available at www.sciencedirect.com

journal homepage: www.elsevier.com/locate/cosrev

Survey

Cycle bases in graphs characterization, algorithms, complexity, and applications

Telikepalli Kavitha[a], *Christian Liebchen*[b], *Kurt Mehlhorn*[c,∗], *Dimitrios Michail*[d], *Romeo Rizzi*[e], *Torsten Ueckerdt*[f], *Katharina A. Zweig*[g]

[a] *Department of Computer Science and Automation, Indian Institute of Science, Bangalore, India*
[b] *DB Schenker Rail Deutschland AG, Mainz, Germany*
[c] *Max-Planck-Institut für Informatik, Saarbrücken, Germany*
[d] *Department of Informatics and Telematics, Harokopio University of Athens 89, Harokopou Street 17671, Kallithea, Athens, Greece*
[e] *Dipartimento di Matematica ed Informatica (DIMI), Università degli Studi di Udine, Udine, Italy*
[f] *Technische Universität Berlin, Berlin, Germany*
[g] *Department of Biological Physics, Eötvös Lorand University, Budapest, Hungary*

ARTICLE INFO

Article history:
Received 3 April 2009
Received in revised form
25 August 2009
Accepted 25 August 2009

ABSTRACT

Cycles in graphs play an important role in many applications, e.g., analysis of electrical networks, analysis of chemical and biological pathways, periodic scheduling, and graph drawing. From a mathematical point of view, cycles in graphs have a rich structure. Cycle bases are a compact description of the set of all cycles of a graph. In this paper, we survey the state of knowledge on cycle bases and also derive some new results. We introduce different kinds of cycle bases, characterize them in terms of their cycle matrix, and prove structural results and a priori length bounds. We provide polynomial algorithms for the minimum cycle basis problem for some of the classes and prove $\mathcal{APX}$-hardness for others. We also discuss three applications and show that they require different kinds of cycle bases.

1. Introduction

Cycles in graphs play an important role in many applications, e.g., analysis of electrical networks, analysis of chemical and biological pathways, periodic scheduling, and graph drawing. From a mathematical point of view, cycles in graphs have a rich structure. Cycle bases are a compact description of the set of all cycles of a graph and cycle bases consisting of short cycles or, in weighted graphs, of small weight cycles are interesting both mathematically and from an application viewpoint. In the applications above, sparse descriptions are to be preferred.

The study of cycle bases dates back to the early days of graph theory; MacLane [1] gave a characterization of planar graphs in terms of cycle bases. Within the last ten years, many new results on cycle bases have been published, most notably a classification of different kinds of cycle bases, structural results, a priori bounds on the length and weight of minimum cycle bases, polynomial time algorithms for constructing exact or approximate minimum cycle bases for some kinds, and hardness results for other kinds of minimum cycle bases.

In this paper, we survey these results and also provide some new ones. Fig. 1 shows the landscape of cycle bases. We

∗ *Corresponding author.*
E-mail address: mehlhorn@mpi-inf.mpg.de (K. Mehlhorn).

1574-0137/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.cosrev.2009.08.001

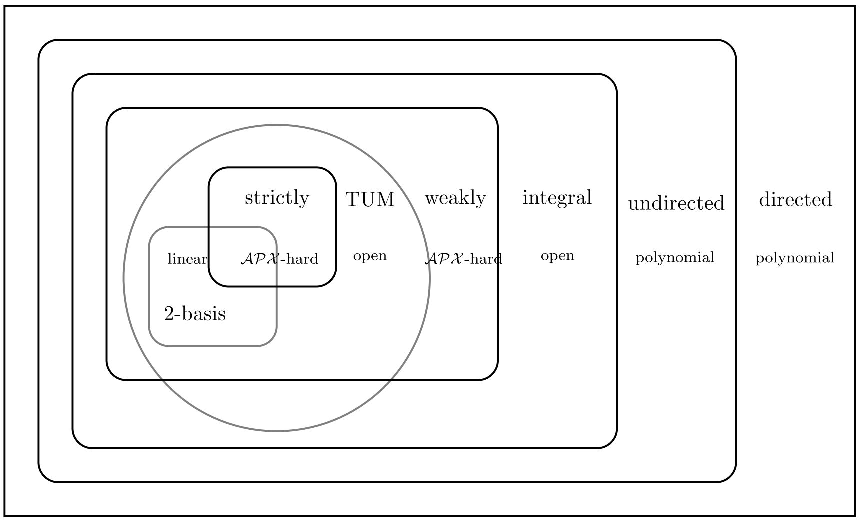


Fig. 1 – The inclusion diagram of cycle bases and the complexity status of their minimum weight cycle basis problems.

will review the different kinds of cycle bases in Sections 2 and 3: directed, undirected, integral, weakly fundamental, totally unimodular, and strictly fundamental bases, and 2-bases. In Section 3, we characterize the different kinds in terms of properties of their cycle matrices. For example, undirected cycle bases are characterized by the fact that the determinant of their cycle matrix is odd and integral cycle bases are characterized by the fact that their determinant is ±1. We will establish the inclusion map and show that different classes lead to different minimum cycle basis problems. We will also establish many structural results.

Section 4 deals with a priori length and weight bounds on minimum cycle bases. We will prove results of the following kind: every graph of n nodes and m edges has a weakly fundamental cycle basis of length $O(m \log m/\log(m/n))$. We will also show that there are graphs for which every basis has length $\Omega(m \log m/\log(m/n))$.

In Section 5, we will give polynomial time algorithms for constructing minimum weight directed, undirected and planar cycle bases. We will also discuss approximation algorithms.

Section 6 treats hardness results; in particular, $\mathcal{APX}$-hardness of the minimum cycle basis problem for weakly fundamental and strictly fundamental bases. Fig. 1 summarizes the complexity results. For two classes the complexity is open:

Open Problem 1. Resolve the complexity status of computing minimum weight integral and minimum weight totally unimodular bases.

Finally, Section 7 discusses three applications of cycle bases; we will see that they require different kinds of cycle bases. The analysis of electrical circuits does not require any particular kind of cycle basis, whereas periodic scheduling requires integral cycle bases, and graph drawing needs strictly fundamental bases.

The paper mostly surveys known results, but it also contains several new ones. In particular, we give additional structural and characterization results, we obtain tight length bounds for weakly fundamental cycle bases for the full spectrum of graph densities, we give a simplified algorithmic treatment of directed cycle bases, and we present the first algorithms for minimum cycle bases in the presence of negative edges. In each section, we also state open problems.

This survey is targeted at mathematicians and computer scientists. We give complete proofs for most results to make the survey self-contained. We wrote the survey because this area has developed quickly in the past decade and is still a rich source of open problems.

2. Definitions

An *(undirected) graph* is a pair $G = (V, E)$, where V is a finite set, and E is a family of unordered pairs of elements of V. The elements of V are called *vertices* or *nodes* and the elements of E are called *edges*. An edge $e = \{v, w\}$ is *incident to* the vertices v and w; v and w are the endpoints of e. The same pair $\{v, w\}$ may occur several times in E; we refer to a pair occurring more than once as a *multiple edge*. Graphs without multiple edges are called *simple*. An edge of the form $\{v, v\}$ is called a *loop*. The *degree* $deg(v)$ of a vertex v is the number of times v occurs as an endpoint of an edge. Observe that a loop $\{v, v\}$ contributes two to the degree of v. We use $\delta(v)$ to denote the set of edges incident to v; a loop $\{v, v\}$ appears twice in $\delta(v)$.

A *(directed) graph* is a pair $D = G = (V, A)$, where V is a finite set, and A is a family of ordered pairs of elements of V. The elements of V are called the vertices or nodes of G, and the elements of A are called the *(directed) edges* or *arcs* of G. We use $G = (V, E)$ to denote directed and undirected graphs and $D = (V, A)$ to denote directed graphs. The vertices v and w are called the *tail* and *head* of the arc $e = (v, w)$, respectively; e is said to leave v and to enter w; it is incident to v and w. The notions *multiple edge*, *simple graph*, and *loop* are defined analogously as for undirected graphs. The *outdegree* $outdeg(v)$ and *indegree* $indeg(v)$ of a vertex v are the number of times v occurs as the tail and head, respectively, of an edge. Observe that a loop (v, v) contributes one to both the indegree and the outdegree of v. We use $\delta^+(v)$ and $\delta^-(v)$ for the edges leaving and entering v, respectively.

We use n and m to denote the number of nodes and edges or arcs, respectively, i.e., $n = |V|$ and $m = |E|$ or $m = |A|$. We use the notation $e = vw$ to denote both directed and undirected edges, i.e., the notation stands for the directed edge (v, w) and the undirected edge $\{v, w\}$. Every directed graph D can be turned into an undirected graph $G(D)$ by ignoring the orientation of the edges and every undirected graph G can be turned into a directed graph by orienting the edges arbitrarily; we call D an *orientation* of G. In this way, we can view every graph as directed.

A *subgraph* $G' = (V', E')$ of G is a graph with $V' \subseteq V$ and $E' \subseteq E$. If V' is a subset of V, $G - V'$ denotes the graph obtained by removing all vertices in V' and their incident edges from G. A *path* P from v to w in G is a subgraph of G with $V' = \{v_0 = v, v_1, \ldots, v_k = w\}$ with $v_i \neq v_j$ and $E' = \{x_0x_1, x_1x_2, \ldots, x_{k-1}x_k\}$. We write $P(v, w)$ if we want to emphasize that P is a path from v to w. The *length of a path* is the number of its edges. An undirected graph is *connected* if there exists a path from any vertex to every other vertex. A vertex v in a connected graph G is called an *articulation point*, or *cut vertex*, if $G - v$ is disconnected. An undirected graph is *biconnected* if it has no articulation point. A directed graph is connected if the underlying undirected graph is connected. Any maximal connected subgraph of G is called a *connected component*. A graph T is a *tree* if it is connected and has $n - 1$ edges. A subgraph G' of a connected graph G is called a *spanning tree* if it constitutes a tree on all vertices in G. If G is not connected, any union of spanning trees for each connected component is called a *spanning forest*.

A *cycle in an undirected graph* is a subgraph in which every vertex has even degree. A cycle is a *circuit* if it is connected and every one of its vertices has degree two. If $C_1, \ldots, C_k$ are cycles, $C_1 + \cdots + C_k$ consists of all edges that are contained in an odd number of C_i's; the sum is again a cycle. An *undirected cycle basis* is a minimal set of circuits such that any cycle can be written as a sum of the circuits in the basis.

We next generalize the notion of an undirected cycle basis. Let κ be a field. A *κ-cycle* C in a directed graph D is a vector in κ^A such that for any vertex v we have

$$\sum_{e \in \delta^+(v)} C_e = \sum_{e \in \delta^-(v)} C_e;$$

here C_e denotes the component of C indexed by e. Instead of C_e, we will also sometimes write $C(e)$. We prefer the latter notation when $C = C_i$ belongs to an indexed family of cycles. In other contexts, cycles are sometimes referred to as *circulations* and the constraint $\sum_{e \in \delta^+(v)} C_e = \sum_{e \in \delta^-(v)} C_e$ is called *flow conservation*. The set

$$\mathcal{C}_\kappa(D) = \{C \mid C \text{ is a } \kappa\text{-cycle of } G\}$$

forms a vector space over κ, the *κ-cycle space* of G; if C_1 and C_2 are cycles and $\lambda \in \kappa$ is a constant, we have

$$(C_1 + C_2)(e) = C_1(e) + C_2(e) \quad \text{and} \quad (\lambda C)(e) = \lambda C(e)$$

for all edges e. The support of a cycle is the set of edges e with $C_e \neq 0$. A cycle is *simple* if $C_e \in \{-1, 0, +1\}$ for all e, and a simple cycle is a *circuit* if its support is connected and for any v there are at most two edges in the support incident to v. A *κ-cycle basis* is a set of circuits forming a basis of the cycle space. Any cycle basis consists of $\nu := m - n + 1$ circuits (see Theorem 2.3). If D and D' are orientations of the same undirected graph G, their cycle spaces $\mathcal{C}_\kappa(D)$ and $\mathcal{C}_\kappa(D')$ are isomorphic. Indeed, if $C \in \kappa^A$ is a cycle in D, the corresponding cycle in D' is obtained by reversing the sign of those components C_e, where e is oriented differently in D and D'. We conclude that the vector space $\mathcal{C}_\kappa(D)$ does not depend on the orientation D; it is uniquely defined by the underlying undirected graph G. Hence, we may also write $\mathcal{C}_\kappa(G)$.

Particularly interesting are the cases $\kappa = \mathbb{Z}_2 = GF(2)$, the field of two elements, and $\kappa = \mathbb{Q}$, the field of rationals. In these cases, the cycle space and cycle basis are referred to as *undirected* or *directed cycle space* and *basis*, respectively.

In $\mathbb{Z}_2$, $-1 = +1$ and $+1$ is the only non-zero element in the field. Thus a $\mathbb{Z}_2$-cycle or simply, a cycle, is a vector $C \in \mathbb{Z}_2^E$ such that $\sum_{e \in \delta(v)} C_e = 0$ for any vertex v. A cycle may alternatively be viewed as a set of edges; e belongs to C iff $C_e = 1$. We use C to denote the vector in $\mathbb{Z}_2^E$, the corresponding subset of E, and also the subgraph (V', C), where V' is the set of vertices having at least one edge in E incident to it. A cycle is an *even* or *Eulerian* subgraph, i.e., every vertex has even degree in C. Conversely, any even subgraph is a cycle.

A $\mathbb{Q}$-cycle C has components in $\mathbb{Q}$; we call it a *directed cycle* if all components of C are integral. Directed cycles may use arcs in *forward* ($C_e > 0$) or *backward* ($C_e < 0$) direction. If any arc is replaced by C_e copies of itself and, in addition, the direction of all arcs e with $C_e < 0$ is reversed, then we end up with a digraph in which the indegree of every vertex is equal to its outdegree.

Let D be a directed graph and let $G = G(D)$ be the underlying undirected graph. For any directed cycle C of D, let $\pi(C) := (C_e \bmod 2)_{e \in E}$. Then $\pi(C)$ is an undirected cycle in G. We call $\pi(C)$ the *projection* of C.

Fig. 2 illustrates these definitions. In addition, it provides a first example showing that directed cycle bases do not necessarily project into undirected cycle bases. However, a set of dependent cycles projects into a set of dependent cycles. Indeed, let C_i, $i \in I$, be a family of dependent directed cycles. Then $\sum_{i \in I} \lambda_i C_i = \mathbf{0}$, with $\lambda_i \in \mathbb{Q}$ not all zero. Here $\mathbf{0}$ denotes the zero-vector in $\mathbb{Q}^E$. We may assume $\lambda_i \in \mathbb{Z}$ not all even. Then $\sum_{i \in I} (\lambda_i \bmod 2)\ \pi(C_i) = \mathbf{0} \bmod 2$ and at least one coefficient $\lambda_i \bmod 2$ is non-zero. Thus the $\pi(C_i)$, $i \in I$, are dependent.

We use $+$ and Σ to denote addition in $\mathbb{Q}$ and in $GF(2)$ (and also in $GF(p)$ for prime p). The distinction will usually be clear from the context. If both fields occur in the same argument, as in the paragraph above, we will emphasize the difference by the additional operator " mod 2".

We may also lift undirected cycles from an undirected graph G to an orientation D of G. Let C' be any undirected cycle in G. We call $C \in \{-1, 0, +1\}^A$ a lifting of C' if C projects to C'. For a circuit C' the lifting is unique up to the sign. Clearly, if C' lifts to C then C projects to C'. Algorithmically, we may lift as follows: assume C' to be connected (components are lifted independently) and consisting of k edges. Since an undirected cycle is a Eulerian subgraph of G, there is a closed traversal $(e_0, \ldots, e_{k-1})$ of the edges of C', i.e., $e_i = \{v_i, v_{i+1}\}$ for $0 \le i < k$ and $v_0 = v_k$. This traversal defines a simple cycle C in D; we have $C_e = 0$ if C' does not contain e and $C_e = +1$ (-1) if the traversal uses e in forward (backward) direction.

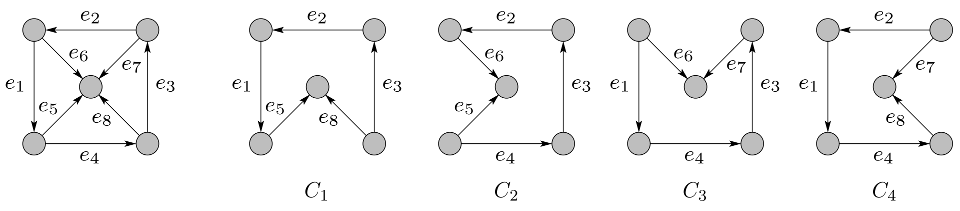


Fig. 2 – An orientation D of the undirected wheel graph W_5, and four circuits C_1 to C_4 in D. The edges of D are numbered from e_1 to e_8. The circuit C_1 uses the edges e_1, e_2, e_3, and e_5 in forward direction and the edge e_8 in backward direction. Thus $C_1 = (1, 1, 1, 0, 1, 0, 0, -1)$. The cycles C_1 to C_4 form a directed cycle basis of D. The cycle C consisting of edges 1 to 4 is represented as: $C = (1, 1, 1, 1, 0, 0, 0, 0) = (C_1 + C_2 + C_3 + C_4)/3$. Let G be the underlying undirected graph, let $\pi(C_i)$ be the undirected cycle corresponding to C_i, and let $\pi(C)$ be the undirected cycle corresponding to C. Then $\pi(C_1) = (1, 1, 1, 0, 1, 0, 0, 1)$ and $\pi(C) = \pi(C_1) \oplus \pi(C_2) \oplus \pi(C_3) \oplus \pi(C_4)$. The circuits $\pi(C_1)$ to $\pi(C_4)$ form an undirected cycle basis of G. The set $\{C_1, C_2, C_3, 2C_4\}$ is also a directed cycle basis of D. However, $\pi(2C_4) = 0$ and hence $\{\pi(C_1), \pi(C_2), \pi(C_3), \pi(2C_4)\}$ is *not* an undirected cycle basis of G. There are less trivial reasons for a directed cycle basis not projecting into an undirected cycle basis.

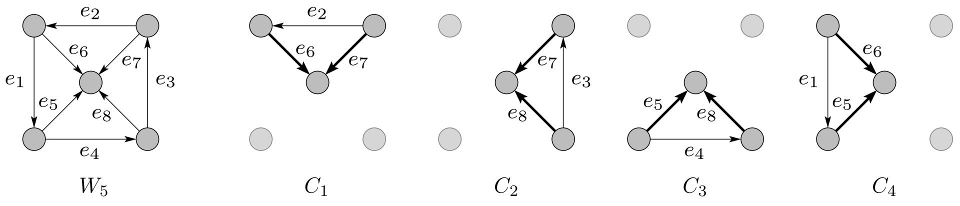


Fig. 3 – An orientation D of the undirected wheel graph W_5 and four circuits C_1 to C_4 in D. The edges are numbered from e_1 to e_8. The edges $\{e_5, e_6, e_7, e_8\}$ form a spanning tree T of D. Circuit C_1 is induced by non-tree edge e_2 and uses edges e_2 and e_6 in forward direction and edge e_7 in backward direction. Thus $C_1 = (0, 1, 0, 0, 0, 1, -1, 0)$. Cycles C_2, C_3, and C_4 are obtained in an analogous way. The set $\{C_1, C_2, C_3, C_4\}$ is a strictly fundamental cycle basis of D.

A *weighted graph* is a graph together with a weight function $w : E \to \mathbb{R}$. If the graph is unweighted, we set $w : E \to 1$ and call w the *uniform weight function*. The weight of a set of edges is the sum of the weights of its members. The *weight and length of a simple cycle* C are

$$w(C) := \sum_e |C_e| w(e) \quad \text{and} \quad |C| := \sum_e |C_e|, \quad \text{respectively.}$$

In an unweighted graph, weight and length are identical. The *weight of a cycle basis* B is the sum of the weights of its cycles, i.e.,

$$w(B) = \sum_{C \in B} w(C).$$

A *minimal κ-cycle basis*, or *MCB*, of G is a κ-cycle basis with minimal weight. We assume that there are no simple cycles of negative weight; such weight functions are called *conservative*. For most of our algorithms, we need to assume that weights are non-negative, i.e., $w : E \to \mathbb{R}^+$.

We close this section with a first theorem. Every graph has a κ-cycle basis and the dimension of the κ-cycle space is given by the graph's *cyclomatic number* $\nu := m - n + CC$, where CC denotes the number of connected components of G. On the way, we get to know a particularly simple set of cycles, the fundamental cycles with respect to a spanning forest. Let G be an (undirected or directed) graph and let T be a spanning forest of G. For any non-tree edge e, let C_T^e be the circuit consisting of e and the tree path connecting the endpoints of e. In the case of a directed graph, we use e in forward direction and traverse the tree path from the head of e to the tail of e; Fig. 3. We call C_T^e the *fundamental circuit* defined by T and e.

Lemma 2.1. *Let G be a graph and let T be any spanning forest of G. Let C be a cycle that uses only edges in T, i.e., $C_e = 0$ for $e \notin T$. Then $C = \mathbf{0}$.*

If C and C' are cycles with $C_e = C'_e$ for all $e \notin T$, then $C = C'$.

Proof. The support of C is contained in T. If the support were non-empty, there would have to be a vertex v having exactly one incident edge with $C_e \neq 0$. This is clearly impossible and hence the support must be empty.

Assume next that C and C' are cycles with $C_e = C'_e$ for all $e \notin T$. Then $C - C'$ is a cycle with $(C - C')_e = 0$ for all $e \notin T$. Thus $C - C' = \mathbf{0}$. □

Lemma 2.2. *Let B be a set of cycles in G and let T be any spanning forest of G. For any cycle $C \in B$, let C' be its restriction to $N := E \setminus T$. The cycles are linearly independent if and only if their restrictions to N are linearly independent.*

Proof. Clearly, linear dependence of the cycles implies linear dependence of their restrictions. Conversely, assume that there is a non-trivial linear combination of the restrictions that yields the zero vector, i.e., $\sum_{C \in B} \lambda_C C' = \mathbf{0}_N$. Here $\mathbf{0}_N$ denotes the zero vector over index set N. Then $\sum_{C \in B} \lambda_C C$ is a cycle that uses only tree edges and hence is equal to $\mathbf{0}$. □

Thus, we may restrict attention to the restricted incidence vectors when discussing questions of linear independence.

Theorem 2.3 (*Dimension of the Cycle Space of a Graph*). *The dimension of the κ-cycle space of a graph G is given by its* cyclomatic number

$$\nu = m - n + CC,$$

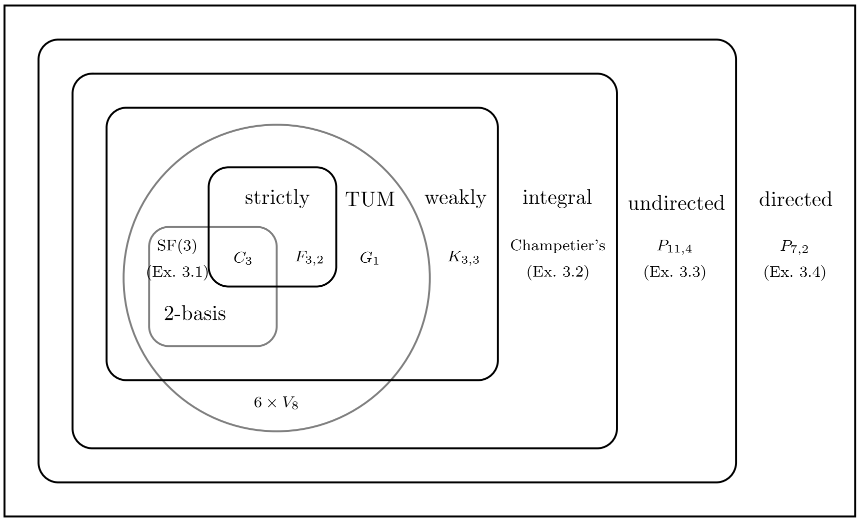


Fig. 4 – The Venn diagram of directed cycle bases: Ex. 3. X refers to examples that are discussed in detail later in this section, $K_{3,3}$ refers to a weighted version of the complete bipartite graph on 3 × 3 vertices, $P_{7,2}$ is a weighted version of a generalized Petersen graph, V_8 is Wagner's graph (see Section 6), $F_{3,2}$ is a fan graph on five vertices, and G_1 is a simple graph on eight vertices; see [2].

where CC denotes the number of connected components of G. Moreover, if T is any spanning forest of G, the set of fundamental circuits with respect to T forms a basis.

Proof. The number of fundamental circuits is equal to ν, because a connected component with m' edges and n' vertices contributes $m' - (n' - 1)$ fundamental cycles. Let N be the set of non-tree edges. The fundamental cycles are clearly independent since any edge $e \in N$ is contained in C_T^e and in no other circuit. It remains to prove that the set of fundamental circuits spans all cycles. Let C be an arbitrary cycle. Consider the cycle

$$\tilde{C} := \sum_{e \in N} C_e C_T^e.$$

We claim that $C = \tilde{C}$. Indeed, for any $e \in N$, we have $\tilde{C}_e = C_e$ and hence $C - \tilde{C}$ is a cycle using only edges of T. Thus $C - \tilde{C} = \mathbf{0}$. □

The following lemma is a first step towards clarifying the relation between directed and undirected cycle bases.

Lemma 2.4. *Let D be a directed graph, let $B = \{C_1, \ldots, C_\nu\}$ be a set of circuits in D, let G be the underlying undirected graph, and let $\pi(B) = \{\pi(C_1), \ldots, \pi(C_\nu)\}$. If $\pi(B)$ is an undirected cycle basis of G then B is a directed cycle basis of D.*

Proof. We have already shown that a set of dependent cycles projects into a set of dependent cycles. Hence $\pi(B)$, being an undirected cycle basis, implies that the cycles in B are independent. Also, ν must be equal to the cyclomatic number of D since $\pi(B)$ is a basis. □

3. Classification of cycle bases

We present seven classes of cycle bases and provide characterizations for them. We will show that each class gives rise to its own minimum cycle basis problem. The complexity of the minimum cycle basis problem differs widely. For three classes the problem is polynomial time, for two classes the problem is NP-complete, and for two classes the status is unknown. This section is mainly based on [2]; the missing proofs can be found there.

Definition 3.1 (*Classes of Cycle Bases*). A directed cycle basis (D-basis) $B = \{C_1, \ldots, C_\nu\}$ of a graph D is called *a(n)*:

1. *undirected* or U-*basis*, if the projections $\pi(C_i)$ of the basic circuits C_i onto the underlying undirected graph $G(D)$ constitute a cycle basis of $G(D)$;
2. *integral* or I-*basis*, if each cycle C of D can be written as an *integer* linear combination of circuits in B, i.e.

   $$\exists \lambda_i \in \mathbb{Z} : C = \lambda_1 C_1 + \cdots + \lambda_\nu C_\nu;$$
3. *zero-one* or[1] TUM-*basis*, if each cycle C' of $G(D)$ has an orientation C that can be written as a linear combination with coefficients in $\{-1, 0, +1\}$ of circuits in B, i.e.

   $$\exists \lambda_i \in \{-1, 0, +1\} : \gamma_C = \lambda_1 C_1 + \cdots + \lambda_\nu C_\nu;$$
4. *weakly fundamental* or W-*basis*, if there exists some permutation σ such that

   $$C_{\sigma(i)} \setminus (C_{\sigma(1)} \cup \cdots \cup C_{\sigma(i-1)}) \neq \emptyset, \quad \forall i = 2, \ldots, \nu;$$
5. *strictly fundamental* or F-*basis*, if there exists some spanning forest $T \subseteq E$ such that $B = \{C_T(e) \mid e \in E \setminus T\}$, where $C_T(e)$ denotes the unique circuit in $T \cup \{e\}$; and
6. *planar*, or *2-basis*, if each arc is contained in at most two basic circuits and the basis is undirected.

Fig. 4 depicts the relationship between these classes: The inclusions are established in Theorem 3.4, and examples for the non-emptiness of the regions will be provided in Section 3.4.

[1] It will become clear in Theorem 3.4 why zero-one bases are called totally unimodular (TUM).

3.1. *Existence*

Except for 2-bases, every graph has a basis of each type. This follows from the fact that every graph has a strictly fundamental cycle basis and that all other classes generalize fundamental cycle bases. In contrast, MacLane [1] established that a graph has a 2-basis if and only if it is planar.

3.2. *Characterizations*

We define the cycle matrix corresponding to a basis and show that the different classes of cycle bases can be characterized in terms of simple properties of this matrix. An important property is the determinant of the cycle basis. The *cycle matrix* corresponding to a D-basis B of D is an $m \times \nu$ matrix whose columns are the incidence vectors of the basic circuits. The cycle matrix is determined up to a permutation of the rows and columns.

The cycle matrix Φ of a fundamental basis has a particularly simple form. Let T be a spanning forest and let N be the set of non-tree arcs. Then, for a suitable permutation of the columns, the $\nu \times \nu$ submatrix Φ' selected by the rows corresponding to non-tree arcs is the identity matrix.

Lemma 3.1 ([3]). *Let B be a directed cycle basis of a directed graph G and let Γ be the corresponding cycle matrix. A $\nu \times \nu$ submatrix Γ' of Γ is nonsingular if and only if the rows of Γ' correspond to the non-tree arcs of some spanning forest of G.*

Proof. To prove sufficiency, consider a spanning forest T of D, and let Φ be the cycle matrix of the fundamental basis with respect to T. BecauseB is a directed cycle basis, any fundamental cycle is a linear combination of cycles in B. Thus there is a matrix $R \in \mathbb{Q}^{\nu\times\nu}$ with $\Phi = \Gamma R$. The restriction of Φ to the non-tree arcs of T is the identity matrix. Hence, R is the inverse of Γ'.

Conversely, assume that the rows which are *not* in Γ' do not form a spanning forest. Then there is a circuit C consisting only of such arcs. As B is a D-basis, we have $C = \Gamma x_C$ for some x_C and clearly $x_C \neq \mathbf{0}$. Restriction to the rows indexing Γ' yields $\mathbf{0} = \Gamma' x_C$, and hence Γ' is singular. □

Lemma 3.2 ([3]). *Let B be a D-basis, let Γ be its cycle matrix, and let A_1 and A_2 be two nonsingular $\nu \times \nu$ submatrices of Γ. Then* $\det A_1 = \pm \det A_2$.

Proof. By Lemma 3.1, the rows of A_i correspond to the non-tree arcs of some spanning forest T_i. It suffices to prove the claim for the case where $T_2 = T_1 + e - f$, for some edges e and f. Let Φ be the cycle matrix of the fundamental basis with respect to T_1. Then $\Gamma = \Phi N$ for some matrix N. Let Φ_i be the submatrix of Φ selected by the non-tree arcs of T_i. Then $\det A_i = \det \Phi_i \cdot \det N$ and therefore it suffices to prove $\det \Phi_2 = \pm \det \Phi_1$. We have $\Phi_1 = I$ and hence $\det \Phi_1 = 1$. Also, since e must lie on the path in T_1 connecting the endpoints of f (otherwise, T_2 would not be a spanning tree), the entry of Φ in row e and column C_f is either $+1$ or -1. Developing $\det \Phi_2$ according to column C_f shows $\det \Phi_2 = \pm 1$. □

The above lemma allows us to define the determinant of a directed cycle basis.

Definition 3.2 (*Determinant of a Set of ν Oriented Circuits*). Let B denote a set of ν circuits in a directed graph D. Consider the matrix Γ with the incidence vectors of B as columns. Let Γ' be the $\nu \times \nu$ submatrix of Γ that arises when deleting the arcs of some spanning forest of D. We define

$$\det B := |\det \Gamma'|.$$

Determinants of directed cycle bases are positive integers. The value of the determinant is invariant under reorienting arcs of D or reorienting circuits of B, because this simply translates to multiplying a row or column by minus one. Thus, starting with a cycle basis of an undirected graph G, orienting the edges of G arbitrarily, and choosing one of the two orientations for each circuit, always results in the same determinant.

How large can the determinant of a cycle basis be? Hadamard's bound gives an upper bound of $\sqrt{n}^{\nu}$, since we are dealing with the determinant of a $\nu \times \nu$ matrix with entries in $\{-1, 0, +1\}$ in which every column has at most n non-zero entries.

Lemma 3.3 ([4]). *Consider the generalized Petersen graph[2] $P_{n,2}$ with $n \geq 5$ and n odd. Let $\mathcal{C}$ denote the set of circuits, each of which contains exactly one inner edge, $n-2$ outer edges and two spokes. $\mathcal{C}$, together with the inner circuit C_I, forms a cycle basis of $P_{n,2}$ and its determinant equals $n-2$.*

Proof. $P_{n,2}$ consists of $2n$ vertices and $3n$ edges. Therefore every cycle basis has to consist of $n+1$ cycles, which is indeed the number of considered circuits. Additionally, it should be mentioned that the inner circuit C_I is indeed a simple cycle since n is odd.

Now let T be a spanning tree of $P_{n,2}$ made up of all but one inner edge and all spokes. Consider the square submatrix Γ' of the cycle matrix Γ obtained by deleting the rows corresponding to T. The non-tree edges and the circuits in $\mathcal{C} \cup \{C_I\}$ can be oriented and permuted such that

$$\Gamma' = \left(\begin{array}{cccccc|c}
1 & \cdots & \cdots & 1 & 0 & 0 & 0 \\
0 & 1 & \ddots & \ddots & 1 & 0 & \\
0 & 0 & 1 & \ddots & \ddots & 1 & \vdots \\
1 & 0 & 0 & 1 & \ddots & \vdots & \\
\vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \\
1 & \cdots & 1 & 0 & 0 & 1 & 0 \\
\hline
* & & \cdots & & & * & 1
\end{array}\right)$$

where the last column and the last row correspond to the inner circuit and the inner edge, respectively. The determinant of Γ' equals the determinant of its $n \times n$ submatrix obtained by deleting the last row and column. The resulting matrix is a *circulant matrix* whose first row has $n-2$ consecutive ones

[2] The generalized Petersen graph $P_{k,\ell}$ consists of $2k$ vertices $\{I_i, O_i \mid 0 \leq i \leq k-1\}$ and edges $\{O_iO_{i+1}, O_iI_i, I_iI_{i+\ell} \mid 0 \leq i \leq k-1\}$. All indices are modulo k. The edges O_iO_{i+1} are called *outer edges*, the edges $I_iI_{i+\ell}$ are called *inner edges*, and the edges O_iI_i are called *spokes*.

followed by two zeros. The entries of every other row result from the row above by a circular shift to the right. We have

$\det \Gamma' = n - 2;$

see [4] for the calculation of the determinant. □

Open Problem 2. Provide better upper and/or lower bounds on the maximal determinant of cycle bases.

