Breaking the $O(m^2n)$ Barrier for Minimum Cycle Bases

Edoardo Amaldi[1], Claudio Iuliano[1], Tomasz Jurkiewicz[2], Kurt Mehlhorn[2], and Romeo Rizzi[3]

[1] Dipartimento di Elettronica e Informazione, Politecnico di Milano, Milano, Italy
[2] Max-Planck-Institut für Informatik, Saarbrücken, Germany
[3] Dipartimento di Matematica ed Informatica, Universitá degli Studi di Udine, Udine, Italy

Abstract. We give improved algorithms for constructing minimum directed and undirected cycle bases in graphs. For general graphs, the new algorithms are Monte Carlo and have running time $O(m^\omega)$, where ω is the exponent of matrix multiplication. The previous best algorithm had running time $\tilde{O}(m^2n)$. For planar graphs, the new algorithm is deterministic and has running time $O(n^2)$. The previous best algorithm had running time $O(n^2 \log n)$. A key ingredient to our improved running times is the insight that the search for minimum bases can be restricted to a set of candidate cycles of total length $O(nm)$.

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, see [KLM+09, Section 7]. 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 to be preferred. We give improved algorithms for computing minimum weight cycle bases. The algorithms run in time $O(m^\omega)$ for general graphs and $O(n^2)$ for planar graphs; here n and m denote the number of nodes and edges, respectively, and ω is the exponent of matrix multiplication. For planar graphs, this is an improvement by a factor of $O(\log n)$; our result implies a similar improvement for the all-pairs minimum cut problem in planar graphs. For general graphs, our algorithm is the first to run faster than $\tilde{O}(m^2n)$. We mention that the previous best algorithms already used fast matrix multiplication and our improvement is due to new structural and algorithmic insights. A key ingredient to our improved running times is the insight that the search for minimum bases can be restricted to a set of candidate cycles of total length $O(nm)$.

Let $G = (V,E)$ be a connected undirected graph. We orient the edges of G arbitrarily and obtain a directed graph (V,A) which we denote by either D or G. We use the notation $e = uv$ to denote both directed and undirected edges, i.e., the notation stands for the directed edge (u,v) and the undirected edge $\{u,v\}$. We use $\delta(v)$ to denote the set of edges incident to v and $\delta^+(v)$ and $\delta^-(v)$ for the directed edges leaving and entering v, respectively.

Let κ be a field. A *κ-cycle* C in D is a vector in κ^E such that for any vertex v we have $\sum_{e\in\delta^+(v)} C_e = \sum_{e\in\delta^-(v)} C_e$. In other contexts, cycles are sometimes referred to as

A. Fiat and P. Sanders (Eds.): ESA 2009, LNCS 5757, pp. 301–312, 2009.

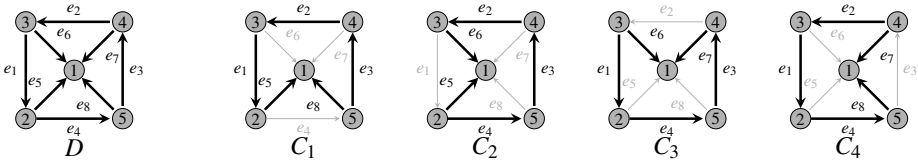

Fig. 1. The figure shows a directed graph D and four circuits C_1 to C_4 in D. The edges of D are e_1 to e_8. The circuit C_1 uses the edges e_1, e_2, e_3, and e_5 in forward direction and the edge e_8 in backward direction. Thus $C_1 = (1,1,1,0,1,0,0,-1)$. The circuits C_1 to C_4 form a directed cycle basis of G. The circuit 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 circuit corresponding to C_i, and let $\pi(C)$ be the undirected circuit corresponding to C. Then $\pi(C_1) = (1,1,1,0,1,0,0,1)$ and $\pi(C) = \pi(C_1) \oplus \pi(C_2) \oplus \pi(C_3) \oplus \pi(C_4)$, where $\oplus$ is addition modulo 2. 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, $\pi(2C_4) = \mathbf{0}$ and hence $\{\pi(C_1), \pi(C_2), \pi(C_3), \pi(2C_4)\}$ is *not* an undirected cycle basis of G.

