Available online at www.sciencedirect.com
ScienceDirect
Discrete Mathematics 309 (2009) 5298–5304
DISCRETE MATHEMATICS
www.elsevier.com/locate/disc

Minimum Eulerian circuits and minimum de Bruijn sequences☆

Martín Matamala[a], Eduardo Moreno[b,*]

[a] *Departamento de Ingeniería Matemática, Universidad de Chile, Centro de Modelamiento Matemático, UMR 2071, UCHILE-CNRS, Casilla 170-3, Correo 3, Santiago, Chile*
[b] *School of Engineering, Universidad Adolfo Ibañez, Santiago, Chile*

Received 29 June 2006; accepted 14 November 2007
Available online 26 December 2007

Abstract

Given a digraph (directed graph) with a labeling on its arcs, we study the problem of finding the Eulerian circuit of lexicographically minimum label. We prove that this problem is NP-complete in general, but if the labelling is *locally injective* (arcs going out from each vertex have different labels), we prove that it is solvable in linear time by giving an algorithm that constructs this circuit. When this algorithm is applied to a de Bruijn graph, it obtains the de Bruijn sequences with lexicographically minimum label.

Keywords: Eulerian circuits; Labelled digraph; De Bruijn sequences

1. Introduction

In this work we consider the problem of finding the Eulerian circuit of minimum lexicographical label. Note that in order to have a well-defined problem, we need to fix a starting vertex, so as to define an order in which vertices are visited. First of all, we study the complexity of this problem, analysing the following decision problem:

MIN-LEX-EULERIAN-CIRCUIT
Instance: An Eulerian digraph G, a labeling $l : A(G) \to \Sigma$ of its arcs, a starting vertex r and a word $X \in \Sigma^{|A(G)|}$.
Question: Is there an Eulerian circuit T starting at r such that $l(T) \leq X$?

We prove the hardness of the problem in Theorem 1.

Theorem 1. MIN-LEX-EULERIAN-CIRCUIT *is NP-complete.*

☆ This work was partially supported by Proyecto Anillo en Redes and Fondecyt 1060825.
* Corresponding author. Tel.: +56 2 369 3637; fax: +56 2 278 4413.
E-mail addresses: mmatamal@dim.uchile.cl (M. Matamala), eduardo.moreno@uai.cl (E. Moreno).

0012-365X/$ - see front matter
doi:10.1016/j.disc.2007.11.027

A labelling is *locally injective* if arcs going out from a given vertex have different labels. In Theorem 2 we prove that for locally injective labelling the problem becomes polynomially solvable.

Theorem 2. MIN-LEX-EULERIAN-CIRCUIT *is polynomially solvable for locally injective labelling.*

1.1. Definitions

Let G be a digraph. For an arc $e = (u, v)$ we denote by $u = t(e)$ and $v = h(e)$ its tail and its head vertices, respectively.

A *trail* is a sequence $T = e_1 e_2 \dots e_k$ of arcs e_j such that $t(e_i) = h(e_{i-1})$ for every $i = 1, 2, \dots, k$ and all arcs are distinct. Let e_i be an arc in the trail T. We denote by Te_i the subtrail $e_1 \dots e_i$, by $e_i T$ the subtrail $e_i \dots e_k$ and by $e_i T e_j$ the subtrail $e_i \dots e_j$ for $i \leq j$. If the tail of e_1 is equal to the head of e_k then T is a closed trail or *circuit*. A circuit is an Eulerian circuit if arcs of T are all arcs of G. An Eulerian digraph is a digraph with an Eulerian circuit.

Let G be a Eulerian digraph and let r be one of its vertices. A subdigraph D of G is called a *prebranching* with root r if each vertex of G, different from r, has out-degree 1 in D. When the underlying graph of D is connected then, D is called a *branching* with root r. In this case, the underlying graph is a tree and for each vertex v in G there is exactly one directed path in D from v to r.

A vertex v is *exhausted* by a trail T if the trail uses all arcs having the vertex v as head or tail.

Let r be a vertex of G and let D be a prebranching of G. A trail T starting at r *disregards* D if for each arc $e = (v, u)$ of D, either the vertex v is exhausted by the subtrail Te or the arc e is not used in T. Intuitively, an arc (v, u) in D is used by T if and only if all other arcs leaving v have already been used by T.

