Directed Planar Reachability Is in Unambiguous Log-Space

CHRIS BOURKE, RAGHUNATH TEWARI, and N. V. VINODCHANDRAN
University of Nebraska–Lincoln

We make progress in understanding the complexity of the graph reachability problem in the context of unambiguous logarithmic space computation; a restricted form of nondeterminism. As our main result, we show a new upper bound on the *directed planar reachability problem* by showing that it can be decided in the class unambiguous logarithmic space (UL). We explore the possibility of showing the same upper bound for the general graph reachability problem. We give a simple reduction showing that the reachability problem for directed graphs with thickness two is complete for the class nondeterministic logarithmic space (NL). Hence an extension of our results to directed graphs with thickness two will unconditionally collapse NL to UL.

Categories and Subject Descriptors: F.1.3 [**Theory of Computation**]: Complexity Measures and Classes—*relations among complexity classes*; G.2.2 [**Mathematics of Computing**]: Graph Theory—*path and circuit problems*

General Terms: Theory, Algorithms

Additional Key Words and Phrases: Directed graph reachability, planar graphs, unambiguous log-space

ACM Reference Format:
Bourke, C., Tewari, R., and Vinodchandran, N. V. 2009. Directed planar reachability is in unambiguous log-space. ACM Trans. Comput. Theory 1, 1, Article 4 (February 2009), 17 pages.
DOI = 10.1145/1490270.1490274. http://doi.acm.org/10.1145/1490270.1490274.

1. INTRODUCTION

Graph reachability problems are fundamental to complexity theory. They capture many important complexity classes. The general *st*-connectivity problem for directed graphs is complete for NL and hence captures the power of

This work was supported by the National Science Foundation (NSF) grant CCF-0430991.
A preliminary version of this article appeared in *Proceedings of the IEEE Conference on Computational Complexity*, IEEE Computer Society Press, Los Alamitos, CA, 2007.

Author's address: C. Bourke, R. Tewari, and N. V. Vinodchandran, Department of Computer Science and Engineering, University of Nebraska–Lincoln, Lincoln, NE 68588; email: {cbourke, rtewari, vinod}@cse.unl.edu.

nondeterminism in the context of logarithmic space. Various restricted versions of this problem characterize other low-level complexity classes such as L, AC^0, and NC^1 [Etessami 1997; Reingold 2005; Barrington et al. 1998; Barrington 1989].

A natural and important restriction of the *st*-connectivity problem is when the graphs involved are planar, which we denote by PLANARREACH in this article. The complexity of this problem is not yet settled satisfactorily. The best known upper bound in terms of space complexity is NL. Though it is hard for L [Etessami 1997], it is not known whether it is complete for NL. Recently there has been progress in understanding the complexity of PLANARREACH. Allender et al. [2005] have shown that PLANARREACH log-space reduces to the reachability problem for a strict subclass of planar graphs called *grid graphs*. We denote the reachability problem for grid graphs as GGR. From this result and the fact that GGR reduces to its complement [Barrington et al. 1998], it follows that PLANARREACH reduces to its complement problem of non-reachability in planar graphs. Allender et al. [2005] also gave a direct log-space reduction from PLANARREACH to its complement.

In this article, we make further progress in understanding the space complexity of PLANARREACH. Building on earlier work, we give a simple argument to show that PLANARREACH can be decided in *unambiguous log-space*.

THEOREM 1.1 (MAIN THEOREM). PLANARREACH $\in$ UL $\cap$ coUL.

UL denotes the unambiguous subclass of NL. A decision problem L is in UL if and only if there exists a nondeterministic log-space machine M deciding L such that, for every instance x, M has at most one accepting computation on input x. Thus, planar reachability can be decided by a nondeterministic machine in log-space with at most one accepting computation.

Unambiguity in nondeterminism is a well-studied notion. In the polynomial-time setting, Valiant [1976] introduced the class UP, the unambiguous version of NP, which proved to be a very useful restriction to study, mainly because of its connection to certain kind of one-way functions [Grollman and Selman 1988]. In the logarithmic space setting, the class UL was first defined and studied by Buntrock et al. [1991] and Àlvarez and Jenner [1993]. Since then, UL and related low-space unambiguous classes have been of interest to researchers [Buntrock et al. 1991; Àlvarez and Jenner 1993; Lange 1997; Allender and Lange 1998; Reinhardt and Allender 2000; Allender et al. 2005].

The class UL is particularly interesting because there is increasing evidence that, in fact, the whole of nondeterministic logarithmic space might be contained in UL. Reinhardt and Allender [2000] show that the nonuniform version of UL contains NL; that is NL $\subseteq$ UL/poly. Can this collapse be made uniform? That is, is it true that NL = UL? To understand this question further it is worthwhile to sketch the structure of their proof. A positively and polynomially weighted graph is said to be *min-unique* if the minimum weight path (if it exists) between any two nodes is unique (the weight of a path is the sum of the weights of its edges). Reinhardt and Allender [2000] show, using a clever adaptation of the inductive counting technique of Immerman

[1988] and Szelepcsényi [1988], that the reachability question in min-unique graphs can be decided in UL. They further argue that the isolation lemma due to Mulmuley et al. [1987] can be used to nonuniformly (or randomly) assign weights to make the given graph min-unique. This gives the nonuniform collapse result. In a later paper, Allender et al. [1999] show that, under the hardness assumption that deterministic linear space has functions that can not be computed by circuits of size $2^{\epsilon n}$, the constructions given by Reinhardt and Allender [2000] can be *derandomized* to show that NL = UL. Thus, it is very likely that NL = UL, but we do not know how to prove this statement unconditionally.

From the above discussion, a very promising approach for showing NL = UL is the following. Consider a class of graphs for which the reachability problem is complete for NL. Prescribe a *deterministic* log-space computable polynomial weight function that makes graphs in this class min-unique. In this paper we prescribe a simple weight function for the class of *grid graphs* and prove that under this weight function, grid graphs are min-unique. Since PLANARREACH reduces to reachability in grid graphs. This leads to our upper bound for PLANARREACH. Although we are unable to come up with a weight function for more general class of graphs that works for solving NL = UL problem, our result indicates that such a task may not be all that difficult.

