On Strings Having the Same Length-k Substrings

Giulia Bernardini
University of Trieste, Italy
CWI, Amsterdam, The Netherlands

Alessio Conte
University of Pisa, Italy

Esteban Gabory
CWI, Amsterdam, The Netherlands

Roberto Grossi
University of Pisa, Italy

Grigorios Loukides
King's College London, UK

Solon P. Pissis
CWI, Amsterdam, The Netherlands
Vrije Universiteit, Amsterdam, The Netherlands

Giulia Punzi
University of Pisa, Italy

Michelle Sweering
CWI, Amsterdam, The Netherlands

Abstract

Let $\mathsf{Substr}_k(X)$ denote the set of length-k substrings of a given string X for a given integer $k > 0$. We study the following basic string problem, called z-Shortest $\mathcal{S}_k$-Equivalent Strings: Given a set $\mathcal{S}_k$ of n length-k strings and an integer $z > 0$, list z shortest distinct strings $T_1, \ldots, T_z$ such that $\mathsf{Substr}_k(T_i) = \mathcal{S}_k$, for all $i \in [1, z]$. The z-Shortest $\mathcal{S}_k$-Equivalent Strings problem arises naturally as an encoding problem in many real-world applications; e.g., in data privacy, in data compression, and in bioinformatics. The 1-Shortest $\mathcal{S}_k$-Equivalent Strings, referred to as Shortest $\mathcal{S}_k$-Equivalent String, asks for a shortest string X such that $\mathsf{Substr}_k(X) = \mathcal{S}_k$.

Our main contributions are summarized below:

- Given a directed graph $G(V, E)$, the Directed Chinese Postman (DCP) problem asks for a shortest closed walk that visits every edge of G at least once. DCP can be solved in $\tilde{\mathcal{O}}(|E||V|)$ time using an algorithm for min-cost flow. We show, via a non-trivial reduction, that if Shortest $\mathcal{S}_k$-Equivalent String over a *binary alphabet* has a near-linear-time solution then so does DCP.
- We show that the length of a shortest string output by Shortest $\mathcal{S}_k$-Equivalent String is in $\mathcal{O}(k + n^2)$. We generalize this bound by showing that the total length of z shortest strings is in $\mathcal{O}(zk + zn^2 + z^2 n)$. We derive these upper bounds by showing (asymptotically tight) bounds on the total length of z shortest Eulerian walks in general directed graphs.
- We present an algorithm for solving z-Shortest $\mathcal{S}_k$-Equivalent Strings in $\mathcal{O}(nk + n^2 \log^2 n + zn^2 \log n + |\mathsf{output}|)$ time. If $z = 1$, the time becomes $\mathcal{O}(nk + n^2 \log^2 n)$ by the fact that the size of the input is $\Theta(nk)$ and the size of the output is $\mathcal{O}(k + n^2)$.

2012 ACM Subject Classification Theory of computation → Pattern matching

Keywords and phrases string algorithms, combinatorics on words, de Bruijn graph, Chinese Postman

Digital Object Identifier 10.4230/LIPIcs.CPM.2022.16

Funding The work in this paper is supported in part by: the MIUR project Algorithms for HArnessing networked Data (AHeAD); and by the PANGAIA and ALPACA projects that have received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreements No 872539 and 956229, respectively.
Giulia Bernardini: Supported by the Netherlands Organisation for Scientific Research (NWO) under project OCENW.GROOT.2019.015 "Optimization for and with Machine Learning (OPTIMAL)".
Grigorios Loukides: Supported by the Leverhulme Trust RPG-2019-399 project.
Michelle Sweering: Supported by the Netherlands Organisation for Scientific Research (NWO) through Gravitation-grant NETWORKS-024.002.003.

33rd Annual Symposium on Combinatorial Pattern Matching (CPM 2022).
Editors: Hideo Bannai and Jan Holub; Article No. 16; pp. 16:1–16:17
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

We start with some basic definitions and notation on strings from [6]. Let $X = X[0] \cdots X[n{-}1]$ be a *string* of length $|X| = n$ over an alphabet Σ whose elements are called *letters*. For any two positions i and $j \geq i$ of X, $X[i \mathinner{.\,.} j]$ is the *fragment* of X starting at position i and ending at position j. The fragment $X[i \mathinner{.\,.} j]$ is an *occurrence* of the underlying *substring* $P = X[i] \cdots X[j]$; we say that P occurs at *position* i in X. A *prefix* of X is a fragment of the form $X[0 \mathinner{.\,.} j]$ and a *suffix* of X is a fragment of the form $X[i \mathinner{.\,.} n{-}1]$. By XY or $X{\cdot}Y$ we denote the *concatenation* of two strings X and Y, i.e., $XY = X[0] \cdots X[|X| - 1]Y[0] \cdots Y[|Y| - 1]$.

Let $\mathsf{Substr}_k(X)$ denote the set of length-k substrings of a finite string X. We consider the following basic problem on strings.

z-Shortest $\mathcal{S}_k$-Equivalent Strings
Input: A set $\mathcal{S}_k$ of n length-k strings over an integer alphabet $\Sigma = [0, nk)$ and an integer $z > 0$.
Output: A list $\mathcal{T}_z = T_1, \ldots, T_z$ of z distinct strings over Σ, such that for all $i \in [1, z]$, $\mathsf{Substr}_k(T_i) = \mathcal{S}_k$ and for every string T' not in $\mathcal{T}_z$ with $\mathsf{Substr}_k(T') = \mathcal{S}_k$, $|T'| \geq |T_i|$, for all $i \in [1, z]$; or FAIL if that is not possible.

In particular, if $z = 1$ the problem consists in finding a shortest string X such that $\mathsf{Substr}_k(X) = \mathcal{S}_k$. In this case, we call the problem Shortest $\mathcal{S}_k$-Equivalent String. We solve z-Shortest $\mathcal{S}_k$-Equivalent Strings by considering the de Bruijn graph of order k of $\mathcal{S}_k$ and reducing this problem to the problem of listing Eulerian walks on directed graphs. Let us first recall a few basic definitions before formally defining the problem in scope. Given $\mathcal{S}_k$, the *de Bruijn graph* (dBG) of order k of $\mathcal{S}_k$ is a directed multigraph $G_{\mathcal{S}_k} = (V, E)$, where V is the set of length-$(k-1)$ substrings of the strings in $\mathcal{S}_k$, and $G_{\mathcal{S}_k}$ contains an edge (u, v) if and only if the string $S = u[0] \cdot v$ is equal to the string $u \cdot v[k-2]$ and $S \in \mathcal{S}_k$. A *walk* in a directed graph $G(V, E)$ is a sequence of edges from E which joins a sequence of nodes from V. An *Eulerian walk* in G is a walk which visits all edges in E at least once. Any string X such that $\mathsf{Substr}_k(X) = \mathcal{S}_k$ corresponds to an Eulerian walk W in the dBG of order k of $\mathcal{S}_k$ and vice-versa. We formally define the problem of listing Eulerian walks in directed graphs.

z-Shortest Eulerian Walks
Input: A directed graph $G(V, E)$ and an integer $z > 0$.
Output: A list $\mathcal{W}_z = W_1, \ldots, W_z$ of z distinct Eulerian walks of G, such that for every Eulerian walk W' of G not in $\mathcal{W}_z$, $|W'| \geq |W_i|$, for all $i \in [1, z]$; or FAIL if that is not possible.

If $z = 1$, we call the problem Shortest Eulerian Walk. Let us denote the total size of $\mathcal{W}_z$ (that is, the total length of the walks) by $||\mathcal{W}_z||$. We show the following result[1].

▶ **Theorem 1.** *The z-Shortest Eulerian Walks problem can be solved in:*
- $\mathcal{O}(|E||V| \log^2 |V| + z|V|^3 + ||\mathcal{W}_z||)$ *time;*
- *or* $\mathcal{O}(|E||V| \log^2 |V| + z(|E||V| + |V|^2 \log |V|) + ||\mathcal{W}_z||)$ *time.*

We also investigate the combinatorial bounds on the total length of z shortest Eulerian walks in directed graphs. We show the following result.

▶ **Theorem 2.** *$||\mathcal{W}_1|| \leq |V|\,|E|$ and this bound is asymptotically tight. Moreover, $||\mathcal{W}_z|| \leq z|V|\,|E| + z^2|V|$ and this bound is asymptotically tight.*

[1] We assume that basic arithmetic operations take constant time, which is the case when $z = \text{poly}(|E|)$.

By employing Theorem 2 we show the following result, where $||\mathcal{T}_z||$ denotes the total length of the strings output for z-Shortest $\mathcal{S}_k$-Equivalent Strings.

▶ **Theorem 3.** *$||\mathcal{T}_1|| = \mathcal{O}(k + n^2)$. Moreover, $||\mathcal{T}_z|| = \mathcal{O}(zk + zn^2 + z^2 n)$.*

By employing Theorem 1 and Theorem 3 we show the following result (recalling that the size of the input in z-Shortest $\mathcal{S}_k$-Equivalent Strings is $\Theta(nk)$).

