THEORY OF COMPUTING, Volume 19 (2), 2023, pp. 1–23
www.theoryofcomputing.org

On Solving Reachability in Grid Digraphs using a Pseudoseparator

Rahul Jain   Raghunath Tewari

Received January 13, 2020; Revised May 25, 2022; Published August 28, 2023

Abstract. The reachability problem asks to decide if there exists a path from one vertex to another in a digraph. In a grid digraph, the vertices are the points of a two-dimensional square grid, and an edge can occur between a vertex and its immediate horizontal and vertical neighbors only.

Asano and Doerr (CCCG'11) presented the first simultaneous time-space bound for reachability in grid digraphs by solving the problem in polynomial time and $O(n^{1/2+\epsilon})$ space. In 2018, the space complexity was improved to $\tilde{O}(n^{1/3})$ by Ashida and Nakagawa (SoCG'18).

In this paper, we show that there exists a polynomial-time algorithm that uses $O(n^{1/4+\epsilon})$ space to solve the reachability problem in a grid digraph containing n vertices. We define and construct a new separator-like device called pseudoseparator to develop a divide-and-conquer algorithm. This algorithm works in a space-efficient manner to solve reachability.

A conference version of this paper appeared in the Proceedings of the 39th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2019) [10].

ACM Classification: G.2.2

AMS Classification: 68Q25

Key words and phrases: digraph reachability, grid digraph, graph algorithm, sublinear space algorithm

 DOI: 10.4086/toc.2023.v019.a002

1 Introduction

Let s and t be vertices of a given directed graph (digraph). The problem of digraph reachability is to decide if there is a path from s to t. This problem has many applications in the field of algorithms and computational complexity theory. Many algorithms for network-related problems use it as a subroutine. Digraph reachability is NL-complete and thus captures the complexity of nondeterministic logarithmic space. Hence designing better algorithms for this problem is of utmost importance to the theory of computing.

Standard traversal algorithms such as DFS and BFS give a linear-time algorithm for this problem, but they require linear space. Savitch's divide-and-conquer algorithm solves reachability in $O(\log^2 n)$ space. However, as a tradeoff, it requires $n^{O(\log n)}$ time [14]. Hence it is natural to ask whether we can get the best of both worlds and design an algorithm for digraph reachability that runs in polynomial time and uses *polylogarithmic* space. Wigderson asked a relaxed version of this question in his survey, whether digraph reachability can be solved by a polynomial-time algorithm that uses $O(n^{1-\epsilon})$ space [16].

Barnes et al. showed that digraph reachability can be decided simultaneously in $n/2^{\Theta(\sqrt{\log n})}$ space and polynomial time [6]. Although this algorithm gives a sublinear space bound, it still does not answer Wigderson's question.

Undirected graphs can be considered as a specific type of digraphs, where adjacency is a symmetric relation. Reachability in undirected graphs can be solved in logspace [13]. Also, for certain classes of digraphs, where the underlying undirected graph is topologically restricted, we know of polynomial time algorithms with space complexity better than linear. Imai et al. presented a polynomial-time algorithm that solves reachability for planar digraphs using $O(n^{1/2+\epsilon})$ space for any $\epsilon > 0$ [9]. Later, Asano et al. presented a polynomial-time algorithm whose space complexity was $\tilde{O}(n^{1/2})$ for the same problem [4]. For digraphs of higher genus, Chakraborty et al. presented a polynomial-time algorithm that uses $\tilde{O}(n^{2/3}g^{1/3})$ space. Their algorithm additionally requires an embedding of the underlying undirected graph on a surface of genus g, as part of the input [7]. They also gave an $\tilde{O}(n^{2/3})$-space algorithm for H-minor-free digraphs which requires a tree decomposition of the underlying undirected graph as part of the input and $O(n^{1/2+\epsilon})$-space algorithms for $K_{3,3}$-minor-free digraphs and for K_5-minor-free digraphs.

For layered planar digraphs Chakraborty and Tewari showed that for every $\epsilon > 0$, there is an $O(n^{\epsilon})$-space algorithm [8]. Stolee and Vinodchandran presented a polynomial-time algorithm that, for any $\epsilon > 0$ solves *reachability* in an acyclic digraph with $O(n^{\epsilon})$ sources and embedded on the surface of genus $O(n^{\epsilon})$ using $O(n^{\epsilon})$ space [15]. For unique-path digraphs Kannan et al. presented an $O(n^{\epsilon})$-space, polynomial-time algorithm [11].

In this paper, grid digraphs are our concern. Grid digraphs are a subclass of planar digraphs. In a grid digraph, the vertices are the points of an $m \times m$ grid. The edges can only occur between a vertex and its immediate vertical and horizontal neighbours. We know that reachability in planar digraphs can be reduced to reachability in grid digraphs in logarithmic space [1]. The reduction, however, causes at least a quadratic blow-up in the size of the digraph. In this paper, we study the simultaneous time–space complexity of reachability in grid graphs.

Asano and Doerr presented a polynomial-time algorithm that uses $O(n^{1/2+\epsilon})$ space for solving reachability in grid digraphs [3]. Ashida and Nakagawa presented an algorithm with improved space complexity of $\widetilde{O}(n^{1/3})$ [5]. The latter algorithm proceeds by first dividing the input grid digraph into subgrids. It then uses a gadget to transform each subgrid into a planar digraph, making the whole of the resultant digraph planar. Finally, it uses the planar reachability algorithm of Imai et al. [9] as a subroutine to get the desired space bound.

In this paper, we present a $O(n^{1/4+\epsilon})$ space and polynomial-time algorithm for grid digraph reachability, thereby significantly improving the space bound of Ashida and Nakagawa.

Theorem 1.1 (Main Theorem)**.** *For every $\epsilon > 0$, there exists a polynomial-time algorithm that solves reachability in an n-vertex grid digraph using $O(n^{1/4+\epsilon})$ space.*

The approach of Ashida and Nakagawa [5] is to reduce the size of the input n-vertex digraph to a digraph of size $O(n^{2/3})$. Their new graph preserves reachability between vertices and is planar. They use the planar-reachability algorithm of Asano et al. [4]. Our approach is slightly different. We convert the input digraph into an *auxiliary digraph* of size $O(n^{1/2+\alpha/2})$ for arbitrarily small α. The auxiliary digraph is created by dividing the given grid digraph into subgrids and replacing paths in each subgrid with a single edge between the boundary vertices. While our auxiliary digraph preserves reachability, it is not planar; and hence we cannot use Asano et al.'s algorithm directly. The auxiliary digraph comes with a drawing with the *crossing property*, that is, if two edges, e and f, cross each other in this drawing, there necessarily exist two more edges, one from the tail of e to the head of f and the other from the tail of f to the head of e, in the digraph. This property allows us to use a device that we call *pseudoseparator* to solve reachability in it. A pseudoseparator designates a set of vertices, a set of edges and a set of components of the digraph, such that a path from one component to another necessarily either takes a vertex of the pseudoseparator or crosses one of the edges of the pseudoseparator in the drawing. We finally solve the problem by recursively solving each of these components and using a traversal algorithm. Due to the crossing property, we are required to store only a small number of vertices for performing the traversal, thereby saving space.

In Section 2, we state the definitions and notation that we use in this paper. In Section 3, we define the auxiliary digraph and state various properties of it that we use later. In Section 4, we discuss the concept of the *pseudoseparator*. We give its formal definition and show how to compute the *pseudoseparator* efficiently. In Section 5, we give the algorithm to solve reachability in an auxiliary digraph and prove its correctness. Finally in Section 6 we use the algorithm of Section 5 to give an algorithm to decide reachability in grid digraphs and thus prove Theorem 1.1.

2 Preliminaries

Let $[n]$ denote the set $\{0, 1, 2, \ldots, n\}$. We denote the vertex set of a digraph G by $V(G)$ and its edge set by $E(G)$. We assume that the vertices of a digraph are *indexed* by integers from 1 to $|V(G)|$. For a subset U of $V(G)$, we denote the sub-digraph of G induced by the vertices of U as $G[U]$ and we denote the sub-digraph of G induced by the vertices $V(G) \setminus U$ as $G \setminus U$. For a digraph G, we write $\mathsf{cc}(G)$ to denote the set of all connected components in the underlying undirected graph of G. By the *underlying undirected graph*, we mean the graph formed by symmetrizing the adjacency relation. To do this, we consider each directed edge in the digraph and create an undirected edge between the corresponding vertices. Henceforth, whenever we talk about connected components, we will mean the connected components of the underlying undirected graph. In a *drawing* of a graph in the plane we map each vertex to a point in the plane and each edge to a simple arc whose endpoints coincide with the images of the end vertices of the edge. Moreover, the image of a vertex does not belong to the interior of an arc corresponding to any edge. We say that a graph is *planar* if there exists a drawing of that graph on a plane such that no two arcs corresponding to the edges of the graph intersect except at the endpoints. Such a drawing is called a *planar embedding*.

A *planar digraph* is a digraph whose underlying undirected graph is planar. Similarly, a *drawing of a digraph* is defined as the drawing of its underlying undirected graph.