Grid graphs are graphs with vertices located on the planar grid and edges connecting a vertex only with its immediate vertical/horizontal neighbors. Reachability in grid graphs is interesting from a complexity-theoretic point of view. Barrington et al. [1998] showed that *st*-connectivity on such graphs with constant width captures the complexity of the AC^0 hierarchy. Long before Reingold [2005] showed that the undirected *st*-connectivity problem is in L, Blum and Kozen [1978] gave a deterministic log-space algorithm for undirected grid graphs. Recently, the complexity of various restrictions of grid graph reachability have been studied [Allender et al. 2005; 2006]. Specifically, Allender et al. [2005] show that the *layered* grid graph reachability problem is in UL by prescribing a weight function that makes such graphs min-unique. A layered grid graph is a grid graph with edges allowed only in three cardinal directions, thus such graphs are acyclic. Our weight function for general grid graphs is a nontrivial extension of the weight function due to Allender et al. [2005].

We also explore the possibility of extending our techniques to solve the NL = UL problem. We give a simple reduction to show that reachability in directed graphs of thickness two is complete for NL. A thickness-two graph is one whose edge set can be partitioned into two planar graphs (by definition planar graphs are thickness-one graphs). Hence, extending our technique to the class of thickness-two graphs will show NL = UL. We show this result by first observing that the reachability problem for 3 dimensional monotone grid graphs is complete for NL. We also give UL upper bounds for reachability in certain classes of nonplanar graphs.

The rest of the article is organized as follows: In the next section we present definitions and techniques needed for the results of this article. We prove the main result in Section 3. In Section 4, we give a few extensions of the main result to certain classes of nonplanar graphs and discuss extending our techniques to general graphs.

2. PRELIMINARIES

We assume familiarity with the basics of complexity theory and in particular the log-space bounded complexity class NL. A language L is in UL if and only if there exists a nondeterministic log-space machine M accepting L such that, for every instance x, M has at most one accepting computation on input x. It is well known that checking for *st*-connectivity for general directed graphs is NL-complete. We consider the *st*-connectivity problem for planar graphs and grid graphs.

A $n \times n$ *grid graph* is a directed graph whose vertices are $[n] \times [n] = \{1, \ldots, n\} \times \{1, \ldots, n\}$ so that if $((i_1, j_1), (i_2, j_2))$ is an edge then $|i_1 - i_2| + |j_1 - j_2| = 1$. Grid graphs are a very natural subclass of planar graphs with vertices identified with the $n \times n$ grid on the x-y plane oriented at $(1, 1)$ with directed edges connecting only the immediate vertical and horizontal neighbors within the grid borders. It is convenient to view the edges according to the cardinal directions. For a vertex (i, j), the edge $(i, j) \rightarrow (i, j+1)$ is a north edge, $(i, j) \rightarrow (i, j-1)$ is a south edge, $(i, j) \rightarrow (i + 1, j)$ is an east edge, and $(i, j) \rightarrow (i - 1, j)$ is a west edge.

The grid graph reachability problem, denoted GGR, is as follows. Given a grid graph G and vertices s and t, determine if there exists a directed path from s to t in G. The directed planar reachability problem denoted as PLANARREACH is the following: Given a planar graph G and vertices s and t, determine if there exists a directed path from s to t in G.

We will not be concerned with details about the representation of planar graphs. We note that the work of Allender and Mahajan [2004], and subsequently Reingold [2005], implies a deterministic logarithmic space algorithm that decides whether or not a given graph is planar and, if it is, outputs a planar embedding. We will use the following result which requires such a planar embedding.

THEOREM 2.1 [ALLENDER ET AL. 2005]. *The* PLANARREACH *problem log-space many-one reduces to* GGR.

Note that because of the above reduction and the fact that UL is closed under log-space many-one reductions, it is enough to show that GGR $\in$ UL $\cap$ coUL to prove our main theorem.

2.1 Reachability in min-unique Graphs Is in UL

Reinhardt and Allender [2000] give a general technique for showing membership in UL which we will make use of.

Definition 2.2. A *min-unique* graph is a directed graph with positive weights associated with each edge where for every pair of vertices u, v, if there is a path from u to v, then there is a unique minimum weight path from u to v. Here, the weight of a path is the sum of the weights on its edges.

Reinhardt and Allender [2000] actually define min-uniqueness for unweighted graphs, but these two definitions are essentially same in our context as one can replace an edge e with a positive, polynomially-bound integer weight $w(e)$, with a path of length $w(e)$. For completeness, we present a (somewhat shorter) version of their proof which uses a clever extension of the inductive counting techniques of Immerman [1988] and Szelepcsényi [1988]. The original proof is for the non-uniform setting and hence requires additional verification to ensure that the advice is "good." This step is not necessary for our application. Other than this, we closely follow their presentation.

THEOREM 2.3 [REINHARDT AND ALLENDER 2000]. *Let $\mathcal{G}$ be a class of graphs and let $H = (V, E) \in \mathcal{G}$. If there is a polynomially bounded log-space computable function f that on input H outputs a weighted graph $f(H)$ so that*

(1) *$f(H)$ is min-unique and*

(2) *H has an st-path if and only if $f(H)$ has an st-path*

then the st-connectivity problem for $\mathcal{G}$ is in UL ∩ coUL.

PROOF. It suffices give a UL ∩ coUL algorithm for the reduced graph. For $H \in \mathcal{G}$, let $G = f(H)$ be a directed graph with a min-unique weight function w on its edges. We first construct an unweighted graph G' from G by replacing every edge e in G with a path of length $w(e)$. It is easy to see that *st*-connectivity is preserved. That is, there is an *st*-path in G if and only if there is one in G'. Since G is min-unique, it is straightforward to argue that the shortest path between any two vertices in G' is unique.