▶ **Theorem 4.** *The z-Shortest $\mathcal{S}_k$-Equivalent Strings problem can be solved in $\mathcal{O}(nk + n^2 \log^2 n + zn^2 \log n + ||\mathcal{T}_z||)$ time. If $z = 1$ this becomes $\mathcal{O}(nk + n^2 \log^2 n)$.*

We complement these results with the following reduction of independent interest. Given a directed graph $G(V, E)$, the Directed Chinese Postman (DCP) problem (also known as the Route Inspection problem) asks for a shortest closed walk that visits every edge of G at least once. DCP can be solved using an algorithm for computing a min-cost flow [8]. To this end, we can use the *network simplex algorithm* to solve DCP in $\tilde{\mathcal{O}}(|E||V|)$ time [28, 32]. We show here, via a non-trivial reduction, that if the Shortest $\mathcal{S}_k$-Equivalent String problem over a *binary alphabet* has a near-linear-time solution then so does DCP.

Motivation and Related Work. The main theoretical motivation for this work comes from the following "gap" in the literature. Let $\mathcal{M}_k$ be a *multiset* (rather than a set) of length-k strings. Counting (resp. listing) all distinct strings whose multiset of length-k strings is $\mathcal{M}_k$ corresponds to counting (resp. listing) all node-distinct Eulerian trails (i.e., walks that do not repeat any edges) in the de Bruijn multigraph of order k of $\mathcal{M}_k$ [16, 17, 22, 3]. Counting all node-distinct Eulerian trails can be done in polynomial time by employing the well-known BEST theorem [34] (see also [21] for the analogous result on strings). Efficient algorithms for listing z node-distinct Eulerian trails are also known [5, 24]. However, the analogous, perhaps more basic, results for counting or listing all strings whose *set* of length-k strings is $\mathcal{S}_k$ are, to the best of our knowledge, unknown. Here we focus on listing by observing that strings whose set of length-k strings is $\mathcal{S}_k$ correspond to Eulerian walks in the dBG of order k of $\mathcal{S}_k$. Counting remains wide open, as an analogous to BEST theorem for Eulerian walks is unknown. Indeed, a fundamental difference is that $\mathcal{M}_k$, by definition, gives the exact length of all strings whose multiset of length-k substrings is $\mathcal{M}_k$, while $\mathcal{S}_k$ gives only a lower bound.

The practical motivation for this work comes from the fact that the z-Shortest $\mathcal{S}_k$-Equivalent Strings problem arises naturally as an encoding problem in many real-world applications. In data privacy, the output strings can be used to construct reverse-safe data structures for pattern matching when $\mathcal{S}_k$ comes from a private string [2, 3]. In data compression, the output strings can be used to represent compactly a set $\mathcal{S}_k$ of length-k strings [26]. In bioinformatics, the output strings correspond to different possible genome reconstructions [29] when $\mathcal{S}_k$ is a set of sequences generated by a sequencing experiment.

Our work is in some sense related to Simon's congruence [31]. Two strings are $\sim_k$-*congruent* if they have the same set of *subsequences* of length at most k. For details on the combinatorial properties of the congruence see [31, 25, 19, 18, 20, 1] and for some algorithmic works see [15, 11, 33, 7, 10, 1, 12]. A long-standing open problem was to design an algorithm which, given two strings S and T, computes the largest k for which $S \sim_k T$. Gawrychowski et al. [12] have recently settled the problem optimally by showing a linear-time algorithm.

Paper Organization. In Section 2 we provide some basic definitions and notation. In Section 3 we present the reduction from DCP to the Shortest $\mathcal{S}_k$-Equivalent String problem. In Section 4 we show the combinatorial bounds on the length of a shortest Eulerian walk and on the total length of z shortest Eulerian walks. From these bounds, we infer the bounds on the length of the strings output for z-Shortest $\mathcal{S}_k$-Equivalent Strings. In Section 5 we present our algorithm for solving the z-Shortest Eulerian Walks problem. From this algorithm, we infer our solution to z-Shortest $\mathcal{S}_k$-Equivalent Strings.

2 Preliminaries

We consider directed graphs $G(V, E)$ such that there is at most one directed edge (u, v) for any $u, v \in V$ (that is, all edges in E have multiplicity 1). For a graph $G(V, E)$, we call *multiplicity function* on G a mapping $E \to \mathbb{N}$. We denote by $G_\mu(V, E)$ the *multigraph* with *underlying graph* G and multiplicity function μ: G_μ is a version of G having $\mu(e)$ copies of each edge $e \in E$. We call G_μ an *extension* of G if $\mu > 0$. We call v the *head* of an edge (u, v), and we call u the edge *tail*.

A *walk* in G is any sequence $W = e_1 e_2 \dots e_{|W|}$ of edges in E such that the head of e_i is equal to the tail of e_{i+1}, for all $i \in [1, |W| - 1]$; $|W|$ is the *length* of W. A walk may traverse any edge multiple times. We denote by $I(W)$ the tail of e_1, by $L(W)$ the head of $e_{|W|}$ and by $\mathsf{mult}_e(W)$ the number of times W visits e, for any $e \in E$. A walk W is *closed* (and in this case we call it a *cycle*) if $I(W) = L(W)$, that is, if it starts and ends at the same node. A walk W is *Eulerian* if it traverses all the edges of G at least once, that is, if the set $\{e_i\}_{i\in[1,|W|]} = E$. A walk that does not traverse any edge twice is a *trail*.

Given a multigraph G_μ, a walk W is Eulerian on G_μ if $\mathsf{mult}_e(W) \geq \mu(e)$ for every $e \in E$ and it is an Eulerian trail on G_μ if $\mathsf{mult}_e(W) = \mu(e)$ for every $e \in E$. A (multi-)graph $G(V, E)$ is *semi-Eulerian* if it admits an Eulerian trail, *Eulerian* if it admits an Eulerian cycle, and *strictly semi-Eulerian* if it is semi-Eulerian but not Eulerian. We denote by $EW(G)$ and $ET(G)$ the set of Eulerian walks and the set of Eulerian trails on G, respectively.

A graph $G(V, E)$ is *strongly connected* if, for any two nodes $u, v \in V$ with $u \neq v$, there exist both a walk from u to v and from v to u. The *strongly connected components* (SCCs in short) of G are its inclusion-maximal strongly connected subgraphs. A graph is *weakly connected* if replacing all of its directed edges with undirected edges yields a connected graph.

▶ **Definition 5** (Flow). *Let $G = (V, E)$ be a directed graph and $\delta : V \to \mathbb{Z}$ be a function called the* supply. *Let m (resp. M) be a function $E \to \mathbb{N} \cup \{+\infty\}$ called* minimal *(resp.* maximal*) capacity. A* flow *on G with supply δ, minimal capacity m and maximal capacity M is a function $f : E \to \mathbb{N}$ such that:*

$$\forall e \in E,\ m(e) \leq f(e) \leq M(e)$$

$$\forall v \in V,\ \sum_{e=(v,w)\in E} f(e) - \sum_{e=(w,v)\in E} f(e) = \delta(v)$$

We denote this set of flows by $\mathcal{F}(G, \delta, m, M)$.

3 Reducing Directed Chinese Postman to Shortest Equivalent String

Directed Chinese Postman (DCP)
Input: A directed graph $G(V, E)$.
Output: A shortest closed Eulerian walk, or FAIL if that is not possible.

The main goal of this section is to reduce DCP to Shortest $\mathcal{S}_k$-Equivalent String. We first show a simple linear-time reduction that uses an alphabet of size $\mathcal{O}(|V|)$. We then show a more involved near-linear-time reduction that uses a binary alphabet. The latter reduction has the following important implication: if Shortest $\mathcal{S}_k$-Equivalent String over a binary alphabet has a near-linear-time solution, then so does DCP.

3.1 Large Alphabet

▶ **Lemma 6.** *Given a directed graph $G(V,E)$, we define $\widetilde{G}(\widetilde{V},\widetilde{E})$ such that $\widetilde{V} = V \cup \{a,b\}$, where a,b are two bogus nodes, and $\widetilde{E} = E \cup \{(a,u),(u,b)\}$ for an arbitrary node $u \in V$.*

Then from any shortest Eulerian walk on $\widetilde{G}$, we can compute a shortest Eulerian closed *walk on G in constant time.*

Proof. Let $\widetilde{W}$ be a shortest Eulerian walk on $\widetilde{G}$. By definition, the bogus node a does not have any ingoing edge, and then since $\widetilde{W}$ must traverse edge (a,u), it has to start with a. By a symmetrical argument, it must end with b. The rest of the walk is on G, and since a and b are connected only to u, the latter is the second and second-to-last node crossed by $\widetilde{W}$. The edges traversed by $\widetilde{W}$ between these two visits of u form a closed Eulerian walk W on G.

