# S-T Connectivity on Digraphs with a Known Stationary Distribution*

Kai-Min Chung†
Harvard University
School of Engineering and Applied Science
33 Oxford St, Cambridge, MA
kmchung@fas.harvard.edu

Omer Reingold‡
Weizmann Institute of Science
Dept. of Computer Science
Rehovot 76100, Israel
omer.reingold@weizmann.ac.il

Salil Vadhan §
Harvard University
School of Engineering and Applied Science
33 Oxford St, Cambridge, MA
salil@eecs.harvard.edu

## Abstract

*We present a deterministic logspace algorithm for solving* S-T CONNECTIVITY *on directed graphs if (i) we are given a stationary distribution for random walk on the graph and (ii) the random walk which starts at the source vertex $s$ has polynomial mixing time. This result generalizes the recent deterministic logspace algorithm for* S-T CONNECTIVITY *on undirected graphs [15]. It identifies knowledge of the stationary distribution as the gap between the* S-T CONNECTIVITY *problems we know how to solve in logspace (*$\mathbf{L}$*) and those that capture all of randomized logspace (*$\mathbf{RL}$*).*

## 1 Introduction

There is a long and beautiful line of work in complexity theory, starting with [3, 21, 13] giving evidence that randomized algorithms are not much more powerful than deterministic algorithms. That is, under a variety of natural complexity assumptions, every randomized algorithm can be fully derandomized with only a small loss in efficiency (e.g. time and space). Like many research directions in complexity theory, a major long-term goal is to obtain similar results *unconditionally*. Unfortunately, recent results loosely show that, when we measure efficiency by *time*, derandomization (e.g. $\mathbf{BPP} = \mathbf{P}$) implies superpolynomial circuit lower bounds (for $\mathbf{NEXP}$), and thus unconditional results may be out of reach [7, 9].

However, when we measure efficiency by *space*, it seems that there is hope for unconditional derandomization, even showing that $\mathbf{RL} = \mathbf{L}$. Indeed, there are highly nontrivial and unconditional deterministic simulations of $\mathbf{RL}$. Most notably, using Nisan's pseudorandom generator for logspace computation [12], Saks and Zhou [17] showed that $\mathbf{RL} \subseteq \mathbf{L}^{3/2}$, where $\mathbf{L}^{3/2}$ denotes the class of problems solvable in space $O(\log^{3/2} n)$. But proving $\mathbf{RL} = \mathbf{L}$ has remained elusive; in fact, there has been no improvement to the Saks and Zhou theorem in over a decade.

Hope for further progress on $\mathbf{RL}$ vs. $\mathbf{L}$ was recently renewed, when Reingold [15] showed how to fully derandomize the classic and most notable example of an $\mathbf{RL}$ algorithm, namely the random-walk algorithm of [1] for UNDIRECTED S-T CONNECTIVITY. (Independently, Trifonov [20] gave an deterministic algorithm for this problem using space $O(\log n \cdot \log\log n)$.) It is well-known that general $\mathbf{RL}$ computations can be viewed as some restricted form of the S-T CONNECTIVITY problem on *directed* graphs (a.k.a. *digraphs*). (S-T CONNECTIVITY on general digraphs is $\mathbf{NL}$-complete, and $\mathbf{RL}$ algorithms correspond to a restricted class of $\mathbf{NL}$ algorithms.) Thus, one can attack the $\mathbf{RL}$ vs. $\mathbf{L}$ question by trying to close the gap between undirected graphs (solvable

*A full version of this paper is available in ECCC [4]

†Supported by NSF grant CCF-0133096.
‡Supported by US-Israel BSF grant 2002246.
§Supported by US-Israel BSF grant 2002246, NSF grant CCF-0133096, and ONR grant N00014-04-1-047.

Twenty-Second Annual IEEE Conference on Computational Complexity (CCC'07)
0-7695-2780-9/07 $20.00 © 2007 IEEE

in $\mathbf{L}$ by [15]) and the types of digraphs corresponding to $\mathbf{RL}$ machines. This approach was pursued in [16].

The first question in this approach is to identify a class of digraphs whose S-T CONNECTIVITY problems capture $\mathbf{RL}$. In [16], it was shown that S-T CONNECTIVITY on digraphs where the random walk converges to the stationary distribution in a polynomial number of steps is complete for $\mathbf{RL}$. For short, we refer to such graphs as *poly-mixing*, and the resulting computational problem as POLY-MIXING S-T CONNECTIVITY.[1] The poly-mixing condition captures what is needed for the random-walk algorithm of [1] to work.[2] Consequently, proving $\mathbf{RL} = \mathbf{L}$ amounts to derandomizing this algorithm, and we may hope to do so by closing the gap between poly-mixing graphs and undirected graphs.

There are two general ways we might hope to place POLY-MIXING S-T CONNECTIVITY in $\mathbf{L}$, corresponding to two common settings for derandomization in general. In the *explicit setting*, we design an algorithm that is given full access to the input graph and can do arbitrary logspace computations on it. This is the most general approach, in the sense that it is equivalent to proving $\mathbf{RL} = \mathbf{L}$. However, many derandomization results are actually done in a more restricted *oblivious setting*. Here the algorithm is not given explicit access to the input graph. Instead, based on just the size and degree of the input graph, it generates "pseudorandom" bits to be used in the randomized algorithm. If the pseudorandom bits are generated from a short seed, then we can get a deterministic algorithm by enumerating all seeds. For example, any pseudorandom generator for space-bounded computation, such as Nisan's [12], yields an oblivious derandomization. (But Nisan's pseudorandom generator does not imply $\mathbf{RL} = \mathbf{L}$ because the seed length is $O(\log^2 n)$ rather than $O(\log n)$.) As noted in [16], to put POLY-MIXING S-T CONNECTIVITY (and hence all of $\mathbf{RL}$) in $\mathbf{L}$, a somewhat weaker notion of pseudorandom generator suffices. Specifically, we only need a method for generating pseudorandom walks on poly-mixing graphs that ensures that the final vertex is distributed close to the stationary distribution; we refer to such a generator as a *pseudorandom walk generator*. Oblivious methods for derandomization tend to be interesting in their own right, and have many applications beyond just proving $\mathbf{RL} = \mathbf{L}$, such as [8, 10, 6, 19]. However, they can be harder to obtain. For example, the derandomization of Saks and Zhou [17] is not oblivious.

The main results of [16] concern the oblivious setting. First, extending the techniques of [15], they exhibit a pseudorandom walk generator for *regular* digraphs that are *consistently labelled*. Regular means that all the in-degrees and out-degrees are the same, and consistently labelled means that it is never the case that the $i$'th neighbor of $u$ is the same as the the $i$'th neighbor of $v$ for distinct vertices $u$ and $v$. Second, they show that a pseudorandom walk generator for arbitrarily labelled regular digraphs implies a pseudorandom walk generator for poly-mixing digraphs, and thus $\mathbf{RL} = \mathbf{L}$. Thus, dealing with inconsistent labelling is the "only" obstacle to proving $\mathbf{RL} = \mathbf{L}$ in the oblivious setting.

In this work, we focus on the explicit setting. Recall that Reingold [15] gave a deterministic logspace algorithm for UNDIRECTED S-T CONNECTIVITY in this setting. In [16], this was extended to (arbitrarily labelled) regular digraphs, and more generally Eulerian digraphs (where every vertex has the same in-degree as out-degree). This result is obtained by noting that there is a simple reduction from S-T CONNECTIVITY in Eulerian digraphs to S-T CONNECTIVITY in consistently labelled regular digraphs, and then applying the pseudorandom walk generator for consistently labelled regular digraphs mentioned above. Thus, after [15, 16], the gap in the explicit setting is between regular or Eulerian digraphs (which are in $\mathbf{L}$) and general poly-mixing digraphs (which are complete for $\mathbf{RL}$).

Regular and Eulerian digraphs have a few properties not shared by general poly-mixing digraphs. It is easy to obtain a stationary distribution for the random walk on such graphs: the uniform distribution in the case of regular graphs, and assigning each vertex mass proportional to its degree in the case of Eulerian digraphs. In addition, this stationary distribution can be computed exactly in logspace and every vertex has non-negligible probability in it (at least $1/(\#\text{edges})$).

In this work, we show that in general, knowing a stationary distribution of the random walk is sufficient to solve POLY-MIXING S-T CONNECTIVITY in deterministic logspace. That is, we consider a further restriction of POLY-MIXING S-T CONNECTIVITY, where we are given the stationary probabilities as part of the input, and show that the resulting (promise) problem, KNOWN-STATIONARY S-T CONNECTIVITY, is in $\mathbf{L}$. We allow the possibility that some vertices have exponentially small stationary probability, and the estimates only need to be accurate to within a $1/\text{poly}(n)$ additive error. We view this result as clarifying the property that makes Reingold's algorithm

[1] Technically, this is a *promise problem*, and thus is complete for the promise-problem analogue of $\mathbf{RL}$.

[2] Technically, we also require that $s$ and $t$ have non-negligible stationary probability, but only require fast mixing on the strongly connected component containing $s$.

Twenty-Second Annual IEEE Conference on Computational Complexity (CCC'07)
0-7695-2780-9/07 $20.00 © 2007 IEEE

and its generalizations possible, and suggesting that future attempts to prove $\mathbf{RL} = \mathbf{L}$ might focus on dealing with unknown stationary distributions.

| Problem | Oblivious | Explicit |
|---|---|---|
| Regular digraph with consistent labelling | $\mathbf{L}$ | $\mathbf{L}$ |
| Regular digraph with arbitrary labelling | $\mathbf{RL}$ | $\mathbf{L}$ |
| Poly-mixing digraph with known stationary distribution | $\mathbf{RL}$ | $\mathbf{L}$ (new) |
| Poly-mixing digraph ($\mathbf{RL}$ complete) | $\mathbf{RL}$ | $\mathbf{RL}$ |

**Table 1.** S-T CONNECTIVITY **problems and $\mathbf{RL}$ vs. $\mathbf{L}$**

An entry of $\mathbf{L}$ means that there is a deterministic logspace solution for the given class of graphs in the given setting (oblivious or explicit). An entry of $\mathbf{RL}$ means that such a solution would imply $\mathbf{RL} = \mathbf{L}$.