Let c_k and Σ_k denote the number of vertices which are at a distance at most k from s and the sum of the lengths of the shortest path to each of them, respectively. Let $d(v)$ denote the length of the shortest path from s to v. If no such path exists, then let $d(v) = |V| + 1$. We have,

$$\Sigma_k = \sum_{\substack{v \in V \\ d(v) \leq k}} d(v)\,.$$

We first give an unambiguous routine (Algorithm 1) to evaluate the predicate "$d(v) \leq k$" when *given* the values of c_k and Σ_k. The algorithm will output the correct value of the predicate (*true*/*false*) on a unique path and outputs "**?**" on the rest of the paths.

We will argue that Algorithm 1 is unambiguous.

(1) If Algorithm 1 incorrectly guesses that $d(x) > k$ for some vertex x then *count* $< c_k$ and so it returns "**?**" in *line* 18. Thus, consider the computation paths that correctly guess the set $\{x \mid d(x) \leq k\}$.

(2) If at any point the algorithm incorrectly guesses the length l of the shortest path to x, then one of the following two cases occur.

    (a) If $d(x) > l$ then no path s to x would be found and the algorithm returns "**?**" in *line* 11.

Algorithm 1. Determining whether $d(v) \leq k$ or not.

```
    Input: $(G, v, k, c_k, \Sigma_k)$
    Output: $true$ if $d(v) \leq k$ else $false$
 1  Initialize $count \leftarrow 0$; $sum \leftarrow 0$; $path.to.v \leftarrow false$
 2  foreach $x \in V$ do
 3      Nondeterministically guess if $d(x) \leq k$
 4      If $guess\ is\ Yes$ then
 5          Guess a path of length $l \leq k$ from $s$ to $x$
 6          if $guess\ is\ correct$ then
 7              Set $count \leftarrow count + 1$
 8              Set $sum \leftarrow sum + l$
 9              if $x = v$then set $path.to.v \leftarrow true$
10          else
11              return "?"
12          end
13      end
14  end
15  if $count = c_k$ and $sum = \Sigma_k$ then
16      return $path.to.v$
17  else
18      return "?"
19  end
```

(b) If $d(x) < l$ then the variable *sum* would be incremented by a value greater than $d(x)$ and thus *sum* would be greater than Σ_k causing the algorithm to return "**?**" in *line* 18.

Thus, there will remain only one computation path where all the guesses are correct and the algorithm will output the correct value of the predicate on this unique path. Finally, we note that Algorithm 1 is easily seen to be log-space computable.

Next, we describe an unambiguous procedure (Algorithm 2) that computes c_k and Σ_k given c_{k-1} and Σ_{k-1}. Algorithm 2 uses Algorithm 1 as a subroutine. Other than calls to Algorithm 1, this routine is deterministic, and so it follows that Algorithm 2 is also unambiguous.

We will argue that Algorithm 2 computes c_k and Σ_k. The subgraph consisting only of s ($d(x) \leq 0$) is trivially min-unique and $c_0 = 1$ and $\Sigma_0 = 0$. Inductively, it is easy to see that

$$\begin{aligned} c_k &= c_{k-1} + \left|\{v \mid d(v) = k\}\right| \\ \Sigma_k &= \Sigma_{k-1} + k\left|\{v \mid d(v) = k\}\right| \end{aligned}$$

In addition, $d(v) = k$ if and only if there exists $(x, v) \in E$ such that $d(x) \leq k - 1$ and $\neg(d(v) \leq k - 1)$. Both of these predicates can be computed using Algorithm 1. Combining these facts, we see that Algorithm 2.1 computes c_k and Σ_k given c_{k-1} and Σ_{k-1}.

As a final step, we give the main routine that invokes Algorithm 2.1 to check for *st*-connectivity in a min-unique graph. Since there is an *st*-path if

Algorithm 2. Computing c_k and Σ_k.

```
Input: $(G, k, c_{k-1}, \Sigma_{k-1})$
Output: $c_k$, $\Sigma_k$
Initialize $c_k \leftarrow c_{k-1}$ and $\Sigma_k \leftarrow \Sigma_{k-1}$
foreach $x \in V$ do
    if $\neg(d(v) \leq k - 1)$ then
        foreach $x$ such that $(x, v) \in E$ do
            if $d(x) \leq k - 1$ then
                Set $c_k \leftarrow c_k + 1$
                Set Set $\Sigma_k \leftarrow \Sigma_k + k$
            end
        end
    end
end
return $c_k$ and $\Sigma_k$
```

Algorithm 3. Determining if there exists a path from s to t in G.

```
Input: A directed graph $G$
Output: true if there is a path from $s$ to $t$, false
        otherwise
Initialize $c_0 \leftarrow 1$, $\Sigma_0 \leftarrow 0, k \leftarrow 0$
for $k = 1, \ldots, n$ do
    Compute $c_k$ and $\Sigma_k$ by invoking Algorithm 2 on
    $(G, k, c_{k-1}, \Sigma_{k-1})$
end
Invoke Algorithm 1 on $(G, t, n, c_n, \Sigma_n)$ and return its value
```

and only if $d(t) \leq n$, it suffices to compute c_n and Σ_n and invoke Algorithm 1 on (G, t, n, c_n, Σ_n). This procedure is presented as Algorithm 3. To ensure that the algorithm runs in log-space, we do not store all intermediate values for c_k, Σ_k. Instead, we only keep the most recently computed values and reuse space. As with Algorithm 2.1, this procedure is deterministic and so the entire routine is unambiguous. Thus, reachability in min-unique graphs can be decided in UL ∩ coUL. □

3. PLANAR REACHABILITY IS IN UL

We now prove our Main Theorem. In light of Theorems 2.1 and 2.3, it suffices to show a log-space computable positive weight function which produces a min-unique graph for grid graphs.