We will represent a planar graph by describing the cyclic ordering of a graph's edges around each vertex. We note that the results in [2] and [13] combine to a *deterministic logarithmic space* algorithm that decides whether a given graph is planar, and if it is, outputs a planar embedding. Hence, when dealing with planar graphs, we will assume without loss of generality that we have a planar embedding whenever required. An $m \times m$ *grid digraph* is a directed graph whose set of vertices is $[m] \times [m] = \{0, 1, \ldots, m\} \times \{0, 1, \ldots, m\}$ so that if $((i_1, j_1), (i_2, j_2))$ is an edge then $|i_1 - i_2| + |j_1 - j_2| = 1$. In other words, we start with an undirected $m \times m$ grid graph. Then, we remove some of the edges and assign directions to the remaining edges while keeping all the vertices. This process results in the formation of an $m \times m$ grid digraph, which has a total of $n = (m+1)^2$ vertices. It follows from the definition that grid digraphs are planar.

We will work with $m \times m$ grid digraphs where $m = O(n^{1/2})$. Hence, a space complexity of $O(m^{1/2+\epsilon})$ will translate into a space complexity of $O(n^{1/4+\epsilon'})$ as required.

3 Auxiliary graph

Let G be an $m \times m$ grid digraph. We divide G into $m^{2\alpha}$ subgrids such that each subgrid is a $m^{1-\alpha} \times m^{1-\alpha}$ grid. Formally, for $0 < \alpha < 1$ and $1 \le i, j \le m^{\alpha}$, the (i, j)-th *subgrid* of G, denoted $G[i, j]$, is the sub-digraph of G induced by the set of vertices, $V(G[i, j])$, where $V(G[i, j])$ is

$$\{(i', j') \mid (i-1) \cdot m^{1-\alpha} \le i' \le i \cdot m^{1-\alpha} \text{ and } (j-1) \cdot m^{1-\alpha} \le j' \le j \cdot m^{1-\alpha}\}\,.$$

For ease of presentation, we will assume without loss of generality that variables like m and α are such that the values like m^{α} and $m^{1-\alpha}$ are integers.

We define the auxiliary digraph $\mathsf{Aux}_{\alpha}(G)[i, j]$ as follows. The vertex set of $\mathsf{Aux}_{\alpha}(G)[i, j]$ is the set of vertices on the *boundary* of $G[i, j]$. That means, $V(\mathsf{Aux}_{\alpha}(G)[i, j])$ is

$$\begin{aligned}&\{(i', j') \mid (i', j') \in V(G[i, j]) \text{ and}\\ &((i-1) \cdot m^{1-\alpha} = i' \text{ or } i \cdot m^{1-\alpha} = i' \text{ or } (j-1) \cdot m^{1-\alpha} = j' \text{ or } j \cdot m^{1-\alpha} = j')\}\,.\end{aligned}$$

For two vertices u, v in $\mathsf{Aux}_{\alpha}(G)[i, j]$, (u, v) is an edge in $\mathsf{Aux}_{\alpha}(G)[i, j]$ if there is a path from u to v in the subgrid $G[i, j]$. We draw $\mathsf{Aux}_{\alpha}(G)[i, j]$ on an Euclidean plane by mapping vertex (x, y)

to the point with coordinates (x, y). The points corresponding to vertices of $\mathsf{Aux}_\alpha(G)[i, j]$ thus lie on a square. We use a straight line segment to represent the edge if u and v do not lie on a single side of this square. and an arc inside the square to represent it otherwise.

We define the *α-auxiliary digraph*, $\mathsf{Aux}_\alpha(G)$ as follows. The vertex set of $\mathsf{Aux}_\alpha(G)$, $V(\mathsf{Aux}_\alpha(G)) = \{(i, j) \mid i = k \cdot m^{1-\alpha} \text{ or } j = l \cdot m^{1-\alpha}, \text{ such that } 0 \le k, l \le m^\alpha\}$. The edges of $\mathsf{Aux}_\alpha(G)$ are the edges of $\mathsf{Aux}_\alpha(G)[i, j]$ taken over all pairs (i, j). Note that $\mathsf{Aux}_\alpha(G)$ might have parallel edges, since an edge on a side of a block might be in the adjacent block as well. In such cases we preserve both the edges, in their different blocks of $\mathsf{Aux}_\alpha(G)$ in the drawing of $\mathsf{Aux}_\alpha(G)$ on the plane. Figure 1 contains an example of a grid digraph partitioned into subgrids and its corresponding auxiliary digraph. Since each block $\mathsf{Aux}_\alpha(G)[i, j]$ contains $4m^{1-\alpha}$ vertices, the total number of vertices in $\mathsf{Aux}_\alpha(G)$ is at most $4m^{1+\alpha}$.

Our algorithm for reachability first constructs $\mathsf{Aux}_\alpha(G)$ by solving each of the $m^{1-\alpha} \times m^{1-\alpha}$ grids recursively. It then uses a polynomial-time subroutine to decide reachability in $\mathsf{Aux}_\alpha(G)$. Note that we do not store the digraph $\mathsf{Aux}_\alpha(G)$ explicitly since that would require too much space. Rather we solve a subgrid recursively whenever the subroutine queries for an edge in that subgrid of $\mathsf{Aux}_\alpha(G)$.

Our strategy is to develop a polynomial-time algorithm which solves reachability in $\mathsf{Aux}_\alpha(G)$ using $\tilde{O}(\tilde{m}^{1/2+\beta/2})$ space where $\tilde{m}$ is the number of vertices in $\mathsf{Aux}_\alpha(G)$. As discussed earlier, $\tilde{m}$ is at most $4m^{1+\alpha}$. Hence, the main algorithm requires $\tilde{O}(m^{1/2+\beta/2+\alpha/2+\alpha\beta/2})$ space. For a fixed constant $\epsilon > 0$, we can pick $\alpha > 0$ and $\beta > 0$ such that the space complexity becomes $O(m^{1/2+\epsilon})$.

Figure 1: A grid digraph G divided into subgrids and its corresponding auxiliary digraph $\mathsf{Aux}_\alpha(G)$

3.1 Properties of the auxiliary digraph

In the following definition, we give ordered labelling to the vertices of a block of the auxiliary digraph. We define the labelling with respect to some vertex in the block.

Definition 3.1. Let G be a $m \times m$ grid digraph, $\ell = \mathsf{Aux}_\alpha(G)[i, j]$ be a block of $\mathsf{Aux}_\alpha(G)$ and $v = (x, y)$ be a vertex in $\mathsf{Aux}_\alpha(G)[i, j]$. Let $t = m^{1-\alpha}$. We define a cyclic permutation c_ℓ on the vertex set of $\mathsf{Aux}_\alpha(G)[i,j]$ as follows:

$$c_l(v) = \begin{cases} (x+1,y) & \text{if } x < (i+1)t \text{ and } y = jt \\ (x,y+1) & \text{if } x = (i+1)t \text{ and } y < (j+1)t \\ (x-1,y) & \text{if } x > it \text{ and } y = (j+1)t \\ (x,y-1) & \text{if } x = it \text{ and } y > jt \end{cases}$$

For any non-negative integer r, we define c_ℓ^r inductively as follows. For $r = 0$, $c_l^r(v) = v$ and otherwise we have $c_l^{r+1}(v) = c_l(c_l^r(v))$.

Note that the permutation c_ℓ can be seen as a counter-clockwise shift. Also note that for a block ℓ and vertices v and w in it, we can write v as $c_l^p(w)$ where p is smallest non-negative integer for which $c_l^p(w) = v$. Next we formalize what it means to say that two edges of the auxiliary digraph cross each other.

Definition 3.2. Let G be a grid digraph and ℓ be a block of $\mathsf{Aux}_\alpha(G)$. For two distinct edges e and f in the block, such that $e = (v, c_l^p(v))$ and $f = (c_l^q(v), c_l^r(v))$. We say that edges e and f cross each other if $\min(q,r) < p < \max(q,r)$.

Note the definition of cross given above is symmetric. That is, if edges e and f cross each other then f and e must cross each other as well. For an edge $f = (c_l^q(v), c_l^r(v))$, we define $\overleftarrow{f} = (c_l^r(v), c_l^q(v))$ and call it the *reverse* of f. We also note that if e and f cross each other, then e and $\overleftarrow{f}$ also cross each other.

In Lemma 3.3 we state an equivalent condition of the crossing of two edges. In Lemmas 3.5 and 3.6 we state some specific properties of the auxiliary digraph that we use later.

Lemma 3.3. *Let G be a grid digraph and ℓ be a block of $\mathsf{Aux}_\alpha(G)$. Let w be an arbitrary vertex in the block ℓ and $e = (c_l^p(w), c_l^q(w))$ and $f = (c_l^r(w), c_l^s(w))$ be two distinct edges in ℓ. Then e and f cross each other if and only if either of the following two conditions hold:*

- $\min(p,q) < \min(r,s) < \max(p,q) < \max(r,s)$
- $\min(r,s) < \min(p,q) < \max(r,s) < \max(p,q)$

Proof. We prove that if $\min(p,q) < \min(r,s) < \max(p,q) < \max(r,s)$ then e and f cross each other. We let $p < r < q < s$. Other cases can be proved by reversing appropriate edges. We thus have integers $r_1 = r - p$, $q_1 = q - p$ and $s_1 = s - p$. Clearly, $r_1 < q_1 < s_1$. Let $v = c_l^p(w)$. Thus we have $e = (v, c_l^q(w)) = (v, c_l^{q_1}(c_l^p(w))) = (v, c_l^{q_1}(v))$ and $f = (c_l^r(w), c_l^s(w)) = (c_l^{r_1}(c_l^p(w)), c_l^{s_1}(c_l^p(w))) = (c_l^{r_1}(v), c_l^{s_1}(v))$ The proof for the second condition is similar.