If a shorter Eulerian cycle on G would exist, we could, by a rotation, make it start and end at u. Then, we could add edges (a,u) and (u,b) at the beginning and at the end, obtaining an Eulerian walk on $\widetilde{G}$ shorter than $\widetilde{W}$, a contradiction. Thus W is a shortest closed Eulerian walk on G, and we can compute it from $\widetilde{W}$ in constant time by removing the first and last edge traversed by $\widetilde{W}$. ◀

▶ **Theorem 7.** *Any instance of DCP can be reduced to an instance of the* Shortest $\mathcal{S}_k$-Equivalent String *problem in linear time.*

Proof. Let $\mathcal{I}_{\text{DCP}} = G(V,E)$ be an instance of DCP. We define $\widetilde{G}(\widetilde{V},\widetilde{E})$ as in Lemma 6. A walk W in $\widetilde{G}$ can be expressed as a sequence $v_0,\ldots,v_{|W|}$ of nodes from $\widetilde{V}$ such that $(v_i,v_{i+1}) \in \widetilde{E}$ for every $i = 0,\ldots,|W|-1$. Equivalently, such a walk W can be seen as a string $v_0 \ldots v_{|W|}$ over $\widetilde{V}$ such that its set of length-2 substrings $\mathcal{S}_2(W)$ is included in $\widetilde{E}$. By definition, W is Eulerian if and only if $\mathcal{S}_2(W)$ is exactly $\widetilde{E}$, so finding a shortest Eulerian walk W in $\widetilde{G}$ corresponds to finding a shortest string over $\widetilde{V}$ such that $\mathcal{S}_2(W) = \widetilde{E}$, which is an instance of the Shortest $\mathcal{S}_2$-Equivalent String problem. Finally, by Lemma 6, a solution to DCP on G can be obtained from an Eulerian walk W on $\widetilde{G}$ in constant time. ◀

3.2 Binary Alphabet

We reduce any instance $G(V,E)$ of the Directed Chinese Postman problem to an instance of Shortest k-Equivalent Strings over a binary alphabet $\Sigma = \{0,1\}$; we assume that G is weakly connected, otherwise the problem has a trivial solution FAIL.

For a given integer ℓ, we denote by $\mathbf{0}_\ell$ the constant string consisting of ℓ zeros. We assign to every $v \in V$ two binary strings v_A, v_B over $\{0,1\}$ such that:

- For every $v \in V$, v_A and v_B are different from each other and from any w_A and w_B for $w \neq v$ in V, so that one can identify v by knowing only v_A or v_B.
- All strings v_A and v_B start and end with a 1, and they all have the same length ℓ.

▶ **Observation 8.** *The length ℓ can be chosen to be in $\mathcal{O}(\log |V|)$.*

Proof. We need two unique binary strings v_A, v_B for every $v \in V$. Since there exist 2^α distinct binary strings of length α, we seek the minimum length ℓ such that $2^{\ell-2} \geq 2|V|$, because we restrict to strings starting and ending with 1. The observation follows. ◀

Let $k = 3\ell$. Our reduction models both nodes and edges of G with appropriate string gadgets. The node gadgets are defined as length-k strings $s(v) := v_A \cdot \mathbf{0}_\ell \cdot v_B$, for every node $v \in V$. Let $S_0 := \{s(v) \mid v \in V\}$ be the set of such gadgets. We model each edge $(u,v) \in E$ as a length-(7ℓ) string gadget $s(u) \cdot \mathbf{0}_\ell \cdot s(v)$. We define the set S of length-k

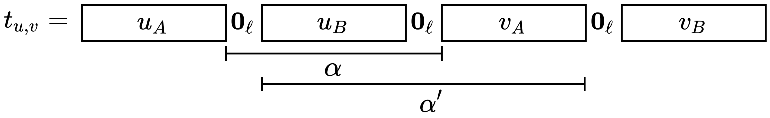

Figure 1 The configuration described in the proof of Lemma 10.

strings input to Shortest $\mathcal{S}_k$-Equivalent String as the set of length-k substrings of all the edge gadgets: $S := \cup_{(u,v)\in E}\mathsf{Substr}_k(s(u)\cdot\mathbf{0}_\ell\cdot s(v))$. Note that $S_0 \subseteq S$. Since $\ell = \mathcal{O}(\log|V|)$, each edge gadget has length $7\ell = \mathcal{O}(\log|V|)$ so it has $\mathcal{O}(\log|V|)$ substrings of length $k = 3\ell = \mathcal{O}(\log|V|)$. Therefore S contains $\mathcal{O}(|E|\log|V|)$ strings of length $\mathcal{O}(\log|V|)$.

▶ **Observation 9.** *Every occurrence of $\mathbf{0}_\ell$ in an edge gadget starts at position ℓ, 3ℓ, or 5ℓ of the edge gadget (i.e., it is one of the occurrences explicitly used for the construction). It follows that there are no spurious (i.e., not coming from one of the described occurrences in the edge gadgets) occurrences of $\mathbf{0}_\ell$ in the strings in S.*

Proof. Every string v_A or v_B starts and ends with a 1, so they cannot overlap an occurrence of $\mathbf{0}_\ell$. Since we never concatenate two copies of $\mathbf{0}_\ell$, the result follows. ◀

We now want to prove the correspondence between walks on G and strings having their length-k substrings in S. The following lemma is a first step in that direction: we show that, given a string T with $\mathsf{Substr}_k(T) \subseteq S$, the node gadgets never overlap in T, and that their succession precisely corresponds to adjacency relations between nodes in G.

▶ **Lemma 10.** *Let $G(V,E)$ be a directed graph and let S be the set of length-k strings defined above. Let T be a string with $\mathsf{Substr}_k(T) \subseteq S$. If, for some $u,v \in V$, T has a substring $t_{u,v}$ such that (i) $t_{u,v}$ has $s(u)$ as a prefix, (ii) $t_{u,v}$ has $s(v)$ as a suffix, and (iii) $t_{u,v}$ has no other substrings of the form $s(w)$ for $w \in V$, then $t_{u,v} = s(u)\cdot\mathbf{0}_\ell\cdot s(v)$ and $(u,v)\in E$.*

Proof. Let $t_{u,v}$ be a substring of T satisfying conditions (i)-(iii), so that $t_{u,v}[0\mathinner{.\,.}k-1] = s(u) = u_A\cdot\mathbf{0}_\ell\cdot u_B$ for some $u \in V$. Note that the length-k substring α starting at position ℓ of $t_{u,v}$ has $\mathbf{0}_\ell$ as a prefix. By the definition of S and Observation 9, every length-k string in S with $\mathbf{0}_\ell$ as a prefix has it also as a suffix, thus $\mathbf{0}_\ell$ is also a suffix of α (inspect Figure 1). Let α' be the length-k substring starting at position 2ℓ of $t_{u,v}$. String α' has u_B as a prefix, and it has an occurrence of $\mathbf{0}_\ell$ at position ℓ (the suffix of α). Since $\alpha' \in S$, and since it has a $\mathbf{0}_\ell$ in a central position, we know that α' is equal to $u_B\cdot\mathbf{0}_\ell\cdot w_A$ for some $w\in V$ such that $(u,w)\in E$. Let now β be the length-k substring of $t_{u,v}$ starting at position 3ℓ. We know that, since $\mathbf{0}_\ell$ is a prefix of β, it is also its length-ℓ suffix. Thus, the length-k substring β' starting at position 4ℓ of $t_{u,v}$, has $w_A\cdot\mathbf{0}_\ell$ as a prefix; by looking at S we find that $\beta' = s(w)$. Now, by definition of $t_{u,v}$, the string β' has to be $s(v)$. Thus, $w = v$, $(u,v)\in E$, and $t_{u,v} = s(u)\cdot\mathbf{0}_\ell\cdot s(v)$. ◀

▶ **Proposition 11.** *Let $G(V,E)$ be a directed graph, let S and S_0 be the sets of strings defined above, and let $\mathcal{T}$ be the set of strings of length at least $k+1$, having prefix and suffix in S_0 and such that $\mathsf{Substr}_k(T)\subseteq S$ for all $T\in\mathcal{T}$. The mapping*

$$\varphi : W = ((v_0,v_1),(v_1,v_2),\ldots,(v_{R-1},v_R)) \mapsto T = s(v_0)\cdot\mathbf{0}_\ell\cdot s(v_1)\cdot\ldots\cdot\mathbf{0}_\ell\cdot s(v_R)$$

defines a bijection between the set of walks on G and $\mathcal{T}$. From any $T\in\mathcal{T}$, $\varphi^{-1}(T)$ can be computed in $\mathcal{O}(|T|)$ time. Moreover, given a walk W, the set of edges traversed by W is exactly the set of $(u,v)\in E$ such that $u_B\cdot\mathbf{0}_\ell\cdot v_A$ are substrings of $\varphi(W)$.