Theorem 3.4 ([2]). *Let B be a directed cycle basis with cycle matrix Γ. Then:*

1. *B is undirected, if and only if $\det B$ is odd.*
2. *B is integral, if and only if $\det B$ is one.*
3. *B is zero-one if and only if Γ is totally unimodular.*[3]
4. *B is weakly fundamental, if and only if Γ can be permuted so as to have a regular upper triangular $\nu \times \nu$ matrix in its last ν rows.*
5. *B is strictly fundamental, if and only if Γ can be permuted so as to have the $\nu \times \nu$ unit matrix in its last ν rows.*
6. *B is a 2-basis, if and only if B is an undirected cycle basis and Γ has at most two non-zero entries per row.*

Proof. *Case* 1. The projections $\pi(C_i)$ of the basic circuits are linearly independent if $\pi(\Gamma)$ has full rank, i.e., if there is a square submatrix $\pi(\Gamma')$ with non-zero determinant over $GF(2)$. The value of the determinant is $(\det \Gamma') \bmod 2$. We conclude that B is undirected if and only if $\det B$ is odd.

Case 2. Let T be some spanning forest, and let Γ' be the square submatrix of Γ indexed by the non-tree arcs of T.

Let Φ be the cycle matrix of the fundamental basis with respect to T. Since B is integral, there is an integral $\nu \times \nu$ matrix R such that $\Phi = \Gamma R$. Restriction to the non-tree arcs of T yields $I = \Gamma' R$. We have $\det \Gamma' \in \mathbb{Z}$ and $\det R \in \mathbb{Z}$, because both matrices are integral. Thus $(\det \Gamma') \cdot (\det R) = 1$ implies $\det \Gamma' = \pm 1$.

Let C be an arbitrary circuit. The representation x_C of C in terms of B satisfies $C = \Gamma x_C$. Restriction to the non-tree arcs of T yields $C' = \Gamma' x_C$ or $x_C = (\Gamma')^{-1} C$. The inverse of Γ' is integral, by Cramer's rule, and since $\det \Gamma' = \pm \det B = \pm 1$. Thus $x_C \in \mathbb{Z}^\nu$.

Case 3. A matrix is totally unimodular if and only if for any subset I of its columns there are coefficients $\lambda_i \in \{-1, +1\}$ such that $\sum_{i \in I} \lambda_i C_i$ is a vector with entries in $\{-1, 0, +1\}$, see [5, Theorem 19.3].

Assume first that B is a zero-one basis. Since zero-one bases are integral, B is an integral cycle basis and hence $\pi(B) = \{\pi(C_i) \mid C_i \in B\}$ is an undirected basis of $G(D)$. Let I be an arbitrary subset of the columns of Γ and consider the $\mathbb{Z}_2$-sum of the projections of the circuits in I, and call the resulting cycle C',

$$\sum_{i \in I} \pi(C_i) = C'.$$

Since B is a zero-one basis, C' has an orientation C that can be written as a linear combination with coefficients $\lambda_i \in \{-1, 0, +1\}$ of the circuits in B, i.e.,

$$\sum_{i=1}^{\nu} \lambda_i C_i = C.$$

Projecting this equation into $\mathbb{Z}_2$, we obtain

$$\sum_{i=1}^{\nu} |\lambda_i| \pi(C_i) = C'.$$

Since the representation of C' with respect to $\pi(B)$ is unique, λ_i is non-zero if and only if $i \in I$. Thus, in the TUM characterization, C is the desired linear combination of the columns selected by I.

Assume conversely that Γ is totally unimodular. Then $\det B = 1$ and hence $\{\pi(C_i) \mid C_i \in B\}$ is a basis of $G(D)$. Let C' be any cycle in $G(D)$. Then $C' = \sum_{i \in I} \pi(C_i) \bmod 2$ for some index set $I \subseteq \{1, \ldots, \nu\}$. Since Γ is totally unimodular, there are coefficients $\lambda_i \in \{-1, +1\}$ such that $\sum_{i \in I} \lambda_i C_i$ is a vector C with components in $\{-1, 0, +1\}$. Clearly, $\pi(C) = C'$ and hence C is the desired orientation of C'.

Case 4. Order the columns of Γ such that $C_{\sigma(i)}$ is in the i-th column for $1 \le i \le \nu$. Order the rows of Γ such that an arc a with $a \in C_{\sigma(i)} \setminus (C_{\sigma(1)} \cup \cdots \cup C_{\sigma(i-1)})$ corresponds to row $\nu - 1 + i$.

Case 5. This is nothing but a reformulation of Sysło's characterization [6] of a strictly fundamental cycle basis B, namely that every circuit in the basis contains an arc that is contained in no other circuit of the basis.

Case 6. This is but a reformulation of the definition of 2-bases. □

The determinant of a set of ν circuits can be computed over any field κ. For directed bases the determinant is non-zero in $\mathbb{Q}$, for undirected bases the determinant is non-zero in $GF(2)$. We therefore also call directed bases $\mathbb{Q}$-bases and undirected bases $GF(2)$-bases. We call a directed basis a $GF(p)$-basis, where p is a prime, if its determinant is non-zero modulo p.

Theorem 3.4 establishes most of the inclusions shown in Fig. 4: Every fundamental basis is both weakly fundamental and totally unimodular, every weakly fundamental or totally unimodular basis is integral, every integral basis is undirected, and every undirected basis is directed. We shall next relate 2-bases to the other classes.

Lemma 3.5. *Every 2-basis is totally unimodular and weakly fundamental.*

Proof. Let $B = \{C_1, \ldots, C_\nu\}$ be a 2-basis of G. MacLane [1] showed that a graph having a 2-basis is planar and that, moreover, the basic circuits correspond to the bounded face cycles of some planar embedding of G. Orient the edges of G arbitrarily and let the C_i's correspond to counterclockwise traversals of the face cycles. Then every row of Γ has at most two non-zero entries; if there are two non-zero entries, one is $+1$ and one is -1. Thus Γ is totally unimodular [5, page 274].

We next show that B is weakly fundamental. Let $C = \{e_1, \ldots, e_k\}$ be the boundary of the infinite face of G. For $i = 1, 2, \ldots, k$, denote by C_{e_i} the unique circuit in B that contains $e_i \in C$. In the first iteration, we define

$$C_{\sigma(\nu)} = C_{e_1}, \qquad C_{\sigma(\nu-1)} = C_{e_2}, \ldots, C_{\sigma(\nu-k+1)} = C_{e_k}.$$

Then, we remove the edges of C from G and proceed in the same way for the 2-connected components of the remaining graph. □

[3] This item is a new result.

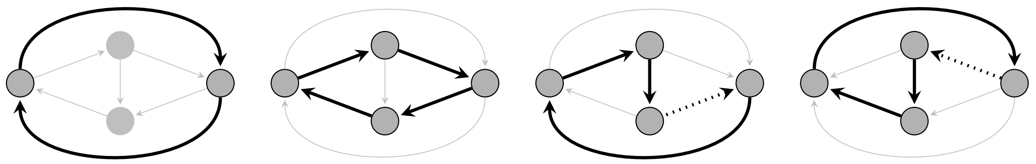


Fig. 5 – A graph and a directed cycle basis. For each of the four circuits, the arcs belonging to the circuit are shown in bold. Arcs used in reversed direction are shown dotted. Every arc is used in exactly two circuits. The determinant of this basis is two. Thus the basis is not totally unimodular. Also, since each arc is used in exactly two circuits, the basis is not weakly fundamental.

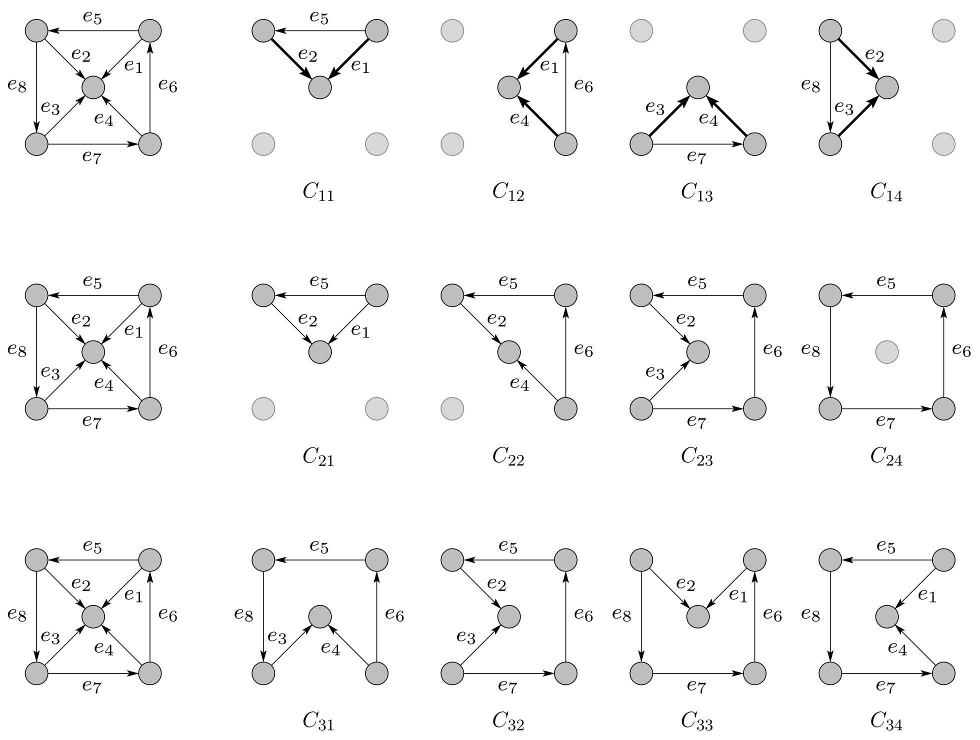


Fig. 6 – Examples of a strictly fundamental cycle basis that is also a 2-basis, a weakly fundamental cycle basis, and a non-integral cycle basis in the wheel graph W_5. The last of these originates from [7].

We required a 2-basis to use every arc at most twice and to be undirected. Fig. 5 shows a graph and a directed basis that uses every arc exactly twice and is neither totally unimodular nor weakly fundamental (Tomasz Jurkiewicz, personal communication).

Open Problem 3. The definition of zero-one bases may seem strange. It would be equally natural to require that every circuit (every simple cycle) is a linear combination of the basic circuits with coefficients in $\{-1, 0, +1\}$. How do these definitions relate?

3.3. *Simple examples*

Fig. 6 presents three cycle bases for the wheel graph W_5: the strictly fundamental cycle basis $B_1 = \{C_{11}, C_{12}, C_{13}, C_{14}\}$, which is also a 2-basis, the weakly fundamental cycle basis $B_2 = \{C_{21}, C_{22}, C_{23}, C_{24}\}$, and the undirected basis $B_3 = \{C_{31}, C_{32}, C_{33}, C_{34}\}$; the lattermost is *not* integral. The strictly fundamental cycle basis B_1 corresponds to the spanning tree $T = \{e_1, e_2, e_3, e_4\}$. The corresponding cycle matrices are as follows:

$$\Gamma_1 = \left(\begin{array}{cccc} -1 & 1 & 0 & 0 \\ 1 & 0 & 0 & -1 \\ 0 & 0 & -1 & 1 \\ 0 & -1 & 1 & 0 \\ \hline 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right), \quad \Gamma_2 = \left(\begin{array}{cccc} -1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 \\ \hline 1 & 1 & 1 & -1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{array}\right),$$

$$\Gamma_3 = \left(\begin{array}{cccc} 0 & 0 & 1 & -1 \\ 0 & 1 & -1 & 0 \\ 1 & -1 & 0 & 0 \\ -1 & 0 & 0 & 1 \\ \hline 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{array}\right).$$

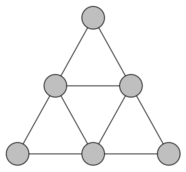


Fig. 7 – The sunflower graph SF(3) has a unique minimum cycle basis that is a 2-basis.

The first four rows correspond to the arcs of T and the last four rows correspond to the non-tree arcs. In Γ_1, every row has at most two non-zero entries and the last four rows constitute a 4×4 unit matrix. Thus B_1 is a 2-basis and is strictly fundamental. In Γ_2, the last four rows constitute a regular upper triangular matrix and so B_2 is weakly fundamental. Finally, in Γ_3 the determinant of the submatrix formed by the last four rows has determinant three. Hence, B_3 is undirected but *not* integral. As a consequence, it can neither be weakly fundamental, and thus its rows and columns cannot be permuted so as to provide a triangular matrix. A direct demonstration that B_3 is not integral is provided by the representation of the circuit C_{24} as a linear combination of the basis B_3, namely

$$C_{24} = \frac{1}{3}C_{31} + \frac{1}{3}C_{32} + \frac{1}{3}C_{33} + \frac{1}{3}C_{34}.$$

3.4. *Variants of the MCB problem*

Each of our classes of cycle bases induces its own variant of the MCB problem. Let D be a directed graph and let $\mathcal{B}$ be a class of cycle bases of D. A minimum (weight) cycle basis of class $\mathcal{B}$ is a basis $B' \in \mathcal{B}$ such that

$$w(B') = \min\{w(B) \mid B \in \mathcal{B}\}.$$

For instance, in the minimum strictly fundamental cycle basis (MFCB) problem we aim at finding a spanning forest in D such that the sum of the weights of its induced fundamental circuits is as small as possible.

Our seven classes define seven different minimum cycle basis problems, i.e., for any two distinct classes $\mathcal{B}_1$ and $\mathcal{B}_2$ there is a directed graph D and a weight function w such that

$$\min\{w(B) \mid B \in \mathcal{B}_1\} \neq \min\{w(B) \mid B \in \mathcal{B}_2\}.$$

In the sequel, we show some of these differences; for the others, we refer our readers to Liebchen and Rizzi [2] and to Fig. 4. In each case, we will exhibit a graph, a weight function, and a basis B, argue that the basis belongs to class $\mathcal{B}_1$, and finally show that every basis of class $\mathcal{B}_2$ must have a larger weight. The graphs that we present next differentiate between the following pairs of the MCB problem:

1. strictly fundamental cycle bases vs. 2-bases and weakly fundamental cycle bases;
2. weakly fundamental cycle bases vs. integral cycle bases;
3. integral cycle bases vs. undirected cycle bases; and
4. undirected vs. directed cycle bases.

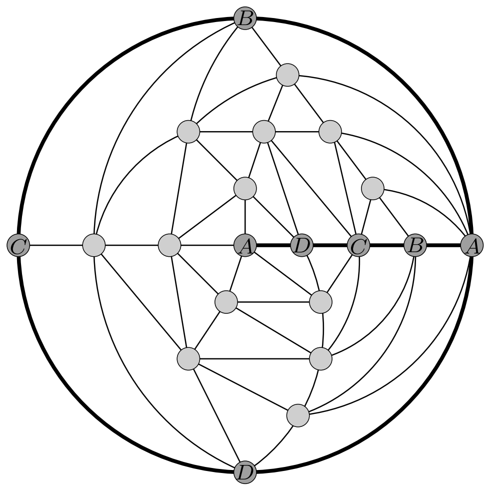


Fig. 8 – The minimum integral cycle basis of Champetier's graph [9] is unique and not weakly fundamental. Nodes with the same label are to be identified.

A graph that distinguishes between the MCB problems for weakly fundamental and totally unimodular cycle bases is given in Fig. 33 of Section 6. For planar graphs, most variants of the minimum cycle basis problem are the same. We will see in Theorem 5.34 that every planar graph has a minimum directed cycle basis that is weakly fundamental, totally unimodular, and integral.

Example 3.1 (F-*bases vs. 2-bases and* W-*bases*). The sunflower graph SF(3) in Fig. 7 contains precisely four circuits with three edges. These are independent and hence constitute its unique minimum cycle basis B. Obviously, B is a 2-basis. And, by Lemma 3.5, B is also weakly fundamental.

However, B is not strictly fundamental, since every edge of the center triangle is contained in two circuits of the basis; cf. case 5 of Theorem 3.4. This example was inspired by Hubicka and Syslo [8].

Example 3.2 (W-*bases vs.* I-*bases*). Champetier [9] introduced the graph shown in Fig. 8. The graph is specified as a node-labelled planar graph. The nodes sharing a label are to be identified. The resulting simple G_{Ch} has 17 vertices and 52 edges. There are precisely 36 triangles in G_{Ch} and they correspond to the finite faces of the underlying planar graph.

Claim 3.1 ([10]). *The 36 triangles in G_{Ch} constitute the unique minimum cycle basis B of G_{Ch}. B is integral but not weakly fundamental.*

Proof. Consider some orientation D of G_{Ch} and orient the circuits in B clockwise, with respect to Fig. 8. Consider the sum C' over $\mathbb{Q}$ of all the triangles, $C' = \sum_{C \in B} C$. In G_{Ch}, all edges except for the ones shown bold in Fig. 8, are part of two triangles. The bold edges belong to three triangles. Thus, C' is the 4-circuit that links the labeled vertices. In Fig. 8, this corresponds to following the outer bold circuit clockwise, or following its representation as a path from left to right.

We now construct a new basis B' by replacing an arbitrarily chosen circuit of B by C'. Let Γ and Γ' be the corresponding

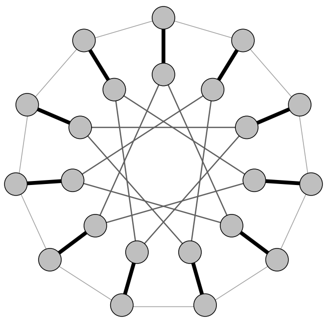


Fig. 9 – A weighted version of the generalized Petersen graph $P_{11,4}$ has a unique minimum cycle basis that is not integral.

cycle matrices. Consider the transformation matrix R such that $\Gamma' = \Gamma R$. With $R = [r_1, \ldots, r_\nu]$, we have $r_i = \mathbf{1}$ for some $i \in \{1, \ldots, \nu\}$, and $r_j = e_j$ for all $j \neq i$. Hence, R constitutes a unimodular transformation and thus B and B' have the same determinant.

The cycle basis B' is weakly fundamental. As in the proof of Lemma 3.5, one can construct a suitable ordering of its circuits. Thus $\det B' = 1$ and hence also $\det B = 1$. We conclude that B is an integral basis. However B is *not* weakly fundamental because every arc is part of two or three triangles. □

We mention that the minimum cycle basis B of Champetier's graph is *not* totally unimodular; cf. [2].

Example 3.3 (U-*bases vs* I-*bases*). Consider the generalized Petersen graph $P_{11,4}$ (Fig. 9) with the following weight function

$$w_{ij} = \begin{cases} 4, & \text{if } i \text{ and } j \text{ are outer vertices,} \\ 5, & \text{if } i \text{ and } j \text{ are inner vertices, and} \\ 12, & \text{otherwise.} \end{cases}$$

Claim 3.2. *$(P_{11,4}, w)$ has precisely 12 circuits of weight 44 or less. These constitute the unique minimum cycle basis.*

Proof. Any cycle basis consists of $\nu = 33 - 22 + 1 = 12$ circuits. We call the edges e with $w_e = 12$ *spokes* and observe that every circuit contains an even number of spokes. There are only two circuits with no spokes; the outer circuit has weight 44 whereas the inner circuit has weight 55. Any circuit with at least four spokes has a weight of at least 48.

We classify the circuits that contain two spokes according to the number of their outer edges. Since there are always two possible choices for the path through the inner edges, we only consider the shorter one in Table 1. Similarly, we may restrict attention to circuits that use at most $5 = \left\lfloor \frac{11}{2} \right\rfloor$ outer edges.

Let B consist of the outer circuit plus the 11 circuits that use precisely one outer edge. We claim that B is an undirected cycle basis. Assume otherwise. Then, there exists a non-trivial linear combination yielding the zero vector over $GF(2)$. If such a combination made use of any of the 11 circuits that use precisely one outer edge, then it would have to use each of these circuits in order to cancel out the spokes. The sum of these 11 circuits is the outer circuit plus the inner circuit. Thus there is no non-trivial linear combination yielding the zero vector. □

Table 1 – Weights of the circuits in $(P_{11,4}, w)$ that use two spokes.

Number of outer edges	1	2	3	4	5
Number of inner edges	3	5	2	1	4
Weight of the shorter circuit	43	57	46	45	64

It remains to show that B is *not* an integral cycle basis. Indeed, its determinant is three, as a simple calculation shows. Alternatively, we observe that the sum of all circuits in B is three times the inner circuit.

The basis B has cost $11 \times 43 + 44 = 517$. The minimum integral cycle basis has cost 518. It consists of the 11 circuits that use exactly one outer edge plus one circuit that uses four consecutive outer edges, two spokes and one inner edge. We leave it to the reader to verify that this basis is integral. It is easy to see that the inner and outer circuits can be obtained as integer linear combinations. For example, subtracting from the circuit that uses four consecutive outer edges the four circuits that use one of these edges each yields a circuit with no outer edge, no spoke, 12 clockwise uses of inner edges and one anti-clockwise use of an inner edge, i.e., the inner circuit results. Adding the 11 circuits using one outer edge each yields the outer circuit plus three copies of the inner circuit.

Example 3.4 (D-*bases vs* U-*bases*). Consider the generalized Petersen graph $P_{7,2}$. A circuit $C_{\ell,k}$ using ℓ consecutive outer edges uses k inner edges where $\ell \pm 2k = 0 \bmod 7$. Summing over all circuits $C_{l,k}$ yields ℓ copies of the outer circuit and k copies of the inner circuit. Thus, over $GF(2)$, the inner circuit C_I is a linear combination of the circuits $C_{2,1}$, however, over $\mathbb{Q}$ it is not. In other words, over $\mathbb{Q}$, the circuits $C_{2,1}$ plus C_I form a basis, and over $GF(2)$ they do not. The weight function

$$w_{ij} = \begin{cases} 3, & \text{if } i \text{ and } j \text{ are outer vertices,} \\ 2, & \text{if } i \text{ and } j \text{ are inner vertices, and} \\ 3, & \text{otherwise} \end{cases}$$

ensures that the circuits $C_{2,1}$ are cheaper than all circuits $C_{\ell,k}$ with $(2, 1) \neq (\ell, k)$, than all circuits with at least four spokes, and than the outer circuit C_O. Also, C_I is cheaper than C_O. We conclude that the circuits $C_{2,1}$ plus C_I form a $\mathbb{Q}$-basis of weight $8 \cdot 14 = 112$ and that any $GF(2)$-basis is more costly.

The minimum weight integral (and hence $GF(2)$-) basis has weight 113. It consists of the seven circuits of type $C_{2,1}$ plus one circuit of type $C_{1,3}$.

3.5. *Directed and GF(p)-bases*

We show that the computation of minimum directed cycle bases can be reduced to the computation of minimum $GF(p)$-bases for suitable primes p.

Lemma 3.6. *Let B be a minimum weight directed cycle basis and let p be a prime. The weight of a minimum weight $GF(p)$-basis is no smaller than the weight of B. If p does not divide the determinant of B, B is also a minimum weight $GF(p)$-basis.*

Proof. Linear dependence over $\mathbb{Q}$ implies linear dependence over $GF(p)$ for any p. Therefore, any $GF(p)$-basis is a directed basis. If the determinant of B is not divided by p, $\det B \bmod p \neq 0$ and B is a $GF(p)$-basis. □

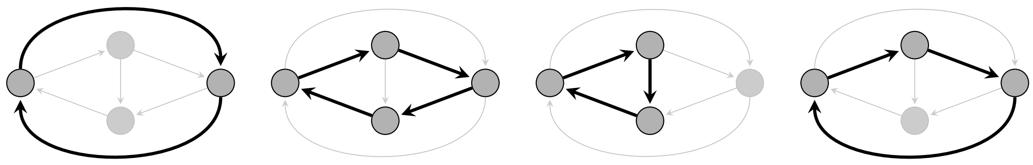


Fig. 10 – A graph and a totally unimodular cycle basis. For each of the four circuits, the edges belonging to the circuit are shown in bold.

To apply the preceding lemma, we need a bound on the determinant of a directed cycle basis. Consider any directed cycle basis B. Its determinant is the determinant of a $\nu \times \nu$ matrix with entries in $\{-1, 0, +1\}$. Moreover, each column of this matrix contains at most n non-zero entries.

Lemma 3.7. *The determinant of a directed cycle basis is an integer bounded by $n^{m/2}$.*

Proof. The determinant is a sum of $\nu!$ terms; each term has an absolute value of at most one. This gives a bound of $\nu! \le \nu^{\nu}$. Hadamard's inequality yields a slightly better bound. The absolute value of the determinant is bounded by the product of the ℓ_2-norms of the column vectors. The norm of each column vector is at most $\sqrt{n}$ and hence we have the bound $\sqrt{n}^{\nu}$. □

Combining the two preceding lemmas, we obtain a characterization of minimum directed bases in terms of minimum $GF(p)$-bases.

Theorem 3.8. *Let P be a set of m primes each of a value of at least n. For each $p \in P$, let B_p be a minimum $GF(p)$-basis, and let p_0 be such that B_{p_0} has minimum weight among the bases B_p. Then:*

1. *B_{p_0} is a minimum weight directed basis.*
2. *Let $p \in P$ be chosen uniformly at random. Then B_p is a minimum weight directed basis with probability at least 1/2.*

We can choose P such that $p \in O(m \log m)$ for all $p \in P$.

Proof. Let B be any minimum directed basis. No more than $m/2$ primes in P can divide the determinant of B.

For an integer s, let $\pi(s)$ be the number of primes less than or equal to s. Then $s/(6 \log s) \le \pi(s) \le 8s/\log s$ [11]. Then there are at most $8n/\log n$ primes less than n. If t is such that $t/(6 \log t) \ge 8n/\log n + m$, then there are at least m primes of a value of at least n less than t; $t = O(m \log m)$ suffices. □

If $p = O(m \log m)$ and hence $\log p = O(\log m)$, the arithmetic in $GF(p)$ takes $O(1)$ time.

3.6. Circuits versus cycles

We defined cycle bases as sets of circuits. Alternatively, we could have defined them as sets of cycles. Is there always a minimum weight basis that consists only of circuits? Is the minimum weight basis of a disconnected graph the union of minimum weight bases of the components? For some of our classes, the answer is yes. For some, the answers to these and related questions are not known.

Theorem 3.9 (*Exchange Theorem*, [12]). *If B is a D or U-basis of G, $C \in B$ and $C = C_1 + C_2$, then either $B \setminus \{C\} \cup \{C_1\}$ or $B \setminus \{C\} \cup \{C_2\}$ is also a cycle basis of G.*

Proof. Let Γ be the cycle matrix for B and let Γ_i be the cycle matrix for $B - C + C_i$, $i = 1, 2$. Let T be a spanning forest of G and let A and A_i be the respective square submatrices indexed by the arcs not in T. Then, using the linearity of the determinant function for the column which corresponds to C, we find that $0 \ne \det A = \det A_1 + \det A_2$. □

The family of linearly independent cycles forms a matroid.

Theorem 3.10. *The set of (directed) cycles of a graph G forms a matroid. The bases of the matroid clearly coincide with the (directed) cycle bases of G.*

Proof. Let $\mathcal{I}$ denote the system of all linear independent sets of cycles in G. It suffices to show the following:

- $\emptyset \in \mathcal{I}$;
- $A \in \mathcal{I}$ and $B \subset A$ implies $B \in \mathcal{I}$;
- For all sets $A, B \in \mathcal{I}$ with $|A| > |B|$ there exists an element $a \in A \setminus B$ such that $B \cup \{a\} \in \mathcal{I}$.

The listed properties hold since the (directed) cycle space of G forms a vector space. □

We will show that Theorem 3.10 does not hold for integral cycle bases, i.e., the system of all subsets of all integral bases in G does not form a matroid. This will cause the computational approaches suitable for U-bases and D-bases to fail for I-bases. In Section 5.8 will we discuss these issues. Now we will examine the validity of Theorem 3.9 for K-bases with K neither D nor U. We first show that Theorem 3.9 does not hold for totally unimodular bases.

Lemma 3.11 (*T. Jurkiewicz, Personal Communication*). *There is a graph G and a totally unimodular basis B of G containing a circuit C and a decomposition $C = C_1 + C_2$ of C such that neither $B \setminus \{C\} \cup \{C_1\}$ nor $B \setminus \{C\} \cup \{C_2\}$ is a totally unimodular basis.*

Proof. Fig. 10 shows a graph and a TUM-basis of this graph. We invite the reader to verify that this basis is TUM. Fig. 11 shows a decomposition of the first circuit into two circuits. Replacing the first circuit by either one of the two circuits shown in Fig. 11 results in a collection of circuits that is not TUM. In both cases, the cycle matrix of the resulting collection contains a 2 by 2 submatrix of the form

$$\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

This matrix has determinant -2; in a TUM-basis, the determinants of all square submatrices must be in $\{-1, 0, +1\}$. □

For weakly fundamental bases, we can show Theorem 3.9 under the additional assumption that C_1 and C_2 use only edges that are also used by C.

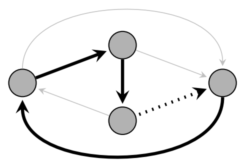


Fig. 11 – Decomposition of the first element of the basis shown in Fig. 10. Edges that are used in reverse direction are shown dotted.

Lemma 3.12. *Let B be a W-basis of G, let $C \in B$ and $C = C_1 + C_2$, where $|C_i(e)| \le |C(e)|$ for all e and $C_i \neq C$ for $i = 1, 2$. Then at least one of $B - C + C_1$ or $B - C + C_2$ is a W-basis of G.*

Proof. If B is a weakly fundamental basis, there is an ordering of the cycles in B such that every cycle introduces a non-tree edge not used in any preceding cycle. Let e be the edge introduced by C. Then $C(e) \neq 0$ and hence at least one of $C_1(e)$ or $C_2(e)$ is non-zero, say the former. We replace C by C_1. Since for any non-zero coefficient of C_1, the corresponding coefficient of C is non-zero, the non-tree part of the new cycle matrix is still lower triangular. □

Observe that Lemma 3.12 is *not* true for strictly fundamental bases. To see this, consider the sunflower graph SF(3) in Fig. 7, and some minimum strictly fundamental cycle basis B of SF(3). If we decompose the 4-circuit $C \in B$ into $C = C_1 + C_2$, where both C_1 and C_2 are triangles, then neither of the $B - C + C_i$ is a strictly fundamental cycle basis. In the next lemma we show that Lemma 3.12 does not hold for integral bases either.

Lemma 3.13. *There is a graph G and an integral basis B of G containing a non-circuit C such that for any decomposition $C = C_1 + C_2$ of C with $|C_i(e)| \le |C(e)|$ for all e and $C_i \neq C$ for $i = 1, 2$, neither $B - C + C_1$ nor $B - C + C_2$ is an integral basis.*

Proof. The graph is $P_{7,3}$ as shown in Fig. 12. It consists of two disjoint cycles of length 7, called the outer and the inner cycle, respectively. We use O_i and I_i, $0 \le i < 7$, to denote the nodes on the outer and inner cycle, respectively. The outer and inner cycles have edges (I_i, I_{i+1}) and (O_i, O_{i+4}), $0 \le i < 7$, respectively. All indices are modulo 7. Furthermore, we have the edges (O_i, I_i), $0 \le i < 7$, called spokes. To summarize, $n = 14$, $m = 21$, and the cyclomatic number ν is thus eight.

The basis B consists of the following cycles. For $0 \le i < 7$, we have the cycle C_i consisting of the edges (O_i, O_{i+1}), (O_{i+1}, O_{i+2}), (O_{i+2}, I_{i+2}), (I_{i+2}, I_{i+6}), (I_{i+6}, I_{i+10}), (I_{i+10}, I_{i+14}), and (I_{i+14}, O_i). Observe that the sum of the C_i's is the nonsimple cycle consisting of two copies of the outer circuit and three copies of the inner circuit. We also have the cycle $D_{a,b}$ which consists of a copies of the outer circuit and b copies of the inner circuit, where $a, b \in \mathbb{Z}$. We will fix a and b later.

We next determine the determinant of the above set of cycles as a function of a and b. We fix a spanning tree T consisting of the spokes and all inner edges except for edge (I_2, I_6). We obtain the following square part of the cycle matrix:

$$\begin{array}{lcccccccc}
 & C_0 & C_1 & C_2 & C_3 & C_4 & C_5 & C_6 & D_{a,b} \\
(O_0, O_1) & 1 & 1 & & & & & & a \\
(O_1, O_2) & & 1 & 1 & & & & & a \\
(O_2, O_3) & & & 1 & 1 & & & & a \\
(O_3, O_4) & & & & 1 & 1 & & & a \\
(O_4, O_5) & & & & & 1 & 1 & & a \\
(O_5, O_6) & & & & & & 1 & 1 & a \\
(O_6, O_1) & 1 & & & & & & 1 & a \\
(I_2, I_6) & 1 & & & 1 & 1 & & & b.
\end{array}$$

Observe that the edge (I_j, I_{j+4}) is used by the cycles C_{j-2}, C_{j-6}, and C_{j-10}. The determinant of the matrix above is $2b - 3a$, as a little calculation, e.g., Gaussian elimination, shows. For $a = b = 1$, the determinant is -1 and hence the basis is integral. The cycle $D_{1,1}$ is not a circuit and uses the outer O and the inner circuit I in the forward direction. The only decomposition of D is $O + I$. The determinant of the basis $B - D + O$ is -3 (use $a = 1$ and $b = 0$ in the formula for the determinant) and the determinant of the basis $B - D + I$ is 2 (use $a = 0$ and $b = 1$ in the formula for the determinant). Thus neither basis is integral. □

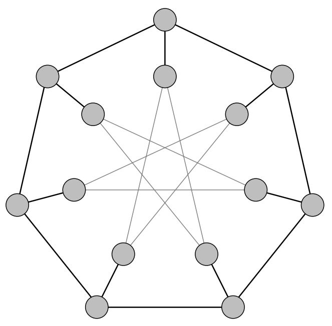


Fig. 12 – The generalized Petersen graph $P_{7,3}$. We provide an integral cycle basis of $P_{7,3}$ which features a nonsimple cycle, but cannot be decomposed into a basis which only consists of circuits.

The next two lemmas provide us with properties of minimum cycle bases which are extremely valuable in practice. These properties are an immediate consequence of Lemma 3.12 and turn out to be true for strictly fundamental cycle bases, too.

Theorem 3.14. *For $K \in \{D, U, W, F\}$, any graph G has a minimum K-basis consisting only of circuits.*

Proof. The cycles in fundamental bases are circuits by definition. For any of the other K's, consider a basis B containing a cycle C that is not a circuit. We may decompose C into a sum of circuits C_i, $1 \le i \le k$. By the preceding lemmas, one of the sets $B - C + C_i$ is a K-basis of G. Also, $w(C_i) \le w(C)$. □

Theorem 3.15. *For $K \in \{D, U, W, F\}$, the union of minimum weight K-bases of its maximal 2-connected components are a minimum weight K-basis.*

Proof. By the preceding theorem, there is a minimum weight K-basis consisting only of circuits. A circuit uses edges only from one 2-connected component. □

Open Problem 4. Does Theorem 3.14 or Theorem 3.15 hold for integral bases or totally unimodular bases? Does Lemma 3.12 hold for totally unimodular bases?

3.7. *Reductions*

We study some simplification rules. At first sight, all might appear quite natural. However, for certain classes of cycle bases, we do not know whether these rules are valid.

For example, is there a simple way to deal with parallel edges? Is there a simple way of handling edges of weight zero?

Let $g = (u, v)$ be a zero weight edge without parallel edges. Let G' be obtained from G by removing g and identifying u and v, i.e., replacing in all edges incident to u or v, u and v by a new vertex uv. The edges of G' correspond to the edges in $E - g$. Let B' be a K-basis of G', where $K \in \{D, U, I, W, TUM, F\}$. Consider the following set B of cycles in G: for any $C' \in B'$ we add a cycle C to B that is obtained from C' by adding g with appropriate multiplicity; the appropriate multiplicity guarantees flow conservation at u and v.