Now, we prove that if $e = (c_l^p(w), c_l^q(w))$ and $f = (c_l^r(w), c_l^s(w))$ cross each other then either of the given two condition holds. We assume that p is the smallest integer among p, q, r and s. Other cases can be proved similarly. Now, let $v = c_l^p(w)$. We thus have integers $q_1 = q - p, r_1 = r - p$ and $s_1 = s - p$ such that $e = (v, c_l^{q_1}(v))$ and $f = (c_l^{r_1}(v), c_l^{s_1}(v))$. Since e and f cross each other, we have $\min(r_1, s_1) < q_1 < \max(r_1, s_1)$. Thus $\min(r_1 + p, s_1 + p) < q_1 + p < \max(r_1 + p, s_1 + p)$. It follows that $\min(r, s) < q < \max(r, s)$. Since we assumed p to be smallest integer among p, q, r and s; we have $\min(p, q) < \min(r, s) < \max(p, q) < \max(r, s)$, thus proving the lemma. □

We see that we can draw an auxiliary digraph on a plane such that the arcs corresponding to two of its edges intersect if and only if the corresponding edges cross each other. Henceforth, we will work with such a drawing.

Definition 3.4. Let G be a grid digraph and ℓ be a block of $\mathsf{Aux}_\alpha(G)$. For a vertex v and edges f, g such that $f = (c_l^q(v), c_l^r(v))$ and $g = (c_l^s(v), c_l^t(v))$, we say that f is closer to v than g if $\min(q, r) < \min(s, t)$.

We say f is *closest* to v if there exists no other edge f' which is closer to v than f.

Lemma 3.5. *Let G be a grid digraph and $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ be two edges in $\mathsf{Aux}_\alpha(G)$. If e_1 and e_2 cross each other, then $\mathsf{Aux}_\alpha(G)$ also contains the edges (u_1, v_2) and (u_2, v_1).*

Proof. Let $e_1 = (v, c_l^p(v))$ and $e_2 = (c_l^q(v), c_l^r(v))$ be two edges that cross each other in $\mathsf{Aux}_\alpha(G)$. Let ℓ be the block of $\mathsf{Aux}_\alpha(G)$ to which e_1 and e_2 belong. Consider the subgrid of G which is solved to construct the block ℓ. Since the edge e_1 exists in block ℓ, there exists a path P from v to $c_l^p(v)$ in the underlying subgrid. This path P divides the subgrid into two parts such that the vertices $c_l^q(v)$ and $c_l^r(v)$ belong to different parts of the subgrid. Thus, a path between $c_l^q(v)$ and $c_l^r(v)$ necessarly take a vertex of path P. Hence, there is a path from v to $c_l^r(v)$ and a path from $c_l^p(v)$ to $c_l^r(v)$. Thus the lemma follows. □

Figure 2: Edge crossings in an auxiliary grid. On the left, there is one block of the auxiliary digraph that contains edges that cross. The dotted edges are the ones whose existence is made necessary by Lemma 3.5. On the right, a subgrid which results in the auxiliary block on the left.

Lemma 3.6. *Let G be a grid digraph and $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$ be two edges in $\mathsf{Aux}_\alpha(G)$. If e_1 and e_2 cross a certain edge $f = (x, y)$, and e_1 is closer to x than e_2, then the edge (u_1, v_2) is also in $\mathsf{Aux}_\alpha(G)$.*

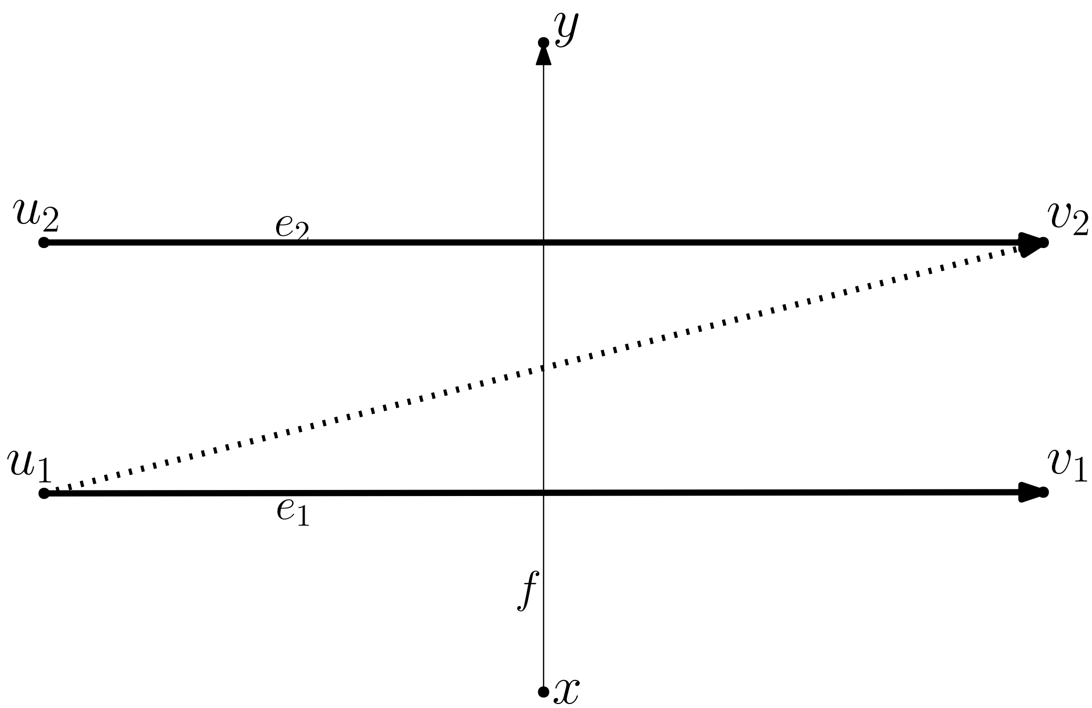

Figure 3: Illustration of Lemma 3.6

Proof. Let $f = (v, c_\ell^p(v))$, $e_1 = (c_\ell^q(v), c_\ell^r(v))$ and $e_2 = (c_\ell^s(v), c_\ell^t(v))$. If $c_\ell^q(v) = c_\ell^s(v)$ then the lemma trivially follows. Otherwise, we have two cases to consider:

Case 1 (e_1 crosses e_2): In this case, we will have $(c_\ell^q(v), c_\ell^t(v))$ in $\mathsf{Aux}_\alpha(G)$ by Lemma 3.5.

Case 2 (e_1 does not cross e_2): In this case, we have $\min(q,r) < \min(s,t) < p < \max(s,t) < \max(q,r)$. Since e_1 crosses f, we have the edge $(c_\ell^q(v), c_\ell^p(v))$ in $\mathsf{Aux}_\alpha(G)$ by Lemma 3.5. This edge will cross e_2. Hence $(c_\ell^q(v), c_\ell^t(v))$ is in $\mathsf{Aux}_\alpha(G)$. □

4 Pseudoseparators in a grid digraph

Imai et al. used a separator construction to solve the reachability problem in planar digraphs [9]. A separator is a set of vertices whose removal disconnects the graph into *small components*. The class of grid digraphs is a subclass of planar digraphs. Grid graphs are known to have small separators. However, for a grid digraph G, the underlying undirected graph of $\mathsf{Aux}_\alpha(G)$ might not have a small separator.

An essential property of a separator is that, for any two vertices, a path between the vertices must contain a separator vertex if the vertices lie in two different components with respect to the separator. This property can then be used to design a divide and conquer algorithm for reachability where only separator vertices need to be marked and stored for traversal. Lemma 3.6

allows us to use edges as well since, at most, one vertex on each side of an edge needs to be stored for traversal (the visited-vertex closest to the tail of the edge). Hence, we can use a slightly weaker structure in place of a separator. We construct a structure called pseudoseparator (see Definition 4.1). It is essentially a set of vertices and edges that can be used to divide the digraph into components, such that a path from one component to another either takes a vertex of the pseudoseparator or crosses an edge of the pseudoseparator.

For a digraph $H = (V_1, E_1)$ given along with its drawing, and a sub-digraph $C = (V_2, E_2)$ of H, define the digraph $H \diamond C = (V_3, E_3)$ as $V_3 = V_1 \setminus V_2$ and $E_3 = E_1 \setminus \{e \in E_1 \mid \exists f \in E_2, e \text{ crosses } f\}$. We note that the digraph H we will be working with throughout the article will be a sub-digraph of an auxiliary digraph. Hence it will always come with a drawing.

Definition 4.1. Let G be a grid digraph and H be an induced sub-digraph of $\mathsf{Aux}_\alpha(G)$ with h vertices. Let $f : \mathbb{N} \to \mathbb{N}$ be a function. A sub-digraph C of H is said to be an $f(h)$-pseudoseparator of $\mathsf{Aux}_\alpha(G)$ if the size of every connected component in $\mathsf{cc}(H \diamond C)$ is at most $f(h)$. The size of C is the number of vertices plus the number of edges of C.