Proof. The mapping is well defined between the given sets and it is injective by the definition of the string gadgets. Consider an arbitrary string $T \in \mathcal{T}$. Let $u, v \in V$ be nodes such that $s(u) \in S_0$ is a prefix of T and $s(v) \in S_0$ is a suffix of T. Let t be the second leftmost occurrence of any string from S_0 in T (the prefix and suffix of T are in S_0, and since the length of T is at least $k+1$, they do not coincide, thus there are at least two occurrences of strings from S_0). We have $t = s(w)$ for some $w \in V$, and then T has a prefix $t_{u,w}$ satisfying conditions (i)-(iii) of Lemma 10. Then $(u, w) \in E$, and $t_{u,w} = s(u) \cdot \mathbf{0}_\ell \cdot s(w)$. By induction, we can find a sequence of nodes $u_0 = u, u_1 = w, \ldots, u_R = v$ such that $T = s(u_0) \cdot \mathbf{0}_\ell \cdot \ldots \cdot \mathbf{0}_\ell \cdot s(u_R)$, and such that $(u_i, u_{i+1}) \in E$ for every $i = 0, \ldots, R-1$. This gives us a walk $W = u_0, u_1, \ldots, u_R$ on G with $\varphi(W) = T$. The mapping φ is therefore surjective, hence it is a bijection. Its inverse map can be computed in $\mathcal{O}(|T|)$ time, by storing the encodings $s(v)$ for $v \in V$ in a dictionary, and linearly constructing the list of v from the occurrences of $s(v)$ in T. ◀

We are now ready to prove the main result of this section.

▶ **Theorem 12.** *Any instance $G(V, E)$ of* Directed Chinese Postman *with output W can be reduced in $\mathcal{O}(|E| \log |V| + |W|)$ time to an instance of* Shortest $\mathcal{S}_k$-Equivalent String *with $n = |\mathcal{S}_k| = \mathcal{O}(|E| \log |V|)$ strings over the binary alphabet and $k = \mathcal{O}(\log |V|)$.*

Proof. Let $G(V, E)$ be a directed graph. We construct a graph $\widetilde{G}$ from G with bogus nodes a and b as in Lemma 6 and we construct sets S_0 and S on $\widetilde{G}$ as described at the beginning of this section. S contains $\mathcal{O}(|E| \log |V|)$ strings, each of them having length $\mathcal{O}(\log |V|)$ bits, and it can be constructed in $\mathcal{O}(|E| \log |V|)$ time by listing the binary encoding of the integers we assign to the nodes. We now prove that we can compute a shortest Eulerian walk in $\widetilde{G}$ from the output of Shortest $\mathcal{S}_k$-Equivalent String, and Lemma 6 concludes the proof.

Indeed, from the bijection in Proposition 11, every string in $\mathcal{T}$ gives rise to a unique walk in $\widetilde{G}$. Let T be a solution to Shortest $\mathcal{S}_k$-Equivalent String with $\mathcal{S}_k = S$. Since node a does not have any ingoing edge in $\widetilde{G}$, by the definition of S, the length-$(k-1)$ prefix of $s(a)$ cannot be a suffix of any string in S. Since T contains every string in S, the string $s(a)$ must be a prefix of T. By symmetry, the string $s(b)$ is a suffix of T. Finally, since $s(a) \neq s(b)$ by definition of the node gadgets, T has length at least $k+1$ and therefore $T \in \mathcal{T}$.

Now, by Proposition 11 we can get in $\mathcal{O}(|T|)$ time a unique walk $W = v_0, \ldots, v_R$ such that $T = s(v_0) \cdot \mathbf{0}_\ell \cdot s(v_1) \cdot \ldots \cdot s(v_R)$. The edges traversed by W are exactly edges $(u, v) \in \widetilde{E}$ such that $u_B \cdot \mathbf{0}_\ell \cdot v_A$ is a substring of T. But since T contains every string from S, and since $u_B \cdot \mathbf{0}_\ell \cdot v_A \in S$ for every $(u, v) \in \widetilde{E}$, the walk W traverses every edge in $\widetilde{G}$ and it is therefore Eulerian. Furthermore, it is minimal, as otherwise some shorter Eulerian walk W' would give rise to $T' = \varphi(W')$ shorter than T. But then T' would be a shorter string with $\mathsf{Substr}_k(T') = S$. It follows that $W = \varphi^{-1}(T)$ is a shortest Eulerian walk of $\widetilde{G}$. By looking at the bijection of Proposition 11, we get $|T| = \mathcal{O}(|W|k) = \mathcal{O}(|W| \log |V|)$ bits, so the total reduction time is $\mathcal{O}(|E| \log |V| + |W|)$. ◀

4 Combinatorial Bounds

The main goal of this section is to prove Theorem 2; in particular, to provide bounds on the quantities $||\mathcal{W}_1||$ and $||\mathcal{W}_z||$ for a directed graph $G(V, E)$. In Section 4.1, we prove the upper bound $||\mathcal{W}_1|| \leq |V| \cdot |E|$, and show that it is asymptotically tight. In Section 4.2, we prove the bound $||\mathcal{W}_z|| \leq z|V| \cdot |E| + z^2|V|$, and show that it is asymptotically tight as well. Finally, in Section 4.3, we employ Theorem 2 to prove Theorem 3.

Firstly, let us observe that there are three types of graphs to consider.

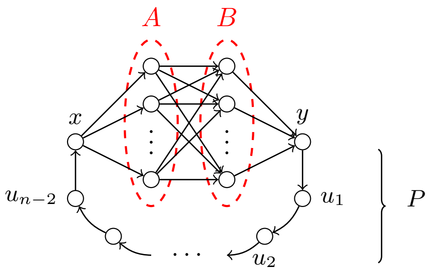

Figure 2 A directed graph where the shortest Eulerian walk has length $\Omega(|V||E|) = \Omega(|V|^3)$.

▶ **Observation 13.** *Every directed graph has either* 0, 1, *or infinitely many distinct Eulerian walks.*

Proof. Any directed path has exactly 1 Eulerian walk. It is easily seen that any other directed acyclic graph has none, and that any directed graph with 2 or more Eulerian walks must have a directed cycle:[2] This cycle may be repeated any number of times, yielding infinitely many distinct Eulerian walks. ◀

As the first two types of graphs are trivial with respect to this analysis, in the rest of the section we focus on graphs having infinitely many distinct Eulerian walks.

4.1 Length of a Shortest Eulerian Walk

▶ **Lemma 14.** *For any directed graph $G(V, E)$, we have $||\mathcal{W}_1|| \leq |V| \cdot |E|$.*

Proof. Let us first assume that G is strongly connected. Letting $e_1, \ldots, e_{|E|}$ be the edges of E, we inductively build a walk which traverses all of these edges in the given order, and which has length at most $|V| \cdot |E|$. The walk starts as the single edge e_1. Inductively, given a walk W' that crosses $e_1, \ldots, e_{h-1}$ and traverses at most $|V| \cdot (h-1)$ edges for any $h \in [2, |E|]$, there is a walk W connecting the head of e_{h-1} to the tail of e_h, since G is strongly connected. Moreover, W traverses at most $|V| - 1$ edges: indeed, if W goes through the same node twice, we can remove the cycle that it forms doing so. As a result, $W'We_h$ is a walk that crosses $e_1, \ldots, e_h$ and traverses at most $|V| \cdot h$ edges. The claim follows when $h = |E|$, noting that a shortest walk cannot be longer than the one that we have just described inductively.

Consider the general case of G and $\mathcal{W}_1 = \{W\}$. Walk W induces a topological order on the SCCs of G, as otherwise the walk cannot traverse all the edges, and these SCCs form a chain graph of the form $G_1 \to G_2 \to \cdots \to G_k$ (see [5]). Since each $G_i(V_i, E_i)$ is strongly connected, we can apply the above argument, so that W traverses at most $|V_i| \cdot |E_i|$ edges there. Overall, since $|E| = k - 1 + \sum_{i=1}^{k} |E_i|$, W traverses at most $k - 1 + \sum_{i=1}^{k} |V_i||E_i| \leq k - 1 + |V| \cdot \sum_{i=1}^{k} |E_i| \leq |V| \cdot |E|$ edges. ◀

We now show in Lemma 15 that the bound in Lemma 14 is asymptotically tight by providing an example of a graph G with $||\mathcal{W}_1|| = \Omega(|V| \cdot |E|) = \Omega(|V|^3)$.

▶ **Lemma 15.** *There is an infinite family of directed graphs, such that each graph $G(V, E)$ satisfies $||\mathcal{W}_1|| = \Omega(|V| \cdot |E|)$.*

[2] Note that some directed graphs with cycles may still allow 0 Eulerian walks.

Proof. We construct G as follows for any given $n \geq 3$ (inspect Figure 2): we start from two sets A, B of n nodes each, and the set of n^2 edges $F = \{(a,b) \mid a \in A, b \in B\}$ (i.e., A, B induce a complete directed bipartite graph with edges directed from A to B). We then add two more nodes x, y and add edges (x, a) for all $a \in A$, and edges (b, y) for all $b \in B$. We further connect y to x by adding $n-2$ new nodes $u_1, \ldots, u_{n-2}$ and all edges of $P = \{(y, u_1)\} \cup \{(u_{n-2}, x)\} \cup \{(u_i, u_{i+1}) \mid i = 1, \ldots, n-3\}$. In total, this graph has $|V| = 3n$ nodes and $|E| = n^2 + 3n - 1 = \Theta(|V|^2)$ edges.