circulations and the constraint $\sum_{e\in\delta^+(v)} C_e = \sum_{e\in\delta^-(v)} C_e$ is called flow conservation. Observe that if C is a cycle, then $-C$ is also a cycle, though a different one. The set

$$\{C; C \text{ is a } \kappa\text{-cycle of } G\}$$

forms a vector space over κ, the *κ-cycle space* of G. 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 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.

Particularly interesting are the cases $\kappa = GF(2)$, 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. Let G be an undirected graph and let D be an orientation of it. For any directed circuit $C \in \{-1,0,+1\}^E$ of D, let $\pi(C) := (C_e \bmod 2)_{e\in E}$. Then $\pi(C)$ is an undirected circuit in G, the *projection* of C. Figure 1 illustrates these definitions. In addition, it provides an example showing that directed cycle bases do not necessarily project onto undirected cycle bases. However, if a set of ν directed circuits projects onto an undirected basis, it forms a directed basis.

A *weighted graph* is a graph together with a non-negative weight function $w : E \to \mathbb{R}_{\geq 0}$. The weight of a set of edges is the sum of the weights of its members. The *weight $w(C)$ and length $|C|$ of a simple cycle C* are

$$w(C) := \sum_e |C_e| w(e) \qquad |C| := \sum_e |C_e| \ ,$$

and the *weight of a cycle basis B* is the sum of the weights of its circuits, i.e.,

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

A *minimal κ-cycle basis* of G is a κ-cycle basis with minimum weight.

Horton [Hor87] gave the first polynomial time algorithm for minimum undirected cycle bases. It had running time $O(m^3n)$. In a sequence of papers [DP95, GH02, BGdV04, KMMP08, MM07], the running time was improved to $\tilde{O}(m^2n)$. Kavitha and Mehlhorn [KM07] gave the first polynomial time algorithm for minimum directed cycle bases. It had running time $O(m^4n)$. In a sequence of papers [LR05, Kav05, HKM08, MM07] the running time was improved to $O(m^3n)$ deterministic time and $\tilde{O}(m^2n)$ Monte Carlo time. We improve the running time to $O(m^\omega)$ Monte Carlo time for undirected and directed bases. For planar graphs, we improve the running time from $O(n^2 \log n)$ [HM94] to $O(n^2)$; the algorithm is deterministic.

This paper is structured as follows. In Section 2 we improve upon a result of Horton [Hor87] and show that the search for cycle bases can be restricted to a set of candidate circuits of total length $O(nm)$; Horton had shown that the search can be restricted to a set of $O(nm)$ circuits. In Section 3, we exploit this structural insight to derive the $O(m^\omega)$ Monte Carlo algorithm for minimum undirected and directed bases. In Section 4, we exploit it to derive the $O(n^2)$ algorithm for minimum bases in planar graphs.

2 Structural Results

For any two nodes u and v, let p_{uv} be a minimum weight path from u to v in G with respect to weight function w. We assume that the collection of minimum weight paths is consistent, i.e., if x and z lie on p_{uv} then p_{xz} is a subpath of p_{uv}. This can be guaranteed for instance by lexicographic ordering. Given an arbitrary numbering of the nodes from 1 to n, a path p between two nodes is considered shorter than a path q of the same total weight if the length of p is strictly smaller than the length of q. In case of ties, the shortest path between p and q will be the one that contains the node with minimum index in the non-common part. For a modified minimum weight path algorithm that ensures lex-shortest paths in time $O(mn + n^2 \log n)$, see [HM94].