PROOF OF THEOREM 1.1. Let G be a grid graph with the rows and columns of G indexed from 1 to n. We define a weight function w on the edges of G as follows:

$$w(e) = \begin{cases} n^4 & \text{if } e \text{ is an east or west edge} \\ i + n^4 & \text{if } e \text{ is a north edge in column } i \\ -i + n^4 & \text{if } e \text{ is a south edge in column } i \end{cases}$$

Clearly, w is log-space computable. Moreover, the weight on any edge is positive since $i \leq n$. Note that according to this weight function, the minimum weight path must be simple. Hence, we need only consider simple paths. Let P be a simple path in G and denote its weight by $w(P)$. The weight of any path is of the form $w(P) = a + bn^4$. For a given path P, let $a(P)$ denote the "a" component and let $b(P)$ denote the "b" component of its weight. Here, $a(P)$ serves to weight a path's north/south edges while $b(P)$ serves to count the total length of the path. Since the largest weight in absolute value of either component for any edge is n and there are no more than n^2 edges in any st-path, it follows that $|a(P)|, b(P) < n^3$ for any path P.

Let P_1 and P_2 be two paths in G having the same weight. Then, we have that $a(P_1) = a(P_2)$ and $b(P_1) = b(P_2)$. To see this, let $w(P_1) = a_1 + b_1 n^4$ and $w(P_2) = a_2 + b_2 n^4$. Then, we have that

$$\begin{array}{rcl} w(P_1) & = & w(P_2) \quad \Rightarrow \\ (a_1 - a_2) + (b_1 - b_2)n^4 & = & 0 \quad \Rightarrow \\ a_1 = a_2 & \text{and} & b_1 = b_2 \end{array}$$

The final implication follows since the $|a_i|$'s and b_i's, and hence their respective differences, are bounded by n^4. Now, we will argue that, with respect to this weight function for any u and v, the minimum weight path from u to v, if it exists, is unique.

First, we prove a very nice property of this weight function: namely, that the "a" component of the weight of any nontrivial simple cycle in G is non-zero. In fact, we prove the following stronger property of this weight function. For a simple cycle C, let $A(C)$ denote the number of unit squares it encloses.

LEMMA 3.1. *Let C be a simple directed cycle in G. Then $a(C) = +A(C)$ if C is a counter-clockwise cycle and $a(C) = -A(C)$ if C is a clockwise cycle.*

PROOF. It suffices to prove the lemma for a counter-clockwise simple cycle. This is because, for a clockwise cycle C, $a(C) = -a(-C)$ where $-C$ is the counter-clockwise cycle obtained by reversing the edges in C.

Let C be a counter-clockwise cycle in G. Consider the restriction of the cycle to the set of edges between two consecutive rows, say j and $j + 1$. We can view this set of edges as an ordered set, where an edge e appears before an edge e' in the ordering if e is to the west of e' in the graph. Denote this ordered set by S_j.

We claim that in S_j, the edges will alternate between south and north edges with the westmost edge being a south edge and the eastmost edge being a north edge. Assume to the contrary that there are two consecutive north edges in S_j (the argument also holds for two consecutive south edges), say $e_1 = (u_1, v_1)$ and $e_2 = (u_2, v_2)$. Consider the simple paths from v_2 to u_1 and from v_1 to u_2 along the simple cycle C. Since the path from v_2 to u_1 does not use any edges in S_j between e_1 and e_2, it must either wrap around v_1 or u_2. Say that it wraps around v_1; that is, v_1 is on the region defined by the path from v_2 to u_1, the edge (u_1, v_1) and row $j + 1$ between v_1 and v_2. Thus, the path v_1 to u_2 must either intersect the path from v_2 to u_1 or cross row $j + 1$ between e_1 and e_2. Since it cannot be the latter, this implies that the paths intersect and hence

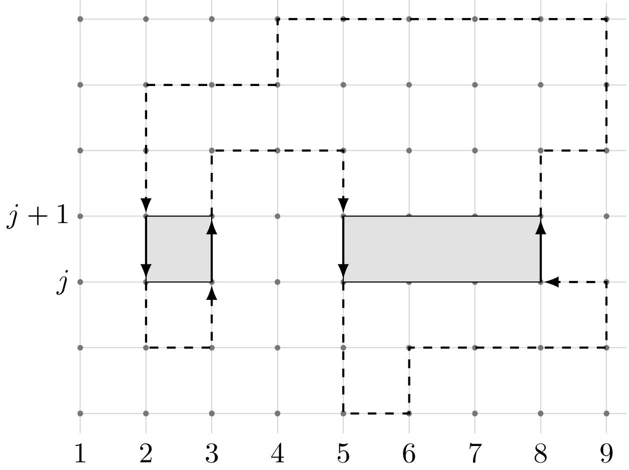

Fig. 1. A view of a grid graph between row j and $j+1$ of a counter-clockwise cycle. The cycle has an "a" component weight of $(3-2)+(8-5)=4$ with respect to rows j and $j+1$, equal to the number of unit squares it encloses.

contradicts the fact that C is a simple cycle. The same argument holds if the path wraps around u_2. In that case, we consider the region defined by the path from v_2 to u_1, the row j between u_1 and u_2 and the edge (u_2, v_2). Since C is a counter-clockwise cycle, the westmost edge is a south edge and the eastmost edge being a north edge.

Now let us look at the set of unit squares that lie between row j and $j+1$. Denote this set by R_j. The function $a(C)$ restricted to S_j counts the number of squares in R_j that lie between adjacent south and north edges, with the north edge being to the east of the south edge (cf. Figure 1). This is because the weight of the kth south edge plus the weight of the kth north edge is equal to the number of squares between these two edges.

A square in R_j is in C if and only if it is between a south and a north edge. This is because if we look at a partition of the set R_j induced by the edges in S_j, then the partitions alternately fall within and outside of C, with the set of squares between the first south and north edge lying within the cycle.

Now we sum our index j from 1 to $n-1$ and thus get that the sum of the edge weights of the cycle is equal to the number of squares it encloses. □

LEMMA 3.2. *Let G be a grid graph. With respect to the weight function w, for any two vertices u and v, the minimum weight path from u to v, if one exists, is unique.*

PROOF. Suppose there exist two different minimum weight paths P_1 and P_2 between u and v. Let u' be the vertex at which P_1 and P_2 diverge for the first time and let v' be the vertex where they meet after their first divergence. Denote the subpath of P_1 from u' to v' by P'_1 and the subpath of P_2 from u' to v' by P'_2 (cf. Figure 2). P_1 and P_2 both start at u and are not the same. The existence of u' implies the existence of v' as both paths end at v.