Let us now see why the length of a shortest Eulerian walk W of G is $\Omega(|V| \cdot |E|)$. By definition, $\mathcal{W}_1$ must traverse all the n^2 edges in the set of edges F. Without loss of generality, let $(a_1, b_1), \ldots, (a_{n^2}, b_{n^2})$ denote the order in which W traverses the edges in F. Let $i < n^2$, and let us consider the point when W has just traversed (a_i, b_i) for the first time: to be able to reach the next (a_{i+1}, b_{i+1}), the walk necessarily needs to traverse the whole of P. This means that W needs to traverse all $n-1$ edges of P once for each $i = 1, \ldots, n^2 - 1$. Furthermore, it must also traverse at least one time all the remaining $2n$ edges of the form (x, a) for all $a \in A$, and (y, b) for all $b \in B$. This leads to a total length of at least $(n-1)(n^2-1) + 2n = n^3 - n^2 + n + 1 = \Omega(|V| \cdot |E|)$, proving the claim. ◀

4.2 Total Length of z Shortest Eulerian Walks

▶ **Lemma 16.** *For any directed graph $G(V, E)$, if $c_1 \leq \ldots \leq c_z$ are the lengths of the walks in $\mathcal{W}_z$, then $c_i \leq |V|(i - 1 + |E|)$, for all $i \in [1, z]$, and we have $||\mathcal{W}_z|| \leq z|V| \cdot |E| + z^2|V|$.*

Proof. First observe that if G is acyclic, then it has at most one Eulerian walk, and by Lemma 14 its length is at most $|V| \cdot |E|$, so the claim holds for $z = 1$.

We can then assume that G has a cycle, let C be the shortest one. Given an Eulerian walk W, the length of the next shortest walk is always bounded by $|W| + |C|$. In fact, let us consider a walk W' defined starting from W, to which we add a tour of cycle C when W traverses for the first time a node of C. Walk W' is Eulerian as it contains W, and it has length exactly $|W| + |C|$. The next shortest walk with respect to W cannot be longer than W', and so its length must be bounded by $|W| + |C| \leq |W| + |V|$.

Now we can prove by a simple induction that the i-th shortest length for an Eulerian walk is at most $||\mathcal{W}_1|| + (i-1)\,|V| \leq |V|(i - 1 + |E|)$, where the base case $i = 1$ is trivial (and bounded by Lemma 14) and the inductive step holds as the length of the shortest longer walk increases by at most $|V|$. Hence, $||\mathcal{W}_z|| \leq \sum_{i=1}^{z}(||\mathcal{W}_1|| + (i-1)\,|V|) \leq z|V||E| + z^2|V|$. ◀

We now show in Lemma 17 that the bound in Lemma 16 is asymptotically tight.

▶ **Lemma 17.** *There are infinite families of directed graphs, such that each graph $G(V, E)$ satisfies either $||\mathcal{W}_z|| = \Omega(z|V| \cdot |E|)$ or $||\mathcal{W}_z|| = \Omega(z^2|V|)$, respectively.*

Proof. As clearly the i-th shortest Eulerian walk, for any i, is not shorter than the shortest one, we have $||\mathcal{W}_z|| \geq z \cdot ||\mathcal{W}_1||$, so the first family follows from Lemma 15 (shown in Figure 2). As for the second, simply consider a directed cycle with an incoming pendant edge and an outgoing one: more formally, the directed cycle $(v_1, v_2), \ldots, (v_{|V|-2}, v_1)$ of length $|V| - 2$, plus two nodes x, y and the edges (x, v_1) and (v_1, y). Any Eulerian walk must start from the source node x, traverse the whole cycle at least once (say, $i > 0$ times), and then follow the edge (v_1, y). As different walks must traverse the cycle a different number of times, it follows that the i-th shortest Eulerian walk has length $2 + i(|V| - 2)$, so $||\mathcal{W}_z|| = \sum_{i=\{1,\ldots,z\}}(2 + i(|V| - 2)) = \Omega(z^2|V|)$. ◀

We can conclude from Lemma 17 that a better worst-case bound than $\Omega(z|V| \cdot |E| + z^2|V|)$ is not possible, so Lemmas 14, 15, 16, 17 yield directly Theorem 2.

4.3 Total Length of z Shortest Equivalent Strings

Let us now prove Theorem 3, which infers upper bounds on the total length of a solution to z-Shortest $\mathcal{S}_k$-Equivalent Strings. We use the following observation:

▶ **Observation 18.** *Let $\mathcal{S}_k$ be a set of n strings each of length k, such that there exists a string T with $\mathsf{Substr}_k(T) = \mathcal{S}_k$. Let $G(V, E)$ be the dBG of order k of $\mathcal{S}_k$. It holds (i) $|E| = n$; and (ii) $|V| \leq n + 1$.*

Proof. (i) It follows from the definition of dBG: each edge corresponds to a string from $\mathcal{S}_k$ and reciprocally, if $S \in \mathcal{S}_k$ there is an edge between $S[0\,..\,k-2]$ and $S[1\,..\,k-1]$, which corresponds to S. Therefore $|E| = |\mathcal{S}_k| = n$.

(ii) Since there exists T with $\mathsf{Substr}_k(T) = \mathcal{S}_k$, we know that there is a walk W traversing every edge in G. It follows that all $e \in E$ (except possibly for the first and last edges traversed by W) are such that the head of e coincides with the tail of some $e' \in E$, and symmetrically, the tail of e coincides with the head of some $e'' \in E$. We conclude that there cannot be more than $|E| + 1$ distinct nodes, thus $|V| \leq |E| + 1 = n + 1$. ◀

Let $\mathcal{S}_k$ be a set of n strings each of length k; we know that if $G(V, E)$ is the dBG of order k of $\mathcal{S}_k$, a (shortest) string T such that $\mathsf{Substr}_k(T) = \mathcal{S}_k$ has length $k - 1 + |W|$, where W is a (shortest) Eulerian walk on G. From Theorem 2 and Observation 18 we get that $||\mathcal{W}_1|| \leq |V| \cdot |E| \leq n^2 + n$ and so $||\mathcal{T}_1|| \leq k - 1 + n^2 + n = \mathcal{O}(k + n^2)$. Using again Theorem 2 and Observation 18, we get that $||\mathcal{W}_z|| \leq z|V| \cdot |E| + z^2|V|$ and so $||\mathcal{T}_z|| \leq z(k - 1 + n^2 + n) + z^2(n + 1) = \mathcal{O}(zk + zn^2 + z^2n)$. We have arrived at Theorem 3.

5 Listing z Shortest Eulerian Walks

The main goal of this section is to prove Theorem 1; in particular, to provide an efficient algorithm for solving z-Shortest Eulerian Walks. In Section 5.1, we start by providing the state-of-the-art results for listing z best flows in a directed graph; the underlying algorithms form the main computational routine of our algorithm, which is provided in detail next. Finally, in Section 5.2, we employ Theorem 1 and Theorem 3 to prove Theorem 4; namely, to provide an efficient algorithm for solving z-Shortest $\mathcal{S}_k$-Equivalent Strings.

5.1 Main Algorithm

In general, we can equip a graph $G(V, E)$ with a *cost* function $C : E \to \mathbb{N}$. Given a flow $f \in \mathcal{F}(G, \delta, m, M)$ for some supply function δ, and minimal (resp. maximal) capacities function m (resp. M), the *cost of f* is $C(f) = \sum_{e \in E} C(e)f(e)$. Let us now formally define the problem of listing z best flows (i.e., flows of minimal cost) from the set $\mathcal{F}(G, \delta, m, M)$ of all feasible flows (with respect to the given conditions) in a directed graph.

z-Best Flows
Input: A directed graph $G = (V, E)$, a supply function δ, a minimal capacity function m, a maximal capacity function M, a cost function C (all taking finite values), a min-cost flow f, and an integer $z > 0$.
Output: A list F of z flows in $\mathcal{F}(G, \delta, m, M)$, such that for every flow f' not in F, $C(f') \geq C(F[i])$, for all $i \in [0, z-1]$, and ordered such that $C(F[0]) \leq \cdots \leq C(F[z-1])$; or FAIL if that is not possible.

A min-cost flow f can be computed for any G using one of the following results.

▶ **Lemma 19** ([13, 27, 28, 32])**.** *Given any directed graph $G(V,E)$, computing a min-cost flow takes $\mathcal{O}((|E|\log|V|)(|E|+|V|\log|V|))$ time or $\mathcal{O}(|E||V|\log|V|\log K|V|)$ time, where K is the maximum cost of any edge of G.*

The following recent result for the z-Best Flows problem is known [23]; see also [14, 30].

▶ **Lemma 20** ([23])**.** *The z-Best Flows problem can be solved in $\mathcal{O}(z|V|^3)$ time or in $\mathcal{O}(z(|E||V|+|V|^2\log|V|))$ time.*

Main Idea. We list Eulerian walks as follows: we construct semi-Eulerian multigraphs G_μ, which are extensions of G obtained by duplicating some edges. Recall that each Eulerian walk in G can be seen as a trail in the semi-Eulerian extension G_μ for some multiplicity function μ such that, for every edge e, $\mu(e)=\mathsf{mult}_e(W)$. We will therefore treat Eulerian walks on G and Eulerian trails on G_μ as if they were the same objects, and use the fact that these semi-Eulerian extensions of G form a partition of the sets of Eulerian walks on G. Our algorithm has two steps: first, listing the extensions, and then, listing the trails in each extension, using the algorithm given in [5]. For the first part, we need the following:

▶ **Proposition 21.** *Let $G=(V,E)$ be a directed graph and let $M=\mathbb{N}^E$ be the set of all possible multiplicity functions on G. We have the following set equality:*[3]

$$EW(G)=\bigsqcup_{\mu\in M} ET(G_\mu) \tag{1}$$

Proof. An Eulerian walk W on G is an Eulerian trail on the semi-Eulerian multigraph obtained by taking as many copies of each edge as the number of times W traverses it. ◀

To list semi-Eulerian extensions, we will use flows. For Eulerian extensions, the balancing conditions are precisely a flow problem, as specified in the following proposition.