For any node x, let T_x be the minimum weight path tree rooted at x, i.e., T_x is the union of the paths p_{xv} for all v. In [Hor87] Horton shows that a polynomial subset of all cycles is guaranteed to contain a minimum cycle basis. The set of Horton candidate cycles, denoted by $\mathscr{H}$, contains all cycles of the form $C_{x,e} := p_{xu} e p_{vx}$, for any possible choice of a node x and an edge $e = uv$ not in T_x, i.e., a co-tree edge. Among these $n\nu$ cycles, we have to consider only the circuits, discarding $C_{x,e}$ if p_{xu} and p_{xv} have more than node x in common (see Figure 2(a)). $\mathscr{H}$ is a multi-set because each circuit C can have different representations $C_{x,e}$ for some of its nodes x. Note that there is no representation for a given node x if C contains more than one co-tree edge with respect to T_x. This is equivalent to the existence of a shortcut between x and another node in C, i.e., the shortest path joining them does not belong to the circuit itself.

A circuit C is called *isometric* if for any two nodes u and v on C, p_{uv} is contained in C. See Figure 2 (b) and (c) for examples of non-isometric circuits. The set of isometric circuits will be denoted by $\mathscr{I}$. Clearly each isometric circuit is a Horton candidate cycle, that is $\mathscr{I} \subseteq \mathscr{H}$.

In fact, we just need to consider isometric circuits.

Proposition 2.1 ([Hor87]). *$\mathscr{I}$ contains a minimum undirected (directed) basis.*

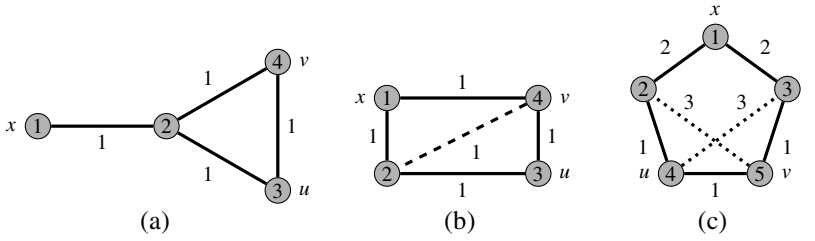

Fig. 2. Examples of non-isometric cycles. (a) $C_{1,\{3,4\}}$ is not a circuit, because p_{13} and p_{14} have also node 2 in common. The contained circuit is obtained as $C_{2,\{3,4\}}$. (b) The minimum weight path connecting 1 and 3 consists of edges $\{1,2\}$ and $\{2,3\}$. $C_{1,\{3,4\}}$ is a non-isometric circuit because of the shortcut, depicted in dashed. The only other representation is $C_{3,\{1,4\}}$. (c) $C_{1,\{4,5\}}$ is a non-isometric circuit with two shortcuts and no representations for any other node.

Moreover, isometric circuits can be characterized in terms of number of representations, namely every isometric circuit C has exactly $|C|$ representations in $\mathscr{H}$.

Property 2.2 (Isometric circuits [Hor87]). *Let C be any isometric circuit and let x be an arbitrary node of C. Then there is an edge $e = uv$ 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 $p_{xv_{i+1}} \neq (v_0, v_1, \ldots, v_i, v_{i+1})$. 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 hence the minimum weight path from x to z is contained in C. ∎

By considering the set of isometric circuits $\mathscr{I}$ instead of $\mathscr{H}$ we have the following simple but fundamental property.

Property 2.3. *The total length of the isometric circuits is at most $n\nu$.*

Proof: An isometric circuit C occurs $|C|$ times in the Horton multi-set and hence $\sum_{C \in \mathscr{I}} |C|$ can be no larger than the size of the Horton multi-set. ∎

Note that we sum only over the isometric circuits, as we have no control over the number of appearances of non-isometric cycles.