In [17,16] it is proved a correspondence between Eulerian circuits in a digraph and its branchings. This result is known as the BEST theorem, an acronym for author's names.

We recast this correspondence in the following manner:

Theorem 3 (*BEST*). *Let G be an Eulerian digraph and let r be a vertex of G. Let D be a prebranching of G rooted at r. Let T be a maximal trail starting at r which disregards D. Then T is an Eulerian circuit if and only if D is a branching of G with root r.*

Proof. If T is an Eulerian circuit which disregards D then, for each vertex v different from r, the last incident arc visited by T is that in D. As T is Eulerian, the underlying graph of D must be a tree. Since T ends at r, we conclude that D is a branching of G rooted at r. This proves the forward implication.

Conversely, let us assume that T does not exhaust all vertices of G and let us choose v as a non-exhausted vertex closest to r in D. As T is a maximal trail starting at r it must stops at r exhausting r, so that $v \neq r$. As T does not exhaust v, there is an arc (v, u) in D not used by T. Hence, u is closer than v to r in T and it is not exhausted, contradicting the choice of v. □

Let $l : A(G) \to \Sigma$ be a labelling of arcs of G over an alphabet Σ. The label of T, $l(T)$, is the word $l(e_1) \dots l(e_k) \in \Sigma^*$. When arcs going out from the same vertex have different labels we say that the labelling is *locally injective*.

Theorem 3 allows us to count the number of Eulerian circuits in a digraph, which grows exponentially fast in terms of the number of vertices. Hence, from the computational complexity point of view, to check any property of Eulerian circuits by enumerative methods is inefficient.

In this work we study the problem of finding the Eulerian circuit of the digraph whose label is lexicographically minimum among labels of all Eulerian circuits. A first approach to solve this problem is to construct a trail by using a greedy strategy: Starting at a given vertex r, follow the unvisited arc (if it exists) of minimum label. A trail constructed by this strategy is called a *alphabetical trail* starting at r.

Let G be a labelled Eulerian digraph and let r be a vertex of G. A prebranching D of G rooted at r is maximal if for each vertex v distinct from r, the arc leaving v in D has a maximum label among all arcs leaving v in G.

Note that an alphabetic trail starting at r is a maximal trail disregarding a prebranching of G rooted at r which is maximal. Although not always there is an alphabetical trail which is an Eulerian circuit (see Fig. 1), from Theorem 3 we can obtain necessary and sufficient condition on the labelling to guarantee that this happens.

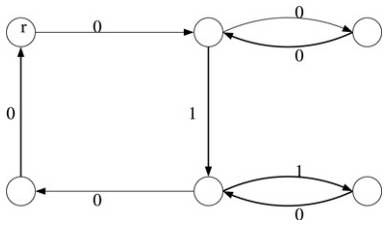

Fig. 1. Example of the alphabetical trail starting at r (of label 000100) which is not an Eulerian circuit because the maximal prebranching (bold arcs) is not a branching.

Corollary 4. *Let G be a labelled Eulerian digraph and let r be a vertex a G. There is an alphabetical trail T starting at r which is an Eulerian circuit if and only if there is a maximal prebranching of G rooted at r which is a branching of G.*

It is worth noting that if the labelling of G is locally injective, then there is only one maximal prebranching of G rooted at r, and there is only one alphabetical trail starting at r.

1.2. Motivation

Our inspiration for studying the complexity of this problem comes from the notion of de Bruijn sequences. A de Bruijn sequence of span n for an alphabet Σ is a periodic sequence such that every n-tuple occurs exactly once. Its first known description appears as a Sanskrit word *yamátárájabhánasalagám* which was a memory aid for Indian drummers, where accented/unaccented syllables represent long/short beats, so all possible triplets of short and long beats are included in the word. De Bruijn sequences are also known as "shift register sequences" and were originally studied by N.G. De Bruijn for the binary alphabet [3]. These sequences have many different applications, such as memory wheels in computers and other technological devices, network models, DNA algorithms, pseudo-random number generation, modern public-key cryptographic schemes, to mention a few (see [15,1,2]). Historically, de Bruijn sequences were studied in an arbitrary alphabet considering the language of all n-tuples. There are a large number of de Bruijn sequences in this case, but only a few can be generated efficiently, see [5] for a survey about this subject. In 1978, Fredricksen and Maiorana [6] give an algorithm to generate a de Bruijn sequence of span n based in the Lyndon words of the language, which happens to be the minimum one in the lexicographic order, and this algorithm was proved to be efficient [13].

