rectangle R^T to the algorithm $\mathcal{A}^T$ and let Q^T be the set of points purchased.

    Add to F the links corresponding to points in $Q^L \cup Q^T$.

Proposition 3.2 essentially shows that OCycAP with cut arrivals is equivalent to an online geometric hitting set problem where each arriving shape is the union of a topless and a leftless rectangle. Algorithm 1 handles the union by purchasing points to cover *both the topless and the leftless rectangle, as long as the arriving union shape is not already hit.*

We now show that this yields the desired competitive guarantee.

Theorem 3.3. *Algorithm 1 is an $O(\log n)$-competitive algorithm for OCycAP with cut arrivals.*

Proof. We will use a dual-fitting argument. First we describe the covering LP for OCAP and its dual.

$$\begin{aligned} \text{minimize} \quad & \sum_{\ell \in L} x_\ell \\ \text{subject to} \quad & \sum_{\ell:\, \ell \text{ covers } C_t} x_\ell \geq 1 \quad \forall t, \\ & x_\ell \geq 0 \quad \forall \ell \in L. \end{aligned} \qquad \begin{aligned} \text{maximize} \quad & \sum_{t=1}^{k} y_t \\ \text{subject to} \quad & \sum_{t:\, \ell \text{ covers } C_t} y_t \leq 1 \quad \forall \ell \in L, \\ & y_t \geq 0 \quad \forall t. \end{aligned}$$

Consider a run of Algorithm 1. We first construct a primal solution from the run of the algorithm. For each link $\ell \in L$, set $x_\ell = 1$ if ℓ is purchased by the algorithm, and set $x_\ell = 0$ otherwise. So x is a feasible primal solution with objective value $|F|$.

We next construct a dual solution. Let T denote the set of iterations at which the arriving cut is not already covered by F. Define $y_t = 1$ if $t \in T$, and $y_t = 0$ otherwise. Thus the value of the dual solution is $|T|$. Also, y is $O(\log n)$-approximately feasible. Indeed, by property 2 of Theorem 1.11, for every link ℓ, the corresponding point p_ℓ is contained in at most $O(\log n)$ unsatisfied rectangles in each of the two geometric instances. Therefore ℓ covers at most $O(\log n)$ cuts C_t for which $y_t = 1$. So there exists a constant c such that $\sum_{t:\, \ell \text{ covers } C_t} y_t \leq c \log n$ for every link $\ell \in L$. By weak duality, we conclude that $\text{OPT} \geq |T| / c \log n$.

Finally, by definition of Algorithm 1, it purchases no points on iterations outside of T, and by property 1 of Theorem 1.11, it purchases at most four links for each iteration in T. Hence $|F| \leq 4|T| \leq (4c \log n) \cdot \text{OPT}$. Therefore, Algorithm 1 is $O(\log n)$-competitive. □

3.2 From OCycAP to general OCAP

We now show how to reduce the OCAP with cut arrivals problem on a general graph G, to the same problem where G is a cycle. This is a simple corollary of the following theorem, due to Dinitz, Karzanov, and Lomonosov [DKL76] (called "Theorem on circumference disposition" in their paper).

Theorem 3.4 (Dinitz, Karzanov, and Lomonosov [DKL76])**.** *Given any connected graph $G = (V, E)$, there exists an (efficiently computable) cycle $H = (V, E')$ such that for all $C \subseteq V$, if $\delta_E(C)$ is a minimum cut in G, then $\delta_{E'}(C)$ is a minimum cut in H.*

Proposition 3.5. *If there is an α-competitive algorithm $\mathcal{A}$ for OCAP with cut-arrivals on a cycle, then there is an α-competitive algorithm for OCAP with cut-arrivals on a general graph. This holds in the weighted case as well.*

Proof. Given an instance of OCAP with online cut-arrivals $C_1, \ldots, C_k$ in a general graph $G = (V, E)$, we apply the reduction given in Theorem 3.4 to obtain a cycle $H = (V, E')$. The link set L considered as a subset of $\binom{V}{2}$ is exactly the same. If the links have costs, these are preserved as well.

Now, whenever a minimum cut C_t arrives to be augmented in G, we will cause the minimum cut C_t to arrive in H. The set of links which hit C_t in G is identical to the set of links which hit C_t in H. The algorithm $\mathcal{A}$ will maintain a set of links F which hits all cuts which have arrived so far in H. This set of links F is also feasible for G. Furthermore, the optimal solutions for covering $C_1, \ldots, C_k$ in G and H are equal. Thus the set of links F is α-competitive for G as well. □

4 An $O(\log n \log \log n)$-competitive algorithm for OWCAP under cut arrivals

Recall that OWCAP reduces to OWCycAP (Theorem 1.7). Also OWCycAP on cycle with n vertices reduces to online TL-rectangle hitting set (Theorem 1.10) with $O(n^2)$ points. Thus, to prove Theorem 1.3, it suffices to give a randomized online algorithm for online TL-rectangle hitting set with expected competitive ratio $O(\log n \log \log n)$, where n now refers to the number of points in the instance. Our approach is to obtain an online fractional hitting set and then round it. Since there is an $O(\log n)$-competitive algorithm for online fractional hitting set in arbitrary set systems [Alo+09], it remains to design an online rounding algorithm for fractional TL-rectangle hitting set with a $O(\log \log n)$ overhead. Thus, the goal in this section is to prove the following theorem.

Theorem 4.1. *If there exists a $\rho(n)$-competitive fractional algorithm for online TL-rectangle hitting set, then there exists a randomized online algorithm for online TL-rectangle hitting set with expected competitive ratio $O(\rho(n) \log \log n)$.*

We prove the theorem in two steps. Recall that in online TL-rectangle hitting set, the requests are unions of topless and leftless rectangles. We first show how to round an online fractional solution for rectangles that are unbounded in a single direction (we canonically use bottomless rectangles

for this section). In Section 4.1, we give a rounding algorithm such that at every time step t, the point p is included in its hitting set with probability at most $O(\log\log n)x_p^t$. This is the crux of our approach. Then in Section 4.2, we show how to use rounding algorithms for topless and leftless rectangles to obtain a rounding algorithm for their union. We give intuitive explanations of these steps below.

Once we have a fractional solution, classical randomized rounding requires us to boost probabilities by a factor of $O(\log n)$ to ensure that the sampled solution is a feasible hitting set. But this loss is more than what we can accommodate. Instead we use an online sampler that exploits properties of the range space (namely its shallow cell complexity), to provide a tighter bound.

The online sampler is based on the Quasi-uniform sampling approach developed by [Cha+12] for the offline setting. To extend this to the online setting, we would ideally create a new point set, which consists of copies of every point in the input, where the number of copies is proportional to the value assigned to that point by the fractional solution. But since the underlying universe of points cannot keep changing in the online setting, we create n copies for each point upfront, but only *activate* an appropriate subset that depends on the fractional value assigned to the point at any given time step. To show that the range system has bounded shallow cell complexity, we imagine the set of points to be in 3D space, with the third dimension being their time of activation. Finally, we show that the arriving sets can be thought of as octants, which can be shown to have the required shallow cell complexity.

To handle shapes which are the union of a topless and a leftless rectangle, we do not deal with the geometry but work directly with the fractional solution, and modify the rounding step. In particular, since x is fractionally feasible, one of the two constituent rectangles is hit to an extent at least $1/2$. So, we consider only the heavier rectangle, double the values assigned to these points (to maintain feasibility), and work with this new fractional solution to do the rounding step. This loses a factor of 2 at most, so we obtain the same $O(\log\log n)$ bound for the union.