The upper bound of Property 2.3 is tight for instance for the complete graph K_n with n vertices and equal weight on the edges. For any node x, the cycle obtained by adding to T_x any co-tree edge is a triangle. $\mathscr{H}$ consists of $n\nu$ triangles that are clearly isometric. Since there are three representations of each possible triangle, obtained by taking as x each one of its 3 nodes, $\mathscr{I}$ consists of $n\nu/3$ triangles. Therefore, the total length of the isometric circuits is exactly $n\nu$.

The total length of the isometric circuits may be much smaller than $n\nu$. Consider an $s \times s$ grid with equal weights on the edges. Since $n = s^2$ and $m = 2s(s-1)$, we

have $\nu = (s-1)^2$, and m and ν are $O(n)$. The isometric circuits are exactly the grid squares and hence their total length is $4(s-1)^2$, that is $O(n)$, whereas the upper bound of Property 2.3 is $n\nu = s^2(s-1)^2$, that is $O(n^2)$.

We will now show that we can extract $\mathscr{I}$ from the Horton multi-set in time $O(nm)$.

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 minimum weight path from x to v. The vectors s_x for all $x \in V$ can be the computed in time $O(n^2)$. Note that a candidate cycle $C = C_{x,e}$, for $e = uv$, is a circuit only when p_{xu} and p_{xv} have only node x in common, i.e., $s_x(u) \neq s_x(v)$ (see Figure 2(a)). The next Lemma shows how to identify different representations of the same isometric circuit and how to discover non-isometric circuits. Given a circuit $C_{x,uv}$, the idea is to check for two specific nodes x' and x'' of C whether the minimum weight path $p_{x'x''}$ between them belongs to C. The nodes x' and x'' are chosen so that a negative answer obviously implies that the circuit is non-isometric whereas a positive answer gives a different representation of C for one of x' and x''. This is achieved by taking $x' = s_x(u)$ and $x'' = v$. In fact, if $p_{x'v}$ belongs to C there are only two possibilities: $p_{x'v} = x'xp_{xv}$ and the other representation for C is for the node x' and is given by $C_{x',uv}$; $p_{x'v} = vup_{ux'}$ and the other representation for C is for the node v and is given by $C_{v,xx'}$. When node x' does not exist because node x coincides with node u, the other representation is for node v and is given by $C_{v,uv}$. Lemma 2.4 explains how to check (in constant time) the conditions that allow to identify the different cases, which are illustrated in Figure 3.

Lemma 2.4. *Let $C = C_{x,e}$, let u be an endpoint of e, and let v be the other endpoint.*

1. *If $s_x(u) \neq s_x(v)$ and $x = u$ then $x \neq v$ and $C = ep_{vu} = C_{v,e}$.*
2. *If $s_x(u) \neq s_x(v)$, $x \neq u$, and $x' = s_x(u)$ is the first node on the minimum weight path from x to u then:*
   (a) if $x = s_{x'}(v)$, then $C = C_{x',e}$,
   (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 $x = u$, $C = uvp_{vu} = p_{vu}uv = C_{v,e}$. This proves the first statement.

If $x \neq u$ and x' is the first vertex on the minimum weight path from x to u, we have $p_{xu} = xx'p_{x'u}$.

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

If x is not the first vertex on the minimum weight path from x' to v and u is the first node on the minimum weight path from v to x' then $C = p_{vx}xx'p_{x'v} = C_{v,xx'}$. This establishes 2b.

If x is not the first vertex on the minimum weight path from x' to v and C is isometric, the minimum weight path from x' to v must be $p_{x'u}$ followed by e. Then u is the first vertex on the minimum weight path from v to x'. This establishes 2c. ∎

Lemma 2.4 allows us to identify different representations of the same isometric circuit. It also allows to exclude some circuits as non-isometric.