An interesting framework to study de Bruijn sequences is graph theory: For a given integer n, a de Bruijn sequence of span n can be constructed as labels of an Eulerian circuit in a corresponding digraph, called de Bruijn graph of span n. In these graphs, vertices are words of length $n-1$, and there is an arc from u to v whenever the suffix of length $n-2$ of u and the prefix of length $n-2$ of v coincide. The arc (u, v) is labelled with the last letter of v. This graph was first defined implicitly in 1894 by Flye Saint-Marie [4] and it was explicitly detailed in 1946 by de Bruijn [3] and Good [8] independently. Note that this labelling is locally injective, since there are no multiple arcs.

In [14], the notion of de Bruijn sequences was extended to a set of n-tuples without a given forbidden word. A further generalization to arbitrary set of n-tuples, called *dictionary*, was proposed by one of the authors. This generalization was carried out by considering strongly connected subgraphs of the de Bruijn graph, induced by words in the dictionary. This idea was first used in [12]. In this framework, the existence of de Bruijn sequences for words in the dictionary is characterized by the existence of Eulerian circuits in the corresponding digraph [9].

Based on Corollary 4, we know that the alphabetical trail is an Eulerian circuit if there are no cycles in the maximal prebranching. On one hand, in the unrestricted case (all words are in the dictionary) the maximal prebranching is a branching and hence the alphabetical trail gives the lexicographically minimum Eulerian circuit. On the other hand, for general dictionary the maximal prebranching could have cycles. In [10] we characterize those dictionaries where the maximal prebranching is a branching.

When the alphabetical trail is not an Eulerian circuit, some vertices are not exhausted. If we restart an alphabetical trail, restricted to unvisited arcs, at the last non-exhausted vertex and we merge both trails, we obtain a longer trail that exhausts more vertices. By iterating this process we obtain a linear time algorithm computing an Eulerian circuit.

We shall prove in Theorem 8 that this algorithm constructs a minimum lexicographically Eulerian circuit, not only for the de Bruijn graphs, but also for any Eulerian digraph with a locally injective labelling, proving Theorem 2. A preliminary version of these results appears in [11].

2. Linear time algorithm for general digraphs

In this section, we prove our main result: the problem MIN-LEX-EULERIAN-CIRCUIT is polynomially solvable for all Eulerian digraphs having a locally injective labelling.

For a trail $T = e_1 \dots e_k$ over G, we define a *failure* of T as a pair (i, j) with $i < j$, $t(e_i) = t(e_j)$, $l(e_i) > l(e_j)$ and such that if $t(e_s) = t(e_i)$ for $s \in \{1, \dots, i-1\}$, then $l(e_s) < l(e_j)$. The vertex $t(e_i)$ is called the *failure vertex* of (i, j) and the subtrail $e_i T e_{j-1}$ is called the *failure subtrail* of (i, j).

Note that an alphabetical trail has no failures.

In order to construct the Eulerian trail of minimum label O, we present an inductive decomposition of O into subtrails O^i. The idea of our algorithm will be to reconstruct O from these subtrails.

First of all, we study the position and properties of failures in O: For a subset of arcs A' we denote by $t(A')$ the set $t(A') := \{t(e) : e \in A'\}$.

Lemma 5. *Let $O = e_1, \dots, e_p$ be the Eulerian trail of minimum label. If (i, j) is a failure, then*

$$t(e_{j+1} O) \cap t(e_i O e_j) = \emptyset.$$

Proof. By sake of contradiction, let e_k and e_s be arcs with $k \geq j+1$ and $i \leq s \leq j$ such that $t(e_k) = t(e_s)$. Depending on the value of s we construct an Eulerian trail $\tilde{O}$ starting with $O e_{i-1}$ and followed by the edge e_j. As $l(e_j) < l(e_i)$, this will contradict the minimality of O.