4.1 Online rounding for bottomless rectangles

We first introduce some necessary terminology on shallow cell complexity.

Definition 4.2 (Traces and shallow cell complexity)**.** *Let $(\mathcal{V}, \mathcal{G})$ be a set system on a finite universe of elements $\mathcal{V}$.*

1. ***Trace:*** *For any subset $W \subseteq \mathcal{V}$, the* trace *of $\mathcal{G}$ on W is the family of induced subsets $\mathcal{G}|_W := \{Q \cap W : Q \in \mathcal{G}\}$.*

2. ***Shallow trace:*** *For an integer $\kappa \geq 1$, a trace $S \in \mathcal{G}|_W$ is called a* κ-shallow *trace if we have $1 \leq |S| \leq \kappa$.*

3. ***Shallow cell complexity (SCC):*** *We say that $(\mathcal{V}, \mathcal{G})$ has shallow cell complexity φ (for function $\varphi : \mathbb{N} \times \mathbb{N} \to \mathbb{R}_{\geq 0}$), if for every subset $W \subseteq \mathcal{V}$ with $|W| = m$,*

$$\left|\{S \in \mathcal{G}|_W : 1 \leq |S| \leq \kappa\}\right| \leq m \cdot \varphi(m, \kappa).$$

In other words, $(\mathcal{V}, \mathcal{G})$ has an SCC of φ if the number of κ-shallow traces on any set of size m is at most $m \cdot \varphi(m, \kappa)$.

Fix $P \subseteq \mathbb{R}^2$, a fixed universe of $n \geq 1$ points with non-negative costs c_p. Requests R_t arrive online and the goal is to maintain an online hitting set H whose total $c(H) = \sum_{p \in H} c_p$ is minimized. We first focus on the case where each request R_t is a bottomless rectangle. Let x be an online fractional solution.

Denote by x_p^t the extent to which point p is bought in response to the first t requests, and define $x^t(S) = \sum_{p \in P \cap S} x_p^t$.

Theorem 4.3. *There exists a randomized algorithm that given an online sequence of bottomless rectangles and an online fractional solution x, maintains an online hitting set H such that $\Pr[p \in H] \leq O(\log \log n) x_p$ for every $p \in P$. The request sequence and x are independent of the algorithm's randomness.*

4.1.1 Online sampling for bounded SCC set systems

Once we have a fractional solution, we will use a specialized form of randomized rounding to obtain an integral solution. Rather than sampling with probabilities simply boosted from the fractional solution as in the classical randomized rounding approach, we use the following online sampler which exploits the geometric properties of our set system. The property used here is that the sets that arrive have a low SCC. The existence of such a sampler follows from a result of Bhore, Gupta and Kumar [BGK26]. We note that, as of October 8, 2026, [BGK26] was retracted from arXiv due to an error in Theorem 3.2. However, Theorem 5.5 in [BGK26] (version 1) is logically separate from the retracted result, and still holds. We state a specialization of this theorem, which is sufficient for our setting.

Lemma 4.4 (Low SCC Online Sampler)**.** *Let $\mathcal{V}$ be a universe of size N, and let $N^{1/3} \leq B \leq N$ and $1 \leq A \leq C_A \log N$ be parameters (for a universal constant C_A, to be determined later). Let $Q_1, Q_2, \ldots$ be an oblivious online hitting set request sequence satisfying $|Q_t| \geq B$ for every t. Let $\mathcal{G} = \{Q_1, Q_2, \ldots\}$, and assume that the set system $(\mathcal{V}, \mathcal{G})$ has shallow cell complexity bounded by*

$$\varphi(m, \kappa) \leq A\kappa^2$$

for all $m, \kappa \geq 1$. Then there exists a randomized online algorithm maintaining an increasing hitting set $J_t \subseteq \mathcal{V}$ such that

$$\Pr[e \in J_t] \leq O\left(\frac{\log(A \log N)}{B}\right)$$

for every element $e \in \mathcal{V}$ and every finite horizon t.

Proof. This lemma is a special case of [BGK26, Theorem 5.5], rephrased for our purposes. In particular, if we write the set family of requests as $\mathcal{G} = \{Q_1, Q_2, \ldots\}$, then [BGK26, Theorem 5.5] provides an online algorithm that maintains an increasing hitting set $J_t \subseteq \mathcal{V}$ with

$$\Pr[e \in J_t] \leq O\left(\frac{1}{B} \log\left(B_{L^*} \cdot \log N \cdot \varphi_{\mathcal{G}}(N_{L^*}, B_{L^*})\right)\right) \tag{1}$$

where B is a parameter satisfying $B > c_1$ for constant c_1 and for which $|Q_t| \geq B$ for every $Q_t \in \mathcal{G}$ (these two conditions are clearly satisfied by the conditions of the lemma). B_ℓ, N_ℓ and L^* are defined as follows.

$$N_\ell := \frac{c_1 N}{2^\ell} \qquad B_\ell := \frac{B}{2^\ell}$$

and L^* is the largest $\ell \in [0, \log(B/c_1)]$ for which

$$h_{\mathcal{G}}(N_\ell, B_\ell) := \sqrt{\frac{8 \ln(B_\ell \lceil \log N_\ell \rceil \, \varphi_{\mathcal{G}}(N_\ell, B_\ell))}{B_\ell}} \leq \frac{1}{2}.$$

Finally, $\varphi_{\mathcal{G}}$ must be a *well-behaved* bound on the *shallow cell complexity* of the set system $\mathcal{G}$. The condition in the statement of our lemma is that $\varphi_{\mathcal{G}}(\cdot, \kappa) \leq A\kappa^2$. And it is easy to verify that this bound is $(3/2, 8)$-well-behaved in the sense of [BGK26, Definition 2.2], since for any $\kappa \geq 8$ we have $A\kappa^2 \leq (A(\kappa/2)^2)^{3/2}$. Thus, all hypotheses of [BGK26, Theorem 5.5] are satisfied.

Since $B \geq N^{1/3}$, we can choose some $\hat{\ell} \in [0, \log(B/c_1)]$ so that $B_{\hat{\ell}} = \Theta(\log N)$. Observe that $N_{\hat{\ell}} \leq O(N)$, and since $A \leq O(\log N)$, we get

$$h_{\mathcal{G}}(N_{\hat{\ell}}, B_{\hat{\ell}}) \leq \sqrt{\frac{8 \ln(B_{\hat{\ell}}^3 \cdot O(\log N) \cdot A)}{B_{\hat{\ell}}}} \leq \sqrt{\frac{O(\log \log N)}{\Theta(\log N)}} = o(1).$$

In particular, for this choice of $\hat{\ell}$, we have $h_{\mathcal{G}}(N_{\hat{\ell}}, B_{\hat{\ell}}) \leq 1/2$ (for sufficiently large N), and hence $L^* \geq \hat{\ell}$ and $B_{L^*} \leq O(\log N)$.

Finally, making the substitutions for $\varphi_{\mathcal{G}}$ and B_{L^*} in (1) yields the desired result. □

4.1.2 Virtual online hitting set instance for sampler

Armed with the online sampler from Lemma 4.4, we wish to use it to sample points with probability proportional to the values assigned to them by the fractional solution. We now describe how to duplicate the point set to simulate this sampling.