For an inducedsub-digraph H of $\mathsf{Aux}_\alpha(G)$, an $f(h)$-pseudoseparator is a sub-digraph C of H that has the property that, if we remove the vertices as well as all the edges that cross one of the edges of the pseudoseparator, the digraph gets disconnected into small pieces. Moreover for every edge e in H, if there exists distinct sets U_1 and U_2 in $\mathsf{cc}(H \diamond C)$ such that one of the endpoints of e is in U_1 and the other is in U_2, then there exists an edge f in C such that e crosses f. Hence any path in H that connects two vertices in different components of $H \diamond C$ must either contain a vertex of C or must contain an edge that crosses an edge of C. We divide the digraph using this pseudoseparator and give an algorithm that recursively solves each sub-digraph and then combines their solution efficiently.

4.1 Constructing a pseudoseparator

Before working out the details below, we briefly comment on how to construct a pseudoseparator of an inducedsub-digraph H of $\mathsf{Aux}_\alpha(G)$. Ideally, we would have liked to have a set of cycles of H such that the surfaces obtained by cutting along these cycles have a bounded number of vertices on it.However, such cycles might not exist. We instead pick a maximal subset of edges from H so that no two edges cross (see Definition 4.2 and Lemma 4.3). Then we triangulate the resulting graph. We show that this triangulated graph has the required cycles and can be found using Imai et al.'s separator algorithm. Call the triangulated graph $\widehat{H}$ and the separator vertices as S. Since the cycles might have triangulated edges, we will add at most a constant number of vertices and edges from H to *shield* these triangulated edges. The vertex set of pseudoseparator of H will thus contain all the vertices of S and at most four additional vertices for each edge of $\widehat{H}[S]$ that is not in H. The edge set of pseudoseparator of H will contain all the edges of H which are also in $\widehat{H}[S]$ and at most four additional edges for each edge of $\widehat{H}[S]$ that is not in H.

Definition 4.2. Let G be a grid digraph and H be an induced sub-digraph of $\mathsf{Aux}_\alpha(G)$. We define $\mathsf{planar}(H)$ as a sub-digraph of H. The vertex set of $\mathsf{planar}(H)$ is same as that of H. For an edge $e \in H$, let ℓ be the block to which e belongs and let w be the lowest indexed vertex in that block.

Then $e = (c^i_\ell(w), c^j_\ell(w))$ is in $\mathsf{planar}(H)$ if there exists no other edge $f = (c^x_\ell(w), c^y_\ell(w))$ in H such that $\min(x, y) < \min(i, j) < \max(x, y) < \max(i, j)$.

In Lemma 4.3 we show that the digraph $\mathsf{planar}(H)$ is indeed planar. We also prove that the subset of edges that we have picked from H is a maximal subset such that no two edges cross.

Lemma 4.3. *Let G be a grid digraph and H be an induced sub-digraph of $\mathsf{Aux}_\alpha(G)$. No two edges of $\mathsf{planar}(H)$ cross each other. Moreover, for any edge e in H that is not in $\mathsf{planar}(H)$, there exists another edge in $\mathsf{planar}(H)$ that crosses e.*

Proof. Let ℓ be a block of $\mathsf{Aux}_\alpha(G)$ and w be the smallest index vertex of ℓ. Let $e = (c^p_\ell(w), c^q_\ell(w))$ and $f = (c^r_\ell(w), c^s_\ell(w))$ be two edges of H that cross. We have, by Lemma 3.3, that either $\min(p, q) < \min(r, s) < \max(p, q) < \max(r, s)$ or $\min(r, s) < \min(p, q) < \max(r, s) < \max(p, q)$. Hence, by our construction of $\mathsf{planar}(H)$, atmost one of e and f belongs to it. Thus no two edges of $\mathsf{planar}(H)$ cross.

For the second part, we will prove by contradiction. Let us assume that there exists edges in H which is not in $\mathsf{planar}(H)$ and also not crossed by an edge in $\mathsf{planar}(H)$. We pick edge $e = (c^p_\ell(w), c^q_\ell(w))$ from them such that $\min(p, q)$ of that edge is minimum. Since this edge is not in $\mathsf{planar}(H)$, we have by definition, an edge $f = (c^r_\ell(w), c^s_\ell(w))$ such that $\min(r, s) < \min(p, q) < \max(r, s) < \max(p, q)$. We pick the edge f for which $\min(r, s)$ is minimum. Now, since this edge f is not in $\mathsf{planar}(H)$, we have another edge $g = (c^i_\ell(w), c^j_\ell(w))$ in $\mathsf{planar}(H)$ such that $\min(i, j) < \min(r, s) < \max(i, j) < \max(r, s)$. We pick g such that $\min(i, j)$ is minimum and break ties by picking one whose $\max(i, j)$ is maximum. Now, we have the following cases:

Case 1 ($i < r < j < s$): Note that H is an induced sub-digraph of $\mathsf{Aux}_\alpha(G)$. Thus, in this case, the edge $(c^i_\ell(w), c^s_\ell(w))$ will be in H due to Lemma 3.5. Since $i < p < s < q$, and $i < \min(r, s)$, this will contradict the way in which edge f was chosen.

Case 2 ($i < s < j < r$): In this case, the edge $(c^r_\ell(w), c^j_\ell(w))$ will be in H due to Lemma 3.5. This edge will cross e and hence not be in $\mathsf{planar}(H)$. Thus, we have an edge $g' = (c^{i'}_\ell(w), c^{j'}_\ell(w))$ in $\mathsf{planar}(H)$ such that $\min(i', j') < j < \max(i', j') < r$. We will thus have two subcases.

Case 2a ($i < \min(i', j')$): Here, we will have $i < \min(i', j') < j < \max(i', j')$. Hence this edge will cross g giving a contradiction to the first part of this Lemma.

Case 2b ($\min(i', j') \leq i$): Here, this edge should have been chosen instead of g contradicting our choice of g.

The analysis of two remaining cases where $j < s < i < r$ and $j < r < i < s$ are similar to Cases 1 and 2 respectively. □

Definition 4.4. Let G be a grid digraph and H be an induced sub-digraph of $\mathsf{Aux}_\alpha(G)$. The graph $\widehat{H}$ is formed by first running Algorithm 1 on H and then triangulating the underlying undirected graph of the result

Input: An induced sub-digraph H of $\mathsf{Aux}_\alpha(G)$

Output all the vertices of H;
Output all the edges of $\mathsf{planar}(H)$;
foreach *block* ℓ *in* H **do**
  foreach *vertex* v *of* H *in* ℓ **do**
    $p \leftarrow$ the smallest positive integer such that $c_\ell^p(v) \in V(H)$;
    if $(v, c_\ell^p(v)) \notin E(\mathsf{planar}(H))$ **then**
      Output the edge $(v, c_\ell^p(v))$;
    end
  end
end

Algorithm 1: Completing the boundary of $\mathsf{planar}(H)$

Note that Algorithm 1 completes the boundary of each block. Thus, for any block, the vertices in its boundary form a simple cycle. Since the interiors of these cycles are disjoint, there is no edge that is drawn through more than one block in $\widehat{H}$. Thus, each edge of $\widehat{H}$ lies either entirely inside *one* of the blocks, or lies completely outside the whole $m \times m$ grid.

Now, for each of those edges added to $\widehat{H}$ as part of triangulation that is inside one of the blocks, we define a set of at most four *shield* edges.

Definition 4.5. Let G be a grid digraph and H be an induced sub-digraph of $\mathsf{Aux}_\alpha(G)$. Let $e = (v, w)$ be an edge in $\widehat{H}$ such that e is inside one of the blocks and e is not in $\mathsf{planar}(H)$. Let p and q be integers such that $w = c_\ell^p(v)$ and $v = c_\ell^q(w)$.

- Let p_1 be the largest integer such that $p_1 < p$ and an edge e_1 with endpoints v and $c_\ell^{p_1}(v)$ exists in $\mathsf{planar}(H)$. e_1 is undefined if no such integer exists.
- Let p_2 be the smallest integer such that $p_2 > p$ and an edge e_2 with endpoints v and $c_\ell^{p_2}(v)$ exists in $\mathsf{planar}(H)$. e_2 is undefined if no such integer exists.
- Let q_1 be the largest integer such that $q_1 < q$ and an edge e_3 with endpoints $c_\ell^{q_1}(w)$ and w exists in $\mathsf{planar}(H)$. e_3 is undefined if no such integer exists.
- Let q_2 be the smallest integer such that $q_2 > q$ and an edge e_4 with endpoints $c_\ell^{q_2}(w)$ and w exists in $\mathsf{planar}(H)$. e_4 is undefined if no such integer exists.

For $i = 1, 2, 3, 4$, the edges e_i which are defined above are called *shield* edges of e.

We will be using the following Lemma, which was proven by Imai et al., to help us in the construction of pseudoseparator.

Lemma 4.6. *[9] For every $\beta > 0$, there exists a polynomial-time and $\tilde{O}(h^{1/2+\beta/2})$-space algorithm that takes a h-vertex planar graph P as input and outputs a set of vertices S, such that $|S|$ is $O(h^{1/2+\beta/2})$ and removal of S disconnects the graph into components of size $O(h^{1-\beta})$.*

Algorithm 2 describes how to construct a sub-digraph of H which we call $\mathsf{psep}(H)$. In Lemma 4.8 we show that $\mathsf{psep}(H)$ is a pseudoseparator of H.