It is interesting to note the very different behavior of the oblivious setting and explicit setting, as summarized in Table 1. Our result shows that in the explicit setting, the gap between $\mathbf{L}$ and $\mathbf{RL}$ centers around whether the stationary distribution is known. But in the oblivious setting, this is not an essential property, as the results of [16] show that handling (arbitrarily labelled) regular digraphs, where the stationary distribution is *uniform*, suffices to solve all of $\mathbf{RL}$. Instead, in the oblivious setting the key property seems to be consistent labelling; note that this property is irrelevant for the explicit setting, where there is a simple reduction from arbitrarily labelled regular graphs to consistently labelled ones.

The idea of restricting to the case that estimates of the stationary probabilities are known is inspired by the work of Raz and Reingold [14], who studied derandomization of $\mathbf{RL}$ machines when estimates of the *state* probabilities of the $\mathbf{RL}$ machine are known. Our model and results are incomparable to those of [14]. They require estimates of the probabilities on walks of every length (in a layered graph), whereas we only require the estimates of the long-term behavior (in a poly-mixing graph). On the other hand, they require only weak multiplicative estimates of the probabilities, whereas we require good additive estimates. Finally, they only derandomize walks of length roughly $2^{\sqrt{\log n}}$, whereas our work allows the walk length/mixing time to be $\mathrm{poly}(n)$.

Our algorithm for KNOWN-STATIONARY S-T CONNECTIVITY is obtained by giving a logspace reduction from the case of poly-mixing digraphs with known stationary probabilities to the case of nearly regular digraphs, and then showing that the algorithm of [16] works even if the graph is nearly regular. This reduction is inspired by the result of [16] showing that a pseudorandom walk generator for (arbitrarily labelled) regular digraphs implies a pseudorandom walk generator for all poly-mixing digraphs. The proof of their theorem works by showing that every poly-mixing digraph can be 'blown up' to a regular digraph such that pseudorandom walks on the regular digraph project down to pseudorandom walks on the poly-mixing digraph. The 'blow up' procedure of [16] is only done in the analysis, and thus need not be computable in logspace (the logspace algorithm only needs to do the projection of walks, which is very simple). Much of the work in our result is in showing that a similar blow up can in fact be done in logspace if estimates of the stationary probabilities are known. To do this, we need to find alternatives to some of the steps taken in the construction of [16], and settle for getting a nearly regular rather than exactly regular graph at the end.

**Organization.** The rest of the paper is organized as follows. We discuss technical preliminaries about random walks on digraphs and Markov chains in Section 2. In Section 3, we give the formal statement of our main result and a high-level overview of the proof. Finally, we present the proof of the main theorem in Section 4. Some proofs are omitted due to space constraint; they can be found in the full version [4].

## 2 Preliminaries

In this paper, we consider *directed graphs* (*digraphs* for short) $G = ([n], E)$, and allow them to have multiple edges and self-loops. A graph $G$ is *outregular* if every vertex has the same number $d$ of edges leaving it; $d$ is called the *out-degree*. $G$ is *regular* if it is both outregular and inregular. We say $G$ is a *$d$-(out)regular graph* if $G$ is a $n$-vertex (out)regular graph of (out-)degree $d$.

Given a graph $G$ on $n$ vertices, we consider the random walk on $G$ described by the transition matrix $M_G$ whose $(u, v)$'th entry equals the number of edges from $u$ to $v$, divided by the out-degree of $v$. $M_G$ is a *Markov chain* on state space $[n]$. Since we are interested in random walks on graphs, when we refer to a Markov chain $M$, there is always an underlying graph $G$ with $M_G = M$.

We say $M$ is *$a$-lazy* if $M(v, v) \geq a$ for every $v$, and $M$ is *lazy* if $M$ is $a$-lazy for some $a > 0$. A distribution $\pi$ on $[n]$ is *stationary* for $M$ if $\pi M = \pi$.

For a distribution $\alpha$ on $[n]$, denote the *support* of $\alpha$ by $\mathrm{supp}(\alpha) \stackrel{\mathrm{def}}{=} \{v : \alpha(v) > 0\}$. Note that the support of a stationary distribution is always the union of disjoint strongly connected components since stationary implies that if there is a path from $u \in \mathrm{supp}(\pi)$ to $v$, then there is also a path from $v$ to $u$.

We are interested in the rate at which a Markov chain $M$ converges to a stationary distribution. In terms of random walk on the graph, how many steps does it take to reach a stationary distribution? It is well-known that for undirected graphs, the rate of convergence is characterized by the second largest (in absolute value) eigenvalue $\lambda_2(M)$ of the matrix $M$. More precisely, let $\alpha_t$ denote the distribution of a random walk after $t$-th step, and $\pi$ be the stationary distribution $\alpha^t$ converges to, then the variation distance of $\alpha_t$ to $\pi$ will decrease in the rate $\lambda_2(M)^t$ (For two distributions $\alpha, \beta$ on $[n]$, their *variation distance* is $\Delta(\alpha, \beta) \stackrel{\mathrm{def}}{=} (1/2)\sum_v |\alpha(v) - \beta(v)|$.)

However, for directed graphs, $\lambda_2(M)$ may even not exist. To estimate the mixing time, Mihail [11] and Fill [5] introduce a generalized parameter, which we call the *spectral expansion* $\lambda_\pi(M)$, and is equal to $\lambda_2(M)$ when $G$ is undirected.

**Definition 2.1** *Let $M$ be a Markov chain and $\pi$ a stationary distribution for $M$. We define the* spectral expansion *of $M$ with respect to $\pi$ to be*

$$\lambda_\pi(M) \stackrel{\mathrm{def}}{=} \max_{x \in \mathbb{R}^n : \sum_{v \in \mathrm{supp}(\pi)} x(v) = 0} \frac{\|xM\|_\pi}{\|x\|_\pi},$$

*where $\|x\|_\pi \stackrel{\mathrm{def}}{=} \sum_{v \in \mathrm{supp}(\pi)} x(v)^2/\pi(v)$.*

The following lemma shows that, like $\lambda_2$, $\lambda_\pi$ measures the rate of convergence to the stationary distribution $\pi$.

**Lemma 2.2 (cf. [16])** *Let $M$ be a Markov chain on $[n]$ and $\pi$ a stationary distribution with $\lambda_\pi(M) < 1$. Let $\alpha^t$ denote the distribution of a random walk after $t$ steps starting from distribution $\alpha^0$ with $\mathrm{supp}(\alpha^0) \subset \mathrm{supp}(\pi)$. Then,*

$$\Delta(\alpha^t, \pi) \le \lambda_\pi(M)^t \cdot \|\alpha^0 - \pi\|_\pi.$$

*In particular, the walk $\alpha^t$ converges to $\pi$.*

Since the above lemma implies that random walks starting at *any* vertex in $\mathrm{supp}(\pi)$ converge to $\pi$, it follows that $\mathrm{supp}(\pi)$ consists of a single strongly connected component and that $\pi$ is the unique stationary distribution supported on this component.

Sometimes it is convenient to use the *spectral gap* $\gamma_\pi(M) \stackrel{\mathrm{def}}{=} 1 - \lambda_\pi(M)$. We will often bound $\lambda_\pi(M)$ (or $\gamma_\pi(M)$) by the *conductance* of $M$.

**Definition 2.3** *Let $M$ be a Markov chain with $n$ vertices and $\pi$ a stationary distribution. The* conductance *of $M$ with respect to $\pi$ is defined to be*

$$h_\pi(M) \stackrel{\mathrm{def}}{=} \min_{A : 0 < \pi(A) \le 1/2} \frac{\sum_{u \in A, v \notin A} \pi(u) M(u,v)}{\pi(A)}.$$

Observe that the denominator $\pi(A)$ is the probability mass contained in $A$, and the numerator $\sum_{u \in A, v \notin A} \pi(u) M(u,v)$ is the probability mass flowing out from $A$. The conductance is a lower bound of the fraction of probability mass in $A$ leaving $A$. Intuitively, if the conductance is large, then the probability mass will mix quickly. Indeed, the following lemma formalizes this intuition.

**Lemma 2.4 ([18, 11, 5])** *Let $M$ be a connected,* $1/2$-lazy *Markov chain and $\pi$ a stationary distribution. Then $\gamma_\pi(M) \ge h_\pi(M)^2/2$.*

To estimate the conductance, we introduce another useful measure of mixing time which is implicitly used in [16].

**Definition 2.5** *Let $M$ be a Markov chain, and $s$ be a vertex of $M$. The* visiting length *of $s$, denoted $\ell_s(M)$, is the smallest number $\ell$ such that for every vertex $v$ reachable from $s$, a random walk of length $\ell$ from $v$ visits $s$ with probability at least $1/2$.*

**Lemma 2.6 ([16])** *Let $M$ be a $1/2$-lazy Markov chain. Let $s$ be a vertex of $M$ with visiting length $\ell$. Then $M$ has a stationary distribution $\pi$ (with $s \in \mathrm{supp}(\pi)$) such that the conductance satisfies $h_\pi(G) \ge 1/(2\ell)$, and hence the spectral gap satisfies $\gamma_\pi(G) \ge 1/(8\ell^2)$.*

Since spectral gap and visiting length are both measures of mixing time, it is not surprising that we can also bound visiting length by spectral gap.

**Lemma 2.7 (implicit in [16])** *Let $M$ be a Markov chain with $n$ vertices such that the underlying graph is $d$-regular, and $\pi$ be a stationary distribution with $\gamma_\pi(M) > 1/k$. Let $s$ be a vertex of $M$ with $\pi(s) > 1/k$, then the visiting length $\ell_s(M)$ is at most $O(nk^3 \log d)$.*

## 3 Main theorem and proof overview

Our main result is a deterministic logspace algorithm to solve S-T CONNECTIVITY problem for digraphs with polynomial mixing time when a good approximation of a stationary distribution is available. Formally, we study the following problems, and solve them in deterministic logspace.

$\delta$-KNOWN-STATIONARY S-T CONNECTIVITY:

- **Input:** $(G, p_1, \ldots, p_n, s, t, 1^k)$, where $G = ([n], E)$ is a $d$-outregular digraph, $s, t \in [n] = V$, and $k \in \mathbb{N}$
- **YES instances:**
  1. There is a stationary distribution $\pi$ such that $\pi(s), \pi(t) \geq 1/k$, and for each $v \in [n]$ that can reach $s$, $|p_v - \pi(v)| \leq \delta$.
  2. If we let $\pi_s$ be the restriction of $\pi$ to the strongly connected component of $\mathrm{supp}(\pi)$ containing $s$,[3] then $\gamma_{\pi_s}(G), \pi_s(s), \pi_s(t) \geq 1/k$.
- **NO instances:** There is no path from $s$ to $t$ in $G$.