New point set: We refer to the given point set P as the physical points, and construct a new universe of virtual points $\mathcal{P} = P \times [n]$, consisting of n virtual copies for each physical point $p \in P$ (yielding $N = |\mathcal{P}| = n^2$). We denote by $\pi : \mathcal{P} \to P$ the projection mapping each virtual point $e = (p, i)$ to its physical parent p, and the corresponding inverse map $\pi^{-1} : P \to \mathcal{P}$ outputs the set of virtual copies for any physical point $p \in P$. But, not all copies of a point $p \in P$ are to be used at a given time step t. Since we require the number of copies of p to be proportional to its fractional cost x_p^t, we only *activate* a fraction of the copies at time t. This is done as follows.

First we define the vector

$$z_p^t = \begin{cases} 0, & x_p^t < 1/(2n), \\ \min\{1, 2x_p^t\}, & x_p^t \geq 1/(2n). \end{cases} \tag{2}$$

and similar to the vector x^t, we also define $z^t(S) = \sum_{p \in P \cap S} z_p^t$. Now, we say that at time step t the first $a_p^t = \lceil n z_p^t \rceil$ copies of point $p \in P$ are active, i.e., $A_t := \bigcup_{p \in P}(\{p\} \times [a_p^t]) \subseteq \mathcal{P}$ is the set of all active copies at time t. For any subset $S \subseteq \mathbb{R}^2$, let $A_t(S) := A_t \cap \pi^{-1}(S)$ denote the set of active copies of points in S at time step t.

We now show that the fractional solution z^t is monotone, feasible, and not too large compared to the original fractional solution x^t.

Lemma 4.5. *The vector z^t is monotone and feasible, the sets A_t are increasing ($A_1 \subseteq A_2 \subseteq \cdots$), and for any request R_t and every point $p \in P$,*

$$|A_t(R_t)| \geq n, \qquad |A_t \cap \pi^{-1}(p)| = a_p^t \leq 4n\, x_p^t.$$

Proof. If there were a single point p' with $x^t_{p'} \geq 1/2$ in a satisfied query set R_t, then $z^t_{p'} \geq 1$, hence also satisfying R_t. Now, we note that we round down a variable x^t_p only if its value is less than $1/(2n)$. Since there are n variables, a value of at most $1/2$ gets discarded, giving us that R_t still has value at least $1/2$, with each variable having value at most $1/2$. Since the new vector doubles the value of all such variables, we would still have $z^t(R_t) \geq 1$, making it a feasible solution. Similarly, since x is monotone, and z doubles the variables which satisfy the threshold of $\geq 1/(2n)$, z is also monotone.

When $z^t_p = 0$, the corresponding $x^t_p \geq 0$, so we trivially have $a^t_p \leq 4n\, x^t_p$. And, for every positive z^t_p, we have $nz^t_p \geq 1$. Therefore

$$a^t_p = \lceil nz^t_p \rceil \leq 2nz^t_p \leq 4n\, x^t_p. \tag{3}$$

Finally,

$$|A_t(R_t)| = \sum_{p \in P \cap R_t} \lceil nz^t_p \rceil \geq n\, z^t(R_t) \geq n. \qquad \square$$

Virtual Requests: Similarly we need to ensure that the sets being sent to the sampler belong to a family $\mathcal{F}$ satisfying the properties required for Lemma 4.4. The sets are created in an online fashion, and when a physical request R_t arrives, we translate this to a virtual request $Q_t \subseteq \mathcal{P}$ to be passed to the online sampler.

Ideally, we would like to send $A_t(R_t)$ as the virtual request. However, in general, this family of sets does not satisfy the low shallow cell complexity condition required to apply Lemma 4.4. Instead, we choose a subset of $A_t(R_t)$ that is sufficiently large and falls within a more structured family of sets (note that hitting a subset of the required set is sufficient to hit the original set).

Before we go into details of the translation, we give a brief introduction to *range trees*, which are a structure used in our construction. Given a set of points on the line, we will construct a data structure that can represent any possible line segment on this set of points. We first start with the sorted list of points (including $-\infty$ and ∞), and consider the segments between consecutive points (called canonical intervals) to be leaves of a tree. On this set as leaves, we construct a balanced binary tree. Each node in this tree represents an interval, which is the union of all the canonical intervals stored in its subtree. And the height of this tree is $O(\log n)$, where n is the number of points in the input set.

With the notion of range trees in hand, the translation procedure works as follows: Consider the entire point set P projected to the x-axis and construct a range tree on this set of points. For a node u of the range tree, denote the set of points in P whose projection lies in the interval corresponding to u as $S_u \subseteq P$.

When a bottomless rectangle request R_t of the form $[a_t, b_t] \times (-\infty, c_t]$ arrives, we consider the deepest node v in the range tree that contains the interval $[a_t, b_t]$, i.e., the node in the range tree such that its corresponding interval contains the range $[a_t, b_t]$, but none of its children's intervals do (or v is a leaf). W.l.o.g., we can assume that a_t, b_t are x-projections of points in P. Among the set of points in S_v, select the points that satisfy $y \leq c_t$ (i.e., the set $R_t \cap P$), and consider the set of their active copies $A_t(R_t)$. If v is a leaf, we simply use $A_t(R_t)$ directly. Otherwise, letting the two child nodes of v be v_ℓ and v_r, we consider the sets $R_t \cap P \cap S_{v_\ell}$ and $R_t \cap P \cap S_{v_r}$, construct the corresponding sets of active copies $A^{\text{left}}_t, A^{\text{right}}_t$, and pick the one with larger number of points.

With the instance as described above, we can now give the rounding algorithm using this instance, and the online sampler:

Algorithm 2: Online Rounding for Bottomless Rectangles

Input: A fixed point set $P \subseteq \mathbb{R}^2$, an arriving sequence of bottomless rectangles R_t, and an $O(\log n)$-competitive online fractional solution x^t.

Output: A monotonically increasing hitting set H, satisfying all requests up until the current time step.

Initialize: Hitting set $H \leftarrow \emptyset$

Construct 1D binary range tree on the horizontal coordinates of P

Initialize the online sampler (Lemma 4.4) on universe $\mathcal{P} = P \times [n]$ with parameters $N = n^2$, $B = \lceil n/2 \rceil$, and $A = \max\{1, 2C_{\text{dom}}(\lceil \log_2 n \rceil + 1)\}$

foreach *arrival of bottomless rectangle* $R_t = [a_t, b_t] \times (-\infty, c_t]$ *with fractional vector* x^t **do**

`// Step 1: Activate copies`

  foreach $p \in P$ **do**

    Let $z_p^t \leftarrow \begin{cases} 0, & x_p^t < 1/(2n) \\ \min\{1, 2x_p^t\}, & x_p^t \geq 1/(2n) \end{cases}$

    Mark the first $a_p^t \leftarrow \lceil n z_p^t \rceil$ copies of p in $\mathcal{P}$ as active

  Update $A_t \subseteq \mathcal{P}$ to be the set of all currently active copies

`// Step 2: Query decomposition via horizontal range tree`

  Find the deepest node v of the range tree containing $[a_t, b_t]$

  if *v is a leaf* **then**

    $Q_t \leftarrow A_t(R_t)$

  else

    Partition $A_t(R_t)$ into subsets A_t^{left} and A_t^{right} associated with v's children

    Retain the heavier piece: $Q_t \leftarrow \arg\max\{|A_t^{\text{left}}|, |A_t^{\text{right}}|\}$

`// Step 3: Online sampling`

  Feed Q_t to the online sampler (Lemma 4.4) and update the sampled hitting set $J_t \subseteq \mathcal{P}$

`// Step 4: Purchase points`

  Purchase physical parents of sampled copies: $H \leftarrow H \cup \{\pi(e) : e \in J_t\}$

4.1.3 Bounding the SCC of the Virtual Instance

Definition 4.6 (Activation time)**.** *For each copy $e = (p, i) \in \mathcal{P}$, its* activation time τ_e *is the first time step at which it becomes active, i.e., $\tau_e = \min\{t \geq 1 : e \in A_t\}$, with $\tau_e = T + 1$ if e remains inactive at the final time step T.*

For the sake of our analysis, we map each copy $e \in \mathcal{P}$ of a physical point $p \in P$ with spatial coordinates (u_p, v_p) to the point $(u_p, v_p, \tau_e) \in \mathbb{R}^3$.