If $s = i$ or $s = j$ then

$$\tilde{O} = (O e_{i-1})(e_j O e_{k-1})(e_i O e_{j-1})(e_k O).$$

If $i < s < j$, then

$$\tilde{O} = (O e_{i-1})(e_j O e_{k-1})(e_s O e_{j-1})(e_i O e_{s-1})(e_k O). \quad \square$$

Let (i, j) and (s, k) be two failures of O with $i < s$. From Lemma 5, it follows that $t(e_i) \neq t(e_s)$. Otherwise, $s = j$, $t(e_k) = t(e_s) = t(e_j)$ and $k > s$, which violates Lemma 5. From Lemma 5 it also follows that either $i < s < k < j$ or $i < j < s < k$. Hence, two failure subtrails $e_i O e_j$ and $e_s O e_k$ either do not intersect or are contained one in the other. This property is also valid for any subtrail of O since by deleting arcs of O we do not create new failures.

The failure (i, j) of a subtrail O' of O with minimum j is called the *initial failure* of O' and its failure vertex the *initial failure vertex* of O'. Notice that there is no failure (s, k) with $i < s < k < j$ in O'. It follows that the failure subtrail T of an initial failure of a subtrail O' of O, is an alphabetical trail in the digraph $(V, A(T))$.

The decomposition of O is the following. Let $O^0 = O$. Let us assume that we have constructed O^i. If O^i has a failure, then we construct O^{i+1} from O^i by removing arcs of I^i, where I^i is the failure subtrail of the initial failure of O^i. Otherwise, the decomposition finishes.

¿From Lemma 5, it is easy to see that the set of failure vertices of O^{i+1} is the set of failure vertices of O^i minus u^i, the initial failure vertex of O^i. If we know O^i, it is easy to find u^i and I^i. Hence, the decomposition of O can be easily obtained.

In order to construct O we will reverse the process. We will construct O, by starting with an alphabetical trail, and adding subtrails in some non-exhausted vertices. In order to prove the correctness of our algorithm we need to establish a connection between non-exhausted vertices of O^{i+1} and failures of O^i. More precisely, we need to identify u^i and I^i, the initial failure vertex and the initial failure subtrail of O^i, from O^{i+1}.

We denote by $LastNotEx(T)$ the last vertex visited by a trail T among all vertices not exhausted by T.

Lemma 6. *Let $\{O^i : i = 0, \dots, q\}$ be the decomposition of the minimum lexicographically Eulerian circuit O. Then, $LastNotEx(O^{i+1})$ is the initial failure vertex of O^i.*

Proof. Let $(l(i), r(i))$ be the initial failure of O^i. Since any initial failure $(l(i), r(i))$ is a failure of O^j, $j < i$, it is easy to see that $r(i) < r(i+1)$, for $i = 0, \ldots, q-2$. Moreover, $A(e_{l(i)}O^i) \subseteq A(e_{l(i)}O)$, for $i = 0, \ldots, q-1$.

Since, for $i = 0, \ldots, q-1$, the vertex u^i is not exhausted in O^{i+1}, we only need to prove that for $i = 0, \ldots, q-1$, every vertex v in $t(e_{r(i)+1}O^{i+1})$ is exhausted in O^{i+1}.

We prove this latter statement by induction on i. The case $i = 0$ is immediate from Lemma 5. Let $i \geq 1$ and let v be a vertex in $t(e_{r(i)+1}O^{i+1})$. Since $(l(i), r(i))$ is a failure of O and $A(e_{l(i)}O^i) \subseteq A(e_{l(i)}O)$, from Lemma 5, $v \notin t(e_{l(i)}Oe_{r(i)})$, hence $v \notin t(e_{l(i)}O^i e_{r(i)})$. ¿From the induction hypothesis, we know that v is exhausted in O^i. Therefore, v is exhausted in O^{i+1}. □

Lemma 7. *I^i is the alphabetical trail starting at u^i over $G \setminus A(O^{i+1})$.*

Proof. Let j be the smallest integer such that $e_j \in I^i$ and there is k with $e_k \notin O^{i+1}$, $t(e_k) = t(e_j)$ and $l(e_k) < l(e_j)$. If such a j does not exist the conclusion is immediate. We shall prove that $e_k \in I^i$ and $k < j$.

Since $e_k \notin O^{i+1}$, then $e_k \in I^s$ for $s \leq i$. By Lemma 5, $j \leq r(s)$. Since $(l(s), r(s))$ is a failure of O and $l(s) \leq k < r(s)$, $j \neq r(s)$.