▶ **Proposition 22.** *Let G be a directed weakly connected graph, and μ be a multiplicity function on G. The multigraph G_μ is Eulerian if and only if the* balancing conditions *hold:*

$$\sum_{(u,w)\in E}\mu(u,w)-\sum_{(w,u)\in E}\mu(w,u)=0 \qquad \textit{for any fixed } u\in V$$

and is semi-Eulerian if the equality above holds or if it exists $(a,b)\in V$ such that:

$$\sum_{(u,w)\in E}\mu(u,w)-\sum_{(w,u)\in E}\mu(w,u)=0 \qquad \textit{for any fixed } u\in V\setminus\{a,b\}$$

$$\sum_{(a,w)\in E}\mu(a,w)-\sum_{(w,a)\in E}\mu(w,a)=1 \qquad \sum_{(b,w)\in E}\mu(b,w)-\sum_{(w,b)\in E}\mu(w,b)=-1$$

Proof. This is a reformulation of the well-known Euler's theorem [9] in the directed case with multiplicities. ◀

For any directed graph $G(V,E)$ and for $v\in V$, we write $\mathbb{1}_v$ for the indicator function on V of v; namely, for every $u\in V$, $\mathbb{1}_v(u)=1 \iff u=v$. We define:

- For every $v,w\in V$, a supply function $\delta_{v,w}: u\mapsto \mathbb{1}_v(u)-\mathbb{1}_w(u)$. If $v=w$, this is the null function, and if $v\neq w$, the function has value 1 on v, -1 on w, and 0 on any other node.
- For every $n\in\mathbb{N}\cup\{\infty\}$, a function c_n on E constantly equal to n.

[3] We use squared cup to underline that the sets $ET(G_\mu)$ are pairwise disjoint.

▶ **Observation 23.** *An extension G_μ of a directed weakly connected graph G with multiplicities μ is semi-Eulerian if and only if*

$$\mu \in \bigcup_{(v,w)\in V^2} \mathcal{F}(G, \delta_{v,w}, c_1, c_\infty)$$

Proof. This is a reformulation of the balancing conditions (Proposition 22). ◀

The semi-Eulerian extensions of G exactly correspond to flows on G with supply function of the form $\delta_{v,w}$, with $(v, w) \in V^2$. However, we would like to treat all cases by solving a single flow problem in order to avoid solving $\mathcal{O}(|V|^2)$ of them separately.

Let $G = (V, E)$ be a directed graph, and $G_{s,t} = (V_{s,t}, E_{s,t})$ be an extension of G with $V_{s,t} = V \cup \{s, t\}$ for some $s, t \notin V$ and $E_{s,t} = E \cup (\bigcup_{v\in V}(s, v)) \cup (\bigcup_{v\in V}(v, t))$. We will compute flows on $G_{s,t}$ by defining a maximal capacity function equal to c_∞ on every edge, a minimal capacity function m with $m|_E = c_1$ and $m|_{(E_{s,t}\setminus E)} = c_0$, and a supply function $\delta_{s,t}$.

Because we set the *minimal* capacity to be 1 on every edge in E, and since the flow can enter and exit G from any node via the edges from s to every $v \in V$, there is almost an equivalence between flows in $\mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)$ and semi-Eulerian extensions of G, except for the following detail: a flow also contains information about the starting and ending point of some trail in the extension, as it specifies which nodes are connected to the bogus nodes. So multiple flows that differ only in the edges they use to connect to the bogus nodes correspond to the same extension of G: this happens when the extension is in fact Eulerian.

▶ **Proposition 24.** *Let $G = (V, E)$ be a directed graph and let $f \in \mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)$. There is a unique $i \in V$ such that $f(s, i) = 1$, and a unique $o \in V$ such that $f(o, t) = 1$. The flow f takes value 0 on every other edge with tail s or head t.*

Proof. The node s has supply 1. Since there are no ingoing edges for s, the outgoing flow has to be 1, so only one node i has $f(s, i) = 1$. The node t has supply -1. Since there are no outgoing edges for t, the ingoing flow has to be 1, so only one node o has $f(o, t) = 1$. ◀

We call $i = \mathbf{en}(f)$ the *entrance* of f and $o = \mathbf{ex}(f)$ its *exit*. Note that $\mathbf{en}(f)$ and $\mathbf{ex}(f)$ are not necessarily distinct. We can now formally describe the relation between the flows in $\mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)$ and the walks in G. We will make use of a function WalkToFlow that takes a walk on G and returns a specific flow on $G_{s,t}$, as defined in the following proposition.

▶ **Proposition 25.** *Let $G = (V, E)$ be a directed weakly connected graph and $G_{s,t}$ the graph extended with the additional nodes s, t, as defined above. For each Eulerian walk W on G with multiplicity function μ, there is a unique flow $\mathsf{WalkToFlow}(W) \in \mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)$ such that (i) $\mathsf{WalkToFlow}(W)|_E = \mu$, (ii) $\mathbf{en}(\mathsf{WalkToFlow}(W)) = I(W)$ and (iii) $\mathbf{ex}(\mathsf{WalkToFlow}(W)) = L(W)$. For every multiplicity function μ, we then have the following partition:*

$$ET(G_\mu) = \bigsqcup_{\substack{f\in\mathcal{F}(G_{s,t},\delta_{s,t},m,c_\infty)\\ f|_E=\mu}} \mathsf{WalkToFlow}^{-1}(f)$$

In particular, the sets $\mathsf{WalkToFlow}^{-1}(f)$ for $f \in \mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)$ are nonempty and pairwise disjoint.

Proof. Let μ be a multiplicity function such that an Eulerian walk W with multiplicity μ exists on G, or in other terms, that G_μ is semi-Eulerian. From Observation 23 we know that μ is a flow in $\mathcal{F}(G, \delta_{v,w}, c_1, c_\infty)$ for some nodes $v, w \in V$, namely $v = I(W)$, $w = L(W)$. It is easy to verify that conditions (i)-(iii) uniquely define the following flow:

$$\mathsf{WalkToFlow}(W) : e \mapsto \begin{cases} \mu(e) & \text{if } e \in E \\ 1 & \text{if } e = (s, I(W)) \text{ or } e = (L(W), t) \\ 0 & \text{otherwise} \end{cases}$$

Now, for every flow $f_0 \in \mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)$, the set $\mathsf{WalkToFlow}^{-1}(f_0)$ is the set of Eulerian trails in $G_{(f_0)|_E}$ starting at $\mathbf{en}(f_0)$ and ending at $\mathbf{ex}(f_0)$. To get all the Eulerian trails having a given multiplicity μ, we need to consider all the flows agreeing with μ on G, regardless of their entrance or exit. Hence we have:

$$ET(G_\mu) = \bigsqcup_{\substack{f \in \mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty) \\ f|_E = \mu}} \mathsf{WalkToFlow}^{-1}(f)$$

The sets $\mathsf{WalkToFlow}^{-1}(f)$ for $f \in \mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)$ are pairwise disjoint because they are defined as reciprocal sets for the mapping $\mathsf{WalkToFlow}$, and are nonempty. Indeed, for a flow $f \in \mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)$, the balancing conditions hold in $G_{f|_E}$ (Observation 23). One can then choose an Eulerian trail W in $G_{f|_E}$ starting at $\mathbf{en}(f)$ and ending at $\mathbf{ex}(f)$. We then obtain a partition of $ET(G_\mu)$. ◀

To compute shortest walks, we need to assign costs to the flows, which are equal to the lengths of the corresponding walks. This is achieved by defining a cost function C_{walks} on $G_{s,t}$ such that $C_{\text{walks}}|_E = c_1$ (the function constantly equal to 1), and $C_{\text{walks}}|_{E_{s,t} \setminus E} = c_0$ (the function constantly equal to 0). Note that this definition coincides with the definition of the minimal capacity function m that we use, but we distinguish the two functions for clarity.