We can show that the sequence of queries we send to the sampler have a small number of shallow traces.

Lemma 4.7 (Shallow traces)**.** *Let $\mathcal{F}$ denote the family of all candidate query sets $Q_t \subseteq \mathcal{P}$ that can be generated across all nodes of the range tree. For any subset $W \subseteq \mathcal{P}$ and integer $\kappa \geq 1$,*

$$|\{Q_t \cap W : Q_t \in \mathcal{F} \text{ and } 1 \leq |Q_t \cap W| \leq \kappa\}| \leq O(\log n \cdot |W| \cdot \kappa^2). \tag{4}$$

To prove the above, we first show an accessory lemma that is implied from earlier work by Afshani and Tsakalidis [AT14]. A short proof is given in Appendix A for completeness.

Theorem 4.8 (3D octant shallow traces [AT14])**.** *There exists an absolute constant C_{dom} such that for any r-element point set in $\mathbb{R}^3$, the number of distinct nonempty traces of size at most κ under octants of a fixed orientation is at most $C_{\text{dom}} r\kappa^2$, for every integer $\kappa \geq 1$.*

Now we may proceed with the proof of Lemma 4.7.

Proof of Lemma 4.7. For the sake of analysis, let us project the set of virtual points $\mathcal{P}$ to $\mathbb{R}^3$ as follows: for a point $p = (u_p, v_p) \in P$, and for each copy $e = (p, i) \in \mathcal{P}$, we consider the projection of the copy to be the point (u_p, v_p, τ_e), where τ_e is the activation time for copy e (so that $s_z \leq t \iff e \in A_t$). We will show that the request sets that are sent to the sampler can be thought of as octants in $\mathbb{R}^3$ (locally on the range tree), and since we can bound the number of shallow traces for octants, we will be able to bound the number of shallow traces for the overall family of queries as well.

Consider the set of points $\pi^{-1}(S_u) \subseteq \mathcal{P}$ contained within the x-interval $[a, b]$ represented by a node u of the range tree. Consider any rectangle query $R_t = [a_t, b_t] \times (-\infty, c_t]$ arriving at time t that we assign to node u for query decomposition. We distinguish two cases:

- **If u is an internal node:** Let the left and right child nodes of u, u_ℓ, u_r respectively, correspond to the intervals $[a, c]$ and $[c, b]$ respectively, where $[a_t, b_t]$ is completely contained in $[a, b]$, but not in $[a, c]$ or $[c, b]$ (i.e., $a_t \leq c \leq b_t$). Since all points in S_{u_ℓ} already satisfy $x \leq c$ and all points in S_{u_r} satisfy $x \geq c$, the candidate sets A_t^{left} and A_t^{right} corresponding to R_t are given by

$$T_{\text{left}} = \{s = (s_x, s_y, s_z) \in \pi^{-1}(S_{u_\ell}) \mid s_x \geq a_t \text{ and } s_y \leq c_t \text{ and } s_z \leq t\}$$

  and

$$T_{\text{right}} = \{s = (s_x, s_y, s_z) \in \pi^{-1}(S_{u_r}) \mid s_x \leq b_t \text{ and } s_y \leq c_t \text{ and } s_z \leq t\}.$$

  Hence, every query set generated from an internal node u is an octant on the points $\pi^{-1}(S_u)$ projected to $\mathbb{R}^3$. The family of queries generated from this node is $\mathcal{F}_u = \mathcal{F}_{\text{left}} \cup \mathcal{F}_{\text{right}}$, which is the union of octants in two different orientations.

- **If u is a leaf:** The algorithm directly outputs $Q_t = A_t(R_t)$. Since $[a_t, b_t] \subseteq [a, b]$, this query is represented by $T_{\text{leaf}} = \{s = (s_x, s_y, s_z) \in \pi^{-1}(S_u) \mid s_x \geq a_t \text{ and } s_y \leq c_t \text{ and } s_z \leq t\}$, which is an octant of a single orientation. Thus, $\mathcal{F}_u$ consists of octants in one direction.

Now, using Theorem 4.8, the number of shallow traces for the family of queries $\mathcal{F}_u$ generated from node u is bounded by

$$\begin{aligned}
&|\{Q_t \cap W : Q_t \in \mathcal{F}_u \text{ and } 1 \leq |Q_t \cap W| \leq \kappa\}| \\
&\quad \leq |\{Q_t \cap W : Q_t \in \mathcal{F}_{\text{left}} \text{ and } 1 \leq |Q_t \cap W| \leq \kappa\}| + |\{Q_t \cap W : Q_t \in \mathcal{F}_{\text{right}} \text{ and } 1 \leq |Q_t \cap W| \leq \kappa\}| \\
&\quad \leq 2 \cdot C_{\text{dom}} \cdot |W \cap \pi^{-1}(S_u)| \cdot \kappa^2
\end{aligned}$$

(when u is a leaf, the bound is at most $C_{\text{dom}} \cdot |W \cap \pi^{-1}(S_u)| \cdot \kappa^2$, which trivially satisfies the same inequality).

Every query sent to the online sampler is processed at some node of the range tree, i.e., the family of queries is $\mathcal{F} = \bigcup_u \mathcal{F}_u$. So, summing the traces generated across all nodes u of the range

tree, we get

$$\begin{aligned}
&|\{Q_t \cap W : Q_t \in \mathcal{F} \text{ and } 1 \le |Q_t \cap W| \le \kappa\}| \\
&\quad \le \sum_u |\{Q_t \cap W : Q_t \in \mathcal{F}_u \text{ and } 1 \le |Q_t \cap W| \le \kappa\}| \\
&\quad \le 2C_{\text{dom}}\kappa^2 \cdot \sum_u |W \cap \pi^{-1}(S_u)| \\
&\quad \le 2C_{\text{dom}}\kappa^2 \cdot |W| \cdot \log n.
\end{aligned}$$

Where the last inequality follows, since every point can lie in at most $O(\log n)$ intervals corresponding to nodes of the range tree. □

We now show that the query sets are heavy.

Lemma 4.9 (Heaviness)**.** *For any request R_t, the query set Q_t produced by Algorithm 2 satisfies $|Q_t| \ge B$. Furthermore, $Q_t \subseteq A_t(R_t)$, so for any $e \in Q_t$, the point $\pi(e) \in P \cap R_t$, and purchasing $\pi(e)$ hits the request R_t.*

Proof. By Lemma 4.5, we know that $|A_t(R_t)| \ge n$. If the node v is a leaf, $Q_t = A_t(R_t)$, and thus $|Q_t| \ge n \ge B$. Otherwise, v is an internal node, and its active copies $A_t(R_t)$ are partitioned into A_t^{left} and A_t^{right}. Since $Q_t = \arg\max\{|A_t^{\text{left}}|, |A_t^{\text{right}}|\}$, we have $|Q_t| \ge \lceil |A_t(R_t)|/2 \rceil \ge \lceil n/2 \rceil = B$.

In either case, $Q_t \subseteq A_t(R_t)$, so any element $e \in Q_t$ is an active copy of some point in $P \cap R_t$. Thus, purchasing $\pi(e)$ hits R_t. □

We now have the ingredients to complete the proof of Theorem 4.3.