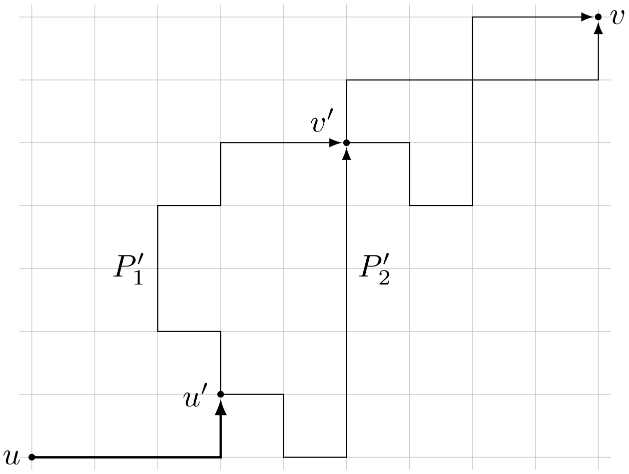

Fig. 2. Paths P_1, P_2 from u to v.

If P'_1 and P'_2 have different weights, then without loss of generality assume $w(P'_1) < w(P'_2)$. This implies that the subpath of P_2 from v' to v has smaller weight than the subpath of P_1 from v' to v. Hence, taking P_1 from u to v' and then taking P_2 to v gets a path of smaller weight from u to v in contradiction to the assumption that P_1 and P_2 are min-unique paths.

On the other hand, suppose P'_1 and P'_2 have the same weight. Then, $a(P'_1) = a(P'_2)$. Now consider the simple cycle C that follows the path P'_1 from u' to v' and then follows the path $-P'_2$ from v' back to u'. Here, for a path P, $-P$ denotes the path obtained by reversing the edges in P. It is clear that $a(-P) = -a(P)$ for any path P. Hence, $a(C) = a(P'_1) - a(P'_2) = 0$. This is a contradiction since C is a nontrivial simple cycle and hence $|a(C)| > 0$ by Lemma 3.1. □

4. EXTENSIONS TO NONPLANAR GRAPHS

In this section we present a few extensions of our main result to certain classes of nonplanar graphs.

Allender et al. [2005] showed that, if given an embedding on the torus of a graph of genus 1, the *st*-connectivity problem is reducible (in deterministic log-space) to the planar case. As a consequence of our Main Theorem, we have the following.

COROLLARY 4.1. *The directed st-connectivity problem for graphs of genus 1 is in* UL ∩ coUL *(when given an embedding).*

Let $\mathcal{G}$ be a class of graphs in which reachability can be decided in complexity class $\mathcal{C}$. Let $G = (V, E) \in \mathcal{G}$. Let $G' = (V, E \cup E')$ be the graph G with an additional *auxiliary* edge set E' that is given as part of the input. We refer to G as the *main* graph and G' the *augmented* graph. We will show that if $|E'|$ is not too large, then reachability for the augmented graph can be decided in $\mathsf{L}^{\mathcal{C}}$.

THEOREM 4.2. *Let $G' = (V, E \cup E')$ be a graph such that reachability in $G = (V, E)$ can be decided in $\mathcal{C}$. Then, if $|E'| = \mathcal{O}(2^{\sqrt{\log n}})$, then reachability in G' can be decided in $\mathsf{L}^{\mathcal{C}}$.*

PROOF. The idea is to reduce reachability in G' to reachability in a smaller graph using reachability for the main graph as an oracle. Construct a graph whose vertices are labeled by edges in E' and there is directed edge from the vertex (u_1, u_2) to (v_1, v_2) in this graph if there is a path in G from u_2 to v_1. Since this new graph is only of size $\mathcal{O}(2^{\sqrt{\log n}})$, we can solve reachability in this graph deterministically in log-space using Savitch's theorem.

Formally, let $\mathsf{isPath}(x, y)$ be a boolean predicate that is true if there is a directed path $p : x \rightsquigarrow y$ in the main graph $G = (V, E)$ (i.e., there is a path $x \rightsquigarrow y$ that does not use auxiliary edges). By assumption, $\mathsf{isPath}(x, y)$ is computable in $\mathcal{C}$. Also, let $a_1 = (u_1, v_1), a_2 = (u_2, v_2), \ldots, a_m = (u_m, v_m)$ be the auxiliary edges (thus, $|E'| = m$).

We construct a new graph $\tilde{G} = (\tilde{V}, \tilde{E})$ as follows. Let

$$\begin{aligned} \tilde{V} &= \{v_{e_i} | e_i \in E'\} \cup \{\tilde{s}, \tilde{t}\} \\ \tilde{E} &= E_1 \cup E_2 \cup E_3 \end{aligned}$$

where

$$\begin{aligned} E_1 &= \{(\tilde{s}, y) \ \mid y = (u_j, v_j) \in E' \text{ and } \mathsf{isPath}(s, u_j) \text{ is true}\}, \\ E_2 &= \{(x, y) \ \mid x = (u_i, v_i), y = (u_j, v_j) \in E' \text{ and } \mathsf{isPath}(v_i, u_j) \text{ is true}\}, \\ E_3 &= \{(x, \tilde{t}) \ \mid x = (u_i, v_i) \in E' \text{ and } \mathsf{isPath}(u_j, t) \text{ is true}\}. \end{aligned}$$

Connectivity from $\tilde{s}$ to $\tilde{t}$ in $\tilde{G}$ can now be accomplished via application of Savitch's Theorem, which requires $\mathcal{O}(\log^2 m)$ space (recall that the size of $\tilde{G}$ is $m + 2$). For $m \leq \mathcal{O}(2^{\sqrt{\log n}})$, this simulation runs in space $\mathcal{O}(\log n)$.