$\delta$-KNOWN-STATIONARY FIND PATH:

- **Input:** $(G, p_1, \ldots, p_n, s, t, 1^k)$, where $G = ([n], E)$ is a $d$-outregular digraph, $s, t \in [n] = V$, and $k \in \mathbb{N}$
- **Promise:**
  1. There is a stationary distribution $\pi$ such that $\pi(s), \pi(t) \geq 1/k$, and for each $v \in [n]$ that can reach $s$, $|p_v - \pi(v)| \leq \delta$.
  2. If we let $\pi_s$ be the restriction of $\pi$ to the strongly connected component of $\mathrm{supp}(\pi)$ containing $s$, then $\gamma_{\pi_s}(G), \pi_s(s), \pi_s(t) \geq 1/k$.
- **Output:** A path from $s$ to $t$ in $G$.

In both of the above problems, $(p_1, \ldots, p_n)$ is called the *input stationary distribution*, and $\delta$ can be a function of the input parameters $n, d$, and $k$. We note that if we remove Condition 1 (regarding the accuracy of the stationary distribution), then the resulting problems, POLY-MIXING S-T CONNECTIVITY and POLY-MIXING FIND PATH become complete for the promise and search version of $\mathbf{RL}$, respectively [16].

Note that the input stationary distribution $(p_1, \ldots, p_n)$ does not necessarily reveal the solution to the decision problem, because $(p_1, \ldots, p_n)$ can be arbitrary on NO instances (in particular, $p_t$ can be larger than $1/k$). Moreover, even if we required Condition 1 (regarding the accuracy of the stationary distribution) to hold on NO instances, $p_t$ could be arbitrary in case that there is no path from $t$ to $s$. However, when there is a path from $t$ to $s$, but no path from $s$ to $t$, any stationary distribution $\pi$ has to have $\pi(t) = 0$, and so $p_t \leq \delta$. In this case the decision problem becomes trivial, but the search version is still interesting. We remark that the reduction from arbitrary $\mathbf{RL}$ problems to POLY-MIXING S-T CONNECTIVITY presented in [16] always gives such instances.

[3] That is, if $S \subset \mathrm{supp}(\pi)$ is the strongly connected component containing $s$, then $\pi_s(v) = \pi(v)/(\sum_{w \in S} \pi(w))$

**Theorem 3.1** *There is a polynomial $p$ such that $(1/p(n,d,k))$-KNOWN-STATIONARY S-T CONNECTIVITY and $(1/p(n,d,k))$-KNOWN-STATIONARY FIND PATH can be solved in logarithmic space.*

To prove Theorem 3.1, it suffices to provide a deterministic logspace algorithm for KNOWN-STATIONARY FIND PATH as we can check whether the path found leads from $s$ to $t$ in order to decide KNOWN-STATIONARY S-T CONNECTIVITY. Let $G$ be an input graph satisfying the promise. Note that the promise implies that $\mathrm{supp}(\pi_s)$ is a strongly connected component containing $s$ and $t$. Our goal is to find a path from $s$ to $t$ in $G$. To simplify the presentation, we first set $\delta = 0$, and use $\pi(\cdot)$ to denote the input stationary distribution. We will explain why the proof still works for some $\delta = 1/\mathrm{poly}(n, d, k)$ at the end.

Recall the idea mentioned in introduction. We first blow up $G$ to a graph $G''$ that is "close" to a consistently labelled regular graph $G_{\mathrm{con}}$, and then apply the pseudorandom walk generator of [16] for consistently labelled regular graphs to $G''$. The pseudorandom walk generator will produce a path in $G''$ which can be projected to a path in $G$, which will visit $t$ with non-negligible probability. Since the pseudorandom walk generator uses a seed of logarithmic length, we can enumerate all possible seeds and find a path from $s$ to $t$ in deterministic logspace.

Although the idea is natural, the construction and analysis are somewhat delicate. The main challenge is that we need to preserve the mixing time throughout the construction. Since the pseudorandom walk generator works for consistently labelled regular graphs, but $G''$ is only close to such a graph $G_{\mathrm{con}}$, there is some "error" accumulated along the pseudorandom walk. It is important to minimize the error produced in each step (by making $G''$ closer to $G_{\mathrm{con}}$) while keeping the walk short by maintaining the mixing time.

We divide the construction into four stages. The algorithm will actually construct two graphs $G'$ and $G''$, and for the purpose of analysis, we will define two regular graphs $G_{\mathrm{reg}}$ and $G_{\mathrm{con}}$. Before we discuss the construction, we first discuss two properties of $G$

we want to preserve throughout the construction. Let $\varepsilon$ be a small error parameter (which we will later set to be $1/\mathrm{poly}(n, d, k)$.)

1. All four graphs preserve the $s$-$t$ connectivity of $G$: Each vertex $v$ in $G$ will become a cloud of vertices in each of the four graphs. The existence of a path from $s$ to $t$ in $G$ implies the existence of a path from the cloud of $s$ to the cloud of $t$ in the four graphs. Furthermore, every path from the cloud of $s$ to the cloud of $t$ in $G'$ or $G''$ can be projected to a path from $s$ to $t$ in $G$ in logspace.

2. $G'$, $G_{\mathrm{reg}}$, and $G_{\mathrm{con}}$ preserve the mixing time of $G$: We want the spectral gaps $\gamma(G'), \gamma(G_{\mathrm{reg}}), \gamma(G_{\mathrm{con}}) \geq 1/\mathrm{poly}(n, d, k)$ for some fixed polynomial *independent* of $\varepsilon$. In this case, we say $G'$, $G_{\mathrm{reg}}$, and $G_{\mathrm{con}}$ have *short* mixing time.

We describe the goal of each stage below.

**STAGE 1** We improve the regularity of $G$ in this stage. We convert $G$ to a nearly $dD$-regular digraph $G'$ with roughly $N$ vertices in logspace, where $N, D$ are blow up factors depending on $\varepsilon$. We use the input stationary distribution to determine how much to blow up each vertex and edge so that $G'$ is nearly regular. More precisely, nearly regular means that except for an $O(\varepsilon)$ fraction of bad vertices in $G'$, every vertex has in-degree and out-degree $(1 \pm O(\varepsilon))dD$. We emphasize that $G'$ has short mixing time.

**STAGE 2** This stage is a mental experiment for the sake of the analysis and is not used by the algorithm. We define a regular graph $G_{\mathrm{reg}}$ "close" to $G'$ by adding an $O(k\varepsilon)$ fraction of edges to $G'$. Adding only a small number of edges is the key property to show that the behavior of pseudorandom walk generator of [16] on $G''$ and $G_{\mathrm{con}}$ are almost the same in Stage 4 below. Since $G'$ is nearly regular, it is easy to get a regular graph by adding small number of edges. The main challenge is to ensure that $G_{\mathrm{reg}}$ has short mixing time, which we do by using a generalization of the notion of visiting length.

**STAGE 3** Now we have (near-)regularity, we work on consistent labelling. There is a simple graph operation that converts a regular graph $G_{\mathrm{reg}}$ to a consistently labelled graph $G_{\mathrm{con}}$. The operation will preserve both the connectivity and mixing time. Note that the algorithm applies the operation to $G'$ to construct $G''$ instead of applying the operation to $G_{\mathrm{reg}}$, as we do not know how to construct $G_{\mathrm{reg}}$ in logspace.

**STAGE 4** The algorithm now applies pseudorandom walk generator of [16] to $G''$. If the pseudorandom walk generator is applied to $G_{\mathrm{con}}$, then by the property of pseudorandom walk generator and short mixing time of $G_{\mathrm{con}}$, a *short* pseudorandom walk will end inside the cloud of $t$ with non-negligible probability. It can be shown that the behavior of pseudorandom walk on $G''$ and $G_{\mathrm{con}}$ are almost the same. (The error is roughly $\varepsilon$ times the length of walk, which can be made small because the walk is short.) Hence, the pseudorandom walk on $G''$ will end inside the cloud of $t$ with positive probability as well. By enumerating all seeds, the algorithm can find a path from the cloud of $s$ to the cloud of $t$ in $G''$, which can then be projected to a path from $s$ to $t$ in $G$.

As mentioned in the introduction, our algorithm is inspired by and shares a similar structure of the result of [16] showing that a pseudorandom walk generator for (arbitrarily labelled) regular digraphs implies a pseudorandom walk generator for all poly-mixing digraphs. There are several reasons that the proof in [16] is not directly applicable:

1. In [16], $G'$ is only part of the analysis and does not to be explicitly constructed by the algorithm in logspace. The reason is that $G'$ and $G_{\mathrm{reg}}$ can be labelled in such a way that the projection of walks from $G'$ to $G$ can be done without actually knowing $G$. However, this labelling is far from consistent (on $G_{\mathrm{reg}}$), which is ok because the result of [16] *assumes* the existence of a pseudorandom walk generator for arbitrary labellings. Here we only want to use the (known) pseudorandom walk generator for consistently labelled graphs. To get a consistent labelling, we construct $G'$ explicitly and then apply Stage 3.

2. Even with knowledge of stationary probability, it is not clear how to carry out the construction of $G'$ in [16] in logspace. In the first step of the proof of [16], they add some edges to $G$ to make the stationary probability of every vertex non-negligible. However, it is not clear how to compute the new stationary distribution in logspace. Therefore, we skip this step and deal with vertices having exponentially small stationary probability directly in our analysis.

## 4 Proof of the main theorem

We prove the main theorem by solving the path finding problem in this section following the outline in the previous section. Let $G$ be an input graph


0-7695-2780-9/07 $20.00 

satisfying the promise. That is, $G$ is a $n$-vertex, $d$-outregular digraph, and vertices $s$ and $t$ in $G$ are connected. Furthermore, let $\pi(\cdot)$ be an (accurate) input stationary distribution, and $\pi_s(\cdot)$ be the restriction of $\pi$ to the strongly connected component of $\mathrm{supp}(\pi)$ containing $S$. We have $\gamma_{\pi_s}, \pi_s(s), \pi_s(t) \geq 1/k$.