Proof of Theorem 4.3. We prove the theorem by showing that Algorithm 2 maintains a hitting set H_t with the property that $\Pr[p \in H_t] \le O(\log\log n)x_p^t$ for every point p and time step t. Consider the requests Q_t sent to the online sampler (Lemma 4.4). By Lemma 4.9, we have that $|Q_t| \ge \lceil n/2 \rceil$. By Lemma 4.7, the query family satisfies the shallow trace condition with $A = 2C_{\text{dom}}(\lceil \log_2 n \rceil + 1)$. Since $N = n^2$, we have $A \le C_A \log N$ by choosing the constant $C_A = 2C_{\text{dom}}/\ln 2$. Together, this allows us to apply Lemma 4.4 with parameters $A = O(\log n)$, $B = \lceil n/2 \rceil$, and $N = n^2$ to get that for every point $p \in P$, each of its active copies is in the sampled hitting set J_t with probability at most $O(\frac{1}{n}\log\log n)$. Since $p \in H_t$ only if one of its active copies is in J_t and p has at most $a_p^t \le 4nx_p^t$ active copies (Equation (3)), we have that $\Pr[p \in H_t] \le O(\log\log n)x_p^t$. □

4.2 Online rounding for unions

Definition 4.10 (Online $\mathcal{F}$-hitting set)**.** *Let U be a universe of n elements and $\mathcal{F}$ be a family of subsets of U. An instance of the* online $\mathcal{F}$-hitting set problem *is a sequence of requests $R_t \in \mathcal{F}$.*

Definition 4.11 ($\alpha(n)$-rounding algorithm)**.** *Let U be a universe of n elements and $\mathcal{F}$ be a family of subsets of U. An* $\alpha(n)$-rounding algorithm *receives at time step t, the request $R_t \in \mathcal{F}$ and the current fractional hitting set x^t, and maintains an online hitting set H such that $\Pr[e \in H] \le \alpha(n) \cdot x_e^t$ for every time step t and every element e.*

We now show that given rounding algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$ for $\mathcal{F}_1$ and $\mathcal{F}_2$, respectively, we can combine them to get a rounding algorithm for $\mathcal{F}_3 = \{S_1 \cup S_2 : S_1 \in \mathcal{F}_1, S_2 \in \mathcal{F}_2\}$.

Theorem 4.12. *Let U be a universe of n elements. Suppose $\mathcal{F}_1$ and $\mathcal{F}_2$ are two families of subsets of U, and let $\mathcal{F}_3 = \{S_1 \cup S_2 : S_1 \in \mathcal{F}_1, S_2 \in \mathcal{F}_2\}$. Furthermore, suppose that $\mathcal{A}_1$ is an $\alpha_1(n)$-rounding algorithm for $\mathcal{F}_1$ and $\mathcal{A}_2$ is an $\alpha_2(n)$-rounding algorithm for $\mathcal{F}_2$. Then, there is a $2(\alpha_1(n) + \alpha_2(n))$-rounding algorithm for $\mathcal{F}_3$.*

Proof. The idea for our algorithm is relatively straightforward. Consider an online $\mathcal{F}_3$-hitting set instance and let x be an online fractional hitting set. The algorithm runs $\mathcal{A}_1$ and $\mathcal{A}_2$; let H_1 and H_2 be their respective hitting sets. At each time step t, the algorithm receives a request $R_t \cup S_t$ where $R_t \in \mathcal{F}_1$ and $S_t \in \mathcal{F}_2$, and x^t, and defines $y_e^t = 2x_e^t$. If $y^t(R_t) \geq 1$, the algorithm feeds R_t and y^t to $\mathcal{A}_1$ and otherwise it feeds S_t and y^t to $\mathcal{A}_2$. Then it updates its hitting set H by setting $H = H_1 \cup H_2$.

We now turn to analyzing the above algorithm. For each time step t, since x is a fractional hitting set, we have $x^t(R_t \cup S_t) \geq 1$ and so, we get $y^t(R_t) \geq 1$ or $y^t(S_t) \geq 1$. Thus, y^t is indeed a fractional hitting set for the sequence of requests sent to $\mathcal{A}_1$ and also for the sequence of requests sent to $\mathcal{A}_2$. The guarantees of $\mathcal{A}_1$ and $\mathcal{A}_2$ imply that for each element $e \in U$ and each time step t, we have $\Pr[e \in H_1] \leq \alpha_1(n) \cdot y_e^t$ and $\Pr[e \in H_2] \leq \alpha_2(n) \cdot y_e^t$. Since $H = H_1 \cup H_2$, we have that $\Pr[e \in H] \leq (\alpha_1(n) + \alpha_2(n)) \cdot y_e^t \leq 2(\alpha_1(n) + \alpha_2(n)) \cdot x_e^t$, as desired. □

Proof of Theorem 4.1. By rotation of the coordinate axes, Theorem 4.3 holds identically for topless and leftless rectangles, yielding an $O(\log \log n)$-rounding algorithm for each family. By Theorem 4.12, these can be combined into an $O(\log \log n)$-rounding algorithm for TL-rectangles (which are unions of topless and leftless rectangles).

Given a $\rho(n)$-competitive fractional solution x, this rounding algorithm maintains a feasible integral hitting set H such that $\Pr[p \in H] \leq O(\log \log n) x_p$ for all $p \in P$. By linearity of expectation, the expected cost of H satisfies:

$$\mathbb{E}[c(H)] = \sum_{p \in P} c_p \Pr[p \in H] \leq O(\log \log n) \sum_{p \in P} c_p x_p \leq O(\rho(n) \log \log n) \cdot \mathrm{OPT}.$$ □

5 Unweighted OCycAP

In this section, we prove Theorem 1.5 by presenting an algorithm achieving a 10-competitive ratio for the OCycAP problem with node arrivals in Section 5.1, and a $\Omega(\log n)$ lower bound in the cut-arrivals setting in Section 5.2.

5.1 $O(1)$-competitive for node arrivals

We now describe a deterministic $O(1)$-competitive algorithm for OCycAP with node arrivals.

Theorem 5.1. *There is a deterministic 10-competitive algorithm for OCycAP with node arrivals.*

5.1.1 Notation and Terminology

To describe the algorithm, we first establish some notation. Since the given graph R is a cycle, we call two links ℓ and ℓ' *intersecting* if they intersect when drawn as straight lines on a planar circular embedding of R (including if they intersect at an endpoint). The *link intersection graph* of R is then defined as the graph on a node set corresponding to the links L, and with two links adjacent

whenever they are intersecting. Recall that in the node arrival setting, our goal is to purchase links so that every requested terminal pair (s,t) is satisfied: it becomes 3-edge-connected in the augmented graph. It is well-known (see, e.g., [Bas+14; TZ23]) that a pair is satisfied if and only if there is a path of purchased links in the link intersection graph from a link with s as an endpoint to a link with t as an endpoint.

Therefore, it useful to study connected component of the link intersection graph for the set of links chosen by the algorithm. For a component S, we abuse notation and say it contains a vertex v of R if any link in S is incident to v.

For any vertex v on the cycle, consider the clockwise labeling $v = v_0, \ldots, v_{n-1}$ of the vertices of the cycle R. Given a set S of links, define its *interval with respect to* v, denoted $I_v(S)$, to consist of the vertices $v_i, v_{i+1}, \ldots, v_j$ where i, j are the smallest and largest indices of the vertices incident to any link in S. Note that even if S is a connected component in the link intersection graph, $I_v(S)$ may contain vertices that are not in the same connected component as S.[1] The following lemma establishes additional structure on the intervals of connected components of the link intersection graph.