▶ **Observation 26.** *Let us equip graph $G_{s,t}$ with the cost function C_{walks}. For any Eulerian walk W on G, the total cost of the flow $\mathsf{WalkToFlow}(W)$ on $G_{s,t}$ is the length of W.*

Proof. By the definition of $\mathsf{WalkToFlow}$ (Proposition 25), for a walk W of length ℓ we have

$$\sum_{e \in E_{s,t}} \mathsf{WalkToFlow}(W)(e) C(e) = \sum_{e \in E} \mathsf{WalkToFlow}(W)(e) = \ell$$ ◀

▶ **Proposition 27.** *Let $G(V, E)$ be a directed weakly connected graph. It holds*

$$EW(G) = \bigsqcup_{f \in \mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)} \mathit{WalkToFlow}^{-1}(f)$$

Proof. It follows directly from Proposition 21 and Proposition 25. ◀

Let $G(V, E)$ be a directed graph, $z \geq 1$ be an integer, and $A \in \mathbb{N} \cup \{\infty\}$. By $\mathcal{F}_{z,A}$ we denote the $\min(z, |\mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_A)|)$ minimal cost flows (with cost function C_{walks} and an arbitrary but fixed order on flows having the same total cost) in $\mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_A)$.

We now prove that we can list z shortest Eulerian walks in a directed graph G by computing only $\mathcal{F}_{z,\infty}$.

▶ **Proposition 28.** *Let $G = (V, E)$ be a directed weakly connected graph and $z > 0$ be an integer. We equip the extension $G_{s,t}$ with the cost function C_{walks}. Consider the set of flows $\mathcal{F}_{z,\infty}$ and a subset F_s of $\mathcal{F}_{z,\infty}$ containing exactly one representative of each class in $\mathcal{F}_{z,\infty}$ under the relation $f \sim f' \iff f|_E = f'|_E$.*

Then, each of the $\min(z, |EW(G)|)$ shortest Eulerian walks on G has multiplicities $f|_E$ for some $f \in F_s$, i.e., each of them is an Eulerian trail in some extension $G_{f|_E}$ with $f \in F_s$.

Proof. We prove that each of the $\min(z, |EW(G)|)$ shortest Eulerian walks on G corresponds to trails in the following set:

$$\mathbf{B} = \bigsqcup_{f \in F_s} ET(G_{f|_E})$$

From the definition of F and Observation 26, the walks in $\mathbf{B}$ are minimal, in the sense that any walk in $EW(G) \setminus \mathbf{B}$ is at least as long as any walk in $\mathbf{B}$. This implies the result if $|\mathbf{B}| \geq \min(z, |EW(G)|)$. To prove that $|\mathbf{B}| \geq \min(z, |EW(G)|)$, we will show that if $|\mathbf{B}| < z$, then $\mathbf{B} = EW(G)$, and this will complete the proof. If $|\mathbf{B}| < z$, we have:

$$\begin{aligned} \mathbf{B} &= \bigsqcup_{f \in F_s} ET(G_{f|_E}) = \bigsqcup_{f \in F_s} \bigsqcup_{f' \sim f} \mathsf{WalkToFlow}^{-1}(f') && \text{(by Proposition 27)} \\ &\supseteq \bigsqcup_{f \in F_s} \bigsqcup_{\substack{f' \in \mathcal{F}_{z,\infty} \\ f' \sim f}} \mathsf{WalkToFlow}^{-1}(f') = \bigsqcup_{f \in \mathcal{F}_{z,\infty}} \mathsf{WalkToFlow}^{-1}(f), \text{ so} \\ |\mathbf{B}| &\geq \sum_{f \in \mathcal{F}_{z,\infty}} |\mathsf{WalkToFlow}^{-1}(f)| \geq |\mathcal{F}_{z,\infty}| \end{aligned}$$

where the last inequality follows from the fact that we are considering the disjoint union of nonempty sets (Proposition 25). Recall that $|\mathcal{F}_{z,\infty}| = \min(z, |\mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)|)$, so if $|\mathbf{B}| < z$, then $|\mathcal{F}_{z,\infty}| < z$ and $\mathcal{F}_{z,\infty} = \mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)$. By Proposition 27, it follows that

$$|EW(G)| = \left| \bigsqcup_{f \in \mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)} \mathsf{WalkToFlow}^{-1}(f) \right| \leq |\mathbf{B}|,$$

so $|\mathbf{B}| \geq |EW(G)|$, and in fact $\mathbf{B} = EW(G)$ from the trivial inclusion. ◀

In order to compute flows in $\mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)$ by means of existing algorithms, we need to define an equivalent problem having finite maximal capacity on each edge, as the known algorithms are not constructed for infinite maximal capacities.

▶ **Lemma 29.** *Let $G(V, E)$ be a directed graph and $z \geq 1$ be an integer. We have that $\mathcal{F}_{z,\infty} = \mathcal{F}_{z,|V|(z-1+|E|)}$.*

Proof. Let $F = f_0, \dots$ (resp. $F' = f'_0, \dots, f'_N$) be the list of feasible flows for the maximal capacity function c_∞ (resp. $c_{|V|(z-1+|E|)}$), ordered by increasing cost and such that the order on flows having the same total cost is arbitrary but fixed. Note that the list F may be infinite. Each flow in the list F' is trivially in the list F. If $F = F'$ the lemma holds. Otherwise, let f_i be the minimal cost flow in F which is not in F', so that $f_j = f'_j$ for every $j < i$. If $i \leq z$, then by Proposition 27 there is at least one Eulerian walk W in $\mathsf{WalkToFlow}^{-1}(f_i)$, and this walk has length $C_{\text{walks}}(f_i)$. But from Lemma 16, the length of W is also at most $|V|(z - 1 + |E|)$, so $f_i \in F'$ (since the flows which appear strictly before in the list F' are less than z), which gives a contradiction. Therefore the first mismatch (if any) between F and F' has to be after the z first elements, and $\mathcal{F}_{z,\infty} = \mathcal{F}_{z,|V|(z-1+|E|)}$. ◀

Algorithm 1 EW-List$(G(V,E),\, z)$.

```
 1: if $G$ is not weakly connected then
 2:     return FAIL
 3: $s \leftarrow |V|+1$; $t \leftarrow |V|+2$; $V_{s,t} \leftarrow V \cup \{s,t\}$          ▷ Add two bogus nodes
 4: $E_{s,t} \leftarrow E \cup \{(s,v),(v,t) \mid v \in V\}$          ▷ Add $2|V|$ bogus edges
 5: Construct $G_{s,t}$, $\delta_{s,t}$, $m$, $c_{|V|(z-1+|E|)}$, and $C_{\text{walks}}$ accordingly
 6: $f \leftarrow$ Best-Flow$(G_{s,t}, \delta_{s,t}, m, c_{|V|(z-1+|E|)}, C_{\text{walks}})$          ▷ Lemma 19
 7: $F \leftarrow z$-Best-Flows$(G_{s,t}, \delta_{s,t}, m, c_{|V|(z-1+|E|)}, C_{\text{walks}}, f, z)$          ▷ Lemma 20
 8: $F_s \leftarrow$ list$(f|_E$ for $f \in F)$     ▷ We keep only the values flows take on the initial graph
 9: Remove duplicates in $F_s$ if any     ▷ Duplicates might arise from the loss of information
10: EW $\leftarrow \emptyset$; $i \leftarrow 0$;
11: while $|\text{EW}| < z$ do
12:     EW $\leftarrow$ EW $\cup \{$ET-List$(G_{F_s[i]}, z - |\text{EW}|)\}$ ▷ List Eulerian trails on extended $G$ [5, 24]
13:     $i \leftarrow i+1$;          ▷ Retrieve the next best flow
14:     if $i = |F_s|$ then          ▷ We have consumed all best flows
15:         break;
16: if $|\text{EW}| \geq z$ then
17:     return EW
18: else
19:     return FAIL
```

▶ **Lemma 30.** *Given a directed graph $G = (V, E)$ and an integer z, Algorithm 1 terminates and solves the z-Shortest Eulerian Walks problem on G.*

Proof. Algorithm 1 computes z flows with minimal cost in $\mathcal{F}(G_{s,t}, \delta_{s,t}, m, c_\infty)$ (for the cost function C_{walks}) as defined in Proposition 28, or all of them if there are less than z (Line 7), from Lemma 29. The set F_s is defined in Lines 8 and 9 as in Proposition 28; and in Line 12, the function ET-List computes, if they exist, the $z - |EW|$ shortest elements from $\text{EW}(G_{f|_E})$ for every $f \in F_s$ (where $|EW|$ is, at each step, the number of Eulerian walks already computed), so we know that no more than z Eulerian walks are computed in the end. The function is implemented by means of the algorithm provided in [5] or the one in [24]. The correctness then directly follows from the equivalence proved in Proposition 28. ◀

▶ **Lemma 31.** *Given a directed graph $G = (V, E)$ and an integer z, Algorithm 1 requires:*

- $\mathcal{O}(|E||V|\log^2|V| + z|V|^3 + ||\mathcal{W}_z||)$ *time;*
- *or* $\mathcal{O}(|E||V|\log^2|V| + z(|E||V| + |V|^2\log|V|) + ||\mathcal{W}_z||)$ *time.*

Proof. Computing a min-cost flow takes $\mathcal{O}(|E||V|\log^2|V|)$ time by Lemma 19 because the maximum cost of any edge is 1. Finding z flows with minimal cost (or all of them if there are less) takes $\mathcal{O}(z|V|^3)$ time or $\mathcal{O}(z(|E||V| + |V|^2\log|V|))$ time by Lemma 20. Listing z Eulerian trails takes $\mathcal{O}(||\mathcal{W}_z||)$ time by applying the algorithm of [5] or the one of [24]. ◀

Lemmas 30 and 31 imply Theorem 1.