Let $\varepsilon = (ndk)^{-c}$ for a constant $c$ to be determined later. Let $N = \lceil 5n/\varepsilon^2 \rceil$, and $D = \lceil 5N/\varepsilon \rceil$ be blow up factors for vertices and edges. By adding self-loops, we may assume without loss of generality that the out-degree $d$ is divisible by 4, and $G$ is $3/4$-lazy.

### 4.1 Construct nearly regular graph $G'$ from $G$

Roughly speaking, our goal is to improve the regularity of $G$ while preserving the mixing time. From $G$, we will construct in logspace a nearly regular digraph $G'$ with a few additional properties. We want to preserve the connectivity of $G$, and want to be able to project a path in $G'$ to a path in $G$. We want $G'$ to have short mixing time, and by the way we control the mixing time, we also need $G'$ to be $1/2$-lazy.

Let us start with regularity. We can think of a random walk on $G$ from a stationary distribution $\pi$ as a flow with no source or sink. Each vertex $v$ has probability mass $\pi(v)$. Each edge $(v, u)$ carries $\pi(v)/d$ flow from $v$ to $u$. Note that regular graphs are characterized by the stationary distribution $\pi$ being uniform — every vertex has equal mass $1/n$, and each edge carries equal flow $1/(nd)$. Observing this, it is natural to attempt to split each vertex and edge proportional to the mass contained in each vertex and the flow carried by each edge.

For motivation, we begin by describing an *ideal construction* in which we ignore round-off errors. Specifically, we convert $G$ to a $N$-vertex $dD$-regular graph $G'$ as follows. For each vertex $v$ in $G$, we blow it up to a cloud of $\pi(v)N$ vertices $C_v$ so that each new vertex contains exactly $1/N$ probability mass. For each edge $(v, u)$ in $G$, we blow it up by a factor $D$ for each vertex in $C_v$. More precisely, for each edge $(v, u)$ in $G$, and each $\hat{v} \in C_v$, $(v, u)$ induces $D$ outgoing edges from $\hat{v}$ to $C_u$ and we spread these edges uniformly over $C_u$. That is, each $\hat{u} \in C_u$ receives $D/|C_u|$ edges from $\hat{v}$. Note that $(v, u)$ carries $\pi(v)/d$ flow in $G$, and blows up to $\pi(v)N \cdot D$ edges in $G'$, so each induced edge shares $(\pi(v)/d)/(\pi(v)N \cdot D) = 1/(dDN)$ flow from the original edge. Intuitively, since each vertex has equal probability mass (namely, $1/N$) and each edge carries equal flow (namely, $1/(dDN)$), we expect $G'$ to be regular.

However, we cannot actually perform the ideal construction, because $\pi(v)N$ and $D/|C_u|$ may not be integers. Thus we consider a more general construction. Let $a(v)$ be the *vertex blow-up factor* and $b(v)$ be the *edge blow-up factor* of vertex $v$ in $G$. Now, for each vertex $v$, we blow up it into a *cloud* of $a(v)$ vertices $C_v$. For each edge $(v, u)$, and each $\hat{v} \in C_v$, $(v, u)$ induces $b(v)$ outgoing edges from $\hat{v}$ to $C_u$, spread as uniformly as possible. That is, each $\hat{u} \in C_u$ receives either $\lceil b(v)/a(u) \rceil$ or $\lfloor b(v)/a(u) \rfloor$ incoming edges from $\hat{v}$. Phrased in this way, the ideal construction sets $a(v) = \pi(v)N$, and $b(v) = D$ for every $v$.

It is natural to set $a(v) = \lceil \pi(v)N \rceil$; we will discuss how to set $b(v)$ shortly. Note that each edge $(v, u)$ in $G$ carries flow $\pi(v)/d$, and induces $a(v) \cdot b(v)$ edges in $G'$. So each induced edge shares $(\pi(v)/d)/(a(v) \cdot b(v))$ flow from $(v, u)$. It turns out that if every edge in $G'$ shares roughly $1/(dDN)$ flow, then the stationary distribution of $G'$ will be well-behaved, and we can show that $indeg(\hat{v}) \approx outdeg(\hat{v})$ for every $\hat{v} \in G'$. Thus, we set $b(v) = \left\lceil \frac{\pi(v)N}{a(v)} D \right\rceil$. Note that when $\pi(v)$ is not too small, $a(v) = \lceil \pi(v)N \rceil \approx \pi(v)N$, $b(v) \approx D$, and $(\pi(v)/d)/(a(v) \cdot b(v)) \approx 1/(dDN)$, similar to the ideal construction.

It can be shown that setting $a(v) = \lceil \pi(v)N \rceil$, and $b(v) = \left\lceil \frac{\pi(v)N}{\lceil \pi(v)N \rceil} D \right\rceil$ indeed makes $G'$ nearly $dD$-regular in the following sense: Except for at most $\varepsilon N$ bad vertices in $G'$, every vertex $\hat{v}$ satisfies $(1 - O(\varepsilon))dD \leq indeg(\hat{v}), outdeg(\hat{v}) \leq (1 + O(\varepsilon))dD$.

Before we actually prove this claim, we discuss one more technical twist to make $G'$ $1/2$-lazy. Recall that $G$ is $3/4$-lazy, so each vertex $v$ has at least $3d/4$ self-loops. We transfer $d/2$ self-loops of $v$ in $G$ to $b(v) \cdot (d/2)$ self-loops for each $\hat{v}$ in $G'$, and use the aforementioned rule to transfer the remaining $d/2$ edges. This clearly makes $G'$ $1/2$-lazy.

Formally, the algorithm to convert $G$ to $G'$ is as follows.

1. For each vertex $v$ in $G$, we blow up $v$ to a cloud $C_v$ in $G'$ with size $a(v) = \lceil \pi(v)N \rceil$.
2. For each vertex $v$ in $G$, and each $\hat{v} \in C_v$, $d/2$ self-loops of $v$ induce $b(v) \cdot (d/2)$ self-loops on $\hat{v}$, and each remaining outgoing edge $(v, u)$ induces $b(v)$ edges which are spread as uniformly as possible from $\hat{v}$ to $C_u$; that is, every $\hat{u} \in C_u$ gets $\lceil b(v)/a(u) \rceil$ or $\lfloor b(v)/a(u) \rfloor$ corresponding edges from $\hat{v}$.

It is not hard to see that given $G$ and the input stationary distribution $\pi(\cdot)$, $G'$ can be constructed in

logspace: For each vertex $v$, the blow-up factors $a(v)$ and $b(v)$ are easy to compute since the arithmetic only involves numbers of logarithmic bit-length,[4] and it is easy to spread $b(v)$ edges to a cloud of size $a(u)$. Note that $C_s$ and $C_t$ have size at least $N/k$, and $G'$ has at most $N+n$ vertices, so the density of $C_s$ and $C_t$ is at least $1/2k$. The following lemma says that the in-degree and out-degree of any $\hat{v}$ in $G'$ are close.

**Lemma 4.1** *For every $\hat{v} \in C_v$, we have $outdeg(\hat{v}) \leq dD$, and:*

$$dD\left(\frac{\pi(v)N}{\lceil\pi(v)N\rceil}\right) \leq outdeg(\hat{v}) \leq dD\left(\frac{\pi(v)N}{\lceil\pi(v)N\rceil}+\varepsilon\right),$$

$$dD\left(\frac{\pi(v)N}{\lceil\pi(v)N\rceil}-\varepsilon\right) \leq indeg(\hat{v}) \leq dD\left(\frac{\pi(v)N}{\lceil\pi(v)N\rceil}+\varepsilon\right).$$

**Proof.** The proof can be found in the full version of the paper [4]. ■

The lemma implies that when $\frac{\pi(v)N}{\lceil\pi(v)N\rceil}$ is close to 1, $\hat{v}$ is nearly $dD$-regular. This leads to the definition of good/bad vertex.

**Definition 4.2** *A vertex $v$ in $G$ is* good *if $\pi(v) > 1/(\varepsilon N) = \Theta(\varepsilon/n)$ $v$ is* bad *otherwise.*

The following two simple lemmas show that good vertices are nearly $dD$-regular, and the number of bad vertices is small.

**Lemma 4.3** *For every good vertex $\hat{v}$ in $G'$, we have*

$$\begin{aligned} outdeg(\hat{v}) &\in [(1-\varepsilon)dD, dD] \\ indeg(\hat{v}) &\in [(1-2\varepsilon)dD, (1+\varepsilon)dD] \end{aligned}$$

**Proof.** The proof can be found in the full version of the paper [4]. ■

**Lemma 4.4** *The number of bad vertices in $G'$ is at most $\varepsilon N$.*

**Proof.** The proof can be found in the full version of the paper [4]. ■

Let us study the relation between $G$ and $G'$. Note that $G'$ is obtained by blowing up each vertex and edge of $G$ by a certain factor, on the cloud level, $G'$ has the same structure as $G$. That is, for every two vertices $u, v$ of $G$ and every $\hat{u} \in C_u$, the fraction of edges leaving $\hat{u}$ that enter $C_v$ equals the fraction of edges leaving $u$ that lead to $v$. Therefore, we can project a path from $\hat{u} \in C_u$ to $\hat{v} \in C_v$ in $G'$ to a path from $u$ to $v$ in $G$, and project a stationary distribution of $G'$ to a stationary distribution of $G$. Furthermore, a random walk on $G'$ is projected to a random walk on $G$.

Conversely, a path (resp., a stationary distribution) on $G$ can be lifted to a path (resp., a stationary distribution) on $G'$. By lifting we mean the projection of a lifted object is again the original object. It is easy to see that for any path from $u$ to $v$ in $G$, there exists lifted paths from some $\hat{u} \in C_u$ to some $\hat{v} \in C_v$. Given a stationary distribution $\pi$ of $G$, we can obtain a lifted stationary distribution of $G'$ by the stationary distribution of a random walk starting from certain distribution. Let $\alpha$ be a distribution on $G'$ defined as follows. For each vertex $v \in G$, set $\alpha(\hat{v}) = \pi(v)$ for some $\hat{v} \in C_v$ and $\alpha(\hat{v}') = 0$ for all other $\hat{v}' \in C_v$. Since $G'$ is $1/2$-lazy, the random walk started at $\alpha$ converges to a stationary distribution $\pi'$. Note that the projection of this random walk is a random walk on $G$ starting from stationary distribution $\pi$. Hence, the projected distribution is always $\pi$, and in particular, the projection of $\pi'$ is $\pi$.