Lemma 3.16. *Let G' be obtained from G by contracting an edge of cost zero not having any parallel edges, let B' be a minimum weight K-basis of G', and let B be obtained from B' as described above. Then B is a minimum weight K-basis of G for $K \in \{D, U, I, W, TUM, F\}$.*

Proof. Let T' be a spanning forest of G' and let Γ' be the cycle matrix corresponding to B'. Let A' be the square submatrix selected by the rows not in T'. Then $T := T' + g$ is a spanning tree of G. We obtain the cycle matrix for B by adding a row for g and setting the entries in this row appropriately. Observe that A' remains the square matrix selected by the non-tree edges. Thus B is a K-basis of G. The weight of B is the weight of B'.

Conversely, let B be any K-basis of G and let B' be obtained from B by identifying u and v. The matrix Γ' for B' is obtained from the matrix Γ for B by deleting the row corresponding to g.

Let C' be any cycle in G'. We lift C' to a cycle C in G. The representation of C with respect to B translates into a representation of C' with respect to B'. Thus B' is also of type K. Also, $w(B') \le w(B)$. □

Lemma 3.17 ([12]). *Let $K \in \{D, U\}$ and let e be any edge. For any minimum weight circuit F containing e, there is a minimum weight K-basis containing F. Any minimum weight K-basis contains a minimum weight circuit containing e.*

Proof. Let B be a minimum weight K-basis. Then $F = \sum_{C \in B} \lambda_C C$. Clearly, there must be a $C \in B$ such that $e \in C$ and $\lambda_C \neq 0$. Then $w(F) \le w(C)$ and $B' := B \setminus C + F$ is a K-basis of weight no larger than the weight of B. Hence B' is a minimum weight K-basis. If $w(F) < w(C)$, then B was not a minimum weight K-basis. □

Lemma 3.17 does not hold for strictly fundamental bases. The sunflower graph SF(3) of Fig. 7 provides an example. Any F-basis contains a circuit of length 4. For the edge inducing this circuit, the shortest circuit through this edge is not contained in B.

Lemma 3.18. *Let g and f be parallel edges with $w(g) \le w(f)$. For $K \in \{D, U, W\}$ a minimum weight K-basis of G can be obtained from a minimum weight K-basis B' of $G' := G - f$ by adding a cheapest circuit through f; call it C.*

Proof. C is clearly independent of B'. Also, C introduces an edge that is not used in any of the other cycles. Thus, if B' is a K-basis of G', $B := B' \cup C$ is a K-basis of G with $w(B) = w(B') + w(C)$. Assume, for the sake of argument, that G has a K-basis $\hat{B}$ with $w(\hat{B}) < w(B)$. We will show that this implies that G' has a K-basis of weight less than $w(B')$.

Assume first that $K \in \{D, U\}$. A cheapest circuit containing f is either the circuit[4] $g \circ f^{-1}$ or has the form $f \circ P$ where P is a cheapest path connecting the endpoints of f in $G \setminus \{f, g\}$. In the latter case, $g \circ P$ is a cheapest cycle containing g. By Lemma 3.17 we may assume that B contains the circuit $g \circ f^{-1}$ in the former case or the circuits $g \circ P$ and $f \circ P$ in the latter case. Assume now, that $\hat{B}$ contains another circuit, say $f \circ Q$, containing f. Replacing this circuit by $g \circ Q$ yields a basis of weight no larger than $\hat{B}$ since $g \circ Q = f \circ Q + g \circ f^{-1}$ in the former case and $g \circ Q = f \circ Q + g \circ P - f \circ P$ in the latter case. We conclude that G has a basis of weight no larger than $\hat{B}$ in which $g \circ f$ or $f \circ P$ is the only circuit containing f. Deleting this circuit from the basis gives us a basis of $G - f$.

For $K = W$, we have to argue differently. Let $\hat{\Gamma}$ be the cycle matrix for $\hat{B}$. We may assume that $\hat{\Gamma}$ has an upper triangular matrix in its last ν rows, with the arcs of some spanning forest T placed above.

Assume first that the row for g is above the row for f. Then f must be a non-tree arc, because otherwise g and f form a circuit in T. Hence, there is a circuit $C_f \in \hat{B}$ "introducing" f, i.e., the diagonal entry in the row indexed by f belongs to C_f. We delete C_f from the basis and replace, in the other basic circuits, occurrences of f by g. Removing the row of f as well, we obtain the cycle matrix of a W-basis for G' of weight $w(\hat{B}) - w(C_f) < w(B) - w(C) = w(B')$, which is a contradiction.

Assume next that the row for f is above the row for g. Then g must be a non-tree arc and hence there is a circuit C_g introducing g; f may be a tree arc or a non-tree arc. If f is a tree arc, we make g a tree arc, replace f by g in all circuits and delete f and C_g. If f is a non-tree arc, the circuit C_f introducing f does not use the arc g, because g was assumed to be arranged below f. We replace f by g in all circuits and delete f and C_g. The circuit C'_f obtained from C_f by replacing f by g now introduces g. In either case, we obtain the cycle matrix of a W-basis for G' of weight $w(\hat{B}) - w(C_g) < w(B) - w(C) = w(B')$, which is again a contradiction. □

Open Problem 5. Extend statements 3.12 to 3.18 to types of cycle bases not covered by the statements.

Open Problem 6. Let $e = \{u, v\}$ be a non-metric edge of a biconnected graph G, i.e., $\text{dist}_G(u, v) < w(e)$. Each minimum K-basis B has precisely one circuit $C \in B$ with $e \in C$. This is true for $K \in \{D, U\}$. Is this true for any other type?

[4] We assume that g and f are oriented in the same way; f^{-1} is the reversal of f and runs anti-parallel to g.

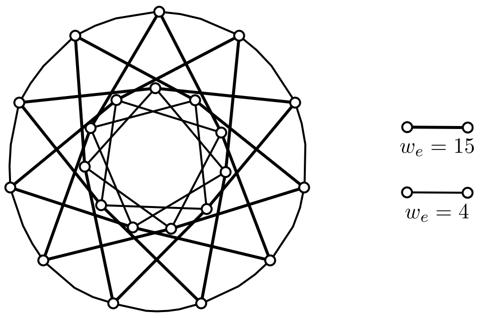


Fig. 13 – A graph G featuring minimal integral cycle bases with different weight sequences. The graph consists of two cycles of length 11 that are connected by 22 spokes. The edges on the outer and inner cycle have weight 4 each, and the spokes have weight 15.

3.8. *Weight sequences*

We consider the sequence of weights of circuits in a minimal K-basis sorted into non-decreasing order. Let $K \in \{D, U\}$ and B and B' be distinct K-bases of G, both of minimal weight. Then their ordered sequences of weights coincide. This is not true for integral bases.

Lemma 3.19. *For $K \in \{D, U\}$, let σ and σ' be the non-decreasing sequences of weights of circuits of two minimal K-bases B and B', respectively. Then $\sigma = \sigma'$.*

Proof. This is true since both the undirected and the directed cycle space form a vector space over $GF(2)$ and $\mathbb{Q}$, respectively. Hence the cycles, together with linear independence, form a *matroid* (cf. Theorem 3.10). Finally, it is a well known fact that the non-decreasing weight sequences of minimal bases in matroids coincide. □

Lemma 3.20. *There is a graph G and two minimal integral bases, B and B', of G whose non-decreasing weight sequences do not coincide.*

Proof. Consider the graph G depicted in Fig. 13. It arises from the generalized Petersen graph $P_{11,3}$ by the addition of an extra set of 11 spokes. We give weight four to all inner and outer edges and weight 15 to all spokes. Let B and B' be the two sets of circuits in G, shown in Figs. 14 and 15, respectively. Either set forms an integral cycle basis of G as the reader may verify. To see minimality of B and B', note that the first 22 circuits in B are in fact the only ones in the graph whose weight does not exceed 42. Besides those, there are only two more circuits – the inner and outer ones – whose weight is at most 44. Replacing the last circuit in B by either the inner or the outer ring yields a non-integral basis (the determinant is either 2 or 3). Every circuit in G other than the so far considered ones has weight at least 46. Hence, there are exactly two sets of 23 circuits whose weight is less than 926. Both form a cycle basis of G, but neither is integral. □

3.9. *Necessary and sufficient conditions for optimality*

We now derive necessary and sufficient conditions for optimality of a given cycle basis. We also show that if no minimum directed basis is integral, then no minimum integral basis is TUM.

Let G be a connected directed graph with spanning tree T, let $N = E \setminus T$ be the set of non-tree arcs, and let B be a directed cycle basis. For any cycle C, let C' be the restriction of C to the non-tree arcs and let Γ' be the $\nu \times \nu$ matrix formed by the restrictions of the circuits in B. Then Γ' is a non-singular matrix and hence has an inverse. We may write the inverse as $(1/\det B)S$ for an integral matrix S. Then $S \cdot \Gamma' = (\det B)I$. That is,

$$\langle S^i, C_j' \rangle = (\det B)\delta_{ij} \quad \text{for all } i \text{ and } j,$$

where $\langle .,. \rangle$ is the inner product of vectors, δ_{ij} is Kronecker's symbol, and S^i denotes the i-th row of S. We will next show that the C_j's must satisfy a local optimality condition.

Theorem 3.21 ([13]). *If $B = \{C_1, \ldots, C_\nu\}$ is a minimum weight K-basis for $K \in \{D, U, I\}$, then there is a $\nu \times \nu$ integral matrix S such that $\langle S^i, C_j' \rangle = 0$ for $i \neq j$, and for all j,*

- *C_j is a minimum weight circuit with $\langle S^j, C_j' \rangle \neq 0$, if $K = D$,*
- *C_j is a minimum weight circuit with $\langle S^j, C_j' \rangle \neq 0 \bmod 2$, if $K = U$,*
- *C_j is a minimum weight circuit with $\langle S^j, C_j' \rangle = \pm 1$, if $K = I$.*

Proof. Let S be defined as in the discussion preceding the theorem. Then, certainly, $\langle S^i, C_j' \rangle = 0$ for $i \neq j$. So assume that there is a j such that C_j does not have the second property. Let D_j have the property and let Γ'' be the matrix obtained by replacing C_j' by D_j'. Then

$$S \cdot \Gamma'' = J,$$

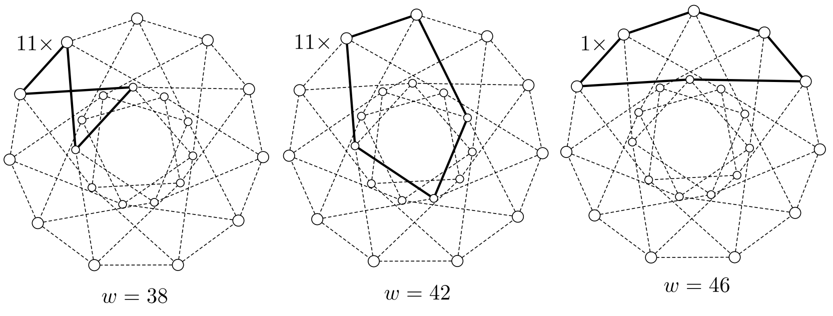


Fig. 14 – A minimal I-basis B of G with weight sequence σ = (38, . . . , 38, 42, . . . , 42, 42, 46).

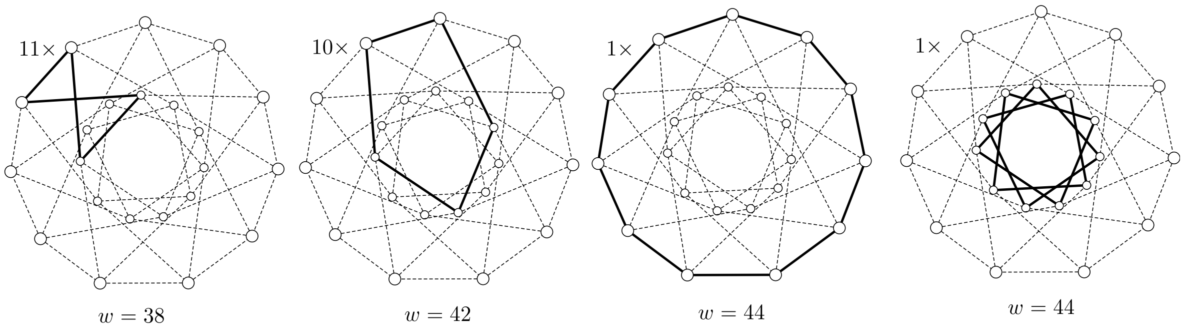


Fig. 15 – A minimal I-basis B' of G with weight sequence $\sigma' = (38, \ldots, 38, 42, \ldots, 42, 44, 44)$.

where J is equal to $(\det B)I$ in all columns except for column j. The j-th column has a non-zero diagonal element for $K = D$, has a non-zero diagonal element modulo 2 for $K = D$, and has diagonal element ± 1 for $K = I$. Thus, $\det J \neq 0$ for $K = D$, $\det J \neq 0 \bmod 2$ for $K = U$, and $\det J = 1$ for $K = I$. Thus $B - C_j + D_j$ is a K-basis cheaper than B. For $K = I$, this follows from the fact that $\det S \in \mathbb{Z}$, $\det \Gamma'' \in \mathbb{Z}$, and $\det S \cdot \det \Gamma'' = \det J = 1$ implies $\det \Gamma'' = 1$. □

For $K \in \{D, U\}$, local optimality implies global optimality.

Theorem 3.22 ([14]). *Let $\kappa \in \{\mathbb{Q}, GF(p)\}$ and let $B = \{C_1, \ldots, C_\nu\}$ be a set of ν circuits. If there is a $\nu \times \nu$ integral matrix S such that $\langle S^i, C'_j\rangle = 0$ for $i \neq j$, and for all j, C_j is a minimum weight circuit with $\langle S^j, C'_j\rangle \neq 0$ (evaluation in κ), then B is a minimum weight κ-basis.*

Proof. Observe first that $S \cdot \Gamma'$ is a diagonal matrix whose diagonal elements are non-zero elements in κ. Thus, $\det \Gamma'$ is a non-zero element of κ and B is a κ-basis.

If B is not a minimum κ-matrix, there is a maximal j such that $\{C_1, \ldots, C_j\}$ can be extended to a minimum κ-basis. We have $j < \nu$. Let $B' = \{C_1, \ldots, C_j, D_{j+1}, \ldots, D_\nu\}$ be a minimum κ-basis. C_{j+1} is a linear combination of the vectors in B', i.e., $C_{j+1} = \sum_{i \le j} \lambda_j C_j + \sum_{i > j} \lambda_j D_j$. Since $\langle S^{j+1}, C'_{j+1}\rangle \neq 0$, there must be an $i > j$ such that $\lambda_j \langle S^{j+1}, D'_i\rangle \neq 0$. Then $w(C_{j+1}) \le w(D_i)$ and hence $B - D_i + C_{j+1}$ is also a minimum weight κ-basis. □

The argument above does not work for integral basis, since the determinant of $B - D_i + C_{j+1}$ may be different from one. Theorem 3.21 has an interesting consequence. A TUM-basis that is a minimum integral basis is also a minimum directed basis.

Theorem 3.23. *If no minimum directed basis of a graph is integral, then no minimum integral basis is TUM.*

Proof. We argue indirectly. Let $B = \{C_1, \ldots, C_\nu\}$ be a minimum integral basis that is also TUM. We show that B is a minimum directed basis. By Theorem 3.21 there is an integral matrix S such that $\langle S^i, C'_j\rangle = 0$ for $i \neq j$, and such that for all j, C'_j is a minimum weight circuit with $\langle S^j, C'_j\rangle = \pm 1$. We claim that C_j is a minimum weight circuit with $\langle S^j, C'_j\rangle \neq 0$. The theorem then follows from Theorem 3.22.

Fix j and consider any circuit C with $\langle S^j, C'\rangle \neq 0$. Since B is TUM, we have $C = \sum_i \lambda_i C_i$ with $\lambda_i \in \{-1, 0, +1\}$. Then $\langle S^j, C\rangle = \lambda_j \langle S^j, C'_j\rangle = \pm 1$ and hence $w(C) \ge w(C_j)$. □

4. Length and weight of cycle bases

In this section we discuss a priori bounds on the length and weight of minimum cycle bases. We state the bounds as functions of the number n of vertices, the number m of arcs, and the total weight W of the edges. Many applications benefit from small length or small weight bases as we will see in Section 7; algorithms for computing minimum or nearly minimum weight bases will be discussed in Section 5. Table 2 summarizes the results. It is interesting to note that all upper bounds have been shown for either weakly or strongly fundamental bases. Although we know that general bases are not always fundamental (see Example 3.3), it seems that fundamentality gives sufficient structure to the problem to make an analysis of their length achievable or at least easier than for general bases.

Open Problem 7. Derive apriori bounds on the weight (length) of directed, undirected, integral, and totally unimodular bases.

Open Problem 8. For $K, K' \in \{D, U, I, TUM, W, F\}$ and a graph G with weight function W, let

$$r_{K,K'}(G) = \frac{\text{weight of a minimum } K\text{-basis}}{\text{weight of a minimum } K'\text{-basis}}$$

and

$$r_{K,K'}(n, m) \\ = \max\{r_{K,K'}(G) \mid G \text{ is a graph with } n \text{ nodes and } m \text{ edges}\}.$$

Derive upper and lower bounds on $r_{K,K'}(n, m)$. For example, $r_{W,D}(n, m) = O(\log n)$, since every graph with n nodes has a W-basis of weight $O(W \log n)$ (see Theorem 4.4) and since every D-basis has a weight of at least W. In the preceding chapter, we established $r_{K,K'}(n, m) > 1$ for certain pairs K and K' and certain values of n and m.

The bounds given in Table 2 are obtained by different methods. There are essentially four approaches:

1. Use of special graph properties like planarity.
2. Induction.
3. Use of clusters, partitions, and spanners.
4. Results of extremal graph theory.

Table 2 – Bounds for minimum weight W- and F-bases. W denotes the total edge weight and $W(\mathrm{MST})$ is the weight of a minimum spanning tree. Bounds for unweighted graphs are only stated if they are better than the bound derived for weighted graphs with $W = m$. In the bound for planar graphs, ϕ is the maximal size of any face.

Graph class	Minimum W-basis	Minimum F-basis
Weighted		
General	$O(W \log n)$, Theorem 4.4	
General	$O(n \cdot W(\mathrm{MST}) + W)$, Theorem 4.2	$O(W \log^2 n \log\log n)$, Theorem 4.11
Planar	$\Theta(W)$	
Unweighted		
General	$O(m \log n \log(m/n))$, Theorems 4.1 and 4.5	$O(n^2)$, Theorem 4.12
Planar		$O(n\sqrt{n\phi})$, Theorem 4.7
Outerplanar		$\Theta(n)$, [15]
d-dim grids	$\Theta(n)$	$\Theta(n \log n)$, Theorem 4.8

We start with some obvious bounds. Throughout this section, we restrict attention to biconnected graphs. There are $m - n + 1$ circuits in a basis and each circuit has a length of at most n. Thus any basis has a length of at most mn and a weight of at most mW. Throughout this section, $W = \sum_{e \in E} w(e)$ denotes the total weight of the edges. Obvious lower bounds are $\Omega(m)$ and $\Omega(W)$, since in biconnected graphs every edge has to belong to at least one circuit of any basis. Extremal graph theory provides a non-trivial lower bound.

Theorem 4.1. *For any integer $h = 2 \bmod 4$ and $h \geq 6$, there is a graph $G_h(n)$ with n nodes and $m = hn/2$ edges, such that any cycle basis for $G_h(n)$ has length $\Omega(m \log n/ \log(m/n))$. In particular, there is a graph family where $m = \Theta(n)$ and any basis has length $\Omega(m \log n)$, and for any integer k, there is a graph family where $m = \Theta(n^{1+1/k})$ and any basis has length $\Omega(mk)$.*

Proof. For any integer h with $h = 2 \bmod 4$ and $h \geq 6$, there exists an infinite family of h-regular graphs, i.e., $m = hn/2$, in which every cycle has length $\Omega(\log n/ \log(m/n))$; see [16]. Since a basis consists of $m - n + 1$ circuits, any basis has length $\Omega(n \log n/ \log(m/n))$. For $h = 6$, we obtain graphs with $m = \Theta(n)$, for which every basis has length $\Omega(m \log n)$. For $h = n^{1/k}$, we obtain graphs with $m = \Theta(n^{1+1/k})$, for which every basis has length $\Omega(mk)$. □

Open Problem 9. Prove a non-trivial lower bound for weighted graphs.

4.1. *Weakly fundamental bases*

The first result for general graphs was given by Horton in 1987; Liebchen [3] observed that the construction yields not only an undirected basis but also a weakly fundamental basis. We generalize Horton's proof to yield an upper bound for weighted graphs.

Theorem 4.2 ([12,3]). *Every simple graph G has a W-basis of a length of at most $3(n-1)(n-2)/2$ and a weight of at most $2nW(\mathrm{MST}) + 2W$, where $W(\mathrm{MST})$ is the weight of a minimum spanning tree.*

Proof. We prove only the upper bound for weighted graphs. For the case of uniform weights, we have $W(\mathrm{MST}) = n - 1$ and $W = m$. This gives a bound of $2n^2 + 2n^2/2 = 3n^2$ for the uniform case.

Let T be an MST of G. The claim clearly holds for $n \leq 3$. So assume that G has more than three vertices and let v be a leaf of T. Our W-basis for G consists of two parts: first, a W-basis $B(G - v)$ of $G - v$ constructed recursively, and second, $d(v) - 1$ cycles passing through v. Observe that a basis for $G - v$ has cardinality $m - d(v) - (n - 2) = m - (n - 1) - (d(v) - 1)$ and hence we are adding the right number of cycles. The graph consisting of T plus the $d(v) - 1$ non-tree edges incident to v is planar. We form $d(v) - 1$ circuits by taking all but one face cycle of this planar graph. The resulting set of circuits is weakly fundamental; this follows from an argument analogous to the one used in the proof of Lemma 3.5.

It remains to argue the bound on the weight. By the induction hypothesis, $B(G - v)$ has a weight of at most $2(n-1) \cdot W(\mathrm{MST} - v) + 2W(G - v)$. The circuits added in the induction step have a combined weight of at most $2W(\mathrm{MST}) + 2W(v)$, where $W(v)$ denotes the sum of the weights of the edges incident to v. Thus the weight of the resulting basis is at most $2nW(\mathrm{MST}) + 2W$. □

The upper bound is tight. Consider the complete graph on n nodes. It has $m = n(n-1)/2$ edges. Since any circuit in any basis contains at least three edges, any cycle basis has a length of at least $3(m - n + 1) = 3(n-1)(n-2)/2$. For sparse graphs, a much better bound is possible. Rizzi [17] proved that every graph has a W-basis of length $O(m \log n)$ and that every weighted graph has a W-basis of weight $O(W \log n)$. The proof given here was found by T. Kavitha and R. Rizzi. The proof makes use of the fact that every graph of minimum degree three contains a logarithmic length cycle.

Lemma 4.3 ([18]). *Any graph with a minimal degree of at least three contains a cycle of a length of at most $2\lceil \log_2 n \rceil$. Moreover, such a cycle can be found in time $O(n)$.*

Proof. Let G be our graph and let v be an arbitrary vertex. Grow a breadth-first search tree rooted at v. As long as only tree edges are encountered, every vertex has at least two children. Thus if $2^0 + 2^1 + \cdots 2^k > n$, there must be at least one non-tree edge incident to a vertex of depth $k - 1$ and hence a cycle of length $2k$ exists. This proves the bound on the length of a shortest cycle. With respect to the time bound, we observe that the first non-tree edge encountered yields the desired cycle. □

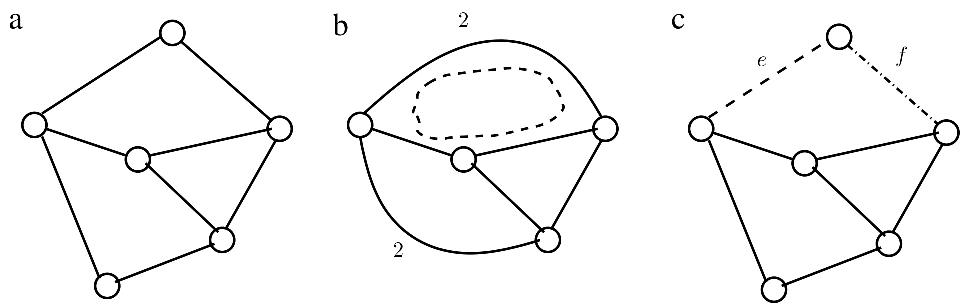


Fig. 16 – In (a), all edge weights are equal to one. In (b), the two indicated super-edges have weight two. The dashed line indicates a short cycle. It consists of three super-edges and has weight four. The heaviest super-edge has weight two. We delete its edges from the graph and make e a non-tree edge and f a tree edge or vice-versa, (c).

Theorem 4.4 ([17]). *Any weighted graph G with total weight W has a weakly fundamental basis of weight $O(W \log n)$. Such a basis can be determined in time $O(nm)$.*

Proof. We build the basis and a spanning tree concurrently. Initially, the basis and the spanning tree are empty. Let G be our current graph, which is initially set to the input graph. If G is empty, we stop. If G has a vertex of degree zero, we delete the vertex, and if G has a vertex of degree one, we delete the vertex and add the incident edge to the spanning tree. So assume that every vertex has degree two or more. We call a maximal path whose interior vertices have degree two a super-edge; an edge whose endpoints both have degree three or more is also a super-edge. The weight of a super-edge is the sum of the weights of the edges forming the super-edge. The endpoints of super-edges have degree three or more in G; see Fig. 16. The graph consisting of the vertices of degree three or more, and the super-edges joining them, contains a circuit $\overline{C}$ consisting of $O(\log n)$ super-edges. Let p be the heaviest super-edge in $\overline{C}$ and let C be the cycle in G represented by $\overline{C}$. Then, $w(C) = O(w(p) \log n)$. We add C to our basis. We also delete all edges belonging to p from G, designate an arbitrary edge of p as a non-tree edge, and add all other edges of p to T. If p consists of k edges, m decreases by k and n decreases by $k-1$. So ν decreases by 1 as it should.

The basis constructed in this way is weakly fundamental because the edge of p designated as a non-tree edge is not used in any cycle constructed later. Also, its weight is $O(W \log n)$ because the cost of the cycle added in an iteration is at most $O(\log n)$ times the weight of the edges deleted in this iteration. □

In the case of uniform weights, Theorem 4.4 establishes the existence of a weakly fundamental basis of length $O(m \log n)$. This is tight for graphs with $m = O(n)$ edges, as Theorem 4.1 asserts the corresponding lower bound. Kaufmann and Michail [19] have recently shown that the lower bound can also be matched for larger values of m. The improvement exploits the fact that graphs with at least $n^{1+1/k}$ edges contain a cycle of a length of at most $2k$ – see [20]. We now proceed as follows. As long as $m \geq n^{1+1/k}$ for a constant k, still to be determined, we find cycles of a length of at most $O(2k)$. We delete one of its edges and charge the cost of the cycle to it. As soon as $m \leq n^{1+1/k}$, we switch to the construction in Theorem 4.4. We construct cycles consisting of $O(\log n)$ super-edges, delete the edges in the heaviest super-edge, and charge $O(\log n)$ to each edge removed. The total charge is

$$O(mk + n^{1+1/k} \log n) \overset{k=2(\log n)/\log(m/n)}{=} O\left(m \frac{\log n}{\log(m/n)}\right).$$

Theorem 4.5 ([19]). *Every graph has a weakly fundamental basis of length $O(m \log n / \log(m/n))$. For $m = \Theta(n^{1+1/k})$, the bound is $O(mk)$ and for $m = n \log^c n$ and positive constant c, the bound is $O(m \log n / \log\log n)$. Finally, for $m = cn$, the bound is $O(m \log n)$.*

In the non-uniform case, a similar improvement is not possible, as the following example shows. Consider a graph $G = G_1 + G_2$, where G_1 is the complete graph on $n/2$ vertices and G_2 is a graph with $m = O(n/2)$ vertices and girth $\Omega(\log n)$. The edges of G_1 have weight zero and the edges of G_2 have weight one. Then, any basis of G has weight $\Omega(W \log n)$ and G has $\Omega(n^2)$ edges. Thus the bound of Theorem 4.4 cannot be improved for dense graphs and general weight functions.

We close our discussion of weakly fundamental bases with some remarks on planar graphs. Every planar graph has a 2-basis and these are weakly fundamental by Lemma 3.5. Thus every planar graph has a W-basis of length $O(n)$ and weight $O(W)$.

4.2. *Fundamental bases*

Upper bounds for strictly fundamental bases are obtained by constructing spanning trees of small diameter or, more generally, spanning trees of small stretch. Clearly, a spanning tree T of diameter D or with $\sum_{e=(u,v)\in E} d_T(x, y)/m \leq D$ gives rise to an F-basis of length $O(Dm)$. Here, $d_T(x, y)$ is the length of the path in T connecting x and y. We review results for planar graphs and for general graphs. The constructions make use of graph separators and graph partitions with suitable properties.

Definition 4.1. A set $S \subset V$ is an (α, β)-separator if $|S| \leq \beta\sqrt{n}$ and any connected component of $G - S$ contains no more than αn vertices.

Lemma 4.6 ([21]). *Any biconnected planar graph with n vertices, m edges, and maximal face size ϕ has an (α, β)-separator with $\alpha = 2/3$ and $\beta = 2\sqrt{\phi/2}$. Moreover, the separator constitutes a simple cycle and is thus called a simple cycle separator (SCS).*

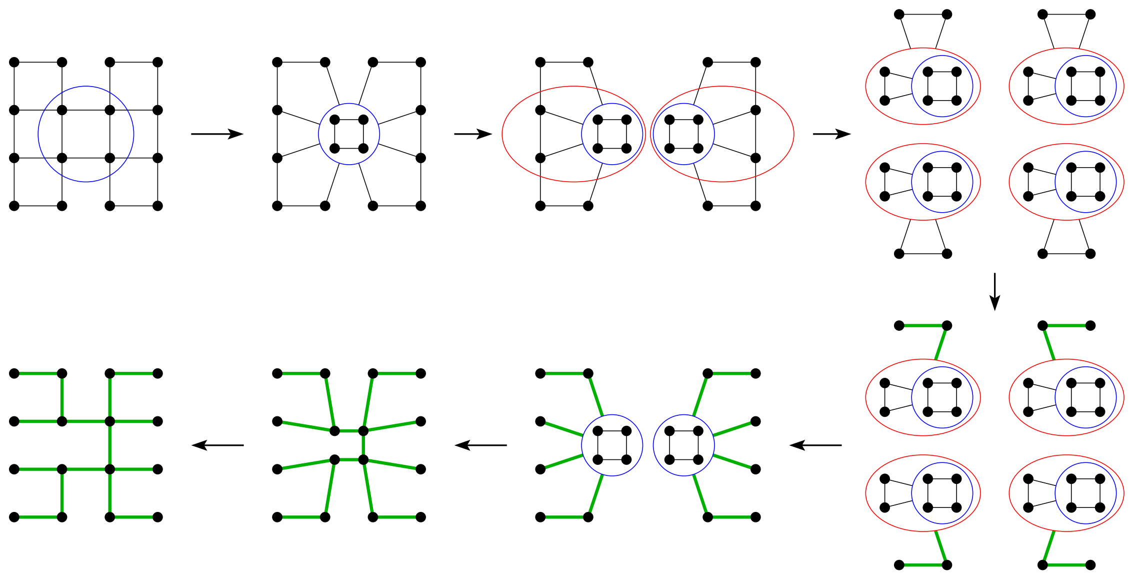


Fig. 17 – Stern and Vavasis's construction of an F-basis for planar graphs.

Cycle separators are the basis for the following theorem.

Theorem 4.7 ([22]). *Any planar graph G with maximal face size ϕ has an F-basis of length $O(n\sqrt{\phi n})$.*

Proof. We may assume that G is biconnected. Fig. 17 illustrates the construction. Let S be a simple cycle separator of size $\beta\sqrt{n}$ in G. We contract S into a single vertex v. Clearly, v becomes an articulation point in the resulting graph. Let $G_1, G_2, \ldots, G_k$ denote the components that would result if v were deleted. We make k copies v_1 to v_k of v, one for each component, and connect v_i with v's neighbors in G_i. Each G_i has at most αn vertices and the maximal face size is no more than ϕ. A spanning tree of G is obtained by taking the cycle S minus one edge plus spanning trees of the components. The spanning trees of the components are constructed recursively. We stop when the components have constant size; any spanning tree can be used for the constant size components.

Let $D(n)$ be the diameter of the spanning tree constructed in this way. Then $D(n) \leq O(\sqrt{\phi n}) + D(\alpha n) = O(\sqrt{\phi n})$. □

Outerplanar graphs have strictly fundamental bases of linear size [15].

Theorem 4.8. *For grid graphs of fixed dimension the minimal length of a fundamental basis is $\Theta(n \log n)$.*

The upper bound for two-dimensional grids was first shown by Stern and Vavasis [22]. A simplified construction that, in addition, applies to any fixed dimension was found by Alon et al. [23]. It is illustrated in Fig. 18 and yields a basis of length no more than $(4/3)n \log n$ as shown by Köhler et al. [24]. The lower bound was also established by Alon et al. [23]; Köhler et al. [24] paid attention to the constant factor and proved, using a different method, that any strictly fundamental basis for the planar grid has a length of at least $(1/12)n \log_2 n - O(n)$.

The first upper bound on the length of *strictly fundamental cycle bases* in general graphs was given by Alon et al. [23].

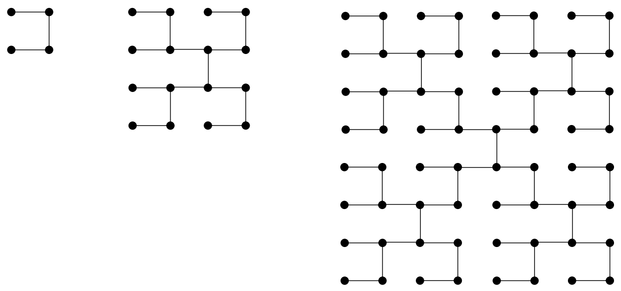


Fig. 18 – A spanning tree for d-dimensional grid graphs with length 2^i in all dimensions [23]. The construction is shown for $d = 2$. If $i = 1$, an optimal spanning tree for the structure is returned. If $i > 1$, the graph is partitioned into 2^d cubes of length 2^{i-1} and trees for the subgraphs are constructed recursively. The set of 2^d vertices in the center of the graph is connected such that they form the same tree that is used at the base of the recursion.

We follow the very descriptive explanation of their technique by Peleg [25]. The construction relies on partitioning a given graph into *clusters* such that the diameter of the clusters and the number of edges between clusters (*intercluster edges*) are controlled at the same time.