It follows from the definition of $\tilde{G}$ that there is a path from s to t in G' if and only if there is a path from $\tilde{s}$ to $\tilde{t}$ in $\tilde{G}$. □

The statement of Theorem 4.2 is intentionally general. It is motivated by the possibility of extending our reachability result to non-planar graphs which may have one or more *crossing edges*. In particular, if we are given an embedding of a graph partitioned into a main graph that is planar and an auxiliary set of crossing edges, as long as there are not many crossing edges, then we can solve st-connectivity in this nonplanar graph in $\mathsf{UL} \cap \mathsf{coUL}$.

COROLLARY 4.3. *Let $\mathcal{G}$ be a class of directed graphs $G = (V, E \cup E')$ such that (V, E) is planar with an auxiliary crossing edge set E' (given separately) with $|E'| \leq \mathcal{O}(2^{\sqrt{\log n}})$ where $n = |V|$. Then, st-connectivity for any graph in $\mathcal{G}$ can be decided in $\mathsf{UL} \cap \mathsf{coUL}$.*

PROOF. Since (V, E) is planar, by Theorem 1.1 reachability queries in (V, E) can be decided in $\mathsf{UL} \cap \mathsf{coUL}$. By Theorem 4.2, reachability in G can be decided in $\mathsf{L}^{\mathsf{UL} \cap \mathsf{coUL}} = \mathsf{UL} \cap \mathsf{coUL}$. □

4.1 Completeness Results

We now discuss the possibility of extending our main result to grid graphs in three dimensions. As it turns out, reachability for a very restricted class of grid graphs in three dimensions, as well as for graphs of thickness-two, is complete for NL. Thus, extending our weight techniques to any of these classes of graphs will show that NL = UL.

A *three-dimensional grid graph* is a directed graph whose vertices are $[n] \times [n] \times [n]$ with edges connecting only immediate neighboring grid points. As before, we identify positive x and y directions with north and east and negative x and y directions with south and west, respectively. An edge in the positive z direction $((i, j, k) \rightarrow (i, j, k+1))$ is an *inward* edge and an edge in the negative z direction is an *outward* edge.

We call a three-dimensional grid graph *monotone* (3D-mGG) if there are only north, east and inward edges. We refer to the *st*-connectivity for such a graph as 3D-MGGR and show that it is complete for NL.

THEOREM 4.4. 3D-MGGR *is complete for* NL.

PROOF. We use the fact that the standard NL-complete reduction which generates the configuration graph of a log-space Turing machine can be easily modified to get a topologically sorted DAG. We simply prepend a timestamp to each configuration which results in a layered acyclic graph. Each layer can use the canonical ordering to induce a total topological order. Thus, without loss of generality, we will reduce such a DAG to a 3D monotone grid graph while preserving *st*-connectivity.

Let $G = (V, E)$ be a DAG with topologically sorted vertices $V = \{v_1, \ldots, v_n\}$. That is, if (v_i, v_j) is an edge, then $i < j$. We construct a 3D monotone grid graph G' as follows. For each vertex v_i, we make i copies at positions (i, i, k) for $k = 1, \ldots, i$. We also connect each of the i copies by an edge in the positive z-direction $((i, i, k) \rightarrow (i, i, k+1)$ for $k = 1, \ldots, i-1)$. This encodes the notion that if there is a path from any copy of a vertex v_i to a copy of a vertex v_k, then there is a path from that copy of v_i to (k, k, k).

The kth xy plane (i.e., the set of vertices $\{(i, j, k) \mid 1 \leq i, j \leq n\}$) encodes edges to and from vertex v_k. We start by adding the path leading in an eastward direction from (k, k, k); $(l, k, k) \rightarrow (l+1, k, k)$ for $l = k, ..., n-1$, so that there is a path from (k, k, k) to (l, k, k) for each $l > k$ (corresponding to all of the possible vertices v_l for which there might be an edge from v_k). Each actual edge (v_k, v_l) is encoded by a path from (l, k, k) north to (l, l, k) (i.e., to the copy of v_l located in the kth xy plane). It is easy to see that there is an edge from v_k to v_l if and only if there is a path in the kth xy plane from (k, k, k) to the copy of v_l in this plane. An example of this construction from a complete DAG of size 4 can be found in Figure 3 (note that we identify the y axis in the vertical while the z axis extends toward the horizon).

The resulting graph is bounded within the $n \times n \times n$ cube. Furthermore, since each edge only requires an index look-up, the construction is clearly log-space computable.

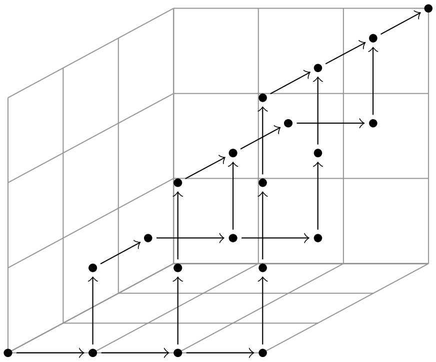

Fig. 3. A mapped graph resulting from a complete DAG on $n = 4$ vertices.

Finally, without loss of generality we can assume that $s = v_1$ and $t = v_n$ and so we map s to the single copy of v_1 and t to the highest numbered copy of v_n, located at (n, n, n). We claim that there exists a path $s \rightsquigarrow t$ in G if and only if there exists a path $(1, 1, 1) \rightsquigarrow (n, n, n)$ in G'. The construction clearly preserves *st*-connectivity. □

This reduction can be further modified to give us a characterization of NL in terms of graph *thickness*. The usual graph-theoretic notion of thickness of a graph G is defined as the minimal number of planar subgraphs whose union is G [Gibbons 1985]. Intuitively, we can think of thickness as the minimal number of transparencies required to draw the graph so that no edges cross within any single transparency. Clearly, a graph is planar if and only if it has thickness-one. Surprisingly, however, thickness-two suffices to capture all of NL.

We'll actually show that completeness holds for an even more restrictive notion of thickness called *geometric thickness* [Hutchinson et al. 1995; Dillencourt et al. 2000]. The geometric thickness of a graph G is defined as the minimal number k such that we can assign planar point locations to the vertices of G, represent each edge as a line segment, and assign each edge to one of k transparencies so that no two lines cross in any one transparency. The difference between these two notions is that geometric thickness requires that all vertex placements be consistent across all transparencies.