Let $\pi'$ and $\pi'_s$ be lifted stationary distributions of the input stationary distribution $\pi$ and $\pi_s$ respectively. For the above discussion, we know that $G'$ preserves the connectivity of $C_s$ and $C_t$, and on the cloud level, preserves the visiting length, which leads to the following definition.

**Definition 4.5** *Let $M$ be a Markov chain, and $S$ be a set of vertices in $M$. The* generalized visiting length *of $S$, denoted $\ell_S(M)$, is the smallest number $\ell$ such that from every vertex $v$ reachable from $S$, a random walk of length $\ell$ from $v$ visits $S$ with probability at least $1/2$.*

The above discussion implies that the generalized visiting length $\ell_{C_s}(G')$ in $G'$ is equal to the visiting length $\ell_s(G)$ in $G$. Note that by Lemma 2.7, $\ell_s(G) = O(nk^3 \log d) = \mathrm{poly}(n,d,k)$ is *short* in the sense that it is independent of $\varepsilon$. Hence, on the cloud level, the mixing time of $G'$ should be short as well. Moreover, the cliques induced by at least $d/4$ self-loops on each vertex should imply quick mixing inside each cloud. We may thus expect $G'$ to have short mixing time. The following lemma formalize this intuition.

**Lemma 4.6** *Let $M$ be a $1/2$-lazy Markov chain. Suppose there is a vertex set $S$ with generalized visiting length $\ell$ satisfies $M(s_1, s_2) \geq 1/(8|S|)$ for every $s_1, s_2 \in S$. Then $M$ has a stationary distribution $\pi$ (with $S \subset \mathrm{supp}(\pi)$) such that the conductance satisfies $h_\pi(M) \geq 1/(32\ell)$, and hence the spectral gap satisfies $\gamma_\pi(M) \geq 1/(2^{11} \cdot \ell^2)$.*

[4] Recall that $1/\mathrm{poly}(n,d,k)$-approximation of $\pi(\cdot)$ is suffices (which we will argue in the end), so $\pi(\cdot)$ can be expressed in logarithmic bits.

**Proof.** The proof can be found in the full version of the paper [4]. ∎

Let $M'$ be the transition matrix for the random walk on $G'$. Note that for all $\hat{s}_1, \hat{s}_2 \in C_s$, $M'(\hat{s}_1, \hat{s}_2) \geq 1/(6|C_s|) > 1/(8|C_s|)$ is satisfied by the cliques on $C_s$ induced by $d/4$ self-loops of $s \in G$. A straightforward application of Lemma 4.6 shows that $\gamma_{\pi'_s}(G') \geq 1/(2^{11} \cdot (\ell_{C_s}(G'))^2) = 1/\mathrm{poly}(n, d, k)$, which means $G'$ has short mixing time, as desired.

To summarize, in this stage we construct $G'$ from $G$ in logspace with the following properties.

1. $G'$ is nearly $dD$-regular: There are at most $\varepsilon N$ bad vertices in $G'$, and for all good vertices $\hat{v} \in G'$, $(1-O(\varepsilon))dD \leq \mathit{indeg}(\hat{v}), \mathit{outdeg}(\hat{v}) \leq (1+O(\varepsilon))dD$.

2. $G'$ preserves the connectivity of $C_s$ and $C_t$, and every path from $C_s$ to $C_t$ in $G'$ can be projected to a path from $s$ to $t$ in $G$ in logspace.

3. $G'$ has short mixing time: The generalized visiting length of $C_s$ in $G'$ is $O(nk^3 \log d) = \mathrm{poly}(n, d, k)$, $M'(\hat{s}_1, \hat{s}_2) \geq 1/(6|C_s|)$ for all $\hat{s}_1, \hat{s}_2 \in C_s$, and the spectral gap of $G'$ satisfies $\gamma_{\pi'_s}(G') \geq 1/(2^{11} \cdot (\ell')^2)$.

4. The sizes of $C_s$ and $C_t$ are at least $N/k$, and their density are at least $1/(2k)$.

### 4.2 $G'$ is close to a regular graph $G_{\mathrm{reg}}$

We emphasize that this stage is a mental experiment for the sake of the analysis and so the algorithm does *nothing* in this stage. We show in analysis that by adding at most $O(k\varepsilon dDN)$ edges, we can get a regular graph $G_{\mathrm{reg}}$ with short mixing time.

Since the out-degree and in-degree of every vertex $\hat{v}$ in $G'$ are bounded by $(1+O(\varepsilon))dD$, we can make $G'$ $(1+O(\varepsilon))dD$-regular by adding $(1+O(\varepsilon))dD - \mathit{outdeg}(\hat{v})$ outgoing edges from each vertex $\hat{v}$. Doing so adds at most $O(\varepsilon dDN)$ edges because $(1 + O(\varepsilon))dD - \mathit{outdeg}(\hat{v}) = O(\varepsilon dD)$ for each good vertex $\hat{v}$, and there are at most $\varepsilon N$ bad vertices. However, note that the stationary distribution of a regular graph is uniform, while there might be some (bad) vertices $\hat{v}$ in $G'$ with exponentially small stationary probability. Thus, the stationary distribution might change dramatically under this operation, making it difficult to directly relate the spectral gap of $G'$ and $G_{\mathrm{reg}}$. Instead, we control the mixing time by maintaining the generalized visiting length and clique structure of $C_s$, as well as the laziness of the entire graph.

To maintain the generalized visiting length of $C_s$, we do not allow the set of vertices reachable from $C_s$ to increase when we add edges. Let $V_1$ be the set of vertices reachable from $s$ in $G$, $V_2$ the set of vertices that can reach $s$ but are not reachable from $s$, and $V_3 = V - V_1 - V_2$. Note that for every stationary distribution $\pi$ of $G$, $\pi(V_2) = 0$. Hence, for every $v \in V_2$, its vertex blow-up factor $a(v)$ is 0, which means $v$ is "deleted" in $G'$.[5] Let $V'_1$ and $V'_3$ be the set of vertices in $G'$ corresponding to $V_1$ and $V_3$, respectively. We have $V' = V'_1 \cup V'_3$, and $V'_1$ and $V'_3$ are disconnected, so we can make $V'_1$ regular by adding edges from $V'_1$ to $V'_1$ itself, which will not increase the set of vertices reachable from $C_s$. We use the following procedure to make $V'_1$ regular.

1. Make out-degree of each vertex at least $(1-\varepsilon)dD$. For each bad vertex $\hat{v} \in V'_1$ with $\mathit{outdeg}(\hat{v}) < (1-\varepsilon)dD$, do the following:
   (a) Add $\lfloor((1-\varepsilon)dD - \mathit{outdeg}(\hat{v}))/2\rfloor$ self-loops to $\hat{v}$.
   (b) Add $\lceil((1-\varepsilon)dD - \mathit{outdeg}(\hat{v}))/2\rceil$ outgoing edges from $\hat{v}$ to $V'_1$, spread as uniformly as possible. That is, every $\hat{v}'$ gets $t$ or $t+1$ edges from $\hat{v}$ for some integer $t$. We always let $\hat{s} \in C_s$ get $t+1$ edges when it is possible.

2. Make the graph regular: Arbitrarily add edges in $V'_1$ to make the graph $(1+2k\varepsilon)dD$-regular.

3. Make the graph $1/2$-lazy: Add $4k\varepsilon dD$ self-loops to each $\hat{v} \in V'_1$.

We need to argue that this procedure is always possible, but first provide some intuition for the construction. If we add only small number of outgoing edges to every vertex, then a short random walk will use original edges with high probability. In this case, the generalized visiting length should not change too much. However, a bad vertex $\hat{v}$ may have small out-degree. We need to add many outgoing edges to $\hat{v}$ to make the graph regular. To make sure that $\hat{v}$ does not increase the in-degree of other vertices too much, and that a random walk can visit $C_s$ from $\hat{v}$ easily, it is natural to spread the extra outgoing edges of $\hat{v}$ as uniformly as possible. In particular, a one-step random walk from $\hat{v}$ using new outgoing edges of $\hat{v}$ will visit $C_s$ with probability at least $|C_s|/|V'_1| = \Omega(1/k)$. Therefore, we can expect the generalized visiting length of $C_s$ remains short.

[5] This is no longer true when the input stationary distribution is not accurate; we will deal with this issue in Section 4.5.

We now argue that the above procedure is always possible. That is, after the first step, the in-degree and out-degree of each vertex is less than $(1 + 2k\varepsilon)dD$. The out-degree part is trivial. For the in-degree part, let us first upper bound the number of incoming edges added in step 1b. The number of bad vertices is at most $\varepsilon N$. Since $C_s \subset V_1'$, $|V_1'| \geq N/k$. Each bad vertex adds at most $\lceil (dD/2)/|V_1'| \rceil \leq dD/(N/k) = kdD/N$ to the in-degree of each $\hat{v} \in V_1'$. Hence the in-degree is increased by at most $(\varepsilon N) \cdot (kdD/N) = k\varepsilon dD$ in step 1b. For a good vertex $\hat{v}$, $\hat{v}$ does not get any self-loops in step 1a. By Lemma 4.3, the in-degree of $\hat{v}$ is at most $(1 + \varepsilon)dD + k\varepsilon dD \leq (1 + 2k\varepsilon)dD$ after step 1b. For a bad vertex $\hat{v}$, by Lemma 4.1, $indeg(\hat{v}) \leq outdeg(\hat{v}) + \varepsilon dD$, so after step 1a, $indeg(\hat{v}) \leq (1+\varepsilon)dD$, and after step 1b, $indeg(\hat{v}) \leq (1 + 2k\varepsilon)dD$. We also note that the resulting graph is $1/2$-lazy because $G'$ is $1/2$-lazy and for each vertex, at least half of the outgoing edges added by the above procedure are self-loops.

We now deal with $V_3'$. Since $V_3'$ does not affect the generalized visiting length of $C_s$, we can make it regular in the naive way described at beginning. To summarize, we construct a $1/2$-lazy, $(1+6k\varepsilon)dD$-regular digraph $G_{\rm reg} = (V_{\rm reg}, E_{\rm reg})$ from $G'$ as follows.