Lemma 4.9 ([25, p. 153]). *Given an unweighted graph $G = (V, E)$, $|V| = n$, and a parameter $x > 1$, there is a partition P of G into clusters C_i such that:*

1. *the radius of each cluster is at most $x \ln m$, and*
2. *the number of intercluster edges is at most m/x.*

Proof. The clusters are grown one by one. As long as there is a vertex not assigned to any cluster, choose one such vertex and grow a cluster C around it in discrete steps. Initially, C consists only of the vertex. Let $E_{out}(C)$ be the set of edges with exactly one endpoint in C, let N_{out} be the endpoints outside C of the edges in $E_{out}(C)$, and let $E_{in}(C)$ be the edges with both

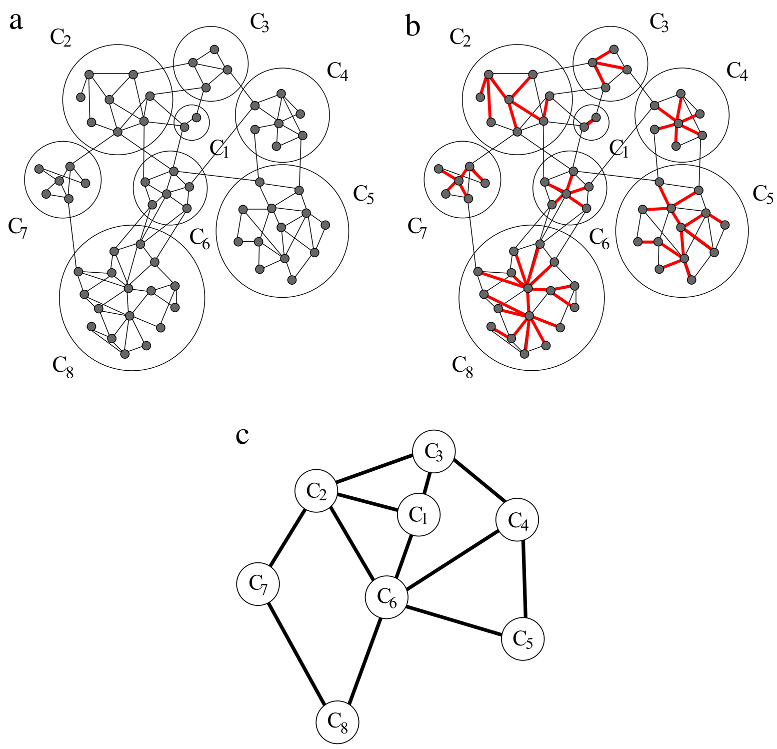


Fig. 19 – Alon et al.'s approach to constructing a spanning tree with average stretch in $\exp(O(\sqrt{\log n \log\log n}))$. Shown is the first level: (a) A partition in 8 clusters, C_1 to C_8, with the properties described in Lemma 4.9; (b) Red edges denote the spanning tree with radius $\leq x \ln m$ for each cluster. All red edges are part of the resulting tree T; (c) Each cluster is contracted to one vertex, possibly introducing multiple edges between clusters. The resulting graph has less than m/x edges. This graph is the starting point for the next level. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

endpoints in C. We add N_{out} to C if $|E_{out}(C)|/|E_{in}(C)| \geq 1/x$. If $|E_{out}(C)|/|E_{in}(C)| < 1/x$, the growth of C is stopped, C is added to the partition and deleted from G, and the next cluster is grown.

Clearly, any edge of G is contained in at most one cluster. Thus the number of intercluster edges is at most m/x. Consider the growth of any particular cluster C. We start with a single vertex v and no edge. In the first iteration, all neighbors of v (that are not assigned to any previous cluster) are added to the cluster. Let m_i be the number of edges added in the i-th iteration. Then $m_i \geq (m_1 + \cdots + m_{i-1})/x$. For the analysis of the growth of the m_i's, assume equality. Then $m_i - m_{i-1} = m_{i-1}/x$ and hence $m_i = (1+x)m_{i-1}/x$. We conclude that

$$m_1 + \cdots + m_i \geq m_i = \Omega\left(\left(\frac{1+x}{x}\right)^i\right).$$

Thus $i \leq (\ln m)/\ln(1 + 1/x) \leq x \ln m$ and we have also established the first property. □

We now come to the construction of the spanning tree. Fig. 19 illustrates the construction. Let P be a partition of $G_1 = G$ as described in the above theorem. For every cluster C_i, let T_i be a spanning tree of diameter $2x \ln m$. Such a tree exists by construction. The union of the T_i form a forest F in G. Any intracluster edge, and there are at most m of them, will give rise to a fundamental circuit of length no greater than $1 + 2x \ln m$. Only the m/x intercluster edges can give rise to longer fundamental circuits.

We contract every cluster C_i to a single vertex v_i and obtain the multi-graph G_2 formed by the intercluster edges. We apply the theorem to G_2 and obtain spanning trees of diameter $2x \ln(m/x) \leq 2x \ln m$ for the clusters of G_2. We add these spanning trees to the forest F. Consider any intracluster edge of G_2. It gives rise to a cycle of length $1 + 2x \ln m$ in G_2. With respect to F, this cycle may have a length up to $(1 + 2x \ln m)^2$ since any vertex representing a cluster of G_1 must be expanded to a path of length $1 + 2x \ln m$. We conclude that we might have m/x fundamental circuits of length $(1 + 2x \ln m)^2$. There are at most m/x^2 intercluster edges in G_2.

The construction continues until graphs of constant size are obtained. The recursion depth is at most $\log_x m$. The total length of the fundamental circuits constructed in this way is

$$\sum_{0 \leq i \leq \log_x m} \frac{m}{x^i}(1 + 2x \ln m)^{i+1} \approx mx(2 \ln m)^{\log_x m}.$$

With $x = \exp(c(\sqrt{\ln n \ln \ln n}))$ for an appropriate constant c, we obtain:

Theorem 4.10 ([23], [25, p. 215]). *Every multi-graph has a strictly fundamental basis of length $m \exp(O(\sqrt{\log n \log \log n}))$.*

A much improved result has been obtained recently.

Theorem 4.11 ([26]). *Every graph has a strictly fundamental cycle basis of length $O(W \log^2 n \log \log n)$.*

Table 3 – Polynomial time algorithms for undirected and directed minimum cycle bases. ω denotes the exponent of matrix multiplication.

Exact algorithms, nonnegative weights			
Undirected bases		Directed bases	
Deterministic	Monte Carlo	Deterministic	Monte Carlo
$O(\frac{m^2 n}{\log n} + mn^2)$ Theorem 5.11	$O(m^\omega)$ Theorem 5.27	$O(m^3 n)$ Theorem 5.12	$O(m^\omega)$ Theorem 5.28
Exact algorithms, conservative weights			
Undirected bases		Directed bases	
Deterministic		Deterministic	Monte Carlo
$O(n^3 \log n + \frac{m^2 n}{\log n} + mn^2)$ Theorem 5.14		$O(m^3 n)$ Theorem 5.15	$O(n^3 \log n + m^2 n)$ Theorem 5.16
$(2k-1)$-approximation, integer $k > 1$, nonnegative weights			
Undirected and directed bases			
Monte Carlo			
$O(mn^{1+1/k} + \min(m, n^{1+1/k})^\omega)$, Theorem 5.39			
Exact algorithms, planar graphs, nonnegative weights			
Undirected bases and directed bases			
$O(n^2)$, Theorem 5.33			

The key ingredient for the improved result is a more refined partitioning procedure, called *star-decomposition*. We refer the reader to [26] for details. We observe in passing that $\exp(O(\sqrt{\ln n \ln \ln n})) = o(n^\epsilon)$ for any $\epsilon > 0$ and hence even for planar graphs, the bounds given in Theorems 4.10 and 4.11 are better than the bound given in Theorem 4.7.

For dense graphs with $m = \Theta(n^2)$, optimal bounds can be achieved. As early as 1982, Deo et al. [27] had conjectured that every simple graph has a fundamental basis of length $O(n^2)$. It took 25 years to prove the conjecture.

Theorem 4.12 ([28]). *Every simple graph on n vertices has a fundamental cycle basis of length $O(n^2)$.*

Proof. Abraham et al. [29] showed that any graph[5] G with n vertices contains a spanning tree T with constant average stretch, averaged over all pairs of vertices, i.e.,

$$\sum_{x,y \in \binom{V}{2}} \frac{d_T(x,y)}{d_G(x,y)} = O(n^2).$$

Here, $d_G(x,y)$ and $d_T(x,y)$ are the distance between x and y in G and T, respectively. Restricting the sum to the edges of G establishes

$$\sum_{e=(x,y)\in E} d_T(x,y) = O(n^2).$$

Since the length of a fundamental cycle closed by a non-tree edge $e = (x,y)$ is $d_T(x,y) + 1$, the theorem follows. $\square$

Open Problem 10. Improve upon Theorem 4.11 or prove a lower bound that is asymptotically larger than $W \log n$ ($m \log n$ in the uniform case).

[5] The result even holds for weighted graphs; we only need it for unweighted graphs here.

5. Polynomial time algorithms for minimum cycle bases

We will now present deterministic and randomized polynomial time algorithms for computing undirected and directed minimum cycle bases. The deterministic algorithms have running time $\Omega(m^2 n/\log n + mn^2)$, and the randomized algorithms have running time $\Omega(m^\omega)$ and hence cannot be used for very large graphs. Therefore, we will also present techniques for computing approximate minimum cycle bases. Table 3 contains a summary of the best running times. The hard variants of the minimum cycle basis problem will be discussed in Section 6.

Open Problem 11. Most algorithms discussed in this chapter have space requirement $\Omega(m^2)$. Are there algorithms with reduced space requirement (and maybe increased running time) and algorithms for external memory?

Recall that a directed basis is a set of ν circuits that are independent over $\mathbb{Q}$ and that an undirected basis is a set of ν circuits that are independent over $GF(2)$. We use κ to denote either $\mathbb{Q}$ or $GF(p)$, where p is a prime, and formulate most of the algorithms in terms of the field κ.

5.1. *The greedy algorithm and the Horton set*

A minimum (directed or undirected) cycle basis can be constructed by a simple greedy algorithm. This is almost a direct consequence of Theorem 3.9. We start with an empty basis and process the circuits of G in order of nondecreasing weight; ties are broken arbitrarily. We add a circuit to the partial cycle basis if it is linearly independent of the circuits in the partial basis. We continue until we have obtained ν linearly independent circuits. Checking linear independence can be easily done by Gaussian elimination.

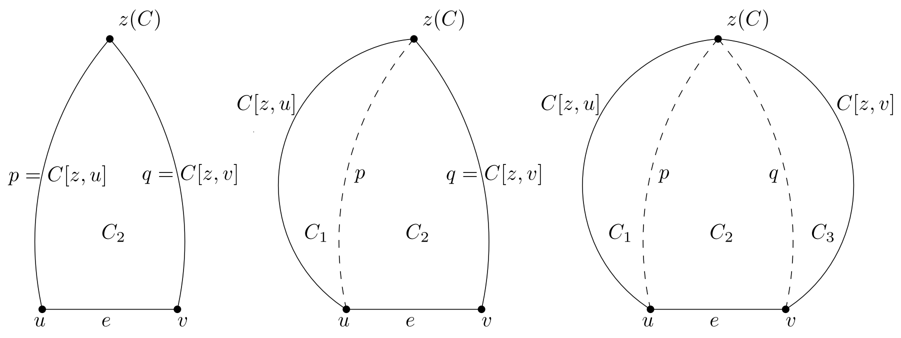


Fig. 20 – The three cases in the proof of Lemma 5.2 (not showing symmetrical cases).

Theorem 5.1. *The greedy algorithm constructs a minimum weight cycle basis.*

Proof. We could appeal to the fact the the greedy algorithm works for matroids [30] and that the set of circuits of a graph form a matroid. We prefer to give a self-contained proof.

Assume that the greedy algorithm does not construct a minimum weight basis and consider the first time in the execution of the algorithm that the partial basis cannot be extended to a minimum weight basis. Say this happens after the addition of the circuit C. Before adding C, we had a partial basis B that could be extended to a minimum weight basis B_{opt}. Let us write C as a linear combination of the circuits in B_{opt}, say $C = \sum_{D \in B_{opt}} \lambda_D D$. Since C is linearly independent of B, there must be a $D \in B_{opt} \setminus B$ with $\lambda_D \neq 0$. Also, since this D is linearly independent of B, we must have $w(C) \leq w(D)$. Thus, $B_{opt} - D + C$ is also a minimum weight basis, which is a contradiction. □

Since a graph may have an exponential number of circuits, the performance of the greedy algorithm in its basic form is miserable. Horton [12] showed that the search for a basis can be restricted to a set of $O(nm)$ circuits. For a vertex v, let T_v be a shortest path tree in G rooted at v. For any two nodes, u and v, we use p_{uv} to denote the shortest path from u to v contained in T_u. We do not assume $p_{uv} = p_{vu}$ or any other consistency requirement.

Definition 5.1 ([12]). For a vertex v and an edge $e = (x, y)$ such that the tree paths from v to x and y, respectively, do not share first edges (this includes the case where one of them is empty), let $C_{v,e}$ be the cycle consisting of the tree path from v to x in T_v, followed by e, followed in turn by the reversal of the tree path from v to y. The *Horton set* $\mathcal{H}$ consists of all such circuits $C_{v,e}$.

The Horton set is really a multi-set, i.e., the mapping $(v, e) \mapsto C_{v,e}$ is not necessarily injective. In fact, we will show in Section 5.7 that if shortest paths are chosen carefully, it is highly non-injective. This will lead to improved algorithms. For now, we content ourselves to show that $\mathcal{H}$ contains an MCB. For a circuit C, let $z(C) \in V \cap C$ be a vertex that minimizes the number of non-tree edges of C w.r.t T_v. We call $z(C)$ the *base node* of C.

Lemma 5.2 ([12,31,32]). *$\mathcal{H}$ contains a minimum cycle basis. Moreover, when the greedy algorithm is executed with $\mathcal{H}$, it extracts a minimum cycle basis.*

Proof. Consider the greedy algorithm run on the set of all circuits. Circuits are ordered lexicographically according to

(weight of C, number of edges outside $T_{z(C)}$, number of edges in C).

Observe that the circuits in $\mathcal{H}$ have second coordinates equal to one and hence come first among cycles of equal weight.

Let C be the first circuit outside $\mathcal{H}$ that is selected by the greedy algorithm. Let $z = z(C)$ and let $e = (u, v)$ be a non-tree edge (with respect to T_z) on C. Write $C = C_{z,u} \circ (u, v) \circ C_{v,z}$ and let p and q be the tree paths in T_z connecting z to u and v, respectively. The cost of p is at most the cost of either cycle path from z to u and the cost of q is at most the cost of either cycle path from z to v. Consider the cycles $C_1 = C_{z,u} \circ p^{rev}$, $C_2 = p \circ e \circ q^{rev}$, $C_3 = q \circ C_{v,z}$ (Fig. 20). The weight of C_1, C_2 and C_3 is at most the weight of C and $C = C_1 + C_2 + C_3$.

We now distinguish cases. Assume first that e is the only non-tree edge on C. Then $C_{z,u}$ and $C_{v,z}$ are contained in T_z. Since C is a circuit and z lies on C, the tree paths to u and v in T_z cannot have a common first edge; thus $C = C_{z,e} \in \mathcal{H}$, which is a contradiction. Assume next that C contains more than one non-tree edge. Then at least one of the cycles, C_1 or C_3, is non-trivial. Also, with respect to T_z all three cycles have at least one fewer non-tree edge than C and hence this is also true of their respective base vertices.

Thus all three cycles are considered by the greedy algorithm before C. Also, at least one of them is independent of the current basis, so it was independent at the time it was considered and hence should have been added. This either contradicts our definition of C (first cycle outside $\mathcal{H}$ added to the basis) or the operation of the greedy algorithm (a cycle not added even though it is independent). □

For undirected cycle bases, Lemma 5.2 was first shown by Horton [12]. Mehlhorn and Michail [31] observed that it suffices to consider a slightly smaller set of circuits. Let Z be a *feedback vertex set* of G, i.e., any circuit in G must contain at least one vertex in Z. It suffices to consider the circuits $C_{z,e}$ where $z \in Z$ and the paths in T_z to the endpoints of e do not have a common first edge. Computing a minimum feedback vertex set is known to be $\mathcal{APX}$-hard, however, a 2-approximation can be computed efficiently [33]. Moreover, Liebchen and Rizzi [32] extended Lemma 5.2 to directed bases.

Lemma 5.2 implies polynomial time algorithms for finding a minimum undirected and directed cycle basis. We first construct $\mathcal{H}$ by solving n single-source shortest-path problems. In the case of non-negative weights, this amounts to n runs of Dijkstra's algorithm and takes $O(nm + n^2 \log n)$ time. We treat the case of conservative weights in Section 5.5. The Horton set consists of $O(mn)$ circuits and a partial basis consists of at most ν circuits. For any circuit in $\mathcal{H}$, we must decide whether it is independent of the current partial basis. Gaussian elimination performs this task with $O(\nu m) = O(m^2)$ arithmetic operations per circuit. Let Γ be the cycle matrix of the current basis. We keep the non-tree part of Γ in upper triangular form. Then independence of a circuit can be checked with $O(\nu m)$ arithmetic operations and, in the case of independence, the cycle matrix can be extended by an additional column with the same number of arithmetic operations. We conclude that a minimum basis can be constructed with $O(m^3 n)$ arithmetic operations. The number of arithmetic operations can be reduced to $O(m^\omega n)$ [34,32], where ω denotes the exponent of matrix multiplication, i.e., $m \times m$ matrices can be multiplied with $O(m^\omega)$ arithmetic operations. It is known that $\omega < 2.376$.

Arithmetic operations over $GF(2)$ take constant time. We conclude that a minimum weight undirected cycle basis of a nonnegatively weighted graph can be computed in time $O(m^\omega n)$. For directed cycle bases we appeal to Theorem 3.8. Let P be a set of m primes of value at least n. The primes $p \in P$ are in $O(m \log m)$ and hence arithmetic in $GF(p)$ takes constant time. Computing a minimum $GF(p)$-basis for all $p \in P$ is guaranteed to find a minimum directed basis. This takes $O(m^{1+\omega} n)$ time. Computing a minimum $GF(p)$-basis for a random $p \in P$ takes $O(m^\omega n)$ time. It finds a minimum directed basis with probability at least 1/2.

5.2. *De Pina's approach*

We will describe an alternative approach for computing minimum cycle bases introduced by de Pina [14] and later refined by Berger et al. [35], Kavitha et al. [36], Hariharan et al. [37], Kavitha et al. [38], Mehlhorn and Michail [31] and Amaldi et al. [39]. Operating in phases, it starts with an empty set of circuits, and adds one circuit per phase. It does *not necessarily* add the circuits in order of increasing weight. This increased flexibility results in faster running time.

For two vectors, C and S, we use $\langle C, S\rangle$ to denote their inner product. Two vectors are *orthogonal* to each other if their inner product is zero. The following theorem is the basis of de Pina's approach; the version given here is due to Mehlhorn and Michail [31] and refines Theorem 3.22.

Theorem 5.3 ([14,31]). *Circuits $C_1, \ldots, C_\nu$ form a minimum κ-basis, where $\kappa = \mathbb{Q}$ or $\kappa = GF(p)$, if there are vectors $S_1, \ldots, S_\nu \in \kappa^E$ such that for all i, $1 \le i \le \nu$, the following hold:*

Prefix orthogonality: *$\langle C_j, S_i\rangle = 0$ for all $1 \le j < i$.*
Non-orthogonality: *$\langle C_i, S_i\rangle \neq 0$.*
Shortness: *C_i is a minimum weight circuit in $\mathcal{H}$ with $\langle C_i, S_i\rangle \neq 0$.*

Proof. We first show linear independence. Let $C := \sum_i \lambda_i C_i$ be a non-trivial linear combination and assume that i_0 is the largest index for which $\lambda_i \neq 0$. Then $\langle C, S_{i_0}\rangle = \lambda_{i_0}\langle C_{i_0}, S_{i_0}\rangle \neq 0$.

We next show that the circuits form a minimum cycle basis of G. Assume otherwise. Then consider the smallest i such that $C_1, \ldots, C_i$ are not contained in any minimum cycle basis consisting only of circuits in the Horton set $\mathcal{H}$. Let B be a minimum weight basis consisting of circuits in the Horton set that contains C_1 to C_{i-1}. We may write C_i as a linear combination of the circuits in B, $C_i = \sum_{C \in B} \lambda_C C$. Since $\langle C_i, S_i\rangle \neq 0$, there exists some $C \in B$ with $\langle C, S_i\rangle \neq 0$. Since C_i is a minimum weight cycle in $\mathcal{H}$ with $\langle C_i, S_i\rangle \neq 0$, we have $w(C_i) \le w(C)$. Also $C \neq C_j$ for $j < i$ since $\langle C_j, S_i\rangle = 0$ for $j < i$.

Let $B' = B \cup \{C_i\} \setminus \{C\}$; B' is a basis according to Theorem 3.9 and $w(B') \le w(B)$. So B' is also a minimum cycle basis. It consists only of circuits in $\mathcal{H}$ and contains C_1 to C_i, which is a contradiction. □

Algorithm 1 An algebraic framework for computing a minimum cycle basis.

```
1: let $T$ be an arbitrary spanning tree.
2: for $i \leftarrow 1, \ldots, \nu$ do
3:     Determine a non-zero vector $S_i$ with $S_i(e) = 0$ for $e \in T$ and orthogonal to $C_1$ to $C_{i-1}$.
4:     Compute a minimum weight cycle $C_i \in \mathcal{H}$ with $\langle C_i, S_i\rangle \neq 0$.
5: end for
```

Theorem 5.3 leads to Algorithm 1. The algorithm operates in ν phases. In each phase, a non-zero vector S, orthogonal to all cycles in the partial basis, is determined and then a shortest circuit $C \in \mathcal{H}$ with $\langle S, C\rangle \neq 0$ is computed and added to the basis. We still need to show that there is always a vector S of the desired form and a circuit to add.

Lemma 5.4. *Let T be a spanning tree of G. For each phase i, $1 \le i \le \nu$: There is a non-zero vector $S_i \in \kappa^E$ such that $\langle S_i, C_j\rangle = 0$ for $j < i$ and $S_i(e) = 0$ for $e \in T$ and there is at least one cycle $C \in \mathcal{H}$ with $\langle C, S_i\rangle \neq 0$.*

Proof. Let C_j' be the restriction of C_j to $N := E \setminus T$. The space spanned by C_1' to C_{i-1}' has dimension $i - 1$ and $i - 1 < \nu$. Thus there is a vector $S' \in k^N$ with $\langle C_j', S'\rangle \neq 0$ for $j < i$. Define S_i by $S_i(e) = S'(e)$ for $e \in N$ and $S_i(e) = 0$ for $e \in T$.

Let e be any edge with $S_i(e) \neq 0$ and let C_e be the fundamental circuit defined by e. Then $\langle C_e, S_i\rangle = S_i(e) \neq 0$. Since the Horton set contains a basis, C_e can be written as a linear combination of circuits in $\mathcal{H}$. Thus, there must be at least one circuit $C \in \mathcal{H}$ with $\langle C, S_i\rangle \neq 0$. □

The requirement that $S_i(e) = 0$ for all $e \in T$ in line (3) of Algorithm 1 is essential. Assume G contains a bridge e. Then the unit vector S_i with $S_i(e) = 1$ is trivially orthogonal to the circuits C_1 to C_{i-1}. However, line (4) will fail because there is no circuit that is non-orthogonal to this S_i. In the next sections we describe how to implement the two main steps of Algorithm 1.

5.3. *Maintaining the orthogonal space*

The vector S_i is a non-trivial solution of the linear system $\langle C_j, S_i\rangle = 0$ for $1 \le j < i$ and $S_i(e) = 0$ for all $e \in T$. The naive way would be to solve this linear system using Gaussian elimination with $O(m^\omega)$ arithmetic operations. Since we need to solve one linear system per phase, the total number of arithmetic operations required would be $O(m^{1+\omega})$.

However, the linear systems to be solved are not independent. Each phase adds one additional equality. De

Algorithm 2 Maintaining a Basis of the Orthogonal Space

```
1: Initialize $S_j$ by $S_j(e_i) = \delta_{ij}$ for $1 \le j \le \nu$ and $1 \le i \le m$.
2: for $i \leftarrow 1, \ldots, \nu$ do
3:     Compute a minimum weight cycle $C_i \in \mathcal{H}$ with $\langle C_i, S_i \rangle \neq 0$
4:     for $j \leftarrow i+1, \ldots, \nu$ do
5:         $S_j = S_j - \frac{\langle C_i, S_j \rangle}{\langle C_i, S_i \rangle} S_i$
6:     end for
7: end for
```

Pina [14] and later Berger et al. [40] observed that it pays to maintain a basis of the solution space of this linear system. The basis is easily updated from one phase to the next.

Let T be an arbitrary spanning tree of G and let e_1 to e_ν be the non-tree edges. We set $S_i(e_i) = 1$ and $S_i(e_j) = 0$ for $j \neq i$. This corresponds to the standard basis of the space κ^N. At the beginning of phase i, we have $S_i, S_{i+1}, \ldots, S_\nu$ that form a basis of the space $\mathcal{C}^\perp$ of all vectors S that are orthogonal to circuits $C_1, \ldots, C_{i-1}$ and have $S(e) = 0$ for all $e \in T$. We use S_i to compute C_i (see Section 5.4) and update vectors $\{S_{i+1}, \ldots, S_\nu\}$ to a basis $\{S'_{i+1}, \ldots, S'_\nu\}$ of the subspace of $\mathcal{C}^\perp$ that is orthogonal to C_i. The update step is as follows. For $i+1 \le j \le \nu$, let

$$S'_j = S_j - \frac{\langle C_i, S_j \rangle}{\langle C_i, S_i \rangle} S_i.$$

Lemma 5.5. *The set $\{S'_{i+1}, \ldots, S'_\nu\}$ forms a basis of the subspace orthogonal to $\{C_1, \ldots, C_i\}$.*

Proof. We will first show that $S'_{i+1}, \ldots, S'_\nu$ are orthogonal to $C_1, \ldots, C_i$. Let $j \ge i+1$ and $\ell \le i$. We have

$$\langle S'_j, C_\ell \rangle = \langle S_j, C_\ell \rangle - \frac{\langle C_i, S_j \rangle}{\langle C_i, S_i \rangle} \langle S_i, C_\ell \rangle.$$

For $\ell < i$, $\langle S_j, C_\ell \rangle = \langle S_i, C_\ell \rangle = 0$. For $\ell = i$, the terms on the right-hand side cancel.

Now we will show that $S'_{i+1}, \ldots, S'_\nu$ are linearly independent. Consider a linear combination

$$\mathbf{0} = \sum_{j \ge i+1} \lambda_j S'_j = \sum_{j \ge i+1} \lambda_j S_j - \left(\sum_{j \ge i+1} \lambda_j \frac{\langle C_i, S_j \rangle}{\langle C_i, S_i \rangle} \right) S_i.$$

Since the S_j, $j \ge i$, are independent, we conclude that $\lambda_j = 0$ for all j. □

Let us now bound the number of arithmetic operations. In each iteration, we update no more than ν vectors at a cost of $O(\nu)$ arithmetic operations each. Thus the total number of arithmetic operations is $O(\nu^3) = O(m^3)$. For undirected bases, this is also the running time.

The vector S_i is only needed in the i-th phase. In particular, the second half of the vectors is only needed in the second half of the computation. We can save time by not updating these vectors at all in the first half of the computation and then computing the cumulative effect of the first half of the computation. We will be able to use fast matrix multiplication for the cumulative update. We now give the details. Let $k = \lfloor \nu/2 \rfloor$. What is the effect of the first k phases on the vectors S_{k+1} to S_ν?

For column vectors v_1 to v_ℓ, we use $[v_1, \ldots, v_\ell]$ to denote the matrix with columns v_1 to v_ℓ. Let S_1 to S_ν denote our vectors before phase 1 and let S'_1 to S'_ν be the vectors after phase k. Then,

$$[S'_{k+1}, \ldots, S'_\nu] = [S_{k+1}, \ldots, S_\nu] - [S'_1, \ldots, S'_k]A$$

for some $k \times (\nu - k)$ matrix A. We want $\langle C_\ell, S'_j \rangle = 0$ for $1 \le \ell \le k$ and $k+1 \le j \le \nu$. Let $C = [C_1, \ldots, C_k]$. Then

$$\mathbf{0} = C^T[S'_{k+1}, \ldots, S'_\nu] = C^T[S_{k+1}, \ldots, S_\nu] - C^T[S'_1, \ldots, S'_k]A$$

and hence

$$A = (C^T[S'_1, \ldots, S'_k])^{-1} C^T[S_{k+1}, \ldots, S_\nu].$$

Since $\langle C_\ell, S'_i \rangle = 0$ for $1 \le \ell < i \le k$ and $\langle C_i, S'_i \rangle \neq 0$, the matrix $C^T[S_1, \ldots, S_k]$ is lower triangular with non-zero entries on the diagonal and hence invertible. We need to compute three matrix products and one matrix inversion. Each of them can be performed with $O(m^\omega)$ arithmetic operations. We conclude that the cumulative update of S_{k+1} to S_ν at the end of phase k requires only $O(m^\omega)$ arithmetic operations instead of the $\Theta(m^3)$ operations for the continuous update. We can carry this idea further by applying it recursively, for example, by not updating $S_{\lfloor k/2 \rfloor + 1}$ to S_k in the first $\lfloor k/2 \rfloor$ phases, but doing a bulk update of these vectors after phase $\lfloor k/2 \rfloor$. We thereby obtain Algorithm 3.

Algorithm 3 Maintaining a Basis of the Orthogonal Space with Bulk Updates

```
1: Initialize $S_j$ by $S_j(e_i) = \delta_{ij}$ for $1 \le j \le \nu$ and $1 \le i \le m$.
2: MinimumCycleBasis(1, $\nu$)
3: where
4: procedure MINIMUMCYCLEBASIS($\ell, u$)  ▷ Adds Circuits $C_\ell$ to $C_u$
5:     if $\ell = u$ then
6:         compute a minimum weight cycle $C_i \in \mathcal{H}$ with $\langle C_i, S_i \rangle \neq 0$;
7:     else
8:         $k \leftarrow \lfloor (\ell + u)/2 \rfloor$;
9:         MinimumCycleBasis($\ell, k$);
10:        $C \leftarrow [C_\ell, \ldots, C_k]$;
11:        $A \leftarrow (C^T[S_\ell, \ldots, S_k])^{-1} C^T[S_{k+1}, \ldots, S_u]$;
12:        $[S_{k+1}, \ldots, S_u] \leftarrow [S_{k+1}, \ldots, S_u] - [S_\ell, \ldots, S_k]A$;  ▷ now $C^T[S_{m+1}, \ldots, S_u] = \mathbf{0}$
13:        MinimumCycleBasis($m+1, u$);
14:    end if
15: end procedure
```

Consider a call of procedure MimimumCycleBasis which is not innermost and let $r = u - \ell + 1$, $s = k - \ell + 1$ and $t = u - k$. In the update of the vectors S_{k+1} to S_u, we perform (s, m, s), (s, m, t), (s, s, t), (m, s, t) matrix multiplications,[6] one inversion of an $s \times s$ matrix, and one addition of two $m \times t$ matrices. If we split all matrices into blocks of $s \times s$ matrices and use fast matrix methods for the blocks, the update requires $O((m/s)s^\omega)$ arithmetic operations. The total number U of arithmetic operations for all updates follows the recursion

$$U(r) = \begin{cases} 0 & \text{if } r = 1 \\ O((ms^{\omega-1})) + U(s) + U(r-s) & \text{if } r > 1 \text{ and } s = \lceil r/2 \rceil. \end{cases}$$

[6] An (a, b, c) matrix multiplication multiplies an $a \times b$ matrix with a $b \times c$ matrix.

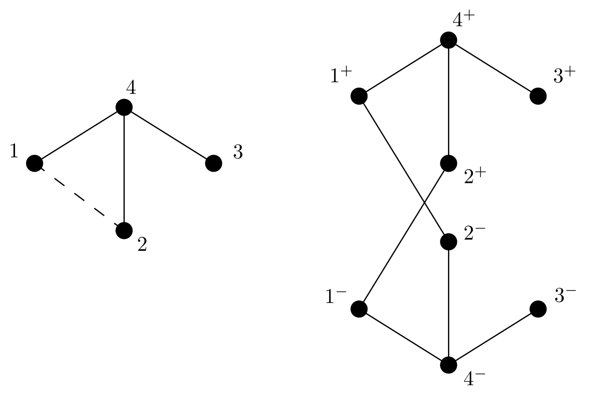


Fig. 21 – An example of the graph G_S, where $S = \{(1,2)\}$. Since the edge $(1,2)$ belongs to S we have the edges $(1^-, 2^+)$ and $(1^+, 2^-)$ connecting vertices on different sides. The edges not in S, i.e., $(1,4)$, $(2,4)$, and $(3,4)$ have copies on both sides.

This recurrence has solution $U(r) = O(mr^{\omega-1})$. In our outermost call, $r = \nu = O(m)$. We conclude that the total number of arithmetic operations in the update steps is $O(m^{\omega})$.

Lemma 5.6. *The total number of arithmetic operations performed in lines* 10 *to* 12 *of Algorithm* 3 *is $O(m^{\omega})$. In a computation over $GF(p)$ with $\log p = O(\log m)$, the time spent in lines* 10 *to* 12 *is $O(m^{\omega})$.*

5.4. *Computing the circuits*

We now come to the second main ingredient of the minimum cycle basis algorithm. Given a non-zero vector S, compute a minimum weight circuit C with $\langle S, C\rangle \neq 0$. We know from Theorem 5.3 that the search can be restricted to $\mathcal{H}$. We will exploit this fact in Sections 5.4 and 5.7. Now, we will show how to find C without this additional knowledge.

We first consider the undirected case and nonnegative edge weights and reduce the computation to n shortest-path computations. Over $GF(2)$, the vector S is zero-one and therefore corresponds to a subset of E; $\langle S, C\rangle \neq 0$ if and only if C uses an odd number of edges in S. The following construction is well known [41,42]. The *signed graph* G_S is defined from $G = (V, E)$ and S in the following manner. G_S has two copies for each vertex $v \in V$. Call them v^+ and v^-. Let $e = (u, v)$ be any edge of G. If $e \notin S$, we put the edges (v^+, u^+) and (v^-, u^-) into G_S and if $e \in S$, we put the edges (v^+, u^-) and (v^-, u^+) into G_S. In either case, the edges inherit the weight of e. Fig. 21 illustrates the construction. The vertices of G_S naturally split into a $+$ side and a $-$ side. Edges of G_S corresponding to edges in $E \setminus S$ connect vertices on the same side, and edges corresponding to edges in S connect vertices on opposite sides.