If $j < l(s)$ then (j, k) is a failure of O. As $j < l(s) \leq k < r(s)$, we get a contradiction with a consequence of Lemma 5. Hence, $l(s) \leq j < r(s)$ and therefore $e_j \in I^s$. Thus $s = i$ and $e_k \in I^i$. Since I^i has no failures in O^i, we conclude that $k < l$. □

Note that after a finite number of steps, we obtain a trail exhausting r with no failure vertex, which is an alphabetical trail starting at r over G.

Theorem 8. *Algorithm* MinLex *computes the Eulerian trail of lexicographically minimum label O in k steps, where k is the number of failure vertices of O. Moreover, it is computed in $\mathcal{O}(|A(G)|)$ time.*

Proof. The correctness of the algorithm comes from Lemmas 6 and 7. Since the algorithm visits each arc at most twice, it can be implemented in $\mathcal{O}(|A(G)|)$ using an adjacency list to represent the digraph.

MinLex(A,r) : Compute the lexicographically minimum Eulerian circuit starting at r on $(V(G), A)$

Require: A an arc-subset of $A(G)$, r a vertex of G.
(1) $T \leftarrow$ AlphabeticTrail(A, r)
(2) **while** $NotEx(T) \neq \emptyset$
(3) $v \leftarrow LastNotEx(T)$
(4) $e_i \leftarrow$ the arc in T after the last visit to v
(5) $T \leftarrow (Te_{i-1})(\text{MinLex}(A \setminus A(T), v))(e_i T)$
(6) **end while**
Return: T

Where AlphabeticTrail(A, r) returns the alphabetical trail starting at r over $(V(G), A)$ and $NotEx(T)$ is the set of not exhausted vertices in T.

3. Minimum Eulerian circuit in general digraphs: An NP-complete problem

A locally injective labelling is a crucial property for the complexity of finding the Eulerian trail of minimum label. In this section, we prove that this problem is NP-complete for a general labelling. In order to prove that, we recall the following decision problem:

MIN-LEX-Eulerian-Circuit
Instance: An Eulerian digraph G, a labelling $l : A(G) \to \Sigma$ of its arcs, a starting vertex r and a word $X \in \Sigma^{|A(G)|}$.
Question: Is there an Eulerian circuit T starting at r such that $l(T) \leq X$?

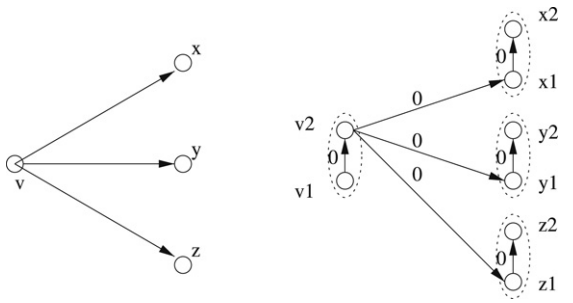

Fig. 2. Transformation of a digraph G (DIRECTED-HAMILTONIAN-CIRCUIT instance) into a labelled digraph H (MIN-LEX-EULERIAN-CIRCUIT instance).

Theorem 1. MIN-LEX-EULERIAN-CIRCUIT *is NP-complete.*

Proof. It is easy to see that one can guess an Eulerian circuit and compare its labels with those of X in polynomial time. Hence, the problem belongs to NP.

We present a reduction of a DIRECTED-HAMILTONIAN-CIRCUIT instance (see [7]) into a MIN-LEX-EULERIAN-CIRCUIT instance, which is polynomially bounded in the size of the input digraph.

Let G be a digraph. We want to test if G contains a directed Hamiltonian circuit. We construct a digraph H in the following way: for each vertex $v \in G$, we include two vertices v_1 and v_2 and an arc (v_1, v_2) in H. Additionally, for each arc $(v, w) \in G$ we include the arc (v_2, w_1) in H (see Fig. 2). Finally, we label all arcs in H with the label 0.

It is easy to see that G has a Hamiltonian circuit if and only if H has a circuit with label $0^{2|V(G)|}$.