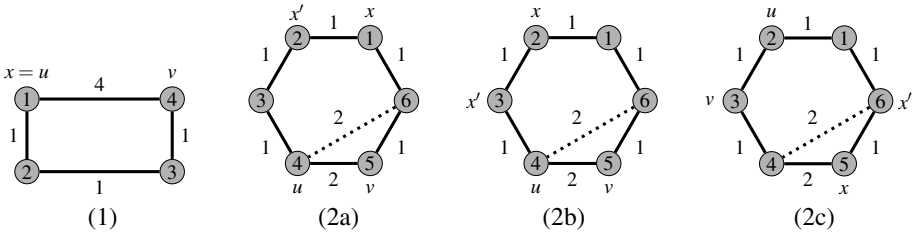

Fig. 3. Examples for the different cases of Lemma 2.4. (1) $C_{1,\{1,4\}}$ where $x = u = 1$. (2a), (2b) and (2c) are three different representations of the same circuit. (2a) $C_{1,\{4,5\}}$ where $s_2(5) = 1$ and we obtain $C_{2,\{4,5\}}$. (2b) $C_{2,\{4,5\}}$ where $s_3(5) \neq 2$ but $s_5(3) = 4$ and we obtain $C_{5,\{2,3\}}$. (2c) $C_{5,\{2,3\}}$ where $s_6(3) \neq 5$ and $s_3(6) \neq 2$, because the minimum weight path connecting 3 and 6 consists of edges $\{3,4\}$ and $\{4,6\}$. This implies that the circuit is not isometric. The shortcut is in dashed line.

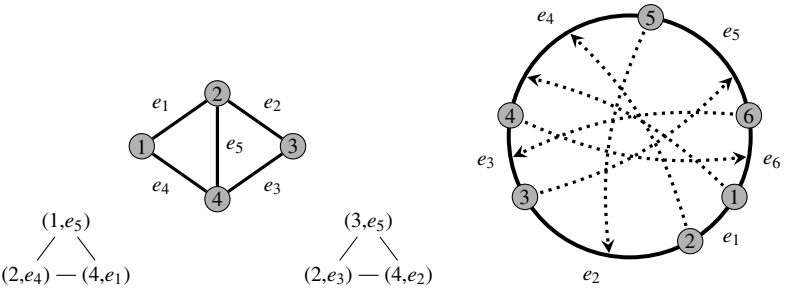

Fig. 4. In the graph on the left all edges have cost one; we select e_1e_2 as the minimum weight path connecting 1 and 3. The circuits C_{1,e_3} and C_{3,e_4} are bad by condition 2c. For the former circuit let $x = 1$, $u = 3$, $v = 4$; then $s_1(3) = 2$ and $s_2(4) \neq 1$ and $s_4(2) \neq 3$. The other circuits are connected as shown below the graph. The figure on the right shows an isometric circuit C embedded on 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 x, we have $C = C_{x,e}$ where e contains the mirror image of x 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.

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), $x \in V$, $e \in E$, if (x,e) is a circuit. We label (x,e) as *bad* if condition 2c holds. We connect two pairs if they satisfy condition 1 or 2a or 2b, see Figure 4.

Lemma 2.5. *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 $e_i = v_iv_{i+1}$, and for any i, $0 \leq i < k$, let $j(i)$ be such that $C = C_{v_i,e_{j(i)}}$. Figure 4 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 edge out of v_i. In this way, the arrow sweeps around the circuit once and links all representations of the same circuit. ∎

Lemma 2.6. *If $C_{x,e}$ is non-isometric then the component of (x,e) contains a bad component.*

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, $0 \le i < k$, 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 minimum weight 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 minimum weight 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 minimum weight path from v_{i+1} to $v_{j(i)+1}$ is not contained in C. For any such i, $C_{v_i, e_{j(i)}}$ will be declared bad. Any non-bad representation of C will be linked to a bad one as described in the preceding Lemma. ∎

Note that checking the conditions of Lemma 2.4 is needed once for each circuit in $\mathscr{H}$.

We summarize the discussion.