A path in G starting at a node v lifts to two paths in G_S, one starting in v^+ and one starting in v^-. The path ends on the other side if and only if it uses an odd number of edges in S. So a circuit passing through v and using an odd number of edges in S lifts to a simple path of the same weight connecting v^+ and v^-. The lifted path does not use both copies in G_S of an edge of G. Conversely, consider a path p connecting v^+ to v^- in G_S. It may use both copies of an edge of G. In our example, the path $\langle 3^+, 4^+, 1^+, 2^-, 4^-, 3^-\rangle$ uses both copies of $(3, 4)$. We split

$$p = \langle v^+, \dots, x^*\rangle\langle x^*, y^\dagger\rangle\langle y^\dagger \dots y^{-\dagger}\rangle\langle y^{-\dagger}, x^{-*}\rangle\langle x^{-*}, \dots, v^-\rangle$$

at the two copies of an edge, say (x, y) such that the "middle part" $q = \langle y^\dagger \dots y^{-\dagger}\rangle$ does not use both copies of any edge; q connects y^+ and y^- and $w(q) \leq w(p)$ since edge weights are nonnegative. We summarize the discussion in:

Lemma 5.7. *For each $v \in V$, let p_v be a minimum weight minimum cardinality path*[7] *from v^+ to v^- in G_S. Let v_0 be such that p_{v_0} has minimum weight among the paths p_v. Break ties in favor of the path containing fewer edges. Let $C = C_{v_0}$ be the projection of p_{v_0} into G. C is a minimum weight cycle in G using an odd number of edges in S.*

The computation of the path p_{v_0} can be performed by computing n shortest (v^+, v^-) paths, one for each vertex $v \in V$, each by Dijkstra's algorithm in G_S and taking their minimum, or by one invocation of an all-pairs shortest-paths algorithm in G_S. This computation takes $O(n(m + n \log n))$ time. Note that depending on the relation between m and n, we may choose which shortest-paths algorithm to use. For example, in the case when the edge weights are integers, or the unweighted case, it is better to use faster all-pairs shortest-paths algorithms than run Dijkstra's algorithm n times.

Computation over $GF(p)$: The signed graph technique extends to computations over $GF(p)$ [43]. The entries of the vector S are now in $\{0, \dots, p-1\}$. Accordingly, we have p levels and p copies v^0 to v^{p-1} of each edge. An edge $e \in E$ with $s = S(v)$ gives rise to edges (v^i, v^{i+s}) for $0 \leq i < p$. Superscripts are to be read modulo p, but everything else is as before. Because of the larger graph, the cost of the shortest-path computation is multiplied by p.

Hariharan et al. [37] were able to remove the factor of p in the running time. Consider a shortest-path computation starting at v^0. The algorithm outlined in the previous paragraph computes for each w and each $i \in \{0, \dots, p-1\}$ a shortest path to w^i. The improved algorithm computes for every w only two paths. Let i_0 be such that the path from v^0 to w^{i_0} is no longer than to any w^i and let i_1 be such that the path from v^0 to w^{i_1} is no longer than to any w^i with $i \neq i_0$. The algorithm computes the paths to w^{i_0} and w^{i_1} and this can be done in Dijkstra-time.

We will not go into more detail since the following section presents a simpler and faster approach which is, furthermore, the same for all $GF(p)$.

Labeled trees: We know from Theorem 5.3 that the search for a shortest circuit C_i with $\langle C_i, S_i\rangle \neq 0$ in line 6 of Algorithm 3 may be restricted to the circuits in $\mathcal{H}$. A compact representation of the circuits in $\mathcal{H}$ is given by the shortest-path trees T_v, $v \in V$. For $v \in V$, each edge $e = (x, y)$ connecting vertices in distinct subtrees of T_v gives rise to the circuit $C_{v,e} \in \mathcal{H}$.

How can we compute $\langle C_{v,e}, S_i\rangle$ efficiently? The idea [31] is to precompute most of the inner product. For any v and w, let

[7] A minimum weight minimum cardinality path from v^+ to v^- is a minimum weight path from v^+ to v^-. Among the minimum weight paths, it has a minimum number of edges.

$p_{v,w}$ be the path from v to w in T_v. We label w in T_v with $\ell_{v,w} = \langle p_{v,w}, S_i \rangle$. For fixed v, the labels $\ell_{v,w}$ can be computed in $O(n)$ arithmetic operations. It takes $O(n^2)$ arithmetic operations to label all trees. Once the labels are available, $\langle C_{v,e}, S_i \rangle$ can be computed with a constant number of arithmetic operations. If $e = (x, y)$,

$$\langle C_{v,e}, S_i \rangle = \ell_{v,x} + S_i(e) - \ell_{v,y}.$$

Lemma 5.8. *If the shortest-path trees T_v, $v \in V$, are available, the minimum weight cycle $C \in \mathcal{H}$ with $\langle C, S_i \rangle \neq 0$ can be found with $O(nm)$ arithmetic operations.*

5.5. *Computing shortest-path trees*

For nonnegative edge weights, we use Dijkstra's algorithm and obtain:

Lemma 5.9. *If edge weights are nonnegative, the shortest-path trees T_v, $v \in V$, can be computed in $O(n(m + n \log n))$ time.*

For conservative edge weights, heavier machinery needs to be used. It is known that computing all-pairs shortest paths in undirected graphs with real edge weights but no negative cycles can be computed by solving a sequence of general weighted matching problems.[8]

Lemma 5.10. *If edge weights are conservative, the shortest-path trees T_v, $v \in V$, can be computed in $O(n^2 m + n^3 \log n)$ time.*

Proof. The single-sink single-source shortest-path problem in a conservatively weighted undirected graph reduces to a weighted perfect matching problem in a graph with $O(n)$ vertices and $O(m)$ edges [30, page 278] and hence can be solved in $O(n(m + n \log n))$ time [44]. The construction of the perfect matching problem consists of n "searches"; each search takes $O(m + n \log n)$ time. The all-pairs shortest-path problem can be reduced to a perfect matching problem plus n^2 searches [30, page 279]. □

5.6. *Putting it together*

We can now put the pieces together.

Theorem 5.11 ([36,31]). *For nonnegative weight functions, a minimum weight undirected cycle basis can be computed in $O(m^2 n / \log n + mn^2)$ time.*

Proof. It takes $O(nm + n^2 \log n)$ time to compute the shortest-path trees (Lemma 5.9), $O(m^\omega)$ time (Lemma 5.6) to compute the S_i, $1 \le i \le \nu$, and $O(nm^2)$ time to determine the cycles C_i, $1 \le i \le \nu$. The total running time is $O(m^2 n)$.

Mehlhorn and Michail [31] have shown that word parallelism on words of $O(\log n)$ bits can be used to extract the cycles in $O(m^2 n / \log n)$ time at the cost of increasing preprocessing time to $O(mn^2)$. □

[8] In directed graphs with no negative cycles, one solves one single-source problem to obtain a potential function. The potential function is then used to obtain an equivalent problem with non-negative edge weights. This reduction does not work for undirected graphs. Also, observe that making an undirected graph bidirected will turn a negative edge into a negative cycle.

Theorem 5.12 ([37,31]). *For nonnegative weight functions, a minimum weight directed cycle basis can be computed in $O(m^3 n)$ time.*

Proof. According to Theorem 3.8, it suffices to compute the minimum $GF(p)$-basis for m primes larger than n. The best such basis is a minimum weight directed cycle basis.

For each fixed p, it takes $O(nm + n^2 \log n)$ time to compute the shortest-path trees (Lemma 5.9), $O(m^\omega)$ time (Lemma 5.6) to compute the S_i, $1 \le i \le \nu$, and $O(nm^2)$ time to determine the cycles C_i, $1 \le i \le \nu$. The total running time is $O(m^2 n)$ for each p and hence $O(m^3 n)$, altogether. □

Theorem 5.13 ([37,31]). *For nonnegative weight functions, a minimum weight directed cycle basis can be computed in $O(m^2 n)$ time with a probability of at least 1/2.*

Proof. According to Theorem 3.8, it suffices to compute the minimum $GF(p)$-basis for a prime p chosen randomly from a set of m primes larger than n. For such a prime the minimum $GF(p)$-basis can be computed in $O(m^2 n)$ time. □

Theorem 5.14. *For conservative weight functions, a minimum undirected cycle basis can be computed in $O(n^3 \log n + m^2 n / \log n + mn^2)$ time.*

Proof. Follows from Lemmas 5.10, 5.6 and 5.8, and the remark made in the proof of Theorem 5.11. □

Theorem 5.15. *For conservative weight functions, a minimum directed cycle basis can be computed in $O(m^3 n)$ time.*

Proof. Follows from Theorem 3.8, and Lemmas 5.10, 5.6 and 5.8. □

Theorem 5.16. *For conservative weight functions, a minimum directed cycle basis can be computed in $O(n^3 \log n + m^2 n)$ time with a probability of at least 1/2.*

Proof. Follows from Theorem 3.8, and Lemmas 5.10, 5.6 and 5.8. □

5.7. *A solution in Monte Carlo running time $O(m^\omega)$*

We will show that minimum weight undirected and directed bases can be computed in Monte Carlo time $O(m^\omega)$. The improvement is based on two observations:

- The search for a minimum cycle basis can be restricted to a subset of the Horton multi-set, namely the set of *isometric circuits*, which have total length $O(nm)$.
- The extraction of the minimum weight basis from the set of isometric circuits can be done in Monte Carlo time $O(m^\omega)$ and with an exponentially small error probability.

We introduce isometric circuits in Section 5.7.2 and show that the set of isometric circuits contains a minimum weight basis and that their total length is $O(nm)$. The proofs require that shortest paths be chosen in a careful way. We therefore discuss unique shortest paths in Section 5.7.1. Finally, the selection of the minimum weight basis from the set of isometric circuits is discussed in Section 5.7.3.

```
let $w'(e) = w(e) + \epsilon$, where $\epsilon$ is a positive infinitesimal. Compute shortest path distances
$d(u, v)$ with respect to $w'$.
for all pairs $(u, v)$ in increasing order of $d(u, v)$ do
    if $d(u, v) = w'(uv)$ then
        $uv$ is the best path connecting $u$ and $v$. Continue with the next pair.
    end if
    for all neighbors $u'$ of $u$ with $d(u, v) = w'(uu') + d(u', v)$ do
        let $vv'$ be the first edge on $p_{vu'}$.
        if $uu'$ is not the first edge on $p_{uv'}$ then
            discard $u'$ and continue with next neighbor of $u$
        end if
        pair $u'$ and $v'$, see Fig. 23.
    end for
    select the pair $(u', v')$ with minimal value of $\min(\min(p_{uv'}), \min(p_{u'v}))$.
    set $p_{uv}$ to $p_{uv'}v'v$    (equivalently $uu'p_{u'v}$).
end for
```

Fig. 22 – Hartvigsen and Mardon's algorithm for constructing best paths.

5.7.1. *Best paths*

For the improved algorithm, we need to select shortest paths carefully. We need to select a collection p_{uv} of shortest paths such that any subpath of any p_{uv} is also in the collection and such that if $p_{uz}p_{zv}$ is a shortest path connecting u and v, then p_{uv} is equal to this path. The latter condition amounts to uniqueness of shortest paths. We discuss two methods for making shortest paths unique, a deterministic method by Hartvigsen and Mardon [45] and a simple randomized method. We will refer to unique shortest paths as *best* paths.

A deterministic solution: Let $E = \{e_1, e_2, \ldots, e_m\}$. We order E via $e_1 < e_2 < \cdots < e_m$. With any set $P = \{e_{i_1} < e_{i_2} < \cdots < e_{i_k}\} \subseteq E$, we associate the tuple $(w(P), k, e_{i_1}, \ldots, e_{i_k})$. If P and Q are distinct sets of edges, we say that P is *better* than Q and write $P \prec Q$ if the tuple for P precedes the tuple for Q in the lexicographic ordering, i.e., if either the weight of P is lower than the weight of Q, or the weights are the same and the cardinality of P is lower, or weights and cardinalities are the same and $\min(P \setminus Q) < \min(Q \setminus P)$. The relation "better" is a linear ordering. Paths and circuits can be viewed as sets of edges and hence we can order paths and circuits by $\prec$.

For any two nodes u and v, let p_{uv} be the best path from u to v in G with respect to the ordering defined above. The empty path is the best path from any node to itself and $p_{uv} = p_{vu}$. Subpaths of best paths are best, i.e., if x and z lie on p_{uv} then the subpath of p_{uv} connecting x and z is equal to p_{xz}.

Hartvigsen and Mardon [45] showed that any shortest-path algorithm can be extended to compute best paths (Fig. 22). We first modify w to w', where $w'(e) = w(e) + \epsilon$ and ϵ is a positive infinitesimal.[9] The effect of this change is that the order of paths with different weights is not changed and that for paths of the same weight, the shorter path is preferred. Let $d(u, v)$ be the shortest path distances according to the modified weight function (computed by any all-pairs shortest-path algorithm).

We now consider the pairs (u, v) in order of increasing $d(u, v)$ values; ties are broken arbitrarily. We assume inductively that p_{xy} is already computed for (x, y) with $d(x, y) < d(u, v)$ and show how to compute p_{uv} in $deg(u)+deg(v)$ time. Hence, the total time is $O(nm)$. p_{uv} is either the edge uv or a proper path. The former is the case if and only if $d(u, v) = w'(uv)$. So assume that p_{uv} is a proper path and let x and y be the neighbors of u and v on p_{uv}. Then $p_{uv} = uxp_{xv} = p_{uy}yv$, $d(u, y) < d(u, v)$ and $d(x, v) < d(u, v)$. Thus p_{uy} and p_{xv} are already available and x and y will be paired in the inner for-loop. For any pair (u', v') that is formed in the inner for-loop, $uu'p_{u'v'}v'v$ is a minimum weight path of minimum length connecting u and v. Moreover, $p_{uv'} = uu'p_{u'v'}$ and $p_{u'v} = p_{u'v'}v'v$; see Fig. 23. How can we select the best among these paths? The crucial observation is that the candidate paths are edge-disjoint; thus if p and q are candidate paths, $\min(p \setminus q) < \min(q \setminus p)$ if and only if $\min p < \min q$. Indeed, consider candidate paths p and q and assume that both pass through z. Then p_{uz} is a prefix of p as well as q and p_{zv} is a suffix of p as well as q. Thus $p = q$.

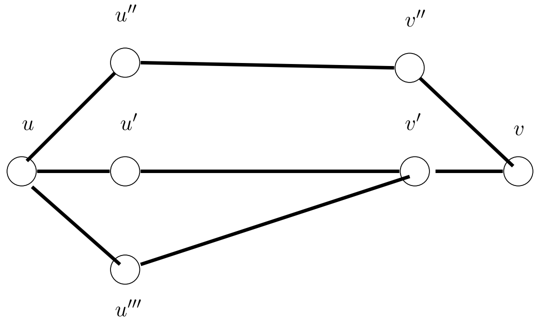


Fig. 23 – Selection of p_{uv}: There are three paths from u to v realizing $d(u, v)$. We pair u' and v' and u'' and v''; u''' is not paired with any neighbor of v.

In summary, the time to compute best paths is the time to solve the all-pair shortest-path problem for the modified weight function w' plus $O(n^2 \log n)$ time to sort the $d(u, v)$'s plus $O(nm)$ time to extract best paths. The sorting step can be avoided, see Lemma 5.17.

Remark The all-pair shortest-path problem can be solved in $O(nm + n^2 \log n)$ time. Pettie [46] improved this recently to $O(nm + n^2 \log\log n)$. For planar graphs there is an $O(n^2)$ algorithm [47] and for undirected graphs with integer edge

[9] Addition and comparison in $\mathbb{R}$ augmented by a positive infinitesimal is as follows: we have $(a+b\epsilon)+(c+d\epsilon) = (a+b)+(c+d)\epsilon$ and $(a + b\epsilon) < (c + d\epsilon)$ if either $a < c$ or $a = c$ and $b < d$.

weights, there is an $O(nm)$ algorithm [48]. Thus, for sparse graphs with $m = O(n)$, the sorting step will be the bottleneck. However, the sorting step is not required.

Lemma 5.17. *Best paths can be computed in $O(APSP + nm)$ time, where APSP is the time to solve the all-pairs shortest-path problem for the modified weight function.*

Proof. We replace sorting by a topological ordering of a suitable directed graph. The vertices are the pairs (u, v) with $u, v \in V$. We have an edge from (u, v') to (u, v) if $v'v \in E$ and $d(u, v') + w'(v'v) = d(u, v)$, and we have an edge from (u', v) to (u, v) if $uu' \in E$ and $w(uu') + d(u'v) = d(u, v)$. The number of edges is $2nm$. We process the nodes in topological order. □

A randomized solution: We set $w'(e) = w(e) + \epsilon + r_e\epsilon^2$, where ϵ is a positive infinitesimal and r_e is a random integer in $[0..M-1]$ for $M = 2n^2m^2$. For nodes x and y and integer ℓ, let $d(x, y, \ell)$ be the minimum weight (with respect to weight function w') of a path of length ℓ connecting x and y. If shortest paths are not unique, there must be an ℓ, $1 \le \ell \le n$, a node x, and edges uy and vy with $u \ne v$ such that

$$d(x, u, \ell - 1) + w'(uy) = d(x, v, \ell - 1) + w'(vy).$$

There are fewer than n^2m^2 such choices. For each choice, the probability that the event happens is at most $1/M$. Thus, the probability that shortest paths are not unique is at most n^2m^2/M.

We run our favorite all-pairs algorithm for weight function w'. Let d be the computed distance function. We perform the following check: For any pair (x, y) with $x \ne y$, we check whether there are two neighbors, u and v, of y with $d(x, y) = d(x, u) + w'(uy) = d(x, v) + w'(vy)$. If this is the case for some pair (x, y), we declare the perturbation a failure, choose new values r_e, and repeat. The check takes $O(nm)$ time. We fail with a probability of at most $1/2$ and hence the expected number of trials is at most 2.

Lemma 5.18. *Best paths can be computed in Las Vegas time $O(APSP+nm)$, where APSP is the time to solve the all-pairs shortest-path problem.*

Circuits are also ordered by the weight function w'. A circuit C is better than a circuit C' if and only if $w'(C) < w(C')$. Distinct circuits may have the same weight.

5.7.2. *Isometric circuits*

For any pair, u and v, of nodes, let p_{uv} be a best path connecting u and v. In the preceding section, we learned how to compute a collection of best paths. Horton [12] introduced the notion of *isometric circuits*. A circuit C is *isometric* if for any two vertices, u and v, on C, p_{uv} is contained in C. We use $\mathcal{I}$ to denote the set of isometric circuits. Actually, Horton called a cycle isometric if for any two vertices, u and v, some shortest path connecting u and v is contained in C. Of course, with this definition the number of isometric circuits may be exponential. With the definition given here and an unfortunate choice of designated shortest paths, the set of isometric paths may be empty, as Fig. 24 shows. With the right choice of designated shortest paths, isometric circuits exist and can be used for a minimum weight basis.

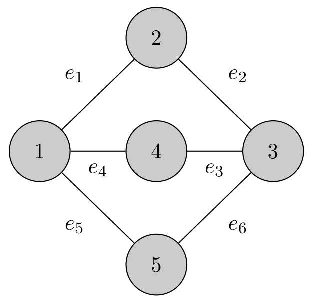


Fig. 24 – All edges have weight zero. If $(1, 5, 3)$ is chosen as $p_{1,3}$, $(2, 1, 5, 3, 4)$ as $p_{2,4}$, and $(4, 1, 2, 3, 5)$ as $p_{4,5}$, then no circuit is isometric. The deterministic strategy of Section 5.7.1 selects all edges as shortest paths and sets $p_{1,3} = (1, 2, 3)$, $p_{2,4} = (2, 1, 4)$, $p_{2,5} = (2, 1, 5)$, and $p_{4,5} = (4, 3, 5)$. The circuits $(1, 2, 3, 4, 1)$ and $(1, 2, 3, 5, 1)$ are isometric, but the circuit $(1, 4, 3, 5, 1)$ is not.

Lemma 5.19 ([39]). *$\mathcal{I}$ contains a minimum weight κ-basis.*

Proof. We run the greedy algorithm for cycle bases on the set of all circuits ordered by the relation "better" defined in the preceding subsection; ties are broken arbitrarily. The algorithm starts with the empty basis and considers the circuits in order of decreasing quality. Whenever a circuit is encountered that is independent of the current basis, the circuit is added to the current basis. We claim that the algorithm chooses only circuits in $\mathcal{I}$.

Consider a circuit $C \notin I$ and let B be the partial basis when C is considered for inclusion in B. There are vertices u and v on C such that C does not contain p_{uv}. Split C at u and v to obtain a path p_1 from u to v and a path p_2 from v to u. Consider the cycles $C_1 = p_1p_{vu}$ and $C_2 = p_2p_{uv}$. We have $C = C_1 + C_2$ and C_1 and C_2 are better than C. Thus, both circuits were considered before C and hence lie in the span of B. Thus, C lies in the span of B and is not added to B. □

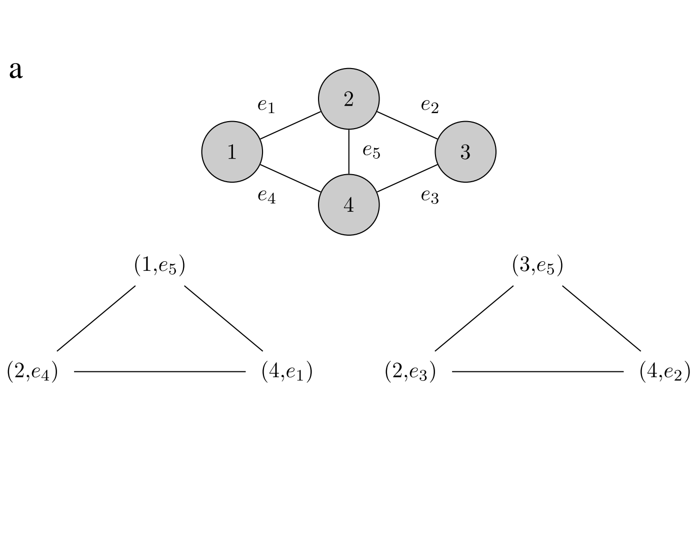
Lemma 5.20 ([12]). *Let C be any isometric circuit and let x be an arbitrary vertex of C. Then there is an edge $e = (u, v)$ on C such that $C = p_{xu}ep_{vx}$. Conversely, if for every $x \in C$, there is such an edge, then C is isometric.*

Proof. Let $C = (x = v_0, v_1, \ldots, v_k = x)$. Since the empty path is the minimum weight path from x to x and C is not the minimum weight path from x to x, there must be an i such that $p_{xv_i} = (v_0, v_1, \ldots, v_i)$ but $p_{xv_{i+1}} \ne (v_0, v_1, \ldots, v_i, v_{i+1})$. Then $p_{xv_{i+1}} = (v_k, v_{k-1}, \ldots, v_{i+1})$ and hence $e = (v_i, v_{i+1})$ is the desired edge.

For the converse, consider any two nodes x and z on C and let $e = uv$ be such that $C = p_{xu}ep_{vx}$; z lies on one of the paths and subpaths of best paths are best paths. Thus C contains p_{xz}. □

Recall the definition of the Horton multi-set $\mathcal{H}$. It consists of all circuits $C_{x,e} = p_{xu}ep_{vx}$, where $e = uv$ and the paths p_{xu} and p_{xv} do not share a first edge. By Lemma 5.20, each isometric circuit C is a $C_{x,e}$ for $|C|$ different nodes x.

Lemma 5.21 ([39]). *The total length of the isometric cycles is at most nm.*



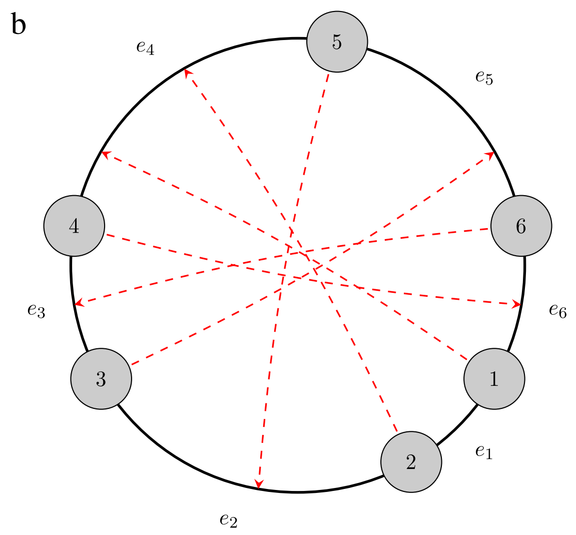


Fig. 25 – In (a) all edges have weight one; we select e_1e_2 as the best path connecting 1 and 3. The pairs $(1, e_2)$ and $(3, e_1)$ do not contribute a circuit to the Horton set. The circuits C_{1,e_3} and C_{3,e_4} are bad by condition 2c. For the former cycle let $x = 1$, $u = 3$, and $v = 4$; then $2 = s_1(3)$ and $1 \neq s_2(4)$ and $3 \neq s_4(2)$. The other circuits are connected as shown below the graph. (b) shows an isometric circuit C embedded onto a circle. The edges correspond to the circular arcs between the vertices and the length of an arc is proportional to the weight of the corresponding edge. For any vertex v, we have $C = C_{v,e}$ where e contains the mirror image of v with respect to the center of the circle. We have the following connections: C_{1,e_4} and C_{2,e_4} are connected by condition 2a, C_{2,e_4} and C_{5,e_2} are connected by condition 2b, and so on.

Proof. An isometric cycle C occurs $|C|$ times in the Horton multi-set and hence $\sum_{C\in\mathcal{I}} |C|$ can be no larger than the cardinality of the Horton multi-set. □

We will next show that we can extract $\mathcal{I}$ from the Horton multi-set in $O(nm)$ time. For any node x, let T_x be the best-path tree rooted at x, i.e., T_x is the union of the paths p_{xv} over all v. For every node $v \neq x$, let $s_x(v)$ be the child of x in T_x containing v in its subtree. In other words, $s_x(v)$ is the first node on the best path from x to v. The vectors s_x for all $x \in V$ can be the computed in $O(n^2)$ time. The following lemma shows how to identify different representations of the same isometric circuit and how to discover non-isometric circuits. The Horton multi-set consists of all $C_{x,e}$ with $e = uv$ and either $s_x(u) \neq s_x(v)$ or $x \in \{u, v\}$ and $e \neq p_{uv}$.

Lemma 5.22 ([39]). *Consider $C = C_{x,e} \in \mathcal{H}$.*

1. *If x is an endpoint of e, say $e = xv$, then $C = ep_{xv} = C_{v,e}$.*
2. *If x is not an endpoint of e, say $e = uv$, and $x' = s_x(u)$ is the first node on the best path from x to u then:*
   (a) *if $x = s_{x'}(v)$, then $C_{x',e} = C$,*
   (b) *if $x \neq s_{x'}(v)$ and $u = s_v(x')$ then $C = C_{v,xx'}$, and*
   (c) *if $x \neq s_{x'}(v)$ and $u \neq s_v(x')$ then C is not isometric.*

Proof. If $e = xv$, we have $C = p_{xv}vx = xvp_{vx} = C_{v,e}$. This establishes 1. Now assume that x is not an endpoint of e. Let $e = uv$ and let x' be the first vertex on the best path from x to u. Then $p_{xu} = xx'p_{x'u}$.

If x is the first vertex on the best path from x' to v, then $p_{ux'}p_{x'v} = p_{ux}p_{xv}$. Thus $C = C_{x',e}$. This establishes 2a.

Now assume that x is not the first vertex on the best path from x' to v. If C is isometric, the best path from x' to v must be $p_{x'u}$ followed by e. Therefore u is the first vertex best path from v to x'. This establishes 2c. Conversely, if u is the first vertex on the best path from v to x', $p_{vx'} = vup_{ux'}$ and hence $C = p_{vx}xx'p_{x'v} = C_{v,xx'}$. This establishes 2b. □

Lemma 5.22 allows us to identify different representations of the same isometric circuit. It also allows us to exclude some circuits as non-isometric. We next show that all representations of an isometric circuit will be identified and all non-isometric circuits will be discovered. We set up a graph whose vertices are the pairs (x, e), $x \in V$, $e \in E$. Let u be either endpoint of e and let v the other endpoint. We label (v, e) as *bad* if either (x, e) does not contribute a circuit to $\mathcal{H}$ or condition 2c holds. We connect two pairs if they satisfy condition 1 or 2a or 2b; see Fig. 25.

Lemma 5.23 ([39]). *All representations of an isometric circuit belong to the same connected component.*

Proof. Let $C = (v_0, v_1, \ldots, v_k = v_0)$ be an isometric circuit, let $e_i = (v_i, v_{i+1})$, and for any i, $0 \leq i < k$, let $j(i)$ be such that $C = C_{v_i,e_{j(i)}}$. Fig. 25 shows how the different representations of C are linked together. In this figure, a representation $C_{v_i,e_{j(i)}}$ is indicated as a dashed arrow from v_i to $e_{j(i)}$. In cases 1 and 2a, v_i and v_{i+1} point to the same edge, i.e., the tail of the arrow advances by one position. In case 2b, we replace the arrow from v_i to $e_{j(i)} = v_{j(i)}v_{j(i)+1}$ by the arrow from $v_{j(i)+1}$ to v_iv_{i+1}, i.e., we reverse the direction of the arrow and it now points from the tail of $e_{j(i)}$ to the cycle edge out of v_i. In this way, the arrow sweeps around the circuit once and links all representations of the same circuit. □

Lemma 5.24 ([39]). *If $C_{v,e}$ is non-isometric then the component of (v, e) contains a bad pair.*

Proof. Let $C = (v_0, v_1, \ldots, v_k = v_0)$ be a non-isometric circuit and let $e_i = (v_i, v_{i+1})$. For some, but not all, i, $0 \leq i < k$, there will be a $j(i)$ such that $C = C_{v_i,e_{j(i)}}$. Observe that if $C = C_{v_i,e_{j(i)}}$, the best paths from v_i to the vertices of C are initial segments of either $p_{v_i v_{j(i)}}$ or $p_{v_i v_{j(i)+1}}$. Also, if the best path from v_{i+1} to $v_{j(i)+1}$ is contained in C, then either $C = C_{v_{i+1},e_{j(i)}}$ or $C = C_{v_{j(i)+1},e_i}$.

Thus, if C is non-isometric, there must be i such that the best path from v_{i+1} to $v_{j(i)+1}$ is not contained in C. For any such i, $(v_i, e_{j(i)})$ will be declared bad. Thus, if $C_{v,e}$ is non-isometric, its component will contain a bad pair. □

Theorem 5.25 ([39]). *In $O(nm)$ time, we can extract for each isometric cycle one pair (v, e) with $C = C_{v,e}$.*

5.7.3. *The algorithm*

We assume $\kappa = GF(p)$. We refine de Pina's approach by selecting in phase i a minimum weight isometric circuit instead of a minimum weight circuit from the Horton set, i.e., line (4) of Algorithm 1 is changed into:

find a minimum weight isometric circuit C_i with $\langle C_i, S_i\rangle \neq 0$.

A probabilistic search technique finds this circuit quickly.

Lemma 5.26 ([39]). *Let $\mathcal{C}$ be a collection of circuits. For each circuit $C \in \mathcal{C}$, let $\lambda_C \in GF(p)$ be chosen randomly and let $D = \sum_{C\in\mathcal{C}} \lambda_C C$. Let S be a non-zero vector in $GF(p)^E$. If all circuits in $\mathcal{C}$ are orthogonal to S, D is orthogonal to S. If $\mathcal{C}$ contains a circuit that is non-orthogonal to S, D is orthogonal to S with a probability of at most $1/p$.*

Proof. Clearly, if every circuit in $\mathcal{C}$ is orthogonal to S, then so is D.

Assume next that $C' \in \mathcal{C}$ is non-orthogonal to S and consider a fixed choice of coefficients λ_C for the circuits $C \in \mathcal{C}$, $C \neq C'$. Also assume that there are two distinct choices α and β for $\lambda_{C'}$ such that $\sum_{C\in\mathcal{C}} \lambda_C C$ are orthogonal to S. Then $\alpha C' + \sum_{C\in\mathcal{C},\mathcal{C}\neq\mathcal{C}'} \lambda_C C$ and $\beta C' + \sum_{C\in\mathcal{C},\mathcal{C}\neq\mathcal{C}'} \lambda_C C$ are orthogonal to S. Thus $(\beta-\alpha)C'$ is orthogonal to S, which is a contradiction. Thus the probability that $\langle D, S\rangle = 0$ is at most $1/p$. □

Consider the $|\mathcal{I}| \leq nm$ isometric circuits. We sort them by nondecreasing weight and put a binary tree (of depth of at most $\log nm$, that is, $O(\log n)$) on top of the sorted list. For each node of the tree, we prepare k random linear combinations of the circuits below the node. We find the cheapest circuit that has non-zero inner product with S_i as follows. Assume the search has reached some node of the tree. We compute the inner product of S_i with the k linear combinations associated with the left child. If one inner product is non-zero, we proceed to the left child. If all k inner products are zero, we proceed to the right child. The move to the left child is always correct. However, the move to the right child may be incorrect. The probability that any specific decision is incorrect is at most p^{-k}. In any search, we make $\log|\mathcal{I}|$ decisions, and we need to find ν circuits. Thus the total number of decisions is $\nu \log|\mathcal{I}|$ and hence the total probability of error is bounded by $\nu \log|\mathcal{I}|p^{-k}$.

Each step of the binary search is a scalar product and hence selecting one circuit takes $O(km \log n)$ time. Selecting all circuits takes $O(km^2 \log n)$ time.

How much time does it take to prepare the random linear combinations? We maintain them as sparse vectors, i.e., as the ordered list of their non-zero entries. In order to prepare one linear combination for each node of the search tree, we choose a random multiplier $\lambda_C \in \kappa$ for each isometric circuit C.

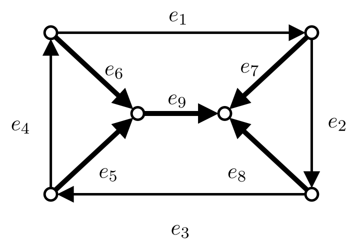


Fig. 26 – The envelope graph.

We then sum the sparse vectors as indicated by the tree. Each non-zero entry of a circuit contributes cost $O(1)$ for each level of the tree and hence the total time to prepare one random linear combination for each node of the search tree is $O(nm \log n)$, by property 5.21. We want k linear combinations for each node and hence require $O(knm \log n)$ time to prepare all of them.

Theorem 5.27 ([39]). *There is a Monte Carlo algorithm for finding a minimum $GF(p)$-basis that works in $O(nm + n^2 \log n + m^\omega + km^2 \log n)$ time and errs with a probability of at most $\nu \log(nm)p^{-k}$. For $k = m^{0.1}$, this is exponentially small, and the running time is $O(m^\omega)$.*

Undirected bases are $GF(2)$-bases and hence we are done. For directed cycle bases we use Theorem 3.8, namely that a minimum $GF(p)$-basis for a random p with $p = \Theta(m \log m)$ is a minimum directed basis with a probability of at least 1/2.

Theorem 5.28 ([39]). *There is a Monte Carlo algorithm for finding a minimum directed cycle basis that works in $O(m^\omega)$ time and errs with a probability of at most 1/2.*

5.8. *A greedy algorithm for integral cycle bases?*

Both the greedy algorithm (Section 5.1) and de Pina's approach (Section 5.2) fundamentally rely on Theorem 3.10, namely the fact that all subsets of K-bases in G constitute a matroid for $K \in \{D, U\}$. This is not true for integral bases.

Theorem 5.29 ([32]). *The system of all subsets of integral cycle bases in G is not a matroid.*

Proof. We exhibit a graph with two integral cycle bases B_1 and B_2 and a circuit $C_1 \in B_1 \setminus B_2$ such that for no circuit $C_2 \in B_2 \setminus B_1$, is $B_1 \setminus \{C_1\} \cup \{C_2\}$ again an integral basis.