- $V_{\rm reg} = V_1' \cup V_3'$.
- We make $V_1'$ $1/2$-lazy and regular by the above procedure.
- We make $V_3'$ $1/2$-lazy and regular by first adding $(1 + 5k\varepsilon)dD - outdeg(\hat{v})$ self-loops to each $\hat{v} \in V_3'$, and then make $V_3'$ $(1 + 6k\varepsilon)dD$-regular arbitrarily.[6]

The construction preserves the connectivity of $C_s$ and $C_t$ because the set of vertices reachable from $C_s$ remains the same. We next bound the generalized visiting length of $C_s$. To simplify the presentation, we introduce the following notation. Let $\tilde{G}' = (V_1', E_1')$ be the subgraph of $G'$ induced by $V_1'$, and $\tilde{G}_{\rm reg}$ be the (strongly) connected component $V_1'$ in $G_{\rm reg}$. Let $B_1$ be the edge set added to $V_1'$ in step 1 above, and $B_2$ be the edge set added to $V_1'$ in steps 2 and 3. Hence, $\tilde{G}_{\rm reg} = (V_1', E_1' \cup B_1 \cup B_2)$. Let $\ell' = \ell_{C_s}(\tilde{G}') = \ell_{C_s}(G')$, and $\ell_{\rm reg} = \ell_{C_s}(\tilde{G}_{\rm reg}) = \ell_{C_s}(G_{\rm reg})$ be the generalized visiting length of $C_s$ in $G'$ and $G_{\rm reg}$, respectively. Then we have:

**Lemma 4.7** $\ell_{\rm reg} = O(\ell' k)$.

[6]The reason that we can not apply the algorithm for $V_1'$ to $V_3'$ is that $|V_3'|$ may be too small. In this case, after step 1, the in-degree will be too large.

**Proof.** Let $\ell = 3a\ell' k$ for some constant $a$ to be determined later. Let $\hat{v}$ be any vertex reachable from $C_s$. Let $w$ denote an $\ell$-step random walk in $\tilde{G}_{\rm reg}$ from $\hat{v}$. We need to show $\Pr_w[w \text{ visits } C_s] \geq 1/2$. Consider the following three events:

- $\mathcal{E}_1$: $w$ uses $\geq ak$ edges in $B_1 \wedge$ never visits $C_s$.
- $\mathcal{E}_2$: $w$ uses $\geq$ one edge in $B_2 \wedge$ never visits $C_s$.
- $\mathcal{E}_3$: $w$ uses less than $ak$ edges in $B_1$, no edge in $B_2$, and never visits $C_s$.

Clearly, $\Pr_w[w \text{ does not visit } S] \leq \Pr_w[\mathcal{E}_1] + \Pr_w[\mathcal{E}_2] + \Pr_w[\mathcal{E}_3]$. Let us upper bound the three events.

Since each vertex $\hat{u}$ in $\tilde{G}_{\rm reg}$ has at most an $O(k\varepsilon)$ fraction of outgoing edges from $B_2$, for each step the probability that $w$ uses a $B_2$ edge is at most $O(k\varepsilon)$. By a union bound,

$$\Pr_w[\mathcal{E}_2] \leq O(k\varepsilon) \cdot \ell = O(a\varepsilon\ell' k^2).$$

To bound $\Pr_w[\mathcal{E}_1]$, we consider the following experiment: Let $w'$ be a random walk starting from the same vertex $\hat{v}$ that continues until it uses $ak$ edges in $B_1$. Each time that $w'$ uses a $B_1$ edge, $w'$ visits $C_s$ with probability at least $1/(4k)$ (at least $1/2$ chance to use an edge added in step 1b, and the density of $C_s$ is at least $1/(2k)$). It follows that

$$\Pr_{w'}[w' \text{ never visits } C_s] \leq \left(1 - \frac{1}{4k}\right)^{ak} \leq e^{-a/4}.$$

To justify this bound, we can think of the random walk in the following way. At each step $w'$ first tosses a biased coin to decide to use edge in $B_1$ or in $E \cup B_2$, and then chooses an edge from the chosen set uniformly. Note that the biased coin depends on the fraction of $B_1$ edges leaving the current vertex. Each time that $w'$ decides to choose a $B_1$ edge, $w'$ will hit $C_s$ with probability at least $1/(4k)$.

Now we can analyze the original walk $w$ as follows:

$$\begin{aligned}\Pr_w[\mathcal{E}_1] &= \Pr_{w'}[w' \text{ never visits } C_s \text{ and } |w'| \leq \ell] \\ &\leq \Pr_{w'}[w' \text{ never visits } C_s] \leq e^{-a/2}\end{aligned}$$

To bound $\Pr_w[\mathcal{E}_3]$, we start with the following observation. Let $r$ be an $\ell'$-step random walk from *any* vertex $\hat{v}$ on $\tilde{G}_{\rm reg}$ and $r'$ be an $\ell'$-step random walk from the same $\hat{v}$ on $G'$. We have

$$\begin{aligned}&\Pr_r[r \text{ only uses edges in } E_1' \text{ and never visits } C_s] \\ \leq\ &\Pr_{r'}[r' \text{ never visits } C_s] \leq \frac{1}{2}\end{aligned}$$

since each walk contained in the LHS event is also contained in the RHS event, and the walk has greater probability mass in the RHS than in the LHS.

We now think of $w$ as $3ak$ consecutive $\ell'$-step random walks in $\tilde{G}_{\mathrm{reg}}$. We call each $\ell'$-step random walk a *segment*. We say that a segment is *bad* if the walk in that segment only uses edges in $E_1'$ and never visits $C_s$. From the above observation, a segment is bad with probability at most $1/2$, even conditioned on the previous segments of the walk. By a Chernoff bound, we have $\Pr_w[\#\text{ of bad segments} \geq 2ak] \leq 2^{-\Omega(ak)}$.

However, note that any walk in $\mathcal{E}_3$ contains at least $2ak$ bad segments, so

$$\Pr_w[\mathcal{E}_3] \leq \Pr_w[\#\text{ of bad segments} \geq 2ak] \leq 2^{-\Omega(ak)}.$$

In sum, we have $\Pr_w[w \text{ does not visit } S] \leq \Pr_w[\mathcal{E}_1]+\Pr_w[\mathcal{E}_2]+\Pr_w[\mathcal{E}_3] \leq O(a\varepsilon\ell' k^2)+e^{-a/4}+2^{-\Omega(ak)} < 1/2$ for $\varepsilon = (ndk)^{-c}$ and a sufficiently large choice of the constants $c$ and $a$.

■

Let $M_{\mathrm{reg}}$ be the transition matrix for the random walk on $G_{\mathrm{reg}}$, and let $\pi_{\mathrm{reg}}$ be the (uniform) stationary distribution on the (strongly) connected component $V_1'$ of $G_{\mathrm{reg}}$. As $G_{\mathrm{reg}}$ is $1/2$-lazy, to apply Lemma 4.6, it remains to check $M_{\mathrm{reg}}(\hat{s}_1, \hat{s}_2) \geq 1/(8|C_s|)$ for all $\hat{s}_1, \hat{s}_2 \in C_s$. Recall that $M'(\hat{s}_1, \hat{s}_2) \geq 1/(6|C_s|)$. Note that we do not remove any edges in $V_1'$ in the construction, and the out-degree increases by only $1+O(k\varepsilon)$ factor. So $M_{\mathrm{reg}}(\hat{s}_1, \hat{s}_2) \geq 1/((1+O(k\varepsilon))\cdot 6|C_s|) \geq 1/(8|C_s|)$. Therefore, by Lemma 4.6, $\gamma_{\pi_{\mathrm{reg}}}(G_{\mathrm{reg}}) \geq 1/(2^{11}\cdot(\ell_{\mathrm{reg}})^2) = 1/\mathrm{poly}(n,d,k)$.

Let $N_{\mathrm{reg}}$ be the number of vertices in $G_{\mathrm{reg}}$, and $D_{\mathrm{reg}} = (1+6k\varepsilon)dD$ be the degree of $G_{\mathrm{reg}}$. We summarize the properties of $G_{\mathrm{reg}}$ as follows.

1. $G_{\mathrm{reg}}$ has at most $O(k\varepsilon D_{\mathrm{reg}} N_{\mathrm{reg}})$ additional edges to $G'$.
2. $G_{\mathrm{reg}}$ preserves the connectivity of $C_s$ and $C_t$.
3. $G_{\mathrm{reg}}$ has short mixing time. The spectral gap of $G_{\mathrm{reg}}$ is $\gamma_{\pi_{\mathrm{reg}}}(G_{\mathrm{reg}}) = 1/\mathrm{poly}(n,d,k)$.
4. $|C_s|, |C_t| \geq N/k \geq N_{\mathrm{reg}}/(2k)$, and $\pi_{\mathrm{reg}}(C_s), \pi_{\mathrm{reg}}(C_t) \geq 1/(2k)$.

### 4.3 Obtain a consistent labelling

The goal of this stage is to obtain a "consistent labelling" of edges so that we can apply the pseudorandom walk generator of [16] in next stage. We achieve it by applying a simple *lift* operation to the graph.[7] Given a labelled graph $H$, the operation will output a labelled graph $L(H)$ preserving the connectivity of $H$. When $H$ is $d$-regular, the lifted graph $L(H)$ is simply the ("tensor" or "AND") product with a $d$-clique (with self-loops), so $L(H)$ has the same spectral gap as $H$. Furthermore, we can consistently label $L(H)$. Hence, if we applied the operation to $G_{\mathrm{reg}}$, the resulting graph $G_{\mathrm{con}} = L(G_{\mathrm{reg}})$ would be a consistently labelled regular graph with short mixing time.

However, since we do not know how to compute $G_{\mathrm{reg}}$ in logspace, our algorithm will compute $G'' = L(G')$ instead. $G''$ is not regular and we do not know its mixing time. Fortunately, we are able to argue that the behavior of *short* (pseudo)random walks on $G''$ and $G_{\mathrm{con}}$ are very "similar", so that we can apply the pseudorandom walk generator to $G''$ instead of $G_{\mathrm{con}}$.

We start with a discussion about labellings.