Input: Aninduced sub-digraph H of $\mathsf{Aux}_\alpha(G)$ and a positive number β
Output: The digraph psep(H)

1 Construct $\widehat{H}$ from H;
2 Find a set S of vertices in $\widehat{H}$ which divides it into components of size $O(n^{1-\beta})$ by applying Lemma 4.6 ;
3 Output all the vertices of S;
4 Output those edges of H which are also (in its undirected form) in $\widehat{H}[S]$;
5 **foreach** *edge* $e = (v, w)$ *of* $\widehat{H}$ **do**
6   **if** *e is* inside *a block of* $\widehat{H}$ ***and*** $e \notin E(\mathsf{planar}(H))$ ***and*** *both endpoints of e are in S* **then**
7   Output all the shield edges of e along with their end vertices.
8   **end**
9 **end**

Algorithm 2: Construction of psep(H)

Lemma 4.7. *Let G be a grid digraph , and H be aninduced sub-digraph of $\mathsf{Aux}_\alpha(G)$. For every $\beta > 0$, there exists a polynomial-time and $\tilde{O}(h^{1/2+\beta/2})$-space algorithm that takes H as input and outputs $\mathsf{psep}(H)$.*

Proof. To prove this lemma, we analyse the space and time complexity of Algorithm 2. We will first see that $\widehat{H}$ is constructed from H in logspace. To see it, we first note that we can decide if an edge e of H belong to $\mathsf{planar}(H)$ in logspace by checking if e satisfies the condition required by Definition 4.2. Next, the boundary of $\mathsf{planar}(H)$ is completed in logspace by using Algorithm 1. Finally we triangulate the resultant graph in logspace: For every face f in the resultant graph, we connect each vertex of f to the lowest-indexed vertex of f. The space complexity of construction of psep(H) is thus dominated by the space required by step 2 of Algorithm 2. By Lemma 4.6, we know that this step requires $O(h^{1/2+\beta/2})$ space. Hence, the space required by Algorithm 2 is $O(h^{1/2+\beta/2})$. Each step of Algorithm 2 is performed in polynomial time, hence the total time complexity is bounded by a polynomial. □

Lemma 4.8. *Let G be a grid digraph , and H be aninduced sub-digraph of $\mathsf{Aux}_\alpha(G)$. The digraph $\mathsf{psep}(H)$ is a $h^{1-\beta}$-*pseudoseparator *of H.*

To prove Lemma 4.8, we first show a property of triangulated graphs that we use in our construction of pseudoseparator. We know that a simple cycle in a planar embedding of a planar graph divides the plane into two parts. We call these two parts the two *sides* of the cycle.

Lemma 4.9. *Let G be a triangulated planar graph, and S be a subset of its vertices. For each pair of vertices u, v belonging to different components of $G \setminus S$, a cycle in $G[S]$ exists, such that u and v belong to different sides of this cycle.*

Proof. To prove the Lemma, we first need some terminology. We call a set of faces a *region of edge-connected faces* if they induce a connected subgraph in the dual graph. We can orient the edges of an undirected planar simple cycle to make it a directed cycle. This can help us identify the two *sides* of the cycle as *interior* (left-side) and *exterior* (right-side).

Let C be a component of $G \setminus S$ and S' be the set of vertices of S adjacent to at least one of the vertices of C in G. Let F be the set of triangle faces of G to which at least one vertex of C belongs. Let $\tilde{G}$ be the dual graph of G and $\tilde{G}[F]$ be the subgraph induced by F on this dual graph.

We first observe that since we have started with a triangulated graph, the vertices of any face f of F will either belong to C or S'.

Let f_1 and f_2 be two arbitrary faces of F. We will first show that F is a region of edge-connected faces by showing that there exists a path between f_1 and f_2 in $\tilde{G}[F]$. Let v_1 be a vertex of C which belongs to f_1. Similarly, let v_2 be a vertex of C which belongs to f_2. Let the length of shortest path between v_1 and v_2 in $G \setminus S$ be k. We prove our claim by induction on k. If $k = 0$, then $v_1 = v_2$. We know that the set of faces that share a vertex induce a connected subgraph in the dual graph, hence our claim holds for $k = 0$. Now, we assume that our claim holds for path length $k = \ell - 1$. To prove that our claim holds for $k = \ell$, let (v_3, v_2) be the last edge in the shortest path from v_1 to v_2. Let f_3 be a face with both v_3 and v_2 in its boundary. Since the length of shortest path from v_1 to v_3 is $\ell - 1$, by induction hypothesis, there exists a path between f_1 and f_3 in $\tilde{G}[F]$. Since f_3 and f_2 share the vertex v_2, there is a path between f_3 and f_2 in $\tilde{G}[F]$. Combining these two paths, we get a path between f_1 and f_2 in $\tilde{G}[F]$ which proves our claim.

Miller proved that the boundary of a region of edge-connected faces is a set of edge-disjoint *simple* cycles which can be oriented such that they have disjoint exteriors [12]. We claim that all the vertices of any boundary cycle of F are contained in S'. We prove this claim by contradiction. Let us assume that v is a vertex in a boundary cycle of F that is not in S'. In this case, v belongs to C. Consider an edge (v, w) of the boundary cycle. Let f_a and f_b be the two faces that shares the edge (v, w). Let v_a and v_b be the third vertex of f_a and f_b respectively. Since v_a, v_b and w are all connected to v, they are in either S' or C. Thus both f_a and f_b are in F. However, this contradicts that (v, w) is at boundary of F.

Thus, we have proved that the vertices of a connected component of $G \setminus S$ are contained inside a set of edge-disjoint simple cycles with disjoint exteriors. The vertices of all such cycles are contained in S. Hence the lemma follows. ◻

Proof of Lemma 4.8. Let $C = \mathsf{psep}(H)$. Let S be the set of vertices obtained from $\widehat{H}$ by using Lemma 4.6. We claim that if any two vertices u and v belong to different connected components of $\widehat{H} \setminus S$, then it belongs to different components of $\mathsf{cc}(H \diamond C)$. We prove this by contradiction. Let us assume that it is not true. Then there is an edge e in H and two distinct sets U_1 and U_2 of $\mathsf{cc}(\widehat{H} \setminus S)$ such that one of the end point of e is in U_1, the other is in U_2 and e does not cross any of the edges of $\mathsf{psep}(H)$. Without loss of generality, let $e = (v, c_\ell^p(v))$, for some block ℓ, where $v \in U_1$ and $c_\ell^p(v)$ is not in U_1 (we pick the edge e such that p is minimum for any choice of v). Due to Lemma 4.9, it follows that there exists an edge f in $\widehat{H}[S]$ such that $f = ((c_\ell^q(v)), c_\ell^r(v))$ and that e crosses f. This edge f is a triangulation edge, for otherwise it is also in psep(H) giving us a contradiction. We orient the triangulation edge so that $q < p < r$. Now, since e is not in planar(H), by Lemma 4.3 there exists at least one edge that crosses it and is in planar(H). Let $g = (c^s_\ell(v), c^t_\ell(v))$ be one such edge such that $t - s$ is maximum[1] We thus have the following cases:

Case 1 ($s < q < p < r < t$): In this case, since g crosses e, by Lemma 3.5, we have that the edge $e' = (c^s_\ell(v), c^p_\ell(v))$ is also in H. e' also crosses f. Since $p - s < p$, existence of e' contradicts our choice of e.

Case 2 ($q < t < p < s < r$): In this case, since g crosses e, by Lemma 3.5, we have that the edge $e' = (v, c^t_\ell(v))$ is also in H. e' also crosses f. Since $t < p$, existence of e' contradicts our choice of e.

Case 3 ($t < q < p < r < s$): In this case, the edge $e' = (c^s_\ell(v), c^p_\ell(v))$ will also be in H. e' will cross f and hence will not be in planar(H). By Lemma 4.3, there exists an edge $g' = (c^{s'}_\ell(v), c^{t'}_\ell(v))$ in planar(H) that crosses e'. Since both g' and g are in planar(H), these edges do not cross each other.

Using the fact that g' crosses e' and g' does not cross g, we get the following:

$$t \leq \min(s', t') < p < \max(s', t') < s$$

Consequently, $t' - s' > t - s$ and g' crosses e. This contradicts our choice of g.

Case 4 ($q < s < p < t < r$): In this case, since g crosses e, by Lemma 3.5, we have that the edge $e' = (v, c^t_\ell(v))$ is in H. e' also crosses f and hence e' was not in planar(H). Thus there will exist an edge in planar(H) which crosses e' by Lemma 4.3. Let $g' = (c^{s'}_\ell(v), c^{t'}_\ell(v))$ be the edge in planar(H) that crosses e' such that $t' - s'$ is maximum. We have the following subcases:

Case 4a ($s' < q < t < r < t'$): In this case, since g' crosses e, by Lemma 3.5, we have that the edge $e'' = (c^{s'}_\ell(v), c^p_\ell(v))$ is also in H. e'' also crosses f. Since $p - s' < p$, existence of e'' contradicts our choice of e.