Lemma 5.2. *For any link intersection graph corresponding to a set of links on R, the set of intervals of connected components forms a laminar family.*

Proof. Observe that if $I_v(S) = [i,j]$ then for every i' such that $i \le i' < j$, there is a link $\ell \in S$ such that $I_v(\ell)$ contains $(i', i'+1)$. Thus, if the intervals $I_v(S)$ and $I_v(S')$ cross, then there is a link $\ell \in S$ and a link $\ell' \in S'$ whose intervals $I_v(\ell)$ and $I_v(\ell')$ cross. Thus, ℓ and ℓ' are adjacent to each other in the link intersection graph. □

5.1.2 Algorithm for Node Arrivals

With this terminology in hand, we can now define the algorithm. In Algorithm 3, we say that a connected component of the link intersection graph is *active* if it contains any terminal seen so far.

Algorithm 3: `OCycAP with Node Arrival`

input : Cycle R, collection L of links, and a sequence $(s_i, t_i) \in \binom{V}{2}$ of terminal pairs.
output: A set of links so that so that (s_i, t_i) are 3-edge-connected.

When terminal pair (s,t) **arrives:**
- Let S and T be the active components containing s and t, respectively.
- Suppose WLOG (by Lemma 5.2, swapping the labels of s and t if necessary) that either $I_t(S)$ and $I_t(T)$ are disjoint or $I_t(S)$ is contained in $I_t(T)$.
- **while** $S \neq T$ **do**
  - (Merge step) If there exist at most 2 links such that buying them merges S with another active component, buy them.
  - (Bracketing step) Otherwise, buy the two links leaving $I_t(S)$ whose endpoints outside of $I_t(S)$ have the smallest and largest index (with respect to the ordering with $v_0 = t$).

[1] For example, suppose S consists of a single link (v_0, v_{n-1}), then $I_v(S)$ contains every vertex.

To prove Theorem 5.1, we show that the total cost of merge steps is at most 8OPT and the total cost of bracketing steps is at most 2OPT for a total cost of at most 10OPT.

Proof of Theorem 5.1. Let r be the total number of distinct terminals requested across all arrivals. Since every valid solution must cover all r distinct terminals, and each link is incident to at most two endpoints, we have that $r/2 \leq \text{OPT}$. Moreover, each merge step strictly reduces the number of active components by one, so starting from at most r initial active components, there can be at most $r-1$ merge steps. Thus, the total cost due to merge steps is at most $2(r-1) \leq 4\text{OPT} \leq 8\text{OPT}$.

We now define a charging scheme to show that the number of bracketing steps is at most OPT. Consider an iteration of the algorithm in which a bracketing step was performed on interval $I_t(S)$. Let A be the set of links bought by the algorithm so far, and let P^* be the shortest path in OPT that connects the component S and t. Since the merge step was not taken in this iteration of the algorithm, we have that $|P^*| \geq 3$. We will charge the bracketing step to a link of P^* that is newly incident to A.

Claim 5.3. *After each bracketing step, there is a link of OPT that is newly incident to A.*

Proof. Let S' be the component containing S after the bracketing step. Since the bracketing step buys the two links ℓ_1, ℓ_2 leaving $I_t(S)$ whose endpoints have the smallest and largest index with respect to $v_0 = t$, respectively, it follows that $I_t(S')$ contains both endpoints of all links incident to $I_t(S)$. Hence, the first link of P^* has an endpoint in $I_t(S)$ and another endpoint in $I_t(S') \setminus I_t(S)$. Moreover, as P^* is the shortest path between S and t, only the first link of P^* has an endpoint in $I_t(S)$. Finally, observe that $I_t(S')$ cannot contain t, since $v_0 = t$ would then imply that one of the links chosen is incident to t, and so the algorithm would have instead taken a merge step. Thus, there exists a link $\ell^* \in P^*$ with one endpoint in $I_t(S') \setminus I_t(S)$ and the other outside of $I_t(S')$.

Observe that ℓ^* is not incident to S. It is incident to at least one of ℓ_1 and ℓ_2, and so it cannot be incident to any other active component since otherwise, the algorithm would have taken a merge step instead of a bracketing step. Therefore, ℓ^* is newly incident to A. □

Overall, there are at most OPT many bracketing steps, each costing two links, so the total cost of the algorithm is at most $8\text{OPT} + 2\text{OPT} = 10\text{OPT}$. □

5.2 An $\Omega(\log n)$ lower bound for OCycAP with cut-arrivals

As discussed previously in Section 1.2, there is no $o(\log n)$-competitive algorithm for OCycAP with cut-arrivals, as it captures general OCAP. We also give a simple direct proof that also rules out $o(\log n)$-competitive fractional algorithms below.

Theorem 5.4. *There is no fractional algorithm for OCycAP with cut arrivals that achieves a competitive ratio better than* $\log n$.

Proof. Let $R = (V, E)$ be a cycle with n nodes $v_0, \ldots, v_{n-1}$. Let $L = \binom{V}{2}$. Recall from Section 3.1 that a min cut C of R can be specified as an interval $[a, b]$ where $1 \leq a \leq b \leq n-1$ where $\{v_a, \ldots, v_b\}$ is one of the two sides of the cut.

Fix a fractional algorithm and let x_ℓ^t be the extent to which it purchased link ℓ in response to the first t cuts. WLOG, we assume that it is lazy, i.e., $x_\ell^t > 0$ only if ℓ covers a cut $C_{t'}$ for $t' \leq t$. The first cut is $C_1 = [1, n/2]$. Each subsequent cut C_t for $t > 1$ is obtained as follows. Let $C_{t-1} = [a_{t-1}, b_{t-1}]$ be the previous cut and $m_{t-1} = \lfloor \frac{a_{t-1}+b_{t-1}}{2} \rfloor$ be the median of $[a_{t-1}, b_{t-1}]$. Let

$A_{t-1} = [a_{t-1}, m_{t-1}]$ and $B_{t-1} = [m_{t-1}+1, b_{t-1}]$. If $\sum_{\ell:\,\ell \text{ covers } A_{t-1}} x_\ell^{t-1} \leq \sum_{\ell:\,\ell \text{ covers } B_{t-1}} x_\ell^{t-1}$ then $C_t = A_{t-1}$ otherwise $C_t = B_{t-1}$. There are a total of $r = \log_2 n - 1$ requests that form a nested family of intervals with the last request being a singleton $[a_r]$.

Observe that OPT $= 1$ as the link (v_0, v_{a_r}) satisfies all cuts. On the other hand, since x is lazy, we have that

$$\sum_{\ell:\,\ell \text{ covers } A_{t-1}} x_\ell^{t-1} + \sum_{\ell:\,\ell \text{ covers } B_{t-1}} x_\ell^{t-1} = \sum_{\ell:\,\ell \text{ covers } C_{t-1}} x_\ell^{t-1} \geq 1$$

so by our choice of C_t

$$\sum_{\ell:\,\ell \text{ covers } C_{t-1}} x_\ell^{t-1} - \sum_{\ell:\,\ell \text{ covers } C_t} x_\ell^{t-1} \geq 1/2.$$

Summing over t gives us that the cost of the fractional solution is $\Omega(\log n)$. □

Independent Work

During a workshop in August 2026, we discovered that another group consisting of Garg, Hommelsheim, and Lindermayr were working on Online Connectivity Augmentation, and that they had independently obtained an $O(1)$-competitive algorithm for OCycAP with node-arrivals. No techniques or other results were discussed, and we continued working independently following this point.