**Labelling.** Let $H$ be a digraph with $n$ vertices such that every vertex has out-degree at most $d_{\mathrm{out}}$ and in-degree at most $d_{\mathrm{in}}$. A *two-way labelling* of $H$ gives each edge $(v,w) \in H$ an outgoing label of $v$ in $[d_{\mathrm{out}}]$, and an incoming label of $w$ in $[d_{\mathrm{in}}]$ such that the outgoing (resp., incoming) labels of each vertex $v \in H$ are all distinct. Such a graph together with its two-way labelling can be specified by a *rotation map* $\mathrm{Rot}_H : [n] \times [d_{\mathrm{out}}] \to ([n] \times [d_{\mathrm{in}}]) \cup \{\perp\}$, where $\mathrm{Rot}_H(v,i) = (w,j)$ if there is an edge numbered $i$ leaving $v$ and it equals the edge numbered $j$ entering $w$, and $\mathrm{Rot}_H(v,i) = \perp$ if there is no edge numbered $i$ leaving $v$. We say a rotation map $\mathrm{Rot}_H$ has *degree* $d$ if $d_{\mathrm{in}} = d_{\mathrm{out}} = d$.

Note that we can compute a degree-$D_{\mathrm{reg}}$ rotation map $\mathrm{Rot}_{G'}$ of $G'$ in logspace. Furthermore, $\mathrm{Rot}_{G'}$ can be extended to a rotation map $\mathrm{Rot}_{G_{\mathrm{reg}}}$ of $G_{\mathrm{reg}}$ in such a way that for every edge present in both $G'$ and $G_{\mathrm{reg}}$, it has the same outgoing and incoming labels in $G'$ and $G_{\mathrm{reg}}$. In the following discussion, we assume $G'$ and $G_{\mathrm{reg}}$ are associated with rotation maps $\mathrm{Rot}_{G'}$ and $\mathrm{Rot}_{G_{\mathrm{reg}}}$ that are *compatible* in this sense.

A *consistent labelling* of a $d$-regular graph $H$ gives each edge only *one* label in $[d]$ such that for each vertex $v \in H$, all of the edges leaving (resp., entering) $v$ have distinct labels. Hence, in terms of rotation maps, $\mathrm{Rot}_H$ defines a consistent labelling iff for all $v, i$, $\mathrm{Rot}_H(v,i) = (w,i)$ for some $w$.

We define the lift operation in terms of rotation maps as follows.

**Definition 4.8** *Let $H$ is a two-way labelled graph on $n$ vertices with rotation map $\mathrm{Rot}_H : [n]\times[d] \to [n]\times[d]$. The* lift *$L(H)$ is a graph on $[n] \times [d]$ vertices whose rotation map $\mathrm{Rot}_{L(H)} : ([n] \times [d]) \times [d^2] \to ([n] \times [d]) \times [d^2]$ is as follows:*

[7] The lift operation defined here is not the same as the "lifts" of [2]

$\mathrm{Rot}_{L(H)}((v,k),(i,j))$**:**

1. *If* $\mathrm{Rot}_H(v,k+i) = (w,l)$*, output* $((w,l+j \bmod d),(i,j))$
2. *If* $\mathrm{Rot}_H(v,k+i) = \perp$*, output* $\perp$*.*

*where all arithmetic on elements of* $[d]$ *is taken modulo* $d$*.*

It is clear that $\mathrm{Rot}_{L(H)}$ can be computed in logspace if $\mathrm{Rot}_H$ can. We can think of the operation as follows. The operation lifts each vertex in $H$ to a *cloud* in $L(H)$. A vertex $(v,k)$ in $L(H)$ is the $k$-th vertex in the cloud of $v$. Staying at vertex $(v,k)$ in $L(H)$ can be interpreted as staying at vertex $v$ in $H$ and facing toward the $k$-th edge. Phrased in this way, the $(i,j)$-th neighbor of vertex $(v,k)$ is obtained by the following steps. Starting at vertex $v$ and facing toward $k$-th edge in $H$, we (i) turn to $(k+i)$-th edge, (ii) go across the $(k+i)$-th edge to vertex $w$ and face to the $l$-th edge, and then (iii) turn to the $(l+j)$-th edge of $w$. We call step (ii) the *$H$-step* of $\mathrm{Rot}_{L(H)}((v,k),(i,j))$. Note that a $H$-step is simply following an edge of $H$.

It is easy to check that $\mathrm{Rot}_{L(H)}$ is a legal two-way labelling. All incoming edges of vertex $(w,l)$ have distinct incoming labels because the above procedure is invertible. Indeed, if an incoming edge of $(w,l)$ is labelled $(i,j)$, then in the second step, it must come from an edge with incoming label $l-j$ at vertex $w$ in $H$. Since $H$ is two-way labelled, there is at most one such edge, say, from the $k$-th outgoing edge of vertex $v$. Thus, the only edge incident to $(w,l)$ with incoming label $(i,j)$ is $(v,k-i)$.

**Lemma 4.9** *If $H$ is $d$-regular, then*

1. *$L(H)$ is consistently labelled.*
2. *$\gamma(L(H)) = \gamma(H)$.*

**Proof.** The proof can be found in the full version of the paper [4]. ∎

By Lemma 4.9, $G_{\mathrm{con}} = L(G_{\mathrm{reg}})$ is a consistently labelled $(N_{\mathrm{reg}} \cdot D_{\mathrm{reg}})$-vertex, $D_{\mathrm{reg}}^2$-regular graph with the same spectral gap as $G_{\mathrm{reg}}$ (with respect to the (uniform) stationary distribution $\pi_{\mathrm{con}}$ on the union of clouds corresponding to vertices in $V_1'$.) Let $\tilde{C}_s = C_s \times [D_{\mathrm{reg}}]$ and $\tilde{C}_t = C_t \times [D_{\mathrm{reg}}]$ be the cloud of $s$ and $t$ in $G''$. We have $\pi_{\mathrm{con}}(\tilde{C}_s) = \pi_{\mathrm{reg}}(C_s)$ and $\pi_{\mathrm{con}}(\tilde{C}_t) = \pi_{\mathrm{reg}}(C_t)$. Let us now consider $G'' = L(G')$. It is easy to check from the definition that $G''$ preserves the connectivity of $\tilde{C}_s$ and $\tilde{C}_t$, and every path from $\tilde{C}_s$ to $\tilde{C}_t$ in $G''$ can be projected to a path from $s$ to $t$ in $G$.

We now study the difference between $G_{\mathrm{con}}$ and $G''$. From the fact that the rotation maps of $G_{\mathrm{reg}}$ and $G'$ are compatible, it follows that the rotation maps of their lifts $G_{\mathrm{con}} = L(G_{\mathrm{reg}})$ and $G'' = L(G)$ are also compatible. Moreover, $\mathrm{Rot}_{G''}((v,k),(i,j)) = \perp$ iff $\mathrm{Rot}_{G'}(v,k+i) = \perp$. That is, an edge of $G_{\mathrm{con}}$ is missing in $G''$ iff the corresponding $G_{\mathrm{reg}}$-step is missing in $G'$. Let $B$ be the set of pairs $(v,k) \in [N_{\mathrm{reg}}] \times [D_{\mathrm{reg}}]$ such that $\mathrm{Rot}_{G'}(v,k) = \perp$. Note that the size of $B$ is only $O(k\varepsilon D_{\mathrm{reg}} N_{\mathrm{reg}})$.

To summarize, in this stage the algorithm first computes a rotation map $\mathrm{Rot}_{G'}$ of $G'$, and then computes $G'' = L(G')$ in logspace. Let $N_{\mathrm{con}} = N_{\mathrm{reg}} \cdot D_{\mathrm{reg}}$ be the number of vertices in $G_{\mathrm{con}}$, and $D_{\mathrm{con}} = D_{\mathrm{reg}}^2$ be the degree of $G_{\mathrm{con}}$. We have the following properties.

1. $G_{\mathrm{con}}$ has short mixing time. The spectral gap of $G_{\mathrm{con}}$ is $\gamma_{\pi_{\mathrm{con}}}(G_{\mathrm{con}}) = \gamma_{\pi_{\mathrm{reg}}}(G_{\mathrm{reg}}) = 1/\mathrm{poly}(n,d,k)$.
2. In both $G''$ and $G_{\mathrm{con}}$, the clouds $\tilde{C}_s$ and $\tilde{C}_t$ are connected, and every path from $\tilde{C}_s$ to $\tilde{C}_t$ in $G''$ can be projected to a path from $s$ to $t$ in $G$ in logspace.
3. An edge of $G_{\mathrm{con}}$ is also an edge of $G''$ iff the $G_{\mathrm{reg}}$-step of the edge is not in $B$. The size of $B$ is only $O(k\varepsilon D_{\mathrm{reg}} N_{\mathrm{reg}})$.
4. $|\tilde{C}_s|, |\tilde{C}_t| \geq N_{\mathrm{con}}/(2k)$, and $\pi_{\mathrm{con}}(\tilde{C}_s), \pi_{\mathrm{con}}(\tilde{C}_t) \geq 1/(2k)$.

### 4.4 Find a path using the pseudorandom walk generator of [16]

We are ready to apply the pseudorandom walk generator of [16] to $G''$ to find a path from $s$ to $t$ in $G$.

**Lemma 4.10 ([16])** *For every $N, D \in \mathbb{N}$, $\delta,\gamma > 0$, there is a generator $\mathrm{PRG} = \mathrm{PRG}_{N,D,\delta,\gamma} : \{0,1\}^r \to [D]^\ell$ with seed length $r = O(\log(ND/\delta\gamma))$, and walk length $\ell = \mathrm{poly}(1/\gamma) \cdot \log(ND/\delta)$, computable in space $O(\log(ND/\delta\gamma))$ such that for every (connected)* consistently labelled *$N$-vertices $D$-regular digraph $G$ with spectral gap $\gamma$ and every vertex $s$ in $G$, taking walk $\mathrm{PRG}(U_r)$ from $s$ ends at a distribution $\delta$-close to uniform (in variation distance).*

Let us first apply the above PRG to $G_{\mathrm{con}}$. Set $\delta = 1/(4k)$. Note that $N_{\mathrm{con}}, D_{\mathrm{con}} = \mathrm{poly}(n,d,k,1/\varepsilon)$, and $\gamma = \gamma_{\pi_{\mathrm{con}}}(G_{\mathrm{con}}) = \mathrm{poly}(n,d,k)$. Hence, PRG is computable in logspace, the seed length $r$ is logarithmic, and the walk length $\ell = \mathrm{poly}(1/\gamma) \cdot$

$\log(ND/\delta) \le (ndk)^a$ for some constant $a$ (*independent* of the constant $c$ in $\varepsilon = (ndk)^{-c}$.) Let us apply $\mathrm{PRG}(U_r)$ to $G_{\mathrm{con}}$ with initial distribution uniform over $\tilde{C}_s$. Therefore, the probability that the walk ends in $\tilde{C}_t$ is at least $1/(2k) - \delta = 1/(4k)$. In particular, this implies there is a vertex $\tilde{s} \in \tilde{C}_s$, a vertex $\tilde{t} \in \tilde{C}_t$, and a seed $x \in \{0,1\}^r$ such that $\mathrm{PRG}(x)$ is a path from $\tilde{s}$ to $\tilde{t}$ in $G_{\mathrm{con}}$.