Theorem 2.7. *In time $O(nm)$ we can extract for each isometric circuit one pair (x,e) with $C = C_{x,e}$.*

3 Improved Algorithms for General Graphs

We refine de Pina's approach [DP95, KLM+09] for computing minimum cycle bases, see Figure 5. It operates in phases. In each phase, one circuit is added to the basis. The algorithm also maintains a basis of the orthogonal space; more precisely, at the beginning of the i-th iteration is has a set $\{S_i, \ldots, S_\nu\}$ of linearly independent vectors $S_j \in \kappa^E$ with $\langle C_j, S_j \rangle = 0$, where $\langle \_, \_ \rangle$ is the inner product of vectors over κ. Throughout this section, $\kappa = GF(p)$ for a prime p with $p = O(m \log m)$. In particular, arithmetic in $GF(p)$ takes constant time. At the start of the computation S_j is initialized to the j-th unit vector for $1 \le j \le \nu$, where the numbering of the edges is such that edges $e_{\nu+1}$ to e_m form a spanning tree of G.

1: Initialize S_j to the j-th unit vector for $1 \le j \le \nu$.
2: **for** $i \leftarrow 1, \ldots, \nu$ **do**
3: Compute a minimum weight isometric circuit C_i 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**
8: Output $\{C_1, \ldots, C_\nu\}$.

Fig. 5. De Pina's algorithm for computing a minimum cycle basis.

Steps (4) and (5) of the algorithm make the S_j, $j > i$, orthogonal to C_i and maintain orthogonality of C_1 to C_{i-1}. Updating the vectors S_j as shown takes time $O(m^2)$ per phase and hence total time $O(m^3)$. In [KMMP08], this was improved to time $O(m^\omega)$. The best known realization of step (3) takes time $\tilde{O}(mn)$ per phase and hence total time $\tilde{O}(m^2 n)$. We describe a Monte Carlo algorithm that improves the total time for step (3) to $o(m^\omega)$. The improved algorithm exploits the new structural result presented in the preceding section.

We start with a simple technical lemma.

Lemma 3.1. *Let $\mathscr{C}$ be a collection of circuits. For each circuit $C \in \mathscr{C}$, let $\lambda_C \in GF(p)$ be chosen randomly and let $D = \sum_{C \in \mathscr{C}} \lambda_C C$. Let S be a nonzero vector in $GF(p)^E$. If all circuits in $\mathscr{C}$ are orthogonal to S, D is orthogonal to S. If $\mathscr{C}$ contains a circuit that is non-orthogonal to S, D is orthogonal to S with probability at most $1/p$.*

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

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

Consider the $|\mathscr{I}| \leq nm$ isometric circuits. We sort them by nondecreasing weight and put a binary tree (of depth 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 nonzero inner product with S_i as follows. Assume the search has arrived in 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 nonzero, 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 $\log|\mathscr{I}|$ decisions, and we need to find ν circuits. Thus the total number of decisions is $\nu \log|\mathscr{I}|$ and hence the total probability of error is bounded by $\nu \log|\mathscr{I}| p^{-k}$.

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

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 nonzero entries. In order to prepare one linear combination for each node of the search tree, we choose a random multiplier $\lambda_C \in k$ for each isometric circuit C. We then sum the sparse vectors as indicated by the tree. Each nonzero 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 2.3. We want k linear combinations for each node and hence require time $O(knm \log n)$ to prepare all of them.

Theorem 3.2. *There is a Monte Carlo algorithm for finding a minimum $GF(p)$-basis that works in time $O(nm + n^2 \log n + m^\omega + km^2 \log n)$ and errs with probability 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 an observation in [KLM+09, Section 3.5], namely that a minimum $GF(p)$-basis for a random p with $p = \Theta(m \log m)$ is a minimum directed basis with probability at least $1/2$.