AI Disclosure

The authors used Gemini 3.8 Flash and Gemini 3.1 Pro-preview for minimal proofreading and editing. The authors also used ChatGPT 5.6 - 6.1 in the development of some of the algorithmic ideas; in particular, the connection to geometric hitting set and the constant-competitive algorithm for OCycAP with node-arrivals was discovered by ChatGPT 6, which also assisted us in adapting the methods of [BGK26] to our setting. ChatGPT 6 was also used to create tikz diagrams, for example in Figure 1. The authors verified the correctness and originality of all content, including references.

References

[Adj19] David Adjiashvili. "Beating Approximation Factor Two for Weighted Tree Augmentation with Bounded Costs". In: *ACM Transactions on Algorithms* 15.2 (2019), 19:1–19:26. DOI: 10.1145/3182395. URL: https://doi.org/10.1145/3182395.

[Alo+09] Noga Alon, Baruch Awerbuch, Yossi Azar, Niv Buchbinder, and Joseph Naor. "The Online Set Cover Problem". In: *SIAM Journal on Computing* 39.2 (2009), pp. 361–370. DOI: 10.1137/060661946. URL: https://doi.org/10.1137/060661946.

[AT14] Peyman Afshani and Konstantinos Tsakalidis. "Optimal Deterministic Shallow Cuttings for 3D Dominance Ranges". In: *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2014, pp. 1389–1398. DOI: 10.1137/1.9781611973402.102. URL: https://doi.org/10.1137/1.9781611973402.102.

[Bas+14] Manu Basavaraju, Fedor V. Fomin, Petr A. Golovach, Pranabendu Misra, M. S. Ramanujan, and Saket Saurabh. "Parameterized Algorithms to Preserve Connectivity". In: *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Proceedings, Part I*. Ed. by Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias. Vol. 8572. Lecture Notes in Computer Science. Springer, 2014, pp. 800–811. DOI: 10.1007/978-3-662-43948-7_66. URL: https://doi.org/10.1007/978-3-662-43948-7_66.

[BGK26] Sujoy Bhore, Anupam Gupta, and Amit Kumar. *Improved Online Hitting Set Algorithms for Structured and Geometric Set Systems*. 2026. arXiv: 2603.14293v1 [cs.DS]. URL: https://arxiv.org/abs/2603.14293v1.

[CG18a] Joseph Cheriyan and Zhihan Gao. "Approximating (Unweighted) Tree Augmentation via Lift-and-Project, Part I: Stemless TAP". In: *Algorithmica* 80.2 (2018), pp. 530–559. DOI: 10.1007/s00453-016-0270-4. URL: https://doi.org/10.1007/s00453-016-0270-4.

[CG18b] Joseph Cheriyan and Zhihan Gao. "Approximating (Unweighted) Tree Augmentation via Lift-and-Project, Part II". In: *Algorithmica* 80.2 (2018), pp. 608–651. DOI: 10.1007/s00453-017-0275-7. URL: https://doi.org/10.1007/s00453-017-0275-7.

[Cha+12] Timothy M. Chan, Elyot Grant, Jochen Könemann, and Malcolm Sharpe. "Weighted Capacitated, Priority, and Geometric Set Cover via Improved Quasi-Uniform Sampling". In: *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2012)*. Ed. by Yuval Rabani. Society for Industrial and Applied Mathematics, 2012, pp. 1576–1585. ISBN: 978-1-61197-210-8. DOI: 10.1137/1.9781611973099.125. URL: https://doi.org/10.1137/1.9781611973099.125.

[Che+25] Chandra Chekuri, Rhea Jain, Sepideh Mahabadi, and Ali Vakilian. "Streaming Algorithms for Network Design". In: *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2025)*. Ed. by Alina Ene and Eshan Chattopadhyay. Vol. 353. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025, 4:1–4:23. ISBN: 978-3-95977-397-3. DOI: 10.4230/LIPIcs.APPROX/RANDOM.2025.4. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.APPROX/RANDOM.2025.4.

[CN13] Nachshon Cohen and Zeev Nutov. "A $(1 + \ln 2)$-Approximation Algorithm for Minimum-Cost 2-Edge-Connectivity Augmentation of Trees with Constant Radius". In: *Theoretical Computer Science* 489–490 (June 2013), pp. 67–74. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2013.04.004. URL: https://doi.org/10.1016/j.tcs.2013.04.004.

[Coh+26] Vincent Cohen-Addad, Marina Drygala, Nathan Klein, and Ola Svensson. "A Strong Linear Programming Relaxation for Weighted Tree Augmentation". In: *Proceedings of the 58th Annual ACM Symposium on Theory of Computing (STOC 2026)*. Ed. by Aditya Bhaskara and Artur Czumaj. New York, NY, USA: Association for Computing Machinery, 2026, pp. 2176–2186. ISBN: 979-8-4007-2536-4. DOI: 10.1145/3798129.3800920. arXiv: 2603.29582 [cs.DS]. URL: https://doi.org/10.1145/3798129.3800920.

[CTZ21] Federica Cecchetto, Vera Traub, and Rico Zenklusen. "Bridging the Gap Between Tree and Connectivity Augmentation: Unified and Stronger Approaches". In: *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. Ed. by Samir Khuller and Virginia Vassilevska Williams. Association for Computing Machinery, 2021, pp. 370–383. DOI: 10.1145/3406325.3451086. URL: https://doi.org/10.1145/3406325.3451086.

[DKL76] Efim A. Dinitz, Alexander V. Karzanov, and Michael V. Lomonosov. "On the Structure of the System of Minimum Edge Cuts of a Graph". In: *Studies in Discrete Optimization*. Ed. by A. A. Fridman. In Russian. Moscow: Nauka, 1976, pp. 290–306. URL: https://www.alexander-karzanov.net/ScannedOld/76_cactus_transl.pdf.

[DST25] Minati De, Satyam Singh, and Csaba D. Tóth. "Online Hitting Sets for Disks of Bounded Radii". In: *33rd Annual European Symposium on Algorithms (ESA 2025)*. Ed. by Anne Benoit, Haim Kaplan, Sebastian Wild, and Grzegorz Herman. Vol. 351. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2025, 50:1–50:16. ISBN: 978-3-95977-395-9. DOI: 10.4230/LIPIcs.ESA.2025.50. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ESA.2025.50.

[ES14] Guy Even and Shakhar Smorodinsky. "Hitting Sets Online and Unique-Max Coloring". In: *Discrete Applied Mathematics* 178 (Dec. 2014), pp. 71–82. ISSN: 0166-218X. DOI: 10.1016/j.dam.2014.06.019. URL: https://doi.org/10.1016/j.dam.2014.06.019.

[ET76] Kapali P. Eswaran and Robert Endre Tarjan. "Augmentation Problems". In: *SIAM Journal on Computing* 5.4 (1976), pp. 653–665. DOI: 10.1137/0205044. URL: https://doi.org/10.1137/0205044.

[Eve+09] Guy Even, Jon Feldman, Guy Kortsarz, and Zeev Nutov. "A 1.8 Approximation Algorithm for Augmenting Edge-Connectivity of a Graph from 1 to 2". In: *ACM Transactions on Algorithms* 5.2 (Mar. 2009), 21:1–21:17. ISSN: 1549-6325. DOI: 10.1145/1497290.1497297. URL: https://doi.org/10.1145/1497290.1497297.