We next show that the PRG can produce a path from $\tilde{C}_s$ to $\tilde{C}_t$ such that all edges in the path are also edges in $G''$. It implies that when we apply PRG to $G''$, we can find a path from $\tilde{C}_s$ to $\tilde{C}_t$, which can be projected to a path from $s$ to $t$ in $G$, as desired.

Let $p_0$ be the uniform distribution on $\tilde{C}_s$. Let $(e_1, \ldots, e_\ell) \in [D_{\mathrm{con}}]^\ell$ be any fixed sequence of edge labels specifying a walk. Starting from $p_0$, let $p_1, \ldots, p_\ell$ be the distribution after each step. Since $G_{\mathrm{con}}$ is consistently labelled, each $p_i$ is a uniform distribution on some set of size $|\tilde{C}_s|$. Now, consider one step that goes from $p_i$ to $p_{i+1}$ using edge label $e_i$. Note that for any two distinct $\tilde{v}_1, \tilde{v}_2 \in G_{\mathrm{con}}$, the $G_{\mathrm{reg}}$-steps of $\mathrm{Rot}_{G_{\mathrm{con}}}(\tilde{v}_1, e_i)$ and $\mathrm{Rot}_{G_{\mathrm{con}}}(\tilde{v}_2, e_i)$ are also distinct, so the probability that the $G_{\mathrm{reg}}$-step of $i$-th step is in $B$ is at most $|B|/|\tilde{C}_s| \le O(k\varepsilon N_{\mathrm{con}})/(N_{\mathrm{con}}/(2k)) = O(k^2\varepsilon)$. By a union bound, the probability that an $\ell$-step walk ever uses an edge whose $G_{\mathrm{reg}}$-step is in $B$ is at most

$$\ell \cdot O(k^2\varepsilon) \le (ndk)^a \cdot O(k^2 \cdot (ndk)^{-c}) \ll \frac{1}{4k}$$

where we set $c = a + 3$. Note that if a walk only uses edges whose $G_{\mathrm{reg}}$-step is not in $B$, then all edges are actually in $G''$ and thus the walk is also a walk in $G''$. Since the above inequality holds for every fixed walk $(e_1, \ldots, e_\ell)$, it holds for a pseudorandom walk starting from $p_0$. Hence,

$$\Pr[\mathrm{PRG}(U_r) \text{ ends in } \tilde{C}_t \text{ and uses only edges in } G''] \ge \frac{1}{4k} - \ell \cdot O(k^2\varepsilon) > 0.$$

We summarize the algorithm and show how it uses PRG to solve KNOWN-STATIONARY FIND PATH in deterministic logspace.

1. Compute $G'$ from $G$ as described in stage 1.
2. Compute a two-way labelling of $G'$ and $G'' = L(G')$ as described in stage 3.
3. For each vertex $\tilde{s} \in \tilde{C}_s$ and seed $x \in \{0,1\}^r$, compute the walk $\mathrm{PRG}(x)$ starting from $\tilde{s}$ in $G''$, and project the path to $G$ (which might fail due to there being no edge with a particular label.
4. If we find a path from $s$ to $t$, then output the path.

The algorithm runs in deterministic logspace because every step does. Since the probability that a pseudorandom walk in $G_{\mathrm{con}}$ starting from uniform distribution on $\tilde{C}_s$ ends in $\tilde{C}_t$ and uses only edges in $G''$ is positive, there exists some $\tilde{s} \in \tilde{C}_s$ and $x \in \{0,1\}^r$ such that the walk $\mathrm{PRG}(x)$ starting from $\tilde{s}$ will end in $\tilde{C}_t$. Such a path can be found by our algorithm and projected to a path from $s$ to $t$ in $G$.

### 4.5 Tolerating additive error in the input stationary distribution

We show that our algorithm still works when the input stationary distribution has a small additive error $\delta = 1/\mathrm{poly}(n, d, k)$. There are two places in the proof we need to take care of this error.

Observe that the only place our algorithm uses the input stationary distribution is in the first stage. We used the input stationary distribution to define the vertex and edge blow-up factors $a(v)$ and $b(v)$, and showed in Lemma 4.1, 4.3, and 4.4 that $G'$ is nearly regular. As the purpose of that analysis is to tolerate the round-off error as compared to the ideal construction, we can expect that our algorithm will still work when the error of the input stationary distribution is as small as the round-off error. Formally, it is not hard to check that when $\delta$ is small enough (e.g., $\delta = 1/ND$), the above three lemmas are still true.

In Stage 2, we argued that since $\pi(V_2) = 0$, all vertices $v \in V_2$ are "deleted" in $G'$. When the input stationary distribution has some additive error, the corresponding vertex set $V_2'$ in $G'$ will be nonempty because the blow-up factors $a(\cdot)$ and $b(\cdot)$ are no longer zero. However, as long as the size of $V_2'$ is small, say $O(\varepsilon N)$ vertices and $O(\varepsilon ND)$ edges, we can simply delete $V_2'$ from $G'$, and still set $V_{\mathrm{reg}} = V_1' \cup V_3'$, while maintaining all properties listed at the end of Section 4.2. When the error $\delta$ is small enough, say $\delta \le 1/ND$, we have $a(v) = \lceil p_v \cdot N \rceil = 1$ and $b(v) = \lceil p_v \cdot ND \rceil = 1$ for every $v \in V_2$, so we delete at most $n \le O(\varepsilon N)$ vertices and $nd \le O(\varepsilon ND)$ edges in $G'$.

Note that in the rest of the proof, we do not use the input stationary distribution. Therefore, our algorithm solves $\delta$-KNOWN-STATIONARY S-T CONNECTIVITY and $\delta$-KNOWN-STATIONARY FIND PATH for $\delta = 1/ND$ in deterministic logspace.

## Acknowledgements

We thank the CCC '07 reviewers for helpful comments on the presentation.

## References


[1] R. Aleliunas, R. M. Karp, R. J. Lipton, L. Lovász, and C. Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *20th Annual Symposium on Foundations of Computer Science*, pages 218–223, San Juan, Puerto Rico, 29–31 Oct. 1979. IEEE.

[2] A. Amit, N. Linial, J. Matousek, and E. Rozenman. Random lifts of graphs. In *SODA*, pages 883–894, 2001.

[3] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal on Computing*, 13(4):850–864, Nov. 1984.

[4] K.-M. Chung, O. Reingold, and S. Vadhan. S-T connectivity on digraphs with a known stationary distribution. *Electronic Colloquium on Computational Complexity* Technical Report TR07-30, March 2007.

[5] J. A. Fill. Eigenvalue bounds on convergence to stationarity for nonreversible markov chains with an application to the exclusion process. *Annals of Applied Probability*, 1:62–87, 1991.

[6] I. Haitner, D. Harnik, and O. Reingold. On the power of the randomized iterate. In *CRYPTO*, pages 22–40, 2006.

[7] R. Impagliazzo, V. Kabanets, and A. Wigderson. In search of an easy witness: exponential time vs. probabilistic polynomial time. *Journal of Computer and System Sciences*, 65(4):672–694, 2002.

[8] P. Indyk. Stable distributions, pseudorandom generators, embeddings and data stream computation. In *41st Annual Symposium on Foundations of Computer Science (Redondo Beach, CA, 2000)*, pages 189–197. IEEE Comput. Soc. Press, Los Alamitos, CA, 2000.

[9] V. Kabanets and R. Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.

[10] E. Kaplan, M. Naor, and O. Reingold. Derandomized constructions of k-wise (almost) independent permutations. In *APPROX-RANDOM*, pages 354–365, 2005.

[11] M. Mihail. Conductance and convergence of markov chains: a combinatorial treatment of expanders. In *Proc. of the 37th Conf. on Foundations of Computer Science*, pages 526–531, 1989.

[12] N. Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

[13] N. Nisan and A. Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, Oct. 1994.

[14] R. Raz and O. Reingold. On recycling the randomness of the states in space bounded computation. In *Proceedings of the Thirty-First Annual ACM Symposium on the Theory of Computing*, Atlanta, GA, May 1999.

[15] O. Reingold. Undirected st-connectivity in log-space. In *Proceedings of the 37th ACM Symposium on Theory of Computing*, pages 376–385, 2005.

[16] O. Reingold, L. Trevisan, and S. Vadhan. Pseudorandom walks in regular digraphs and the RL vs. L problem. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC '06)*, pages 457–466, 21–23 May 2006. Preliminary version on *ECCC*, February 2005.

[17] M. Saks and S. Zhou. $\mathrm{BP_hSPACE(S)} \subseteq \mathrm{SPACE(S)}$. *Journal of Computer and System Sciences*, 58(2):376–403, 1999. 36th IEEE Symposium on the Foundations of Computer Science (Milwaukee, WI, 1995).

[18] A. Sinclair and M. Jerrum. Approximate counting, uniform generation and rapidly mixing Markov chains. *Inform. and Comput.*, 82(1):93–133, 1989.

[19] D. Sivakumar. Algorithmic derandomization via complexity theory. In *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing*, pages 619–626 (electronic), New York, 2002. ACM.

[20] V. Trifonov. An O(log n log log n) space algorithm for undirected st-connectivity. In *STOC*, pages 626–633, 2005.

[21] A. C. Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science*, pages 80–91, Chicago, Illinois, 3–5 Nov. 1982. IEEE.