Consider the directed envelope graph shown in Fig. 26 and the spanning tree T indicated by the bold edges. The bases B_1 and B_2 are given by the cycle matrices (only the parts corresponding to non-tree edges are shown)

$$\Gamma_1 = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad \Gamma_2 = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}.$$

The bases are integral since $|\det \Gamma_1| = |\det \Gamma_2| = 1$. Now choose the circuit in the first column of Γ_1 – call it C_1 – to exit the basis. Of course, neither the third nor the fourth circuit in B_2 can replace C_1 since both already appear in $B_1 \setminus \{C_1\}$. But adding the first or the second circuit of B_2 results in a cycle basis of determinant 2 or 3, respectively. □

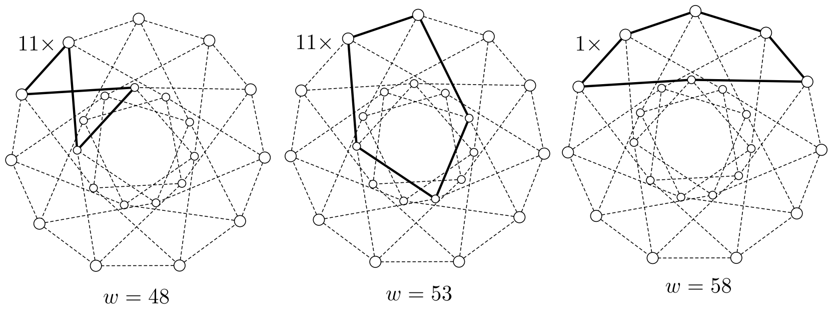


Fig. 27 – An integral cycle basis B_1 of G with a total weight of 1169.

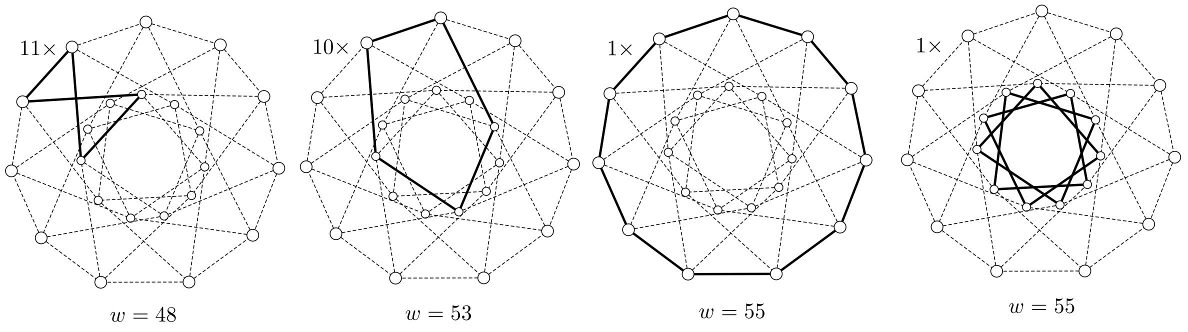


Fig. 28 – The (unique) minimum integral cycle basis B_2 of G with a total weight of 1168.

Theorem 5.29 does not yet imply the failure of the greedy algorithm nor of de Pina's approach, since the weights of cycles in G cannot be chosen independently for each cycle. A greedy algorithm for minimum integral bases would consider circuits in order of increasing weight. It would maintain a partial basis that can be extended to an integral basis and add a circuit to the current basis if this property is maintained. It is not known how to implement this strategy efficiently. In any case, it would not work.

Theorem 5.30. *The greedy algorithm may end up with a non-optimal integral cycle basis of G.*

Proof. We again consider the graph introduced in Lemma 3.20 together with the same two integral bases B_1 and B_2 depicted in Figs. 27 and 28. In contrast to Lemma 3.20, we assign other weights to the edges. Let every inner and outer edge have a weight of 5 whereas every spoke has a weight of 19. Then the first 22 cycles in B_1 are the only ones in G whose weights do not exceed 53. Moreover, there are exactly two cycles, the inner cycle C_I and the outer cycle C_O, with weight 55 and the weight of every other cycle is at least 58. Under this assignment of weights, B_1 has a total weight of 1169 whereas the weight of B_2 is 1168.

As a consequence, B_2 is the unique minimum integral cycle basis. On the other hand, the greedy algorithm first picks the 22 cycles of weight of at most 53. These cannot be extended to an integral basis by adding C_I nor C_O and hence the greedy algorithm will end up with a basis similar to B_1 and thus with a non-optimal basis. □

Finally, we observe that the basis B_1 in the preceding proof, although non-optimal, constitutes a *locally optimal* integral cycle basis of G, i.e. B_1 cannot be improved by an exchange of a single cycle in G. This is true since the only two exchanges which would decrease the weight of B_1 are the replacement of the 58-circuit by either C_I or C_O, but both result in a non-integral basis. Hence a local-search procedure fails in general; de Pina's approach can be interpreted as such a local search.

5.9. *Planar graphs*

For planar graphs, a minimum undirected cycle basis can be computed in $O(n^2)$ time, a minimum 2-basis can be computed in linear time, and the notions of minimum directed, undirected, integral, and weakly fundamental and totally unimodular bases coincide. The algorithm was found by Hartvigsen and Mardon [45]; Amaldi et al. [39] improved the running time from $O(n^2 \log n)$ to $O(n^2)$.

Let G be a plane graph, i.e., a planar graph that is embedded into the plane. A plane graph divides the plane into maximal open connected sets of points that we call *faces*. Any circuit C divides the plane into two maximal open connected sets of points, one bounded and one unbounded. We use *interior*(C) to denote the bounded set. If *interior*(C) agrees with one of the faces of G, we call C a *face circuit*. A collection of circuits is called *nested* if, for any two circuits C and D in the collection, the interiors are either disjoint or the interior of one is contained in the interior of the other.

For a collection B of circuits, let F_B be the face circuits that do not belong to B. We define the directed inclusion graph D_B with vertex set $B \cup F_B$ as follows. Let C and C' be circuits in $B \cup F_B$. We have an edge from C to C' if *interior*$(C) \subset$ *interior*(C') and there is no circuit $C'' \in B \cup F_B$ such that *interior*$(C) \subset$

$interior(C'') \subset interior(C')$. The inclusion graph is acyclic; the sources of the inclusion graph are precisely the face circuits of G. The inclusion graph is a forest if and only if B is nested.

Theorem 5.31 ([45]). *Let G be a plane graph. G has a minimum (directed or undirected) cycle basis that is nested. The number of isometric cycles is at most twice the number of faces of G.*

Proof. The circuits in a basis are isometric (Lemma 5.19). In a plane graph, any two isometric circuits have either disjoint interiors or the interior of one is contained in the interior of the other.

Let B be the set of all isometric circuits. The inclusion graph D_B is a forest with f leaves, where f is the number of faces of G. Each non-leaf has an indegree of at least two. Thus, the number of nonleaves is at most $f - 1$. □

Theorem 5.32 ([45]). *Let G be a plane graph. A nested collection B of circuits is a minimum (directed or undirected) cycle basis iff B is a minimum weight collection of circuits satisfying the following three properties:*

1. *the inclusion graph D_B is a forest,*
2. *every non-leaf in D_B has exactly one child in F_B, and*
3. *the circuits in F_B have parents in D_B.*

Proof. Assume first that B is a basis. We first observe that the number of circuits in B that are not face circuits is equal to the number of face circuits that do not belong to B since the face circuits form a basis and all bases have the same cardinality.

If B is nested, the inclusion graph is a forest. Consider any non-leaf C of D_B. If no child of C belongs to F_B, C is the sum of its children and B is not a basis. Thus any non-leaf C has at least one child in F_B. The nonleaves of the inclusion graph are precisely the circuits in B that are not face circuits. Thus, any non-leaf has exactly one child in F_B and every circuit in F_B must have a parent.

Conversely, assume that B is a minimum cost collection of circuits satisfying (1) to (3). Since D_B is a forest, B is nested. Since the circuits in F_B have parents in D_B and these parents are distinct, the number of nonleaves in D_B is exactly the number of circuits in F_B. So B has the right number of circuits for a basis. Finally, any face circuit is representable as a sum of circuits in B. This is obvious for the face circuits that belong to B. For the face circuits in F_B, it follows from (2) and (3). □

We now come to the algorithm for finding a minimum weight basis. We start by computing the best-path trees T_v for all vertices v; by Lemma 5.17 this takes $O(n^2)$ time plus the time to solve the all-pair shortest-path problem. Frederickson [47] showed how to compute all-pair shortest paths in planar graphs in $O(n^2)$ time. The Horton multi-set consists of $O(n^2)$ cycles. In $O(n^2)$ time, we extract one copy of each isometric circuit from it (Theorem 5.25). The number of isometric circuits is $O(n)$ (Theorem 5.31). We sort the isometric circuits by weight; it takes $O(n^2)$ time to determine the weights and $O(n \log n)$ time to sort.

We construct the incidence matrix A between isometric circuits and the faces of G. The entry corresponding to a circuit C and a face F is one if $F \subseteq interior(C)$. This matrix can clearly be computed in $O(n^2)$ time.

We initialize the basis B to the empty set and set up the corresponding inclusion graph D_B. The vertices of D_B are the face circuits and there are no edges. As long as B does not have the right number of circuits, meaning D_B does not satisfy (2) and (3), we do the following. If there is a non-leaf node C that has two children in F_B (case 1), let R_1 and R_2 be two face circuits in F_B having C as their common parent. If there is no such non-leaf node, there must be a face circuit in F_B without a parent (case 2). Let R_1 be this face and let R_2 be the unbounded face. In either case, we find the least weight circuit D containing exactly one of R_1 or R_2 in its interior. We can find D in time $O(n)$ by scanning the columns of A.

We add D to B and update D_B. If D is a face circuit, we only have to remove D from F_B. The inclusion graph stays the same. If D is not a face circuit, we determine, starting from the face circuits in $interior(D)$ (we can find them in matrix A), the maximal subtrees of D_B that are contained in $interior(D)$. They become children of D. D either becomes a root (in case 2) or a child of C (in case 1). Updating D_B takes $O(n)$ time.

We conclude that we time $O(n)$ time per base circuit for a total of $O(n^2)$.

Theorem 5.33 ([45,39]). *A minimum (directed or undirected) circuit basis of a planar graph can be found in $O(n^2)$ time.*

Hartvigsen and Mardon [45] observed that the minimum cycle basis problem is dual to the all-pairs minimum cut problem. Hence the all-pairs minimum cut problem in planar graphs can also be solved in $O(n^2)$ time.

Theorem 5.34. *Every planar graph has a minimum directed cycle basis that is weakly fundamental, totally unimodular, and integral.*

Proof. By the above, every planar graph has a minimum directed cycle basis that is nested. Let B be such a basis. We first show that B is totally unimodular. We need to show that any circuit is a linear combination of the circuits in B with coefficients in $\{-1, 0, +1\}$. Let C be any circuit. Then,

$$C = \sum_{F \text{ is a face circuit contained in } interior(C)} F.$$

A face circuit either belongs to B or is the difference between the parent of F in D_B and the sum of the siblings of F in D_B. Thus,

$$C = \sum_{F \in B} F + \sum_{F \in F_B} \left(p(F) - \sum_{D \in B \text{ and } D \text{ is a sibling of } F \text{ in } D_B} D \right).$$

If a circuit D occurs twice in the representation of D, it occurs once as a parent and once as a child. As a parent, its coefficient is $+1$, and as a child, it is -1, and hence the two occurrences cancel. Thus, every circuit is a linear combination of the circuits in B with coefficients in $\{-1, 0, +1\}$.

We next show that B is weakly fundamental. We need to show an ordering $C_1, \ldots, C_\nu$ of the circuits in B such that $C_i \setminus (C_{i+1} \cup \cdots \cup C_\nu) \neq \emptyset$ for all i. Let D_B be the inclusion graph corresponding to B. If F_B is empty, every bounded face circuit belongs to B. There must be an edge on the unbounded face that belongs to at most one circuit in B. Take this circuit as C_1 and continue in this fashion. Assume next that F_B is non-empty. Since every circuit in F_B has a parent, we have a non-leaf node C in D_B, all of whose children are face circuits. One of these face circuits, say F, belongs to F_B and the others belong to B. There must be at least one edge on the boundary of F that does not belong to C because, otherwise, $C = F$. Let F' be the other face circuit incident to e and let p be the maximal path containing e and having all interior vertices of degree two. We remove the edges of p from the graph, assign F' to e, delete F' from B, and add the edges of $p \setminus e$ to the spanning tree. Removal of p merges F and F' and $B \setminus F'$ is a basis for the modified graph. Continuing in this way constructs an elimination order for the edges.

The proof is completed by the fact that any weakly fundamental basis is integral. □

5.10. *Approximation*

Minimum directed and undirected cycle bases can be computed in polynomial time. However, the running times are fairly high degree polynomials, too high for applications, e.g., circuit analysis, that need to find cycle bases of graphs with several million vertices and edges. However, in these applications, a *nearly* optimal basis is almost as good as an optimal basis. It is therefore natural to explore approximation algorithms. The results presented in this section are based on [36,38]. We present two approximation techniques, the first of which uses de Pina's approach and replaces shortest-path computations by approximate shortest-path computations. The second technique uses Horton's approach and replaces the Horton set $\mathcal{H}$ by a smaller set of circuits that is guaranteed to contain a 2-approximate cycle basis. We will start with lower bounds that we will use in our quality estimates.

Lemma 5.35 ([14]). *Let $R_1, \ldots, R_\nu$ be linearly independent vectors in κ^ν and let A_i be a shortest cycle in G such that $\langle A_i, R_i \rangle \neq 0$. Then $\sum_{i=1}^{\nu} w(A_i)$ is a lower bound on the weight of any κ-basis.*

Proof. Let $\{C_1, \ldots, C_\nu\}$ be a κ-basis. We may assume without loss of generality that the A_i's and C_i's are sorted by weight, that is, $w(A_1) \le w(A_2) \le \cdots \le w(A_\nu)$ and $w(C_1) \le w(C_2) \le \cdots \le w(C_\nu)$. The former may require a renumbering of the R_i's. We will show that $w(A_i) \le w(C_i)$ for all i.

Consider a fixed i and observe that $\langle C_k, R_\ell \rangle \neq 0$ for some k and ℓ with $1 \le k \le i \le \ell \le \nu$. Otherwise, the $\nu - i + 1$ linearly independent vectors $R_i, R_{i+1}, \ldots, R_\nu$ belong to the subspace orthogonal to $C_1, \ldots, C_i$; however, this subspace has dimension $\nu - i$, which is a contradiction. Thus, $w(A_\ell) \le w(C_k)$ since A_ℓ is a shortest cycle with $\langle A_\ell, R_\ell \rangle \neq 0$ and hence $w(A_i) \le w(A_\ell) \le w(C_k) \le w(C_i)$. □

Corollary 5.36. *Let G be a graph. For any edge e, let SC_e be the minimum weight cycle containing e. Then $\sum_{e \in E} w(SC_e)$ is a lower bound on the weight of any cycle basis.*

Proof. Let R_e be the unit vector whose entry corresponding to e is one. The vectors R_e, $e \in E$, are clearly independent (over $\mathbb{Q}$ and over $GF(p)$) and $\langle R_e, SC_e \rangle = 1 \neq 0$. Clearly SC_e is a shortest cycle C with $\langle R_e, C \rangle \neq 0$. □

5.10.1. *Approximate shortest paths*

De Pina's approach works in phases. In each phase, we compute a support vector S and a shortest circuit C with $\langle S, C \rangle \neq 0$. If instead of searching for a shortest circuit, we search for a t-approximation of it, we should obtain a t-approximate cycle basis. We next show how to realize this idea for any integer $k > 1$ and $t = 2k - 1$.

Algorithm 4 Approximation algorithm. Best performance for sparse graphs.

```
1: procedure SPANNER-APPROX-SPARSE(Graph G)
2:     Construct a $(2k-1)$-spanner $G'$ with $O(n^{1+1/k})$ edges. Let $e_1, \ldots, e_\lambda$ be the edges of $G \setminus G'$.
3:     For $1 \le i \le \lambda$ let $C_i = e_i + p_i$ where $e_i = (u_i, v_i)$ and $p_i$ is a shortest path in $G'$ from $u_i$ to $v_i$.
4:     Find linearly independent $S_{\lambda+1}, \ldots, S_\nu$ in the subspace orthogonal to cycles $C_1, \ldots, C_\lambda$.
5:     Call the recursive algorithm in Section 5.3 with input: the graph G, sets $\{C_1, \ldots, C_\lambda\}$, $\{S_{\lambda+1}, \ldots, S_\nu\}$ and $\nu - \lambda$ to compute $(2k-1)$-approximate cycles $C_{\lambda+1}, \ldots, C_\nu$.
6:     Return $\{C_1, \ldots, C_\lambda\} \cup \{C_{\lambda+1} \ldots, C_\nu\}$.
7: end procedure
```

A t-spanner of an undirected graph G is a subgraph G' of G such that for any two vertices u and v, the distance from u to v in G' is at most t times their distance in G. Althöfer et al. [49] showed that every weighted undirected graph on n vertices has a $(2k-1)$-spanner with $O(n^{1+1/k})$ edges. Such a spanner is easily constructed incrementally. We start with an empty graph G' and consider the edges of G in non-decreasing order of weight. When an edge is considered, we add it to G' if its endpoints are not already connected by a path using at most $2k-1$ edges of G'; otherwise, we discard it. At any stage, G' is a $(2k-1)$-spanner of the edges already considered, and its unweighted girth[10] is at least $2k+1$, so it has only $O(n^{1+1/k})$ edges. The above procedure can be implemented to run in $O(mn^{1+1/k})$ time. From now on, $G' = (V, E')$ denotes a t-spanner of G. Let $\lambda = |E \setminus E'|$ and $m' = |E'| = m - \lambda$. Observe that $\nu' = m' - n + 1 = m - n + 1 - \lambda = \nu - \lambda$.

For each edge $e = (v, w) \in E \setminus E'$, let C_e be the circuit consisting of e and the shortest path, say p, in G' connecting v and w. Then

$$w(C_e) = w(e) + w(p) \le w(e) + t\, dist_G(u, v) \le t\, w(SC_e).$$

The circuits C_e, $e \in E \setminus E'$, are clearly independent and form the first λ circuits in our t-approximate basis. The cost of constructing these λ circuits is the cost of λ shortest-path computations in G' and hence bounded by $O(\lambda \cdot (n^{1+1/k} + n \log n))$. Since $\lambda \le m$ we can compute both the spanner and the λ circuits in $O(mn^{1+1/k})$ time.

We need an additional $\nu - \lambda$ circuits for a basis. We outline one approach and then discuss a second approach in more detail. The first approach now switches to the recursive algorithm in Section 5.3. It first computes a basis $S_{\lambda+1}, \ldots, S_\nu$ of the subspace orthogonal to C_e, $e \in E \setminus E'$ and then proceeds as in Section 5.3; see Algorithm 4. Instead of computing a shortest cycle in each phase, it computes a t-approximate shortest path using the *approximate distance oracle* of [50]. This data structure answers $(2k-1)$-approximate shortest path queries in $O(k)$ time. The structure requires $O(kn^{1+1/k})$ space and can be constructed in $O(kmn^{1/k})$ expected time.

Theorem 5.37 ([38,31]). *For any integer $k \ge 2$, Algorithm 4 computes a $(2k-1)$-approximate undirected cycle basis in $O(kmn^{1+2/k} + mn^{(1+1/k)(\omega-1)})$ expected time.*

[10] The girth of a graph is the minimum number of edges in any circuit.

Algorithm 5 Approximation algorithm. Best performance for dense graphs.

```
1: procedure SPANNER-APPROX-DENSE(Graph G)
2:     Construct a $(2k-1)$-spanner $G'$ with $O(n^{1+1/k})$ edges. Let $e_1,\dots,e_\lambda$ be the edges of $G \setminus G'$.
3:     For $1 \le i \le \lambda$ let $C_i = e_i + p_i$ where $e_i = (u_i, v_i)$ and $p_i$ is the shortest path in $G'$ from $u_i$ to $v_i$.
4:     Call the best exact algorithm to find an MCB of $G'$. Let these cycles be $C_{\lambda+1},\dots,C_\nu$.
5:     Return $\{C_1,\dots,C_\lambda\} \cup \{C_{\lambda+1},\dots,C_\nu\}$.
6: end procedure
```

The second approach is even simpler. We complete the basis by computing a minimum cycle basis of the t-spanner G'; see Algorithm 5. The dimension of the cycle space of G' is $\nu' = \nu - \lambda$ and thus we have the right number of circuits. Let $C_{\lambda+1},\dots,C_\nu$ be a minimum cycle basis of G'. Circuits $\{C_1,\dots,C_\lambda\} \cup \{C_{\lambda+1},\dots,C_\nu\}$ are, by definition, linearly independent and we are also going to prove that they form a t-approximation of an MCB of G.

For $1 \le i \le \lambda$, we have $C_i = e_i + p_i$, where p_i is a shortest path in G' between the endpoints of e_i. In order to show that cycles $C_1,\dots,C_\nu$ constitute a t-approximation of the MCB, we again define appropriate linearly independent vectors $S_1,\dots,S_\nu \in \kappa^m$ and use Lemma 5.35. Consider the exact algorithm in Section 5.3, executing with the t-spanner G' as its input. Other than the cycles $C_{\lambda+1},\dots,C_\nu$, the algorithm also returns the vectors $R_{\lambda+1},\dots,R_\nu \in \kappa^{m'}$ such that $\langle C_i, R_j\rangle = 0$ for $\lambda+1 \le i < j \le \nu$ and C_i is a shortest cycle in G' such that $\langle C_i, R_i\rangle \ne 0$ for $\lambda+1 \le i \le \nu$. Moreover, the $(\nu-\lambda)\times m'$ matrix whose j-th row is R_j is lower triangular with 1 in its diagonal. This implies that the R_j's are linearly independent. Given any vector $S \in \kappa^m$, let $\tilde{S}$ be the projection of S onto its last m' coordinates. In other words, $\tilde{S}$ is the restriction of S to the edge set of G'. We define S_j for $1 \le j \le \nu$ as follows. Let $S_1,\dots,S_\lambda$ be the first λ unit vectors of κ^m. For $\lambda+1 \le j \le \nu$ define S_j as:

$$S_j = (-\langle \tilde{C}_1, R_j\rangle, \dots, -\langle \tilde{C}_\lambda, R_j\rangle, R_{j,1}, R_{j,2}, \dots, R_{j,m'}),$$

where $R_{j,1},\dots,R_{j,m'}$ are the coordinates of the vector $R_j \in \kappa^{m'}$. Note that the vectors S_j for $1 \le j \le \nu$, defined above, are linearly independent. This is because the $\nu\times\nu$ matrix whose j-th row is S_j is lower triangular with non-zeros in its diagonal. The above definition of S_j's is motivated by the property that for each $1 \le i \le \lambda$, we have $\langle C_i, S_j\rangle = -\langle \tilde{C}_i, R_j\rangle + \langle \tilde{C}_i, R_j\rangle = 0$, since the cycle C_i has 0 in the first λ coordinates except the i-th coordinate, which is non-zero. Lemma 5.38, shown below, together with Lemma 5.35, implies the correctness of our approach.

Lemma 5.38. *Consider the above defined S_j for $1 \le j \le \nu$ and let D_j be a shortest cycle in G such that $\langle D_j, S_j\rangle \ne 0$. Cycle C_j returned by the algorithm in Fig. 5 has a weight of at most t times the weight of D_j.*

Proof. This is obvious for $1 \le j \le \lambda$ since D_j is a shortest cycle in G which uses edge e_j and $C_j = e_j + p_j$, where p_j is a t-approximate shortest path between the endpoints of e_j. Consider now D_j for $\lambda+1 \le j \le \nu$. If D_j uses any edge e_i for $1 \le i \le \lambda$, we replace it with the corresponding shortest path in the spanner. This is the same as saying consider the cycle $D_j - C_i$ instead of D_j. Let $D'_j = D_j - \sum_{1\le i\le\lambda}(e_i \in D_j)C_i$ where $(e_i \in D_j)$ is 1 if $e_i \in D_j$ and 0 if $e_i \notin D_j$. Then,

$$\langle D'_j, S_j\rangle = \langle D_j, S_j\rangle + \sum_{1\le i\le\lambda}(e_i \in D_j)\langle C_i, S_j\rangle.$$

But recall that our definition of S_j ensures that $\langle C_i, S_j\rangle = 0$ for $1 \le i \le \lambda$. This implies that $\langle D'_j, S_j\rangle = \langle D_j, S_j\rangle \ne 0$. But D'_j, by definition, has 0 in the first λ coordinates and $\tilde{S}_j = R_j$, which in turn implies that

$$\langle \tilde{D}'_j, R_j\rangle = \langle \tilde{D}'_j, \tilde{S}_j\rangle = \langle D'_j, S_j\rangle \ne 0.$$

C_j is a shortest cycle in G' such that $\langle C_j, R_j\rangle \ne 0$. Thus, C_j weighs no more than $\tilde{D}'_j$ (which is the same cycle as D'_j), and by construction, D'_j weighs at most t times the weight of D_j. □

Thus, we have shown that the cost of our approximate basis is at most t times the cost of an optimal basis. As a t-spanner, we will again use a $(2k-1)$-spanner. The spanner has $O(\min(m, n^{1+1/k}))$ edges and hence its minimum undirected cycle basis can be computed in $O(\min(m, n^{1+1/k})^\omega)$ time.

Theorem 5.39 ([38,31]). *For any integer $k \ge 2$, Algorithm 4 computes a $(2k-1)$-approximate undirected cycle basis in Monte Carlo time $O(mn^{1+1/k} + \min(m, n^{1+1/k})^\omega)$.*

Directed graphs: Algorithm 5 readily extends to the directed case. For the spanner computation we view our directed graph G as undirected and we compute a $(2k-1)$-spanner G'. We then give to the edges of G' the orientation that they have in G.

As in the undirected case, we return two sets of cycles. The first set is constructed as follows. For each edge $e_i \in E \setminus E'$ for $1 \le i \le \lambda$ we compute the cycle $e_i + p_i$ where p_i is the shortest path in G' between the endpoints of e_i when G' is viewed as an undirected graph. Then, we traverse each such cycle in an arbitrary orientation and form our directed cycles based on the direction of the edges in G. The second set is simply the set of cycles of a directed MCB of G'. The time bound is the same as in Theorem 5.39.

5.10.2. *2-approximation*

A direct consequence of the technique in Section 5.4 is that any reduction in size of the candidate collection $\mathcal{H}$ would immediately imply better algorithmic bounds than those in Section 5.6. In this section we show that a set of $O(m\sqrt{n\log n})$ cycles, which is a subset of $\mathcal{H}$, contains a 2-approximate minimum cycle basis.

Definition 5.2. For $v, x \in V$ and $S \subset V$, *bunch*(v, S) consists of all vertices closer to v than to any vertex in S and *cluster*(x, S) consists of all vertices v with $x \in$ *bunch*(v, S).

Lemma 5.40 ([51]). *Given a weighted graph $G = (V, E)$ and $0 < q < 1$, one can compute a set $S \subset V$ of size $O(nq\log n)$ in $O(m/q\log n)$ expected time such that $|cluster(x, S)| \le 4/q$ for all $x \in V$.*

We take $q = 1/\sqrt{n\log n}$ and first compute, as given in Lemma 5.40, a set S of $O(\sqrt{n\log n})$ vertices. This takes $O(m\sqrt{n}\log^{3/2} n)$ expected time and ensures that *cluster*(v, S) has size $\sqrt{n\log n}$ for all $v \in V$. Also, *bunch*(v, S) for all v can be computed in $O(m/q)$ expected time [50], which is $O(m\sqrt{n\log n})$. We use two types of cycles:

- the $O(m\sqrt{n\log n})$ cycles $C_{s,e}$ for all $s \in S$ and $e \in E$,
- the cycles $C_{u,e}$ for each $u \in V$ and $e = (v,w) \in E$ and either v or w in $bunch(u,S)$. The number of such cycles is $\sum_{u\in V}\sum_{v\in bunch(u,S)} deg(v)$. Rewriting this sum, we obtain $\sum_{v\in V} deg(v)\cdot|cluster(v,S)|$, which in turn is at most $\sqrt{n\log n}\sum_{v\in V} deg(v) = m\sqrt{n\log n}$.

Thus, our collection has $O(m\sqrt{n\log n})$ cycles. We need to show that it contains a 2-approximate cycle basis. Let $B_1,\ldots,B_\nu$ be the minimum cycle basis of G determined by Horton's algorithm in order of non-decreasing weight, i.e., $w(B_1) \le w(B_2) \le \cdots \le w(B_\nu)$.

Lemma 5.41. *For all $1 \le i \le \nu$ we have $B_i = \sum_{C\in\mathcal{C}_i} C$ where $\mathcal{C}_i$ is a subset of our collection and each cycle in $\mathcal{C}_i$ weighs at most $2\cdot w(B_i)$.*

Proof. Consider any B_i. If B_i belongs to our collection, we set $\mathcal{C}_i = \{B_i\}$. Otherwise, $B_i = C_{u,e}$, where $e = (v,w)$ and neither v nor w is in $bunch(u,S)$. Let $s \in S$ be the nearest vertex in S to u. Then, $w(SP(s,u)) \le w(SP(u,v))$ and $w(SP(s,u)) \le w(SP(u,w))$.

For any edge $f \in B_i$, the cycle $C(s,f)$ is in our collection and $B_i = \sum_{f\in B_i} C(s,f)$ since the paths from s to the endpoints of the edges in B_i appear twice in this sum and cancel out. We set $\mathcal{C}_i = \{C(s,f) \mid f \in B_i\}$. It remains to show that $w(C(s,f)) \le 2w(B_i)$ for all $f \in B_i$.

We now distinguish two cases. Assume first that $f \ne e$. Then $f \in SP(u,v)$ or $f \in SP(u,w)$. We may assume w.l.o.g. that the former is the case. Then $w(C(s,f)) \le w(SP(s,u)) + w(SP(u,v)) + w(SP(v,s))$ since $C(s,f)$ consists of f and the shortest paths from s to the endpoints of f and $w(SP(v,s)) \le w(SP(s,u)) + w(SP(u,v))$ by the triangle inequality. Thus, $w(C(s,f)) \le 2(w(SP(s,u)) + w(SP(u,v))) \le 2w(B_i)$ since $w(SP(s,u)) \le w(SP(u,w))$.

Assume next that $f = e$. Then,

$$\begin{aligned} w(C(s,f)) &= w(SP(s,v)) + c(e) + w(SP(w,s)) \\ &\le w(SP(s,u)) + w(SP(u,v)) + c(e) \\ &\quad + w(SP(s,u)) + w(SP(u,w)) \\ &\le 2w(SP(u,v)) + c(e) + 2w(SP(u,w)) \\ &\le 2w(B_i). \quad \square \end{aligned}$$

Lemma 5.42. *The collection defined above contains ν linearly independent cycles $A_1,\ldots,A_\nu$ with $w(A_i) \le 2\cdot w(B_i)$ for $i = 1,\ldots,\nu$.*

Proof. The lemma follows from Lemma 5.41. Assuming otherwise, let j be minimal such that $\cup_{i\le j}\mathcal{C}_i$ contains less than j linearly independent vectors with $w(A_i) \le 2\cdot w(B_i)$ for $i = 1,\ldots,j$. Then $j \ge 1$ and $\cup_{i\le j-1}\mathcal{C}_i$ contains at least $j-1$ linearly independent vectors with $w(A_i) \le 2\cdot w(B_i)$ for $i = 1,\ldots,j-1$. Also, $\cup_{i\le j}\mathcal{C}_i$ spans $\{B_1,\ldots,B_j\}$ and hence contains at least j linearly independent vectors. Thus, it contains a vector A_j linearly independent of $\{A_1,\ldots,A_{j-1}\}$. Furthermore, $A_j \in \mathcal{C}_i$ for some $i \le j$ and hence $w(A_j) \le 2w(B_i) \le 2w(B_j)$, which is a contradiction. □

It is now straightforward to extract the 2-approximate MCB using the techniques that we have discussed so far. The resulting running time is better than those in Section 5.6 but not better than those in Theorems 5.27 and 5.28.

5.11. *Algorithm engineering*

Both exact and approximate algorithms for minimum cycle bases have a fairly large worst-case running time. In this section, we discuss heuristic improvements and algorithm engineering issues. The hope is that in many cases heuristics and algorithm engineering techniques will improve upon the worst-case running time. We restrict attention to computing minimum undirected bases. Implementations of cycle basis algorithms are described in [52–56]. There is no implementation of the Monte Carlo algorithm of Section 5.7 yet.

The first decision to be made is to choose between the two main approaches. Horton's approach first computes $O(nm)$ cycles and then uses Gaussian elimination to find an optimum basis. No heuristics are known that improve upon the worst-case. The situation is different for the algebraic approach of Algorithm 1 where in each phase first a support vector and then a cycle is computed.

How should we represent cycles and support vectors, as sparse or as dense vectors? There are two arguments in favor of a sparse representation. The theoretical argument is that we know of the existence of bases of weight $O(W\log n)$; in such a basis, we expect most circuits to have $o(n)$ edges. The engineering argument is that a dense representation immediately introduces an $\Omega(m^2)$ lower bound; we are constructing m vectors of length m. Thus, the sparse representation is preferred.

The next major question is how to compute each cycle. In Algorithm 1, each cycle is computed after a support vector is found. However, there are two possible ways for doing this: (a) use the candidate set $\mathcal{H}$ or some other collection that contains a minimum cycle basis and the labeled trees representation, or (b) use the signed graph approach.

Although the labeled trees approach is faster for sparse graphs by a logarithmic factor and is faster for dense graphs when the extra technique of bit-packing from Mehlhorn and Michail [31] is used, it has a major practical drawback which needs to be addressed. It introduces a lower bound on the best case of the algorithm. The labeled trees approach maintains n shortest-path trees. In each of the ν phases of the algorithm, each of these shortest-path trees is traversed, in order to update the labels based on the current support vector. Thus, the technique introduces an $\Omega(mn^2)$ lower bound. For this reason we believe that the signed graph approach is better.

The signed graph approach constructs a graph $G_i(S_i)$, where S_i is the support vector during phase i of the algorithm. In this graph, it executes n single source shortest path computations. There are, however, some heuristics that can be used to reduce the number of such computations. During phase i we might perform up to n shortest-path computations in order to compute a shortest cycle C_i with an odd intersection with the vector S_i. We can use the shortest path found so far as an upper bound on the shortest path. This is implemented as follows: a node is only added in the priority queue of Dijkstra's implementation if its correct upper distance is not more than our current upper bound.

We come to the most important heuristic. In each of the ν phases, we are performing n shortest-path computations. This results in $\Omega(mn)$ shortest path computations. Let $S = \{e_1, e_2, \ldots, e_k\}$ be a support vector at some point of the execution. We need to compute a shortest cycle C such that $\langle C, S\rangle = 1$. We can reduce the number of shortest path computations based on the following observation.