[Fio+18] Samuel Fiorini, Martin Groß, Jochen Könemann, and Laura Sanità. "Approximating Weighted Tree Augmentation via Chvátal-Gomory Cuts". In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Ed. by Artur Czumaj. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2018, pp. 817–831. ISBN: 978-1-61197-503-1. DOI: 10.1137/1.9781611975031.53. URL: https://doi.org/10.1137/1.9781611975031.53.

[FJ81] Greg N. Frederickson and Joseph F. JáJá. "Approximation Algorithms for Several Graph Augmentation Problems". In: *SIAM Journal on Computing* 10.2 (May 1981), pp. 270–283. DOI: 10.1137/0210019. URL: https://doi.org/10.1137/0210019.

[Gál+21] Waldo Gálvez, Fabrizio Grandoni, Afrouz Jabal Ameli, and Krzysztof Sornat. "On the Cycle Augmentation Problem: Hardness and Approximation Algorithms". In: *Theory of Computing Systems* 65.6 (2021), pp. 985–1008. DOI: 10.1007/s00224-020-10025-6. URL: https://doi.org/10.1007/s00224-020-10025-6.

[GKR12] Anupam Gupta, Ravishankar Krishnaswamy, and R. Ravi. "Online and Stochastic Survivable Network Design". In: *SIAM Journal on Computing* 41.6 (2012), pp. 1649–1672. ISSN: 0097-5397. DOI: 10.1137/09076725X. URL: https://doi.org/10.1137/09076725X.

[GKZ18] Fabrizio Grandoni, Christos Kalaitzis, and Rico Zenklusen. "Improved Approximation for Tree Augmentation: Saving by Rewiring". In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*. New York, NY, USA: Association for Computing Machinery, June 2018, pp. 632–645. ISBN: 978-1-4503-5559-9. DOI: 10.1145/3188745.3188898. URL: https://doi.org/10.1145/3188745.3188898.

[GS26] Mohit Garg and Aditya Subramanian. "Online Connectivity Augmentation". In: *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2026, Vancouver, BC, Canada, January 11-14, 2026*. Ed. by Kasper Green Larsen and Barna Saha. SIAM, 2026, pp. 85–104. DOI: 10.1137/1.9781611978971.4. URL: https://doi.org/10.1137/1.9781611978971.4.

[Jin+24] Ce Jin, Michael Kapralov, Sepideh Mahabadi, and Ali Vakilian. "Streaming Algorithms for Connectivity Augmentation". In: *51st International Colloquium on Automata, Languages, and Programming (ICALP 2024)*. Ed. by Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson. Vol. 297. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024, 93:1–93:20. ISBN: 978-3-

95977-322-5. DOI: 10.4230/LIPIcs.ICALP.2024.93. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.ICALP.2024.93.

[Kha+23] Arindam Khan, Aditya Lonkar, Saladi Rahul, Aditya Subramanian, and Andreas Wiese. "Online and Dynamic Algorithms for Geometric Set Cover and Hitting Set". In: *39th International Symposium on Computational Geometry (SoCG 2023)*. Ed. by Erin W. Chambers and Joachim Gudmundsson. Vol. 258. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023, 46:1–46:17. ISBN: 978-3-95977-273-0. DOI: 10.4230/LIPIcs.SoCG.2023.46. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.SoCG.2023.46.

[NUW22] Joseph Naor, Seeun William Umboh, and David P. Williamson. "Tight Bounds for Online Weighted Tree Augmentation". In: *Algorithmica* 84.2 (2022), pp. 304–324. DOI: 10.1007/s00453-021-00888-7. URL: https://doi.org/10.1007/s00453-021-00888-7.

[TZ22] Vera Traub and Rico Zenklusen. "Local Search for Weighted Tree Augmentation and Steiner Tree". In: *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Ed. by Joseph (Seffi) Naor and Niv Buchbinder. SIAM, 2022, pp. 3253–3272. DOI: 10.1137/1.9781611977073.128. URL: https://doi.org/10.1137/1.9781611977073.128.

[TZ23] Vera Traub and Rico Zenklusen. "A $(1.5+\varepsilon)$-Approximation Algorithm for Weighted Connectivity Augmentation". In: *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC)*. Ed. by Barna Saha and Rocco A. Servedio. ACM, 2023, pp. 1820–1833. DOI: 10.1145/3564246.3585122. URL: https://doi.org/10.1145/3564246.3585122.

[TZ25] Vera Traub and Rico Zenklusen. "Better-Than-2 Approximations for Weighted Tree Augmentation and Applications to Steiner Tree". In: *Journal of the ACM* 72.2 (Apr. 2025), pp. 1–40. DOI: 10.1145/3722101. URL: https://doi.org/10.1145/3722101.

A Proof of Theorem 4.8

Let $W \subset \mathbb{R}^3$ be a set of r points, and let $\mathcal{G}$ denote the family of octants of a fixed orientation. Without loss of generality, assume these are standard 3D dominance ranges of the form $Q_q = (-\infty, q_x] \times (-\infty, q_y] \times (-\infty, q_z]$ defined by a query vertex $q = (q_x, q_y, q_z)$. We wish to bound the number of distinct shallow traces $Q_q \cap W$ of size at most κ.

To do this, we rely on the existence of optimal shallow cuttings for 3D dominance ranges, as established by Afshani and Tsakalidis [AT14]. A *κ-shallow cutting* for W is a collection of spatial cells (regions) $\mathcal{C}$ with the following properties:

1. **Covering:** The union of the cells in $\mathcal{C}$ completely covers the κ-shallow region of $\mathbb{R}^3$. That is, for any query q such that $|Q_q \cap W| \leq \kappa$, the vertex q lies in at least one cell $C \in \mathcal{C}$.

2. **Size:** The number of cells in the cutting is $|\mathcal{C}| = O(r/\kappa)$.

3. **Conflict bound:** For each cell $C \in \mathcal{C}$, let $W_C \subseteq W$ be the subset of points that can be dominated by any query $q \in C$. The cutting guarantees that $|W_C| = O(\kappa)$.

Every query octant Q_q that captures at most κ points has its vertex q located inside some cell $C \in \mathcal{C}$. Consequently, the trace of Q_q on W is entirely a subset of W_C, meaning $Q_q \cap W = Q_q \cap W_C$. Thus, to bound the total number of distinct shallow traces of size at most κ over all of $\mathbb{R}^3$, it suffices to bound the number of distinct traces that can be formed within each cell C, and sum this over all cells in $\mathcal{C}$.

For a fixed cell C, the number of points interacting with queries in C is $|W_C| = O(\kappa)$. Any distinct trace generated by an octant Q_q can be uniquely identified by its minimal bounding octant. Because an octant is defined by exactly 3 orthogonal planes, this minimal bounding octant is pinned down by at most 3 extreme points from the trace (the points with the maximum x, y, and z coordinates, respectively).

Therefore, the number of distinct traces that can be formed on W_C is bounded by the number of subsets of W_C of size at most 3. This gives:

$$\text{Number of traces in } C \leq \sum_{i=1}^{3} \binom{|W_C|}{i} = O(|W_C|^3) = O(\kappa^3).$$

Summing this bound over all $|\mathcal{C}| = O(r/\kappa)$ cells in the shallow cutting, the total number of distinct traces of size at most κ is at most:

$$\sum_{C \in \mathcal{C}} O(\kappa^3) = O\left(\frac{r}{\kappa}\right) \cdot O(\kappa^3) = O(r\kappa^2).$$

Letting C_{dom} be the absolute constant absorbing the asymptotic notation, we conclude that the number of distinct nonempty traces of size at most κ is at most $C_{\text{dom}} r\kappa^2$, completing the proof. $\square$