Case 4b ($q < t' < t < s' < r$): In this case, we see that since g and g' are both in planar(H), they do not cross each other. Therefore. $t' \leq s$ and consequently $t' < p$. Thus, g' crosses e and by Lemma 3.5, the edge $e'' = (v, c^{t'}_\ell(v))$ is also in H. e'' also crosses f. Since $t' < p$, existence of e'' contradicts our choice of e.

Case 4c ($t' < q < t < r < s'$): In this case, the edge $e'' = (c^{s'}_\ell(v), c^t_\ell(v))$ will also be in H. e'' will cross f and hence will not be in planar(H). By Lemma 4.3, there exists an edge $g'' = (c^{s''}_\ell(v), c^{t''}_\ell(v))$ in planar(H) that crosses e''. Since g'', g' and g are all in planar(H), these edges do not cross each other.

Using the fact that g'' crosses e'', g'' does not cross g' and g'' does not cross g; we get the following:

[1]We note that we do *not* consider the *absolute* value of the $t - s$, rather we consider the actual value. $t - s$ can hence be negative for an edge.

(a) The s-t path takes a vertex of the separator

(b) The s-t path crosses an edge of the separator

Figure 4

$$t' \le \min(s'', t'') < s < t < \max(s'', t'') < s'$$

Consequently, $t'' - s'' > t' - s'$ and g'' crosses e'. This contradicts our choice of g'.

Case 4d ($q < s' < t < t' < r$): In this case, we see that since g and g' are both in $\mathsf{planar}(H)$, they do not cross each other. Therefore $s' \le s$ and consequently $s' < p$. Thus, g' crosses e and $t' - s' > t - s$. This contradicts our choice of g.

In other cases, if g is picked such that one of its vertices is common with f, then e will cross a $shield$ edge of f, giving a contradiction. If g is picked such that g cross f, then it contradicts the fact that both of them are in $\widehat{H}$, which is a planar digraph . ☐

Summarizing Lemmas 4.7 and 4.8 we have Theorem 4.10.

Theorem 4.10. *Let G be a grid digraph and H be aninduced subgraph of $\mathsf{Aux}_\alpha(G)$ with h vertices. For any constant $\beta > 0$, there exists an $\widetilde{O}(h^{1/2+\beta/2})$-space and polynomial-time algorithm that takes H as input and outputs an $h^{1-\beta}$-*pseudoseparator *of size $O(h^{1/2+\beta/2})$.*

5 Algorithm to solve reachability in the auxiliary digraph

In this section, we discuss the grid digraph reachability algorithm. Let G be a grid digraph having $\widetilde{n}$ vertices. By induction, we assume that we have access to an induced subgraph H of $\mathsf{Aux}_\alpha(G)$, containing h vertices. Below we describe a recursive procedure $\mathsf{AuxReach}(H, x, y)$ that outputs true if there is a path from x to y in H and outputs false otherwise.

5.1 Description of the algorithm AuxReach

First we construct a $h^{1-\beta}$-pseudoseparator C of H, using Theorem 4.10. We also ensure that x and y are part of C (if not then we add them). Let $I_1, I_2, \ldots, I_\ell$ be the connected components of $H \diamond C$.

We maintain an array called $\mathsf{visited}$ of size $|C|$ to mark vertices or edges of the pseudoseparator C. Each cell of $\mathsf{visited}$ corresponds to a distinct vertex or edge of C. For a vertex v in C, we set $\mathsf{visited}[v] := 1$ if there is a path from x to v in H, else it is set to 0. For an edge $e = (u, v)$ in C, we set $\mathsf{visited}[e] := u'$ if (i) there is an edge $f = (u', v')$ that crosses e, (ii) there is a path from x to u'

in H and (iii) f is the closest such edge to u. Else $\mathsf{visited}[e]$ is set to NULL. Initially, for all vertex $v \in C$, $\mathsf{visited}[v] := 0$ and for all edges $e \in C$, $\mathsf{visited}[e] := \mathsf{NULL}$. We say that a vertex v is *marked* if either $\mathsf{visited}[v] = 1$ or $\mathsf{visited}[e] = v$ for some edge e.

First set $\mathsf{visited}[x] := 1$. We then perform an outer loop with h iterations and in each iteration update certain entries of the array $\mathsf{visited}$ as follows. For every vertex $v \in C$, the algorithm sets $\mathsf{visited}[v] := 1$ if there is a path from a marked vertex to v such that the internal vertices of that path all belong to only one component I_i. Similarly, for each edge $e = (u, v)$ of C, the algorithm sets $\mathsf{visited}[e] := u'$ if (i) there exists an edge $f = (u', v')$ which crosses e, (ii) there is a path from a marked vertex to u' such that the internal vertices of that path all belong to only one component I_i and, (iii) f is the closest such edge to u. Finally we output true if $\mathsf{visited}[y] = 1$ else output false. We use the procedure $\mathsf{AuxReach}$ recursively to check if there is a path between two vertices in a single connected component of $H \diamond C$. A formal description of $\mathsf{AuxReach}$ is given in Algorithm 3.

5.2 Proof of correctness of AuxReach

Let P be a path from x to y in H. Suppose P passes through the components $I_{\sigma_1}, I_{\sigma_2}, \ldots, I_{\sigma_L}$ in this order. The length of this sequence is at most $|H|$. As the path leaves the component I_{σ_j} and goes into $I_{\sigma_{j+1}}$, it can do in the following two ways only:

1. The path exits I_{σ_j} through a vertex w of pseudoseparator as shown in Figure 4a. In this case, Algorithm 3 marks the vertex w.

2. The path exits I_{σ_j} through an edge (u, v) whose other endpoint is in $I_{\sigma_{j+1}}$. By Lemma 3.6, this edge will cross an edge $e = (x', y')$ of the pseudoseparator. In this case, Algorithm 3 marks the vertex u', such that there is an edge (u', v') that crosses e as well and (u', v') is closer than (u, v) to x' and there is a path in I_{σ_j} from a marked vertex to u'. By Lemma 3.6, the edge (u', v) is in H as well.

Thus after the j-th iteration, $\mathsf{AuxReach}$ traverses the fragment of the path in the component I_{σ_j} and either marks its endpoint or a vertex which is closer to the edge e of C which the path crosses. Finally, y is marked after L iterations if and only if there is a path from x to y in H. We give a formal proof of correctness in Lemma 5.1. For a path $P = (u_1, u_2, \ldots u_t)$, we define $\mathsf{tail}(P) := u_1$ and $\mathsf{head}(P) := u_t$.

Lemma 5.1. *Let G be a grid digraph and H be aninduced subgraph of $\mathsf{Aux}_\alpha(G)$. Then for any two vertices x, y in H, there is a path from x to y in H if and only if $\mathsf{AuxReach}(H, x, y)$ returns true.*

Proof. Firstly observe that, if a vertex is marked, then there is a path from some other marked vertex to that vertex in H. Hence if there is no path from x to y then y is never marked by $\mathsf{AuxReach}$ and hence $\mathsf{AuxReach}$ returns false.

Now let P be a path from x to y in H. We divide the path into subpaths $P_1, P_2, \ldots, P_\ell$, such that for each i, all vertices of P_i belong to $U \cup V(C)$ for some connected component U in $\mathsf{cc}(H \diamond C)$ and either (i) $\mathsf{head}(P_i) = \mathsf{tail}(P_{i+1})$, or (ii) $e_i = (\mathsf{head}(P_i), \mathsf{tail}(P_{i+1}))$ is an edge that crosses some edge $f_i \in C$. By Definition 4.1, we have that if condition (i) is true then $\mathsf{head}(P_i)$ is a vertex in C,

Input: An induced subgraph H of $\mathsf{Aux}_\alpha(G)$ and two vertices x and y in H (let G be an $m \times m$ grid digraph and $h = |V(H)|$)

Output: true if there is a path from x to y in H and false otherwise

```
if $h \leq m^{1/8}$ then Use DFS to solve the problem; /* $m$ is a global variable where $G$ is an $m \times m$ grid digraph */
else
    Compute a $h^{1-\beta}$-pseudoseparator $C$ of $H$ using Theorem 4.10;
    $C \leftarrow C \cup \{x, y\}$;
    foreach edge $e$ in $C$ do visited$[e] \leftarrow$ NULL;
    foreach vertex $v$ in $C$ do visited$[v] \leftarrow 0$;
    visited$[x] \leftarrow 1$;
    for $i = 1$ to $|H|$ do
        foreach edge $e = (u, v) \in C$ do
            closestedge $\leftarrow$ NULL;
            foreach marked vertex $w$ do
                foreach $U \in cc(H \diamond C)$ do
                    foreach edge $f = (u', v')$ such that $f$ crosses $e$ do
                        if AuxReach$(H[U \cup \{w, u'\}], w, u') =$ true then
                            if closestedge = NULL or $f$ is closer to $e$ than closestedge then
                                visited$[e] \leftarrow u'$;
                                closestedge $\leftarrow f$;
                            end
                        end
                    end
                end
            end
        end
        foreach vertex $v \in C$ do
            if $\exists w \exists U$ such that $w$ is a marked vertex and $U \in cc(H \diamond C)$ and
             AuxReach$(H[U \cup \{w, v\}], w, v) =$ true)) then
                visited$[v] \leftarrow 1$;
            end
        end
    end
    if visited$[y] = 1$ then return true;
    else return false;
end
```

Algorithm 3: AuxReach(H, s, t)

and if condition (ii) is true then $\mathsf{head}(P_i)$ and $\mathsf{tail}(P_{i+1})$ belong to two different components of $\mathsf{cc}(H \diamond C)$ and e_i is the edge between them.