THEOREM 4.5. *The st-connectivity problem for (geometric) thickness-two graphs is complete for* NL. *Moreover, each transparency is a monotone grid graph.*

PROOF. We will make use of the 3D monotone grid graph that results by applying the reduction in Theorem 4.4. We start by embedding each xy-layer (identified as L_k, $1 \leq k \leq n$) in the first transparency. We do so by laying each layer above the previous layer, shifting it one unit eastward. That is, the lower left corner of each layer L_k is mapped to $(k, (k-1)n + 1)$ while the upper-right corner is mapped to $((k + n - 1), nk)$. This results in a $2(n - 1) \times n^2 - 1$ sized grid.

We now embed the inward z-edges using the second transparency. We will do so by routing them inside the grid defined by the xy planes. In order to do this, we first expand the grid by 3: each unit square is replaced by a 3×3 grid, leading to a *fine-grid*. Thus, each (i, j) coordinate in the first grid maps to

$$\big((3i-2),(3j-2)\big)$$

in the fine grid.

We now have room to route the z-edges through the second transparency. Consider the z edges between layer L_k and L_{k+1}: $(i, i, k) \rightarrow (i, i, k+1)$ for $i = k+1, \ldots, n$ in the original 3D-mGG. The initial and final vertex get mapped to

$$((k+(i-1)),(k-1)n+1+(i-1)) = (k+i-1,(k-1)n+i)$$

and

$$((k+1)+(i-1),(k+1-1)n+1+(i-1)) = (k+i, nk+i)$$

respectively. In the expanded fine grid they are located at

$$\big(3[k+i-1]-2, 3[(k-1)n+i]-2\big)$$

and

$$\big(3[k+i]-2, 3[nk+i]-2\big),$$

respectively. We will capture the connectivity of this edge by routing a path between these two grid points on the second transparency, avoiding contact with other z-edge/paths in the second transparency. First, we travel east one edge in the fine-grid:

$$\big(3[k+i-1]-2+1, 3[(k-1)n+i]-2\big) = \big(3[k+i-1]-1, 3[(k-1)n+i]-2\big).$$

We then travel north until we have cleared the sub-grid corresponding to L_k; that is to

$$y = 3nk-1$$

on the fine-grid. We then travel east again for one edge;

$$\big(3[k+i-1]-1+1, 3nk-1\big) = \big(3[k+i-1], 3nk-1\big)$$

and continue north again to the same row as the final vertex:

$$y = 3[nk+i]-2.$$

At this point, we simply travel east again one more edge and arrive at

$$\big(3[k+i-1]+1, 3[nk+i]-2\big) = \big(3[k+i]-2, 3[nk+i]-2\big)$$

the intended final vertex.

This reduction is illustrated (cf. Figure 4) for the first few layers resulting from the complete DAG on 4 vertices from Figure 3.

It is not difficult to see that the reduction results in only two transparencies each of which avoids any edge crossings. Moreover, the reduction is clearly log-space computable. □

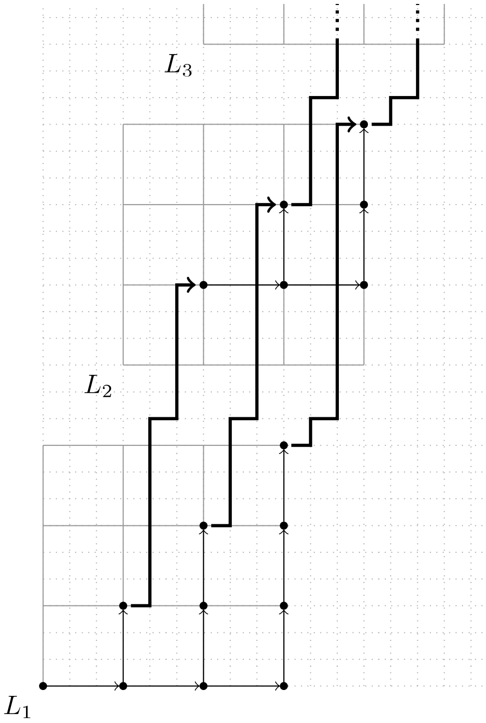

Fig. 4. First two layers of the thickness-two reduction on the DAG in Figure 3. Lighter directed edges correspond to the first transparency while the darker (routing) paths are on the second transparency.

In fact, the reduction in Theorem 4.5 is even stronger: each transparency is actually a directed forest embedded as a monotone grid graph.

Extending our Main Theorem by defining a weight function for 3D monotone grid graphs or thickness-two graphs appears difficult however. One of the key arguments used in our proof exploits the fact that equal weight paths necessarily intersect on the 2D grid. This is not necessarily true in the 3D case as the obvious extensions of our weight function allow equal weight paths that do not intersect.

5. CONCLUSION

We have shown that the *st*-connectivity problem for directed planar graphs can be decided in UL ∩ coUL, improving over the known upper bound of NL. We have also given several extensions to nonplanar graphs and some completeness results.

The most direct and important open question is to show that NL = UL unconditionally. We believe that the result of this paper is a definite step towards solving this problem. We gave an easy log-space reduction from general directed graph reachability to reachability in graphs with thickness two. Can we reduce general directed graph reachability to PLANARREACHor even to reachability in graphs with genus one? This would show NL = UL. On the other hand, as there are no known complete problems for UL∩coUL, there might be a better upper bound for PLANARREACH. Specifically, can we show PLANARREACH∈ L?

Another general question is to place more problems of interest in UL. Very recently, there has been some progress reported in this direction. After the publication of the conference version of this paper, our main result was used to establish new, similar upper bounds. Thierauf and Wagner [2008] extended our result to show that shortest distance in a planar graphs can also be computed in UL. They use this fact to show that the isomorphism problem for planar 3-connected graphs can be decided in UL ∩ coUL. Limaye et al. [2009] have shown that the longest path problem for planar DAGs also is solvable in UL. Finally, Datta et al. [2007; 2008] use similar weighting techniques to establish improved upper bounds for bipartite planar matching problems.