Theorem 3.3. *There is a Monte Carlo algorithm for finding a minimum directed cycle basis that works in time $O(m^\omega)$ and errs with probability at most $1/2$.*

4 Planar Graphs

Hartvigsen and Mardon [HM94] have shown that minimum undirected cycle bases in planar graphs can be computed in time $O(n^2 \log n)$. In this section, we summarize their result, improve the running time to $O(n^2)$, and also show that for planar graphs, the notions of minimum directed, undirected, integral, weakly fundamental, and totally unimodular bases coincide, see [KLM+09, Section 3] and the proof of Theorem 4.2 for a definition of the latter terms.

Let G be a plane graph, a planar graph 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 $\mathit{interior}(C)$ to denote the bounded set. If $\mathit{interior}(C)$ agrees with one of the faces of G, we call C a *face circuit*. Note that the number of edges and the number of face circuits are both $O(n)$. A collection of circuits is called *nested* if for any two circuits 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 $\mathit{interior}(C) \supset \mathit{interior}(C')$ and there is no circuit $C'' \in B \cup F_B$ such that $\mathit{interior}(C) \supset \mathit{interior}(C'') \supset \mathit{interior}(C')$. The inclusion graph is acyclic; the nodes of D_B with no outgoing edges are precisely the face circuits of G. The inclusion graph is a forest if and only if B is nested.

In [HM94] Hartvigsen and Mardon show that the number of isometric circuits is at most twice the number of face circuits of any planar graph G and there is at least a minimum cycle basis (directed or undirected) that is nested. Moreover, a nested collection of cycles B is a minimum cycle basis iff B is a minimum weight collection of circuits satisfying three properties: (1) every non-leaf in D_B has exactly one child in F_B, (2) the circuits in F_B have parents in D_B, (3) the inclusion graph D_B is a forest.

Our algorithm for finding a minimum weight basis differs from that of [HM94] in two points. First, we use the all-pairs minimum weight paths method for planar graph in $O(n^2)$ proposed in [Fre87]. Then, the main improvement is to exploit the procedure implied by Theorem 2.7 to obtain the set of isometric circuits in $O(n^2)$. This way, the bottleneck of $O(n^2 \log n)$ decreases to $O(n^2)$. The rest of the algorithm proceeds as in [HM94] and we summarize it below for completeness. Recall that the number of isometric circuits is $O(n)$ and that sorting by nondecreasing weight is $O(n \log n)$.

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

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 and hence D_B does not satisfy properties (1) and (2), we do the following.

If there is a non-leaf node C that has more than one child in F_B (case 1), let R_1 and R_2 be two faces of G limited by 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 the face limited by this face circuit 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. So assume that D is not a face circuit. Starting from the face circuits in $interior(D)$ (we can find them in matrix A), we determine 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 time $O(n)$.

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

Theorem 4.1. *A minimum (directed or undirected) circuit basis of a planar graph can be found in time $O(n^2)$.*

[HM94] 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 time $O(n^2)$.

Theorem 4.2. *Every planar graph has a minimum directed cycle basis that is weakly fundamental, totally unimodular, and integral.*

Proof: 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 can be obtained as the sum of the face circuits that limit faces in $interior(C)$. A face circuit either belongs to B or is equal to the difference of its parent $p(F)$ in D_B and the sum of the other children of $p(F)$ in D_B. Thus

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

If a circuit D occurs twice in the representation of C, it occurs once as a parent and once as a child. As a parent, its coefficient is $+1$, and as a child, its coefficient 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 exhibit an ordering $C_1, \ldots, C_\nu$ of the circuits in B such that $C_i \setminus (C_1 \cup \ldots \cup C_{i-1}) \neq \emptyset$ for all i. Let D_B be the inclusion graph corresponding to B. If F_B is empty, every face circuit belongs to B. We determine a reverse ordering of the circuits $C_\nu, \ldots, C_1$ as described in [LR07]. Starting