We claim that after i-th iteration of loop in Line 8 of Algorithm 3, either of the following two statements hold:

1. $\mathsf{head}(P_i)$ is a vertex in C and $\mathsf{visited}[\mathsf{head}(P_i)] = 1$.

2. There exists an edge $f_i = (u_i, v_i)$ of C such that the edge $e_i = (\mathsf{head}(P_i), \mathsf{tail}(P_{i+1}))$ crosses f_i and there is an edge $g_i = (u'_i, v'_i)$ which crosses f_i as well, such that g_i is closer to u_i than e_i and $\mathsf{visited}[f_i] = u'_i$.

We prove the claim by induction. The base case holds since x is marked at the beginning. We assume that the claim is true after the $(i-1)$-th iteration. We have that P_i belongs to $U \cup V(C)$ for some connected component U in $\mathsf{cc}(H \diamond C)$.

Case 1 ($\mathsf{head}(P_{i-1}) = \mathsf{tail}(P_i) = w$ (say)): By induction hypothesis w was marked after the $(i-1)$-th iteration. If $\mathsf{head}(P_i)$ is a vertex in C then it will be marked after the i-th iteration in Line 26. On the other hand if $e_i = (\mathsf{head}(P_i), \mathsf{tail}(P_{i+1}))$ is an edge that crosses some edge $f_i = (u_i, v_i) \in C$ then in the i-th iteration in Line 16, the algorithm marks a vertex u'_i such that, $g_i = (u'_i, v'_i)$ is the closest edge to u_i that crosses f_i and there is a path from w to u'_i.

Case 2 ($e_{i-1} = (\mathsf{head}(P_{i-1}), \mathsf{tail}(P_i))$ crosses some edge $f_{i-1} = (u_{i-1}, v_{i-1}) \in C$): By induction hypothesis, there is an edge $g_{i-1} = (u'_{i-1}, v'_{i-1})$ which crosses f_{i-1} as well, such that g_{i-1} is closer to u_{i-1} than e_{i-1} and $\mathsf{visited}[f_{i-1}] = u'_{i-1}$. By Lemma 3.6 there is an edge in H between u'_{i-1} and $\mathsf{tail}(P_i)$ as well. Now if $\mathsf{head}(P_i)$ is a vertex in C then it will be marked after the i-th iteration in Line 26 by querying the digraph $H[U \cup \{u'_{i-1}, \mathsf{head}(P_i)\}]$. On the other hand if $e_i = (\mathsf{head}(P_i), \mathsf{tail}(P_{i+1}))$ is an edge that crosses some edge $f_i = (u_i, v_i) \in C$ then in the i-th iteration in Line 16, $\mathsf{AuxReach}$ queries the digraph $H[U \cup \{u'_{i-1}, u'_i\}]$ and marks a vertex u'_i such that, $g_i = (u'_i, v'_i)$ is the closest edge to u_i that crosses f_i and there is a path from u'_{i-1} to u'_i. □

Our subroutine solves reachability in a subgraph H (having size h) of $\mathsf{Aux}_\alpha(G)$. We do not explicitly store a component of $\mathsf{cc}(H \diamond C)$, since it might be too large. Instead, we identify a component with the lowest indexed vertex in it and use Reingold's algorithm on $H \diamond C$ to determine if a vertex is in that component. We require $\widetilde{O}(h^{1/2+\beta/2})$ space to compute a $h^{1-\beta}$-pseudoseparator by Theorem 4.10. We can potentially mark all the vertices of the pseudoseparator and for each edge of the pseudoseparator we mark at most one additional vertex. Since the size of the pseudoseparator is at most $O(h^{1/2+\beta/2})$, we require $\widetilde{O}(h^{1/2+\beta/2})$ space. The algorithm recurses on a digraph with $h^{1-\beta}$ vertices. Since we stop the recursion when $h \leq m^{1/8}$ (Line 1 of Algorithm 3), the depth of the recursion is at most $\lceil -3/\log(1-\beta) \rceil$, which is a constant.

Since the digraph H is given implicitly in our algorithm, an additional polynomial overhead is involved in obtaining its vertices and edges. However, the total time complexity remains a polynomial in the number of vertices since the recursion depth is constant.

Lemma 5.2. *Let G be an $m \times m$ grid digraph and H beaninduced subgraph of $\mathsf{Aux}_\alpha(G)$ with h vertices. For every $\beta > 0$, AuxReach runs in $\widetilde{O}(h^{1/2+\beta/2})$ space and polynomial time.*

Proof. Since the size of a component U in $\mathsf{cc}(H \diamond C)$ might be too large, we will not explicitly store it. Instead we identify a component by the lowest index vertex in it and use Reingold's algorithm on $H \diamond C$ to determine if a vertex is in U. Let $S(m, h)$ and $T(m, h)$ denote the space and time complexity functions respectively of AuxReach, where G is an $m \times m$ grid digraph and h is the number of vertices in the digraph H. As noted earlier the depth of the recursion is at most $d := \lceil -3/\log(1-\beta) \rceil$.

Consider $S(m, h)$ for any $h > m^{1/8}$. By Theorem 4.10, we require $\widetilde{O}(h^{1/2+\beta/2})$ space to execute Line 3. We can potentially mark all the vertices of C and for each edge e of C we store at most one additional vertex in $\mathsf{visited}[e]$. Since the size of C is at most $O(h^{1/2+\beta/2})$, we require $\widetilde{O}(h^{1/2+\beta/2})$ space to store C. By induction, a call to AuxReach in line 16 and 26 requires $S(m, h^{1-\beta})$ space which can be subsequently reused. Hence the space complexity satisfies the following recurrence. Then,

$$S(m,h) = \begin{cases} S(m, h^{1-\beta}) + \widetilde{O}(h^{1/2+\beta/2}) & h > m^{1/8} \\ \widetilde{O}(h) & \text{otherwise.} \end{cases}$$

Solving we get $S(m, h) = \widetilde{O}(h^{1/2+\beta/2} + m^{1/4})$.

Next we measure the time complexity of AuxReach. Consider the case when $h > m^{1/8}$. The total number of steps in AuxReach is some polynomial in h, say p. Moreover AuxReach makes q calls to AuxReach, where q is some other polynomial in h. Hence $q(h) \leq p(h)$. Then,

$$T(m,h) = \begin{cases} q \cdot T(h^{1-\beta}) + p & h > m^{1/8} \\ O(h) & \text{otherwise.} \end{cases}$$

Solving the above recurrence we get $T(m, h) = O(p \cdot q^d + m^{1/4}) = O(p^{2d} + m^{1/4})$. ☐

6 Solving grid digraph

Let G be an $m \times m$ grid digraph . As mentioned in the introduction, our objective is to run the algorithm 3 on the digraph $\mathsf{Aux}_\alpha(G)$. Consider two vertices x and y of $\mathsf{Aux}_\alpha(G)$. Note that, by definition, y is reachable from x in $\mathsf{Aux}_\alpha(G)$ if and only if y is reachable from x in G. Hence it is sufficient to work with the digraph $\mathsf{Aux}_\alpha(G)$. However, we do not have explicit access to the edges of $\mathsf{Aux}_\alpha(G)$. Note that we can obtain the edges of $\mathsf{Aux}_\alpha(G)$ by solving the corresponding subgrid of G to which that edge belongs. If the subgrid is small enough, then we use a standard linear space-traversal algorithm. Otherwise, we use our algorithm recursively on the subgrid. Algorithm 4 outlines this method.

Consider an $\widehat{m} \times \widehat{m}$ grid digraph $\widehat{G}$. Let $S(\widehat{m})$ be the space complexity and $T(\widehat{m})$ be the time complexity of executing GridReach on $\widehat{G}$. Note that the size of $\mathsf{Aux}_\alpha(\widehat{G})$ is at most $\widehat{m}^{1+\alpha}$. For $\widehat{m} > m^{1/8}$, the space required to solve the grid digraph

Input: A grid digraph $\widehat{G}$ and two vertices $\widehat{s}, \widehat{t}$ of $\widehat{G}$ and a positive integer m
Output: true if there is a path from s to t in G and false otherwise
if $\widehat{G}$ *is smaller than* $m^{1/8} \times m^{1/8}$ *grid* **then**
  Use Depth-First Search to solve the problem;
end
else
  Use ImplicitAuxReach($\mathsf{Aux}_\alpha(G), \widehat{s}, \widehat{t}$) to solve the problem;
  /* ImplicitAuxReach executes the same way as AuxReach except for the case when it queries an edge (u, v) in a block B of $\mathsf{Aux}_\alpha(G)$. In this case, the query is answered by calling GridReach(B, u, v, m) where B is the subgrid in which edge (u, v) might belong. */
end