We can extend the digraph H to an Eulerian digraph $\bar{H}$ with additional vertices and arcs with label 1 in the following way: we add a vertex c and we connect every vertex v in H to c with two arcs (v, c) and (c, v) of label 1. With these connections, the resulting digraph is strongly connected even if we remove all arcs in H. Finally, to each arc (x, y) in H we add an arc (y, x) with label 1. These arcs provide the equality between the in-degree and the out-degree of each vertex in $\bar{H}$. Hence, the resulting digraph is Eulerian. Moreover, if G has a Hamiltonian circuit then we can remove arcs of its associated circuit of label $0^{2|V(G)|}$ in $\bar{H}$ and the remaining digraph is still Eulerian.

Therefore, G has a Hamiltonian circuit if and only if $\bar{H}$ has an Eulerian circuit starting at any vertex $r \in H$ with label smaller or equal to $0^{2|V(G)|}1^{|A(\bar{H})|-2|V(G)|}$. □

4. Conclusion

We first note that by interchanging the role of minimum and maximum all our results apply in the study of the lexicographically maximum Eulerian circuit problem.

What are the capabilities/limitations of MINLEX algorithm? On one hand, one can construct examples of a labeled Eulerian digraph where the MINLEX algorithm does not obtain a lexicographically minimum Eulerian circuit. On the other hand, it is easy to construct examples where MINLEX does obtain the lexicographically minimum Eulerian circuit, even if the labelling is not locally injective. Hence, a natural question is to characterize those labellings where the MINLEX algorithm obtains a lexicographically minimum Eulerian circuit.

Acknowledgements

We would like to thank referees for their constructive comments. Their specific comments helped us to obtain the characterization given in Corollary 4 and to improve the overall structure of the paper.

References

[1] J.-C. Bermond, R.W. Dawes, F.Ö Ergincan, De Bruijn and Kautz bus networks, Networks 30 (3) (1997) 205–218.
[2] F. Chung, P. Diaconis, R. Graham, Universal cycles for combinatorial structures, Discrete Math. 110 (1–3) (1992) 43–59.
[3] N.G. de Bruijn, A combinatorial problem, Nederl. Akad. Wetensch., Proc. 49 (1946) 758–764.
[4] C. Flye Sainte-Marie, Question 48, L'Intermédiaire Math. 1 (1894) 107–110.
[5] H. Fredricksen, A survey of full length nonlinear shift register cycle algorithms, SIAM Rev. 24 (2) (1982) 195–221.

[6] H. Fredricksen, J. Maiorana, Necklaces of beads in k colors and k-ary de Bruijn sequences, Discrete Math. 23 (1978) 207–210.
[7] M.R. Garey, D.S. Johnson, Computers and Intractability, a Guide to the Theory of NP-completeness, in: A Series of Books in the Mathematical Sciences, W.H. Freeman and Co., San Francisco, California, 1979.
[8] I.J. Good, Normal recurring decimals, J. London Math. Soc. 21 (1946) 167–169.
[9] E. Moreno, De Bruijn sequences and de Bruijn graphs for a general language, Inform. Process. Lett. 96 (6) (2005) 214–219.
[10] E. Moreno, M. Matamala, Minimal de Bruijn sequence in a language with forbidden subtrings, Lect. Notes Comput. Sci. 3353 (2004) A complete and revised version of this article is published at arXiv:0711.1695.
[11] E. Moreno, M. Matamala, Minimal eulerian circuit in a labeled digraph, Lect. Notes Comput. Sci. 3887 (2006) 737–744.
[12] G. Rauzy, Suites à termes dans un alphabet fini, in: Seminar on Number Theory, Univ. Bordeaux I (Talence 1982/1983), 1983, pp. Exp. No. 25, 16.
[13] F. Ruskey, C. Savage, T.M. Wang, Generating necklaces, J. Algorithms 13 (3) (1992) 414–430.
[14] F. Ruskey, J. Sawada, Generating necklaces and strings with forbidden substrings, Lect. Notes Comput. Sci. 1858 (2000) 330–339.
[15] S.K. Stein, The mathematician as an explorer, Sci. Amer. 204 (5) (1961) 148–158.
[16] W.T. Tutte, C.A.B. Smith, On Unicursal Paths in a Network of Degree 4, Amer. Math. Monthly 48 (4) (1941) 233–237.
[17] T. van Aardenne-Ehrenfest, N.G. de Bruijn, Circuits and trees in oriented linear graphs, Simon Stevin 28 (1951) 203–217.