ACKNOWLEDGMENTS

The third author is very grateful to Eric Allender for inviting him to Rutgers in the summer of 2006 where he was inspired to work on this problem. The authors thank Meena Mahajan for interesting email discussions and valuable comments on an earlier draft of the paper. We also thank several anonymous reviewers for their suggestions.

REFERENCES

ALLENDER, E., BARRINGTON, D. A. M., CHAKRABORTY, T., DATTA, S., AND ROY, S. 2006. Grid graph reachability problems. In *Proceedings of the Annual IEEE Conference on Computational Complexity*. IEEE Computer Society Press, Los Alamitos, CA. 299–313.

ALLENDER, E., DATTA, S., AND ROY, S. 2005. The directed planar reachability problem. In *Proceedings of the 25th Annual Conference on Foundations of Software Technology and Theoretical Computer Science*. Springer-Verlag, Berlin, Germany.

ALLENDER, E. AND LANGE, K.-J. 1998. $\mathsf{RUSPACE}(\log n) \subseteq \mathsf{DSPACE}(\log^2 n/\log\log n)$. *Theory of Computing Systems 31*, 539–550. (Special issue devoted to the 7th Annual International Symposium on Algorithms and Computation (ISAAC'96).)

ALLENDER, E. AND MAHAJAN, M. 2004. The complexity of planarity testing. *Inf. Comput. 189*, 117–134.

ALLENDER, E., REINHARDT, K., AND ZHOU, S. 1999. Isolation, matching, and counting: Uniform and nonuniform upper bounds. *J. Comput. Syst. Sci. 59*, 164–181.

ÀLVAREZ, C. AND JENNER, B. 1993. A very hard log-space counting class. *Theoret. Comput. Sci. 107*, 3–30.

BARRINGTON, D. A. M. 1989. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. *J. Comput. Syst. Sci. 38*, 150–164.

BARRINGTON, D. A. M., LU, C.-J., MILTERSEN, P. B., AND SKYUM, S. 1998. Searching constant width mazes captures the AC^0 hierarchy. In *Proceedings of the 15th International Symposium on Theoretical Aspects of Computer Science (STACS)*. Lecture Notes in Computer Science, vol. 1373. Springer-Verlag, Berlin, Germany. 74–83.

BLUM, M. AND KOZEN, D. 1978. On the power of the compass (or, why mazes are easier to search than graphs). In *Proceedings of the IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society Press, Los Alamitos, CA. 132–142.

BUNTROCK, G., JENNER, B., LANGE, K.-J., AND ROSSMANITH, P. 1991. Unambiguity and fewness for logarithmic space. In *Proceedings of the 8th International Conference on Fundamentals of Computation Theory (FCT'91)*. Lecture Notes in Computer Science, vol. 529. Springer-Verlag, Berlin, Germany. 168–179.

DATTA, S., KULKARNI, R., LIMAYE, N., AND MAHAJAN, M. 2007. Planarity, determinants, permanents, and (unique) perfect matchings. In *Proceedings of 2nd International Computer Science Symposium in Russia CSR*. Lecture Notes in Computer Science, vol. 4649. Springer-Verlag, New York. 115–126.

DATTA, S., KULKARNI, R., AND ROY, S. 2008. Deterministically isolating a perfect matching in bipartite planar graphs. In *Proceedings of the 25th International Symposium on Theoretical Aspects of Computer Science*. pp. 229–240, www.stacs-conf.org.

DILLENCOURT, M. B., EPPSTEIN, D., AND HIRSCHBERG, D. S. 2000. Geometric thickness of complete graphs. *J. Graph Algor. Appl. 4*, 3, 5–17.

ETESSAMI, K. 1997. Counting quantifiers, successor relations, and logarithmic space. *J. Comput. Syst. Sci. 54*, 3 (June), 400–411.

GIBBONS, A. 1985. *Algorithmic Graph Theory*. Cambridge University Press, Cambridge, UK.

GROLLMAN, J. AND SELMAN, A. L. 1988. Complexity measures for public-key cryptosystems. *SIAM J. Comput. 17*, 309–335.

HUTCHINSON, J. P., SHERMER, T. C., AND VINCE, A. 1995. On representations of some thickness-two graphs. In *GD'95: Proceedings of the Symposium on Graph Drawing*. Springer-Verlag, Berlin, Germany. 324–332.

IMMERMAN, N. 1988. Nondeterministic space is closed under complement. *SIAM J. Comput. 17*, 935–938.

LANGE, K.-J. 1997. An unambiguous class possessing a complete set. In *Proceedings of the 14th International Symposium on Theoretical Aspects of Computer Science (STACS)*. Springer-Verlag, Berlin, Germany. 339–350.

LIMAYE, N., MAHAJAN, M., AND NIMBHORKAR, P. 2009. Longest path in planar DAGs in unambiguous logspace. In *Proceedings of the Australasian Theory Symposium (ATS)*, to appear.

MULMULEY, K., VAZIRANI, U., AND VAZIRANI, V. 1987. Matching is as easy as matrix inversion. *Combinatorica 7*, 105–113.

REINGOLD, O. 2005. Undirected st-connectivity in log-space. In *STOC'05: Proceedings of the 37th Annual ACM Symposium on Theory of Computing*. IEEE Computer Society Press, Los Alamitos, CA. 376–385.

REINHARDT, K. AND ALLENDER, E. 2000. Making nondeterminism unambiguous. *SIAM J. Comput. 29*, 1118–1131. (An earlier version appeared in FOCS 1997, pp. 244–253).

SZELEPCSÉNYI, R. 1988. The method of forced enumeration for nondeterministic automata. *Acta Inf. 26*, 279–284.

THIERAUF, T. AND WAGNER, F. 2008. The isomorphism problem for planar 3-connected graphs is in unambiguous logspace. In *Proceedings of the 25th International Symposium on Theoretical Aspects of Computer Science*, pp. 633–644, www.stacs-conf.org.

VALIANT, L. 1976. The relative complexity of checking and evaluating. *Inf. Proc. Lett. 5*, 20–23.

Received July 2008; revised July 2008; accepted August 2008