Algorithm 4: GridReach($\widehat{G}, \widehat{s}, \widehat{t}, m$)

is $S(\widehat{m}) = S(\widehat{m}^{1-\alpha}) + \widetilde{O}((\widehat{m}^{1+\alpha})^{1/2+\beta/2})$. This is because, a query whether $(u, v) \in \widehat{G}$invokes a recursion which requires $S(\widehat{m}^{1-\alpha})$ space and the main computation of ImplicitAuxReach can be done using $\widetilde{O}((\widehat{m}^{1+\alpha})^{1/2+\beta/2})$ space. Hence we get the following recurrence for space complexity.

$$S(\widehat{m}) = \begin{cases} S(\widehat{m}^{1-\alpha}) + \widetilde{O}((\widehat{m}^{1+\alpha})^{1/2+\beta/2}) & \widehat{m} > m^{1/8} \\ \widetilde{O}(\widehat{m}^{1/4}) & \text{otherwise} \end{cases}$$

Similar to the analysis of AuxReach, for appropriate polynomials p and q, the time complexity satisfies the following recurrence:

$$T(\widehat{m}) = \begin{cases} q(\widehat{m}) \cdot T(\widehat{m}^{1-\alpha}) + p(\widehat{m}) & \widehat{m} > m^{1/8} \\ O(\widehat{m}) & \text{otherwise.} \end{cases}$$

Solving we get $S(m) = \widetilde{O}(m^{1/2+\beta/2+\alpha/2+\alpha\beta/2})$ and $T(m) = \mathsf{poly}(m)$. For any constant $\epsilon > 0$, we can chose α and β such that $S(m) = O(m^{1/2+\epsilon})$.

7 Conclusion

Our result improves upon the known time–space bounds on the grid digraph . Asano et al. [4] used a clever idea of exploiting the grid structure of the input digraph to reduce its size. Their exploitation destroyed the grid structure of the input digraph , but they did not go far enough to destroy its planar structure as well. Our exploitation goes a step further. We get a non-planar auxiliary digraph that is smaller as a result and has just enough structure to solve reachability in a space-efficient manner. It remains to be seen if the structure could be further exploited to get a smaller *auxiliary-like* digraph in which reachability can be solved.

It is known that reachability in planar digraph s can be reduced to reachability in grid digraph s in logarithmic space [1]. However, such a reduction results in at least a quadratic blow-up in the size of the digraph . In principle, it would seem that an improvement to the state of the art for the planar digraph can be obtained by improving the result for grid digraph s. However, we feel that this would be a difficult direction to go. Note that we solve an h-vertex auxiliary digraph in $O(h^{1/2+\beta/2})$ space as a subroutine for our griddigraph algorithm. A planar digraph with its embedding can be thought of as an auxiliary digraph , and hence our algorithm contains within itself a solution to the planar digraph as well. For this reason, we feel that directly solving a planar digraph would be easier than going down the grid digraph routine. A significantly different approach would be required to directly design an algorithm for grid digraph , one which does not use a solution for the class of planar digraph s, or its superclass, as a subroutine.

References

[1] Eric Allender, David A. Mix Barrington, Tanmoy Chakraborty, Samir Datta, and Sambuddha Roy: Planar and grid graph reachability problems. *Theory Computing Sys.*, 45(4):675–723, 2009. [doi:10.1007/s00224-009-9172-z] 2, 20

[2] Eric Allender and Meena Mahajan: The complexity of planarity testing. *Inform. Comput.*, 189(1):117–134, 2004. Preliminary version in STACS'00. [doi:10.1016/j.ic.2003.09.002] 4

[3] Tetsuo Asano and Benjamin Doerr: Memory-constrained algorithms for shortest path problem. In *Proc. 23rd Canad. Conf. Comput. Geom. (CCCG'11)*, pp. 1–4, 2011. cccg.ca. 3

[4] Tetsuo Asano, David Kirkpatrick, Kotaro Nakagawa, and Osamu Watanabe: $\tilde{O}(\sqrt{n})$-space and polynomial-time algorithm for planar directed graph reachability. In *Proc. Internat. Symp. Math. Foundations of Comp. Sci. (MFCS'14)*, pp. 45–56. Springer, 2014. [doi:10.1007/978-3-662-44465-8_5, ECCC:TR14-071] 2, 3, 20

[5] Ryo Ashida and Kotaro Nakagawa: $\tilde{O}(n^{1/3})$-space algorithm for the grid graph reachability problem. In *Proc. 34th Internat. Symp. Comput. Geom. (SoCG'18)*, pp. 5:1–13. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2018. [doi:10.4230/LIPIcs.SoCG.2018.5, arXiv:1803.07097] 3

[6] Greg Barnes, Jonathan F. Buss, Walter L. Ruzzo, and Baruch Schieber: A sublinear space, polynomial time algorithm for directed s-t connectivity. *SIAM J. Comput.*, 27(5):1273–1282, 1998. Preliminary version in SCT'92. [doi:10.1137/S0097539793283151] 2

[7] Diptarka Chakraborty, Aduri Pavan, Raghunath Tewari, N. V. Vinodchandran, and Lin F. Yang: New time-space upperbounds for directed reachability in high-genus and H-minor-free graphs. In *Proc. 34th Found. Softw. Techn. Theoret. Comp. Sci. Conf. (FSTTCS'14)*, pp. 585–595. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014. [doi:10.4230/LIPIcs.FSTTCS.2014.585, ECCC:TR14-035] 2

[8] Diptarka Chakraborty and Raghunath Tewari: An $O(n^{\epsilon})$ space and polynomial time algorithm for reachability in directed layered planar graphs. *ACM Trans. Comput. Theory*,

9(4):19:1–11, 2017. Preliminary version in ISAAC'15. [doi:10.1145/3154857, arXiv:1501.05828, ECCC:TR15-016] 2

[9] TATSUYA IMAI, KOTARO NAKAGAWA, ADURI PAVAN, N. V. VINODCHANDRAN, AND OSAMU WATANABE: An $O(n^{\frac{1}{2}+\epsilon})$-space and polynomial-time algorithm for directed planar reachability. In *Proc. 28th IEEE Conf. on Comput. Complexity (CCC'13)*, pp. 277–286. IEEE Comp. Soc., 2013. [doi:10.1109/CCC.2013.35] 2, 3, 8, 11

[10] RAHUL JAIN AND RAGHUNATH TEWARI: An $O(n^{1/4+\epsilon})$ space and polynomial algorithm for grid graph reachability. In *Proc. 39th Found. Softw. Techn. Theoret. Comp. Sci. Conf. (FSTTCS'19)*, pp. 19:1–14. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019. [doi:10.4230/LIPIcs.FSTTCS.2019.19] 1

[11] SAMPATH KANNAN, SANJEEV KHANNA, AND SUDEEPA ROY: STCON in directed unique-path graphs. In *Proc. 28th Found. Softw. Techn. Theoret. Comp. Sci. Conf. (FSTTCS'08)*, pp. 256–267. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2008. [doi:10.4230/LIPIcs.FSTTCS.2008.1758] 2

[12] GARY L. MILLER: Finding small simple cycle separators for 2-connected planar graphs. *J. Comput. System Sci.*, 32(3):265–279, 1986. Preliminary version in STOC'84. [doi:10.1016/0022-0000(86)90030-9] 13

[13] OMER REINGOLD: Undirected connectivity in log-space. *J. ACM*, 55(4):17:1–24, 2008. Preliminary version in STOC'05. [doi:10.1145/1391289.1391291] 2, 4

[14] WALTER J. SAVITCH: Relationships between nondeterministic and deterministic tape complexities. *J. Comput. System Sci.*, 4(2):177–192, 1970. [doi:10.1016/S0022-0000(70)80006-X] 2

[15] DERRICK STOLEE AND N. V. VINODCHANDRAN: Space-efficient algorithms for reachability in surface-embedded graphs. In *Proc. 27th IEEE Conf. on Comput. Complexity (CCC'12)*, pp. 326–333. IEEE Comp. Soc., 2012. [doi:10.1109/CCC.2012.15, ECCC:TR10-154] 2

[16] AVI WIGDERSON: The complexity of graph connectivity. In *Proc. Internat. Symp. Math. Foundations of Comp. Sci. (MFCS'92)*, pp. 112–132. Springer, 1992. [doi:10.1007/3-540-55808-X_10] 2

AUTHORS

Rahul Jain
Research associate
Department of Theoretical Computer Science
Fernuniversität in Hagen
Germany
rahul.jain@fernuni-hagen.de
https://www.fernuni-hagen.de/ti/en/team/rahul.jain

Raghunath Tewari
Associate professor
Department of Computer Science and Engineering
Indian Institute of Technology Kanpur
rtewari@cse.iit.ac.in
https://www.cse.iitk.ac.in/users/rtewari

ABOUT THE AUTHORS

RAHUL JAIN is a postdoctoral research associate at Fernuniversität in Hagen, Germany. He obtained his doctorate at the Indian Institute of Technology Kanpur, India in 2020 under Dr. Raghunath Tewari.

RAGHUNATH TEWARI is an Associate Professor in the Computer Science and Engineering Department at the Indian Institute of Technology Kanpur. He obtained his undergraduate degree in mathematics and computer science from Chennai Mathematical Institute in 2005. Thereafter he did his Masters in 2007 and Ph. D. in 2011 in computational complexity theory at the University of Nebraska – Lincoln under Dr. N. V. Vinodchandran. He is interested in computational complexity theory, algorithms and graph theory.