from the circuit C that limits the unbounded face, we add the face circuits with an edge in common with C. After removing the edges of C from G, we proceed in the same way. We now extend the previous result to the general case when F_B is not empty. Since every face circuit in F_B has a parent, we have a non-leaf node D in D_B whose children are all face circuits. One of these face circuits, say F, belongs to F_B and all the others belong to B. The same idea for constructing a reverse ordering is then applied to the circuits in B corresponding to the children of D starting from F. The face circuits among these with an edge in common with F are added and the edges of F that are not in D are removed. Then we proceed in the same way considering the circuit that limits the new face. After that all children of D are added, we delete them from D_B. We repeat this until all nodes in D_B are isolated. By applying the procedure in the remaining graph for the case where there are only face circuits, we find a reverse ordering of the circuits. Thus, the same result holds for general cycle bases.

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

5 Conclusion

We have shown that minimum cycle bases can be computed in time $O(m^\omega)$ by a Monte Carlo algorithm. A further improvement would have to do away with the maintenance of a basis of the orthogonal subspace.

References

[BGdV04] Berger, F., Gritzmann, P., de Vries, S.: Minimum cycle bases for network graphs. Algorithmica 40(1), 51–62 (2004)

[DP95] De Pina, J.C.: Applications of shortest path methods. PhD thesis, University of Amsterdam, The Netherlands (1995)

[Fre87] Frederickson, G.N.: Fast algorithms for shortest paths in planar graphs, with applications. SIAM J. Computing 16(6), 1004–1022 (1987)

[GH02] Golynski, A., Horton, J.D.: A polynomial time algorithm to find the minimum cycle basis of a regular matroid. In: Penttonen, M., Schmidt, E.M. (eds.) SWAT 2002. LNCS, vol. 2368, pp. 200–209. Springer, Heidelberg (2002)

[HKM08] Hariharan, R., Kavitha, T., Mehlhorn, K.: Faster deterministic and randomized algorithms for minimum cycle basis in directed graphs. SIAM J. Computing 38(4), 1430–1447 (2008)

[HM94] Hartvigsen, D., Mardon, R.: The all-pairs min cut problem and the minimum cycle basis problem on planar graphs. SIAM J. Discrete Math. 7(3), 403–418 (1994)

[Hor87] Horton, J.D.: A polynomial-time algorithm to find the shortest cycle basis of a graph. SIAM J. Computing 16(2), 358–366 (1987)

[Kav05] Kavitha, T.: An $\tilde{O}(m^2n)$ randomized algorithm to compute a minimum cycle basis of a directed graph. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 273–284. Springer, Heidelberg (2005)

[KLM+09] Kavitha, T., Liebchen, C., Mehlhorn, K., Michail, D., Rizzi, R., Ueckerdt, T., Zweig, K.: Cycle bases in graphs: Characterization, algorithms, complexity, and applications, 78 pages (submitted for publication) (March 2009)

[KM07] Kavitha, T., Mehlhorn, K.: Algorithms to compute minimum cycle basis in directed graphs. Theory of Computing Systems 40(4), 485–505 (2007); A preliminary version of this paper appeared in STACS 2005, vol. 3404, pp. 654–665

[KMMP08] Kavitha, T., Mehlhorn, K., Michail, D., Paluch, K.E.: An $\tilde{O}(m^2n)$ algorithm for minimum cycle basis of graphs. Algorithmica 52(3), 333–349 (2008); A preliminary version of this paper appeared in ICALP 2004, vol. 3142, pp. 846–857

[LR05] Liebchen, C., Rizzi, R.: A greedy approach to compute a minimum cycle basis of a directed graph. Information Processing Letters 94(3), 107–112 (2005)

[LR07] Liebchen, C., Rizzi, R.: Classes of cycle bases. Discrete Applied Mathematics 155(3), 337–355 (2007)

[MM07] Mehlhorn, K., Michail, D.: Minimum cycle bases: Faster and simpler. Accepted for publication in ACM Transactions on Algorithms (2007)