Let $C_{\geq i}$ be the shortest cycle in G such that $\langle C_{\geq i}, S\rangle = 1$, $C_{\geq i} \cap \{e_1, \dots, e_{i-1}\} = \emptyset$, and $e_i \in C_{\geq i}$. Then,

$$C = \min_{i=1,\dots,k} C_{\geq i}.$$

We can compute $C_{\geq i}$ in the following way. We delete edges $\{e_1, \dots, e_i\}$ from G and the corresponding edges from the signed graph G_i. Let $e_i = (v, u) \in G$. Then we compute a shortest path in G_i from v^+ to u^+. The path computed will have an even number of edges from the set S and, together with e_i, an odd number. Since we deleted edges $\{e_1, \dots, e_i\}$, the resulting cycle does not contain any edges from $\{e_1, \dots, e_{i-1}\}$.

Using the above observation we can compute each cycle in $O(k \cdot \mathrm{SP}(n, m))$ time when $|S| = k < n$ and in $O(n \cdot \mathrm{SP}(n, m))$ time when $|S| \geq n$. Here $\mathrm{SP}(n, m)$ is the time of a single-source shortest-path computation in a graph with n nodes and m edges. In this way, the total cost of computing the basic circuits becomes

$$\mathrm{SP}(n, m) \cdot \sum_{i=1,\dots,\nu} \min(n, |S_i|).$$

Another issue that needs to be discussed is the use of fast matrix multiplication when computing the support vectors. Experiments in [56] with random graphs suggest that the use of fast matrix multiplication is not necessary, even for medium to large instances. The reason is that the cycles computation part of the algorithm dominates the running time, even though in theory it is the other way around. The reason is that support vectors are typically sparse. Thus, the technique of Algorithm 2, which has a worst-case bound of $O(m^3)$, is sufficient. Moreover, due to its simplicity, it is very easy to implement efficiently.

Moving to the approximation algorithms of Section 5.10, we note that they significantly improve the running times. This is not only a theoretical observation but is true in practice as well. Algorithm 5 reduces the computation of an approximate MCB to the computation of: (a) a spanner of the input graph, and (b) the MCB of a sparse graph (the previously computed spanner). This approximation algorithm is much faster than any exact algorithm. Moreover, the approximation algorithms do not really require fast matrix multiplication. Algorithm 5 requires $O(n^{3+2/k}/\log n + n^{3+1/k})$ time in order to compute a $(2k - 1)$-approximate MCB. If we do not use fast matrix multiplication, the running time increases to $O(n^{3+3/k})$. We conclude that Algorithm 5, where the support vectors are maintained as in Algorithm 2 and the cycles are computed using the signed graph approach of Section 5.4, will be an effective way of computing approximate minimum cycle bases.

5.12. *Relevant circuits*

In general, minimum cycle bases are not unique. In some applications, e.g. in chemistry [10], it is useful to know *all* minimum cycle bases. A circuit that belongs to some minimum cycle basis is called *relevant*. As their number could be exponential, the goal is to compute a set of prototype circuits from which all relevant circuits can then be derived easily. Vismara [57] presented an algorithm that, in a fashion similar to Horton's algorithm, extracts these prototypes from a polynomially sized set of candidate circuits using Gaussian elimination. Vismara's algorithm runs in $O(m^4)$ time. From these prototypes, relevant circuits can be computed in $O(n|\mathcal{C}_R|)$ time, where $\mathcal{C}_R$ denotes the set of relevant circuits.

6. Hardness results

We will show that the minimization problems for strictly and weakly fundamental cycle bases are $\mathcal{APX}$-hard. A minimization problem belongs to class $\mathcal{APX}$ if it has a constant-factor approximation algorithm and is *$\mathcal{APX}$-hard* if any problem in $\mathcal{APX}$ can be reduced to it by an L-reduction; $\mathcal{APX}$-hard problems do not have polynomial time approximation schemes unless $\mathcal{P} = \mathcal{NP}$ [58,59].

An *L-reduction* from an optimization problem P_1 to an optimization problem P_2 consists of two polynomially computable functions t_1 and t_2 with the following properties:

1. t_1 maps instances of P_1 to instances of P_2 such that

   $$opt_{P_2}(t_1(I)) \leq \beta_1\, opt_{P_1}(I)$$

   for any instance I of P_1. Here $opt_{P_i}(X)$ denotes the optimum value for instance X of problem P_i, and β_1 is some constant.
2. t_2 associates with any instance I of P_1 and any feasible solution S' of the corresponding instance $I' := t_1(I)$ of P_2 a feasible solution $S := t_2(I, S')$ of I such that

   $$|opt_{P_1}(I) - val_{P_1}(I, S)| \leq \beta_2 |val_{P_2}(I', S') - opt_{P_2}(I')|.$$

   Here, $val_{P_i}(X, Y)$ denotes the objective function value of the feasible solution Y for instance X of problem P_i, and β_2 is some constant.

L-reductions preserve approximability. If S' is an ϵ-approximation of the optimum solution of I', i.e., $|opt_{P_2}(I') - val_{P_2}(I', S')| \leq \epsilon \cdot opt_{P_2}(I')$, then $|opt_{P_1}(I) - val_{P_1}(I, S)| \leq \beta_2 \cdot \epsilon \cdot opt_{P_2}(I') \leq \beta_1 \cdot \beta_2 \cdot \epsilon \cdot opt_{P_1}(I)$, i.e., S is a $\beta_1\beta_2\epsilon$-approximation to the optimum solution of I.

6.1. *Strictly fundamental cycle bases*

Recall that a strictly fundamental cycle basis consists of the fundamental circuits with respect to some spanning tree. We saw in Theorem 4.11 that any graph has a strictly fundamental basis of weight $O(W \log^2 n \log\log n)$ and of length $O(n^2)$. Deo et al. [27] showed the $\mathcal{NP}$-hardness of the minimum strictly fundamental cycle basis problem. We will now sketch a proof of its $\mathcal{APX}$-hardness [60]. The proof consists of an L-reduction from the following special case of the maximum satisfiability problem.

MAX-3-SAT-NAE-UN-9: Given a set $X = \{x_1, \dots, x_n\}$ of Boolean variables and a collection $\mathcal{C} = \{C_1, \dots, C_m\}$ of disjunctive clauses with exactly 3 variables per clause, where all variables appear unnegated and each occurs in at most 9 clauses, find a truth assignment to the variables maximizing the number of clauses containing both a *true* and a *false* variable.

MAX-3-SAT-NAE-UN-9 is a not-all-equal version of MAX-3-SAT restricted to instances with unnegated variables, each

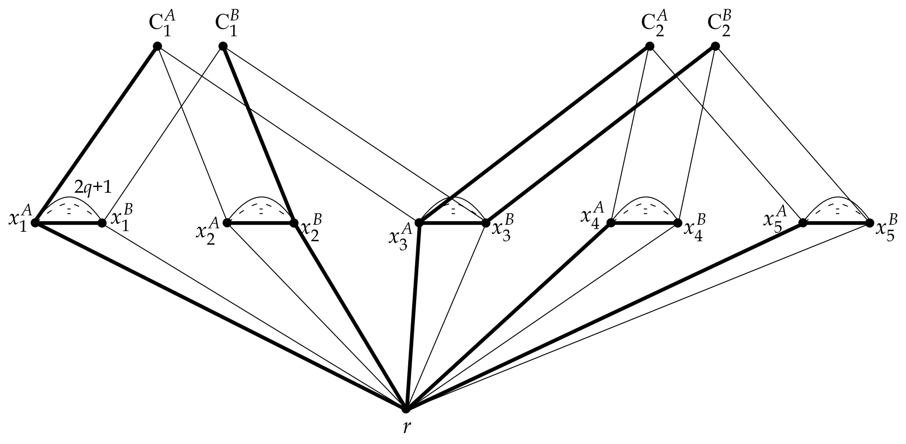


Fig. 29 – The graph G_I associated with the MAX-3-SAT-NAE-UN-9 instance I with variable set $X = \{x_1, x_2, x_3, x_4, x_5\}$ and clause collection $\mathcal{C} = \{x_1 \vee x_2 \vee x_3, x_3 \vee x_4 \vee x_5\}$. The spanning tree T of G_I derived from the truth assignment $\Phi = (true, false, true, true, true)$ is shown in bold.

variable having at most 9 occurrences. In [60], MAX-3-SAT-NAE-UN-9 was shown to be $\mathcal{APX}$-hard by means of a sequence of standard L-reductions starting from MAX CUT-3, the problem of finding a cut containing the maximum number of edges in an undirected graph where all vertices have a degree of at most 3. In turn, MAX CUT-3 has been shown to be $\mathcal{APX}$-hard in [61].

The main reduction. We now describe the L-reduction from MAX-3-SAT-NAE-UN-9 to the minimum strictly fundamental cycle basis problem (MSFCB). Let q and M be integer constants, which we will fix later. Let I be an instance of MAX-3-SAT-NAE-UN-9 with variable set $X = \{x_1, \ldots, x_n\}$ and clause collection $\mathcal{C} = \{C_1, \ldots, C_m\}$. We construct an instance I' of MSFCB, i.e., a weighted graph G_I, as follows. The set of vertices is

$$V(G_I) = \{r, c_1^A, c_2^A, \ldots, c_m^A, c_1^B, c_2^B, \ldots, c_m^B, x_1^A, x_2^A, \ldots, x_n^A, x_1^B, x_2^B, \ldots, x_n^B\},$$

and the edges in $E(G_I)$ together with the corresponding weights are:

- for each $i = 1, 2, \ldots, n$, we have two edges $\{r, x_i^A\}$ and $\{r, x_i^B\}$, each of weight 1;
- for each $i = 1, 2, \ldots, n$, we have $2q + 1$ parallel edges connecting vertices x_i^A and x_i^B, and all of them have weight 1;
- for each $j = 1, 2, \ldots, m$ and for each variable x_i occurring in C_j, we have the edges $\{c_j^A, x_i^A\}$ and $\{c_j^B, x_i^B\}$, each of weight M.

We remark that edges of weight M may be replaced by a path of length M and parallel edges may be split by an additional intermediate vertex. In other words, the graph G_I could also be constructed as an unweighted simple graph. The following easy-direction lemma indicates the intention behind the reduction.

Lemma 6.1. *Let $\Phi : \{x_1, x_2, \ldots, x_n\} \mapsto \{true, false\}^n$ be a truth assignment such that there are t clauses in $\mathcal{C}$ containing both a variable with value true and a variable with value false. Then G_I has a fundamental cycle basis of weight $n(4q+3) + m(8M+12) - t$.*

Proof. By relabeling the clauses, we can assume w.l.o.g. that, for $j = 1, 2, \ldots, t$, clause C_j contains a variable with value *true* as well as a variable with value *false*. So, for $t < j \le m$, all variables in C_j have the same truth value under Φ. Construct a spanning tree T as follows (Fig. 29):

- for each $i = 1, 2, \ldots, n$, include a single edge connecting x_i^A and x_i^B;
- for each $i = 1, 2, \ldots, n$, include $\{r, x_i^A\}$ if $\Phi(x_i) = true$ and include $\{r, x_i^B\}$ if $\Phi(x_i) = false$;
- for each clause C_j, with $1 \le j \le t$, select one variable $\bar{x}$ with $\Phi(\bar{x}) = true$ and one variable $\tilde{x}$ with $\Phi(\tilde{x}) = false$, and include the edges $\{c_j^A, \bar{x}^A\}$ and $\{c_j^B, \tilde{x}^B\}$;
- for each clause C_j, with $j = t + 1, \ldots, m$, select a single arbitrary variable x_i occurring in C_j and include both edges $\{c_j^A, x_i^A\}$ and $\{c_j^B, x_i^B\}$.

We next compute the costs of the fundamental cycles induced by the non-tree edges. We distinguish the following cases:

- For each $i = 1, 2, \ldots, n$, T contains exactly one of the two edges $\{r, x_i^A\}$ or $\{r, x_i^B\}$. The other edge induces a fundamental cycle of cost 3, for a total of $3n$.
- For each $i = 1, 2, \ldots, n$, $2q$ edges connecting vertices x_i^A and x_i^B are not in T. Each of them induces a fundamental cycle of cost 2, for a total cost of $4qn$.
- For each $j = 1, 2, \ldots, t$, the four non-tree edges incident to c_j^A or c_j^B induce four cycles. Exactly one of these cycles has cost $2M + 2$, while the others have cost $2M + 3$. The corresponding costs sum to $t(8M + 11)$.
- For each $j = t+1, \ldots, m$, each one of the two non-tree edges incident to c_j^A induces a cycle of cost $2M + 2$ if all variables in C_j are *true* and of cost $2M + 4$ if all these variables are *false*. Analogously, each one of the two non-tree edges incident to c_j^B induces a cycle of cost $2M + 4$ if all variables in C_j are *true* and of cost $2M + 2$ if all these variables are *false*. These costs sum to $(m - t)(8M + 12)$.

Therefore the fundamental cycle basis induced by T has a cost of

$$3n + 4qn + t(8M + 11) + (m - t)(8M + 12)$$
$$= n(4q + 3) + m(8M + 12) - t. \quad \square$$

A key property of the reduction is that the type of spanning tree considered in the lemma above gives rise to a minimum strictly fundamental basis.

Definition 6.1. A spanning tree T of G_I is *well-behaved* if it satisfies the following properties:

1. for each $j = 1, 2, \ldots, m$, the vertices c_j^A and c_j^B have degree 1 in T,
2. for each $i = 1, 2, \ldots, n$, exactly one edge of the $2q + 1$ edges $\{x_i^A, x_i^B\}$ belongs to T,
3. for each $j = 1, 2, \ldots, m$, either for some i both edges $\{c_j^A, x_i^A\}$ and $\{c_j^B, x_i^B\}$ belong to T or for some i_1 and i_2, with $1 \leq i_1 \leq n$, $1 \leq i_2 \leq n$ and $i_1 \neq i_2$, both edges $\{c_j^A, x_{i_1}^A\}$ and $\{x_{i_1}^A, r\}$ as well as both edges $\{c_j^B, x_{i_2}^B\}$ and $\{x_{i_2}^B, r\}$ belong to T.

Lemma 6.2 ([60]). *Assume $q \geq 9$ and $M \geq 4$. For any spanning tree T of graph G_I, we can, in polynomial time, derive from T a well-behaved spanning tree T' such that the weight of the basis induced by T' is no larger than the weight of the basis induced by T.*

We can now state the main result of the section.

Theorem 6.3 ([60]). *The minimum strictly fundamental cycle basis problem is $\mathcal{APX}$-hard.*

Proof. It suffices to verify that the reduction presented is an L-reduction. For any instance $I = (X, \mathcal{C})$ of MAX-3-SAT-NAE-UN-9, the corresponding instance I' of MSFCB can obviously be constructed in polynomial time.

The simple randomized argument implying that any MAX-SAT instance with m clauses admits a truth assignment satisfying at least $m/2$ clauses, is also valid for MAX-3-SAT-NAE-UN-9. Thus $opt(I) \geq m/2$.

According to Lemma 6.1, where $q = 9$, we have $opt(I') \leq n(4q + 3) + m(8M + 12)$. Since we may assume that $n \leq 3m$ (otherwise some variable would occur in no clause), $opt(I') \leq 3m(4q+3)+m(8M+12) = m(12q+8M+21) \leq m(24q+16M+42)opt(I)$. We set $\beta_1 = 24q + 16M + 42$.

By Lemma 6.2, from any spanning tree T of G_I we can derive a well-behaved spanning T' without increasing the weight of the associated fundamental cycle basis. Now, the three properties characterizing well-behaved spanning trees make sure that it is possible to reverse the construction described in the proof of Lemma 6.1. Therefore, to derive a truth assignment Φ from any spanning tree T of G_I, it suffices first to derive a well-behaved spanning tree T' from T and then to set $\Phi(x_i) = true$ when $\{x_i^A, r\} \in T'$, and $\Phi(x_i) = false$ when $\{x_i^B, r\} \in T'$. Condition (ii) of an L-reduction is then satisfied with $\beta_2 = 1$. $\square$

We close this section with some open problems.

Open Problem 12. Is there an $O(\log n)$ approximation algorithm for minimum F-bases? Is there one for planar graphs? We remark that the approximability of the bottleneck version, in which one looks for a strictly fundamental cycle basis where the weight of the maximum cycle is minimum, has been addressed in [62].

Open Problem 13. Is the minimum F-basis problem in $\mathcal{APX}$, i.e., does it have constant factor approximation?

Open Problem 14. What is the complexity of the minimum F-basis problem for planar graphs? We remark that the related problem of computing a spanning tree with shortest fundamental circuit is NP-complete for planar graphs [63].

6.2. *Weakly fundamental cycle bases*

We know from Theorem 4.4 that any graph has a weakly fundamental cycle basis (W-basis) of weight $O(W \log n)$. Thus the weight of a minimum W-basis can be approximated within a factor of $O(\log n)$; no better approximation factor is known. Rizzi [17] has shown that the minimum W-basis problem is $\mathcal{APX}$-hard and we will sketch his proof in this section.

Open Problem 15. Is the minimum W-basis problem in $\mathcal{APX}$?

We first introduce a compact way of representing W-bases. For T a spanning tree of G, any ordering $e_1, e_2, \ldots, e_\nu$ of the non-tree edges is called a *removal sequence*. Let $\mathcal{C}$ be a W-basis of a connected graph G. If $\mathcal{C} \neq \emptyset$, that is, if G is not a tree, then there exists an edge e of G that is contained in precisely one circuit of $\mathcal{C}$. Let C_e be the only circuit in $\mathcal{C}$ that contains e. Notice that $G \setminus e$ is connected and $\mathcal{C} \setminus C_e$ is a W-basis of $G \setminus e$. If this process is iterated over $G \setminus e$, we end up with a spanning tree T of G. Furthermore, if we label the i-th edge that has been removed in the process as e_i, then the sequence $s = e_1, e_2, \ldots, e_\nu$ is a removal sequence having the edges in T as tree edges and certifying that $\mathcal{C}$ is a W-basis according to Definition 3.1. We say that the spanning tree T, the ordering $e_1, e_2, \ldots, e_\nu$, and the W-basis $\mathcal{C}$, from which we started, are *compatible*. Notice that at any iteration of the edge removal process it may be possible that more than one edge of the current graph is contained in precisely one circuit of $\mathcal{C}$. Actually, there is always some freedom of choice when removing the last edge. Thus, in general, a W-basis of G may be compatible with more than one spanning tree of G. Furthermore, even w.r.t. any particular tree T of G, a W-basis may be compatible with more than one ordering of the edges of $G \setminus T$. Conversely, any removal sequence $e_1, e_2, \ldots, e_\nu$ of G, might be compatible with more than one W-basis of G. However, among the W-bases of G which are compatible with the ordering $e_1, e_2, \ldots, e_\nu$, we can efficiently find one of minimum cost by resorting to any shortest-path algorithm as a subroutine. And we can also enforce the uniqueness of this W-basis by adopting a lexicographic scheme to resolve ties among circuits of the same weight. Indeed, given any removal sequence $e_1, e_2, \ldots, e_\nu$ of G, the unique W-basis of G associated with the sequence $e_1, e_2, \ldots, e_\nu$ is obtained as described in AlgorithmW-DECODER.

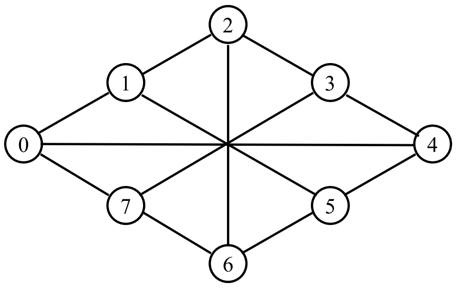


Fig. 30 – Wagner's graph V_8.

```
Algorithm W-Decoder ($e_1, e_2, \ldots, e_\nu$)
start with $G' := G$, $\mathcal{C} = \emptyset$, and,
 for $i = 1, 2, \ldots, \nu$, do,
   add to $\mathcal{C}$ the unique[11] cheapest circuit of $G'$ containing
   edge $e_i$;
   remove edge $e_i$ from $G'$.
return $\mathcal{C}$.
```

Notice that not every W-basis of G admits a removal sequence encoding it such that the above algorithm can reproduce it. We say that a W-basis $\mathcal{C}$ of G is *locally-optimal* if there exists a removal sequence s of G such that the execution of Algorithm W-Decoder (s) produces $\mathcal{C}$. Indeed, the above remarks inspire a natural local search approach for the minimum W-basis problem, where, given any WFCB $\mathcal{C}$ of G, we first obtain a removal sequence s compatible with $\mathcal{C}$ and then substitute $\mathcal{C}$ with the W-basis $\mathcal{C}'$ produced by Algorithm W-Decoder (s). Notice that $\mathcal{C}'$ is locally-optimal and its cost never exceeds the cost of $\mathcal{C}$.

A fundamental gadget: Our $\mathcal{APX}$-hardness proof is based on a single gadget. The gadget is derived from a graph first described by Leibchen and Rizzi [2]; every W-basis of this graph is strictly more expensive than the cheapest undirected cycle basis. Indeed, since the minimum U-basis problem is in $\mathcal{P}$, the graphs produced by a reduction from a generic $\mathcal{APX}$-hard optimization problem to the minimum W-basis problem are bound to involve such graphs.

[11] Uniqueness is enforced by the adoption of a lexicographic scheme.

Although, the inapproximability result also holds in the unweighted case, we find it convenient to allow the use of small natural weights (actually, all in $\{1, 2, 3\}$) in the constructions and in the gadgets to follow. Clearly, an edge of weight w may be replaced by a path of w edges and $w - 1$ new intermediate nodes without changing the essence of the cycle basis problem. The transformation is polynomial as long as the weights are polynomially bounded. So there is no harm in using small integer weights. We start by describing a graph for which no minimum U-basis is weakly fundamental. Consider first the graph V_8 in Fig. 30. Here, $m = 12$, $n = 8$, $\nu = 5$ and the circuits $C_1 = 2-3-7-6$, $C_2 = 3-4-0-7$, $C_3 = 4-5-1-0$, $C_4 = 5-6-2-1$, and $C_5 = 6-5-4-3-2$ form a W-basis of V_8. Indeed, the edge $\{6, 7\}$ is only contained in C_1, $\{0, 7\}$ only in C_2, $\{0, 1\}$ only in C_3, and $\{1, 2\}$ only in C_4. For later convenience, we also certify the independence of C_1 to C_5 by giving five sets Σ_1 to Σ_5 such that Σ_i and C_j have an odd-sized intersection if and only if $i = j$.

$\Sigma_1 = \{\{2, 6\}, \{5, 6\}\}$ with odd-sized intersection only with C_1 (see Fig. 31 on the left);
$\Sigma_2 = \{\{2, 6\}, \{5, 6\}, \{3, 7\}\}$ with odd-sized intersection only with C_2 (see Fig. 31 on the right);
$\Sigma_3 = \{\{1, 5\}, \{2, 3\}, \{2, 6\}\}$ with odd-sized intersection only with C_3 (mirror image of Σ_2);
$\Sigma_4 = \{\{2, 3\}, \{2, 6\}\}$ with odd-sized intersection only with C_4 (mirror image of Σ_1);
$\Sigma_5 = \{\{2, 3\}, \{2, 6\}, \{5, 6\}\}$ with odd-sized intersection only with C_5 (see Fig. 31 in the middle).

We will use the name *petal* for the weighted graph obtained from V_8 by assigning cost 2 to the edges $\{6, 7\}$, $\{7, 0\}$, $\{0, 1\}$, $\{1, 2\}$, also called *glue edges*, and assigning cost 1 to all other edges, called *internal edges*, Notice that all circuits of the petal have a cost of at least 5, and C_1, C_2, C_3, C_4, C_5 are actually its only 5 circuits of a cost of precisely 5, hence they form the unique minimum basis of the petal. A removal sequence compatible with this locally-optimal W-basis of the petal is illustrated in Fig. 32.

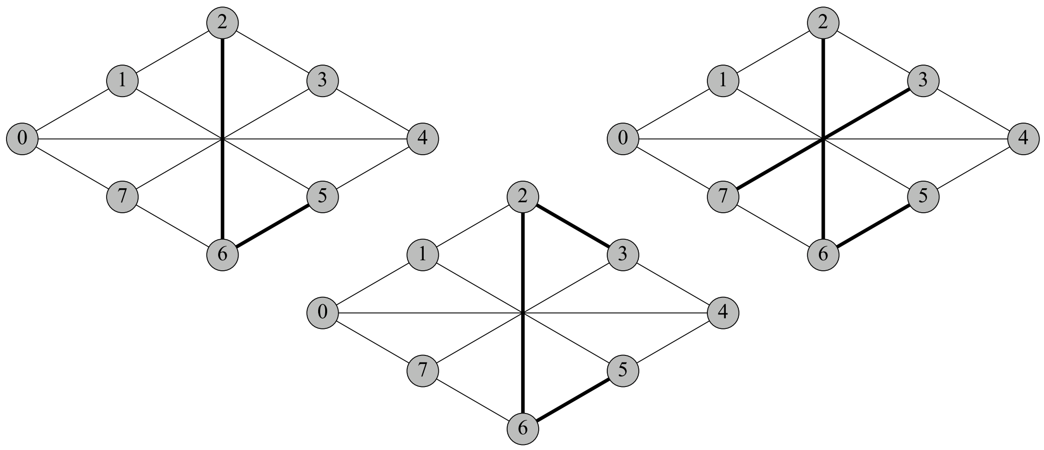


Fig. 31 – Certificates of independence for C_1 (left), C_2 (right), and C_5 (middle).

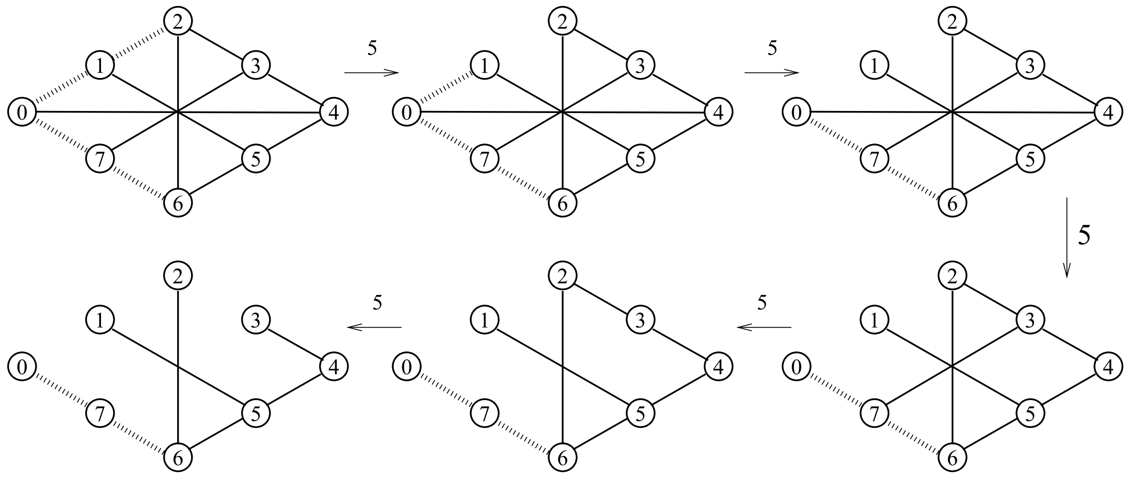


Fig. 32 – A removal sequence of a W-basis of weight 25.

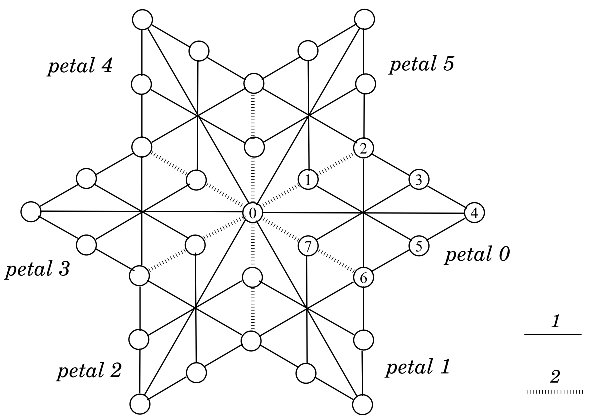


Fig. 33 – A weighted graph F whose unique minimum U-basis is not weakly fundamental.

Consider now the weighted graph F obtained by gluing together 6 distinct petals as shown in Fig. 33: Let petal i, $0 \leq i \leq 5$, be on nodes $\{v_0^i, v_1^i, \ldots, v_7^i\}$; we glue the petals through the following node identifications: $v_0^0 \leftrightarrow v_0^1 \leftrightarrow v_0^2 \leftrightarrow v_0^3 \leftrightarrow v_0^4 \leftrightarrow v_0^5$, and $v_1^{i+1 \bmod 6} \leftrightarrow v_7^i$ and $v_2^{i+1 \bmod 6} \leftrightarrow v_6^i$ for $0 \leq i \leq 5$. Edges that become parallel through this identification process are replaced by a single edge of weight 2.

Clearly, $n_F = 31$, $m_F = 60$, and $\nu_F = 30$. The six copies of the circuits C_1, C_2, C_3, C_4, and C_5 form a collection of 30 circuits in F whose independence can be established by taking the 6 corresponding copies of each of the odd sets $\Sigma_1, \Sigma_2, \Sigma_3, \Sigma_4, \Sigma_5$. Hence, these 30 circuits form a U-basis of F. Each of these 30 circuits has weight 5. Every other circuit of F has cost at least 6; hence these 30 circuits form the unique minimum U-basis of F. This cycle basis is not weakly fundamental since each edge of F is contained in at least 2 of these circuits.

It is also relevant to our discussion to exhibit a cheap W-basis of F. In fact, F has a W-basis whose weight is only one larger than the weight of the unique minimum W-basis introduced above. Indeed, consider first one single petal of F and its W-basis as encoded by the removal sequence displayed in Fig. 34. This W-basis has weight 26 and leaves all glue edges as tree edges. It is easy to extend this removal sequence for one petal, say petal 0, to a removal sequence for F that encodes a W-basis of weight 151: simply append to it, for each one of the other 5 petals taken in clockwise order, a removal sequence like the one in Fig. 32 (see the proof of Fact 6.1 for the details).

Fact 6.1. *There are precisely 30 circuits of cost 5 in F. These 30 circuits constitute the unique minimum U-basis of F. This basis has weight 150 and is not weakly fundamental. Furthermore, F admits a W-basis of weight 151.*

Proof. AlgorithmWFCB-DECODER constructs a W-basis of weight 151 from the following removal sequence: first, within petal 0, remove $v_5^0v_6^0$, $v_2^0v_6^0$, $v_3^0v_7^0$, $v_0^0v_4^0$, and $v_4^0v_5^0$ in this order; next, for $i = 1, 2, 3, 4, 5$ and in sequence within petal i, remove $v_1^iv_2^i$, $v_0^iv_1^i$, $v_0^iv_4^i$, $v_3^iv_7^i$, and $v_2^iv_3^i$ in this order. □

We are now ready to produce a weighted graph G (the gadget) with the following properties: (1) G contains 4 nodes x, y, z and w, and the edges wx, wy and wz, all of weight 1; (2) The minimum U-basis of G has weight B and G has a W-basis of weight B. (3) Let T be any spanning tree of G, compatible with some W-basis of weight B. Then, the distance in T between any two of the 4 nodes x, y, z and w is at least 3; (4) G has a W-basis of weight $B + 1$, for which $wx, wy, wz \in T$.

The intended functioning of the gadget is as follows. Several copies of the gadget will be part of the graph G_H representing the minimum W-basis problem instance constructed by the L-reduction proposed in Section 6.2. Each such gadget (copy) is attached to the rest of G_H by means of the 4 nodes x, y, z and w, and, actually, occurs as an induced subgraph of G_H. Each removal sequence for G_H contains, in a natural way, a removal sequence for each one of these gadgets. Indeed, a set of edges whose removal makes G_H acyclic also intersects all circuits of any given subgraph of G_H. Notice that after the removal of the sole edges prescribed by a removal sequence for a gadget, the nodes of that gadget are still connected within the gadget. In order to intersect all circuits of G_H, a removal sequence for G_H will need to include further edges. When obtaining a short W-basis of G_H, the following Boolean choice has to be made for each one of these gadgets:

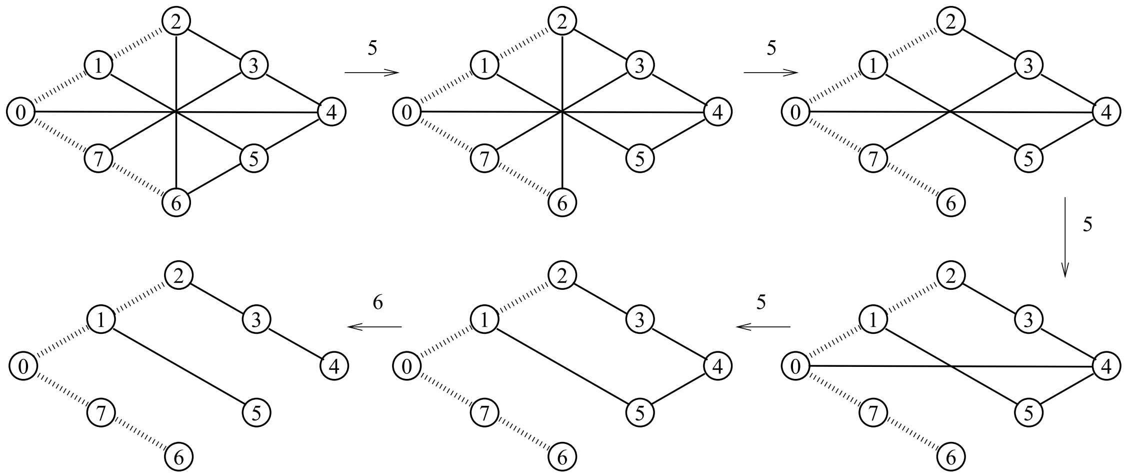


Fig. 34 – The removal sequence of a W-basis of weight 26. Notice that all glue edges are tree edges.

either locally pay $B + 1$: pay the extra price of $+1$ by locally applying a removal sequence avoiding the "cheap" edges wx, wy, and wz, hence making it possible to disconnect cheaply the 4 nodes w, x, y, and z with later removals;
or locally pay B: pay just the minimum B, but then these 4 nodes will remain connected within the gadget since disconnecting them later would cost significantly more than the $+1$. Indeed, by points (3) and (1) above, the cost of disconnecting any two nodes among the 4 nodes w, x, y will be at least $3 - 1 = 2$.

The above informal description will make full sense only later, after all the pieces of the proposed reduction are in place. We now present our gadget; see Fig. 35. The weighted subgraph of G induced by the nodes a, b, c, y and w is called *the chamber*. The gadget graph G is similar to the flower F from Fig. 33 but it has 14 petals and 1 chamber, plus two edges wx and wz, dubbed the *jump edges*. In Fig. 35, the 14 petals are only hinted at for reasons of legibility. The numbers that label some of the nodes represent the distances from node w within the weighted graph $G \setminus \{wx, wz\}$ and, as such, certify the truth of the last two properties listed in the following lemma.