5.2 Listing z Shortest Equivalent Strings

Any instance of the z-Shortest $\mathcal{S}_k$-Equivalent Strings problem can be reduced to some instance of the z-Shortest Eulerian Walks problem in linear time. In particular, this reduction consists in constructing the dBG of order k of $\mathcal{S}_k$ in $\mathcal{O}(nk)$ time [4]. The resultant dBG is the directed graph $G(V, E)$ given as input to z-Shortest Eulerian Walks. By Observation 18, the total number of nodes in V is $\mathcal{O}(n)$ and the total number of edges in E is also $\mathcal{O}(n)$. Any Eulerian walk W in G corresponds to a string of length $k - 1 + |W|$: $k - 1$ is the length of the first node of the walk to which we concatenate one letter for each of the $|W|$ edges of the walk. Thus by employing Theorem 1 and Theorem 3 we obtain Theorem 4.

References

1 Laura Barker, Pamela Fleischmann, Katharina Harwardt, Florin Manea, and Dirk Nowotka. Scattered factor-universality of words. In *Developments in Language Theory – 24th International Conference*, volume 12086 of *Lecture Notes in Computer Science*, pages 14–28. Springer, 2020. doi:10.1007/978-3-030-48516-0_2.

2 Giulia Bernardini, Huiping Chen, Gabriele Fici, Grigorios Loukides, and Solon P. Pissis. Reverse-safe data structures for text indexing. In *Proceedings of the Symposium on Algorithm Engineering and Experiments, ALENEX*, pages 199–213, 2020. doi:10.1137/1.9781611976007.16.

3 Giulia Bernardini, Huiping Chen, Gabriele Fici, Grigorios Loukides, and Solon P. Pissis. Reverse-safe text indexing. *ACM J. Exp. Algorithmics*, 26, 2021. doi:10.1145/3461698.

4 Bastien Cazaux, Thierry Lecroq, and Eric Rivals. Linking indexing data structures to de Bruijn graphs: Construction and update. *J. Comput. Syst. Sci.*, 104:165–183, 2019. doi:10.1016/j.jcss.2016.06.008.

5 Alessio Conte, Roberto Grossi, Grigorios Loukides, Nadia Pisanti, Solon P. Pissis, and Giulia Punzi. Beyond the BEST theorem: Fast assessment of Eulerian trails. In *Fundamentals of Computation Theory – 23rd International Symposium,*, volume 12867 of *Lecture Notes in Computer Science*, pages 162–175. Springer, 2021. doi:10.1007/978-3-030-86593-1_11.

6 Maxime Crochemore, Christophe Hancart, and Thierry Lecroq. *Algorithms on strings.* Cambridge University Press, 2007.

7 Maxime Crochemore, Borivoj Melichar, and Zdenek Tronícek. Directed acyclic subsequence graph – overview. *J. Discrete Algorithms*, 1(3-4):255–280, 2003. doi:10.1016/S1570-8667(03)00029-7.

8 Jack R. Edmonds and Ellis L. Johnson. Matching, Euler tours and the Chinese Postman. *Math. Program.*, 5(1):88–124, 1973. doi:10.1007/BF01580113.

9 Leonhard Euler. Solutio problematis ad geometriam situs pertinentis. *Euler Archive – All Works*, 53:15, 1741.

10 Lukas Fleischer and Manfred Kufleitner. Testing Simon's congruence. In *43rd International Symposium on Mathematical Foundations of Computer Science*, volume 117 of *LIPIcs*, pages 62:1–62:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.MFCS.2018.62.

11 Emmanuelle Garel. Minimal separators of two words. In *Combinatorial Pattern Matching, 4th Annual Symposium*, volume 684 of *Lecture Notes in Computer Science*, pages 35–53. Springer, 1993. doi:10.1007/BFb0029795.

12 Pawel Gawrychowski, Maria Kosche, Tore Koß, Florin Manea, and Stefan Siemer. Efficiently testing Simon's congruence. In *38th International Symposium on Theoretical Aspects of Computer Science,*, volume 187 of *LIPIcs*, pages 34:1–34:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.STACS.2021.34.

13 Andrew V. Goldberg and Robert Endre Tarjan. Solving minimum-cost flow problems by successive approximation. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 7–18. ACM, 1987. doi:10.1145/28395.28397.

14 Horst W. Hamacher. A note on K best network flows. *Ann. Oper. Res.*, 57(1):65–72, 1995. doi:10.1007/BF02099691.

15 Jean-Jacques Hébrard. An algorithm for distinguishing efficiently bit-strings by their subsequences. *Theor. Comput. Sci.*, 82(1):35–49, 1991. doi:10.1016/0304-3975(91)90170-7.

16 Joan P. Hutchinson. On words with prescribed overlapping subsequences. *Utilitas Mathematica*, 7:241–250, 1975.

17 Joan P. Hutchinson and Herbert S. Wilf. On Eulerian circuits and words with prescribed adjacency patterns. *J. Comb. Theory, Ser. A*, 18(1):80–87, 1975. doi:10.1016/0097-3165(75)90068-0.

18 Prateek Karandikar, Manfred Kufleitner, and Philippe Schnoebelen. On the index of Simon's congruence for piecewise testability. *Inf. Process. Lett.*, 115(4):515–519, 2015. doi:10.1016/j.ipl.2014.11.008.

19 Prateek Karandikar and Philippe Schnoebelen. The height of piecewise-testable languages with applications in logical complexity. In *25th EACSL Annual Conference on Computer Science Logic*, volume 62 of *LIPIcs*, pages 37:1–37:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CSL.2016.37.

20 Prateek Karandikar and Philippe Schnoebelen. The height of piecewise-testable languages and the complexity of the logic of subwords. *Log. Methods Comput. Sci.*, 15(2), 2019. doi:10.23638/LMCS-15(2:6)2019.

21 Juhani Karhumäki, Svetlana Puzynina, Michaël Rao, and Markus A. Whiteland. On cardinalities of k-abelian equivalence classes. *Theor. Comput. Sci.*, 658:190–204, 2017. doi:10.1016/j.tcs.2016.06.010.

22 Carl Kingsford, Michael C. Schatz, and Mihai Pop. Assembly complexity of prokaryotic genomes using short reads. *BMC Bioinform.*, 11:21, 2010. doi:10.1186/1471-2105-11-21.

23 David Könen, Daniel R. Schmidt, and Christiane Spisla. Finding all minimum cost flows and a faster algorithm for the K best flow problem. *CoRR*, abs/2105.10225, 2021. arXiv:2105.10225.

24 Kazuhiro Kurita and Kunihiro Wasa. Constant amortized time enumeration of Eulerian trails. *CoRR*, abs/2101.10473, 2021. arXiv:2101.10473.

25 M. Lothaire. *Combinatorics on Words*. Cambridge Mathematical Library. Cambridge University Press, 2nd edition, 1997. doi:10.1017/CBO9780511566097.

26 Alessio Orlandi and Rossano Venturini. Space-efficient substring occurrence estimation. *Algorithmica*, 74(1):65–90, 2016. doi:10.1007/s00453-014-9936-y.

27 James B. Orlin. A faster strongly polynomial minimum cost flow algorithm. *Oper. Res.*, 41(2):338–350, 1993. doi:10.1287/opre.41.2.338.

28 James B. Orlin. A polynomial time primal network simplex algorithm for minimum cost flows. *Math. Program.*, 77:109–129, 1997. doi:10.1007/BF02614365.

29 Pavel A. Pevzner, Haixu Tang, and Michael S. Waterman. An Eulerian path approach to DNA fragment assembly. *Proc Natl Acad Sci*, 98(17):9748–9753, 2001. doi:10.1073/pnas.171285098.

30 Antonio Sedeño-Noda and Juan José Espino-Martín. On the K best integer network flows. *Comput. Oper. Res.*, 40(2):616–626, 2013. doi:10.1016/j.cor.2012.08.014.

31 Imre Simon. Piecewise testable events. In *Automata Theory and Formal Languages, 2nd GI Conference*, volume 33 of *Lecture Notes in Computer Science*, pages 214–222. Springer, 1975. doi:10.1007/3-540-07407-4_23.

32 Robert Endre Tarjan. Dynamic trees as search trees via Euler tours, applied to the network simplex algorithm. *Math. Program.*, 77:169–177, 1997. doi:10.1007/BF02614369.

33 Zdenek Tronícek. Common subsequence automaton. In *Implementation and Application of Automata, 7th International Conference*, volume 2608 of *Lecture Notes in Computer Science*, pages 270–275. Springer, 2002. doi:10.1007/3-540-44977-9_28.

34 Tatyana van Aardenne-Ehrenfest and Nicolaas G. de Bruijn. Circuits and trees in oriented linear graphs. In Ira Gessel and Gian-Carlo Rota, editors, *Classic Papers in Combinatorics*, pages 149–163. Birkhäuser Boston, Boston, MA, 1987. doi:10.1007/978-0-8176-4842-8_12.