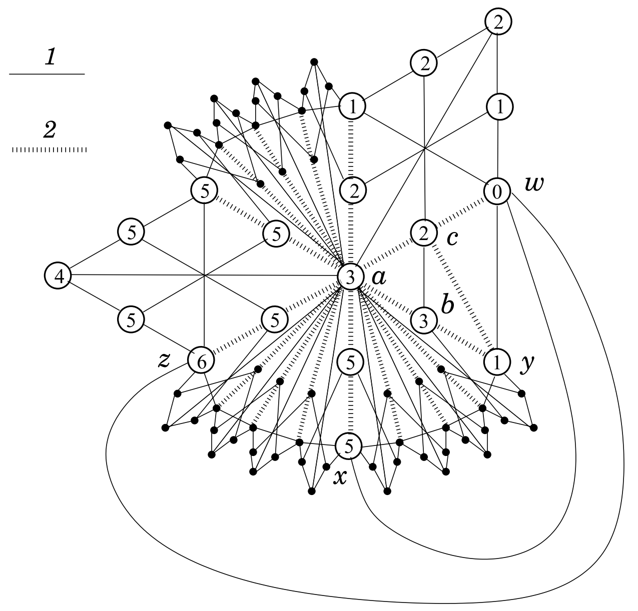


Fig. 35 – A gadget graph G with 11 petals, one chamber, and two jump edges wx and wz.

Lemma 6.4. *The graph G in Fig. 35 has 14 petals, one chamber, $n = 73$, $m = 147$, and $\nu = 75$. It has has a unique minimum U-basis $\mathcal{C}_{min}$; this basis is weakly fundamental and has weight $B := 377$. The edges wx, wy, and wz are all non-tree edges w.r.t. any removal sequence encoding $\mathcal{C}_{min}$. There exists a W-basis of G of weight $B+1$, and a removal sequence encoding this basis, w.r.t. which the edges wx, wy, and wz are all tree edges. The distance between w and y in $G \setminus yw$ is 4. The distance between w and x in $G \setminus xw$ is 5. The distance between w and z is 5 in $G \setminus zw$ and 6 in $G \setminus \{zw, xw\}$. The distance between any two nodes in $\{x, y, z\}$ is at least 4 in $G \setminus \{xw, yw, zw\}$.*

Proof. All claims in the first sentence are readily verified. As for the last four sentences, their truth can be readily verified through shortest-path computations, and the distance values reported in Fig. 35 may partially support the reader in this task.

The remaining properties follow from the properties of the petals and from the structure of G. We refer to Rizzi [17] for detailed arguments. □

The source problem of the L-reduction: Hypergraphs generalize graphs. A hypergraph is a pair $H = (V, E)$, where V is a finite set of nodes and E is a finite set of *hyperedges*. A hyperedge is a set of nodes. When all hyperedges have size t, H is called *t-uniform*. Graphs are 2-uniform hypergraphs. A circuit in a hypergraph is an alternating sequence of nodes and edges $v_0, e_0, v_1, e_1, v_2, \ldots, v_k, e_k$ such that, for every $i = 0, 1, 2, \ldots, k$, we have $v_i \in e_i$ and $v_{i+1 \bmod k} \in e_i$. The *length* of the circuit is k, the number of edges comprising it. The *girth* γ_H of a hypergraph H is the minimum length of a circuit in H. A hypergraph is *acyclic* if it contains no circuit. A *feedback hyperedge set (FHS)* of a hypergraph $H = (V, E)$ is a set $F \subseteq E$ such that $(V, E \setminus F)$ is acyclic. Given a hypergraph H, the MINIMUM FEEDBACK HYPEREDGE SET (MFHS) problem seeks a minimum cardinality FHS in H.

Lemma 6.5 ([17]). *There exists a constant $\alpha > 0$ such that the* MFHS *problem is $\mathcal{APX}$-hard even when restricted to 4-uniform hypergraphs with $\gamma \geq 6$ in which a minimum cardinality FHS has a size of at least $\alpha |E|$.*

The main reduction: The main *L*-reduction is from the MFHS problem to the minimum W-basis problem. Let $H = (V, E)$ be a 4-uniform hypergraph with $\gamma_H \geq 6$. Consider the weighted graph G_H obtained as follows:

1. Start from node set V;
2. Add a further node r adjacent to all nodes in V through edges of weight 3;
3. For each hyperedge $e = \{v_1, v_2, v_3, v_4\} \in E$, add a new and private copy C_e of the gadget graph and perform the following 4 node identifications: $v_1 \leftrightarrow x$, $v_2 \leftrightarrow y$, $v_3 \leftrightarrow z$, and $v_4 \leftrightarrow w$.

The following lemma establishes that the above poly-time construction is an *L*-reduction from the MFHS problem restricted to instances conforming to the properties in Lemma 6.5 to the minimum W-basis problem. As a consequence, the MWFCB problem is $\mathcal{APX}$-hard.

Lemma 6.6. *The hypergraph $H = (V, E)$ admits an FHS of size t iff G_H admits a* W*-basis of weight $(21 + B)m + t$, where $n = |V|$, $m = |E|$.*

Proof. Assume first that the hypergraph $H = (V, E)$ admits an FHS $F \subseteq E$ with $|F| = t$. We construct a W-basis $\mathcal{C}_F$ of G_H and a removal sequence s encoding $\mathcal{C}_F$. We start with $\mathcal{C}_F := \emptyset$ and $s := \emptyset$ and set $G' := G_H$.

For each $e \in F$, we proceed as follows. First, at cost $(B + 1)$, we put in $\mathcal{C}_F$ all circuits of the W-basis of C_e of weight $(B + 1)$. By Lemma 6.4, this basis can be encoded by a removal sequence with respect to which the edges xw, yw, and zw of C_e are all tree edges. Append this removal sequence to s meanwhile removing from G' the 75 edges it prescribes. After the removal of these edges C_e is acyclic. Furthermore, the only edges of C_e which are not bridges of G' are the edges xw, yw, and zw. Next, at cost $7+7+7 = 21$, remove from G' these three edges of component C_e, meanwhile appending them to s and adding to $\mathcal{C}_F$ the three triangles they form together with node r. Each of these triangles costs $7 = 3 + 3 + 1$. Clearly, after the removal of these three edges, no circuit of G' can go through an edge of C_e. After this has been performed for each $e \in F$, we have paid $(B + 1 + 21)\,t = (21 + B)\,t + t$ in total.

At this point, the number of connected components of $(V, E \setminus F)$ is $n - 3(m - t)$; for each connected component C of $(V, E \setminus F)$, remove from G' all edges of the form rc, $c \in C$, except one. Each removal has cost $7 = 3 + 3 + 1$ (explained in more detail below) and adds a triangle through node r to $\mathcal{C}_F$. Once these edges have been removed, no circuit of G' contains r. We now explain in more detail how the removal of these edges can be performed within the claimed costs. First, select a node a of C and a spanning tree A of $G'[C]$. Then, let $A' := A$. Consider A' as a tree rooted at a. While $A' \neq \{a\}$, consider any leaf q of A' and let p be the father of q in A'; remove from G' the edge qr and append it to s, meanwhile inserting into $\mathcal{C}_F$ the triangle $q - r - p$ (at a cost of $7 = 3 + 3 + 1$); remove node q from the rooted tree A'. In this way, we remove a total of $n - (n - 3(m - t)) = 3(m - t)$ edges, for a total cost of $21(m - t)$. Up to this point, we have paid $(B + 1)\,t + 21\,m$ in total.

Finally, for each $e \in E \setminus F$, put in $\mathcal{C}_F$ all circuits of the W-basis of C_e of weight B. Also, append to s and remove from G' all non-tree edges of C_e w.r.t. any removal sequence encoding this W-basis of C_e. After this, G' is a spanning tree of G_H. In particular, the acyclicity of G' follows from the acyclicity of $E \setminus F$. Thus, $\mathcal{C}_F$ is a W-basis of G_H. In total, this last step has cost $(m - t)\,B$ and hence the total cost of $\mathcal{C}_F$ is $(21 + B)\,m + t$.

For the reverse direction, let $\mathcal{C}$ be a W-basis of G_H of a cost of at most $(21 + B)\,m + t$. We may assume $\mathcal{C}$ to be locally-optimal and encoded by means of a removal sequence s. Let T be the spanning tree of G_H made of the tree edges w.r.t. s. Let E' be the set of those hyperedges $e \in E$ such that $C_e \cap T$ is a spanning tree of C_e. Notice that the hypergraph (V, E') is acyclic since T is acyclic. Let $F := E \setminus E'$. It follows that F is an FHS of the hypergraph H. Let $f := |F|$. We will show that $|F| \leq t$.

Let ν denote the cyclomatic number of G_H. By Lemma 6.4, the cyclomatic number of each gadget C_e is 75. Therefore $\nu = 75\,m + 3\,m = 78\,m$ since, to make G_H acyclic, we need to remove $3\,m$ further edges after having made each C_e acyclic. Let $G^0 := G_H$, and, for $i = 1, 2, \ldots, \nu$, let G^i be the weighted graph obtained from G^{i-1} by removing the i-th edge e_i from the removal sequence s. For every $e \in E$, we denote by V_e the nodes of the gadget C_e, and we say that edge e_i is *pertinent* to C_e if e_i is an edge of C_e and if the induced graphs $G^i[V_e]$ and $G^{i-1}[V_e]$ have the same number of connected components. Clearly, the removal sequence $s = e_1, e_2, \ldots, e_\nu$ contains precisely $\nu_{C_e} = 75$ edges pertinent to C_e and the subsequence s_e of s comprising these 75 edges encodes a W-basis of C_e. Since the girth of H is at least 6, every circuit of G_H which is not a circuit of some C_e costs at least 7. As a consequence, for every $e \in E$, the total cost of the circuits in $\mathcal{C}$ associated with the edges in s_e is at least B. Besides the $\nu = 75\,m$ removals considered up to now (which are precisely enough to make each C_e gadget acyclic), we have $3\,m$ further removals. None of these further removals can cost less than 7, since none of the corresponding circuits in $\mathcal{C}$ is entirely contained in one single C_e gadget. Furthermore, for every $e \in F$, the best edge removal for making $G_H[V_e]$ acyclic and disconnecting the subgraph $G_H[V_e] \cap T$ has cost $B + 1$. Indeed, for every $e \in F$, there exists an e_i in s such that $G^i[V_e]$ and $G^{i-1}[V_e]$ have the same number of connected components. As a consequence, the corresponding circuit C_i in $\mathcal{C}$ contains an edge (the edge e_i) in C_e but is not entirely contained in C_e. Now, if e_i is neither the xw, nor the yw, nor the zw edge of C_e, then the cost of C_i is strictly greater than 7 (actually, at least 10); otherwise, the total cost of the circuits in $\mathcal{C}$ associated with the edges in s_e is at least $B + 1$. In total, the cost is at least $(m - f)B + f(B + 1) + 21\,m = m\,B + 21\,m + f$. Since we know that this cost is at most $(21 + B)\,m + t$, we conclude $f \leq t$. □

7. Applications

Cycle bases arise in a wide range of engineering situations. Here, we discuss three of them, which require different kinds of cycle bases: The analysis of electrical circuits can be carried out with any kind of cycle basis, whereas solving periodic scheduling problems in traffic planning require integral bases, and a graph drawing method requires strictly fundamental bases.

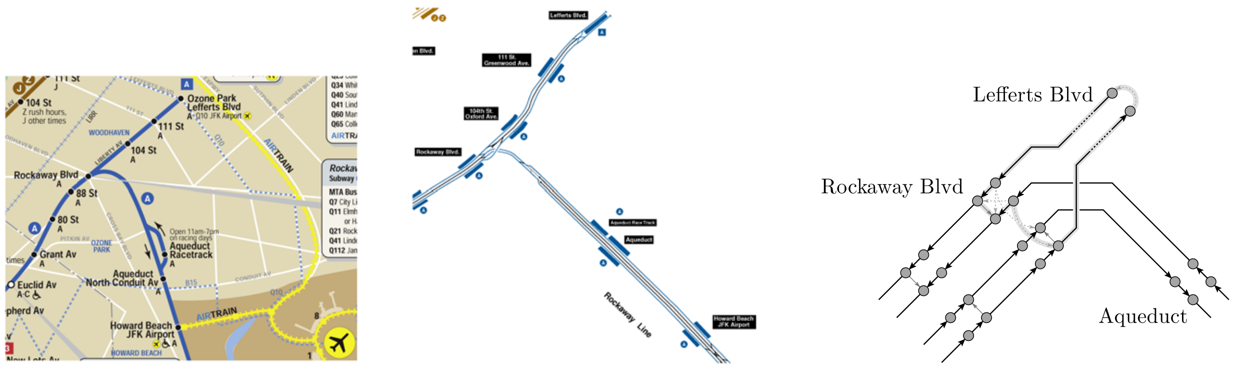


Fig. 36 – The representations of the the New York City subway network near JFK airport: the line plan, the track map, and the PESP graph. In the lattermost, light gray dotted arcs model passenger transfers or turnarounds of trains, and dark gray solid arcs model minimum headways and/or coordinated departures. A circuit in the PESP graph is highlighted.

7.1. *Kirchhoff's voltage law*

Kirchhoff's circuits laws govern the behavior of electrical circuits. The zero-sum property states that the directed sum of the electrical potential differences around any closed circuit must be zero [64]. Cycle bases are relevant for circuit analysis, since the zero-sum property holds for all circuits if it holds for the circuits in a basis. Thus, circuit analysis can restrict attention to the circuits in a basis. Indeed, consider the cycle matrix Γ of some directed cycle basis and some arbitrary cycle C. Then $C = \Gamma\lambda$ for some coefficient vector λ. Now, if some vector x of potential differences satisfies Kirchoff's law for every circuit in the basis, i.e., $x^T\Gamma = 0$, then $x^TC = x^T(\Gamma\lambda) = (x^T\Gamma)\lambda = \mathbf{0}^T\lambda = 0$. For a more detailed exposition of this application of cycle bases, we refer the reader to Bollobás [65]. An in-depth presentation of how cycle bases can be used for index reduction of differential algebraic systems is given in Bächle [66].

7.2. *Periodic scheduling in traffic planning*

Periodic scheduling problems arise frequently in traffic planning. Two examples are scheduling traffic lights and timetabling public transport. They share a common mathematical model that can be traced back to early work by Gartner et al. [67] and Rüger [68] and that was put into its final form by Serafini and Ukovich [69].

In the *Periodic Event Scheduling Problem (PESP)* we are given a directed graph $D = (V, A)$, vectors ℓ and u on the arcs, and a scalar T called *cycle time* or *period*. For an arc a, $\ell(a)$ and $u(a)$ are lower and upper bounds, respectively, for the travel time across a. In the feasibility version of the problem, the question is whether or not a node potential π exists, such that

$$\ell_a \le \pi_j - \pi_i + Tp_a \le u_a, \quad \forall a = (i, j) \in A,$$

where p constitutes an *integral* vector on the arcs. Then π_i is the event time at node i modulo the period T and p_a translates between periods. For example, if $T = 60$, $a = (i, j)$, $\ell(a) = 20$, $u(a) = 30$, $\pi_i = 45$, $\pi_j = 10$, then $p_a = 1$. One may further add an objective function, which will depend on the application. We will next discuss two applications in somewhat more detail and then make the connection to cycle bases.

Traffic light coordination. The task is to plan the red/green timings of traffic lights. We assume that a desired cycle time has already been determined, e.g., 60 s, and that minimum durations for the green phases of the individual signal groups (left turn lane, straight traffic, etc.) have been derived from the traffic loads of the origin-destination pairs. It is then necessary to schedule the events for each signal group, i.e., when signals turn from green to red and from red to green. Typical objectives are the minimization of the number of red lights for drivers and the total travel time within the network. Wünsch [70] discusses the traffic light coordination problem and related problems in greater detail.

Periodic timetabling in public transport. The German train system runs essentially on either a one- or a two-hour period. Main lines run on a one-hour period, and secondary lines operate on a two-hour period. Of course, the period is not maintained during night hours. The Sunday schedule of the Berlin subway runs on a 10-min period. Shorter periods are used on weekdays, particularly during rush hours. The comprehensive process of timetabling is highly complex, in particular, when different train operation companies intend to use the same track for the same time slot. We concentrate here on purely periodic schedules.

A periodic timetable assigns arrival and departure times to all pairs of lines and stations. For example, Berlin metro line U9 leaves Zoo station southbound at minute 02 and arrives at the next station, Kurfürstendamm, at minute 03. Many constraints have to be respected. These include minimum spacing intervals between two trains using the same track, collision-free service on a single track, and maximum durations for stops in intermediate stations. Among the most important objectives are short transfer times for the passengers as well as short turnaround times for the trains in their terminus stations, where both also have to respect certain minimum durations, too. In Fig. 36, these types of arcs are shown for a small part of New York City.

Cycle bases for PESP. The practical performance of mixed integer programming solvers on PESP instances as formulated above is rather poor. We now describe a more efficient problem formulation that makes use of integral cycle bases.

Given an (in-) feasible solution (π, p) for the PESP, consider the function $x(a)$ on the arcs,

$$x_a := \pi_j - \pi_i + Tp_a, \quad \text{where } a = (i, j).$$

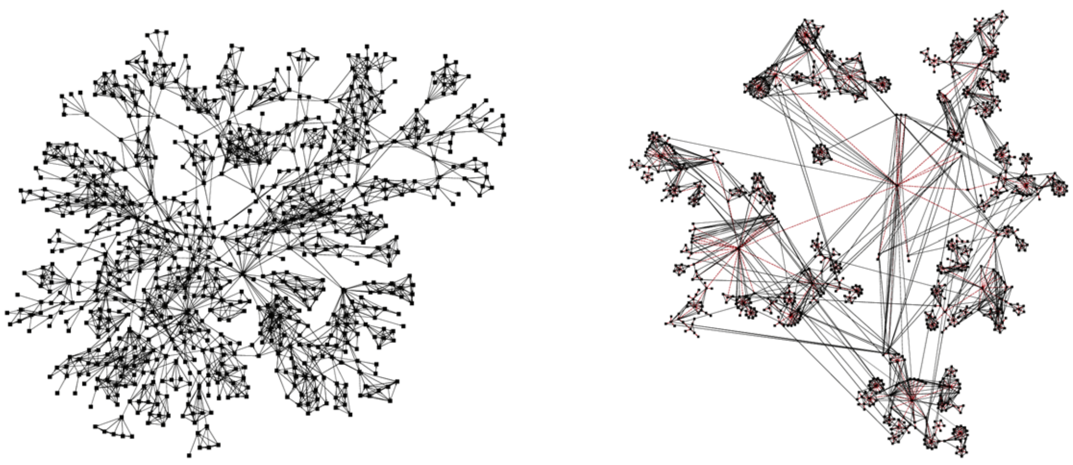


Fig. 37 – Two drawings of the same graph: The drawing on the left is generated by a refined version of a classic force-directed layout approach and the drawing on the right is generated by first computing an F-basis that is then used to draw the whole graph as described in [75]. The edges of the spanning tree are marked in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The vector x is sometimes referred to as a *periodic tension* for (π, p), and models the duration between its two events i and j. Summation of this equality for the arcs of any circuit C yields the *cycle periodicity property*

$$\frac{1}{T}\left(\sum_{a\in C^+} x_a - \sum_{a\in C^-} x_a\right) \in \mathbb{Z};$$

the important observation is that the sum on the left must have an integral value for all circuits C. Nachtigall [71] observed that if the cycle periodicity property holds for the circuits in some strictly fundamental basis it holds for all circuits. Liebchen and Peeters [72] generalized this result to integral cycle bases.

Theorem 7.1 ([72,71]). *Let x be some vector on the arcs of a directed graph. There exists a pair (π, p) such that x is a periodic tension of (π, p), if and only if x satisfies the cycle periodicity property for all circuits of some integral cycle basis.*

Indeed, assume that the cycle periodicity holds for all the circuits in an integral basis and let C be an arbitrary circuit. Then $C = \sum_i \lambda_i C_i$, where the C_i's are the basic circuits and the λ_i are integral. Then

$$\begin{aligned}\sum_a C(a)x_a &= \sum_a \left(\sum_i \lambda_i C_i(a)\right) x_a \\ &= \sum_i \lambda_i \left(\sum_a C_i(a)x_a\right) = \sum_i \lambda_i q_{C_i} T\end{aligned}$$

and hence the net travel time along C is an integral multiple of the period. For this argument to hold, it is essential that C be an *integral* linear combination of the basic circuits.

Practical use. For both applications, the mathematical model sketched above has made its way into practice — including the computation of short integral cycle bases as a preprocessing subroutine. For the traffic light scheduling problem, Wünsch [70] reports, that since 2008, the method has been commercially available as a module in one of the major software suites for traffic planning. In periodic timetabling, Liebchen [73] reports that the first mathematically optimized railway timetable went into service in 2005, for the Berlin subway network. About two years later, even a national railway company reported that their new timetable was designed with the help of combinatorial algorithms [74].

7.3. Graph drawing

Graph drawing is concerned with embedding graphs into the plane in an aesthetically pleasing way. A position is assigned to each vertex and each edge is drawn as a (poly-)line. The goal is to obtain a clear, easily interpretable drawing of the graph. [75] have shown that minimum or near minimum strictly fundamental cycle bases are very useful in this context.

They start with the observation that many real-world graphs, such as social networks, are *sparse* and simultaneously *clustered* in the sense that the neighbors of a vertex are frequently also connected directly to each other. These edges will then form triangles. More generally, most edges of real-world graphs belong to triangles or at least short cycles. This is in contrast to sparse random graphs. However, there are usually also some edges that connect seemingly random vertices with each other [76]. Edges of the first category are often called *local* edges and edges of the second category are called *global edges*. Although there is no clear definition of either of these categories, it is frequently desirable to show either the local structure or the global structure of the graph. The spanning tree underlying a (near) minimal cycle basis will provide the right scaffold. Moreover, it can easily be drawn in linear time with a tree drawing method [77].

With this spanning tree as a scaffold, global edges can now be defined as those edges that connect vertices with at least a given threshold distance in the tree. By adding them to the spanning tree, the global structure of the graph can be emphasized. Analogously, by adding the other, non-global edges to the spanning tree, the local, clustered structure is prominently displayed. Thus, this method provides a neat way to show both the local and global structure of a given graph next to each other. Fig. 37 shows an example.

8. Summary

Cycle bases of graphs are a rich subject with many applications. We surveyed structural, algorithmic, and complexity-theoretic results and compiled a list of open problems. We also proved several new results. In particular, we gave additional structural and characterization results, obtained tight length bounds for weakly fundamental cycle bases for the full spectrum of graph densities, simplified the algorithmic treatment of directed cycle bases, and presented the first algorithms for minimum cycle bases in the presence of negative edges.

Acknowledgements

The third author, KM, based part of a recent course on "Selected Topics in Algorithms" on this survey. He thanks the students of the course for numerous comments. All authors thank Markus Geyer and Gregor Wünsch for discussions and comments. The authors also would like to thank Marc Begin for improving the linguistic quality of the paper. For the second author, part of this work was carried out at the DFG Research Center MATHEON in Berlin. The last author was supported by a post-doctoral scholarship from the Deutsche Akademie der Naturforscher Leopoldina (BMBF LPD 9901/8-182).

REFERENCES

[1] S. MacLane, A combinatorial condition for planar graphs, Fund. Math. 28 (1937) 22–32.

[2] C. Liebchen, R. Rizzi, Classes of cycle bases, Discrete Appl. Math. 155 (3) (2007) 337–355.

[3] C. Liebchen, Finding short integral cycle bases for cyclic timetabling, in: G.D. Battista, U. Zwick (Eds.), ESA, in: Lecture Notes in Computer Science, vol. 2832, Springer, 2003, pp. 715–726.

[4] T. Ueckerdt, Berechnung kurzer ganzzahliger Kreisbasen in Graphen, Master's Thesis, Technische Universität Berlin, 2008.

[5] A. Schrijver, Theory of Linear and Integer Programming, Wiley, 1986.

[6] M.M. Sysło, On cycle bases of a graph, Networks 9 (1979) 123–132.

[7] D. Hartvigsen, E. Zemel, Is every cycle basis fundamental? J. Graph Theory 13 (1) (1989) 117–137.

[8] E. Hubicka, M.M. Sysło, Minimal bases of cycles of a graph, in: M. Fiedler (Ed.), Recent advances in graph theory, Academia Praha, 1975, pp. 283–293.

[9] C. Champetier, On the null-homotopy of graphs, Discrete Mathematics 64 (1987) 97–98.

[10] P.M. Gleiss, Short cycles — minimum cycle bases of graphs from Chemistry and Biochemistry, Ph.D. Thesis, Fakultät für Naturwissenschaften und Mathematik der Universität Wien, 2001.

[11] T. Apostol, Introduction to Analytic Number Theory, Springer, 1997.

[12] J. Horton, A polynomial-time algorithm to find the shortest cycle basis of a graph, SIAM J. Comput. 16 (1987) 359–366.

[13] T. Kavitha, K.V. Krishna, An improved heuristic for computing short integral cycle bases, ACM J. Exp. Algorithmics 13 (2008).

[14] J.C. de Pina, Applications of Shortest-Path Methods, Ph.D. Thesis, Universiteit van Amsterdam, 1995.

[15] A. Reich, Streng fundamentale Kreisbasen planarer Graphen, Master's Thesis, Mathematisches Institut der Brandenburgischen Technischen Universität Cottbus, 2007.

[16] A. Lubotzky, R. Phillips, P. Sarnak, Ramanujan graphs, Combinatorica 3 (1988) 261–277.

[17] R. Rizzi, Minimum weakly fundamental cycle bases are hard to find, Algorithmica (2007) doi:10.1007/s00453-007-9112-8. online first.

[18] B. Bollobás, Extremal Graph Theory, Academic Press, New York, 1978.

[19] M. Kaufmann, D. Michail, Personal communication, 2008.

[20] N. Alon, S. Hoory, N. Linial, The Moore bound for irregular graphs, Graphs Combin. 18 (1) (2002) 53–57.

[21] G. Miller, Finding small simple cycle separators for 2-connected planar graphs, J. Comp. Syst. Sci. 32 (1986) 265–279.

[22] J.M. Stern, S.A. Vavasis, Nested dissection for sparse nullspace bases, Technical report, Department of Computer Science, Cornell University, Ithaca, 1990.

[23] N. Alon, R.M. Karp, D. Peleg, D.B. West, A graph-theoretic game and its application to the k-server problem, SIAM J. Comput. 24 (1) (1995) 78–100.

[24] E. Köhler, C. Liebchen, R. Rizzi, G. Wünsch, Lower bounds for strictly fundamental cycle bases in grid graphs, Networks 53 (2) (2009).

[25] D. Peleg, Distributed computing — a locality sensitive approach, SIAM Monogr. Discrete Math. Appl. (2000).

[26] M. Elkin, Y. Emek, D.A. Spielman, S.-H. Teng, Lower-stretch spanning trees, SIAM J. Comput. 38 (2) (2008) 608–628.

[27] N. Deo, G.M. Prabhu, M. Krishnamoorthy, Algorithms for generating fundamental cycles in a graph, ACM Trans. Math. Softw. 8 (1) (1982) 26–42.

[28] M. Elkin, C. Liebchen, R. Rizzi, New length bounds for cycle bases, Inf. Process. Lett. 104 (5) (2007) 186–193.

[29] I. Abraham, Y. Bartal, O. Neiman, Embedding metrics into ultrametrics and graphs into spanning trees with constant average distortion, in: SODA, 2007, pp. 502–511.

[30] B. Korte, J. Vygen, Combinatorial Optimization, Springer, 2005.

[31] K. Mehlhorn, D. Michail, Minimum cycle bases: Faster and simpler, ACM Trans. Algorithms (2008) (in press).

[32] C. Liebchen, R. Rizzi, A greedy approach to compute a minimum cycle basis of a directed graph, Inf. Process. Lett. 94 (3) (2005) 107–112.

[33] V. Bafna, P. Berman, T. Fujito, A 2-approximation algorithm for the undirected feedback vertex set problem, SIAM J. Discrete Math. 12 (3) (1999) 289–297.

[34] A. Golynski, J.D. Horton, A polynomial time algorithm to find the minimum cycle basis of a regular matroid, in: M. Penttonen, E.M. Schmidt (Eds.), SWAT, in: Lecture Notes in Computer Science, Vol. 2368, Springer, 2002, pp. 200–209.

[35] F. Berger, P. Gritzmann, S. de Vries, Minimum cycle bases for network graphs, Algorithmica 40 (2004) 51–62.

[36] T. Kavitha, K. Mehlhorn, D. Michail, K.E. Paluch, A faster algorithm for minimum cycle bases of graphs, in: J. Díaz, J. Karhumäki, A. Lepistö, D. Sannella (Eds.), ICALP, in: Lecture Notes in Computer Science, vol. 3142, Springer, 2004, pp. 846–857.

[37] R. Hariharan, T. Kavitha, K. Mehlhorn, A faster deterministic algorithm for minimum cycle bases in directed graphs, in: M. Bugliesi, B. Preneel, V. Sassone, I. Wegener (Eds.), ICALP (1), in: Lecture Notes in Computer Science, vol. 4051, Springer, 2006, pp. 250–261.

[38] T. Kavitha, K. Mehlhorn, D. Michail, New approximation algorithms for minimum cycle bases of graphs, in: W. Thomas, P. Weil (Eds.), STACS, in: Lecture Notes in Computer Science, vol. 4393, Springer, 2007, pp. 512–523.

[39] E. Amaldi, C. Iuliano, T. Jurkiewicz, K. Mehlhorn, R. Rizzi, Breaking through the $O(m^2n)$ barrier for minimum cycle bases, in: LNCS, vol. 5757, 2009, pp. 301–312. http://www.mpi-inf.mpg.de/~mehlhorn/ftp/BarrierCycleBasis.pdf.

[40] F. Berger, C. Flamm, P.M. Gleiss, J. Leydold, P.F. Stadler, Counterexamples in chemical ring perception, J. Chem. Inform. Comput. Sci. 44 (2) (2004) 323–331.
[41] F. Barahona, A.R. Mahjoub, On the cut polytope, Math. Program. 36 (1986) 157–173.
[42] M. Grötschel, L. Lovász, A. Schrijver, Geometric Algorithms and Combinatorial Optimization, Springer, 1988.
[43] T. Kavitha, K. Mehlhorn, A polynomial time algorithm for minimum cycle bases in directed graphs, in: V. Diekert, B. Durand (Eds.), STACS, in: Lecture Notes in Computer Science, vol. 3404, Springer, 2005, pp. 654–665.
[44] H. Gabow, Data structures for weighted matching and nearest common ancestor with linking, in: SODA, 1990, pp. 434–443.
[45] D. Hartvigsen, R. Mardon, The all-pairs min cut problem and the minimum cycle basis problem on planar graphs, SIAM J. Discrete Math. 7 (3) (1994) 403–418.
[46] S. Pettie, A new approach to all-pairs shortest paths on real-weighted graphs, Theoret. Comput. Sci. 312 (1) (2004) 47–74.
[47] G. Frederickson, Fast algorithms for shortest paths in planar graphs, with applications, SIAM J. Comput. 16 (1987) 1004–1022.
[48] M. Thorup, Undirected single-source shortest paths with positive integer weights in linear time, JACM 46 (1999) 362–394.
[49] I. Althöfer, G. Das, D. Dobkin, D. Joseph, J. Soares, On sparse spanners of weighted graphs, Discrete Comput. Geom. 9 (1) (1993) 81–100.
[50] M. Thorup, U. Zwick, Approximate distance oracles, in: ACM Symposium on Theory of Computing, 2001, pp. 183–192.
[51] M. Thorup, U. Zwick, Compact routing schemes, in: Proceedings of 13th ACM Symposium on Parallel Algorithms and Architecture, 2001, pp. 1–10.
[52] P.M. Gleiss, Cycdeco, 2001. http://www.tbi.univie.ac.at/~pmg/cycdeco/cycdeco.html.
[53] M. Huber, Implementation of algorithms for sparse cycle bases of graphs, 2003. http://www-m9.ma.tum.de/dm/cycles/mhuber.
[54] F. Berger, Kreisbasenbibliothek CyBaL, 2004. http://www-m9.ma.tum.de/m9old/dm/projects/cycles/cybal/.
[55] U. Bauer, Minimum cycle basis algorithms for the chemistry development toolkit, 2004. http://geom.mi.fu-berlin.de/ubauer/CDK-Ringsearch.pdf.
[56] K. Mehlhorn, D. Michail, Implementing minimum cycle basis algorithms, ACM J. Experimental Algorithms 11 (2006) 1–14.
[57] P. Vismara, Union of all the minimum cycle bases of a graph, Electron. J. Combin. 4 (1) (1997) 73–87.
[58] C.M. Papadimitriou, Computational Complexity, Addison-Wesley, 1994.
[59] L. Trevisan, Inapproximability of combinatorial optimization problems. Electronic Colloquium on Computational Complexity (ECCC), 2004.
[60] G. Galbiati, R. Rizzi, E. Amaldi, On the approximability of the minimum strictly fundamental cycle bases problem, 2007 (submitted for publication).
[61] P. Alimonti, V. Kann, Some APX-completeness results for cubic graphs, Theor. Comput. Sci. 237 (1–2) (2000) 123–134.
[62] G. Galbiati, On finding cycle bases and fundamental cycle bases with a shortest maximal cycle, Inf. Process. Lett. 88 (4) (2003) 155–159.
[63] S.P. Fekete, J. Kremer, Tree spanners in planar graphs, Discrete Appl. Math. 108 (1–2) (2001) 85–103.
[64] G. Kirchhoff, Über die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Vertheilung galvanischer Ströme geführt wird, Annal. Phys. Chem. 72 (12) (1847) 497–508.
[65] B. Bollobás, Modern Graph Theory, in: Graduate Texts in Mathematics, vol. 184, Springer, 2002.
[66] S. Bächle, Numerical solution of differential-algebraic systems arising in circuit simulation, Ph.D. Thesis, Technische Universität Berlin, 2007.
[67] N.H. Gartner, J.D. Little, H. Gabbay, Optimization of traffic signal settings by mixed-integer linear programming, Part I: The network coordination problem, Transp. Sci. 9 (1975) 321–343.
[68] S. Rüger, Transporttechnologie städtischer öffentlicher Personenverkehr, Transpress, Verlag für Verkehrswesen, 1986.
[69] P. Serafini, W. Ukovich, A mathematical model for periodic scheduling problems, SIAM J. Discrete Math. 2 (4) (1989) 550–581.
[70] G. Wünsch, Coordination of traffic signals in networks and related graph theoretical problems on spanning trees, Ph.D. Thesis, Technische Universität Berlin, Institute of Mathematics, 2008.
[71] K. Nachtigall, Periodic network optimization with different arc frequencies, Discrete Appl. Math. 69 (1–2) (1996) 1–17.
[72] C. Liebchen, L. Peeters, Integral cycle bases for cyclic timetabling, Discrete Optim. 6 (1) (2009) 98–109.
[73] C. Liebchen, The first optimized railway timetable in practice, Transp. Sci. 42 (4) (2008) 420–435.
[74] L. Kroon, D. Huisman, E. Abbink, P.-J. Fioole, M. Fischetti, G. Maróti, L. Schrijver, A. Steenbeek, R. Ybema, The new Dutch timetable: The OR revolution, Technical Report EI2008-19, Erasmus University Rotterdam, Econometric Institute. (Accepted for publication in Interfaces as winning paper of the 2008 INFORMS Franz Edelman Award, 2008).
[75] K.A. Lehmann, S. Kottler, Visualizing large and complex networks, in: Proceedings of the 14th International Symposium on Graph Drawing, GD'06, 2007.
[76] D.J. Watts, S.H. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (1998) 440–442.
[77] M. Kaufmann, D. Wagner (Eds.), Drawing Graphs, in: Lecture Notes in Computer Science, vol. 2025, Springer, 2001.